User's Manual

32

# RA4M2 Group

## User's Manual: Hardware

32-Bit MCU
Renesas Advanced (RA) Family
Renesas RA4 Series

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (http://www.renesas.com).

Renesas Electronics
www.renesas.com

Rev.1.20 May 2023

# Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

## Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

## Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

## Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

© 2023 Renesas Electronics Corporation. All rights reserved.

# General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between $V_{IL}$ (Max.) and $V_{IH}$ (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between $V_{IL}$ (Max.) and $V_{IH}$ (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

# Preface

## 1. About this document

This manual is generally organized into an overview of the product, descriptions of the CPU, system control functions, peripheral functions, electrical characteristics, and usage notes. This manual describes the product specification of the microcontroller (MCU) superset. Depending on your product, some pins, registers, or functions might not exist. Address space that store unavailable registers are reserved.

## 2. Audience

This manual is written for system designers who are designing and programming applications using the Renesas Microcontroller. The user is expected to have basic knowledge of electrical circuits, logic circuits, and the MCU.

## 3. Renesas Publications

Renesas provides the following documents. Before using any of these documents, visit www.renesas.com for the most up-to-date version of the document.

| Component | Document Type | Description |
|---|---|---|
| Microcontrollers | Data sheet | Features, overview, and electrical characteristics of the MCU |
| | User's Manual: Hardware | MCU specifications such as pin assignments, memory maps, peripheral functions, electrical characteristics, timing diagrams, and operation descriptions |
| | Application Notes | Technical notes, board design guidelines, and software migration information |
| | Technical Update (TU) | Preliminary reports on product specifications such as restriction and errata |
| Software | User's Manual: Software | API reference and programming information |
| | Application Notes | Project files, guidelines for software programming, and application examples to develop embedded software applications |
| Tools & Kits, Solutions | User's Manual: Development Tools | User's manual and quick start guide for developing embedded software applications with Development Kits (DK), Starter Kits (SK), Promotion Kits (PK), Product Examples (PE), and Application Examples (AE) |
| | User's Manual: Software | |
| | Quick Start Guide | |
| | Application Notes | Project files, guidelines for software programming, and application examples to develop embedded software applications |

# 4. Numbering Notation

The following numbering notation is used throughout this manual:

| Example | Description |
|---|---|
| 011b | Binary number. For example, the binary equivalent of the number 3 is 011b. |
| 0x1F | Hexadecimal number. For example, the hexadecimal equivalent of the number 31 is described 0x1F. In some cases, a hexadecimal number is shown with the suffix "h". |
| 1234 | Decimal number. A decimal number is followed by this symbol only when the possibility of confusion exists. Decimal numbers are generally shown without a suffix. |

# 5. Typographic Notation

The following typographic notation is used throughout this manual:

| Example | Description |
|---|---|
| WDT.WDTRCR.RSTIRQS | Periods separated a function module symbol (WDT), register symbol (WDTRCR), and bit field symbol (RSTIRQS). |
| WDT.WDTRCR | A period separated a function module symbol (WDT) and register symbol (WDTRCR). |
| WDTRCR.RSTIRQS | A period separated a register symbol (WDTRCR) and bit field symbol (RSTIRQS). |
| CKS[3:0] | Numbers in brackets expresses a bit number. For example, CKS[3:0] occupies bits 3 to 0 of the WDT Control Register (WDTCR) register. |

# 6. Unit and Unit Prefix

The following units and unit prefixes are sometimes misleading. Those unit prefixes are described throughout this manual with the following meaning:

| Symbol | Name | Description |
|---|---|---|
| b | Binary Digit | Single 0 or 1 |
| B | Byte | This unit is generally used for memory specification of the MCU and address space. |
| k | kilo- | $1000 = 10^3$. k is also used to denote 1024 ($2^{10}$) but this unit prefix is used to denote 1000 ($10^3$) throughout this manual. |
| K | Kilo- | $1024 = 2^{10}$. This unit prefix is used to denote 1024 ($2^{10}$) not 1000 ($10^3$) throughout this manual. |

# 7. Special Terms

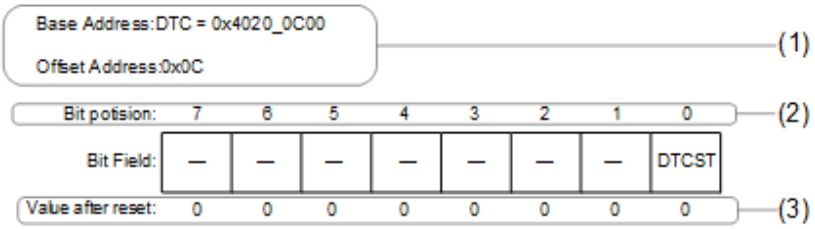
The following terms have special meanings.

| Term | Description |
|---|---|
| NC | Not connected pin. NC means that pin is not connected to the MCU. |
| Hi-Z | High impedance. |

# 8. Register Description

Each register description includes both a register diagram that shows the bit assignments and a register bit table that describes the content of each bit. The example of symbols used in these tables are described in the sections that follow. The following is an example of a register description and associated bit field definition.

XXXX DTCST : DTC Module Start Register




(4) (5) (6)

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DTCST | DTC Module Start<br>0:DTC module stop<br>1:DTC module start | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

### (1) Function module symbol, register symbol, and address assignment

Function module symbol, register symbol, and address assignment of this register are generally expressed. Base Address and Offset Address mean DTC Module Start Register (DTCST) of Data Transfer Controller (DTC) is assigned to address 0x4020_0C00.

### (2) Bit number

This number indicates the bit number. This bits are shown in order from bits 31 to 0 for 32-bit register, from bits 15 to 0 for 16-bit register, and from bits 7 to 0 for 8-bit register.

### (3) Value after reset

This symbol or number indicate the value of each bit after a hard reset. The value is shown in binary unless specified otherwise.

- 0: Indicates that the value is 0 after a reset.
- 1: Indicates that the value is 1 after a reset.
- x: Indicates that the value is undefined after a reset.

### (4) Symbol

Symbol indicates the short name of bit field. Reserved bit is expressed with a —.

### (5) Function

Function indicates the full name of the bit field and enumerated values.

### (6) R/W

The R/W column indicates access type whether the bit field is readable or writable.

- R/W: The bit field is readable and writable.
- R: The bit field is readable only. Writing to this bit field has no effect.
- W: The bit field is writable only. The read value is the same as after a reset unless specified otherwise.

# 9. Abbreviations

Abbreviations used in this document are shown in the following table.

| Abbreviation | Description |
|---|---|
| AES | Advanced Encryption Standard |
| AHB | Advanced High-performance Bus |
| AHB-AP | AHB Access Port |
| APB | Advanced Peripheral Bus |
| ARC | Alleged RC |
| ATB | Advanced Trace Bus |
| BCD | Binary Coded Decimal |
| BSDL | Boundary Scan Description Language |
| DES | Data Encryption Standard |
| DSA | Digital Signature Algorithm |
| ETB | Embedded Trace Buffer |
| ETM | Embedded Trace Macrocell |
| FLL | Frequency Locked Loop |
| FPU | Floating Point Unit |
| HMI | Human Machine Interface |
| IrDA | Infrared Data Association |
| LSB | Least Significant Bit |
| MSB | Most Significant Bit |
| NVIC | Nested Vector Interrupt Controller |
| PC | Program Counter |
| PFS | Port Function Select |
| PLL | Phase Locked Loop |
| POR | Power-on reset |
| PWM | Pulse Width Modulation |
| RSA | Rivest Shamir Adleman |
| SHA | Secure Hash Algorithm |
| S/H | Sample and Hold |
| SP | Stack Pointer |
| SWD | Serial Wire Debug |
| SW-DP | Serial Wire-Debug Port |
| TRNG | True Random Number Generator |
| UART | Universal Asynchronous Receiver/Transmitter |
| VCO | Voltage Controlled Oscillator |

## 10. Proprietary Notice

All text, graphics, photographs, trademarks, logos, artwork and computer code, collectively known as content, contained in this document is owned, controlled or licensed by or to Renesas, and is protected by trade dress, copyright, patent and trademark laws, and other intellectual property rights and unfair competition laws. Except as expressly provided herein, no part of this document or content may be copied, reproduced, republished, posted, publicly displayed, encoded, translated, transmitted or distributed in any other medium for publication or distribution or for any commercial enterprise, without prior written consent from Renesas.

Arm® and Cortex® are registered trademarks of Arm Limited.CoreSight™ is a trademark of Arm Limited.

CoreMark® is a registered trademark of the Embedded Microprocessor Benchmark Consortium.

Other brands and names mentioned in this document may be the trademarks or registered trademarks of their respective holders.

## 11. Feedback on the product

If you have any comments or suggestions about this product, go to Contact Us.

# Contents

**Features** ..... **51**

**1. Overview** ..... **52**

1.1 Function Outline ..... 52
1.2 Block Diagram ..... 57
1.3 Part Numbering ..... 57
1.4 Function Comparison ..... 60
1.5 Pin Functions ..... 62
1.6 Pin Assignments ..... 66
1.7 Pin Lists ..... 68

**2. CPU** ..... **71**

2.1 Overview ..... 71
2.1.1 CPU ..... 71
2.1.2 Debug ..... 71
2.1.3 Operating Frequency ..... 72
2.1.4 Block Diagram ..... 72
2.2 Implementation Options ..... 73
2.3 Trace Interface ..... 74
2.4 JTAG/SWD Interface ..... 74
2.5 Security Attribution for Memory ..... 74
2.6 Debug Function ..... 75
2.6.1 Debugger connectivity ..... 75
2.6.2 Emulator Connection ..... 76
2.6.3 Self-Hosted Debug Function ..... 76
2.6.4 Effect of Debug Function ..... 77
2.7 Programmers Model ..... 77
2.7.1 Address Spaces ..... 77
2.7.2 Peripheral Address Map ..... 78
2.7.3 CoreSight ROM Table ..... 78
2.7.4 DBGREG Module ..... 80
2.7.5 OCDREG Module ..... 82
2.7.6 CPUDSAR : CPU Debug Security Attribution Register ..... 84
2.7.7 Processing on Error response generated by CPU access ..... 84
2.8 CoreSight Cross Trigger Interface (CTI) ..... 86
2.9 CoreSight ATB Funnel ..... 87
2.10 Break Point Unit ..... 88
2.11 CoreSight Time Stamp Generator ..... 88
2.12 SysTick Timer ..... 88

2.13 OCD Emulator Connection ..... 88
2.13.1 DBGEN ..... 89
2.13.2 Restrictions on Connecting an OCD emulator ..... 89
2.14 References ..... 90
**3. Operating Modes ..... 91**
3.1 Overview ..... 91
3.2 Details of Operating Modes ..... 91
3.2.1 Single-Chip Mode ..... 91
3.2.2 SCI Boot Mode ..... 91
3.2.3 USB Boot Mode ..... 91
3.3 Operating Modes Transitions ..... 91
3.3.1 Operating Mode Transitions as Determined by the Mode-Setting Pin ..... 91
**4. Address Space ..... 92**
4.1 Address Space ..... 92
**5. Resets ..... 93**
5.1 Overview ..... 93
5.2 Register Descriptions ..... 98
5.2.1 RSTSAR : Reset Security Attribution Register ..... 98
5.2.2 RSTSR0 : Reset Status Register 0 ..... 99
5.2.3 RSTSR1 : Reset Status Register 1 ..... 100
5.2.4 RSTSR2 : Reset Status Register 2 ..... 103
5.3 Operation ..... 103
5.3.1 RES Pin Reset ..... 103
5.3.2 Power-On Reset ..... 103
5.3.3 Voltage Monitor Reset ..... 104
5.3.4 Deep Software Standby Reset ..... 105
5.3.5 Independent Watchdog Timer Reset ..... 106
5.3.6 Watchdog Timer Reset ..... 106
5.3.7 Software Reset ..... 106
5.3.8 Determination of Cold/Warm Start ..... 106
5.3.9 Determination of Reset Generation Source ..... 107
**6. Option-Setting Memory ..... 108**
6.1 Overview ..... 108
6.2 Register Descriptions ..... 109
6.2.1 OFS0 : Option Function Select Register 0 ..... 109
6.2.2 SAS : Startup Area Setting Register ..... 112
6.2.3 OFS1, OFS1_SEC, OFS1_SEL : Option Function Select Register 1 ..... 113
6.2.4 BPS, BPS_SEC, BPS_SEL : Block Protect Setting Register ..... 114
6.2.5 PBPS, PBPS_SEC : Permanent Block Protect Setting Register ..... 115

6.3 Setting Option-Setting Memory ..... 115
6.3.1 Allocation of Data in Option-Setting Memory ..... 115
6.3.2 Setting Data for Programming Option-Setting Memory ..... 115
6.3.3 Security attribution of option-setting memory ..... 116
6.3.4 Timing of the Setting Value ..... 116
6.4 Usage Notes ..... 116
6.4.1 Data for Programming Reserved Areas and Reserved Bits in the Option-Setting Memory 116
**7. Low Voltage Detection (LVD) ..... 117**
7.1 Overview ..... 117
7.2 Register Descriptions ..... 119
7.2.1 LVDSAR : Low Voltage Detection Security Attribution Register ..... 119
7.2.2 LVD1CMPCR : Voltage Monitoring 1 Comparator Control Register ..... 119
7.2.3 LVD2CMPCR : Voltage Monitoring 2 Comparator Control Register ..... 120
7.2.4 LVD1CR0 : Voltage Monitor 1 Circuit Control Register 0 ..... 121
7.2.5 LVD2CR0 : Voltage Monitor 2 Circuit Control Register 0 ..... 122
7.2.6 LVD1CR1 : Voltage Monitor 1 Circuit Control Register ..... 123
7.2.7 LVD1SR : Voltage Monitor 1 Circuit Status Register ..... 124
7.2.8 LVD2CR1 : Voltage Monitor 2 Circuit Control Register 1 ..... 124
7.2.9 LVD2SR : Voltage Monitor 2 Circuit Status Register ..... 125
7.3 VCC Input Voltage Monitor ..... 125
7.3.1 Monitoring Vdet0 ..... 125
7.3.2 Monitoring Vdet1 ..... 125
7.3.3 Monitoring Vdet2 ..... 126
7.4 Reset from Voltage Monitor 0 ..... 126
7.5 Interrupt and Reset from Voltage Monitor 1 ..... 127
7.6 Interrupt and Reset from Voltage Monitor 2 ..... 129
7.7 Event Link Controller (ELC) Output ..... 132
7.7.1 Interrupt Handling and Event Linking ..... 133
**8. Clock Generation Circuit ..... 134**
8.1 Overview ..... 134
8.2 Register Descriptions ..... 138
8.2.1 CGFSAR : Clock Generation Function Security Attribute Register ..... 138
8.2.2 SCKDIVCR : System Clock Division Control Register ..... 140
8.2.3 SCKSCR : System Clock Source Control Register ..... 142
8.2.4 PLLCCR : PLL Clock Control Register ..... 144
8.2.5 PLLCR : PLL Control Register ..... 145
8.2.6 PLL2CCR : PLL2 Clock Control Register ..... 146
8.2.7 PLL2CR : PLL2 Control Register ..... 147
8.2.8 MOSCCR : Main Clock Oscillator Control Register ..... 148

8.2.9 SOSCCR : Sub-Clock Oscillator Control Register ..... 149
8.2.10 LOCOCR : Low-Speed On-Chip Oscillator Control Register ..... 150
8.2.11 HOCOCR : High-Speed On-Chip Oscillator Control Register ..... 150
8.2.12 MOCOCR : Middle-Speed On-Chip Oscillator Control Register ..... 151
8.2.13 FLLCR1 : FLL Control Register1 ..... 152
8.2.14 FLLCR2 : FLL Control Register2 ..... 153
8.2.15 OSCSF : Oscillation Stabilization Flag Register ..... 154
8.2.16 OSTDCR : Oscillation Stop Detection Control Register ..... 155
8.2.17 OSTDSR : Oscillation Stop Detection Status Register ..... 156
8.2.18 MOSCWTCR : Main Clock Oscillator Wait Control Register ..... 157
8.2.19 MOMCR : Main Clock Oscillator Mode Oscillation Control Register ..... 158
8.2.20 SOMCR : Sub-Clock Oscillator Mode Control Register ..... 158
8.2.21 CKOCR : Clock Out Control Register ..... 159
8.2.22 LOCOUTCR : LOCO User Trimming Control Register ..... 160
8.2.23 MOCOUTCR : MOCO User Trimming Control Register ..... 161
8.2.24 HOCOUTCR : HOCO User Trimming Control Register ..... 161
8.2.25 USBCKDIVCR : USB Clock Division Control Register ..... 162
8.2.26 USBCKCR : USB Clock Control Register ..... 162
8.2.27 TRCKCR : Trace Clock Control Register ..... 163
8.3 Main Clock Oscillator ..... 164
8.3.1 Connecting a Crystal Resonator ..... 164
8.3.2 External Clock Input ..... 164
8.3.3 Notes on External Clock Input ..... 165
8.4 Sub-Clock Oscillator ..... 165
8.4.1 Connecting a 32.768-kHz Crystal Resonator ..... 165
8.4.2 Pin Handling When the Sub-Clock Oscillator Is Not Used ..... 166
8.5 Oscillation Stop Detection Function ..... 166
8.5.1 Oscillation Stop Detection and Operation after Detection ..... 166
8.5.2 Oscillation Stop Detection Interrupts ..... 168
8.6 PLL Circuit ..... 169
8.7 Internal Clock ..... 169
8.7.1 System Clock (ICLK) ..... 169
8.7.2 Peripheral Module Clock (PCLKA, PCLKB, PCLKC, PCLKD) ..... 170
8.7.3 FlashIF Clock (FCLK) ..... 170
8.7.4 USB Clock (USBCLK) ..... 171
8.7.5 CAN Clock (CANMCLK) ..... 171
8.7.6 CAC Clock (CACCLK) ..... 171
8.7.7 RTC-Dedicated Clock (RTCSCLK, RTCLCLK) ..... 171
8.7.8 IWDT-Dedicated Clock (IWDTCLK) ..... 171
8.7.9 AGT-Dedicated Clock (AGTSCLK, AGTLCLK) ..... 171

8.7.10 SysTick Timer-Dedicated Clock (SYSTICCLK) ..... 171
8.7.11 External Pin Output Clock (CLKOUT) ..... 171
8.7.12 JTAG Clock (JTAGTCK) ..... 172
8.8 Usage Notes ..... 172
8.8.1 Notes on Clock Generation Circuit ..... 172
8.8.2 Notes on Resonator ..... 172
8.8.3 Notes on Board Design ..... 172
8.8.4 Notes on Resonator Connect Pin ..... 173
8.8.5 Notes on Using Sub-Clock Oscillator ..... 173
**9. Clock Frequency Accuracy Measurement Circuit (CAC) ..... 174**
9.1 Overview ..... 174
9.2 Register Descriptions ..... 175
9.2.1 CACR0 : CAC Control Register 0 ..... 175
9.2.2 CACR1 : CAC Control Register 1 ..... 176
9.2.3 CACR2 : CAC Control Register 2 ..... 176
9.2.4 CAICR : CAC Interrupt Control Register ..... 177
9.2.5 CASTR : CAC Status Register ..... 178
9.2.6 CAULVR : CAC Upper-Limit Value Setting Register ..... 179
9.2.7 CALLVR : CAC Lower-Limit Value Setting Register ..... 179
9.2.8 CACNTBR : CAC Counter Buffer Register ..... 180
9.3 Operation ..... 180
9.3.1 Measuring Clock Frequency ..... 180
9.3.2 Digital Filtering of Signals on CACREF Pin ..... 181
9.4 Interrupt Requests ..... 181
9.5 Usage Notes ..... 182
9.5.1 Settings for the Module-Stop Function ..... 182
**10. Low Power Modes ..... 183**
10.1 Overview ..... 183
10.2 Register Descriptions ..... 187
10.2.1 LPMSAR : Low Power Mode Security Attribution Register ..... 187
10.2.2 DPFSAR : Deep Software Standby Interrupt Factor Security Attribution Register ..... 188
10.2.3 SBYCR : Standby Control Register ..... 190
10.2.4 MSTPCRA : Module Stop Control Register A ..... 191
10.2.5 MSTPCRB : Module Stop Control Register B ..... 192
10.2.6 MSTPCRC : Module Stop Control Register C ..... 193
10.2.7 MSTPCRD : Module Stop Control Register D ..... 194
10.2.8 MSTPCRE : Module Stop Control Register E ..... 196
10.2.9 OPCCR : Operating Power Control Register ..... 197
10.2.10 SOPCCR : Sub Operating Power Control Register ..... 198

10.2.11 SNZCR : Snooze Control Register ..... 199
10.2.12 SNZEDCR0 : Snooze End Control Register 0 ..... 200
10.2.13 SNZEDCR1 : Snooze End Control Register 1 ..... 201
10.2.14 SNZREQCR0 : Snooze Request Control Register 0 ..... 202
10.2.15 SNZREQCR1 : Snooze Request Control Register 1 ..... 204
10.2.16 DPSBYCR : Deep Software Standby Control Register ..... 204
10.2.17 DPSWCR : Deep Software Standby Wait Control Register ..... 206
10.2.18 DPSIER0 : Deep Software Standby Interrupt Enable Register 0 ..... 206
10.2.19 DPSIER1 : Deep Software Standby Interrupt Enable Register 1 ..... 207
10.2.20 DPSIER2 : Deep Software Standby Interrupt Enable Register 2 ..... 208
10.2.21 DPSIER3 : Deep Software Standby Interrupt Enable Register 3 ..... 209
10.2.22 DPSIFR0 : Deep Software Standby Interrupt Flag Register 0 ..... 209
10.2.23 DPSIFR1 : Deep Software Standby Interrupt Flag Register 1 ..... 210
10.2.24 DPSIFR2 : Deep Software Standby Interrupt Flag Register 2 ..... 211
10.2.25 DPSIFR3 : Deep Software Standby Interrupt Flag Register 3 ..... 213
10.2.26 DPSIEGR0 : Deep Software Standby Interrupt Edge Register 0 ..... 214
10.2.27 DPSIEGR1 : Deep Software Standby Interrupt Edge Register 1 ..... 215
10.2.28 DPSIEGR2 : Deep Software Standby Interrupt Edge Register 2 ..... 215
10.2.29 SYOCDCR : System Control OCD Control Register ..... 216
10.2.30 LDOSCR:LDO Stop Control Register ..... 217
10.2.31 PL2LDOSCR:PLL2-LDO Stop Control Register ..... 219
10.3 Reducing Power Consumption by Switching Clock Signals ..... 220
10.4 Module-Stop Function ..... 220
10.5 Function for Lower Operating Power Consumption ..... 220
10.5.1 Setting Operating Power Control Mode ..... 220
10.6 Sleep Mode ..... 221
10.6.1 Transitioning to Sleep Mode ..... 221
10.6.2 Canceling Sleep Mode ..... 222
10.7 Software Standby Mode ..... 222
10.7.1 Transitioning to Software Standby Mode ..... 222
10.7.2 Canceling Software Standby Mode ..... 224
10.7.3 Example of Software Standby Mode Application ..... 225
10.8 Snooze Mode ..... 226
10.8.1 Transition to Snooze Mode ..... 226
10.8.2 Canceling Snooze Mode ..... 226
10.8.3 Returning from Snooze Mode to Software Standby Mode ..... 227
10.8.4 Snooze Operation Example ..... 228
10.9 Deep Software Standby Mode ..... 232
10.9.1 Transitioning to Deep Software Standby Mode ..... 232
10.9.2 Cancelling Deep Software Standby Mode ..... 232

10.9.3 Pin States when Deep Software Standby mode is Canceled ..... 233
10.9.4 Example of Deep Software Standby Mode Application ..... 233
10.9.5 Usage Flow for Deep Software Standby Mode ..... 234
10.10 Usage Notes ..... 235
10.10.1 Register Access ..... 235
10.10.2 I/O Port pin states ..... 237
10.10.3 Module-Stop State of DTC, DMAC ..... 237
10.10.4 Internal Interrupt Sources ..... 237
10.10.5 Input Buffer Control by DIRQnE Bit ..... 237
10.10.6 Transitioning to Low Power Modes ..... 237
10.10.7 Timing of WFI Instruction ..... 237
10.10.8 Writing to the WDT/IWDT Registers by DTC or DMAC in Sleep Mode or Snooze Mode ..... 238
10.10.9 Oscillators in Snooze Mode ..... 238
10.10.10 Snooze Mode Entry by RXD0 Falling Edge ..... 238
10.10.11 Using UART of SCI0 in Snooze Mode ..... 238
10.10.12 Conditions of A/D Conversion Start in Snooze Mode ..... 238
10.10.13 Conditions of CTSU in Snooze Mode ..... 238
10.10.14 ELC Events in Snooze Mode ..... 238
10.10.15 Module-Stop Bit Write Timing ..... 239

**11. Battery Backup Function ..... 240**
11.1 Overview ..... 240
11.1.1 Features of Battery Backup Function ..... 240
11.1.2 Battery Power Supply Switch ..... 240
11.1.3 Backup Registers ..... 240
11.1.4 Time Capture Pin Detection ..... 240
11.2 Register Descriptions ..... 242
11.2.1 BBFSAR : Battery Backup Function Security Attribute Register ..... 242
11.2.2 VBATTMNSELR : Battery Backup Voltage Monitor Function Select Register ..... 243
11.2.3 VBATTMONR : Battery Backup Voltage Monitor Register ..... 244
11.2.4 VBTBER : VBATT Backup Enable Register ..... 244
11.2.5 VBTBKR[n] : VBATT Backup Register (n = 0 to 127) ..... 245
11.2.6 VBTICTLR : VBATT Input Control Register ..... 246
11.3 Operation ..... 246
11.3.1 Battery Backup Function ..... 246
11.3.2 VBATT Battery Power Supply Switch Usage ..... 248
11.3.3 VBATT Backup Register Usage ..... 248
11.4 Usage Notes ..... 248

**12. Register Write Protection ..... 249**
12.1 Overview ..... 249

12.2 Register Descriptions ..... 249
12.2.1 PRCR : Protect Register ..... 249
**13. Interrupt Controller Unit (ICU) ..... 251**
13.1 Overview ..... 251
13.2 Register Descriptions ..... 252
13.2.1 ICUSARA : Interrupt Controller Unit Security Attribution Register A ..... 253
13.2.2 ICUSARB : Interrupt Controller Unit Security Attribution Register B ..... 253
13.2.3 ICUSARC : Interrupt Controller Unit Security Attribution Register C ..... 254
13.2.4 ICUSARD : Interrupt Controller Unit Security Attribution Register D ..... 255
13.2.5 ICUSARE : Interrupt Controller Unit Security Attribution Register E ..... 255
13.2.6 ICUSARF : Interrupt Controller Unit Security Attribution Register F ..... 256
13.2.7 ICUSARG : Interrupt Controller Unit Security Attribution Register G ..... 257
13.2.8 ICUSARH : Interrupt Controller Unit Security Attribution Register H ..... 257
13.2.9 ICUSARI : Interrupt Controller Unit Security Attribution Register I ..... 258
13.2.10 IRQCRi : IRQ Control Register i (i = 0 to 15) ..... 258
13.2.11 NMISR : Non-Maskable Interrupt Status Register ..... 259
13.2.12 NMIER : Non-Maskable Interrupt Enable Register ..... 262
13.2.13 NMICLR : Non-Maskable Interrupt Status Clear Register ..... 263
13.2.14 NMICR : NMI Pin Interrupt Control Register ..... 265
13.2.15 IELSRn : ICU Event Link Setting Register n (n = 0 to 95) ..... 266
13.2.16 DELSRn : DMAC Event Link Setting Register n (n = 0 to 7) ..... 267
13.2.17 SELSR0 : SYS Event Link Setting Register ..... 268
13.2.18 WUPEN0 : Wake Up Interrupt Enable Register 0 ..... 269
13.2.19 WUPEN1 : Wake Up interrupt enable register 1 ..... 270
13.3 Vector Table ..... 271
13.3.1 Interrupt Vector Table ..... 271
13.3.2 Event Number ..... 274
13.4 Interrupt Operation ..... 280
13.4.1 Detecting Interrupts ..... 280
13.5 Interrupt setting procedure ..... 281
13.5.1 Enabling Interrupt Requests ..... 281
13.5.2 Disabling Interrupt Requests ..... 281
13.5.3 Polling for interrupts ..... 281
13.5.4 Selecting Interrupt Request Destinations ..... 281
13.5.5 Digital Filter ..... 283
13.5.6 External Pin Interrupts ..... 284
13.6 Non-Maskable Interrupt Operation ..... 284
13.6.1 Correspondence to TrustZone-M by NMI ..... 285
13.7 Return from Low Power Modes ..... 286
13.7.1 Return from Sleep Mode ..... 286

13.7.2 Return from Software Standby Mode ..... 287
13.7.3 Return from Snooze Mode ..... 287
13.8 Using the WFI Instruction with Non-Maskable Interrupts ..... 287
13.9 Reference ..... 287

**14. Buses ..... 288**
14.1 Overview ..... 288
14.2 Description of Buses ..... 289
14.2.1 Arbitration ..... 289
14.2.2 Parallel Operation ..... 289
14.2.3 Restrictions ..... 290
14.3 Register Descriptions ..... 290
14.3.1 BUSSARA : BUS Security Attribution Register A ..... 290
14.3.2 BUSSARB : BUS Security Attribution Register B ..... 291
14.3.3 BUSSCNT<slave> : Slave Bus Control Register (<slave> = FHBIU, FLBIU, S0BIU, EQBIU) ..... 292
14.3.4 BUSSCNT<slave> : Slave Bus Control Register (<slave> = PSBIU, PLBIU, PHBIU) ..... 292
14.3.5 BUSnERRADD : BUS Error Address Register (n = 1 to 3) ..... 293
14.3.6 BUSnERRRW : BUS Error Read Write Register (n = 1 to 3) ..... 293
14.3.7 BTZFnERRADD : BUS TZF Error Address Register (n = 1 to 3) ..... 294
14.3.8 BTZFnERRRW : BUS TZF Error Read Write Register (n = 1 to 3) ..... 295
14.3.9 BUSnERRSTAT : BUS Error Status Register n (n = 1 to 3) ..... 295
14.3.10 DMACDTCERRSTAT : DMAC/DTC Error Status Register ..... 297
14.3.11 BUSnERRCLR : BUS Error Clear Register n (n = 1 to 3) ..... 297
14.3.12 DMACDTCERRCLR : DMAC/DTC Error Clear Register ..... 298
14.4 Bus Error Monitoring Section ..... 298
14.4.1 Bus Error Types ..... 298
14.4.2 Operations When a Bus Error Occurs ..... 298
14.4.3 Conditions Leading to Illegal Address Access Errors ..... 300
14.4.4 Time-out ..... 301
14.5 References ..... 301

**15. Memory Protection Unit (MPU) ..... 302**
15.1 Overview ..... 302
15.2 Arm MPU ..... 302
15.3 Bus Master MPU ..... 302
15.3.1 Register Descriptions ..... 303
15.3.2 Operation ..... 311
15.4 References ..... 314

**16. DMA Controller (DMAC) ..... 315**
16.1 Overview ..... 315

16.2 Register Descriptions ..... 317
16.2.1 DMACSAR : DMAC Controller Security Attribution Register ..... 317
16.2.2 DMSAR : DMA Source Address Register ..... 317
16.2.3 DMSRR : DMA Source Reload Address Register ..... 318
16.2.4 DMDAR : DMA Destination Address Register ..... 318
16.2.5 DMDRR : DMA Destination Reload Address Register ..... 318
16.2.6 DMCRA : DMA Transfer Count Register ..... 319
16.2.7 DMCRB : DMA Block Transfer Count Register ..... 320
16.2.8 DMTMD : DMA Transfer Mode Register ..... 321
16.2.9 DMINT : DMA Interrupt Setting Register ..... 322
16.2.10 DMAMD : DMA Address Mode Register ..... 323
16.2.11 DMOFR : DMA Offset Register ..... 326
16.2.12 DMCNT : DMA Transfer Enable Register ..... 326
16.2.13 DMREQ : DMA Software Start Register ..... 327
16.2.14 DMSTS : DMA Status Register ..... 328
16.2.15 DMSBS : DMA Source Buffer Size Register ..... 329
16.2.16 DMDBS : DMA Destination Buffer Size Register ..... 330
16.2.17 DMAST : DMA Module Activation Register ..... 331
16.2.18 DMECHR : DMAC Error Channel Register ..... 332
16.3 Operation ..... 333
16.3.1 Transfer Mode ..... 333
16.3.2 Extended Repeat Area Function ..... 342
16.3.3 Free-running Function ..... 344
16.3.4 Address Update Function using Offset ..... 345
16.3.5 Address Update Function in Repeat-Block Transfer Mode ..... 350
16.3.6 Example of Using Repeat-Block Transfer Mode ..... 352
16.3.7 Activation Sources ..... 355
16.3.8 Operation Timing ..... 355
16.3.9 DMAC Execution Cycles ..... 356
16.3.10 Activating the DMAC ..... 357
16.3.11 Starting DMA Transfer ..... 359
16.3.12 Registers during DMA Transfer ..... 359
16.3.13 Channel Priority ..... 360
16.3.14 Channel Security ..... 360
16.3.15 Master TrustZone Filter in DMAC ..... 361
16.4 Ending DMA Transfer ..... 362
16.4.1 Transfer End by Completion of Specified Total Number of Transfer Operations ..... 362
16.4.2 Transfer End by Repeat Size End Interrupt ..... 362
16.4.3 Transfer End by Interrupt on Extended Repeat Area Overflow ..... 363
16.5 Processing on DMA Transfer Error ..... 363

16.5.1 Processing on NMI handler ..... 363
16.5.2 Processing on Error response detection interrupt request (DMA_TRANSERR) handler ..... 366
16.6 Interrupts ..... 372
16.6.1 Transfer End Interrupt ..... 372
16.6.2 Transfer Error Interrupt ..... 374
16.7 Event Link ..... 375
16.8 Low-Power Consumption Function ..... 375
16.9 Usage Notes ..... 375
16.9.1 Access to the Registers during DMA Transfer ..... 375
16.9.2 DMA Transfer to Reserved Areas ..... 376
16.9.3 Setting of DMAC Event Link Setting Register of the Interrupt Controller Unit (ICU.DELSRn n = 0 to 7) ..... 376
16.9.4 Suspending or Restarting DMAC Activation ..... 376
16.9.5 Precautions for Resuming DMA Transfer ..... 376
**17. Data Transfer Controller (DTC) ..... 378**
17.1 Overview ..... 378
17.2 Register Descriptions ..... 379
17.2.1 DTCSAR : DTC Controller Security Attribution Register ..... 380
17.2.2 MRA : DTC Mode Register A ..... 380
17.2.3 MRB : DTC Mode Register B ..... 381
17.2.4 SAR : DTC Transfer Source Register ..... 382
17.2.5 DAR : DTC Transfer Destination Register ..... 382
17.2.6 CRA : DTC Transfer Count Register A ..... 383
17.2.7 CRB : DTC Transfer Count Register B ..... 383
17.2.8 DTCCR : DTC Control Register ..... 384
17.2.9 DTCCR_SEC : DTC Control Register for secure Region ..... 384
17.2.10 DTCVBR : DTC Vector Base Register ..... 385
17.2.11 DTCVBR_SEC : DTC Vector Base Register for secure Region ..... 385
17.2.12 DTCST : DTC Module Start Register ..... 385
17.2.13 DTCSTS : DTC Status Register ..... 386
17.2.14 DTEVR : DTC Error Vector Register ..... 387
17.3 Activation Sources ..... 388
17.3.1 Allocating Transfer Information and DTC Vector Table ..... 388
17.4 Operation ..... 390
17.4.1 Transfer Information Read Skip Function ..... 392
17.4.2 Transfer Information Write-Back Skip Function ..... 392
17.4.3 Normal Transfer Mode ..... 393
17.4.4 Repeat Transfer Mode ..... 394
17.4.5 Block Transfer Mode ..... 395
17.4.6 Chain Transfer ..... 396

17.4.7 Operation Timing ..... 397
17.4.8 Execution Cycles of DTC ..... 399
17.4.9 DTC Bus Mastership Release Timing ..... 400
17.4.10 Vector Security ..... 400
17.4.11 Master TrustZone Filter in DTC ..... 400
17.5 DTC Setting Procedure ..... 400
17.6 Examples of DTC Usage ..... 401
17.6.1 Normal Transfer ..... 401
17.6.2 Chain transfer ..... 402
17.6.3 Chain Transfer when Counter = 0 ..... 403
17.7 Processing on DTC Transfer Error ..... 405
17.7.1 Processing on NMI handler ..... 406
17.7.2 Processing on Error response detection interrupt request (DMA_TRANSERR) handler ..... 409
17.8 Interrupt ..... 415
17.8.1 Interrupt Request of Transfer End ..... 415
17.8.2 Interrupt Request of Transfer Error ..... 415
17.9 Event Link ..... 416
17.10 Low Power Consumption Function ..... 416
17.11 Usage Notes ..... 417
17.11.1 Transfer Information Start Address ..... 417
**18. Event Link Controller (ELC) ..... 418**
18.1 Overview ..... 418
18.2 Register Descriptions ..... 419
18.2.1 ELCR : Event Link Controller Register ..... 419
18.2.2 ELSEGRn : Event Link Software Event Generation Register n (n = 0, 1) ..... 420
18.2.3 ELSRn : Event Link Setting Register n (n = 0 to 9, 12 to 18) ..... 421
18.2.4 ELCSARA : Event Link Controller Security Attribution Register A ..... 426
18.2.5 ELCSARB : Event Link Controller Security Attribution Register B ..... 427
18.2.6 ELCSARC : Event Link Controller Security Attribution Register C ..... 427
18.3 Operation ..... 428
18.3.1 Relation between Interrupt Handling and Event Linking ..... 428
18.3.2 Linking Events ..... 428
18.3.3 Example of Procedure for Linking Events ..... 428
18.4 Usage Notes ..... 428
18.4.1 Linking DMAC/DTC Transfer End Signals as Events ..... 428
18.4.2 Setting Clocks ..... 429
18.4.3 Module-Stop Function Setting ..... 429
18.4.4 ELC Delay Time ..... 429
**19. I/O Ports ..... 430**

19.1 Overview ..... 430
19.2 Register Descriptions ..... 431
19.2.1 PCNTR1/PODR/PDR : Port Control Register 1 ..... 431
19.2.2 PCNTR2/EIDR/PIDR : Port Control Register 2 ..... 432
19.2.3 PCNTR3/PORR/POSR : Port Control Register 3 ..... 433
19.2.4 PCNTR4/EORR/EOSR : Port Control Register 4 ..... 434
19.2.5 PmnPFS/PmnPFS_HA/PmnPFS_BY : Port mn Pin Function Select Register (m = 0 to 7, n = 00 to 15) ..... 435
19.2.6 PWPR : Write-Protect Register ..... 437
19.2.7 PWPRS : Write-Protect Register for Secure ..... 438
19.2.8 PmSAR : Port Security Attribution register (m = 0 to 7) ..... 438
19.3 Operation ..... 439
19.3.1 General I/O Ports ..... 439
19.3.2 Port Function Select ..... 439
19.3.3 Port Group Function for ELC ..... 439
19.4 Handling of Unused Pins ..... 441
19.5 Usage Notes ..... 442
19.5.1 Procedure for Specifying the Pin Functions ..... 442
19.5.2 Procedure for Using Port Group Input ..... 442
19.5.3 Port Output Data Register (PODR) Summary ..... 442
19.5.4 Notes on Using Analog Functions ..... 443
19.5.5 I/O Buffer Specification ..... 443
19.6 Peripheral Select Settings for Each Product ..... 444

**20. Port Output Enable for GPT (POEG) ..... 450**

20.1 Overview ..... 450
20.2 Register Descriptions ..... 452
20.2.1 POEGGn : POEG Group n Setting Register (n = A to D) ..... 452
20.3 Output-Disable Control Operation ..... 453
20.3.1 Pin Input Level Detection Operation ..... 453
20.3.2 Output-Disable Requests from the GPT ..... 454
20.3.3 Output-Disable Control Using Detection of Stopped Oscillation ..... 454
20.3.4 Output-Disable Control Using Registers ..... 454
20.3.5 Release from Output-Disable ..... 454
20.4 Interrupt Sources ..... 455
20.5 External Trigger Output to the GPT ..... 455
20.6 Usage Notes ..... 456
20.6.1 Transition to Software Standby Mode ..... 456
20.6.2 Specifying Pins Associated with the GPT ..... 456

**21. General PWM Timer (GPT) ..... 457**

21.1 Overview ..... 457

21.2 Register Descriptions ..... 460
21.2.1 GTWP : General PWM Timer Write-Protection Register ..... 460
21.2.2 GTSTR : General PWM Timer Software Start Register ..... 462
21.2.3 GTSTP : General PWM Timer Software Stop Register ..... 463
21.2.4 GTCLR : General PWM Timer Software Clear Register ..... 464
21.2.5 GTSSR : General PWM Timer Start Source Select Register ..... 464
21.2.6 GTPSR : General PWM Timer Stop Source Select Register ..... 467
21.2.7 GTCSR : General PWM Timer Clear Source Select Register ..... 471
21.2.8 GTUPSR : General PWM Timer Up Count Source Select Register ..... 474
21.2.9 GTDNSR : General PWM Timer Down Count Source Select Register ..... 477
21.2.10 GTICASR : General PWM Timer Input Capture Source Select Register A ..... 481
21.2.11 GTICBSR : General PWM Timer Input Capture Source Select Register B ..... 484
21.2.12 GTCR : General PWM Timer Control Register ..... 487
21.2.13 GTUDDTYC : General PWM Timer Count Direction and Duty Setting Register ..... 489
21.2.14 GTIOR : General PWM Timer I/O Control Register ..... 491
21.2.15 GTINTAD : General PWM Timer Interrupt Output Setting Register ..... 495
21.2.16 GTST : General PWM Timer Status Register ..... 496
21.2.17 GTBER : General PWM Timer Buffer Enable Register ..... 500
21.2.18 GTCNT : General PWM Timer Counter ..... 501
21.2.19 GTCCRk : General PWM Timer Compare Capture Register k (k = A to F) ..... 502
21.2.20 GTPR : General PWM Timer Cycle Setting Register ..... 502
21.2.21 GTPBR : General PWM Timer Cycle Setting Buffer Register ..... 502
21.2.22 GTDTCR : General PWM Timer Dead Time Control Register ..... 503
21.2.23 GTDVU : General PWM Timer Dead Time Value Register U ..... 504
21.2.24 GTICLF : General PWM Timer Inter Channel Logical Operation Function Setting Register ..... 504
21.2.25 GTPC : General PWM Timer Period Count Register ..... 506
21.2.26 GTSECSR : General PWM Timer Operation Enable Bit Simultaneous Control Channel Select Register ..... 507
21.2.27 GTSECR : General PWM Timer Operation Enable Bit Simultaneous Control Register ..... 508
21.2.28 OPSCR : Output Phase Switching Control Register ..... 509
21.3 Operation ..... 512
21.3.1 Basic Operation ..... 512
21.3.2 Buffer Operation ..... 520
21.3.3 PWM Output Operating Mode ..... 527
21.3.4 Automatic Dead Time Setting Function ..... 537
21.3.5 Count Direction Changing Function ..... 541
21.3.6 Function of Output Duty 0% and 100% ..... 541
21.3.7 Hardware Count Start/Count Stop and Clear Operation ..... 543
21.3.8 Synchronized Operation ..... 548
21.3.9 PWM Output Operation Examples ..... 552

21.3.10 Period Count Function ..... 558
21.3.11 Phase Counting Function ..... 559
21.3.12 Output Phase Switching (GPT_OPS) ..... 569
21.3.13 Inter Channel Logical Operation Function ..... 576
21.4 Interrupt Sources ..... 578
21.4.1 Interrupt Sources ..... 578
21.4.2 DMAC and DTC Activation ..... 580
21.5 Operations Linked by ELC ..... 580
21.5.1 Event Signal Output to ELC ..... 580
21.5.2 Event Signal Inputs from ELC ..... 580
21.6 Noise Filter Function ..... 580
21.7 Protection Function ..... 581
21.7.1 Write-Protection for Registers ..... 581
21.7.2 Disabling of Buffer Operation ..... 581
21.7.3 GTIOCnm Pin Output Negate Control (n = 0 to 7, m = A, B) ..... 583
21.8 Initialization Method of Output Pins ..... 584
21.8.1 Pin Settings after Reset ..... 584
21.8.2 Pin Initialization Due to Error during Operation ..... 585
21.9 Usage Notes ..... 585
21.9.1 Module-Stop Function Setting ..... 585
21.9.2 GTCCRn Settings during Compare Match Operation (n = A to F) ..... 585
21.9.3 Setting Range for GTCNT Counter ..... 586
21.9.4 Starting and Stopping the GTCNT Counter ..... 586
21.9.5 Priority Order of Each Event ..... 586

**22. Low Power Asynchronous General Purpose Timer (AGT) ..... 588**

22.1 Overview ..... 588
22.2 Register Descriptions ..... 590
22.2.1 AGT : AGT Counter Register ..... 590
22.2.2 AGTCMA : AGT Compare Match A Register ..... 590
22.2.3 AGTCMB : AGT Compare Match B Register ..... 591
22.2.4 AGTCR : AGT Control Register ..... 591
22.2.5 AGTMR1 : AGT Mode Register 1 ..... 593
22.2.6 AGTMR2 : AGT Mode Register 2 ..... 593
22.2.7 AGTIOC : AGT I/O Control Register ..... 595
22.2.8 AGTISR : AGT Event Pin Select Register ..... 596
22.2.9 AGTCMSR : AGT Compare Match Function Select Register ..... 597
22.2.10 AGTIOSEL : AGT Pin Select Register ..... 597
22.3 Operation ..... 598
22.3.1 Reload Register and Counter Rewrite Operation ..... 598
22.3.2 Reload Register and AGT Compare Match A/B Register Rewrite Operation ..... 600

22.3.3 Timer Mode ..... 601
22.3.4 Pulse Output Mode ..... 602
22.3.5 Event Counter Mode ..... 603
22.3.6 Pulse Width Measurement Mode ..... 604
22.3.7 Pulse Period Measurement Mode ..... 605
22.3.8 Compare Match function ..... 606
22.3.9 Output Settings for Each Mode ..... 607
22.3.10 Standby Mode ..... 608
22.3.11 Interrupt Sources ..... 609
22.3.12 Event Signal Output to ELC ..... 609
22.4 Usage Notes ..... 610
22.4.1 Count Operation Start and Stop Control ..... 610
22.4.2 Access to Counter Register ..... 610
22.4.3 When Changing Mode ..... 610
22.4.4 Output pin setting ..... 610
22.4.5 Digital Filter ..... 610
22.4.6 How to Calculate Event Number, Pulse Width, and Pulse Period ..... 610
22.4.7 When Count is Forcibly Stopped by TSTOP Bit ..... 611
22.4.8 When Selecting AGTn (n = 0, 2, 4) Underflow as the Count Source ..... 611
22.4.9 Module-stop function ..... 611
22.4.10 When Switching Source Clock ..... 611
**23. Realtime Clock (RTC) ..... 612**
23.1 Overview ..... 612
23.2 Register Descriptions ..... 613
23.2.1 R64CNT : 64-Hz Counter ..... 614
23.2.2 RSECCNT : Second Counter (in Calendar Count Mode) ..... 614
23.2.3 RMINCNT : Minute Counter (in Calendar Count Mode) ..... 615
23.2.4 RHRCNT : Hour Counter (in Calendar Count Mode) ..... 615
23.2.5 RWKCNT : Day-of-Week Counter (in Calendar Count Mode) ..... 616
23.2.6 BCNTn : Binary Counter n (n = 0 to 3) (in Binary Count Mode) ..... 616
23.2.7 RDAYCNT : Day Counter ..... 616
23.2.8 RMONCNT : Month Counter ..... 617
23.2.9 RYRCNT : Year Counter ..... 617
23.2.10 RSECAR : Second Alarm Register (in Calendar Count Mode) ..... 618
23.2.11 RMINAR : Minute Alarm Register (in Calendar Count Mode) ..... 619
23.2.12 RHRAR : Hour Alarm Register (in Calendar Count Mode) ..... 619
23.2.13 RWKAR : Day-of-Week Alarm Register (in Calendar Count Mode) ..... 620
23.2.14 BCNTnAR : Binary Counter n Alarm Register (n = 0 to 3) (in Binary Count Mode) ..... 621
23.2.15 RDAYAR : Date Alarm Register (in Calendar Count Mode) ..... 621
23.2.16 RMONAR : Month Alarm Register (in Calendar Count Mode) ..... 622

23.2.17 RYRAR : Year Alarm Register (in Calendar Count Mode) ..... 622
23.2.18 RYRAREN : Year Alarm Enable Register (in Calendar Count Mode) ..... 623
23.2.19 BCNTnAER : Binary Counter n Alarm Enable Register (n = 0, 1) (in Binary Count Mode). 623
23.2.20 BCNT2AER : Binary Counter 2 Alarm Enable Register (in Binary Count Mode) ..... 624
23.2.21 BCNT3AER : Binary Counter 3 Alarm Enable Register (in Binary Count Mode) ..... 624
23.2.22 RCR1 : RTC Control Register 1 ..... 625
23.2.23 RCR2 : RTC Control Register 2 (in Calendar Count Mode) ..... 626
23.2.24 RCR2 : RTC Control Register 2 (in Binary Count Mode) ..... 628
23.2.25 RCR4 : RTC Control Register 4 ..... 629
23.2.26 RFRL : Frequency Register L ..... 629
23.2.27 RFRH : Frequency Register H ..... 630
23.2.28 RADJ : Time Error Adjustment Register ..... 630
23.2.29 RTCCRn : Time Capture Control Register n (n = 0 to 2) ..... 631
23.2.30 RSECCPn : Second Capture Register n (n = 0 to 2) (in Calendar Count Mode) ..... 632
23.2.31 RMINCPn : Minute Capture Register n (n = 0 to 2) (in Calendar Count Mode) ..... 633
23.2.32 RHRCPn : Hour Capture Register n (n = 0 to 2) (in Calendar Count Mode) ..... 633
23.2.33 RDAYCPn : Date Capture Register n (n = 0 to 2) (in Calendar Count Mode) ..... 634
23.2.34 RMONCPn : Month Capture Register n (n = 0 to 2) (in Calendar Count Mode) ..... 634
23.2.35 BCNTnCPm : BCNTn Capture Register m (n= 0 to 3, m = 0 to 2) (in Binary Count Mode) 635
23.3 Operation ..... 635
23.3.1 Outline of Initial Settings of Registers after Power On ..... 635
23.3.2 Clock and Count Mode Setting Procedure ..... 635
23.3.3 Setting the Time ..... 636
23.3.4 30-Second Adjustment ..... 637
23.3.5 Reading 64-Hz Counter and Time ..... 638
23.3.6 Alarm Function ..... 640
23.3.7 Procedure for Disabling Alarm Interrupt ..... 641
23.3.8 Time Error Adjustment Function ..... 641
23.3.9 Capturing the time ..... 643
23.4 Interrupt Sources ..... 644
23.5 Event Link Output ..... 645
23.5.1 Interrupt Handling and Event Linking ..... 646
23.6 Usage Notes ..... 646
23.6.1 Register Writing during Counting ..... 646
23.6.2 Use of Periodic Interrupts ..... 646
23.6.3 RTCOUT (1-Hz/64-Hz) Clock Output ..... 647
23.6.4 Transitions to Low Power Modes after Setting Registers ..... 647
23.6.5 Notes on Writing to and Reading from Registers ..... 647
23.6.6 Changing the Count Mode ..... 647
23.6.7 Initialization Procedure When the RTC Is Not to Be Used ..... 647

23.6.8 When Switching Source Clock ..... 648

**24. Watchdog Timer (WDT) ..... 649**

24.1 Overview ..... 649

24.2 Register Descriptions ..... 650

24.2.1 WDTRR : WDT Refresh Register ..... 650

24.2.2 WDTCR : WDT Control Register ..... 651

24.2.3 WDTSR : WDT Status Register ..... 653

24.2.4 WDTRCR : WDT Reset Control Register ..... 654

24.2.5 WDTCSTPR : WDT Count Stop Control Register ..... 655

24.2.6 Option Function Select Register 0 (OFS0) ..... 655

24.3 Operation ..... 655

24.3.1 Count Operation in each Start Mode ..... 655

24.3.2 Controlling Writes to the WDTCR, WDTRCR, and WDTCSTPR Registers ..... 659

24.3.3 Refresh Operation ..... 659

24.3.4 Status Flags ..... 660

24.3.5 Reset Output ..... 660

24.3.6 Interrupt Sources ..... 660

24.3.7 Reading the Down-Counter Value ..... 661

24.3.8 Association between Option Function Select Register 0 (OFS0) and WDT Registers ..... 661

24.4 Output to the Event Link Controller (ELC) ..... 662

24.5 Usage Notes ..... 662

24.5.1 ICU Event Link Setting Register n (IELSRn) Setting ..... 662

**25. Independent Watchdog Timer (IWDT) ..... 663**

25.1 Overview ..... 663

25.2 Register Descriptions ..... 664

25.2.1 IWDTRR : IWDT Refresh Register ..... 664

25.2.2 IWDTSR : IWDT Status Register ..... 665

25.2.3 OFS0 : Option Function Select Register 0 ..... 666

25.3 Operation ..... 668

25.3.1 Auto Start Mode ..... 668

25.3.2 Refresh Operation ..... 669

25.3.3 Status Flags ..... 670

25.3.4 Reset Output ..... 671

25.3.5 Interrupt Sources ..... 671

25.3.6 Reading the Down-Counter Value ..... 671

25.4 Output to the Event Link Controller (ELC) ..... 671

25.5 Usage Notes ..... 672

25.5.1 Refresh Operations ..... 672

25.5.2 Clock Division Ratio Setting ..... 672

25.5.3 Constraints on the ICU Event Link Setting Register n (IELSRn) Setting ..... 672

**26. USB 2.0 Full-Speed Module (USBFS) ..... 673**

26.1 Overview ..... 673

26.2 Register Descriptions ..... 674

26.2.1 SYSCFG : System Configuration Control Register ..... 674

26.2.2 SYSSTS0 : System Configuration Status Register 0 ..... 676

26.2.3 DVSTCTR0 : Device State Control Register 0 ..... 677

26.2.4 CFIFO/CFIFOL : CFIFO Port Register ..... 679

26.2.5 DnFIFO/DnFIFOL : DnFIFO Port Register (n = 0, 1) ..... 680

26.2.6 CFIFOSEL : CFIFO Port Select Register ..... 681

26.2.7 DnFIFOSEL : DnFIFO Port Select Register (n = 0, 1) ..... 683

26.2.8 CFIFOCTR : CFIFO Port Control Register ..... 685

26.2.9 DnFIFOCTR : DnFIFO Port Control Register (n = 0, 1) ..... 686

26.2.10 INTENB0 : Interrupt Enable Register 0 ..... 688

26.2.11 INTENB1 : Interrupt Enable Register 1 ..... 688

26.2.12 BRDYENB : BRDY Interrupt Enable Register ..... 689

26.2.13 NRDYENB : NRDY Interrupt Enable Register ..... 690

26.2.14 BEMPENB : BEMP Interrupt Enable Register ..... 691

26.2.15 SOFCFG : SOF Output Configuration Register ..... 692

26.2.16 INTSTS0 : Interrupt Status Register 0 ..... 693

26.2.17 INTSTS1 : Interrupt Status Register 1 ..... 695

26.2.18 BRDYSTS : BRDY Interrupt Status Register ..... 698

26.2.19 NRDYSTS : NRDY Interrupt Status Register ..... 699

26.2.20 BEMPSTS : BEMP Interrupt Status Register ..... 699

26.2.21 FRMNUM : Frame Number Register ..... 700

26.2.22 DVCHGR : Device State Change Register ..... 701

26.2.23 USBADDR : USB Address Register ..... 701

26.2.24 USBREQ : USB Request Type Register ..... 702

26.2.25 USBVAL : USB Request Value Register ..... 703

26.2.26 USBINDX : USB Request Index Register ..... 704

26.2.27 USBLENG : USB Request Length Register ..... 704

26.2.28 DCPCFG : DCP Configuration Register ..... 705

26.2.29 DCPMAXP : DCP Maximum Packet Size Register ..... 705

26.2.30 DCPCTR : DCP Control Register ..... 706

26.2.31 PIPESEL : Pipe Window Select Register ..... 709

26.2.32 PIPECFG : Pipe Configuration Register ..... 710

26.2.33 PIPEMAXP : Pipe Maximum Packet Size Register ..... 711

26.2.34 PIPEPERI : Pipe Cycle Control Register ..... 712

26.2.35 PIPEnCTR : PIPEn Control Registers (n = 1 to 5) ..... 713

26.2.36 PIPEnCTR : PIPEn Control Registers (n = 6 to 9) ..... 717

26.2.37 PIPEnTRE : PIPEn Transaction Counter Enable Register (n = 1 to 5) ..... 719
26.2.38 PIPEnTRN : PIPEn Transaction Counter Register (n = 1 to 5) ..... 720
26.2.39 BCCTRL1 : Battery Charging Control Register 1 ..... 721
26.2.40 BCCTRL2 : Battery Charging Control Register 2 ..... 723
26.2.41 DEVADDn : Device Address n Configuration Register (n = 0 to 5) ..... 723
26.2.42 PHYSECTRL : PHY Single-ended Receiver Control Register ..... 724
26.2.43 DPUSR0R : Deep Software Standby USB Transceiver Control/Pin Monitor Register ..... 724
26.2.44 DPUSR1R : Deep Software Standby USB Suspend/Resume Interrupt Register ..... 725
26.3 Operation ..... 727
26.3.1 System Control ..... 727
26.3.2 Interrupts ..... 737
26.3.3 Interrupt Descriptions ..... 741
26.3.4 Pipe Control ..... 750
26.3.5 FIFO Buffer ..... 754
26.3.6 FIFO Buffer Clearing ..... 755
26.3.7 FIFO Port Functions ..... 755
26.3.8 DMA Transfers (D0FIFO and D1FIFO Ports) ..... 756
26.3.9 Control Transfers Using the DCP ..... 757
26.3.10 Bulk Transfers (Pipes 1 to 5) ..... 759
26.3.11 Interrupt Transfers (Pipes 6 to 9) ..... 759
26.3.12 Isochronous Transfers (Pipes 1 and 2) ..... 759
26.3.13 SOF Interpolation Function ..... 766
26.3.14 Pipe Schedule ..... 767
26.3.15 Battery Charging Detection Processing ..... 768
26.4 Usage Notes ..... 772
26.4.1 Settings for the Module-Stop State ..... 772
26.4.2 Clearing the Interrupt Status Register on Canceling Software Standby Mode ..... 772
26.4.3 Clearing the Interrupt Status Register after Setting Up the Port Function ..... 772

**27. Serial Communications Interface (SCI) ..... 774**
27.1 Overview ..... 774
27.2 Register Descriptions ..... 779
27.2.1 RSR : Receive Shift Register ..... 779
27.2.2 RDR : Receive Data Register ..... 779
27.2.3 RDRHL : Receive Data Register for Non-Manchester mode (MMR.MANEN = 0) ..... 779
27.2.4 RDRHL_MAN : Receive Data Register for Manchester mode (MMR.MANEN = 1) ..... 780
27.2.5 FRDRHL/FRDRH/FRDRL : Receive FIFO Data Register ..... 780
27.2.6 TDR : Transmit Data Register ..... 782
27.2.7 TDRHL : Transmit Data Register for Non-Manchester mode (MMR.MANEN = 0) ..... 782
27.2.8 TDRHL_MAN : Transmit Data Register for Manchester mode (MMR.MANEN = 1) ..... 783
27.2.9 FTDRHL/FTDRH/FTDRL : Transmit FIFO Data Register ..... 784

27.2.10 TSR : Transmit Shift Register ..... 784
27.2.11 SMR : Serial Mode Register for Non-Smart Card Interface Mode (SCMR.SMIF = 0) ..... 785
27.2.12 SMR_SMCI : Serial Mode Register for Smart Card Interface Mode (SCMR.SMIF = 1) ..... 786
27.2.13 SCR : Serial Control Register for Non-Smart Card Interface Mode (SCMR.SMIF = 0) ..... 788
27.2.14 SCR_SMCI : Serial Control Register for Smart Card Interface Mode (SCMR.SMIF = 1) ..... 790
27.2.15 SSR : Serial Status Register for Non-Smart Card Interface and Non-FIFO Mode (SCMR.SMIF = 0, FCR.FM = 0, and MMR.MANEN = 0) ..... 791
27.2.16 SSR_FIFO : Serial Status Register for Non-Smart Card Interface and FIFO Mode (SCMR.SMIF = 0, FCR.FM = 1, and MMR.MANEN = 0) ..... 793
27.2.17 SSR_SMCI : Serial Status Register for Smart Card Interface Mode (SCMR.SMIF = 1, and MMR.MANEN = 0) ..... 796
27.2.18 SSR_MANC : Serial Status Register for Manchester Mode (SCMR.SMIF = 0, and MMR.MANEN = 1) ..... 798
27.2.19 SCMR : Smart Card Mode Register ..... 801
27.2.20 BRR : Bit Rate Register ..... 803
27.2.21 MDDR : Modulation Duty Register ..... 812
27.2.22 SEMR : Serial Extended Mode Register ..... 814
27.2.23 SNFR : Noise Filter Setting Register ..... 817
27.2.24 SIMR1 : IIC Mode Register 1 ..... 818
27.2.25 SIMR2 : IIC Mode Register 2 ..... 819
27.2.26 SIMR3 : IIC Mode Register 3 ..... 820
27.2.27 SISR : IIC Status Register ..... 821
27.2.28 SPMR : SPI Mode Register ..... 822
27.2.29 FCR : FIFO Control Register ..... 823
27.2.30 FDR : FIFO Data Count Register ..... 825
27.2.31 LSR : Line Status Register ..... 825
27.2.32 CDR : Compare Match Data Register ..... 826
27.2.33 DCCR : Data Compare Match Control Register ..... 827
27.2.34 SPTR : Serial Port Register ..... 828
27.2.35 ACTR : Adjustment Communication Timing Register ..... 830
27.2.36 MMR : Manchester Mode Register ..... 831
27.2.37 TMPR : Transmit Manchester Preface Setting Register ..... 833
27.2.38 RMPR : Receive Manchester Preface Setting Register ..... 834
27.2.39 MESR : Manchester Extended Error Status Register ..... 835
27.2.40 MECR : Manchester Extended Error Control Register ..... 836
27.2.41 ESMER : Extended Serial Module Enable Register ..... 837
27.2.42 CR0 : Control Register 0 ..... 838
27.2.43 CR1 : Control Register 1 ..... 838
27.2.44 CR2 : Control Register 2 ..... 839
27.2.45 CR3 : Control Register 3 ..... 840
27.2.46 PCR : Port Control Register ..... 840

27.2.47 ICR : Interrupt Control Register ..... 840
27.2.48 STR : Status Register ..... 841
27.2.49 STCR : Status Clear Register ..... 842
27.2.50 CF0DR : Control Field 0 Data Register ..... 842
27.2.51 CF0CR : Control Field 0 Compare Enable Register ..... 843
27.2.52 CF0RR : Control Field 0 Receive Data Register ..... 843
27.2.53 PCF1DR : Primary Control Field 1 Data Register ..... 844
27.2.54 SCF1DR : Secondary Control Field 1 Data Register ..... 844
27.2.55 CF1CR : Control Field 1 Compare Enable Register ..... 844
27.2.56 CF1RR : Control Field 1 Receive Data Register ..... 845
27.2.57 TCR : Timer Control Register ..... 845
27.2.58 TMR : Timer Mode Register ..... 845
27.2.59 TPRE : Timer Prescaler Register ..... 846
27.2.60 TCNT : Timer Count Register ..... 846
27.3 Operation in Asynchronous Mode ..... 846
27.3.1 Serial Data Transfer Format ..... 847
27.3.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode ..... 848
27.3.3 Clock ..... 849
27.3.4 Double-Speed Operation and Frequency of 6 Times the Bit Rate ..... 850
27.3.5 CTS and RTS Functions ..... 850
27.3.6 Address Match (Receive Data Match Detection) Function ..... 850
27.3.7 SCI Initialization in Asynchronous Mode ..... 853
27.3.8 Serial Data Transmission in Asynchronous Mode ..... 855
27.3.9 Serial Data Reception in Asynchronous Mode ..... 860
27.3.10 The function of adjust receive sampling timing (Asynchronous Mode) ..... 873
27.3.11 The function of adjust transmit timing (Asynchronous Mode) ..... 877
27.4 Multi-Processor Communication Function ..... 882
27.4.1 Multi-Processor Serial Data Transmission ..... 883
27.4.2 Multi-Processor Serial Data Reception ..... 886
27.5 Operation in Manchester mode ..... 891
27.5.1 Frame Format ..... 892
27.5.2 Clock ..... 896
27.5.3 Initialization of the SCI in Manchester Mode ..... 896
27.5.4 Double-speed operation ..... 897
27.5.5 CTS and RTS functions ..... 897
27.5.6 Serial data transmission in Manchester mode ..... 898
27.5.7 Serial Data Reception in Manchester Mode ..... 901
27.5.8 Operation When Multi-Processor Bit Is Used ..... 905
27.5.9 Receive Retiming ..... 905
27.5.10 Polarity Setting for Manchester Code ..... 906

27.5.11 Errors in Manchester Mode ..... 907
27.6 Operation in Clock Synchronous Mode ..... 912
27.6.1 Clock ..... 912
27.6.2 CTS and RTS Functions ..... 912
27.6.3 SCI Initialization in Clock Synchronous Mode ..... 913
27.6.4 Serial Data Transmission in Clock Synchronous Mode ..... 914
27.6.5 Serial Data Reception in Clock Synchronous Mode ..... 918
27.6.6 Simultaneous Serial Data Transmission and Reception in Clock Synchronous Mode ..... 923
27.7 Operation in Smart Card Interface Mode ..... 926
27.7.1 Example Connection ..... 926
27.7.2 Data Format (Except in Block Transfer Mode) ..... 926
27.7.3 Block Transfer Mode ..... 928
27.7.4 Receive Data Sampling Timing and Reception Margin ..... 928
27.7.5 SCI Initialization (Smart Card Interface Mode) ..... 929
27.7.6 Serial Data Transmission (Except in Block Transfer Mode) ..... 930
27.7.7 Serial Data Reception (Except in Block Transfer Mode) ..... 932
27.7.8 Clock Output Control ..... 934
27.8 Operation in Simple IIC Mode ..... 935
27.8.1 Generation of Start, Restart, and Stop Conditions ..... 936
27.8.2 Clock Synchronization ..... 937
27.8.3 SDAn Output Delay ..... 938
27.8.4 SCI Initialization in Simple IIC Mode ..... 938
27.8.5 Operation in Master Transmission in Simple IIC Mode ..... 939
27.8.6 Master Reception in Simple IIC Mode ..... 942
27.9 Operation in Simple SPI Mode ..... 943
27.9.1 States of Pins in Master and Slave Modes ..... 944
27.9.2 SS Function in Master Mode ..... 945
27.9.3 SS Function in Slave Mode ..... 945
27.9.4 Relationship between Clock and Transmit/Receive Data ..... 945
27.9.5 SCI Initialization in Simple SPI Mode ..... 946
27.9.6 Transmission and Reception of Serial Data in Simple SPI Mode ..... 946
27.10 Bit Rate Modulation Function ..... 946
27.11 Extended Serial Mode Control Section: Description of Operation ..... 947
27.11.1 Serial Transfer Protocol ..... 947
27.11.2 Transmitting a Start Frame ..... 948
27.11.3 Receiving a Start Frame ..... 951
27.11.4 Detection of Bus Collisions ..... 957
27.11.5 Digital Filter for Input on the RXDXn (n = 1, 2) Pin ..... 958
27.11.6 Bit Rate Measurement ..... 959
27.11.7 Selectable Timing for Sampling Data Received through RXDXn (n = 1, 2) ..... 960

27.11.8 Timer ..... 960
27.12 Interrupt Sources ..... 962
27.12.1 Buffer Operation for SCIn_TXI and SCIn_RXI Interrupts (Non-FIFO Selected) ..... 962
27.12.2 Buffer Operation for SCIn_TXI and SCIn_RXI Interrupts (FIFO Selected) ..... 963
27.12.3 Interrupts in Asynchronous, Manchester, Clock Synchronous, and Simple SPI Modes ..... 963
27.12.4 Interrupts in Smart Card Interface Mode ..... 964
27.12.5 Interrupts in Simple IIC Mode ..... 965
27.12.6 Interrupt Requests from the Extended Serial Mode Control Section ..... 966
27.13 Event Linking ..... 966
27.14 Address Non-match Event Output (SCI0_DCUF) ..... 967
27.15 Noise Cancellation Function ..... 967
27.16 Usage Notes ..... 968
27.16.1 Settings for the Module-Stop Function ..... 968
27.16.2 SCI Operation during Low Power State ..... 968
27.16.3 Break Detection and Processing ..... 973
27.16.4 Mark State and Production of Breaks ..... 974
27.16.5 Receive Error Flags and Transmit Operation in Clock Synchronous Mode and Simple SPI Mode ..... 974
27.16.6 Restrictions on Clock Synchronous Transmission in Clock Synchronous Mode and Simple SPI Mode ..... 974
27.16.7 Restrictions on Using DTC or DMAC ..... 975
27.16.8 Notes on Starting Transfer ..... 976
27.16.9 External Clock Input in Clock Synchronous Mode and Simple SPI Mode ..... 976
27.16.10 Limitations on Simple SPI Mode ..... 976
27.16.11 Notes on Transmit Enable Bit (SCR.TE) ..... 977
27.16.12 Limitation 1 on Usage of the Extended Serial Mode Control Section ..... 977
27.16.13 Limitation 2 on Usage of the Extended Serial Mode Control Section ..... 977
27.16.14 Note on Stopping Reception When Using the RTS Function in Asynchronous Mode ..... 978

**28. I2C Bus Interface (IIC) ..... 979**
28.1 Overview ..... 979
28.2 Register Descriptions ..... 981
28.2.1 ICCR1 : I$^2$C Bus Control Register 1 ..... 981
28.2.2 ICCR2 : I$^2$C Bus Control Register 2 ..... 983
28.2.3 ICMR1 : I$^2$C Bus Mode Register 1 ..... 986
28.2.4 ICMR2 : I$^2$C Bus Mode Register 2 ..... 987
28.2.5 ICMR3 : I$^2$C Bus Mode Register 3 ..... 988
28.2.6 ICFER : I$^2$C Bus Function Enable Register ..... 990
28.2.7 ICSER : I$^2$C Bus Status Enable Register ..... 991
28.2.8 ICIER : I$^2$C Bus Interrupt Enable Register ..... 993

28.2.9 ICSR1 : I2C Bus Status Register 1 ..... 994
28.2.10 ICSR2 : I2C Bus Status Register 2 ..... 996
28.2.11 ICWUR : I2C Bus Wakeup Unit Register ..... 999
28.2.12 ICWUR2 : I2C Bus Wakeup Unit Register 2 ..... 1000
28.2.13 SARLy : Slave Address Register Ly (y = 0 to 2) ..... 1002
28.2.14 SARUy : Slave Address Register Uy (y = 0 to 2) ..... 1002
28.2.15 ICBRL : I2C Bus Bit Rate Low-Level Register ..... 1003
28.2.16 ICBRH : I2C Bus Bit Rate High-Level Register ..... 1003
28.2.17 ICDRT : I2C Bus Transmit Data Register ..... 1005
28.2.18 ICDRR : I2C Bus Receive Data Register ..... 1005
28.2.19 ICDRS : I2C Bus Shift Register ..... 1005
28.3 Operation ..... 1005
28.3.1 Communication Data Format ..... 1005
28.3.2 Initial Settings ..... 1006
28.3.3 Master Transmit Operation ..... 1007
28.3.4 Master Receive Operation ..... 1011
28.3.5 Slave Transmit Operation ..... 1016
28.3.6 Slave Receive Operation ..... 1018
28.4 SCL Synchronization Circuit ..... 1020
28.5 SDA Output Delay Function ..... 1021
28.6 Digital Noise Filter Circuits ..... 1022
28.7 Address Match Detection ..... 1023
28.7.1 Slave-Address Match Detection ..... 1023
28.7.2 Detection of General Call Address ..... 1026
28.7.3 Device-ID Address Detection ..... 1027
28.7.4 Host Address Detection ..... 1028
28.8 Wakeup Function ..... 1029
28.8.1 Normal Wakeup Mode 1 ..... 1030
28.8.2 Normal Wakeup Mode 2 ..... 1033
28.8.3 Command Recovery Mode and EEP Response Mode (Special Wakeup Modes) ..... 1035
28.9 Automatic Low-Hold Function for SCL ..... 1038
28.9.1 Function to Prevent Wrong Transmission of Transmit Data ..... 1038
28.9.2 NACK Reception Transfer Suspension Function ..... 1039
28.9.3 Function to Prevent Failure to Receive Data ..... 1040
28.10 Arbitration-Lost Detection Functions ..... 1041
28.10.1 Master Arbitration-Lost Detection (MALE Bit) ..... 1041
28.10.2 Function to Detect Loss of Arbitration during NACK Transmission (NALE Bit) ..... 1043
28.10.3 Slave Arbitration-Lost Detection (SALE Bit) ..... 1044
28.11 Start, Restart, and Stop Condition Issuing Function ..... 1045

28.11.1 Issuing a Start Condition ..... 1045
28.11.2 Issuing a Restart Condition ..... 1045
28.11.3 Issuing a Stop Condition ..... 1047
28.12 Bus Hanging ..... 1047
28.12.1 Timeout Function ..... 1047
28.12.2 Extra SCL Clock Cycle Output Function ..... 1048
28.12.3 IIC Reset and Internal Reset ..... 1049
28.13 SMBus Operation ..... 1049
28.13.1 SMBus Timeout Measurement ..... 1050
28.13.2 Packet Error Code (PEC) ..... 1051
28.13.3 SMBus Host Notification Protocol (Notify ARP Master Command) ..... 1051
28.14 Interrupt Sources ..... 1051
28.14.1 Buffer Operation for IICn_TXI and IICn_RXI Interrupts ..... 1052
28.15 State of Registers When Issuing Each Condition ..... 1052
28.16 Event Link Output ..... 1053
28.16.1 Interrupt Handling and Event Linking ..... 1053
28.17 Usage Notes ..... 1054
28.17.1 Settings for the Module-Stop Function ..... 1054
28.17.2 Notes on Starting Transfer ..... 1054

**29. Controller Area Network (CAN) Module ..... 1055**
29.1 Overview ..... 1055
29.2 Register Descriptions ..... 1057
29.2.1 CTLR : Control Register ..... 1057
29.2.2 BCR : Bit Configuration Register ..... 1060
29.2.3 MKR[k] : Mask Register k (k = 0 to 7) ..... 1062
29.2.4 FIDCRk : FIFO Received ID Compare Register k (k = 0, 1) ..... 1062
29.2.5 MKIVLR : Mask Invalid Register ..... 1063
29.2.6 Mailbox Registers ..... 1064
29.2.7 MIER : Mailbox Interrupt Enable Register ..... 1067
29.2.8 MIER_FIFO : Mailbox Interrupt Enable Register for FIFO Mailbox Mode ..... 1067
29.2.9 MCTL_TX[j] : Message Control Register for Transmit (j = 0 to 31) ..... 1068
29.2.10 MCTL_RX[j] : Message Control Register for Receive (j = 0 to 31) ..... 1070
29.2.11 RFCR : Receive FIFO Control Register ..... 1072
29.2.12 RFPCR : Receive FIFO Pointer Control Register ..... 1074
29.2.13 TFCR : Transmit FIFO Control Register ..... 1075
29.2.14 TFPCR : Transmit FIFO Pointer Control Register ..... 1076
29.2.15 STR : Status Register ..... 1077
29.2.16 MSMR : Mailbox Search Mode Register ..... 1079
29.2.17 MSSR : Mailbox Search Status Register ..... 1079
29.2.18 CSSR : Channel Search Support Register ..... 1080

29.2.19 AFSR : Acceptance Filter Support Register ..... 1081
29.2.20 EIER : Error Interrupt Enable Register ..... 1082
29.2.21 EIFR : Error Interrupt Factor Judge Register ..... 1083
29.2.22 RECR : Receive Error Count Register ..... 1085
29.2.23 TECR : Transmit Error Count Register ..... 1086
29.2.24 ECSR : Error Code Store Register ..... 1086
29.2.25 TSR : Time Stamp Register ..... 1087
29.2.26 TCR : Test Control Register ..... 1088
29.3 Operation Modes ..... 1089
29.3.1 CAN Reset Mode ..... 1090
29.3.2 CAN Halt Mode ..... 1091
29.3.3 CAN Sleep Mode ..... 1092
29.3.4 CAN Operation Mode (Excluding Bus-Off State) ..... 1092
29.3.5 CAN Operation Mode (Bus-Off State) ..... 1093
29.4 Data Transfer Rate Configuration ..... 1093
29.4.1 Clock Setting ..... 1093
29.4.2 Bit Timing Setting ..... 1094
29.4.3 Data Transfer Rate ..... 1094
29.5 Mailbox and Mask Register Structure ..... 1094
29.6 Acceptance Filtering and Masking Functions ..... 1096
29.7 Reception and Transmission ..... 1098
29.7.1 Reception ..... 1098
29.7.2 Transmission ..... 1100
29.8 Interrupts ..... 1101
29.9 Usage Notes ..... 1102
29.9.1 Settings for the Module-Stop State ..... 1102
29.9.2 Settings for the Operating Clock ..... 1102
**30. Serial Peripheral Interface (SPI) ..... 1103**
30.1 Overview ..... 1103
30.2 Register Descriptions ..... 1105
30.2.1 SPCR : SPI Control Register ..... 1105
30.2.2 SSLP : SPI Slave Select Polarity Register ..... 1107
30.2.3 SPPCR : SPI Pin Control Register ..... 1107
30.2.4 SPSR : SPI Status Register ..... 1108
30.2.5 SPDR/SPDR_HA/SPDR_BY : SPI Data Register ..... 1112
30.2.6 SPSCR : SPI Sequence Control Register ..... 1114
30.2.7 SPSSR : SPI Sequence Status Register ..... 1115
30.2.8 SPBR : SPI Bit Rate Register ..... 1116
30.2.9 SPDCR : SPI Data Control Register ..... 1117
30.2.10 SPCKD : SPI Clock Delay Register ..... 1118

30.2.11 SSLND : SPI Slave Select Negation Delay Register ..... 1119
30.2.12 SPND : SPI Next-Access Delay Register ..... 1119
30.2.13 SPCR2 : SPI Control Register 2 ..... 1120
30.2.14 SPCMDm : SPI Command Register m (m = 0 to 7) ..... 1121
30.2.15 SPDCR2 : SPI Data Control Register 2 ..... 1123
30.2.16 SPCR3 : SPI Control Register 3 ..... 1124
30.3 Operation ..... 1125
30.3.1 Overview of SPI Operation ..... 1125
30.3.2 Controlling the SPI Pins ..... 1126
30.3.3 SPI System Configuration Examples ..... 1127
30.3.4 Data Formats ..... 1133
30.3.5 Transfer Formats ..... 1144
30.3.6 Data Transfer Modes ..... 1146
30.3.7 Transmit Buffer Empty and Receive Buffer Full Interrupts ..... 1148
30.3.8 Communication End Interrupt ..... 1150
30.3.9 Error Detection ..... 1156
30.3.10 Initializing the SPI ..... 1161
30.3.11 SPI Operation ..... 1162
30.3.12 Clock Synchronous Operation ..... 1176
30.3.13 Loopback Mode ..... 1182
30.3.14 Self-Diagnosis of Parity Bit Function ..... 1183
30.3.15 Interrupt Sources ..... 1184
30.4 Event Link Controller Event Output ..... 1185
30.4.1 Receive Buffer Full Event Output ..... 1185
30.4.2 Transmit Buffer Empty Event Output ..... 1185
30.4.3 Mode-Fault, Underrun, Overrun, or Parity Error Event Output ..... 1186
30.4.4 SPI Idle Event Output ..... 1186
30.4.5 Communication End Event Output ..... 1186
30.5 Usage Notes ..... 1188
30.5.1 Settings for the Module-Stop State ..... 1188
30.5.2 Constraint on Low-Power Functions ..... 1188
30.5.3 Constraints on Starting Transfer ..... 1188
30.5.4 Constraints on Mode-Fault, Underrun, Overrun, or Parity Error Event Output ..... 1188
30.5.5 Constraints on the SPSR.SPRF and SPSR.SPTEF Flags ..... 1188
**31. Quad Serial Peripheral Interface (QSPI) ..... 1189**
31.1 Overview ..... 1189
31.2 Register Descriptions ..... 1190
31.2.1 SFMSMD : Transfer Mode Control Register ..... 1190
31.2.2 SFMSSC : Chip Selection Control Register ..... 1191
31.2.3 SFMSKC : Clock Control Register ..... 1192

31.2.4 SFMSST : Status Register ..... 1193
31.2.5 SFMCOM : Communication Port Register ..... 1194
31.2.6 SFMCMD : Communication Mode Control Register ..... 1195
31.2.7 SFMCST : Communication Status Register ..... 1195
31.2.8 SFMSIC : Instruction Code Register ..... 1196
31.2.9 SFMSAC : Address Mode Control Register ..... 1196
31.2.10 SFMSDC : Dummy Cycle Control Register ..... 1197
31.2.11 SFMSPC : SPI Protocol Control Register ..... 1198
31.2.12 SFMPMD : Port Control Register ..... 1198
31.2.13 SFMCNT1 : External QSPI Address Register ..... 1199
31.3 Memory Map ..... 1199
31.3.1 External Bus Space ..... 1199
31.3.2 Address Width of the SPI Space and SPI Bus ..... 1200
31.4 SPI Bus ..... 1201
31.4.1 SPI Protocol ..... 1201
31.4.2 SPI Mode ..... 1204
31.5 SPI Bus Timing Adjustment ..... 1205
31.5.1 SPI Bus Reference Cycles ..... 1205
31.5.2 QSPCLK Signal Duty Ratio ..... 1206
31.5.3 Minimum High-Level Width for the QSSL Signal ..... 1206
31.5.4 QSSL Signal Setup Time ..... 1207
31.5.5 QSSL Signal Hold Time ..... 1207
31.5.6 Hold Time of the Serial Data Output Enable ..... 1207
31.5.7 Setup Time for Serial Data Output ..... 1208
31.5.8 Hold Time for Serial Data Output ..... 1209
31.5.9 Serial Data Receiving Latency ..... 1209
31.6 SPI Instruction Set Used for Serial Flash Memory Access ..... 1210
31.6.1 SPI Instructions That Are Automatically Generated ..... 1210
31.6.2 Standard Read Instruction ..... 1211
31.6.3 Fast Read Instruction ..... 1212
31.6.4 Fast Read Dual Output Instruction ..... 1213
31.6.5 Fast Read Dual I/O Instruction ..... 1214
31.6.6 Fast Read Quad Output Instruction ..... 1215
31.6.7 Fast Read Quad I/O Instruction ..... 1216
31.6.8 Enter 4-Byte Mode Instruction ..... 1217
31.6.9 Exit 4-Byte Mode Instruction ..... 1218
31.6.10 Write Enable Instruction ..... 1218
31.7 SPI Bus Cycle Arrangement ..... 1219
31.7.1 Serial Flash Memory Read Based on Individual Conversion ..... 1219
31.7.2 Serial Flash Memory Read Using the Prefetch Function ..... 1219

31.7.3 Halt of Prefetching ... 1220
31.7.4 Direct Specification of Prefetch Destination ... 1220
31.7.5 Prefetch State Polling ... 1220
31.7.6 SPI Bus Cycle Extension Function ... 1221
31.8 XIP Control ... 1221
31.8.1 Setting XIP Mode ... 1222
31.8.2 Releasing the XIP Mode ... 1223
31.9 QIO2 and QIO3 Pin States ... 1224
31.10 Direct Communication Mode ... 1225
31.10.1 About Direct Communication ... 1225
31.10.2 Using Direct Communication Mode ... 1225
31.10.3 Generating the SPI Bus Cycle during Direct Communication ... 1225
31.11 Interrupts ... 1229
31.12 Usage Note ... 1229
31.12.1 Settings for the Module-Stop Function ... 1229
31.12.2 Procedure for Changing Settings in Multiple Control Registers ... 1229
**32. Serial Sound Interface Enhanced (SSIE) ... 1231**
32.1 Overview ... 1231
32.2 Features ... 1231
32.3 Block Diagram ... 1232
32.4 Register Descriptions ... 1234
32.4.1 SSICR : Control Register ... 1234
32.4.2 SSISR : Status Register ... 1244
32.4.3 SSIFCR : FIFO Control Register ... 1255
32.4.4 SSIFSR : FIFO Status Register ... 1261
32.4.5 SSIFTDR : Transmit FIFO Data Register ... 1264
32.4.6 SSIFRDR : Receive FIFO Data Register ... 1267
32.4.7 SSIOFR : Audio Format Register ... 1269
32.4.8 SSISCR : Status Control Register ... 1272
32.5 Communication Formats ... 1273
32.5.1 I2S Format ... 1274
32.5.2 Monaural Format ... 1274
32.5.3 TDM Format ... 1276
32.6 Communication Modes ... 1277
32.6.1 Slave-mode Communication ... 1277
32.6.2 Master-mode Communication ... 1277
32.6.3 Transmission ... 1278
32.6.4 Reception ... 1278
32.6.5 Transmission and Reception ... 1278
32.7 Operation ... 1278

32.7.1 Idle State ..... 1278
32.7.2 Communication States ..... 1280
32.8 Communication Operation ..... 1284
32.8.1 Start Communication ..... 1284
32.8.2 Transmission ..... 1285
32.8.3 Reception ..... 1286
32.8.4 Transmission and Reception ..... 1287
32.8.5 Halt Communication ..... 1287
32.8.6 Error Handling ..... 1288
32.8.7 Resume Communication ..... 1289
32.9 Interrupts ..... 1290
32.9.1 SSIE0_SSIF Interrupt ..... 1291
32.9.2 SSIE0_SSITXI Interrupt ..... 1292
32.9.3 SSIE0_SSIRXI Interrupt ..... 1292
32.10 Software Resets ..... 1292
32.10.1 Software Reset Procedure ..... 1292
32.11 Notes ..... 1293
32.11.1 Notes for Slave-mode Communication ..... 1293
32.11.2 Notes for Master-mode Communication ..... 1294
32.11.3 Notes for Communication Flow ..... 1294
32.11.4 Write Access Restriction ..... 1295

**33. SD/MMC Host Interface (SDHI) ..... 1297**

33.1 Overview ..... 1297
33.2 Register Descriptions ..... 1298
33.2.1 SD_CMD : Command Type Register ..... 1298
33.2.2 SD_ARG : SD Command Argument Register ..... 1299
33.2.3 SD_ARG1 : SD Command Argument Register 1 ..... 1299
33.2.4 SD_STOP : Data Stop Register ..... 1300
33.2.5 SD_SECCNT : Block Count Register ..... 1301
33.2.6 SD_RSPi : SD Card Response Register i (i = 10, 32, 54) ..... 1301
33.2.7 SD_RSPj : SD Card Response Register j (j = 1, 3, 5) ..... 1302
33.2.8 SD_RSP76 : SD Card Response Register 76 ..... 1302
33.2.9 SD_RSP7 : SD Card Response Register 7 ..... 1302
33.2.10 SD_INFO1 : SD Card Interrupt Flag Register 1 ..... 1303
33.2.11 SD_INFO2 : SD Card Interrupt Flag Register 2 ..... 1306
33.2.12 SD_INFO1_MASK : SD INFO1 Interrupt Mask Register ..... 1310
33.2.13 SD_INFO2_MASK : SD INFO2 Interrupt Mask Register ..... 1311
33.2.14 SD_CLK_CTRL : SD Clock Control Register ..... 1312
33.2.15 SD_SIZE : Transfer Data Length Register ..... 1313
33.2.16 SD_OPTION : SD Card Access Control Option Register ..... 1313

33.2.17 SD_ERR_STS1 : SD Error Status Register 1 ..... 1315
33.2.18 SD_ERR_STS2 : SD Error Status Register 2 ..... 1316
33.2.19 SD_BUF0 : SD Buffer Register ..... 1317
33.2.20 SDIO_MODE : SDIO Mode Control Register ..... 1317
33.2.21 SDIO_INFO1 : SDIO Interrupt Flag Register ..... 1319
33.2.22 SDIO_INFO1_MASK : SDIO INFO1 Interrupt Mask Register ..... 1320
33.2.23 SD_DMAEN : DMA Mode Enable Register ..... 1321
33.2.24 SOFT_RST : Software Reset Register ..... 1321
33.2.25 SDIF_MODE : SD Interface Mode Setting Register ..... 1322
33.2.26 EXT_SWAP : Swap Control Register ..... 1323
33.3 Operation ..... 1323
33.3.1 SD/MMC Interface ..... 1323
33.3.2 Card Detect/Write Protect ..... 1324
33.3.3 Interrupt Request and DMA Transfer Request ..... 1326
33.3.4 Communication Errors and Timeouts ..... 1327
33.3.5 Command without Data Transfer (SD/MMC) ..... 1329
33.3.6 Single Block Read (SD/MMC) ..... 1330
33.3.7 Single Block Write (SD/MMC) ..... 1332
33.3.8 Multiple Block Read (SD/MMC) ..... 1333
33.3.9 Multiple Block Write (SD/MMC Using Internal Timer) ..... 1335
33.3.10 Multiple Block Write (MMC using external timer) ..... 1337
33.3.11 IO_RW_DIRECT Command (SD: CMD52) ..... 1339
33.3.12 IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Read) ..... 1339
33.3.13 IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Write) ..... 1341
33.3.14 DMA Transfer (SD/MMC) ..... 1343
33.3.15 Example of SD_CMD Register Setting ..... 1345
33.4 Usage Notes ..... 1349
33.4.1 SD_BUF Illegal Write Access (SD/MMC) ..... 1349
33.4.2 Block Number Constraint for Multiple Block Read (SD) ..... 1349
33.4.3 Automatic Control of SD/MMC Clock Output (SD/MMC) ..... 1349
33.4.4 Control of the C52PUB Setting for Multiple Block Write (SD) ..... 1350
33.4.5 Notes on SD_CLK_CTRL Register Settings (SD/MMC) ..... 1350
33.4.6 Specification Limitations ..... 1350
33.4.7 STP Bit Setting during Multiple Block Read (SD/MMC) ..... 1350
33.4.8 Register Setting Notes ..... 1351

**34. Cyclic Redundancy Check (CRC) ..... 1352**

34.1 Overview ..... 1352
34.2 Register Descriptions ..... 1353
34.2.1 CRCCR0 : CRC Control Register 0 ..... 1353
34.2.2 CRCDIR/CRCDIR_BY : CRC Data Input Register ..... 1353

34.2.3 CRCDOR/CRCDOR_HA/CRCDOR_BY : CRC Data Output Register ..... 1354
34.3 Operation ..... 1354
34.3.1 Basic Operation ..... 1354
34.4 Usage Notes ..... 1357
34.4.1 Settings for the Module-Stop State ..... 1357
34.4.2 Note on Transmission ..... 1357
**35. Boundary Scan ..... 1359**
35.1 Overview ..... 1359
35.2 Register Descriptions ..... 1359
35.2.1 JTIR : Instruction Register ..... 1360
35.2.2 JTIDR : ID Code Register ..... 1360
35.2.3 JTBPR : Bypass Register ..... 1361
35.2.4 JTBSR : Boundary Scan Register ..... 1361
35.3 Operation ..... 1361
35.3.1 TAP Controller ..... 1361
35.3.2 Commands ..... 1362
35.4 Usage Notes ..... 1363
**36. Secure Cryptographic Engine (SCE9) ..... 1365**
36.1 Overview ..... 1365
36.2 Operation ..... 1366
36.2.1 Encryption Engine ..... 1366
36.2.2 Encryption and Decryption ..... 1367
36.3 Usage Notes ..... 1368
36.3.1 Software Standby Mode ..... 1368
36.3.2 Module-Stop Function Setting ..... 1368
**37. 12-Bit A/D Converter (ADC12) ..... 1369**
37.1 Overview ..... 1369
37.2 Register Descriptions ..... 1372
37.2.1 ADDRn : A/D Data Registers n (n = 0 to 8, 11 to 13, 16) ..... 1372
37.2.2 ADDBLDR : A/D Data Duplexing Register ..... 1374
37.2.3 ADDBLDRn : A/D Data Duplexing Register n (n = A, B) ..... 1375
37.2.4 ADTSDR : A/D Temperature Sensor Data Register ..... 1376
37.2.5 ADOCDR : A/D Internal Reference Voltage Data Register ..... 1378
37.2.6 ADRD : A/D Self-Diagnosis Data Register ..... 1379
37.2.7 ADCSR : A/D Control Register ..... 1380
37.2.8 ADANSA0 : A/D Channel Select Register A0 ..... 1383
37.2.9 ADANSA1 : A/D Channel Select Register A1 ..... 1383
37.2.10 ADANSB0 : A/D Channel Select Register B0 ..... 1384
37.2.11 ADANSB1 : A/D Channel Select Register B1 ..... 1384

37.2.12 ADADS0 : A/D-Converted Value Addition/Average Channel Select Register 0 ..... 1385
37.2.13 ADADS1 : A/D-Converted Value Addition/Average Channel Select Register 1 ..... 1386
37.2.14 ADADC : A/D-Converted Value Addition/Average Count Select Register ..... 1387
37.2.15 ADCER : A/D Control Extended Register ..... 1388
37.2.16 ADSTRGR : A/D Conversion Start Trigger Select Register ..... 1389
37.2.17 ADEXICR : A/D Conversion Extended Input Control Registers ..... 1390
37.2.18 ADSSTRn/ADSSTRL/ADSSTRT/ADSSTRO : A/D Sampling State Register (n = 0 to 8, 11 to 13) ..... 1392
37.2.19 ADDISCR : A/D Disconnection Detection Control Register ..... 1392
37.2.20 ADGSPCR : A/D Group Scan Priority Control Register ..... 1393
37.2.21 ADCMPCR : A/D Compare Function Control Register ..... 1395
37.2.22 ADCMPANSR0 : A/D Compare Function Window A Channel Select Register 0 ..... 1396
37.2.23 ADCMPANSR1 : A/D Compare Function Window A Channel Select Register 1 ..... 1397
37.2.24 ADCMPANSER : A/D Compare Function Window A Extended Input Select Register ..... 1397
37.2.25 ADCMPLR0 : A/D Compare Function Window A Comparison Condition Setting Register 0 ..... 1398
37.2.26 ADCMPLR1 : A/D Compare Function Window A Comparison Condition Setting Register 1 ..... 1399
37.2.27 ADCMPLER : A/D Compare Function Window A Extended Input Comparison Condition Setting Register ..... 1400
37.2.28 ADCMPDRn : A/D Compare Function Window A Lower-Side/Upper-Side Level Setting Register (n = 0, 1) ..... 1401
37.2.29 ADWINnLB : A/D Compare Function Window B Lower-Side/Upper-Side Level Setting Register (n = L, U) ..... 1402
37.2.30 ADCMPSR0 : A/D Compare Function Window A Channel Status Register 0 ..... 1404
37.2.31 ADCMPSR1 : A/D Compare Function Window A Channel Status Register1 ..... 1404
37.2.32 ADCMPSER : A/D Compare Function Window A Extended Input Channel Status Register ..... 1405
37.2.33 ADCMPBNSR : A/D Compare Function Window B Channel Select Register ..... 1406
37.2.34 ADCMPBSR : A/D Compare Function Window B Status Register ..... 1407
37.2.35 ADWINMON : A/D Compare Function Window A/B Status Monitor Register ..... 1408
37.2.36 ADBUFEN : A/D Data Buffer Enable Register ..... 1409
37.2.37 ADBUFPTR : A/D Data Buffer Pointer Register ..... 1410
37.2.38 ADBUFn : A/D Data Buffer Registers n (n = 0 to 15) ..... 1410
37.3 Operation ..... 1411
37.3.1 Scanning Operation ..... 1411
37.3.2 Single Scan Mode ..... 1412
37.3.3 Continuous Scan Mode ..... 1417
37.3.4 Group Scan Mode ..... 1420
37.3.5 Compare Function for Windows A and B ..... 1429
37.3.6 Analog Input Sampling and Scan Conversion Time ..... 1433
37.3.7 Usage Example of A/D Data Register Automatic Clearing Function ..... 1435

37.3.8 A/D-Converted Value Addition/Average Mode ..... 1436
37.3.9 Disconnection Detection Assist Function ..... 1436
37.3.10 Starting A/D Conversion with an Asynchronous Trigger ..... 1438
37.3.11 Starting A/D Conversion with a Synchronous Trigger from a Peripheral Module ..... 1439
37.3.12 Using Data Buffers ..... 1439
37.4 Interrupt Sources and DTC, DMAC Transfer Requests ..... 1440
37.4.1 Interrupt Requests ..... 1440
37.5 Event Link Function ..... 1442
37.5.1 Event Output to the ELC ..... 1442
37.5.2 ADC12 Operation through an Event from the ELC ..... 1442
37.6 Usage Notes ..... 1443
37.6.1 Constraints on Setting the Registers ..... 1443
37.6.2 Constraints on Reading the Data Registers ..... 1443
37.6.3 Constraints on Stopping A/D Conversion ..... 1443
37.6.4 A/D Conversion Restart and Termination Timing ..... 1445
37.6.5 Constraints on Scan End Interrupt Handling ..... 1445
37.6.6 Settings for the Module-Stop Function ..... 1445
37.6.7 Notes on Entering the Low-Power States ..... 1445
37.6.8 Error in Absolute Accuracy When Disconnection Detection Assistance Is in Use ..... 1445
37.6.9 Constraints on Operating Modes and Status Bits ..... 1445
37.6.10 Notes on Board Design ..... 1446
37.6.11 Constraints on Noise Prevention ..... 1446
37.6.12 Port Settings When Using the ADC12 Input ..... 1446
37.6.13 Notes on Canceling Software Standby Mode ..... 1446
37.6.14 Calculation for Sampling Time ..... 1447

**38. 12-Bit D/A Converter (DAC12) ..... 1448**

38.1 Overview ..... 1448
38.2 Register Descriptions ..... 1449
38.2.1 DADRn : D/A Data Register n (n = 0, 1) ..... 1449
38.2.2 DACR : D/A Control Register ..... 1449
38.2.3 DADPR : DADRn Format Select Register ..... 1451
38.2.4 DAADSCR : D/A A/D Synchronous Start Control Register ..... 1451
38.2.5 DAAMPCR : D/A Output Amplifier Control Register ..... 1452
38.2.6 DAASWCR : D/A Amplifier Stabilization Wait Control Register ..... 1452
38.2.7 DAADUSR : D/A A/D Synchronous Unit Select Register ..... 1453
38.3 Operation ..... 1453
38.3.1 Reducing Interference between D/A and A/D Conversion ..... 1454
38.4 Event Link Operation Setting Procedure ..... 1456
38.4.1 DA0 Event Link Operation Setting Procedure ..... 1456
38.4.2 DA1 Event Link Operation Setting Procedure ..... 1456

38.5 Usage Notes on Event Link Operation ..... 1456
38.6 Usage Notes ..... 1456
38.6.1 Settings for the Module-Stop Function ..... 1456
38.6.2 DAC12 Operation in the Module-Stop State ..... 1456
38.6.3 DAC12 Operation in Software Standby Mode ..... 1457
38.6.4 Constraint on Entering Deep Software Standby Mode ..... 1457
38.6.5 Initialization Procedure with the Output Amplifier ..... 1457
38.6.6 Constraint on Usage When Interference Reduction between D/A and A/D Conversion Is Enabled ..... 1457

**39. Temperature Sensor (TSN) ..... 1458**
39.1 Overview ..... 1458
39.2 Register Descriptions ..... 1459
39.2.1 TSCR : Temperature Sensor Control Register ..... 1459
39.2.2 TSCDR : Temperature Sensor Calibration Data Register ..... 1459
39.3 Using the Temperature Sensor ..... 1460
39.3.1 Preparation for Using the Temperature Sensor ..... 1460
39.3.2 Procedures for Using the Temperature Sensor ..... 1460
39.4 Usage Notes ..... 1462
39.4.1 Settings for the Module-Stop Function ..... 1462

**40. Capacitive Touch Sensing Unit (CTSU) ..... 1463**
40.1 Overview ..... 1463
40.2 Register Descriptions ..... 1465
40.2.1 CTSUCR0 : CTSU Control Register 0 ..... 1465
40.2.2 CTSUCR1 : CTSU Control Register 1 ..... 1467
40.2.3 CTSUSDPRS : CTSU Synchronous Noise Reduction Setting Register ..... 1468
40.2.4 CTSUSST : CTSU Sensor Stabilization Wait Control Register ..... 1469
40.2.5 CTSUMCH0 : CTSU Measurement Channel Register 0 ..... 1469
40.2.6 CTSUMCH1 : CTSU Measurement Channel Register 1 ..... 1470
40.2.7 CTSUCHAC0 : CTSU Channel Enable Control Register 0 ..... 1470
40.2.8 CTSUCHAC1 : CTSU Channel Enable Control Register 1 ..... 1471
40.2.9 CTSUCHTRC0 : CTSU Channel Transmit/Receive Control Register 0 ..... 1471
40.2.10 CTSUCHTRC1 : CTSU Channel Transmit/Receive Control Register 1 ..... 1471
40.2.11 CTSUDCLKC : CTSU High-Pass Noise Reduction Control Register ..... 1472
40.2.12 CTSUST : CTSU Status Register ..... 1472
40.2.13 CTSUSSC : CTSU High-Pass Noise Reduction Spectrum Diffusion Control Register ..... 1474
40.2.14 CTSUSO0 : CTSU Sensor Offset Register 0 ..... 1475
40.2.15 CTSUSO1 : CTSU Sensor Offset Register 1 ..... 1475
40.2.16 CTSUSC : CTSU Sensor Counter ..... 1477
40.2.17 CTSURC : CTSU Reference Counter ..... 1477
40.2.18 CTSUERRS : CTSU Error Status Register ..... 1478

40.2.19 CTSUTRMR : CTSU Reference Current Calibration Register ..... 1479
40.3 Operation ..... 1479
40.3.1 Principles of Measurement Operation ..... 1479
40.3.2 Measurement Modes ..... 1481
40.3.3 Parameters Common to Multiple Modes ..... 1490
40.4 Usage Notes ..... 1492
40.4.1 Measurement Result Data (CTSUSC and CTSURC Counters) ..... 1492
40.4.2 Constraint on Software Trigger ..... 1492
40.4.3 Constraints on External Triggers ..... 1493
40.4.4 Constraints on Forced Stops ..... 1493
40.4.5 TSCAP Pin ..... 1493
40.4.6 Constraints on Measurement Operation (CTSUCR0.CTSUSTRT Bit = 1) ..... 1493
**41. Data Operation Circuit (DOC) ..... 1494**
41.1 Overview ..... 1494
41.2 DOC Register Descriptions ..... 1494
41.2.1 DOCR : DOC Control Register ..... 1494
41.2.2 DODIR : DOC Data Input Register ..... 1495
41.2.3 DODSR : DOC Data Setting Register ..... 1496
41.3 Operation ..... 1496
41.3.1 Data Comparison Mode ..... 1496
41.3.2 Data Addition Mode ..... 1496
41.3.3 Data Subtraction Mode ..... 1497
41.4 Interrupt Source ..... 1497
41.5 Output of an Event Signal to the Event Link Controller (ELC) ..... 1498
41.6 Usage Notes ..... 1498
41.6.1 Settings for the Module-Stop State ..... 1498
**42. SRAM ..... 1499**
42.1 Overview ..... 1499
42.2 Register Descriptions ..... 1499
42.2.1 SRAMSAR : SRAM Security Attribution Register ..... 1499
42.2.2 PARIOAD : SRAM Parity Error Operation After Detection Register ..... 1500
42.2.3 SRAMPRCR : SRAM Protection Register ..... 1501
42.2.4 SRAMWTSC : SRAM Wait State Control Register ..... 1501
42.2.5 SRAMPRCR2 : SRAM Protection Register 2 ..... 1502
42.2.6 ECCMODE : ECC Operating Mode Control Register ..... 1502
42.2.7 ECC2STS : ECC 2-Bit Error Status Register ..... 1503
42.2.8 ECC1STSEN : ECC 1-Bit Error Information Update Enable Register ..... 1503
42.2.9 ECC1STS : ECC 1-Bit Error Status Register ..... 1504
42.2.10 ECCPRCR : ECC Protection Register ..... 1504

42.2.11 ECCPRCR2 : ECC Protection Register 2 ..... 1505
42.2.12 ECCETST : ECC Test Control Register ..... 1505
42.2.13 ECCOAD : SRAM ECC Error Operation After Detection Register ..... 1506
42.3 Operation ..... 1506
42.3.1 Module Stop Function ..... 1506
42.3.2 Correction of ECC errors ..... 1507
42.3.3 ECC Error Interrupt Function ..... 1507
42.3.4 ECC Decoder Testing ..... 1507
42.3.5 Parity Calculation Function ..... 1508
42.3.6 TrustZone Filter function ..... 1510
42.3.7 Interrupt Source ..... 1511
42.3.8 Wait state ..... 1511
42.3.9 Access Cycle ..... 1512
42.3.10 ECC encode specification ..... 1512
**43. Standby SRAM ..... 1513**
43.1 Overview ..... 1513
43.2 Register Descriptions ..... 1513
43.2.1 STBRAMSAR : Standby RAM memory Security Attribution Register ..... 1513
43.3 Operation ..... 1514
43.3.1 Data Retention ..... 1514
43.3.2 Setting for the Module-stop Function ..... 1514
43.3.3 Parity Calculation Function ..... 1514
43.3.4 TrustZone Filter function ..... 1515
43.3.5 Access Cycle ..... 1515
43.4 Usage Notes ..... 1515
43.4.1 Instruction Fetch from the Standby SRAM Area ..... 1515
**44. Flash Memory ..... 1516**
44.1 Overview ..... 1516
44.2 Structure of Memory ..... 1518
44.3 Address Space ..... 1519
44.4 Register Descriptions ..... 1520
44.4.1 FCACHEE : Flash Cache Enable Register ..... 1520
44.4.2 FCACHEIV : Flash Cache Invalidate Register ..... 1520
44.4.3 FLWT : Flash Wait Cycle Register ..... 1521
44.4.4 FSAR : Flash Security Attribution Register ..... 1521
44.4.5 UIDRn : Unique ID Registers n (n = 0 to 3) ..... 1522
44.4.6 PNRn : Part Numbering Register n (n = 0 to 3) ..... 1522
44.4.7 MCUVER : MCU Version Register ..... 1523
44.4.8 FWEPROR : Flash P/E Protect Register ..... 1523

44.4.9 FASTAT : Flash Access Status Register ..... 1524
44.4.10 FAEINT : Flash Access Error Interrupt Enable Register ..... 1525
44.4.11 FRDYIE : Flash Ready Interrupt Enable Register ..... 1526
44.4.12 FSADDR : FACI Command Start Address Register ..... 1526
44.4.13 FEADDR : FACI Command End Address Register ..... 1527
44.4.14 FMEPROT : Flash P/E Mode Entry Protection Register ..... 1527
44.4.15 FBPROT0 : Flash Block Protection Register ..... 1528
44.4.16 FBPROT1 : Flash Block Protection for Secure Register ..... 1529
44.4.17 FSTATR : Flash Status Register ..... 1529
44.4.18 FENTRYR : Flash P/E Mode Entry Register ..... 1533
44.4.19 FSUINITR : Flash Sequencer Setup Initialization Register ..... 1534
44.4.20 FCMDR : FACI Command Register ..... 1535
44.4.21 FBCCNT : Blank Check Control Register ..... 1536
44.4.22 FBCSTAT : Blank Check Status Register ..... 1536
44.4.23 FPSADDR : Data Flash Programming Start Address Register ..... 1536
44.4.24 FSUASMON : Flash Startup Area Select Monitor Register ..... 1537
44.4.25 FCPSR : Flash Sequencer Processing Switching Register ..... 1538
44.4.26 FPCKAR : Flash Sequencer Processing Clock Notification Register ..... 1538
44.4.27 FSUACR : Flash Startup Area Control Register ..... 1539
44.4.28 FCKMHZ : Data Flash Access Frequency Register ..... 1539
44.5 Flash Cache ..... 1540
44.5.1 Feature of flash cache ..... 1540
44.6 Operating Modes Associated with Flash Memory ..... 1541
44.7 Overview of Functions ..... 1542
44.8 Operating Modes of the Flash Sequencer ..... 1543
44.9 FACI Commands ..... 1544
44.9.1 List of FACI Commands ..... 1544
44.9.2 Relationship between the Flash Sequencer State and FACI Commands ..... 1545
44.9.3 Usage of FACI Commands ..... 1547
44.10 Suspend Operation ..... 1564
44.11 Protection Function ..... 1564
44.11.1 Software Protection ..... 1564
44.11.2 Error Protection ..... 1566
44.11.3 Start-Up Program Protection ..... 1568
44.12 Security Function ..... 1572
44.12.1 Security Flag for Startup Area Select ..... 1572
44.12.2 Permanent Block Protect Setting ..... 1573
44.12.3 Flash Memory Protection for TrustZone ..... 1574
44.13 Boot Mode ..... 1582
44.13.1 Boot Mode (for the SCI Interface) ..... 1583

44.13.2 Boot Mode (for the USB Interface) ..... 1584
44.14 Using the Serial Programmer for Rewriting ..... 1584
44.14.1 Environments for Serial Programming ..... 1584
44.15 Programming through Self-Programming ..... 1585
44.15.1 Overview ..... 1585
44.15.2 Background Operation ..... 1586
44.16 Reading Flash Memory ..... 1586
44.16.1 Reading Code Flash Memory ..... 1586
44.16.2 Reading Data Flash Memory ..... 1586
44.16.3 Access Cycle ..... 1586
44.17 Usage Notes ..... 1587
**45. Internal Voltage Regulator ..... 1589**
45.1 Overview ..... 1589
45.2 Operation ..... 1589
**46. Security Features ..... 1590**
46.1 Features ..... 1590
46.2 Arm TrustZone Security ..... 1590
46.2.1 Arm TrustZone Technology ..... 1590
46.2.2 Memory Security Attribution ..... 1590
46.2.3 Peripheral Security Attribution ..... 1592
46.2.4 Flash Sequencer Security Attribution ..... 1592
46.2.5 Address Space Security Attribution ..... 1593
46.2.6 TrustZone Access Error ..... 1593
46.3 Device Lifecycle Management ..... 1594
46.3.1 Changing the Lifecycle State ..... 1595
46.3.2 Debug access level ..... 1596
46.3.3 Serial Programming ..... 1596
46.3.4 Lifecycle changing example ..... 1596
46.3.5 Failure analysis ..... 1596
46.4 Key Injection ..... 1597
46.5 Register Description ..... 1599
46.5.1 PSARB : Peripheral Security Attribution Register B ..... 1599
46.5.2 PSARC : Peripheral Security Attribution Register C ..... 1600
46.5.3 PSARD : Peripheral Security Attribution Register D ..... 1601
46.5.4 PSARE : Peripheral Security Attribution Register E ..... 1602
46.5.5 MSSAR : Module Stop Security Attribution Register ..... 1604
46.5.6 CFSAMONA : Code Flash Security Attribution Monitor Register A ..... 1604
46.5.7 CFSAMONB : Code Flash Security Attribution Monitor Register B ..... 1605
46.5.8 DFSAMON : Data Flash Security Attribution Monitor Register ..... 1605

46.5.9 SSAMONA : SRAM Security Attribution Monitor Register A ..... 1605
46.5.10 SSAMONB : SRAM Security Attribution Monitor Register B ..... 1606
46.5.11 DLMMON : Device Lifecycle Management State Monitor Register ..... 1606
46.5.12 TZFSAR : TrustZone Filter Security Attribution Register ..... 1607
46.5.13 TZFOAD : TrustZone Filter Operation After Detection Register ..... 1607
46.5.14 TZFPT : TrustZone Filter Protect Register ..... 1608
46.6 Usage Notes ..... 1608
46.6.1 Restrictions on setting the security attribution ..... 1608
46.6.2 SAU setting ..... 1609
46.6.3 Non-secure exception during the setting of FACI registers ..... 1609
46.6.4 FCU interrupt usage ..... 1609
**47. Electrical Characteristics ..... 1610**
47.1 Absolute Maximum Ratings ..... 1610
47.2 DC Characteristics ..... 1611
47.2.1 Tj/Ta Definition ..... 1611
47.2.2 I/O VIH, VIL ..... 1611
47.2.3 I/O IOH, IOL ..... 1613
47.2.4 I/O VOH, VOL, and Other Characteristics ..... 1614
47.2.5 Operating and Standby Current ..... 1615
47.2.6 VCC Rise and Fall Gradient and Ripple Frequency ..... 1618
47.2.7 Thermal Characteristics ..... 1619
47.3 AC Characteristics ..... 1623
47.3.1 Frequency ..... 1623
47.3.2 Clock Timing ..... 1624
47.3.3 Reset Timing ..... 1626
47.3.4 Wakeup Timing ..... 1627
47.3.5 NMI and IRQ Noise Filter ..... 1630
47.3.6 I/O Ports, POEG, GPT, AGT, and ADC12 Trigger Timing ..... 1631
47.3.7 CAC Timing ..... 1633
47.3.8 SCI Timing ..... 1634
47.3.9 SPI Timing ..... 1640
47.3.10 QSPI Timing ..... 1644
47.3.11 IIC Timing ..... 1646
47.3.12 SSIE Timing ..... 1649
47.3.13 SD/MMC Host Interface Timing ..... 1651
47.4 USB Characteristics ..... 1652
47.4.1 USBFS Timing ..... 1652
47.5 ADC12 Characteristics ..... 1654
47.6 DAC12 Characteristics ..... 1657
47.7 TSN Characteristics ..... 1657

47.8 OSC Stop Detect Characteristics ..... 1657
47.9 POR and LVD Characteristics ..... 1658
47.10 VBATT Characteristics ..... 1661
47.11 CTSU Characteristics ..... 1662
47.12 Flash Memory Characteristics ..... 1662
47.12.1 Code Flash Memory Characteristics ..... 1662
47.12.2 Data Flash Memory Characteristics ..... 1664
47.12.3 Option Setting Memory Characteristics ..... 1665
47.13 Boundary Scan ..... 1666
47.14 Joint Test Action Group (JTAG) ..... 1667
47.15 Serial Wire Debug (SWD) ..... 1668
47.16 Embedded Trace Macro Interface (ETM) ..... 1669

**Appendix 1. Port States in Each Processing Mode ..... 1671**

**Appendix 2. Package Dimensions ..... 1672**

**Appendix 3. I/O Registers ..... 1676**
3.1 Peripheral Base Addresses ..... 1676
3.2 Access Cycles ..... 1678

**Revision History ..... 1680**

Leading-performance 100 MHz Arm Cortex-M33 core, up to 512 KB code flash memory with background operation, 8 KB Data flash memory, and 128 KB SRAM with Parity/ECC. High-integration with USB 2.0 Full-Speed, SDHI, Quad SPI, and advanced analog. Integrated Secure Crypto Engine with cryptography accelerators, key management support, tamper detection and power analysis resistance in concert with Arm TrustZone for integrated secure element functionality.

# Features

## Arm® Cortex®-M33 Core

- Armv8-M architecture with the main extension
- Maximum operating frequency: 100 MHz
- Arm Memory Protection Unit (Arm MPU)
  - Protected Memory System Architecture (PMSAv8)
  - Secure MPU (MPU_S): 8 regions
  - Non-secure MPU (MPU_NS): 8 regions
- SysTick timer
  - Embeds two Systick timers: Secure and Non-secure instance
  - Driven by LOCO or system clock
- CoreSight™ ETM-M33

## Memory

- Up to 512-KB code flash memory
- 8-KB data flash memory (100,000 program/erase (P/E) cycles)
- 128-KB SRAM

## Connectivity

- Serial Communications Interface (SCI) × 6
  - Asynchronous interfaces
  - 8-bit clock synchronous interface
  - Smart card interface
  - Simple IIC
  - Simple SPI
  - Manchester coding (SCI3, SCI4)
- $I^2C$ bus interface (IIC) × 2
- Serial Peripheral Interface (SPI)
- Quad Serial Peripheral Interface (QSPI)
- USB 2.0 Full-Speed Module (USBFS)
- Control Area Network module (CAN)
- SD/MMC Host Interface (SDHI)
- Serial Sound Interface Enhanced (SSIE)

## Analog

- 12-bit A/D Converter (ADC12)
- 12-bit D/A Converter (DAC12) × 2
- Temperature Sensor (TSN)

## Timers

- General PWM Timer 32-bit (GPT32) × 4
- General PWM Timer 16-bit (GPT16) × 4
- Low Power Asynchronous General Purpose Timer (AGT) × 6

## Security and Encryption

- Secure Crypto Engine 9
  - Symmetric algorithms: AES
  - Asymmetric algorithms: RSA, ECC, and DSA
  - Hash-value generation: SHA224, SHA256, GHASH
  - 128-bit unique ID
- Arm® TrustZone®
  - Up to three regions for the code flash
  - Up to two regions for the data flash
  - Up to three regions for the SRAM
  - Individual secure or non-secure security attribution for each peripheral
- Device lifecyle management
- Pin function
  - Up to three tamper pins
  - Secure pin multiplexing

## System and Power Management

- Low power modes
- Battery backup function (VBATT)
- Realtime Clock (RTC) with calendar and VBATT support
- Event Link Controller (ELC)
- Data Transfer Controller (DTC)
- DMA Controller (DMAC) × 8
- Power-on reset
- Low Voltage Detection (LVD) with voltage settings
- Watchdog Timer (WDT)
- Independent Watchdog Timer (IWDT)

## Human Machine Interface (HMI)

- Capacitive Touch Sensing Unit (CTSU)

## Multiple Clock Sources

- Main clock oscillator (MOSC) (8 to 24 MHz)
- Sub-clock oscillator (SOSC) (32.768 kHz)
- High-speed on-chip oscillator (HOCO) (16/18/20 MHz)
- Middle-speed on-chip oscillator (MOCO) (8 MHz)
- Low-speed on-chip oscillator (LOCO) (32.768 kHz)
- IWDT-dedicated on-chip oscillator (15 kHz)
- Clock trim function for HOCO/MOCO/LOCO
- PLL/PLL2
- Clock out support

## General-Purpose I/O Ports

- 5-V tolerance, open drain, input pull-up, switchable driving ability

## Operating Voltage

- VCC: 2.7 to 3.6 V

## Operating Temperature and Packages

- Ta = -40°C to +105°C
  - 100-pin LQFP (14 mm × 14 mm, 0.5 mm pitch)
  - 64-pin LQFP (10 mm × 10 mm, 0.5 mm pitch)
  - 48-pin LQFP (7 mm × 7 mm, 0.5 mm pitch)
  - 48-pin QFN (7 mm × 7 mm, 0.5 mm pitch)

# 1. Overview

The MCU integrates multiple series of software- and pin-compatible Arm®-based 32-bit cores that share a common set of Renesas peripherals to facilitate design scalability and efficient platform-based product development.

The MCU in this series incorporates a high-performance Arm Cortex®-M33 core running up to 100 MHz with the following features:

- Up to 512 KB code flash memory
- 128 KB SRAM
- Quad Serial Peripheral Interface (QSPI)
- USBFS, SD/MMC Host Interface
- Capacitive Touch Sensing Unit (CTSU)
- Analog peripherals
- Security and safety features

## 1.1 Function Outline

**Table 1.1 Arm core**

| Feature | Functional description |
|---|---|
| Arm Cortex-M33 core | • Maximum operating frequency: up to 100 MHz<br>• Arm Cortex-M33 core:<br>– Armv8-M architecture with security extension<br>– Revision: r0p4-00rel0<br>• Arm Memory Protection Unit (Arm MPU)<br>– Protected Memory System Architecture (PMSAv8)<br>– Secure MPU (MPU_S): 8 regions<br>– Non-secure MPU (MPU_NS): 8 regions<br>• SysTick timer<br>– Embeds two Systick timers: Secure and Non-secure instance<br>– Driven by SysTick timer clock (SYSTICCLK) or system clock (ICLK)<br>• CoreSight™ ETM-M33 |

**Table 1.2 Memory**

| Feature | Functional description |
|---|---|
| Code flash memory | Maximum 512 KB of code flash memory.<br>See section 44, Flash Memory. |
| Data flash memory | 8 KB of data flash memory.<br>See section 44, Flash Memory. |
| Option-setting memory | The option-setting memory determines the state of the MCU after a reset.<br>See section 6, Option-Setting Memory. |
| SRAM | On-chip high-speed SRAM with either parity bit or Error Correction Code (ECC).<br>See section 42, SRAM. |

**Table 1.3 System (1 of 2)**

| Feature | Functional description |
|---|---|
| Operating modes | Two operating modes:<br>• Single-chip mode<br>• SCI/USB boot mode<br>See section 3, Operating Modes. |
| Resets | The MCU provides 13 resets.<br>See section 5, Resets. |

**Table 1.3 System (2 of 2)**

| Feature | Functional description |
|---|---|
| Low Voltage Detection (LVD) | The Low Voltage Detection (LVD) module monitors the voltage level input to the VCC pin. The detection level can be selected by register settings. The LVD module consists of three separate voltage level detectors (LVD0, LVD1, LVD2). LVD0, LVD1, and LVD2 measure the voltage level input to the VCC pin. LVD registers allow your application to configure detection of VCC changes at various voltage thresholds.<br>See section 7, Low Voltage Detection (LVD). |
| Clocks | • Main clock oscillator (MOSC)<br>• Sub-clock oscillator (SOSC)<br>• High-speed on-chip oscillator (HOCO)<br>• Middle-speed on-chip oscillator (MOCO)<br>• Low-speed on-chip oscillator (LOCO)<br>• IWDT-dedicated on-chip oscillator<br>• PLL/PLL2<br>• Clock out support<br>See section 8, Clock Generation Circuit. |
| Clock Frequency Accuracy Measurement Circuit (CAC) | The Clock Frequency Accuracy Measurement Circuit (CAC) counts pulses of the clock to be measured (measurement target clock) within the time generated by the clock selected as the measurement reference (measurement reference clock), and determines the accuracy depending on whether the number of pulses is within the allowable range. When measurement is complete or the number of pulses within the time generated by the measurement reference clock is not within the allowable range, an interrupt request is generated.<br>See section 9, Clock Frequency Accuracy Measurement Circuit (CAC). |
| Interrupt Controller Unit (ICU) | The Interrupt Controller Unit (ICU) controls which event signals are linked to the Nested Vector Interrupt Controller (NVIC), the DMA Controller (DMAC), and the Data Transfer Controller (DTC) modules. The ICU also controls non-maskable interrupts.<br>See section 13, Interrupt Controller Unit (ICU). |
| Low power modes | Power consumption can be reduced in multiple ways, including setting clock dividers, stopping modules, selecting power control mode in normal operation, and transitioning to low power modes.<br>See section 10, Low Power Modes. |
| Battery backup function | A battery backup function is provided for partial powering by a battery. The battery-powered area includes the RTC, SOSC, backup memory, and switch between VCC and VBATT.<br>See section 11, Battery Backup Function. |
| Register write protection | The register write protection function protects important registers from being overwritten due to software errors. The registers to be protected are set with the Protect Register (PRCR).<br>See section 12, Register Write Protection. |
| Memory Protection Unit (MPU) | The MCU has one Memory Protection Unit (MPU).<br>See section 15, Memory Protection Unit (MPU). |

**Table 1.4 Event link**

| Feature | Functional description |
|---|---|
| Event Link Controller (ELC) | The Event Link Controller (ELC) uses the event requests generated by various peripheral modules as source signals to connect them to different modules, allowing direct link between the modules without CPU intervention.<br>See section 18, Event Link Controller (ELC). |

**Table 1.5 Direct memory access**

| Feature | Functional description |
|---|---|
| Data Transfer Controller (DTC) | A Data Transfer Controller (DTC) module is provided for transferring data when activated by an interrupt request.<br>See section 17, Data Transfer Controller (DTC). |
| DMA Controller (DMAC) | The MCU includes an 8-channel direct memory access controller (DMAC) that can transfer data without intervention from the CPU. When a DMA transfer request is generated, the DMAC transfers data stored at the transfer source address to the transfer destination address.<br>See section 16, DMA Controller (DMAC). |

**Table 1.6 External bus interface**

| Feature | Functional description |
|---|---|
| External bus | • QSPI area (EQBIU): Connected to the QSPI (external device interface) |

**Table 1.7 Timers**

| Feature | Functional description |
|---|---|
| General PWM Timer (GPT) | The General PWM Timer (GPT) is a 32-bit timer with GPT32 × 4 channels and a 16-bit timer with GPT16 × 4 channels. PWM waveforms can be generated by controlling the up-counter, down-counter, or the up- and down-counter. In addition, PWM waveforms can be generated for controlling brushless DC motors. The GPT can also be used as a general-purpose timer.<br>See section 21, General PWM Timer (GPT). |
| Port Output Enable for GPT (POEG) | The Port Output Enable (POEG) function can place the General PWM Timer (GPT) output pins in the output disable state<br>See section 20, Port Output Enable for GPT (POEG). |
| Low Power Asynchronous General Purpose Timer (AGT) | The Low Power Asynchronous General Purpose Timer (AGT) is a 16-bit timer that can be used for pulse output, external pulse width or period measurement, and counting external events. This timer consists of a reload register and a down counter. The reload register and the down counter are allocated to the same address, and can be accessed with the AGT register.<br>See section 22, Low Power Asynchronous General Purpose Timer (AGT). |
| Realtime Clock (RTC) | The realtime clock (RTC) has two counting modes, calendar count mode and binary count mode, that are used by switching register settings. For calendar count mode, the RTC has a 100-year calendar from 2000 to 2099 and automatically adjusts dates for leap years. For binary count mode, the RTC counts seconds and retains the information as a serial value. Binary count mode can be used for calendars other than the Gregorian (Western) calendar.<br>See section 23, Realtime Clock (RTC). |
| Watchdog Timer (WDT) | The Watchdog Timer (WDT) is a 14-bit down counter that can be used to reset the MCU when the counter underflows because the system has run out of control and is unable to refresh the WDT. In addition, the WDT can be used to generate a non-maskable interrupt or an underflow interrupt.<br>See section 24, Watchdog Timer (WDT). |
| Independent Watchdog Timer (IWDT) | The Independent Watchdog Timer (IWDT) consists of a 14-bit down counter that must be serviced periodically to prevent counter underflow. The IWDT provides functionality to reset the MCU or to generate a non-maskable interrupt or an underflow interrupt. Because the timer operates with an independent, dedicated clock source, it is particularly useful in returning the MCU to a known state as a fail-safe mechanism when the system runs out of control. The IWDT can be triggered automatically by a reset, underflow, refresh error, or a refresh of the count value in the registers.<br>See section 25, Independent Watchdog Timer (IWDT). |

**Table 1.8 Communication interfaces (1 of 2)**

| Feature | Functional description |
|---|---|
| Serial Communications Interface (SCI) | The Serial Communications Interface (SCI) × 6 channels have asynchronous and synchronous serial interfaces:<br>• Asynchronous interfaces (UART and Asynchronous Communications Interface Adapter (ACIA))<br>• 8-bit clock synchronous interface<br>• Simple IIC (master-only)<br>• Simple SPI<br>• Smart card interface<br>• Manchester interface<br>• Extended Serial interface<br>The smart card interface complies with the ISO/IEC 7816-3 standard for electronic signals and transmission protocol. SCIn (n = 0, 3, 4, 9) has FIFO buffers to enable continuous and full-duplex communication, and the data transfer speed can be configured independently using an on-chip baud rate generator.<br>See section 27, Serial Communications Interface (SCI). |
| $I^2C$ bus interface (IIC) | The $I^2C$ bus interface (IIC) has 2 channels. The IIC module conforms with and provides a subset of the NXP $I^2C$ (Inter-Integrated Circuit) bus interface functions.<br>See section 28, I2C Bus Interface (IIC). |

**Table 1.8 Communication interfaces (2 of 2)**

| Feature | Functional description |
|---|---|
| Serial Peripheral Interface (SPI) | The Serial Peripheral Interface (SPI) has 1 channel. The SPI provides high-speed full-duplex synchronous serial communications with multiple processors and peripheral devices.<br>See section 30, Serial Peripheral Interface (SPI). |
| Control Area Network (CAN) | The Controller Area Network (CAN) module uses a message-based protocol to receive and transmit data between multiple slaves and masters in electromagnetically noisy applications. The module complies with the ISO 11898-1 (CAN 2.0A/CAN 2.0B) standard and supports up to 32 mailboxes, which can be configured for transmission or reception in normal mailbox and FIFO modes. Both standard (11-bit) and extended (29-bit) messaging formats are supported. The CAN module requires an additional external CAN transceiver.<br>See section 29, Controller Area Network (CAN) Module. |
| USB 2.0 Full-Speed module (USBFS) | The USB 2.0 Full-Speed module (USBFS) can operate as a host controller or device controller. The module supports full-speed and low-speed (host controller only) transfer as defined in Universal Serial Bus Specification 2.0. The module has an internal USB transceiver and supports all of the transfer types defined in Universal Serial Bus Specification 2.0. The USB has buffer memory for data transfer, providing a maximum of 10 pipes. Pipes 1 to 9 can be assigned any endpoint number based on the peripheral devices used for communication or based on your system.<br>See section 26, USB 2.0 Full-Speed Module (USBFS). |
| Quad Serial Peripheral Interface (QSPI) | The Quad Serial Peripheral Interface (QSPI) is a memory controller for connecting a serial ROM (nonvolatile memory such as a serial flash memory, serial EEPROM, or serial FeRAM) that has an SPI-compatible interface.<br>See section 31, Quad Serial Peripheral Interface (QSPI). |
| Serial Sound Interface Enhanced (SSIE) | The Serial Sound Interface Enhanced (SSIE) peripheral provides functionality to interface with digital audio devices for transmitting $I^2S$/Monaural/TDM audio data over a serial bus. The SSIE supports an audio clock frequency of up to 50 MHz, and can be operated as a slave or master receiver, transmitter, or transceiver to suit various applications. The SSIE includes 32-stage FIFO buffers in the receiver and transmitter, and supports interrupts and DMA-driven data reception and transmission.<br>See section 32, Serial Sound Interface Enhanced (SSIE). |
| SD/MMC Host Interface (SDHI) | The SDHI and MultiMediaCard (MMC) interface module provides the functionality required to connect a variety of external memory cards to the MCU. The SDHI supports both 1- and 4-bit buses for connecting memory cards that support SD, SDHC, and SDXC formats. When developing host devices that are compliant with the SD Specifications, you must comply with the SD Host/Ancillary Product License Agreement (SD HALA). The MMC interface supports 1-bit, and 4-bit MMC buses that provide eMMC 4.51 (JEDEC Standard JESD 84-B451) device access. This interface also provides backward compatibility and supports high-speed SDR transfer modes.<br>See section 33, SD/MMC Host Interface (SDHI). |

**Table 1.9 Analog**

| Feature | Functional description |
|---|---|
| 12-bit A/D Converter (ADC12) | A 12-bit successive approximation A/D converter (ADC12) is provided. Analog input channels are selectable up to 22. The temperature sensor output and an internal reference voltage are selectable for conversion.<br>See section 37, 12-Bit A/D Converter (ADC12). |
| 12-bit D/A Converter (DAC12) | A 12-bit D/A converter (DAC12) is provided.<br>See section 38, 12-Bit D/A Converter (DAC12). |
| Temperature Sensor (TSN) | The on-chip Temperature Sensor (TSN) determines and monitors the die temperature for reliable operation of the device. The sensor outputs a voltage directly proportional to the die temperature, and the relationship between the die temperature and the output voltage is fairly linear. The output voltage is provided to the ADC12 for conversion and can be further used by the end application.<br>See section 39, Temperature Sensor (TSN). |

**Table 1.10 Human machine interfaces**

| Feature | Functional description |
|---|---|
| Capacitive Touch Sensing Unit (CTSU) | The Capacitive Touch Sensing Unit (CTSU) measures the electrostatic capacitance of the touch sensor. Changes in the electrostatic capacitance are determined by software that enables the CTSU to detect whether a finger is in contact with the touch sensor. The electrode surface of the touch sensor is usually enclosed with an electrical conductor so that a finger does not come into direct contact with the electrode.<br>See section 40, Capacitive Touch Sensing Unit (CTSU). |

**Table 1.11 Data processing**

| Feature | Functional description |
|---|---|
| Cyclic Redundancy Check (CRC) calculator | The Cyclic Redundancy Check (CRC) generates CRC codes to detect errors in the data. The bit order of CRC calculation results can be switched for LSB-first or MSB-first communication. Additionally, various CRC-generation polynomials are available.<br>See section 34, Cyclic Redundancy Check (CRC). |
| Data Operation Circuit (DOC) | The Data Operation Circuit (DOC) compares, adds, and subtracts 16-bit data. When a selected condition applies, 16-bit data is compared and an interrupt can be generated.<br>See section 41, Data Operation Circuit (DOC). |

**Table 1.12 Security**

| Feature | Functional description |
|---|---|
| Securty function | • ARMv8-M TrustZone security<br>• Device lifecycle management<br>• Debug access level<br>• Key injection<br>• Secure pin multiplexing |
| Secure Crypto Engine 9 (SCE9) | • Symmetric algorithms: AES<br>• Asymmetric algorithms: RSA, ECC, and DSA<br>• Hash-value generation: SHA224, SHA256, GHASH<br>• 128-bit unique ID.<br>See section 36, Secure Cryptographic Engine (SCE9). |

**Table 1.13 I/O ports**

| Feature | Functional description |
|---|---|
| Programmable I/O ports | • I/O ports for the 100-pin LQFP<br>– I/O pins: 77<br>– Input pins: 1<br>– Pull-up resistors: 78<br>– N-ch open-drain outputs: 77<br>– 5-V tolerance: 14<br>• I/O ports for the 64-pin LQFP<br>– I/O pins: 43<br>– Input pins: 1<br>– Pull-up resistors: 44<br>– N-ch open-drain outputs: 43<br>– 5-V tolerance: 9<br>• I/O ports for the 48-pin LQFP, 48-pin QFN<br>– I/O pins: 29<br>– Input pins: 1<br>– Pull-up resistors: 30<br>– N-ch open-drain outputs: 29<br>– 5-V tolerance: 4 |

## 1.2 Block Diagram

Figure 1.1 shows a block diagram of the MCU superset. Some individual devices within the group have a subset of the features.

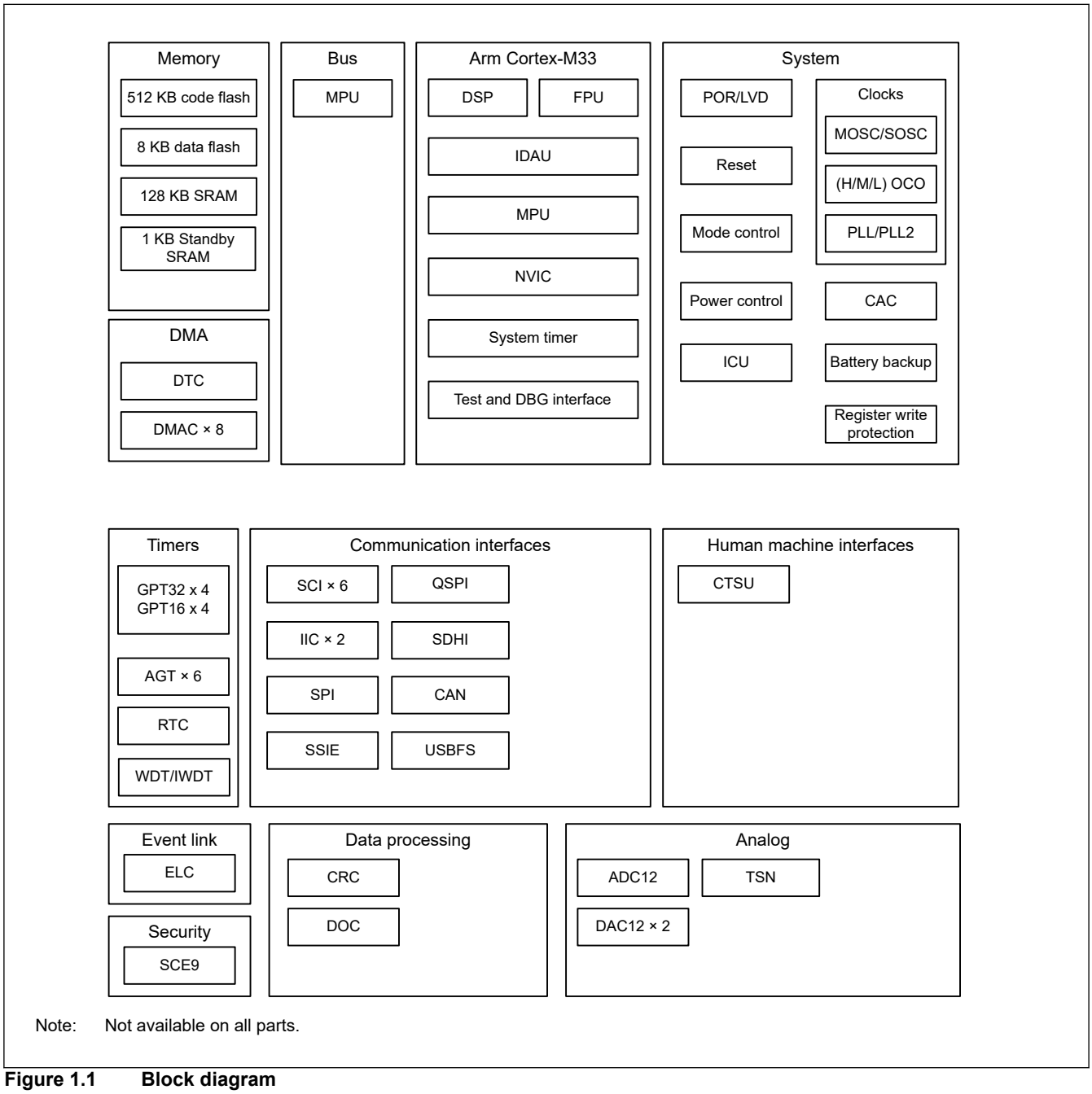
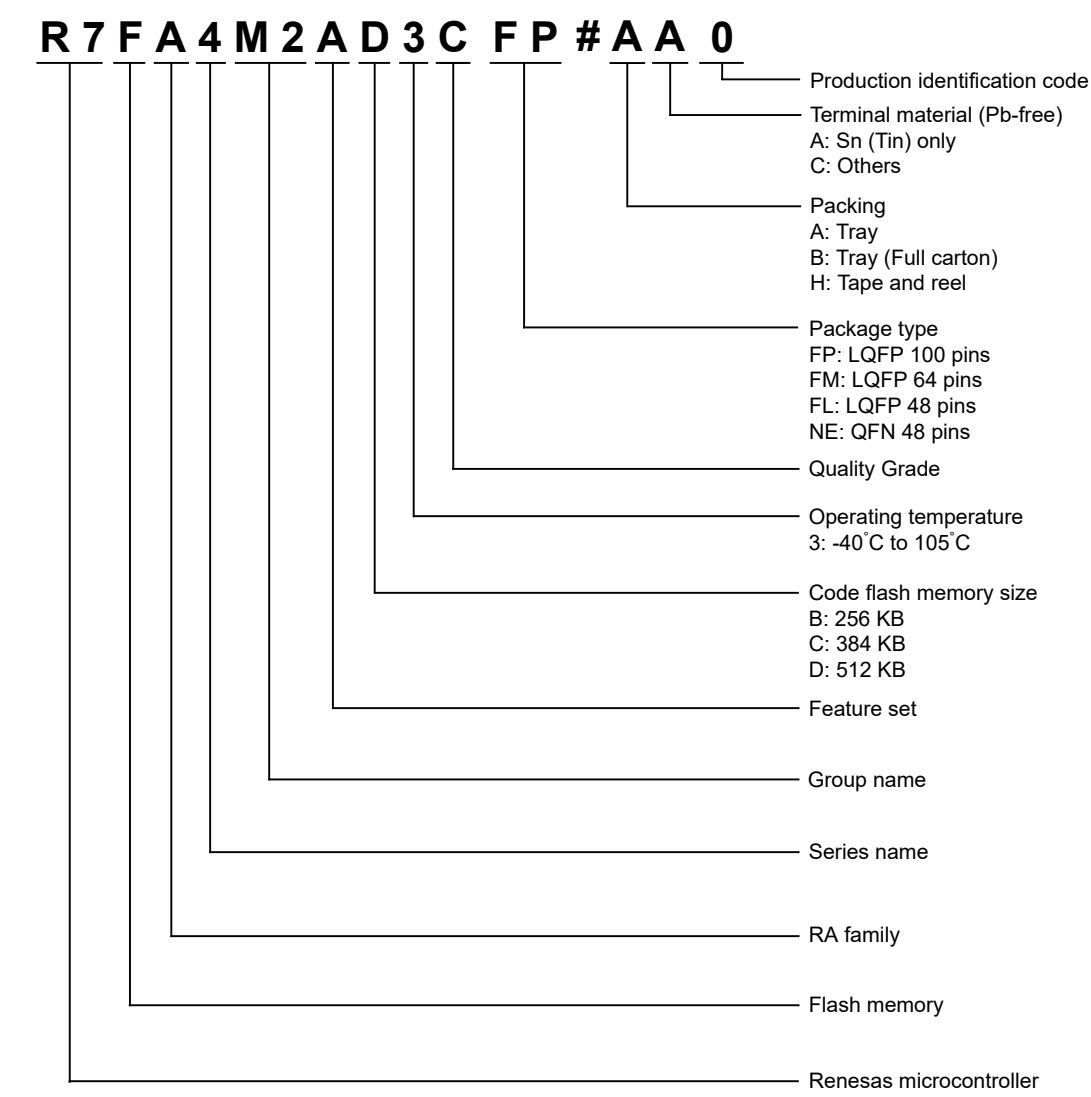



**Figure 1.1 Block diagram**

## 1.3 Part Numbering

Figure 1.2 shows the product part number information, including memory capacity and package type. Table 1.14 shows a list of products.




Note: Check the order screen for each product on the Renesas website for valid symbols after the #.

**Figure 1.2 Part numbering scheme**

**Table 1.14 Product list (1 of 2)**

| Product part number | Package code | Code flash | Data flash | SRAM | Operating temperature |
|---|---|---|---|---|---|
| R7FA4M2AD3CFP | PLQP0100KB-B | 512 KB | 8 KB | 128 KB | -40 to +105°C |
| R7FA4M2AD3CFM | PLQP0064KB-C | | | | |
| R7FA4M2AD3CFL | PLQP0048KB-B | | | | |
| R7FA4M2AD3CNE | PWQN0048KC-A | | | | |
| R7FA4M2AC3CFP | PLQP0100KB-B | 384 KB | 8 KB | 128 KB | -40 to +105°C |
| R7FA4M2AC3CFM | PLQP0064KB-C | | | | |
| R7FA4M2AC3CFL | PLQP0048KB-B | | | | |
| R7FA4M2AC3CNE | PWQN0048KC-A | | | | |

**Table 1.14 Product list (2 of 2)**

| Product part number | Package code | Code flash | Data flash | SRAM | Operating temperature |
|---|---|---|---|---|---|
| R7FA4M2AB3CFP | PLQP0100KB-B | 256 KB | 8 KB | 128 KB | -40 to +105°C |
| R7FA4M2AB3CFM | PLQP0064KB-C | | | | |
| R7FA4M2AB3CFL | PLQP0048KB-B | | | | |
| R7FA4M2AB3CNE | PWQN0048KC-A | | | | |

## 1.4 Function Comparison

**Table 1.15 Function Comparison (1 of 2)**

| Parts number | | R7FA4M2AD3CFP<br>R7FA4M2AC3CFP<br>R7FA4M2AB3CFP | R7FA4M2AD3CFM<br>R7FA4M2AC3CFM<br>R7FA4M2AB3CFM | R7FA4M2AD3CFL<br>R7FA4M2AC3CFL<br>R7FA4M2AB3CFL | R7FA4M2AD3CNE<br>R7FA4M2AC3CNE<br>R7FA4M2AB3CNE |
|---|---|---|---|---|---|
| Pin count | | 100 | 64 | 48 | 48 |
| Package | | LQFP | | | QFN |
| Code flash memory | | 512KB<br>384KB<br>256KB | | | |
| Data flash memory | | 8 KB | | | |
| SRAM | | 128 KB | | | |
| | Parity | 64 KB | | | |
| | ECC | 64 KB | | | |
| Standby SRAM | | 1 KB | | | |
| DMA | DTC | Yes | | | |
| | DMAC | 8 | | | |
| System | CPU clock | 100 MHz (max.) | | | |
| | CPU clock sources | MOSC, SOSC, HOCO, MOCO, LOCO, PLL | | | |
| | CAC | Yes | | | |
| | WDT/IWDT | Yes | | | |
| | Backup register | 128 B | | | |
| Communication | SCI | 6 | | | |
| | IIC | 2 | | 1 | |
| | SPI | 1 | | | |
| | CAN | 1 | | | |
| | USBFS | Yes | | | |
| | QSPI | Yes | | | |
| | SSIE | Yes | No | | |
| | SDHI/MMC | Yes | No | | |
| Timers | GPT32[*1] | 4 | | | |
| | GPT16[*1] | 4 | | | |
| | AGT[*1] | 6 | | | |
| | RTC | Yes | | | |
| Analog | ADC12 | Unit 0: 13 | Unit 0: 9 | Unit 0: 7 | |
| | DAC12 | 2 | | | |
| | TSN | Yes | | | |
| HMI | CTSU | 12 | 7 | 4 | |
| Data processing | CRC | Yes | | | |
| | DOC | Yes | | | |
| Event control | ELC | Yes | | | |
| Security | | SCE9, TrustZone, and Lifecycle management | | | |

**Table 1.15 Function Comparison (2 of 2)**

| Parts number | | R7FA4M2AD3CFP<br>R7FA4M2AC3CFP<br>R7FA4M2AB3CFP | R7FA4M2AD3CFM<br>R7FA4M2AC3CFM<br>R7FA4M2AB3CFM | R7FA4M2AD3CFL<br>R7FA4M2AC3CFL<br>R7FA4M2AB3CFL | R7FA4M2AD3CNE<br>R7FA4M2AC3CNE<br>R7FA4M2AB3CNE |
|---|---|---|---|---|---|
| I/O ports | I/O pins | 77 | 43 | 29 | |
| | Input pins | 1 | 1 | 1 | |
| | Pull-up resistors | 78 | 44 | 30 | |
| | N-ch open-drain outputs | 77 | 43 | 29 | |
| | 5-V tolerance | 14 | 9 | 4 | |

Note 1. Available pins depend on the Pin count, about details see section 1.7. Pin Lists.

## 1.5 Pin Functions

**Table 1.16 Pin functions (1 of 4)**

| Function | Signal | I/O | Description |
|---|---|---|---|
| Power supply | VCC | Input | Power supply pin. Connect it to the system power supply. Connect this pin to VSS by a 0.1-μF capacitor. The capacitor should be placed close to the pin. |
| | VCL | I/O | Connect this pin to the VSS pin by the smoothing capacitor used to stabilize the internal power supply. Place the capacitor close to the pin. |
| | VBATT | Input | Battery Backup power pin |
| | VSS | Input | Ground pin. Connect it to the system power supply (0 V). |
| Clock | XTAL | Output | Pins for a crystal resonator. An external clock signal can be input through the EXTAL pin. |
| | EXTAL | Input | |
| | XCIN | Input | Input/output pins for the sub-clock oscillator. Connect a crystal resonator between XCOUT and XCIN. |
| | XCOUT | Output | |
| | CLKOUT | Output | Clock output pin |
| Operating mode control | MD | Input | Pin for setting the operating mode. The signal level on this pin must not be changed during operation mode transition on release from the reset state. |
| System control | RES | Input | Reset signal input pin. The MCU enters the reset state when this signal goes low. |
| CAC | CACREF | Input | Measurement reference clock input pin |
| On-chip emulator | TMS | Input | On-chip emulator or boundary scan pins |
| | TDI | Input | |
| | TCK | Input | |
| | TDO | Output | |
| | TCLK | Output | Output clock for synchronization with the trace data |
| | TDATA0 to TDATA3 | Output | Trace data output |
| | SWO | Output | Serial wire trace output pin |
| | SWDIO | I/O | Serial wire debug data input/output pin |
| | SWCLK | Input | Serial wire clock pin |
| Interrupt | NMI | Input | Non-maskable interrupt request pin |
| | IRQn | Input | Maskable interrupt request pins |
| | IRQn-DS | Input | Maskable interrupt request pins that can also be used in Deep Software Standby mode |

**Table 1.16 Pin functions (2 of 4)**

| Function | Signal | I/O | Description |
|---|---|---|---|
| GPT | GTETRGA, GTETRGB, GTETRGC, GTETRGD | Input | External trigger input pins |
| | GTIOCnA, GTIOCnB | I/O | Input capture, output compare, or PWM output pins |
| | GTIU | Input | Hall sensor input pin U |
| | GTIV | Input | Hall sensor input pin V |
| | GTIW | Input | Hall sensor input pin W |
| | GTOUUP | Output | 3-phase PWM output for BLDC motor control (positive U phase) |
| | GTOULO | Output | 3-phase PWM output for BLDC motor control (negative U phase) |
| | GTOVUP | Output | 3-phase PWM output for BLDC motor control (positive V phase) |
| | GTOVLO | Output | 3-phase PWM output for BLDC motor control (negative V phase) |
| | GTOWUP | Output | 3-phase PWM output for BLDC motor control (positive W phase) |
| | GTOWLO | Output | 3-phase PWM output for BLDC motor control (negative W phase) |
| AGT | AGTEEn | Input | External event input enable signals |
| | AGTIOn | I/O | External event input and pulse output pins |
| | AGTOn | Output | Pulse output pins |
| | AGTOAn | Output | Output compare match A output pins |
| | AGTOBn | Output | Output compare match B output pins |
| RTC | RTCOUT | Output | Output pin for 1-Hz or 64-Hz clock |
| | RTCICn | Input | Time capture event input pins |
| SCI | SCKn | I/O | Input/output pins for the clock (clock synchronous mode) |
| | RXDn | Input | Input pins for received data (asynchronous mode/clock synchronous mode) |
| | TXDn | Output | Output pins for transmitted data (asynchronous mode/clock synchronous mode) |
| | CTSn_RTSn | I/O | Input/output pins for controlling the start of transmission and reception (asynchronous mode/clock synchronous mode), active-low. |
| | CTSn | Input | Input for the start of transmission. |
| | SCLn | I/O | Input/output pins for the IIC clock (simple IIC mode) |
| | SDAn | I/O | Input/output pins for the IIC data (simple IIC mode) |
| | SCKn | I/O | Input/output pins for the clock (simple SPI mode) |
| | MISOn | I/O | Input/output pins for slave transmission of data (simple SPI mode) |
| | MOSIn | I/O | Input/output pins for master transmission of data (simple SPI mode) |
| | RXDXn | Input | Input pins for received data (Extended Serial Mode) |
| | TXDXn | Output | Output pins for transmitted data (Extended Serial Mode) |
| | SIOXn | I/O | Input/output pins for receivde or tramsmitted data (Extended Serial Mode) |
| | SSn | Input | Chip-select input pins (simple SPI mode), active-low |
| IIC | SCLn | I/O | Input/output pins for the clock |
| | SDAn | I/O | Input/output pins for data |

**Table 1.16 Pin functions (3 of 4)**

| Function | Signal | I/O | Description |
|---|---|---|---|
| SPI | RSPCKA | I/O | Clock input/output pin |
| | MOSIA | I/O | Input or output pins for data output from the master |
| | MISOA | I/O | Input or output pins for data output from the slave |
| | SSLA0 | I/O | Input or output pin for slave selection |
| | SSLA1 to SSLA3 | Output | Output pins for slave selection |
| CAN | CRXn | Input | Receive data |
| | CTXn | Output | Transmit data |
| USBFS | VCC_USB | Input | Power supply pin |
| | VSS_USB | Input | Ground pin |
| | USB_DP | I/O | D+ pin of the USB on-chip transceiver. Connect this pin to the D+ pin of the USB bus. |
| | USB_DM | I/O | D- pin of the USB on-chip transceiver. Connect this pin to the D- pin of the USB bus. |
| | USB_VBUS | Input | USB cable connection monitor pin. Connect this pin to VBUS of the USB bus. The VBUS pin status (connected or disconnected) can be detected when the USB module is operating as a function controller. |
| | USB_EXICEN | Output | Low-power control signal for external power supply (OTG) chip |
| | USB_VBUSEN | Output | VBUS (5 V) supply enable signal for external power supply chip |
| | USB_OVRCURA, USB_OVRCURB | Input | Connect the external overcurrent detection signals to these pins. Connect the VBUS comparator signals to these pins when the OTG power supply chip is connected. |
| | USB_OVRCURA-DS | Input | Overcurrent pins for USBFS that can also be used in Deep Software Standby mode. Connect the external overcurrent detection signals to these pins. Connect the VBUS comparator signals to these pins when the OTG power supply chip is connected. |
| | USB_ID | Input | Connect the MicroAB connector ID input signal to this pin during operation in OTG mode |
| QSPI | QSPCLK | Output | QSPI clock output pin |
| | QSSL | Output | QSPI slave output pin |
| | QIO0 to QIO3 | I/O | Data0 to Data3 |
| SSIE | SSIBCK0 | I/O | SSIE serial bit clock pins |
| | SSILRCK0/SSIFS0 | I/O | LR clock/frame synchronization pins |
| | SSITXD0 | Output | Serial data output pin |
| | SSIRXD0 | Input | Serial data input pin |
| | AUDIO_CLK | Input | External clock pin for audio (input oversampling clock) |
| SDHI/MMC | SD0CLK | Output | SD clock output pins |
| | SD0CMD | I/O | Command output pin and response input signal pins |
| | SD0DAT0 to SD0DAT3 | I/O | SD and MMC data bus pins |
| | SD0CD | Input | SD card detection pins |
| | SD0WP | Input | SD write-protect signals |

**Table 1.16 Pin functions (4 of 4)**

| Function | Signal | I/O | Description |
|---|---|---|---|
| Analog power supply | AVCC0 | Input | Analog voltage supply pin. This is used as the analog power supply for the respective modules. Supply this pin with the same voltage as the VCC pin. |
| | AVSS0 | Input | Analog ground pin. This is used as the analog ground for the respective modules. Supply this pin with the same voltage as the VSS pin. |
| | VREFH | Input | Analog reference voltage supply pin for the D/A Converter. Connect this pin to AVCC0 when not using the D/A Converter. |
| | VREFL | Input | Analog reference ground pin for the D/A Converter. Connect this pin to AVSS0 when not using the D/A Converter. |
| | VREFH0 | Input | Analog reference voltage supply pin for the ADC12. Connect this pin to AVCC0 when not using the ADC12. |
| | VREFL0 | Input | Analog reference ground pin for the ADC12. Connect this pin to AVSS0 when not using the ADC12. |
| ADC12 | ANmn | Input | Input pins for the analog signals to be processed by the A/D converter.<br>(m: ADC unit number, n: pin number) |
| | ADTRGm | Input | Input pins for the external trigger signals that start the A/D conversion, active-low. |
| DAC12 | DAn | Output | Output pins for the analog signals processed by the D/A converter. |
| CTSU | TSn | Input | Capacitive touch detection pins (touch pins) |
| | TSCAP | I/O | Secondary power supply pin for the touch driver |
| I/O ports | Pmn | I/O | General-purpose input/output pins<br>(m: port number, n: pin number) |
| | P200 | Input | General-purpose input pin |

## 1.6 Pin Assignments

The following figures show the pin assignments from the top view.

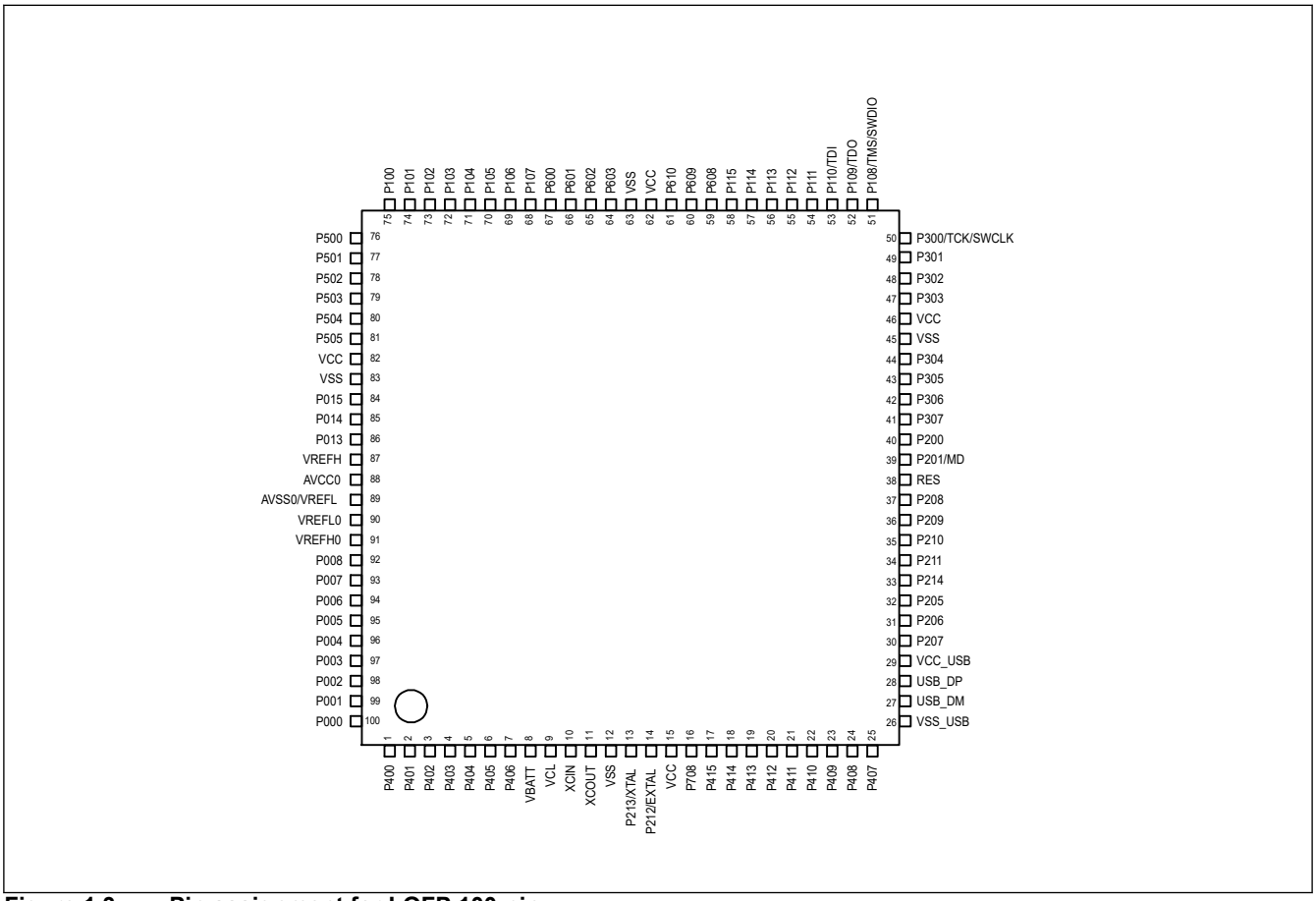



**Figure 1.3 Pin assignment for LQFP 100-pin**

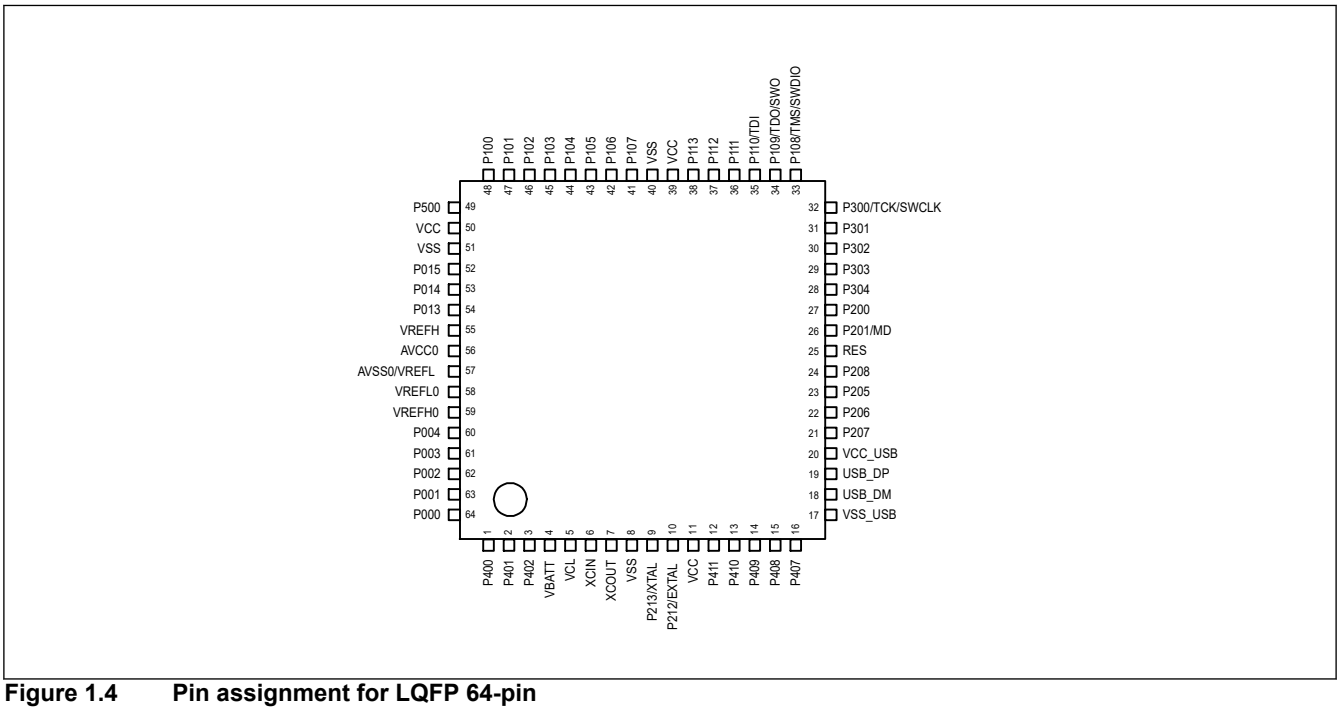



**Figure 1.4 Pin assignment for LQFP 64-pin**

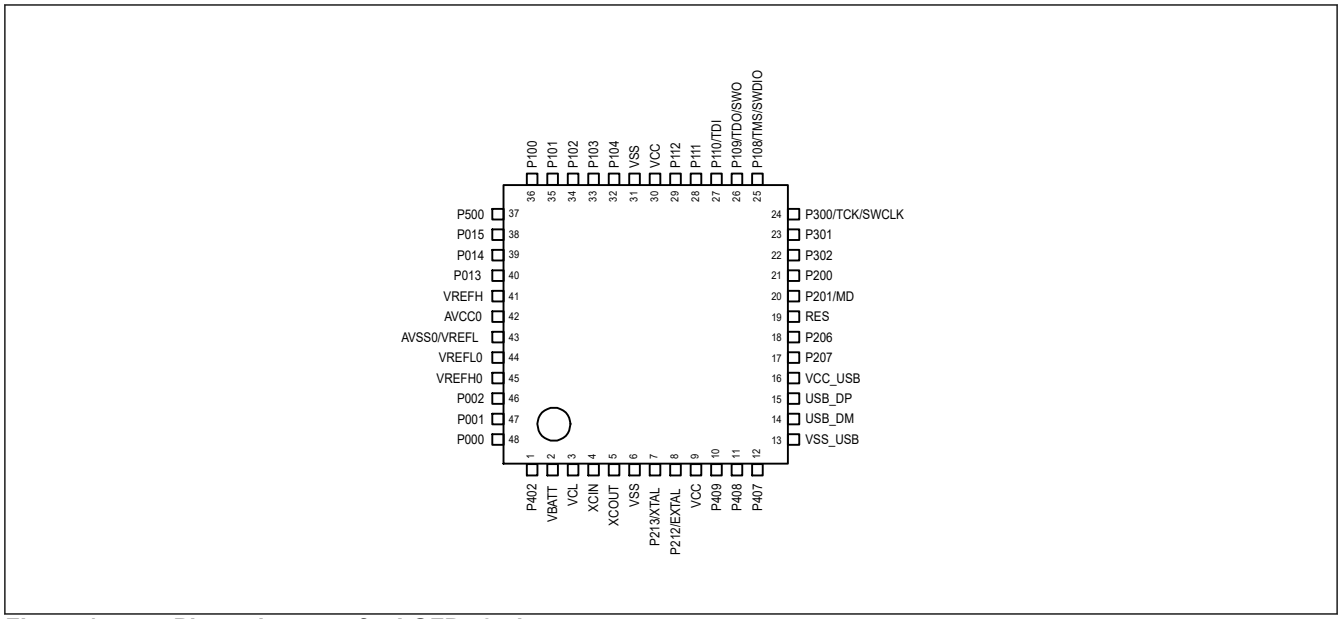



**Figure 1.5 Pin assignment for LQFP 48-pin**

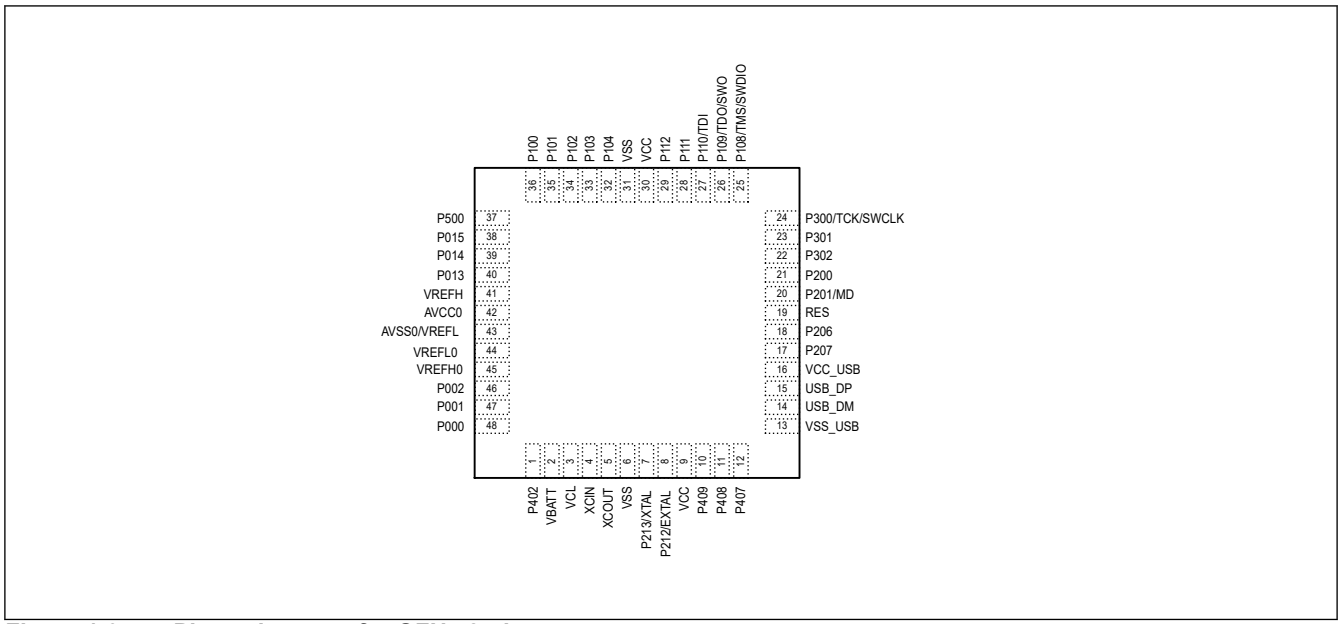



**Figure 1.6 Pin assignment for QFN 48-pin**

## 1.7 Pin Lists

**Table 1.17 Pin list (1 of 3)**

| LQFP100 | LQFP64 | LQFP48, QFN48 | Power, System, Clock, Debug, CAC | I/O ports | Ex. Interrupt | SCI/IIC/SPI/CAN/USBFS/QSPI/SSIE/SDHI/MMC | GPT/AGT/RTC | ADC12/DAC12 | CTSU |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | — | — | P400 | IRQ0 | SCK4/SCL0_A/AUDIO_CLK | GTIOC6A/AGTIO1 | — | — |
| 2 | 2 | — | — | P401 | IRQ5-DS | CTS4_RTS4/SS4/SDA0_A/CTX0 | GTETRGA/GTIOC6B | — | — |
| 3 | 3 | 1 | CACREF | P402 | IRQ4-DS | CTS4/CRX0/AUDIO_CLK | AGTIO0/AGTIO1/AGTIO2/AGTIO3/ RTCIC0 | — | — |
| 4 | — | — | — | P403 | IRQ14-DS | SSIBCK0_A | GTIOC3A/AGTIO0/AGTIO1/AGTIO2/ AGTIO3/RTCIC1 | — | — |
| 5 | — | — | — | P404 | IRQ15-DS | SSILRCK0/SSIFS0_A | GTIOC3B/AGTIO0/AGTIO1/AGTIO2/ AGTIO3/RTCIC2 | — | — |
| 6 | — | — | — | P405 | — | SSITXD0_A | GTIOC1A | — | — |
| 7 | — | — | — | P406 | — | SSIRXD0_A | GTIOC1B/AGTO5 | — | — |
| 8 | 4 | 2 | VBATT | — | — | — | — | — | — |
| 9 | 5 | 3 | VCL | — | — | — | — | — | — |
| 10 | 6 | 4 | XCIN | — | — | — | — | — | — |
| 11 | 7 | 5 | XCOUT | — | — | — | — | — | — |
| 12 | 8 | 6 | VSS | — | — | — | — | — | — |
| 13 | 9 | 7 | XTAL | P213 | IRQ2 | TXD1/MOSI1/SDA1/TXDX1/SIOX1 | GTETRGC/GTIOC0A/AGTEE2 | — | — |
| 14 | 10 | 8 | EXTAL | P212 | IRQ3 | RXD1/MISO1/SCL1/RXDX1 | GTETRGD/GTIOC0B/AGTEE1 | — | — |
| 15 | 11 | 9 | VCC | — | — | — | — | — | — |
| 16 | — | — | CACREF | P708 | IRQ11 | RXD1/MISO1/SCL1/RXDX1/AUDIO_CLK | — | — | TS12 |
| 17 | — | — | — | P415 | IRQ8 | USB_VBUSEN/SD0CD | GTIOC0A/AGTIO4 | — | TS11 |
| 18 | — | — | — | P414 | IRQ9 | CTS0/SD0WP | GTIOC0B/AGTIO5 | — | TS10 |
| 19 | — | — | — | P413 | — | CTS0_RTS0/SS0/SD0CLK_A | GTOUUP/AGTEE3 | — | TS09 |
| 20 | — | — | — | P412 | — | SCK0/CTS3/SD0CMD_A | GTOULO/AGTEE1 | — | TS08 |
| 21 | 12 | — | — | P411 | IRQ4 | TXD0/MOSI0/SDA0/CTS3_RTS3/SS3/SD0DAT0_A | GTOVUP/AGTOA1 | — | TS07 |
| 22 | 13 | — | — | P410 | IRQ5 | RXD0/MISO0/SCL0/SCK3/SD0DAT1_A | GTOVLO/AGTOB1 | — | TS06 |
| 23 | 14 | 10 | — | P409 | IRQ6 | TXD3/MOSI3/SDA3/USB_EXICEN | GTOWUP/AGTOA2 | — | TS05 |
| 24 | 15 | 11 | — | P408 | IRQ7 | CTS4/RXD3/MISO3/SCL3/SCL0_B/USB_ID | GTOWLO/GTIOC6B/AGTOB2 | — | TS04 |
| 25 | 16 | 12 | — | P407 | — | CTS4_RTS4/SS4/SDA0_B/USB_VBUS | GTIOC6A/AGTIO0/RTCOUT | ADTRG0 | TS03 |
| 26 | 17 | 13 | VSS_USB | — | — | — | — | — | — |
| 27 | 18 | 14 | USB_DM | — | — | — | — | — | — |
| 28 | 19 | 15 | USB_DP | — | — | — | — | — | — |
| 29 | 20 | 16 | VCC_USB | — | — | — | — | — | — |
| 30 | 21 | 17 | — | P207 | — | TXD4/MOSI4/SDA4/QSSL | — | — | TSCAP |
| 31 | 22 | 18 | — | P206 | IRQ0-DS | RXD4/MISO4/SCL4/CTS9/SDA1_B/USB_VBUSEN/ SD0DAT2_A | GTIU | — | TS02 |
| 32 | 23 | — | CLKOUT | P205 | IRQ1-DS | TXD4/MOSI4/SDA4/CTS9_RTS9/SS9/SCL1_B/ USB_OVRCURA-DS/SD0DAT3_A | GTIV/GTIOC4A/AGTO1 | — | TS01 |
| 33 | — | — | TCLK | P214 | — | QSPCLK/SD0CLK_B | GTIU/AGTO5 | — | — |
| 34 | — | — | TDATA0 | P211 | — | QIO0/SD0CMD_B | GTIV/AGTOA5 | — | — |
| 35 | — | — | TDATA1 | P210 | — | QIO1/SD0CD | GTIW/AGTOB5 | — | — |
| 36 | — | — | TDATA2 | P209 | — | QIO2/SD0WP | GTOVUP/AGTEE5 | — | — |
| 37 | 24 | — | TDATA3 | P208 | — | QIO3/SD0DAT0_B | GTOVLO | — | — |
| 38 | 25 | 19 | RES | — | — | — | — | — | — |
| 39 | 26 | 20 | MD | P201 | — | — | — | — | — |
| 40 | 27 | 21 | — | P200 | NMI | — | — | — | — |
| 41 | — | — | — | P307 | — | QIO0 | GTOUUP/AGTEE4 | — | — |
| 42 | — | — | — | P306 | — | QSSL | GTOULO/AGTOA2 | — | — |
| 43 | — | — | — | P305 | IRQ8 | QSPCLK | GTOWUP/AGTOB2 | — | — |
| 44 | 28 | — | — | P304 | IRQ9 | — | GTOWLO/GTIOC7A/AGTEE2 | — | — |
| 45 | — | — | VSS | — | — | — | — | — | — |

**Table 1.17 Pin list (2 of 3)**

| LQFP100 | LQFP64 | LQFP48, QFN48 | Power, System, Clock, Debug, CAC | I/O ports | Ex. Interrupt | SCI/IIC/SPI/CAN/USBFS/QSPI/SSIE/SDHI/MMC | GPT/AGT/RTC | ADC12/DAC12 | CTSU |
|---|---|---|---|---|---|---|---|---|---|
| 46 | — | — | VCC | — | — | — | — | — | — |
| 47 | 29 | — | — | P303 | — | CTS9 | GTIOC7B | — | — |
| 48 | 30 | 22 | — | P302 | IRQ5 | TXD2/MOSI2/SDA2/TXDX2/SIOX2/SSLA3 | GTOUUP/GTIOC4A | — | — |
| 49 | 31 | 23 | — | P301 | IRQ6 | RXD2/MISO2/SCL2/RXDX2/CTS9_RTS9/SS9/SSLA2 | GTOULO/GTIOC4B/AGTIO0 | — | — |
| 50 | 32 | 24 | TCK/SWCLK | P300 | — | SSLA1 | GTOUUP/GTIOC0A | — | — |
| 51 | 33 | 25 | TMS/SWDIO | P108 | — | CTS9_RTS9/SS9/SSLA0 | GTOULO/GTIOC0B/AGTOA3 | — | — |
| 52 | 34 | 26 | TDO/SWO/CLKOUT | P109 | — | TXD9/MOSI9/SDA9/MOSIA | GTOVUP/GTIOC1A/AGTOB3 | — | — |
| 53 | 35 | 27 | TDI | P110 | IRQ3 | CTS2_RTS2/SS2/RXD9/MISO9/SCL9/MISOA | GTOVLO/GTIOC1B/AGTEE3 | — | — |
| 54 | 36 | 28 | — | P111 | IRQ4 | SCK2/SCK9/RSPCKA | GTIOC3A/AGTOA5 | — | — |
| 55 | 37 | 29 | — | P112 | — | TXD2/MOSI2/SDA2/TXDX2/SIOX2/SCK1/SSLA0/QSSL/ SSIBCK0_B | GTIOC3B/AGTOB5 | — | — |
| 56 | 38 | — | — | P113 | — | RXD2/MISO2/SCL2/RXDX2/SSILRCK0/SSIFS0_B | GTIOC2A/AGTEE5 | — | — |
| 57 | — | — | — | P114 | — | CTS9/SSIRXD0_B | GTIOC2B/AGTIO5 | — | — |
| 58 | — | — | — | P115 | — | SSITXD0_B | GTIOC4A | — | — |
| 59 | — | — | — | P608 | — | — | GTIOC4B | — | — |
| 60 | — | — | — | P609 | — | — | GTIOC5A/AGTO5 | — | — |
| 61 | — | — | — | P610 | — | — | GTIOC5B/AGTO4 | — | — |
| 62 | 39 | 30 | VCC | — | — | — | — | — | — |
| 63 | 40 | 31 | VSS | — | — | — | — | — | — |
| 64 | — | — | — | P603 | — | CTS9_RTS9/SS9 | GTIOC7A/AGTIO4 | — | — |
| 65 | — | — | — | P602 | — | TXD9/MOSI9/SDA9 | GTIOC7B/AGTO3 | — | — |
| 66 | — | — | — | P601 | — | RXD9/MISO9/SCL9 | GTIOC6A/AGTEE3 | — | — |
| 67 | — | — | CACREF/CLKOUT | P600 | — | SCK9 | GTIOC6B/AGTIO3 | — | — |
| 68 | 41 | — | — | P107 | — | — | AGTOA0 | — | — |
| 69 | 42 | — | — | P106 | — | — | AGTOB0 | — | — |
| 70 | 43 | — | — | P105 | IRQ0 | — | GTETRGA/GTIOC1A/AGTO2 | — | — |
| 71 | 44 | 32 | — | P104 | IRQ1 | QIO2 | GTETRGB/GTIOC1B/AGTEE2 | — | — |
| 72 | 45 | 33 | — | P103 | — | CTS0_RTS0/SS0/CTX0/QIO3 | GTOWUP/GTIOC2A/AGTIO2 | — | — |
| 73 | 46 | 34 | — | P102 | — | SCK0/CRX0/QIO0 | GTOWLO/GTIOC2B/AGTO0 | ADTRG0 | — |
| 74 | 47 | 35 | — | P101 | IRQ1 | TXD0/MOSI0/SDA0/CTS1_RTS1/SS1/QIO1 | GTETRGB/GTIOC5A/AGTEE0 | — | — |
| 75 | 48 | 36 | — | P100 | IRQ2 | RXD0/MISO0/SCL0/SCK1/QSPCLK | GTETRGA/GTIOC5B/AGTIO0 | — | — |
| 76 | 49 | 37 | CACREF | P500 | — | USB_VBUSEN/QSPCLK | GTIU/AGTOA0 | AN016 | — |
| 77 | — | — | — | P501 | IRQ11 | USB_OVRCURA/QSSL | GTIV/AGTOB0 | — | — |
| 78 | — | — | — | P502 | IRQ12 | USB_OVRCURB/QIO0 | GTIW/AGTOA2 | — | — |
| 79 | — | — | — | P503 | — | USB_EXICEN/QIO1 | GTETRGC/AGTOB2 | — | — |
| 80 | — | — | — | P504 | — | USB_ID/QIO2 | GTETRGD/AGTOA3 | — | — |
| 81 | — | — | — | P505 | IRQ14 | QIO3 | AGTOB3 | — | — |
| 82 | 50 | — | VCC | — | — | — | — | — | — |
| 83 | 51 | — | VSS | — | — | — | — | — | — |
| 84 | 52 | 38 | — | P015 | IRQ13 | — | — | AN013/DA1 | — |
| 85 | 53 | 39 | — | P014 | — | — | — | AN012/DA0 | — |
| 86 | 54 | 40 | — | P013 | — | — | — | AN011 | — |
| 87 | 55 | 41 | VREFH | — | — | — | — | — | — |
| 88 | 56 | 42 | AVCC0 | — | — | — | — | — | — |
| 89 | 57 | 43 | AVSS0/VREFL | — | — | — | — | — | — |
| 90 | 58 | 44 | VREFL0 | — | — | — | — | — | — |
| 91 | 59 | 45 | VREFH0 | — | — | — | — | — | — |
| 92 | — | — | — | P008 | IRQ12-DS | — | — | AN008 | — |
| 93 | — | — | — | P007 | — | — | — | AN007 | — |
| 94 | — | — | — | P006 | IRQ11-DS | — | — | AN006 | — |

**Table 1.17 Pin list (3 of 3)**

| LQFP100 | LQFP64 | LQFP48, QFN48 | Power, System, Clock, Debug, CAC | I/O ports | Ex. Interrupt | SCI/IIC/SPI/CAN/USBFS/QSPI/SSIE/SDHI/MMC | GPT/AGT/RTC | ADC12/DAC12 | CTSU |
|---|---|---|---|---|---|---|---|---|---|
| 95 | — | — | — | P005 | IRQ10-DS | — | — | AN005 | — |
| 96 | 60 | — | — | P004 | IRQ9-DS | — | — | AN004 | — |
| 97 | 61 | — | — | P003 | — | — | — | AN003 | — |
| 98 | 62 | 46 | — | P002 | IRQ8-DS | — | — | AN002 | — |
| 99 | 63 | 47 | — | P001 | IRQ7-DS | — | — | AN001 | — |
| 100 | 64 | 48 | — | P000 | IRQ6-DS | — | — | AN000 | — |

Note: Several pin names have the added suffix of _A, _B, and _C. The suffix can be ignored when assigning functionality.

# 2. CPU

The MCU is based on the Arm® Cortex®-M33 core.

## 2.1 Overview

### 2.1.1 CPU

- Arm Cortex-M33
  - Revision: r0p4-00rel1
  - Armv8-M architecture profile
  - Single Precision Floating-Point Unit compliant with the ANSI/IEEE Std 754-2008
- SAU (Security Attribution Unit): 0 region
- IDAU (Implementation Defined Attribution Unit): 8 regions
  - Code flash (secure/non-secure callable/non-secure)
  - Data Flash (secure/non-secure)
  - SRAM0 (secure/non-secure callable/non-secure)
- Memory Protection Unit (MPU)
  - Armv8 Protected Memory System Architecture (PMSAv8)
  - Secure MPU (MPU_S): 8 regions
  - Non-secure MPU (MPU_NS): 8 regions
- SysTick timer
  - Two Systick timers: Secure and Non-secure instance
  - Driven by SysTick timer clock (SYSTICCLK) or system clock (ICLK)

See reference 1. and reference 2. in section 2.14. References for details.

### 2.1.2 Debug

- Arm® CoreSight™ ETM-M33
  - Revision: r0p2-00rel0
  - ARM ETM Architecture version 4.2
- Instrumentation Trace Macrocell (ITM)
- Data Watchpoint and Trace Unit (DWT)
  - 4 comparators for watchpoints and triggers
- Breakpoint Unit (BPU)
  - Breakpoint function is available.
    - 8 instruction comparators
    - 0 literal comparators
- Time Stamp Generator (TSG)
  - Time stamp for ETM and ITM
  - Driven by CPU clock
- Debug Register Module (DBGREG)
  - Reset control
  - Halt control
- Debug Access Port (DAP)

  - JTAG Debug Port (JTAG-DP)
  - Serial Wire Debug Port (SW-DP)
- Cortex-M33 Trace Port Interface Unit (TPIU)
  - 4 bits TPIU formatter output
  - Serial Wire Output
- Cross Trigger Interface (CTI)
- Embedded Trace Buffer (ETB)
  - CoreSight Trace Memory Controller with ETB configuration
  - Buffer size: 2 KB

See reference 1. and reference 2. in section 2.14. References for details.

### 2.1.3 Operating Frequency

The operating frequencies for the MCU are as follows:

- CPU: maximum 100 MHz
- 4-bit TPIU trace interface: maximum 50 MHz
- Serial Write Output (SWO) trace interface: maximum 50 MHz
- Joint Test Action Group (JTAG) interface: maximum 25 MHz
- Serial Wire Debug (SWD) interface: maximum 25 MHz

### 2.1.4 Block Diagram

Figure 2.1 shows a block diagram of the Cortex-M33 core.

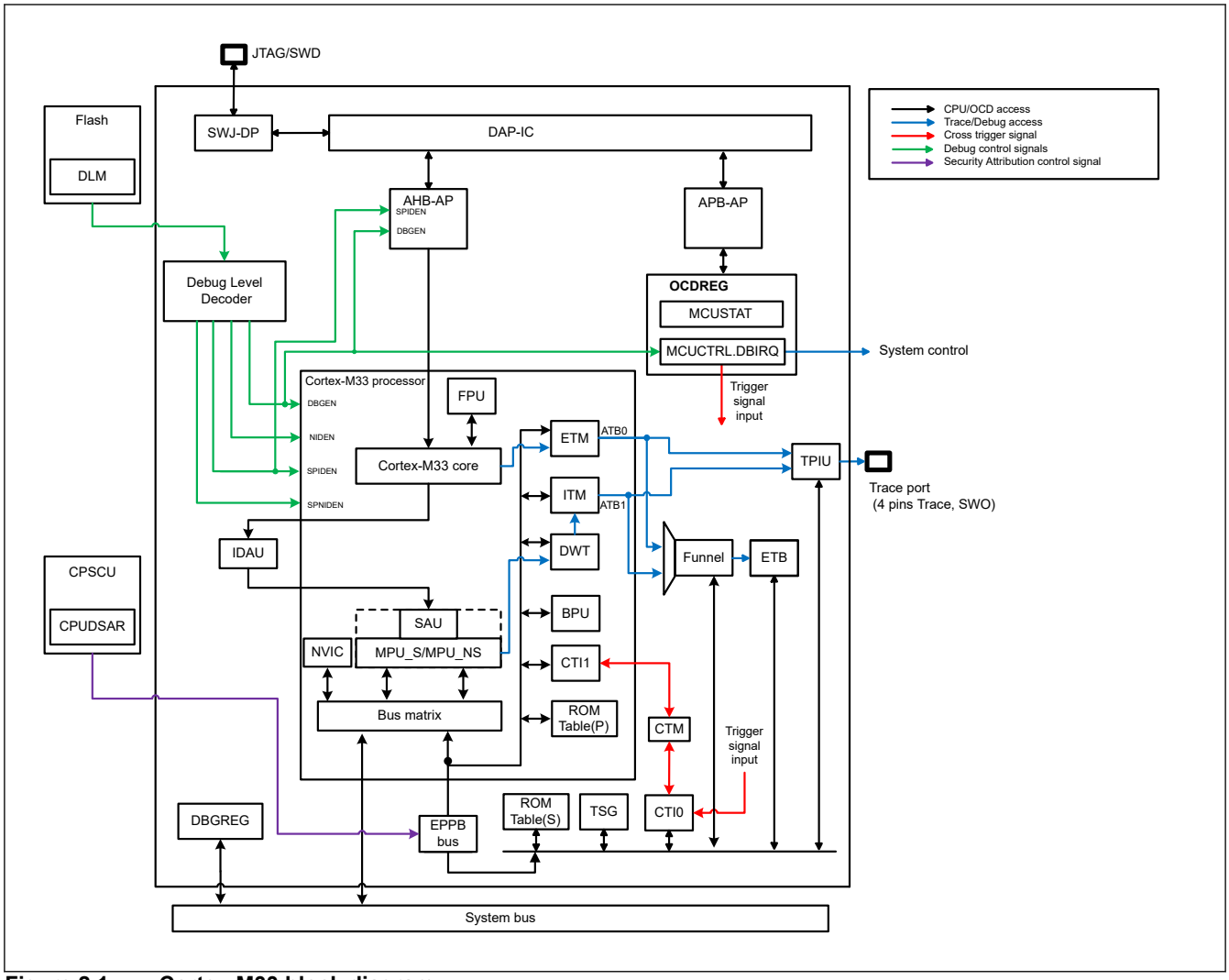



**Figure 2.1 Cortex-M33 block diagram**

## 2.2 Implementation Options

Table 2.1 shows the implementation options of the MCU.

**Table 2.1 Implementation options (1 of 2)**

| Option | Implementation |
|---|---|
| SAU | Not included |
| IDAU | Included, 8 regions |
| MPU | Included, 8 regions for Secure and 8 regions for Non-secure |
| BPU | Included |
| Cross Trigger Interface (CTI) | Included |
| DWT | Included |
| Number of Wakeup Interrupt Controllers (WIC) | Not included<br>ICU can wake up CPU instead of WIC. See section 13, Interrupt Controller Unit (ICU) for details. |
| TPIU | Included<br>• 4 bits TPIU formatter output<br>• Serial Wire Output |
| FPU | Included |
| DSP | Included |

**Table 2.1 Implementation options (2 of 2)**

| Option | Implementation |
|---|---|
| Embedded Trace Macrocell (ETM) | Included |
| Sleep mode power saving | Sleep mode and other low power modes are supported. For more details, see section 10, Low Power Modes.<br>Note: SCB.SCR.SLEEPDEEP is ignored. |
| Interrupts | 98 |
| Priority bits | 4 bits (16 levels) |
| Endianness | Little-endian |
| SysTick | Included |
| SYST_CALIB register (0x4000_0147) | Bit [31] = 0 Reference clock provided<br>Bit [30] = 1 TERMS value is inexact<br>Bits [29:24] = 0x00 Reserved<br>Bits [23:0] = 0x000147 TERM: (32768 × 10 ms) - 1/32.768 kHz<br>= 326.66 decimal<br>= 327 with skew<br>= 0x000147 |
| Event input/output | Not implemented |
| Global exclusive monitor | Not implemented |
| System reset request output | The SYSRESETREQ bit in Application Interrupt and Reset Control Register causes a CPU reset |

## 2.3 Trace Interface

A Trace Port Interface Unit (TPIU) and Serial Wire Output (SWO) provide trace output. Table 2.2 shows the MCU pins for the function. These pins are multiplexed with other functions.

**Table 2.2 Trace function pins**

| Name | I/O | Function | When not in use |
|---|---|---|---|
| TCLK | Output | Trace clock | Open |
| TDATA0 | Output | Trace data output 0 | Open |
| TDATA1 | Output | Trace data output 1 | Open |
| TDATA2 | Output | Trace data output 2 | Open |
| TDATA3 | Output | Trace data output 3 | Open |
| TDO/SWO | Output | Serial wire output multiplexed with JTAG TDO pin | Open |

## 2.4 JTAG/SWD Interface

Table 2.3 shows the JTAG/SWD pins.

**Table 2.3 JTAG/SWD pins**

| Name | I/O | Function | When not in use |
|---|---|---|---|
| TDI | Input | JTAG TDI pin | Pull-up |
| TDO/SWO | Output | JTAG TDO pin multiplexed with serial wire output | Open |
| TCK/SWCLK | Input | JTAG clock pin<br>Serial wire clock pin | Pull-up |
| TMS/SWDIO | I/O | JTAG TMS pin<br>Serial wire data I/O pin | Pull-up |

## 2.5 Security Attribution for Memory

In this MCU, SAU is not implemented and IDAU performs region definition for memory. IDAU divides the memory into 8 different areas as shown in Figure 2.2.

The code flash, the data flash, and the SRAM are divided into Secure (S), Non-secure (NS) and Non-secure callable (NSC) regions. These memory security attributions are set into the nonvolatile memory by the serial programming command when the device lifecycle is in SSD state. These memory security attributions are loaded into the IDAU and the memory controller before application execution. These memory security attributions cannot be updated by application but can read through the dedicated registers.

Note: When configuring, the memory regions should satisfy the setting condition of minimum address unit shown in Table 2.4.

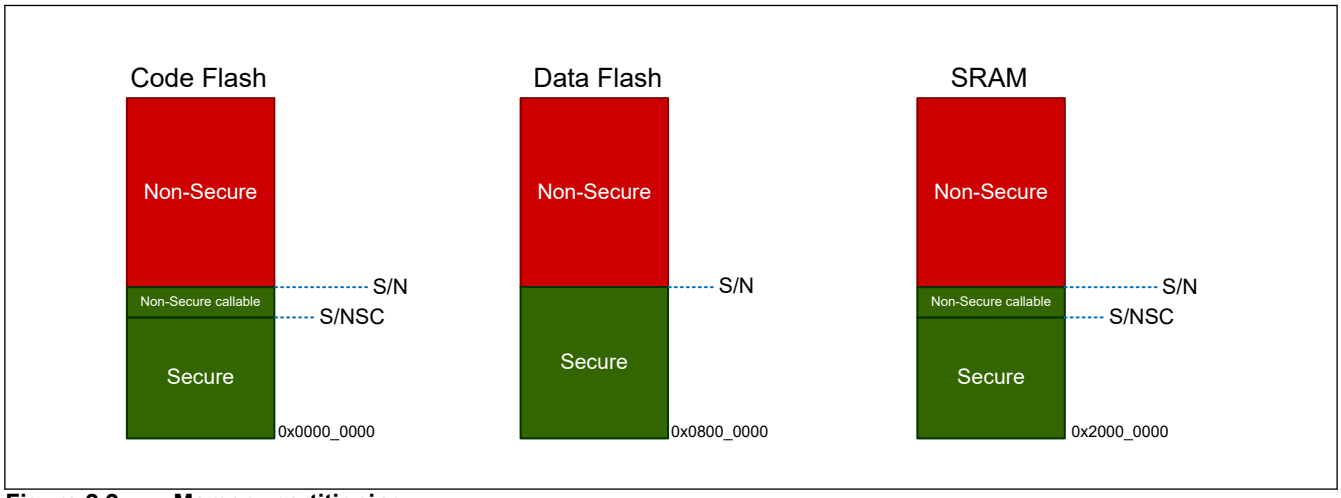



**Figure 2.2 Memory partitioning**

**Table 2.4 S/NS and S/NSC boundary list**

| Boundary | Code flash | Data flash | SRAM |
|---|---|---|---|
| S/NS | 32 KB | 1 KB | 8 KB |
| S/NSC | 1 KB | — | 1 KB |

Each region has its dedicated ID as follows. For more details, see section 2.14. References.

| IREGION (IDAU region number) | Description |
|---|---|
| 0x0D | Non-secure SRAM |
| 0x0E | Non-secure callable SRAM |
| 0x0F | Secure SRAM |
| 0x09 | Non-secure data flash |
| 0x0B | Secure data flash |
| 0x05 | Non-secure code flash |
| 0x06 | Non-secure callable code flash |
| 0x07 | Secure code flash |

## 2.6 Debug Function

### 2.6.1 Debugger connectivity

In this MCU, debug function is considered in three levels, DBG0, DBG1, DBG2. At DBG0, no debug function is available. DBG1 level is defined as non-secure debug in ARMv-8 and the debugger can only access defined non-secure debug accessible regions. DBG2 level is defined as secure debug in ARMv-8 and at this level, nonsecure and secure debug function is enabled and can be accessible from the debugger.

Debug level is determined by the Device Lifecycle Management (DLM) state of the product.

See Figure 2.1 for debugger accessible regions.

Table 2.5 shows the CPU debug function and conditions.

**Table 2.5 CPU debug function and conditions**

| Condition | | | Permitted debug function |
|---|---|---|---|
| OCD connect*1 | DLM State | Debug level | Description |
| Connected | CM | DBG2 | All debug functions are available |
| Connected | SSD | DBG2 | All debug functions are available |
| Connected | NSECSD | DBG1 | Only Non-secure debug function is available |
| Connected | DPL | DBG0 | Debugger connection is not available |
| Connected | LCK_DBG | DBG0 | Debugger connection is not available |
| Connected | LCK_BOOT | DBG0 | Debugger connection is not available |
| Connected | RMA_REQ | DBG0 | Debugger connection is not available |
| Connected | RMA_ACK | DBG2 | All debug functions are available |

Note 1. OCD connect is determined by the CDBGPWRUPREQ bit output in the SWJ-DP register. The bit can only be written by the OCD. However, the level of the bit can be confirmed by reading the DBGSTR.CDBGPWRUPREQ bit.

## 2.6.2 Emulator Connection

Renesas provides the emulator which supports both debugging using SWD or JTAG communication and serial programming using SCI communication. This emulator makes it easy to switch between debug-ging and serial programming.

Table 2.6 shows the pinout of 10 pin or 20 pin socket pinouts when using this emulator. The pinout of SWD and JTAG is ARM standard, and MD, TXD, RXD pins are added for the serial programming using SCI communication.

The serial programming interface must be used to program the TrustZone IDAU boundary register settings.

It is recommended to connect P300/SWCLK/TCK and P201/MD pins using wired OR circuit on the board to use both debugging and serial programming.

**Table 2.6 Pin assign for emulator**

| Pin No. | SWD | JTAG | Serial Programming using SCI |
|---|---|---|---|
| 1 | VCC | VCC | VCC |
| 2 | P108/SWDIO | P108/TMS | NC |
| 4 | P300/SWCLK<br>Wired OR with P201/MD | P300/TCK<br>Wired OR with P201/MD | P201/MD |
| 6 | P109/SWO/TXD9 | P109/SWO/TXD9 | P109/TXD9 |
| 8 | P110/RXD9 | P110/TDI/RXD9 | P110/RXD9 |
| 9 | GNDdetect | GNDdetect | GNDdetect |
| 10 | nRESET | nRESET | nRESET |
| 12 | P214/TCLK | P214/TCLK | NC |
| 14 | P211/TDATA[0] | P211/TDATA[0] | NC |
| 16 | P210/TDATA[1] | P210/TDATA[1] | NC |
| 18 | P209/TDATA[2] | P209/TDATA[2] | NC |
| 20 | P208/TDATA[3] | P208/TDATA[3] | NC |
| 3, 5, 15, 17, 19 | GND | GND | GND |
| 7 | NC | NC | NC |
| 11, 13 | NC | NC | NC |

## 2.6.3 Self-Hosted Debug Function

As described in section 2.7.6. CPUDSAR : CPU Debug Security Attribution Register, at the initial setting access from the CPU in non-secure state to CoreSight debug components is protected, that is, the non-secure access to CoreSight debug

components from the self-hosted debugger is not allowed when the debug level is DBG2 at the initial setting. Therfore, the CPUDSAR.CPUDSA0 must be set to 1 to enable the self-hosted debug for CPU in non-secure state.

Note: There is no restriction for the self-hosted debug function while the CPU is in the secure state.

### 2.6.4 Effect of Debug Function

The debug function effects inside and outside of CPU.

#### 2.6.4.1 Low power mode

All CoreSight debug components can store the register settings even when the CPU enters Software Standby, Snooze or Deep Software Standby mode. However, AHB-AP cannot respond to On-Chip Debug (OCD) access in these low power modes. The OCD must wait for cancellation of the low power mode to access the CoreSight debug components. To request low power mode cancellation, the OCD can set the DBIRQ bit in the MCUCTRL register. For details, see section 2.7.5.2. MCUCTRL : MCU Control Register.

#### 2.6.4.2 Reset

In OCD mode, some resets depend on the CPU status and the DBGSTOPCR register setting.

**Table 2.7 Reset or interrupt and mode setting**

| Reset or interrupt name | Control in On-Chip Debug (OCD) mode | |
|---|---|---|
| | OCD break mode | OCD run mode |
| RES pin reset | Same as user mode | |
| Power-on reset | Same as user mode | |
| Independent watchdog timer reset/interrupt | Does not occur*1 | Depends on DBGSTOPCR setting |
| Watchdog timer reset/interrupt | Does not occur*1 | Depends on DBGSTOPCR setting |
| Voltage monitor 0 reset | Depends on DBGSTOPCR setting | |
| Voltage monitor 1 reset/interrupt | Depends on DBGSTOPCR setting | |
| Voltage monitor 2 reset/interrupt | Depends on DBGSTOPCR setting | |
| SRAM parity error reset/interrupt | Depends on DBGSTOPCR setting | |
| SRAM ECC error reset/interrupt | Depends on DBGSTOPCR setting | |
| Bus master MPU error reset/interrupt | Same as user mode | |
| Deep software standby reset | Same as user mode | |
| Software reset | Same as user mode | |

Note: In OCD break mode, the CPU is halted. In OCD run mode, the CPU is in OCD mode and the CPU is not halted.
Note 1. The IWDT and WDT always stop in this mode.

## 2.7 Programmers Model

### 2.7.1 Address Spaces

The MCU debug system includes two CoreSight Access Ports (AP):

- AHB-AP, which is connected to the CPU bus matrix and has the same access to the system address space as the CPU
- APB-AP, which has a dedicated address space (OCD address space) and is connected to the OCDREG registers.

Figure 2.3 shows a block diagram of the AP connection and address spaces.

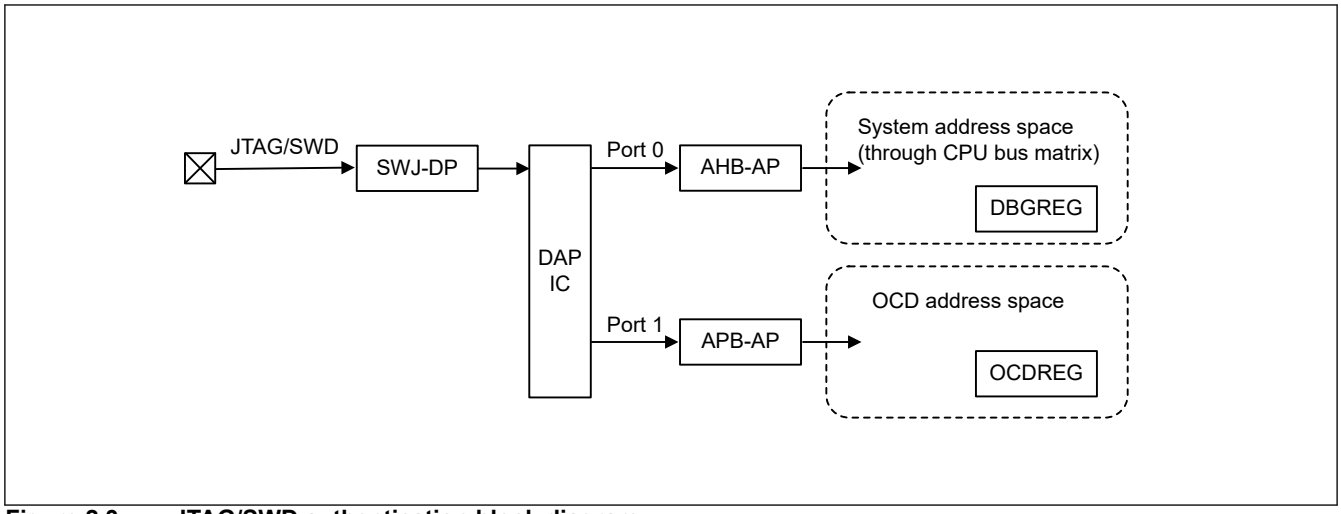



**Figure 2.3 JTAG/SWD authentication block diagram**

For debugging purposes, there are two register modules, DBGREG and OCDREG. DBGREG is located in the system address space and can be accessed from the OCD emulator, the CPU, and other bus masters in the MCU. OCDREG is located in the OCD address space and can only be accessed from the OCD tool. The CPU and other bus masters cannot access OCDREG.

## 2.7.2 Peripheral Address Map

In system address space, the Cortex-M33 core has a Private Peripheral Bus (PPB) which can be accessed only from CPU and OCD emulator. The PPB is expanded from the original implementation of the Cortex-M33 core for this MCU. Table 2.8 shows the address map of the MCU.

**Table 2.8 Peripheral address map**

| Component name | Start address | End address | Note |
|---|---|---|---|
| ITM | 0xE000_0000 | 0xE000_0FFF | See reference 2. in section 2.14. References |
| DWT | 0xE000_1000 | 0xE000_1FFF | See reference 2. in section 2.14. References |
| BPU | 0xE000_2000 | 0xE000_2FFF | See reference 2. in section 2.14. References |
| Secure SCS/SCS | 0xE000_E000 | 0xE000_EFFF | See reference 1. in section 2.14. References |
| Non-Secure SCS | 0xE002_E000 | 0xE002_EFFF | See reference 2. in section 2.14. References |
| TPIU | 0xE004_0000 | 0xE004_0FFF | See reference 3. in section 2.14. References |
| ETM | 0xE004_1000 | 0xE004_1FFF | See reference 1. in section 2.14. References |
| CTI1 | 0xE004_2000 | 0xE004_2FFF | See reference 2. in section 2.14. References |
| CTI0 | 0xE004_4000 | 0xE004_4FFF | See reference 4. in section 2.14. References |
| ATB Funnel | 0xE004_7000 | 0xE004_7FFF | See section 2.9. CoreSight ATB Funnel and reference 4. in section 2.14. References |
| ETB | 0xE004_8000 | 0xE004_8FFF | See reference 4. in section 2.14. References |
| Time Stamp Generator | 0xE004_9000 | 0xE004_9FFF | See section 2.11. CoreSight Time Stamp Generator and reference 4. in section 2.14. References |
| System ROM Table | 0xE00F_E000 | 0xE00F_EFFF | See reference 3. in section 2.14. References |
| Processor ROM Table | 0xE00F_F000 | 0xE00F_FFFF | See reference 2. in section 2.14. References |

## 2.7.3 CoreSight ROM Table

The MCU contains two CoreSight ROM Tables, the processor and system ROM Tables. The Processor ROM Table contains entries which hold a list of debug components inside the processor. The System ROM Table contains entries of Processor ROM Table and others debug components outside the processor.

### 2.7.3.1 ROM entries

ROM entries hold a list of components in the system. OCD emulator can use the ROM entries to determine which components are implemented in a system.

Table 2.9 and Table 2.10 show the System ROM entries and Processor ROM entries. See reference 5. in section 2.14. References for details.

**Table 2.9 System ROM entries**

| # | Address | Access size | R/W | Value | Target module pointer |
|---|---|---|---|---|---|
| 0 | 0xE00F_E000 | 32 bits | R | 0xFFF46003 | CTI0 |
| 1 | 0xE00F_E004 | 32 bits | R | 0xFFF49003 | Funnel |
| 2 | 0xE00F_E008 | 32 bits | R | 0xFFF4A003 | ETB |
| 3 | 0xE00F_E00C | 32 bits | R | 0xFFF4B003 | TSG |
| 4 | 0xE00F_E010 | 32 bits | R | 0xFFF42003 | TPIU |
| 5 | 0xE00F_E014 | 32 bits | R | 0x00001003 | Processor ROM table |
| 6 | 0xE00F_E018 | 32 bits | R | 0x00000000 | End of entries |

**Table 2.10 Processor ROM Entries**

| # | Address | Access size | R/W | Value | Target module pointer |
|---|---|---|---|---|---|
| 0 | 0xE00F_F000 | 32 bits | R | 0xFFF0F003 | SCS |
| 1 | 0xE00F_F004 | 32 bits | R | 0xFFF02003 | DWT |
| 2 | 0xE00F_F008 | 32 bits | R | 0xFFF03003 | BPU |
| 3 | 0xE00F_F00C | 32 bits | R | 0xFFF01003 | ITM |
| 4 | 0xE00F_F014 | 32 bits | R | 0xFFF42003 | ETM |
| 5 | 0xE00F_F018 | 32 bits | R | 0xFFF43003 | CTI1 |
| 6 | 0xE00F_F020 | 32 bits | R | 0x00000000 | End of entries |

### 2.7.3.2 CoreSight component registers

The CoreSight ROM Table lists the CoreSight component registers defined in the Arm CoreSight architecture.

Table 2.11 shows the registers. See reference 5. in section 2.14. References for details of each register.

**Table 2.11 CoreSight component registers in the CoreSight ROM Table**

| Name | Address | Access size | R/W | Initial value |
|---|---|---|---|---|
| PID4 | 0xE00F_EFD0 | 32 bits | R | 0x00000004 |
| PID5 | 0xE00F_EFD4 | 32 bits | R | 0x00000000 |
| PID6 | 0xE00F_EFD8 | 32 bits | R | 0x00000000 |
| PID7 | 0xE00F_EFDC | 32 bits | R | 0x00000000 |
| PID0 | 0xE00F_EFE0 | 32 bits | R | 0x00000034 |
| PID1 | 0xE00F_EFE4 | 32 bits | R | 0x00000030 |
| PID2 | 0xE00F_EFE8 | 32 bits | R | 0x0000000A |
| PID3 | 0xE00F_EFEC | 32 bits | R | 0x00000000 |
| CID0 | 0xE00F_EFF0 | 32 bits | R | 0x0000000D |
| CID1 | 0xE00F_EFF4 | 32 bits | R | 0x00000010 |
| CID2 | 0xE00F_EFF8 | 32 bits | R | 0x00000005 |
| CID3 | 0xE00F_EFFC | 32 bits | R | 0x000000B1 |

## 2.7.4 DBGREG Module

The DBGREG module controls the debug functionalities and is implemented as a CoreSight-compliant component.

Table 2.12 shows the DBGREG registers other than the CoreSight component registers.

**Table 2.12 Non-CoreSight DBGREG registers**

| Name | | DAP port | Address | Access size | R/W |
|---|---|---|---|---|---|
| Debug Status Register | DBGSTR | Port 0 | 0x4001_B000 | 32 bits | R |
| Debug Stop Control Register | DBGSTOPCR | Port 0 | 0x4001_B010 | 32 bits | R/W |

### 2.7.4.1 DBGSTR : Debug Status Register

Base address: DBG = 0x4001_B000

Offset address: 0x00

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | CDBGPWRUPACK | CDBGPWRUPREQ | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 27:0 | — | These bits are read as 0. | R |
| 28 | CDBGPWRUPREQ | Debug power-up request<br>0: OCD is not requesting debug power up<br>1: OCD is requesting debug power up | R |
| 29 | CDBGPWRUPACK | Debug power-up acknowledge<br>0: Debug power-up request is not acknowledged<br>1: Debug power-up request is acknowledged | R |
| 31:30 | — | These bits are read as 0. | R |

The DBGSTR register is a status register which indicates the state of the debug power-up request to the MCU from the emulator.

### 2.7.4.2 DBGSTOPCR : Debug Stop Control Register

Base address: DBG = 0x4001_B000

Offset address: 0x10

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | DBGSTOP_RECCR | DBGSTOP_RPER | — | — | — | — | — | DBGSTOP_LVD2 | DBGSTOP_LVD1 | DBGSTOP_LVD0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | DBGSTOP_WDT | DBGSTOP_IWDT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DBGSTOP_IWDT | Mask bit for IWDT reset/interrupt in the OCD run mode<br>In the OCD break mode, the reset/interrupt is masked and IWDT counter is stopped, regardless of this bit value.<br>0: Enable IWDT reset/interrupt<br>1: Mask IWDT reset/interrupt and stop IWDT counter | R/W |
| 1 | DBGSTOP_WDT | Mask bit for WDT reset/interrupt in the OCD run mode<br>In the OCD break mode, the reset/interrupt is masked and WDT counter is stopped, regardless of this bit value.<br>0: Enable WDT reset/interrupt<br>1: Mask WDT reset/interrupt and stop WDT counter | R/W |
| 15:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | DBGSTOP_LVD0 | Mask bit for LVD0 reset<br>0: Enable LVD0 reset<br>1: Mask LVD0 reset | R/W |
| 17 | DBGSTOP_LVD1 | Mask bit for LVD1 reset/interrupt<br>0: Enable LVD1 reset/interrupt<br>1: Mask LVD1 reset/interrupt | R/W |
| 18 | DBGSTOP_LVD2 | Mask bit for LVD2 reset/interrupt<br>0: Enable LVD2 reset/interrupt<br>1: Mask LVD2 reset/interrupt | R/W |
| 23:19 | — | These bits are read as 0. The write value should be 0. | R/W |
| 24 | DBGSTOP_RPER | Mask bit for SRAM parity error reset/interrupt<br>0: Enable SRAM parity error reset/interrupt<br>1: Mask SRAM parity error reset/interrupt | R/W |
| 25 | DBGSTOP_RECCR | Mask bit for SRAM ECC error reset/interrupt<br>0: Enable SRAM ECC error reset/interrupt<br>1: Mask SRAM ECC error reset/interrupt | R/W |
| 31:26 | — | These bits are read as 0. The write value should be 0. | R/W |

The Debug Stop Control Register (DBGSTOPCR) controls the functional stop in OCD mode. All bits in the register are regarded as 0 when the MCU is not in OCD mode.

### 2.7.4.3 DBGREG CoreSight component registers

The DBGREG module provides the CoreSight component registers defined in the Arm CoreSight architecture.

Table 2.13 shows the registers. See reference 4. in section 2.14. References for details of each register.

**Table 2.13 DBGREG CoreSight component registers**

| Name | Address | Access size | R/W | Initial value |
|---|---|---|---|---|
| PID4 | 0x4001_BFD0 | 32 bits | R | 0x00000004 |
| PID5 | 0x4001_BFD4 | 32 bits | R | 0x00000000 |
| PID6 | 0x4001_BFD8 | 32 bits | R | 0x00000000 |
| PID7 | 0x4001_BFDC | 32 bits | R | 0x00000000 |
| PID0 | 0x4001_BFE0 | 32 bits | R | 0x00000005 |
| PID1 | 0x4001_BFE4 | 32 bits | R | 0x00000030 |
| PID2 | 0x4001_BFE8 | 32 bits | R | 0x0000000A |
| PID3 | 0x4001_BFEC | 32 bits | R | 0x00000000 |
| CID0 | 0x4001_BFF0 | 32 bits | R | 0x0000000D |
| CID1 | 0x4001_BFF4 | 32 bits | R | 0x000000F0 |
| CID2 | 0x4001_BFF8 | 32 bits | R | 0x00000005 |
| CID3 | 0x4001_BFFC | 32 bits | R | 0x000000B1 |

## 2.7.5 OCDREG Module

The OCDREG module are only accessible by the On-Chip Debug (OCD) emulator. OCDREG is implemented as a CoreSight-compliant component.

Table 2.14 lists the OCDREG registers.

**Table 2.14 OCDREG registers**

| Name | | DAP port | Address | Access size | R/W |
|---|---|---|---|---|---|
| MCU Status Register | MCUSTAT | Port 1 | 0x8000_0400 | 32 bits | R |
| MCU Control Register | MCUCTRL | Port 1 | 0x8000_0410 | 32 bits | R/W |

Note: OCDREG is located in the dedicated OCD address space. This address map is independent from the system address map.

### 2.7.5.1 MCUSTAT : MCU Status Register

Base address: CPU_OCD = 0x8000_0000

Offset address: 0x400

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | SECDBG | DBGFUNCEN | BOOTMD | — | — | — | — | — | — | — | — | CPUSTOPCLK | CPUSLEEP | — |
| Value after reset: | 0 | 0 | 1/0*1 | 1/0*1 | 1/0*1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1/0*1 | 1/0*1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | These bits are read as 0. | R |
| 1 | CPUSLEEP | Sleep mode status<br>This bit is unpredictable when the MCU is in Software Standby mode, Snooze mode, or Deep Software Standby mode.<br>0: CPU is not in Sleep mode<br>1: CPU in Sleep mode | R |
| 2 | CPUSTOPCLK | CPU clock status<br>This bit is unpredictable when the MCU is in Deep Software Standby mode.<br>0: CPU clock is not stopped.<br>1: CPU clock is stopped. | R |
| 7:3 | — | These bits are read as 0. | R |
| 8 | — | These bits are read as 1. | R |
| 10:9 | — | These bits are read as 0. | R |
| 11 | BOOTMD | Boot mode status<br>0: Device is not in Boot mode<br>1: Device is in Boot mode | R |
| 12 | DBGFUNCEN | Debugger status<br>0: Debugger connection is not available<br>1: Debugger function is enabled | R |
| 13 | SECDBG | Secure Debug status<br>0: Secure Debug is not available<br>1: Secure Debug is available | R |
| 31:14 | — | These bits are read as 0. | R |

Note 1. Depends on the MCU status.

### 2.7.5.2 MCUCTRL : MCU Control Register

Base address: CPU_OCD = 0x8000_0000

Offset address: 0x410

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | CPUWAIT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | DBIRQ | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | DBIRQ*2 | Debug Interrupt Request<br>Writing 1 to the bit wakes up the MCU from low power mode. The condition can be cleared by writing 0 to the DBIRQ bit.<br>0: Debug interrupt not requested<br>1: Debug interrupt requested | R/W |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | CPUWAIT*2 | CPU Wait Setting<br>Write 1 to assert CPUWAIT, write 0 to deassert CPUWAIT*1.<br>0: Clear CPUWAIT to low<br>1: Set CPUWAIT to high | R/W |
| 31:17 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. CPUWAIT is used to prevent the processor from executing code immediately after reset.
Note 2. Access (R/W) to bit is valid only when Debug Level is DBG1 or DBG2.

### 2.7.5.3 OCDREG CoreSight component registers

The OCDREG module provides the CoreSight component registers defined in the Arm CoreSight architecture.

Table 2.15 shows the registers. See reference 4. in section 2.14. References for details of each register.

**Table 2.15 OCDREG CoreSight component registers**

| Name | Address | Access size | R/W | Initial value |
|---|---|---|---|---|
| PID4 | 0x8000_0FD0 | 32 bits | R | 0x00000004 |
| PID5 | 0x8000_0FD4 | 32 bits | R | 0x00000000 |
| PID6 | 0x8000_0FD8 | 32 bits | R | 0x00000000 |
| PID7 | 0x8000_0FDC | 32 bits | R | 0x00000000 |
| PID0 | 0x8000_0FE0 | 32 bits | R | 0x00000004 |
| PID1 | 0x8000_0FE4 | 32 bits | R | 0x00000030 |
| PID2 | 0x8000_0FE8 | 32 bits | R | 0x0000000A |
| PID3 | 0x8000_0FEC | 32 bits | R | 0x00000000 |
| CID0 | 0x8000_0FF0 | 32 bits | R | 0x0000000D |
| CID1 | 0x8000_0FF4 | 32 bits | R | 0x000000F0 |
| CID2 | 0x8000_0FF8 | 32 bits | R | 0x00000005 |
| CID3 | 0x8000_0FFC | 32 bits | R | 0x000000B1 |

## 2.7.6 CPUDSAR : CPU Debug Security Attribution Register

Base address: CPSCU = 0x4000_8000

Offset address: 0x1B0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | CPUDSA0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CPUDSA0 | CPU Debug Security Attribution 0<br>0: Secure<br>1: Non-secure | R/W |
| 31:1 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only secure access can write to this register. Both secure access and non-secure read access are allowed. Non-secure write access is denied, and no TrustZone access error is generated.

Note: This register is write-protected by PRCR register.

When Debug level of the MCU is DBG2, by guarding entire EPPB bus, the non-secure access from CPU to debug related components is completely controlled by the current value of the CPUDSA0 bit. Since this bit is modifiable only when CPU is in secure state, user must be aware of the CPUDSAR register before using CoreSight debug components.

### CPUDSA0 bit (CPU Debug Security Attribution 0)

Security attributes of register for accessing the debug component of the CPU.

0: Debug component can only be accessed with secure access.

1: There is no restriction on accessing the debug component.

## 2.7.7 Processing on Error response generated by CPU access

In addition to the specific-error detection specification of the Arm Cortex-M33 processor, this MCU also provides additional error information which is described in section 14, Buses.

This section describes how to handle the additional error information with no conflict to that of the Arm Cortex-M33 processor.

Table 2.16 shows error detection modules, which are also described in section 14, Buses. These error detection modules not only provide error information on the bus module, but also notify the processor to trigger the exception handler.

**Table 2.16 Error detection modules**

| | NMI/RESET request | Interrupt | Bus error status register | Error address register Error RW register |
|---|---|---|---|---|
| Slave TZF | NMISR.TZFST | Bus Fault*1 (Hard Fault) | BUS.BUSnERRSTAT.STERRSTAT | BUS.BTZFnERRADD<br>BUS.BTZFnERRRW |
| Slave bus error | — | Bus Fault*1 (Hard Fault) | BUS.BUSnERRSTAT.SLERRSTAT | BUS.BUSnERRADD<br>BUS.BUSnERRRW |
| Illegal address access error | — | Bus Fault*1 (Hard Fault) | BUS.BUSnERRSTAT.ILERRSTAT | BUS.BUSnERRADD<br>BUS.BUSnERRRW |

Note 1. A Bus Fault can be treated as HardFault. For details, see *ARM® Cortex®-M33 Device Generic User Guide* in the section 2.14. References.

To prevent unexpected operation, when handling the exception, additional operation should be added into exception routing.

BusFault when occurred by error detected as shown in Table 2.16:

- See section 14, Buses for the error information in the corresponding register
- Clear the data in cache for the error address
- Clear the Error Status register in the bus module
- Service exception handling with Arm-guided operation

For a Bus Fault that is not detected in the Renesas-specific error detection module (occurred inside the Arm Cortex-M33 core), see the *ARM® Cortex®-M33 Device Generic User Guide* to handle this case.

In the system bus specification, there is a specific case for Slave TrustZone Filter, that is, if an error is selected to generate an NMI, then before the processor handles the Bus Fault exception, NMI with higher priority takes the exception first. Therefore, use the BusFault handler and not NMI handler to handle this error. In other words, the NMI status should be cleared but the error status bit should not be cleared to ensure that BusFault captures all the error information.

Figure 2.4 and Figure 2.5 show the recommended flows for NMI handler and BusFault handler for the errors described in Table 2.16.

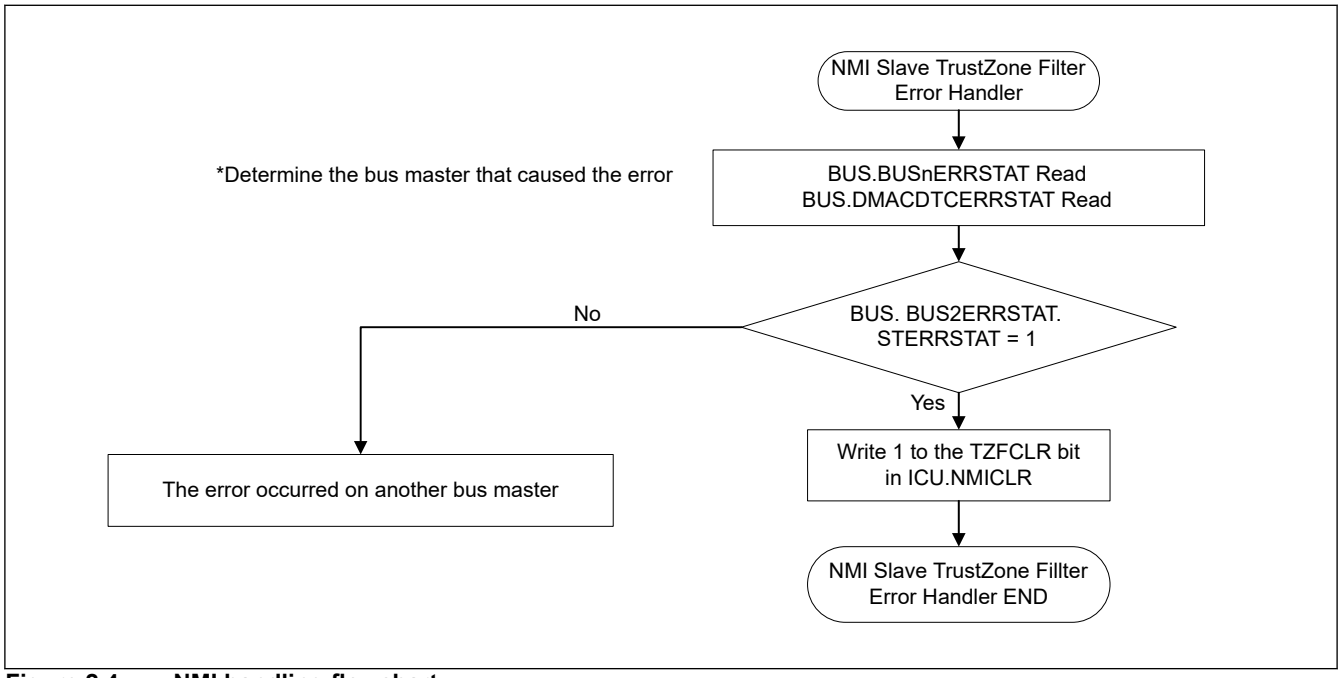



**Figure 2.4 NMI handling flowchart**

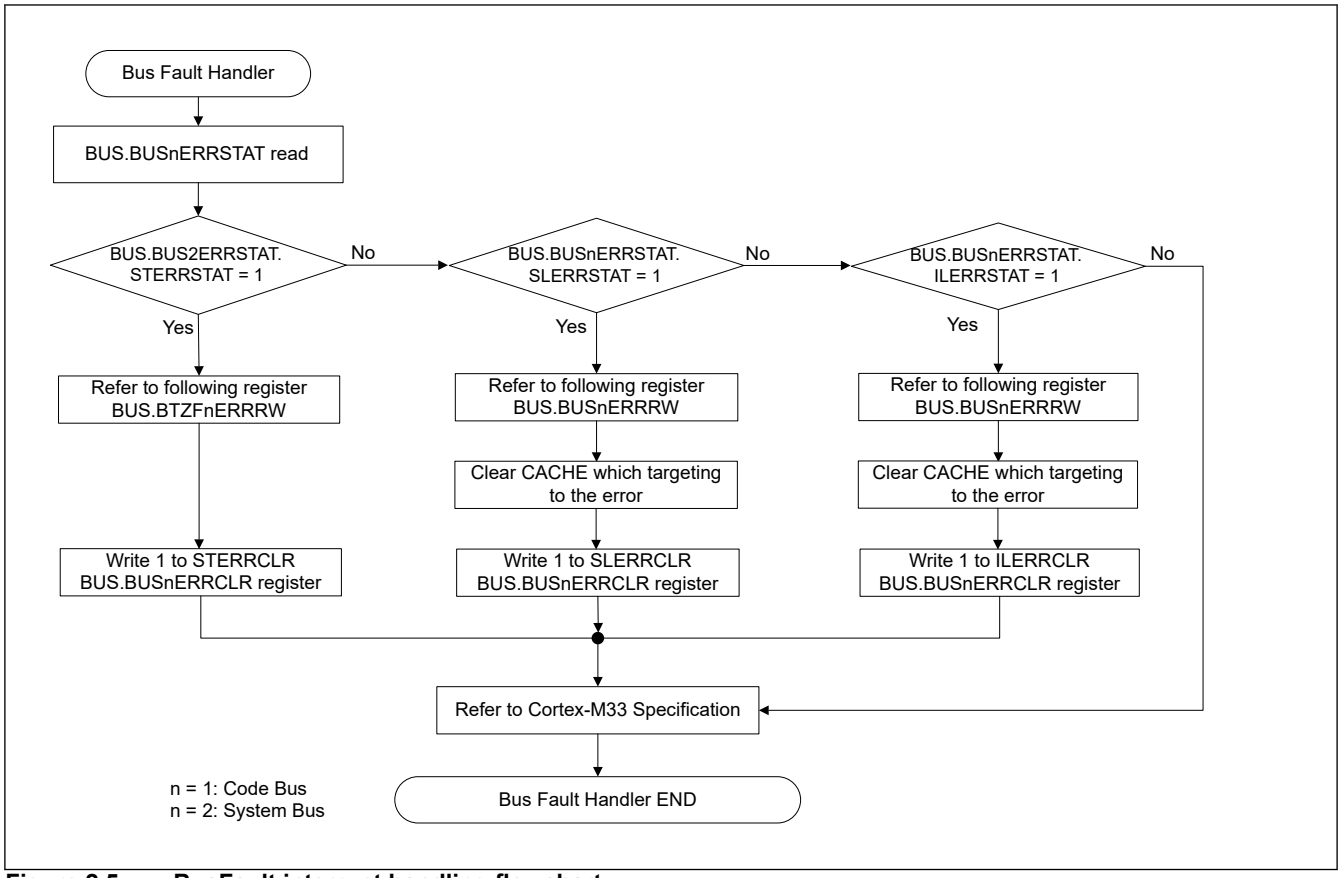



**Figure 2.5 BusFault interrupt handling flowchart**

## 2.8 CoreSight Cross Trigger Interface (CTI)

As shown in Figure 2.6, the input and output of a Cross Trigger Interface (CTI) interact with each other through four CTM channels. Input of a CTI can be used to trigger the output of another CTI using the four CTM channels.

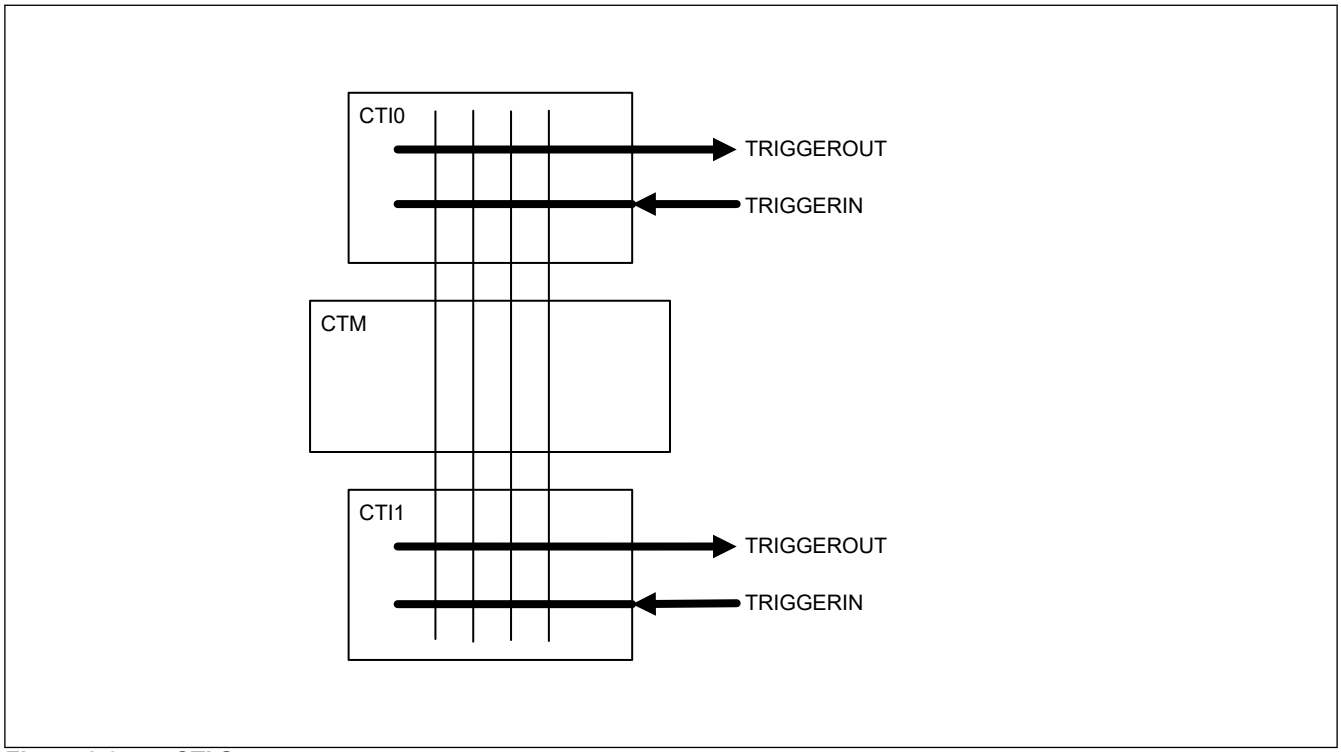



**Figure 2.6 CTI System**

Debug Interrupt Request (DBGIRQ) is controlled by MCUCTRL register in OCDREG module.

**Table 2.17 CTI Trigger signals**

| Number of CTI channel | CTITRIGIN | | CTITRIGOUT | |
|---|---|---|---|---|
| CTI0 (Debug common) | 0 | ACQCOMP | 0 | — |
| | 1 | FULL | 1 | — |
| | 2 | DBIRQ | 2 | ETB FLUSHIN |
| | 3 | — | 3 | ETB TRIGIN |
| | 4 | — | 4 | — |
| | 5 | — | 5 | — |
| | 6 | — | 6 | — |
| | 7 | — | 7 | — |
| CTI1 (CPU) | 0 | Processor Halted | 0 | Processor debug request |
| | 1 | DWT Comparator Output 0 | 1 | Processor Restart |
| | 2 | DWT Comparator Output 1 | 2 | CTIIRQ[0] (Connected to IRQ96) |
| | 3 | DWT Comparator Output 2 | 3 | CTIIRQ[1] (Connected to IRQ97) |
| | 4 | ETM Event Output 0 | 4 | ETM Event Input 0 |
| | 5 | ETM Event Output 1 | 5 | ETM Event Input 1 |
| | 6 | — | 6 | ETM Event Input 2 |
| | 7 | — | 7 | ETM Event Input 3 |

## 2.9 CoreSight ATB Funnel

There is one CoreSight ATB funnel in the MCU. The funnel has two ATB slaves and one ATB master, and it selects the debug trace source from ETM and ITM to ETB. Figure 2.7 shows the CoreSight ATB connection in the MCU.

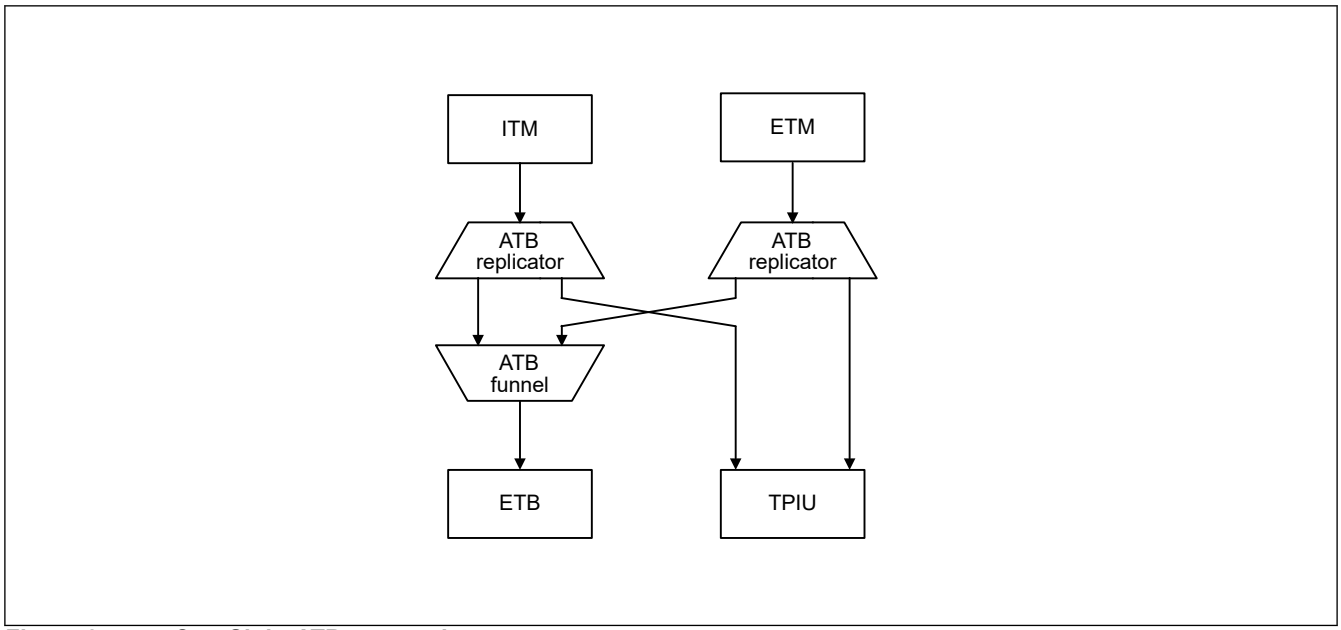



**Figure 2.7 CoreSight ATB connection**

Table 2.18 shows the ATB slave connection for the funnel.

**Table 2.18 ATB slave connection (1 of 2)**

| ATB slave number | Connected trace source |
|---|---|
| #0 | ITM |

**Table 2.18 ATB slave connection (2 of 2)**

| ATB slave number | Connected trace source |
|---|---|
| #1 | ETM |

See reference 4. in section 2.14. References for details of the ATB and funnel.

## 2.10 Break Point Unit

The MCU has Break Point Unit. See BreakPoint unit chapter of reference 1. in section 2.14. References for details about register description of this module.

## 2.11 CoreSight Time Stamp Generator

A CoreSight Time Stamp Generator provides a CPU clock-based timestamp to ITM and ETM.The timestamp is generated by a 64-bit counter. See reference 4. in section 2.14. References for details.

## 2.12 SysTick Timer

The MCU has SysTick timer that provides two 24-bit down counters, non-secure and secure counters. The timer can select SysTick timer clock (SYSTICCLK) or System clock (ICLK).

See section 8, Clock Generation Circuit and reference 1. in section 2.14. References for details.

Note: SysTick timer counter operation is enabled by signal synchronized with CPU clock. Therefore, the counter might not operate correctly if the CPU clock is slower than the SysTick timer clock. In other words, clock setting must satisfy the following: CPU clock ≥ SysTick timer clock (LOCO: 32.768 kHz).

## 2.13 OCD Emulator Connection

In this product, the MCU confirms the access permission for Non-secure debug and Non-secure chip resources by checking Debug level is DBG1 or higher. For full access permission for debug and chip resources, Secure debug level DBG2 is required.

Figure 2.8 shows a block diagram of SWD authentication mechanism.

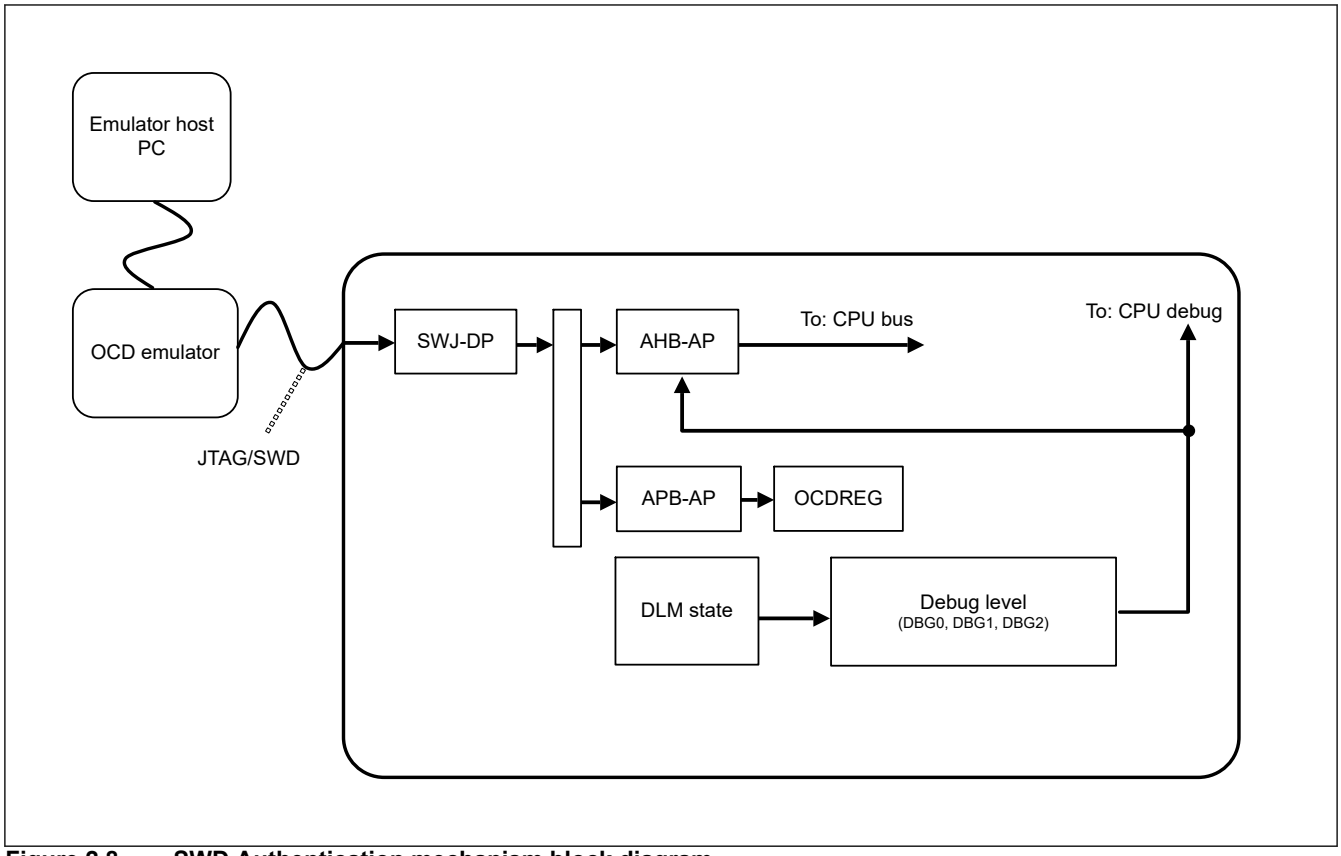



**Figure 2.8 SWD Authentication mechanism block diagram**

Three levels of debug capability are available, DBG0, DBG1, and DBG2, which correspond to the Device Level Management (DLM) states. When debug level is DBG0, access to debug components and system bus from OCD emulator is not permitted. When debug level is DBG1 or DBG2, the corresponding non-secure or secure debug components and system bus can be accessed from the OCD emulator. See Table 2.5 for more information about debug levels.

## 2.13.1 DBGEN

After the OCD emulator gets access permission, the OCD emulator must set the DBGEN bit in the System Control OCD Control Register (SYOCDCR). In addition, the OCD emulator must clear the DBGEN bit before disconnecting it. See section 10, Low Power Modes for details.

## 2.13.2 Restrictions on Connecting an OCD emulator

This section describes the restrictions on emulator access.

### 2.13.2.1 Starting connection while in low power mode

When starting a JTAG/SWD connection from an OCD emulator, the MCU must be in Normal or Sleep mode. If the MCU is in Software Standby, Snooze, or Deep Software Standby mode, the OCD emulator can cause the MCU to hang.

### 2.13.2.2 Changing low power mode while in OCD mode

When the MCU is in OCD mode, the low power mode can be changed. However, system bus access from AHB-AP is prohibited in Software Standby, Snooze or Deep Software Standby mode. Only SWJ-DP, APB-AP, and OCDREG can be accessed from the OCD emulator in these modes. Table 2.19 shows the restrictions.

**Table 2.19 Restrictions by mode (1 of 2)**

| Active mode | Start OCD emulator connection | Change low power mode | Access AHB-AP and system bus | Access APB-AP and OCDREG |
|---|---|---|---|---|
| Normal | Yes | Yes | Yes | Yes |

**Table 2.19 Restrictions by mode (2 of 2)**

| Active mode | Start OCD emulator connection | Change low power mode | Access AHB-AP and system bus | Access APB-AP and OCDREG |
|---|---|---|---|---|
| Sleep | Yes | Yes | Yes | Yes |
| Software Standby | No | Yes | No | Yes |
| Snooze | No | Yes | No | Yes |
| Deep Software Standby | No | Yes | No | Yes |

If system bus access is required in Software Standby, Snooze, or Deep Software Standby mode, set the MCUCTRL.DBIRQ bit in OCDREG to wake up the MCU from the low power modes. Simultaneously, by asserting the MCUCTRL.DBIRQ bit in OCDREG, the OCD emulator can wake up the MCU without starting CPU execution by using a CPU break.

#### 2.13.2.3 Connecting sequence and JTAG/SWD authentication

1. Connect the OCD debugger to the MCU through the JTAG or SWD interface.
2. Set up SWJ-DP to access DAP bus.
   In the setup, the OCD emulator must assert CDBGPWRUPREQ in the SWJDP.
   Control Status Register, and then wait until CDBGPWRUPACK in the same register is asserted.
3. Set the APB-AP to access OCDREG. This APB-AP is connected to DAP bus port 1.
4. Set MCUCTRL.CPUWAIT = 1.
5. Confirm the debug capability of device by reading MCUSTAT:
   - If Debug function is prohibited, this device is not able to debug.
   - If Debug function is enabled and secure debug is not available, only non-secure debug is available.
   - If Debug function is enabled and secure debug is available, full debug functions are available.

   If Debug function is available, set debug-related register then clear MCUCTRL.CPUWAIT = 0.
6. Set up the AHB-AP to access the system address space. The AHB-AP is connected to DAP bus port 0.
7. Set SYOCDCR.DBGEN to 1.
8. Start accessing the CPU debug resources using the AHB-AP.

Note: Debug level is determined by the current DLM state of product.

## 2.14 References

1. *ARM®v8-M Architecture Reference Manual* (ARM DDI 0553B.a)
2. *ARM® Cortex®-M33 Processor Technical Reference Manual* (ARM 100230)
3. *ARM® Cortex®-M33 Device Generic User Guide* (ARM 100235)
4. *ARM® CoreSight™ SoC-400 Technical Reference Manual* (ARM DDI 0480G)
5. *ARM® CoreSight™ Architecture Specification* (ARM IHI 0029E)

# 3. Operating Modes

## 3.1 Overview

Table 3.1 shows the selection of operating modes by the mode-setting pin. For details, see section 3.2. Details of Operating Modes. Operation starts with the on-chip flash memory enabled, regardless of the mode in which operation started.

**Table 3.1 Selection of operating modes by the mode-setting pin**

| Mode-setting pin (MD) | Operating mode | On-chip Flash |
|---|---|---|
| 1 | Single-chip mode | Enable |
| 0 | SCI / USB boot mode | Enable |

## 3.2 Details of Operating Modes

### 3.2.1 Single-Chip Mode

In single-chip mode, all I/O pins are available for use as input or output port, inputs or outputs for peripheral functions, or as interrupt inputs.

When a reset is released while the MD pin is high, the MCU starts in single-chip mode and the on-chip flash is enabled.

### 3.2.2 SCI Boot Mode

In this mode, the on-chip flash memory programming routine (SCI boot program), stored in the boot area within the MCU, is used. The on-chip flash, including code flash memory and data flash memory, can be modified from outside the MCU by using a universal asynchronous receiver/transmitter (UART) SCI. For details, see section 44, Flash Memory. The MCU starts in SCI boot mode if the MD pin is held low on release from the reset state.

### 3.2.3 USB Boot Mode

In this mode, the on-chip flash memory programming routine (USB boot program), stored in the boot area within the MCU, is used. The on-chip flash, including the code flash memory and data flash memory, can be modified from outside the MCU by using the USB. For details, see section 44, Flash Memory. The MCU starts in USB boot mode if the MD pin is held low on release from the reset state.

## 3.3 Operating Modes Transitions

### 3.3.1 Operating Mode Transitions as Determined by the Mode-Setting Pin

Figure 3.1 shows operating mode transitions determined by the MD pin settings.

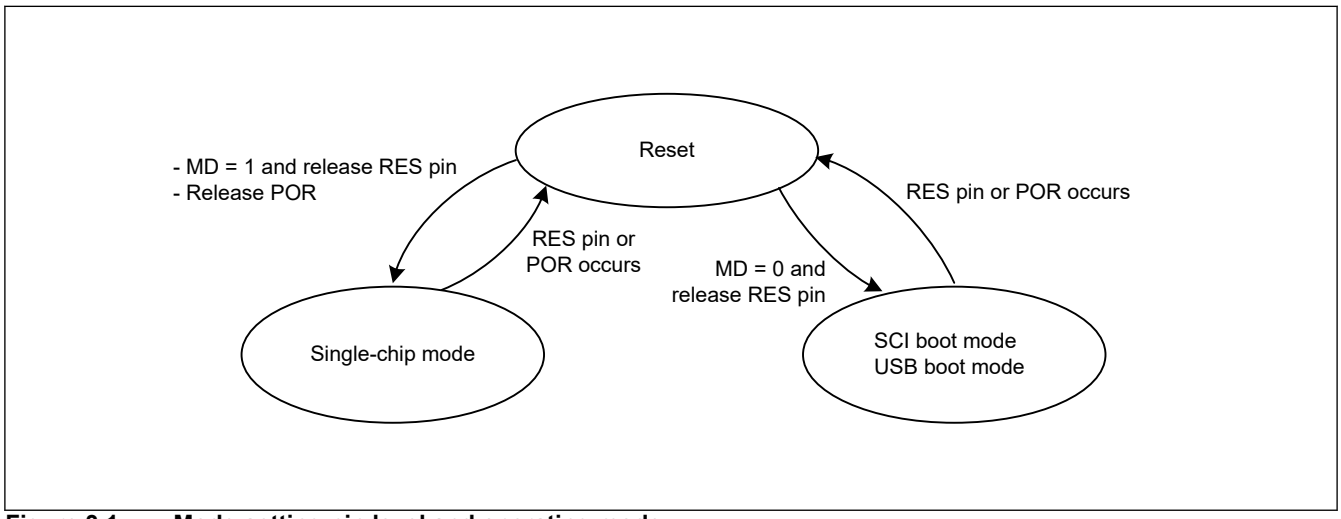
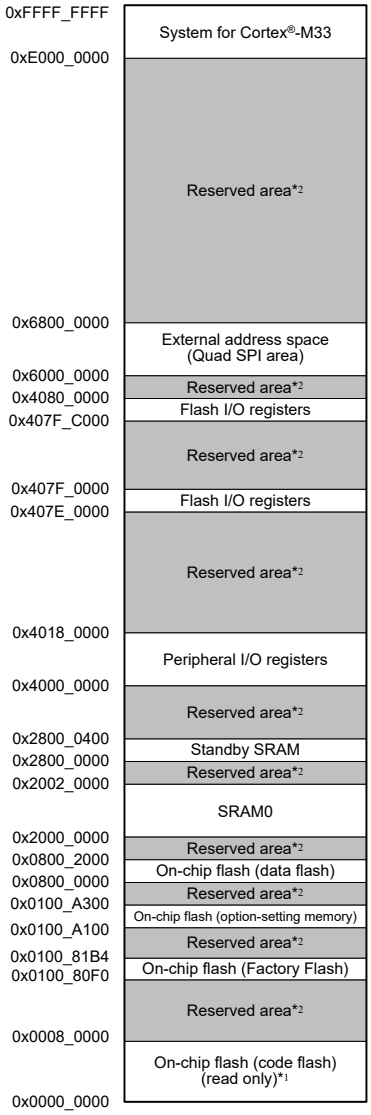



**Figure 3.1 Mode-setting pin level and operating mode**

# 4. Address Space

## 4.1 Address Space

The MCU supports a 4-GB linear address space ranging from 0x0000_0000 to 0xFFFF_FFFF that can contain both program and data. Figure 4.1 shows the memory map.




Note 1. See Table 4.1. The capacity of the flash differs depending on the product.
Note 2. Do not access reserved areas.

**Figure 4.1 Memory map**

**Table 4.1 Capacity of the code flash memory, data flash memory, and SRAM0**

| Code flash memory | | Data flash memory | | SRAM0 | |
|---|---|---|---|---|---|
| **Capacity** | **Address** | **Capacity** | **Address** | **Capacity** | **Address** |
| 512 KB | 0x0000_0000 - 0x0007_FFFF | 8 KB | 0x0800_0000 - 0x0800_1FFF | 128 KB | 0x2000_0000 - 0x2001_FFFF |
| 384 KB | 0x0000_0000 - 0x0005_FFFF | | | | |
| 256 KB | 0x0000_0000 - 0x0003_FFFF | | | | |

# 5. Resets

## 5.1 Overview

The MCU provides 13 resets.

Table 5.1 lists the reset names and sources.

**Table 5.1 Reset names and sources**

| Reset name | Source |
|---|---|
| RES pin reset | Voltage input to the RES pin is driven low |
| Power-on reset | VCC rise (voltage detection $V_{POR}$)*1 |
| Independent watchdog timer reset | IWDT underflow or refresh error |
| Watchdog timer reset | WDT underflow or refresh error |
| Voltage monitor 0 reset | VCC fall (voltage detection $V_{det0}$)*1 |
| Voltage monitor 1 reset | VCC fall (voltage detection $V_{det1}$)*1 |
| Voltage monitor 2 reset | VCC fall (voltage detection $V_{det2}$)*1 |
| SRAM parity error reset | SRAM parity error detection |
| SRAM ECC error reset | SRAM ECC error detection |
| Bus master MPU error reset | Bus master MPU error detection |
| TrustZone error reset | TrustZone error detection |
| Deep software standby reset | Deep software standby mode is canceled by an interrupt |
| Software reset | Register setting (use the software reset bit AIRCR.SYSRESETREQ) |

Note 1. For details on the voltages to be monitored ($V_{POR}$, $V_{det0}$, $V_{det1}$, and $V_{det2}$), see section 7, Low Voltage Detection (LVD) and section 47, Electrical Characteristics.

The internal state and pins are initialized by a reset. Table 5.2 and Table 5.3 list the targets initialized by resets.

**Table 5.2 Reset detect flags initialized by each reset source (1 of 3)**

| Flag to be initialized | Reset source | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | RES pin reset | Power-on reset | Voltage monitor 0 reset | Independent watchdog timer reset | Watchdog timer reset | Voltage monitor 1 reset | Voltage monitor 2 reset | Software reset |
| Power-On Reset Detect Flag (RSTSR0.PORF) | ✓ | — | — | — | — | — | — | — |
| Voltage Monitor 0 Reset Detect Flag (RSTSR0.LVD0RF) | ✓ | ✓ | — | — | — | — | — | — |
| Independent Watchdog Timer Reset Detect Flag (RSTSR1.IWDTRF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| Watchdog Timer Reset Detect Flag (RSTSR1.WDTRF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| Voltage Monitor 1 Reset Detect Flag (RSTSR0.LVD1RF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| Voltage Monitor 2 Reset Detect Flag (RSTSR0.LVD2RF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| Software Reset Detect Flag (RSTSR1.SWRF) | ✓ | ✓ | ✓ | — | — | — | — | — |

**Table 5.2 Reset detect flags initialized by each reset source (2 of 3)**

| Flag to be initialized | Reset source | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | RES pin reset | Power-on reset | Voltage monitor 0 reset | Independent watchdog timer reset | Watchdog timer reset | Voltage monitor 1 reset | Voltage monitor 2 reset | Software reset |
| SRAM Parity Error Reset Detect Flag (RSTSR1.RPERF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| SRAM ECC Error Reset Detect Flag (RSTSR1.REERF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| Bus Master MPU Error Reset Detect Flag (RSTSR1.BUSMRF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| TrustZone Error Reset Detect Flag (RSTSR1.TZERF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| Deep Software Standby Reset Detect Flag (RSTSR0.DPSRSTF) | ✓ | ✓ | ✓ | — | — | — | — | — |
| Cold Start/Warm Start Determination Flag (RSTSR2.CWSF) | — | ✓ | — | — | — | — | — | — |

**Table 5.2 Reset detect flags initialized by each reset source (3 of 3)**

| Flag to be initialized | Reset source | | | | | |
|---|---|---|---|---|---|---|
| | SRAM parity error reset | SRAM ECC error reset | Bus master MPU error reset | TrustZone reset error | Deep Software Standby reset | |
| | | | | | DEEPCUT[0] = 0 | DEEPCUT[0] = 1 |
| Power-On Reset Detect Flag (RSTSR0.PORF) | — | — | — | — | — | — |
| Voltage Monitor 0 Reset Detect Flag (RSTSR0.LVD0RF) | — | — | — | — | — | — |
| Independent Watchdog Timer Reset Detect Flag (RSTSR1.IWDTRF) | — | — | — | — | ✓ | ✓ |
| Watchdog Timer Reset Detect Flag (RSTSR1.WDTRF) | — | — | — | — | ✓ | ✓ |
| Voltage Monitor 1 Reset Detect Flag (RSTSR0.LVD1RF) | — | — | — | — | — | — |
| Voltage Monitor 2 Reset Detect Flag (RSTSR0.LVD2RF) | — | — | — | — | — | — |
| Software Reset Detect Flag (RSTSR1.SWRF) | — | — | — | — | ✓ | ✓ |
| SRAM Parity Error Reset Detect Flag (RSTSR1.RPERF) | — | — | — | — | ✓ | ✓ |
| SRAM ECC Error Reset Detect Flag (RSTSR1.REERF) | — | — | — | — | ✓ | ✓ |
| Bus Master MPU Error Reset Detect Flag (RSTSR1.BUSMRF) | — | — | — | — | ✓ | ✓ |
| TrustZone Error Reset Detect Flag (RSTSR1.TZERF) | — | — | — | — | ✓ | ✓ |
| Deep Software Standby Reset Detect Flag (RSTSR0.DPSRSTF) | — | — | — | — | — | — |
| Cold Start/Warm Start Determination Flag (RSTSR2.CWSF) | — | — | — | — | — | — |

Note: ✓ : Initialized to 0
— : Not initialized

**Table 5.3 Module-related registers initialized by each reset source (1 of 4)**

| Registers to be initialized | | Reset source | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | RES pin reset | Power-on reset | Voltage monitor 0 reset | Independent watchdog timer reset | Watchdog timer reset | Voltage monitor 1 reset | Voltage monitor 2 reset | Software reset |
| Independent watchdog timer registers | IWDTRR, IWDTSR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Watchdog timer registers | WDTRR, WDTCR, WDTSR, WDTRCR, WDTCSTPR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Voltage monitor function 1 registers | LVD1CR0,LVD1CMPCR | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — |
| | LVD1CR1/LVD1SR | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — |
| Voltage monitor function 2 registers | LVD2CR0, LVD2CMPCR | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — |
| | LVD2CR1/LVD2SR | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — |
| SOSC register | SOSCCR | — | ✓*1 | — | — | — | — | — | — |
| | SOMCR | — | — | — | — | — | — | — | — |
| LOCO registers | LOCOCR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | LOCOUTCR | — | ✓ | ✓ | — | — | ✓ | ✓ | — |
| MOSC register | MOMCR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Realtime Clock (RTC) register*2 | | — | — | — | — | — | — | — | — |
| AGTn registers (n = 0 to 3) | | — | ✓ | ✓ | — | — | ✓ | ✓ | — |
| AGTn registers (n = 4, 5) | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| USBFS registers | Except DPUSR0R, DPUSR1R | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | DPUSR0R, DPUSR1R | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Bus, MPU and TrustZone error registers*4 | BUS_ERROR_ADDRESS Register BUS_ERROR_STATUS Register | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Pin states (except XCIN/XCOUT pin) | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Pin states (XCIN/XCOUT pin) | | — | — | — | — | — | — | — | — |
| Low-power function registers | DPSBYCR, DPSIER0 to DPSIER3, DPSIFR0 to DPSIFR3, DPSIEGR0 to DPSIEGR2 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | SYOCDCR | — | ✓ | — | — | — | — | — | — |
| Stop Control Register | PL2LDOSCR | — | ✓ | — | — | — | — | — | — |

**Table 5.3 Module-related registers initialized by each reset source (2 of 4)**

| Registers to be initialized | | Reset source | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | RES pin reset | Power-on reset | Voltage monitor 0 reset | Independent watchdog timer reset | Watchdog timer reset | Voltage monitor 1 reset | Voltage monitor 2 reset | Software reset |
| Security Attribute Registers | CPUDSAR, RSTSAR, LVDSAR, CGFSAR, LPMSAR, DPFSAR, BBFSAR, ICUSARx, BUSSARA, BUSSARB, MMPUSARA, MMPUSARB, DMACSAR, DTCSAR, ELCSARA, ELCSARB, ELCSARC, PmSAR, SRAMSAR, STBRAMSAR, FSAR, PSARB, PSARC, PSARD, PSARE, MSSAR, TZFSAR | ✓*5 | ✓ | ✓*5 | ✓*5 | ✓*5 | ✓*5 | ✓*5 | ✓*5 |
| Battery backup register | VBTBKRn,VBTICTLR | — | — | — | — | — | — | — | — |
| | VBTBER | — | ✓ | — | — | — | — | — | — |
| Registers other than those shown, CPU, and internal state | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

**Table 5.3 Module-related registers initialized by each reset source (3 of 4)**

| Registers to be initialized | | Reset source | | | | | |
|---|---|---|---|---|---|---|---|
| | | SRAM parity error reset | SRAM ECC error reset | Bus master MPU error reset | TrustZone error reset | Deep Software Standby reset | |
| | | | | | | DEEPCUT[0] = 0 | DEEPCUT[0] = 1 |
| Independent watchdog timer registers | IWDTRR, IWDTSR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Watchdog timer registers | WDTRR, WDTCR, WDTSR, WDTRCR, WDTCSTPR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Voltage monitor function 1 registers | LVD1CR0,LVD1CMPCR | — | — | — | — | — | — |
| | LVD1CR1 / LVD1SR | — | — | — | — | ✓ | ✓ |
| Voltage monitor function 2 registers | LVD2CR0, LVD2CMPCR | — | — | — | — | — | — |
| | LVD2CR1/LVD2SR | — | — | — | — | ✓ | ✓ |
| SOSC register | SOSCCR | — | — | — | — | — | — |
| | SOMCR | — | — | — | — | — | — |
| LOCO registers | LOCOCR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | LOCOUTCR | — | — | — | — | — | ✓ |
| MOSC register | MOMCR | ✓ | ✓ | ✓ | ✓ | — | — |
| Realtime Clock (RTC) register*2 | | — | — | — | — | — | — |
| AGTn registers (n = 0 to 3) | | — | — | — | — | — | ✓ |
| AGTn registers (n = 4,5) | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

**Table 5.3 Module-related registers initialized by each reset source (4 of 4)**

| Registers to be initialized | | Reset source | | | | | |
|---|---|---|---|---|---|---|---|
| | | SRAM parity error reset | SRAM ECC error reset | Bus master MPU error reset | TrustZone error reset | Deep Software Standby reset | |
| | | | | | | DEEPCUT[0] = 0 | DEEPCUT[0] = 1 |
| USBFS registers | Except DPUSR0R, DPUSR1R,DPBCCTRL | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | DPUSR0R, DPUSR1R,DPBCCTRL | ✓ | ✓ | ✓ | ✓ | — | ✓ |
| Bus, MPU and TrustZone error registers*4 | BUS_ERROR_ADDRESS Register BUS_ERROR_STATUS Register | ✓ | ✓ | — | — | ✓ | ✓ |
| Pin states (except XCIN/XCOUT pin) | | ✓ | ✓ | ✓ | ✓ | *3 | *3 |
| Pin states (XCIN/XCOUT pin) | | — | — | — | — | — | — |
| Low-power function registers | DPSBYCR, DPSIER0 to DPSIER3, DPSIFR0 to DPSIFR3, DPSIEGR0 to DPSIEGR2 | ✓ | ✓ | ✓ | ✓ | — | — |
| | SYOCDCR | — | — | — | — | — | — |
| Stop Control Register | PL2LDOSCR | — | — | — | — | — | — |
| Security Attribute Registers | CPUDSAR, RSTSAR, LVDSAR, CGFSAR, LPMSAR, DPFSAR, BBFSAR, ICUSARx, BUSSARA, BUSSARB, MMPUSARA, MMPUSARB, DMACSAR, DTCSAR, ELCSARA, ELCSARB, ELCSARC, PmSAR, SRAMSAR, STBRAMSAR, FSAR, PSARB, PSARC, PSARD, PSARE, MSSAR, TZFSAR | ✓*5 | ✓*5 | ✓*5 | ✓*5 | ✓*6 | ✓*6 |
| Battery backup register | VBTBKRn,VBTICTLR | — | — | — | — | — | — |
| | VBTBER | — | — | — | — | — | — |
| Registers other than those shown, CPU, and internal state | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Note: ✓ : Initialized
— : Not initialized
Note 1. For the initial value of each register, see section 8, Clock Generation Circuit.
Note 2. The RTC has a software reset. Some control bits are not initialized by all types of resets. For details on the target bits, see section 23, Realtime Clock (RTC).
Note 3. Depends on the setting of DPSBYCR.IOKEEP.
Note 4. Some control bits are not initialized by all types of resets. For details on the target bits, see section 14, Buses
Note 5. Reset does not occur while the debugger is connected (DBGSTR.CDBGPWRUPREQ = 1) even if On-chip debugger is disabled (SYOCDCR.DBGEN = 0).
Note 6. Reset does not occur while On-chip debugger is enabled (SYOCDCR.DBGEN = 1).

The RTC is not initialized by any reset source. SOSC and LOCO can be selected as the clock sources of the RTC.

Table 5.4 and Table 5.5 show the states of SOSC and LOCO when a reset occurs.

**Table 5.4 States of SOSC when a reset occurs**

| | | Reset source | |
|---|---|---|---|
| | | POR | Other |
| SOSC | Enable or disable | Initialized to enable | Continue with the state that was selected before the reset occurred |
| | Drive capability | Continue with the state that was selected before the reset occurred | |

**Table 5.5 States of LOCO when a reset occurs**

| | | Reset source | |
|---|---|---|---|
| | | POR, LVD0, LVD1, LVD2, Deep Software Standby (DEEPCUT[0] = 1) | Other |
| LOCO | Enable or disable | Initialized to enable | |
| | Oscillation accuracy*1 | Initialized to accuracy before trimming by power-on (accuracy: ± 10%) | Continue with the accuracy that was trimmed by LOCOUTCR |

Note 1. The LOCO User Trimming Control Register (LOCOUTCR) is reset by POR, LVD0, LVD1, LVD2, and Deep Software Standby (DEEPCUT[0] = 1) resets, returning the LOCO to the default oscillation accuracy. This can affect RTC accuracy if the RTC uses the LOCO (with a user trimming value in LOCOUTCR) as the RTC source clock. To restore the pre-reset LOCO oscillation accuracy, reload the required trimming value into LOCOUTCR after any of these resets.

When a reset is released, reset exception handling starts.

Table 5.6 lists the pin related to the reset function.

**Table 5.6 Pin related to reset**

| Pin name | I/O | Function |
|---|---|---|
| RES | Input | Reset pin |

## 5.2 Register Descriptions

### 5.2.1 RSTSAR : Reset Security Attribution Register

Base address: SYSC = 0x4001_E000

Offset address: 0x3C4

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | NONSEC2 | NONSEC1 | NONSEC0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | NONSEC0 | Non Secure Attribute bit 0<br>Target register: Reset Status Register 0<br>0: Secure<br>1: Non Secure | R/W |
| 1 | NONSEC1 | Non Secure Attribute bit 1<br>Target register: Reset Status Register 1<br>0: Secure<br>1: Non Secure | R/W |
| 2 | NONSEC2 | Non Secure Attribute bit 2<br>Target register: Reset Status Register 2<br>0: Secure<br>1: Non Secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:3 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

### NONSEC0 bit (Non Secure Attribute bit 0)

This bit controls the security attribute of RSTSR0.

### NONSEC1 bit (Non Secure Attribute bit 1)

This bit controls the security attribute of RSTSR1.

### NONSEC2 bit (Non Secure Attribute bit 2)

This bit controls the security attribute of RSTSR2.

## 5.2.2 RSTSR0 : Reset Status Register 0

Base address: SYSC = 0x4001_E000

Offset address: 0x410

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DPSRSTF | — | — | — | LVD2RF | LVD1RF | LVD0RF | PORF |
| Value after reset: | x*1 | 0 | 0 | 0 | x*1 | x*1 | x*1 | x*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PORF | Power-On Reset Detect Flag<br>0: Power-on reset not detected<br>1: Power-on reset detected | R/W*2 |
| 1 | LVD0RF | Voltage Monitor 0 Reset Detect Flag<br>0: Voltage monitor 0 reset not detected<br>1: Voltage monitor 0 reset detected | R/W*2 |
| 2 | LVD1RF | Voltage Monitor 1 Reset Detect Flag<br>0: Voltage monitor 1 reset not detected<br>1: Voltage monitor 1 reset detected | R/W*2 |
| 3 | LVD2RF | Voltage Monitor 2 Reset Detect Flag<br>0: Voltage monitor 2 reset not detected<br>1: Voltage monitor 2 reset detected | R/W*2 |
| 6:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | DPSRSTF | Deep Software Standby Reset Detect Flag<br>0: Deep software standby mode cancellation not requested by an interrupt.<br>1: Deep software standby mode cancellation requested by an interrupt. | R/W*2 |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. The value after reset depends on the reset source.
Note 2. The register is cleared when a reset source listed in Table 5.2 occurs or when 0 is written to clear a flag. Bits other than the flag that is cleared should be set to 1.

### PORF flag (Power-On Reset Detect Flag)

The PORF flag indicates that a power-on reset occurred.

[Setting condition]

- When a power-on reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When PORF is read as 1 and then 0 is written to PORF

#### LVD0RF flag (Voltage Monitor 0 Reset Detect Flag)

The LVD0RF flag indicates that the VCC voltage fell below $V_{det0}$.

[Setting condition]

- When a voltage monitor 0 reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When LVD0RF is read as 1 and then 0 is written to LVD0RF.

#### LVD1RF flag (Voltage Monitor 1 Reset Detect Flag)

The LVD1RF flag indicates that the VCC voltage fell below $V_{det1}$.

[Setting condition]

- When a voltage monitor 1 reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When LVD1RF is read as 1 and then 0 is written to LVD1RF

#### LVD2RF flag (Voltage Monitor 2 Reset Detect Flag)

The LVD2RF flag indicates that the VCC voltage fell below $V_{det2}$.

[Setting condition]

- When a voltage monitor 2 reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When LVD2RF is read as 1 and then 0 is written to LVD2RF

#### DPSRSTF flag (Deep Software Standby Reset Detect Flag)

The DPSRSTF flag indicates that Deep Software Standby mode has been canceled by an external or internal interrupt and that an internal reset (Deep Software Standby reset) occurred when the exception from Deep Software Standby Mode occur.

[Setting condition]

- When Deep Software Standby mode is cancelled by an external or an internal interrupt. For details, see section 10, Low Power Modes.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs.
- When DPSRSTF is read as 1 and then 0 is written to DPSRSTF

### 5.2.3 RSTSR1 : Reset Status Register 1

Base address: SYSC = 0x4001_E000

Offset address: 0x0C0

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | TZERF | — | BUSMRF | — | REERF | RPERF | — | — | — | — | — | SWRF | WDTRF | IWDTRF |
| Value after reset: | 0 | 0 | x*1 | 0 | x*1 | 0 | x*1 | x*1 | 0 | 0 | 0 | 0 | 0 | x*1 | x*1 | x*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IWDTRF | Independent Watchdog Timer Reset Detect Flag<br>0: Independent watchdog timer reset not detected<br>1: Independent watchdog timer reset detected | R/W*2 |
| 1 | WDTRF | Watchdog Timer Reset Detect Flag<br>0: Watchdog timer reset not detected<br>1: Watchdog timer reset detected | R/W*2 |
| 2 | SWRF | Software Reset Detect Flag<br>0: Software reset not detected<br>1: Software reset detected | R/W*2 |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | RPERF | SRAM Parity Error Reset Detect Flag<br>0: SRAM parity error reset not detected<br>1: SRAM parity error reset detected | R/W*2 |
| 9 | REERF | SRAM ECC Error Reset Detect Flag<br>0: SRAM ECC error reset not detected<br>1: SRAM ECC error reset detected | R/W*2 |
| 10 | — | This bit is read as 0. The write value should be 0. | R/W |
| 11 | BUSMRF | Bus Master MPU Error Reset Detect Flag<br>0: Bus master MPU error reset not detected<br>1: Bus master MPU error reset detected | R/W*2 |
| 12 | — | This bit is read as 0. The write value should be 0. | R/W |
| 13 | TZERF | TrustZone Error Reset Detect Flag<br>0: TrustZone error reset not detected.<br>1: TrustZone error reset detected. | R/W*2 |
| 15:14 | — | This bit is read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. The value after reset depends on the reset source.
Note 2. Only 0 can be written to clear the flag. The flag must be cleared by writing 0 after 1 is read.

### IWDTRF flag (Independent Watchdog Timer Reset Detect Flag)

The IWDTRF flag indicates that an independent watchdog timer reset occurs.

[Setting condition]

- When an independent watchdog timer reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When 1 is read and then 0 is written to IWDTRF.

### WDTRF flag (Watchdog Timer Reset Detect Flag)

The WDTRF flag indicates that a watchdog timer reset occurs.

[Setting condition]

- When a watchdog timer reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When 1 is read and then 0 is written WDTRF.

### SWRF flag (Software Reset Detect Flag)

The SWRF flag indicates that a software reset occurs.

[Setting condition]

- When a software reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When 1 is read and then 0 is written to SWRF.

#### RPERF flag (SRAM Parity Error Reset Detect Flag)

The RPERF flag indicates that an SRAM parity error reset occurs.

[Setting condition]

- When an SRAM parity error reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When 1 is read as 1 and then 0 is written to RPERF.

#### REERF flag (SRAM ECC Error Reset Detect Flag)

The REERF flag indicates that an SRAM ECC error reset occurs.

[Setting condition]

- When an SRAM ECC error reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When 1 is read as 1 and then 0 is written to REERF.

#### BUSMRF flag (Bus Master MPU Error Reset Detect Flag)

The BUSMRF flag indicates that a bus master MPU error reset occurs.

[Setting condition]

- When a bus master MPU error reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When 1 is read and then 0 is written to BUSMRF.

#### TZERF flag (TrustZone Error Reset Detect Flag)

The TZERF flag indicates that a TrustZone error reset has occurred.

[Setting condition]

- When a TrustZone error reset occurs.

[Clearing conditions]

- When a reset listed in Table 5.2 occurs
- When 1 is read then and 0 is written to TZERF.

### 5.2.4 RSTSR2 : Reset Status Register 2

Base address: SYSC = 0x4001_E000

Offset address: 0x411

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | CWSF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CWSF | Cold/Warm Start Determination Flag<br>0: Cold start<br>1: Warm start | R/W*2 |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. The value after reset depends on the reset source.
Note 2. Only 1 can be written to set the flag.

RSTSR2 determines whether a power-on reset caused the reset processing (cold start) or a reset signal input during operation caused the reset processing (warm start).

#### CWSF flag (Cold/Warm Start Determination Flag)

The CWSF flag indicates the type of reset processing, either cold start or warm start. The determines whether a power-on reset caused the reset processing (cold start) or a reset signal input during operation caused the reset processing (warm start). CWSF flag is initialized by a power-on reset. It is not initialized by a reset signal generated by the RES pin.

[Setting condition]

- When 1 is written by software.

[Clearing condition]

- When a reset listed in Table 5.2 occurs.

## 5.3 Operation

### 5.3.1 RES Pin Reset

The RES pin generates this reset. When the RES pin is driven low, all the processing in progress is aborted and the MCU enters a reset state. To successfully reset the MCU, the RES pin must be held low for the power supply stabilization time specified at power-on.

When the RES pin is driven high from low, the internal reset is canceled after the post-RES cancellation wait time ($t_{RESWT}$) elapses. The CPU then starts the reset exception handling.

For details, see section 47, Electrical Characteristics.

### 5.3.2 Power-On Reset

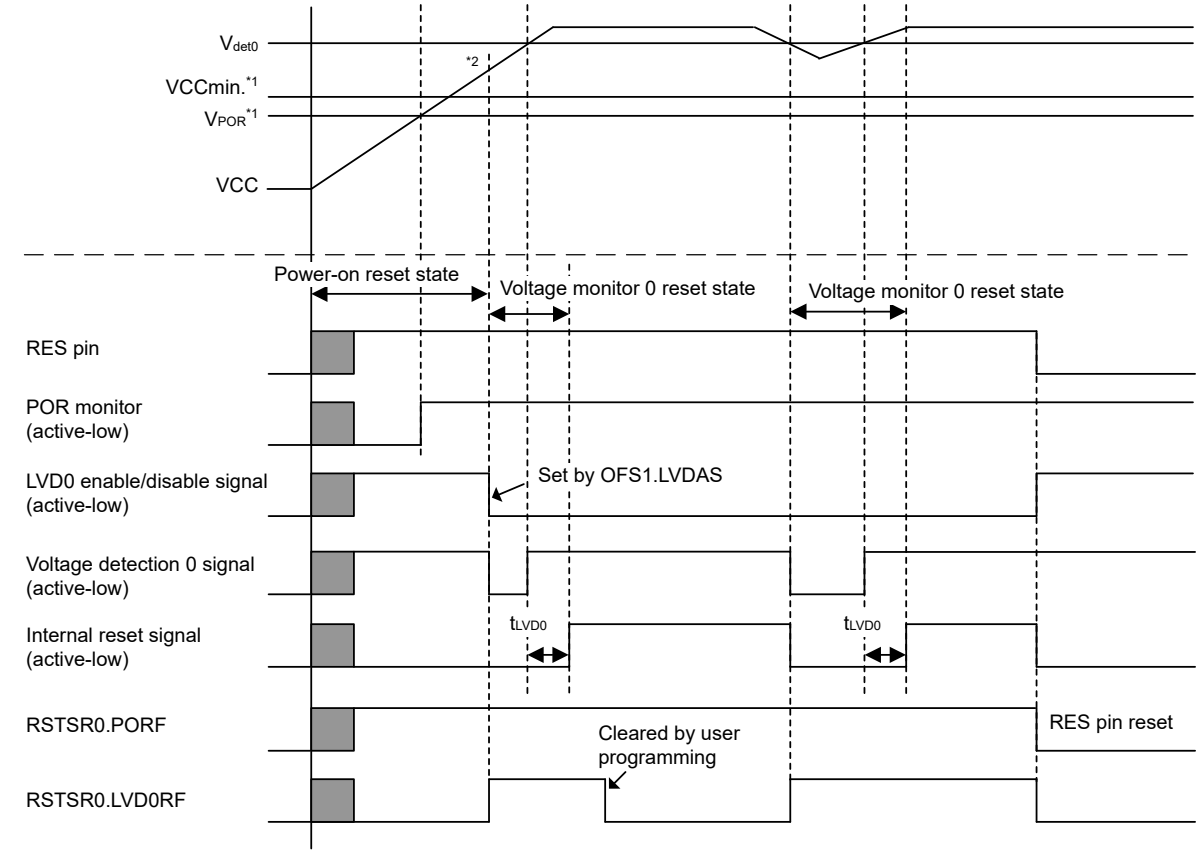
The power-on reset (POR) is an internal reset generated by the power-on reset circuit. A power-on reset is generated under the following conditions.

1. If the RES pin is in a high level state when power is supplied
2. If the RES pin is in a high level state when VCC is below $V_{POR}$

After VCC exceeds $V_{POR}$ and the specified power-on reset time ($t_{POR}$) elapses, the CPU starts the reset exception handling. The power-on reset time is a stabilization period of the external power supply and the MCU circuit.

After a power-on reset is generated, the PORF flag in the RSTSR0 is set to 1. The PORF flag is initialized by the RES pin reset. When VCC falls below $V_{POR}$, a power-on reset state is occurred.

Figure 5.1 shows example of operations during a power-on reset.




Note: For details on the electrical characteristics, see section 47, Electrical Characteristics.
Note 1. $V_{POR}$ shows a power-on reset detection level, and VCCmin shows the minimum guaranteed voltage of MCU.
Note 2. At power-on, VCC should rise to the minimum guaranteed voltage before the power-on reset is released.

**Figure 5.1 Example of operations during a power-on reset**

### 5.3.3 Voltage Monitor Reset

The voltage monitor i (i = 0, 1, 2) reset is an internal reset generated by the voltage monitor i circuit. If the Voltage Detection 0 Circuit Start (LVDAS) bit in the Option Function Select Register 1 (OFS1) is 0 (voltage monitor 0 reset is enabled after a reset) and VCC falls below $V_{det0}$, the RSTSR0.LVD0RF flag becomes 1 and the voltage detection circuit generates voltage monitor 0 reset. Clear the OFS1.LVDAS bit to 0 if the voltage monitor 0 reset is to be used. After VCC exceeds $V_{det0}$ and the voltage monitor 0 reset time ($t_{LVD0}$) elapses, the internal reset is canceled and the CPU starts the reset exception handling.

When the Voltage Monitor 1 Interrupt/Reset Enable bit (RIE) is set to 1 (enabling generation of a reset or interrupt by the voltage detection circuit) and the Voltage Monitor 1 Circuit Mode Select bit (RI) is set to 1 (selecting generation of a reset in response to detection of a low voltage) in Voltage Monitor 1 Circuit Control Register 0 (LVD1CR0), the RSTSR0.LVD1RF flag is set to 1 and the voltage detection circuit generates a voltage monitor 1 reset if VCC falls to or below $V_{det1}$.

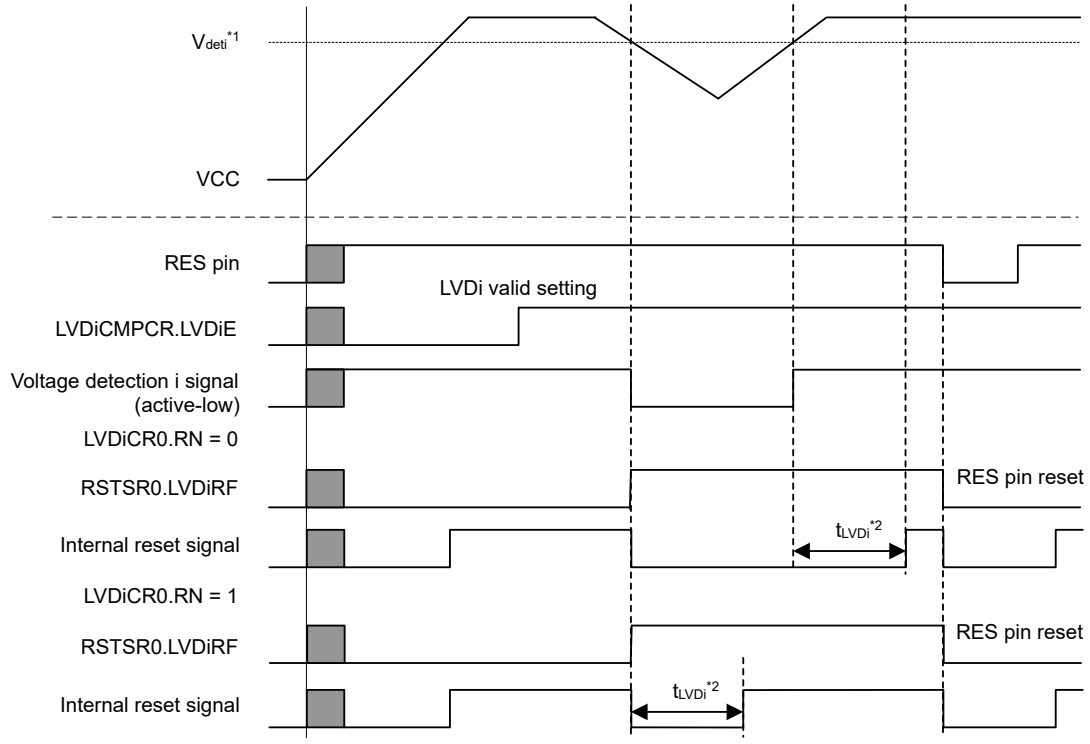
Likewise, when the Voltage Monitor 2 Interrupt/Reset Enable bit (RIE) is set to 1 (enabling generation of a reset or interrupt by the voltage detection circuit) and the Voltage Monitor 2 Circuit Mode Select bit (RI) is set to 1 (selecting generation of a reset in response to detection of a low voltage) in Voltage Monitor 2 Circuit Control Register 0 (LVD2CR0), the RSTSR0.LVD2RF flag is set to 1 and the voltage detection circuit generates a voltage monitor 2 reset if VCC falls to or below $V_{det2}$.

Similarly, timing for release from the voltage monitor 1 reset state is selectable with the Voltage Monitor 1 Reset Negate Select bit (RN) in the LVD1CR0. When the LVD1CR0.RN bit is 0 and VCC falls to or below $V_{det1}$, the CPU is released from the internal reset state and starts reset exception handling when the LVD1 reset time ($t_{LVD1}$) elapses after VCC rises above $V_{det1}$. When the LVD1CR0.RN bit is 1 and VCC falls to or below $V_{det1}$, the CPU is released from the internal reset state and starts reset exception handling when the LVD1 reset time ($t_{LVD1}$) elapses.

Likewise, timing for release from the voltage monitor 2 reset state is selectable by setting the Voltage Monitor 2 Reset Negate Select bit (RN) in the LDV2CR0 register.

Detection levels $V_{det1}$ and $V_{det2}$ can be changed in the Voltage Monitoring Comparator Control Register (LVD1CMPCR/ LVD2CMPCR).

Figure 5.2 shows example of operations during voltage monitor 1 and 2 resets. For details on the voltage monitor 1 reset and voltage monitor 2 reset, see section 7, Low Voltage Detection (LVD).




Note: For details on the electrical characteristics, see section 47, Electrical Characteristics.
Note 1. $V_{deti}$ indicates the detection level of a voltage monitor 1 reset and a voltage monitor 2 reset (i = 1, 2).
Note 2. $t_{LVDi}$ indicates the time for a voltage monitor 1 reset and a voltage monitor 2 reset (i = 1, 2).

**Figure 5.2 Example of operations during voltage monitor 1 and voltage monitor 2 resets**

### 5.3.4 Deep Software Standby Reset

This is an internal reset generated when Deep Software Standby mode is canceled by an interrupt.

When a Deep Software Standby mode cancelation source is generated, a Deep Software Standby reset is generated. The Deep Software Standby reset is canceled after $t_{DSBY}$ (return time after Deep Software Standby mode cancelation) has elapsed. At the same time, Deep Software Standby mode is also canceled.

When $t_{DSBYWT}$ (wait time after Deep Software Standby mode cancelation) has elapsed after Deep Software Standby mode has been canceled, the internal reset is canceled and the CPU starts the reset exception handling.

For details of the Deep Software Standby reset, see section 10, Low Power Modes.

### 5.3.5 Independent Watchdog Timer Reset

The independent watchdog timer reset is an internal reset generated from the Independent Watchdog Timer (IWDT). Output of the reset from the IWDT can be selected in the Option Function Select Register 0 (OFS0).

When output of the independent watchdog timer reset is selected, the reset is generated if the IWDT underflows, or if data is written when refresh operation is disabled. When the internal reset time ($t_{RESW2}$) elapses after the independent watchdog timer reset is generated, the internal reset is canceled and the CPU starts the reset exception handling.

For details on the independent watchdog timer reset, see section 25, Independent Watchdog Timer (IWDT).

### 5.3.6 Watchdog Timer Reset

The watchdog timer reset is an internal reset generated from the Watchdog Timer (WDT). Output of the reset from the WDT can be selected in the WDT Reset Control Register (WDTRCR) or Option Function Select register 0 (OFS0).

When output of the watchdog timer reset is selected, a watchdog timer reset is generated if the WDT underflows, or if data is written when refresh operation is disabled. When the internal reset time ($t_{RESW2}$) elapses after the watchdog timer reset is generated, the internal reset is canceled and the CPU starts the reset exception handling.

For details on the watchdog timer reset, see section 24, Watchdog Timer (WDT).

### 5.3.7 Software Reset

The software reset is an internal reset generated by a software setting of the SYSRESETREQ bit in the AIRCR register in the Arm core. When the SYSRESETREQ bit is set to 1, a software reset is generated. When the internal reset time ($t_{RESW2}$) elapses after the software reset is generated, the internal reset is canceled and the CPU starts the reset exception handling.

For details on the SYSRESETREQ bit, see the *ARM® Cortex®-M33 Technical Reference Manual*.

### 5.3.8 Determination of Cold/Warm Start

Read the CWSF flag in RSTSR2 to determine the cause of reset processing. This flag indicates whether a power-on reset caused the reset processing (cold start) or a reset signal input during operation caused the reset processing (warm start).

The CWSF flag is set to 0 when a power-on reset occurs (cold start), otherwise the flag is not set to 0. The flag is set to 1 when 1 is written to it through software. It is not set to 0 even on writing 0 to it.

Figure 5.3 shows an example of cold/warm start determination operation.

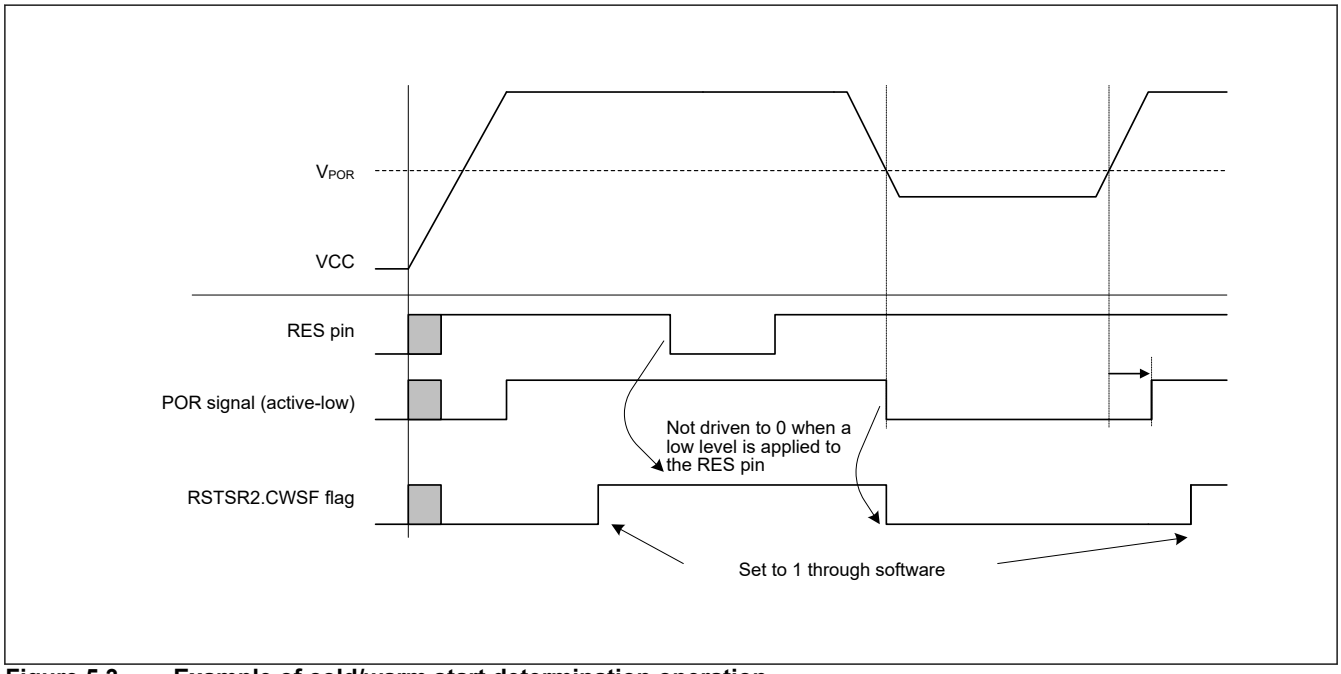
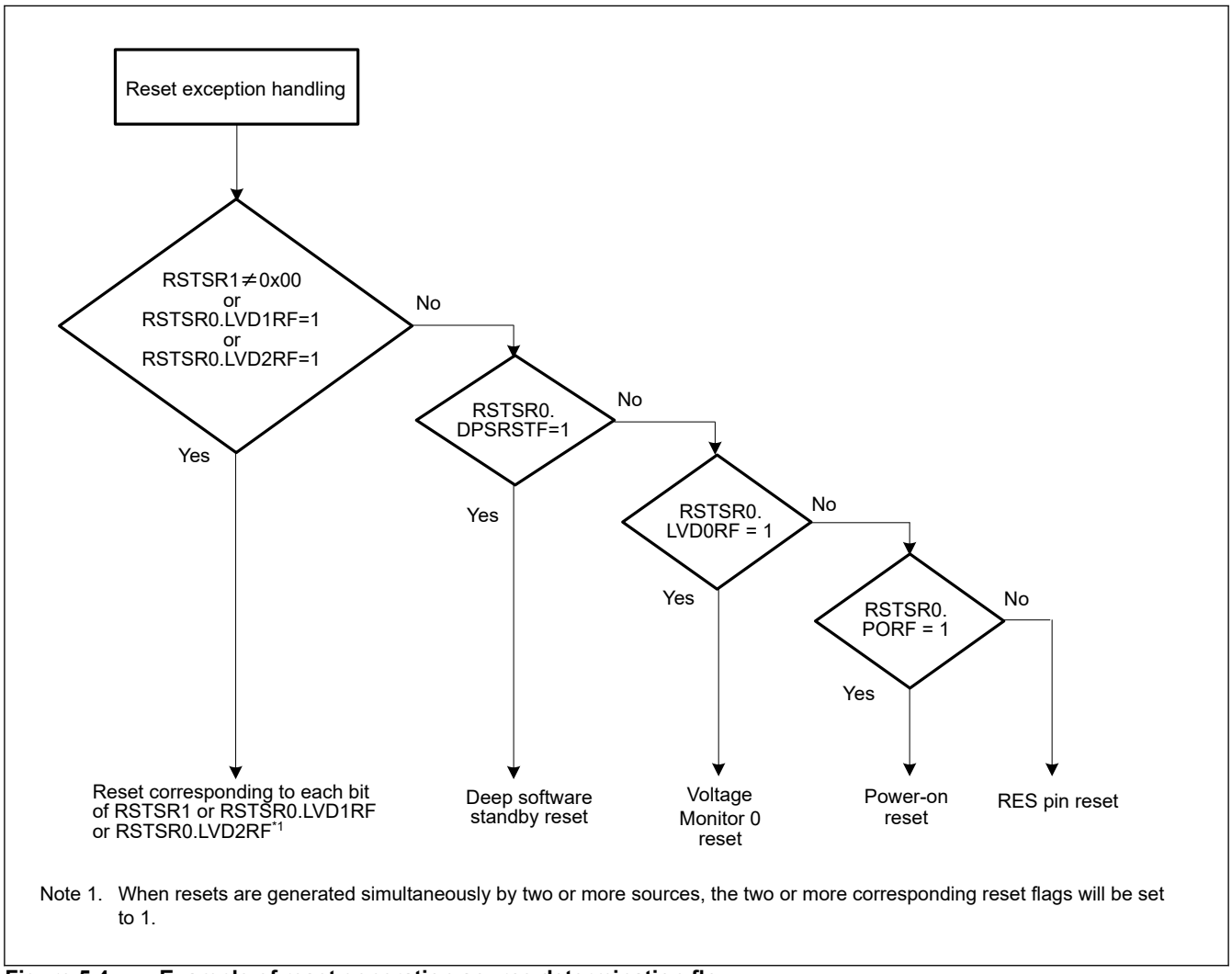



**Figure 5.3 Example of cold/warm start determination operation**

### 5.3.9 Determination of Reset Generation Source

Read RSTSR0 and RSTSR1 to determine which reset executes the reset exception handling.

Figure 5.4 shows an example of the flow to identify a reset generation source. The reset flag must be written with 0 after it is read as 1.




Note 1. When resets are generated simultaneously by two or more sources, the two or more corresponding reset flags will be set to 1.

**Figure 5.4 Example of reset generation source determination flow**

# 6. Option-Setting Memory

## 6.1 Overview

The option-setting memory determines the state of the MCU after a reset. The option-setting memory is allocated to the configuration setting area of the flash memory.

Figure 6.1 shows the option-setting memory area. The option-setting memory area has secure region. Table 6.1 shows the programming condition of the option-setting memory area.

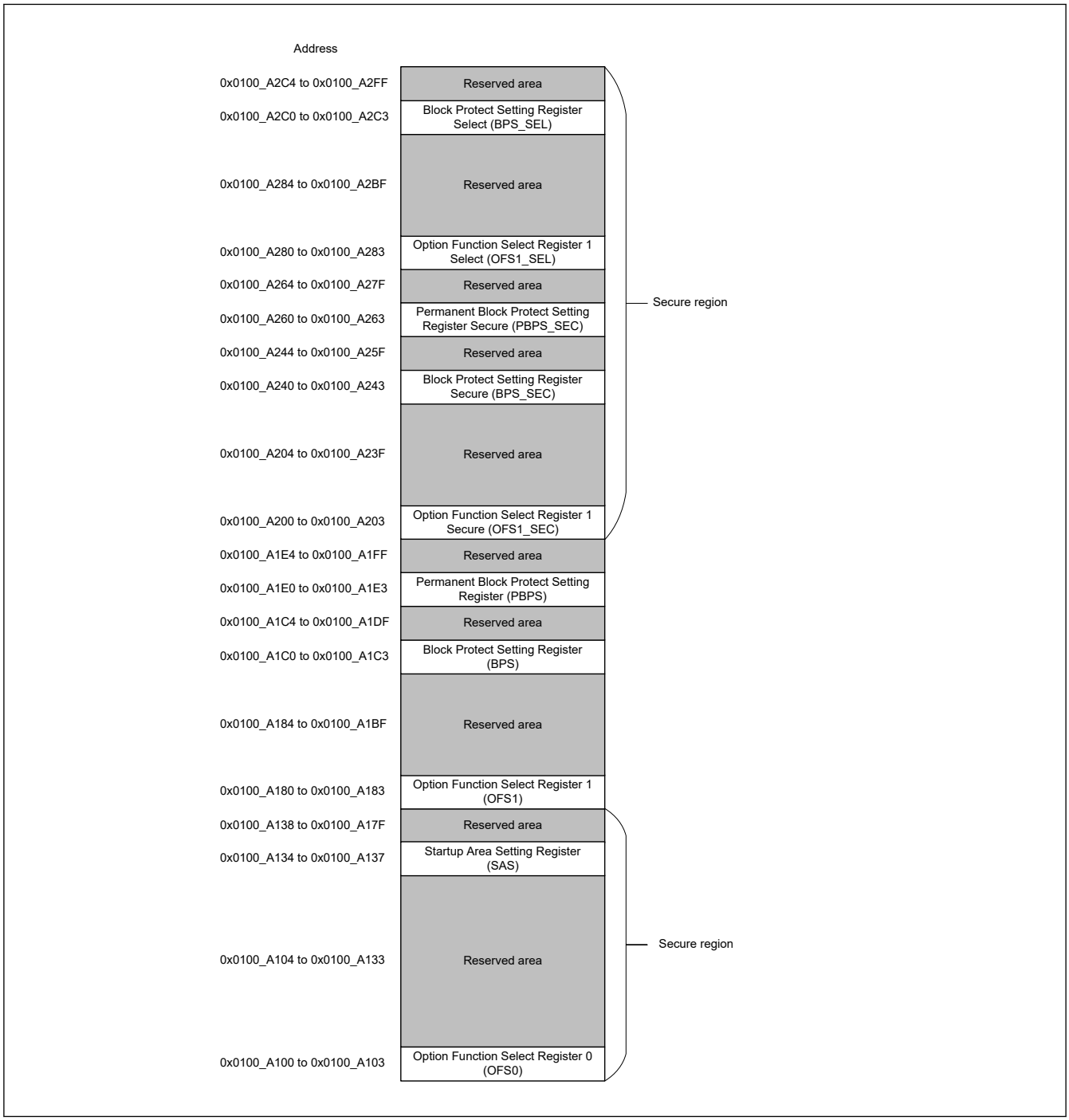



**Figure 6.1 Option-setting memory area**

**Table 6.1 The programming condition of the option-setting memory area**

| | Self programming | Serial programming | Programming by the on-chip debugger |
|---|---|---|---|
| Secure region | Programming commands issued by secure access. | Programming commands issued when the device life cycle is SSD. | Programming commands issued when the debug level is DBG2. |
| Other region | Programming commands issued by secure or non-secure access. | Programming commands issued when the devise life cycle is SSD or NSECSD. | Programming commands issued when the debug level is DBG2 or DBG1. |

## 6.2 Register Descriptions

### 6.2.1 OFS0 : Option Function Select Register 0

Address: 0x0100_A100

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | WDTSTPCTL | — | WDTRSTIRQS | WDTRPSS[1:0] | | WDTRPES[1:0] | | WDTCKS[3:0] | | | | WDTTOPS[1:0] | | WDTSTRT | — |
| Value after reset: | User setting*1 | | | | | | | | | | | | | | | |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | IWDTSTPCTL | — | IWDTRSTIRQS | IWDTRPSS[1:0] | | IWDTRPES[1:0] | | IWDTCKS[3:0] | | | | IWDTTOPS[1:0] | | IWDTSTRT | — |
| Value after reset: | User setting*1 | | | | | | | | | | | | | | | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | When read, this bit returns the written value. The write value should be 1. | R |
| 1 | IWDTSTRT | IWDT Start Mode Select<br>0: Automatically activate IWDT after a reset (auto start mode)<br>1: Disable IWDT after a reset | R |
| 3:2 | IWDTTOPS[1:0] | IWDT Timeout Period Select<br>0 0: 128 cycles (0x007F)<br>0 1: 512 cycles (0x01FF)<br>1 0: 1024 cycles (0x03FF)<br>1 1: 2048 cycles (0x07FF) | R |
| 7:4 | IWDTCKS[3:0] | IWDT-Dedicated Clock Frequency Division Ratio Select<br>0x0: × 1<br>0x2: × 1/16<br>0x3: × 1/32<br>0x4: × 1/64<br>0xF: × 1/128<br>0x5: × 1/256<br>Others: Setting prohibited | R |
| 9:8 | IWDTRPES[1:0] | IWDT Window End Position Select<br>0 0: 75%<br>0 1: 50%<br>1 0: 25%<br>1 1: 0% (no window end position setting) | R |
| 11:10 | IWDTRPSS[1:0] | IWDT Window Start Position Select<br>0 0: 25%<br>0 1: 50%<br>1 0: 75%<br>1 1: 100% (no window start position setting) | R |
| 12 | IWDTRSTIRQS | IWDT Reset Interrupt Request Select<br>0: Interrupt<br>1: Reset | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 13 | — | When read, this bit returns the written value. The write value should be 1. | R |
| 14 | IWDTSTPCTL | IWDT Stop Control<br>0: Continue counting<br>1: Stop counting when in Sleep, Snooze, or Software Standby mode | R |
| 16:15 | — | When read, these bits return the written value. The write value should be 1. | R |
| 17 | WDTSTRT | WDT Start Mode Select<br>0: Automatically activate WDT after a reset (auto start mode)<br>1: Stop WDT after a reset (register start mode) | R |
| 19:18 | WDTTOPS[1:0] | WDT Timeout Period Select<br>0 0: 1024 cycles (0x03FF)<br>0 1: 4096 cycles (0x0FFF)<br>1 0: 8192 cycles (0x1FFF)<br>1 1: 16384 cycles (0x3FFF) | R |
| 23:20 | WDTCKS[3:0] | WDT Clock Frequency Division Ratio Select<br>0x1: PCLKB divided by 4<br>0x4: PCLKB divided by 64<br>0xF: PCLKB divided by 128<br>0x6: PCLKB divided by 512<br>0x7: PCLKB divided by 2048<br>0x8: PCLKB divided by 8192<br>Others: Setting prohibited | R |
| 25:24 | WDTRPES[1:0] | WDT Window End Position Select<br>0 0: 75%<br>0 1: 50%<br>1 0: 25%<br>1 1: 0% (no window end position setting) | R |
| 27:26 | WDTRPSS[1:0] | WDT Window Start Position Select<br>0 0: 25%<br>0 1: 50%<br>1 0: 75%<br>1 1: 100% (no window start position setting) | R |
| 28 | WDTRSTIRQS | WDT Reset Interrupt Request Select<br>0: Interrupt<br>1: Reset | R |
| 29 | — | When read, these bits return the written value. The write value should be 1. | R |
| 30 | WDTSTPCTL | WDT Stop Control<br>0: Continue counting<br>1: Stop counting when entering Sleep mode | R |
| 31 | — | When read, these bits return the written value. The write value should be 1. | R |

Note 1. The value in a blank product is 0xFFFFFFFF. It is set to the value written by your application.

### IWDTSTRT bit (IWDT Start Mode Select)

The IWDTSTRT bit selects the mode in which the IWDT is activated after a reset (stopped state or activated state).

### IWDTTOPS[1:0] bits (IWDT Timeout Period Select)

The IWDTTOPS[1:0] bits specify the timeout period, that is, the time it takes for the down counter to underflow, as 128, 512, 1024, or 2048 cycles of the frequency-divided clock set in the IWDTCKS[3:0] bits. The time it takes for the counter to underflow after a refresh operation is determined by the combination of the IWDTCKS[3:0] and IWDTTOPS[1:0] bits.

For details, see section 25, Independent Watchdog Timer (IWDT).

### IWDTCKS[3:0] bits (IWDT-Dedicated Clock Frequency Division Ratio Select)

The IWDTCKS[3:0] bits specify the division ratio of the prescaler for dividing the frequency of the clock for the IWDT as 1/1, 1/16, 1/32, 1/64, 1/128, and 1/256. Using this setting combined with the IWDTTOPS[1:0] bits setting, the IWDT counting period can be set from 128 to 524288 IWDT clock cycles.

For details, see section 25, Independent Watchdog Timer (IWDT).

#### IWDTRPES[1:0] bits (IWDT Window End Position Select)

The IWDTRPES[1:0] bits specify the position where the window for the down counter ends as 0%, 25%, 50%, or 75% of the count value. The value of the window end position must be smaller than the value of the window start position, otherwise only the value for the window start position is valid.

The counter values associated with the settings for the start and end positions of the window in the IWDTRPSS[1:0] and IWDTRPES[1:0] bits vary with the setting in the IWDTTOPS[1:0] bits.

For details, see section 25, Independent Watchdog Timer (IWDT).

#### IWDTRPSS[1:0] bits (IWDT Window Start Position Select)

The IWDTRPSS[1:0] bits specify the position where the window for the down counter starts as 25%, 50%, 75%, or 100% of the counted value. The point at which counting starts is 100% and the point at which an underflow occurs is 0%. The interval between the window starts and ends positions becomes the period in which a refresh is possible. Refresh is not possible outside this period.

For details, see section 25, Independent Watchdog Timer (IWDT).

#### IWDTRSTIRQS bit (IWDT Reset Interrupt Request Select)

The IWDTRSTIRQS bit selects the operation on an underflow of the down counter or generation of a refresh error. The operation is selectable to an independent watchdog timer reset, a non-maskable interrupt request, or an interrupt request.

For details, see section 25, Independent Watchdog Timer (IWDT).

#### IWDTSTPCTL bit (IWDT Stop Control)

The IWDTSTPCTL bit specifies whether to stop counting when entering Sleep mode, Snooze mode, or Software Standby mode.

For details, see section 25, Independent Watchdog Timer (IWDT).

#### WDTSTRT bit (WDT Start Mode Select)

The WDTSTRT bit selects the mode in which the WDT is activated after a reset (stopped state or activated in auto start mode). When WDT is activated in auto start mode, the OFS0 register setting for the WDT is valid.

#### WDTTOPS[1:0] bits (WDT Timeout Period Select)

The WDTTOPS[1:0] bits specify the timeout period, that is, the time it takes for the down counter to underflow as 1024, 4096, 8192, or 16384 cycles of the frequency-divided clock set in the WDTCKS[3:0] bits. The number of PCLKB cycles that takes to underflow after a refresh operation is determined by a combination of the WDTCKS[3:0] and WDTTOPS[1:0] bits.

For details, see section 24, Watchdog Timer (WDT).

#### WDTCKS[3:0] bits (WDT Clock Frequency Division Ratio Select)

The WDTCKS[3:0] bits specify the division ratio of the prescaler for dividing the frequency of PCLKB as 1/4, 1/64, 1/128, 1/512, 1/2048, and 1/8192. Using this setting combined with the WDTTOPS[1:0] bits setting, the WDT counting period can be set from 4096 to 134217728 PCLKB cycles.

For details, see section 24, Watchdog Timer (WDT).

#### WDTRPES[1:0] bits (WDT Window End Position Select)

The WDTRPES[1:0] bits specify the position where the window on the down counter ends as 0%, 25%, 50%, or 75% of the counted value. The value of the window end position must be smaller than the value of the window start position, otherwise only the value for the window start position is valid.

The counter values associated with the settings for the start and end positions of the window in the WDTRPSS[1:0] and WDTRPES[1:0] bits vary with the setting of the WDTTOPS[1:0] bits.

For details, see section 24, Watchdog Timer (WDT).

### WDTRPSS[1:0] bits (WDT Window Start Position Select)

The WDTRPSS[1:0] bits specify the position where the window for the down counter starts as 25%, 50%, 75%, or 100% of the counted value. The point at which counting starts is 100% and the point at which an underflow occurs is 0%. The interval between the positions where the window starts and ends becomes the period in which a refresh is possible.

Refresh is not possible outside this period.

For details, see section 24, Watchdog Timer (WDT).

### WDTRSTIRQS bit (WDT Reset Interrupt Request Select)

The WDTRSTIRQS bit selects the operation on an underflow of the down-counter or generation of a refresh error. The operation is selectable to a watchdog timer reset, a non-maskable interrupt request, or an interrupt request.

For details, see section 24, Watchdog Timer (WDT).

### WDTSTPCTL bit (WDT Stop Control)

The WDTSTPCTL bit specifies whether to stop counting when entering Sleep mode.

For details, see section 24, Watchdog Timer (WDT).

## 6.2.2 SAS : Startup Area Setting Register

Address: 0x0100_A134

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BTFLG | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | User setting | | | | | | | | | | | | | | | |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | FSPR | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | User setting | | | | | | | | | | | | | | | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14:0 | — | When read, these bits return the written value. The write value should be 1. | R |
| 15 | FSPR | Protection of Startup Area Select Function<br>This bit controls the programming of the write/erase protection for the Startup Area Select flag (SAS.BTFLG), and the temporary boot swap control. When this bit is set to 0, it cannot be changed to 1.<br>0: Executing the configuration setting command for programming the Startup Area Select flag (SAS.BTFLG) is invalid.<br>1: Executing the configuration setting command for programming the Startup Area Select flag (SAS.BTFLG) is valid. | R |
| 30:16 | — | When read, these bits return the written value. The write value should be 1. | R |
| 31 | BTFLG | Startup Area Select Flag<br>This bit specifies whether the address of the startup area is exchanged for the boot swap function or not.<br>0: First 8-KB area (0x0000_0000 to 0x0000_1FFF) and second 8-KB area (0x0000_2000 to 0x0000_3FFF) are exchanged.<br>1: First 8-KB area (0x0000_0000 to 0x0000_1FFF) and second 8-KB area (0x0000_2000 to 0x0000_3FFF) are not exchanged. | R |

## 6.2.3 OFS1, OFS1_SEC, OFS1_SEL : Option Function Select Register 1

Address: OFS1: 0x0100_A180
OFS1_SEC: 0x0100_A200
OFS1_SEL: 0x0100_A280

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | The value set by the user*1 | | | | | | | | | | | | | | | |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | HOCOFRQ0[1:0] | | HOCOEN | — | — | — | — | — | LVDAS | VDSEL[1:0] | |
| Value after reset: | The value set by the user*1 | | | | | | | | | | | | | | | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | VDSEL[1:0] | Voltage Detection 0 Level Select<br>0 0: Setting prohibited<br>0 1: Select 2.94 V<br>1 0: Select 2.87 V<br>1 1: Select 2.80 V | R |
| 2 | LVDAS | Voltage Detection 0 Circuit Start<br>0: Enable voltage monitor 0 reset after a reset<br>1: Disable voltage monitor 0 reset after a reset | R |
| 7:3 | — | When read, these bits return the written value. The write value should be 1. | R |
| 8 | HOCOEN | HOCO Oscillation Enable<br>0: Enable HOCO oscillation after a reset<br>1: Disable HOCO oscillation after a reset | R |
| 10:9 | HOCOFRQ0[1:0] | HOCO Frequency Setting 0<br>0 0: 16 MHz<br>0 1: 18 MHz<br>1 0: 20 MHz<br>1 1: Setting prohibited | R |
| 31:11 | — | When read, these bits return the written value. The write value should be 1. | R |

Note 1. The value in a blank product is 0xFFFFFFFF. It is set to the value written by your application.

Only secure developer can program OFS1_SEC and OFS1_SEL registers. OFS1_SEC register is for secure developer, and OFS1 register is for non-secure developer. The applied setting value is determined by the setting value of the corresponding bit in OFS1_SEL register. For details, see section 6.3.3. Security attribution of option-setting memory.

### VDSEL[1:0] bits (Voltage Detection 0 Level Select)

The VDSEL[1:0] bits select the voltage detection level of the voltage detection 0 circuit.

### LVDAS bits (Voltage Detection 0 Circuit Start)

The LVDAS bit selects whether the voltage monitor 0 reset is enabled or disabled after a reset.

### HOCOEN bit (HOCO Oscillation Enable)

The HOCOEN bit selects whether the HOCO oscillation is enabled or disabled after a reset. Setting this bit to 0 allows the HOCO oscillation to start before the CPU starts operation, which reduces the wait time for oscillation stabilization.

Note: When the HOCOEN bit is set to 0, the system clock source is not switched to HOCO. The system clock source is only switched to HOCO by setting the Clock Source Select bits (SCKSCR.CKSEL[2:0]). To use the HOCO clock, you must set the HOCO Frequency Setting 0 bits (OFS1.HOCOFRQ0[1:0]) to an optimum value.

### HOCOFRQ0[1:0] bits (HOCO Frequency Setting 0)

The HOCOFRQ0[1:0] bits specify the HOCO frequency after a reset as 16, 18, or 20 MHz.

### 6.2.4 BPS, BPS_SEC, BPS_SEL : Block Protect Setting Register

address:

BPS: 0x0100_A1C0
BPS_SEC: 0x0100_A240
BPS_SEL: 0x0100_A2C0

| Bit position: | 31 | 0 |
|---|---|---|
| Bit field: | | |
| Value after reset: | User setting*1 | |

Note 1. The value in a blank product is 0xFFFFFFFF. It is set to the value written by your application.

Only secure developer can program BPS_SEC and BPS_SEL registers. BPS_SEC register is for secure developer, and BPS register is for non-secure developer. The applied setting value is determined by the setting value of the corresponding bit in BPS_SEL register. For details, see section 6.3.3. Security attribution of option-setting memory.

The BPS and BPS_SEC registers invalidate the programming and erasure to the code flash memory. When the bit of this register is set to 0, the programming and erasure to the corresponding block are invalid. Figure 6.2 shows the code flash block structure of each product. Figure 6.3 shows the relationship between the bit of register and the block number. Unused bits are reserved and should be set to 1.

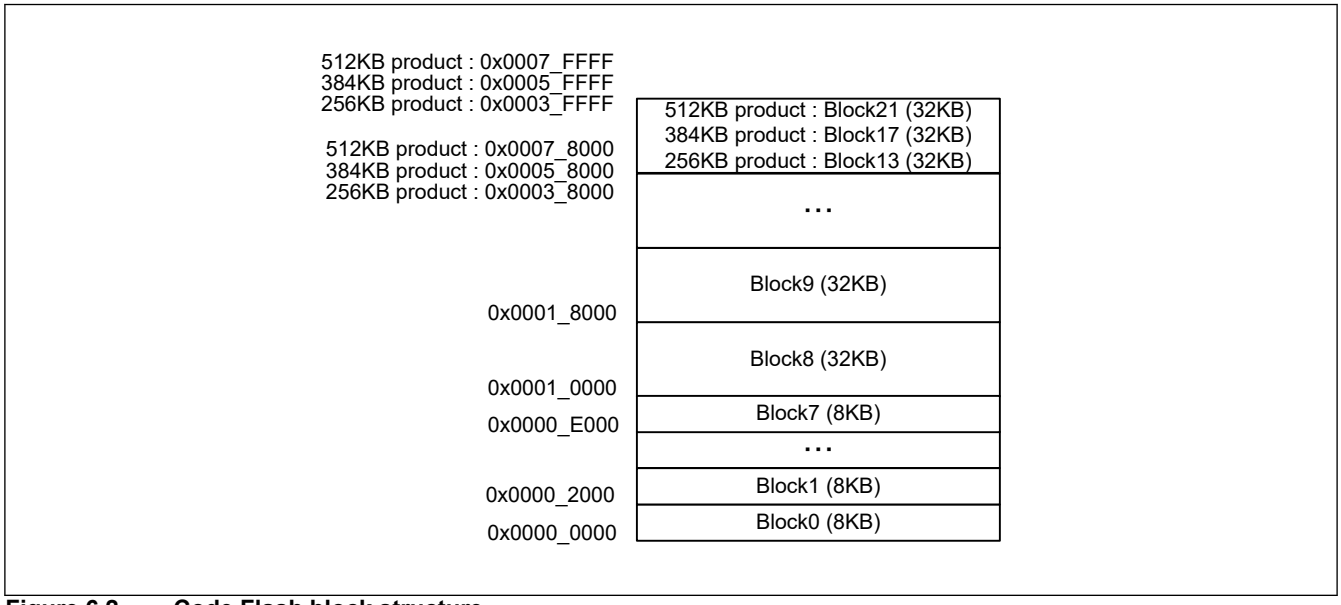



**Figure 6.2 Code Flash block structure**

| Regiser | Address | +31 | +30 | +29 | +28 | +27 | +26 | +25 | +24 | +23 | +22 | +21 | +20 | +19 | +18 | +17 | +16 | +15 | +14 | +13 | +12 | +11 | +10 | +9 | +8 | +7 | +6 | +5 | +4 | +3 | +2 | +1 | +0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BPS_SEL | 0x0100_A2C0 | | | | | | | | | | | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ⋮ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BPS_SEC | 0x0100_A240 | | | | | | | | | | | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ⋮ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BPS | 0x0100_A1C0 | | | | | | | | | | | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

**Figure 6.3 The relationship between the bit of register and the block number**

### 6.2.5 PBPS, PBPS_SEC : Permanent Block Protect Setting Register

Address: PBPS: 0x0100_A1E0
PBPS_SEC: 0x0100_A260

| Bit position: | 31 | 0 |
|---|---|---|
| Bit field: | | |
| Value after reset: | User setting*1 | |

Note 1. The value in a blank product is 0xFFFFFFFF. It is set to the value written by your application.

Only secure developer can program PBPS_SEC register. PBPS_SEC register is for secure developer, and PBPS register is for non-secure developer. The applied setting value is determined by the setting value of the corresponding bit in BPS_SEL register. For details, see section 6.3.3. Security attribution of option-setting memory. The security attribution register is same BPS_SEL register between the block protection and permanent block protection.

The PBPS and PBPS_SEC registers invalidate writes to bits of BPS and BPS_SEC. The bit of this register can be set to 0 when corresponding bit of BPS and BPS_SEC is set to 0. When the bit of this register is set to 0, writing the corresponding bit of BPS and BPS_SEC register is invalid. Once the bit of this register is set to 0, it is impossible to change the bit to 1. Table 6.2 shows the relationship between the bit of applied PBPS and bit of applied BPS.

The relationship between the bit of this register and the block number is same as BPS and BPS_SEC registers (section 6.2.4. BPS, BPS_SEC, BPS_SEL : Block Protect Setting Register). Unused bits are reserved and should be set to 1.

**Table 6.2 The relationship between the bit of PBPS, PBPS_SEC and bit of BPS, BPS_SEC**

| The bit of applied PBPS | The bit of applied BPS | Content |
|---|---|---|
| 1 | 1 | Programming and erasure to the corresponding block is valid. |
| 1 | 0 | Programming and erasure to the corresponding block is invalid. This protection can be canceled by FBPROT0 or FBPROT1 registers. |
| 0 | 1 | Cannot set this condition |
| 0 | 0 | Programming and erasure to the corresponding block is invalid permanently |

## 6.3 Setting Option-Setting Memory

### 6.3.1 Allocation of Data in Option-Setting Memory

Programming data is allocated to the addresses in the option-setting memory shown in Figure 6.1. The allocated data is used by tools such as a flash programming software or an on-chip debugger.

Note: Programming formats vary depending on the compiler. See the compiler manual for details.

### 6.3.2 Setting Data for Programming Option-Setting Memory

Allocating data according to the procedure described in section 6.3.1. Allocation of Data in Option-Setting Memory, alone does not actually write the data to the option-setting memory. You must also follow one of the actions described in this section.

#### (1) Changing the option-setting memory by self-programming

Use the configuration setting command to write data to the option-setting memory in the configuration setting area.

The option-setting memory does not support background operations (BGO). When write the option-setting memory, jump to SRAM after copying writing softwere to SRAM.

For details of the configuration setting command, see section 44, Flash Memory.

#### (2) Debugging through an OCD or programming by a flash writer

This procedure depends on the tool in use, see the tool manual for details.

The MCU provides two setting procedures:

- Read the data allocated as described in section 6.3.1. Allocation of Data in Option-Setting Memory, from an object file or Motorola S-format file generated by the compiler, and write the data to the MCU
- Use the GUI interface of the tool to program the same data as allocated in section 6.3.1. Allocation of Data in Option-Setting Memory.

### 6.3.3 Security attribution of option-setting memory

Some functionality has 3 registers for non-secure (FUNC NAME), and secure (FUNC NAME_SEC), and security attribution (FUNC NAME_SEL). Only secure developer can set the registers for secure and security attribution. As shown in Figure 6.4, when the bit of security attribution register is set to 0, the corresponding bit of secure register is applied. When the bit of security attribution register is set to 1, the corresponding bit of non-secure register is applied.

For example, if the secure developer wants to configure LVD of OFS1 as secure, HOCO of OFS1 as non-secure, the secure developer needs to set OFS1_SEL as follows.

OFS1_SEL = 0xFFFF_FFF8

By this setting, LVDAS and VDSEL[1:0] values of OFS1_SEC and HOCOFRQ0[1:0] and HOCOEN values of OFS1 are applied to MCU. The reserved bits of the security attribution register (FUNC NAME_SEL) should be set to 1.

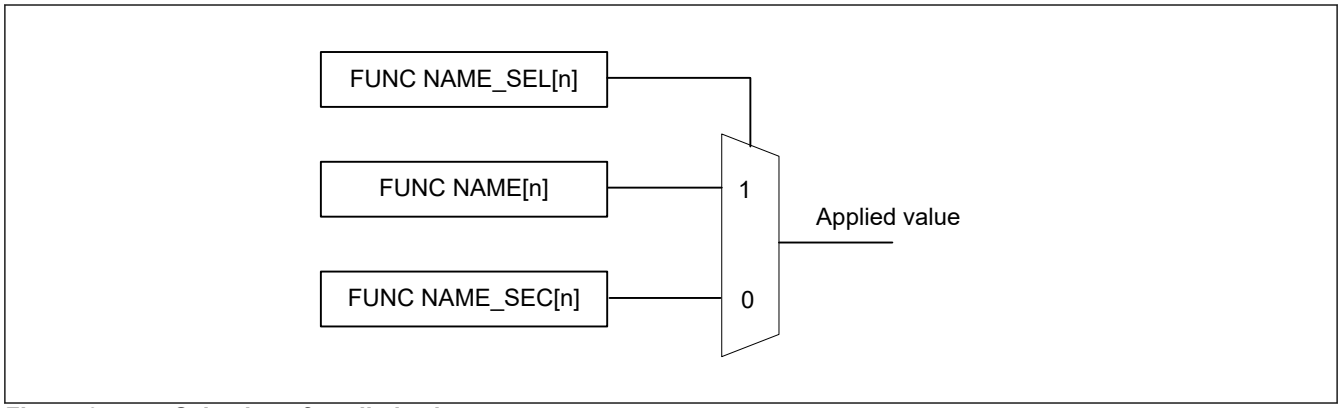



**Figure 6.4 Selection of applied value**

### 6.3.4 Timing of the Setting Value

For SAS, BPS, BPS_SEC, PBPS, and PBPS_SEC registers, the setting value of the related startup area and block protection is applied immediately after programming. For other registers, the setting value is applied after the MCU is reset.

In case the programming using the serial programming mode in customer’s factory, be careful that the block protection for secure user is applied after MCU is reset. Because initial value of the security attribution registers of block protection (BPS_SEL) is 1 (non-secure), the block protection setting for secure developer (BPS_SEC/PBPS_SEC) is not applied until MCU is reset even if the corresponding bit of BPS_SEL is programmed to 0 (secure).

## 6.4 Usage Notes

### 6.4.1 Data for Programming Reserved Areas and Reserved Bits in the Option-Setting Memory

When reserved areas and reserved bits in the option-setting memory are within the scope of programming, write 1 to all bits of reserved areas and all reserved bits. If 0 is written to these bits, normal operation cannot be guaranteed.

# 7. Low Voltage Detection (LVD)

## 7.1 Overview

The Low Voltage Detection (LVD) module monitors the voltage level input to the VCC pin. The detection level can be selected by register settings. The LVD module consists of three separate voltage level detectors (LVD0, LVD1, LVD2). LVD0, LVD1, and LVD2 measure the voltage level input to the VCC pin. LVD registers allow your application to configure detection of VCC changes at various voltage thresholds.

Voltage monitor registers are used to configure the LVD to trigger an interrupt, event link output, or reset when the thresholds are crossed.

Table 7.1 lists the LVD specifications. Figure 7.1 shows a block diagram of the voltage monitor 0 reset generation circuit. Figure 7.2 shows a block diagram of the voltage monitor 1 interrupt and reset circuit, and Figure 7.3 shows a block diagram of the voltage monitor 2 interrupt and reset circuit.

**Table 7.1 LVD specifications**

| Parameter | | Voltage monitor 0 | Voltage monitor 1 | Voltage monitor 2 |
|---|---|---|---|---|
| Means for setting up operation | | OFS1 register | Registers | Registers |
| Target for monitoring | | VCC pin input voltage | VCC pin input voltage | VCC pin input voltage |
| Monitored voltage | | $V_{det0}$ | $V_{det1}$ | $V_{det2}$ |
| Detected event | | Voltage falls past $V_{det0}$ | Voltage rises or falls past $V_{det1}$ | Voltage rises or falls past $V_{det2}$ |
| Detection voltage | | Selectable from 3 different levels in the OFS1.VDSEL[1:0] bits | Selectable from 3 different levels in the LVD1CMPCR.LVD1LVL[4:0] bits | Selectable from 3 different levels in the LVD2CMPCR.LVD2LVL[2:0] bits |
| Monitoring flag | | None | LVD1SR.MON flag: Monitors whether voltage is higher or lower than $V_{det1}$ | LVD2SR.MON flag: Monitors whether voltage is higher or lower than $V_{det2}$ |
| | | | LVD1SR.DET flag: $V_{det1}$ passage detection | LVD2SR.DET flag: $V_{det2}$ passage detection |
| Process on voltage detection | Reset | Voltage monitor 0 reset | Voltage monitor 1 reset | Voltage monitor 2 reset |
| | | Reset when $V_{det0}$ > VCC<br>CPU restart after specified time with VCC > $V_{det0}$ | Reset when $V_{det1}$ > VCC<br>CPU restart timing selectable:<br>after specified time with VCC > $V_{det1}$ or $V_{det1}$ > VCC | Reset when $V_{det2}$ > VCC<br>CPU restart timing selectable:<br>after specified time with either VCC > $V_{det2}$ or $V_{det2}$ > VCC |
| | Interrupt | No interrupt | Voltage monitor 1 interrupt | Voltage monitor 2 interrupt |
| | | | Non-maskable or maskable interrupt selectable | Non-maskable or maskable interrupt selectable |
| | | | Interrupt request issued when $V_{det1}$ > VCC and VCC > $V_{det1}$ or either | Interrupt request issued when $V_{det2}$ > VCC and VCC > $V_{det2}$ or either |
| Digital filter | Switching between enable and disable | No digital filter function | Available | Available |
| | Sampling time | — | 1/n LOCO frequency × 2 (n: 2, 4, 8, 16) | 1/n LOCO frequency × 2 (n: 2, 4, 8, 16) |
| Event link function | | None | Available<br>Output of event signals on detection of $V_{det1}$ crossings | Available<br>Output of event signals on detection of $V_{det2}$ crossings |
| TrustZone Filter | | — | Security attribution can be set for each registers | |

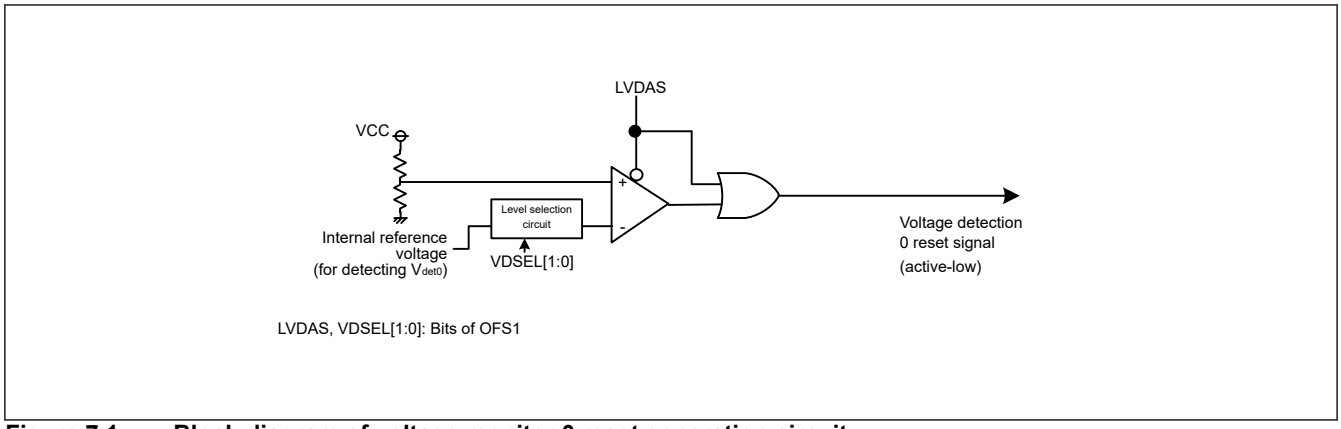



**Figure 7.1 Block diagram of voltage monitor 0 reset generation circuit**

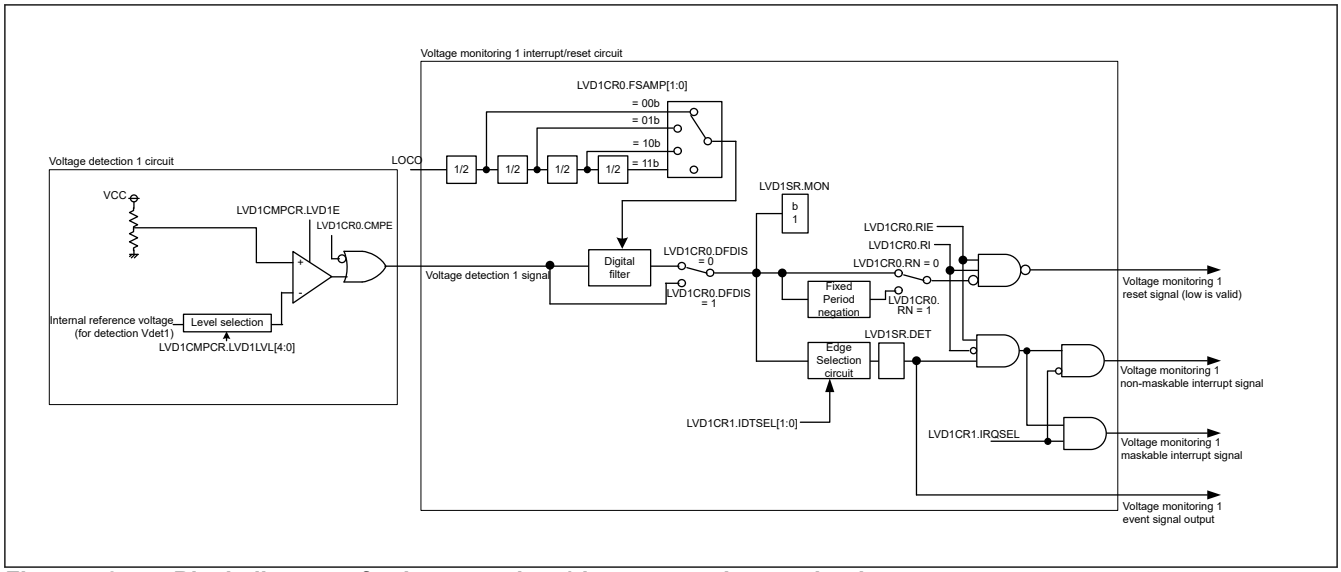



**Figure 7.2 Block diagram of voltage monitor 1 interrupt and reset circuit**

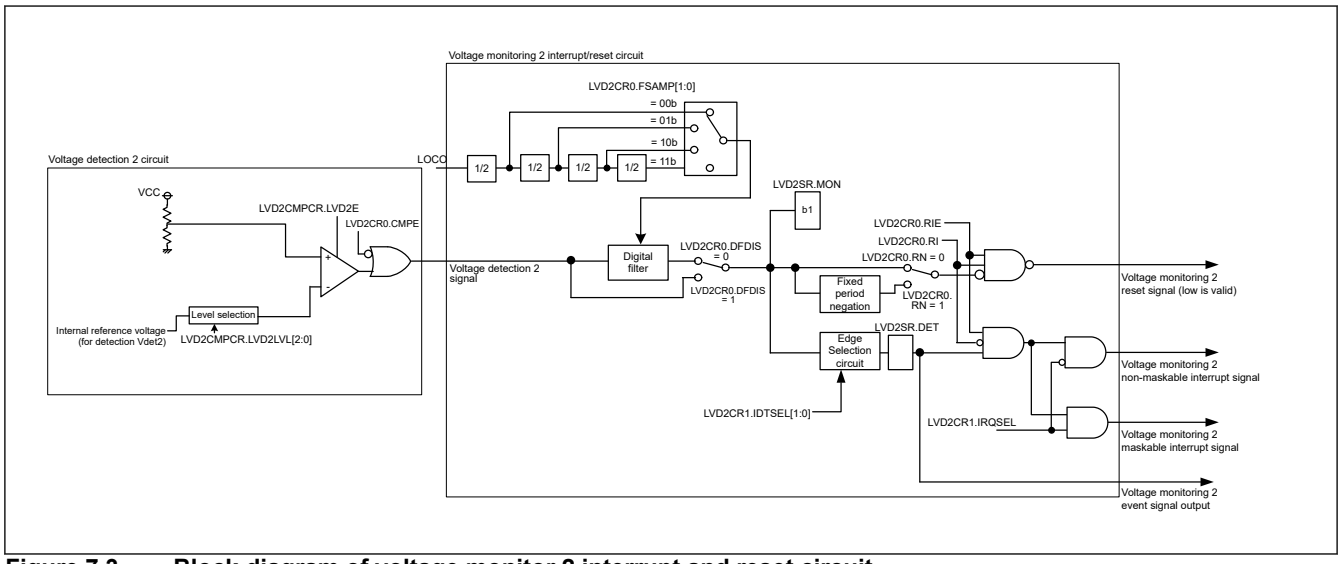



**Figure 7.3 Block diagram of voltage monitor 2 interrupt and reset circuit**

## 7.2 Register Descriptions

### 7.2.1 LVDSAR : Low Voltage Detection Security Attribution Register

Base address: SYSC = 0x4001_E000

Offset address: 0x3CC

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | NONSEC1 | NONSEC0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | NONSEC0 | Non Secure Attribute bit 0<br>Target register: registers for LVD1<br>0: Secure<br>1: Non Secure | R/W |
| 1 | NONSEC1 | Non Secure Attribute bit 1<br>Target register: registers for LVD2<br>0: Secure<br>1: Non Secure | R/W |
| 31:2 | — | These bits are read as 1. The write value must be 1 when it is possible to write. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

The LVDSAR register controls the secure attribute of LVD registers.

**NONSEC0 bit (Non Secure Attribute bit 0)**

This bit controls the security attribute of LVD1CMPCR, LVD1CR0, LVD1CR1, LVD1SR.

**NONSEC1 bit (Non Secure Attribute bit 1)**

This bit controls the security attribute of LVD2CMPCR, LVD2CR0, LVD2CR1, LVD2SR.

### 7.2.2 LVD1CMPCR : Voltage Monitoring 1 Comparator Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x417

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | LVD1E | — | — | LVD1LVL[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | LVD1LVL[4:0] | Voltage Detection 1 Level Select (Standard voltage during drop in voltage)<br>0x11: 2.99 V (Vdet1_1)<br>0x12: 2.92 V (Vdet1_2)<br>0x13: 2.85 V (Vdet1_3)<br>Others: Setting prohibited | R/W |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | LVD1E | Voltage Detection 1 Enable<br>0: Voltage detection 1 circuit disabled<br>1: Voltage detection 1 circuit enabled | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (write enabled) before rewriting this register.

The LVD1CMPCR.LVD1LVL can be changed only if the LVD1CMPCR.LVD1E and LVD2CMPCR.LVD2E bits are both 0. The voltage detection circuits 1 and 2 should not be set at the same voltage detection level.

Do not change LVD1CMPCR.LVD1LVL and LVD1CMPCR.LVD1E at the same time.

#### LVD1E bit (Voltage Detection 1 Enable)

When using voltage detection 1 interrupt/reset or the LVD1SR.MON bit, set the LVD1E bit to 1. The voltage detection 1 circuit starts once $t_{d(E-A)}$ passes after the LVD1E bit value is changed from 0 to 1. When using the voltage detection 1 circuit in Deep Software Standby mode, do not set the DPSBYCR.DEEPCUT[1:0] bits to 11b.

### 7.2.3 LVD2CMPCR : Voltage Monitoring 2 Comparator Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x418

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | LVD2E | — | — | — | — | LVD2LVL[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | LVD2LVL[2:0] | Voltage Detection 2 Level Select (Standard voltage during drop in voltage)<br>1 0 1: 2.99 V (Vdet2_1)<br>1 1 0: 2.92 V (Vdet2_2)<br>1 1 1: 2.85 V (Vdet2_3)<br>Others: Setting prohibited | R/W |
| 6:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | LVD2E | Voltage Detection 2 Enable<br>0: Voltage detection 2 circuit disabled<br>1: Voltage detection 2 circuit enabled | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (write enabled) before rewriting this register.

The LVD2CMPCR.LVD2LVL can be changed only if the LVD1CMPCR.LVD1E and LVD2CMPCR.LVD2E bits are both 0. The voltage detection circuits 1 and 2 should not be set at the same voltage detection level.

Do not change LVD2CMPCR.LVD2LVL and LVD2CMPCR.LVD2E at the same time.

#### LVD2E bit (Voltage Detection 2 Enable)

When using voltage detection 2 interrupt/reset or the LVD2SR.MON bit, set the LVD2E bit to 1. The voltage detection 2 circuit starts once $t_{d(E-A)}$ passes after the LVD2E bit value is changed from 0 to 1. When using the voltage detection 2 circuit in Deep Software Standby mode, do not set the DPSBYCR.DEEPCUT[1:0] bits to 11b.

## 7.2.4 LVD1CR0 : Voltage Monitor 1 Circuit Control Register 0

Base address: SYSC = 0x4001_E000

Offset address: 0x41A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | RN | RI | FSAMP[1:0] | | — | CMPE | DFDIS | RIE |
| Value after reset: | 1 | 0 | 0 | 0 | x | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RIE | Voltage Monitor 1 Interrupt/Reset Enable<br>0: Disable<br>1: Enable | R/W |
| 1 | DFDIS | Voltage monitor 1 Digital Filter Disabled Mode Select<br>0: Enable the digital filter<br>1: Disable the digital filter | R/W |
| 2 | CMPE | Voltage Monitor 1 Circuit Comparison Result Output Enable<br>0: Disable voltage monitor 1 circuit comparison result output<br>1: Enable voltage monitor 1 circuit comparison result output | R/W |
| 3 | — | The read value is undefined. The write value should be 1. | R/W |
| 5:4 | FSAMP[1:0] | Sampling Clock Select<br>0 0: 1/2 LOCO frequency<br>0 1: 1/4 LOCO frequency<br>1 0: 1/8 LOCO frequency<br>1 1: 1/16 LOCO frequency | R/W |
| 6 | RI | Voltage Monitor 1 Circuit Mode Select<br>0: Generate voltage monitor 1 interrupt on $V_{det1}$ crossing<br>1: Enable voltage monitor 1 reset when the voltage falls to and below $V_{det1}$ | R/W |
| 7 | RN | Voltage Monitor 1 Reset Negate Select<br>0: Negate after a stabilization time ($t_{LVD1}$) when VCC > $V_{det1}$ is detected<br>1: Negate after a stabilization time ($t_{LVD1}$) on assertion of the LVD1 reset | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (write enabled) before rewriting this register.

### RIE bit (Voltage Monitor 1 Interrupt/Reset Enable)

The RIE bit enables or disables voltage monitor 1 interrupt/reset. Ensure that neither a voltage monitor 1 interrupt nor a voltage monitor 1 reset is generated during programming or erasure of the flash memory.

### DFDIS bit (Voltage monitor 1 Digital Filter Disabled Mode Select)

The DFDIS bit disables the digital filter circuit. Set the LOCOCR.LCSTP bit to 0 (the LOCO operates) when this bit is 0(enabled). Set this bit to 1 (disabled) when using the voltage monitor 1 circuit in Software Standby mode or in Deep Software Standby mode.

### CMPE bit (Voltage Monitor 1 Circuit Comparison Result Output Enable)

The CMPE bit enables or disables voltage monitor 1 circuit comparison result output. Set the CMPE bit to 1 after the voltage detection 1 circuit enables and stabilization time ($t_{d(E-A)}$) elapses. When stopping the voltage detection 1 circuit, disable the voltage detection 1 circuit after setting the CMPE bit is 0.

### FSAMP[1:0] bits (Sampling Clock Select)

The FSAMP[1:0] bits can be rewritten only when the LVD1CR0.DFDIS bit is 1 (digital filter circuit disabled). Do not rewrite these bits if the LVD1CR0.DFDIS bit is 0 (digital filter circuit enabled).

### RI bit (Voltage Monitor 1 Circuit Mode Select)

When the RI bit is 1 (voltage monitor 1 reset selected), transition to Deep Software Standby mode cannot be made. In this case, transition to Software Standby mode is made. To enter Deep Software Standby mode, set the RI bit to 0 (voltage monitor 1 interrupt selected).

### RN bit (Voltage Monitor 1 Reset Negate Select)

If the RN bit is set to 1 (negation follows a stabilization time on assertion of the LVD1 reset signal), set the LOCOCR.LCSTP bit to 0 (the LOCO operates). In addition, for a transition to Software Standby or Deep Software Standby mode, the only possible value for the RN bit is 0 (negation follows stabilization time when VCC > $V_{det1}$ is detected). Do not set the RN bit to 1 when this is the case.

## 7.2.5 LVD2CR0 : Voltage Monitor 2 Circuit Control Register 0

Base address: SYSC = 0x4001_E000

Offset address: 0x41B

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | RN | RI | FSAMP[1:0] | | — | CMPE | DFDIS | RIE |
| Value after reset: | 1 | 0 | 0 | 0 | x | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RIE | Voltage Monitor 2 Interrupt/Reset Enable<br>0: Disable<br>1: Enable | R/W |
| 1 | DFDIS | Voltage monitor 2 Digital Filter Disabled Mode Select<br>0: Enable the digital filter<br>1: Disable the digital filter | R/W |
| 2 | CMPE | Voltage Monitor 2 Circuit Comparison Result Output Enable<br>0: Disable voltage monitor 2 circuit comparison result output<br>1: Enable voltage monitor 2 circuit comparison result output | R/W |
| 3 | — | The read value is undefined. The write value should be 1. | R/W |
| 5:4 | FSAMP[1:0] | Sampling Clock Select<br>0 0: 1/2 LOCO frequency<br>0 1: 1/4 LOCO frequency<br>1 0: 1/8 LOCO frequency<br>1 1: 1/16 LOCO frequency | R/W |
| 6 | RI | Voltage Monitor 2 Circuit Mode Select<br>0: Generate voltage monitor 2 interrupt on $V_{det2}$ crossing<br>1: Enable voltage monitor 2 reset when the voltage falls to and below $V_{det2}$ | R/W |
| 7 | RN | Voltage Monitor 2 Reset Negate Select<br>0: Negate after a stabilization time ($t_{LVD2}$) when VCC > $V_{det2}$ is detected<br>1: Negate after a stabilization time ($t_{LVD2}$) on assertion of the LVD2 reset | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (write enabled) before rewriting this register.

### RIE bit (Voltage Monitor 2 Interrupt/Reset Enable)

The RIE bit enables or disables the voltage monitor 2 interrupt/reset. Ensure that neither a voltage monitor 2 interrupt nor a voltage monitor 2 reset is generated during programming or erasure of the flash memory.

### DFDIS bit (Voltage monitor 2 Digital Filter Disabled Mode Select)

The DFDIS bit disables the digital filter circuit. Set the LOCOCR.LCSTP bit to 0 (the LOCO operates) when this bit is 0 (digital filter enabled). Set this bit to 1 (digital filter disabled) when using the voltage monitor 2 circuit in Software Standby mode or in Deep Software Standby mode.

### CMPE bit (Voltage Monitor 2 Circuit Comparison Result Output Enable)

The CMPE bit enables or disables voltage monitor 2 circuit comparison result output. Set the CMPE bit to 1 after the voltage detection 2 circuit enables and stabilization time ($t_{d(E-A)}$) elapses. When stopping the voltage detection 2 circuit, disable the voltage detection 2 circuit after setting the CMPE bit is 0.

### FSAMP[1:0] bits (Sampling Clock Select)

The FSAMP[1:0] bits can be rewritten only when the LVD2CR0.DFDIS bit is 1 (digital filter circuit disabled). Do not rewrite these bits if the LVD2CR0.DFDIS bit is 0 (digital filter circuit enabled).

### RI bit (Voltage Monitor 2 Circuit Mode Select)

When the RI bit is 1 (voltage monitor 2 reset selected), transition to Deep Software Standby mode cannot be made. In this case, transition to Software Standby mode is made. To enter Deep Software Standby mode, set the RI bit to 0 (voltage monitor 2 interrupt selected).

### RN bit (Voltage Monitor 2 Reset Negate Select)

If the RN bit is set to 1 (negating LVD2 reset in a specified time after its assertion), set the LOCOCR.LCSTP bit to 0 (the LOCO operates). Additionally, for a transition to Software Standby or Deep Software Standby mode, the only possible value for the RN bit is 0 (negation follows a stabilization time when VCC > $V_{det2}$ is detected). Do not set the RN bit to 1 (negation follows a stabilization time after assertion of the LVD2 reset signal) when this is the case.

## 7.2.6 LVD1CR1 : Voltage Monitor 1 Circuit Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x0E0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | IRQSEL | IDTSEL[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | IDTSEL[1:0] | Voltage Monitor 1 Interrupt Generation Condition Select<br>0 0: When VCC ≥ $V_{det1}$ (rise) is detected<br>0 1: When VCC < $V_{det1}$ (fall) is detected<br>1 0: When fall and rise are detected<br>1 1: Settings prohibited | R/W |
| 2 | IRQSEL | Voltage Monitor 1 Interrupt Type Select<br>0: Non-maskable interrupt<br>1: Maskable interrupt*1 | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (writing enabled) before rewriting this register.

Note 1. When enabling maskable interrupts, do not change the NMIER.LVD1EN bit value in the ICU from the reset state.

## 7.2.7 LVD1SR : Voltage Monitor 1 Circuit Status Register

Base address: SYSC = 0x4001_E000

Offset address: 0x0E1

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | MON | DET |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DET | Voltage Monitor 1 Voltage Variation Detection Flag<br>0: Not detected<br>1: $V_{det1}$ crossing is detected | R/W*1 |
| 1 | MON | Voltage Monitor 1 Signal Monitor Flag<br>0: VCC < $V_{det1}$<br>1: VCC ≥ $V_{det1}$ or MON is disabled | R |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (write enabled) before rewriting this register.

Note 1. Only 0 can be written to this bit. After writing 0 to this bit, 2 system clock cycles are required for the bit to be read as 0.

### DET flag (Voltage Monitor 1 Voltage Variation Detection Flag)

The DET flag is enabled when the LVD1CMPCR.LVD1E bit is 1 (voltage detection 1 circuit enabled) and the LVD1CR0.CMPE bit is 1 (voltage monitor 1 circuit comparison result output enabled).

When detecting $V_{det1}$, set the DET flag to 0 after setting LVD1CR0.RIE is 0 (disabled). When setting LVD1CR0.RIE bit to 1 (enabled) after setting it to 0, wait for 2 or more PCLKB cycles which have elapsed.

### MON flag (Voltage Monitor 1 Signal Monitor Flag)

The MON flag is enabled when the LVD1CMPCR.LVD1E bit is 1 (voltage detection 1 circuit enabled) and the LVD1CR0.CMPE bit is 1 (voltage monitor 1 circuit comparison result output enabled).

## 7.2.8 LVD2CR1 : Voltage Monitor 2 Circuit Control Register 1

Base address: SYSC = 0x4001_E000

Offset address: 0x0E2

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | IRQSEL | IDTSEL[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | IDTSEL[1:0] | Voltage Monitor 2 Interrupt Generation Condition Select<br>0 0: When VCC ≥ $V_{det2}$ (rise) is detected<br>0 1: When VCC < $V_{det2}$ (fall) is detected<br>1 0: When fall and rise are detected<br>1 1: Settings prohibited | R/W |
| 2 | IRQSEL | Voltage Monitor 2 Interrupt Type Select<br>0: Non-maskable interrupt<br>1: Maskable interrupt*1 | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (writing enabled) before rewriting this register.
Note 1. When enabling maskable interrupts, do not change the NMIER.LVD2EN bit value in the ICU from the reset state.

### 7.2.9 LVD2SR : Voltage Monitor 2 Circuit Status Register

Base address: SYSC = 0x4001_E000

Offset address: 0x0E3

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | MON | DET |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DET | Voltage Monitor 2 Voltage Variation Detection Flag<br>0: Not detected<br>1: $V_{det2}$ crossing is detected | R/W*1 |
| 1 | MON | Voltage Monitor 2 Signal Monitor Flag<br>0: VCC < $V_{det2}$<br>1: VCC ≥ $V_{det2}$ or MON is disabled | R |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (write enabled) before rewriting this register.
Note 1. Only 0 can be written to this bit. After writing 0 to this bit, 2 system clock cycles are required for the bit to be read as 0.

#### DET flag (Voltage Monitor 2 Voltage Variation Detection Flag)

The DET flag is enabled when the LVD2CMPCR.LVD2E bit is 1 (voltage detection 2 circuit enabled) and the LVD2CR0.CMPE bit is 1 (voltage monitor 2 circuit comparison result output enabled).

When detecting $V_{det2}$, set the DET flag to 0 after setting LVD2CR0.RIE is 0 (disabled). When setting LVD2CR0.RIE bit to 1 (enabled) after setting it to 0, wait for 2 or more PCLKB cycles which have elapsed.

#### MON flag (Voltage Monitor 2 Signal Monitor Flag)

The MON flag is enabled when the LVD2CMPCR.LVD2E bit is 1 (voltage detection 2 circuit enabled) and the LVD2CR0.CMPE bit is 1 (voltage monitor 2 circuit comparison result output enabled).

## 7.3 VCC Input Voltage Monitor

### 7.3.1 Monitoring $V_{det0}$

The comparison results from voltage monitor 0 are not available for reading.

### 7.3.2 Monitoring $V_{det1}$

Table 7.2 shows the procedures to set up monitoring against $V_{det1}$. After the settings are complete, the comparison results from voltage monitor 1 can be monitored with the LVD1SR.MON flag.

**Table 7.2 Procedures to set up monitoring against $V_{det1}$**

| Step | | Monitoring the comparison results from voltage monitor 1 |
|---|---|---|
| Setting up the voltage detection 1 circuit | 1 | Set LVD1CMPCR.LVD1E = 0 to disable voltage detection 1 before writing to the LVD1CMPCR.LVD1LVL[4:0] bits. |
| | 2 | Select the detection voltage in the LVD1CMPCR.LVD1LVL[4:0] bits. |
| | 3 | Set LVD1CMPCR.LVD1E = 1 to enable the voltage detection 1 circuit. |
| | 4 | Wait for at least $t_{d\ (E-A)}$ for the LVD1 operation stabilization time after LVD1 is enabled.*1 |
| Setting the digital filter*2 | 5 | Select the sampling clock for the digital filter in the LVD1CR0.FSAMP[1:0] bits. |
| | 6 | Set LVD1CR0.DFDIS = 0 to enable the digital filter. |
| | 7 | Wait for at least 2n + 3 cycles of the LOCO, where n = 2, 4, 8, or 16, and the sampling clock for the digital filter is the LOCO frequency-divided by n. |
| Enabling output | 8 | Set LVD1CR0.CMPE = 1 to enable output of the comparison results from voltage monitor 1. |

Note 1. Steps 5 to 7 can be performed during the wait time of step 4. For details of $t_{d(E-A)}$, see section 47, Electrical Characteristics.
Note 2. Steps 5 to 7 are not required if the digital filter is not in use.

### 7.3.3 Monitoring $V_{det2}$

Table 7.3 shows the procedures to set up monitoring against $V_{det2}$. After the settings are complete, the comparison results from voltage monitor 2 can be monitored in the LVD2SR.MON flag.

**Table 7.3 Procedures to set up monitoring against $V_{det2}$**

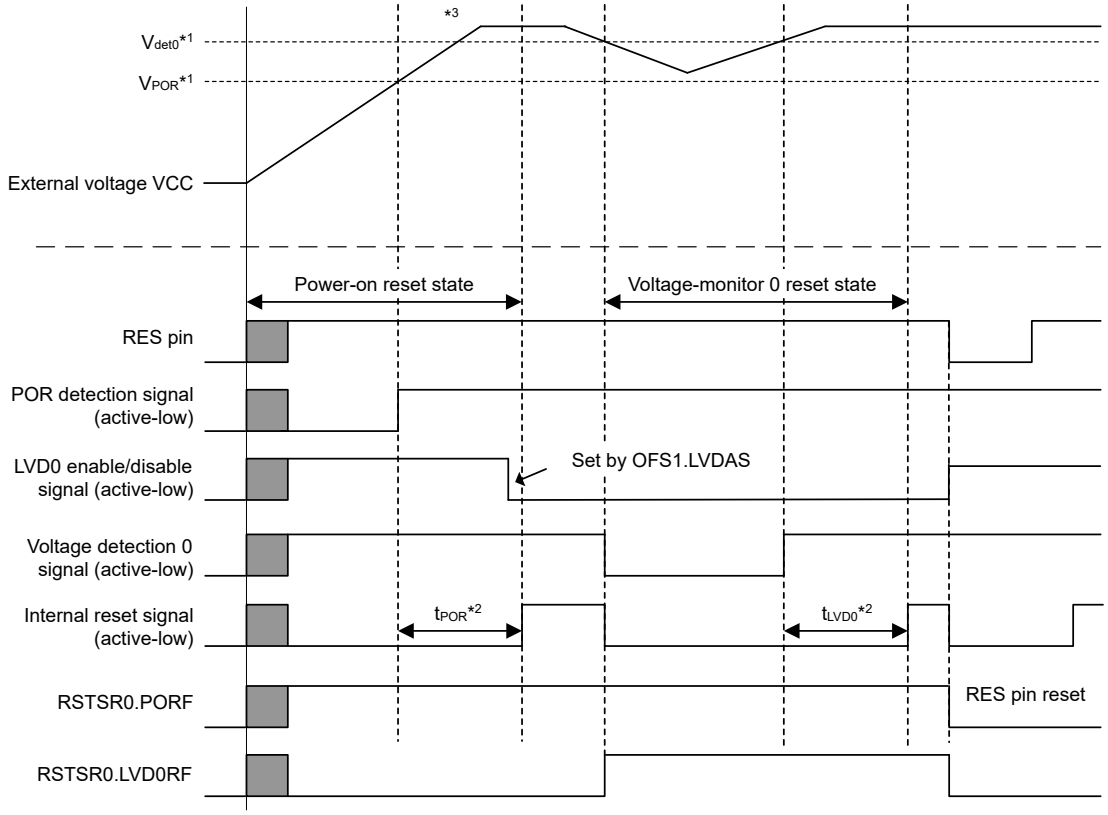
| Step | | Monitoring the results of comparison by voltage monitor 2 |
|---|---|---|
| Setting up the voltage detection 2 circuit | 1 | Set LVD2CMPCR.LVD2E = 0 to disable voltage detection 2 before writing to the LVD2CMPCR.LVD2LVL[2:0] bits. |
| | 2 | Select the detection voltage in the LVD2CMPCR.LVD2LVL[2:0] bits. |
| | 3 | Set LVD2CMPCR.LVD2E = 1 to enable the voltage detection 2 circuit. |
| | 4 | Wait for at least $t_{d\ (E-A)}$ for the LVD2 operation stabilization time after LVD2 is enabled.*1 |
| Setting the digital filter*2 | 5 | Select the sampling clock for the digital filter in the LVD2CR0.FSAMP[1:0] bits. |
| | 6 | Set LVD2CR0.DFDIS = 0 to enable the digital filter. |
| | 7 | Wait for at least 2n + 3 cycles of the LOCO, where n = 2, 4, 8, or 16, and the sampling clock for the digital filter is the LOCO frequency-divided by n. |
| Enabling output | 8 | Set LVD2CR0.CMPE = 1 to enable output of the comparison results from voltage monitor 2. |

Note 1. Steps 5 to 7 can be performed during the wait time of step 4. For details of $t_{d(E-A)}$, see section 47, Electrical Characteristics.
Note 2. Steps 5 to 7 are not required if the digital filter is not in use.

## 7.4 Reset from Voltage Monitor 0

When using the reset from voltage monitor 0, clear the OFS1.LVDAS bit to 0 to enable the voltage monitor 0 reset after a reset. However, at boot mode, the reset from voltage monitor 0 is disabled regardless of the value of the OFS1.LVDAS bit.

Figure 7.4 shows an example of operations for a voltage monitor 0 reset.




Note: For details of the electrical characteristics, see section 47, Electrical Characteristics.
Note 1. $V_{POR}$ indicates the detection level for a power-on reset and $V_{det0}$ indicates the detection level for a voltage monitor 0 reset.
Note 2. $t_{POR}$: the period until the power-on reset is released.
$t_{LVD0}$: the period until the LVD0 reset is released.
Note 3. At power-on, the POR reset is released after VCC has reached the minimum guaranteed voltage.

**Figure 7.4 Example of voltage monitor 0 reset operation**

## 7.5 Interrupt and Reset from Voltage Monitor 1

An interrupt or reset can be generated in response to the comparison results from the voltage monitor 1 circuit.

Table 7.4 shows the procedures for setting bits related to the voltage monitor 1 interrupt/reset so that voltage monitoring occurs. Table 7.5 shows the procedures for setting bits related to the voltage monitor 1 interrupt/reset so that voltage monitoring stops. Figure 7.5 shows an example of operations for a voltage monitor 1 interrupt. For the operation of the voltage monitor 1 reset, see Figure 5.2 in section 5, Resets.

When using the voltage monitor 1 circuit in Software Standby mode or Deep Software Standby mode, set up the circuit using the procedures in this section.

### (1) Setting in Software Standby mode

- Disable the digital filter (LVD1CR0.DFDIS = 1).
- When VCC > $V_{det1}$ is detected, negate the voltage monitor 1 reset signal (LVD1CR0.RN = 0) following a stabilization time.

### (2) Settings in Deep Software Standby mode

- Disable the digital filter (LVD1CR0.DFDIS = 1).

- Enable the voltage monitor 1 interrupt (LVD1CR0.RI = 0). If the voltage monitor 1 reset is enabled (LVD1CR0.RI = 1), a transition to Deep Software Standby mode is not possible, and the operation transitions to Software Standby mode instead.
- When the DPSBYCR.DEEPCUT[1:0] bits are 11b, the voltage monitor 1 circuit stops. To use the voltage monitor 1 circuit in Deep Software Standby mode, set the DPSBYCR.DEEPCUT[1:0] bits to a value other than 11b.

**Table 7.4 Procedures for setting bits related to voltage monitor 1 interrupt and voltage monitor 1 reset so that voltage monitoring occurs**

| Step | | Voltage monitor 1 interrupt (voltage monitor 1 ELC event output) | Voltage monitor 1 reset |
|---|---|---|---|
| Setting up the voltage detection 1 circuit | 1 | Set LVD1CMPCR.LVD1E = 0 to disable voltage detection 1 before writing to the LVD1CMPCR register. | |
| | 2 | Select the detection voltage in the LVD1CMPCR.LVD1LVL[4:0] bits. | |
| | 3 | Set LVD1CMPCR.LVD1E = 1 to enable the voltage detection 1 circuit. | |
| | 4 | Wait for at least $t_{d\ (E-A)}$ for the LVD1 operation stabilization time after LVD1 is enabled.*1 | |
| Setting the digital filter*3 | 5 | Select the sampling clock for the digital filter in the LVD1CR0.FSAMP[1:0] bits. | |
| | 6 | Set LVD1CR0.DFDIS = 0 to enable the digital filter. | |
| | 7 | Wait for at least 2n + 3 LOCO cycles, where n = 2, 4, 8, or 16, and the sampling clock for the digital filter is the LOCO frequency-divided by n.*4 | |
| Setting up the voltage monitor 1 interrupt or reset | 8 | Set LVD1CR0.RI = 0 to select the voltage monitor 1 interrupt. | • Set LVD1CR0.RI = 1 to select the voltage monitor 1 reset.<br>• Select the type of reset negation in the LVD1CR0.RN bit. |
| | 9 | • Select the interrupt request condition in the LVD1CR1.IDTSEL[1:0] bits.<br>• Select the interrupt type in the LVD1CR1.IRQSEL bit. | — |
| Enabling output | 10 | Set LVD1SR.DET = 0. | |
| | 11 | Set LVD1CR0.RIE = 1 to enable the voltage monitor 1 interrupt or reset.*2 | |
| | 12 | Set LVD1CR0.CMPE = 1 to enable output of the comparison results from voltage monitor 1. | |

Note 1. Steps 5 to 11 can be performed during the wait time in step 4. For details on $t_{d\ (E-A)}$, see section 47, Electrical Characteristics.
Note 2. Step 11 is not required if only the ELC event signal is to be output.
Note 3. Steps 5 to 7 are not required if the digital filter is not in use.
Note 4. Steps 8 to 11 can be performed during the wait time of step 7.

**Table 7.5 Procedures for setting bits related to voltage monitor 1 interrupt and voltage monitor 1 reset so that voltage monitoring stops**

| Step | | Voltage monitor 1 interrupt (voltage monitor 1 ELC event output), voltage monitor 1 reset |
|---|---|---|
| Stopping the enabling output | 1 | Set LVD1CR0.CMPE = 0 to disable output of the comparison results from voltage monitor 1. |
| | 2 | Wait for at least 2n + 3 cycles of the LOCO, where n = 2, 4, 8, or 16, and the sampling clock for the digital filter is the LOCO frequency-divided by n.*2 |
| | 3 | Set LVD1CR0.RIE = 0 to disable the voltage monitor 1 interrupt or reset.*1 |
| Stopping the digital filter | 4 | Set LVD1CR0.DFDIS = 1 to disable the digital filter.*2 *3 |
| Stopping the voltage detection 1 circuit | 5 | Set LVD1CMPCR.LVD1E = 0 to disable the voltage detection 1 circuit. |

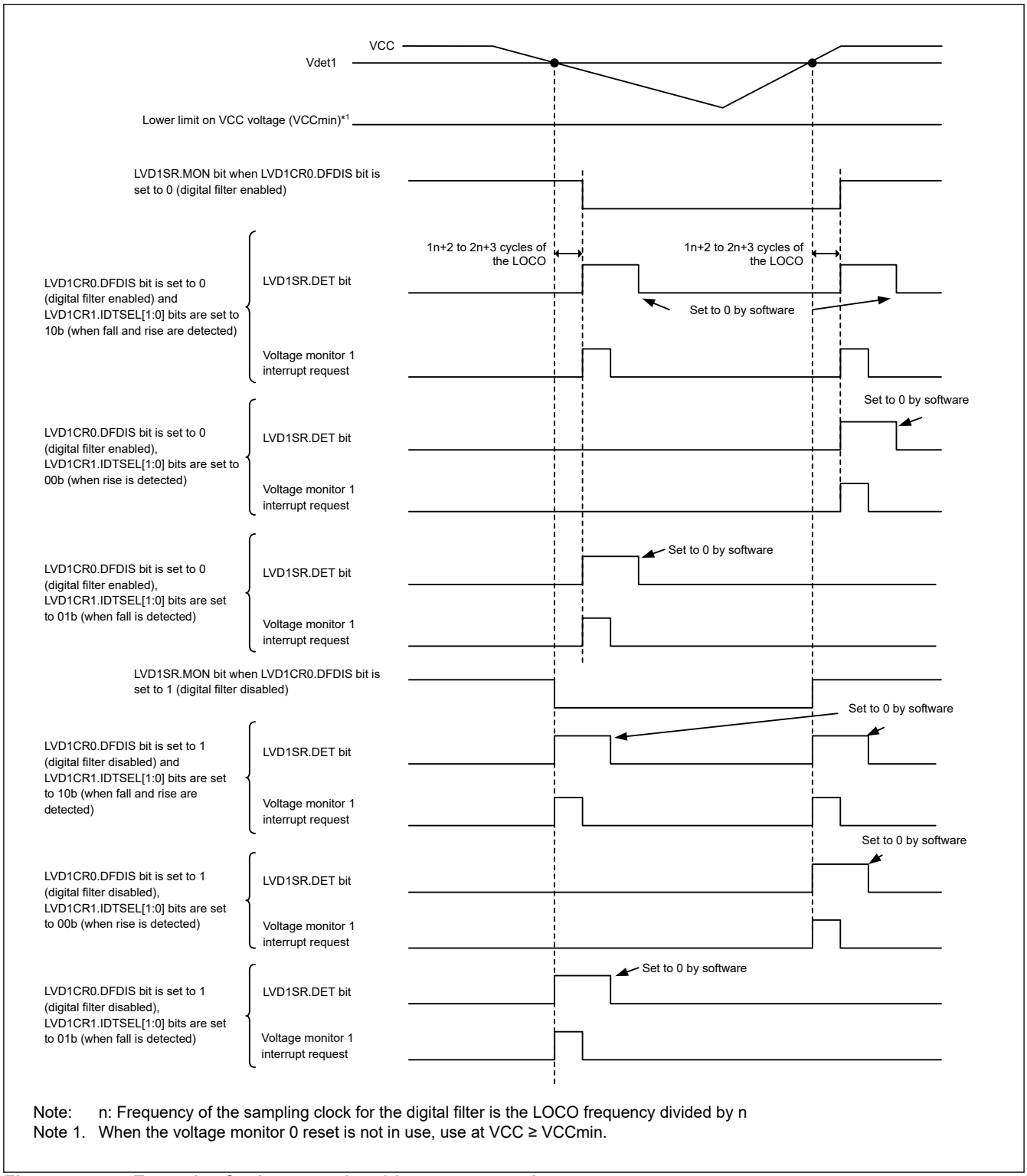
Note 1. Step 3 is not required if only the ELC event signal is to be output.
Note 2. Steps 2 and 4 are not required if the digital filter is not in use.
Note 3. To disable the digital filter from its enabled state and then re-enable it, disable it and wait for at least 2 LOCO clock cycles before re-enabling it.

If the voltage monitor 1 interrupt or reset setting is to be made again after it is used and stopped once, you can omit the following steps in the procedures for stopping and setting, depending on the conditions:

- Setting the voltage detection 1 circuit is not required if the settings for the circuit do not change.
- Setting the digital filter is not required if the settings for the circuit do not change.

- Setting the voltage monitor 1 interrupt or reset is not required if the settings for the voltage monitor 1 interrupt or voltage monitor 1 reset do not change.

Figure 7.5 shows an example of the voltage monitor 1 interrupt operation.




Note: n: Frequency of the sampling clock for the digital filter is the LOCO frequency divided by n
Note 1. When the voltage monitor 0 reset is not in use, use at VCC ≥ VCCmin.

**Figure 7.5 Example of voltage monitor 1 interrupt operation**

## 7.6 Interrupt and Reset from Voltage Monitor 2

An interrupt or reset can be generated in response to the comparison results from the voltage monitor 2 circuit.

Table 7.6 shows the procedures for setting bits related to the voltage monitor 2 interrupt/reset so that voltage monitoring occurs. Table 7.7 shows the procedures for setting bits related to the voltage monitor 2 interrupt/reset so that voltage monitoring stops. Figure 7.6 shows an example of operations for a voltage monitor 2 interrupt. For the operation of the voltage monitor 2 reset, see Figure 5.2 in section 5, Resets.

When using the voltage monitor 2 circuit in Software Standby mode or Deep Software Standby mode, set up the circuit with the following procedures.

### (1) Setting in Software Standby mode

- Disable the digital filter (LVD2CR0.DFDIS = 1)
- When VCC > $V_{det2}$ is detected, negate the voltage monitor 2 reset signal (LVD2CR0.RN = 0) following a LVD2 stabilization time.

### (2) Settings in Deep Software Standby mode

- Disable the digital filter (LVD2CR0.DFDIS = 1).
- Enable the voltage monitor 2 interrupt (LVD2CR0.RI = 0). If the voltage monitor 2 reset is enabled (LVD2CR0.RI = 1), a transition to Deep Software Standby mode is not possible, and the operation transitions to Software Standby mode instead.
- When the DPSBYCR.DEEPCUT[1:0] bits are 11b, the voltage monitor 2 circuit stops. To use the voltage monitor 2 circuit in Deep Software Standby mode, set the DPSBYCR.DEEPCUT[1:0] bits to a value other than 11b.

**Table 7.6 Procedures for setting bits related to voltage monitor 2 interrupt and voltage monitor 2 reset so that voltage monitoring occurs**

| Step | | Voltage monitor 2 interrupt (voltage monitor 2 ELC event output) | Voltage monitor 2 reset |
|---|---|---|---|
| Setting up the voltage detection 2 circuit | 1 | Set LVD2CMPCR.LVD2E = 0 to disable voltage detection 2 before writing to the LVD2CMPCR register. | |
| | 2 | Select the detection voltage in the LVD2CMPCR.LVD2LVL[2:0] bits. | |
| | 3 | Set LVD2CMPCR.LVD2E = 1 to enable the voltage detection 2 circuit. | |
| | 4 | Wait for at least $t_{d\ (E-A)}$ for the LVD2 operation stabilization time after LVD2 is enabled.*1 | |
| Setting the digital filter*3 | 5 | Select the sampling clock for the digital filter in the LVD2CR0.FSAMP[1:0] bits. | |
| | 6 | Set LVD2CR0.DFDIS = 0 to enable the digital filter. | |
| | 7 | Wait for at least 2n + 3 LOCO cycles, where n = 2, 4, 8, or 16, and the sampling clock for the digital filter is the LOCO frequency-divided by n.*4 | |
| Setting up the voltage monitor 2 interrupt or reset | 8 | Set LVD2CR0.RI = 0 to select the voltage monitor 2 interrupt. | • Set LVD2CR0.RI = 1 to select the voltage monitor 2 reset.<br>• Select the type of reset negation in the LVD2CR0.RN bit. |
| | 9 | • Select the interrupt request condition in the LVD2CR1.IDTSEL[1:0] bits.<br>• Select the interrupt type in the LVD2CR1.IRQSEL bit. | — |
| Enabling output | 10 | Set LVD2SR.DET = 0. | |
| | 11 | Set LVD2CR0.RIE = 1 to enable the voltage monitor 2 interrupt or reset.*2 | |
| | 12 | Set LVD2CR0.CMPE = 1 to enable output of the comparison results from voltage monitor 2. | |

Note 1. Steps 5 to 11 can be performed during the wait time in step 4. For details on $t_{d(E-A)}$, see section 47, Electrical Characteristics.
Note 2. Step 11 is not required if only the ELC event signal is to be output.
Note 3. Steps 5 to 7 are not required if the digital filter is not in use.
Note 4. Steps 8 to 11 can be performed during the wait time of step 7.

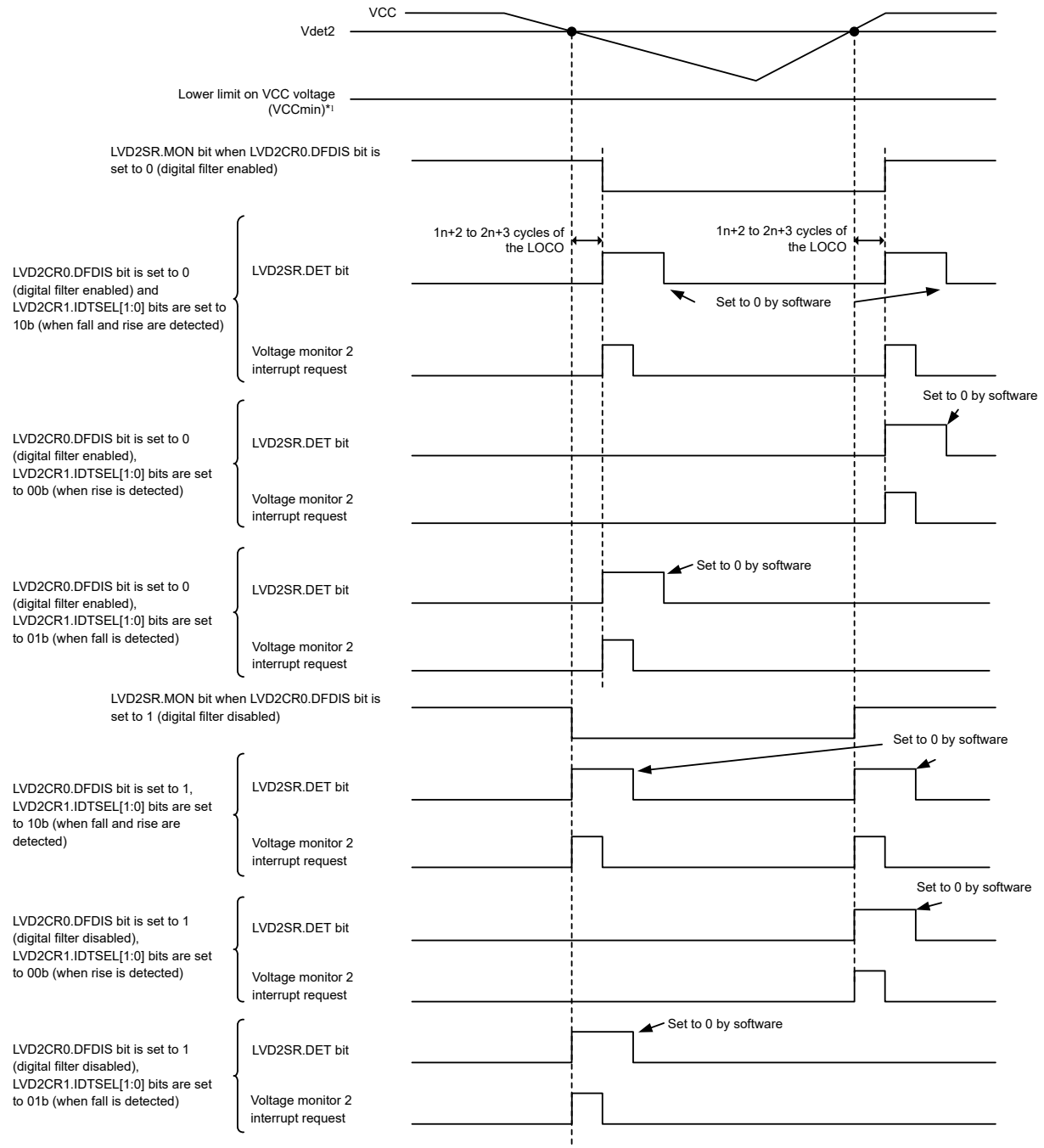
**Table 7.7 Procedures for setting bits related to voltage monitor 2 interrupt and voltage monitor 2 reset so that voltage monitoring stops**

| Step | | Voltage monitor 2 interrupt (voltage monitor 2 ELC event output), voltage monitor 2 reset |
|---|---|---|
| Settings to stop enabling output | 1 | Set LVD2CR0.CMPE = 0 to disable output of the comparison results from voltage monitor 2. |
| | 2 | Wait for at least 2n + 3 cycles of the LOCO, where n = 2, 4, 8, or 16, and the sampling clock for the digital filter is the LOCO frequency-divided by n.*2 |
| | 3 | Set LVD2CR0.RIE = 0 to disable the voltage monitor 2 interrupt or reset.*1 |
| Stopping the digital filter | 4 | Set LVD2CR0.DFDIS = 1 to disable the digital filter. *2 *3 |
| Stopping the voltage detection 2 circuit | 5 | Set LVD2CMPCR.LVD2E = 0 to disable the voltage detection 2 circuit. |

Note 1. Step 3 is not required if only the ELC event signal is to be output.
Note 2. Steps 2 and 4 are not required if the digital filter is not in use.
Note 3. To disable the digital filter from its enabled state and then re-enable it, disable it and wait for at least 2 LOCO clock cycles before re-enabling it.

If the voltage monitor 2 interrupt or reset setting is to be made again after it is used and stopped once, you can omit the following steps in the procedures for stopping and setting, depending on the conditions:

- Setting the voltage detection 2 is not required if the settings for the circuit do not change.
- Setting the digital filter is not required if the settings for the circuit do not change.
- Setting the voltage monitor 2 interrupt or reset is not required if the settings for the voltage monitor 2 interrupt or voltage monitor 2 reset do not change.




Note: n: Frequency of the sampling clock for the digital filter is the LOCO frequency divided by n
Note 1. When the voltage monitor 0 reset is not in use, use at VCC ≥ VCCmin.

**Figure 7.6 Example of voltage monitor 2 interrupt operation**

## 7.7 Event Link Controller (ELC) Output

The LVD can output the event signals to the Event Link Controller (ELC).

### (1) $V_{det1}$ Crossing Detection Event

The LVD outputs the event signal when it detects that the voltage has passed the $V_{det1}$ voltage while both the voltage detection 1 circuit and the voltage monitor 1 circuit comparison result output are enabled.

### (2) $V_{det2}$ Crossing Detection Event

The LVD outputs the event signal when it detects that the voltage has passed the $V_{det2}$ voltage while both the voltage detection 2 circuit and the voltage monitor 2 circuit comparison result output are enabled.

When enabling the event link output function of the LVD, you must enable the LVD before enabling the LVD event link function of the ELC. To stop the event link output function of the LVD, you must stop the LVD before disabling the LVD event link function of the ELC.

## 7.7.1 Interrupt Handling and Event Linking

The LVD provides bits to separately enable or disable the voltage monitor 1 and 2 interrupts. When an interrupt source is generated and the interrupt is enabled by the interrupt enable bit, the interrupt signal is output to the CPU.

In contrast, as soon as an interrupt source is generated, an event link signal is output as the event signal to the other module through the ELC, regardless of the state of the interrupt enable bit.

It is possible to output voltage monitor 1 and 2 interrupts in Software Standby and Deep Software Standby modes. The event signals for the ELC in Software Standby and Deep Software Standby modes are output as follows:

- When a $V_{det1}$ or $V_{det2}$ passage events is detected in Software Standby mode, event signals are not generated for the ELC because the clock is not supplied in Software Standby mode. Because the $V_{det1}$ and $V_{det2}$ passage detection flags are saved, when the clock supply resumes after returning from Software Standby mode, the event signals for the ELC are output based on the state of the $V_{det1}$ and $V_{det2}$ detection flags.
- When a Vdet1 or Vdet2 passage events are detected in Deep Software Standby mode, event signals are not generated for the ELC.

# 8. Clock Generation Circuit

## 8.1 Overview

The MCU provides a clock generation circuit. Table 8.1 and Table 8.2 list the clock generation circuit specifications. Figure 8.1 show a block diagram, and Table 8.3 lists the I/O pins.

**Table 8.1 Clock generation circuit specifications for the clock sources**

| Clock source | Description | Specification |
|---|---|---|
| Main clock oscillator (MOSC) | Resonator frequency | 8 MHz to 24 MHz<br>8, 10, 16, 20, 24 MHz (USB boot mode) |
| | External clock input frequency | Up to 24 MHz |
| | External resonator or additional circuit | ceramic resonator, crystal |
| | Connection pins | EXTAL, XTAL |
| | Drive capability switching | Available |
| | Oscillation stop detection function | Available |
| Sub-clock oscillator (SOSC) | Resonator frequency | 32.768 kHz |
| | External resonator or additional circuit | crystal resonator |
| | Connection pins | XCIN, XCOUT |
| | Drive capability switching | Available |
| PLL circuit | Input clock source | MOSC, HOCO |
| | Input pulse frequency division ratio | Selectable from 1, 2, and 3 |
| | Input frequency | 8 MHz to 24 MHz |
| | Frequency multiplication ratio | Selectable from 10 to 30 (0.5 steps) |
| | Output pulse frequency division ratio | Unavailable |
| | PLL Output frequency | 100 MHz to 200 MHz |
| PLL2 circuit | Input clock source | MOSC, HOCO |
| | Input pulse frequency division ratio | Selectable from 1, 2, and 3 |
| | Input frequency | 8 MHz to 24 MHz |
| | Frequency multiplication ratio | Selectable from 10 to 30 (0.5 steps) |
| | Output pulse frequency division ratio | Unavailable |
| | PLL Output frequency | 120 MHz to 240 MHz |
| | PLL2-LDO stop function | Available |
| High-speed on-chip oscillator (HOCO) | Oscillation frequency | 16/18/20 MHz |
| | FLL function | Available |
| | User trimming | Available |
| Middle-speed on-chip oscillator (MOCO) | Oscillation frequency | 8 MHz |
| | User trimming | Available |
| Low-speed on-chip oscillator (LOCO) | Oscillation frequency | 32.768 kHz |
| | User trimming | Available |
| IWDT-dedicated on-chip oscillator (IWDTLOCO) | Oscillation frequency | 15 kHz |
| | User trimming | Unavailable |
| External clock input for JTAG (TCK) | Input clock frequency | Up to 25 MHz |
| External clock input for SWD (SWCLK) | Input clock frequency | Up to 25 MHz |

**Table 8.2 Clock generation circuit specifications for the internal clocks**

| Item | Clock source | Clock supply | Specification |
|---|---|---|---|
| System clock (ICLK) | MOSC/SOSC/HOCO/MOCO/ LOCO/PLL | CPU, DTC, DMAC, Flash, RAM | Up to 100 MHz<br>Division ratios: 1/2/4/8/16/32/64 |
| Peripheral module clock A (PCLKA) | MOSC/SOSC/HOCO/MOCO/ LOCO/PLL | Peripheral modules (QSPI, SCI, SPI, CRC, DOC, ADC12, DAC12, SCE9, GPT bus clock) | Up to 100 MHz<br>Division ratio: 1/2/4/8/16/32/64 |
| Peripheral module clock B (PCLKB) | MOSC/SOSC/HOCO/MOCO/ LOCO/PLL | Peripheral modules (CAC, ELC, I/O ports, POEG, RTC, WDT, IWDT, AGT, IIC, CAN, USBFS, SSIE, SDHI, TSN, CTSU, Standby SRAM) | Up to 50 MHz<br>Division ratio: 1/2/4/8/16/32/64 |
| Peripheral module clock C (PCLKC) | MOSC/SOSC/HOCO/MOCO/ LOCO/PLL | Peripheral module (ADC12 conversion clock) | Up to 50 MHz<br>Division ratio: 1/2/4/8/16/32/64 |
| Peripheral module clock D (PCLKD) | MOSC/SOSC/HOCO/MOCO/ LOCO/PLL | Peripheral module(GPT count clock) | Up to 100 MHz<br>Division ratio: 1/2/4/8/16/32/64 |
| FlashIF clock (FCLK) | MOSC/SOSC/HOCO/MOCO/ LOCO/PLL | FlashIF | 4 MHz to 50 MHz(P/E)<br>Up to 50 MHz(read)<br>Division ratio: 1/2/4/8/16/32/64 |
| USB clock (USBCLK) | PLL/PLL2 | USBFS | 48 MHz<br>Division ratio: 3/4/5 |
| CAN clock (CANMCLK) | MOSC | CAN | 8 MHz to 24 MHz |
| AGT clock (AGTSCLK) | SOSC | AGT | 32.768 kHz |
| AGT clock (AGTLCLK) | LOCO | AGT | 32.768 kHz |
| CAC Main clock (CACMCLK) | MOSC | CAC | Up to 24 MHz |
| CAC Sub clock (CACSCLK) | SOSC | CAC | 32.768 kHz |
| CAC LOCO clock (CACLCLK) | LOCO | CAC | 32.768 kHz |
| CAC MOCO clock (CACMOCLK) | MOCO | CAC | 8 MHz |
| CAC HOCO clock (CACHCLK) | HOCO | CAC | 16/18/20 MHz |
| CAC IWDTLOCO clock (CACILCLK) | IWDTLOCO | CAC | 15 kHz |
| RTC clock (RTCCLK) | SOSC/LOCO | RTC | 32.768 kHz |
| IWDT clock (IWDTCLK) | IWDTLOCO | IWDT | 15 kHz |
| SysTick timer clock (SYSTICCLK) | LOCO | SysTick timer | 32.768 kHz |
| JTAG clock (JTAGTCK) | TCK | JTAG | Up to 25 MHz |
| Serial wire clock (SWCLK) | SWCLK | OCD | Up to 25 MHz |
| Trace clock (TRCLK) | MOSC/SOSC/HOCO/MOCO/ LOCO/PLL | CPU-OCD | Up to 50 MHz, Division ratio: 1/2/4 |
| TCLK pin output (TCLK) | 1/2 TRCLK | TCLK pin | Up to 25 MHz |
| Clock/buzzer output (CLKOUT) | MOSC/SOSC/LOCO/MOCO/ HOCO | CLKOUT pin | Up to 60 MHz<br>Division ratios: 1/2/4/8/16/32/64/128 |

Note: Restrictions on setting clock frequency: ICLK ≥ PCLKA ≥ PCLKB, PCLKD ≥ PCLKA ≥ PCLKB
ICLK ≥ FCLK
Restrictions on clock frequency ratio: (N: integer, and up to 64)
ICLK:FCLK = N:1, ICLK:PCLKA = N:1, ICLK:PCLKB = N:1, ICLK:PCLKC = N:1 or 1:N , ICLK:PCLKD= N:1 or 1:N, ICLK:TRCLK= N:1 or 1:N
If the A/D converter is enabled, the clock frequency ratio is constrained as follows:
PCLKA:PCLKC ＝ 1:1 or 2:1 or 4:1 or 8:1 or 1:2 or 1:4

Note: Restrictions on the minimum FCLK frequency 4MHz when P/E.

Note: The multiplication of PLL and PLL2 should be set to be within the output frequency range of PLL and PLL2, taking the frequency of HOCO into consideration when not using the FLL function.

Note: Clocks have a permissible frequency range (See Table 8.2).

Flash memory and SRAM also have a permissible operating frequency range in each wait cycle setting. (See section 42, SRAM, section 44, Flash Memory)
Those clock frequency ranges must be satisfied even if the HOCO has its maximum or minimum frequency when not using FLL function. (See section 47, Electrical Characteristics).

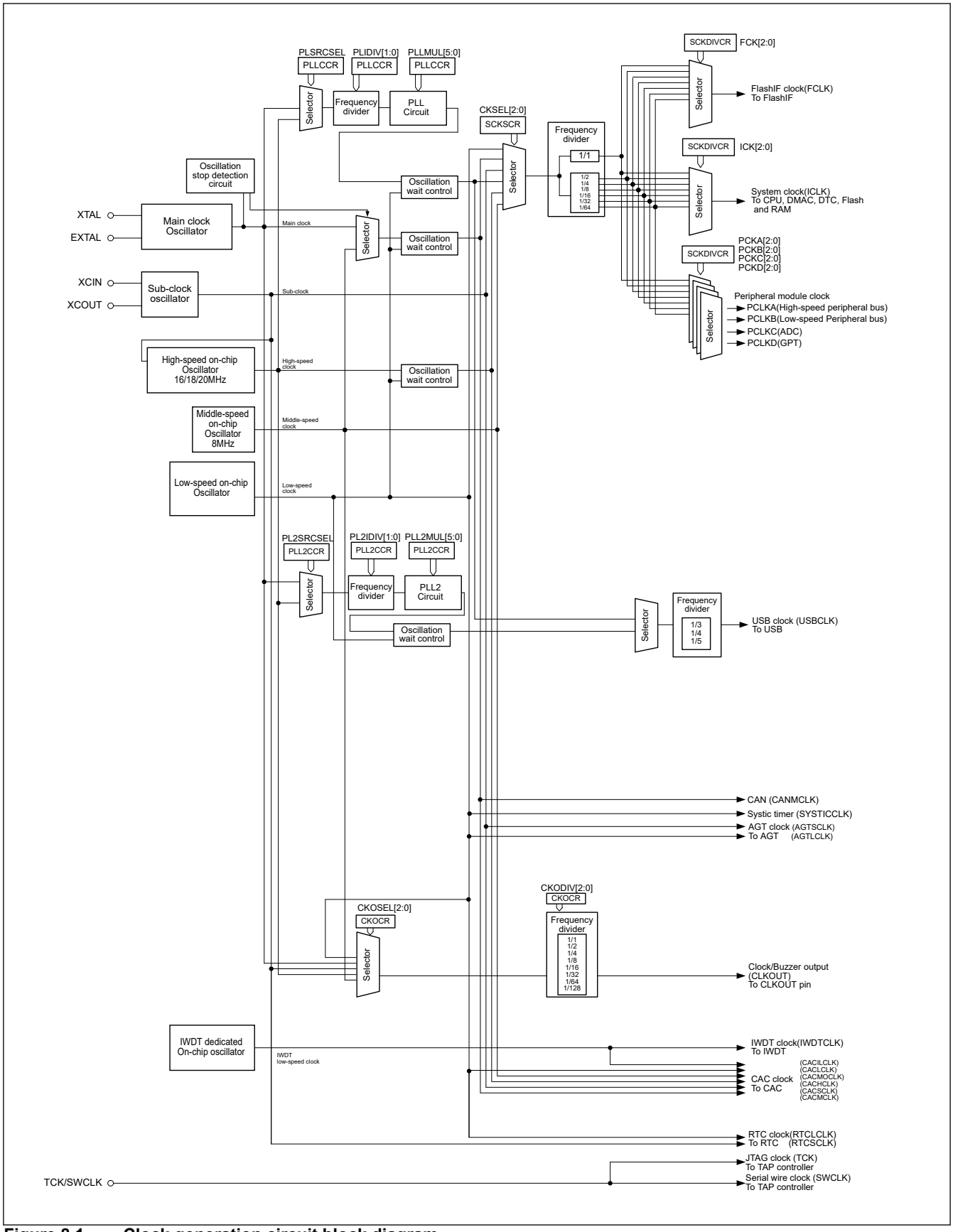



**Figure 8.1 Clock generation circuit block diagram**

Table 8.3 lists the input/output pins of the clock generation circuit.

**Table 8.3 Input/Output Pins of Clock Generation Circuit**

| Pin name | I/O | Description |
|---|---|---|
| XTAL | Output | These pins are used to connect a ceramic resonator or crystal resonator. The EXTAL pin can also be used to input an external clock. For details, see section 8.3.2. External Clock Input. |
| EXTAL | Input | |
| XCIN | Input | These pins are used to connect a 32.768-kHz crystal resonator |
| XCOUT | Output | |
| TCK/SWCLK | Input | This pin is used to input the clock for the JTAG/SWD |
| CLKOUT | Output | This pin is used to output the CLKOUT/BUZZER clock |

## 8.2 Register Descriptions

### 8.2.1 CGFSAR : Clock Generation Function Security Attribute Register

Base address: SYSC = 0x4001_E000

Offset address: 0x3C0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | NONSEC16 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | NONSEC11 | — | NONSEC09 | NONSEC08 | NONSEC07 | NONSEC06 | NONSEC05 | NONSEC04 | NONSEC03 | NONSEC02 | — | NONSEC00 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | NONSEC00*1 | Non Secure Attribute bit 00<br>Target register: SCKDIVCR, SCKSCR<br>Target factor: system clock control<br>0: Secure<br>1: Non Secure | R/W |
| 1 | — | This bit is read as 1. The write value should be 1. | R/W |
| 2 | NONSEC02*1 | Non Secure Attribute bit 02<br>Target register: HOCOCR, FLLCR1, FLLCR2, HOCOUTCR<br>Target factor: HOCO<br>0: Secure<br>1: Non Secure | R/W |
| 3 | NONSEC03*1 | Non Secure Attribute bit 03<br>Target register: MOCOCR, MOCOUTCR<br>Target factor: MOCO<br>0: Secure<br>1: Non Secure | R/W |
| 4 | NONSEC04 | Non Secure Attribute bit 04<br>Target register: LOCOCR, LOCOUTCR<br>Target factor: LOCO<br>0: Secure<br>1: Non Secure | R/W |
| 5 | NONSEC05 | Non Secure Attribute bit 05<br>Target register: MOSCCR, MOSCWTCR, MOMCR<br>Target factor: MOSC<br>0: Secure<br>1: Non Secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6 | NONSEC06 | Non Secure Attribute bit 06<br>Target register: OSTDCR, OSTDSR<br>Target factor: oscillation stop detection control<br>0: Secure<br>1: Non Secure | R/W |
| 7 | NONSEC07 | Non Secure Attribute bit 07<br>Target register: SOSCCR, SOMCR<br>Target factor: SOSC<br>0: Secure<br>1: Non Secure | R/W |
| 8 | NONSEC08*1 | Non Secure Attribute bit 08<br>Target register: PLLCCR, PLLCR<br>Target factor: PLL<br>0: Secure<br>1: Non Secure | R/W |
| 9 | NONSEC09 | Non Secure Attribute bit 09<br>Target register: PLL2CCR, PLL2CR<br>Target factor: PLL2<br>0: Secure<br>1: Non Secure | R/W |
| 10 | — | This bit is read as 1. The write value should be 1. | R/W |
| 11 | NONSEC11 | Non Secure Attribute bit 11<br>Target register: CKOCR<br>Target factor: CLKOUT control<br>0: Secure<br>1: Non Secure | R/W |
| 15:12 | — | These bits are read as 1. The write value should be 1. | R/W |
| 16 | NONSEC16 | Non Secure Attribute bit 16<br>Target register: USBCKDIVCR, USBCKCR<br>Target factor: USBCLK<br>0: Secure<br>1: Non Secure | R/W |
| 17 | — | This bit is read as 1. The write value should be 1. | R/W |
| 18 | — | This bit is read as 1. The write value should be 1. | R/W |
| 19 | — | This bit is read as 1. The write value should be 1. | R/W |
| 20 | — | This bit is read as 1. The write value should be 1. | R/W |
| 31:21 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.
Note 1. It is recommended that these bits are configured as Non Secure when the device life cycle is NSECSD (DLMMON.DLMMON[3:0] = 0011b). See section 46.6.1. Restrictions on setting the security attribution for the details.

CGFSAR register controls the secure attribute of Clock Generation Function registers.

### NONSEC00 bit (Non Secure Attribute bit 00)

This bit controls the security attribute of SCKDIVCR, SCKSCR.

### NONSEC02 bit (Non Secure Attribute bit 02)

This bit controls the security attribute of HOCOCR, FLLCR1, FLLCR2, HOCOUTCR.

### NONSEC03 bit (Non Secure Attribute bit 03)

This bit controls the security attribute of MOCOCR, MOCOUTCR.

### NONSEC04 bit (Non Secure Attribute bit 04)

This bit controls the security attribute of LOCOCR, LOCOUTCR.

**NONSEC05 bit (Non Secure Attribute bit 05)**

This bit controls the security attribute of MOSCCR, MOSCWTCR, MOMCR.

**NONSEC06 bit (Non Secure Attribute bit 06)**

This bit controls the security attribute of OSTDCR, OSTDSR.

**NONSEC07 bit (Non Secure Attribute bit 07)**

This bit controls the security attribute of SOSCCR, SOMCR.

**NONSEC08 bit (Non Secure Attribute bit 08)**

This bit controls the security attribute of PLLCCR, PLLCR.

**NONSEC09 bit (Non Secure Attribute bit 09)**

This bit controls the security attribute of PLL2CCR, PLL2CR.

**NONSEC11 bit (Non Secure Attribute bit 11)**

This bit controls the security attribute of CKOCR.

**NONSEC16 bit (Non Secure Attribute bit 16)**

This bit controls the security attribute of USBCKDIVCR, USBCKCR.

## 8.2.2 SCKDIVCR : System Clock Division Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x020

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | FCK[2:0] | | | — | ICK[2:0] | | | — | — | — | — | — | RSV | | |
| Value after reset: | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | PCKA[2:0] | | | — | PCKB[2:0] | | | — | PCKC[2:0] | | | — | PCKD[2:0] | | |
| Value after reset: | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | PCKD[2:0]*3 | Peripheral Module Clock D (PCLKD) Select<br>0 0 0: × 1/1<br>0 0 1: × 1/2<br>0 1 0: × 1/4<br>0 1 1: × 1/8<br>1 0 0: × 1/16<br>1 0 1: × 1/32<br>1 1 0: × 1/64<br>Others: Setting prohibited. | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6:4 | PCKC[2:0]*3 | Peripheral Module Clock C (PCLKC) Select<br>0 0 0: × 1/1<br>0 0 1: × 1/2<br>0 1 0: × 1/4<br>0 1 1: × 1/8<br>1 0 0: × 1/16<br>1 0 1: × 1/32<br>1 1 0: × 1/64<br>Others: Setting prohibited. | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 10:8 | PCKB[2:0]*2 | Peripheral Module Clock B (PCLKB) Select<br>0 0 0: × 1/1<br>0 0 1: × 1/2<br>0 1 0: × 1/4<br>0 1 1: × 1/8<br>1 0 0: × 1/16<br>1 0 1: × 1/32<br>1 1 0: × 1/64<br>Others: Setting prohibited. | R/W |
| 11 | — | This bit is read as 0. The write value should be 0. | R/W |
| 14:12 | PCKA[2:0]*2 | Peripheral Module Clock A (PCLKA) Select<br>0 0 0: × 1/1<br>0 0 1: × 1/2<br>0 1 0: × 1/4<br>0 1 1: × 1/8<br>1 0 0: × 1/16<br>1 0 1: × 1/32<br>1 1 0: × 1/64<br>Others: Setting prohibited. | R/W |
| 15 | — | This bit is read as 0. The write value should be 0. | R/W |
| 18:16 | RSV | Reserved. Set these bits to the same value as PCKB[2:0].<br>0 0 0: × 1/1<br>0 0 1: × 1/2<br>0 1 0: × 1/4<br>0 1 1: × 1/8<br>1 0 0: × 1/16<br>1 0 1: × 1/32<br>1 1 0: × 1/64<br>Others: Settings prohibited | R/W |
| 23:19 | — | These bits are read as 0. The write value should be 0. | R/W |
| 26:24 | ICK[2:0]*1*2*3*4*5 | System Clock (ICLK) Select<br>0 0 0: × 1/1<br>0 0 1: × 1/2<br>0 1 0: × 1/4<br>0 1 1: × 1/8<br>1 0 0: × 1/16<br>1 0 1: × 1/32<br>1 1 0: × 1/64<br>Others: Setting prohibited. | R/W |
| 27 | — | This bit is read as 0. The write value should be 0. | R/W |
| 30:28 | FCK[2:0]*1 | FlashIF Clock (FCLK) Select<br>0 0 0: × 1/1<br>0 0 1: × 1/2<br>0 1 0: × 1/4<br>0 1 1: × 1/8<br>1 0 0: × 1/16<br>1 0 1: × 1/32<br>1 1 0: × 1/64<br>Others: Setting prohibited. | R/W |
| 31 | — | This bit is read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

Note 1. The following relation is required between the frequencies of the system clock (ICLK) and the FlashIF clock (FCLK).
ICLK:FCLK=N:1 (N: integer)

Note 2. The following relation is required between the frequencies of the system clock (ICLK) and the peripheral module clocks (PCLKA, PCLKB)
ICLK:PCLKA = N:1, ICLK:PCLKB = N:1 (N: integer)

Note 3. The following relation is required between the frequencies of the system clock (ICLK) and the peripheral module clocks (PCLKC, PCLKD):
ICLK:PCLKC,PCLKD = N:1or1:N (N: integer)
Note 4. Selecting the division-by-1 to ICLK is prohibited when SCKSCR.CKSEL[2:0] bits select the system clock source faster than 100 MHz.
Note 5. The frequency of the system clock (ICLK) is limited to the flash wait cycle register (FLWT). See section 44, Flash Memory.

SCKDIVCR selects the frequencies of the system clock (ICLK), peripheral module clock (PCLKA, PCLKB, PCLKC, PCLKD), FlashIF clock (FCLK).

When the PLL is selected as the clock source, set the following modules to the module-stop state before changing the value of SCKDIVCR:

- SCE9

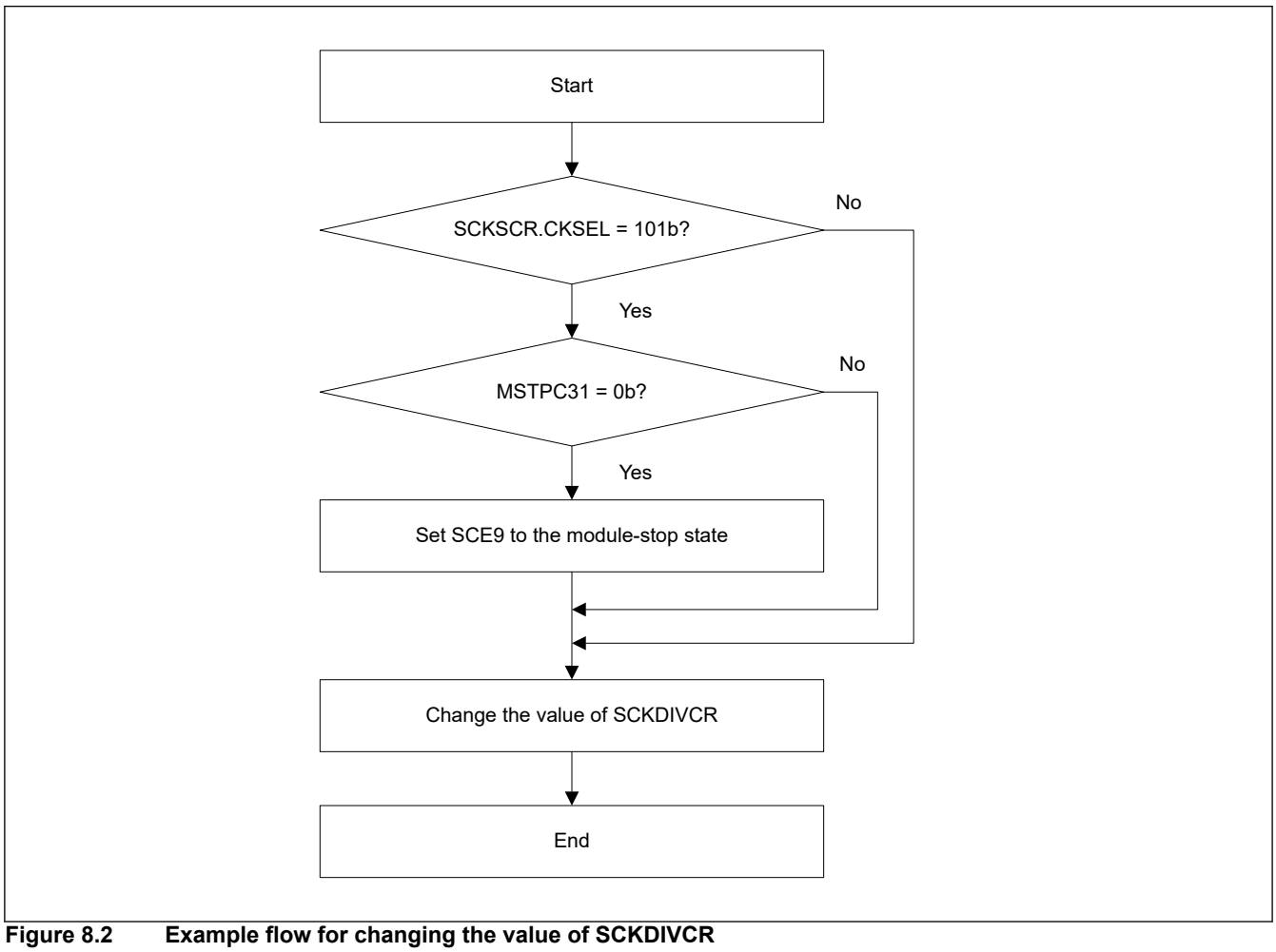



**Figure 8.2 Example flow for changing the value of SCKDIVCR**

## 8.2.3 SCKSCR : System Clock Source Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x026

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | CKSEL[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | CKSEL[2:0] | Clock Source Select<br>0 0 0: HOCO<br>0 0 1: MOCO<br>0 1 0: LOCO<br>0 1 1: Main clock oscillator (MOSC)<br>1 0 0: Sub-clock oscillator (SOSC)<br>1 0 1: PLL<br>1 1 0: Setting prohibited<br>1 1 1: Setting prohibited | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.
Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The SCKSCR register selects the clock source for the system clock.

When changing the value of SCKSCR to either select or deselect the PLL, set the following modules to the module-stop state before changing the SCKSCR value:

- SCE9

Figure 8.3 shows an example flow for changing the value of SCKSCR.

### CKSEL[2:0] bits (Clock Source Select)

The CKSEL[2:0] bits select the source for the following modules:

- System clock (ICLK)
- Peripheral module clocks (PCLKA, PCLKB, PCLKC, and PCLKD)
- FlashIF clock (FCLK)

The bits select from one of the following sources:

- Low-speed on-chip oscillator (LOCO)
- Middle-speed on-chip oscillator (MOCO)
- High-speed on-chip oscillator (HOCO)
- Main clock oscillator (MOSC)
- Sub-clock oscillator (SOSC)
- PLL

The operating state of each clock source is controlled not only by the clock oscillation enable settings but also by the operating modes of the product. Some clock sources might be forcibly stopped depending on the product operating mode being used.

Check the operation state of clock sources in each product operating mode, and do not select the clock source to be stopped in SCKSCR. The clock sources should be switched when there are no occurring internal asynchronous interrupt. For details, see section 10, Low Power Modes.

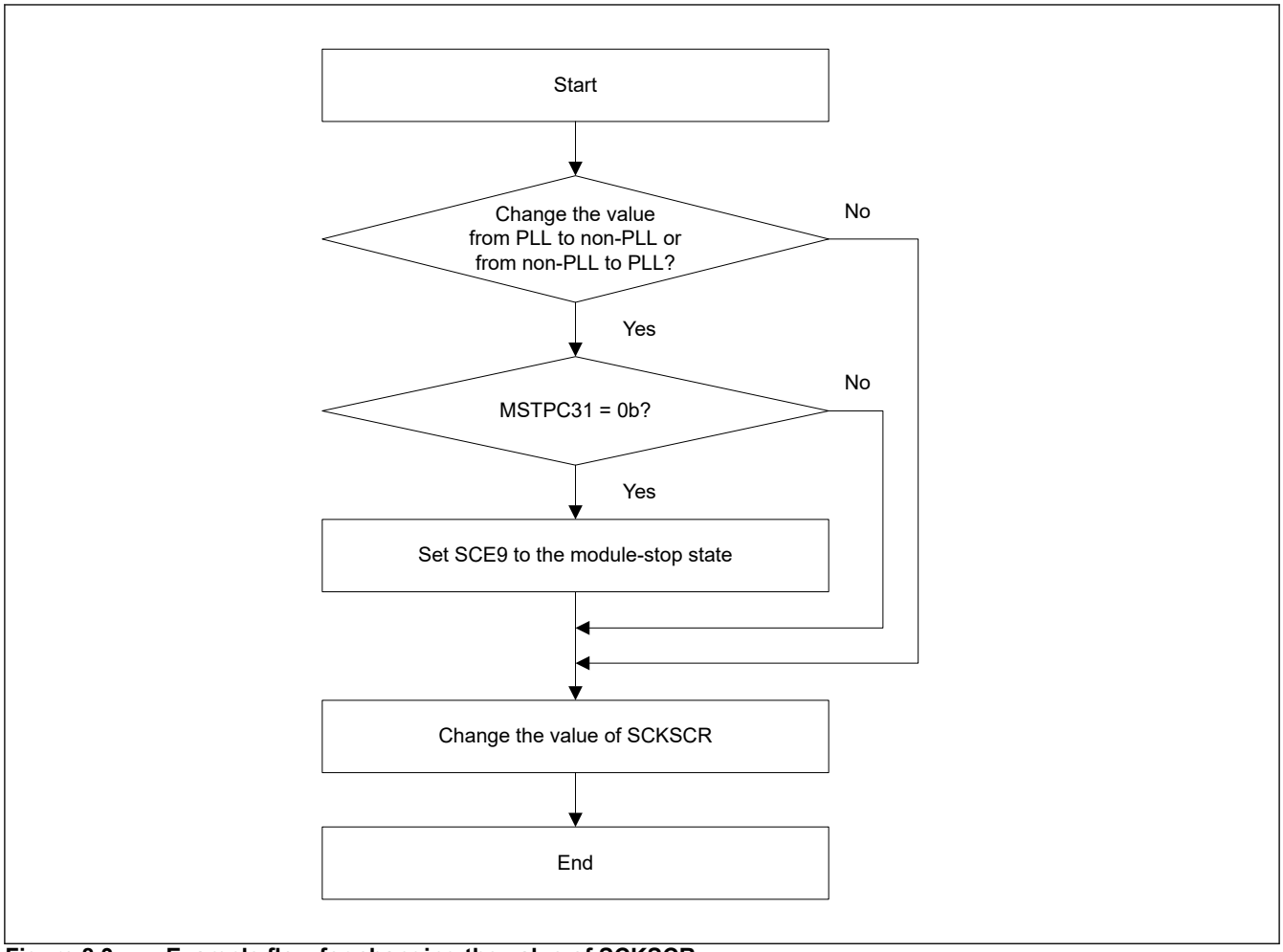



**Figure 8.3 Example flow for changing the value of SCKSCR**

## 8.2.4 PLLCCR : PLL Clock Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x028

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | PLLMUL[5:0] | | | | | | — | — | — | PLSR CSEL | — | — | PLIDIV[1:0] | |
| Value after reset: | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | PLIDIV[1:0]*1 | PLL Input Frequency Division Ratio Select<br>0 0: /1<br>0 1: /2<br>1 0: /3<br>Others: Setting prohibited. | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | PLSRCSEL | PLL Clock Source Select<br>0: Main clock oscillator<br>1: HOCO*3 | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 13:8 | PLLMUL[5:0]*2 | PLL Frequency Multiplication Factor Select<br>0x13: × 10.0 (value after reset)<br>0x14: × 10.5<br>0x15: × 11.0<br>⋮<br>0x1C: × 14.5<br>0x1D: × 15.0<br>0x1E: × 15.5<br>⋮<br>0x3A: × 29.5<br>0x3B: × 30.0<br>Others: Setting prohibited. | R/W |
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.
Note 1. PLIDIV[1:0] should be set so that the frequency of PLL input signal is within the range of section 8.1. Overview.
Note 2. PLLMUL[5:0] should be set so that the frequency of PLL output signal is within the range of section 8.1. Overview.
Note 3. The FLL function must be enabled when using USBCLK.

The PLLCCR register sets the operation of the PLL circuit.

Writing to the PLLCCR is prohibited when the PLLCR.PLLSTP bit is 0 (the PLL operates).

### PLIDIV[1:0] bits (PLL Input Frequency Division Ratio Select)

These bits select the frequency division ratio of the PLL clock source.

### PLSRCSEL bit (PLL Clock Source Select)

This bit selects the clock source for the PLL.

### PLLMUL[5:0] bits (PLL Frequency Multiplication Factor Select)

These bits select the frequency multiplication factor of the PLL circuit.

## 8.2.5 PLLCR : PLL Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x02A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | PLLSTP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PLLSTP | PLL Stop Control<br>0: PLL is operating<br>1: PLL is stopped. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The PLLCR register controls the operation of the PLL circuit.

**PLLSTP bit (PLL Stop Control)**

This bit runs or stops the PLL circuit.

If the main clock oscillator is to be selected as the clock source for the PLL by the PLLCCR.PLSRCSEL bit, the Main Clock Oscillator Wait Control Register (MOSCWTCR) must be set.

After the PLLSTP bit setting is changed to run the PLL, only use the PLL clock after confirming that the OSCSF.PLLSF bit is set to 1. That is, a fixed time for stabilization is required after starting the PLL operation. A fixed time is also required for oscillation to stop after stopping the PLL operation. Additionally, apply the following limitations when starting and stopping the PLL operation by the PLLSTP bit:

- After stopping the PLL, confirm that the OSCSF.PLLSF bit is 0 before restarting the PLL.
- Confirm that the PLL is operating and that the OSCSF.PLLSF bit is 1 before stopping the PLL.
- Regardless of whether the PLL clock is selected as the system clock, confirm that the OSCSF.PLLSF is set to 1 before executing a WFI instruction to place the MCU in Software Standby or Deep Software Standby mode after operating the PLL.
- When transitioning to Software Standby or Deep Software Standby mode after stopping the PLL, confirm that the OSCSF.PLLSF bit is cleared to 0 before executing a WFI instruction.

Writing 1 to the PLLSTP bit is prohibited when SCKSCR.CKSEL[2:0] = 101 (system clock source = PLL).

Confirm the following conditions before writing 0 to PLLSTP:

- When PLL source clock = MOSC: MOSCCR.MOSTP = 0 (MOSC is enabled)
- When PLL source clock = HOCO: HOCOCR.HCSTP = 0 (HOCO is enabled).

## 8.2.6 PLL2CCR : PLL2 Clock Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x048

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | PLL2MUL[5:0] | | | | | | — | — | — | PL2SRCSEL | — | — | PL2IDIV[1:0] | |
| Value after reset: | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | PL2IDIV[1:0]*1 | PLL2 Input Frequency Division Ratio Select<br>0 0: /1 (value after reset)<br>0 1: /2<br>1 0: /3<br>Others: Setting prohibited. | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | PL2SRCSEL | PLL2 Clock Source Select<br>0: Main clock oscillator<br>1: HOCO*3 | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13:8 | PLL2MUL[5:0]*2 | PLL2 Frequency Multiplication Factor Select<br>0x13: × 10.0 (value after reset)<br>0x14: × 10.5<br>0x15: × 11.0<br>⋮<br>0x1C: × 14.5<br>0x1D: × 15.0<br>0x1E: × 15.5<br>⋮<br>0x3A: × 29.5<br>0x3B: × 30.0<br>Others: Setting prohibited. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.
Note 1. PL2IDIV[1:0] should be set so that the frequency of PLL2 input signal is within the range of section 8.1. Overview.
Note 2. PLL2MUL[5:0] should be set so that the frequency of PLL2 output signal is within the range of section 8.1. Overview.
Note 3. The FLL function must be enabled when using USBCLK.

The PLL2CCR register sets the operation of the PLL2 circuit.

Writing to the PLL2CCR register is prohibited when the PLL2CR.PLL2STP bit is 0 (the PLL2 operates).

### PL2IDIV[1:0] bits (PLL2 Input Frequency Division Ratio Select)

These bits select the frequency division ratio of the PLL2 clock source.

### PL2SRCSEL bit (PLL2 Clock Source Select)

This bit selects the clock source for the PLL2.

### PLL2MUL[5:0] bits (PLL2 Frequency Multiplication Factor Select)

These bits select the frequency multiplication factor of the PLL2 circuit.

## 8.2.7 PLL2CR : PLL2 Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x04A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | PLL2STP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PLL2STP | PLL2 Stop Control<br>0: PLL2 is operating<br>1: PLL2 is stopped. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The PLL2CR register controls the operation of the PLL2 circuit.

### PLL2STP bit (PLL2 Stop Control)

This bit runs or stops the PLL2 circuit.

If the main clock oscillator is to be selected as the clock source for the PLL2 by the PLL2CCR.PL2SRCSEL bit, the Main Clock Oscillator Wait Control Register (MOSCWTCR) must be set.

After the PLL2STP bit setting is changed to run the PLL2, only use the PLL2 clock after confirming that the OSCSF.PLL2SF bit is set to 1. That is, a fixed time for stabilization is required after starting the PLL2 operation. A fixed time is also required for oscillation to stop after stopping the PLL2 operation. Additionally, apply the following limitations when starting and stopping the PLL2 operation by the PLL2STP bit:

- After stopping the PLL2, confirm that the OSCSF.PLL2SF bit is 0 before restarting the PLL2.
- Confirm that the PLL2 is operating and that the OSCSF.PLL2SF bit is 1 before stopping the PLL2.

- Confirm that the OSCSF.PLL2SF bit is set to 1 before executing a WFI instruction to place the MCU in Software Standby or Deep Software Standby mode after operating the PLL2.
- When transitioning to Software Standby or Deep Software Standby mode after stopping the PLL2, confirm that the OSCSF.PLL2SF bit is cleared to 0 before executing a WFI instruction.

Confirm the following conditions before writing 0 to PLL2STP:

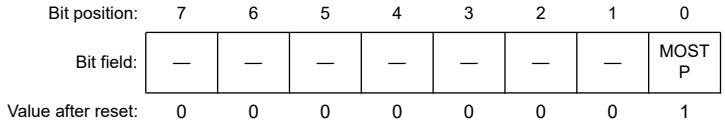
- When the PLL2 source clock = MOSC: MOSCCR.MOSTP = 0 (MOSC is enabled)
- When the PLL2 source clock = HOCO: HOCOCR.HCSTP = 0 (HOCO is enabled).

If PLL2 is not used, PLL2-LDO can be stopped by the PL2LDOSCR register to reduce current consumption. For details, see section 10.2.31. PL2LDOSCR:PLL2-LDO Stop Control Register.

## 8.2.8 MOSCCR : Main Clock Oscillator Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x032




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MOSTP | Main Clock Oscillator Stop<br>0: Operate the main clock oscillator*1<br>1: Stop the main clock oscillator | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.
Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. MOMCR register must be set before setting MOSTP to 0.

The MOSCCR register controls the main clock oscillator.

### MOSTP bit (Main Clock Oscillator Stop)

The MOSTP bit starts or stops the main clock oscillator.

When changing the value of the MOSTP bit, execute subsequent instructions only after reading the bit to check that the value is updated.

When using the main clock, the Main Clock Oscillator Mode Oscillation Control Register (MOMCR) and the Main Clock Oscillator Wait Control Register (MOSCWTCR) must be set before setting MOSTP to 0. After setting the MOSTP bit to 0, confirm that the OSCSF.MOSCSF bit is set to 1 before using the main clock oscillator.

A fixed stabilization wait time is required after setting the main clock oscillator to start operation. A fixed wait time is also required for oscillation to stop after stopping the main clock oscillator.

The following restrictions apply when starting and stopping operation:

- After stopping the main clock oscillator, confirm that the OSCSF.MOSCSF bit is 0 before restarting the main clock oscillator
- Confirm that the main clock oscillator operates and that the OSCSF.MOSCSF bit is 1 before stopping the main clock oscillator
- Regardless of whether the main clock oscillator is selected as the system clock, confirm that the OSCSF.MOSCSF bit is set to 1 before executing a WFI instruction to place the MCU in Software Standby after operating the main clock oscillator or Deep Software Standby mode.

- When a transition to Software Standby or Deep Software Standby mode is to follow the setting to stop the main clock oscillator, confirm that the OSCSF.MOSCSF bit is set to 0 before executing the WFI instruction.

Writing 1 to MOSTP is prohibited under the following condition:

- SCKSCR.CKSEL[2:0] = 011b (system clock source = MOSC).
- PLLCCR.PLSRCSEL = 0 (PLL source clock = MOSC) and SCKSCR.CKSEL[2:0] = 101b (system clock source = PLL)
- PLLCCR.PLSRCSEL = 0 (PLL source clock = MOSC) and PLLCR.PLLSTP = 0 (PLL is operating)
- PLL2CCR.PL2SRCSEL = 0 (PLL2 source clock = MOSC) and PLL2CR.PLL2STP = 0 (PLL2 is operating)

## 8.2.9 SOSCCR : Sub-Clock Oscillator Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x480

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | SOSTP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SOSTP | Sub Clock Oscillator Stop<br>0: Operate the sub-clock oscillator*1<br>1: Stop the sub-clock oscillator | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.
Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. The SOMCR register must be set before setting SOSTP to 0.

The SOSCCR register controls the sub-clock oscillator.

### SOSTP bit (Sub Clock Oscillator Stop)

The SOSTP bit starts or stops the sub-clock oscillator. When changing the value of the SOSTP bit, only execute subsequent instructions after reading the bit to check that the value is updated. Use the SOSTP bit when using the sub-clock oscillator as the source for a peripheral module, for example the RTC. When using the sub-clock oscillator, set the Sub-Clock Oscillator Mode Control Register (SOMCR) before setting SOSTP to 0.

The following restrictions apply when starting and stopping the operation:

- After stopping the sub-clock oscillator, allow a stop interval of at least 5 SOSC clock cycles before restarting it
- After setting the SOSTP bit to 0, use the sub-clock only after the sub-clock oscillation stabilization time ($t_{SUBOSCWT}$) has elapsed.
- Regardless of whether the sub-clock oscillator is selected as the system clock, confirm that the sub-clock oscillation is stable before executing a WFI instruction to place the MCU in Software Standby or Deep Software Standby mode
- When a transition to Software Standby or Deep Software Standby mode is to follow the setting to stop the sub-clock oscillator, wait for at least 3 SOSC clock cycles before executing the WFI instruction.

Writing 1 to SOSTP is prohibited under the following condition:

- SCKSCR.CKSEL[2:0] = 100b (system clock source = SOSC).

## 8.2.10 LOCOCR : Low-Speed On-Chip Oscillator Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x490

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | LCSTP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | LCSTP | LOCO Stop<br>0: Operate the LOCO clock<br>1: Stop the LOCO clock | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The LOCOCR register controls the LOCO clock.

### LCSTP bit (LOCO Stop)

The LCSTP bit starts or stops the LOCO clock.

After setting the LCSTP bit to 0 to start the LOCO clock, only use the clock after the LOCO clock-oscillation stabilization wait time ($t_{LOCOWT}$) elapses. A fixed stabilization wait time is required after setting the LOCO clock to start operation. A fixed wait time is also required after setting the LOCO clock to stop.

The following restrictions apply when starting and stopping operation:

- After stopping the LOCO clock, allow a stop interval of at least 5 LOCO clock cycles before restarting it
- Confirm that LOCO oscillation is stable before stopping the LOCO clock
- Regardless of whether the LOCO is selected as the system clock, confirm that LOCO oscillation is stable before executing a WFI instruction to place the MCU in Software Standby or Deep Software Standby mode
- When a transition to Software Standby or Deep Software Standby mode is to follow the setting to stop the LOCO clock, wait for at least 3 LOCO cycles before executing the WFI instruction.

Writing 1 to LCSTP is prohibited under the following condition:

- SCKSCR.CKSEL[2:0] = 010b (system clock source = LOCO).

Because the LOCO clock measures the wait time for other oscillators, it continues to oscillate while measuring this time, regardless of the setting in LOCOCR.LCSTP. As a result, the LOCO clock might be unintentionally supplied even when the LCSTP is set to stop.

## 8.2.11 HOCOCR : High-Speed On-Chip Oscillator Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x036

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | HCSTP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | HCSTP | HOCO Stop<br>0: Operate the HOCO clock *2<br>1: Stop the HOCO clock | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.
Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. The HCSTP bit value after a reset is 0 when the OFS1.HOCOEN bit is 0. It is 1 when the OFS1.HOCOEN bit is 1.
Note 2. If you are using the HOCO (HCSTP = 0), set the OFS1.HOCOFRQ0[1:0] bit to an optimum value.

The HOCOCR register controls the HOCO clock.

### HCSTP bit (HOCO Stop)

The HCSTP bit starts or stops the HOCO clock.

After setting the HCSTP bit to 0 to start the HOCO clock, confirm that the OSCSF.HOCOSF is set to 1 before using the clock. When OFS1.HOCOEN is set to 0, confirm that OSCSF.HOCOSF is also set to 1 before using the HOCO clock. A fixed stabilization wait time is required after setting the HOCO clock to start operation. A fixed wait time is also required after setting the HOCO clock to stop.

The following limitations apply when starting and stopping operation:

- After stopping the HOCO clock, confirm that the OSCSF.HOCOSF is 0 before restarting the HOCO clock.
- Confirm that the HOCO clock operates and that the OSCSF.HOCOSF is 1 before stopping the HOCO clock.
- Regardless of whether the HOCO clock is selected as the system clock, confirm that the OSCSF.HOCOSF is set to 1 before executing a WFI instruction to place the MCU in Software Standby or Deep Software Standby mode after setting HOCO operation with the HCSTP bit.
- When a transition to Software Standby or Deep Software Standby mode is to follow the setting of the HOCO clock to stop, confirm that the OSCSF.HOCOSF is set to 0 after setting the HOCO clock and before executing the WFI instruction.

Writing 1 to HCSTP is prohibited under the following conditions:

- SCKSCR.CKSEL[2:0] = 000b (system clock source = HOCO).
- PLLCCR.PLSRCSEL = 1 (PLL source clock = HOCO) and SCKSCR.CKSEL[2:0] = 101b (system clock source = PLL)
- PLLCCR.PLSRCSEL = 1 (PLL source clock = HOCO) and PLLCR.PLLSTP = 0 (PLL is operating)
- PLL2CCR.PL2SRCSEL = 1 (PLL2 source clock = HOCO) and PLL2CR.PLL2STP = 0 (PLL2 is operating)

## 8.2.12 MOCOCR : Middle-Speed On-Chip Oscillator Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x038

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | MCSTP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MCSTP | MOCO Stop<br>0: MOCO clock is operating<br>1: MOCO clock is stopped | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.
Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The MOCOCR register controls the MOCO clock.

### MCSTP bit (MOCO Stop)

The MCSTP bit starts or stops the MOCO clock.

After setting MCSTP to 0, use the MOCO clock only after the MOCO clock oscillation stabilization time ($t_{MOCOWT}$) elapses. A fixed stabilization wait time is required after setting the MOCO clock to start operation. A fixed wait time is also required for oscillation to stop after setting the MOCO clock to stop operation.

The following restrictions apply when starting and stopping the oscillator:

- After stopping the MOCO clock, allow a stop interval of at least 5 MOCO clock cycles before restarting it
- Confirm that MOCO clock oscillation is stable before stopping the MOCO clock
- Regardless of whether the MOCO clock is selected as the system clock, confirm that MOCO clock oscillation is stable before executing a WFI instruction to place the MCU in Software Standby or Deep Software Standby mode
- When a transition to Software Standby or Deep Software Standby mode is to follow the setting to stop the MOCO clock, wait for at least 3 MOCO clock cycles before executing the WFI instruction.

Writing 1 to MCSTP is prohibited under the following condition:

- SCKSCR.CKSEL[2:0] = 001b (system clock source = MOCO).

Writing 1 to the MCSTP bit (stopping the MOCO) is prohibited if oscillation stop detection is enabled in the Oscillation Stop Detection Control Register (OSTDCR.OSTDE).

## 8.2.13 FLLCR1 : FLL Control Register1

Base address: SYSC = 0x4001_E000

Offset address: 0x039

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | FLLEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FLLEN | FLL Enable<br>0: FLL function is disabled<br>1: FLL function is enabled. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: HOCO must be stopped (HOCOCR.HCSTP = 1) before FLLCR1.FLLEN is modified.
Note: SOSC must be operating with stabilization while FLL is enabled (FLLCR1.FLLEN = 1).
Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The FLLCR1 register controls the FLL function of the HOCO.

### FLLEN bit (FLL Enable)

This bit enables or disables the FLL function of the HOCO.

If FLL is enabled, the frequency accuracy is guaranteed after FLL is stabilized. The FLL stabilization can be checked by the CAC frequency measurement, but it must be executed after HOCO stabilization.

In addition, you must disable FLL by setting the FLLEN bit to 0 before transitioning to Software Standby mode.

Table 8.4 show an example flow of the FLL setting for each case.

**Table 8.4 FLL setting flow**

| Step | | Operation |
|---|---|---|
| After reset release/Deep Software Standby cancellation | 1 | Start (After reset release / Deep Software Standby cancellation) |
| | 2 | FLL setting (FLLCR2.FLLCNTL) |
| | 3 | Enable FLL (FLLCR1.FLLEN = 1)<br>Note: SOSC must be running with the oscillation stabilization. |
| | 4 | Enable HOCO (HOCOCR.HCSTP = 0) |
| | 5 | Wait for the FLL stabilization ($t_{FLLWT}$) |
| | 6 | Check the HOCO stabilization (OSCSF.HOCOSF = 1) |
| | 7 | End (HOCO can be used.) |
| Software standby transition/ cancellation | 1 | Start (FLL is being used.) |
| | 2 | Stop HOCO (HOCOCR.HCSTP = 1)<br>Note: If HOCO is used as the system clock or the PLL reference clock, these clock source must be changed to another clock before HOCO is stopped. |
| | 3 | Disable FLL (FLLCR1.FLLEN = 0) |
| | 4 | WFI instruction |
| | 5 | Software standby mode |
| | 6 | Software standby cancellation |
| | 7 | Enable FLL (FLLCR1.FLLEN = 1) |
| | 8 | Enable HOCO (HOCOCR.HCSTP = 0) |
| | 9 | Wait for the FLL stabilization ($t_{FLLWT}$) |
| | 10 | Check the HOCO stabilization (OSCSF.HOCOSF = 1) |
| | 11 | End (HOCO can be used.) |

## 8.2.14 FLLCR2 : FLL Control Register2

Base address: SYSC = 0x4001_E000

Offset address: 0x03A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | FLLCNTL[10:0] | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 10:0 | FLLCNTL[10:0] | FLL Multiplication Control<br>When OFS1.HOCOFRQ[1:0] is 00b (16MHz), these bits must be set to 0x1E9.<br>When OFS1.HOCOFRQ[1:0] is 01b (18MHz), these bits must be set to 0x226.<br>When OFS1.HOCOFRQ[1:0] is 10b (20MHz), these bits must be set to 0x263.<br>Other settings are prohibited. | R/W |
| 15:11 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The FLLCR2 register controls the FLL function of the HOCO.

### FLLCNTL[10:0] bits (FLL Multiplication Control)

These bits select the multiplication ratio of the FLL reference clock.

These bits must be set before FLL is enabled (FLLCR1.FLLEN=1).

## 8.2.15 OSCSF : Oscillation Stabilization Flag Register

Base address: SYSC = 0x4001_E000

Offset address: 0x03C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | PLL2SF | PLLSF | — | MOSCSF | — | — | HOCOSF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | HOCOSF | HOCO Clock Oscillation Stabilization Flag<br>0: The HOCO clock is stopped or is not yet stable<br>1: The HOCO clock is stable, so is available for use as the system clock | R |
| 2:1 | — | These bits are read as 0. | R |
| 3 | MOSCSF | Main Clock Oscillation Stabilization Flag<br>0: The main clock oscillator is stopped (MOSTP = 1) or is not yet stable*2<br>1: The main clock oscillator is stable, so is available for use as the system clock | R |
| 4 | — | This bit is read as 0. | R |
| 5 | PLLSF | PLL Clock Oscillation Stabilization Flag<br>0: The PLL clock is stopped, or oscillation of the PLL clock is not stable yet<br>1: The PLL clock is stable, so is available for use as the system clock | R |
| 6 | PLL2SF | PLL2 Clock Oscillation Stabilization Flag<br>0: The PLL2 clock is stopped, or oscillation of the PLL2 clock is not stable yet<br>1: The PLL2 clock is stable | R |
| 7 | — | These bits are read as 0. | R |

Note 1. The value after reset depends on the OFS1.HOCOEN setting.
When OFS1.HOCOEN = 1 (disable HOCO), the value after reset of HOCOSF is 0.
When OFS1.HOCOEN = 0 (enable HOCO), the HOCOSF value is set to 0 immediately after reset is released, and the HOCOSF value is set to 1 after the HOCO oscillation stabilization wait time elapses.

Note 2. This is true when an appropriate value is set in the Wait Control register for the main clock oscillator. If the wait time value is not sufficient, the oscillation stabilization flag is set to 1 and supply of the clock signal to the internal circuits starts before oscillation is stable.

This register is not controlled by CGFSAR register.

The OSCSF register contains flags to indicate the operating status of the counters in the oscillation stabilization wait circuits for the individual oscillators. After oscillation starts, these counters measure the wait time until each oscillator output clock is supplied to the internal circuits. An overflow of a counter indicates that the clock supply is stable and available for the associated circuit.

### HOCOSF flag (HOCO Clock Oscillation Stabilization Flag)

The HOCOSF flag indicates the operating status of the counter that measures the wait time for the high-speed clock oscillator (HOCO). When OFS1.HOCOEN is set to 0, confirm that OSCSF.HOCOSF is set to 1 before using the HOCO clock.

[Setting condition]

- When the HOCO clock is stopped and the HOCOCR.HCSTP bit is set to 0, and then the HOCO oscillation stabilization time is counted by the LOCO clock and supply of the HOCO clock within the MCU is started. For the HOCO oscillation stabilization time, see section 47, Electrical Characteristics.

[Clearing condition]

- When the HOCO clock is operating and then is deactivated because the HOCOCR.HCSTP bit is set to 1.

### MOSCSF flag (Main Clock Oscillation Stabilization Flag)

The MOSCSF flag indicates the operating status of the counter that measures the wait time for the main clock oscillator.

[Setting condition]

- When the main clock oscillator is stopped and the MOSCCR.MOSTP bit is set to 0, and then the number of LOCO clock cycles corresponding to the setting of the MOSCWTCR register is counted and supply of the main clock within the MCU is started.

[Clearing condition]

- When the main clock oscillator is operating and then is deactivated because the MOSCCR.MOSTP bit is set to 1.

### PLLSF flag (PLL Clock Oscillation Stabilization Flag)

The PLLSF flag indicates the operating state of the counter that measures the wait time of the PLL.

[Setting condition]

- When the PLL is stopped and the PLLCR.PLLSTP bit is set to 0, and then the PLL oscillation stabilization time is counted by the LOCO clock and supply of the PLL clock within the MCU is started. If oscillation by the PLL clock source is not stable when the PLLCR.PLLSTP bit is set to 0, counting of the LOCO cycles continues even after the PLL clock source oscillation is stabilized. For the PLL oscillation stabilization time, see section 47, Electrical Characteristics.

[Clearing condition]

- When the PLL is operating and then is deactivated because the PLLCR.PLLSTP bit is set to 1.

### PLL2SF flag (PLL2 Clock Oscillation Stabilization Flag)

The PLL2SF flag indicates the operating state of the counter that measures the wait time of the PLL2.

[Setting condition]

- When the PLL2 is stopped and the PLL2CR.PLL2STP bit is set to 0, and then the PLL2 oscillation stabilization time is counted by the LOCO clock and supply of the PLL2 clock within the MCU is started. If oscillation by the PLL2 clock source is not stable when the PLL2CR.PLL2STP bit is set to 0, counting of the LOCO cycles continues even after the PLL2 clock source oscillation is stabilized. For the PLL2 oscillation stabilization time, see section 47, Electrical Characteristics.

[Clearing condition]

- When the PLL2 is operating and then is deactivated because the PLL2CR.PLL2STP bit is set to 1.

## 8.2.16 OSTDCR : Oscillation Stop Detection Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x040

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | OSTDE | — | — | — | — | — | — | OSTDIE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | OSTDIE | Oscillation Stop Detection Interrupt Enable<br>0: Disable oscillation stop detection interrupt (do not notify the POEG)<br>1: Enable oscillation stop detection interrupt (notify the POEG) | R/W |
| 6:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | OSTDE | Oscillation Stop Detection Function Enable<br>0: Disable oscillation stop detection function<br>1: Enable oscillation stop detection function | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The OSTDCR register controls the oscillation stop detection function.

### OSTDIE bit (Oscillation Stop Detection Interrupt Enable)

The OSTDIE bit enables the oscillation stop detection function interrupt. It also controls whether oscillation stop detection is reported to the POEG.

If the Oscillation Stop Detection flag in the Oscillation Stop Detection Status Register (OSTDSR.OSTDF) requires clearing, set the OSTDIE bit to 0 before clearing OSTDF. Wait for at least 2 PCLKB cycles before setting the OSTDIE bit to 1. By reading the I/O register whose access cycle number is defined by PCLKB, it is possible to secure waiting time of 2 or more cycles of PCLKB.

### OSTDE bit (Oscillation Stop Detection Function Enable)

The OSTDE bit enables the oscillation stop detection function.

When the OSTDE bit is 1 (enabled), the MOCO stop bit (MOCOCR.MCSTP) is set to 0 and the MOCO operation starts. The MOCO clock cannot be stopped while the oscillation stop detection function is enabled. Writing 1 to the MOCOCR.MCSTP bit (MOCO stopped) is invalid.

When the Oscillation Stop Detection flag in the Oscillation Stop Detection Status Register (OSTDSR.OSTDF) is 1 (main clock oscillation stop detected), writing 0 to the OSTDE bit is invalid.

The OSTDE bit must be set to 0 before transitioning to Software Standby or Deep Software Standby mode. To transition to Software Standby or Deep Software Standby mode, first set the OSTDE bit to 0, then execute the WFI instruction.

The following restrictions apply when using the oscillation stop detection function:
In low-speed mode, selecting division by 1, 2, 4, 8 for ICLK, FCLK, PCLKA, PCLKB, PCLKC, and PCLKD is prohibited.

## 8.2.17 OSTDSR : Oscillation Stop Detection Status Register

Base address: SYSC = 0x4001_E000

Offset address: 0x041

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | OSTDF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | OSTDF | Oscillation Stop Detection Flag<br>0: Main clock oscillation stop not detected<br>1: Main clock oscillation stop detected | R/W*1 |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. This bit can only be set to 0. This bit is cleared to 0 by writing 0 after reading it as 1.

The OSTDSR register indicates the stop detection status of the main clock oscillator.

### OSTDF flag (Oscillation Stop Detection Flag)

The OSTDF flag indicates the main clock oscillator status. When this flag is 1, it indicates that the main clock oscillation stop was detected. After this stop is detected, the OSTDF flag is not set to 0 even when the main clock oscillation is restarted. The OSTDF bit is cleared to 0 by writing 0 after reading it as 1.

At least 3 ICLK cycles of wait time are required between writing 0 to OSTDF and reading it as 0. If the OSTDF flag is set to 0 when the main clock oscillation is stopped, the OSTDF flag becomes 0 then returns to 1.

The OSTDF flag cannot be set to 0 under the following conditions:

- SCKSCR.CKSEL[2:0] = 011b (system clock source = MOSC).
- PLLCCR.PLSRCSEL = 0 (PLL source clock = MOSC) and SCKSCR.CKSEL[2:0] = 101b (System clock source = PLL)

The OSTDF flag must be set to 0 after switching the clock source to sources other than the main clock oscillator and PLL.

[Setting condition]

- The main clock oscillator is stopped when OSTDCR.OSTDE = 1 (oscillation stop detection function enabled).

[Clearing condition]

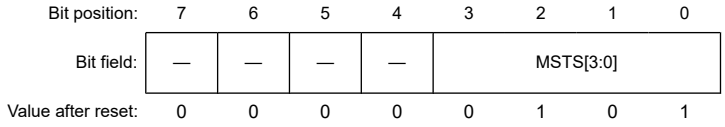
When writing 0 after reading 1. However, it will not be 0 under the following conditions.

- SCKSCR.CKSEL[2:0] = 011b (system clock source = MOSC).
- PLLCCR.PLSRCSEL = 0 (PLL source clock = MOSC) and SCKSCR.CKSEL[2:0] = 101b (System clock source =PLL).

## 8.2.18 MOSCWTCR : Main Clock Oscillator Wait Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x0A2




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | MSTS[3:0] | Main Clock Oscillator Wait Time Setting<br>0x0: Wait time = 3 cycles (11.4 µs)<br>0x1: Wait time = 35 cycles (133.5 µs)<br>0x2: Wait time = 67 cycles (255.6 µs)<br>0x3: Wait time = 131 cycles (499.7 µs)<br>0x4: Wait time = 259 cycles (988.0 µs)<br>0x5: Wait time = 547 cycles (2086.6 µs)<br>0x6: Wait time = 1059 cycles (4039.8 µs)<br>0x7: Wait time = 2147 cycles (8190.2 µs)<br>0x8: Wait time = 4291 cycles (16368.9 µs)<br>0x9: Wait time = 8163 cycles (31139.4 µs)<br>Others: Setting prohibited | R/W |
| 7:4 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### MSTS[3:0] bits (Main Clock Oscillator Wait Time Setting)

The MSTS[3:0] bits specify the oscillation stabilization wait time for the main clock oscillator.

Set the main clock oscillation stabilization time to a period longer than or equal to the stabilization time recommended by the oscillator manufacturer. When the main clock is input externally, set these bits to 0x0 because the oscillation stabilization time is not required.

The wait time set in these bits is counted using: 1 cycle (µs) = $1/(f_{LOCO}[MHz] \times 8) = 1/(0.032768 \times 8) = 3.81$ (µs) (min.)

The LOCO clock automatically oscillates when necessary, regardless of the value of the LOCO.LCSTP bit. After the

specified wait time elapses, supply of the main clock starts internally in the MCU, and the OSCSF.MOSCSF flag is set to 1. If the specified wait time is short, supply of the main clock starts before oscillation of the clock becomes stable.

Only rewrite the MOSCWTCR register when the MOSCCR.MOSTP bit is 1 and the OSCSF.MOSCSF flag is 0. Do not rewrite this register under any other conditions.

## 8.2.19 MOMCR : Main Clock Oscillator Mode Oscillation Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x413

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | MOSEL | MODRV[1:0] | | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5:4 | MODRV[1:0] | Main Clock Oscillator Drive Capability 0 Switching<br>0 0: 20 MHz to 24 MHz<br>0 1: 16 MHz to 20 MHz<br>1 0: 8 MHz to 16 MHz<br>1 1: 8 MHz | R/W |
| 6 | MOSEL | Main Clock Oscillator Switching<br>0: Resonator<br>1: External clock input | R/W |
| 7 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: The EXTAL/XTAL pins are also used as ports. In the initial state, the pin is set as a port.
Note: The MOSCCR.MOSTP bit must be 1 (MOSC is stopped) before changing this register.
Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

### MODRV[1:0] bit (Main Clock Oscillator Drive Capability 0 Switching)

The MODRV[1:0] bit switches the drive capability of the main clock oscillator.

### MOSEL bit (Main Clock Oscillator Switching)

The MOSEL bit switches the source for the main clock oscillator.

## 8.2.20 SOMCR : Sub-Clock Oscillator Mode Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x481

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | SODRV | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 1 | SODRV | Sub-Clock Oscillator Drive Capability Switching<br>0: Standard<br>1: Low | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The SOMCR register must be modified when SOSCCR.SOSTP is 1 (SOSC is stopped).

### SODRV bits (Sub-Clock Oscillator Drive Capability Switching)

The SODRV bits switch the drive capability of the sub-clock oscillator. SODRV is undefined at the first Power up, but value after reset of SOSCCR.SOSTP is 0 (SOSC is operated). And therefore, please set the SOSC as follows when the first Power up:

1. Set the SOSCCR.SOSTP to 1 (SOSC is stopped)
2. Set this bit to a value corresponding to the using capacitor.
3. Clear the SOSCCR.SOSTP to 0 (SOSC is operated)

## 8.2.21 CKOCR : Clock Out Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x03E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CKOEN | CKODIV[2:0] | | | — | CKOSEL[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | CKOSEL[2:0] | Clock Out Source Select<br>0 0 0: HOCO (value after reset)<br>0 0 1: MOCO<br>0 1 0: LOCO<br>0 1 1: MOSC<br>1 0 0: SOSC<br>1 0 1: Setting prohibited<br>Others: Setting prohibited | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6:4 | CKODIV[2:0] | Clock Output Frequency Division Ratio<br>0 0 0: × 1/1<br>0 0 1: × 1/2<br>0 1 0: × 1/4<br>0 1 1: × 1/8<br>1 0 0: × 1/16<br>1 0 1: × 1/32<br>1 1 0: × 1/64<br>1 1 1: × 1/128 | R/W |
| 7 | CKOEN | Clock Out Enable<br>0: Disable clock out<br>1: Enable clock out | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### CKOSEL[2:0] bits (Clock Out Source Select)

The CKOSEL[2:0] bits select the source of the clock to be output from the CLKOUT pin. When changing the clock source, set the CKOEN bit to 0.

### CKODIV[2:0] bits (Clock Output Frequency Division Ratio)

The CKODIV[2:0] bits specify the clock division ratio. Set the CKOEN bit to 0 when changing the division ratio.

### CKOEN bit (Clock Out Enable)

The CKOEN bit enables output from the CLKOUT pin.

When this bit is set to 1, the selected clock is output. When this bit is set to 0, low is output. When changing this bit, confirm that the clock out source clock selected in the CKOSEL[2:0] bits is stable. Otherwise, a glitch might be generated in the output.

Clear this bit before entering Software Standby or Deep Software Standby mode if the selecting clock out source clock is stopped in that mode.

## 8.2.22 LOCOUTCR : LOCO User Trimming Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x492

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | LOCOUTRM[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | LOCOUTRM[7:0] | LOCO User Trimming<br>0x80: -128<br>0x81: -127<br>⋮<br>0xFF: -1<br>0x00: Center Code<br>0x01: +1<br>⋮<br>0x7E: +126<br>0x7F: +127 | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The LOCOUTCR register is added to the original LOCO trimming data.

MCU operation is not guaranteed when LOCOUTCR is set to a value that causes the LOCO frequency to be outside of the specification range. When LOCOUTCR is modified, the frequency stabilization time corresponds to the frequency stabilization time at the start of MCU operation. When the ratio of the LOCO frequency and the other oscillation frequency is an integer value, changing the LOCOUTCR value is prohibited.

Changing LOCOUTCR during RTC operation is prohibited.

## 8.2.23 MOCOUTCR : MOCO User Trimming Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x061

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | MOCOUTRM[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | MOCOUTRM[7:0] | MOCO User Trimming<br>0x80: -128<br>0x81: -127<br>⋮<br>0xFF: -1<br>0x00: Center Code<br>0x01: +1<br>⋮<br>0x7E: +126<br>0x7F: +127 | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The MOCOUTCR register is added to the original MOCO trimming data.

MCU operation is not guaranteed when MOCOUTCR is set to a value that causes the MOCO frequency to be outside of the specification range. When MOCOUTCR is modified, the frequency stabilization wait time corresponds to the frequency stabilization wait time at the start of the MCU operation. When the ratio of the MOCO frequency and the other oscillation frequency is an integer value, changing the MOCOUTCR value is prohibited.

## 8.2.24 HOCOUTCR : HOCO User Trimming Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x062

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | HOCOUTRM[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | HOCOUTRM[7:0] | HOCO User Trimming<br>0x80: -128<br>0x81: -127<br>⋮<br>0xFF: -1<br>0x00: Center Code<br>0x01: +1<br>⋮<br>0x7E: +126<br>0x7F: +127 | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The HOCOUTCR register is added to the original HOCO trimming data.

MCU operation is not guaranteed when HOCOUTCR is set to a value that causes the HOCO frequency to be outside of the specification range. When HOCOUTCR is modified, the frequency stabilization wait time corresponds to the frequency stabilization wait time at the start of the MCU operation.

These bits must be 0x00 when FLL is enabled (FLLCR1.FLLEN = 1).

## 8.2.25 USBCKDIVCR : USB Clock Division Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x06C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | USBCKDIV[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | USBCKDIV[2:0] | USB Clock (USBCLK) Division Select<br>0 1 0: /4<br>1 0 1: /3<br>1 1 0: /5<br>Others: Setting prohibited. | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The USBCKDIVCR register controls the USB clock.

### USBCKDIV[2:0] bits (USB Clock (USBCLK) Division Select)

These bits select the frequency of the USB clock (USBCLK) and must be modified when USBCKCR.USBCKSRDY = 1.

## 8.2.26 USBCKCR : USB Clock Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x074

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | USBCKSRDY | USBCKSREQ | — | — | — | USBCKSEL[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | USBCKSEL[2:0] | USB Clock (USBCLK) Source Select<br>1 0 1: PLL<br>1 1 0: PLL2<br>Others: Setting prohibited. | R/W |
| 5:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | USBCKSREQ | USB Clock (USBCLK) Switching Request<br>0: No request<br>1: Request switching. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | USBCKSRDY | USB Clock (USBCLK) Switching Ready state flag<br>0: Impossible to Switch<br>1: Possible to Switch | R |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

The USBCKCR register controls the USB clock.

When switching the clock source, ensure that the clock before the switch and the clock after the switch generate stable output. To change the set value of USBCKDIVCR.USBCKDIV[2:0] and USBCKSEL[2:0], use the following procedure:

1. Write 1 to USBCKSREQ.
2. Poll until USBCKSRDY is read as 1. While USBCKSRDY = 1, no clock is output to USBCLK.
3. Write to USBCKDIVCR.USBCKDIV[2:0] and USBCKSEL[2:0].
4. Write 0 to USBCKSREQ.
5. Poll until USBCKSRDY is read as 0.
6. When USBCKSRDY becomes 0, USBCLK starts to output. Clock switching is complete.

When transitioning to Software Standby or Deep Software Standby mode, do not execute the WFI instruction while performing clock switching. That is, do not execute the WFI instruction when USBCKSREQ = 1 and USBCKSRDY = 0, or when USBCKSREQ = 0 and USBCKSRDY = 1.

#### USBCKSEL[2:0] bits (USB Clock (USBCLK) Source Select)

These bits select the clock source of the USB clock (USBCLK) and must be modified when USBCKCR.USBCKSRDY = 1.

#### USBCKSREQ bit (USB Clock (USBCLK) Switching Request)

This bit selects the USBCLK switching request.

#### USBCKSRDY flag (USB Clock (USBCLK) Switching Ready state flag)

This flag indicates the state of switching ready for the USBCLK. When USBCKSRDY = 1, no clock is output to USBCLK.

### 8.2.27 TRCKCR : Trace Clock Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x03F

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TRCKEN | — | — | — | TRCK[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | TRCK[3:0] | Trace Clock operating frequency select<br>0x0: /1<br>0x1: /2 (value after reset)<br>0x2: /4<br>Others: Setting prohibited | R/W |
| 6:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | TRCKEN | Trace Clock operating Enable<br>0: Stop<br>1: Operation enable | R/W |

Note: Set the PRCR.PRC0 bit to 1 (write enabled) before rewriting this register.

Trace Clock Control Register controls switching the trace clock.

TRCKCR can be written only when the debugger is connected (DBGSTR.CDBGPWRUPREQ = 1).

When the debugger is not connected, it is not necessary to change TRCK[3:0] even if the TRCLK frequency is higher than the maximum value of the specification.

Change the TRCLK frequency in the state of TRCKEN = 0.

Factor of the initialization of TRCKCR register is all resets.

## 8.3 Main Clock Oscillator

To supply the clock signal to the main clock oscillator, use one of the following ways:

- Connect an oscillator
- Connect the input of an external clock signal.

### 8.3.1 Connecting a Crystal Resonator

Figure 8.4 shows an example of connecting a crystal resonator. A damping resistor (Rd) can be added, if required.

Because the resistor values vary according to the resonator and the oscillation drive capability, use values recommended by the resonator manufacturer. If the manufacturer recommends using an external feedback resistor (Rf), insert an Rf between EXTAL and XTAL by following the instructions.

When connecting a resonator to supply the clock, the frequency of the resonator must be in the frequency range of the resonator for the main clock oscillator as described in Table 8.1.

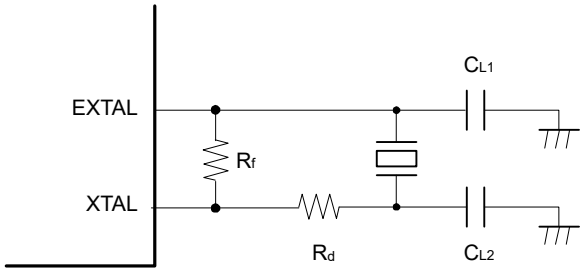



**Figure 8.4 Example of crystal resonator connection**

Figure 8.5 shows an equivalent circuit of the crystal resonator.

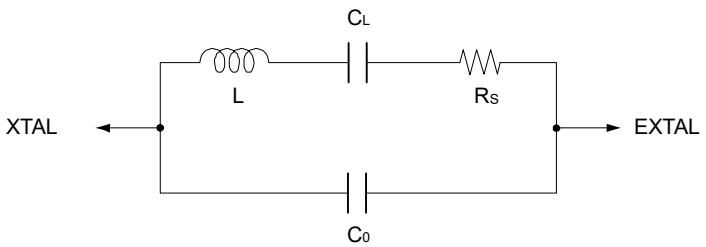



**Figure 8.5 Equivalent circuit of the crystal resonator**

### 8.3.2 External Clock Input

Figure 8.6 shows an example of connecting an external clock input. To operate the oscillator with an external clock signal, set the MOMCR.MOSEL bit to 1. The XTAL pin becomes high impedance.

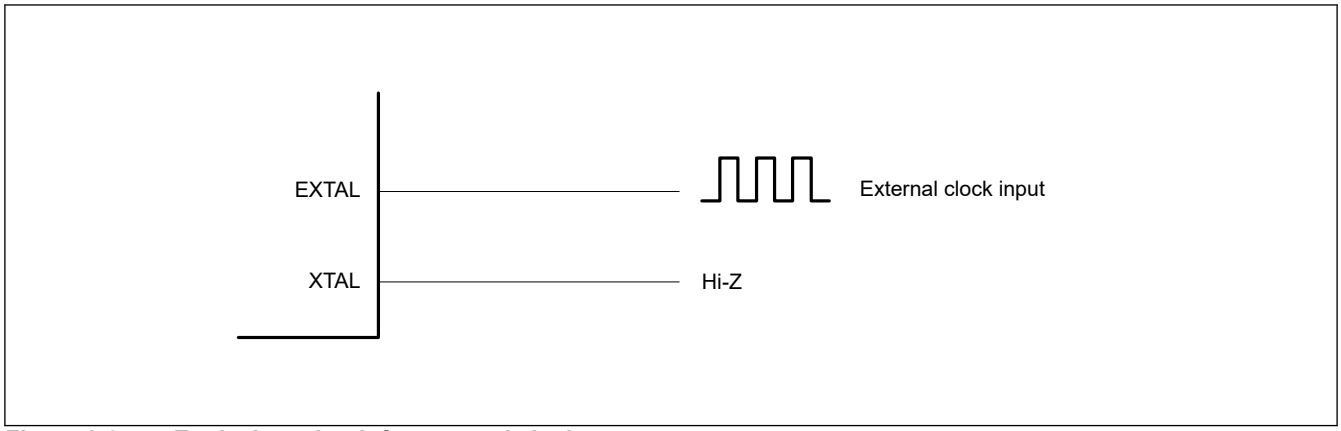



**Figure 8.6 Equivalent circuit for external clock**

### 8.3.3 Notes on External Clock Input

The frequency of the external clock input can only be changed when the main clock oscillator is stopped. Do not change the frequency of the external clock input when the setting of the Main Clock Oscillator Stop bit (MOSCCR.MOSTP) is 0.

## 8.4 Sub-Clock Oscillator

The only way of supplying a clock signal to the sub-clock oscillator is by connecting a crystal oscillator.

### 8.4.1 Connecting a 32.768-kHz Crystal Resonator

To supply a clock to the sub-clock oscillator, connect a 32.768-kHz crystal resonator as shown in Figure 8.7. A damping resistor (Rd) can be added, if necessary. Because the resistor values vary according to the resonator and the oscillation drive capability, use values recommended by the resonator manufacturer. If the resonator manufacturer recommends the use of an external feedback resistor (Rf), insert an Rf between XCIN and XCOUT by following the instructions. When connecting a resonator to supply the clock, the frequency of the resonator must be in the frequency range of the resonator for the sub-clock oscillator as described in Table 8.1.

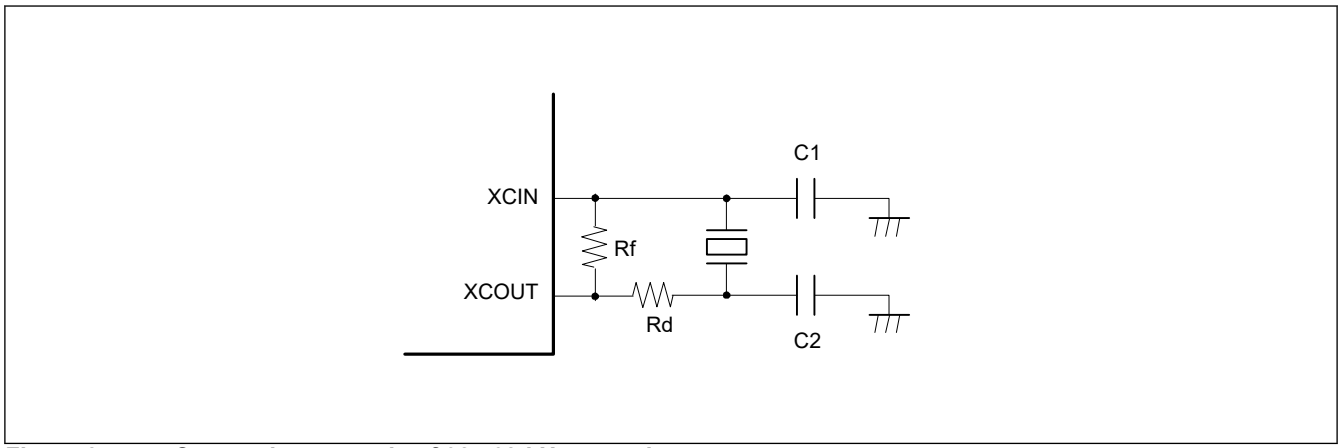



**Figure 8.7 Connection example of 32.768-kHz crystal resonator**

Figure 8.8 shows an equivalent circuit for the 32.768-kHz crystal resonator.

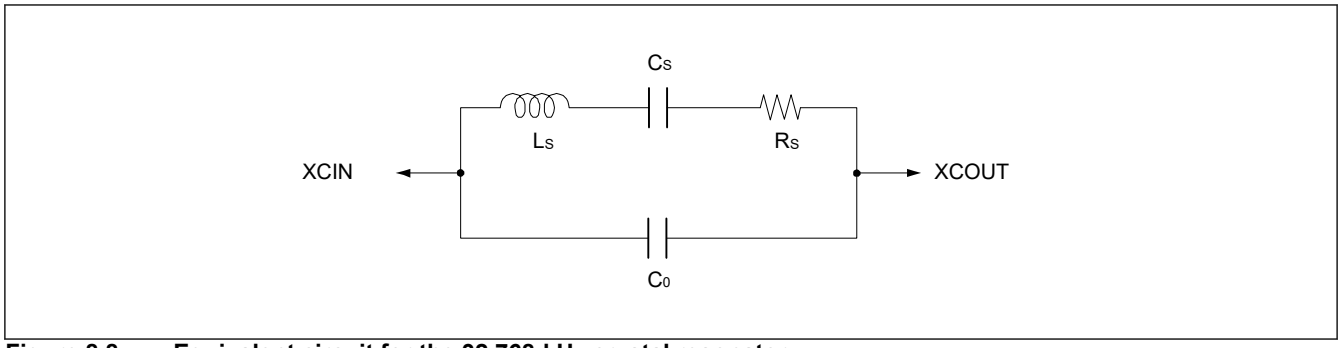



**Figure 8.8 Equivalent circuit for the 32.768-kHz crystal resonator**

### 8.4.2 Pin Handling When the Sub-Clock Oscillator Is Not Used

When the sub-clock oscillator is not in use, connect the XCIN pin to VSS through a resistor (to pull VSS down) and leave the XCOUT pin open as shown in Figure 8.9. In addition, if an oscillator is not connected, set the Sub-Clock Oscillator Stop bit (SOSCCR.SOSTP) to 1 to stop the oscillator.

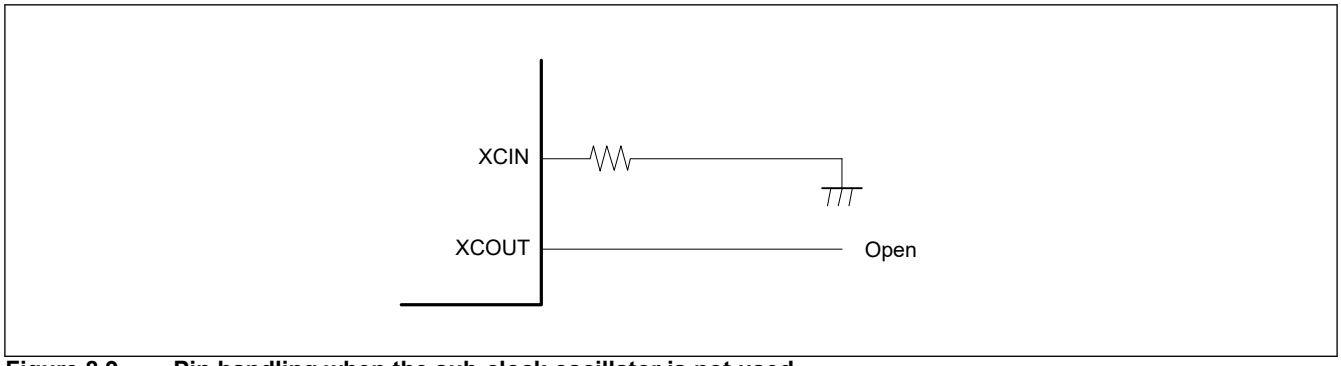



**Figure 8.9 Pin handling when the sub-clock oscillator is not used**

## 8.5 Oscillation Stop Detection Function

### 8.5.1 Oscillation Stop Detection and Operation after Detection

The oscillation stop detection function detects the main clock oscillator stop. When oscillation stop is detected, the system clock switches as follows:

- If an oscillation stop is detected with SCKSCR.CKSEL[2:0] = 011b (system clock source = MOSC), the system clock source switches to the MOCO clock.
- If an oscillation stop is detected with PLLCCR.PLSRCSEL = 0 (PLL source clock = MOSC) and SCKSCR.CKSEL[2:0] = 101b (system clock source = PLL), PLL clock remains the system clock source. However, the frequency becomes a free-running oscillation frequency.

An oscillation stop detection interrupt request can be generated when an oscillation stop is detected. In addition, the General PWM Timer (GPT) output can be forced to a high-impedance state on detection.

The main clock oscillation stop is detected when the input clock remains at 0 or 1 for a certain period, for example, when a malfunction occurs in the main clock oscillator. See section 47, Electrical Characteristics.

Switching between the main clock oscillator and the MOCO clock or between the PLL clock and PLL free-running clock is controlled by the Oscillation Stop Detection Flag (OSTDSR.OSTDF).

OSTDF controls the switched clock as follows:

- When SCKSCR.CKSEL[2:0] = 011b (system clock source = MOSC):
  - When OSTDF changes from 0 to 1, the clock source switches to the MOCO clock.
  - When OSTDF changes from 1 to 0, the clock source switches back to MOSC.
- When PLLCCR.PLSRCSEL = 0 (PLL source clock = MOSC) and SCKSCR.CKSEL[2:0] = 101b (System clock source = PLL):

- When OSTDF changes 0 to 1, the clock source switches to the PLL free-running oscillation clock.
- When OSTDF changes 1 to 0, the clock source switches back to PLL.

To switch the clock source to the main clock or PLL clock again after the oscillation stop detection, set the CKSEL[2:0] bits to a clock source other than the main clock or PLL clock and clear the OSTDF flag to 0. Also, check that the OSTDF flag is not 1, then set the CKSEL[2:0] bits to the main clock or PLL clock after the specified oscillation stabilization time elapses.

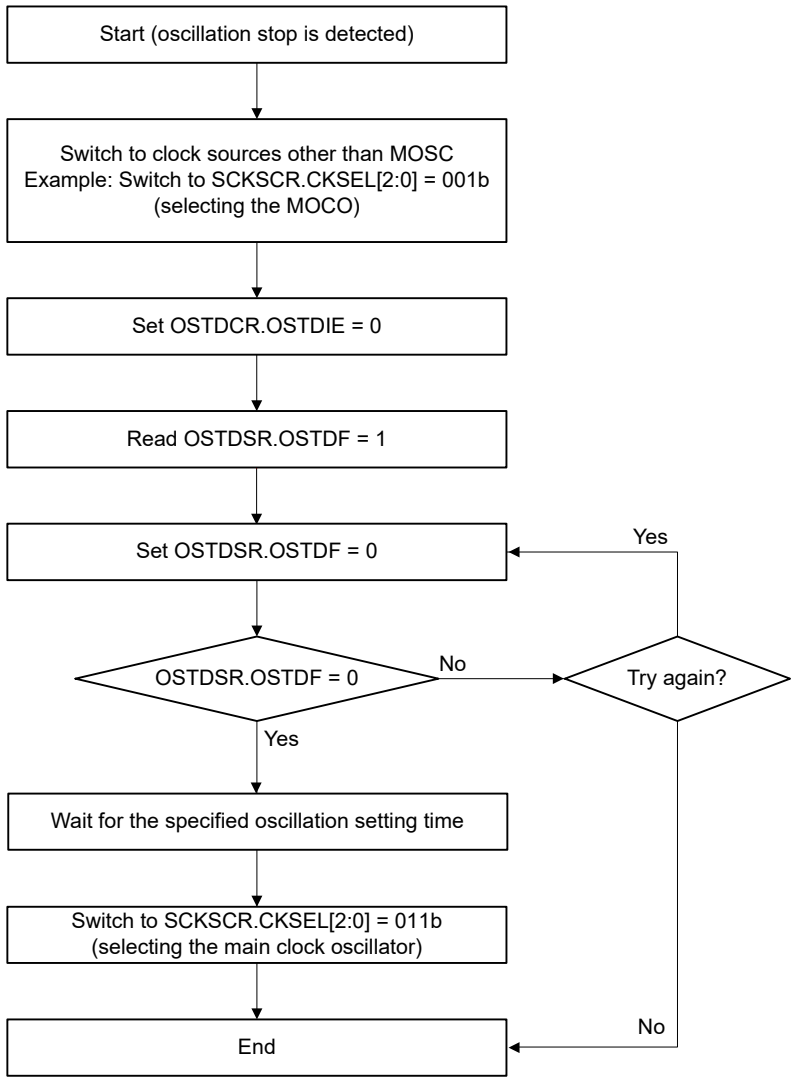
After a reset release, the main clock oscillator is stopped and the oscillation stop detection function is disabled. To enable the oscillation stop detection function, activate the main clock oscillator and write 1 to the oscillation stop detection function enable bit (OSTDCR.OSTDE) after a specified oscillation stabilization time elapses.

The oscillation stop detection function detects when the main clock is stopped by an external cause. Therefore, the oscillation stop detection function must be disabled before the main clock oscillator is stopped by software or a transition is made to Software Standby or Deep Software Standby mode.

The oscillation stop detection function switches all clocks that can be selected as the MOSC clock except CLKOUT to the MOCO (when system clock is MOSC) or PLL free-running (when system clock is PLL).

The system clock (ICLK) frequency during the MOCO(when system clock is MOSC) or PLL free-running (when system clock is PLL) operation is specified by the MOCO oscillation frequency and the division ratio set by the system clock select bits (SCKDIVCR.ICK[2:0])

Example of returning when CKSEL[2:0] = 011b (selecting the main clock oscillator) after an oscillation stop is detected




Note: On return from the oscillation-stopped state, the factor responsible for stopping the main clock oscillation circuit must be removed from the system to allow oscillation to resume.

**Figure 8.10 Flow of recovery on detection of oscillator stop**

## 8.5.2 Oscillation Stop Detection Interrupts

An oscillation stop detection interrupt (MOSC_STOP) is generated when the Oscillation Stop Detection Flag (OSTDSR.OSTDF) is 1 and the Oscillation Stop Detection Interrupt Enable bit in the Oscillation Stop Detection Control Register (OSTDCR.OSTDIE) is 1 (enabled). The Port Output Enable for GPT (POEG) is notified of the main clock oscillator stop. On receiving the notification, the POEG sets the Oscillation Stop Detection Flag in the POEG Group n Setting Register (POEGGn.OSTPF) to 1 (n = A, B, C, D).

After the oscillation stop is detected, wait at least 10 PCLKB clock cycles before writing to the POEGGn.OSTPF flag. When the OSTDSR.OSTDF flag requires clearing, do so after clearing the Oscillation Stop Detection Interrupt Enable bit in the Oscillation Stop Detection Control Register (OSTDCR.OSTDIE). Wait at least 2 PCLKB clock cycles before setting the OSTDCR.OSTDIE bit to 1 again. A longer PCLKB wait time might be required, depending on the number of cycles required to read a given I/O register.

The oscillation stop detection interrupt is a non-maskable interrupt. Because non-maskable interrupts are disabled in the initial state after a reset release, enable non-maskable interrupts through software before using oscillation stop detection interrupts. For details, see section 13, Interrupt Controller Unit (ICU).

## 8.6 PLL Circuit

The PLL circuit has a function to multiply the frequency from the oscillator.

## 8.7 Internal Clock

Clock sources for the internal clock signals include:

- Main clock
- Sub-clock
- HOCO clock
- MOCO clock
- LOCO clock
- PLL clock
- PLL2 clock
- IWDT-dedicated clock
- JTAG clock

The following internal clocks are produced from these sources.

- Operating clock of the CPU, DMAC, DTC, Flash, and RAM: System clock (ICLK)
- Operating clocks of peripheral modules: Peripheral module clocks (PCLKA, PCLKB, PCLKC, PCLKD)
- Operating clock of the FlashIF: FlashIF clock (FCLK)
- Operating clock for the USBFS clock (USBCLK)
- Operating clock for the CAN: CAN clock (CANMCLK)
- Operating clocks for the CAC: CAC clock (CACCLK)
- Operating clock for the RTC: RTC-dedicated LOCO clock (RTCLCLK)
- Operating clock for the RTC: RTC-dedicated sub clock (RTCSCLK)
- Operating clock for the IWDT: IWDT-dedicated clock (IWDTCLK)
- Operating clock for the AGT: AGT-dedicated LOCO clock (AGTLCLK)
- Operating clock for the AGT: AGT-dedicated sub clock (AGTSCLK)
- Operating clock for the Systick Timer: Systick Timer-dedicated clock (SYSTICCLK)
- Clock for external pin output: Clock/Buzzer output clock(CLKOUT)
- Operating clock for the JTAG: JTAG clock (JTAGTCK)

For details on the registers used to set the frequencies of the internal clocks, see section 8.7.1. System Clock (ICLK) to section 8.7.12. JTAG Clock (JTAGTCK)

If the value of any of these bits is changed, subsequent operation is at the frequency determined by the new value.

### 8.7.1 System Clock (ICLK)

The system clock (ICLK) is the operating clock of the CPU, DMAC, DTC, Flash, and SRAM.

The ICLK frequency is specified by the ICK[2:0] bits in SCKDIVCR, the CKSEL[2:0] bits in SCKSCR, the PLLMUL[5:0] bits and PLIDIV[1:0] bits in PLLCCR, and the HOCOFRQ0[1:0] bits in OFS1.

When the ICLK clock source is switched, the duration of the ICLK clock cycle becomes longer during the clock source transition period. See Figure 8.11 and Figure 8.12.

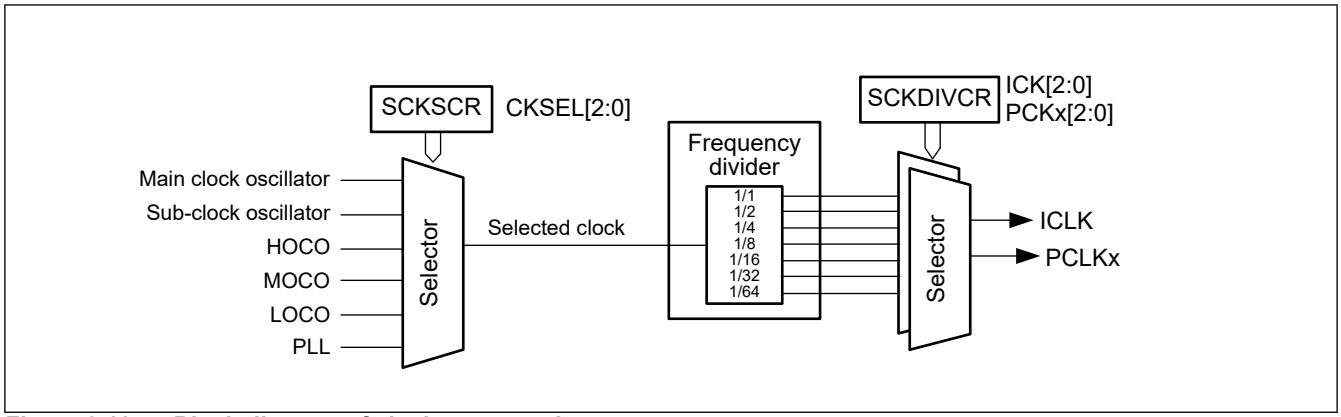



**Figure 8.11 Block diagram of clock source selector**

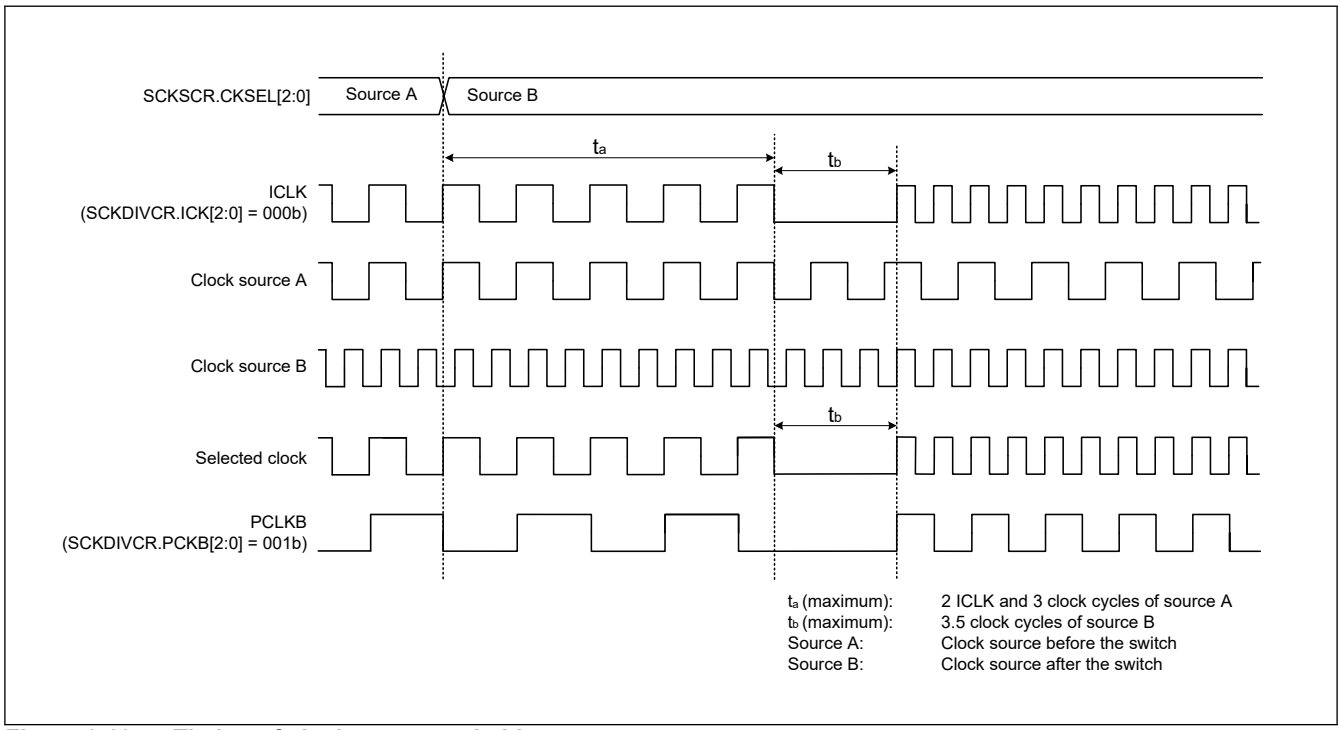



**Figure 8.12 Timing of clock source switching**

## 8.7.2 Peripheral Module Clock (PCLKA, PCLKB, PCLKC, PCLKD)

The peripheral module clocks (PCLKA, PCLKB, PCLKC and PCLKD) are the operating clocks for the peripheral modules.

The frequency of the given clock is specified in the following bits:

- PCKA[2:0], PCKB[2:0], PCKC[2:0] and PCKD[2:0] bits in SCKDIVCR
- CKSEL[2:0] bits in SCKSCR
- PLLMUL[5:0] and PLIDIV[1:0] bits in PLLCCR
- HOCOFRQ0[1:0] bits in OFS1.

When the clock source of the peripheral module clock is switched, the duration of the peripheral module clock cycle becomes longer during the clock source transition period. See Figure 8.11 and Figure 8.12.

## 8.7.3 FlashIF Clock (FCLK)

The flash interface clock (FCLK) is the operating clock for the flash memory interface. In addition to reading from the data flash, FCLK is used for the programming and erasure of the code flash and data flash.

The FCLK frequency is specified in the following bits:

- FCK[2:0] bits in SCKDIVCR
- CKSEL[2:0] bits in SCKSCR
- PLLMUL[5:0] and PLIDIV[1:0] bits in PLLCCR
- HOCOFRQ0[1:0] bits in OFS1.

### 8.7.4 USB Clock (USBCLK)

The USB clock (USBCLK) is the operating clock for the USBFS module.

A 48-MHz clock must be supplied to the USB module. When the USB module is used, setting must be made so that USBCLK is 48 MHz.

The USBCLK frequency is specified in the following bits:

- USBCKSEL[2:0] bits in USBCKCR
- USBCKDIV[2:0] bits in USBCKDIVCR
- PLLMUL[5:0] and PLIDIV[1:0] bits in PLLCCR

### 8.7.5 CAN Clock (CANMCLK)

The CAN clock, CANMCLK, is the operating clock for the CAN module. CANMCLK is generated by the main clock oscillator.

### 8.7.6 CAC Clock (CACCLK)

The CAC clock, CACCLK, is the operating clock for the CAC. CACCLK is generated by the following oscillators:

- Main clock oscillator
- Sub-clock oscillator
- High-speed clock oscillator (HOCO)
- Middle-speed clock oscillator (MOCO)
- Low-speed on-chip oscillator (LOCO)
- IWDT-dedicated on-chip oscillator. (IWDTLOCO)

### 8.7.7 RTC-Dedicated Clock (RTCSCLK, RTCLCLK)

The RTC-dedicated clock (RTCSCLK, RTCLCLK) is the operating clock for the RTC.

RTCSCLK is generated by the sub-clock oscillator, and RTCLCLK is generated by the LOCO clock.

### 8.7.8 IWDT-Dedicated Clock (IWDTCLK)

The IWDT-dedicated clock (IWDTCLK) is the operating clock for the IWDT. IWDTCLK is internally generated by the IWDT-dedicated on-chip oscillator.

### 8.7.9 AGT-Dedicated Clock (AGTSCLK, AGTLCLK)

The AGT-dedicated clocks (AGTSCLK and AGTLCLK) are the operating clocks for the AGT. AGTSCLK is generated by the sub-clock oscillator, and AGTLCLK is generated by the LOCO clock.

### 8.7.10 SysTick Timer-Dedicated Clock (SYSTICCLK)

The SysTick timer-dedicated clock, SYSTICCLK, is the operating clock for the SysTick timer. SYSTICCLK is generated by the LOCO clock.

### 8.7.11 External Pin Output Clock (CLKOUT)

The CLKOUT is output externally from the CLKOUT pin for the clock or buzzer output. The CLKOUT is output to the CLKOUT pin when the CKOCR.CKOEN bit is set to 1. Only change the value in the CKODIV[2:0] bits or CKOSEL[2:0] bits in CKOCR when the CKOCR.CKOEN bit is 0.

The CLKOUT clock frequency is specified in the following bits:

- CKODIV[2:0] bits or CKOSEL[2:0] bits in CKOCR
- HOCOFRQ0[1:0] bits in OFS1

### 8.7.12 JTAG Clock (JTAGTCK)

The JTAG clock (JTAGTCK) is the clock for the JTAG.

JTAGTCK is generated by the JTAG external clock (TCK).

## 8.8 Usage Notes

### 8.8.1 Notes on Clock Generation Circuit

The frequency of the following clocks supplied to each module changes according to the setting of the SCKDIVCR register:

- System clock (ICLK)
- Peripheral module clocks (PCLKA, PCLKB, PCLKC, and PCLKD)
- FlashIF clock (FCLK)

Each frequency must meet the following conditions:

- Each frequency must be selected within the operation-guaranteed range of the operating frequency (f) specified in the AC characteristics. See section 47, Electrical Characteristics.
- The system clock, peripheral module clock must be set according to Table 8.2.

To ensure correct processing after the clock frequency changes, first write to the relevant Clock Control register to change the frequency, then read the value from the register, and finally perform the subsequent processing.

### 8.8.2 Notes on Resonator

Because various resonator characteristics relate closely to your board design, adequate evaluation is required before use. See the resonator connection example in Figure 8.7. The circuit constants for the resonator depend on the resonator to be used and the stray capacitance of the mounting circuit. Therefore, consult the resonator manufacturer when determining the circuit constants. The voltage to be applied between the resonator pins must be within the absolute maximum rating.

### 8.8.3 Notes on Board Design

When using a crystal resonator, place the resonator and its load capacitors as close to the XTAL and EXTAL pins as possible. Other signal lines should be routed away from the oscillation circuit as shown in Figure 8.13 to prevent electromagnetic induction from interfering with correct oscillation. Figure 8.13 shows the case which the main clock oscillator is used. In case of sub-clock oscillator, it is also same as Figure 8.13.

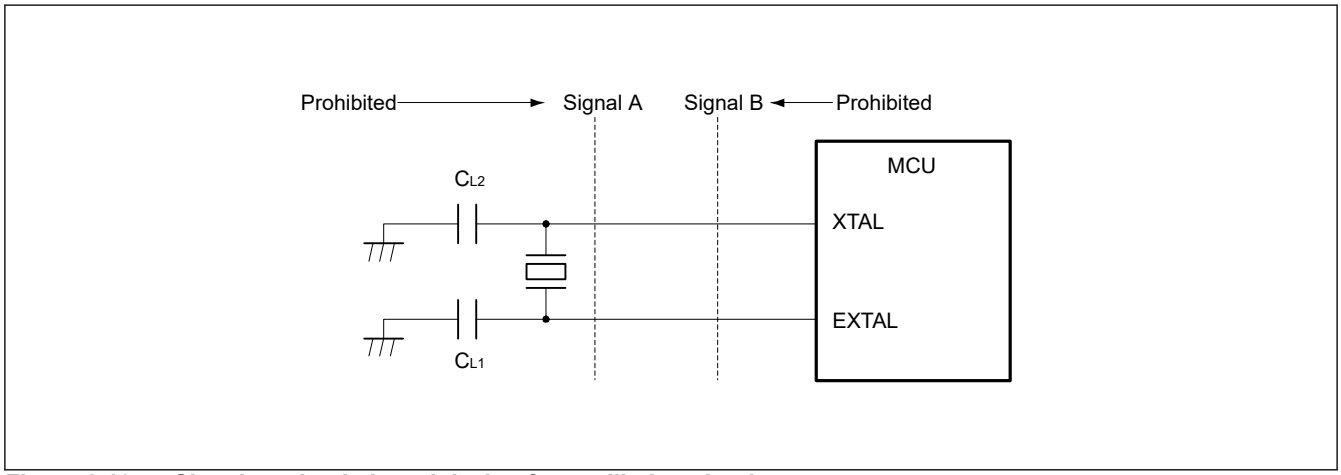



**Figure 8.13 Signal routing in board design for oscillation circuit**

### 8.8.4 Notes on Resonator Connect Pin

When the main clock is not used, the EXTAL and XTAL pins can be used as general ports. When these pins are used as general ports, the main clock must be stopped (MOSCCR.MOSTP bit should be set to 1).

### 8.8.5 Notes on Using Sub-Clock Oscillator

The output of the P212 (EXTAL), P213 (XTAL), and P406 pins may affect the oscillation by the sub-clock oscillator.

If the sub-clock oscillator is used, implement board design so as not to affect the oscillation. Renesas strongly recommends setting the PmnPFS.DSCR[1:0] bits to 00b or 01b when using the P212 (EXTAL), P213 (XTAL), and P406 as output pins and using the sub-clock oscillator.

In addition, when using the sub-clock oscillator in low drive capability (SOMCR.SODRV1 = 1), Renesas recommecds setting the PmnPFS.DSCR[1:0] bits to 00b when using the P212 (EXTAL), P213 (XTAL), and P406 as output pins and using the sub-clock oscillator.

# 9. Clock Frequency Accuracy Measurement Circuit (CAC)

## 9.1 Overview

The Clock Frequency Accuracy Measurement Circuit (CAC) counts pulses of the clock to be measured (measurement target clock) within the time generated by the clock selected as the measurement reference (measurement reference clock), and determines the accuracy depending on whether the number of pulses is within the allowable range. When measurement is complete or the number of pulses within the time generated by the measurement reference clock is not within the allowable range, an interrupt request is generated.

Table 9.1 lists the CAC specifications, Figure 9.1 shows the CAC block diagram, and Table 9.2 lists the CAC I/O pin.

**Table 9.1 CAC specifications**

| Parameter | Specifications |
|---|---|
| Measurement target clocks | Frequency can be measured for:<br>• Main clock oscillator<br>• Sub-clock oscillator<br>• HOCO clock<br>• MOCO clock<br>• LOCO clock<br>• Peripheral module clock B (PCLKB)<br>• IWDT-dedicated clock |
| Measurement reference clocks | Frequency can be referenced to:<br>• External clock input to the CACREF pin<br>• Main clock oscillator<br>• Sub-clock oscillator<br>• HOCO clock<br>• MOCO clock<br>• LOCO clock<br>• Peripheral module clock B (PCLKB)<br>• IWDT-dedicated clock |
| Selectable function | Digital filter |
| Interrupt sources | • Measurement end<br>• Frequency error<br>• Overflow |
| Module-stop function | Module-stop state can be set to reduce power consumption |
| TrustZone Filter | Security attribution can be set |

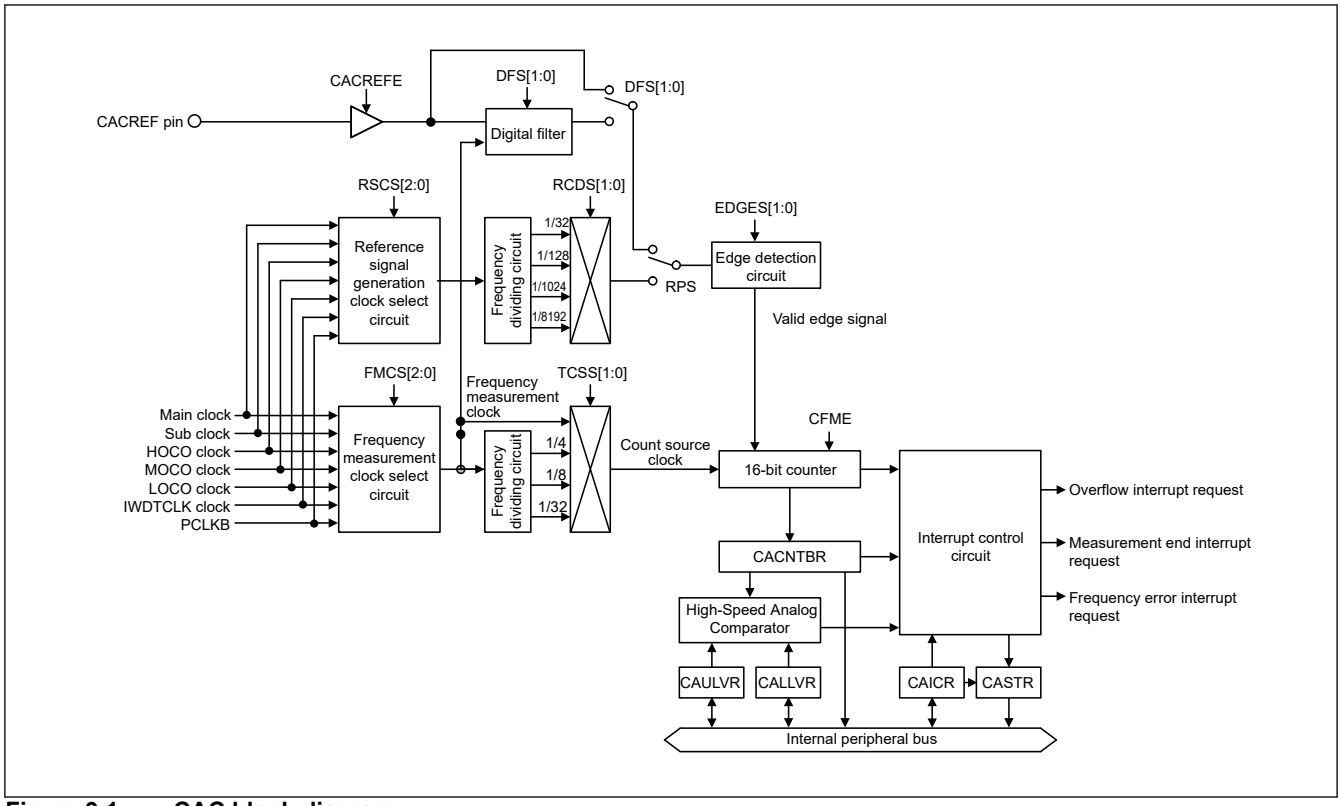



**Figure 9.1 CAC block diagram**

**Table 9.2 CAC I/O pin**

| Function | Pin name | I/O | Description |
|---|---|---|---|
| CAC | CACREF | Input | Measurement reference clock input pin |

## 9.2 Register Descriptions

### 9.2.1 CACR0 : CAC Control Register 0

Base address: CAC = 0x4008_3600

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | CFME |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CFME | Clock Frequency Measurement Enable<br>0: Disable<br>1: Enable | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

**CFME bit (Clock Frequency Measurement Enable)**

The CFME bit enables clock frequency measurement. Changes made to this bit are not immediately reflected to the internal circuit. Read the bit to confirm that the change has been reflected.

## 9.2.2 CACR1 : CAC Control Register 1

Base address: CAC = 0x4008_3600

Offset address: 0x01

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | EDGES[1:0] | | TCSS[1:0] | | FMCS[2:0] | | | CACREFE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CACREFE | CACREF Pin Input Enable<br>0: Disable<br>1: Enable | R/W |
| 3:1 | FMCS[2:0] | Measurement Target Clock Select<br>0 0 0: Main clock oscillator<br>0 0 1: Sub-clock oscillator<br>0 1 0: HOCO clock<br>0 1 1: MOCO clock<br>1 0 0: LOCO clock<br>1 0 1: Peripheral module clock B (PCLKB)<br>1 1 0: IWDT-dedicated clock<br>1 1 1: Setting prohibited | R/W |
| 5:4 | TCSS[1:0] | Timer Count Clock Source Select<br>0 0: No division<br>0 1: × 1/4 clock<br>1 0: × 1/8 clock<br>1 1: × 1/32 clock | R/W |
| 7:6 | EDGES[1:0] | Valid Edge Select<br>0 0: Rising edge<br>0 1: Falling edge<br>1 0: Both rising and falling edges<br>1 1: Setting prohibited | R/W |

Note: Set the CACR1 register when the CACR0.CFME bit is 0.

**CACREFE bit (CACREF Pin Input Enable)**

The CACREFE bit enables the CACREF pin input.

**FMCS[2:0] bits (Measurement Target Clock Select)**

The FMCS[2:0] bits select the measurement target clock whose frequency is to be measured.

**TCSS[1:0] bits (Timer Count Clock Source Select)**

The TCSS[1:0] bits select the division ratio of the measurement target clock.

**EDGES[1:0] bits (Valid Edge Select)**

The EDGES[1:0] bits select the valid edge for the reference signal.

## 9.2.3 CACR2 : CAC Control Register 2

Base address: CAC = 0x4008_3600

Offset address: 0x02

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DFS[1:0] | | RCDS[1:0] | | RSCS[2:0] | | | RPS |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RPS | Reference Signal Select<br>0: CACREF pin input<br>1: Internal clock (internally generated signal) | R/W |
| 3:1 | RSCS[2:0] | Measurement Reference Clock Select<br>0 0 0: Main clock oscillator<br>0 0 1: Sub-clock oscillator<br>0 1 0: HOCO clock<br>0 1 1: MOCO clock<br>1 0 0: LOCO clock<br>1 0 1: Peripheral module clock B (PCLKB)<br>1 1 0: IWDT-dedicated clock<br>1 1 1: Setting prohibited | R/W |
| 5:4 | RCDS[1:0] | Measurement Reference Clock Frequency Division Ratio Select<br>0 0: × 1/32 clock<br>0 1: × 1/128 clock<br>1 0: × 1/1024 clock<br>1 1: × 1/8192 clock | R/W |
| 7:6 | DFS[1:0] | Digital Filter Select<br>0 0: Disable digital filtering<br>0 1: Use sampling clock for the digital filter as the frequency measuring clock<br>1 0: Use sampling clock for the digital filter as the frequency measuring clock divided by 4<br>1 1: Use sampling clock for the digital filter as the frequency measuring clock divided by 16. | R/W |

Note: Set the CACR2 register when the CACR0.CFME bit is 0.

### RPS bit (Reference Signal Select)

The RPS bit selects whether to use the CACREF pin input or an internal clock (internally generated signal) as the reference signal.

### RSCS[2:0] bits (Measurement Reference Clock Select)

The RSCS[2:0] bits select the reference clock for measurement.

### RCDS[1:0] bits (Measurement Reference Clock Frequency Division Ratio Select)

The RCDS[1:0] bits select the frequency-divisor of the reference clock for measurement when an internal reference clock is selected. When RPS = 0 (CACREF pin is used as the reference clock source), the reference clock is not divided.

### DFS[1:0] bits (Digital Filter Select)

The DFS[1:0] bits enable or disable the digital filter and selects its sampling clock.

## 9.2.4 CAICR : CAC Interrupt Control Register

Base address: CAC = 0x4008_3600

Offset address: 0x03

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | OVFFCL | MENDFCL | FERRFCL | — | OVFIE | MENDIE | FERRIE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FERRIE | Frequency Error Interrupt Request Enable<br>0: Disable<br>1: Enable | R/W |
| 1 | MENDIE | Measurement End Interrupt Request Enable<br>0: Disable<br>1: Enable | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2 | OVFIE | Overflow Interrupt Request Enable<br>0: Disable<br>1: Enable | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | FERRFCL | FERRF Clear<br>0: No effect<br>1: The CASTR.FERRF flag is cleared | W |
| 5 | MENDFCL | MENDF Clear<br>0: No effect<br>1: The CASTR.MENDF flag is cleared | W |
| 6 | OVFFCL | OVFF Clear<br>0: No effect<br>1: The CASTR.OVFF flag is cleared. | W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |

### FERRIE bit (Frequency Error Interrupt Request Enable)

The FERRIE bit enables or disables the frequency error interrupt request.

### MENDIE bit (Measurement End Interrupt Request Enable)

The MENDIE bit enables or disables the measurement end interrupt request.

### OVFIE bit (Overflow Interrupt Request Enable)

The OVFIE bit enables or disables the overflow interrupt request.

### FERRFCL bit (FERRF Clear)

Setting the FERRFCL bit to 1 clears the CASTR.FERRF flag.

### MENDFCL bit (MENDF Clear)

Setting the MENDFCL bit to 1 clears the CASTR.MENDF flag.

### OVFFCL bit (OVFF Clear)

Setting the OVFFCL bit to 1 clears the CASTR.OVFF flag.

## 9.2.5 CASTR : CAC Status Register

Base address: CAC = 0x4008_3600

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | OVFF | MENDF | FERRF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FERRF | Frequency Error Flag<br>0: Clock frequency is within the allowable range<br>1: Clock frequency has deviated beyond the allowable range (frequency error). | R |
| 1 | MENDF | Measurement End Flag<br>0: Measurement is in progress<br>1: Measurement ended | R |
| 2 | OVFF | Overflow Flag<br>0: Counter has not overflowed<br>1: Counter overflowed | R |
| 7:3 | — | These bits are read as 0. | R |

### FERRF flag (Frequency Error Flag)

The FERRF flag indicates a deviation of the clock frequency from the set value (frequency error).

[Setting condition]

- The clock frequency is outside the allowable range defined in the CAULVR and CALLVR registers.

[Clearing condition]

- 1 is written to the FERRFCL bit.

### MENDF flag (Measurement End Flag)

The MENDF flag indicates the end of measurement.

[Setting condition]

- Measurement ends.

[Clearing condition]

- 1 is written to the MENDFCL bit.

### OVFF flag (Overflow Flag)

The OVFF flag indicates that the counter overflowed.

[Setting condition]

- The counter overflows.

[Clearing condition]

- 1 is written to the CAICR.OVFFCL bit.

## 9.2.6 CAULVR : CAC Upper-Limit Value Setting Register

Base address: CAC = 0x4008_3600

Offset address: 0x06

| Bit position: | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | The Upper Value of the Allowable Range<br>The CAULVR register is a 16-bit read/write register that specifies the upper value of the allowable range. When the counter value exceeds the value specified in this register, a frequency error is detected. Write to this register when the CACR0.CFME bit is 0.<br>The counter value stored in CACNTBR can vary depending on the difference between the phases of the digital filter and edge-detection circuit, and the signal on the CACREF pin. Ensure that this setting allows an adequate margin. | R/W |

## 9.2.7 CALLVR : CAC Lower-Limit Value Setting Register

Base address: CAC = 0x4008_3600

Offset address: 0x08

| Bit position: | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | The Lower Value of the Allowable Range<br>The CALLVR register is a 16-bit read/write register that specifies the lower value of the allowable range. When the counter value falls below the value specified in this register, a frequency error is detected. Write to this register when the CACR0.CFME bit is 0.<br>The counter value stored in CACNTBR can vary depending on the difference between the phases of the digital filter and edge-detection circuit, and the signal on the CACREF pin. Ensure that this setting allows an adequate margin. | R/W |

### 9.2.8 CACNTBR : CAC Counter Buffer Register

Base address: CAC = 0x4008_3600

Offset address: 0x0A

| Bit position: | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | The Measurement Result<br>The CACNTBR register is a 16-bit read-only register that stores the measurement result. | R |

## 9.3 Operation

### 9.3.1 Measuring Clock Frequency

The CAC measures the clock frequency using the CACREF pin input or an internal clock as a reference. Figure 9.2 shows an operating example of the CAC.

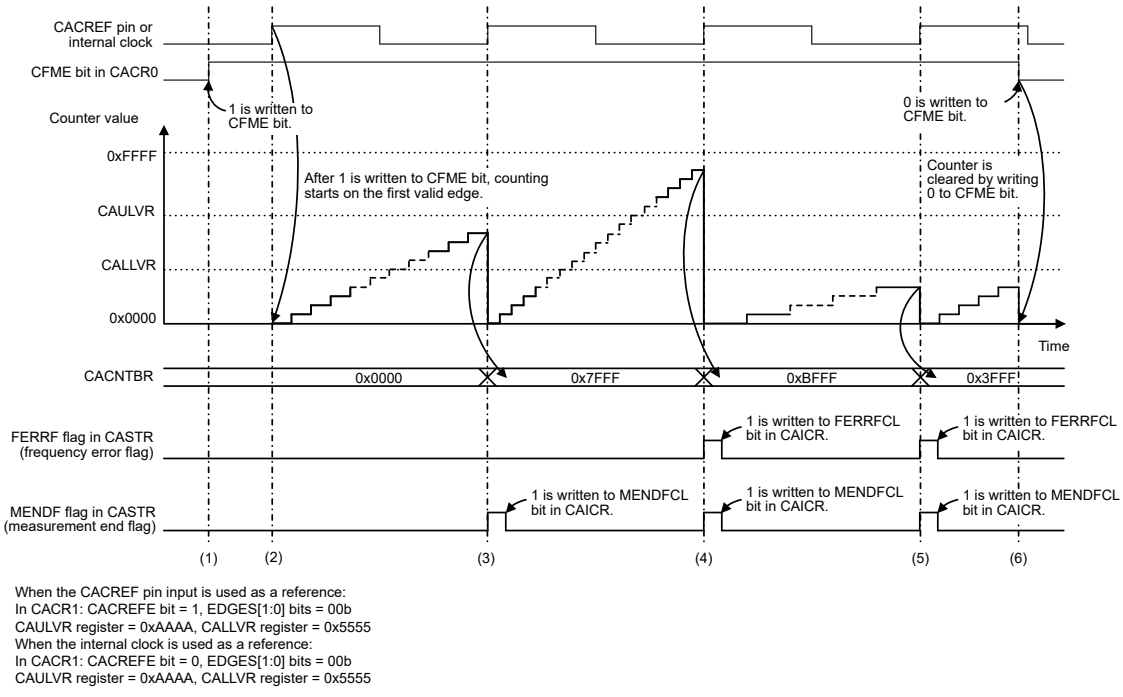



**Figure 9.2 CAC operating example**

The events in Figure 9.2 are:

1. When the CACREF pin input is used as reference (CACR1.CACREFE = 1), frequency measurement is enabled by writing 1 to the CACR0.CFME bit while the CACR2.RPS bit is set to 0 and the CACR1.CACREFE bit is set to 1. When the internal clock is used as reference (CACR1.CACREFE = 0), frequency measurement is enabled by writing 1 to the CACR0.CFME bit while the CACR2.RPS bit is set to 1.
2. When the CACREF pin input is used as reference, after 1 is written to the CFME bit, the timer starts up-counting if the valid edge selected by the CACR1.EDGES[1:0] bits (rising edge (CACR1.EDGES[1:0] = 00b) in Figure 9.2) is input from the CACREF pin. When the internal clock is used as reference, after 1 is written to the CFME bit, the timer starts up-counting if the valid edge selected by the CACR1.EDGES[1:0] bits (rising edge (CACR1.EDGES[1:0] = 00b) in Figure 9.2) is input based on the clock source selected by the CACR2.RSCS[2:0] bits.
3. When the next valid edge is input, the counter value is transferred to CACNTBR and compared with the values in CAULVR and CALLVR. If both CACNTBR ≤ CAULVR and CACNTBR ≥ CALLVR are true, only the MENDF flag in CASTR is set to 1, because the clock frequency is correct. If the MENDIE bit in CAICR is 1, a measurement end interrupt is generated.
4. When the next valid edge is input, the counter value is transferred to CACNTBR and compared with the values in CAULVR and CALLVR. If CACNTBR > CAULVR, the FERRF flag in CASTR is set to 1, because the clock frequency is erroneous. If the FERRIE bit in CAICR is 1, a frequency error interrupt is generated. The MENDF flag in CASTR is set to 1 at the end of measurement. If the MENDIE bit in CAICR is 1, a measurement end interrupt is generated.
5. When the next valid edge is input, the counter value is transferred to CACNTBR and compared with the values in CAULVR and CALLVR. If CACNTBR < CALLVR, the FERRF flag in CASTR is set to 1, because the clock frequency is erroneous. If the FERRIE bit in CAICR is 1, a frequency error interrupt is generated. The MENDF flag in CASTR is set to 1 at the end of measurement. If the MENDIE bit in CAICR is 1, a measurement end interrupt is generated.
6. When the CFME bit in CACR0 is 1, the counter value is transferred to CACNTBR and compared with the values in CAULVR and CALLVR every time a valid edge is input. Writing 0 to the CFME bit in CACR0 clears the counter and stops up-counting.

### 9.3.2 Digital Filtering of Signals on CACREF Pin

The CACREF pin has a digital filter, and levels on the CACREF pin are transmitted to the internal circuitry after three consecutive matches in the selected sampling interval. The same level continues to be transmitted internally until the level on the pin has three consecutive matches again. Enabling or disabling of the digital filter and its sampling clock are selectable.

The counter value transferred to CACNTBR might be in error by up to 1 cycle of the sampling clock because of the difference between the phases of the digital filter and the signal input to the CACREF pin. When a frequency dividing clock is selected as a count source clock, the counter value error is obtained using the following formula:

$$\text{Counter value error} = (1 \text{ cycle of the count source clock}) / (1 \text{ cycle of the sampling clock})$$

## 9.4 Interrupt Requests

The CAC generates three types of interrupt requests:

- Frequency error interrupt
- Measurement end interrupt
- Overflow interrupt

When an interrupt source is generated, the associated status flag is set to 1. Table 9.3 provides information on the CAC interrupt requests.

**Table 9.3 CAC interrupt requests (1 of 2)**

| Interrupt request | Interrupt enable bit | Status flag | Interrupt sources |
|---|---|---|---|
| Frequency error interrupt | CAICR.FERRIE | CASTR.FERRF | The result of comparing CACNTBR with CAULVR and CALLVR is either CACNTBR > CAULVR or CACNTBR < CALLVR |

**Table 9.3 CAC interrupt requests (2 of 2)**

| Interrupt request | Interrupt enable bit | Status flag | Interrupt sources |
|---|---|---|---|
| Measurement end interrupt | CAICR.MENDIE | CASTR.MENDF | • Valid edge is input from the CACREF pin or internal clock<br>• Measurement end interrupt does not occur at the first valid edge after writing 1 to the CACR0.CFME bit |
| Overflow interrupt | CAICR.OVFIE | CASTR.OVFF | Counter overflows |

## 9.5 Usage Notes

### 9.5.1 Settings for the Module-Stop Function

The Module Stop Control Register C (MSTPCRC) can enable or disable CAC operation. The CAC module is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

# 10. Low Power Modes

## 10.1 Overview

The MCU has several functions for reducing power consumption, such as setting clock dividers, stopping modules, selecting power control mode in Normal mode, and transitioning to low power modes.

Table 10.1 lists the specifications of the low power mode functions. Table 10.2 lists the conditions to transition to low power modes, the states of the CPU and peripheral modules, and the method for canceling each mode. After a reset, the MCU enters the program execution state, but only the DTC, DMAC and SRAM operate.

**Table 10.1 Specifications of the low power mode functions**

| Item | Specification |
|---|---|
| Reducing power consumption by switching clock signals | The frequency division ratio can be selected independently for the system clock (ICLK), peripheral module clocks (PCLKA, PCLKB, PCLKC, PCLKD), and flash interface clock (FCLK). *1 |
| Module stop | Functions can be stopped independently for each peripheral module |
| Low-power modes | • Sleep mode<br>• Software Standby mode<br>• Snooze mode<br>• Deep Software Standby mode |
| Power control modes | • Power consumption can be reduced in Normal, Seep and Snooze modes by selecting an appropriate operating power control mode according to the operating frequency.<br>• Three operating power control modes are available:<br>High-speed mode<br>Low-speed mode<br>Subosc-speed mode |
| TrustZone Filter | Security attribution can be set for each registers |

Note 1. For details, see section 8, Clock Generation Circuit

**Table 10.2 Operating conditions of each low power mode (1 of 2)**

| Item | Sleep mode | Software Standby mode | Snooze mode | Deep Software Standby mode |
|---|---|---|---|---|
| Transition condition | WFI instruction while SBYCR.SSBY = 0 | WFI instruction while SBYCR.SSBY = 1 and DPSBYCR.DPSBY = 0 | Snooze request trigger in Software Standby mode. SNZCR.SNZE=1. | WFI instruction while SBYCR.SSBY = 1 and DPSBYCR.DPSBY = 1 |
| Canceling method | All interrupts.<br>Any reset available in the mode. | Interrupts shown in Table 10.3.<br>Any reset available in the mode. | Interrupts shown in Table 10.3.<br>Any reset available in the mode. | Interrupts shown in Table 10.3.<br>Any reset available in the mode. |
| State after cancellation by an interrupt | Program execution state (interrupt processing) | Program execution state (interrupt processing) | Program execution state (interrupt processing) | Reset state |
| State after cancellation by a reset | Reset state | Reset state | Reset state | Reset state |
| Main clock oscillator | Selectable | Stop | Selectable*5 | Stop |
| Sub-clock oscillator | Selectable | Selectable | Selectable | Selectable |
| High-speed on-chip oscillator | Selectable | Stop | Selectable | Stop |
| Middle-speed on-chip oscillator | Selectable | Stop | Selectable | Stop |
| Low-speed on-chip oscillator | Selectable | Selectable | Selectable | Selectable*8 |
| IWDT-dedicated on-chip oscillator | Selectable*1 | Selectable*1 | Selectable*1 | Stop |
| PLL | Selectable | Stop | Selectable*5 | Stop |
| PLL2 | Selectable | Stop | Selectable*5 | Stop |
| Oscillation stop detection function | Selectable | Operation prohibited | Operation prohibited | Operation prohibited |
| Clock/buzzer output function | Selectable | Selectable*2 | Selectable | Stop (Undefined) |
| CPU | Stop (Retained) | Stop (Retained) | Stop (Retained) | Stop (Undefined) |
| SRAMn (n = 0) | Selectable | Stop (Retained) | Selectable | Stop (Undefined) |

**Table 10.2 Operating conditions of each low power mode (2 of 2)**

| Item | Sleep mode | Software Standby mode | Snooze mode | Deep Software Standby mode |
|---|---|---|---|---|
| Standby SRAM | Selectable | Stop (Retained) | Selectable | Stop (Retained/Undefined)*9 |
| Flash memory | Operating | Stop (Retained) | Stop (Retained) | Stop (Retained) |
| DMA Controller (DMAC) | Selectable | Stop (Retained) | Operation prohibited | Stop (Undefined) |
| Data Transfer Controller (DTC) | Selectable | Stop (Retained) | Selectable | Stop (Undefined) |
| USB 2.0 Full-Speed (USBFSn, n = 0) | Selectable | Stop (Retained). Detection of USB resumption is possible. | Operation prohibited. Detection of USB resumption is possible. | Stop (Retained/Undefined) Detection of USB resumption is possible.*10 |
| Watchdog Timer (WDT) | Selectable*1 | Stop (Retained) | Stop (Retained) | Stop (Undefined) |
| Independent Watchdog Timer (IWDT) | Selectable*1 | Selectable*1 | Selectable*1 | Stop (Undefined) |
| Realtime clock (RTC) | Selectable | Selectable | Selectable | Selectable*11 |
| Low Power Asynchronous General Purpose Timer (AGTn (n = 0 to 3)) | Selectable | Selectable*3 | Selectable*3 | Selectable*3 |
| Low Power Asynchronous General Purpose Timer (AGTn (n = 4 to 5)) | Selectable | Selectable*14 | Selectable*14 | Stop (Undefined) |
| 12-Bit A/D Converter (ADC12) | Selectable | Stop (Retained) | Selectable*15 | Stop (Undefined) |
| 12-Bit D/A Converter (DAC12) | Selectable | Stop (Retained) | Selectable | Stop (Undefined) |
| Capacitive Touch Sensing Unit (CTSU) | Selectable | Stop (Retained) | Selectable | Stop (Undefined) |
| Data Operation Circuit (DOC) | Selectable | Stop (Retained) | Selectable | Stop (Undefined) |
| Serial Communications Interface (SCI0) | Selectable | Stop (Retained) | Selectable (RXD0 falling edge is available, to enter snooze mode) (only in asynchronous mode).*6 | Stop (Undefined) |
| Serial Communications Interface (SCIn (n = 1 to 4, 9)) | Selectable | Stop (Retained) | Operation prohibited | Stop (Undefined) |
| I2C Bus Interface (IIC0) | Selectable | Selectable*4 | Selectable*4 Only wakeup interrupt is available. | Stop (Undefined) |
| I2C Bus Interface (IIC1) | Selectable | Stop (Retained) | Operation prohibited | Stop (Undefined) |
| Event Link Controller (ELC) | Selectable | Stop (Retained) | Selectable*7 | Stop (Undefined) |
| IRQn (n = 0 to 15) pin interrupt | Selectable | Selectable | Selectable | Stop (Undefined) |
| NMI, IRQn-DS (n = 0, 1, 4 to 12, 14, 15) pin interrupt | Selectable | Selectable | Selectable | Selectable |
| Low voltage detection (LVD) | Selectable | Selectable | Selectable | Selectable*12 |
| Power-on reset circuit | Operating | Operating | Operating | Operating*13 |
| Other peripheral modules | Selectable | Stop (Retained) | Operation prohibited | Stop (Undefined) |
| I/O Ports | Operating | Retained | Operating | Retained |

Note: Selectable means that operating or not operating can be selected by the control registers.
Stop (Retained) means that the contents of the internal registers are retained but the operations are suspended.
Operation prohibited means that the function must be stopped before entering Software Standby mode.
Stop (Undefined) means that the contents of the internal registers are undefined and power to the internal circuit is cut off.
All modules whose module-stop bits are 0 start as soon as PCLKs are supplied after entering Snooze mode. In order to avoid increase in power consumption in Snooze mode, module-stop bit of modules which are unnecessary in Snooze mode must be set to 1 before entering Software Standby mode.

Note 1. In IWDT-dedicated on-chip oscillator and IWDT, operating or stopping is selected by setting the IWDT Stop Control bit (IWDTSTPCTL) in Option Function Select register 0 (OFS0) in IWDT auto start mode. In WDT, operating or stopping is selected by setting the WDT Stop Control bit (WDTSTPCTL) in Option Function Select Register 0 (OFS0) in WDT auto start mode.Power consumption can be reduced in Normal and Sleep modes by selecting an appropriate operating power control mode according to the operating frequency.

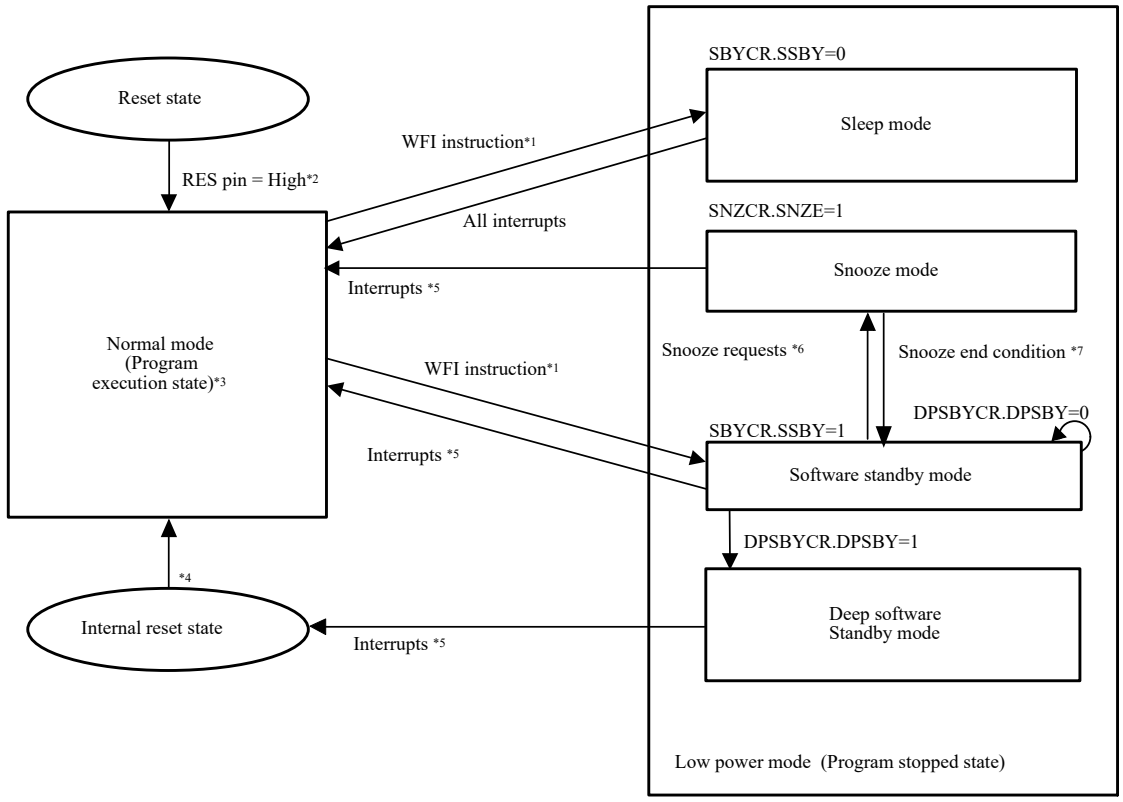
Note 2. Stopped when the clock output source select bits (CKOCR.CKOSEL[2:0]) are set to a value other than 010b (LOCO) and 100b (SOSC).

Note 3. AGT0/AGT2 operation is possible when 100b (AGTLCLK) or 110b (AGTSCLK) is selected by the AGT0/2.AGTMR1.TCK[2:0] bits. AGT1/AGT3 operation is possible when 100b (AGTLCLK), 110b (AGTSCLK) or 101b (Underflow event signal from AGT0/AGT2) is selected by the AGT1/3.AGTMR1.TCK[2:0] bits.
When 100b (AGTLCLK) is selected by AGTn.AGTMR1.TCK[2:0] bits (n = 0, 1, 2, 3), the DPSBYCR.DEEPCUT[1:0] bits must set to 00b before entering Deep Software Standby mode.
Note 4. IIC0 wakeup interrupt is available.
Note 5. When using SCI0 in Snooze mode, MOSCCR.MOSTP and PLLCR.PLLSTP and PLL2CR.PLL2STP bits must be 1.
Note 6. Serial communication modes of SCI0 is only in asynchronous mode.
Note 7. Event lists the restrictions described in section 10.10.14. ELC Events in Snooze Mode.
Note 8. If the DPSBYCR.DEEPCUT[1:0] bits are 00b, the oscillator status is the same as before entering Deep Software Standby mode. When the DPSBYCR.DEEPCUT[1:0] bits are not 00b, the oscillator stops when the MCU enters Deep Software Standby mode.
Note 9. If the DPSBYCR.DEEPCUT[1:0] bits are 00b, data in the Standby SRAM is retained in Deep Software Standby mode. When the DPSBYCR.DEEPCUT[1:0] bits are not 00b, data in the Standby SRAM is undefined in Deep Software Standby mode.
Note 10. If the DPSBYCR.DEEPCUT[1:0] bits are 00b, the values of the USB resume detection circuit registers are retained and detection of USB resumption is enabled, and the values of other registers are undefined in Deep Software Standby mode. When the DPSBYCR.DEEPCUT[1:0] bits are not 00b, the values of all registers are undefined in Deep Software Standby mode.
Note 11. When the RCR4.RCKSEL bit set to 1 (LOCO), the DPSBYCR.DEEPCUT[1:0] bits must set to 00b before entering Deep Software Standby mode.
Note 12. When using LVD in Deep Software Standby mode, DPSBYCR.DEEPCUT[1:0] bits must be 00b or 01b before entering Deep Software Standby mode.
Note 13. When the MCU enters Deep Software Standby mode with the DPSBYCR.DEEPCUT[1:0] bits set to 11b, the LVD circuit stops and the low-power function of the power-on reset circuit is enabled.
Note 14. AGT4 operation is possible when 100b (AGTLCLK) or 110b (AGTSCLK) is selected by the AGT4.AGTMR1.TCK[2:0] bits. AGT5 operation is possible when 100b (AGTLCLK), 110b (AGTSCLK) or 101 (Underflow event signal from AGT4) is selected by the AGT5.AGTMR1.TCK[2:0] bits.
Note 15. When using the 12-bit A/D Converter in Snooze mode, the ADCMPCR.CMPAE and ADCMPCR.CMPBE bits must be 1.

**Table 10.3 Interrupt Source for canceling Snooze, Software Standby and Deep Software Standby Modes**

| Interrupt source | Name | Software Standby Mode | Snooze Mode | Deep Software Standby Mode |
|---|---|---|---|---|
| NMI | | Yes | Yes | Yes |
| Port | PORT_IRQn (n = 0 to 15) | Yes | Yes | No |
| | PORT_IRQn-DS (n = 0, 1, 4 to 12, 14, 15) | Yes | Yes | Yes |
| LVD | LVD_LVD1 | Yes | Yes | Yes |
| | LVD_LVD2 | Yes | Yes | Yes |
| IWDT | IWDT_NMIUNDF | Yes | Yes | No |
| USBFS0 | USBFS0_USBR | Yes | Yes | Yes |
| RTC | RTC_ALM | Yes | Yes | Yes |
| | RTC_PRD | Yes | Yes | Yes |
| AGT1 | AGT1_AGTI | Yes | Yes*3 | Yes |
| | AGT1_AGTCMAI | Yes | Yes | No |
| | AGT1_AGTCMBI | Yes | Yes | No |
| AGT3 | AGT3_AGTI | Yes | Yes*3 | Yes |
| | AGT3_AGTCMAI | Yes | Yes | No |
| | AGT3_AGTCMBI | Yes | Yes | No |
| IIC0 | IIC0_WUI | Yes | Yes | No |
| ADC12n (n = 0) | ADC12n_WCMPM | No | Yes with SELSR0*1 *3 | No |
| | ADC12n_WCMPUM | No | Yes with SELSR0*1 *3 | No |
| SCI0 | SCI0_AM | No | Yes with SELSR0*1 *2 | No |
| | SCI0_RXI_OR_ERI | No | Yes with SELSR0*1 *2 | No |
| DTC | DTC_COMPLETE | No | Yes with SELSR0*1 *3 | No |
| DOC | DOC_DOPCI | No | Yes with SELSR0 *1 | No |
| CTSU | CTSU_CTSUFN | No | Yes with SELSR0*1 | No |

Note 1. To use the interrupt request as a trigger for exiting the Snooze mode, the request must be selected in SELSR0 . See section 13, Interrupt Controller Unit (ICU) for the setting of SELSR0. When a trigger selected in SELSR0 occurs after executing WFI instruction and during the transition from Normal mode to Software Standby mode, the request might or might not be accepted, depending on the timing of the occurrence.
Note 2. Only one of either SCI0_AM or SCI0_RXI_OR_ERI can be set.
Note 3. The event which is enabled by the SNZEDCRn must not be used.




Note 1. When an interrupt that acts as a trigger for cancel is received during a transition to the program-stopped state after the execution of a WFI instruction, the MCU executes interrupt exception handling instead of transitioning to low power mode.
Note 2. The MOCO clock is the source of the operating clock following a transition from the reset state to Normal mode.
Note 3. The transition to Normal mode is made because of an interrupt from Sleep mode, Snooze mode, or Software Standby mode. The clock source is the same as before entering the low power mode.
Note 4. When an available interrupt request is generated, an internal reset (Deep Software Standby reset) is generated over a fixed period. Canceling of Deep Software Standby mode accompanies release from the internal reset state, and then the MCU transitions to Normal mode and executes reset exception processing with the MOCO clock as the source of the operating clock.
Note 5. See Table 10.3
Note 6. See Table 10.7
Note 7. See Table 10.9

**Figure 10.1 Mode Transitions**

## 10.2 Register Descriptions

### 10.2.1 LPMSAR : Low Power Mode Security Attribution Register

Base address: SYSC = 0x4001_E000

Offset address: 0x3C8

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | NONSEC13 | — | — | — | NONSEC9 | NONSEC8 | — | — | — | NONSEC4 | — | NONSEC2 | — | NONSEC0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | NONSEC0*1 | Non Secure Attribute bit 0<br>Target register: OPCCR, SOPCCR<br>0: Secure<br>1: Non Secure | R/W |
| 1 | — | This bit is read as 1. The write value should be 1. | R/W |
| 2 | NONSEC2 | Non Secure Attribute bit 2<br>Target register: SBYCR<br>0: Secure<br>1: Non Secure | R/W |
| 3 | — | This bit is read as 1. The write value should be 1. | R/W |
| 4 | NONSEC4 | Non Secure Attribute bit 4<br>Target register: SNZCR, SNZEDCRn, SNZREQCRn<br>0: Secure<br>1: Non Secure | R/W |
| 7:5 | — | These bits are read as 1. The write value should be 1. | R/W |
| 8 | NONSEC8 | Non Secure Attribute bit 8<br>Target register: DPSBYCR<br>0: Secure<br>1: Non Secure | R/W |
| 9 | NONSEC9 | Non Secure Attribute bit 9<br>Target register: DPSWCR<br>0: Secure<br>1: Non Secure | R/W |
| 12:10 | — | These bits are read as 1. The write value should be 1. | R/W |
| 13 | NONSEC13 | Non Secure Attribute bit 13<br>Target register: LDOSCR, PL2LDOSCR<br>0: Secure<br>1: Non Secure | R/W |
| 31:14 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

Note 1. It is recommended that these bits are configured as Non Secure when the device life cycle is NSECSD (DLMMON.DLMMON[3:0] = 0011b). See section 46.6.1. Restrictions on setting the security attribution for details.

The LPMSAR register controls the secure attribute of Low Power Mode registers.

**NONSEC0 bit (Non Secure Attribute bit 0)**

This bit controls the security attribute of OPCCR, SOPCCR.

**NONSEC2 bit (Non Secure Attribute bit 2)**

This bit controls the security attribute of SBYCR.

**NONSEC4 bit (Non Secure Attribute bit 4)**

This bit controls the security attribute of SNZCR, SNZEDCRn, SNZREQCRn

**NONSEC8 bit (Non Secure Attribute bit 8)**

This bit controls the security attribute of DPSBYCR.

**NONSEC9 bit (Non Secure Attribute bit 9)**

This bit controls the security attribute of DPSWCR.

**NONSEC13 bit (Non Secure Attribute bit 13)**

This bit controls the security attribute of LDOSCR and PL2LDOSCR.

## 10.2.2 DPFSAR : Deep Software Standby Interrupt Factor Security Attribution Register

Base address: SYSC = 0x4001_E000

Offset address: 0x3E0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DPFSAn (n = 16 to 31) | | | | | | | | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DPFSA15 | DPFSA14 | — | DPFSA12 | DPFSA11 | DPFSA10 | DPFSA09 | DPFSA08 | DPFSA07 | DPFSA06 | DPFSA05 | DPFSA04 | — | — | DPFSA01 | DPFSA00 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | DPFSA01, DPFSA00 | Deep Software Standby Interrupt Factor Security Attribute bit n (n = 0, 1)<br>Target register: DPSIER0.bn, DPSIFR0.bn, DPSIEGR0.bn (n = 0, 1)<br>Target factor : IRQn-DS Pin (n = 0, 1)<br>0: Secure<br>1: Non Secure | R/W |
| 3:2 | — | These bits are read as 1. The write value should be 1. | R/W |
| 12:4 | DPFSA12 to DPFSA04 | Deep Software Standby Interrupt Factor Security Attribute bit n (n = 4 to 12)<br>Target register: DPSIER0.bn, DPSIFR0.bn, DPSIEGR0.bn (n = 4 to 7), DPSIER1.bn, DPSIFR1.bn, DPSIEGR1.bn (n = 0 to 4)<br>Target factor : IRQn-DS Pin (n = 4 to 12)<br>0: Secure<br>1: Non Secure | R/W |
| 13 | — | This bit is read as 1. The write value should be 1. | R/W |
| 15:14 | DPFSA15, DPFSA14 | Deep Software Standby Interrupt Factor Security Attribute bit n (n = 14, 15)<br>Target register: DPSIER1.bn, DPSIFR1.bn, DPSIEGR1.bn (n = 6, 7)<br>Target factor : IRQn-DS Pin (n = 14, 15)<br>0: Secure<br>1: Non Secure | R/W |
| 16 | DPFSA16 | Deep Software Standby Interrupt Factor Security Attribute bit 16<br>Target register: DPSIER2.b0, DPSIFR2.b0, DPSIEGR2.b0<br>Target factor : LVD1<br>0: Secure<br>1: Non Secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 17 | DPFSA17 | Deep Software Standby Interrupt Factor Security Attribute bit 17<br>Target register: DPSIER2.b1, DPSIFR2.b1, DPSIEGR2.b1<br>Target factor : LVD2<br>0: Secure<br>1: Non Secure | R/W |
| 18 | DPFSA18 | Deep Software Standby Interrupt Factor Security Attribute bit 18<br>Target register: DPSIER2.b2, DPSIFR2.b2<br>Target factor : RTC Interval<br>0: Secure<br>1: Non Secure | R/W |
| 19 | DPFSA19 | Deep Software Standby Interrupt Factor Security Attribute bit 19<br>Target register: DPSIER2.b3, DPSIFR2.b3<br>Target factor : RTC Alarm<br>0: Secure<br>1: Non Secure | R/W |
| 20 | DPFSA20 | Deep Software Standby Interrupt Factor Security Attribute bit 20<br>Target register: DPSIER2.b4, DPSIFR2.b4, DPSIEGR2.b4<br>Target factor : NMI Pin<br>0: Secure<br>1: Non Secure | R/W |
| 23:21 | — | These bits are read as 1. The write value should be 1. | R/W |
| 24 | DPFSA24 | Deep Software Standby Interrupt Factor Security Attribute bit 24<br>Target register: DPSIER3.b0, DPSIFR3.b0<br>Target factor : USBFS0 Suspend/Resume<br>0: Secure<br>1: Non Secure | R/W |
| 25 | — | This bit is read as 1. The write value should be 1. | R/W |
| 26 | DPFSA26 | Deep Software Standby Interrupt Factor Security Attribute bit 26<br>Target register: DPSIER3.b2, DPSIFR3.b2<br>Target factor : AGT1 Underflow<br>0: Secure<br>1: Non Secure | R/W |
| 27 | DPFSA27 | Deep Software Standby Interrupt Factor Security Attribute bit 27<br>Target register: DPSIER3.b3, DPSIFR3.b3<br>Target factor : AGT3 Underflow<br>0: Secure<br>1: Non Secure | R/W |
| 31:28 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

The DPFSAR register controls the secure attribute of Deep Software Standby Interrupt Factor control registers.

### DPFSA01, DPFSA00 bits (Deep Software Standby Interrupt Factor Security Attribute bit n (n = 0, 1))

These bits control the security attribute of DPSIER0.bn, DPSIFR0.bn, and DPSIEGR0.bn (n = 0, 1).

Target factor is IRQn-DS Pin (n = 0, 1).

### DPFSA04 to DPFSA12 bits (Deep Software Standby Interrupt Factor Security Attribute bit n (n = 4 to 12))

These bits control the security attribute of DPSIER0.bn, DPSIFR0.bn, DPSIEGR0.bn (n = 4 to 7), DPSIER1.bn, DPSIFR1.bn, and DPSIEGR1.bn (n = 0 to 4).

Target factor is IRQn-DS Pin (n = 4 to 12).

### DPFSA15, DPFSA14 bits (Deep Software Standby Interrupt Factor Security Attribute bit n (n = 14, 15))

These bits control the security attribute of DPSIER1.bn, DPSIFR1.bn, and DPSIEGR1.bn (n = 6, 7).

Target factor is IRQn-DS Pin (n = 14, 15).

**DPFSA16 bit (Deep Software Standby Interrupt Factor Security Attribute bit 16)**

This bit controls the security attribute of DPSIER2.b0, DPSIFR2.b0, and DPSIEGR2.b0.

Target factor is LVD1.

**DPFSA17 bit (Deep Software Standby Interrupt Factor Security Attribute bit 17)**

This bit controls the security attribute of DPSIER2.b1, DPSIFR2.b1, and DPSIEGR2.b1.

Target factor is LVD2.

**DPFSA18 bit (Deep Software Standby Interrupt Factor Security Attribute bit 18)**

This bit controls the security attribute of DPSIER2.b2 and DPSIFR2.b2.

Target factor is RTC Interval.

**DPFSA19 bit (Deep Software Standby Interrupt Factor Security Attribute bit 19)**

This bit controls the security attribute of DPSIER2.b3 and DPSIFR2.b3.

Target factor is RTC Alarm.

**DPFSA20 bit (Deep Software Standby Interrupt Factor Security Attribute bit 20)**

This bit controls the security attribute of DPSIER2.b4, DPSIFR2.b4, and DPSIEGR2.b4.

Target factor is NMI Pin.

**DPFSA24 bit (Deep Software Standby Interrupt Factor Security Attribute bit 24)**

This bit controls the security attribute of DPSIER3.b0 and DPSIFR3.b0.

Target factor is USBFS0 Suspend/Resume.

**DPFSA26 bit (Deep Software Standby Interrupt Factor Security Attribute bit 26)**

This bit controls the security attribute of DPSIER3.b2 and DPSIFR3.b2.

Target factor is AGT1 Underflow.

**DPFSA27 bit (Deep Software Standby Interrupt Factor Security Attribute bit 27)**

This bit controls the security attribute of DPSIER3.b3 and DPSIFR3.b3.

Target factor is AGT3 Underflow.

## 10.2.3 SBYCR : Standby Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x00C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SSBY | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14:0 | — | These bits are read as reset value.The write value should be reset value | R/W |
| 15 | SSBY | Software Standby Mode Select<br>0: Sleep mode<br>1: Software Standby mode. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

### SSBY bit (Software Standby Mode Select)

The SSBY bit specifies the transition destination after a WFI instruction is executed.

When the SSBY bit is set to 1, the MCU enters Software Standby mode after execution of a WFI instruction. When the MCU returns to Normal mode from Software Standby mode by an interrupt, the SSBY bit remains 1. The SSBY bit can be cleared by writing 0 to it.

While the OSTDCR.OSTDE bit is 1, setting of the SSBY bit is ignored. Even if SSBY bit is 1, the MCU enters Sleep mode on execution of a WFI instruction.

While the FENTRYR.FENTRYC bit is 1 setting of the SSBY bit is ignored. Even if SSBY bit is 1, the MCU enters Sleep mode on execution of a WFI instruction.

## 10.2.4 MSTPCRA : Module Stop Control Register A

Base address: MSTP = 0x4008_4000

Offset address: 0x000

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | MSTPA22 | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | MSTPA7 | — | — | — | — | — | — | MSTPA0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MSTPA0 | SRAM0 Module Stop<br>Target module: SRAM0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 6:1 | — | These bits are read as 1. The write value should be 1. | R/W |
| 7 | MSTPA7 | Standby SRAM Module Stop<br>Target module: Standby SRAM<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 21:8 | — | These bits are read as 1. The write value should be 1. | R/W |
| 22 | MSTPA22 | DMA Controller/Data Transfer Controller Module Stop*1<br>Target module: DTC, DMAC<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 31:23 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. When rewriting the MSTPA22 bit from 0 to 1, disable the DMAC and DTC before setting the MSTPA22 bit.

## 10.2.5 MSTPCRB : Module Stop Control Register B

Base address: MSTP = 0x4008_4000

Offset address: 0x004

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MSTPB31 | MSTPB30 | MSTPB29 | MSTPB28 | MSTPB27 | — | — | — | — | MSTPB22 | — | — | MSTPB19 | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | MSTPB11 | — | MSTPB9 | MSTPB8 | — | MSTPB6 | — | — | — | MSTPB2 | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | This bit is read as 1. The write value should be 1. | R/W |
| 1 | — | This bit is read as 1. The write value should be 1. | R/W |
| 2 | MSTPB2 | Controller Area Network 0 Module Stop*1<br>Target module: CAN0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 5:3 | — | These bits are read as 1. The write value should be 1. | R/W |
| 6 | MSTPB6 | Quad Serial Peripheral Interface Module Stop<br>Target module: QSPI<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 7 | — | This bit is read as 1. The write value should be 1. | R/W |
| 8 | MSTPB8 | $I^2C$ Bus Interface 1 Module Stop<br>Target module: IIC1<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 9 | MSTPB9 | $I^2C$ Bus Interface 0 Module Stop<br>Target module: IIC0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 10 | — | This bit is read as 1. The write value should be 1. | R/W |
| 11 | MSTPB11 | Universal Serial Bus 2.0 FS Interface 0 Module Stop*2<br>Target module: USBFS0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 12 | — | This bit is read as 1. The write value should be 1. | R/W |
| 15:13 | — | These bits are read as 1. The write value should be 1. | R/W |
| 17:16 | — | These bits are read as 1. The write value should be 1. | R/W |
| 18 | — | These bits are read as 1. The write value should be 1. | R/W |
| 19 | MSTPB19 | Serial Peripheral Interface 0 Module Stop<br>Target module: SPI0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 21:20 | — | These bits are read as 1. The write value should be 1. | R/W |
| 22 | MSTPB22 | Serial Communication Interface 9 Module Stop<br>Target module: SCI9<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 26:23 | — | These bits are read as 1. The write value should be 1. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 27 | MSTPB27 | Serial Communication Interface 4 Module Stop<br>Target module: SCI4<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 28 | MSTPB28 | Serial Communication Interface 3 Module Stop<br>Target module: SCI3<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 29 | MSTPB29 | Serial Communication Interface 2 Module Stop<br>Target module: SCI2<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 30 | MSTPB30 | Serial Communication Interface 1 Module Stop<br>Target module: SCI1<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 31 | MSTPB31 | Serial Communication Interface 0 Module Stop<br>Target module: SCI0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. The MSTPBi bit must be written while the oscillation of the clock controlled by this bit is stabilized. For entering Software Standby mode after writing the MSTPBi bit, wait for CAN clock (CANMCLK) cycle after writing, and then execute a WFI instruction (i = 2).

Note 2. The MSTPBi bit must be written while the oscillation of the clock controlled by this bit is stabilized. For entering Software Standby mode after writing the MSTPBi bit, wait for two USB clock (USBCLK) cycles after writing, and then execute a WFI instruction (i = 11).

## 10.2.6 MSTPCRC : Module Stop Control Register C

Base address: MSTP = 0x4008_4000

Offset address: 0x008

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MSTPC31 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | MSTPC14 | MSTPC13 | MSTPC12 | — | — | — | MSTPC8 | — | — | — | — | MSTPC3 | — | MSTPC1 | MSTPC0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MSTPC0 | Clock Frequency Accuracy Measurement Circuit Module Stop*1<br>Target module: CAC<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 1 | MSTPC1 | Cyclic Redundancy Check Calculator Module Stop<br>Target module: CRC<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 2 | — | This bit is read as 1. The write value should be 1. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | MSTPC3 | Capacitive Touch Sensing Unit Module Stop<br>Target module: CTSU<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 7:4 | — | These bits are read as 1. The write value should be 1. | R/W |
| 8 | MSTPC8 | Serial Sound Interface Enhanced Module Stop<br>Target module: SSIE<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 11:9 | — | These bits are read as 1. The write value should be 1. | R/W |
| 12 | MSTPC12 | Secure Digital HOST IF / Multi Media Card 0 Module Stop<br>Target module: SDHI/MMC0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 13 | MSTPC13 | Data Operation Circuit Module Stop<br>Target module: DOC<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 14 | MSTPC14 | Event Link Controller Module Stop<br>Target module: ELC<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 30:15 | — | These bits are read as 1. The write value should be 1. | R/W |
| 31 | MSTPC31 | SCE9 Module Stop<br>Target module: SCE9<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. The MSTPC0 bit must be written while the oscillation of the clock to be controlled by this bit is stable. To enter Software Standby mode after writing this bit, wait for 2 cycles of the slowest clock from the clocks output by the oscillators, then execute a WFI instruction.

## 10.2.7 MSTPCRD : Module Stop Control Register D

Base address: MSTP = 0x4008_4000

Offset address: 0x00C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | MSTPD22 | — | MSTPD20 | — | — | — | MSTPD16 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | MSTPD14 | MSTPD13 | MSTPD12 | MSTPD11 | — | — | — | — | — | — | — | MSTPD3 | MSTPD2 | MSTPD1 | MSTPD0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MSTPD0 | Low Power Asynchronous General Purpose Timer 3 Module Stop*3<br>Target module: AGT3<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | MSTPD1 | Low Power Asynchronous General Purpose Timer 2 Module Stop*4<br>Target module: AGT2<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 2 | MSTPD2 | Low Power Asynchronous General Purpose Timer 1 Module Stop*1<br>Target module: AGT1<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 3 | MSTPD3 | Low Power Asynchronous General Purpose Timer 0 Module Stop*2<br>Target module: AGT0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 10:4 | — | These bits are read as 1. The write value should be 1. | R/W |
| 11 | MSTPD11 | Port Output Enable for GPT Group D Module Stop<br>Target module: POEGGD<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 12 | MSTPD12 | Port Output Enable for GPT Group C Module Stop<br>Target module: POEGGC<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 13 | MSTPD13 | Port Output Enable for GPT Group B Module Stop<br>Target module: POEGGB<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 14 | MSTPD14 | Port Output Enable for GPT Group A Module Stop<br>Target module: POEGGA<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 15 | — | This bit is read as 1. The write value should be 1. | R/W |
| 16 | MSTPD16 | 12-bit A/D Converter 0 Module Stop<br>Target module: ADC120<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 19:17 | — | These bits are read as 1. The write value should be 1. | R/W |
| 20 | MSTPD20 | 12-bit D/A Converter Module Stop<br>Target module: DAC12<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 21 | — | This bit is read as 1. The write value should be 1. | R/W |
| 22 | MSTPD22 | Temperature Sensor Module Stop<br>Target module: Temperature Sensor<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 31:23 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. When the count source is sub-clock oscillator or LOCO, AGT1 counting does not stop even if MSTPD2 is set to 1. If the count source is the sub-clock oscillator or LOCO, this bit must be set to 1 except when accessing the AGT1 registers.

Note 2. When the count source is sub-clock oscillator or LOCO, AGT0 counting does not stop even if MSTPD3 is set to 1. If the count source is the sub-clock oscillator or LOCO, this bit must be set to 1 except when accessing the AGT0 registers.

Note 3. When the count source is sub-clock oscillator or LOCO, AGT3 counting doesn't stop even if MSTPD0 is set to 1. If the count source is the sub-clock oscillator or LOCO, this bit must be set to 1 except when accessing the AGT3 registers.

Note 4. When the count source is sub-clock oscillator or LOCO, AGT2 counting doesn't stop even if MSTPD1 is set to 1. If the count source is the sub-clock oscillator or LOCO, this bit must be set to 1 except when accessing the AGT2 registers.

## 10.2.8 MSTPCRE : Module Stop Control Register E

Base address: MSTP = 0x4008_4000

Offset address: 0x010

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MSTPE31 | MSTPE30 | MSTPE29 | MSTPE28 | MSTPE27 | MSTPE26 | MSTPE25 | MSTPE24 | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MSTPE15 | MSTPE14 | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 13:0 | — | These bits are read as 1. The write value should be 1. | R/W |
| 14 | MSTPE14 | Low Power Asynchronous General Purpose Timer 5 Module Stop*1<br>Target module: AGT5<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 15 | MSTPE15 | Low Power Asynchronous General Purpose Timer 4 Module Stop*2<br>Target module: AGT4<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 23:16 | — | These bits are read as 1. The write value should be 1. | R/W |
| 24 | MSTPE24 | GPT7 Module Stop<br>Target module: GPT7<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 25 | MSTPE25 | GPT6 Module Stop<br>Target module: GPT6<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 26 | MSTPE26 | GPT5 Module Stop<br>Target module: GPT5<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 27 | MSTPE27 | GPT4 Module Stop<br>Target module: GPT4<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 28 | MSTPE28 | GPT3 Module Stop<br>Target module: GPT3<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 29 | MSTPE29 | GPT2 Module Stop<br>Target module: GPT2<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 30 | MSTPE30 | GPT1 Module Stop<br>Target module: GPT1<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |
| 31 | MSTPE31 | GPT0 Module Stop<br>Target module: GPT0<br>0: Cancel the module-stop state<br>1: Enter the module-stop state | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. When the count source is sub-clock oscillator or LOCO, AGT5 counting does not stop even if MSTPE14 is set to 1. If the count source is the sub-clock oscillator or LOCO, this bit must be set to 1 except when accessing the AGT5 registers.

Note 2. When the count source is sub-clock oscillator or LOCO, AGT4 counting does not stop even if MSTPE15 is set to 1. If the count source is the sub-clock oscillator or LOCO, this bit must be set to 1 except when accessing the AGT4 registers.

## 10.2.9 OPCCR : Operating Power Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x0A0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | OPCMTSF | — | — | OPCM[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | OPCM[1:0] | Operating Power Control Mode Select<br>0 0: High-speed mode<br>0 1: Setting prohibited<br>1 0: Setting prohibited<br>1 1: Low-speed mode | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | OPCMTSF | Operating Power Control Mode Transition Status Flag<br>0: Transition completed<br>1: During transition | R |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The OPCCR register is used to reduce power consumption in Normal and Sleep modes by specifying a lower operating frequency. For the procedure to change the operating power control modes, see section 10.5. Function for Lower Operating Power Consumption.

When transitioning from Software Standby mode to Normal or Snooze mode, the settings in the OPCCR.OPCM[1:0] and SOPCCR.SOPCM bits are as follows, regardless of their settings before entering Software Standby mode:

- OPCCR.OPCM[1:0] = 00b (High-speed mode)
- SOPCCR.SOPCM = 0b (not Subosc-speed mode).

If Software Standby mode is canceled before the transition to Software Standby completes, the OPCCR.OPCM[1:0] and SOPCCR.SOPCM bits retain their settings from before the WFI instruction is executed. If this causes any problem, set the MCU to High-speed mode during the exception handling procedure when canceling Software Standby mode.

### OPCM[1:0] bits (Operating Power Control Mode Select)

The OPCM[1:0] bits select the operating power control mode in Normal and Sleep modes. Table 10.4 shows the relationship between the operating power control modes and the OPCM[1:0] and SOPCM settings.

### OPCMTSF flag (Operating Power Control Mode Transition Status Flag)

The OPCMTSF flag indicates the switching control state when the operating power control mode is switched. This flag becomes 1 when the OPCM bit is written, and 0 when mode transition completes. Read this flag and confirm that it is 0 before proceeding.

## 10.2.10 SOPCCR : Sub Operating Power Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x0AA

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | SOPCMTSF | — | — | — | SOPCM |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SOPCM | Sub Operating Power Control Mode Select<br>0: Other than Subosc-speed mode<br>1: Subosc-speed mode | R/W |
| 3:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | SOPCMTSF | Operating Power Control Mode Transition Status Flag<br>0: Transition completed<br>1: During transition | R |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The SOPCCR register is used to reduce power consumption in Normal mode and Sleep mode. Setting this register initiates entry to and exit from Subosc-speed mode. Subosc-speed mode is available only when using the sub-clock oscillator or LOCO without dividing the frequency.

For the procedure to change operating power control modes, see section 10.5. Function for Lower Operating Power Consumption.

### SOPCM bit (Sub Operating Power Control Mode Select)

The SOPCM bit selects the operating power control mode in Normal and Sleep modes. Setting this bit to 1 allows transition to Subosc-speed mode. Setting this bit to 0 allows a return to the operating mode (operating mode set by OPCCR.OPCM[1:0]) that was active before the transition to Subosc-speed mode.

When transitioning from Software Standby mode to Normal mode or Snooze mode, the OPCCR.OPCM[1:0] and SOPCCR.SOPCM settings are as follows, regardless of their settings before entering Software Standby mode:

- OPCCR.OPCM[1:0] = 00b (High-speed mode)
- SOPCCR.SOPCM = 0b (not Subosc-speed mode)

If Software Standby mode is canceled before the transition to Software Standby completes, the OPCCR.OPCM[1:0] and SOPCCR.SOPCM bits retain their settings from before the WFI instruction is executed. If this causes any problem, set the MCU to High-speed mode during the exception handling procedure when canceling Software Standby mode.

Table 10.4 shows the relationship between the operating power control modes, the OPCM[1:0], and SOPCM bits settings.

### SOPCMTSF flag (Operating Power Control Mode Transition Status Flag)

The SOPCMTSF flag indicates the switching control state when the operating power control mode is switched to or from Subosc-speed mode. This flag becomes 1 when the SOPCM bit is written, and 0 when mode transition completes. Read this flag and confirm that it is 0 before proceeding.

Table 10.4 shows each operating power control mode.

**Table 10.4 Operating power control mode**

| Operating power control mode | OPCM[1:0] bits | SOPCM bit | Power consumption |
|---|---|---|---|
| High-speed mode | 00b | 0 | High |
| Low-speed mode | 11b | 0 | ↓ |
| Subosc-speed mode | xxb | 1 | Low |

For details about the operating frequency range, see section 47, Electrical Characteristics.

Each operating power control mode is described below.

- High-speed mode
  After a reset cancellation, the MCU is activated in this mode.
- Low-speed mode
  The following constraints apply in low-speed mode:
  – Programming and erasure operations for the flash memory are prohibited
  – Using the PLL or PLL2 is prohibited. See section 10.10.1. Register Access

  In this mode, lower power consumption is possible than in High-speed mode when the same operation is performed under the same conditions, such as operating frequency.
- Subosc-speed mode
  The following constraints apply in Subosc-speed mode:
  – Programming and erasure operations for the flash memory are prohibited
  – Reading of the data flash is prohibited
  – Using MOSC, PLL, PLL2, MOCO, or HOCO is prohibited. See section 10.10.1. Register Access
  – Using the divided clock for ICK or FCK is prohibited. See section 10.10.1. Register Access
  – Using the oscillation stop detection function of the main clock oscillator is prohibited.

  In this mode, lower power consumption is possible than in low-speed mode when the same operation is performed under the same conditions, such as operating frequency.

## 10.2.11 SNZCR : Snooze Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x092

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SNZE | — | — | — | — | — | SNZDTCEN | RXDREQEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RXDREQEN | RXD0 Snooze Request Enable<br>0: Ignore RXD0 falling edge in Software Standby mode<br>1: Detect RXD0 falling edge in Software Standby mode | R/W |
| 1 | SNZDTCEN | DTC Enable in Snooze mode<br>0: Disable DTC operation<br>1: Enable DTC operation | R/W |
| 6:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | SNZE | Snooze mode Enable<br>0: Disable Snooze mode<br>1: Enable Snooze mode | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

### RXDREQEN bit (RXD0 Snooze Request Enable)

The RXDREQEN bit specifies whether to detect a falling edge of the RXD0 pin in Software Standby mode. This bit can be used only when SCI0 is operating in asynchronous mode. To detect a falling edge of the RXD0 pin, set this bit before entering Software Standby mode. When this bit is set to 1, a falling edge of the RXD0 pin in Software Standby mode causes the MCU to enter Snooze mode.

### SNZDTCEN bit (DTC Enable in Snooze mode)

The SNZDTCEN bit specifies whether to use the DTC and SRAM in Snooze mode. To use the DTC and SRAM in Snooze mode, set this bit to 1 before entering Software Standby mode. When this bit is set to 1, the DTC can be activated by setting IELSRn register.

### SNZE bit (Snooze mode Enable)

The SNZE bit specifies whether to enable a transition from Software Standby mode to Snooze mode. To use Snooze mode, set this bit to 1 before entering Software Standby mode. When this bit is set to 1, a trigger as shown in Table 10.7 in Software Standby mode causes the MCU to enter Snooze mode. After the MCU transitions from Software Standby mode or Snooze mode to Normal mode, set 0 to the SNZE bit once then set it before re-entering Software Standby mode. For details, see section 10.8. Snooze Mode.

## 10.2.12 SNZEDCR0 : Snooze End Control Register 0

Base address: SYSC = 0x4001_E000

Offset address: 0x094

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SCI0UMTED | — | — | AD0UMTED | AD0MATED | DTCNZRED | DTCZRED | AGTUNFED |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | AGTUNFED | AGT1 Underflow Snooze End Enable<br>0: Disable the snooze end request<br>1: Enable the snooze end request | R/W |
| 1 | DTCZRED | Last DTC Transmission Completion Snooze End Enable<br>0: Disable the snooze end request<br>1: Enable the snooze end request | R/W |
| 2 | DTCNZRED | Not Last DTC Transmission Completion Snooze End Enable<br>0: Disable the snooze end request<br>1: Enable the snooze end request | R/W |
| 3 | AD0MATED | ADC12 Compare Match Snooze End Enable<br>0: Disable the snooze end request<br>1: Enable the snooze end request | R/W |
| 4 | AD0UMTED | ADC12 Compare Mismatch Snooze End Enable<br>0: Disable the snooze end request<br>1: Enable the snooze end request | R/W |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | SCI0UMTED | SCI0 Address Mismatch Snooze End Enable<br>0: Disable the snooze end request<br>1: Enable the snooze end request | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The SNZEDCR0 register controls the condition of switching from Snooze mode to Software Standby mode. In order to use a trigger shown in Table 10.8 as a condition to switch from Snooze mode to Software Standby mode, the corresponding bit in the SNZEDCR0 register must be set to 1.

The event that is used to return from snooze mode to normal mode as shown in Table 10.3 must not be enabled in the SNZEDCR0 register.

### AGTUNFED bit (AGT1 Underflow Snooze End Enable)

The AGTUNFED bit specifies whether to enable a transition from Snooze mode to Software Standby mode on an AGT1 underflow. For details on the trigger conditions, see section 22, Low Power Asynchronous General Purpose Timer (AGT).

### DTCZRED bit (Last DTC Transmission Completion Snooze End Enable)

The DTCZRED bit specifies whether to enable a transition from Snooze mode to Software Standby mode on completion of the last DTC transmission, that is, when CRA or CRB registers in the DTC is 0. For details on the trigger conditions, see section 17, Data Transfer Controller (DTC).

### DTCNZRED bit (Not Last DTC Transmission Completion Snooze End Enable)

The DTCNZRED bit specifies whether to enable a transition from Snooze mode to Software Standby mode on completion of each DTC transmission, that is, when CRA or CRB registers in the DTC is not 0. For details on the trigger conditions, see section 17, Data Transfer Controller (DTC).

### AD0MATED bit (ADC12 Compare Match Snooze End Enable)

The AD0MATED bit specifies whether to enable a transition from Snooze mode to Software Standby mode on an ADC120 event when a conversion result matches the expected data. For details on the trigger conditions, see section 37, 12-Bit A/D Converter (ADC12).

### AD0UMTED bit (ADC12 Compare Mismatch Snooze End Enable)

The AD0UMTED bit specifies whether to enable a transition from Snooze mode to Software Standby mode on an ADC120 event when the conversion result does not match the expected data. For details on the trigger conditions, see section 37, 12-Bit A/D Converter (ADC12).

### SCI0UMTED bit (SCI0 Address Mismatch Snooze End Enable)

The SCI0UMTED bit specifies whether to enable a transition from Snooze mode to Software Standby mode on an SCI0 event when an address received in Software Standby mode does not match the expected data. For details on the trigger conditions, see section 27, Serial Communications Interface (SCI). Only set this bit to 1 when SCI0 operates in asynchronous mode.

## 10.2.13 SNZEDCR1 : Snooze End Control Register 1

Base address: SYSC = 0x4001_E000

Offset address: 0x095

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | AGT3UNFED |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | AGT3UNFED | AGT3 underflow Snooze End Enable<br>0: Disable the Snooze End request<br>1: Enable the Snooze End request | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The SNZEDCR1 register controls the condition of switching from Snooze mode to Software Standby mode. In order to use a trigger shown in Table 10.8 as a condition to switch from Snooze mode to Software Standby mode, the corresponding bit in the SNZEDCR1 register must be set to 1.

The event that is used to return from Snooze mode to normal operating mode as shown in Table 10.3 must not be enabled in the SNZEDCR1 register.

### AGT3UNFED bit (AGT3 underflow Snooze End Enable)

The AGT3UNFED bit specifies whether to enable a transition from Snooze mode to Software Standby mode by an underflow of the AGT3. For the detail of the condition of the trigger, see section 22, Low Power Asynchronous General Purpose Timer (AGT).

## 10.2.14 SNZREQCR0 : Snooze Request Control Register 0

Base address: SYSC = 0x4001_E000

Offset address: 0x098

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | SNZREQEN30 | SNZREQEN29 | SNZREQEN28 | — | — | SNZREQEN25 | SNZREQEN24 | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SNZREQEN15 | SNZREQEN14 | SNZREQEN13 | SNZREQEN12 | SNZREQEN11 | SNZREQEN10 | SNZREQEN9 | SNZREQEN8 | SNZREQEN7 | SNZREQEN6 | SNZREQEN5 | SNZREQEN4 | SNZREQEN3 | SNZREQEN2 | SNZREQEN1 | SNZREQEN0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SNZREQEN0 | Enable IRQ0 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 1 | SNZREQEN1 | Enable IRQ1 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 2 | SNZREQEN2 | Enable IRQ2 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 3 | SNZREQEN3 | Enable IRQ3 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 4 | SNZREQEN4 | Enable IRQ4 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 5 | SNZREQEN5 | Enable IRQ5 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 6 | SNZREQEN6 | Enable IRQ6 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 7 | SNZREQEN7 | Enable IRQ7 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 8 | SNZREQEN8 | Enable IRQ8 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9 | SNZREQEN9 | Enable IRQ9 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 10 | SNZREQEN10 | Enable IRQ10 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 11 | SNZREQEN11 | Enable IRQ11 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 12 | SNZREQEN12 | Enable IRQ12 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 13 | SNZREQEN13 | Enable IRQ13 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 14 | SNZREQEN14 | Enable IRQ14 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 15 | SNZREQEN15 | Enable IRQ15 pin snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 23:16 | — | These bits are read as 0. The write value should be 0. | R/W |
| 24 | SNZREQEN24 | Enable RTC alarm snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 25 | SNZREQEN25 | Enable RTC period snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 27:26 | — | These bits are read as 0. The write value should be 0. | R/W |
| 28 | SNZREQEN28 | Enable AGT1 underflow snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 29 | SNZREQEN29 | Enable AGT1 compare match A snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 30 | SNZREQEN30 | Enable AGT1 compare match B snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 31 | — | This bit is read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The SNZREQCR0 register controls which trigger causes the MCU to switch from Software Standby mode to Snooze mode. If a trigger is selected as a request to cancel Software Standby mode by setting the WUPENn register, see section 13, Interrupt Controller Unit (ICU), the MCU enters Normal mode when the trigger is generated while the associated bit of the SNZREQCR0 is 1. The setting of the WUPENn register always has higher priority than the setting of the SNZREQCR0 register. For details, see section 10.8. Snooze Mode and section 13, Interrupt Controller Unit (ICU).

## 10.2.15 SNZREQCR1 : Snooze Request Control Register 1

Base address: SYSC = 0x4001_E000

Offset address: 0x088

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | SNZREQEN2 | SNZREQEN1 | SNZREQEN0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SNZREQEN0 | Enable AGT3 underflow snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 1 | SNZREQEN1 | Enable AGT3 compare match A snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 2 | SNZREQEN2 | Enable AGT3 compare match B snooze request<br>0: Disable the snooze request<br>1: Enable the snooze request | R/W |
| 31:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The SNZREQCR1 register controls which trigger causes the MCU to switch from Software Standby mode to Snooze mode. If a trigger is selected as a request to cancel Software Standby mode by setting the WUPENn register, see section 13, Interrupt Controller Unit (ICU), the MCU enters Normal mode when the trigger is generated while the associated bit of the SNZREQCR1 is 1. The setting of the WUPENn register always has higher priority than the setting of the SNZREQCR1 register. For details, see section 10.8. Snooze Mode and section 13, Interrupt Controller Unit (ICU).

## 10.2.16 DPSBYCR : Deep Software Standby Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x400

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DPSBY | IOKEEP | — | — | — | — | DEEPCUT[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | DEEPCUT[1:0] | Power-Supply Control<br>0 0: Power to the standby RAM, Low-speed on-chip oscillator, AGTn (n = 0 to 3), and USBFS resume detecting unit is supplied in Deep Software Standby mode.<br>0 1: Power to the standby RAM, Low-speed on-chip oscillator, AGT, and USBFS resume detecting unit is not supplied in Deep Software Standby mode.<br>1 0: Setting prohibited<br>1 1: Power to the standby RAM, Low-speed on-chip oscillator, AGT, and USBFS resume detecting unit is not supplied in Deep Software Standby mode. In addition, LVD is disabled and the low power function in a power-on reset circuit is enabled. | R/W |
| 5:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | IOKEEP | I/O Port Rentention<br>0: When the Deep Software Standby mode is canceled, the I/O ports are in the reset state.<br>1: When the Deep Software Standby mode is canceled, the I/O ports are in the same state as in the Deep Software Standby mode. | R/W |
| 7 | DPSBY | Deep Software Standby<br>0: Sleep mode (SBYCR.SSBY=0) / Software Standby mode (SBYCR.SSBY=1)<br>1: Sleep mode (SBYCR.SSBY=0) / Deep Software Standby mode (SBYCR.SSBY=1) | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The DPSBYCR register controls the Deep Software Standby mode.

DPSBYCR is not initialized by the internal reset signal that is the source to cancel the Deep Software Standby mode. For details, see section 5, Resets.

### DEEPCUT[1:0] bit (Power-Supply Control)

The DEEPCUT[1:0] bits control the internal power supply to the standby RAM, Low-speed on-chip oscillator, AGT, and USBFS resume detecting unit in Deep Software Standby mode. In addition, these bits control the state of LVD and power-on reset circuit in Deep Software Standby mode.

When a USBFS suspend/resume interrupt is used as a Deep Software Standby mode Cancelling source, the DEEPCUT[1:0] bits must be set to 00b.

When an LVD interrupt is used in Deep Software Standby mode, the DEEPCUT[1:0] bits must be set to 00b or 01b.

For lower power consumption, set the DEEPCUT[1:0] bits to 11b so that the LVD is stopped and the low power consumption function of the power-on reset circuit is enabled.

Regardless of the DEEPCUT[1:0] bit setting, during Deep Software Standby mode, internal power supply to SRAM other than standby SRAM is stopped.

When a Deep Software Standby mode is used, set DPSWCR.WTSTS bits depending on the value of DEEPCUT[1] before entering Deep Software Standby mode.

### IOKEEP bit (I/O Port Rentention)

In Deep Software Standby mode, I/O ports keep the same states as in the Software Standby mode. The IOKEEP bit specifies whether to reset the state of the I/O ports or not when the Deep Software Standby mode is canceled.

### DPSBY bit (Deep Software Standby)

The DPSBY bit controls transitions to Deep Software Standby mode.

When the WFI instruction is executed while SBYCR.SSBY bit and DPSBYCR.DPSBY bit are both 1, the MCU enters Deep Software Standby mode through Software Standby mode.

The DPSBY bit remains 1 when Deep Software Standby mode is canceled by certain pins which are sources of external pin interrupts (NMI, IRQn-DS (n = 0, 1, 4 to 12, 14, 15)) or a peripheral interrupt (RTC alarm, RTC interval, USB suspend/resume, voltage monitor 1, or voltage monitor 2). Write 0 to this bit to clear it.

The DPSBY bit setting is invalid when OFS0.IWDTSTPCTL bit is 0 (counting continues) regardless of the setting in OFS0.IWDTSTRT bit. In that case, even when SBYCR.SSBY bit is 1 and the DPSBY bit is 1, the transition after the execution of a WFI instruction is to Software Standby mode.

The setting of the DPSBY bit is invalid when voltage monitor 1 reset is enabled ( LVD1CR0.RI = 1) or when a voltage monitor 2 reset is enabled (LVD2CR0.RI = 1). In this case, even when the SBYCR.SSBY bit is 1 and the DPSBY bit is 1, the transition after the execution of a WFI instruction is to Software Standby mode.

## 10.2.17 DPSWCR : Deep Software Standby Wait Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x401

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | WTSTS[5:0] | | | | | |
| Value after reset: | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | WTSTS[5:0] | Deep Software Wait Standby Time Setting Bit<br>0x0E: Wait cycle for fast recovery<br>0x19: Wait cycle for slow recovery<br>Others: Setting prohibited | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The DPSWCR register sets the wait stabilization time when a Deep Software Standby mode is canceled by certain pins that are the sources of external pin interrupts or a peripheral interrupt such as RTC alarm, RTC interval, and USB suspend/ resume.

During a wait stabilization period set in this register, a Deep Software Standby reset occurs, and the MCU is initialized.

The DPSWCR register is not initialized with the internal reset signal by the cancellation of the Deep Software Standby mode. For details, see section 5, Resets.

When a Deep Software Standby mode is used, set DPSWCR.WTSTS bits according to the value of DPSBYCR.DEEPCUT[1] before entering Deep Software Standby mode.

When DPSBYCR.DEEPCUT[1]=0, you can set DPSWCR.WTSTS to the wait cycle for fast recovery.

When DPSBYCR.DEEPCUT[1]=1, you must set DPSWCR.WTSTS to the wait cycle for slow recovery.

## 10.2.18 DPSIER0 : Deep Software Standby Interrupt Enable Register 0

Base address: SYSC = 0x4001_E000

Offset address: 0x402

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DIRQ7E | DIRQ6E | DIRQ5E | DIRQ4E | — | — | DIRQ1E | DIRQ0E |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DIRQ0E | IRQ0-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | DIRQ1E | IRQ1-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | DIRQ4E | IRQ4-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 5 | DIRQ5E | IRQ5-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 6 | DIRQ6E | IRQ6-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 7 | DIRQ7E | IRQ7-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

DPSIER0 is not initialized by the internal reset signal used as Deep Software Standby mode Cancelling source. For details, see section 5, Resets.

After the setting of DPSIER0 is modified, an edge may be internally generated depending on the state of the pin, resulting in DPSIFR0 being set to 1. Therefore, DPSIFR0 should be cleared to 0 before entering Deep Software Standby mode.

## 10.2.19 DPSIER1 : Deep Software Standby Interrupt Enable Register 1

Base address: SYSC = 0x4001_E000

Offset address: 0x403

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DIRQ15E | DIRQ14E | — | DIRQ12E | DIRQ11E | DIRQ10E | DIRQ9E | DIRQ8E |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DIRQ8E | IRQ8-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 1 | DIRQ9E | IRQ9-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 2 | DIRQ10E | IRQ10-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 3 | DIRQ11E | IRQ11-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 4 | DIRQ12E | IRQ12-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6 | DIRQ14E | IRQ14-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 7 | DIRQ15E | IRQ15-DS Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

DPSIER1 is not initialized by the internal reset signal used as Deep Software Standby mode Cancelling source. For details, see section 5, Resets.

After the setting of DPSIER1 is modified, an edge may be internally generated depending on the state of the pin, resulting in DPSIFR1 being set to 1. Therefore, DPSIFR1 should be cleared to 0 before entering Deep Software Standby mode.

## 10.2.20 DPSIER2 : Deep Software Standby Interrupt Enable Register 2

Base address: SYSC = 0x4001_E000

Offset address: 0x404

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | DNMIE | DRTCAIE | DRTCIIE | DLVD2IE | DLVD1IE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DLVD1IE | LVD1 Deep Software Standby Cancel Signal Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 1 | DLVD2IE | LVD2 Deep Software Standby Cancel Signal Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 2 | DRTCIIE | RTC Interval interrupt Deep Software Standby Cancel Signal Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 3 | DRTCAIE | RTC Alarm interrupt Deep Software Standby Cancel Signal Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 4 | DNMIE | NMI Pin Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W*1 |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

Note 1. 1 can be written only once. Once 1 is written to this bit, subsequent write accesses are disabled.

DPSIER2 is not initialized by the internal reset signal used as Deep Software Standby mode Cancelling source. For details, see section 5, Resets.

After the setting of DPSIER2 is modified, an edge may be internally generated depending on the state of the pin, resulting in DPSIFR2 being set to 1. Therefore, DPSIFR2 should be cleared to 0 before entering Deep Software Standby mode.

## 10.2.21 DPSIER3 : Deep Software Standby Interrupt Enable Register 3

Base address: SYSC = 0x4001_E000

Offset address: 0x405

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | DAGT 3IE | DAGT 1IE | — | DUSB FS0IE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DUSBFS0IE | USBFS0 Suspend/Resume Deep Software Standby Cancel Signal Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 1 | — | This bit is read as 0. The write value should be 0. | R/W |
| 2 | DAGT1IE | AGT1 Underflow Deep Software Standby Cancel Signal Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 3 | DAGT3IE | AGT3 Underflow Deep Software Standby Cancel Signal Enable<br>0: Cancelling Deep Software Standby mode is disabled<br>1: Cancelling Deep Software Standby mode is enabled | R/W |
| 7:4 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

DPSIER3 is not initialized by the internal reset signal used as Deep Software Standby mode Cancelling source. For details, see section 5, Resets.

After the setting of DPSIER3 is modified, an edge may be internally generated depending on the state of the pin, resulting in DPSIFR3 being set to 1. Therefore, DPSIFR3 should be cleared to 0 before entering Deep Software Standby mode.

## 10.2.22 DPSIFR0 : Deep Software Standby Interrupt Flag Register 0

Base address: SYSC = 0x4001_E000

Offset address: 0x406

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DIRQ7F | DIRQ6F | DIRQ5F | DIRQ4F | — | — | DIRQ1F | DIRQ0F |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DIRQ0F | IRQ0-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 1 | DIRQ1F | IRQ1-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | DIRQ4F | IRQ4-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5 | DIRQ5F | IRQ5-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 6 | DIRQ6F | IRQ6-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 7 | DIRQ7F | IRQ7-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Writing 0 clears the flag. Writing 1 is ignored.
Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

Each flag is set to 1 when a cancel request specified by DPSIEGR0 is generated.

Each flag may be set to 1 when a cancel request is generated in any mode (not only in Deep Software Standby mode) or when the setting of DPSIER0 is modified. Therefore, a transition to Deep Software Standby mode should be made after DPSIFR0 is cleared to 0x00.

To clear DPSIFR0 to 0x00 after modifying DPSIER0, wait for at least 6 PCLKB cycles, read DPSIFR0, and then write 0 to DPSIFR0. Six or more PCLKB cycles can be secured, for example, by reading DPSIER0.

DPSIFR0 is not initialized by the internal reset signal used as Deep Software Standby mode Cancelling source. To clear DPSIFR0 to 0x00 after modifying DPSIER0, wait for at least 6 PCLKB cycles, read DPSIFR0, and then write 0 to DPSIFR0. Six or more PCLKB cycles can be secured, for example, by reading DPSIER0. For details, see section 5, Resets.

#### DIRQnF flag (IRQn-DS Pin Deep Software Standby Cancel Flag) (n = 0, 1, 4 to 7)

The DIRQnF flag indicates that a cancel request by the IRQn-DS pin has been generated.

[Setting condition]

A cancel request by the IRQn-DS pin specified by DPSIEGR0 is generated.

[Clearing condition]

Writing 0 to each flag after 1 is read.

## 10.2.23 DPSIFR1 : Deep Software Standby Interrupt Flag Register 1

Base address: SYSC = 0x4001_E000

Offset address: 0x407

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DIRQ15F | DIRQ14F | — | DIRQ12F | DIRQ11F | DIRQ10F | DIRQ9F | DIRQ8F |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DIRQ8F | IRQ8-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 1 | DIRQ9F | IRQ9-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 2 | DIRQ10F | IRQ10-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | DIRQ11F | IRQ11-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 4 | DIRQ12F | IRQ12-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | DIRQ14F | IRQ14-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 7 | DIRQ15F | IRQ15-DS Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Writing 0 clears the flag. Writing 1 is ignored.
Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

Each flag is set to 1 when a cancel request specified by DPSIEGR1 is generated.

Each flag may be set to 1 when a cancel request is generated in any mode (not only in Deep Software Standby mode) or when the setting of DPSIER1 is modified. Therefore, a transition to Deep Software Standby mode should be made after DPSIFR1 is cleared to 0x00.

To clear DPSIFR1 to 0x00 after modifying DPSIER1, wait for at least 6 PCLKB cycles, read DPSIFR1, and then write 0 to DPSIFR1. Six or more PCLKB cycles can be secured, for example, by reading DPSIER1.

DPSIFR1 is not initialized by the internal reset signal used as Deep Software Standby mode Cancelling source. For details, see section 5, Resets.

#### DIRQnF flag (IRQn-DS Pin Deep Software Standby Cancel Flag) (n = 8 to 12, 14, 15)

The DIRQnF flag indicates that a cancel request by the IRQn-DS pin has been generated.

[Setting condition]

A cancel request by the IRQn-DS pin specified by DPSIEGR1 is generated.

[Clearing condition]

Writing 0 to each flag after 1 is read.

### 10.2.24 DPSIFR2 : Deep Software Standby Interrupt Flag Register 2

Base address: SYSC = 0x4001_E000

Offset address: 0x408

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | DNMIF | DRTCAIF | DRTCIIF | DLVD2IF | DLVD1IF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DLVD1IF | LVD1 Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 1 | DLVD2IF | LVD2 Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2 | DRTCIIF | RTC Interval Interrupt Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 3 | DRTCAIF | RTC Alarm Interrupt Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 4 | DNMIF | NMI Pin Deep Software Standby Cancel Flag<br>0: The cancel request is not generated<br>1: The cancel request is generated | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Writing 0 clears the flag. Writing 1 is ignored.
Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

Each flag is set to 1 when a cancel request specified by DPSIEGR2 is generated.

Each flag may be set to 1 when a cancel request is generated in any mode (not only in Deep Software Standby mode) or when the setting of DPSIER2 is modified. Therefore, a transition to Deep Software Standby mode should be made after DPSIFR2 is cleared to 0x00.

To clear DPSIFR2 to 0x00 after modifying DPSIER2, wait for at least 6 PCLKB cycles, read DPSIFR2, and then write 0 to DPSIFR2. Six or more PCLKB cycles can be secured, for example, by reading DPSIER2.

DPSIFR2 is not initialized by the internal reset signal used as Deep Software Standby mode Cancelling source. For details, see section 5, Resets.

### DLVDmIF flag (LVDm Deep Software Standby Cancel Flag) (m = 1 to 2)

The DLVDmIF flag indicates that a cancel request by the voltage monitor m signal has been generated.

[Setting condition]

A cancel request is generated by the voltage monitor m signal that is selected in DPSIEGR2.

[Clearing condition]

Writing 0 to each flag after 1 is read.

### DRTCIIF flag (RTC Interval Interrupt Deep Software Standby Cancel Flag)

This flag indicates that a cancel request by the RTC interval interrupt signal has been generated.

[Setting condition]

A cancel request by the RTC interval interrupt signal is generated

[Clearing condition]

Writing 0 to each flag after 1 is read.

### DRTCAIF flag (RTC Alarm Interrupt Deep Software Standby Cancel Flag)

This flag indicates that a cancel request by the RTC alarm interrupt signal has been generated.

[Setting condition]

A cancel request by the RTC alarm interrupt signal is generated

[Clearing condition]

Writing 0 to each flag after 1 is read.

### DNMIF flag (NMI Pin Deep Software Standby Cancel Flag)

This flag indicates that a cancel request by the NMI pin has been generated.

[Setting condition]

A cancel request by the NMI pin specified by DPSIEGR2 is generated

[Clearing condition]

Writing 0 to each flag after 1 is read.

## 10.2.25 DPSIFR3 : Deep Software Standby Interrupt Flag Register 3

Base address: SYSC = 0x4001_E000

Offset address: 0x409

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | DAGT 3IF | DAGT 1IF | — | DUSB FS0IF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DUSBFS0IF | USBFS0 Suspend/Resume Deep Software Standby Cancel Flag<br>0: The cancel request is not generated.<br>1: The cancel request is generated. | R/W |
| 1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 2 | DAGT1IF | AGT1 Underflow Deep Software Standby Cancel Flag<br>0: The cancel request is not generated.<br>1: The cancel request is generated. | R/W |
| 3 | DAGT3IF | AGT3 Underflow Deep Software Standby Cancel Flag<br>0: The cancel request is not generated.<br>1: The cancel request is generated. | R/W |
| 7:4 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Writing 0 clears the flag. Writing 1 is ignored.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

Each flag is set to 1 when the corresponding cancel request is generated.

Each flag may be set to 1 when a cancel request is generated in any mode (not only in Deep Software Standby mode) or when the setting of DPSIER3 is modified. Therefore, a transition to Deep Software Standby mode should be made after DPSIFR3 is cleared to 0x00.

To clear DPSIFR3 to 0x00 after modifying DPSIER3, wait for at least 6 PCLKB cycles, read DPSIFR3, and then write 0 to DPSIFR3. Six or more PCLKB cycles can be secured, for example, by reading DPSIER3.

DPSIFR3 is not initialized by the internal reset signal used as Deep Software Standby mode Cancelling source. For details, see section 5, Resets.

### DUSBFS0IF flag (USBFS0 Suspend/Resume Deep Software Standby Cancel Flag)

The DUSBFS0IF flag is the flag for USBFS0 that indicates that a cancel request by the USBFS0 suspend/resume has been generated.

[Setting condition]

A cancel request by the USBFS0 suspend/resume is generated

[Clearing condition]

Writing 0 to each flag after 1 is read.

### DAGT1IF flag (AGT1 Underflow Deep Software Standby Cancel Flag)

This flag indicates that a cancel request by the AGT1 underflow has been generated.

[Setting condition]

A cancel request by the AGT1 underflow is generated

[Clearing condition]

Writing 0 to each flag after 1 is read.

**DAGT3IF flag (AGT3 Underflow Deep Software Standby Cancel Flag)**

This flag indicates that a cancel request by the AGT3 underflow has been generated.

[Setting condition]

A cancel request by the AGT3 underflow is generated

[Clearing condition]

Writing 0 to each flag after 1 is read.

## 10.2.26 DPSIEGR0 : Deep Software Standby Interrupt Edge Register 0

Base address: SYSC = 0x4001_E000

Offset address: 0x40A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DIRQ7EG | DIRQ6EG | DIRQ5EG | DIRQ4EG | — | — | DIRQ1EG | DIRQ0EG |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DIRQ0EG | IRQ0-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge<br>1: A cancel request is generated at a rising edge | R/W |
| 1 | DIRQ1EG | IRQ1-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge<br>1: A cancel request is generated at a rising edge | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | DIRQ4EG | IRQ4-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge<br>1: A cancel request is generated at a rising edge | R/W |
| 5 | DIRQ5EG | IRQ5-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge<br>1: A cancel request is generated at a rising edge | R/W |
| 6 | DIRQ6EG | IRQ6-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge<br>1: A cancel request is generated at a rising edge | R/W |
| 7 | DIRQ7EG | IRQ7-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge<br>1: A cancel request is generated at a rising edge | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

DPSIEGR0 is not initialized by the internal reset signal that is the source to cancel the Deep Software Standby mode. For details, see section 5, Resets.

## 10.2.27 DPSIEGR1 : Deep Software Standby Interrupt Edge Register 1

Base address: SYSC = 0x4001_E000

Offset address: 0x40B

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DIRQ15EG | DIRQ14EG | — | DIRQ12EG | DIRQ11EG | DIRQ10EG | DIRQ9EG | DIRQ8EG |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DIRQ8EG | IRQ8-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge.<br>1: A cancel request is generated at a rising edge. | R/W |
| 1 | DIRQ9EG | IRQ9-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge.<br>1: A cancel request is generated at a rising edge. | R/W |
| 2 | DIRQ10EG | IRQ10-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge.<br>1: A cancel request is generated at a rising edge | R/W |
| 3 | DIRQ11EG | IRQ11-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge.<br>1: A cancel request is generated at a rising edge. | R/W |
| 4 | DIRQ12EG | IRQ12-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge.<br>1: A cancel request is generated at a rising edge. | R/W |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | DIRQ14EG | IRQ14-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge.<br>1: A cancel request is generated at a rising edge. | R/W |
| 7 | DIRQ15EG | IRQ15-DS Pin Edge Select<br>0: A cancel request is generated at a falling edge.<br>1: A cancel request is generated at a rising edge. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

DPSIEGR1 is not initialized by the internal reset signal that is the source to cancel the Deep Software Standby mode. For details, see section 5, Resets.

## 10.2.28 DPSIEGR2 : Deep Software Standby Interrupt Edge Register 2

Base address: SYSC = 0x4001_E000

Offset address: 0x40C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | DNMIEG | — | — | DLVD2EG | DLVD1EG |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DLVD1EG | LVD1 Edge Select<br>0: A cancel request is generated when VCC < $V_{det1}$ (fall) is detected<br>1: A cancel request is generated when VCC ≥ $V_{det1}$ (rise) is detected | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | DLVD2EG | LVD2 Edge Select<br>0: A cancel request is generated when VCC < $V_{det2}$ (fall) is detected<br>1: A cancel request is generated when VCC ≥ $V_{det2}$ (rise) is detected | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | DNMIEG | NMI Pin Edge Select<br>0: A cancel request is generated at a falling edge<br>1: A cancel request is generated at a rising edge | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

DPSIEGR2 is not initialized by the internal reset signal that is the source to cancel the Deep Software Standby mode. For details, see section 5, Resets.

## 10.2.29 SYOCDCR : System Control OCD Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x040E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DBGEN | — | — | — | — | — | — | DOCDF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DOCDF | Deep Software Standby OCD flag<br>0: DBIRQ is not generated<br>1: DBIRQ is generated | R/W*1 |
| 6:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | DBGEN | Debugger Enable bit<br>Set to 1 first in on-chip debug mode.<br>0: On-chip debugger is disabled<br>1: On-chip debugger is enabled | R/W |

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

Note 1. Writing 0 clears the flag. Writing 1 is ignored

This register is not controlled by any security attribute register (eg. LPMSAR, DPFSAR).

SYOCDCR can be written when DBGSTR.CDBGPWRUPREQ = 1 (the debugger is connected).

SYOCDCR is not initialized by the internal reset signal that is the source to cancel the Deep Software Standby mode.

### DOCDF flag (Deep Software Standby OCD flag)

DOCDF flag indicates that a cancel request of Deep Software Standby mode by the MCUCTRL.DBIRQ bit has been generated. DOCDF flag is set to 1 when a cancel request is generated. This flag may be set to 1 when a cancel request is generated in any mode (not only in Deep Software Standby mode). Therefore, a transition to Deep Software Standby mode must be made after DOCDF flag is cleared to 0.

[Setting condition]

- A cancel request by the MCUCTRL.DBIRQ is generated

[Clearing condition]

- Writing 0 to the flag after reading the bit as 1
- When DBGEN bit is 0

**DBGEN bit (Debugger Enable bit)**

The DBGEN bit enables the on-chip debug mode. This bit must be set to 1 first in the on-chip debugger mode.

[Setting condition]

- Writing 1 to the bit when the debugger is connected.

[Clearing condition]

- Power-on reset is generated
- Writing 0 to the bit.

Note: Certain restrictions apply in terms of the MCU states in which the DBGEN bit can be set to 1. For details, see section 2.13.2. Restrictions on Connecting an OCD emulator.

Note: DBGEN setting should be changed when the LDOSCR.LDOSTPn (n = 0 ,1) bits are 0.

## 10.2.30 LDOSCR:LDO Stop Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x0440

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | LDOSTP1 | LDOSTP0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | LDOSTP0 | LDO0 Stop<br>0: LDO0 is enabled<br>1: LDO0 is stopped | R/W |
| 1 | LDOSTP1 | LDO1 Stop<br>0: LDO1 is enabled<br>1: LDO1 is stopped | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

The internal voltage is supplied from multiple built-in linear regulators (LDO). If you do not use the SCE9, LDO self-power consumption can be reduced by setting LDOSCR register to stop LDO0 and LDO1. The setting of the LDOSTPn bit must be changed under the following conditions:

- PLL, PLL2 and MOSC are stopped
- CPU clock and peripheral module clock are 2MHz or less
- All modules other than the DMAC, DTC, and SRAM modules are in the module-stop state

After changing the value of LDOSTPn, you must also insert a wait time of at least 10 µs to stabilize the power supply capacity of the LDO. The recommended method to measure the wait time is through software. Be sure to consider the worst-case conditions to ensure that the required wait time elapses.

This register is initialized by all reset factors. In addition, this register is initialized in the Deep Software Standby mode.

Figure 10.2 shows the flow for stopping LDO0 and LDO1, Figure 10.3 shows the flow for restarting LDO0 and LDO1.

Note: SYOCDCR.DBGEN setting should be changed when the LDOSTPn bits are 0. It is possible to set the LDOSTPn bits to 1 after changing the SYOCDCR.DBGEN setting.

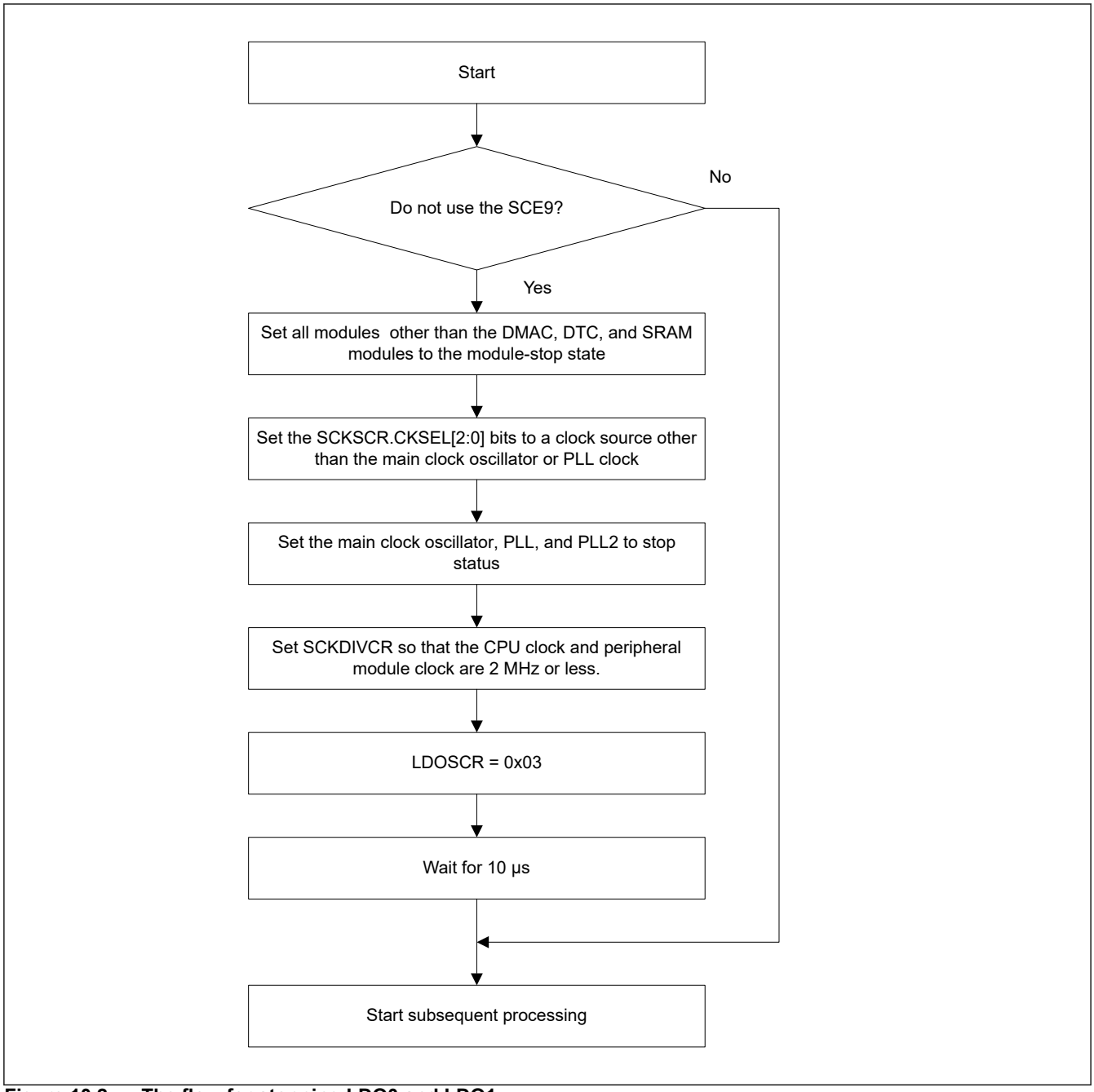



**Figure 10.2 The flow for stopping LDO0 and LDO1**

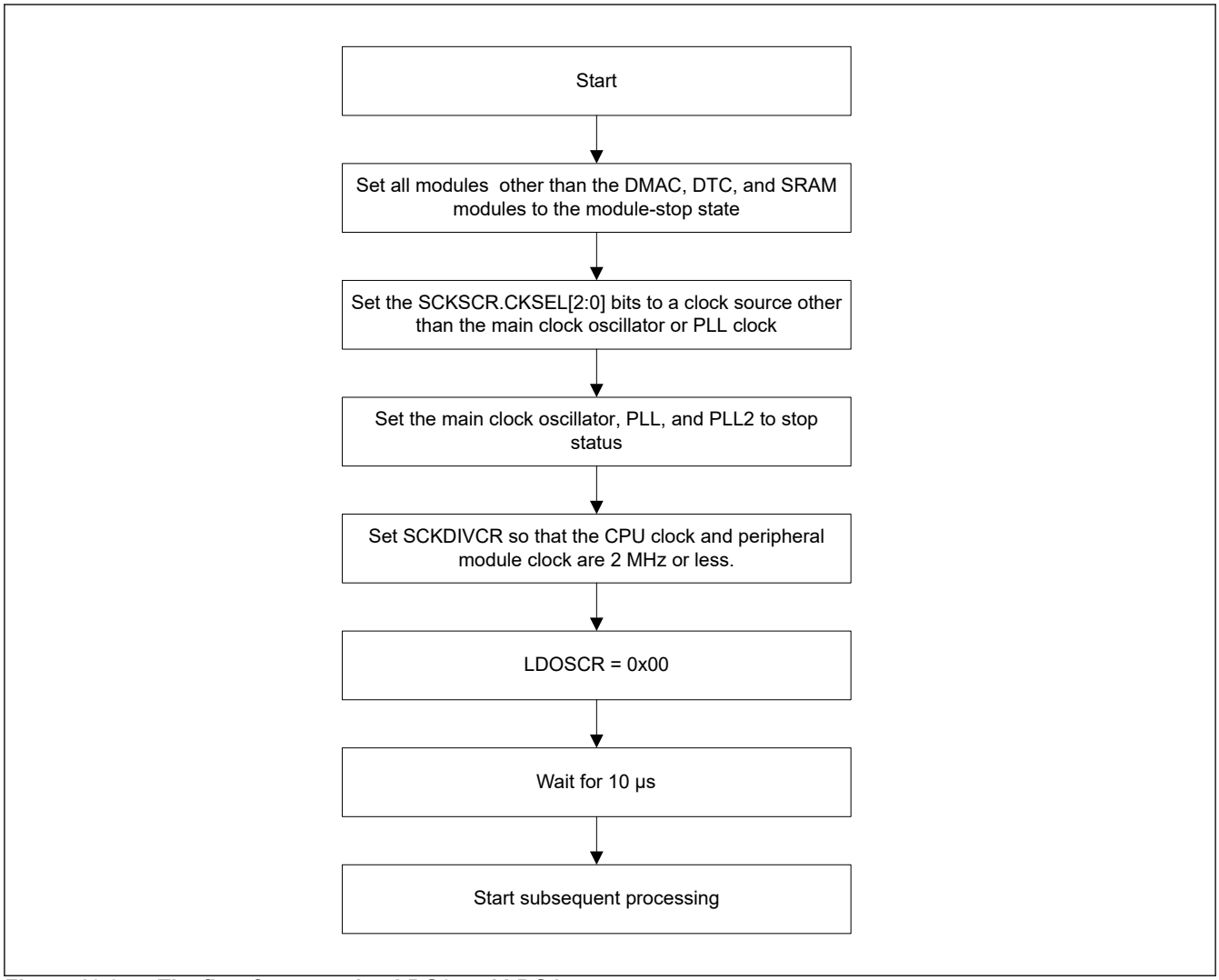



**Figure 10.3 The flow for restarting LDO0 and LDO1**

## 10.2.31 PL2LDOSCR:PLL2-LDO Stop Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x0444

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | PL2LDOSTP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PL2LDOSTP | PLL2-LDO Stop<br>0: PLL2-LDO is enabled<br>1: PLL2-LDO is stopped | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

This MCU has a built-in LDO dedicated to PLL2. If you never use PLL2, the PL2LDOSCR can be configured to stop the LDO dedicated to PLL2 in order to reduce the quiescent current of that LDO.

Do not write 0 again after writing 1 to PL2LDOSTP.

When PL2LDOSTP is 1, do not write 0 to PLL2CR.PLL2STP.

When PLL2CR.PLL2STP is 0, do not write 1 to PL2LDOSTP.

This register is initialized by only POR reset.

## 10.3 Reducing Power Consumption by Switching Clock Signals

The clock frequency changes when the SCKDIVCR register is set.

For information on module and clock associations, see section 8.2.2. SCKDIVCR : System Clock Division Control Register.

## 10.4 Module-Stop Function

The module stop function can stop the clock supply set for each peripheral module.

When the MSTPmi bit (m = A to E, i = 31 to 0) in MSTPCRn (n = A to E) is set to 1, the specified module stops operating and enters the module-stop state, but the CPU continues to operate independently. Setting the MSTPmi bit to 0 cancels the module-stop state, allowing the module to resume operation at the end of the bus cycle.

After a reset is canceled, all modules other than the DMAC, DTC and SRAMn modules are placed in the module-stop state. Do not access the module while the corresponding MSTPmi bit is 1. Additionally, do not set 1 to the MSTPmi bit while the corresponding module is accessed.

## 10.5 Function for Lower Operating Power Consumption

By selecting an appropriate operating power consumption control mode according to the operating frequency, power consumption can be reduced in Normal mode, Sleep mode, and Snooze mode.

### 10.5.1 Setting Operating Power Control Mode

Ensure the operating condition such as the frequency range is always within the specified range before and after switching the operating power control modes.

This section provides example procedures for switching operating power control modes.

**Table 10.5 Available oscillators in each mode**

| Mode | Oscillator | | | | | | |
|---|---|---|---|---|---|---|---|
| | PLL, PLL2 | High-speed on-chip oscillator | Middle-speed on-chip oscillator | Low-speed on-chip oscillator | Main clock oscillator | Sub-clock oscillator | IWDT-dedicated on-chip oscillator |
| High-speed | Available | Available | Available | Available | Available | Available | Available |
| Low-speed | N/A | Available | Available | Available | Available | Available | Available |
| Subosc-speed | N/A | N/A | N/A | Available | N/A | Available | Available |

#### (1) Switching from a higher power mode to a lower power mode

Example 1: From High-speed mode to Low-speed mode:

(Operation begins in High-speed mode)

1. Change the oscillator to what is used in Low-speed mode. Set the frequency of each clock lower than or equal to the maximum operating frequency in Low-speed mode.
2. Turn off the oscillator that is not required in Low-speed mode.
3. Confirm that the OPCCR.OPCMTSF flag is 0 (indicates transition completed).
4. Set the OPCCR.OPCM[1:0] bits to 11b (Low-speed mode).
5. Confirm that OPCCR.OPCMTSF flag is 0 (indicates transition completed).

(Operation is now in Low-speed mode)

Example 2: From High-speed mode to Subosc-speed mode

(Operation begins in High-speed mode)

1. Switch the clock source to sub-clock oscillator. Turn off PLL, PLL2, HOCO, MOCO, LOCO and main oscillator.
2. Confirm that all clock sources other than the sub-clock oscillator are stopped.
3. Confirm that the SOPCCR.SOPCMTSF flag is 0 (indicates transition completed).
4. Set the SOPCCR.SOPCM bit to 1 (Subosc-speed mode).
5. Confirm that the SOPCCR.SOPCMTSF flag is 0 (indicates transition completed).

(Operation is now in Subosc-speed mode)

#### (2) Switching from a lower power mode to a higher power mode

Example 1: From Subosc-speed mode to High-speed mode

(Operation begins in Subosc-speed mode)

1. Confirm that the SOPCCR.SOPCMTSF flag is 0 (indicates transition completed).
2. Set the SOPCCR.SOPCM bit to 0 (High-speed mode).
3. Confirm that the SOPCCR.SOPCMTSF flag is 0 (indicates transition completed).
4. Turn on the required oscillator in High-speed mode.
5. Set the frequency of each clock lower than or equal to the maximum operating frequency for High-speed mode.

(Operation is now in High-speed mode)

Example 2: From Low-speed mode to High-speed mode

(Operation begins in Low-speed mode)

1. Confirm that the OPCCR.OPCMTSF flag is 0 (indicates transition completed).
2. Set the OPCCR.OPCM[1:0] bits to 00b (High-speed mode).
3. Confirm that the OPCCR.OPCMTSF flag is 0 (indicates transition completed).
4. Turn on any required oscillator in High-speed mode.
5. Set the frequency of each clock lower than or equal to the maximum operating frequency for High-speed mode.

(Operation is now in High-speed mode)

## 10.6 Sleep Mode

### 10.6.1 Transitioning to Sleep Mode

When a WFI instruction is executed while SBYCR.SSBY bit is 0, the MCU enters Sleep mode. In this mode, the CPU stops operating but the contents of its internal registers are retained. Other peripheral functions do not stop. Available resets or interrupts in Sleep mode cause the MCU to cancel Sleep mode. All interrupt sources are available. If using an interrupt to cancel Sleep mode, you must set the associated IELSRn register before executing a WFI instruction. For details, see section 13, Interrupt Controller Unit (ICU).

Counting by IWDT stops when the MCU enters Sleep mode while the IWDT is in auto start mode and the OFS0.IWDTSTPCTL bit is 1 (IWDT stops in Sleep mode, Software Standby mode, or Snooze mode).

Counting by IWDT continues when the MCU enters Sleep mode while the IWDT is in auto start mode and the OFS0.IWDTSTPCTL bit is 0 (IWDT does not stop in Sleep mode, Software Standby mode, or Snooze mode).

Counting by WDT stops when the MCU enters Sleep mode while the WDT is in auto start mode and the OFS0.WDTSTPCTL bit is 1 (WDT stops in Sleep mode). Similarly, counting by WDT stops when the MCU enters Sleep mode while the WDT is in register start mode and the WDTCSTPR.SLCSTP bit is 1 (WDT stops in Sleep mode).

Counting by WDT continues when the MCU enters Sleep mode while the WDT is in auto start mode and the OFS0.WDTSTPCTL bit is 0 (WDT does not stop in Sleep mode). Similarly, counting by WDT continues when the MCU enters Sleep mode while the WDT is in register start mode and the WDTCSTPR.SLCSTP bit is 0 (WDT does not stop in Sleep mode).

### 10.6.2 Canceling Sleep Mode

Sleep mode is canceled by:

- An interrupt
- A RES pin reset
- A power-on reset
- A voltage monitor reset
- An SRAM parity error reset
- An SRAM ECC error reset
- A bus master MPU error reset
- A TrustZone error reset
- A reset caused by an IWDT or a WDT underflow

The operations are as follows:

1. Canceling by an interrupt
   When an interrupt request is generated, Sleep mode is canceled and the MCU starts the interrupt handling.
2. Canceling by RES pin reset
   When the RES pin is driven low, the MCU enters the reset state. Be sure to keep the RES pin low for the time period specified in section 47, Electrical Characteristics. When the RES pin is driven high after the specified time period, the CPU starts the reset exception handling.
3. Canceling by IWDT reset
   - Sleep mode is canceled by an internal reset generated by an IWDT underflow and the MCU starts the reset exception handling. However, IWDT stops in Sleep mode and an internal reset for canceling Sleep mode is not generated in the following conditions:
   - OFS0.IWDTSTRT = 0 and OFS0.IWDTSTPCTL = 1.
4. Canceling by WDT reset
   Sleep mode is canceled by an internal reset generated by a WDT underflow and the MCU starts the reset exception handling. However, WDT stops in Sleep mode even when counting in Normal mode and an internal reset for canceling Sleep mode is not generated in the following conditions:
   - OFS0.WDTSTRT = 0 (auto start mode) and OFS0.WDTSTPCTL = 1
   - OFS0.WDTSTRT = 1 (register start mode) and WDTCSTPR.SLCSTP = 1.
5. Canceling by other resets available in Sleep mode
   Sleep mode is canceled by other resets and the MCU starts the reset exception handling.

Note: For details on proper setting of the interrupts, see section 13, Interrupt Controller Unit (ICU).

## 10.7 Software Standby Mode

### 10.7.1 Transitioning to Software Standby Mode

When a WFI instruction is executed while SBYCR.SSBY bit is 1 and DPSBYCR.DPSBY bit is 0, the MCU enters Software Standby mode. In this mode, the CPU, most of the on-chip peripheral functions and the oscillators stop. However, the contents of the CPU internal registers and SRAM data, the states of on-chip peripheral functions and the I/O ports are retained. Software Standby mode allows significant reduction in power consumption because most of the oscillators stops in this mode. Table 10.2 shows the status of each on-chip peripheral functions and oscillators. Available resets or interrupts in Software Standby mode make the MCU to cancel Software Standby mode. See Table 10.3 for available interrupt sources and section 13.2.18. WUPEN0 : Wake Up Interrupt Enable Register 0, section 13.2.19. WUPEN1 : Wake Up interrupt

enable register 1 for information on waking up the MCU from Software Standby mode. If using an interrupt to cancel an interrupt, you must set the associated IELSRn register before executing a WFI instruction. For details, see section 13, Interrupt Controller Unit (ICU).

Clear DMAST.DMST bit and DTCST.DTCST bit to 0 before executing WFI instruction except when using DTC in Snooze mode. If DTC is required in Snooze mode, set DTCST.DTCST bit to 1 before executing a WFI instruction.

Counting by the IWDT stops if the MCU enters Software Standby mode while the IWDT is in auto start mode and the OFS0.IWDTSTPCTL bit is 1 (IWDT stops in Sleep, Software Standby or Snooze mode).

Counting by the IWDT continues if the MCU enters Software Standby mode while the IWDT is in auto start mode and the OFS0.IWDTSTPCTL bit is 0 (IWDT does not stop in Sleep, Software Standby or Snooze mode).

WDT stops counting when the MCU enters Software Standby mode because the PCLKB stops.

Do not enter Software Standby mode while OSTDCR.OSTDE = 1 (oscillation stop detection function is enabled). To enter Software Standby mode, execute a WFI instruction after disabling the oscillation stop detection function (OSTDCR.OSTDE = 0). In case of executing WFI instruction while OSTDCR.OSTDE = 1, the MCU enters Sleep mode even if SBYCR.SSBY = 1.

Do not enter Software Standby mode while the flash memory is programming or erasing. To enter Software Standby mode, execute a WFI instruction after programming or erasing procedure completes.

When the PLL is selected as the clock source, set the following modules to the module-stop state before executing a WFI instruction:

- SCE9

Table 10.6 shows the setting of the related control bits and the modes to enter after executing WFI instruction.

Figure 10.4 shows an example flow for transitioning to Software Standby or Deep Software Standby mode.

**Table 10.6 Bit settings that affect modes when executing a WFI instruction**

| | | SBYCR.SSBY and PSBYCR.DPSBY bit settings | | | |
|---|---|---|---|---|---|
| | | SSBY = 0, DPSBY = 0 | SSBY = 0, DPSBY = 1 | SSBY = 1, DPSBY = 0 | SSBY = 1, DPSBY = 1 |
| OSTDCR.OSTDE | 0 | Sleep | Sleep | Software Standby | Deep Software Standby |
| | 1 | | | Sleep | Sleep |
| FENTRYR.FENTRYC FENTRYR.FENTRYD | 0 | Sleep | Sleep | Software Standby | Deep Software Standby |
| | 1 | | | Sleep | Sleep |
| OFS0.IWDTSTPCTL | 0 | Sleep | Sleep | Software Standby | Software Standby |
| | 1 | | | | Deep Software Standby |
| LVD1CR0.RI | 0 | Sleep | Sleep | Software Standby | Deep Software Standby |
| | 1 | | | | Software Standby |
| LVD2CR0.RI | 0 | Sleep | Sleep | Software Standby | Deep Software Standby |
| | 1 | | | | Software Standby |

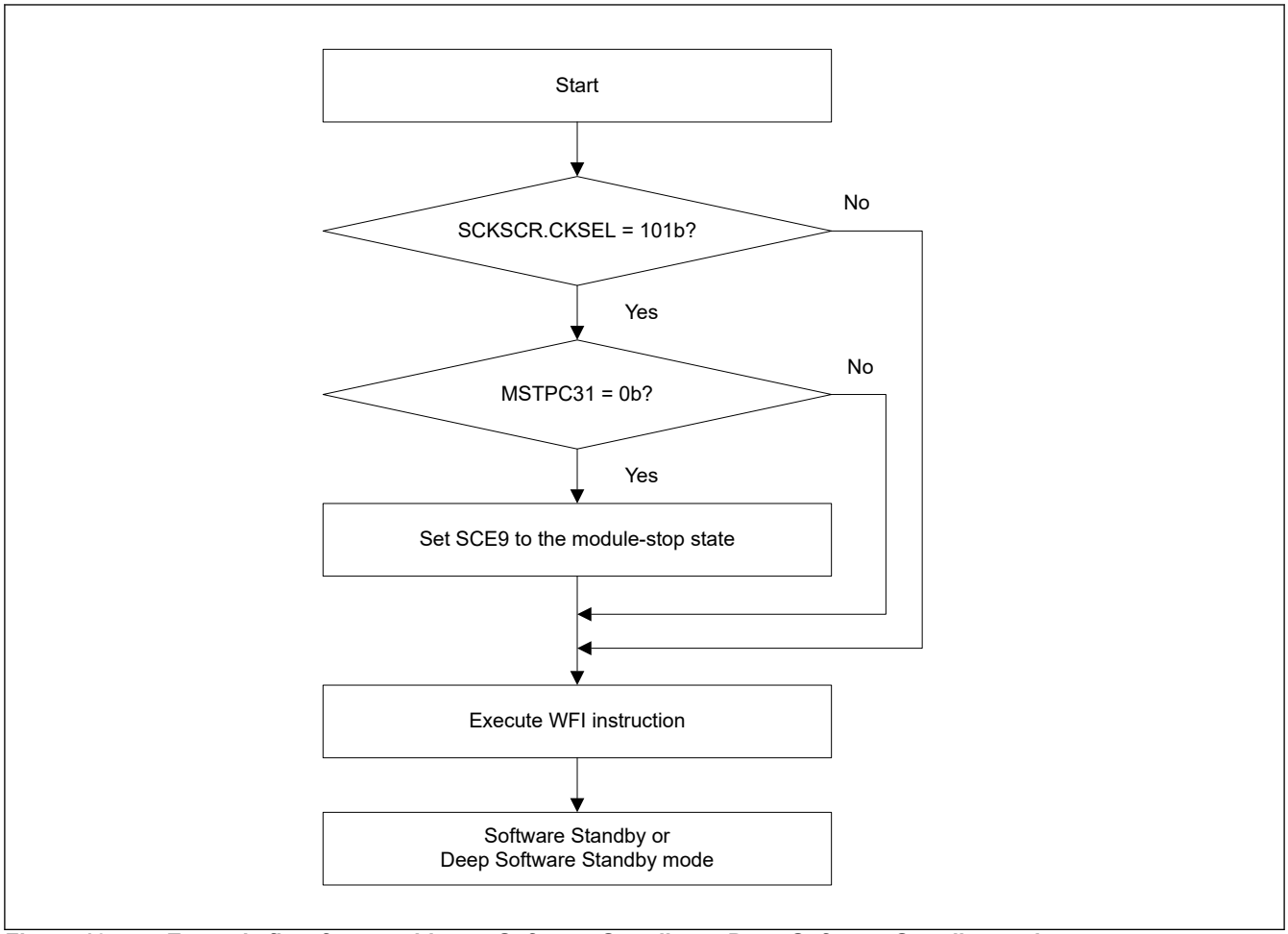



**Figure 10.4 Example flow for transition to Software Standby or Deep Software Standby mode**

## 10.7.2 Canceling Software Standby Mode

Software Standby mode is canceled by:

- An available interrupt shown in Table 10.3
- A RES pin reset
- A power-on reset
- A voltage monitor reset
- A reset caused by an IWDT underflow.

On exiting Software Standby mode, the oscillators that operate before the transition to the mode restart. After all the oscillators are stabilized, the MCU returns to Normal mode from Software Standby mode. See section 13.2.18. WUPEN0 : Wake Up Interrupt Enable Register 0, section 13.2.19. WUPEN1 : Wake Up interrupt enable register 1 for information on how to wake up the MCU from Software Standby mode.

You can cancel Software Standby mode in any of the following ways:

1. Canceling by an interrupt
   When an available interrupt request (see Table 10.3) is generated, an oscillator that operates before the transition to Software Standby mode restarts. After all the oscillators are stabilized, the MCU returns to Normal mode from Software Standby mode and starts the interrupt handling.
2. Canceling by a RES pin reset
   When the RES pin is driven low, the MCU enters the reset state, and the oscillators whose default status is operating, start the oscillation. Be sure to keep the RES pin low for the time period specified in section 47, Electrical Characteristics. When the RES pin is driven high after the specified time period, the CPU starts the reset exception handling.

3. Canceling by a power-on reset
   Software Standby mode is canceled by a power-on reset and the MCU starts the reset exception handling.
4. Canceling by a voltage monitor reset
   Software Standby mode is canceled by a voltage monitor reset from the voltage detection circuit and the MCU starts the reset exception handling.
5. Canceling by IWDT reset
   Software Standby mode is canceled by an internal reset generated by an IWDT underflow and the MCU starts the reset exception handling. However, IWDT stops in Software Standby mode and an internal reset for canceling Software Standby mode is not generated in the following condition:
   - OFS0.IWDTSTRT = 0 and OFS0.IWDTSTPCTL = 1.

### 10.7.3 Example of Software Standby Mode Application

Figure 10.5 shows an example of entry to Software Standby mode on detection of a falling edge of the IRQn pin, and exit from Software Standby mode by a rising edge of the IRQn pin.

In this example, an IRQn pin interrupt is accepted with the IRQCRi.IRQMD[1:0] bits of the ICU set to 00b (falling edge) in Normal mode, and the IRQCRi.IRQMD[1:0] bits are set to 01b (rising edge). After that, the SBYCR.SSBY bit is set to 1 and a WFI instruction is executed. As a result, entry to Software Standby mode completes and exit from Software Standby mode is initiated by a rising edge of the IRQn pin.

Setting the ICU is also required to exit Software Standby mode. For details, see section 13, Interrupt Controller Unit (ICU). The oscillation stabilization time in Figure 10.5 is specified in section 47, Electrical Characteristics.

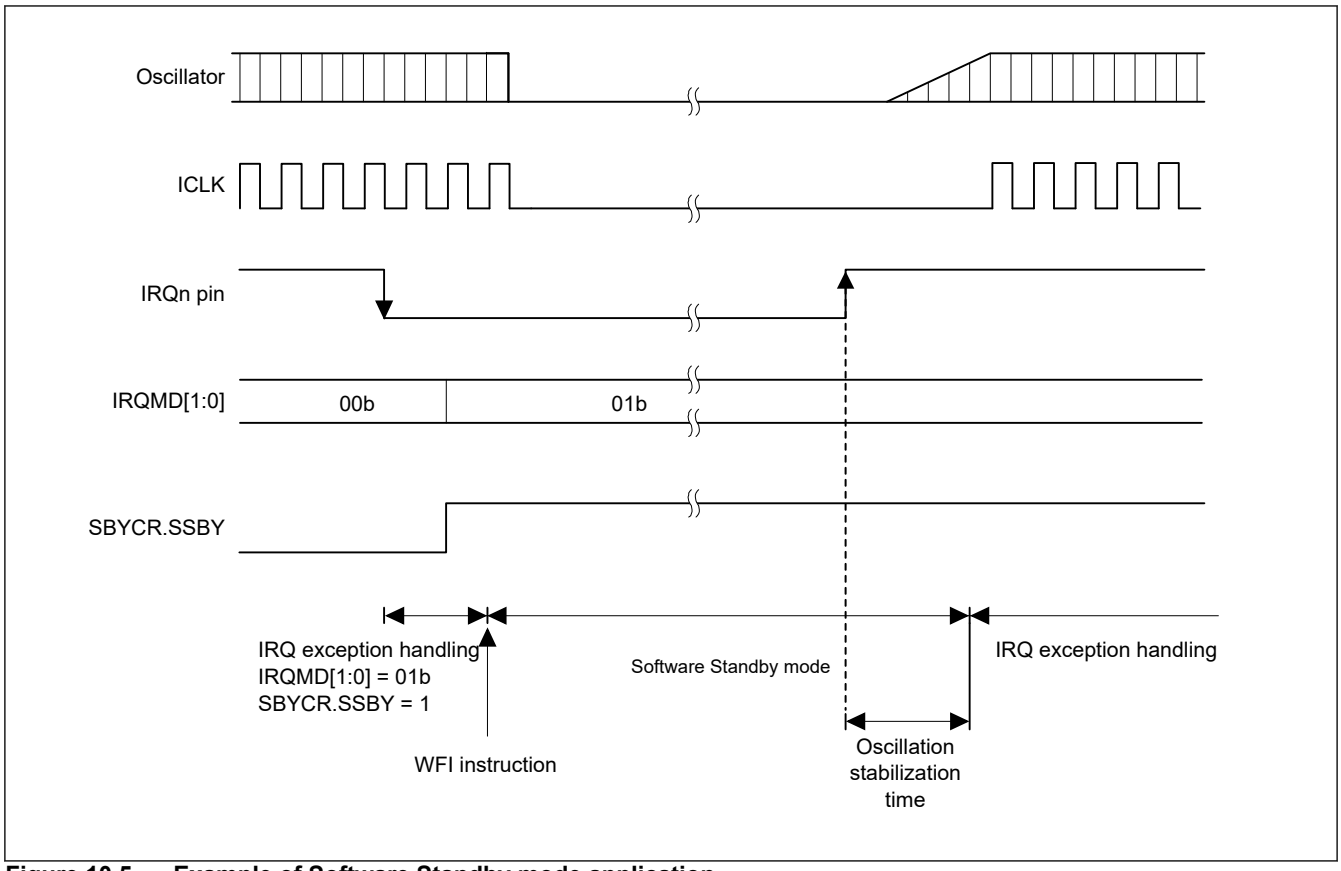



**Figure 10.5 Example of Software Standby mode application**

## 10.8 Snooze Mode

### 10.8.1 Transition to Snooze Mode

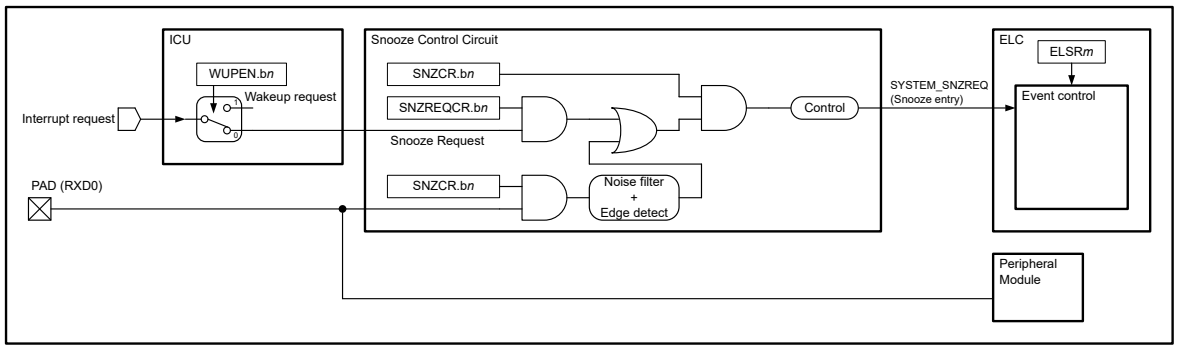



**Figure 10.6 Snooze mode entry configuration**

When the snooze control circuit accepts an available snooze request in Software Standby mode, the MCU transfers to Snooze mode. In this mode, some peripheral modules operates without waking the CPU. The peripheral modules that can operate in Snooze mode are shown in Table 10.2. Also, DTC operation in Snooze mode can be selected by the setting of SNZCR.SNZDTCEN bit.

Table 10.7 shows the Snooze requests that switch the MCU from Software Standby mode to Snooze mode. To use the listed Snooze requests as a trigger to switch to Snooze mode, the corresponding SNZREQENn bit of the SNZREQCRn register or RXDREQEN bit of SNZCR register must be set before entering Software Standby mode.

**Table 10.7 Available snooze requests to switch to Snooze mode**

| Snooze request | Control Register | |
|---|---|---|
| | Register | Bit*1 *3 |
| PORT_IRQn (n = 0 to 15) | SNZREQCR0 | SNZREQENn (n = 0 to 15) |
| RTC_ALM | SNZREQCR0 | SNZREQEN24 |
| RTC_PRD | SNZREQCR0 | SNZREQEN25 |
| AGT1_AGTI | SNZREQCR0 | SNZREQEN28 |
| AGT1_AGTCMAI | SNZREQCR0 | SNZREQEN29 |
| AGT1_AGTCMBI | SNZREQCR0 | SNZREQEN30 |
| AGT3_AGTI | SNZREQCR1 | SNZREQEN0 |
| AGT3_AGTCMAI | SNZREQCR1 | SNZREQEN1 |
| AGT3_AGTCMBI | SNZREQCR1 | SNZREQEN2 |
| RXD0 falling edge | SNZCR | RXDREQEN*2 |

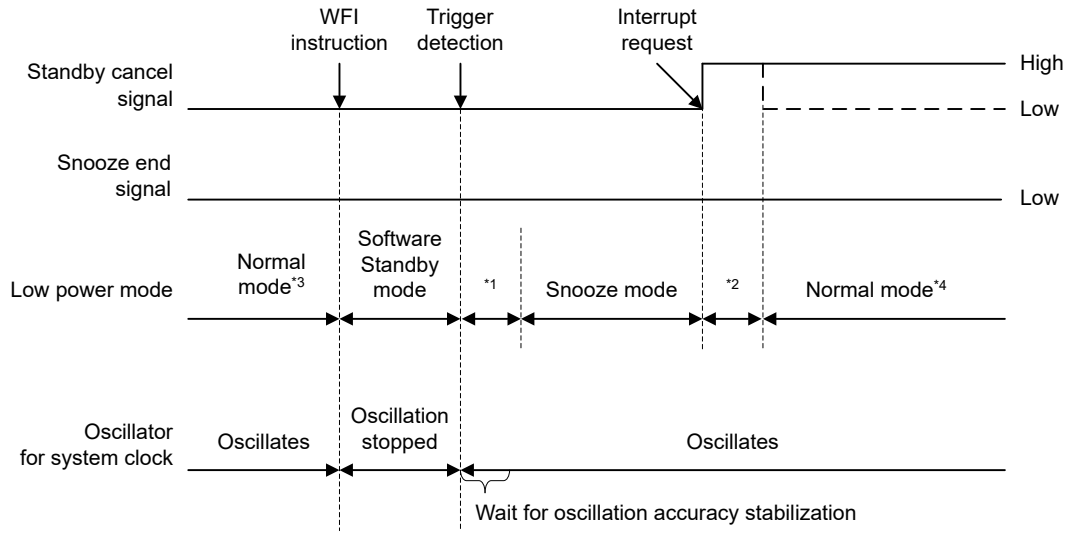
Note 1. Do not enable multiple snooze requests at the same time.
Note 2. Do not set the RXDREQEN bit to 1 except in asynchronous mode.
Note 3. When AGT1 is used for Snooze request factor, Snooze end factor is not allowed to set AGT3 (only AGT1).
When AGT3 is used for Snooze request factor, Snooze end factor is not allowed to set AGT1 (only AGT3).

Clear the DMAST.DMST and DTCST.DTCST bits to 0 before executing a WFI instruction, except when using the DTC in Snooze mode. If the DTC is required in Snooze mode, set the DTCST.DTCST bit to 1 before executing a WFI instruction.

### 10.8.2 Canceling Snooze Mode

Snooze mode is canceled by an interrupt request that is available in Software Standby mode or a reset. Table 10.3 shows the requests that can be used to exit each mode. After canceling the Snooze mode, the MCU enters Normal mode and proceeds with exception processing for the given interrupt or reset. The action triggered by the interrupt requests, selected in

SELSR0, cancels Snooze mode. Interrupt canceling Snooze mode must be selected in IELSRn to link to the NVIC for the corresponding interrupt handling. See section 13, Interrupt Controller Unit (ICU) for information on SELSR0 and IELSRn registers.




Note 1. Transition time from Software Standby mode to Snooze mode.
Note 2. Transition time from Snooze mode to Normal mode.
Note 3. Enable Snooze mode (SNZCR.SNZE = 1) immediately before switching to Software Standby mode.
Note 4. Disable Snooze mode (SNZCR.SNZE = 0) immediately after returning to Normal mode.

**Figure 10.7 Canceling of Snooze mode when an interrupt request signal is generated**

### 10.8.3 Returning from Snooze Mode to Software Standby Mode

Table 10.8 shows the snooze end request that can be used as triggers to return to Software Standby mode. The snooze end requests are available only in Snooze mode. If the requests are generated when the MCU is not in Snooze mode, they are ignored. When multiple requests are selected, each of the requests invokes transition to Software Standby mode from Snooze mode.

Table 10.9 shows the snooze end conditions that consist of the snooze end requests and the conditions of the peripheral modules. The CTSU, SCI0, ADC12, and DTC modules can keep the MCU in Snooze mode until they complete the operation. However, an AGTn (n = 1, 3) underflow as a trigger to return to Software Standby mode cancels Snooze mode without waiting for the completion of SCI0 operation.

Figure 10.8 shows the timing diagram for the transition from Snooze mode to Software Standby mode. This mode transition occurs according to which snooze end requests are set in the SNZEDCR0 register. A snooze request is cleared automatically after returning to Software Standby mode.

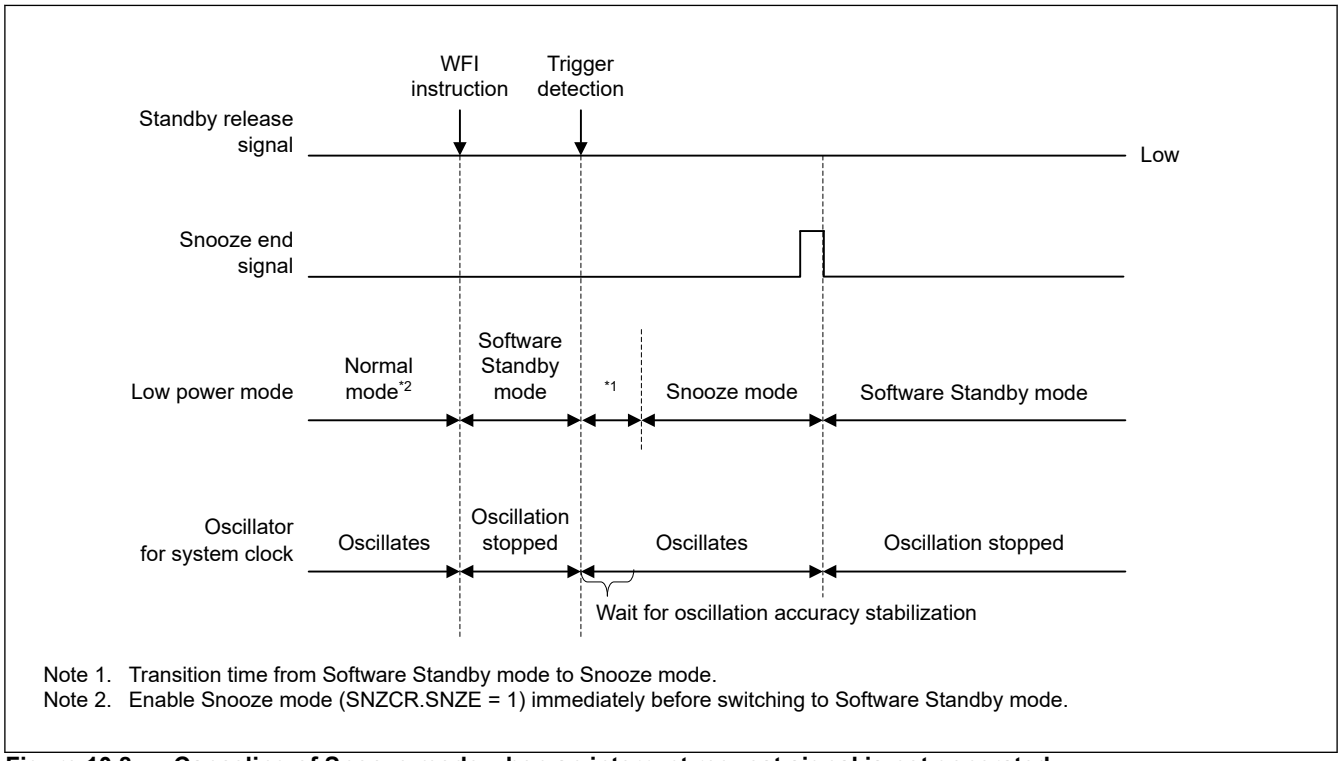
**Table 10.8 Available snooze end requests (triggers to return to Software Standby mode)**

| Peripheral Module | Snooze end request | Enable/Disable Control | |
|---|---|---|---|
| | | Register | Symbol |
| AGT1 | AGT1 underflow (AGT1_AGTI) | SNZEDCR0 | AGTUNFED |
| DTC | Last DTC transmission completion (DTC_COMPLETE) | SNZEDCR0 | DTCZRED |
| DTC | Not Last DTC Transmission Completion (DTC_TRANSFER) | SNZEDCR0 | DTCNZRED |
| ADC120 | Window A/B compare match (ADC120_WCMPM) | SNZEDCR0 | AD0MATED |
| ADC120 | Window A/B compare mismatch (ADC120_WCMPUM) | SNZEDCR0 | AD0UMTED |
| SCI0 | SCI0 address mismatch (SCI0_DCUF) | SNZEDCR0 | SCI0UMTED |
| AGT3 | Underflow or measurement complete (AGT3_AGTI) | SNZEDCR1 | AGT3UNFED |

**Table 10.9 Snooze end conditions**

| Operating module when a snooze end request occurs | Snooze end request | |
|---|---|---|
| | AGT1/AGT3 underflow | Other than AGT1/AGT3 underflow |
| DTC | The MCU transfers to the Software Standby mode after all of the modules listed in this table complete operation. | The MCU transfers to the Software Standby mode after all of the modules listed to the left of this column complete the operation. |
| ADC12n | | |
| CTSU | | |
| SCI0 | The MCU transfers to the Software Standby mode immediately after the snooze end request is generated. | |
| Other than specified | The MCU transfers to the Software Standby mode immediately after a snooze end request is generated. | |

Note: If the DTC is used to activate the ADC12n, CTSU, or SCI, the MCU transitions to Software Standby mode immediately after a snooze end request is generated.

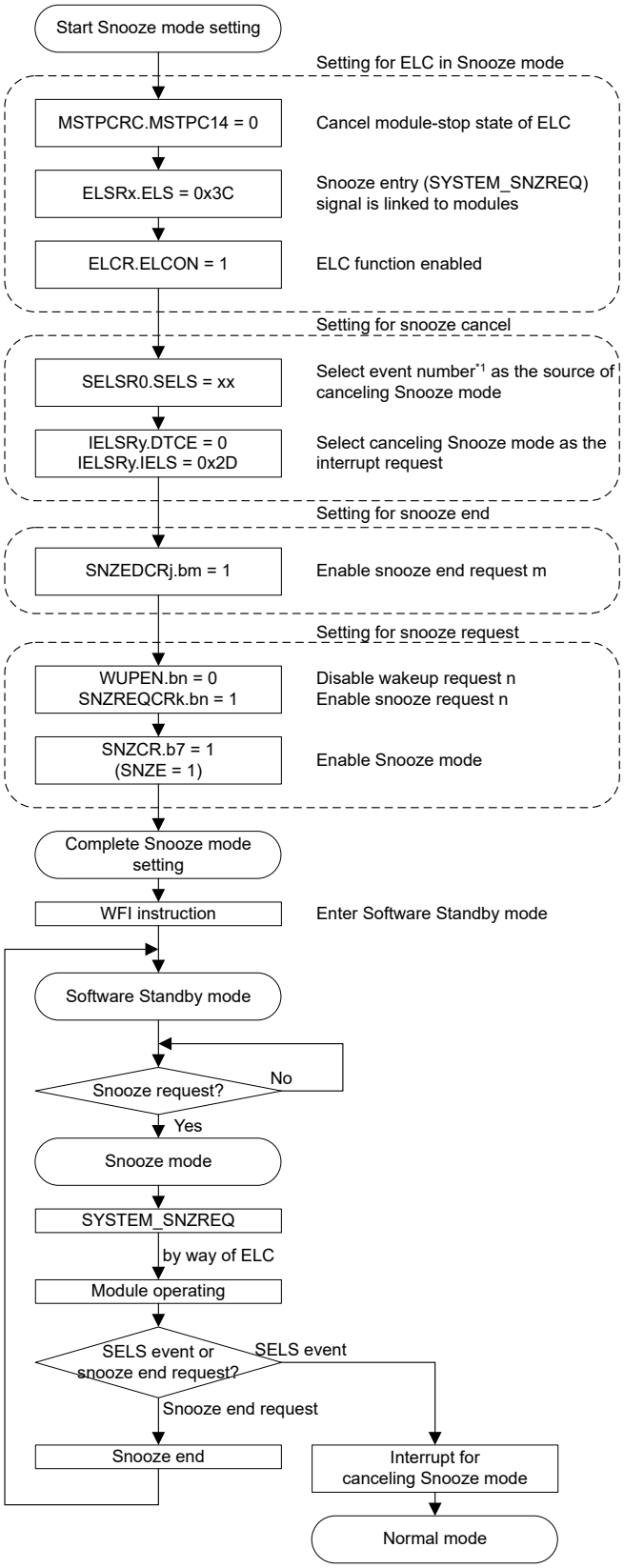



Note 1. Transition time from Software Standby mode to Snooze mode.
Note 2. Enable Snooze mode (SNZCR.SNZE = 1) immediately before switching to Software Standby mode.

**Figure 10.8 Canceling of Snooze mode when an interrupt request signal is not generated**

## 10.8.4 Snooze Operation Example

Figure 10.9 shows an example setting for using ELC in Snooze mode.




Note 1. See Table 13.4

**Figure 10.9 Setting example of using ELC in Snooze mode**

The MCU can transmit and receive data in SCI0 asynchronous mode without CPU intervention. When using the SCI0 in Snooze mode, use either High-speed mode or Low-speed mode.

Do not use Subosc-speed mode. Table 10.10 shows the maximum transfer rate of SCI0 in Snooze mode.

**Table 10.10 HOCO: ± 1.4% (Ta = -20°C to 105°C) (Unit: bps)**

| Maximum division ratio of ICLK, PCLKA, PCLKB, PCLKC, PCLKD, FCLK, and TRCLK | HOCO frequency | | | | | |
|---|---|---|---|---|---|---|
| | LOCO is not operating | | | LOCO is operating | | |
| | 16 MHz | 18 MHz | 20 MHz | 16 MHz | 18 MHz | 20 MHz |
| 1 | 2400 | | | 4800 | | |
| 2 | | | | | | |
| 4 | | | | | | |
| 8 | | | | | | |
| 16 | | | | | | |
| 32 | 1200 | | | 2400 | | |
| 64 | | | | | | |

When using SCI0 in Snooze mode, use the following setting: BGDM = 0, ABCS = 0, ABCSE = 0. See section 27, Serial Communications Interface (SCI) for information on these bits.

Figure 10.10 shows a setting example for using SCI0 in Snooze mode entry.

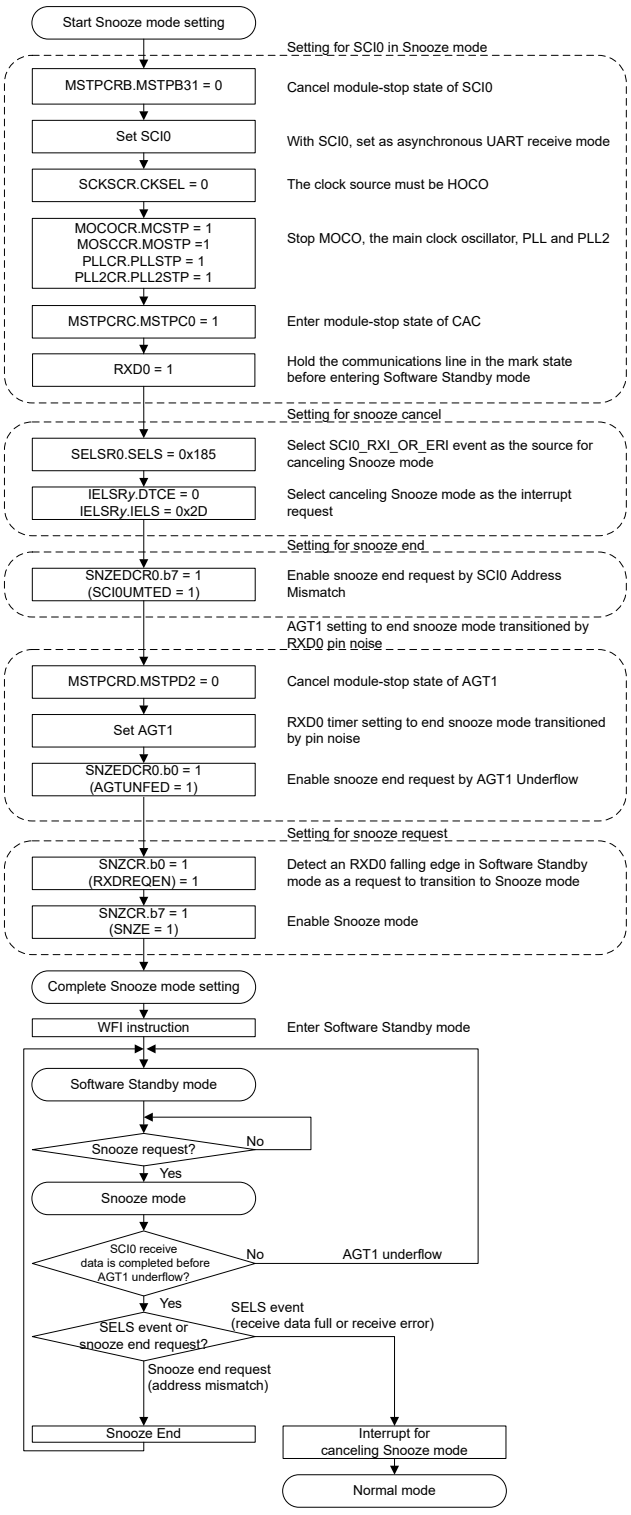



**Figure 10.10 Setting example of using SCI0 in Snooze mode entry**

## 10.9 Deep Software Standby Mode

### 10.9.1 Transitioning to Deep Software Standby Mode

When a WFI instruction is executed with the SBYCR.SSBY and DPSBYCR.DPSBY bits set to 1, the MCU enters Deep Software Standby mode. See Table 10.6 for the setting of the related control bits. In this mode, the CPU, on-chip peripheral functions (except for RTC alarm, RTC interval, and USB suspend/resume detecting unit), SRAM (except for standby RAM), and all oscillators (except for Sub-clock oscillator and Low-speed on-chip oscillator) are stopped. Also because the internal power supply to these modules is stopped, power consumption is remarkably reduced. The contents of all CPU registers and internal peripheral modules (except for RTC alarm, RTC interval, and USB suspend/resume detecting unit) become undefined.

Data in the standby SRAM are preserved if the setting of the DEEPCUT[1:0] bits are 00b. If the setting of the DEEPCUT[1:0] bits are 01b, the internal power supply to the standby SRAM and the USB resume detecting unit is cut off, reducing power consumption. Data in the standby SRAM becomes undefined at this time. If the setting of the DEEPCUT[1:0] bits are 11b, the internal power supply to the standby SRAM, and the USB resume detecting unit is cut off, the LVD is stopped, and the low-power-consumption function of the power-on reset circuit is enabled, so power consumption is further reduced. For details, see section 47, Electrical Characteristics.

When the MCU enters Deep Software Standby mode while the IWDT is in auto start mode and the OFS0.IWDTSTPCTL bit is 1, power supply to the IWDT-dedicated clock and the IWDT is cut off, and counting by the IWDT stops.

When OFS0.IWDTSTPCTL bit is 0, the MCU enters Software Standby mode instead of Deep Software Standby mode, regardless of the setting of OFS0.IWDTSTRT bit or DPSBYCR.DPSBY bit. If OFS0.IWDTSTPCTL bit is 0 while OFS0.IWDTSTRT bit is 0 (auto start mode), IWDT-dedicated clock and IWDT continues the operation.

When LVD1CR0.RI = 1 (voltage monitor 1 reset selected) or LVD2CR0.RI = 1 (voltage monitor 2 reset selected), the MCU enters Software Standby mode instead of Deep Software Standby mode. The I/O port states are the same as in Software Standby mode.

When the PLL is selected as the clock source, set the following modules to the module-stop state before executing a WFI instruction:

- SCE9

Note: Conditions on the DTC, DMAC, and IWDT for transitioning to Software Standby mode should be met before the WFI instruction is executed. For details, see section 10.7. Software Standby Mode.

### 10.9.2 Cancelling Deep Software Standby Mode

Deep Software Standby mode is canceled by:

- An interrupt shown in Table 10.3
- A RES pin reset
- A power-on reset
- A voltage monitor 0 reset.

#### (1) Cancelling by an interrupt

Cancelling by interrupts is controlled by DPSIERn (n = 0 to 3) and DPSIFRn (n = 0 to 3). When a Deep Software Standby Cancelling interrupt is generated, the corresponding flag in DPSIFRn is set to 1. If the interrupt is enabled in DPSIERn, Deep Software Standby mode is canceled. Rising edge or falling edge can be selected by DPSIEGRn (n = 0 to 2). The interrupts for which an edge can be selected are the NMI, IRQn-DS (n = 0, 1, 4 to 12, 14, 15), voltage monitor 1, and voltage monitor 2 interrupts. When a Deep Software Standby mode canceling request occurs, the internal power is supplied and MOCO starts oscillating, and an internal reset (Deep Software Standby reset) is generated for the entire MCU.

The stable MOCO clock is supplied to the entire MCU and Deep Software Standby reset is canceled. The MCU starts reset exception handling.

When Deep Software Standby mode is canceled by an external interrupt pin or internal interrupt signal, the RSTSR0.DPSRSTF flag is set to 1.

#### (2) Cancelling by RES pin reset

When the RES pin is driven low, the MCU cancels Deep Software Standby mode and enters the reset state. Keep the RES pin low for the time specified in section 47, Electrical Characteristics. When RES pin is driven high after the specified time period, the CPU starts the reset exception handling.

#### (3) Cancelling by a power-on reset

Deep Software Standby mode is canceled by a power-on reset and the MCU starts the reset exception handling.

#### (4) Cancelling by a voltage monitor 0 reset

Deep Software Standby mode is canceled by a voltage monitor 0 reset from the voltage detection circuit and the MCU starts the reset exception handling.

### 10.9.3 Pin States when Deep Software Standby mode is Canceled

In Deep Software Standby mode, the I/O ports retain the same states from Software Standby mode. The MCU is initialized by an internal reset generated when Deep Software Standby mode is canceled, and reset exception handling starts immediately. The DPSBYCR.IOKEEP bit setting determines whether to initialize the I/O ports or to retain the I/O ports states for Software Standby mode. The following is the state of the I/O ports for each bit setting:

- When the DPSBYCR.IOKEEP bit = 0
  I/O ports are initialized by an internal reset generated when Deep Software Standby mode is canceled.
- When the DPSBYCR.IOKEEP bit = 1
  Although the MCU is initialized by an internal reset generated when Deep Software Standby mode is canceled, the I/O ports retain their states from Software Standby mode regardless of the MCU internal state. The I/O ports states remain unchanged from Software Standby mode even when settings are made to the I/O ports or peripheral modules. The retained I/O ports states are released by clearing the DPSBYCR.IOKEEP bit to 0, and the MCU operates according to the internal state. The DPSBYCR.IOKEEP bit is not initialized by any internal reset generated when Deep Software Standby mode is canceled.

### 10.9.4 Example of Deep Software Standby Mode Application

#### (1) Entering and exiting Deep Software Standby mode

Figure 10.11 shows an example where a transition to Deep Software Standby mode is made at the falling edge of the IRQn-DS pin, and exiting Deep Software Standby mode is made at the rising edge of the IRQn-DS pin. In this example, an IRQn interrupt is accepted with the IRQCRi.IRQMD[1:0] bits of the ICU set to 00b (falling edge). After the DPSIEGRy.DIRQnEG (y = 0 or 1, n = 0 to 15) bit is set to 1 (rising edge) and the SBYCR.SSBY bit and DPSBYCR.DPSBY bit are both set to 1, the WFI instruction is executed. As a result, the MCU transitions to Deep Software Standby mode. Deep Software Standby mode is then canceled on the rising edge of the IRQn-DS pin.

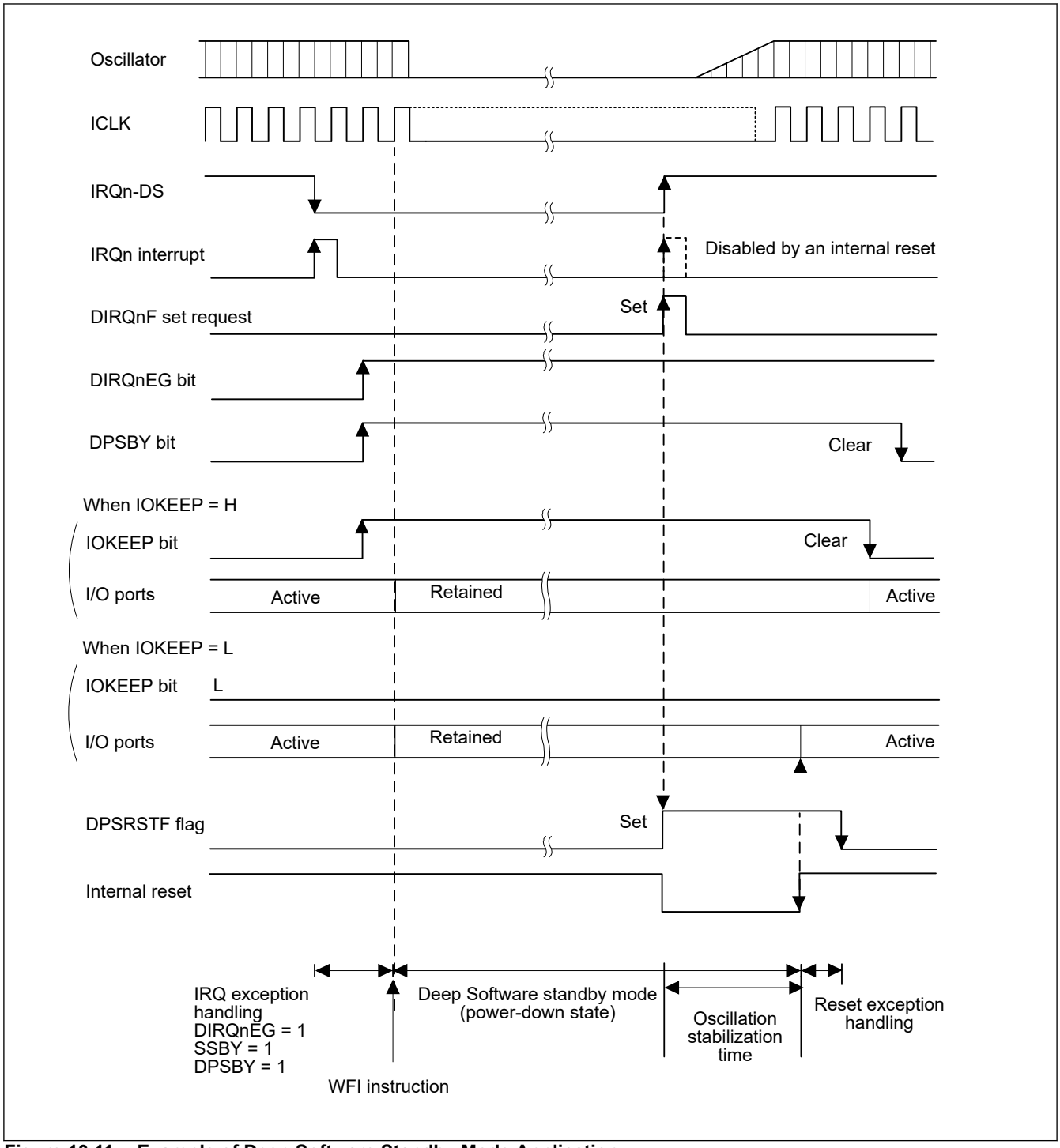



**Figure 10.11 Example of Deep Software Standby Mode Application**

## 10.9.5 Usage Flow for Deep Software Standby Mode

Figure 10.12 shows an example flow for using Deep Software Standby mode.

In this example, the RSTSR0.DPSRSTF flag of the reset function is read after the reset exception handling to determine whether a reset was generated by the RES pin or by the cancellation of Deep Software Standby mode.

For a reset by the RES pin, the MCU transitions to Deep Software Standby mode after the required register settings are made.

For a reset by cancellation of Deep Software Standby mode, the DPSBYCR.IOKEEP bit is cleared to 0 after the I/O port settings are made.

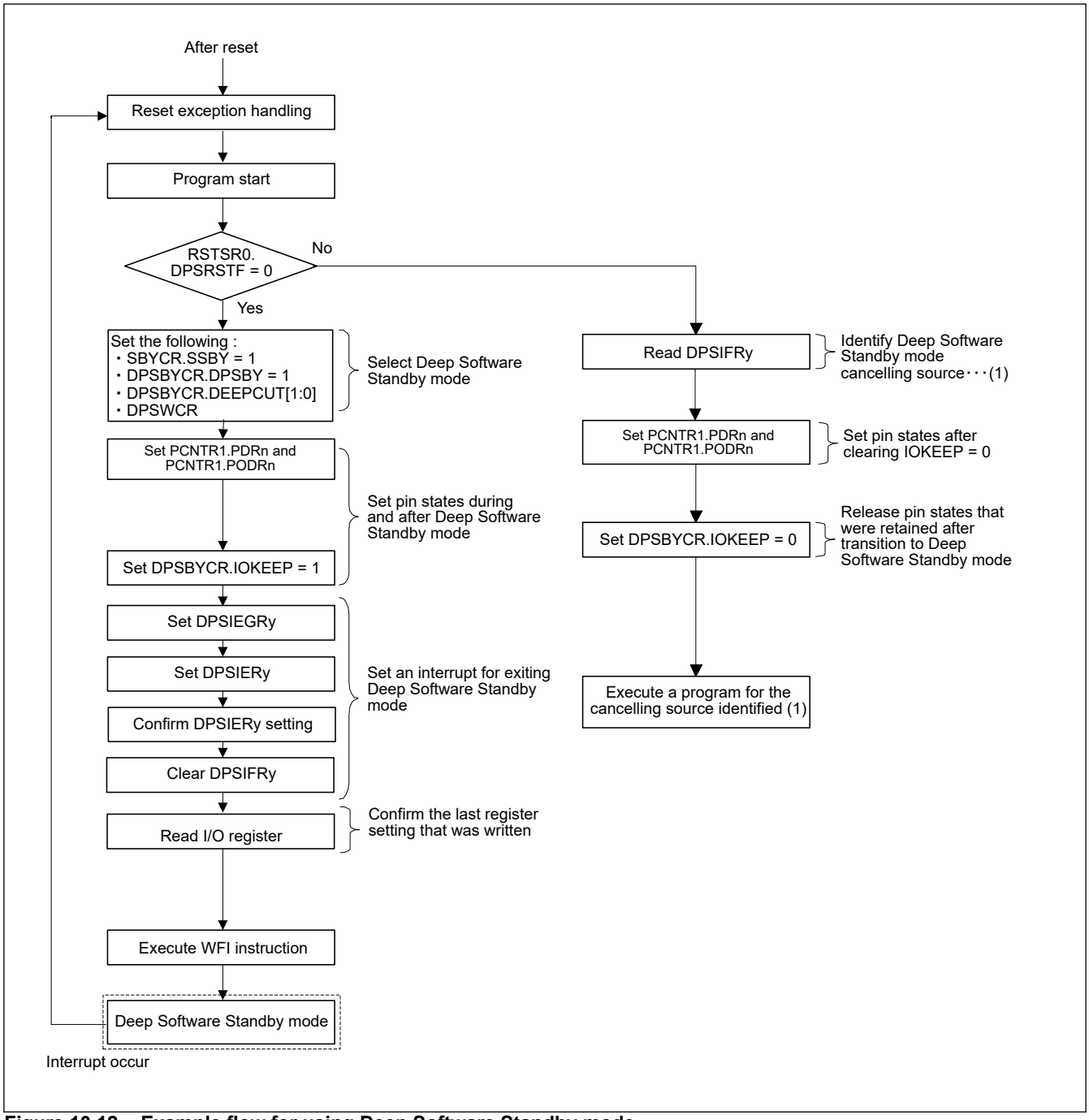



**Figure 10.12 Example flow for using Deep Software Standby mode**

## 10.10 Usage Notes

### 10.10.1 Register Access

#### (1) Invalid register write accesses during specific modes or transitions

Do not write to registers under any of the conditions listed in this section.

[Registers]

- All registers with a peripheral name of SYSTEM.

[Conditions]

- OPCCR.OPCMTSF = 1 or SOPCCR.SOPCMTSF = 1 (during transition of the operating power control mode)

- During the time period from executing a WFI instruction to returning to Normal mode
- FENTRYR.FENTRY0 = 1 or FENTRYR.FENTRYD = 1 (flash P/E mode, data flash P/E mode)

### (2) Valid setting for the clock-related registers

Table 10.11 and Table 10.12 show the valid settings of the clock-related registers in each operating power control mode. Do not write any value other than the valid setting. Each register has certain prohibited settings under conditions other than those related to the operating power control modes. See section 8, Clock Generation Circuit for another condition of each register.

**Table 10.11 Valid settings for the clock-related registers (1)**

| Mode | Valid settings | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | SCKSCR.<br>CKSEL[2:0]<br>CKOCR.<br>CKOSEL[2:0] | SCKDIVCR.<br>FCK[2:0]<br>ICK[2:0] | PLLCR.<br>PLLSTP | PLL2CR.<br>PLL2STP | HOCOCR.<br>HCSTP | MOCOCR.<br>MCSTP | LOCOCR.<br>LCSTP | MOSCCR.<br>MOSTP | SOSCCR.<br>SOSTP |
| High-speed | 000b (HOCO)<br>001b (MOCO)<br>010b (LOCO)<br>011b (Main clock)<br>100b (Sub clock)<br>101b (PLL) *1 | 000b (1/1)<br>001b (1/2)<br>010b (1/4)<br>011b (1/8)<br>100b (1/16)<br>101b (1/32)<br>110b (1/64) | 0 (operating)<br>1 (stop) | 0 (operating)<br>1 (stop) | 0 (operating)<br>1 (stopped) | 0 (operating)<br>1 (stopped) | 0 (operating)<br>1 (stopped) | 0 (operating)<br>1 (stopped) | 0 (operating)<br>1 (stopped) |
| Low-speed | 000b (HOCO)<br>001b (MOCO)<br>010b (LOCO)<br>011b (Main clock)<br>100b (Sub clock) | | 1 (stop) | 1 (stop) | | | | | |
| Subosc-speed | 010b (LOCO)<br>100b (SOSC) | 000b (1/1) | 1 (stop) | 1 (stop) | 1 (stopped) | 1 (stopped) | 0 (operating)<br>1 (stopped) | 1 (stopped) | 0 (operating)<br>1 (stopped) |

Note 1. SCKSCR.CKSEL[2:0] only

**Table 10.12 Valid settings for the clock-related registers (2)**

| Operating oscillator | Valid settings | |
|---|---|---|
| | SOPCCR.SOPCM | OPCCR.OPCM[1:0] |
| PLL, PLL2 | 0 | 00b |
| High-speed on-chip oscillator | 0 | 00b, 11b |
| Middle-speed on-chip oscillator | | |
| Main clock oscillator | | |
| Low-speed on-chip oscillator | 0, 1 | 00b, 11b |
| Sub-clock oscillator | | |
| IWDT-dedicated on-chip oscillator | | |

### (3) Invalid register write accesses in subosc-speed mode

Do not write to registers under the listed condition in this section.

[Registers]

- SCKSCR, OPCCR.

[Condition]

- SOPCCR.SOPCM = 1 (Subosc-speed mode).

### (4) Invalid register write accesses by the DTC or DMAC

Do not write to registers listed in this section by the DTC or DMAC.

[Registers]

- MSTPCRA, MSTPCRB, MSTPCRC, MSTPCRD, MSTPCRE

#### (5) Invalid register write accesses in Snooze mode

Do not write to registers listed in this section in Snooze mode. They must be set before entering Software Standby mode.

[Registers]

- SNZCR, SNZEDCRn, SNZREQCRn.

#### (6) Invalid write access to FLWT.FLWT[2:0]

Do not write any value other than 000b to the FLWT.FLWT[2:0] bits under the listed condition.

[Conditions]

- SOPCCR.SOPCM = 1 (Subosc-speed mode)

#### (7) Invalid write access when PRCR.PRC1 is 0

Do not write to registers listed in this section when the PRCR.PRC1 bit is 0.

[Registers]

- SBYCR, SNZCR, SNZEDCRn, SNZREQCRn, OPCCR, SOPCCR, DPSBYCR, DPSWCR, DPSIERn, DPSIFRn, DPSIGRn, SYOCDCR, LDOSCR, PL2LDOSCR

#### (8) Invalid write access when when PRCR.PRC4 bit is 0

Do not write to registers listed in this section when the PRCR.PRC4 bit is 0.

[Registers]

- LPMSAR, DPFSAR

### 10.10.2 I/O Port pin states

The I/O port pin states in Software Standby mode, Deep Software Standby and Snooze mode, unless modifying in Snooze mode, are the same before entering the modes. Therefore, power consumption is not reduced while the output signals are held high.

### 10.10.3 Module-Stop State of DTC, DMAC

Before writing 1 to MSTPCRA.MSTPA22, clear the DMAST.DMST bit of the DMAC and the DTCST.DTCST bit of the DTC to 0. For details, see section 16, DMA Controller (DMAC) and section 17, Data Transfer Controller (DTC).

### 10.10.4 Internal Interrupt Sources

Interrupts do not operate in the module-stop state. If setting the module-stop bit while an interrupt request is generated, a CPU interrupt source or a DTC or DMAC startup source cannot be cleared. Always disable the associated interrupts before setting the module-stop bits.

### 10.10.5 Input Buffer Control by DIRQnE Bit

Setting the DPSIERy.DIRQnE (y = 0 or 1, n = 0 to 15) bit to 1 enables the associated input buffer of the IRQn-DS (n = 0, 1, 4 to 12, 14, 15) pins. Although inputs to these pins are sent to the DPSIFRy.DIRQnF (y = 0 or 1, n = 0 to 15) bits, they are not sent to the interrupt controller (ICU), peripheral modules, and I/O ports.

### 10.10.6 Transitioning to Low Power Modes

Because the MCU does not support wakeup by events, do not enter the low power modes such as Sleep mode, Software Standby mode or Deep Software Standby Mode by executing a WFE instruction. Also, do not set the SLEEPDEEP bit of the System Control Register in the Cortex-M33 core because the MCU does not support low power modes by SLEEPDEEP.

### 10.10.7 Timing of WFI Instruction

It is possible for the WFI instruction to be executed before I/O register write is completed, in which case operation might not be as intended. This can happen if the WFI is placed immediately after a write to an I/O register. To avoid this problem, read back the register that was written to confirm that the write completed.

### 10.10.8 Writing to the WDT/IWDT Registers by DTC or DMAC in Sleep Mode or Snooze Mode

Do not write to the WDT or IWDT registers by the DTC or DMAC while WDT or IWDT is stopped after entering Sleep mode or Snooze mode.

### 10.10.9 Oscillators in Snooze Mode

Oscillators that stop on entering Software Standby mode automatically restart when a trigger for switching to Snooze mode is generated. The MCU does not enter Snooze mode until all the oscillators stabilize. If in Snooze mode, you must disable oscillators that are not required in Snooze mode before entering Software Standby mode. Otherwise, the transition from Software Standby mode to Snooze mode takes longer.

### 10.10.10 Snooze Mode Entry by RXD0 Falling Edge

When the SNZCR.RXDREQEN bit is 1, the falling edge of RXD0 pin is used to switch MCU from Software Standby mode to Snooze mode when using UART of SCI0 in Snooze mode. In this case an interrupt such as SCI0_ERI, SCI0_RXI or an address mismatch event is used as the source for canceling Snooze mode. However noise on the RXD0 pin might cause the MCU to transfer from Software Standby mode to Snooze mode unexpectedly. In this case if the MCU does not receive RXD0 data after the noise, an interrupt such as SCI0_ERI or SCI0_RXI, or an address mismatch event is not generated and the MCU stays in Snooze mode. This can be avoided by using AGTn (n = 1, 3) underflow interrupt to return to Software Standby mode or Normal mode unless otherwise UART receive data is completed before AGTn (n = 1, 3) underflow. However, do not use the AGTn (n = 1, 3) underflow as a source to return to Software Standby mode during an UART communication. This causes the UART to stop the operation in a half-finished state.

### 10.10.11 Using UART of SCI0 in Snooze Mode

When using UART in Snooze mode, ensure that the snooze request (RXD0 falling edge) does not conflict with the wakeup requests set by the WUPEN register, otherwise UART cannot be guaranteed.

When using UART in Snooze mode, the following conditions must be satisfied:

- The clock source must be HOCO
- MOCO, PLL, PLL2, and the main clock oscillator must be stopped before entering Software Standby mode
- The RXD0 pin must be kept high before entering Software Standby mode
- A transition to Software Standby mode must not occur during an SCI0 communication
- The MSTPCRC.MSTPC0 bit must be 1 before entering Software Standby mode.

### 10.10.12 Conditions of A/D Conversion Start in Snooze Mode

ADC120 can only be triggered by the ELC in Snooze mode. Do not use software trigger or ADTRGn (n = 0, 1) pin.

### 10.10.13 Conditions of CTSU in Snooze Mode

The CTSU can only be started by the ELC in Snooze mode.

### 10.10.14 ELC Events in Snooze Mode

This section lists available ELC events in Snooze mode. Do not use any other events. If starting peripheral modules for the first time after entering Snooze mode, the Event Link Setting Register (ELSRn) must set a Snooze mode entry event (SYSTEM_SNZREQ) as the trigger.

- Snooze mode entry (SYSTEM_SNZREQ)
- DTC transfer end (DTC_DTCEND)
- ADC120 window A/B compare match (ADC120_WCMPM)
- ADC120 window A/B compare mismatch (ADC120_WCMPUM)
- Data operation circuit interrupt (DOC_DOPCI).

### 10.10.15 Module-Stop Bit Write Timing

It is possible that access to I/O register may be executed before the corresponding module-stop bit write completed. In this case, access to I/O register may not proceed as intended. To avoid this issue, before accessing I/O register, read back the module-stop bit that was written to confirm that the write completed.

# 11. Battery Backup Function

## 11.1 Overview

The MCU provides a battery backup function that maintains partial battery powering in the event of a power loss. Switching between VCC and VBATT, the battery-powered area includes RTC, SOSC, and backup memory.

During normal operation, the battery-powered area is powered by the main power supply, the VCC pin. When a VCC voltage drop is detected, the power source switches to the dedicated battery backup power pin, the VBATT pin. When the voltage rises again, the power source switches back from VBATT to VCC.

### 11.1.1 Features of Battery Backup Function

The features include:

- Battery power supply switch
- Backup registers
- Time capture pin detection

### 11.1.2 Battery Power Supply Switch

When the voltage applied to the VCC pin drops, this feature switches the power supply from the VCC pin to the VBATT pin. When the voltage rises, it switches the power supply from the VBATT pin back to the VCC pin.

### 11.1.3 Backup Registers

The battery-powered area provides 128-byte backup registers. These registers retain data when power is supplied from the VBATT pin even when the VCC pin is in the power-off state.

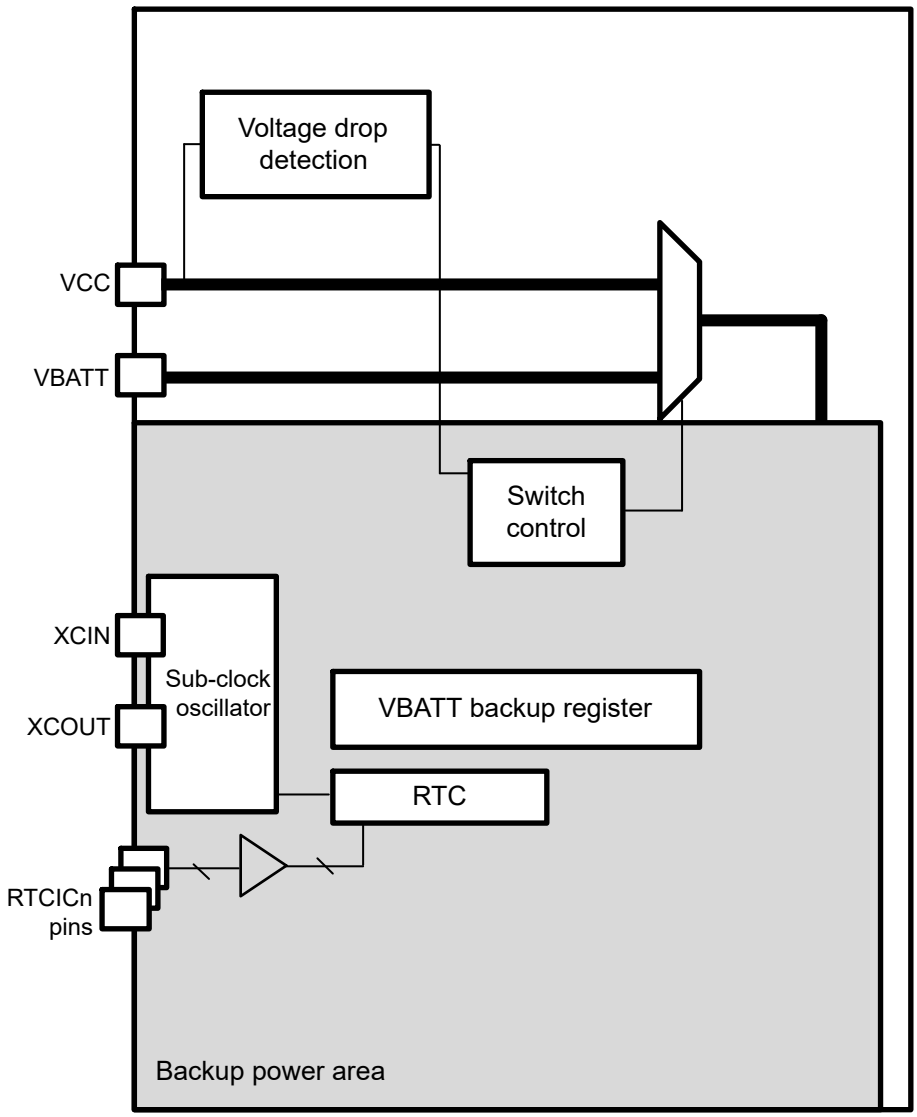
### 11.1.4 Time Capture Pin Detection

The RTC detects input level changes on the time capture pin. For more information, see section 23, Realtime Clock (RTC).

Note: When VCC is < $V_{DETBATT}$ and > (VBATT + 0.6 V), the injected current flows from the VCC to the VBATT pin through an internal diode. If the power supply battery connected to the VBATT pin cannot support this current injection, for example if the battery is not rechargeable, Renesas strongly recommends that you connect through a low-voltage threshold diode between the power supply battery and the VBATT pin.

Note: You must enable voltage monitor 0 reset to use the battery backup function. The voltage monitor 0 level must be higher than the VBATT switch level.

Figure 11.1 shows the configuration of the battery backup function.




Note: VCC : Main power supply pin
VBATT: Battery backup power supply pin
XCIN/XCOUT: SOSC input/output
RTCICn (n = 0 to 2): Input port for battery backup function

**Figure 11.1 Configuration of the battery backup function**

## 11.2 Register Descriptions

### 11.2.1 BBFSAR : Battery Backup Function Security Attribute Register

Base address: SYSC = 0x4001_E000

Offset address: 0x3D0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | NONSEC23 | NONSEC22 | NONSEC21 | NONSEC20 | NONSEC19 | NONSEC18 | NONSEC17 | NONSEC16 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | NONSEC2 | NONSEC1 | NONSEC0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | NONSEC0 | Non Secure Attribute bit 0<br>Target register: VBATTMNSELR<br>0: Secure<br>1: Non Secure | R/W |
| 1 | NONSEC1 | Non Secure Attribute bit 1<br>Target register: VBTBER<br>0: Secure<br>1: Non Secure | R/W |
| 2 | NONSEC2 | Non Secure Attribute bit 2<br>Target register: VBTICTLR<br>0: Secure<br>1: Non Secure | R/W |
| 15:3 | — | These bits are read as 1. The write value should be 1. | R/W |
| 16 | NONSEC16 | Non Secure Attribute bit 16<br>Target register: VBTBKRn (n = 0 to 15)<br>0: Secure<br>1: Non Secure | R/W |
| 17 | NONSEC17 | Non Secure Attribute bit 17<br>Target register: VBTBKRn (n = 16 to 31)<br>0: Secure<br>1: Non Secure | R/W |
| 18 | NONSEC18 | Non Secure Attribute bit 18<br>Target register: VBTBKRn (n = 32 to 47)<br>0: Secure<br>1: Non Secure | R/W |
| 19 | NONSEC19 | Non Secure Attribute bit 19<br>Target register: VBTBKRn (n = 48 to 63)<br>0: Secure<br>1: Non Secure | R/W |
| 20 | NONSEC20 | Non Secure Attribute bit 20<br>Target register: VBTBKRn (n = 64 to 79)<br>0: Secure<br>1: Non Secure | R/W |
| 21 | NONSEC21 | Non Secure Attribute bit 21<br>Target register: VBTBKRn (n = 80 to 95)<br>0: Secure<br>1: Non Secure | R/W |
| 22 | NONSEC22 | Non Secure Attribute bit 22<br>Target register: VBTBKRn (n = 96 to 111)<br>0: Secure<br>1: Non Secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 23 | NONSEC23 | Non Secure Attribute bit 23<br>Target register: VBTBKRn (n = 112 to 127)<br>0: Secure<br>1: Non Secure | R/W |
| 31:24 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

The BBFSAR register controls the secure attribute of the battery backup function registers.

**NONSEC0 bit (Non Secure Attribute bit 0)**

This bit controls the security attribute of VBATTMNSELR.

**NONSEC1 bit (Non Secure Attribute bit 1)**

This bit controls the security attribute of VBTBER.

**NONSEC2 bit (Non Secure Attribute bit 2)**

This bit controls the security attribute of VBTICTLR.

**NONSEC16 bit (Non Secure Attribute bit 16)**

This bit controls the security attribute of VBTBKRn (n = 0 to 15).

**NONSEC17 bit (Non Secure Attribute bit 17)**

This bit controls the security attribute of VBTBKRn (n = 16 to 31).

**NONSEC18 bit (Non Secure Attribute bit 18)**

This bit controls the security attribute of VBTBKRn (n = 32 to 47).

**NONSEC19 bit (Non Secure Attribute bit 19)**

This bit controls the security attribute of VBTBKRn (n = 48 to 63).

**NONSEC20 bit (Non Secure Attribute bit 20)**

This bit controls the security attribute of VBTBKRn (n = 64 to 79).

**NONSEC21 bit (Non Secure Attribute bit 21)**

This bit controls the security attribute of VBTBKRn (n = 80 to 95).

**NONSEC22 bit (Non Secure Attribute bit 22)**

This bit controls the security attribute of VBTBKRn (n = 96 to 111).

**NONSEC23 bit (Non Secure Attribute bit 23)**

This bit controls the security attribute of VBTBKRn (n = 112 to 127).

## 11.2.2 VBATTMNSELR : Battery Backup Voltage Monitor Function Select Register

Base address: SYSC = 0x4001_E000

Offset address: 0x41D

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | VBATTMNSEL |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | VBATTMNSEL | VBATT Low Voltage Detect Function Select Bit<br>0: Disables VBATT low voltage detect function<br>1: Enables VBATT low voltage detect function | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC3 bit to 1 (write enabled) before rewriting this register.

VBATTMNSELR is the register which controls VBATT low voltage detect function.

This register is initialized by all reset sources including Deep Software Standby reset.

### VBATTMNSEL bit (VBATT Low Voltage Detect Function Select Bit)

Select VBATT low voltage detect function

Consumption current increases while VBATTMNSEL = 1. So, after monitoring the VBATT voltage level, clear VBATTMNSEL to 0 in order to reduce power consumption of VBATT power supply.

## 11.2.3 VBATTMONR : Battery Backup Voltage Monitor Register

Base address: SYSC = 0x4001_E000

Offset address: 0x41E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | VBATTMON |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | VBATTMON | VBATT Voltage Monitor Bit<br>Check VBATT voltage level<br>0 can be read when VBATT Low Voltage Detect Function Select Bit is 0.<br>0: VBATT ≥ Vbattldet*1<br>1: VBATT < Vbattldet | R |
| 7:1 | — | These bits are read as 0. | R |

Note 1. Vbattldet is VBATT low voltage detection level . For more details, see section 47, Electrical Characteristics.

VBATTMONR is the register that can check VBATT voltage level when VBATTMNSELR.VBATTMNSEL = 1 and also VCC is supplied.

This register is initialized by all reset sources including Deep Software Standby reset.

### VBATTMON bit (VBATT Voltage Monitor Bit)

Monitor VBATT voltage level

## 11.2.4 VBTBER : VBATT Backup Enable Register

Base address: SYSC = 0x4001_E000

Offset address: 0x4C0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | VBAE | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 3 | VBAE | VBATT backup register access enable bit<br>0: Disable to access VBTBKR<br>1: Enable to access VBTBKR | R/W |
| 7:4 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

### VBAE bit (VBATT backup register access enable bit)

You must write 1 to VBAE before accessing VBTBKR and you must write 0 to VBAE after finishing all access (write or read) to VBTBKR. If you do not write 0 to VBAE, the data of VBTBKR is not kept in VBATT mode.

To access VBTBKR, wait for at least 500 ns after writing 1 to VBAE, and then access VBTBKR.

Before entering the Deep Software Standby mode, you must write 0 to VBAE.

To enter the Deep Software Standby mode, wait for at least 250 ns after writing 0 to VBAE, then enter the Deep Software Standby mode.

If you do not use VBTBKR, you should change VBAE to 0 to reduce power consumption of VBTBKR.

## 11.2.5 VBTBKR[n] : VBATT Backup Register (n = 0 to 127)

Base address: SYSC = 0x4001_E000

Offset address: 0x500 + 0x001 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | VBTBKR | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | VBTBKR | VBATT Backup Register<br>The value of this register is retained even in VBATT mode.<br>This register is not initialized by any reset sources. | R/W |

Note: If the security attribution is configured as Secure,
- Secure access is allowed,
- Non-secure access is ignored, but TrustZone access error is not generated.

If the security attribution is configured as Non-secure,
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

**Table 11.1 VBATT Backup Register**

| Address | Symbol |
|---|---|
| 0x4001_E500 to 0x4001_E50F | VBTBKR[0] to VBTBKR[15] |
| 0x4001_E510 to 0x4001_E51F | VBTBKR[16] to VBTBKR[31] |
| 0x4001_E520 to 0x4001_E52F | VBTBKR[32] to VBTBKR[47] |
| 0x4001_E530 to 0x4001_E53F | VBTBKR[48] to VBTBKR[63] |
| 0x4001_E540 to 0x4001_E54F | VBTBKR[64] to VBTBKR[79] |
| 0x4001_E550 to 0x4001_E55F | VBTBKR[80] to VBTBKR[95] |
| 0x4001_E560 to 0x4001_E56F | VBTBKR[96] to VBTBKR[111] |
| 0x4001_E570 to 0x4001_E57F | VBTBKR[112] to VBTBKR[127] |

### 11.2.6 VBTICTLR : VBATT Input Control Register

Base address: SYSC = 0x4001_E000

Offset address: 0x4BB

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | VCH2INEN | VCH1INEN | VCH0INEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | VCH0INEN | VBATT CH0 Input Enable<br>0: RTCIC0 input disable<br>1: RTCIC0 input enable | R/W |
| 1 | VCH1INEN | VBATT CH1 Input Enable<br>0: RTCIC1 input disable<br>1: RTCIC1 input enable | R/W |
| 2 | VCH2INEN | VBATT CH2 Input Enable<br>0: RTCIC2 input disable<br>1: RTCIC2 input enable | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: Set the PRCR.PRC1 bit to 1 (write enabled) before rewriting this register.

VBTICTLR is the register that can select VBATT I/O direction as input. This register is not initialized by any reset.

#### VCHnINEN bits (VBATT CHn Input Enable Bits) (n = 0 to 2)

The VCHnINEN bit enables the input direction on the associated VBATT channel.

See section 19.5.5. I/O Buffer Specification.

## 11.3 Operation

### 11.3.1 Battery Backup Function

When the voltage on the VCC pin drops, power can be supplied to the RTC and sub-clock oscillator from the VBATT pin. When a drop of power supply from VCC pin is detected, the power connection switches from VCC pin to the VBATT pin. The power supply from the VCC pin is resumed when the voltage on the VCC pin exceeds $V_{DETBATT}$. This power supply change does not affect the RTC operation.

You must enable voltage monitor 0 reset to use the battery backup function. The RTC supports time capture detection, triggered by a change of the time capture pin input level.

The VBATT pin supplies power to the following modules:

- RTC
- Sub-clock oscillator (including XCIN and XCOUT pins)
- VBATT Backup Register

Table 11.2 shows the operating states in VBATT mode.

**Table 11.2 Operating States in VBATT Mode (1 of 2)**

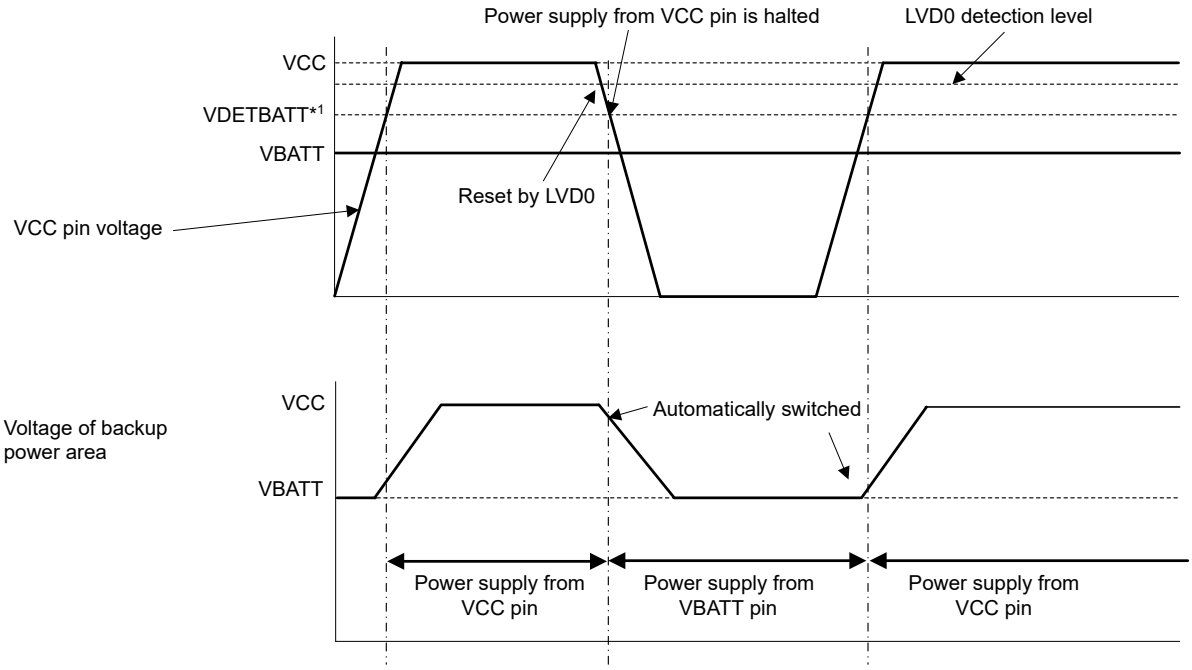
| Operating state | VBATT Mode |
|---|---|
| Transition condition | Detection of VCC voltage drop |
| Canceling method other than reset | Detection of VCC voltage rise |

**Table 11.2 Operating States in VBATT Mode (2 of 2)**

| Operating state | VBATT Mode |
|---|---|
| State after cancellation by an interrupt | — |
| State after cancellation by a reset | — |
| Main clock oscillator | Stop |
| Sub-clock oscillator | Operating |
| High-speed on-chip oscillator | Stop |
| Middle-speed on-chip oscillator | Stop |
| Low-speed on-chip oscillator | Stop |
| IWDT-dedicated on-chip oscillator | Stop |
| PLL | Stop |
| PLL2 | Stop |
| CPU | Stop (Undefined) |
| SRAM (ECC RAM included) | Stop (Undefined) |
| Standby SRAM | Stop (Undefined) |
| VBATT Backup Register | Stop (Retained) |
| Flash memory | Stop (Retained) |
| Realtime clock (RTC) | Selectable when selecting clock which is operating as the count source. |
| AGTn (n = 0 to 3) | Stop (Undefined) |
| Low voltage detection circuit (LVD) | Stop |
| Power-on reset circuit | Stop |
| Other Peripheral modules | Stop (Undefined) |
| I/O ports | RTCICn ports (n = 0 to 2): Operating<br>All ports not specified here: Undefined |

Note: Selectable means that operating or stopped is selectable in the control registers. Some modules are also controlled by the associated module-stop bit.
Note: Stop (retained) means that the contents of the internal registers are retained but the operations are suspended.
Note: Stop (undefined) means that the contents of the internal registers are undefined and power to the internal circuit is cut off.

Figure 11.2 shows switching sequence of Battery backup function.




Note: For details, see section 47, Electrical Characteristics.
Note 1. $V_{DETBATT}$ indicates the threshold level of the power supply change between the VCC pin and the VBATT pin.

**Figure 11.2 Switching sequence of Battery backup function.**

### 11.3.2 VBATT Battery Power Supply Switch Usage

The battery power supply switch can switch the power supply from the VCC pin to the VBATT pin when the voltage being applied to the VCC pin drops. When the voltage rises, this switch changes the power supply from the VBATT pin to the VCC pin.

Note: The battery backup function should be used after the voltage monitor 0 reset is enabled (OFS1.LVDAS bit is 0). Voltage monitor 0 level should be higher than VBATT switch level.

### 11.3.3 VBATT Backup Register Usage

Use the VBATT backup registers VBTBKRn, where n = 0 to 127, to store or restore data with an 8-bit read or write operation.

## 11.4 Usage Notes

1. Operation of the sub-clock oscillator and RTC are not guaranteed when the voltage level on VBATT is lower than the guaranteed operation range. Initialize the RTC when the VBATT pin falls below the guaranteed operating voltage and then powers up again.
2. A reset generated while writing to registers described in this section might destroy the register value.
3. When VCC is higher than $V_{DETBATT}$, the VCC pin and VBATT pin are separated. When VCC is lower than $V_{DETBATT}$ and the switch is connected to the VBATT pin, and if the voltage on VBATT drops lower than (VCC - 0.6 V), current might flow into the VBATT pin through the parasitic diode between the VCC and VBATT pins.
4. During RTC operation using the voltage from the VBATT pin and the I/O ports within the backup, the power supply area can only be used as time capture event input pins for the RTC.

# 12. Register Write Protection

## 12.1 Overview

The register write protection function protects important registers from being overwritten due to software errors. The registers to be protected are set with the Protect Register (PRCR).

Table 12.1 lists the association between the bits in the PRCR register and the registers to be protected.

**Table 12.1 Association between the bits in the PRCR register and registers to be protected**

| PRCR bit | Register to be protected |
|---|---|
| PRC0 | • Registers related to the clock generation circuit:<br>SCKDIVCR, SCKSCR, PLLCCR, PLLCR, MOSCCR, HOCOCR, MOCOCR, FLLCR1, FLLCR2, CKOCR, TRCKCR, OSTDCR, OSTDSR, PLL2CCR, PLL2CR, MOCOUTCR, HOCOUTCR, USBCKDIVCR, USBCKCR, MOSCWTCR, MOMCR, SOSCCR, SOMCR, LOCOCR, LOCOUTCR |
| PRC1 | • Registers related to the low power modes:<br>SBYCR, SNZCR, SNZEDCR0, SNZEDCR1, SNZREQCR0, SNZREQCR1, OPCCR, SOPCCR, DPSBYCR, DPSWCR, DPSIER0-3, DPSIFR0-3, DPSIEGR0-2, SYOCDCR, LDOSCR, PL2LDOSCR<br>• Register related to the battery backup function:<br>VBTBER, VBTICTLR, VBTBKRn (n = 0 to 127) |
| PRC3 | • Registers related to the LVD:<br>LVD1CR1, LVD1SR, LVD2CR1, LVD2SR, LVD1CMPCR, LVD2CMPCR, LVD1CR0, LVD2CR0, VBATTMNSELR |
| PRC4 | • Registers related to the security function:<br>CGFSAR, RSTSAR, LPMSAR, LVDSAR, BBFSAR, DPFSAR, CSAR, SRAMSAR, STBRAMSAR, DTCSAR, DMACSAR, ICUSARx, BUSSARx, MMPUSARx, TZFSAR, CPUDSAR, FSAR, PSARx, MSSAR, PmSAR, ELCSARx |

## 12.2 Register Descriptions

### 12.2.1 PRCR : Protect Register

Base address: SYSC = 0x4001_E000

Offset address: 0x3FE

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PRKEY[7:0] | | | | | | | | — | — | — | PRC4 | PRC3 | — | PRC1 | PRC0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PRC0 | Enable writing to the registers related to the clock generation circuit<br>0: Disable writes<br>1: Enable writes | R/W |
| 1 | PRC1 | Enable writing to the registers related to the low power modes, and the battery backup function<br>0: Disable writes<br>1: Enable writes | R/W |
| 2 | — | This bit is read as 0. The write value should be 0. | R/W |
| 3 | PRC3 | Enable writing to the registers related to the LVD<br>0: Disable writes<br>1: Enable writes | R/W |
| 4 | PRC4 | Enables writing to the registers related to the security function<br>0: Disable writes<br>1: Enable writes | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | PRKEY[7:0] | PRC Key Code<br>These bits control the write access to the PRCR register. To modify the PRCR register, write 0xA5 to the upper 8 bits and the target value to the lower 8 bits as a 16-bit unit. | W |

### PRCn bits (Protect bit n) (n = 0, 1, 3, 4)

The PRCn bits enable or disable writing to the protected registers listed in Table 12.1. Setting the PRCn bits to 1 or 0 enables or disables writing, respectively.

The register controlled by PRC4 may not reflect the PRC4 change when PRCR and its controlled registers are continuously written access. Avoid continuous write access or read the PRCR after PRC4 change, and then write access the PRC4-controlled register.

# 13. Interrupt Controller Unit (ICU)

## 13.1 Overview

The Interrupt Controller Unit (ICU) controls which event signals are linked to the Nested Vector Interrupt Controller (NVIC), the DMA Controller (DMAC), and the Data Transfer Controller (DTC) modules. The ICU also controls non-maskable interrupts.

Table 13.1 lists the ICU specifications, Figure 13.1 shows a block diagram, and Table 13.2 lists the I/O pins.

**Table 13.1 ICU specifications**

| Item | | Description |
|---|---|---|
| Maskable interrupts | Peripheral function interrupts | • Interrupts from peripheral modules<br>Number of sources: 212 (select factor within event list numbers 32 to 511) |
| | External pin interrupts | • Interrupt detection on low level*4, falling edge, rising edge, rising and falling edges. One of these detection methods can be set for each source<br>• Digital filter function supported<br>• 16 sources, with interrupts from IRQi (i = 0 to 15) pins. |
| | Interrupt requests to CPU (NVIC) | • 96 interrupt requests are output to NVIC. |
| | DMAC control | • The DMAC can be activated using interrupt sources*1<br>• The target interrupt source can be selected individually for every DMAC channels. |
| | DTC control | • The DTC can be activated using interrupt sources*1<br>• The method for selecting an interrupt source is the same as that of the interrupt request to NVIC. |
| Non-maskable interrupts*2 | NMI pin interrupt | • Interrupt from the NMI pin<br>• Interrupt detection on falling edge or rising edge<br>• Digital filter function supported |
| | WDT underflow/refresh error*3 | Interrupt on an underflow of the down-counter or occurrence of a refresh error |
| | IWDT underflow/refresh error*3 | Interrupt on an underflow of the down-counter or occurrence of a refresh error |
| | Low voltage detection 1*3 | Voltage monitor 1 interrupt of the voltage monitor 1 circuit (LVD_LVD1) |
| | Low voltage detection 2*3 | Voltage monitor 2 interrupt of the voltage monitor 2 circuit (LVD_LVD2) |
| | RPEST*5 | Interrupt on SRAM parity error |
| | RECCST*5 | Interrupt on SRAM ECC error |
| | TZFST*5 | Interrupt on TrustZone Filter error |
| | Oscillation stop detection interrupt*3 | Interrupt on detecting that the main oscillation has stopped |
| | Bus master MPU error*5 | Interrupt on bus master MPU error |
| Low power modes | | • Sleep mode: return is initiated by non-maskable interrupts or any other interrupt source<br>• Software Standby mode: return is initiated by non-maskable interrupts. Interrupt can be selected in the WUPEN register.<br>• Snooze mode: return is initiated by non-maskable interrupts. Interrupt can be selected in the SELSR0 and WUPEN registers.<br>See section 13.2.17. SELSR0 : SYS Event Link Setting Register and section 13.2.18. WUPEN0 : Wake Up Interrupt Enable Register 0, section 13.2.19. WUPEN1 : Wake Up interrupt enable register 1. |
| TrustZone Filter | | Available |

Note 1. For the DMAC and DTC activation sources, see Table 13.4.
Note 2. Non-maskable interrupts can be enabled only once after a reset release.
Note 3. These non-maskable interrupts can also be used as maskable interrupts. When used as maskable interrupts, do not change the value of the NMIER register from the reset state. To enable voltage monitor 1 and voltage monitor 2 interrupts, set the LVD1CR1.IRQSEL and LVD2CR1.IRQSEL bits to 1.
Note 4. Low level: interrupt detection is not canceled if you do not clear it after a detection.
Note 5. These non-maskable interrupt sources cannot be recovered if the request source clock is stopped during low power mode.

Figure 13.1 shows the ICU block diagram.

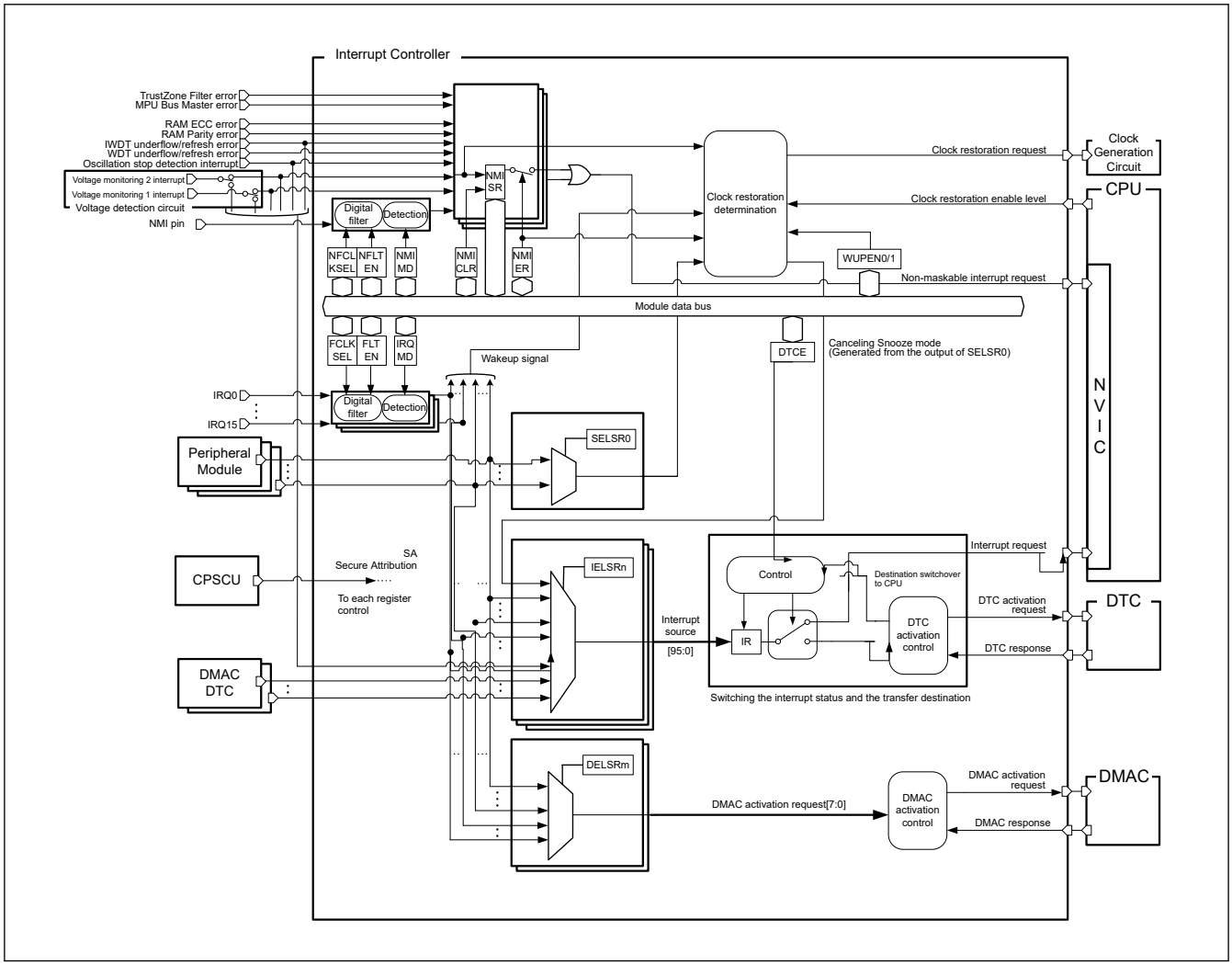



**Figure 13.1 ICU block diagram**

Table 13.2 lists the ICU input/output pins.

**Table 13.2 ICU I/O pins**

| Pin name | I/O | Description |
|---|---|---|
| NMI | Input | Non-maskable interrupt request pin |
| IRQi (i = 0 to 15) | Input | External interrupt request pins |

## 13.2 Register Descriptions

This chapter does not describe the Arm® NVIC internal registers. For information about these registers, see ARM Limited., ARM® Cortex®-M33 Processor Technical Reference Manual (ARM 100230).

## 13.2.1 ICUSARA : Interrupt Controller Unit Security Attribution Register A

Base address: CPSCU = 0x4000_8000

Offset address: 0x40

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SAIRQ CR15 | SAIRQ CR14 | SAIRQ CR13 | SAIRQ CR12 | SAIRQ CR11 | SAIRQ CR10 | SAIRQ CR9 | SAIRQ CR8 | SAIRQ CR7 | SAIRQ CR6 | SAIRQ CR5 | SAIRQ CR4 | SAIRQ CR3 | SAIRQ CR2 | SAIRQ CR1 | SAIRQ CR0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | SAIRQCR15 to SAIRQCR0 | Security attributes of registers for the IRQCRn register<br>0: Secure<br>1: Non-secure | R/W |
| 31:16 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

### SAIRQCRn bits (Security attributes of registers for the IRQCRn register)

The target registers are as follows:

- IRQCR0 to IRQCR15 registers
- WUPEN0.IRQWUPEN[15:0] bits

## 13.2.2 ICUSARB : Interrupt Controller Unit Security Attribution Register B

Base address: CPSCU = 0x4000_8000

Offset address: 0x44

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | SANMI |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SANMI | Security attributes of registers for nonmaskable interrupt<br>0: Secure<br>1: Non-secure | R/W |
| 31:1 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

### SANMI bit (Security attributes of registers for nonmaskable interrupt)

Security attributes of registers for non-maskable interrupt. The target registers are as follows:

- NMIER

- NMICLR
- NMICR

The value of AIRCR.BFHFNMINS bit [13] in Application Interrupt and Reset Control Register of ARM CPU should be the same as the value of security attribution. The initial values of AIRCR.BFHFNMINS and the SANMI bits are different. AIRCR.BFHFNMINS is secure and SANMI is non-secure. Polarity has the same meaning so program these to match.

Note: Only one of Secure and Non-Secure can set security attribution for non-maskable interrupt-related registers. If you program the Secure attribute as secure, it always goes to the Secure interrupt handler. To release any of the non-maskable interrupt sources to the non-secure user, write a function to execute a nonsecure program from the interrupt handler for Secure.

## 13.2.3 ICUSARC : Interrupt Controller Unit Security Attribution Register C

Base address: CPSCU = 0x4000_8000

Offset address: 0x48

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | SADMAC7 | SADMAC6 | SADMAC5 | SADMAC4 | SADMAC3 | SADMAC2 | SADMAC1 | SADMAC0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | SADMAC7 to SADMAC0 | Security attributes of registers for DMAC channel<br>0: Secure<br>1: Non-secure | R/W |
| 31:8 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

### SADMACn bits (Security attributes of registers for DMAC channel)

Security attributes of registers for DMAC channel. This register is referred to as the security attribute of the ICU and DMAC registers.

The controlled ICU register is:

- DELSRn

The controlled DMAC registers are:

- DMACn.DMSAR
- DMACn.DMSRR
- DMACn.DMDAR
- DMACn.DMDRR
- DMACn.DMCRA
- DMACn.DMCRB
- DMACn.DMTMD
- DMACn.DMINT
- DMACn.DMAMD
- DMACn.DMOFR

- DMACn.DMCNT
- DMACn.DMREQ
- DMACn.DMSTS
- DMACn.DMSBS
- DMACn.DMDBS

For details on DMAC registers, see section 16, DMA Controller (DMAC).

### 13.2.4 ICUSARD : Interrupt Controller Unit Security Attribution Register D

Base address: CPSCU = 0x4000_8000

Offset address: 0x4C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | SASELSR0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SASELSR0 | Security attributes of registers for SELSR0<br>0: Secure<br>1: Non-secure | R/W |
| 31:1 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

### 13.2.5 ICUSARE : Interrupt Controller Unit Security Attribution Register E

Base address: CPSCU = 0x4000_8000

Offset address: 0x50

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SAIIC0WUP | SAAGT1CBWUP | SAAGT1CAWUP | SAAGT1UDWUP | SAUSBFS0WUP | — | SARTCPRDWUP | SARTCALMWUP | — | — | — | — | SALVD2WUP | SALVD1WUP | — | SAIWDTWUP |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | — | These bits are read as 1. The write value should be 1. | R/W |
| 16 | SAIWDTWUP | Security attributes of registers for WUPEN0.b16<br>0: Secure<br>1: Non-secure | R/W |
| 17 | — | This bit is read as 1. The write value should be 1. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 18 | SALVD1WUP | Security attributes of registers for WUPEN0.b18<br>0: Secure<br>1: Non-secure | R/W |
| 19 | SALVD2WUP | Security attributes of registers for WUPEN0.b19<br>0: Secure<br>1: Non-secure | R/W |
| 23:20 | — | These bits are read as 1. The write value should be 1. | R/W |
| 24 | SARTCALMWUP | Security attributes of registers for WUPEN0.b24<br>0: Secure<br>1: Non-secure | R/W |
| 25 | SARTCPRDWUP | Security attributes of registers for WUPEN0.b25<br>0: Secure<br>1: Non-secure | R/W |
| 26 | — | These bits are read as 1. The write value should be 1. | R/W |
| 27 | SAUSBFS0WUP | Security attributes of registers for WUPEN0.b27<br>0: Secure<br>1: Non-secure | R/W |
| 28 | SAAGT1UDWUP | Security attributes of registers for WUPEN0.b28<br>0: Secure<br>1: Non-secure | R/W |
| 29 | SAAGT1CAWUP | Security attributes of registers for WUPEN0.b29<br>0: Secure<br>1: Non-secure | R/W |
| 30 | SAAGT1CBWUP | Security attributes of registers for WUPEN0.b30<br>0: Secure<br>1: Non-secure | R/W |
| 31 | SAIIC0WUP | Security attributes of registers for WUPEN0.b31<br>0: Secure<br>1: Non-secure | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

## 13.2.6 ICUSARF : Interrupt Controller Unit Security Attribution Register F

Base address: CPSCU = 0x4000_8000

Offset address: 0x54

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | SAAGT3CBWUP | SAAGT3CAWUP | SAAGT3UDWUP |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SAAGT3UDWUP | Security attributes of registers for WUPEN1.b0<br>0: Secure<br>1: Non-secure | R/W |
| 1 | SAAGT3CAWUP | Security attributes of registers for WUPEN1.b1<br>0: Secure<br>1: Non-secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2 | SAAGT3CBWUP | Security attributes of registers for WUPEN1.b2<br>0: Secure<br>1: Non-secure | R/W |
| 31:3 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

## 13.2.7 ICUSARG : Interrupt Controller Unit Security Attribution Register G

Base address: CPSCU = 0x4000_8000

Offset address: 0x70

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SAIELSR31 | SAIELSR30 | SAIELSR29 | SAIELSR28 | SAIELSR27 | SAIELSR26 | SAIELSR25 | SAIELSR24 | SAIELSR23 | SAIELSR22 | SAIELSR21 | SAIELSR20 | SAIELSR19 | SAIELSR18 | SAIELSR17 | SAIELSR16 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SAIELSR15 | SAIELSR14 | SAIELSR13 | SAIELSR12 | SAIELSR11 | SAIELSR10 | SAIELSR9 | SAIELSR8 | SAIELSR7 | SAIELSR6 | SAIELSR5 | SAIELSR4 | SAIELSR3 | SAIELSR2 | SAIELSR1 | SAIELSR0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | SAIELSR31 to SAIELSR0 | Security attributes of registers for IELSR31 to IELSR0<br>0: Secure<br>1: Non-secure | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

### SAIELSRn bits (Security attributes of registers for IELSR31 to IELSR0)

The Secure Attribute managed within the Arm CPU NVIC must match the security attribution of IELSEn ( n = 0 to 31 ). NVIC internal registers are in NVIC_ITNS0[31:0]. The initial values of NVIC_ITNS0 and ICUSARG are different. NVIC_ITNS0 is secure and ICUSARG is non-secure. Polarity has the same meaning so program these to match.

## 13.2.8 ICUSARH : Interrupt Controller Unit Security Attribution Register H

Base address: CPSCU = 0x4000_8000

Offset address: 0x74

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SAIELSR63 | SAIELSR62 | SAIELSR61 | SAIELSR60 | SAIELSR59 | SAIELSR58 | SAIELSR57 | SAIELSR56 | SAIELSR55 | SAIELSR54 | SAIELSR53 | SAIELSR52 | SAIELSR51 | SAIELSR50 | SAIELSR49 | SAIELSR48 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SAIELSR47 | SAIELSR46 | SAIELSR45 | SAIELSR44 | SAIELSR43 | SAIELSR42 | SAIELSR41 | SAIELSR40 | SAIELSR39 | SAIELSR38 | SAIELSR37 | SAIELSR36 | SAIELSR35 | SAIELSR34 | SAIELSR33 | SAIELSR32 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | SAIELSR63 to SAIELSR32 | Security attributes of registers for IELSR63 to IELSR32<br>0: Secure<br>1: Non-secure | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

### SAIELSRn bits (Security attributes of registers for IELSR63 to IELSR32)

The Secure Attribute managed within the ARM CPU NVIC must match the security attribution of IELSEn ( n = 32 to 63 ). NVIC internal registers are in NVIC_ITNS1[31:0]. The initial values of NVIC_ITNS1 and ICUSARH are different. NVIC_ITNS1 is secure and ICUSARH is non-secure. Polarity has the same meaning so program these to match.

## 13.2.9 ICUSARI : Interrupt Controller Unit Security Attribution Register I

Base address: CPSCU = 0x4000_8000

Offset address: 0x78

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SAIELSR95 | SAIELSR94 | SAIELSR93 | SAIELSR92 | SAIELSR91 | SAIELSR90 | SAIELSR89 | SAIELSR88 | SAIELSR87 | SAIELSR86 | SAIELSR85 | SAIELSR84 | SAIELSR83 | SAIELSR82 | SAIELSR81 | SAIELSR80 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SAIELSR79 | SAIELSR78 | SAIELSR77 | SAIELSR76 | SAIELSR75 | SAIELSR74 | SAIELSR73 | SAIELSR72 | SAIELSR71 | SAIELSR70 | SAIELSR69 | SAIELSR68 | SAIELSR67 | SAIELSR66 | SAIELSR65 | SAIELSR64 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | SAIELSR95 to SAIELSR64 | Security attributes of registers for IELSR95 to IELSR64<br>0: Secure<br>1: Non-secure | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

### SAIELSRn bits (Security attributes of registers for IELSR95 to IELSR64)

The Secure Attribute managed within the ARM CPU NVIC must match the security attribution of IELSEn ( n = 64 to 95 ). NVIC internal registers are in NVIC_ITNS2[31:0]. The initial values of NVIC_ITNS2 and ICUSARI are different. NVIC_ITNS2 is secure and ICUSARI is non-secure. Polarity has the same meaning so program these to match.

## 13.2.10 IRQCRi : IRQ Control Register i (i = 0 to 15)

Base address: ICU = 0x4000_6000

Offset address: 0x000 + 0x1 × i

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | FLTEN | — | FCLKSEL[1:0] | | — | — | IRQMD[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | IRQMD[1:0] | IRQi Detection Sense Select<br>0 0: Falling edge<br>0 1: Rising edge<br>1 0: Rising and falling edges<br>1 1: Low level | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:4 | FCLKSEL[1:0] | IRQi Digital Filter Sampling Clock Select<br>0 0: PCLKB<br>0 1: PCLKB/8<br>1 0: PCLKB/32<br>1 1: PCLKB/64 | R/W |
| 6 | — | This bit is read as 0. The write value should be 0. | R/W |
| 7 | FLTEN | IRQi Digital Filter Enable<br>0: Digital filter is disabled<br>1: Digital filter is enabled. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

IRQCRi register changes must satisfy the following conditions:

- For a CPU interrupt or DTC trigger:
  Change the IRQCRi register value before setting the target IELSRn register (n = 0 to 95).
  The register value should be changed only when the value of the target IELSRn register is 0x0000.
- For a DMAC trigger:
  Change the IRQCRi register value before setting the target DELSRn register (n = 0 to 7).
  The register value should be changed only when the value of the target DELSRn register is 0x0000.
- For a wakeup enable signal:
  Change the IRQCRi register setting before setting the target WUPEN0.IRQWUPEN[n] (n = 0 to 15). The register value should be changed when the target WUPEN0.IRQWUPEN[n] is 0.

#### IRQMD[1:0] bits (IRQi Detection Sense Select)

The IRQMD[1:0] bits set the detection sensing method for the IRQi external pin interrupt sources. For setting method when using external pin interrupt, see section 13.5.6. External Pin Interrupts.

#### FCLKSEL[1:0] bits (IRQi Digital Filter Sampling Clock Select)

The FCLKSEL[1:0] bits select the digital filter sampling clock for the IRQi external pin interrupt request pins, selectable to:

- PCLKB (every cycle)
- PCLKB/8 (once every 8 cycles)
- PCLKB/32 (once every 32 cycles)
- PCLKB/64 (once every 64 cycles)

For details of the digital filter, see section 13.5.5. Digital Filter.

#### FLTEN bit (IRQi Digital Filter Enable)

The FLTEN bit enables the digital filter used for the IRQi external pin interrupt sources. The digital filter is enabled when the IRQCRi.FLTEN bit is 1 and disabled when the IRQCRi.FLTEN bit is 0. The IRQi pin level is sampled at the clock cycle specified in the IRQCRi.FCLKSEL[1:0] bits. When the sampled level matches three times, the output level from the digital filter changes. For details of the digital filter, see section 13.5.5. Digital Filter.

### 13.2.11 NMISR : Non-Maskable Interrupt Status Register

Base address: ICU = 0x4000_6000

Offset address: 0x140

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | TZFST | — | BUSMST | — | RECCST | RPEST | NMIST | OSTST | — | — | LVD2ST | LVD1ST | WDTST | IWDTST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IWDTST | IWDT Underflow/Refresh Error Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 1 | WDTST | WDT Underflow/Refresh Error Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 2 | LVD1ST | Voltage Monitor 1 Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 3 | LVD2ST | Voltage Monitor 2 Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 5:4 | — | These bits are read as 0. | R |
| 6 | OSTST | Main Clock Oscillation Stop Detection Interrupt Status Flag<br>0: Interrupt not requested for main clock oscillation stop<br>1: Interrupt requested for main clock oscillation stop | R |
| 7 | NMIST | NMI Pin Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 8 | RPEST | SRAM Parity Error Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 9 | RECCST | SRAM ECC Error Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 10 | — | This bit is read as 0. | R |
| 11 | BUSMST | Bus Master MPU Error Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 12 | — | This bit is read as 0. | R |
| 13 | TZFST | TrustZone Filter Error Interrupt Status Flag<br>0: Interrupt not requested<br>1: Interrupt requested | R |
| 15:14 | — | These bits are read as 0. | R |

The NMISR register monitors the status of non-maskable interrupt sources. Writes to the NMISR register are ignored. The setting in the Non-Maskable Interrupt Enable Register (NMIER) does not affect the status flags in this register. Before the end of the non-maskable interrupt handler, check that all of the bits in this register are set to 0 to confirm that no other NMI requests are generated during handler processing.

### IWDTST flag (IWDT Underflow/Refresh Error Interrupt Status Flag)

The IWDTST flag indicates an IWDT underflow/refresh error interrupt request. It is read-only and cleared by the NMICLR.IWDTCLR bit.

[Setting condition]

When the IWDT underflow/refresh error interrupt is generated and this interrupt source is enabled.

[Clearing condition]

When 1 is written to the NMICLR.IWDTCLR bit.

### WDTST flag (WDT Underflow/Refresh Error Interrupt Status Flag)

The WDTST flag indicates a WDT underflow/refresh error interrupt request. It is read-only and cleared by the NMICLR.WDTCLR bit.

[Setting condition]

When the WDT underflow/refresh error interrupt is generated.

[Clearing condition]

When 1 is written to the NMICLR.WDTCLR bit.

#### LVD1ST flag (Voltage Monitor 1 Interrupt Status Flag)

The LVD1ST flag indicates a request for voltage monitor 1 interrupt. It is read-only and cleared by the NMICLR.LVD1CLR bit.

[Setting condition]

When the voltage monitor 1 interrupt is generated and this interrupt source is enabled.

[Clearing condition]

When 1 is written to the NMICLR.LVD1CLR bit.

#### LVD2ST flag (Voltage Monitor 2 Interrupt Status Flag)

The LVD2ST flag indicates a request for voltage monitor 2 interrupt. It is read-only and cleared by the NMICLR.LVD2CLR bit.

[Setting condition]

When the voltage monitor 2 interrupt is generated and this interrupt source is enabled.

[Clearing condition]

When 1 is written to the NMICLR.LVD2CLR bit.

#### OSTST flag (Main Clock Oscillation Stop Detection Interrupt Status Flag)

The OSTST flag indicates a main clock oscillation stop detection interrupt request. It is read-only and cleared by the NMICLR.OSTCLR bit.

[Setting condition]

When the main clock oscillation stop detection interrupt is generated.

[Clearing condition]

When 1 is written to the NMICLR.OSTCLR bit.

#### NMIST flag (NMI Pin Interrupt Status Flag)

The NMIST flag indicates an NMI pin interrupt request. It is read-only and cleared by the NMICLR.NMICLR bit.

[Setting condition]

When an edge specified by the NMICR.NMIMD bit is input to the NMI pin.

[Clearing condition]

When 1 is written to the NMICLR.NMICLR bit.

#### RPEST flag (SRAM Parity Error Interrupt Status Flag)

The RPEST flag indicates an SRAM parity error interrupt request.

[Setting condition]

When an interrupt is generated in response to an SRAM parity error.

[Clearing condition]

When 1 is written to the NMICLR.RPECLR bit.

#### RECCST flag (SRAM ECC Error Interrupt Status Flag)

The RECCST flag indicates an SRAM ECC error interrupt request.

[Setting condition]

When an interrupt is generated in response to an SRAM ECC error.

[Clearing condition]

When 1 is written to the NMICLR.RECCCLR bit.

**BUSMST flag (Bus Master MPU Error Interrupt Status Flag)**

The BUSMST flag indicates a bus master error interrupt request.

[Setting condition]

When an interrupt is generated in response to a bus master error.

[Clearing condition]

When 1 is written to the NMICLR.BUSMCLR bit.

**TZFST flag (TrustZone Filter Error Interrupt Status Flag)**

This flag indicates the TrustZone Filter error interrupt request.

[Setting condition]

When an interrupt is generated in response to a TrustZone Filter error

[Clearing condition]

When 1 is written to the NMICLR.TZFCLR bit

## 13.2.12 NMIER : Non-Maskable Interrupt Enable Register

Base address: ICU = 0x4000_6000

Offset address: 0x120

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | TZFEN | — | BUSMEN | — | RECCEN | RPEEN | NMIEN | OSTEN | — | — | LVD2EN | LVD1EN | WDTEN | IWDTEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IWDTEN | IWDT Underflow/Refresh Error Interrupt Enable<br>0: Disabled<br>1: Enabled. | R/W*1 *2 |
| 1 | WDTEN | WDT Underflow/Refresh Error Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 *2 |
| 2 | LVD1EN | Voltage monitor 1 Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 *2 |
| 3 | LVD2EN | Voltage monitor 2 Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 *2 |
| 5:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | OSTEN | Main Clock Oscillation Stop Detection Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 *2 |
| 7 | NMIEN | NMI Pin Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 |
| 8 | RPEEN | SRAM Parity Error Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 |
| 9 | RECCEN | SRAM ECC Error Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 |
| 10 | — | This bit is read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 11 | BUSMEN | Bus Master MPU Error Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 |
| 12 | — | This bit is read as 0. The write value should be 0. | R/W |
| 13 | TZFEN | TrustZone Filter Error Interrupt Enable<br>0: Disabled<br>1: Enabled | R/W*1 |
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. You can write 1 to this bit only once after reset. Subsequent write accesses are invalid. Writing 0 to this bit is invalid.
Note 2. Do not write 1 to this bit when the source is used as an event signal.

#### IWDTEN bit (IWDT Underflow/Refresh Error Interrupt Enable)

The IWDTEN bit enables IWDT underflow/refresh error interrupt as an NMI trigger.

#### WDTEN bit (WDT Underflow/Refresh Error Interrupt Enable)

The WDTEN bit enables WDT underflow/refresh error interrupt as an NMI trigger.

#### LVD1EN bit (Voltage monitor 1 Interrupt Enable)

The LVD1EN bit enables voltage monitor 1 interrupt as an NMI trigger.

#### LVD2EN bit (Voltage monitor 2 Interrupt Enable)

The LVD2EN bit enables voltage monitor 2 interrupt as an NMI trigger.

#### OSTEN bit (Main Clock Oscillation Stop Detection Interrupt Enable)

The OSTEN bit enables main clock oscillation stop detection interrupt as an NMI trigger.

#### NMIEN bit (NMI Pin Interrupt Enable)

The NMIEN bit enables NMI pin interrupt as an NMI trigger.

#### RPEEN bit (SRAM Parity Error Interrupt Enable)

The RPEEN bit enables SRAM parity error interrupt as an NMI trigger.

#### RECCEN bit (SRAM ECC Error Interrupt Enable)

The RECCEN bit enables SRAM ECC error interrupt as an NMI trigger.

#### BUSMEN bit (Bus Master MPU Error Interrupt Enable)

The BUSMEN bit enables bus master error interrupt as an NMI trigger.

#### TZFEN bit (TrustZone Filter Error Interrupt Enable)

TZFEN bit enables the TrustZone Filter error interrupt as an NMI trigger.

### 13.2.13 NMICLR : Non-Maskable Interrupt Status Clear Register

Base address: ICU = 0x4000_6000

Offset address: 0x130

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | TZFCLR | — | BUSMCLR | — | RECCCLR | RPECLR | NMICLR | OSTCLR | — | — | LVD2CLR | LVD1CLR | WDTCLR | IWDTCLR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IWDTCLR | IWDT Underflow/Refresh Error Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.IWDTST flag | R/W*1 |
| 1 | WDTCLR | WDT Underflow/Refresh Error Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.WDTST flag | R/W*1 |
| 2 | LVD1CLR | Voltage Monitor 1 Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.LVD1ST flag | R/W*1 |
| 3 | LVD2CLR | Voltage Monitor 2 Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.LVD2ST flag. | R/W*1 |
| 5:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | OSTCLR | Oscillation Stop Detection Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.OSTST flag | R/W*1 |
| 7 | NMICLR | NMI Pin Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.NMIST flag | R/W*1 |
| 8 | RPECLR | SRAM Parity Error Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.RPEST flag | R/W*1 |
| 9 | RECCCLR | SRAM ECC Error Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.RECCST flag | R/W*1 |
| 10 | — | This bit is read as 0. The write value should be 0. | R/W |
| 11 | BUSMCLR | Bus Master MPU Error Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.BUSMST flag | R/W*1 |
| 12 | — | This bit is read as 0. The write value should be 0. | R/W |
| 13 | TZFCLR | TrustZone Filter Error Interrupt Status Flag Clear<br>0: No effect<br>1: Clear the NMISR.TZFCLR flag | R/W*1 |
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. Only write 1 to this bit.

### IWDTCLR bit (IWDT Underflow/Refresh Error Interrupt Status Flag Clear)

Writing 1 to the IWDTCLR bit clears the NMISR.IWDTST flag. This bit is read as 0.

### WDTCLR bit (WDT Underflow/Refresh Error Interrupt Status Flag Clear)

Writing 1 to the WDTCLR bit clears the NMISR.WDTST flag. This bit is read as 0.

### LVD1CLR bit (Voltage Monitor 1 Interrupt Status Flag Clear)

Writing 1 to the LVD1CLR bit clears the NMISR.LVD1ST flag. This bit is read as 0.

### LVD2CLR bit (Voltage Monitor 2 Interrupt Status Flag Clear)

Writing 1 to the LVD2CLR bit clears the NMISR.LVD2ST flag. This bit is read as 0.

### OSTCLR bit (Oscillation Stop Detection Interrupt Status Flag Clear)

Writing 1 to the OSTCLR bit clears the NMISR.OSTST flag. This bit is read as 0.

### NMICLR bit (NMI Pin Interrupt Status Flag Clear)

Writing 1 to the NMICLR bit clears the NMISR.NMIST flag. This bit is read as 0.

### RPECLR bit (SRAM Parity Error Interrupt Status Flag Clear)

Writing 1 to the RPECLR bit clears the NMISR.RPEST flag. This bit is read as 0.

### RECCCLR bit (SRAM ECC Error Interrupt Status Flag Clear)

Writing 1 to the RECCCLR bit clears the NMISR.RECCST flag. This bit is read as 0.

### BUSMCLR bit (Bus Master MPU Error Interrupt Status Flag Clear)

Writing 1 to the BUSMCLR bit clears the NMISR.BUSMST flag. This bit is read as 0.

### TZFCLR bit (TrustZone Filter Error Interrupt Status Flag Clear)

Writing 1 to the TZFCLR bit clears the NMISR.TZFST flag. This bit is read as 0.

## 13.2.14 NMICR : NMI Pin Interrupt Control Register

Base address: ICU = 0x4000_6000

Offset address: 0x100

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | NFLTEN | — | NFCLKSEL[1:0] | | — | — | — | NMIMD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | NMIMD | NMI Detection Set<br>0: Falling edge<br>1: Rising edge | R/W |
| 3:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5:4 | NFCLKSEL[1:0] | NMI Digital Filter Sampling Clock Select<br>0 0: PCLKB<br>0 1: PCLKB/8<br>1 0: PCLKB/32<br>1 1: PCLKB/64 | R/W |
| 6 | — | This bit is read as 0. The write value should be 0. | R/W |
| 7 | NFLTEN | NMI Digital Filter Enable<br>0: Disabled.<br>1: Enabled. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Change the NMICR register settings before enabling NMI pin interrupts, that is, before setting NMIER.NMIEN to 1.

### NMIMD bit (NMI Detection Set)

The NMIMD bit selects the detection sensing method for the NMI pin interrupts.

### NFCLKSEL[1:0] bits (NMI Digital Filter Sampling Clock Select)

The NFCLKSEL[1:0] bits select the digital filter sampling clock for the NMI pin interrupts, selectable to:

- PCLKB (every cycle)
- PCLKB/8 (once every 8 cycles)
- PCLKB/32 (once every 32 cycles)
- PCLKB/64 (once every 64 cycles)

For details of the digital filter, see section 13.5.5. Digital Filter.

### NFLTEN bit (NMI Digital Filter Enable)

The NFLTEN bit enables the digital filter used for NMI pin interrupts. The filter is enabled when NFLTEN is 1, and disabled when NFLTEN is 0. The NMI pin level is sampled at the clock cycle specified in NFCLKSEL[1:0]. When the sampled level matches three times, the output level from the digital filter changes. For details of the digital filter, see section 13.5.5. Digital Filter.

## 13.2.15 IELSRn : ICU Event Link Setting Register n (n = 0 to 95)

Base address: ICU = 0x4000_6000

Offset address: 0x300 + 0x4 × n

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | DTCE | — | — | — | — | — | — | — | IR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | IELS[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | IELS[8:0] | ICU Event Link Select<br>0x00: Disable interrupts to the associated NVIC or DTC module<br>Others: Event signal number to be linked. For details, see section 13.3.2. Event Number. | R/W |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | IR | Interrupt Status Flag<br>0: No interrupt request generated.<br>1: An interrupt request is generated. | R/W*1 |
| 23:17 | — | These bits are read as 0. The write value should be 0. | R/W |
| 24 | DTCE | DTC Activation Enable<br>0: DTC activation is disabled.<br>1: DTC activation is enabled. | R/W |
| 31:25 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: This register requires halfword or word access.

Note 1. Writing 1 to the IR flag is prohibited.

The IELSRn register selects the IRQi source used by the NVIC. For details, see Table 13.3. IELSRn corresponds to the NVIC IRQ input source number, where n = 0 to 95.

### IELS[8:0] bits (ICU Event Link Select)

The IELS[8:0] bits link an event signal to the associated NVIC or DTC module. Event options are classified into 8 groups (groups 0 to 7). For details, see Table 13.3 and Table 13.4.

### IR flag (Interrupt Status Flag)

The IR status flag indicates an individual interrupt request from the event specified in IELS[8:0].

[Setting condition]

When an interrupt request is received from the associated peripheral module or IRQi pin.

[Clearing condition]

- The IR flag is cleared to 0 by writing 0.

- At the time other than the final transfer transfer end in DTC transfer during DTCE = 1, IR flag repeat set and cleared by Hardware.

When DTC transfer except last transfer is completed (DTCE bit is changed from 1 to 0).

During DTCE = 1, write 0 to IR register is prohibited.

In the case of level detection, clear of the IR flag should follow the steps below.

1. Negate the input interrupt signal.
2. Read access the peripheral once and wait for 2 clock cycles of the target module clock.
3. Clear the IR flag by writing 0.

### DTCE bit (DTC Activation Enable)

When the DTCE bit is set to 1, the associated event is selected as the source for DTC activation.

[Setting condition]

- When 1 is written to the DTCE bit.

[Clearing condition]

- When the specified number of transfers is complete. For chain transfers, when the specified number of transfers for the last chain transfer is complete.
- When 0 is written to the DTCE bit.

Note: Error during DTC transfer

If an error response occurs during DTC transfer, the DTC notifies the ICU that an error has occurred. ICU clears all bits of the target IELSRn (n = 0 to 95). IELSRn that is not the target is not cleared.

Note: DTC transfer error in snooze mode

When an error occurs in DTC transfer in Snooze mode, ICU issues a wakeup request. However, interrupt requests are not issued automatically. See section 17, Data Transfer Controller (DTC) for how to set the interrupt when a DTC error occurs.

## 13.2.16 DELSRn : DMAC Event Link Setting Register n (n = 0 to 7)

Base address: ICU = 0x4000_6000

Offset address: 0x280 + 0x4 × n

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | IR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | DELS[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | DELS[8:0] | DMAC Event Link Select<br>0x00: Disable interrupts to the associated DMAC module.<br>Others: Event signal number to be linked. For details, see Table 13.4. | R/W |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | IR | DMAC Activation Request Status Flag<br>0: No DMAC activation request occurred.<br>1: DMAC activation request occurred. | R/W*1 |
| 31:17 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed

- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:

- Secure and Non-secure access are allowed.

Note 1. Writing 1 to the IR flag is prohibited.

### DELS[8:0] bit (DMAC Event Link Select)

The DELS[8:0] bits link an event signal to the associated DMAC module. Do not set the same event number in multiple DELSRn registers.

### IR flag (DMAC Activation Request Status Flag)

The IR flag is the status flag of a DMAC activation request. This flag is associated with the DELS[8:0] bits of this register.

[Setting condition]

The flag is set to 1 when a DMAC activation request is generated from the associated peripheral module or IRQi pin.

[Clearing conditions]

- When 0 is written to the IR flag.
- At the start of a DMA transfer after the DMAC activation request is issued.

Note: The IR flag is automatically cleared after completion of a DMA transfer. Therefore, do not write 0 unless an abort occurs. When 0 is written, DMA transfer operation cannot be guaranteed.

Note: Error during DMAC transfer

If an error response occurs during a DMAC transfer, the DMAC notifies the ICU that an error has occurred.

The ICU clears all bits of the target channel of DELSRn (n = 0 to 7). DELSRn that is not the target channel is not cleared.

## 13.2.17 SELSR0 : SYS Event Link Setting Register

Base address: ICU = 0x4000_6000

Offset address: 0x200

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | SELS[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | SELS[8:0] | SYS Event Link Select<br>0x00: Disable event output to the associated low-power mode module<br>Others: Event signal number to be linked. For details, see Table 13.4. | R/W |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:

- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:

- Secure and Non-secure access are allowed.

The SELSR0 register selects the events that wake up the CPU from Snooze mode. You can use only the events listed in Table 13.4 checked as “Canceling Snooze mode”. When ICU_SNZCANCEL is selected in the IELSRn.IELS[8:0] bits, an interrupt is generated that cancels snooze mode.

**Caution: For security attribution added to parts related to a series of actions, make sure to match all security attribution so that security holes cannot be created.**

**About security attribution to be matched**

- **Event source to be set to SELSR0.**
- **SELSR0**
- **IELSRn (n = 0 to 95) to receive event No. 45 (ICU_SNZCANCEL).**
- **NVIC internal registers in the CPU of the interrupt specified in the previous item.**
- **Interrupt Handler.**

## 13.2.18 WUPEN0 : Wake Up Interrupt Enable Register 0

Base address: ICU = 0x4000_6000

Offset address: 0x1A0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | IIC0WUPEN | AGT1CBWUPEN | AGT1CAWUPEN | AGT1UDWUPEN | USBFS0WUPEN | — | RTCPRDWUPEN | RTCALMWUPEN | — | — | — | — | LVD2WUPEN | LVD1WUPEN | — | IWDTWUPEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | IRQWUPEN[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | IRQWUPEN[15:0] | IRQn Interrupt Software Standby/Snooze Mode Returns Enable bit (n = 0 to 15)<br>0: Software Standby/Snooze Mode returns by IRQn interrupt is disabled<br>1: Software Standby/Snooze Mode returns by IRQn interrupt is enabled*1 | R/W |
| 16 | IWDTWUPEN | IWDT Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by IWDT interrupt is disabled<br>1: Software Standby/Snooze Mode returns by IWDT interrupt is enabled | R/W |
| 17 | — | This bit is read as 0. The write value should be 0. | R/W |
| 18 | LVD1WUPEN | LVD1 Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by LVD1 interrupt is disabled<br>1: Software Standby/Snooze Mode returns by LVD1 interrupt is enabled | R/W |
| 19 | LVD2WUPEN | LVD2 Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by LVD2 interrupt is disabled<br>1: Software Standby/Snooze Mode returns by LVD2 interrupt is enabled | R/W |
| 23:20 | — | These bits are read as 0. The write value should be 0. | R/W |
| 24 | RTCALMWUPEN | RTC Alarm Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by RTC alarm interrupt is disabled<br>1: Software Standby/Snooze Mode returns by RTC alarm interrupt is enabled | R/W |
| 25 | RTCPRDWUPEN | RTC Period Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by RTC period interrupt is disabled<br>1: Software Standby/Snooze Mode returns by RTC period interrupt is enabled | R/W |
| 26 | — | This bit is read as 0. The write value should be 0. | R/W |
| 27 | USBFS0WUPEN | USBFS0 Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by USBFS0 interrupt is disabled<br>1: Software Standby/Snooze Mode returns by USBFS0 interrupt is enabled | R/W |
| 28 | AGT1UDWUPEN | AGT1 Underflow Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by AGT1 underflow interrupt is disabled<br>1: Software Standby/Snooze Mode returns by AGT1 underflow interrupt is enabled | R/W |
| 29 | AGT1CAWUPEN | AGT1 Compare Match A Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by AGT1 compare match A interrupt is disabled<br>1: Software Standby/Snooze Mode returns by AGT1 compare match A interrupt is enabled | R/W |
| 30 | AGT1CBWUPEN | AGT1 Compare Match B Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by AGT1 compare match B interrupt is disabled<br>1: Software Standby/Snooze Mode returns by AGT1 compare match B interrupt is enabled | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31 | IIC0WUPEN | IIC0 Address Match Interrupt Software Standby/Snooze Mode Returns Enable bit<br>0: Software Standby/Snooze Mode returns by IIC0 address match interrupt is disabled<br>1: Software Standby/Snooze Mode returns by IIC0 address match interrupt is enabled | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. Description is a description of each bit.

Note: The security attribution of this register is set for each wakeup event.

To avoid the occurrence of a security hole, the target event of a wakeup and the security attribution added to this bit must match.

## 13.2.19 WUPEN1 : Wake Up interrupt enable register 1

Base address: ICU = 0x4000_6000

Offset address: 0x1A4

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | AGT3 CBWU PEN | AGT3 CAWU PEN | AGT3 UDWU PEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | AGT3UDWUPEN | AGT3 Underflow Interrupt Software Standby Return Enable bit<br>0: Software standby returns by AGT3 underflow interrupt is disabled<br>1: Software standby returns by AGT3 underflow interrupt is enabled | R/W |
| 1 | AGT3CAWUPEN | AGT3 Compare Match A Interrupt Software Standby Return Enable bit<br>0: Software standby returns by AGT3 compare match A interrupt is disabled<br>1: Software standby returns by AGT3 compare match A interrupt is enabled | R/W |
| 2 | AGT3CBWUPEN | AGT3 Compare Match B Interrupt Software Standby Return Enable bit<br>0: Software standby returns by AGT3 compare match B interrupt is disabled<br>1: Software standby returns by AGT3 compare match B interrupt is enabled | R/W |
| 31:3 | — | These bits are read as 0. The write value should be 0. | R*1 |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. This bit is read only

### AGT3UDWUPEN bit (AGT3 Underflow Interrupt Software Standby Return Enable bit)

This bit is the enable bit to control the use of the AGT3 underflow interrupt as an Software standby return factor.

### AGT3CAWUPEN bit (AGT3 Compare Match A Interrupt Software Standby Return Enable bit)

This bit is the enable bit to control the use of the AGT3 compare match A interrupt as an Software standby return factor.

### AGT3CBWUPEN bit (AGT3 Compare Match B Interrupt Software Standby Return Enable bit)

This bit is the enable bit to control the use of the AGT3 compare match B interrupt as an Software standby return factor.

Note: The security attribution of this register is set for each wakeup event.
To avoid the occurrence of a security hole, the target event of a wakeup and the security attribution added to this bit must match.

## 13.3 Vector Table

The ICU detects maskable and non-maskable interrupts. Interrupt priorities are set up in the Arm NVIC. For information about these registers, see section 13.9. Reference.

### 13.3.1 Interrupt Vector Table

Table 13.3 describes the interrupt vector table. The interrupt vector addresses conform to the NVIC specifications.

**Table 13.3 Interrupt vector table (1 of 3)**

| Exception number | IRQ number | Vector offset | Source | Description |
|---|---|---|---|---|
| 0 | — | 0x000 | Arm | Initial stack pointer |
| 1 | — | 0x004 | Arm | Initial program counter (reset vector) |
| 2 | — | 0x008 | Arm | Non-Maskable Interrupt (NMI) |
| 3 | — | 0x00C | Arm | Hard Fault |
| 4 | — | 0x010 | Arm | MemManage fault |
| 5 | — | 0x014 | Arm | BusFault |
| 6 | — | 0x018 | Arm | UsageFault |
| 7 | — | 0x01C | Arm | SecureFault |
| 8 | — | 0x020 | Arm | Reserved |
| 9 | — | 0x024 | Arm | Reserved |
| 10 | — | 0x028 | Arm | Reserved |
| 11 | — | 0x02C | Arm | Supervisor Call (SVCall) |
| 12 | — | 0x030 | Arm | DebugMonitor |
| 13 | — | 0x034 | Arm | Reserved |
| 14 | — | 0x038 | Arm | Pendable request for system service (PendableSrvReq) |
| 15 | — | 0x03C | Arm | System Tick Timer (SysTick) |
| 16 | 0 | 0x040 | ICU.IELSR0 | Event selected in the ICU.IELSR0 register |
| 17 | 1 | 0x044 | ICU.IELSR1 | Event selected in the ICU.IELSR1 register |
| 18 | 2 | 0x048 | ICU.IELSR2 | Event selected in the ICU.IELSR2 register |
| 19 | 3 | 0x04C | ICU.IELSR3 | Event selected in the ICU.IELSR3 register |
| 20 | 4 | 0x050 | ICU.IELSR4 | Event selected in the ICU.IELSR4 register |
| 21 | 5 | 0x054 | ICU.IELSR5 | Event selected in the ICU.IELSR5 register |
| 22 | 6 | 0x058 | ICU.IELSR6 | Event selected in the ICU.IELSR6 register |
| 23 | 7 | 0x05C | ICU.IELSR7 | Event selected in the ICU.IELSR7 register |
| 24 | 8 | 0x060 | ICU.IELSR8 | Event selected in the ICU.IELSR8 register |
| 25 | 9 | 0x064 | ICU.IELSR9 | Event selected in the ICU.IELSR9 register |
| 26 | 10 | 0x068 | ICU.IELSR10 | Event selected in the ICU.IELSR10 register |
| 27 | 11 | 0x06C | ICU.IELSR11 | Event selected in the ICU.IELSR11 register |
| 28 | 12 | 0x070 | ICU.IELSR12 | Event selected in the ICU.IELSR12 register |
| 29 | 13 | 0x074 | ICU.IELSR13 | Event selected in the ICU.IELSR13 register |
| 30 | 14 | 0x078 | ICU.IELSR14 | Event selected in the ICU.IELSR14 register |
| 31 | 15 | 0x07C | ICU.IELSR15 | Event selected in the ICU.IELSR15 register |

**Table 13.3 Interrupt vector table (2 of 3)**

| Exception number | IRQ number | Vector offset | Source | Description |
|---|---|---|---|---|
| 32 | 16 | 0x080 | ICU.IELSR16 | Event selected in the ICU.IELSR16 register |
| 33 | 17 | 0x084 | ICU.IELSR17 | Event selected in the ICU.IELSR17 register |
| 34 | 18 | 0x088 | ICU.IELSR18 | Event selected in the ICU.IELSR18 register |
| 35 | 19 | 0x08C | ICU.IELSR19 | Event selected in the ICU.IELSR19 register |
| 36 | 20 | 0x090 | ICU.IELSR20 | Event selected in the ICU.IELSR20 register |
| 37 | 21 | 0x094 | ICU.IELSR21 | Event selected in the ICU.IELSR21 register |
| 38 | 22 | 0x098 | ICU.IELSR22 | Event selected in the ICU.IELSR22 register |
| 39 | 23 | 0x09C | ICU.IELSR23 | Event selected in the ICU.IELSR23 register |
| 40 | 24 | 0x0A0 | ICU.IELSR24 | Event selected in the ICU.IELSR24 register |
| 41 | 25 | 0x0A4 | ICU.IELSR25 | Event selected in the ICU.IELSR25 register |
| 42 | 26 | 0x0A8 | ICU.IELSR26 | Event selected in the ICU.IELSR26 register |
| 43 | 27 | 0x0AC | ICU.IELSR27 | Event selected in the ICU.IELSR27 register |
| 44 | 28 | 0x0B0 | ICU.IELSR28 | Event selected in the ICU.IELSR28 register |
| 45 | 29 | 0x0B4 | ICU.IELSR29 | Event selected in the ICU.IELSR29 register |
| 46 | 30 | 0x0B8 | ICU.IELSR30 | Event selected in the ICU.IELSR30 register |
| 47 | 31 | 0x0BC | ICU.IELSR31 | Event selected in the ICU.IELSR31 register |
| 48 | 32 | 0x0C0 | ICU.IELSR32 | Event selected in the ICU.IELSR32 register |
| 49 | 33 | 0x0C4 | ICU.IELSR33 | Event selected in the ICU.IELSR33 register |
| 50 | 34 | 0x0C8 | ICU.IELSR34 | Event selected in the ICU.IELSR34 register |
| 51 | 35 | 0x0CC | ICU.IELSR35 | Event selected in the ICU.IELSR35 register |
| 52 | 36 | 0x0D0 | ICU.IELSR36 | Event selected in the ICU.IELSR36 register |
| 53 | 37 | 0x0D4 | ICU.IELSR37 | Event selected in the ICU.IELSR37 register |
| 54 | 38 | 0x0D8 | ICU.IELSR38 | Event selected in the ICU.IELSR38 register |
| 55 | 39 | 0x0DC | ICU.IELSR39 | Event selected in the ICU.IELSR39 register |
| 56 | 40 | 0x0E0 | ICU.IELSR40 | Event selected in the ICU.IELSR40 register |
| 57 | 41 | 0x0E4 | ICU.IELSR41 | Event selected in the ICU.IELSR41 register |
| 58 | 42 | 0x0E8 | ICU.IELSR42 | Event selected in the ICU.IELSR42 register |
| 59 | 43 | 0x0EC | ICU.IELSR43 | Event selected in the ICU.IELSR43 register |
| 60 | 44 | 0x0F0 | ICU.IELSR44 | Event selected in the ICU.IELSR44 register |
| 61 | 45 | 0x0F4 | ICU.IELSR45 | Event selected in the ICU.IELSR45 register |
| 62 | 46 | 0x0F8 | ICU.IELSR46 | Event selected in the ICU.IELSR46 register |
| 63 | 47 | 0x0FC | ICU.IELSR47 | Event selected in the ICU.IELSR47 register |
| 64 | 48 | 0x100 | ICU.IELSR48 | Event selected in the ICU.IELSR48 register |
| 65 | 49 | 0x104 | ICU.IELSR49 | Event selected in the ICU.IELSR49 register |
| 66 | 50 | 0x108 | ICU.IELSR50 | Event selected in the ICU.IELSR50 register |
| 67 | 51 | 0x10C | ICU.IELSR51 | Event selected in the ICU.IELSR51 register |
| 68 | 52 | 0x110 | ICU.IELSR52 | Event selected in the ICU.IELSR52 register |
| 69 | 53 | 0x114 | ICU.IELSR53 | Event selected in the ICU.IELSR53 register |
| 70 | 54 | 0x118 | ICU.IELSR54 | Event selected in the ICU.IELSR54 register |
| 71 | 55 | 0x11C | ICU.IELSR55 | Event selected in the ICU.IELSR55 register |
| 72 | 56 | 0x120 | ICU.IELSR56 | Event selected in the ICU.IELSR56 register |

**Table 13.3 Interrupt vector table (3 of 3)**

| Exception number | IRQ number | Vector offset | Source | Description |
|---|---|---|---|---|
| 73 | 57 | 0x124 | ICU.IELSR57 | Event selected in the ICU.IELSR57 register |
| 74 | 58 | 0x128 | ICU.IELSR58 | Event selected in the ICU.IELSR58 register |
| 75 | 59 | 0x12C | ICU.IELSR59 | Event selected in the ICU.IELSR59 register |
| 76 | 60 | 0x130 | ICU.IELSR60 | Event selected in the ICU.IELSR60 register |
| 77 | 61 | 0x134 | ICU.IELSR61 | Event selected in the ICU.IELSR61 register |
| 78 | 62 | 0x138 | ICU.IELSR62 | Event selected in the ICU.IELSR62 register |
| 79 | 63 | 0x13C | ICU.IELSR63 | Event selected in the ICU.IELSR63 register |
| 80 | 64 | 0x140 | ICU.IELSR64 | Event selected in the ICU.IELSR64 register |
| 81 | 65 | 0x144 | ICU.IELSR65 | Event selected in the ICU.IELSR65 register |
| 82 | 66 | 0x148 | ICU.IELSR66 | Event selected in the ICU.IELSR66 register |
| 83 | 67 | 0x14C | ICU.IELSR67 | Event selected in the ICU.IELSR67 register |
| 84 | 68 | 0x150 | ICU.IELSR68 | Event selected in the ICU.IELSR68 register |
| 85 | 69 | 0x154 | ICU.IELSR69 | Event selected in the ICU.IELSR69 register |
| 86 | 70 | 0x158 | ICU.IELSR70 | Event selected in the ICU.IELSR70 register |
| 87 | 71 | 0x15C | ICU.IELSR71 | Event selected in the ICU.IELSR71 register |
| 88 | 72 | 0x160 | ICU.IELSR72 | Event selected in the ICU.IELSR72 register |
| 89 | 73 | 0x164 | ICU.IELSR73 | Event selected in the ICU.IELSR73 register |
| 90 | 74 | 0x168 | ICU.IELSR74 | Event selected in the ICU.IELSR74 register |
| 91 | 75 | 0x16C | ICU.IELSR75 | Event selected in the ICU.IELSR75 register |
| 92 | 76 | 0x170 | ICU.IELSR76 | Event selected in the ICU.IELSR76 register |
| 93 | 77 | 0x174 | ICU.IELSR77 | Event selected in the ICU.IELSR77 register |
| 94 | 78 | 0x178 | ICU.IELSR78 | Event selected in the ICU.IELSR78 register |
| 95 | 79 | 0x17C | ICU.IELSR79 | Event selected in the ICU.IELSR79 register |
| 96 | 80 | 0x180 | ICU.IELSR80 | Event selected in the ICU.IELSR80 register |
| 97 | 81 | 0x184 | ICU.IELSR81 | Event selected in the ICU.IELSR81 register |
| 98 | 82 | 0x188 | ICU.IELSR82 | Event selected in the ICU.IELSR82 register |
| 99 | 83 | 0x18C | ICU.IELSR83 | Event selected in the ICU.IELSR83 register |
| 100 | 84 | 0x190 | ICU.IELSR84 | Event selected in the ICU.IELSR84 register |
| 101 | 85 | 0x194 | ICU.IELSR85 | Event selected in the ICU.IELSR85 register |
| 102 | 86 | 0x198 | ICU.IELSR86 | Event selected in the ICU.IELSR86 register |
| 103 | 87 | 0x19C | ICU.IELSR87 | Event selected in the ICU.IELSR87 register |
| 104 | 88 | 0x1A0 | ICU.IELSR88 | Event selected in the ICU.IELSR88 register |
| 105 | 89 | 0x1A4 | ICU.IELSR89 | Event selected in the ICU.IELSR89 register |
| 106 | 90 | 0x1A8 | ICU.IELSR90 | Event selected in the ICU.IELSR90 register |
| 107 | 91 | 0x1AC | ICU.IELSR91 | Event selected in the ICU.IELSR91 register |
| 108 | 92 | 0x1B0 | ICU.IELSR92 | Event selected in the ICU.IELSR92 register |
| 109 | 93 | 0x1B4 | ICU.IELSR93 | Event selected in the ICU.IELSR93 register |
| 110 | 94 | 0x1B8 | ICU.IELSR94 | Event selected in the ICU.IELSR94 register |
| 111 | 95 | 0x1BC | ICU.IELSR95 | Event selected in the ICU.IELSR95 register |

### 13.3.2 Event Number

The following table lists heading details for Table 13.4, which describes each event number.

| Heading | Description |
|---|---|
| Interrupt request source | Name of the source generating the interrupt request |
| Name | Name of the interrupt |
| Connect to NVIC | " ✓" indicates the interrupt can be used as a CPU interrupt |
| Invoke DTC | " ✓" indicates the interrupt can be used to request DTC activation |
| Invoke DMAC | "✓" indicates the interrupt can be used to request DMAC activation |
| Canceling Snooze | " ✓" indicates the interrupt can be used to request a return from Snooze mode |
| Canceling Software Standby | " ✓" indicates the interrupt can be used to request a return from Software Standby mode |
| Canceling Deep Software Standby | "✓" indicates the interrupt can be used to request a return from Deep Software Standby mode |

**Table 13.4 Event table (1 of 7)**

| Event number | Interrupt request source | Name | IELSRn | | DELSRn | Canceling Snooze | Canceling Software Standby | Canceling Deep Software Standby |
|---|---|---|---|---|---|---|---|---|
| | | | Connect to NVIC | Invoke DTC | Invoke DMAC | | | |
| 0x001 | Port | PORT_IRQ0 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x002 | | PORT_IRQ1 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x003 | | PORT_IRQ2 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x004 | | PORT_IRQ3 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x005 | | PORT_IRQ4 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x006 | | PORT_IRQ5 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x007 | | PORT_IRQ6 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x008 | | PORT_IRQ7 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x009 | | PORT_IRQ8 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x00A | | PORT_IRQ9 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x00B | | PORT_IRQ10 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x00C | | PORT_IRQ11 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x00D | | PORT_IRQ12 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x00E | | PORT_IRQ13 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x00F | | PORT_IRQ14 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x010 | | PORT_IRQ15 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x020 | DMAC0 | DMAC0_INT | ✓ | ✓ | — | — | — | — |
| 0x021 | DMAC1 | DMAC1_INT | ✓ | ✓ | — | — | — | — |
| 0x022 | DMAC2 | DMAC2_INT | ✓ | ✓ | — | — | — | — |
| 0x023 | DMAC3 | DMAC3_INT | ✓ | ✓ | — | — | — | — |
| 0x024 | DMAC4 | DMAC4_INT | ✓ | ✓ | — | — | — | — |
| 0x025 | DMAC5 | DMAC5_INT | ✓ | ✓ | — | — | — | — |
| 0x026 | DMAC6 | DMAC6_INT | ✓ | ✓ | — | — | — | — |
| 0x027 | DMAC7 | DMAC7_INT | ✓ | ✓ | — | — | — | — |
| 0x029 | DTC | DTC_COMPLETE | ✓ | — | — | ✓*3 | — | — |
| 0x02B | DMAC/DTC | DMA_TRANSERR | ✓ | — | — | ✓ | — | — |
| 0x02D | ICU | ICU_SNZCANCEL | ✓ | — | — | ✓ | — | — |

**Table 13.4 Event table (2 of 7)**

| Event number | Interrupt request source | Name | IELSRn | | DELSRn | Canceling Snooze | Canceling Software Standby | Canceling Deep Software Standby |
|---|---|---|---|---|---|---|---|---|
| | | | Connect to NVIC | Invoke DTC | Invoke DMAC | | | |
| 0x030 | FCU | FCU_FIFERR | ✓ | — | — | — | — | — |
| 0x031 | | FCU_FRDYI | ✓ | — | — | — | — | — |
| 0x038 | LVD | LVD_LVD1 | ✓ | — | — | ✓ | ✓ | ✓ |
| 0x039 | | LVD_LVD2 | ✓ | — | — | ✓ | ✓ | ✓ |
| 0x03B | MOSC | MOSC_STOP | ✓ | — | — | — | — | — |
| 0x03C | LPW | SYSTEM_SNZREQ | — | ✓ | — | — | — | — |
| 0x040 | AGT0 | AGT0_AGTI | ✓ | ✓ | ✓ | — | — | — |
| 0x041 | | AGT0_AGTCMAI | ✓ | ✓ | ✓ | — | — | — |
| 0x042 | | AGT0_AGTCMBI | ✓ | ✓ | ✓ | — | — | — |
| 0x043 | AGT1 | AGT1_AGTI | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x044 | | AGT1_AGTCMAI | ✓ | ✓ | ✓ | ✓ | ✓ | — |
| 0x045 | | AGT1_AGTCMBI | ✓ | ✓ | ✓ | ✓ | ✓ | — |
| 0x046 | AGT2 | AGT2_AGTI | ✓ | ✓ | ✓ | — | — | — |
| 0x047 | | AGT2_AGTCMAI | ✓ | ✓ | ✓ | — | — | — |
| 0x048 | | AGT2_AGTCMBI | ✓ | ✓ | ✓ | — | — | — |
| 0x049 | AGT3 | AGT3_AGTI | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 0x04A | | AGT3_AGTCMAI | ✓ | ✓ | ✓ | ✓ | ✓ | — |
| 0x04B | | AGT3_AGTCMBI | ✓ | ✓ | ✓ | ✓ | ✓ | — |
| 0x04C | AGT4 | AGT4_AGTI | ✓ | ✓ | ✓ | — | — | — |
| 0x04D | | AGT4_AGTCMAI | ✓ | ✓ | ✓ | — | — | — |
| 0x04E | | AGT4_AGTCMBI | ✓ | ✓ | ✓ | — | — | — |
| 0x04F | AGT5 | AGT5_AGTI | ✓ | ✓ | ✓ | — | — | — |
| 0x050 | | AGT5_AGTCMAI | ✓ | ✓ | ✓ | — | — | — |
| 0x051 | | AGT5_AGTCMBI | ✓ | ✓ | ✓ | — | — | — |
| 0x052 | IWDT | IWDT_NMIUNDF | ✓ | — | — | ✓ | ✓ | — |
| 0x053 | WDT | WDT_NMIUNDF | ✓ | — | — | — | — | — |
| 0x054 | RTC | RTC_ALM | ✓ | — | — | ✓ | ✓ | ✓ |
| 0x055 | | RTC_PRD | ✓ | — | — | ✓ | ✓ | ✓ |
| 0x056 | | RTC_CUP | ✓ | — | — | — | — | — |
| 0x06B | USBFS | USBFS0_D0FIFO | ✓ | ✓ | ✓ | — | — | — |
| 0x06C | | USBFS0_D1FIFO | ✓ | ✓ | ✓ | — | — | — |
| 0x06D | | USBFS0_USBI | ✓ | — | — | — | — | — |
| 0x06E | | USBFS0_USBR | ✓ | — | — | ✓ | ✓ | ✓ |
| 0x073 | IIC0 | IIC0_RXI | ✓ | ✓ | ✓ | — | — | — |
| 0x074 | | IIC0_TXI | ✓ | ✓ | ✓ | — | — | — |
| 0x075 | | IIC0_TEI | ✓ | — | — | — | — | — |
| 0x076 | | IIC0_EEI | ✓ | — | — | — | — | — |
| 0x077 | | IIC0_WUI | ✓ | — | — | ✓ | ✓ | — |

**Table 13.4 Event table (3 of 7)**

| Event number | Interrupt request source | Name | IELSRn | | DELSRn | Canceling Snooze | Canceling Software Standby | Canceling Deep Software Standby |
|---|---|---|---|---|---|---|---|---|
| | | | Connect to NVIC | Invoke DTC | Invoke DMAC | | | |
| 0x078 | IIC1 | IIC1_RXI | ✓ | ✓ | ✓ | — | — | — |
| 0x079 | | IIC1_TXI | ✓ | ✓ | ✓ | — | — | — |
| 0x07A | | IIC1_TEI | ✓ | — | — | — | — | — |
| 0x07B | | IIC1_EEI | ✓ | — | — | — | — | — |
| 0x082 | SDHI MMC | SDHI_MMC0_ACCS | ✓ | — | — | — | — | — |
| 0x083 | | SDHI_MMC0_SDIO | ✓ | — | — | — | — | — |
| 0x084 | | SDHI_MMC0_CARD | ✓ | — | — | — | — | — |
| 0x085 | | SDHI_MMC0_ODMSDBREQ | — | ✓ | ✓ | — | — | — |
| 0x08A | SSI | SSI0_SSITXI | ✓ | ✓ | ✓ | — | — | — |
| 0x08B | | SSI0_SSIRXI | ✓ | ✓ | ✓ | — | — | — |
| 0x08D | | SSI0_SSIF | ✓ | — | — | — | — | — |
| 0x09A | CTSU | CTSU_CTSUWR | ✓ | ✓ | ✓ | — | — | — |
| 0x09B | | CTSU_CTSURD | ✓ | ✓ | ✓ | — | — | — |
| 0x09C | | CTSU_CTSUFN | ✓ | — | — | ✓*3 | — | — |
| 0x09E | CAC | CAC_FERRI | ✓ | — | — | — | — | — |
| 0x09F | | CAC_MENDI | ✓ | — | — | — | — | — |
| 0x0A0 | | CAC_OVFI | ✓ | — | — | — | — | — |
| 0x0A1 | CAN0 | CAN0_ERS | ✓ | — | — | — | — | — |
| 0x0A2 | | CAN0_RXF | ✓ | — | — | — | — | — |
| 0x0A3 | | CAN0_TXF | ✓ | — | — | — | — | — |
| 0x0A4 | | CAN0_RXM | ✓ | — | — | — | — | — |
| 0x0A5 | | CAN0_TXM | ✓ | — | — | — | — | — |
| 0x0B1 | PORT | IOPORT_GROUP1 | ✓ | ✓*1 | ✓*1 | — | — | — |
| 0x0B2 | | IOPORT_GROUP2 | ✓ | ✓*1 | ✓*1 | — | — | — |
| 0x0B3 | | IOPORT_GROUP3 | ✓ | ✓*1 | ✓*1 | — | — | — |
| 0x0B4 | | IOPORT_GROUP4 | ✓ | ✓*1 | ✓*1 | — | — | — |
| 0x0B5 | ELC | ELC_SWEVT0 | ✓*2 | ✓ | — | — | — | — |
| 0x0B6 | | ELC_SWEVT1 | ✓*2 | ✓ | — | — | — | — |
| 0x0B7 | POEG | POEG_GROUPA | ✓ | — | — | — | — | — |
| 0x0B8 | | POEG_GROUPB | ✓ | — | — | — | — | — |
| 0x0B9 | | POEG_GROUPC | ✓ | — | — | — | — | — |
| 0x0BA | | POEG_GROUPD | ✓ | — | — | — | — | — |

**Table 13.4 Event table (4 of 7)**

| Event number | Interrupt request source | Name | IELSRn | | DELSRn | Canceling Snooze | Canceling Software Standby | Canceling Deep Software Standby |
|---|---|---|---|---|---|---|---|---|
| | | | Connect to NVIC | Invoke DTC | Invoke DMAC | | | |
| 0x0C0 | GPT0 | GPT0_CCMPA | ✓ | ✓ | ✓ | — | — | — |
| 0x0C1 | | GPT0_CCMPB | ✓ | ✓ | ✓ | — | — | — |
| 0x0C2 | | GPT0_CMPC | ✓ | ✓ | ✓ | — | — | — |
| 0x0C3 | | GPT0_CMPD | ✓ | ✓ | ✓ | — | — | — |
| 0x0C4 | | GPT0_CMPE | ✓ | ✓ | ✓ | — | — | — |
| 0x0C5 | | GPT0_CMPF | ✓ | ✓ | ✓ | — | — | — |
| 0x0C6 | | GPT0_OVF | ✓ | ✓ | ✓ | — | — | — |
| 0x0C7 | | GPT0_UDF | ✓ | ✓ | ✓ | — | — | — |
| 0x0C8 | | GPT0_PC | ✓ | ✓ | ✓ | — | — | — |
| 0x0C9 | GPT1 | GPT1_CCMPA | ✓ | ✓ | ✓ | — | — | — |
| 0x0CA | | GPT1_CCMPB | ✓ | ✓ | ✓ | — | — | — |
| 0x0CB | | GPT1_CMPC | ✓ | ✓ | ✓ | — | — | — |
| 0x0CC | | GPT1_CMPD | ✓ | ✓ | ✓ | — | — | — |
| 0x0CD | | GPT1_CMPE | ✓ | ✓ | ✓ | — | — | — |
| 0x0CE | | GPT1_CMPF | ✓ | ✓ | ✓ | — | — | — |
| 0x0CF | | GPT1_OVF | ✓ | ✓ | ✓ | — | — | — |
| 0x0D0 | | GPT1_UDF | ✓ | ✓ | ✓ | — | — | — |
| 0x0D1 | | GPT1_PC | ✓ | ✓ | ✓ | — | — | — |
| 0x0D2 | GPT2 | GPT2_CCMPA | ✓ | ✓ | ✓ | — | — | — |
| 0x0D3 | | GPT2_CCMPB | ✓ | ✓ | ✓ | — | — | — |
| 0x0D4 | | GPT2_CMPC | ✓ | ✓ | ✓ | — | — | — |
| 0x0D5 | | GPT2_CMPD | ✓ | ✓ | ✓ | — | — | — |
| 0x0D6 | | GPT2_CMPE | ✓ | ✓ | ✓ | — | — | — |
| 0x0D7 | | GPT2_CMPF | ✓ | ✓ | ✓ | — | — | — |
| 0x0D8 | | GPT2_OVF | ✓ | ✓ | ✓ | — | — | — |
| 0x0D9 | | GPT2_UDF | ✓ | ✓ | ✓ | — | — | — |
| 0x0DB | GPT3 | GPT3_CCMPA | ✓ | ✓ | ✓ | — | — | — |
| 0x0DC | | GPT3_CCMPB | ✓ | ✓ | ✓ | — | — | — |
| 0x0DD | | GPT3_CMPC | ✓ | ✓ | ✓ | — | — | — |
| 0x0DE | | GPT3_CMPD | ✓ | ✓ | ✓ | — | — | — |
| 0x0DF | | GPT3_CMPE | ✓ | ✓ | ✓ | — | — | — |
| 0x0E0 | | GPT3_CMPF | ✓ | ✓ | ✓ | — | — | — |
| 0x0E1 | | GPT3_OVF | ✓ | ✓ | ✓ | — | — | — |
| 0x0E2 | | GPT3_UDF | ✓ | ✓ | ✓ | — | — | — |

**Table 13.4 Event table (5 of 7)**

| Event number | Interrupt request source | Name | IELSRn | | DELSRn | Canceling Snooze | Canceling Software Standby | Canceling Deep Software Standby |
|---|---|---|---|---|---|---|---|---|
| | | | Connect to NVIC | Invoke DTC | Invoke DMAC | | | |
| 0x0E4 | GPT4 | GPT4_CCMPA | ✓ | ✓ | ✓ | — | — | — |
| 0x0E5 | | GPT4_CCMPB | ✓ | ✓ | ✓ | — | — | — |
| 0x0E6 | | GPT4_CMPC | ✓ | ✓ | ✓ | — | — | — |
| 0x0E7 | | GPT4_CMPD | ✓ | ✓ | ✓ | — | — | — |
| 0x0E8 | | GPT4_CMPE | ✓ | ✓ | ✓ | — | — | — |
| 0x0E9 | | GPT4_CMPF | ✓ | ✓ | ✓ | — | — | — |
| 0x0EA | | GPT4_OVF | ✓ | ✓ | ✓ | — | — | — |
| 0x0EB | | GPT4_UDF | ✓ | ✓ | ✓ | — | — | — |
| 0x0EC | | GPT4_PC | ✓ | ✓ | ✓ | — | — | — |
| 0x0ED | GPT5 | GPT5_CCMPA | ✓ | ✓ | ✓ | — | — | — |
| 0x0EE | | GPT5_CCMPB | ✓ | ✓ | ✓ | — | — | — |
| 0x0EF | | GPT5_CMPC | ✓ | ✓ | ✓ | — | — | — |
| 0x0F0 | | GPT5_CMPD | ✓ | ✓ | ✓ | — | — | — |
| 0x0F1 | | GPT5_CMPE | ✓ | ✓ | ✓ | — | — | — |
| 0x0F2 | | GPT5_CMPF | ✓ | ✓ | ✓ | — | — | — |
| 0x0F3 | | GPT5_OVF | ✓ | ✓ | ✓ | — | — | — |
| 0x0F4 | | GPT5_UDF | ✓ | ✓ | ✓ | — | — | — |
| 0x0F5 | | GPT5_PC | ✓ | ✓ | ✓ | — | — | — |
| 0x0F6 | GPT6 | GPT6_CCMPA | ✓ | ✓ | ✓ | — | — | — |
| 0x0F7 | | GPT6_CCMPB | ✓ | ✓ | ✓ | — | — | — |
| 0x0F8 | | GPT6_CMPC | ✓ | ✓ | ✓ | — | — | — |
| 0x0F9 | | GPT6_CMPD | ✓ | ✓ | ✓ | — | — | — |
| 0x0FA | | GPT6_CMPE | ✓ | ✓ | ✓ | — | — | — |
| 0x0FB | | GPT6_CMPF | ✓ | ✓ | ✓ | — | — | — |
| 0x0FC | | GPT6_OVF | ✓ | ✓ | ✓ | — | — | — |
| 0x0FD | | GPT6_UDF | ✓ | ✓ | ✓ | — | — | — |
| 0x0FE | | GPT6_PC | ✓ | ✓ | ✓ | — | — | — |
| 0x0FF | GPT7 | GPT7_CCMPA | ✓ | ✓ | ✓ | — | — | — |
| 0x100 | | GPT7_CCMPB | ✓ | ✓ | ✓ | — | — | — |
| 0x101 | | GPT7_CMPC | ✓ | ✓ | ✓ | — | — | — |
| 0x102 | | GPT7_CMPD | ✓ | ✓ | ✓ | — | — | — |
| 0x103 | | GPT7_CMPE | ✓ | ✓ | ✓ | — | — | — |
| 0x104 | | GPT7_CMPF | ✓ | ✓ | ✓ | — | — | — |
| 0x105 | | GPT7_OVF | ✓ | ✓ | ✓ | — | — | — |
| 0x106 | | GPT7_UDF | ✓ | ✓ | ✓ | — | — | — |
| 0x150 | GPT | GPT_UVWEDGE | ✓ | — | — | — | — | — |

**Table 13.4 Event table (6 of 7)**

| Event number | Interrupt request source | Name | IELSRn | | DELSRn | Canceling Snooze | Canceling Software Standby | Canceling Deep Software Standby |
|---|---|---|---|---|---|---|---|---|
| | | | Connect to NVIC | Invoke DTC | Invoke DMAC | | | |
| 0x160 | ADC120 | ADC120_ADI | ✓ | ✓ | ✓ | — | — | — |
| 0x161 | | ADC120_GBADI | ✓ | ✓ | ✓ | — | — | — |
| 0x162 | | ADC120_CMPAI | ✓ | — | — | — | — | — |
| 0x163 | | ADC120_CMPBI | ✓ | — | — | — | — | — |
| 0x164 | | ADC120_WCMPM | — | ✓ | ✓ | ✓*3 | — | — |
| 0x165 | | ADC120_WCMPUM | — | ✓ | ✓ | ✓*3 | — | — |
| 0x180 | SCI0 | SCI0_RXI | ✓ | ✓ | ✓ | — | — | — |
| 0x181 | | SCI0_TXI | ✓ | ✓ | ✓ | — | — | — |
| 0x182 | | SCI0_TEI | ✓ | — | — | — | — | — |
| 0x183 | | SCI0_ERI | ✓ | — | — | — | — | — |
| 0x184 | | SCI0_AM | ✓ | — | — | ✓*3 | — | — |
| 0x185 | | SCI0_RXI_OR_ERI | — | — | — | ✓*3 | — | — |
| 0x186 | SCI1 | SCI1_RXI | ✓ | ✓ | ✓ | — | — | — |
| 0x187 | | SCI1_TXI | ✓ | ✓ | ✓ | — | — | — |
| 0x188 | | SCI1_TEI | ✓ | — | — | — | — | — |
| 0x189 | | SCI1_ERI | ✓ | — | — | — | — | — |
| 0x18C | SCI2 | SCI2_RXI | ✓ | ✓ | ✓ | — | — | — |
| 0x18D | | SCI2_TXI | ✓ | ✓ | ✓ | — | — | — |
| 0x18E | | SCI2_TEI | ✓ | — | — | — | — | — |
| 0x18F | | SCI2_ERI | ✓ | — | — | — | — | — |
| 0x192 | SCI3 | SCI3_RXI | ✓ | ✓ | ✓ | — | — | — |
| 0x193 | | SCI3_TXI | ✓ | ✓ | ✓ | — | — | — |
| 0x194 | | SCI3_TEI | ✓ | — | — | — | — | — |
| 0x195 | | SCI3_ERI | ✓ | — | — | — | — | — |
| 0x196 | | SCI3_AM | ✓ | — | — | — | — | — |
| 0x198 | SCI4 | SCI4_RXI | ✓ | ✓ | ✓ | — | — | — |
| 0x199 | | SCI4_TXI | ✓ | ✓ | ✓ | — | — | — |
| 0x19A | | SCI4_TEI | ✓ | — | — | — | — | — |
| 0x19B | | SCI4_ERI | ✓ | — | — | — | — | — |
| 0x19C | | SCI4_AM | ✓ | — | — | — | — | — |
| 0x1B6 | SCI9 | SCI9_RXI | ✓ | ✓ | ✓ | — | — | — |
| 0x1B7 | | SCI9_TXI | ✓ | ✓ | ✓ | — | — | — |
| 0x1B8 | | SCI9_TEI | ✓ | — | — | — | — | — |
| 0x1B9 | | SCI9_ERI | ✓ | — | — | — | — | — |
| 0x1BA | | SCI9_AM | ✓ | — | — | — | — | — |
| 0x1BC | SCIX0*4 | SCIX0_SCIX0 | ✓ | — | — | — | — | — |
| 0x1BD | | SCIX0_SCIX1 | ✓ | — | — | — | — | — |
| 0x1BE | | SCIX0_SCIX2 | ✓ | — | — | — | — | — |
| 0x1BF | | SCIX0_SCIX3 | ✓ | — | — | — | — | — |

**Table 13.4 Event table (7 of 7)**

| Event number | Interrupt request source | Name | IELSRn | | DELSRn | Canceling Snooze | Canceling Software Standby | Canceling Deep Software Standby |
|---|---|---|---|---|---|---|---|---|
| | | | Connect to NVIC | Invoke DTC | Invoke DMAC | | | |
| 0x1C0 | SCIX1*5 | SCIX1_SCIX0 | ✓ | — | — | — | — | — |
| 0x1C1 | | SCIX1_SCIX1 | ✓ | — | — | — | — | — |
| 0x1C2 | | SCIX1_SCIX2 | ✓ | — | — | — | — | — |
| 0x1C3 | | SCIX1_SCIX3 | ✓ | — | — | — | — | — |
| 0x1C4 | SPI0 | SPI0_SPRI | ✓ | ✓ | ✓ | — | — | — |
| 0x1C5 | | SPI0_SPTI | ✓ | ✓ | ✓ | — | — | — |
| 0x1C6 | | SPI0_SPII | ✓ | — | — | — | — | — |
| 0x1C7 | | SPI0_SPEI | ✓ | — | — | — | — | — |
| 0x1C8 | | SPI0_SPCEND | ✓ | — | — | — | — | — |
| 0x1DA | QSPI | QSPI_INTR | ✓ | — | — | — | — | — |
| 0x1DB | DOC | DOC_DOPCI | ✓ | — | — | ✓*3 | — | — |

Note 1. Only the first edge detection is valid.
Note 2. Only interrupts after DTC transfer are supported.
Note 3. Using SELSR0.
Note 4. SCIX0_SCIXn (n = 0 to 3) links to extended serial interface of SCI1.
Note 5. SCIX1_SCIXn (n = 0 to 3) links to extended serial interface of SCI2.

## 13.4 Interrupt Operation

The ICU performs the following functions:

- Detecting interrupts
- Enabling and disabling interrupts
- Selecting interrupt request destinations such as CPU interrupt, DTC activation, or DMAC activation.

### 13.4.1 Detecting Interrupts

The ICU selects an event source input from a peripheral function interrupt or an external pin interrupt with IELSRn.IELS [8:0].

The accepted interrupt source sets the IELSRn.IR to 1 and sends an interrupt request to the NVIC.

External pin interrupt requests are detected by either:

- Edges (falling edge, rising edge, or rising and falling edges)
- Level (low level) of the interrupt signal.

Set the IRQCRi.IRQMD[1:0] bits to select the detection mode for the IRQi pins. For interrupt sources associated with peripheral modules, see Table 13.3 and Table 13.4. Events must be accepted by the NVIC before an interrupt occurs and is accepted by the CPU.

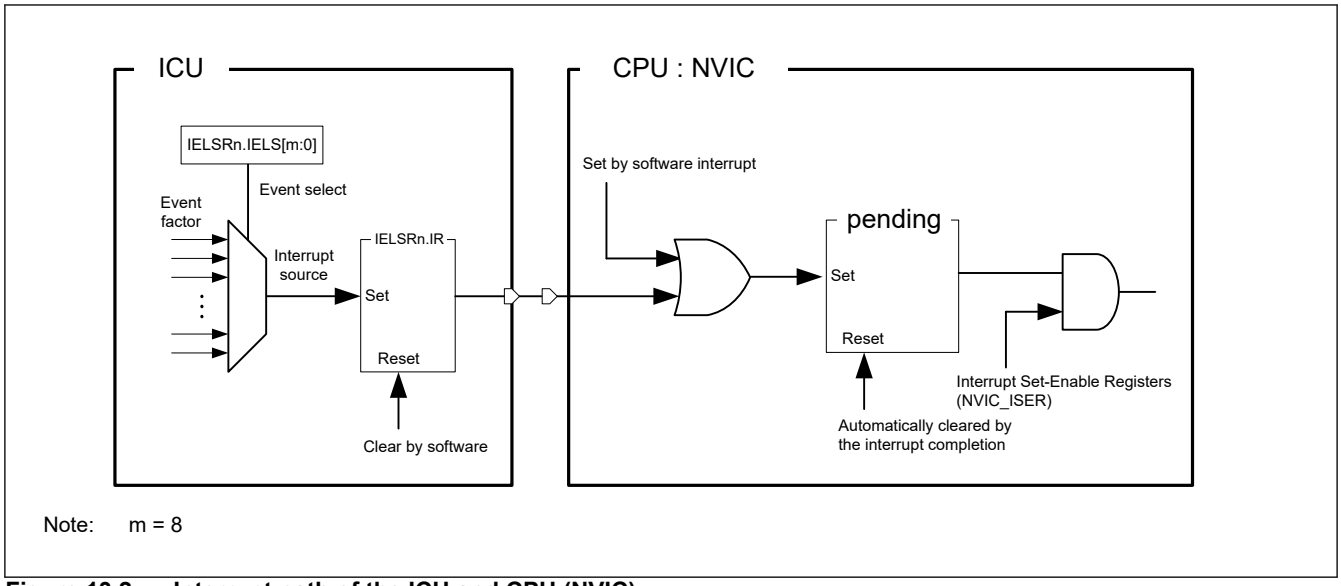



**Figure 13.2 Interrupt path of the ICU and CPU (NVIC)**

## 13.5 Interrupt setting procedure

### 13.5.1 Enabling Interrupt Requests

The procedure for enabling an interrupt request is as follows:

1. Set the Interrupt Set-Enable register (NVIC_ISER).
2. Set the IELSRn.IELS[8:0] bits as the interrupt source.
3. Specify the operation settings for the event source, such as DMAC activation (DELSRn.DELS[8:0]), snooze mode cancellation (SELSR0.SELS[8:0]), software standby mode cancellation (WUPEN register setting).

### 13.5.2 Disabling Interrupt Requests

The procedure to disable the interrupt request is as follows:

1. Disable the operation settings for the event source, such as DMAC activation (DELSRn.DELS[8:0]), snooze mode cancellation (SELSR0.SELS[8:0]), software standby mode cancellation (WUPEN register setting).
2. Clear the interrupt source setting (IELSRn.IELS[8:0] = 0x00).
3. Clear the interrupt status flag (IELSRn.IR = 0).
4. Clear the interrupt Clear-Enable register (NVIC_ICER) and interrupt Clear-Pending register (NVIC_ICPR).

### 13.5.3 Polling for interrupts

The procedure for polling for interrupt requests is as follows:

1. Set the Interrupt Clear-Enable register (NVIC_ICER).
2. Set the IELSRn.IELS[8:0] bits as the interrupt source.
3. Specify the operation settings for the event source, such as DMAC activation (DELSRn.DELS[8:0]), snooze mode cancellation (SELSR0.SELS[8:0]), software standby mode cancellation (WUPEN register setting).
4. Poll the interrupt Set-Pending register (NVIC_ISPR).

### 13.5.4 Selecting Interrupt Request Destinations

The available destinations are fixed for each interrupt, as described in Table 13.3, Table 13.4.

The interrupt output destination, CPU, DMAC, or DTC can be independently selected for each interrupt source.

Use an interrupt request destination setting that is indicated by a "✓" in the event list (see section 13.3.2. Event Number).

Note: Setting the same interrupt source for IELSRn and DELSRn is prohibited.

If the DMAC or DTC is selected as the destination for requests from an IRQi pin, you must set the IRQCRi.IRQMD[1:0] bits for that interrupt to select edge detection.

#### 13.5.4.1 CPU interrupt request

When IELSRn.DTCE = 0, the event specified in the IELSRn register is output to the NVIC. Set the IELSRn.IELS[8:0] bits to the target event and set the IELSRn.DTCE bit to 0.

#### 13.5.4.2 DTC activation

When IELSRn.DTCE = 1, the event specified in the IELSRn register is output to the DTC. Use the following procedure:

1. Set the IELSRn.IELS[8:0] bits to the target event and set the IELSRn.DTCE bit to 1.
2. Set the DTC Module Start bit (DTCST.DTCST) to 1.

Table 13.5 shows operation when the DTC is the interrupt request destination.

**Table 13.5 Operation when DTC becomes interrupt request destination**

| Interrupt request destination | DISEL*1 | Remaining transfer operations | Operation per request | IR*2 | Interrupt request destination after transfer |
|---|---|---|---|---|---|
| DTC*3 | 1 | ≠ 0 | DTC transfer → CPU interrupt | Cleared on interrupt acceptance by the CPU | DTC |
| | | = 0 | DTC transfer → CPU interrupt | Cleared on interrupt acceptance by the CPU | CPU (IELSRn.DTCE bit is automatically cleared) |
| | 0 | ≠ 0 | DTC transfer | Cleared at the start of DTC data transfer after reading DTC transfer data | DTC |
| | | = 0 | DTC transfer → CPU interrupt | Cleared on interrupt acceptance by the CPU | CPU (IELSRn.DTCE bit is automatically cleared) |

Note 1. DTC.MRB.DISEL bit controles the interrupt generates timing from DTC to CPU.
Note 2. When the IELSRn.IR flag is 1, an interrupt request (DTC activation request) that occurs again is ignored.
Note 3. For chain transfers, DTC transfer continues until the last chain transfer ends. The DISEL bit state and the remaining transfer count determine whether a CPU interrupt occurs, the IELSRn.IR flag clear timing, and the interrupt request destination after transfer. See Table 17.2 in section 17, Data Transfer Controller (DTC).

Note: Error during DTC transfer

If an error response occurs during DTC transfer, the DTC notifies the ICU that an error has occurred. ICU clears all bits of the target IELSRn (n = 0 to 95). IELSRn that is not the target is not cleared.

Note: DTC transfer error in snooze mode

When an error occurs in DTC transfer in Snooze mode, ICU issues a Wake Up request. However, interrupt requests are not issued automatically. See section 17, Data Transfer Controller (DTC) chapter for information on how to set the interrupt when a DTC error occurs.

#### 13.5.4.3 DMAC Activation

Events specified in the DELSRn registers are output to the DMAC.

To set the interrupt source for DMAC, use the following procedure:

1. Set the DELSRn.DELS[8:0] bits to the event to activate the DMAC.
2. When using interrupts to CPU, set the IELSRn.IELS bit to factor of DMAC interrupt and IELSRn.DTCE bit to 0.
3. Set the activation source for the target DMAC channel (DMACm.DMTMD.DCTG[1:0]) to 01b (interrupt module detection).
4. Set the DMAC transfer enable bit for the target DMAC channel (DMACm.DMCNT.DTE) to 1.
5. Set the DMAC operation enable bit (DMAST.DMST) to 1.

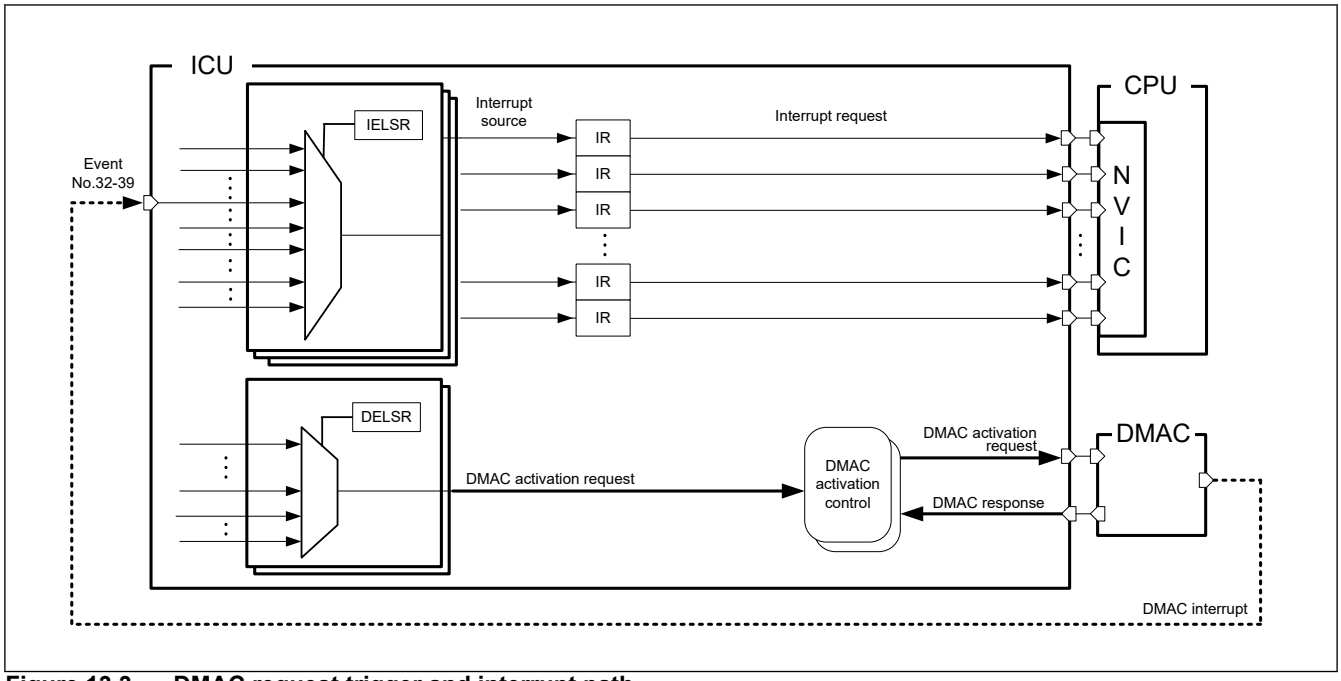



**Figure 13.3 DMAC request trigger and interrupt path**

Note: Error during DMAC transfer

If an error response occurs during DMAC transfer, the DMAC notifies the ICU that an error has occurred.

The ICU clears all bits of the target channel of DELSRn (n = 0 to 7). DELSRn that is not the target channel is not cleared.

### 13.5.5 Digital Filter

A digital filter function is provided for the external interrupt request pins IRQi, (i = 0 to 15) and the NMI pin interrupt. It samples input signals on the filter PCLKB sampling clock and removes any signal with a pulse width less than 3 sampling cycles.

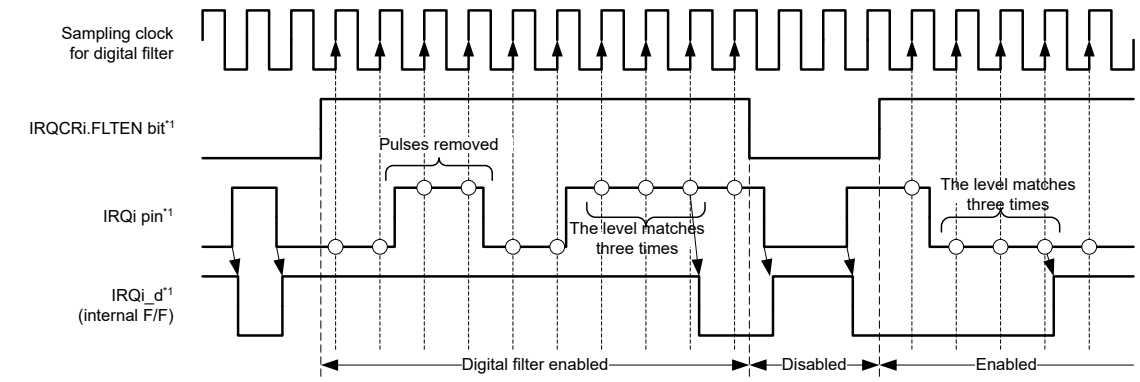
To use the digital filter for an IRQi pin:

1. Set the sampling clock cycle to PCLKB, PCLKB/8, PCLKB/32, or PCLKB/64 in the IRQCRi.FCLKSEL[1:0] bits (i = 0 to 15).
2. Set the IRQCRi.FLTEN bit (i = 0 to 15) to 1 (digital filter enabled).

To use the digital filter for an NMI pin:

1. Set the sampling clock cycle to PCLKB, PCLKB/8, PCLKB/32, or PCLKB/64 in the NMICR.NFCLKSEL[1:0] bits.
2. Set the NMICR.NFLTEN bit to 1 (digital filter enabled).

Figure 13.4 shows an example of digital filter operation.




Note 1. i = 0 to 7

**Figure 13.4 Digital filter operation example**

Before entering Software Standby mode, disable the digital filters by clearing the IRQCRi.FLTEN and NMICR.NFLTEN bits. The ICU clock stops in Software Standby mode.

On exiting Software Standby mode, the circuit detects the edge by comparing the state before standby to the state after standby release. If the input changes during Software Standby mode, an incorrect edge might be detected. The digital filters can be enabled again after exiting Software Standby mode.

### 13.5.6 External Pin Interrupts

To use external pin interrupts:

1. Configure I/O ports settings.
2. Clear the IRQCRi.FLTEN bit (i = 0 to 15) to 0 (digital filter disabled).
3. Set the IRQMD[1:0] bits of the given IRQCRi register (i = 0 to 15) to select the senses of detection.
4. Set the FCLKSEL[1:0] bits, and the FLTEN bit of the IRQCRi register.
5. Select the IRQ pin as follows:
   - If the IRQ pin is to be used for CPU interrupt requests, set the IELSRn.IELS[8:0] bits and the IELSRn.DTCE bit to 0.
   - If the IRQ pin is to be used for DTC activation, set the IELSRn.IELS[8:0] bits and the IELSRn.DTCE bit to 1.
   - If the IRQ pin is to be used for DMAC activation, set the DELSRn.DELS[8:0] bits.

## 13.6 Non-Maskable Interrupt Operation

The following sources can trigger a non-maskable interrupt:

- NMI pin interrupt
- Oscillation stop detection interrupt
- WDT underflow/refresh error interrupt
- IWDT underflow/refresh error interrupt
- Voltage monitor 1 interrupt
- Voltage monitor 2 interrupt
- SRAM parity error interrupt
- SRAM ECC error interrupt

- Bus master MPU error interrupt
- TrustZone filter error interrupt

Non-maskable interrupts can only be used with the CPU, not to activate the DTC or DMAC. Non-maskable interrupts take precedence over all other interrupts. The non-maskable interrupt states can be verified in the Non-Maskable Interrupt Status Register (NMISR). Confirm that all bits in the NMISR are 0 before returning from the NMI handler.

Non-maskable interrupts are disabled by default. To use non-maskable interrupts:

1. Clear the NMICR.NFLTEN bit to 0 (digital filter disabled).
2. Set the NMIMD bit, NFCLKSEL[1:0] bits, and NFLTEN bit of NMICR register.
3. Write 1 to the NMICLR.NMICLR bit to clear the NMISR.NMIST flag to 0.
4. Enable the non-maskable interrupt by writing 1 to the associated bit in the Non-Maskable Interrupt Enable Register (NMIER).

After 1 is written to the NMIER register, subsequent write access to the NMIEN bit in NMIER is ignored. An NMI cannot be disabled when enabled, except by a reset.

The secure attribution managed within the Application Interrupt and Reset Control Register (AIRCR) of the Arm CPU must match the security attribution of NMI.

The NMI secure of the CPU is changed by AIRCR.BFHFNMINS. It is managed by software developers who manage Secure program.

### 13.6.1 Correspondence to TrustZone-M by NMI

Although there is only one NMI per CPU, multiple factors can be set. This section describes the procedure for mixing Secure and NonSecure programs of NMI. When doing so, the NMI-related registers of the ICU are set to Secure.

NMI-related registers:

- NMIER
- NMICLR
- NMICR

Set the ICUSARB.SANMI bit to 0.

The value of AIRCR.BFHFNMINS[13] in the Application Interrupt and Reset Control Register of the ARM CPU must be the same as the value of security attribution. The initial values of AIRCR.BFHFNMINS and ICUSARB.SANMI are different.

AIRCR.BFHFNMINS is for secure and ICUSARB.SANMI is for non-secure. Polarity has the same meaning so program these to match.

If NMI is issued, jump to the NMI handler. When mixing secure and non-secure program, the NMI handler must branch according to the TrustZone-M rule. Figure 13.5 shows the flow.

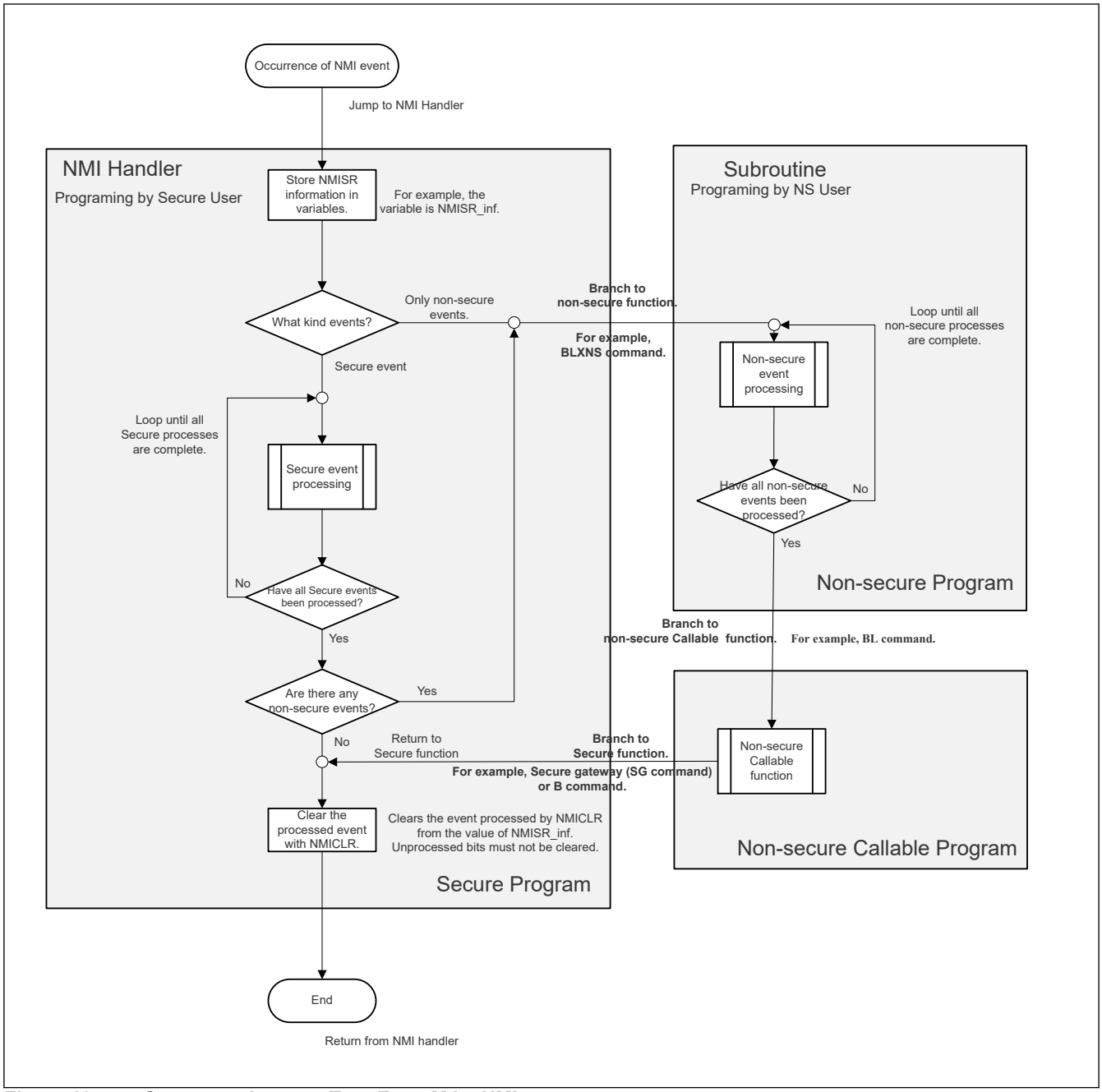



**Figure 13.5 Correspondence to TrustZone-M by NMI**

See the Arm documentation for details on how to move between secure and non-secure programs.

## 13.7 Return from Low Power Modes

Table 13.4 lists the interrupt sources that can be used to exit Sleep, Snooze, or Software Standby mode. For more information, see section 10, Low Power Modes.

### 13.7.1 Return from Sleep Mode

To return from Sleep mode in response to an interrupt:

**non-maskable interrupt**

- Use the NMIER register to enable the target interrupt request.

**maskable interrupt**

- Select the CPU as the interrupt request destination.

- Enable the interrupt in the NVIC.

### 13.7.2 Return from Software Standby Mode

The ICU returns from Software Standby mode using a non-maskable interrupt or a maskable interrupt. For maskable interrupt of canceling source, see Table 13.4.

To return from Software Standby mode:

1. Select the interrupt source that enables return from Software Standby:
   - For non-maskable interrupts, use the NMIER register to enable the target interrupt request
   - For maskable interrupts, use the WUPEN register to enable the target interrupt request.
2. Select the CPU as the interrupt request destination
3. Enable the interrupt in the NVIC.

Interrupt requests through the IRQn pins that do not satisfy these conditions are not detected while the clock is stopped in Software Standby mode.

Similarly, request for a non-maskable interrupt from a request source whose clock is stopped in Software Standby mode cannot be detected.

#### Transition to/from Software Standby mode

1. Before Software Standby mode is entered, disable the digital filter for the interrupt source as a return target (IRQCRi.FLTEN = 0, NMICR.NFLTEN = 0).
2. To use the digital filter again after returning from Software Standby mode, enable the digital filter (IRQCRi.FLTEN = 1, NMICR.NFLTEN = 1).

### 13.7.3 Return from Snooze Mode

The ICU can return to Normal mode from Snooze mode using the interrupts provided for this mode.

To return to Normal mode from Snooze mode:

1. Set the number of the required interrupt request in SELSR0.SELS[8:0]
2. Set the value 0x02D (ICU_SNZCANCEL) in IELSRn.IELS[8:0] (n = 0 to 95).
3. Select the CPU as the interrupt request destination.
4. Enable the interrupt in the NVIC.

Interrupt requests through the non-maskable that do not satisfy the above conditions are not detected while the clock is stopped in snooze mode.

Note: In Snooze mode, a clock is supplied to the ICU. If an event selected in IELSRn is detected, the CPU acknowledges the interrupt after returning to Normal mode from Software Standby mode. If an event selected in DELSRn is detected, the DMAC can acknowledge the interrupt after returning to Normal mode from Software Standby mode.

## 13.8 Using the WFI Instruction with Non-Maskable Interrupts

Whenever a WFI instruction is executed, confirm that all status flags in the NMISR register are 0.

## 13.9 Reference

- ARM Limited., ARM® Cortex®-M33 Processor Technical Reference Manual (ARM 100230)

# 14. Buses

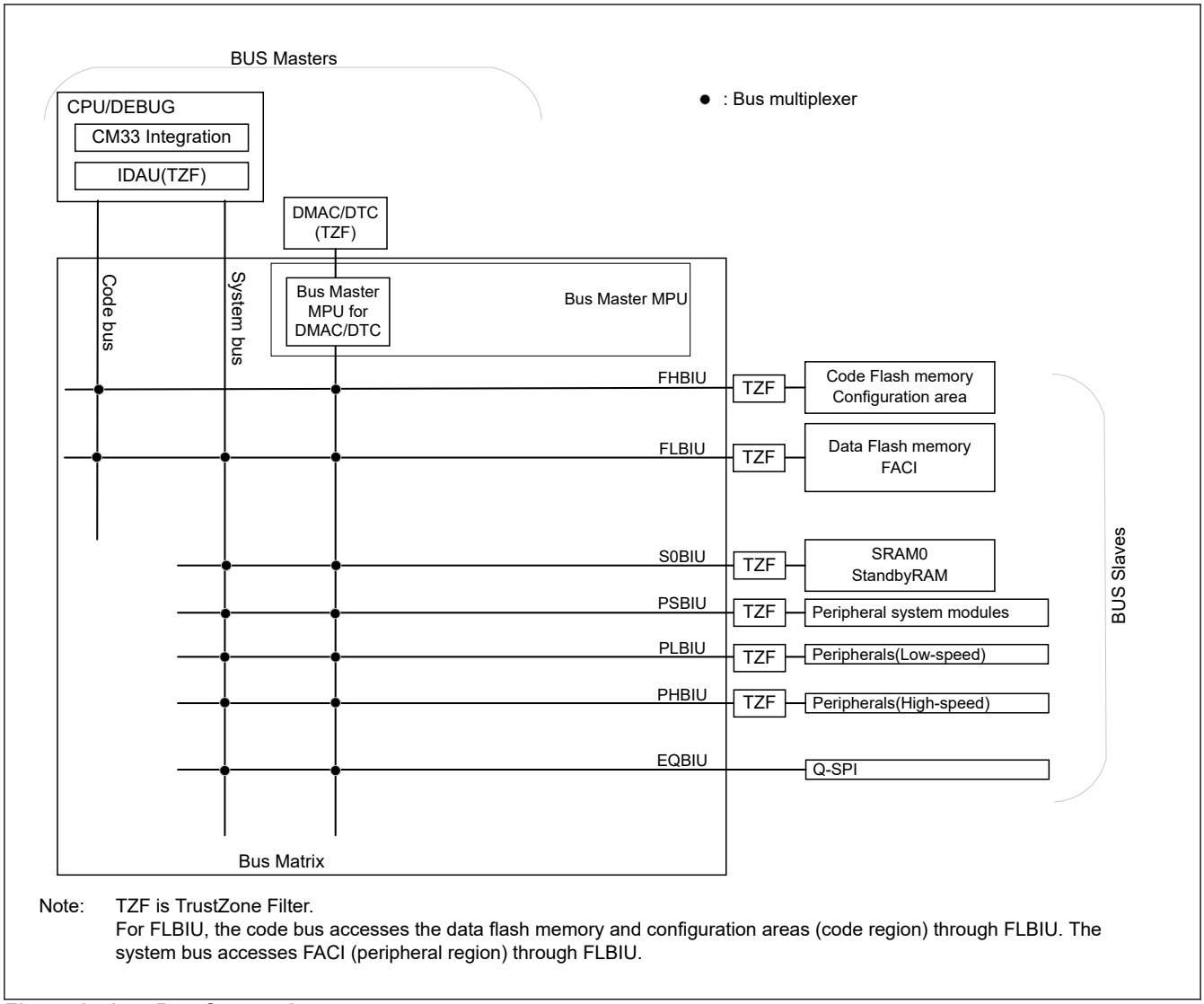
## 14.1 Overview

The buses consists of 32 bits AHB bus matrix. Table 14.1 lists the bus masters and bus slaves and Figure 14.1 shows the bus configuration.

**Table 14.1 Bus Specifications**

| Classification | Bus Master/Slave name | Bus I/F Max Freq | Sync Clock | Specifications |
|---|---|---|---|---|
| Bus Masters | Code bus (Cortex-M33) | 100 MHz | ICLK | Connected to the CPU for instructions and operands |
| | System bus (Cortex-M33) | 100 MHz | ICLK | Connected to the CPU for system |
| | DMAC / DTC | 100 MHz | ICLK | Connected to the DMAC/DTC |
| Bus Slaves | FHBIU | 100 MHz | ICLK | Connected to Code Flash memory and Configuration area |
| | FLBIU | 50 MHz | FCLK | Connected to Data Flash memory, FACI |
| | S0BIU | 100 MHz | ICLK | Connected to SRAM0 (Standby RAM) |
| | PSBIU | 100 MHz | ICLK | Connected to peripheral system modules (DTC, DMAC, ICU, Flash, MPU, SRAM, Debug/Trace module, System controller and BUS controller) |
| | PLBIU | 50 MHz | PCLKB | Connected to peripheral modules (CAC, ELC, I/O ports, POEG, RTC, WDT, IWDT, AGT, IIC, CAN, USBFS, SDHI, SSIE, TSN, and CTSU) |
| | PHBIU | 100 MHz | PCLKA | Connected to peripheral modules (GPT, SCI, SPI, CRC, DOC, ADC12, DAC12 and SCE9) |
| | EQBIU (QSPI area) | 100 MHz | PCLKA | Connected to the QSPI (External Memory Interface) |

Note: FHBIU: Flash High speed Bus Interface Unit.
FLBIU: Flash Low speed Bus Interface Unit.
S0BIU: SRAM0 Bus Interface Unit.
PSBIU: Peripheral System Bus Interface Unit.
PLBIU: Peripheral Low speed Bus Interface Unit.
PHBIU: Peripheral High speed Bus Interface Unit.
EQBIU: External memory interface Qspi Bus Interface Unit.




Note: TZF is TrustZone Filter.
For FLBIU, the code bus accesses the data flash memory and configuration areas (code region) through FLBIU. The system bus accesses FACI (peripheral region) through FLBIU.

**Figure 14.1 Bus Connection**

## 14.2 Description of Buses

### 14.2.1 Arbitration

For arbitration between masters in each slave, fixed-priority and round-robin methods can be selected for each master. For details, see section 14.3.3. BUSSCNT<slave> : Slave Bus Control Register (<slave> = FHBIU, FLBIU, S0BIU, EQBIU), section 14.3.4. BUSSCNT<slave> : Slave Bus Control Register (<slave> = PSBIU, PLBIU, PHBIU).

### 14.2.2 Parallel Operation

Parallel operation is possible when different bus-master modules are requesting access to different slave modules. For example, if the CPU is fetching an instruction from Code Flash and an operand from SRAM0, the DMAC can handle transfer between a peripheral module at the same time.

Figure 14.2 shows an example of parallel operations. In this example, the CPU uses code bus and system bus for simultaneous access to FHBIU and S0BIU, respectively. Furthermore, the DMAC/DTC simultaneously accesses the peripheral bus during access to FHBIU and S0BIU by the CPU.

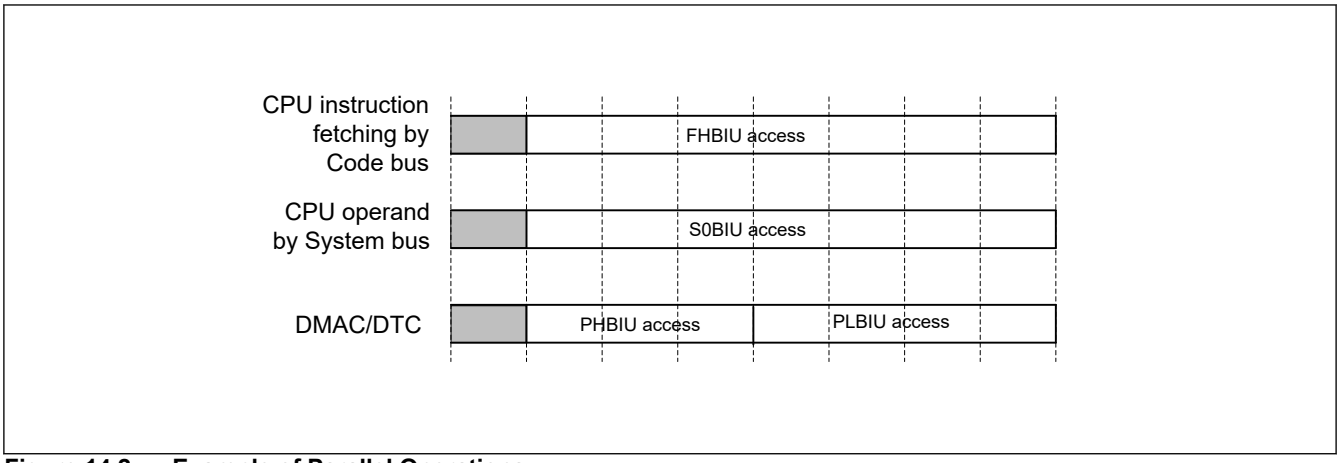



**Figure 14.2 Example of Parallel Operations**

### 14.2.3 Restrictions

#### (1) Restriction on Endian

Memory space must be little-endian in order to execute Cortex code.

#### (2) Bufferable write access

When CPU perform Bufferable Write access to PLBIU or PHBIU, if an STZF error occurs then the error response is invalidated. So there is no error flag will be set and no NMI / RESET request is generated.

When CPU perform Bufferable Write access to PHBIU, if a Slave BUS error occurs then the error response will become invalid and the error flag will not be set.

If error response is required, set the bus master to non-bufferable access.

#### (3) Access to reserved area of FLBIU and S0BIU

Access to the reserved area of FLBIU and S0BIU is prohibited. Operation is not guaranteed if accessed.

#### (4) Clock setting

The clock division ratio prohibits setting changes during bus access.

## 14.3 Register Descriptions

### 14.3.1 BUSSARA : BUS Security Attribution Register A

Base address: CPSCU = 0x4000_8000

Offset address: 0x0100

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | BUSSA0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BUSSA0 | BUS Security Attribution A0<br>0: Secure<br>1: Non-Secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:1 | — | These bits are read as 1. | R |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

### BUSSA0 bit (BUS Security Attribution A0)

The correspondence between register and BIU name is as follows

Connection (BUSSCNT<slave> = FHBIU/FLBIU/S0BIU/PSBIU/PLBIU/PHBIU/EQBIU)

See to Figure 14.1 for connection between BIU and BUS

- BUSSCNTFHBIU
- BUSSCNTFLBIU
- BUSSCNTS0BIU
- BUSSCNTPSBIU
- BUSSCNTPLBIU
- BUSSCNTPHBIU
- BUSSCNTEQBIU

## 14.3.2 BUSSARB : BUS Security Attribution Register B

Base address: CPSCU = 0x4000_8000

Offset address: 0x0104

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | BUSSB0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BUSSB0 | BUS Security Attribution B0<br>0: Secure<br>1: Non-Secure | R/W |
| 31:1 | — | These bits are read as 1. | R |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

### BUSSB0 bit (BUS Security Attribution B0)

The BUSSB0 bit specifies the security attributes of registers for Bus Error Clear registers and DMAC/DTC Error Clear register.

BUS1ERRCLR: Code bus

BUS2ERRCLR: System bus

BUS3ERRCLR: DMAC/DTC

DMACDTCERRCLR: DMAC/DTC (Master-TZF)

See Figure 14.1 for connection of each BUS.

## 14.3.3 BUSSCNT<slave> : Slave Bus Control Register (<slave> = FHBIU, FLBIU, S0BIU, EQBIU)

Base address: BUS = 0x4000_3000

Offset address: 0x1100 (BUSSCNTFHBIU)
0x1104 (BUSSCNTFLBIU)
0x1110 (BUSSCNTS0BIU)
0x1140 (BUSSCNTEQBIU)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | ARBS[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | ARBS[1:0] | Arbitration Select for two masters<br>Specify the priority between bus master.<br>> : Fixed Priority<br>↔: Round-Robin<br>0 0: DMAC/DTC > CPU<br>0 1: DMAC/DTC ↔ CPU<br>1 0: Setting prohibited<br>1 1: Setting prohibited | R/W |
| 15:2 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note:
- BUSSCNT<slave> : <slave> is bus interface unit name for Slave
- The change is prohibited from initial value (0) to reserved bit. Operation when changing is not guaranteed.

### ARBS[1:0] bits (Arbitration Select for two masters)

The ARBS bits sets the arbitration method of each master.

## 14.3.4 BUSSCNT<slave> : Slave Bus Control Register (<slave> = PSBIU, PLBIU, PHBIU)

Base address: BUS = 0x4000_3000

Offset address: 0x1120 (BUSSCNTPSBIU)
0x1130 (BUSSCNTPLBIU)
0x1134 (BUSSCNTPHBIU)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | ARBS |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ARBS | Arbitration Select for two masters<br>Specify the priority between bus master.<br>> : Fixed Priority<br>↔: Round-Robin<br>0: DMAC/DTC > CPU<br>1: DMAC/DTC ↔ CPU | R/W |
| 15:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note:
- BUSSCNT<slave> : <slave> is bus interface unit name for Slave

- The change is prohibited from initial value (0) to reserved bit. Operation when changing is not guaranteed.

### ARBS bit (Arbitration Select for two masters)

The ARBS bits sets the arbitration method of each master.

## 14.3.5 BUSnERRADD : BUS Error Address Register (n = 1 to 3)

Base address: BUS = 0x4000_3000

Offset address: 0x1800 + 0x10 × (n - 1)

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BERAD[31:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | BERAD[31:0] | Bus Error Address<br>When a bus error occurs, these bits store the error address | R |

This register is cleared by reset other than MPU- and TZF-related resets which are Bus Master MPU Error Reset and TrustZone Filter Error Reset.

For detail of MPU- and TZF-related resets, see section 5, Resets, section 15, Memory Protection Unit (MPU) and section 46.2. Arm TrustZone Security .

The following bus errors correspond to the master bus:

BUS1ERRADD : Code bus

BUS2ERRADD : System bus

BUS3ERRADD : DMAC/DTC

### BERAD[31:0] bits (Bus Error Address)

The BERAD[31:0] bits indicate the address when an error occurs on the associated bus. For detail of error that occurs by bus, see section 14.3.9. BUSnERRSTAT : BUS Error Status Register n (n = 1 to 3) and section 14.4. Bus Error Monitoring Section.

When an error occurs on the bus, the corresponding bit of ILERRSTAT, MMERRSTAT, SLERRSTAT in BUSnERRSTAT (n = 1 to 3) is set to 1, at the same time, the BERAD[31:0] bits store the address of the bus error access.

BERAD[31:0] bits are only valid when ILERRSTAT, MMERRSTAT, and SLERRSTAT in BUSnERRSTAT (n = 1 to 3) are set to 1.

## 14.3.6 BUSnERRRW : BUS Error Read Write Register (n = 1 to 3)

Base address: BUS = 0x4000_3000

Offset address: 0x1804 + 0x10 × (n - 1)

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | RWSTAT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RWSTAT | Error Access Read/Write Status<br>The status at the time of the error<br>0: Read access<br>1: Write access | R |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

This register is cleared by reset other than MPU- and TZF-related resets which are Bus Master MPU Error Reset and TrustZone Filter Error Reset.

For detail of MPU related reset, see section 5, Resets and section 15, Memory Protection Unit (MPU).

The following bus errors correspond to the master bus:

BUS1ERRRW : Code bus

BUS2ERRRW : System bus

BUS3ERRRW : DMAC/DTC

#### RWSTAT bit (Error Access Read/Write Status)

The RWSTAT bit indicates the access status, (write access or read access) when an error occurs on the associated bus. For detail of error that occurs by bus, see section 14.3.9. BUSnERRSTAT : BUS Error Status Register n (n = 1 to 3) and section 14.4. Bus Error Monitoring Section.

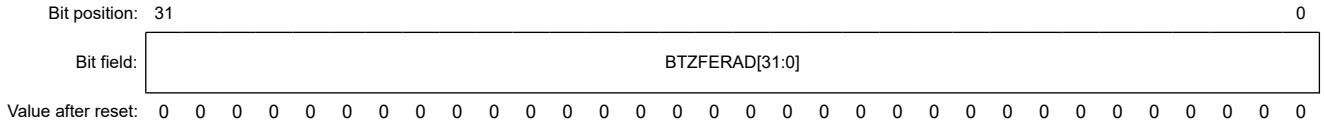
When an error occurs on the bus, the corresponding bit of ILERRSTAT, MMERRSTAT, SLERRSTAT in BUSnERRSTAT (n = 1 to 3) is set to 1, at the same time, the RWSTAT bits store the read/write status of the bus error access.

RWSTAT bit is only valid when ILERRSTAT, MMERRSTAT, and SLERRSTAT in BUSnERRSTAT (n = 1 to 3) are set to 1.

### 14.3.7 BTZFnERRADD : BUS TZF Error Address Register (n = 1 to 3)

Base address: BUS = 0x4000_3000

Offset address: 0x1900 + 0x10 × (n - 1)




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | BTZFERAD[31:0] | Bus TrustZone Filter Error Address<br>When a bus error occurs, these bits store the error address | R |

This register is cleared by reset other than MPU- and TZF-related resets which are Bus Master MPU Error Reset and TrustZone Filter Error Reset.

For detail of MPU- and TZF-related resets, see section 5, Resets, section 15, Memory Protection Unit (MPU) and section 46.2. Arm TrustZone Security .

The following bus errors correspond to the master bus:

BTZF1ERRADD : Code bus

BTZF2ERRADD : System bus

BTZF3ERRADD : DMAC/DTC

See Figure 14.1 for connection of each BUS.

#### BTZFERAD[31:0] bits (Bus TrustZone Filter Error Address)

The BTZFERAD[31:0] bits indicate the address, when an error occurs on the associated bus. For detail of error that occurs by bus, see section 14.3.9. BUSnERRSTAT : BUS Error Status Register n (n = 1 to 3) and section 14.4. Bus Error Monitoring Section.

When an error occurs on the bus, the corresponding bit of STERRSTAT in BUSnERRSTAT (n = 1 to 3) is set to 1, at the same time, the BTZFERAD[31:0] bits store the address of the bus error access.

BTZFERAD[31:0] bits are only valid when STERRSTAT in BUSnERRSTAT (n = 1 to 3) is set to 1.

## 14.3.8 BTZFnERRRW : BUS TZF Error Read Write Register (n = 1 to 3)

Base address: BUS = 0x4000_3000

Offset address: 0x1904 + 0x10 × (n - 1)

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | TRWSTAT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TRWSTAT | TrustZone filter error access Read/Write Status<br>The status at the time of the error<br>0: Read access<br>1: Write access | R |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

This register is cleared by reset other than MPU- and TZF-related resets which are Bus Master MPU Error Reset and TrustZone Filter Error Reset.

For detail of MPU- and TZF-related resets, see section 5, Resets, section 15, Memory Protection Unit (MPU) and section 46.2. Arm TrustZone Security .

The following bus errors correspond to the master bus:

BTZF1ERRRW : Code bus

BTZF2ERRRW : System bus

BTZF3ERRRW : DMAC/DTC

See Figure 14.1 for connection of each BUS.

### TRWSTAT bit (TrustZone filter error access Read/Write Status)

The TRWSTAT bit indicates the access status, (write access or read access), when an error occurs on the associated bus. For detail of error that occurs by bus, see section 14.3.9. BUSnERRSTAT : BUS Error Status Register n (n = 1 to 3) and section 14.4. Bus Error Monitoring Section.

When an error occurs on the bus, the corresponding bit of STERRSTAT in BUSnERRSTAT (n = 1 to 3) is set to 1, at the same time, the TRWSTAT bits store the read/write status of the bus error access. The TRWSTAT bit is only valid when STERRSTAT in BUSnERRSTAT (n = 1 to 3) is set to 1.

## 14.3.9 BUSnERRSTAT : BUS Error Status Register n (n = 1 to 3)

Base address: BUS = 0x4000_3000

Offset address: 0x1A00 + 0x10 × (n - 1)

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | ILERRSTAT | MMERRSTAT | — | STERRSTAT | SLERRSTAT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SLERRSTAT | Slave bus Error Status<br>0: No error occurred<br>1: Error occurred | R |
| 1 | STERRSTAT | Slave TrustZone filter Error Status<br>0: No error occurred<br>1: Error occurred | R |
| 2 | — | This bit is read as 0. | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | MMERRSTAT | Master MPU Error Status<br>0: No error occurred<br>1: Error occurred | R |
| 4 | ILERRSTAT | Illegal address access Error Status<br>0: No error occurred<br>1: Error occurred | R |
| 7:5 | — | These bits are read as 0. | R |

This register is cleared by reset other than MPU- and TZF-related resets which are Bus Master MPU Error Reset and TrustZone Filter Error Reset.

For detail of MPU- and TZF-related resets, see section 5, Resets, section 15, Memory Protection Unit (MPU) and section 46.2. Arm TrustZone Security .

The following bus errors correspond to the master bus:

BUS1ERRSTAT : Code bus

BUS2ERRSTAT : System bus

BUS3ERRSTAT : DMAC/DTC

See Figure 14.1 for connection of each BUS

When an illegal access error, master MPU error, and slave bus error all occurred at the same time, the STAT bit is only valid in the following order of priority. The left side has higher priority.

Master MPU Error > Illegal access error, slave bus error

Note: Illegal access error and slave bus error do not occur at the same time.

If one of ILERRSTAT, MMERRSTAT or SLERRSTAT is set, these bits are not renewed until it is cleared.

#### SLERRSTAT bit (Slave bus Error Status)

When slave error occurs by bus, BUSnERRSTAT.SLERRSTAT (n = 1 to 3) is set to 1. Clear condition is reset or set BUSnERRCLR.SLERRCLR (n = 1 to 3) to 1. Slave error is an error that occurs on a slave such as a timeout. For detail of slave error that occurs by bus, see section 14.4. Bus Error Monitoring Section.

#### STERRSTAT bit (Slave TrustZone filter Error Status)

When slave TrustZone filter error occurs by bus, BUSnERRSTAT.STERRSTAT (n = 1 to 3) is set to 1. Clear condition is reset or set BUSnERRCLR.STERRCLR (n = 1 to 3) to 1. The STERRSTAT bit is not set when the debugger accesses the security area. For detail of slave TrustZone filter error that occurs by bus, see section 46, Security Features .

#### MMERRSTAT bit (Master MPU Error Status)

When master MPU error occurs by bus, BUSnERRSTAT.MMERRSTAT (n = 1 to 3) is set to 1. Clear condition is reset or set BUSnERRCLR.MMERRCLR (n = 1 to 3) to 1. For detail of master MPU error that occurs by bus, see section 15, Memory Protection Unit (MPU).

Note: At master MPU error is occur in DMAC or DTC access, if error address value is not as master MPU area, Illegal address access error or slave error is occurring before DMAC or DTC access. Decide the what error was happened by referring the error address value.

#### ILERRSTAT bit (Illegal address access Error Status)

When illegal address access error occurs by bus, BUSnERRSTAT.ILERRSTAT (n = 1 to 3) is set to 1. Clear condition is reset or set BUSnERRCLR.ILERRCLR (n = 1 to 3) to 1. For detail of illegal address access error that occurs by bus, see section 14.4. Bus Error Monitoring Section.

## 14.3.10 DMACDTCERRSTAT : DMAC/DTC Error Status Register

Base address: BUS = 0x4000_3000

Offset address: 0x1A24

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | MTERRSTAT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MTERRSTAT | Master TrustZone Filter Error Status<br>0: No error occurred<br>1: Error occurred | R |
| 7:1 | — | These bits are read as 0. | R |

This register is cleared by reset other than MPU- and TZF-related resets which are Bus Master MPU Error Reset and TrustZone Filter Error Reset.

For detail of MPU- and TZF-related resets, see section 5, Resets, section 15, Memory Protection Unit (MPU) and section 46.2. Arm TrustZone Security .

### MTERRSTAT bit (Master TrustZone Filter Error Status)

When a master TrustZone filter error occurs by DMAC or DTC, DMACDTCERRSTAT.MTERRSTAT is set to 1. Clear condition is reset or set DMACDTCERRCLR.MTERRCLR to 1.

For detail of master TrustZone filter error that occurs by DMAC or DTC, see section 16, DMA Controller (DMAC) and section 17, Data Transfer Controller (DTC)

## 14.3.11 BUSnERRCLR : BUS Error Clear Register n (n = 1 to 3)

Base address: BUS = 0x4000_3000

Offset address: 0x1A08 + 0x10 × (n - 1)

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | ILERRCLR | MMERRCLR | — | STERRCLR | SLERRCLR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SLERRCLR | Slave bus Error Clear<br>Writing 1 to the SLERRCLR bit clears the BUSnERRSTAT.SLERRSTAT (n = 1 to 3) | R/W*1 |
| 1 | STERRCLR | Slave TrustZone filter Error Clear<br>Writing 1 to the STERRCLR bit clears the BUSnERRSTAT.STERRSTAT (n = 1 to 3) | R/W*1 |
| 2 | — | This bit is read as 0. The write value should be 0. | R/W |
| 3 | MMERRCLR | Master MPU Error Clear<br>Writing 1 to the MMERRCLR bit clears the BUSnERRSTAT.MMERRSTAT (n = 1 to 3) | R/W*1 |
| 4 | ILERRCLR | Illegal Address Access Error Clear<br>Writing 1 to the ILERRCLR bit clears the BUSnERRSTAT.ILERRSTAT (n = 1 to 3) | R/W*1 |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. Only 1 can be written to this bit. This bit is read as 0. Writing 0 to this bit has no effect.

The following bus errors correspond to the master bus:

BUS1ERRCLR : Code bus

BUS2ERRCLR : System bus

BUS3ERRCLR : DMAC/DTC

When writing 1 to BUSnERRCLR (n = 1 to 3), stop the bus access that causes an error in the corresponding bus master.

### 14.3.12 DMACDTCERRCLR : DMAC/DTC Error Clear Register

Base address: BUS = 0x4000_3000

Offset address: 0x1A2C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | MTERRCLR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MTERRCLR | Master TrustZone filter Error Clear<br>Writing 1 to this bit clears the DMACDTCERRSTAT.MTERRSTAT flag. | R/W*1 |
| 7:1 | — | This bit is read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. Only 1 can be written to this bit. This bit is read as 0. Writing 0 to this bit has no effect.

When writing 1 to DMACDTCERRCLR, stop the bus access that causes an error in DMAC/DTC.

## 14.4 Bus Error Monitoring Section

The bus error monitoring system monitors each individual area, and when an error is detected, an error is returned to the requesting master IP using the AHB-Lite error response protocol.

### 14.4.1 Bus Error Types

The following types of errors can occur on each bus:

- Illegal address access
- Bus master MPU error
- TrustZone Filter error
- Bus error transmitted from each slave IP

Table 14.2 lists the address ranges where access leads to illegal address access errors. The reserved area in the slave does not trigger an illegal address access error. For more information on the bus master MPU, see section 15, Memory Protection Unit (MPU).

### 14.4.2 Operations When a Bus Error Occurs

When a bus error occurs, operation is not guaranteed and the error is returned to the requesting master IP.

Figure 14.3 shows operation from each error detection to user notification on the bus.

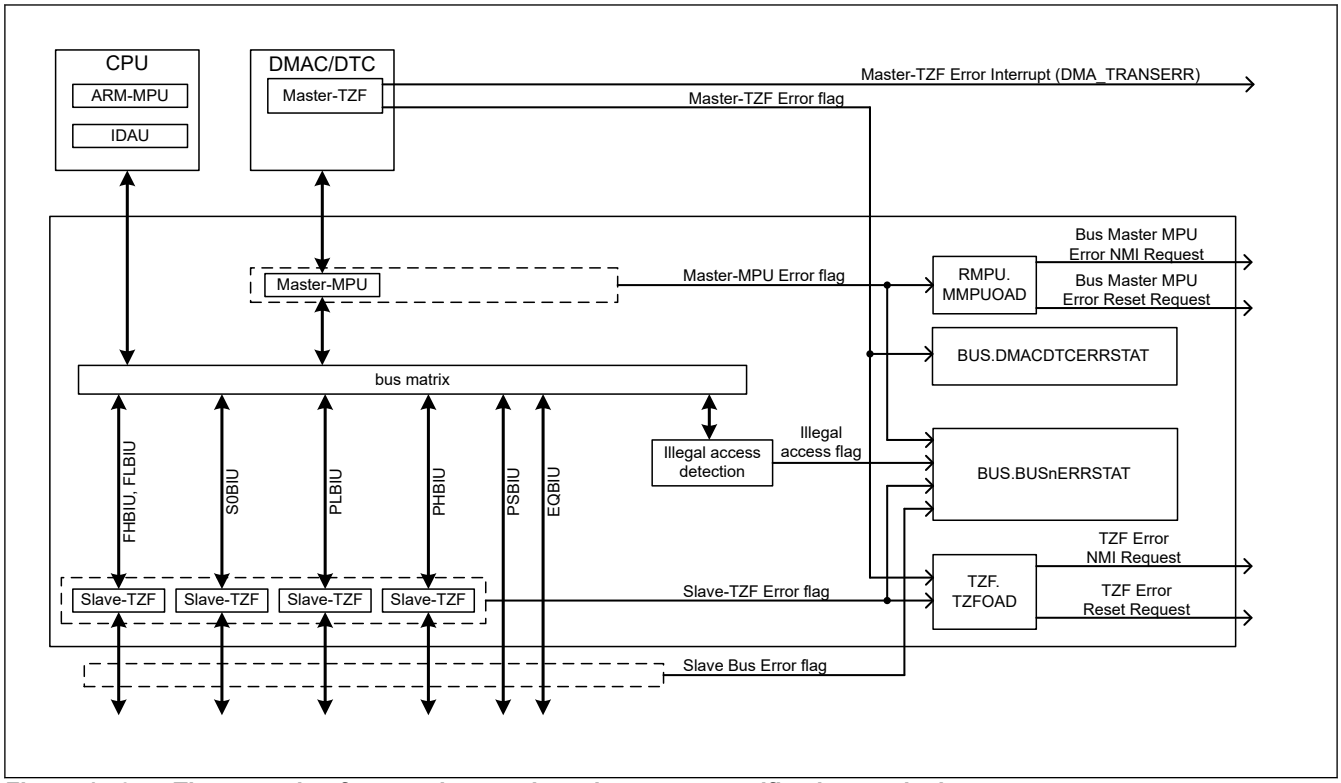



**Figure 14.3 The operation from each error detection to user notification on the bus**

## (1) Bus Master MPU Error

The bus master of DMAC/DTC has a master MPU for access control of the set address area. The CPU does not have a master MPU because it has an Arm MPU. When a bus master MPU error is detected, an Error response is returned to the master. At the same time, perform the following steps:

1. Store the address of the error in BUSnERRADD (n = 3).
2. Store the read/write information of the error in BUSnERRRW (n = 3).
3. Set 1 to MMERRSTAT bit of BUSnERRSTAT (n = 3).

An NMI request or a reset request is generated according to the MMPUOAD.OAD setting (see section 15, Memory Protection Unit (MPU)). Since BUSnERRADD (n = 3), BUSnERRRW (n = 3), and BUSnERRSTAT (n = 3) are held until reset other than MPU- and TZF-related resets or cleared by BUSnERRCLR (n = 3), they can be verified in the NMI handler or after reset.

An NMI request is generated only on the first bus master MPU error after a reset or clearing of BUSnERRSTAT.MMERRSTAT (n = 3) bit by BUSnERRCLR (n = 3).

## (2) Illegal Access Error

section 14.4.3. Conditions Leading to Illegal Address Access Errors, describes illegal access errors. When an illegal access error is detected, an Error response is returned to the master. At the same time, perform the following steps:

1. Store the address of the error in BUSnERRADD (n = 1 to 3).
2. Store the read/write information of the error in BUSnERRRW (n = 1 to 3).
3. Set 1 to ILERRSTAT bit of BUSnERRSTAT (n = 1 to 3).

NMI request and reset request are not generated. Since BUSnERRADD (n = 1 to 3), BUSnERRRW (n = 1 to 3), BUSnERRSTAT (n = 1 to 3) are held until reset other than MPU- and TZF-related resets or cleared by BUSnERRCLR (n = 1 to 3), they can be confirmed in the Bus Fault handler or the interrupt handler.

### (3) Master-TZF Error

As described in section 46, Security Features , DMAC/DTC has Master-TZF errors. When a Master-TZF error is detected, 1 is set to MTERRSTAT bit of DMACDTCERRSTAT, and because the DMAC/DTC does not perform bus access, no bus error information is stored in BTZF3ERRADD and BTZF3ERRRW.

An NMI request or reset request is generated according to the setting of TZFOAD.OAD. See section 16, DMA Controller (DMAC), section 17, Data Transfer Controller (DTC) for details on Master-TZF errors. Because DMACDTCERRSTAT is held until reset other than MPU- and TZF-related resets or cleared by DMACDTCERRCLR, they can be verified in the NMI handler or after reset.

An NMI request is generated only on the first Master-TZF error after reset or clearing of the DMACDTCERRSTAT.MTERRSTAT bit by DMACDTCERRCLR.

### (4) Slave-TZF Error

As described in section 46, Security Features , FHBIU (code flash), FLBIU (data flash), S0BIU (SRAM), PHBIU and PLBIU have Slave-TZF errors. When a Slave-TZF error is detected, perform the following steps:

1. Store the address of the error in BTZFnERRADD (n = 1 to 3).
2. Store the read/write information of the error in BTZFnERRRW (n = 1 to 3).
3. Set 1 to STERRSTAT bit of BUSnERRSTAT (n = 1 to 3).

NMI request or reset request is generated according to the setting in TZFOAD.OAD. Since BTZFnERRADD (n = 1 to 3), BTZFnERRRW (n = 1 to 3), and BUSnERRSTAT (n = 1 to 3) are held until reset other than MPU- and TZF-related resets or cleared by BUSnERRCLR (n = 1 to 3), they can be verified in the NMI handler or after reset.

An NMI request is generated only on the first Slave-TZF error after a reset or clearing of the BUSnERRSTAT.STERRSTAT (n = 1 to 3) bit by BUSnERRCLR (n = 1 to 3).

### (5) Slave Bus Error

Slave Bus Error occurs in the slave. When Slave Bus Error is detected, an Error response is returned to the master. At the same time, perform the following steps:

1. Store the address of the error in BUSnERRADD (n = 1 to 3)
2. Store the read/write information of the error in BUSnERRRW (n = 1 to 3)
3. Set 1 to SLERRSTAT bit of BUSnERRSTAT (n = 1 to 3).

An NMI request and reset request are not generated. Since BUSnERRADD (n = 1 to 3), BUSnERRRW (n = 1 to 3), and BUSnERRSTAT (n = 1 to 3) are held until reset other than MPU- and TZF-related resets or cleared by BUSnERRCLR (n = 1 to 3), they can be verified in the Bus Fault handler or interrupt handler. When a bus slave MPU error occurs, the error is returned to the requesting master IP and operation is not guaranteed.

## 14.4.3 Conditions Leading to Illegal Address Access Errors

Table 14.2 lists the address spaces for each bus that trigger illegal address access errors.

**Table 14.2 Conditions leading to illegal address access errors (1 of 2)**

| Address | Slave bus | Master bus | | |
|---|---|---|---|---|
| | | CPU | | DMA |
| | | Code | System | |
| 0x0000_0000 to 0x01FF_FFFF | FHBIU | — | | — |
| 0x0200_0000 to 0x07FF_FFFF | Reserved | E | | E |
| 0x0800_0000 to 0x0803_FFFF | FLBIU | — | | — |
| 0x0804_0000 to 0x0FFF_FFFF | Reserved | E | | E |
| 0x1000_0000 to 0x100F_FFFF | Reserved | — | | E |
| 0x1010_0000 to 0x1FFF_FFFF | Reserved | E | | E |

**Table 14.2 Conditions leading to illegal address access errors (2 of 2)**

| Address | Slave bus | Master bus | | |
|---|---|---|---|---|
| | | CPU | | DMA |
| | | Code | System | |
| 0x2000_0000 to 0x2800_FFFF | S0BIU | | — | — |
| 0x2801_0000 to 0x3FFF_FFFF | Reserved | | E | E |
| 0x4000_0000 to 0x4007_FFFF | PSBIU | | — | — |
| 0x4008_0000 to 0x400F_FFFF | PLBIU | | — | — |
| 0x4010_0000 to 0x4017_FFFF | PHBIU | | — | — |
| 0x4018_0000 to 0x407D_FFFF | Reserved | | E | E |
| 0x407E_0000 to 0x407F_FFFF | FLBIU | | — | — |
| 0x4080_0000 to 0x5FFF_FFFF | Reserved | | E | E |
| 0x6000_0000 to 0x67FF_FFFF | EQBIU | | — | — |
| 0x6800_0000 to 0x87FF_FFFF | Reserved | | E | E |
| 0x8800_0000 to 0xDFFF_FFFF | Reserved | | E | E |
| 0xE000_0000 to 0xFFFF_FFFF | System for Cortex®-M33 | | | E |

Note: "E" : A bus error occurs.
" " : Transfer does not occur.
"—" : A bus error has not occurred. Even if there has reserved area, a bus error has not occurred.
Do not access reserved area in FLBIU and S0BIU. If accessed, a slave TZF error might occur.

### 14.4.4 Time-out

For some peripheral modules, a timeout error occurs with the module-stop function. When there is no response from the slave for a certain period of time, a timeout error is detected. A timeout error is returned to the requesting master IP using the AHB-Lite error response protocol.

## 14.5 References

1. ARM Limited, *ARM v8-M Architecture Reference Manual* (ARM DDI0553B.g)
2. ARM Limited, *ARM Cortex-M33 Processor Technical Reference Manual Revision:r0p4* (ARM 100230_0004_00_en)
3. ARM Limited, *ARM AMBA 5 AHB Protocol Specification AHB5, AHB-Lite* (ARM IHI 0033B.b)
4. ARM Limited, *ARM AMBA AXI and ACE Protocol Specification AXI3, AXI4, and AXI4-Lite, ACE and ACE-Lite* (ARM IHI 0022D)
5. ARM Limited, *ARM AMBA APB Protocol Specification Version: 2.0* (ARM IHI 0024C)

# 15. Memory Protection Unit (MPU)

## 15.1 Overview

The MCU has one Memory Protection Unit (MPU).

Table 15.1 lists the MPU specifications, and Table 15.2 shows the behavior on detection of each MPU error.

**Table 15.1 MPU specifications**

| Classification | Module/Function | Specifications |
|---|---|---|
| Illegal memory access | Arm® Cortex®-M33 CPU | • Arm CPU has a default memory map. If the CPU makes an illegal access, an exception interrupt occurs<br>• The MPU can change a default memory map. |
| Memory protection | Arm MPU | Memory protection function for the CPU:<br>• (8+8) region MPU with sub regions and background region for secure and non-secure. |
| | Bus master MPU | Memory protection function for each bus master except for the CPU:<br>• DMAC/DTC: 8 regions |

**Table 15.2 Behavior on MPU error detection**

| MPU type | Notification type | Error Response by HRESP signal of AHB I/F | Bus Access on error detection | Storing of error access information |
|---|---|---|---|---|
| Arm MPU | • Hard fault | Not supported | • Does not correctly write access<br>• Does not correctly read access | Stored in the Cortex-M33 processor |
| Bus Master MPU | • Reset or Non-maskable interrupts<br>• Hard fault | Supported | • Write access ignore<br>• Read access is read as 0 | Stored |

For information on error access for the Arm MPU, see section 15.4. References. For information on error access for other MPUs, see section 14.3. Register Descriptions and section 14.4. Bus Error Monitoring Section in section 14, Buses.

## 15.2 Arm MPU

The Arm MPU monitors the addresses accessed by the CPU across the entire address space (0x0000_0000 to 0xFFFF_FFFF) and provides support for:

- (8 + 8) protected regions
- When memory regions overlap, the processor generates a fault if a core access hits the overlapping regions
- Setting access permissions to protected region (Read, Write, Execution)
- Export of memory attributes to the system.

Arm MPU mismatches and permission violations invoke the programmable-priority MemManage fault (Hard Fault) handler. For details, see section 15.4. References.

## 15.3 Bus Master MPU

The bus master MPU monitors the addresses accessed by the bus masters in the entire address space (0x0000_0000 to 0xFFFF_FFFF). Access-control information can be set up to 8 regions in DMAC/DTC and monitor for access to each region is in accord with this information.

If access to a protected region is detected, the bus master MPU generates an internal reset or a non-maskable interrupt. For information on error access, see section 14.3. Register Descriptions and section 14.4. Bus Error Monitoring Section in section 14, Buses.

The access control information for each area consists of protected/not-protected to read or write.

Table 15.3 lists the specifications of the bus master MPU.

**Table 15.3 Bus master MPU specifications**

| Parameter | Description |
|---|---|
| Protected master groups | • DMAC, DTC |
| Protected regions | 0x0000_0000 to 0xFFFF_FFFF |
| Number of regions | • DMAC/DTC: 8 regions |
| Address specification for individual regions | • Specifying start and end address for individual regions |
| Enable or disable setting for memory protection in individual regions | • Enabling or disabling setting for the associated region |
| Access-control settings for individual regions | • Permission for read and write |
| Operation on error detection | • Reset or non-maskable interrupts |
| Register protection | • Protecting registers from illegal writes |
| TrustZone Filter | • DMAC: Security attribution can be set for each regions |

## 15.3.1 Register Descriptions

Bus access must be stopped before writing to MPU registers.

### 15.3.1.1 MMPUSARA : Master Memory Protection Unit Security Attribution Register A

Base address: CPSCU = 0x4000_8000

Offset address: 0x130

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | MMPUASA7 | MMPUASA6 | MMPUASA5 | MMPUASA4 | MMPUASA3 | MMPUASA2 | MMPUASA1 | MMPUASA0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | MMPUASAn | MMPUA Security Attribution (n = 0 to 7)<br>0: Secure<br>1: Non-Secure | R/W |
| 31:8 | — | These bits are read as 1. | R*1 |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

Note 1. This bit is read only.

#### MMPUASAn bits (MMPUA Security Attribution (n = 0 to 7))

The MMPUASAn bits specify the security attributes of registers for the Bus Master MPU Region Setting register. The target registers are:

- MMPUSDMACn (n = 0 to 7)
- MMPUEDMACn (n = 0 to 7)
- MMPUACDMACn (n = 0 to 7)

### 15.3.1.2 MMPUSARB : Master Memory Protection Unit Security Attribution Register B

Base address: CPSCU = 0x4000_8000

Offset address: 0x134

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | MMPUBSA0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MMPUBSA0 | MMPUB Security Attribution<br>0: Secure<br>1: Non-Secure | R/W |
| 31:1 | — | These bits are read as 1. | R*1 |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

Note 1. This bit is read-only.

#### MMPUBSA0 bit (MMPUB Security Attribution)

The MMPUBSA0 bit specifies the security attributes of registers for the Bus Master MPU Region Setting register, Protect register, and OAD register. The target registers are:

- MMPUENDMAC
- MMPUENPTDMAC
- MMPURPTDMAC
- MMPURPTDMAC_SEC
- MMPUOAD
- MMPUOADPT

The Secure user provides a Secure API to Non-secure user for the modification of the MMPURPTDMAC value when MMPUBSA0 bit is set to 0 (Secure).

### 15.3.1.3 MMPUSDMACn : MMPU Start Address Register for DMAC (n = 0 to 7)

Base address: RMPU = 0x4000_0000

Offset address: 0x0204 + 0x010 × n

| Bit position: | 31 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Bit field: | MMPUS[31:5] | — | — | — | — | — | — |
| Value after reset: | x x x x x x x x x x x x x x x x x x x x x x x x x x x | 0 | 0 | 0 | 0 | 0 | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 31:5 | MMPUS[31:5] | Region start address register<br>Address where the region starts, for use in region determination | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The MMPUSDMACn (n = 0 to 7) register specifies the start address where the region starts.

This register requires word access. Byte access and half word access is prohibited. When byte access and half word access is executed, operation is not guaranteed.

Regions set by MMPUSDMACn (n = 0 to 7), MMPUEDMACn (n = 0 to 7) and MMPUACDMACn (n = 0 to 7) registers, can be set for a secure access or a non-secure access with the MMPUSARA register. If the corresponding MMPUSARA.MMPUASAn (n = 0 to 7) bit is set to 1, only non-secure access is permitted for that region. On the other hand, if the corresponding MMPUSARA.MMPUASAn (n = 0 to 7) bit is set to 0, only secure access is permitted for that region.

### 15.3.1.4 MMPUEDMACn : MMPU End Address Register for DMAC (n = 0 to 7)

Base address: RMPU = 0x4000_0000

Offset address: 0x0208 + 0x010 × n

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | 5 | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MMPUE[31:5] | | | | | | | | | | | | | | | | | | | | | | | — | — | — | — | — | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | 1 | 1 | 1 | 1 | 1 | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | — | These bits are read as 1. The write value should be 1. | R/W |
| 31:5 | MMPUE[31:5] | Region end address register<br>Address where the region end, for use in region determination | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The MMPUEDMAC (n = 0 to 7) register specifies the end address where the region ends.

This register requires word access. Byte access and half word access is prohibited. When byte access and half word access is executed, operation is not guaranteed.

### 15.3.1.5 MMPUACDMACn : MMPU Access Control Register for DMAC (n = 0 to 7)

Base address: RMPU = 0x4000_0000

Offset address: 0x0200 + 0x010 × n

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | WP | RP | ENABLE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ENABLE | Region enable<br>0: DMAC Region n unit is disabled<br>1: DMAC Region n unit is enabled | R/W |
| 1 | RP | Read protection<br>0: Read permission<br>1: Read protection | R/W |
| 2 | WP | Write protection<br>0: Write permission<br>1: Write protection | R/W |
| 15:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Region n unit sets the ENABLE bit, the RP bit, and the WP bit each.

### ENABLE bit (Region enable)

The ENABLE bit controls the enable or disable of DMAC/DTC region n (n = 0 to 7) unit.

When the ENABLE bit is set to 1, the RP bit and the WP bit control access permission for read and write protection to MMPUSDMACn (n = 0 to 7) and MMPUEDMACn (n = 0 to 7).

When the ENABLE bit is set to 0, access to DMAC region n (n = 0 to 7) is the outside region.

### RP bit (Read protection)

The RP bit enables or disables read protection of DMAC/DTC region n (n = 0 to 7).

When the ENABLE bit is set to 1, the RP bit is available.

### WP bit (Write protection)

The WP bit enables or disables write protection of DMAC/DTC region n (n = 0 to 7).

When the ENABLE bit is set to 1, the WP bit is available.

**Table 15.4 Function of Region Control Circuit for DMAC**

| MMPUACDMACn (n = 0 to 7) | | | Access | Region | Output of DMAC Region n unit (n = 0 to 7) |
|---|---|---|---|---|---|
| ENABLE | RP | WP | | | |
| 0 | — | — | Read | — | Outside region |
| | | | Write | | Outside region |
| 1 | 0 | 0 | Read | Inside | Permitted region |
| | | | | Outside | Outside region |
| | | | Write | Inside | Permitted region |
| | | | | Outside | Outside region |
| | 0 | 1 | Read | Inside | Permitted region |
| | | | | Outside | Outside region |
| | | | Write | Inside | Protection region |
| | | | | Outside | Outside region |
| | 1 | 0 | Read | Inside | Protection region |
| | | | | Outside | Outside region |
| | | | Write | Inside | Permitted region |
| | | | | Outside | Outside region |
| | 1 | 1 | Read | Inside | Protection region |
| | | | | Outside | Outside region |
| | | | Write | Inside | Protection region |
| | | | | Outside | Outside region |

Note: Each regions of DMAC / DTC are set for secure access and non-secure access by MMPUSARA register. In this case, Non-Secure regions in secure access and secure regions in Non-Secure access are outside regions.

**Table 15.5 Function of Master Control Circuit for DMAC**

| MMPUENDMAC ENABLE | Output of DMAC Region 0 unit | Output of DMAC Region 1 unit | Output of DMAC Region 2-7 unit | Function of DMAC |
|---|---|---|---|---|
| 1 | Protected region | Don't care | Don't care | Generate error |
| | Don't care | Protected region | Don't care | Generate error |
| | Don't care | Don't care | Protected region | Generate error |
| | Outside region | Outside region | Outside region | Generate error |
| Other cases | | | | No error |

A master MPU error occurs on the following conditions:

1. MMPUENDMAC.ENABLE = 1, and output of one or more Region n unit is protected region.
2. MMPUENDMAC.ENABLE = 1, and output of all Region n unit are outside region.

Other cases are handled as permitted region.

### 15.3.1.6 MMPUENDMAC : MMPU Enable Register for DMAC

Base address: RMPU = 0x4000_0000

Offset address: 0x0100

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | ENABLE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ENABLE | Bus Master MPU of DMAC enable<br>0: Bus Master MPU of DMAC is disabled.<br>1: Bus Master MPU of DMAC is enabled. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code<br>These bits enable or disable writes to the ENABLE bit. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: It is necessary to write by half word access.
Byte write access is prohibited. When byte write access is executed, operation is not guaranteed.

#### ENABLE bit (Bus Master MPU of DMAC enable)

The ENABLE bit controls enable or disable of the bus master MPU function of each master group.

When the ENABLE bit is set to 1, MMPUACDMACn (n = 0 to 7) is valid. When the ENABLE bit is set to 0, MMPUACDMACn (n = 0 to 7) is invalid for all regions. The bus master MPU function sets the ENABLE bit of each master group. When the ENABLE bit is set, write 0xA5 in KEY[7:0] at the same time.

#### KEY[7:0] bits (Key Code)

The KEY[7:0] bits enable or disable writing to the ENABLE bit. When writing to the ENABLE bit, write 0xA5 in KEY[7:0] bits at the same time. When values other than 0xA5 are written to KEY[7:0] bits, the ENABLE bit is not updated. The KEY[7:0] bits are always read as 0x00.

### 15.3.1.7 MMPUENPTDMAC : MMPU Enable Protect Register for DMAC

Base address: RMPU = 0x4000_0000

Offset address: 0x0104

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | PROTECT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PROTECT | Protection of register<br>0: MMPUENDMAC register writes are possible.<br>1: MMPUENDMAC register writes are protected. Read is possible. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code<br>These bits enable or disable writes to the PROTECT bit. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: It is necessary to write by half word access.
Byte write access is prohibited. When byte write access is executed, operation is not guaranteed.

#### PROTECT bit (Protection of register)

The PROTECT bit enables or disables writes to the MMPUENDMAC register.

When writing to the PROTECT bit, write 0xA5 in KEY[7:0] at the same time.

#### KEY[7:0] bits (Key Code)

The KEY[7:0] bits enable or disable writes to the PROTECT bit. When writing to the PROTECT bit, write 0xA5 in KEY[7:0] at the same time. When values other than 0xA5 are written in KEY[7:0] bits, the PROTECT bit is not updated. The KEY[7:0] bits are always read as 0x00.

### 15.3.1.8 MMPURPTDMAC : MMPU Regions Protect Register for DMAC

Base address: RMPU = 0x4000_0000

Offset address: 0x0108

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | PROTECT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PROTECT | Protection of register<br>0: Bus Master MPU register for DMAC writing is possible.<br>1: Bus Master MPU register for DMAC writing is protected. Read is possible. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code<br>These bits enable or disable writes to the PROTECT bit. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:

- Secure and Non-secure access are allowed.

Note: It is necessary to write by half word access.
Byte write access is prohibited. When byte write access is executed, operation is not guaranteed.

**PROTECT bit (Protection of register)**

The PROTECT bit enables or disables writes to the associated registers to be protected.

MMPURPTDMAC.PROTECT controls the following registers:

- MMPUSDMACn (n = 0 to 7) of Non-Secure program
- MMPUEDMACn (n = 0 to 7) of Non-Secure program
- MMPUACDMACn (n = 0 to 7) of Non-Secure program

When writing to the PROTECT bit, write 0xA5 simultaneously to the KEY[7:0] bits, using halfword access.

**KEY[7:0] bits (Key Code)**

The KEY[7:0] bits enable or disable writes to the PROTECT bit. When writing to the PROTECT bit, write 0xA5 simultaneously to the KEY[7:0] bits. When other values are written, the PROTECT bit is not updated.

The KEY[7:0] bits always read as 0x00.

### 15.3.1.9 MMPURPTDMAC_SEC : MMPU Regions Protect register for DMAC Secure

Base address: RMPU = 0x4000_0000

Offset address: 0x010C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | PROTECT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PROTECT | Protection of register<br>0: Bus master MPU register for DMAC secure writes are possible.<br>1: Bus master MPU register for DMAC secure writes are protected. Read is possible. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code<br>These bits enable or disable writes to the PROTECT bit. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: It is necessary to write by half word access.
Byte write access is prohibited. When byte write access is executed, operation is not guaranteed.

**PROTECT bit (Protection of register)**

The PROTECT bit enables or disables writes to the associated registers to be protected.

MMPURPTDMAC_SEC.PROTECT controls the following registers:

- MMPUSDMACn (n = 0 to 7) of Secure program
- MMPUEDMACn (n = 0 to 7) of Secure program
- MMPUACDMACn (n = 0 to 7) of Secure program

When writing to the PROTECT bit, write 0xA5 simultaneously to the KEY[7:0] bits, using halfword access.

### KEY[7:0] bits (Key Code)

The KEY[7:0] bits enable or disable writes to the PROTECT bit. When writing to the PROTECT bit, write 0xA5 simultaneously to the KEY[7:0] bits. When other values are written, the PROTECT bit is not updated.

The KEY[7:0] bits are always read as 0x00.

## 15.3.1.10 MMPUOAD : MMPU Operation After Detection Register

Base address: RMPU = 0x4000_0000

Offset address: 0x0000

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | OAD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | OAD | Operation after detection<br>0: Non-maskable interrupt<br>1: Reset | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code<br>This bit enables or disables writes to the OAD bit. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: It is necessary to write by half word access.
Byte write access is prohibited. When byte write access is executed, operation is not guaranteed.

### OAD bit (Operation after detection)

The OAD bit is specified to generate either reset or non-maskable interrupt when the access to the protect region is detected by the BUS Master MPU.

When writing to the OAD bit, write 0xA5 simultaneously to the KEY[7:0] bits using halfword access.

### KEY[7:0] bits (Key Code)

The KEY[7:0] bits enable or disable writing to the OAD bit. When writing to the OAD bit, write 0xA5 simultaneously to the KEY[7:0] bits. When other values are written, the OAD bit is not updated.

The KEY[7:0] bits always read as 0x00.

## 15.3.1.11 MMPUOADPT : MMPU Operation After Detection Protect Register

Base address: RMPU = 0x4000_0000

Offset address: 0x0004

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | PROTECT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PROTECT | Protection of register<br>0: MMPUOAD register writes are possible.<br>1: MMPUOAD register writes are protected. Read is possible. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key code<br>These bits enable or disable writes to the PROTECT bit. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: It is necessary to write by half word access.
Byte write access is prohibited. When byte write access is executed, operation is not guaranteed.

**PROTECT bit (Protection of register)**

The PROTECT bit enables or disables writes to the associated registers to be protected.

MMPUOADPT.PROTECT controls the following register:

- MMPUOAD

When the PROTECT bit is set simultaneously, write 0xA5 to the KEY[7:0] bits using half word access.

**KEY[7:0] bits (Key code)**

The KEY[7:0] bits enable or disable writes to the PROTECT bit. When writing to the PROTECT bit simultaneously, write 0xA5 to the KEY[7:0] bits. When other values are written to the KEY[7:0] bits, the PROTECT bit is not updated. The KEY[7:0] bits are always read as 0x00.

## 15.3.2 Operation

### 15.3.2.1 Memory protection

The bus master MPU monitors memory access using control settings made individually for the access control regions. If access to a protected region is detected, the bus master MPU generates a memory protection error.

Bus Master MPU can be set for up to 8 protection regions. It is protection region when set up of permission region and protection region overlaps. It is protection region when set up of two protection region overlaps.

Bus Master MPU has master groups of DMAC/DTC.

Memory protection checks the address of the bus which the master group unified. Therefore, all the access of a master group is detected by memory protection.

The region setting registers of the Bus Master MPU for DMAC/DTC can be set for secure access and Non-Secure access using the MMPUSARA register. Make secure access and Non-Secure access settings the same for each DMAC/DTC channel and the corresponding region setting registers of the Bus Master MPU.

Bus Master MPU is permission of all regions after reset. All region is protected by setting MMPUENDMAC.ENABLE = 1.

Each region sets up a permission region on the protection region. If access to the protected region is detected, Bus Master MPU will generate an error.

Figure 15.1 shows the use case of a bus master MPU.

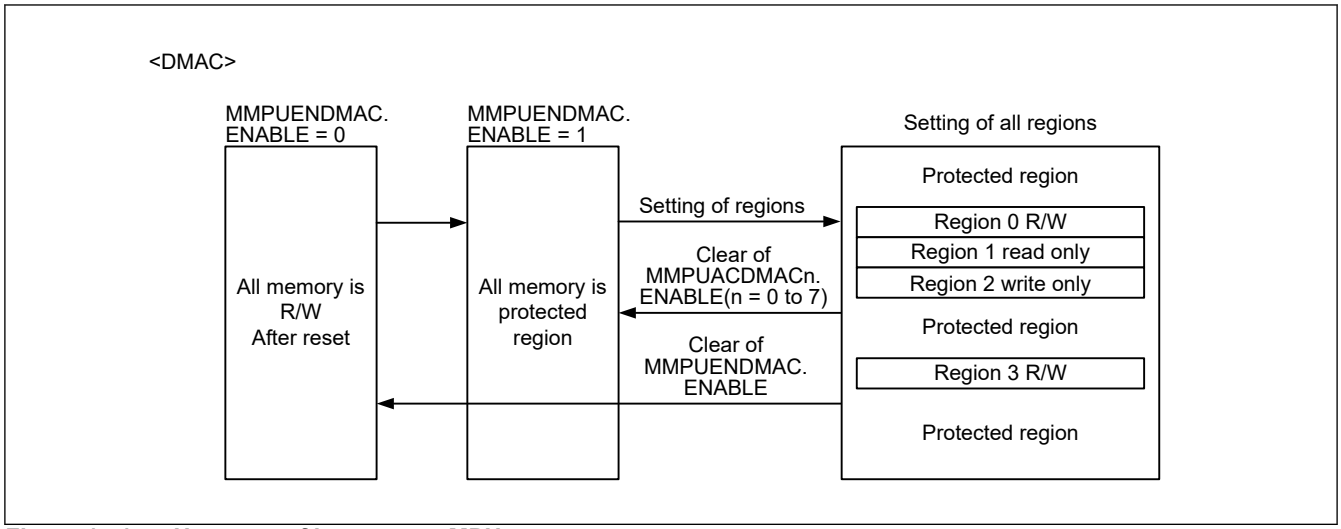



**Figure 15.1 Use case of bus master MPU**

Figure 15.2 shows the access permission or protection by the overlapping bus master MPU regions.

Access control for the overlapping regions is as follows:

- The region is handled as a protected region when output of one or more region units is a protected region
- The region is handled as a protected region when output of all region units is outside of the regions
- Other cases are handled as permitted regions.

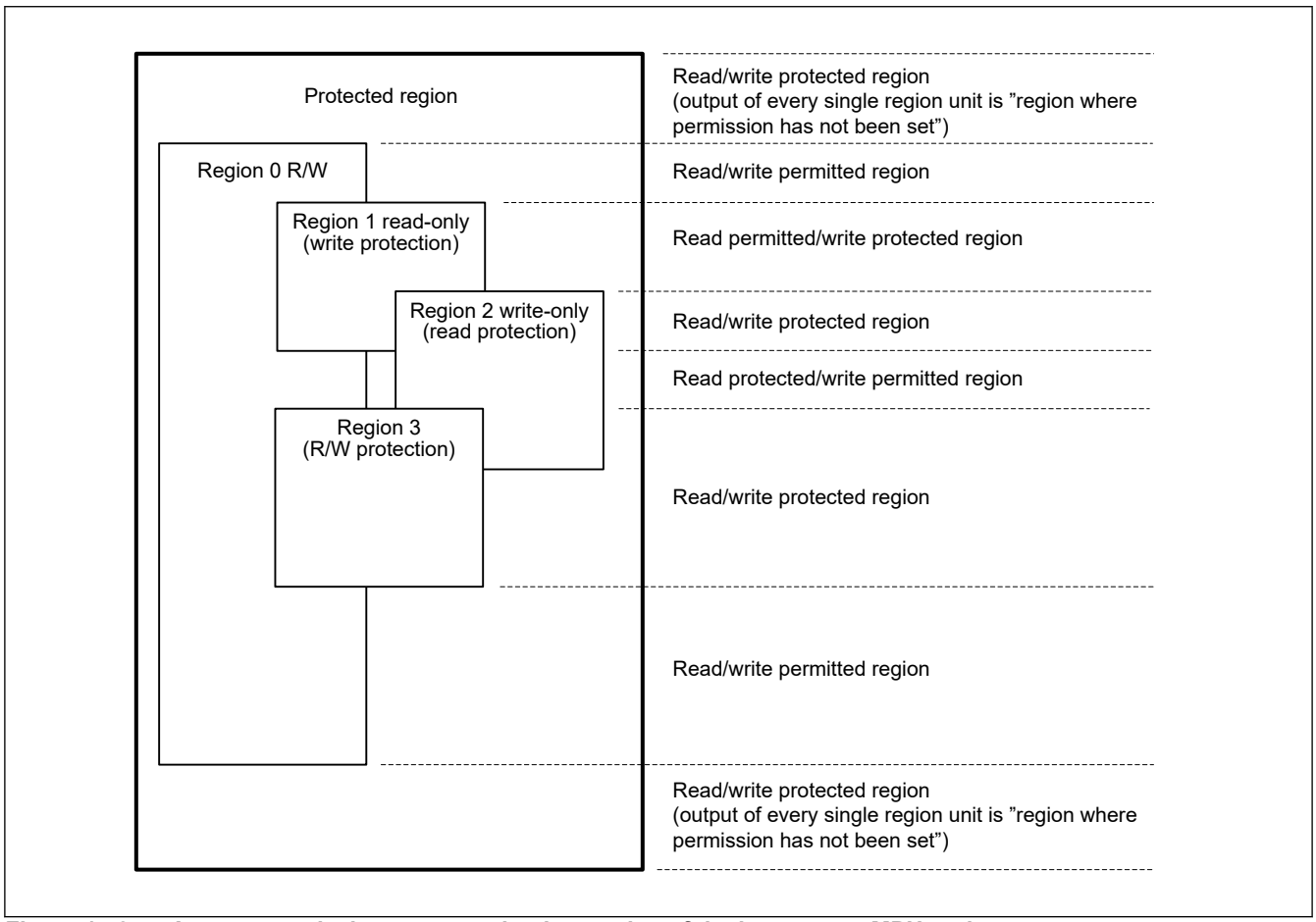



**Figure 15.2 Access permission or protection by overlap of the bus master MPU regions**

Figure 15.3 shows the register setting flow after reset. During this register setting, stop all bus masters except the CPU.

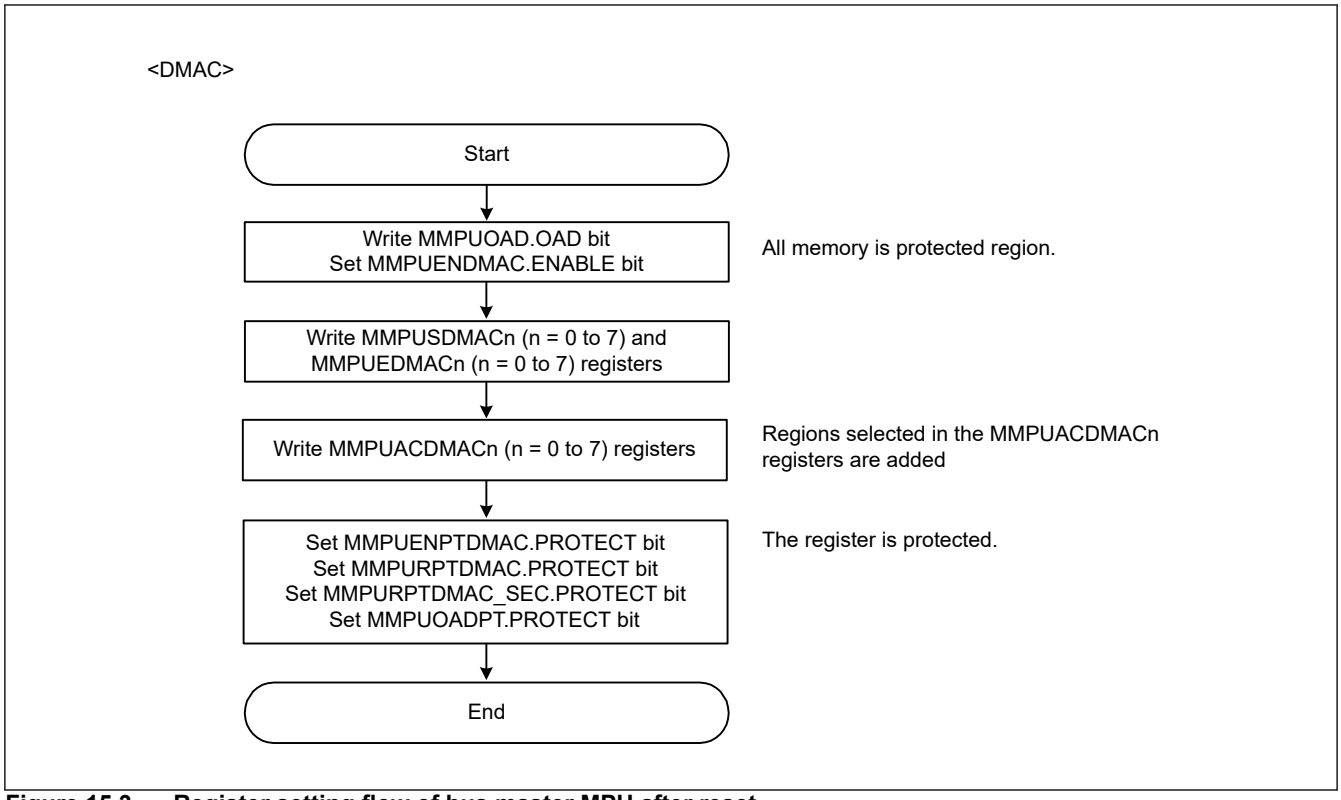



**Figure 15.3 Register setting flow of bus master MPU after reset**

Figure 15.4 shows the register setting flow for adding regions. During this register setting, stop all masters except the CPU.

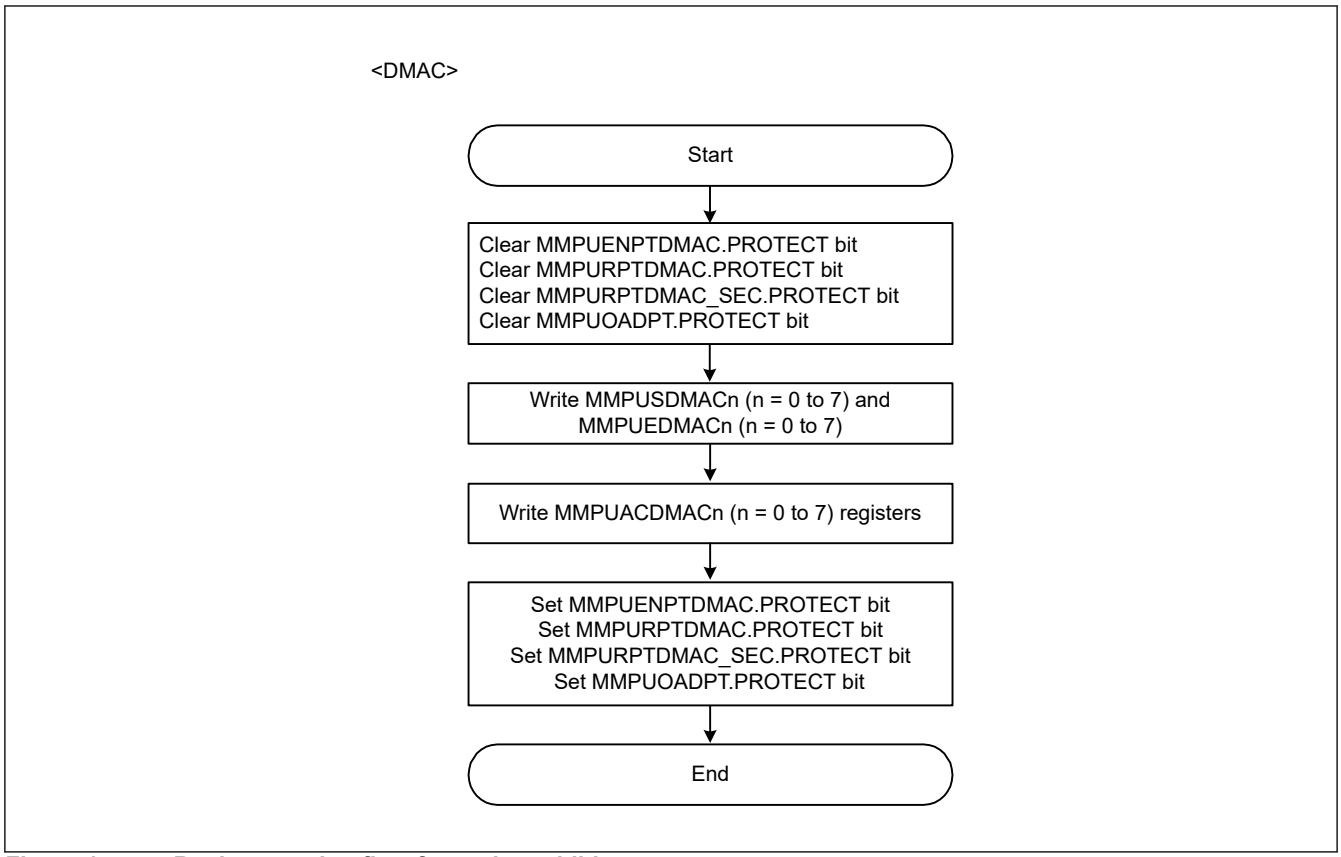



**Figure 15.4 Register setting flow for region addition**

#### 15.3.2.2 Protecting the registers

Registers related to the Bus Master MPU can be protected with the PROTECT bit in the MMPUENPTDMA, MMPURPTDMAC, MMPURPTDMAC_SEC and MMPUOADPT registers.

**Table 15.6 PROTECT bit and Protected target registers**

| PROTECT bit | Protect target registers |
|---|---|
| MMPUENPTDMAC.PROTECT | MMPUENDMAC |
| MMPURPTDMAC.PROTECT | The following registers set to Non-Secure by MMPUSARA.MMPUASAn (n = 0 to 7).<br>MMPUSDMACn (n = 0 to 7)<br>MMPUEDMACn (n = 0 to 7)<br>MMPUACDMACn (n = 0 to 7) |
| MMPURPTDMAC_SEC.PROTECT | The following registers set to Secure by MMPUSARA.MMPUASAn (n = 0 to 7).<br>MMPUSDMACn (n = 0 to 7)<br>MMPUEDMACn (n = 0 to 7)<br>MMPUACDMACn (n = 0 to 7) |
| MMPUOADPT.PROTECT | MMPUOAD |

#### 15.3.2.3 Memory protection error

If access to a protected region is detected, the bus master MPU generates an error. Set the OAD bit to select whether the error is reported as a non-maskable interrupt or a reset.

The non-maskable interrupt status is indicated in ICU.NMISR.BUSMST. For details, see section 13, Interrupt Controller Unit (ICU). The reset status is indicated in SYSTEM.RSTSR1.BUSMRF. For details, see section 5, Resets.

## 15.4 References

1. *Arm®v8-M Architecture Reference Manual* (ARM DDI0553B.g)
2. *Arm® Cortex®-M33 Processor Technical Reference Manual* (ARM 100230_0004_00_en)

# 16. DMA Controller (DMAC)

## 16.1 Overview

The MCU includes an 8-channel direct memory access controller (DMAC) that can transfer data without intervention from the CPU. When a DMA transfer request is generated, the DMAC transfers data stored at the transfer source address to the transfer destination address.

Table 16.1 lists the DMAC specifications, and Figure 16.1 shows a block diagram of the DMAC.

**Table 16.1 DMAC specifications (1 of 2)**

| Item | | Description |
|---|---|---|
| Number of channels | | 8 channels (DMACn (n = 0 to 7)) |
| Transfer space | | 4 GB<br>(0x00000000 to 0xFFFFFFFF excluding reserved areas) |
| Maximum transfer volume | | 64 M data<br>(Maximum number of transfers in block transfer mode: 1,024 data/block × 65,536 blocks) |
| DMAC activation source | | Selectable for each channel:<br>● Software trigger<br>● Interrupt requests from peripheral modules or trigger from external interrupt input pins.*1 |
| Channel priority | | Channel 0 > Channel 1 > Channel 2 > Channel 3… > Channel 7 (Channel 0: Highest) |
| Transfer data | Single data | Bit length: 8, 16, 32 bits |
| | Block size | Number of data: 1 to 1,024 |
| Transfer mode | Normal transfer mode | ● One data transfer by one DMA transfer request<br>● Free-running function (setting in which total number of data transfers is not specified) settable |
| | Repeat transfer mode | ● One data transfer by one DMA transfer request<br>● Program returns to the transfer start address on completion of the repeat size of data transfer specified for the transfer source or destination.<br>● Maximum settable repeat size: 1,024<br>● Selectable free-running function |
| | Repeat-block transfer mode | ● One block data transfer by one DMA transfer request<br>● Maximum settable block size: 1,024<br>● Block transfer can be repeated<br>● Maximum settable repeat size: 64K<br>● Selectable free-running function |
| | Block transfer mode | ● One block data transfer by one DMA transfer request<br>● Maximum settable block size: 1,024 data<br>● Selectable free-running function |
| Selective functions | Extended repeat area function | ● Function in which data can be transferred by repeating the address values in the specified range with the upper bit values in the transfer address register fixed.<br>● Area of 2 bytes to 128 Mbytes separately settable as extended repeat area for transfer source and destination. |
| Processing on DMAC transfer error | | ● When a DMAC transfer error occurs, the transfer on the channel that caused the error is stopped.<br>● A request to clear the register for activation request of DMAC error channel is sent to ICU. |
| Interrupt (DMACn_INT) | Transfer end interrupt | Generated on completion of transferring data volume specified by the transfer counter. |
| | Transfer escape end interrupt | ● Generated when the repeat size of data transfer is completed.<br>● Generated when the source address extended repeat area overflows.<br>● Generated when the destination address extended repeat area overflows. |
| Interrupt (DMA_TRANSERR) | Error response detection interrupt | Generated when the DMAC transfer error occurs. |
| Event link activation (DMACn_INT) | | An event link request is generated after each data transfer (for block transfer, after each block is transferred). |

**Table 16.1 DMAC specifications (2 of 2)**

| Item | Description |
|---|---|
| Master TrustZone Filter | TrustZone violation area of Flash and SRAM is detected before a non-secure channel access the bus. |
| Power consumption reduction function | Module-stop state can be set. |
| TrustZone Filter | Security attribution can be set for each channels |

Note: Security attribution Register of DMAC channel is described in ICU.ICUSARC
Note 1. For details on DMAC activation sources, see Table 13.4 in section 13, Interrupt Controller Unit (ICU).

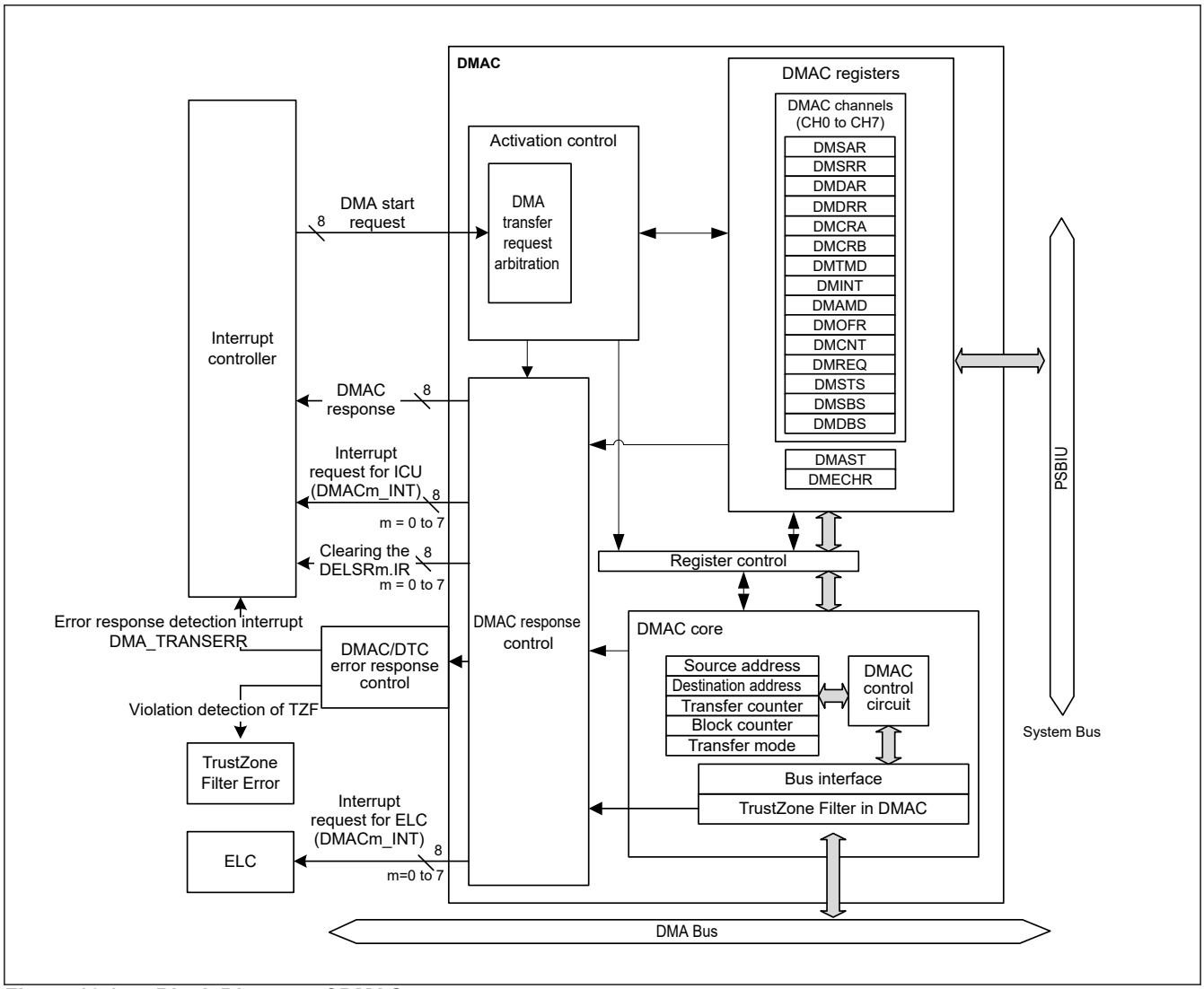



**Figure 16.1 Block Diagram of DMAC**

## 16.2 Register Descriptions

### 16.2.1 DMACSAR : DMAC Controller Security Attribution Register

Base address: CPSCU = 0x4000_8000

Offset address: 0x34

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | DMASTSA |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DMASTSA | DMAST Security Attribution<br>0: Secure<br>1: Non-secure | R/W |
| 31:1 | — | These bits are read as 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

For DMAC, security attribution is set for each channel. However, this register only sets the DMAST register security attribute. The security attribution setting of each channel is described in the section 13.2.3. ICUSARC : Interrupt Controller Unit Security Attribution Register C.

#### DMASTSA bit (DMAST Security Attribution)

Security attributes of registers for DMAST. Do not write to DMASTSA bit while DMA transfer is enabled or a bus master is writing to the DMA registers.

### 16.2.2 DMSAR : DMA Source Address Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x00

| Bit position: | 31 | 0 |
|---|---|---|
| Bit field: | | |
| Value after reset: | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | Specifies the transfer source start address<br>Setting range is 0x0000_0000 to 0xFFFF_FFFF (4 Gbytes). | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

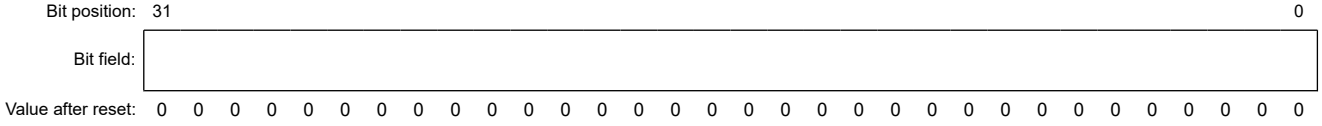
Set DMSAR while DMAC activation is disabled (DMAST.DMST = 0) or DMA transfer is disabled (DMCNT.DTE = 0).

Note: Address alignment in this register must match the Transfer Data Size value selected in the DMTMD.SZ bits.

### 16.2.3 DMSRR : DMA Source Reload Address Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x20




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | Specifies the transfer source reload address<br>Setting range is 0x0000_0000 to 0xFFFF_FFFF (4 Gbytes). | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Set DMSRR while DMAC activation is disabled (DMAST.DMST = 0) or DMA transfer is disabled (DMCNT.DTE = 0).

DMSRR is the initial value of DMSAR. In repeat-block transfer mode, DMSAR reloads the value of DMSRR after the specified transfer is finished.

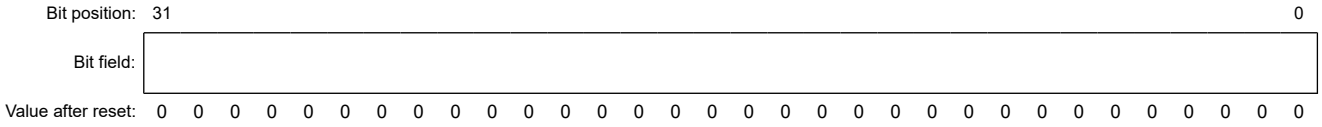
In normal transfer mode, repeat transfer mode, and block transfer mode DMSRR is not used. The setting is invalid.

Note: Address alignment in this register must match the Transfer Data Size value selected in the DMTMD.SZ bits.

### 16.2.4 DMDAR : DMA Destination Address Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x04




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | Specifies the transfer destination start address<br>Setting range is 0x0000_0000 to 0xFFFF_FFFF (4 Gbytes). | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

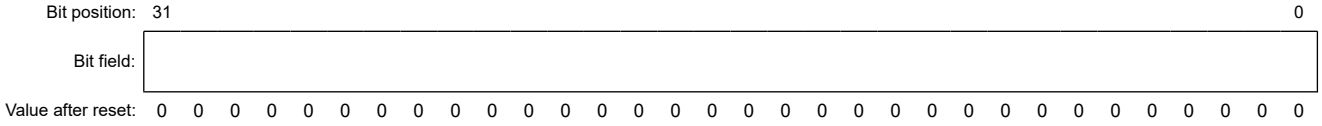
Set DMDAR while DMAC activation is disabled (DMAST.DMST = 0) or DMA transfer is disabled (DMCNT.DTE = 0).

Note: Address alignment in this register must match the Transfer Data Size value selected in the DMTMD.SZ bits.

### 16.2.5 DMDRR : DMA Destination Reload Address Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x24



| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | Specifies the transfer destination reload address<br>Setting range is 0x0000_0000 to 0xFFFF_FFFF (4 Gbytes). | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Set DMDRR while DMAC activation is disabled (DMAST.DMST = 0) or DMA transfer is disabled (DMCNT.DTE = 0).

DMDRR is the initial value of DMDAR. In repeat-block transfer mode, DMDAR reloads the value of DMDRR after the specified transfer is finished.

In normal transfer mode, repeat transfer mode and block transfer mode, DMDRR is not used. The setting is invalid.

Note: Address alignment in this register must match the Transfer Data Size value selected in the DMTMD.SZ bits.

## 16.2.6 DMCRA : DMA Transfer Count Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x08

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | DMCRAH[9:0] | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DMCRAL[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | DMCRAL[15:0] | Lower bits of transfer count<br>Specifies the number of transfer operations. | R/W |
| 25:16 | DMCRAH[9:0] | Upper bits of transfer count<br>Specifies the number of transfer operations. | R/W |
| 31:26 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Set the same value for DMCRAH and DMCRAL in repeat transfer mode, block transfer mode, and repeat-block transfer mode. Bits 15 to 10 are fixed to 0 in repeat transfer mode, block transfer mode, and repeat-block transfer mode.

### (1) Normal Transfer Mode (DMTMD.MD[1:0] = 00b)

DMCRAL functions as a 16-bit transfer counter.

The number of transfer operations is one when the setting is 0x0001, and 65,535 when it is 0xFFFF. The value is decremented by one each time data is transferred.

When the setting is 0x0000, no specific number of transfer operations is set; data transfer is performed with the transfer counter stopped (free running function).

Free running function is not selected by DMTMD.TKP bit in normal transfer mode.

DMCRAH is not used in normal transfer mode. Write 0x0000 to DMCRAH.

### (2) Repeat Transfer Mode (DMTMD.MD[1:0] = 01b)

DMCRAH specifies the repeat size and DMCRAL functions as a 10-bit transfer counter.

The number of transfer operations is one when the setting is 0x001, 1023 when it is 0x3FF, and 1024 when it is 0x000. In repeat transfer mode, a value in the range of 0x000 to 0x3FF (1 to 1024) can be set for DMCRAH and DMCRAL.

Setting bits 15 to 10 in DMCRAL is invalid. Write 0 to these bits.

The value in DMCRAL is decremented by one each time data is transferred until it reaches 0x000, at which the value in DMCRAH is loaded into DMCRAL.

### (3) Block Transfer Mode (DMTMD.MD[1:0] = 10b)

DMCRAH specifies the block size and DMCRAL functions as a 10-bit block size counter.

The block size is one when the setting is 0x001, 1023 when it is 0x3FF, and 1024 when it is 0x000. In block transfer mode, a value in the range of 0x000 to 0x3FF can be set for DMCRAH and DMCRAL.

Setting bits 15 to 10 in DMCRAL is invalid. Write 0 to these bits.

The value in DMCRAL is decremented by one each time data is transferred until it reaches 0x000, at which the value in DMCRAH is loaded into DMCRAL.

### (4) Repeat-Block Transfer Mode (DMTMD.MD[1:0] = 11b)

DMCRAH specifies the block size and DMCRAL functions as a 10-bit block size counter.

The block size is one when the setting is 0x001, 1023 when it is 0x3FF, and 1024 when it is 0x000. In repeat-block transfer mode, a value in the range of 0x000 to 0x3FF can be set for DMCRAH and DMCRAL.

Setting bits 15 to 10 in DMCRAL is invalid. Write 0 to these bits.

The value in DMCRAL is decremented by one each time data is transferred until it reaches 0x000, at which the value in DMCRAH is loaded into DMCRAL.

## 16.2.7 DMCRB : DMA Block Transfer Count Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x0C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DMCRBH[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Bit field: | DMCRBL[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | DMCRBL[15:0] | Functions as a number of block, repeat or repeat-block transfer counter.<br>0x0001 to 0xFFFF (1 to 65535)<br>0x0000 (65536) | R/W |
| 31:16 | DMCRBH[15:0] | Specifies the number of block, repeat or repeat-block transfer operations.<br>0x0001 to 0xFFFF (1 to 65535)<br>0x0000 (65536) | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Set the same value for DMCRBH and DMCRBL in repeat transfer mode, block transfer mode and repeat-block transfer mode.

DMCRBH specifies the number of block, repeat and repeat-block transfer operations, and DMCRBL functions as a 16-bit the number of block counter in block, repeat, and repeat-block transfer mode, respectively.

The number of transfer operations is one when the setting is 0x0001, 65535 when it is 0xFFFF, and 65536 when it is 0x0000.

In repeat transfer mode, the value is decremented by one when the final data of one repeat size is transferred.

In block transfer mode and repeat-block transfer mode, the value is decremented by one when the final data of one block size is transferred.

In normal transfer mode, DMCRB is not used. The setting is invalid.

When DMTMD.TKP is 1 and the final data of one repeat size or one block size is transferred, DMCRBL reloads the value of DMCRBH automatically.

## 16.2.8 DMTMD : DMA Transfer Mode Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x10

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MD[1:0] | | DTS[1:0] | | — | TKP | SZ[1:0] | | — | — | — | — | — | — | DCTG[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | DCTG[1:0] | Transfer Request Source Select<br>0 0: Software request<br>0 1: Hardware request*1<br>1 0: Setting prohibited<br>1 1: Setting prohibited | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 9:8 | SZ[1:0] | Transfer Data Size Select<br>0 0: 8 bits<br>0 1: 16 bits<br>1 0: 32 bits<br>1 1: Setting prohibited | R/W |
| 10 | TKP | Transfer Keeping<br>0: Transfer is stopped by completion of specified total number of transfer operations.<br>1: Transfer is not stopped by completion of specified total number of transfer operations (free-running). | R/W |
| 11 | — | This bit is read as 0. The write value should be 0. | R/W |
| 13:12 | DTS[1:0] | Repeat Area Select<br>0 0: The destination is specified as the repeat area or block area.<br>0 1: The source is specified as the repeat area or block area.<br>1 0: The repeat area or block area is not specified.<br>1 1: Setting prohibited. | R/W |
| 15:14 | MD[1:0] | Transfer Mode Select<br>0 0: Normal transfer<br>0 1: Repeat transfer<br>1 0: Block transfer<br>1 1: Repeat-block transfer | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. To select the DMAC activation source, use the DELSRn registers of the ICU. For details on DMAC activation sources, see Table 13.4 in section 13, Interrupt Controller Unit (ICU).

### DTS[1:0] bits (Repeat Area Select)

DTS[1:0] select either the source or destination as the repeat area in repeat or block transfer mode. In normal or repeat-block transfer mode, setting these bits is invalid.

### TKP bit (Transfer Keeping)

TKP selects either stopping transfer or keeping transfer by completion of specified total number of transfer operations in repeat, block or repeat-block transfer mode. In normal transfer mode, setting this bit is invalid.

## 16.2.9 DMINT : DMA Interrupt Setting Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x13

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | DTIE | ESIE | RPTIE | SARIE | DARIE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DARIE | Destination Address Extended Repeat Area Overflow Interrupt Enable<br>0: Disables an interrupt request for an extended repeat area overflow on the destination address.<br>1: Enables an interrupt request for an extended repeat area overflow on the destination address. | R/W |
| 1 | SARIE | Source Address Extended Repeat Area Overflow Interrupt Enable<br>0: Disables an interrupt request for an extended repeat area overflow on the source address.<br>1: Enables an interrupt request for an extended repeat area overflow on the source address. | R/W |
| 2 | RPTIE | Repeat Size End Interrupt Enable<br>0: Disables the repeat size end interrupt request.<br>1: Enables the repeat size end interrupt request. | R/W |
| 3 | ESIE | Transfer Escape End Interrupt Enable<br>0: Disables the transfer escape end interrupt request.<br>1: Enables the transfer escape end interrupt request. | R/W |
| 4 | DTIE | Transfer End Interrupt Enable<br>0: Disables the transfer end interrupt request.<br>1: Enables the transfer end interrupt request. | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### DARIE bit (Destination Address Extended Repeat Area Overflow Interrupt Enable)

When an extended repeat area overflow on the destination address occurs while DARIE bit is set to 1, the DMCNT.DTE bit is cleared to 0. At the same time, the DMSTS.ESIF flag is set to 1 to indicate that an interrupt by an extended repeat area overflow on the destination address is requested.

When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting 1 in the DMCNT.DTE bit of the channel for which a transfer has been stopped, the transfer is resumed from the state when the transfer is stopped.

When the extended repeat area is not specified for the destination address, this bit is ignored.

When set to repeat-block transfer mode, do not use this bit.

### SARIE bit (Source Address Extended Repeat Area Overflow Interrupt Enable)

When an extended repeat area overflow on the source address occurs while SARIE bit is set to 1, the DMCNT.DTE bit is cleared to 0. At the same time, the DMSTS.ESIF flag is set to 1 to indicate that an interrupt by an extended repeat area overflow on the source address is requested.

When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting 1 in the DMCNT.DTE bit of the channel for which a transfer has been stopped, the transfer is resumed from the state when the transfer is stopped.

When the extended repeat area is not specified for the source address, this bit is ignored.

When set to repeat-block transfer mode, do not use this bit.

### RPTIE bit (Repeat Size End Interrupt Enable)

When RPTIE bit is set to 1 in repeat transfer mode, the DMCNT.DTE bit is cleared to 0 after completion of a 1-repeat size data transfer. At the same time, the DMSTS.ESIF flag is set to 1 to indicate that the repeat size end interrupt request has been generated. The repeat size end interrupt request can be generated even when the DMTMD.DTS[1:0] bits are 10b (= repeat area or block area is not specified).

When this bit is set to 1 in block transfer mode, the DMCNT.DTE bit is cleared to 0 after completion of a 1-block data transfer in the same way as repeat transfer mode. At the same time, the DMSTS.ESIF flag is set to 1 to indicate that the repeat size end interrupt request has been generated. The repeat size end interrupt request can be generated even when the DMTMD.DTS[1:0] bits are 10b (= repeat area or block area is not specified).

When set to repeat-block transfer mode, do not use this bit.

### ESIE bit (Transfer Escape End Interrupt Enable)

ESIE bit enables or disables the transfer escape end interrupt requests (repeat size end interrupt request and extended repeat area overflow interrupt request) that are generated during DMA transfer.

The transfer escape end interrupt is generated when the DMSTS.ESIF flag is set to 1 with this bit set to 1. The transfer escape end interrupt is cleared by clearing this bit or the DMSTS.ESIF flag to 0.

### DTIE bit (Transfer End Interrupt Enable)

DTIE bit enables or disables the transfer end interrupt request to be generated on completion of a specified number of data transfers.

The transfer end interrupt is generated when the DMSTS.DTIF flag is set to 1 with this bit set to 1. The transfer end interrupt is cleared by clearing this bit or the DMSTS.DTIF flag to 0.

## 16.2.10 DMAMD : DMA Address Mode Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x14

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SM[1:0] | | SADR | SARA[4:0] | | | | | DM[1:0] | | DADR | DARA[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | DARA[4:0] | Destination Address Extended Repeat Area<br>Specifies the extended repeat area on the destination address. For details on the settings, see Table 16.2. | R/W |
| 5 | DADR | Destination Address Update Select After Reload<br>0: Only reloading.<br>1: Add index after reloading. | R/W |
| 7:6 | DM[1:0] | Destination Address Update Mode<br>0 0: Destination address is fixed.<br>0 1: Offset addition.<br>1 0: Destination address is incremented.<br>1 1: Destination address is decremented. | R/W |
| 12:8 | SARA[4:0] | Source Address Extended Repeat Area<br>Specifies the extended repeat area on the source address. For details on the settings, see Table 16.2. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 13 | SADR | Source Address Update Select After Reload<br>0: Only reloading.<br>1: Add index after reloading. | R/W |
| 15:14 | SM[1:0] | Source Address Update Mode<br>0 0: Source address is fixed.<br>0 1: Offset addition.<br>1 0: Source address is incremented.<br>1 1: Source address is decremented. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### DARA[4:0] bits (Destination Address Extended Repeat Area)

DARA[4:0] bits specify the extended repeat area on the destination address. The extended repeat area function is realized by updating the specified lower address bits with the remaining upper address bits fixed. The size of the extended repeat area can be any power of two between 2 bytes and 128 Mbytes.

When the lower address overflows the extended repeat area by address increment, the start address of the extended repeat area is set. Similarly, when the lower address underflows the extended repeat area by address decrement, the end address of the extended repeat area is set.

When the repeat area or block area is specified as a transfer destination, do not specify the extended repeat area on the destination address. When repeat transfer or block transfer is selected, and when DMTMD.DTS[1:0] = 00b (the transfer destination is specified as the repeat area or block area), write 00000b in the DARA[4:0] bits.

In repeat-block transfer mode, write 00000b in the DARA[4:0] bits.

An interrupt can be requested when an overflow or underflow occurs in the extended repeat area with the DMINT.DARIE bit set to 1. Table 16.2 lists the settings and the corresponding extended repeat areas.

### DADR bits (Destination Address Update Select After Reload)

In repeat-block transfer mode, this bit specifies the behavior of DMDAR after reloading DMDRR.

When this bit is set to 1, an index value ((DMDBSH-DMDBSL) × DataSize) is added to DMDAR after reloading DMDRR.

When this bit is set to 0, DMDAR only reloads DMDRR. This behavior is described in Table 16.13.

In normal, repeat or block transfer mode, this bit is ignored.

### DM[1:0] bits (Destination Address Update Mode)

DM[1:0] bits select the mode of updating the destination address.

When increment is selected and the DMTMD.SZ[1:0] bits are set to 00b, 01b, or 10b, the destination address is incremented by 1, 2, or 4, respectively.

When decrement is selected and the DMTMD.SZ[1:0] bits are set to 00b, 01b, or 10b, the destination address is decremented by 1, 2, or 4, respectively.

When offset addition is selected, the offset specified by the DMOFR register is added to the address.

### SARA[4:0] bits (Source Address Extended Repeat Area)

SARA[4:0] bits specify the extended repeat area on the source address. The extended repeat area function is realized by updating the specified lower address bits with the remaining upper address bits fixed. The size of the extended repeat area can be any power of two between 2 bytes and 128 Mbytes.

When the lower address overflows the extended repeat area by address increment, the start address of the extended repeat area is set. Similarly, when the lower address underflows the extended repeat area by address decrement, the end address of the extended repeat area is set.

When the repeat area or block area is specified as a transfer source, do not specify the extended repeat area on the source address. When repeat transfer or block transfer is selected, and when DMTMD.DTS[1:0] = 01b (the transfer source is specified as the repeat area or block area), write 00000b in the SARA[4:0] bits.

In repeat-block transfer mode, write 00000b in the SARA[4:0] bits.

An interrupt can be requested when an overflow or underflow occurs in the extended repeat area with the DMINT.SARIE bit set to 1. Table 16.2 lists the settings and the corresponding extended repeat areas.

### SADR bits (Source Address Update Select After Reload)

In repeat-block transfer mode, this bit specifies the behavior of DMSAR after reloading DMSRR.

When this bit is set to 1, an index value ((DMSBSH-DMSBSL) × DataSize) is added to DMSAR after reloading DMSRR.

When this bit is set to 0, DMSAR only reloads DMSRR. This behavior is described in Table 16.12.

In normal, repeat or block transfer mode, this bit is ignored.

### SM[1:0] bits (Source Address Update Mode)

SM[1:0] bits select the mode of updating the source address.

When increment is selected and the DMTMD.SZ[1:0] bits are set to 00b, 01b, or 10b, the source address is incremented by 1, 2, or 4, respectively.

When decrement is selected and the DMTMD.SZ[1:0] bits are set to 00b, 01b, or 10b, the source address is decremented by 1, 2, or 4, respectively.

When offset addition is selected, the offset specified by the DMOFR register is added to the address.

**Table 16.2 SARA[4:0] or DARA[4:0] settings and corresponding repeat areas (1 of 2)**

| SARA[4:0] or DARA[4:0] settings | Extended repeat area |
|---|---|
| 00000b | Not specified |
| 00001b | 2 bytes specified as extended repeat area by the lower 1 bit of the address |
| 00010b | 4 bytes specified as extended repeat area by the lower 2 bits of the address |
| 00011b | 8 bytes specified as extended repeat area by the lower 3 bits of the address |
| 00100b | 16 bytes specified as extended repeat area by the lower 4 bits of the address |
| 00101b | 32 bytes specified as extended repeat area by the lower 5 bits of the address |
| 00110b | 64 bytes specified as extended repeat area by the lower 6 bits of the address |
| 00111b | 128 bytes specified as extended repeat area by the lower 7 bits of the address |
| 01000b | 256 bytes specified as extended repeat area by the lower 8 bits of the address |
| 01001b | 512 bytes specified as extended repeat area by the lower 9 bits of the address |
| 01010b | 1 Kbyte specified as extended repeat area by the lower 10 bits of the address |
| 01011b | 2 Kbytes specified as extended repeat area by the lower 11 bits of the address |
| 01100b | 4 Kbytes specified as extended repeat area by the lower 12 bits of the address |
| 01101b | 8 Kbytes specified as extended repeat area by the lower 13 bits of the address |
| 01110b | 16 Kbytes specified as extended repeat area by the lower 14 bits of the address |
| 01111b | 32 Kbytes specified as extended repeat area by the lower 15 bits of the address |
| 10000b | 64 Kbytes specified as extended repeat area by the lower 16 bits of the address |
| 10001b | 128 Kbytes specified as extended repeat area by the lower 17 bits of the address |
| 10010b | 256 Kbytes specified as extended repeat area by the lower 18 bits of the address |
| 10011b | 512 Kbytes specified as extended repeat area by the lower 19 bits of the address |
| 10100b | 1 Mbyte specified as extended repeat area by the lower 20 bits of the address |
| 10101b | 2 Mbytes specified as extended repeat area by the lower 21 bits of the address |
| 10110b | 4 Mbytes specified as extended repeat area by the lower 22 bits of the address |
| 10111b | 8 Mbytes specified as extended repeat area by the lower 23 bits of the address |
| 11000b | 16 Mbytes specified as extended repeat area by the lower 24 bits of the address |
| 11001b | 32 Mbytes specified as extended repeat area by the lower 25 bits of the address |

**Table 16.2 SARA[4:0] or DARA[4:0] settings and corresponding repeat areas (2 of 2)**

| SARA[4:0] or DARA[4:0] settings | Extended repeat area |
|---|---|
| 11010b | 64 Mbytes specified as extended repeat area by the lower 26 bits of the address |
| 11011b | 128 Mbytes specified as extended repeat area by the lower 27 bits of the address |
| 11100b to 11111b | Setting prohibited. |

## 16.2.11 DMOFR : DMA Offset Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x18

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | Specifies the offset when offset addition is selected as the address update mode for transfer source or destination.<br>0x00000000 to 0x00FFFFFF (0 bytes to (16 M – 1) bytes)<br>0xFF000000 to 0xFFFFFFFF (–16 Mbytes to –1 byte) | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Write to this register while the DMAC operation is stopped or DMA transfer is disabled (not during data transfer).

Setting bits 31 to 25 is invalid; a value of bit 24 is extended to bits 31 to 25. Reading DMOFR returns the extended value.

In repeat-block transfer mode, the offset is not specified by DMOFR when offset addition is selected, write 0 to DMOFR.

## 16.2.12 DMCNT : DMA Transfer Enable Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x1C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | DTE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DTE | DMA Transfer Enable<br>0: Disables DMA transfer.<br>1: Enables DMA transfer. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### DTE bit (DMA Transfer Enable)

When the DMAST.DMST bit is set to 1 (DMAC activation is enabled) and this bit is set to 1 (DMA transfer is enabled), DMA transfer can be started for the corresponding channel.

[Setting condition]

- When 1 is written to this bit.

[Clearing conditions]

- When 0 is written to this bit.
- When the specified total volume of data transfer is completed.
- When DMA transfer is stopped by the repeat size end interrupt.
- When DMA transfer is stopped by the extended repeat area overflow interrupt.
- When DMA transfer is stopped by the access error occurs. See section 16.5. Processing on DMA Transfer Error.

## 16.2.13 DMREQ : DMA Software Start Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x1D

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | CLRS | — | — | — | SWREQ |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SWREQ | DMA Software Start<br>0: DMA transfer is not requested.<br>1: DMA transfer is requested. | R/W |
| 3:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | CLRS | DMA Software Start Bit Auto Clear Select<br>0: SWREQ bit is cleared after DMA transfer is started by software.<br>1: SWREQ bit is not cleared after DMA transfer is started by software. | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### SWREQ bit (DMA Software Start)

When 1 is written to SWREQ bit, a DMA transfer request is generated. After DMA transfer is started in response to the request, this bit is cleared to 0 if the CLRS bit is set to 0. This bit is not cleared to 0 while the CLRS bit is set to 1. In this case, a DMA transfer request can be issued again after completion of a transfer.

Note that, however, setting this bit is valid and DMA transfer by software is enabled only when the DMTMD.DCTG[1:0] bits are set to 00b (DMAC activation source is software).

Setting this bit is invalid when the DMTMD.DCTG[1:0] bits are set to a value other than 00b.

To start DMA transfer by software with the CLRS bit being 0, ensure that the SWREQ bit is 0, and then write 1 to the SWREQ bit.

[Setting condition]

- When 1 is written to this bit.

[Clearing conditions]

- When a DMA transfer request by software is accepted and DMA transfer is started while the CLRS bit is set to 0 (the SWREQ bit is cleared after DMA transfer is started by software).
- When 0 is written to this bit.

### CLRS bit (DMA Software Start Bit Auto Clear Select)

CLRS bit specifies whether to clear the SWREQ bit to 0 after DMA transfer is started in response to the DMA transfer request generated by setting the SWREQ bit to 1. With this bit set to 0, the SWREQ bit is cleared to 0 after DMA transfer is started. With this bit set to 1, the SWREQ bit is not cleared to 0. In this case, a DMA transfer request can be issued again after completion of a transfer.

## 16.2.14 DMSTS : DMA Status Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x1E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ACT | — | — | DTIF | — | — | — | ESIF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ESIF | Transfer Escape End Interrupt Flag<br>0: A transfer escape end interrupt has not been generated.<br>1: A transfer escape end interrupt has been generated. | R/W*1 |
| 3:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | DTIF | Transfer End Interrupt Flag<br>0: A transfer end interrupt has not been generated.<br>1: A transfer end interrupt has been generated. | R/W*1 |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | ACT | DMAC Active Flag<br>0: DMAC is in the idle state.<br>1: DMAC is operating. | R |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. Only 0 can be written to clear the flag.

### ESIF flag (Transfer Escape End Interrupt Flag)

This flag indicates that the transfer escape end interrupt has been generated.

[Setting conditions]

- When 1-repeat size data transfer is completed in repeat transfer mode with the DMINT.RPTIE bit set to 1.
- When 1-block data transfer is completed in block transfer mode with the DMINT.RPTIE bit set to 1.
- When an extended repeat area overflow on the source address occurs while the DMINT.SARIE bit is set to 1 and the DMAMD.SARA[4:0] bits are set to a value other than 00000b (extended repeat area is specified on the transfer source address).
- When an extended repeat area overflow on the destination address occurs while the DMINT.DARIE bit is set to 1 and the DMAMD.DARA[4:0] bits are set to a value other than 00000b (extended repeat area is specified on the transfer destination address).

[Clearing conditions]

- When 0 is written to this bit.
- When 1 is written to the DMCNT.DTE bit.

### DTIF flag (Transfer End Interrupt Flag)

This flag indicates that the transfer end interrupt has been generated.

[Setting conditions]

- When the specified number of unit-transfers are completed in normal transfer mode (the value of DMCRAL becoming 0 on completion of transfer).
- When the specified number of repeat transfer operations are completed in repeat transfer mode (the value of DMCRBL becoming 0 on completion of transfer with DMTMD.TKP = 0 or the value of DMCRBL reloading DMCRBH with DMTMD.TKP = 1).
- When the specified number of blocks have been transferred in block transfer mode and repeat-block transfer mode (the value of DMCRBL becoming 0 on completion of transfer with DMTMD.TKP = 0 or the value of DMCRBL reloading DMCRBH with DMTMD.TKP = 1).

[Clearing conditions]

- When 0 is written to this bit.
- When 1 is written to the DMCNT.DTE bit.

### ACT flag (DMAC Active Flag)

This flag indicates whether the DMAC is in the idle or active state.

[Setting condition]

- When the DMAC starts data transfer operation.

[Clearing condition]

- When data transfer in response to one transfer request is completed.

## 16.2.15 DMSBS : DMA Source Buffer Size Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x28

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DMSBSH[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DMSBSL[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | DMSBSL[15:0] | Functions as data transfer counter in repeat-block transfer mode<br>See Table 16.3 for available settings. | R/W |
| 31:16 | DMSBSH[15:0] | Specifies the repeat-area size in repeat-block transfer mode<br>See Table 16.3 for available settings. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Set the same value for DMSBSH and DMSBSL in repeat-block transfer mode. Write 0x00000000 to DMSBS in normal, repeat and block transfer mode.

DMSBSH specifies buffer size and DMSBSL functions as a 16-bit buffer size counter in repeat-block transfer mode. In repeat-block transfer mode, source repeat area is specified by DMSBSH.

When address update mode is incremented address or decremented address, this register means the numbers of data of whole buffer. When address update mode is offset addition, this register means the numbers of data of an individual buffer. In offset addition, setting DMSBSH and DMSBSL to 0x0000 is prohibited. When final data of one buffer size is transferred, DMSBSL reloads value of DMSBSH. When address update mode is fixed address, this register is ignored. Table 16.3 shows

the setting values of DMA Source Buffer Size Register corresponding to Transfer Data Size in Source Address Update Mode.

**Table 16.3 Available setting for DMSBS register in repeat-block transfer mode**

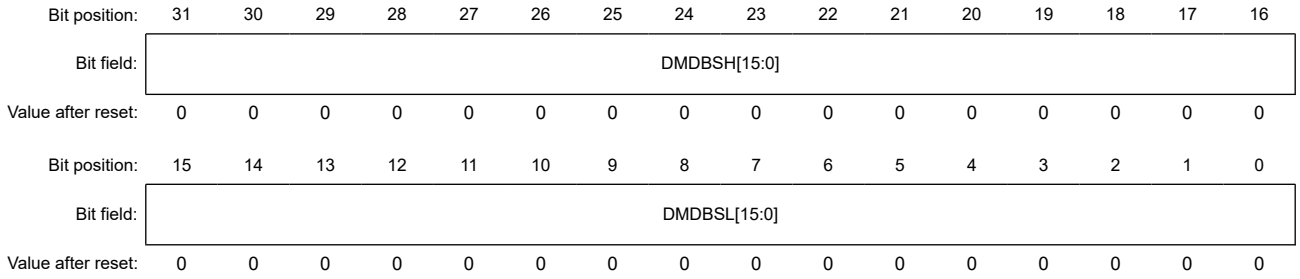
| Source address update mode (DMAMD.SM) | Transfer data size (DMTMD.SZ) | Available setting for DMSBSH and DMSBSL bits |
|---|---|---|
| Source address is fixed (SM = 00b) | Don't care | 0x0000 (DMSBS is not used) |
| Offset addition (SM = 01b) | 8 bits (SZ = 00b) | 0x0001 to 0xFFFF (1 to 65535) |
| | 16 bits (SZ = 01b) | 0x0001 to 0x7FFF (1 to 32767) |
| | 32 bits (SZ = 10b) | 0x0001 to 0x3FFF (1 to 16383) |
| Source address is incremented or decremented (SM = 1xb) | Don't care | 0x0000 (infinite)<br>0x0001 to 0xFFFF (1 to 65535) |

In normal, repeat and block transfer mode, DMSBS is not used. The setting is invalid.

## 16.2.16 DMDBS : DMA Destination Buffer Size Register

Base address: DMACn = 0x4000_5000 + 0x0040 × n (n = 0 to 7)

Offset address: 0x2C




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | DMDBSL[15:0] | Functions as data transfer counter in repeat-block transfer mode.<br>See Table 16.4 for available settings. | R/W |
| 31:16 | DMDBSH[15:0] | Specifies the repeat-area size in repeat-block transfer mode.<br>See Table 16.4 for available settings. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Set the same value for DMDBSH and DMDBSL in repeat-block transfer mode. Write 0x00000000 to DMDBS in normal, repeat and block transfer mode.

DMDBSH specifies buffer size and DMDBSL functions as a 16-bit buffer size counter in repeat-block transfer mode. In repeat-block transfer mode, destination repeat area is specified by DMDBSH.

When address update mode is incremented address or decremented address, this register means the numbers of data of whole buffer. When address update mode is offset addition, this register means the numbers of data of an individual buffer. In offset addition, setting DMDBSH and DMDBSL to 0x0000 is prohibited. When final data of one buffer size is transferred, DMDBSL reloads value of DMDBSH. When address update mode is fixed address, this register is ignored. Table 16.4 shows the setting values of Destination Buffer Size Register corresponding to Transfer Data Size in Destination Address Update Mode.

**Table 16.4 Available setting for DMDBS register in repeat-block transfer mode (1 of 2)**

| Destination address update mode (DMAMD.DM) | Transfer data size (DMTMD.SZ) | Available setting for DMDBSH and DMDBSL bits |
|---|---|---|
| Destination address is fixed (DM = 00b) | Don't care | 0x0000 (DMDBS is not used) |

**Table 16.4 Available setting for DMDBS register in repeat-block transfer mode (2 of 2)**

| Destination address update mode (DMAMD.DM) | Transfer data size (DMTMD.SZ) | Available setting for DMDBSH and DMDBSL bits |
|---|---|---|
| Offset addition (DM = 01b) | 8 bits (SZ = 00b) | 0x0001 to 0xFFFF (1 to 65535) |
| | 16 bits (SZ = 01b) | 0x0001 to 0x7FFF (1 to 32767) |
| | 32 bits (SZ = 10b) | 0x0001 to 0x3FFF (1 to 16383) |
| Destination address is incremented or decremented (DM = 1xb) | Don't care | 0x0000 (infinite)<br>0x0001 to 0xFFFF (1 to 65535) |

In normal, repeat and block transfer mode, DMDBS is not used. The setting is invalid.

## 16.2.17 DMAST : DMA Module Activation Register

Base address: DMA = 0x4000_5200

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | DMST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DMST | DMAC Operation Enable<br>0: DMAC activation is disabled.<br>1: DMAC activation is enabled. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### DMST bit (DMAC Operation Enable)

Setting the DMAST.DMST to 1 enables DMAC activation for all channels. When the DMST bit is set to 1 (DMAC activation is enabled), and 1 is written to the DMCNT.DTE bit (DMA transfer is enabled) for multiple channels, all associated channels can be placed in the transfer request ready state at the same time.

When the DMST bit clears to 0 during DMA transfer, DMA transfer is suspended after the current data transfer associated with a single transfer request completes. To resume DMA transfer, set the DMST bit to 1 again.

[Setting condition]

- When 1 is written to this bit.

[Clearing condition]

- When 0 is written to this bit.

## 16.2.18 DMECHR : DMAC Error Channel Register

Base address: DMA = 0x4000_5200

Offset address: 0x40

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | DMESTA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | DMECHSAM | — | — | — | — | — | DMECH | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | DMECH | DMAC Error channel<br>Indicates the channel number causing the error<br>0 0 0: Error occurred on Channel 0<br>0 0 1: Error occurred on Channel 1<br>0 1 0: Error occurred on Channel 2<br>⋮ ⋮<br>1 1 1: Error occurred on Channel 7 | R |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R |
| 8 | DMECHSAM | DMAC Error channel Security Attribution Monitor<br>Indicates the security attribution of a channel causing the error<br>0: secure channel<br>1: non-secure channel | R |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R |
| 16 | DMESTA | DMAC Error Status<br>0: No DMA transfer error occurred<br>1: DMA transfer error occurred | R/W*1 |
| 31:17 | — | These bits are read as 0. The write value should be 0. | R |

Note 1. Writing to DMESTA depends on the value of DMECHSAM

### DMECH[2:0] bits (DMAC Error channel)

When a transfer error due to DMA transfer occurs, the DMECH[2:0] bits store the violating DMAC channel.

When reset is selected in MPU.MMPUOAD.OAD and TZF.TZFOAD.OAD, this register is also reset. Select NMI when you want to debug the program.

[Set condition]

- When the DMAC transfer error occurs and DMESTA = 0.

[Clearing condition]

- When 1 is written to DMESTA.

### DMECHSAM bit (DMAC Error channel Security Attribution Monitor)

When a transfer error due to DMA transfer occurs, the DMECHSAM bit indicates the security attribution of the violating DMAC channel.

When reset is selected in MPU.MMPUOAD.OAD and TZF.TZFOAD.OAD, this register is also reset. Select NMI when you want to debug the program.

[Set condition]

- When the DMAC transfer error occurs and DMESTA = 0.

[Clearing condition]

- When 1 is written to DMESTA.

**DMESTA bit (DMAC Error Status)**

The DMESTA bit indicates whether or not a DMA transfer error occurred.

DMECH, DMECHSAM, DMESTA are cleared by writing 1 to DMESTA. Writing 0 to DMESTA is ignored.

When reset is selected in MPU.MMPUOAD.OAD and TZF.TZFOAD.OAD, this register is also reset. Select NMI when you want to debug the program.

[Set condition]

- When the DMAC transfer error occurs.

[Clearing condition]

- When 1 is written to DMESTA.

Note: When DMECHSAM = 1, it can be cleared in the secure state and non-secure state. DMECHSAM = 0, it cannot be cleared in the non-secure state.

## 16.3 Operation

### 16.3.1 Transfer Mode

#### 16.3.1.1 Normal Transfer Mode

In normal transfer mode, one data is transferred by one transfer request. A maximum of 65535 can be set as the number of transfer operations using the DMCRAL register. When these bits are set to 0x0000, no specific number of transfer operations is set; data transfer is performed with the transfer counter stopped (free-running function). Setting DMCRB register is invalid in normal transfer mode. Except in free-running function, a transfer end interrupt request can be generated after completion of the specified number of transfer operations.

Table 16.5 summarizes the register update operation in normal transfer mode, and Figure 16.2 shows the operation in normal transfer mode.

**Table 16.5 Register update operation in normal transfer mode**

| Register | Function | Update operation after completion of a transfer by one transfer request |
|---|---|---|
| DMSAR | Transfer source address | Increment/decrement/fixed/offset addition |
| DMDAR | Transfer destination address | Increment/decrement/fixed/offset addition |
| DMCRAL | Transfer count | Decremented by one/not updated (in free running function) |
| DMCRAH | — | Not updated (Not used in normal transfer mode) |
| DMCRB | — | Not updated (Not used in normal transfer mode) |

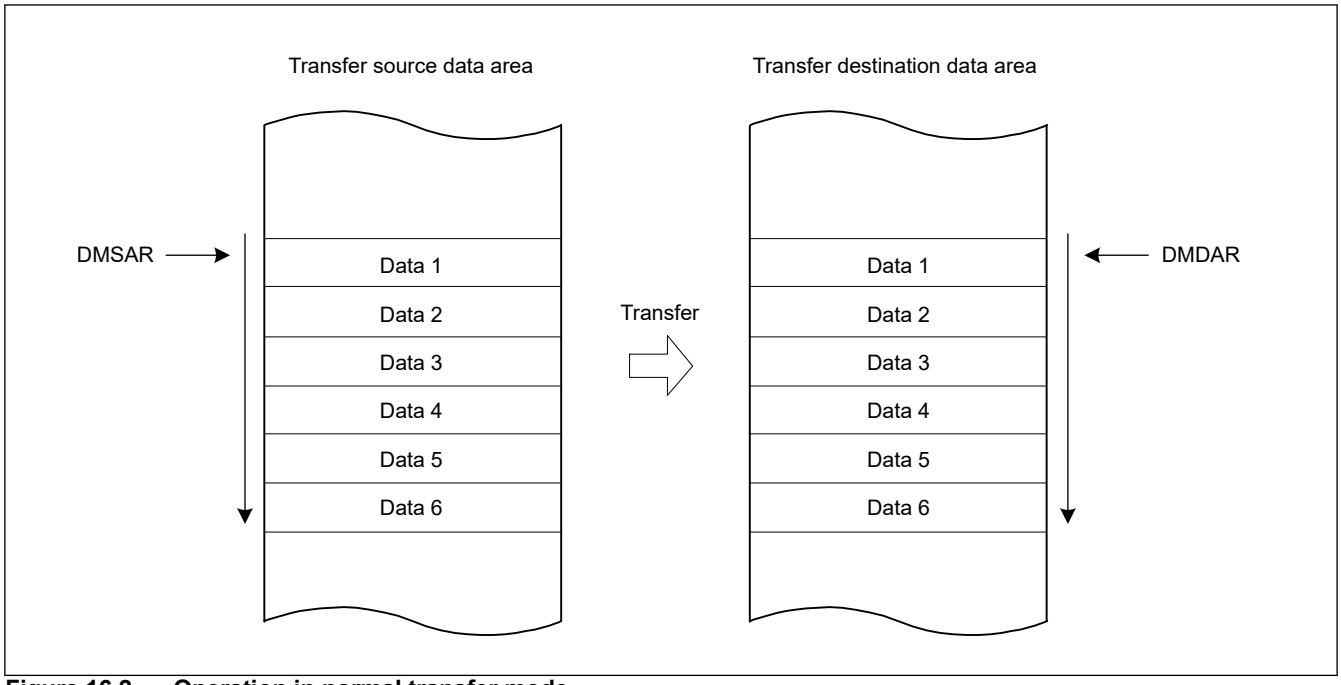



**Figure 16.2 Operation in normal transfer mode**

### 16.3.1.2 Repeat Transfer Mode

In repeat transfer mode, one data is transferred by one transfer request.

A maximum of 1K data can be set as a total repeat transfer size using DMCRA register.

A maximum of 64K can be set as the number of repeat transfer operations using DMCRB register; therefore, a maximum of 64M data (1K data × 64K counts of repeat transfer operations) can be set as a total data transfer size.

Either the transfer source or transfer destination can be specified as a repeat area. When transfer of the repeat size data is completed, the address of the specified repeat area (DMSAR or DMDAR) returns to the transfer start address. When data of the specified repeat size has all been transferred in repeat transfer mode, DMA transfer can be stopped, and the repeat size end interrupt can be requested. DMA transfer can be resumed by writing 1 to the DMCNT.DTE bit in the repeat size end interrupt handling.

A transfer end interrupt request can be generated after completion of the specified number of repeat transfer operations.

Table 16.6 summarizes the register update operation in repeat transfer mode, and Figure 16.3 shows the operation in repeat transfer mode.

**Table 16.6 Register update operation in repeat transfer mode (1 of 2)**

| Register | Function | Update operation after completion of a transfer by one transfer request | |
|---|---|---|---|
| | | When DMCRAL register is not 1 | When DMCRAL register is 1 (Transfer of the last data in repeat size) |
| DMSAR | Transfer source address | Increment/decrement/fixed/offset addition | • DMTMD.DTS[1:0] = 00b Increment/decrement/fixed/offset addition<br>• DMTMD.DTS[1:0] = 01b Initial value of DMSAR<br>• DMTMD.DTS[1:0] = 10b Increment/decrement/fixed/offset addition |
| DMDAR | Transfer destination address | Increment/decrement/fixed/offset addition | • DMTMD.DTS[1:0] = 00b Initial value of DMDAR<br>• DMTMD.DTS[1:0] = 01b Increment/decrement/fixed/offset addition<br>• DMTMD.DTS[1:0] = 10b Increment/decrement/fixed/offset addition |
| DMCRAH | Repeat size | Not updated | Not updated |
| DMCRAL | Transfer count | Decremented by one | DMCRAH |

**Table 16.6 Register update operation in repeat transfer mode (2 of 2)**

| Register | Function | Update operation after completion of a transfer by one transfer request | |
|---|---|---|---|
| | | When DMCRAL register is not 1 | When DMCRAL register is 1 (Transfer of the last data in repeat size) |
| DMCRBH | Number of repeat transfer operations | Not updated | Not updated |
| DMCRBL | Count of repeat transfer operations | Not updated | Decremented by one |

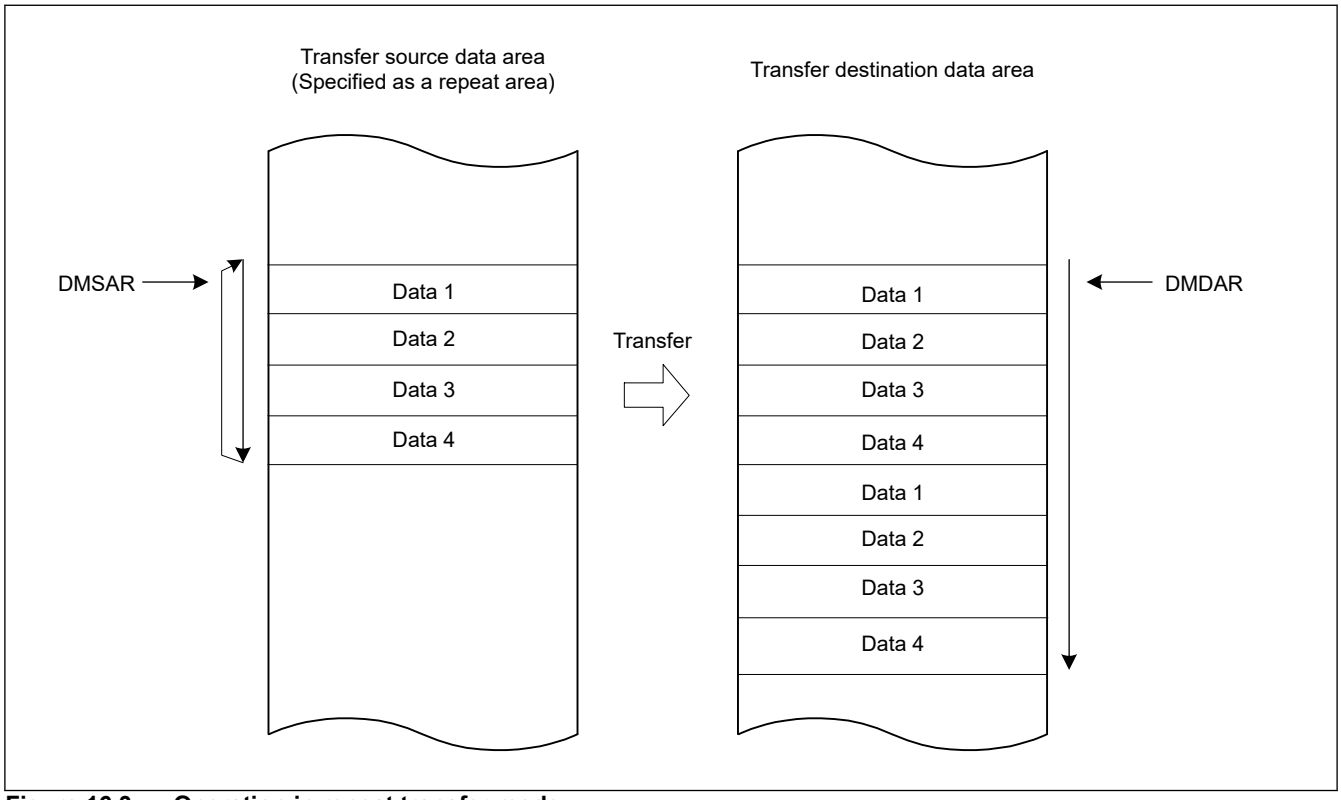



**Figure 16.3 Operation in repeat transfer mode**

### 16.3.1.3 Block Transfer Mode

In block transfer mode, a single block data is transferred by one transfer request.

A maximum of 1K data can be set as a total block transfer size using DMCRA register.

A maximum of 64K can be set as the number of block transfer operations using DMCRB register; therefore, a maximum of 64M data (1K data × 64K counts of block transfer operations) can be set as a total data transfer size.

Either the transfer source or transfer destination can be specified as a block area. When transfer of a single block data is completed, the address of the specified block area (DMSAR or DMDAR) returns to the transfer start address. When a single block data has all been transferred in block transfer mode, DMA transfer can be stopped, and the repeat size end interrupt can be requested. DMA transfer can be resumed by writing 1 to the DMCNT.DTE bit in the repeat size end interrupt handling.

Transfer end interrupt request can be generated after completion of the specified number of block transfer operations.

Table 16.7 summarizes the register update operation in block transfer mode, and Figure 16.4 shows the operation in block transfer mode.

**Table 16.7 Register update operation in block transfer mode**

| Register | Function | Update operation after completion of single-block transfer by one transfer request |
|---|---|---|
| DMSAR | Transfer source address | • DMTMD.DTS[1:0] = 00b<br>Increment/decrement/fixed/offset addition<br>• DMTMD.DTS[1:0] = 01b<br>Initial value of DMSAR<br>• DMTMD.DTS[1:0] = 10b<br>Increment/decrement/fixed/offset addition |
| DMDAR | Transfer destination address | • DMTMD.DTS[1:0] = 00b<br>Initial value of DMDAR<br>• DMTMD.DTS[1:0] = 01b<br>Increment/decrement/fixed/offset addition<br>• DMTMD.DTS[1:0] = 10b<br>Increment/decrement/fixed/offset addition |
| DMCRAH | Block size | Not updated |
| DMCRAL | Transfer count | DMCRAH |
| DMCRBH | Number of block transfer operations | Not updated |
| DMCRBL | Count of block transfer operations | Decremented by one |

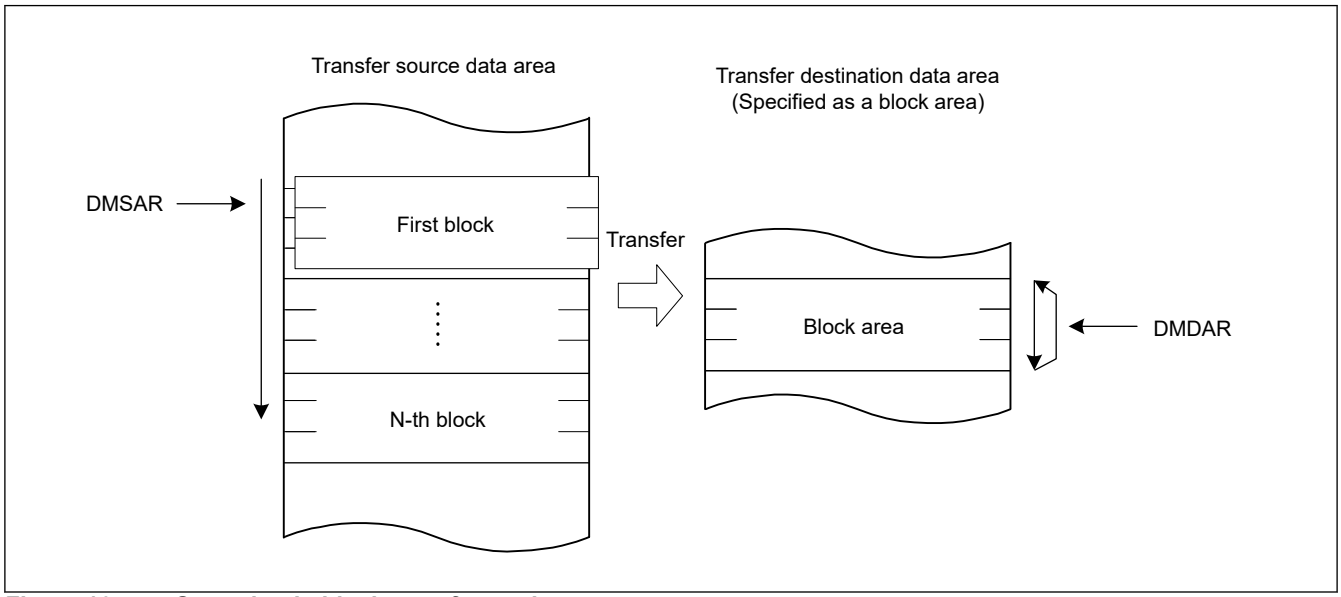



**Figure 16.4 Operation in block transfer mode**

### 16.3.1.4 Repeat-Block Transfer Mode

Repeat-block transfer is the operation mode with the following functions added to the block transfer function.

Repeat function: Added function (ring buffer) to repeat specified address area.

Offset function: Multiple areas with offset can be specified within one block transfer.

The repeat function and the offset function can be used for both the transfer source and the transfer destination of repeat-block transfer.

Figure 16.5 shows an example of adding a repeat function to the transfer destination.

Figure 16.6 shows repeat-block transfer with an offset to the transfer destination.

In repeat-block transfer mode, a single block data is transferred by one transfer request.

A maximum of 1K data can be set as a total block transfer size using DMCRA of the DMACn.

A maximum of 64K can be set as the number of block transfer operations using DMCRB of the DMACn; therefore, a maximum of 64M data (1K data × 64K counts of block transfer operations) can be set as a total data transfer size.

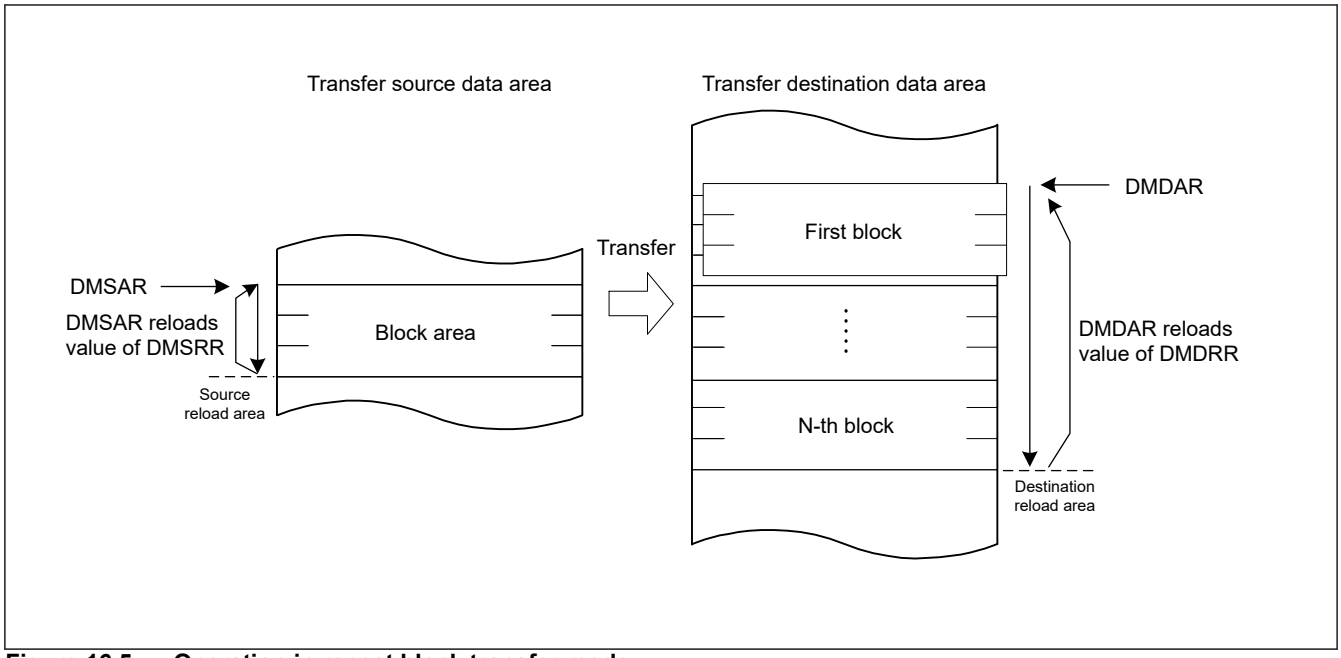



**Figure 16.5 Operation in repeat block transfer mode**

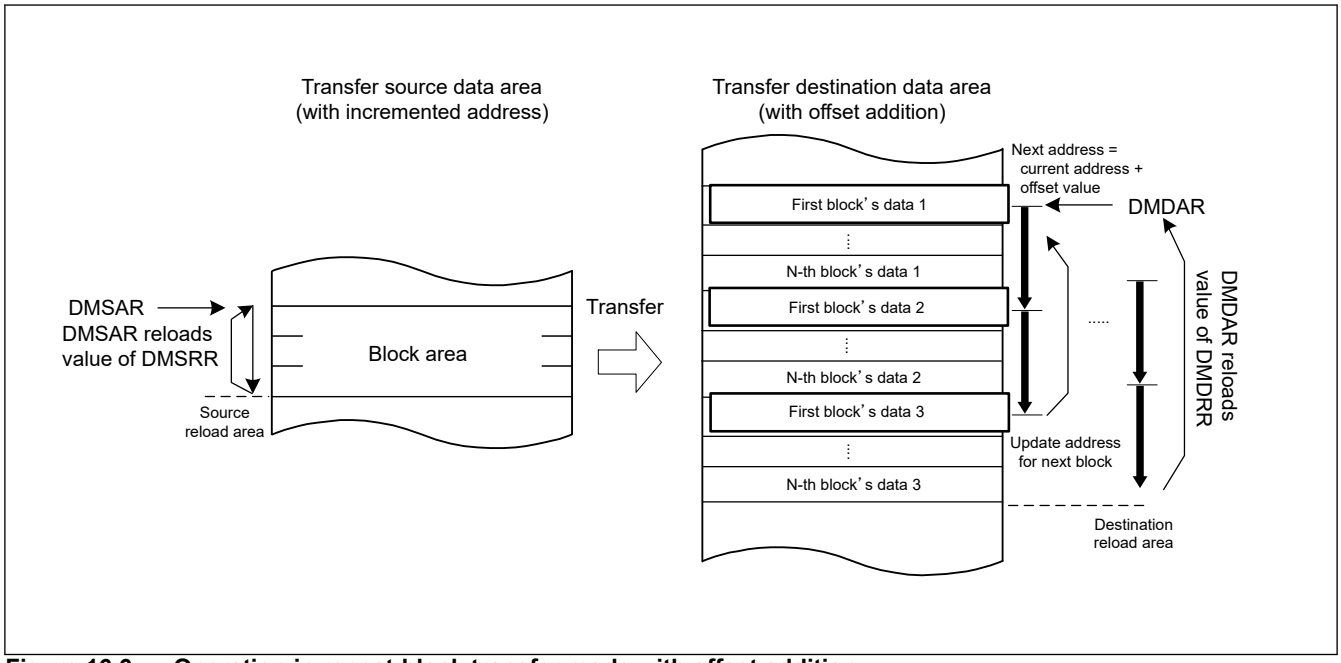



**Figure 16.6 Operation in repeat-block transfer mode with offset addition**

Table 16.8 to Table 16.13 summarize the register update operations in repeat-block transfer mode.

**Table 16.8 Register update operation associated with source area in repeat-block transfer mode (fixed address DMAMD.SM[1:0] = 00b) (1 of 2)**

| Register | Function | Update operation after single data is transferred | | |
|---|---|---|---|---|
| | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMSRR | Transfer source reload address | Not updated | Not updated | Not updated |
| DMSAR | Transfer source address | Not updated | Not updated | Not updated |
| DMCRAH[9:0] | Block size | Not updated | Not updated | Not updated |

**Table 16.8 Register update operation associated with source area in repeat-block transfer mode (fixed address DMAMD.SM[1:0] = 00b) (2 of 2)**

| Register | Function | Update operation after single data is transferred | | |
|---|---|---|---|---|
| | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMCRAL[15:0] | Block size count | Decremented by 1 | DMCRAH[9:0] | DMCRAH[9:0] |
| DMCRBH[15:0] | Number of block transfer operations | Not updated | Not updated | Not updated |
| DMCRBL[15:0] | Count of block transfer operations when DMTMD.TKP = 0 | Not updated | Decremented by 1 | 0 |
| | Count of block transfer operations when DMTMD.TKP = 1 | | | DMCRBH[15:0] |

**Table 16.9 Register update operation associated with destination area in repeat-block transfer mode (fixed address DMAMD.DM[1:0] = 00b)**

| Register | Function | Update operation after single data is transferred | | |
|---|---|---|---|---|
| | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMDRR | Transfer destination reload address | Not updated | Not updated | Not updated |
| DMDAR | Transfer destination address | Not updated | Not updated | Not updated |
| DMCRAH[9:0] | Block size | Not updated | Not updated | Not updated |
| DMCRAL[15:0] | Block size count | Decremented by 1 | DMCRAH[9:0] | DMCRAH[9:0] |
| DMCRBH[15:0] | Number of block transfer operations | Not updated | Not updated | Not updated |
| DMCRBL[15:0] | Count of block transfer operations when DMTMD.TKP = 0 | Not updated | Decremented by 1 | 0 |
| | Count of block transfer operations when DMTMD.TKP = 1 | | | DMCRBH[15:0] |

**Table 16.10 Register update operation associated with source area in repeat-block transfer mode (incremented or decremented address DMAMD.SM[1:0] = 10b or 11b) (1 of 2)**

| Register | Function | Update operation after single data is transferred | | | | | |
|---|---|---|---|---|---|---|---|
| | | DMSBSL[15:0] is not 1 | | | DMSBSL[15:0] is 1 | | |
| | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMSRR | Transfer source reload address | Not updated | Not updated | Not updated | Not updated | Not updated | Not updated |

**Table 16.10 Register update operation associated with source area in repeat-block transfer mode (incremented or decremented address DMAMD.SM[1:0] = 10b or 11b) (2 of 2)**

| Register | Function | Update operation after single data is transferred | | | | | |
|---|---|---|---|---|---|---|---|
| | | DMSBSL[15:0] is not 1 | | | DMSBSL[15:0] is 1 | | |
| | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMSAR | Transfer source address when DMTMD.SM[1:0] = 10b | Incremented by Data Size | | | DMSRR | | |
| | Transfer source address when DMTMD.SM[1:0] = 11b | Decremented by Data Size | | | DMSRR | | |
| DMCRAH[9:0] | Block size | Not updated | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMCRAL[15:0] | Block size count | Decremented by 1 | DMCRAH[9:0] | DMCRAH[9:0] | Decremented by 1 | DMCRAH[9:0] | DMCRAH[9:0] |
| DMSBSH[15:0] | Source buffer size (Repeat-size) | Not updated | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMSBSL[15:0] | Count of transfer data in source buffer | Decremented by 1 | Decremented by 1 | Decremented by 1 | DMSBSH | DMSBSH | DMSBSH |
| DMCRBH[15:0] | Number of block transfer operations | Not updated | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMCRBL[15:0] | Count of block transfer operations when DMTMD.TKP = 0 | Not updated | Decremented by 1 | 0 | Not updated | Decremented by 1 | 0 |
| | Count of block transfer operations when DMTMD.TKP = 1 | | | DMCRBH[15:0] | | | DMCRBH[15:0] |

**Table 16.11 Register update operation associated with destination area in repeat-block transfer mode (incremented or decremented address DMAMD.DM[1:0] = 10b or 11b) (1 of 2)**

| Register | Function | Update operation after single data is transferred | | | | | |
|---|---|---|---|---|---|---|---|
| | | DMDBSL[15:0] is not 1 | | | DMDBSL[15:0] is 1 | | |
| | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMDRR | Transfer destination reload address | Not updated | Not updated | Not updated | Not updated | Not updated | Not updated |

**Table 16.11 Register update operation associated with destination area in repeat-block transfer mode (incremented or decremented address DMAMD.DM[1:0] = 10b or 11b) (2 of 2)**

| Register | Function | Update operation after single data is transferred | | | | | |
|---|---|---|---|---|---|---|---|
| | | DMDBSL[15:0] is not 1 | | | DMDBSL[15:0] is 1 | | |
| | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMDAR | Transfer destination address when DMTMD.DM[1:0] = 10b | Incremented by Data Size | | | DMDRR | | |
| | Transfer destination address when DMTMD.DM[1:0] = 11b | Decremented by Data Size | | | DMDRR | | |
| DMCRAH[9:0] | Block size | Not updated | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMCRAL[15:0] | Block size count | Decremented by 1 | DMCRAH[9:0] | DMCRAH[9:0] | Decremented by 1 | DMCRAH[9:0] | DMCRAH[9:0] |
| DMDBSH[15:0] | Destina-tion buffer size (Repeat-size) | Not updated | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMDBSL[15:0] | Count of transfer data in destination buffer | Decremented by 1 | Decremented by 1 | Decremented by 1 | DMDBSH | DMDBSH | DMDBSH |
| DMCRBH[15:0] | Number of block transfer operations | Not updated | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMCRBL[15:0] | Count of block transfer operations when DMTMD.TKP = 0 | Not updated | Decremented by 1 | 0 | Not updated | Decremented by 1 | 0 |
| | Count of block transfer operations when DMTMD.TKP = 1 | | | DMCRBH[15:0] | | | DMCRBH[15:0] |

**Table 16.12 Register update operation associated with source area in repeat-block transfer mode (offset addition DMAMD.SM[1:0] = 01b) (1 of 2)**

| Register | Function | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | | | |
|---|---|---|---|---|---|---|
| | | | DMSBSL[15:0] is not 1 | | DMSBSL[15:0] is 1 | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMSRR | Transfer source reload address | Not updated | Not updated | Not updated | Not updated | Not updated |

**Table 16.12 Register update operation associated with source area in repeat-block transfer mode (offset addition DMAMD.SM[1:0] = 01b) (2 of 2)**

| Register | Function | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | | | |
|---|---|---|---|---|---|---|
| | | | DMSBSL[15:0] is not 1 | | DMSBSL[15:0] is 1 | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMSAR | Transfer source address when DMAMD.SADR = 0 | Offset addition by DMSBSH | DMSRR | | DMSRR | |
| | Transfer source address when DMAMD.SADR = 1 | | DMSRR + (DMS-BSH - DMSBSL) × DataSize | | | |
| DMCRAH[9:0] | Block size | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMCRAL[15:0] | Block size count | Decremented by 1 | DMCRAH[9:0] | DMCRAH[9:0] | DMCRAH[9:0] | DMCRAH[9:0] |
| DMSBSH[15:0] | Source buffer size (Repeat-size) | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMSBSL[15:0] | Count of transfer data in source buffer | Not updated | Decremented by 1 | Decremented by 1 | DMSBSH | DMSBSH |
| DMCRBH[15:0] | Number of block transfer operations | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMCRBL[15:0] | Count of block transfer operations when DMTMD.TKP = 0 | Not updated | Decremented by 1 | 0 | Decremented by 1 | 0 |
| | Count of block transfer operations when DMTMD.TKP = 1 | | | DMCRBH[15:0] | | DMCRBH[15:0] |

**Table 16.13 Register update operation associated with destination area in repeat-block transfer mode (offset addition DMAMD.DM[1:0] = 01b) (1 of 2)**

| Register | Function | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | | | |
|---|---|---|---|---|---|---|
| | | | DMDBSL[15:0] is not 1 | | DMDBSL[15:0] is 1 | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMDRR | Transfer destination reload address | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMSAR | Transfer destination address when DMAMD.DADR = 0 | Offset addition by DMDBSH | DMDRR | | DMDRR | |
| | Transfer destination address when DMAMD.DADR = 1 | | DMDRR + (DMDBSH - DMDBSL) × DataSize | | | |
| DMCRAH[9:0] | Block size | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMCRAL[15:0] | Block size count | Decremented by 1 | DMCRAH[9:0] | DMCRAH[9:0] | DMCRAH[9:0] | DMCRAH[9:0] |

**Table 16.13 Register update operation associated with destination area in repeat-block transfer mode (offset addition DMAMD.DM[1:0] = 01b) (2 of 2)**

| Register | Function | DMCRAL[15:0] is not 1 | DMCRAL[15:0] is 1 (single block is transferred) | | | |
|---|---|---|---|---|---|---|
| | | | DMDBSL[15:0] is not 1 | | DMDBSL[15:0] is 1 | |
| | | | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 | DMCRBL[15:0] is not 1 | DMCRBL[15:0] is 1 |
| DMDBSH[15:0] | Destina-tion buffer size (Repeat-size) | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMDBSL[15:0] | Count of transfer data in destination buffer | Not updated | Decremented by 1 | Decremented by 1 | DMDBSH | DMDBSH |
| DMCRBH[15:0] | Number of block transfer operations | Not updated | Not updated | Not updated | Not updated | Not updated |
| DMCRBL[15:0] | Count of block transfer operations when DMTMD.TKP = 0 | Not updated | Decremented by 1 | 0 | Decremented by 1 | 0 |
| | Count of block transfer operations when DMTMD.TKP = 1 | | | DMCRBH[15:0] | | DMCRBH[15:0] |

## 16.3.2 Extended Repeat Area Function

The DMAC supports a function to specify the extended repeat areas on the transfer source and destination addresses. With the extended repeat areas set, the address registers repeatedly indicate the addresses of the specified extended repeat areas.

The extended repeat areas can be specified separately to the transfer source address register (DMSAR) and transfer destination address register (DMDAR).

The extended repeat area on the source address is specified by the DMAMD.SARA[4:0] bits. The extended repeat area on the destination address is specified by the DMAMD.DARA[4:0] bits. The size can be specified separately for the source and destination sides.

However, the area (of transfer source or transfer destination) which is specified as the repeat area or block area should not be specified as the extended repeat area.

When the address register value reaches the end address of the extended repeat area and the extended repeat area overflows, DMA transfer is stopped and an interrupt by an extended repeat area overflow can be requested. When an overflow occurs in the extended repeat area on the transfer source while the DMINT.SARIE bit is set to 1, the DMSTS.ESIF flag is set to 1 and the DMCNT.DTE bit is cleared to 0 to stop DMA transfer. At this time, if the DMINT.ESIE bit is set to 1, an interrupt by an extended repeat area overflow is requested. When the DMINT.DARIE bit is set to 1, the destination address register becomes a target to apply the function. DMA transfer can be resumed by writing 1 to the DMCNT.DTE bit in the interrupt handling.

Figure 16.7 shows an example of the extended repeat area operation.

Eight bytes are specified as an extended repeat area by the lower three bits of DMSAR (DMAMD.SARA[4:0] bits = 00011b). The data size is eight bits (DMTMD.SZ[1:0] = 00b).

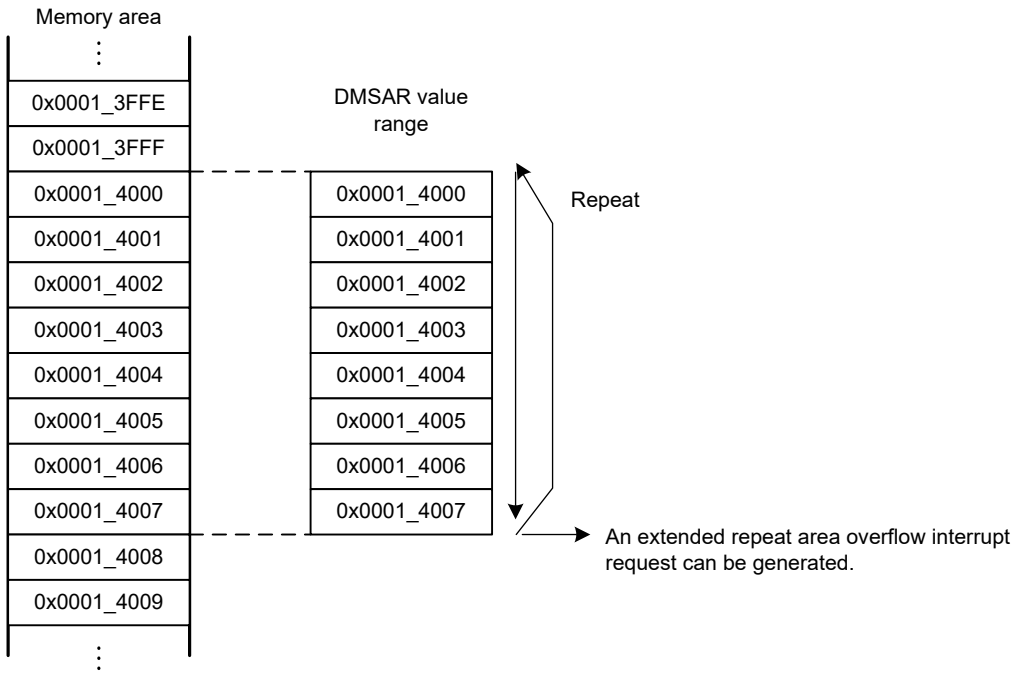



**Figure 16.7 Example of extended repeat area operation**

When an interrupt by an extended repeat area overflow is used in block transfer mode, the following should be taken into consideration.

When a transfer is stopped by an interrupt by an extended repeat area overflow, the address register must be set so that the block size is a power of 2 or the block size boundary is aligned with the extended repeat area boundary. When an overflow on the extended repeat area occurs during a transfer of one block, the interrupt by the overflow is suspended until transfer of the block is completed, and the transfer overruns.

Figure 16.8 shows an example when the extended repeat area function is used in block transfer mode.

Eight bytes are specified as an extended repeat area by the lower three bits of DMSAR (DMAMD.SARA[4:0] bits = 00011b), block transfer mode with block size 5 is set (DMCRA = 0x0005_0005), and transfer source address is not specified as a block area. Data size is eight bits (DMTMD.SZ[1:0] bits = 00b).

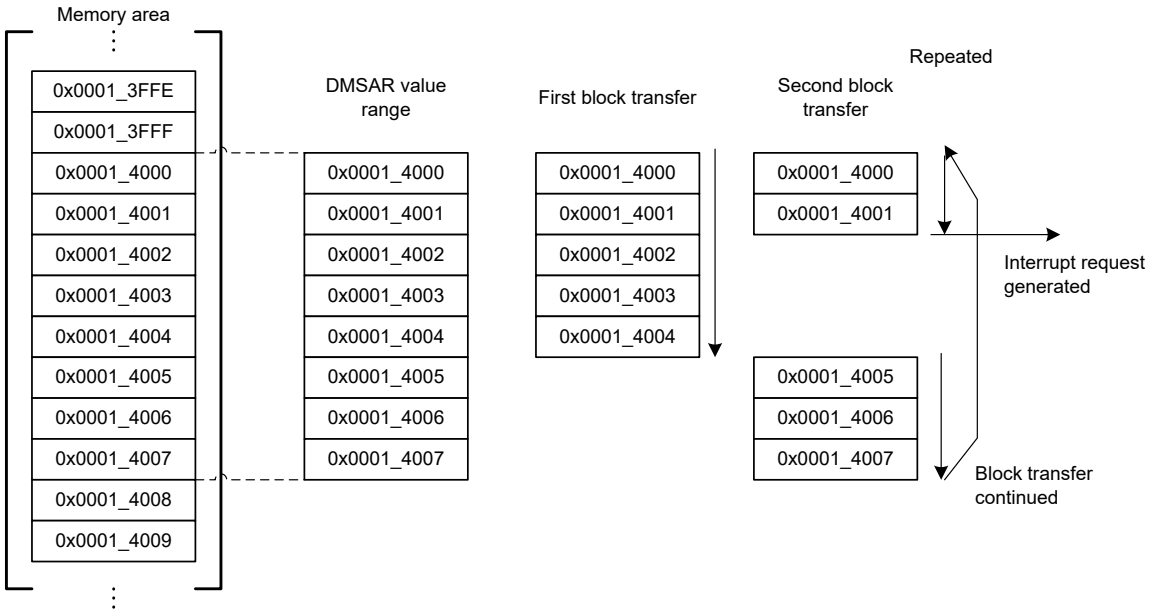



**Figure 16.8 Example of extended repeat area function in block transfer mode**

## 16.3.3 Free-running Function

The DMAC supports free-running function. This function allows to transfer repeatedly without reconfiguring in interrupt handler.

### 16.3.3.1 In Normal Transfer Mode

In normal transfer mode, when DMCRA.DMCRAL bits are set to 0x0000, no specific number of transfer operations is set; data transfer is performed with the transfer counter stopped.

For more information, see section 16.3.1.1. Normal Transfer Mode.

### 16.3.3.2 In Other Transfer Modes

In repeat, block and repeat-block transfer mode, the DMAC supports free-running function using the DMTMD.TKP bit. If the DMTMD.TKP bit is to be set to 1, the transfer is not stopped by completion of specified total number of transfer operations and reloads DMCRBH repeatedly.

Figure 16.9 show an example of block transfer operation without free-running function.

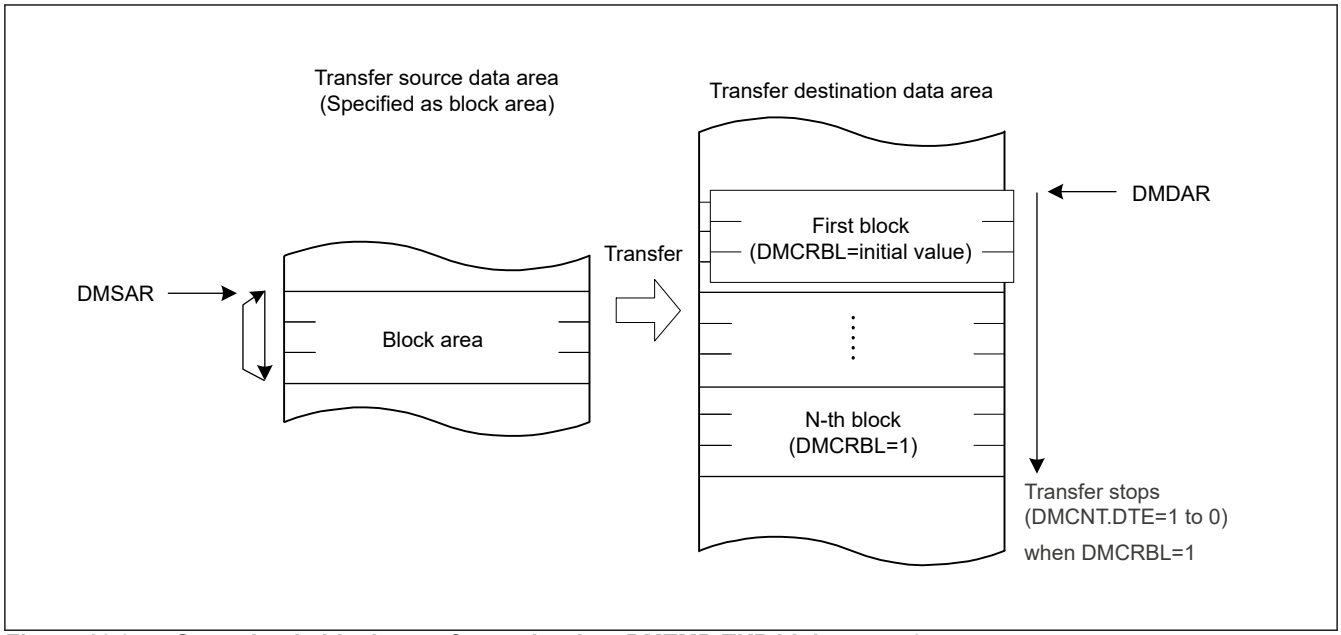



**Figure 16.9 Operation in block transfer mode when DMTMD.TKP bit is set to 0**

Figure 16.10 show an example of block transfer operation with free-running function.

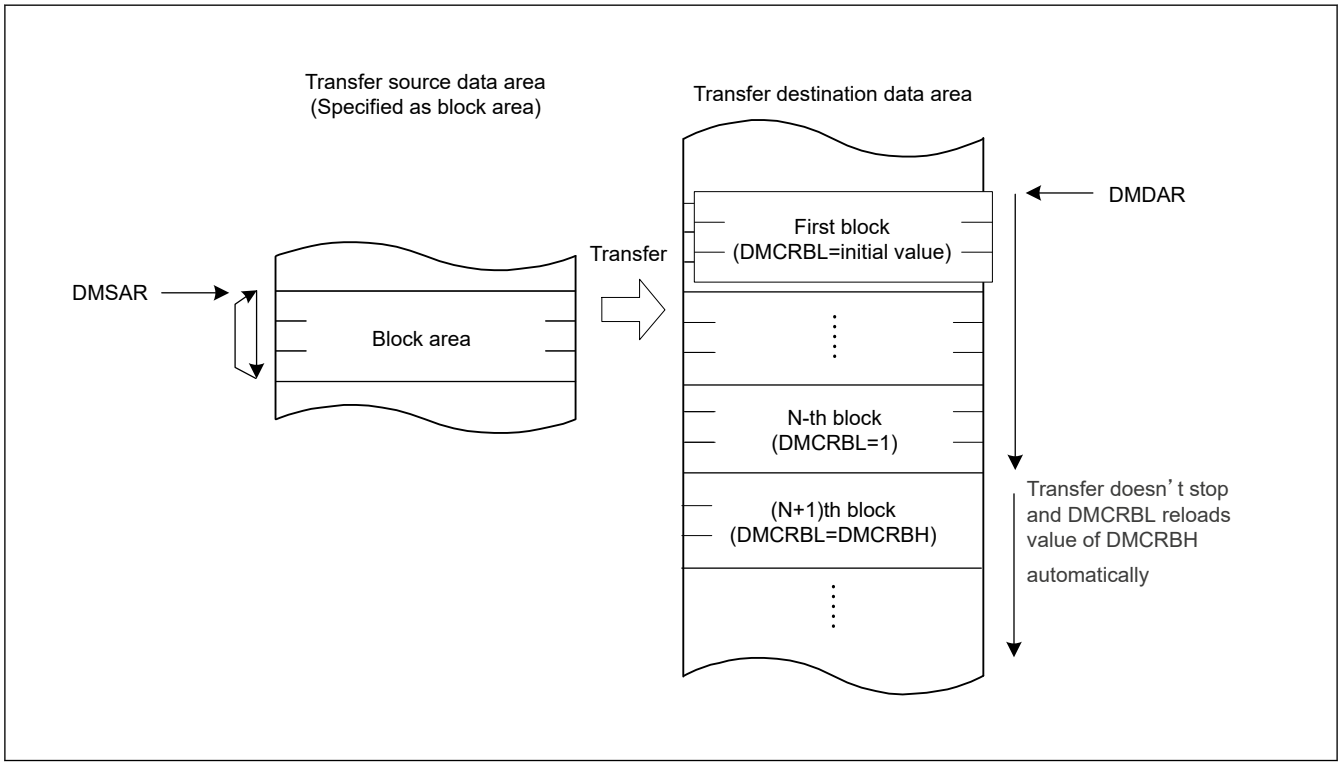



**Figure 16.10 Operation in block transfer mode when DMTMD.TKP bit is set to 1**

## 16.3.4 Address Update Function using Offset

The source and destination addresses can be updated by fixing, increment, decrement, or offset addition. In normal, repeat and block transfer mode, when the offset addition is selected, the offset specified by the DMA offset register (DMOFR) is added to the address every time the DMAC performs one data transfer. This function realizes a data transfer where addresses are allocated to separated areas.

Offset subtraction can also be realized by setting a negative value in DMOFR. In this case, the negative value must be 2's complement.

DMSBS or DMDBS are used instead of DMOFR in repeat-block transfer mode. For more information section 16.3.1.4. Repeat-Block Transfer Mode

Table 16.14 shows the address update method in each address update mode.

**Table 16.14 Address update method in each address update mode**

| Address update mode | Settings of DMAMD.SM[1:0] and DMAMD.DM[1:0] for address update modes | Address update method (for different SZ[1:0] settings in DMTMD) | | |
|---|---|---|---|---|
| | | SZ[1:0] = 00b | SZ[1:0] = 01b | SZ[1:0] = 10b |
| Address fixed | 00b | Fixed | | |
| Offset addition | 01b | +DMOFR*1 | | |
| Increment | 10b | +1 | +2 | +4 |
| Decrement | 11b | –1 | –2 | –4 |

Note 1. When setting a negative value in the DMA Offset Register, the value must be in two's complement, obtained by the following formula:
two's complement of a negative offset value = ~ (offset) + 1 (~ = bit inversion)

### 16.3.4.1 Basic Transfer Using Offset Addition

Figure 16.11 shows an example of address updating using offset addition.

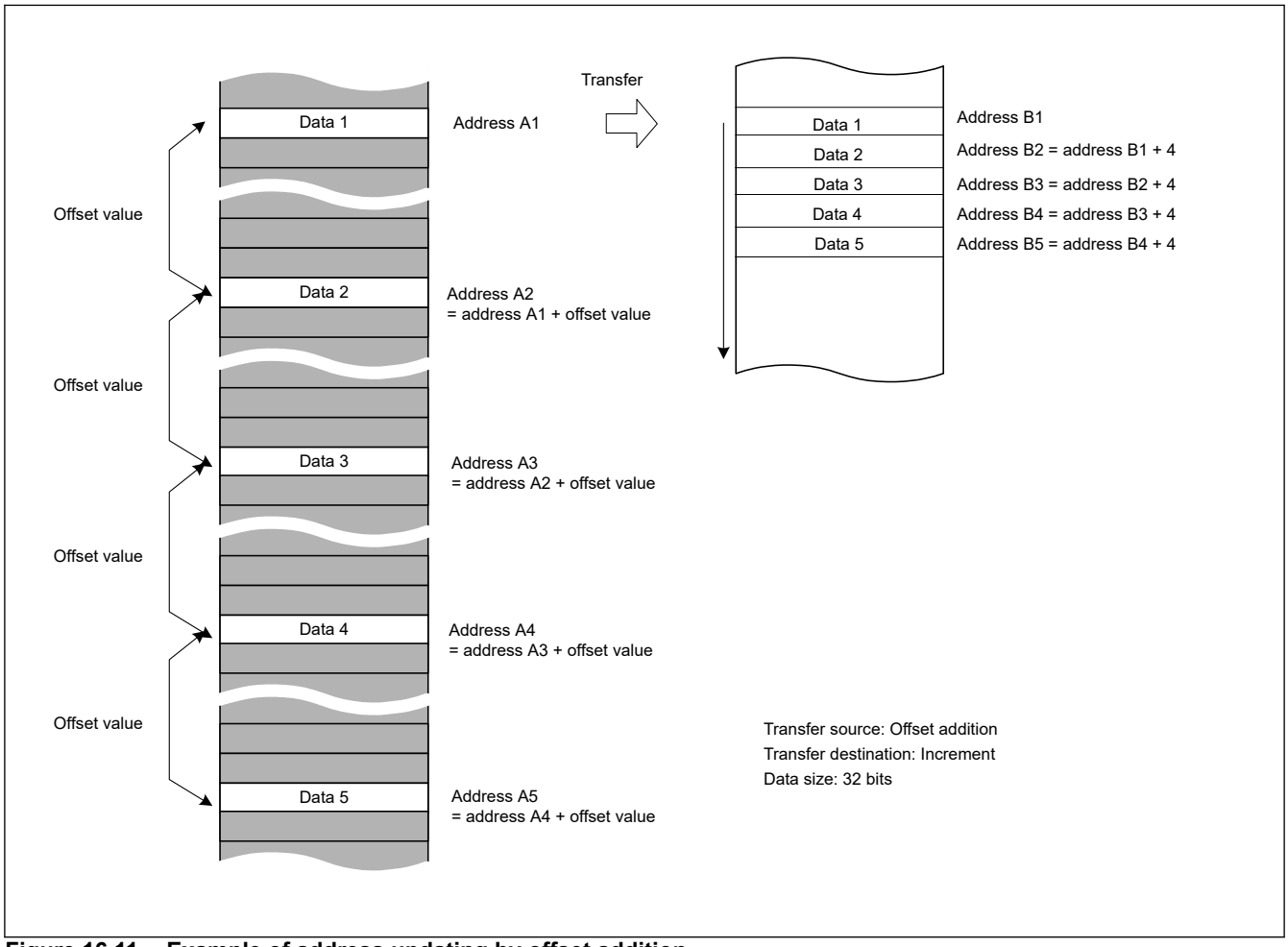



**Figure 16.11 Example of address updating by offset addition**

Figure 16.11 shows the setting of the following.

- The transfer data is 32 bits long.
- Offset addition is set as the transfer source address update mode.

- Increment is set as the transfer destination address update mode.

The second and subsequent data is each read from the transfer source address obtained by adding the offset value to the previous address. The data read from the addresses at the specified intervals is written to the continuous locations on the destination.

### 16.3.4.2 Example of XY Conversion Using Offset Addition

Figure 16.12 shows the XY conversion using offset addition in repeat transfer mode.

Settings are as follows:

- DMAMD.SM — Transfer source address update mode: Offset addition.
- DMAMD.DM — Transfer destination address update mode: Destination address is incremented.
- DMTMD.SZ — Transfer data size select: 32 bits.
- DMTMD.MD — Transfer mode select: Repeat transfer.
- DMTMD.DTS — Repeat area select: The source is specified as the repeat area.
- DMOFR — Offset address: 0x10.
- DMCRA — Repeat size: 0x4.
- DMINT.RPTIE — The repeat size end interrupt is enabled.

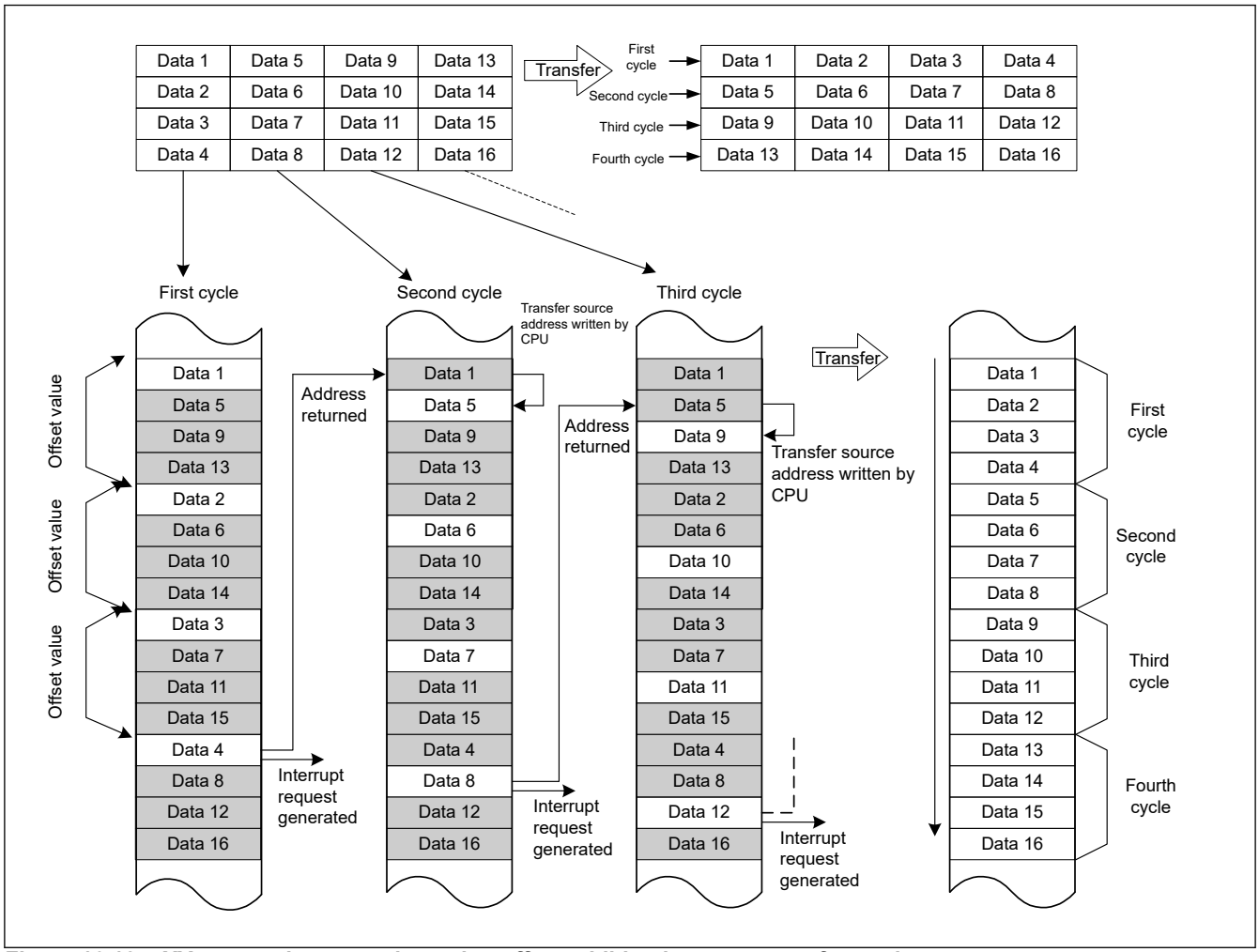



**Figure 16.12 XY conversion operation using offset addition in repeat transfer mode**

When a transfer starts, the offset value is added to the transfer source address every time data is transferred. The transfer data is written to continuous transfer destination addresses. When data 4 is transferred:

- The repeat size of data transfer is complete.

- The transfer source address returns to the transfer start address (the address of data 1 on the transfer source).
- A repeat size end interrupt is requested.

During the time this interrupt pauses the transfer, the following operations are performed.

- DMSAR — Rewrite the DMA transfer source address to the address of data 5
  (with the above example, the data 1 address + 4).
- DMCNT — Set the DTE bit to 1.

The DMA transfer is resumed from the state when the DMA transfer is stopped. After that, the operations described above are repeated until the transfer source data is transposed to the destination area (XY conversion).

Figure 16.13 shows a flowchart of the XY conversion.

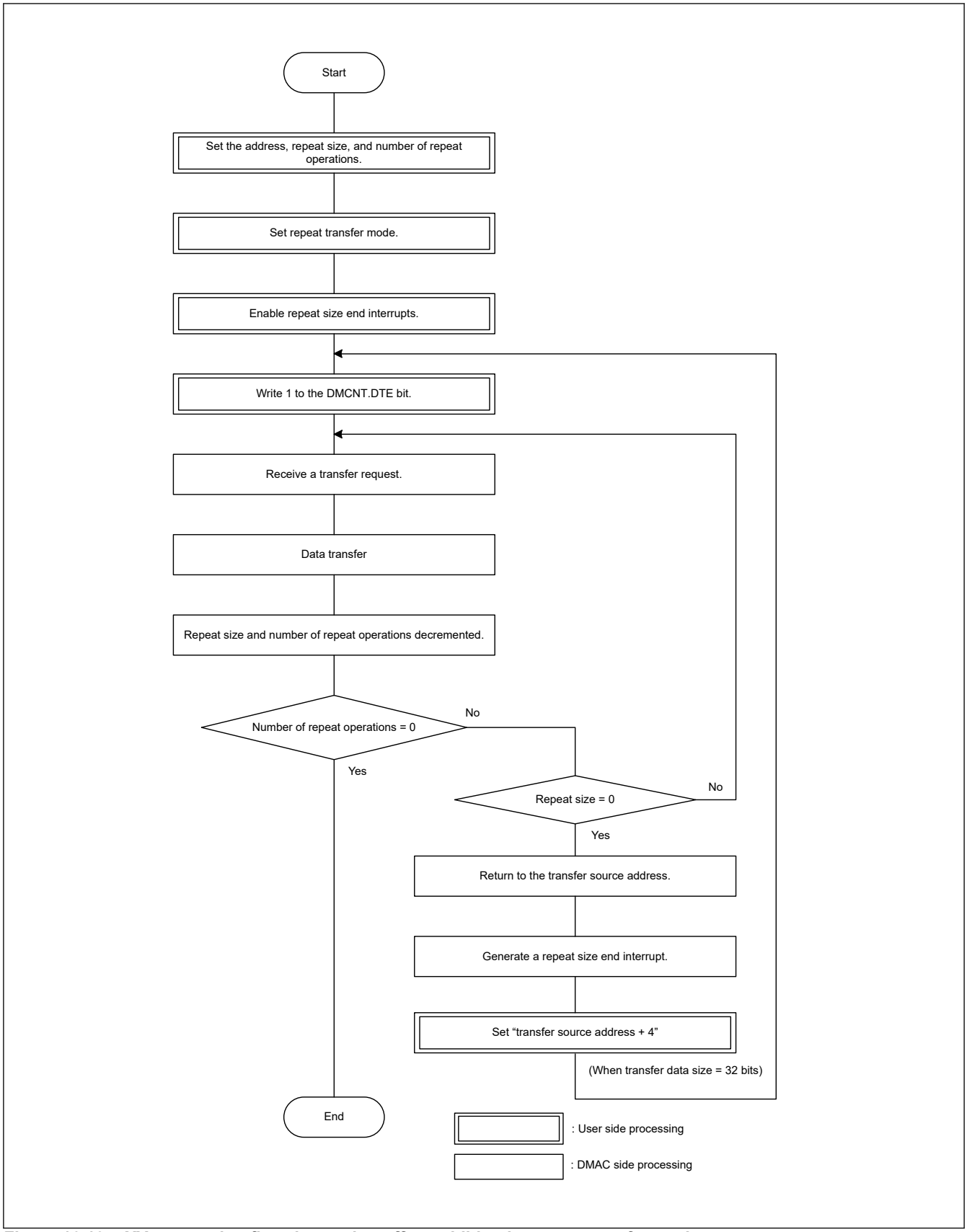



**Figure 16.13 XY conversion flowchart using offset addition in repeat transfer mode**

### 16.3.5 Address Update Function in Repeat-Block Transfer Mode

Repeat-block transfer mode is an extension of repeat transfer mode and block transfer mode. However, the detailed behavior of the address update is different from these two modes. Here are the details of the address update function in repeat-block transfer mode.

#### 16.3.5.1 Fixed Address Mode

When DMAMD.SM[1:0] is set to 00b, the address update mode of the source is fixed address. And when DMAMD.DM[1:0] is set to 00b, the address update mode of the destination is fixed address.

In fixed address, the address is not updated from the initial value of DMSAR and DMDAR. If the block size (DMCRA) is larger than 1, the same data will be transferred multiple times for one request.

Figure 16.14 shows address update in fixed address.

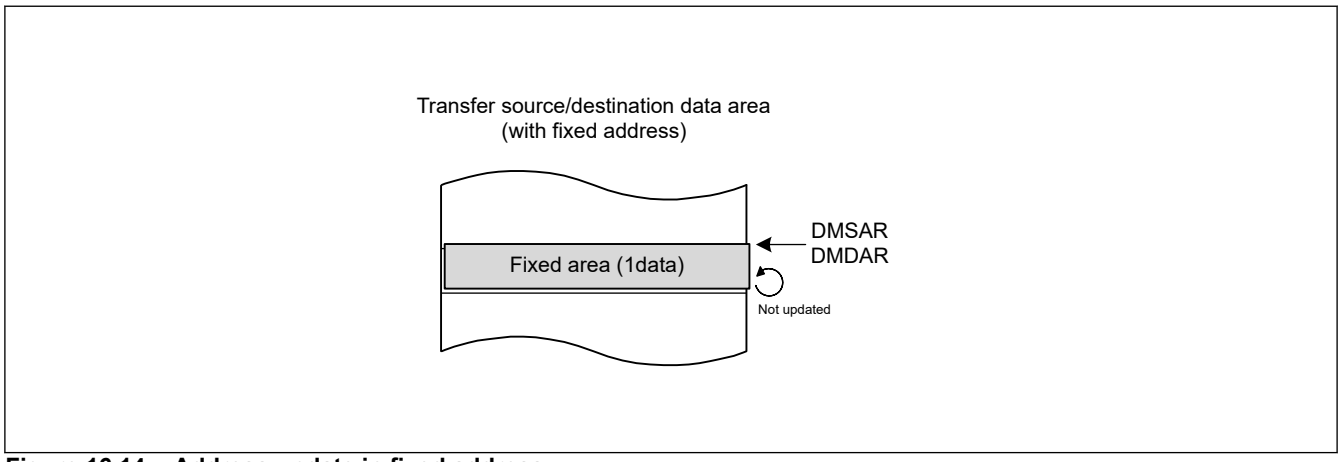



**Figure 16.14 Address update in fixed address**

#### 16.3.5.2 Incremental and Decremental Address Mode

When DMAMD.SM[1:0] is set to 10b, the address update mode of the source is incremental address. And when DMAMD.DM[1:0] is set to 10b, the address update mode of the destination is incremental address. When DMAMD.SM[1:0] is set to 11b, the address update mode of the source is decremental address. And when DMAMD.DM[1:0] is set to 11b, the address update mode of the destination is decremental address.

In these update modes, the address is incremented or decremented according to the setting of DMTMD.SZ[1:0].

In these update modes DMSBS and DMDBS indicates a reload area. The unit of DMSBS and DMDBS is "number of data". At the start of transfer, DMSBSL and DMDBSL, which is the lower 16 bits of DMSBS and DMDBS, operates as a down counter and decrements each time one data transfer is performed. When the value becomes 1, DMSAR and DMDAR reloads the value of DMSRR and DMDRR.

Figure 16.15 shows address update in incremental address.

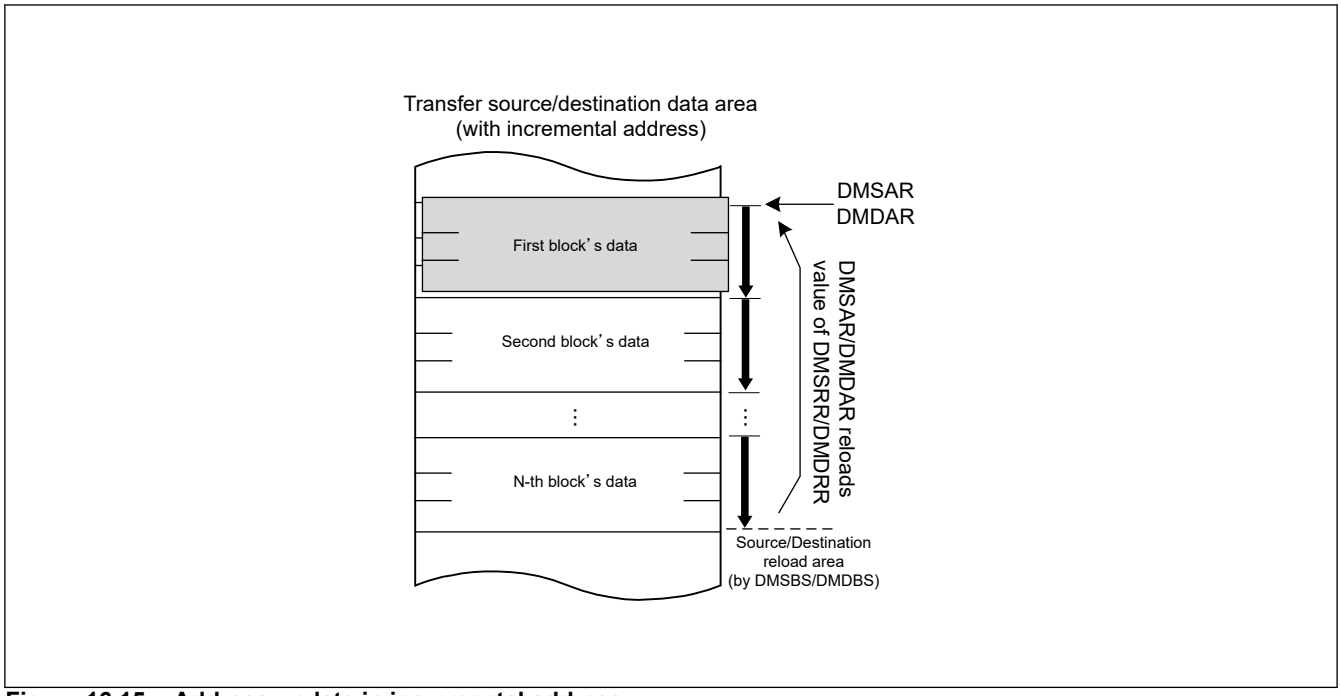



**Figure 16.15 Address update in incremental address**

### 16.3.5.3 Offset Addition Mode

When DMAMD.SM[1:0] is set to 01b, the address update mode of the source is offset addition. And when DMAMD.DM[1:0] is set to 01b, the address update mode of the destination is offset addition.

In offset addition, DMSBS and DMDBS indicates reload area and also works as an access offset value. Unlike other transfer modes, DMOFR register is not used in repeat-block transfer mode. In offset addition, the unit of DMSBS and DMDBS is the number of blocks. When the transfer starts, DMCRAL operates as a down counter, DMSAR and DMDAR reloads the value of DMSRR and DMDRR every time one block is transferred. In addition, DMSBSL and DMDBSL, which is the lower 16 bits of DMSBS and DMDBS, also operates as a down counter and decrements every time one block is transferred. When the DMSBS and DMDBS value becomes 1, DMSAR and DMDAR reloads the value of DMSRR and DMDRR.

When DMAMD.SADR and DMAMD.DADR is set to 0, offset addition operation of the same area is repeated. DMDAR only reloads DMDRR. Figure 16.16 shows address update in offset addition with DMAMD.SADR and DMAMD.DADR=0.

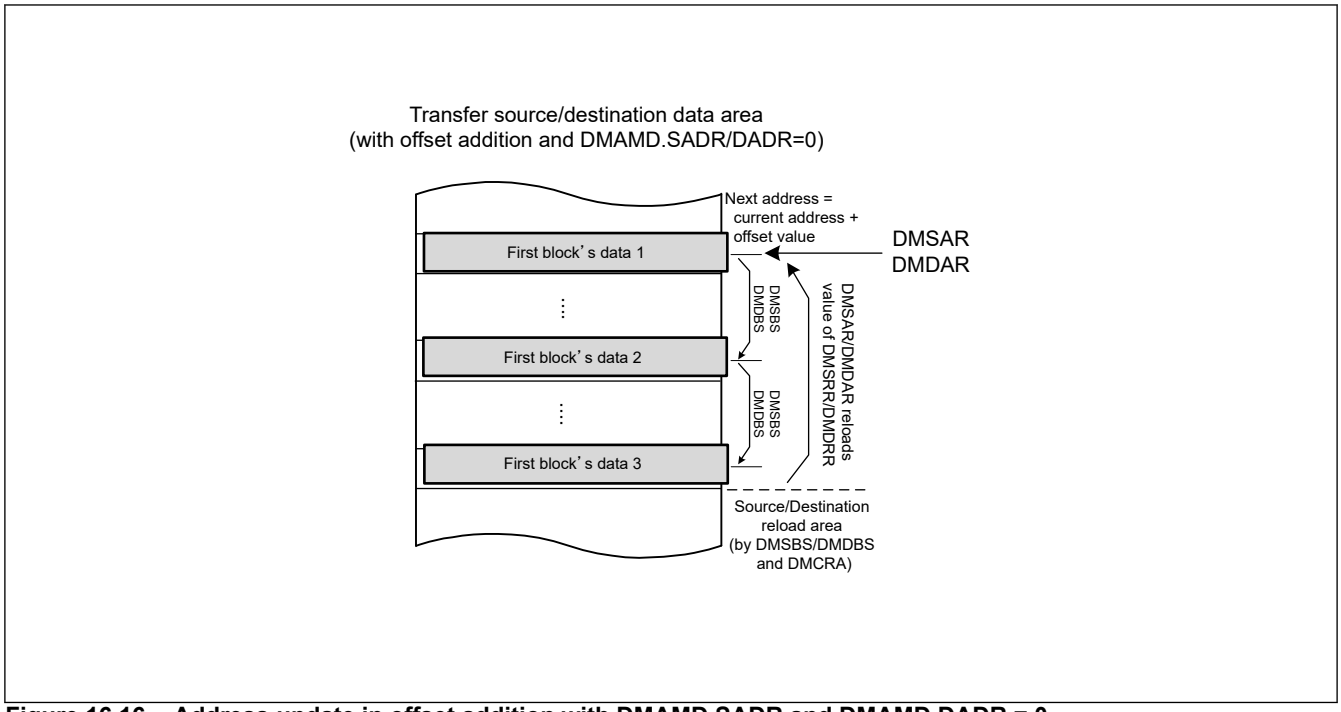



**Figure 16.16 Address update in offset addition with DMAMD.SADR and DMAMD.DADR = 0**

When DMAMD.SADR and DMAMD.DADR is set to 1, the address is incremented by one data unit after DMSRR and DMDRR is reloaded by DMCRAL=1. In other words, an index value ((DMDBSH-DMDBSL) × DataSize) is added to DMDAR after DMDRR is reloaded. This behavior is used to implement multiple ring buffers. Figure 16.17 shows address update in offset addition with DMAMD.SADR and DMAMD.DADR=1.

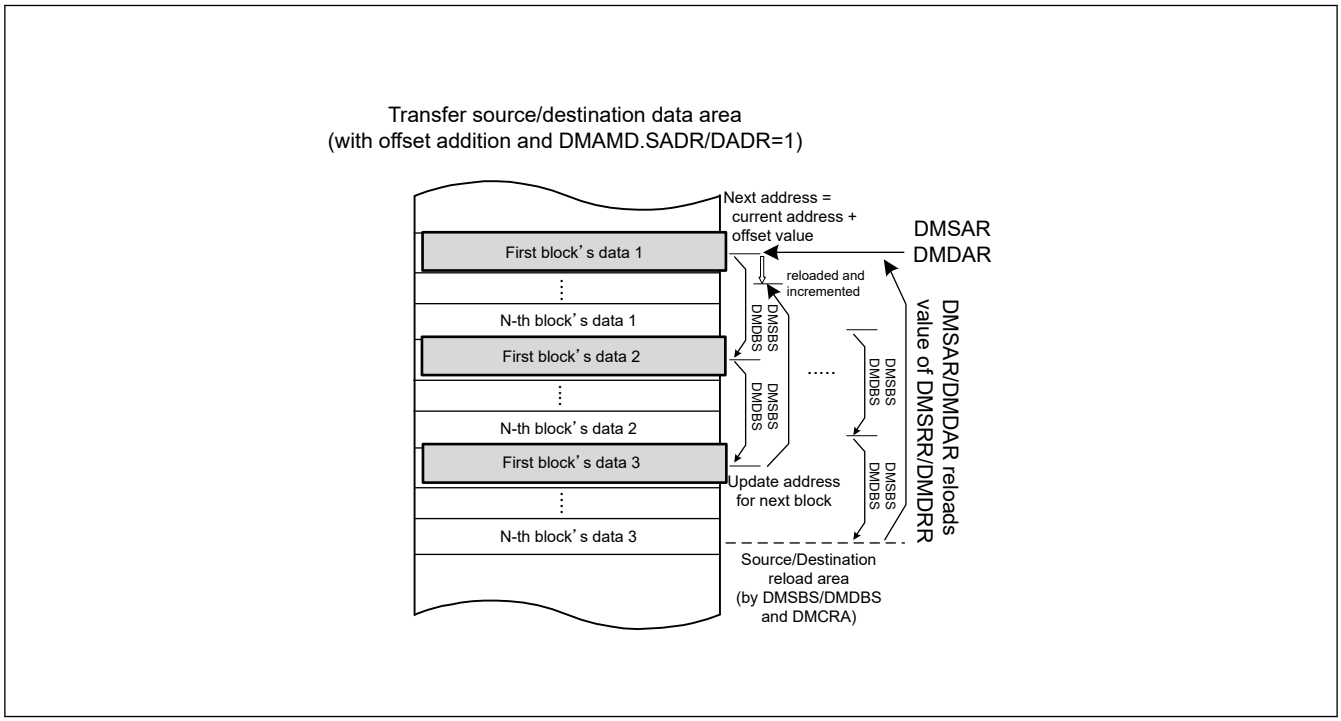



**Figure 16.17 Address update in offset addition with DMAMD.SADR and DMAMD.DADR = 1**

## 16.3.6 Example of Using Repeat-Block Transfer Mode

In repeat-block transfer mode, it is possible to realize repeated access to interval data and single or multiple ring buffers by combining the above address update modes. Following sections shows some usage examples.

### 16.3.6.1 Interval Address to Single Ring Buffer

Figure 16.18 shows an example of reading interval ADDRn registers (data register) of ADC12 module and storing it in single ring buffer. It transfers 2 data every 4 halfwords per 1 request. DMSAR is incremented by one data every one request. This can be achieved by setting the transfer source to offset addition and DMAMD.SADR=1, the block size (DMCRA) to 2, and the transfer source offset (DMSBS) to 4. Table 16.15 shows setting of this example.

**Table 16.15 Setting of use case: from interval address to single ring buffer**

| Register | Value | Description |
|---|---|---|
| DMSAR, DMSRR | 0x4017_0020 | Initial source address |
| DMDAR, DMDRR | 0x2000_0000 | Initial destination address |
| DMTMD.SZ[1:0] | 01b | Data size is halfword |
| DMAMD.SADR | 1 | Incremental source address after reloading |
| DMAMD.SM[1:0] | 01b | Source update mode is offset addition |
| DMAMD.DM[1:0] | 10b | Destination update mode is incremental address |
| DMCRAH, DMCRAL | 2 | Transfer block size |
| DMSBSH, DMSBSL | 4 | Source whole buffer size (unit is ‘blocks’) and Source access offset (unit is ‘data’) |
| DMDBSH, DMDBSL | N × 2 (DMCRA) | Destination buffer size (unit is ‘data’) |

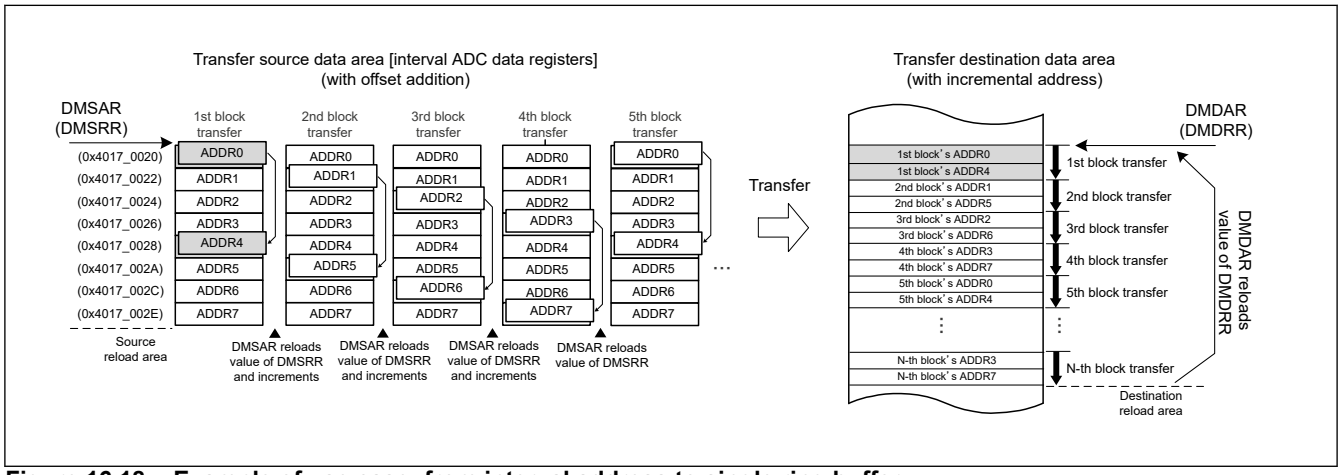



**Figure 16.18 Example of use case: from interval address to single ring buffer**

### 16.3.6.2 Unaligned Ring Buffer to Single Ring Buffer

Figure 16.19 shows an example of reading ADBUFn registers of ADC12 module (conversion result storage ring buffer) incrementally and storing it in single ring buffer. In this example, wrapping occurs because ADBUFn overflows in the fourth scan, but transfer source address of DMAC is also updated accordingly. This can be realized by setting the transfer source to incremental address and setting the DMSBS register to 16 which is the length of ADBUFn. This makes it possible to continue transfer without performing CPU processing using interrupts. Table 16.16 shows setting of this example.

**Table 16.16 Setting of use case: from unaligned ring buffer to single ring buffer (1 of 2)**

| Register | Value | Description |
|---|---|---|
| DMSAR, DMSRR | 0x4017_00B0 | Initial source address |
| DMDAR, DMDRR | 0x2000_0000 | Initial destination address |
| DMTMD.SZ[1:0] | 01b | Data size is halfword |
| DMAMD.SM[1:0] | 10b | Source update mode is incremental address |
| DMAMD.DM[1:0] | 10b | Destination update mode is incremental address |

**Table 16.16 Setting of use case: from unaligned ring buffer to single ring buffer (2 of 2)**

| Register | Value | Description |
|---|---|---|
| DMCRAH, DMCRAL | 5 | Transfer block size |
| DMSBSH, DMSBSL | 16 | Source buffer size (unit is ‘data’) |
| DMDBSH, DMDBSL | N × 5(DMCRA) | Destination buffer size (unit is ‘data’) |

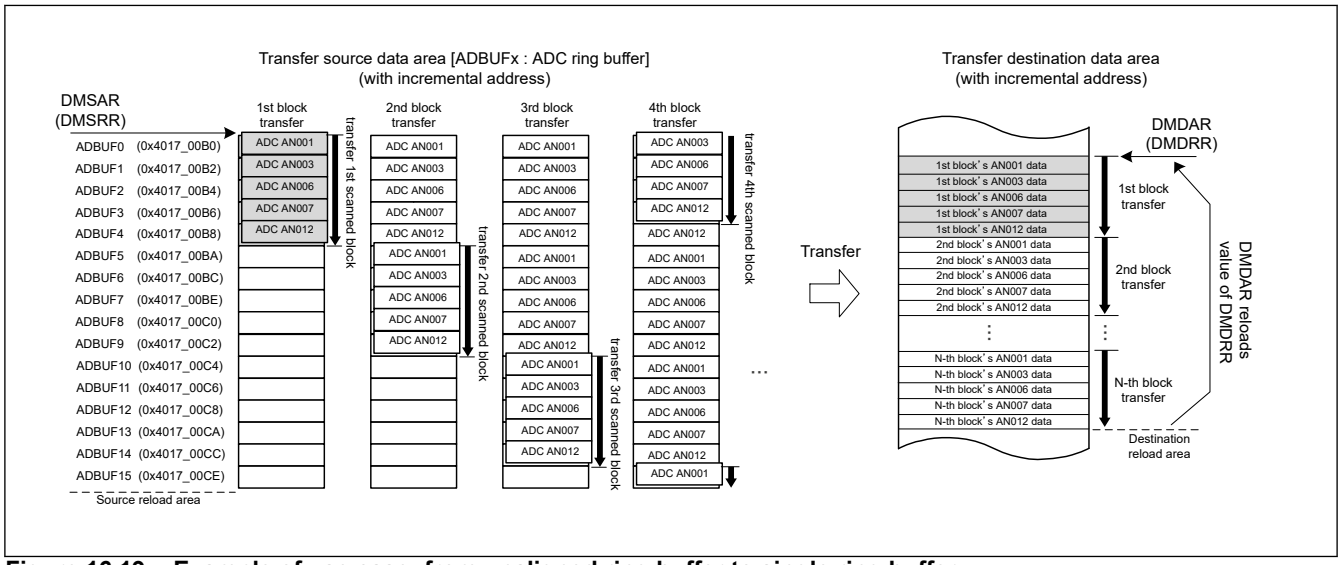



**Figure 16.19 Example of use case: from unaligned ring buffer to single ring buffer**

### 16.3.6.3 Single Block to Multi Ring Buffer

Figure 16.20 shows an example of storing the continuous ADDRn registers (data register) of ADC12 module individually in multiple ring buffers. In this example, a ring buffer in which only the first element (ADDR0) in a single block is arranged in transfer order is created at the destination. Also, in the next area, create a ring buffer in which only the second element (ADDR1) is arranged in transfer order. In the following case, create a ring buffer of length N, which is defined by DMDBS. And the number of data elements in the block is 3, which is defined by DMCRA. Table 16.17 shows setting of this example.

**Table 16.17 Setting of use case: from single block to multi ring buffer**

| Register | Value | Description |
|---|---|---|
| DMSAR, DMSRR | 0x4017_0020 | Initial source address |
| DMDAR, DMDRR | 0x2000_0000 | Initial destination address |
| DMTMD.SZ[1:0] | 01b | Data size is halfword |
| DMAMD.DADR | 1 | Incremental destination address after reloading |
| DMAMD.SM[1:0] | 10b | Source update mode is incremental address |
| DMAMD.DM[1:0] | 01b | Destination update mode is offset addition |
| DMCRAH, DMCRAL | 3 | Transfer block size |
| DMSBSH, DMSBSL | 3 | Source buffer size (unit is ‘data’) |
| DMDBSH, DMDBSL | N | Destination whole buffer size (unit is ‘blocks’) and Destination access offset (unit is ‘data’) |

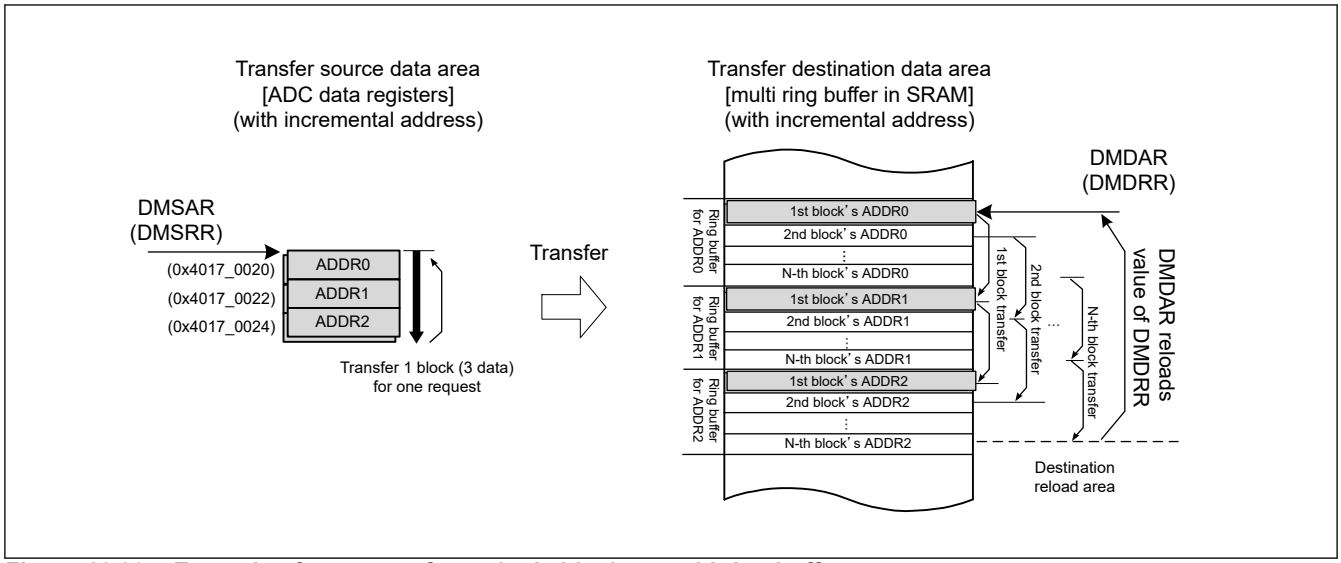



**Figure 16.20 Example of use case: from single block to multi ring buffer**

## 16.3.7 Activation Sources

Software, interrupt requests from the peripheral modules, and external interrupt requests can all be specified as DMAC activation sources. Set the DMTMD.DCTG[1:0] bits to select the activation source.

### 16.3.7.1 DMAC Activation by Software

When DMA transfer is started by software, follow below procedure.

1. Set the DMTMD.DCTG[1:0] bits to 00b.
2. Set the DMCNT.DTE bit to 1 (DMA transfer is enabled).
3. Set the DMAST.DMST bit set to 1 (DMAC activation enabled).
4. Set the DMREQ.SWREQ bit to 1 (DMA requested).

When the DMAC is activated by software while the DMREQ.CLRS bit is 0, the DMREQ.SWREQ bit is cleared to 0 after data transfer is started in response to a DMA transfer request.

When the DMAC is activated by software while the CLRS bit is 1, the SWREQ bit is not cleared to 0 after data transfer is started. In this case, a DMA transfer request is issued again after completion of a transfer.

### 16.3.7.2 DMAC Activation through Interrupt Requests from On-Chip Peripheral Modules or External Interrupt Requests

You can specify interrupt requests from on-chip peripheral modules and external interrupt requests as DMAC activation sources. The activation sources can be selected individually for each channel in ICU.DELSRn.DELS[8:0] (n = 0 to 7).

To start DMA transfer through an interrupt request from an on-chip peripheral module or an external interrupt request, follow the procedures as indicated below.

1. Set ICU.DELSRn.DELS[8:0] (n = 0 to 7) to the event number (select the DMAC event link).
2. Set the DMTMD.DCTG[1:0] bits to 01b (interrupts from the peripheral modules and the external interrupt pins).
3. Set the DMCNT.DTE bit to 1 (enable DMA transfer).
4. Set the DMAST.DMST bit set to 1 (DMAC activation enabled).

For interrupt requests specified as DMAC activation sources, see Table 13.3, in section 13, Interrupt Controller Unit (ICU).

## 16.3.8 Operation Timing

The following timing charts show the minimum number of execution cycles.

Figure 16.21 and Figure 16.22 show DMAC operation timing examples.

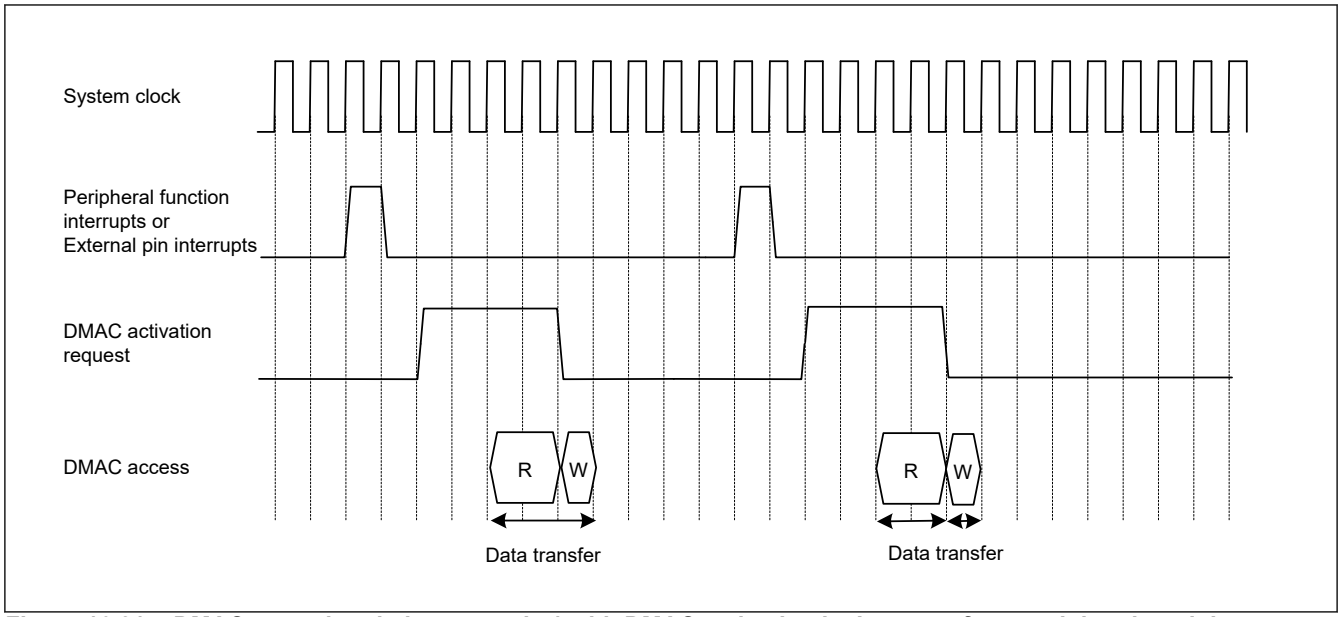



**Figure 16.21 DMAC operation timing example 1 with DMAC activation by Interrupt from peripheral module or external interrupt input pin, in normal transfer mode or repeat transfer mode**

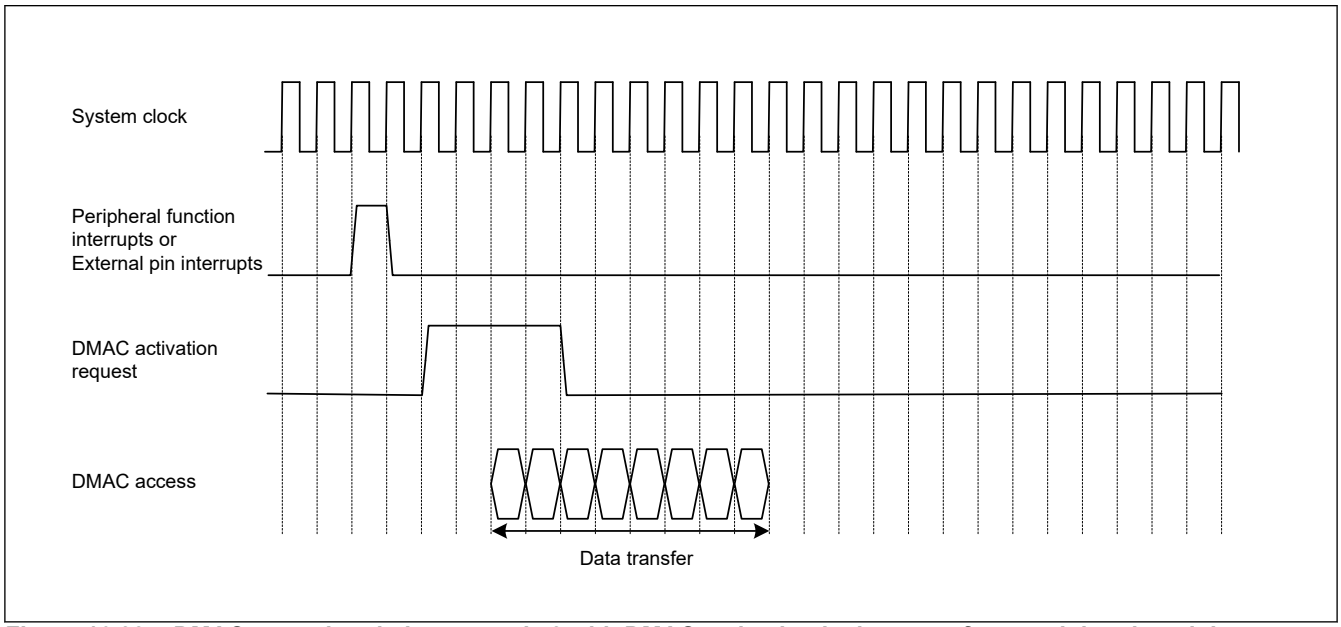



**Figure 16.22 DMAC operation timing example 2 with DMAC activation by interrupt from peripheral module or external interrupt input pin, in block transfer mode with block size = 4**

## 16.3.9 DMAC Execution Cycles

Table 16.18 lists execution cycles in one DMAC data transfer operation.

**Table 16.18 DMAC execution cycles**

| Transfer mode | Data transfer (read) | Data transfer (write) |
|---|---|---|
| Normal | Cr+1 | Cw |
| Repeat | Cr+1 | Cw |
| Block*1 | P × Cr | P × Cw |

Note: P: Block size (DMCRAH register setting)
Cr: Data read destination access cycle
Cw: Data write destination access cycle

Note 1. This is the case when the block size is 2 or more. When the block size is 1, normal transfer cycle is applied.

Cr and Cw depend on the access destination. For the number of cycles for each access destination, see section 42, SRAM, section 44, Flash Memory, and section 14, Buses. The frequency ratio of the system clock and the peripheral clock is also taken into consideration.

The unit for +1 in “Data Transfer (Read)” column is one system clock cycle (ICLK). For the operation example, see section 16.3.8. Operation Timing.

### 16.3.10 Activating the DMAC

Table 16.19 shows the register setting procedure of normal, repeat and block transfer mode and Table 16.20 shows register setting procedure of repeat-block transfer mode.

**Table 16.19 Register setting procedure of normal transfer mode, repeat transfer mode and block transfer mode (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Disable the peripheral function as the DMACn request source. | To use peripheral function interrupts as DMAC activation sources.<br>Disable the control register for the peripheral function. |
| 2 | Disable the IRQn pin as the DMACn request source. | To use external pin interrupts as DMAC activation sources. |
| 3 | Set the DMACn Event Link select (ICU.DELSRn.DELS[8:0]) to 0x00. | Disable the DMACn request. |
| 4 | Clear the DMCNT.DTE bit to 0. | Disable DMA transfer. |
| 5 | Set the interrupt request as a DMACn request source in the DMAC Event Link Setting Register (ICU.DELSRn) by using the ICU. | To use internal peripheral interrupts or external pin interrupts as DMAC activation sources.<br>Enable the interrupt bit for the activation source. Set the DMACn activation source. |
| 6 | Set the peripheral module as a DMACn request source. | To use peripheral function interrupt as a DMAC activation source.<br>Set the control register for the peripheral function without starting it. |
| 7 | Set the IRQn pin function by using the ICU. | To use external pin interrupt as a DMAC activation source.<br>Set the IRQn pin function by using the Interrupt Controller Unit. |
| 8 | Set the DMAMD.DM[1:0] bits.<br>Set the DMAMD.SM[1:0] bits.<br>Set the DMAMD.DARA[4:0] bits.<br>Set the DMAMD.SARA[4:0] bits. | Set the Transfer destination address update mode bits.<br>Set the Transfer source address update mode bits.<br>Set the Transfer destination address extended repeat area bits.<br>Set the Transfer source address extended repeat area bits. |
| 9 | Set the DMTMD.DCTG[1:0] bits.<br>Set the DMTMD.SZ[1:0] bits.<br>Set the DMTMD.DTS[1:0] bits.<br>Set the DMTMD.MD[1:0] bits.<br>Set the DMTMD.TKP bit. | Set the Transfer request select bits.<br>Set the Data transfer size bits.<br>Set the Repeat area select bits.<br>Set the Transfer mode select bits.<br>Set the transfer keeping select bit. |
| 10 | Set the DMSAR register.<br>Set the DMDAR register.<br>Set the DMCRA register. | Set the transfer source start address.<br>Set the transfer destination start address.<br>Set the number of transfer operations. |
| 11 | Set the DMCRB register. | To use block transfer mode or repeat transfer mode.<br>Set the number of block transfer operations. |
| 12 | Set the DMOFR register. | To use the address update function with offset.<br>Set the offset value. |
| 13 | Set the DMINT.DTIE bit to 1. | To use the DMA transfer end interrupts.<br>Enable DMACn transfer end interrupts. |
| 14 | Set the DMINT.RPTIE bit.<br>Set the DMINT.SARIE bit.<br>Set the DMINT.DARIE bit.<br>Set the DMINT.ESIE bit to 1. | To use the DMA transfer escape end interrupts<br>Set the repeat size end interrupt.<br>Set the transfer source address extended repeat area overflow interrupt.<br>Set the transfer destination address extended repeat area overflow interrupt.<br>Enable the DMA transfer escape end interrupt. |
| 15 | Set the DMCNT.DTE bit to 1. | Enable DMA transfer. |
| 16 | Set the DMAST.DMST bit to 1. | Enable DMAC operation. *1<br>Common settings for DMAC |

**Table 16.19 Register setting procedure of normal transfer mode, repeat transfer mode and block transfer mode (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 17 | Start the peripheral function as a DMACn request source. | To use peripheral function interrupt as a DMAC activation source |
| 18 | Enable the IRQn pin as a DMACn request source. | To use external pin interrupt as a DMAC activation source |
| 19 | End of initial settings. | For activation by software<br>On completion of the initial settings, writing 1 to the DMA software start bit (DMREQ.SWREQ) starts DMA transfer. |

Note: n: DMAC channel (n = 0 to 7)
Note 1. The DMAST.DMST bit setting does not necessarily have to follow the settings for the individual activation sources.

**Table 16.20 Register setting procedure of repeat-block transfer mode (1 of 2)**

| No. | Step name | Description |
|---|---|---|
| 1 | Disable the peripheral function as the DMACn request source. | To use peripheral function interrupts as DMA activation sources.<br>Disable the control register for the peripheral function. |
| 2 | Disable the IRQ pin as the DMACn request source. | To use external pin interrupts as DMA activation sources. |
| 3 | Set the DMACn Event Link select (ICU.DELSRn.DELS[8:0]) to 0x00. | Disable the DMACn request. |
| 4 | Clear the DMCNT.DTE bit to 0. | Disable DMACn transfer. |
| 5 | Set the interrupt request as a DMACn request source in the DMACn Event Link Setting Register (ICU.DELSRn) by using the ICU. | To use internal peripheral interrupts or external pin interrupts as DMA activation sources.<br>Enable the interrupt bit for the activation source.<br>Set the DMACn activation source. |
| 6 | Set the peripheral module as a DMACn request source. | To use peripheral function interrupt as a DMA activation source.<br>Set the control register for the peripheral function without starting it. |
| 7 | Set the IRQ pin function by using the Interrupt Controller Unit. | To use external pin interrupt as a DMA activation source.<br>Set the IRQ pin function by using the Interrupt Controller Unit. |
| 8 | Set the DMAMD.DM[1:0] bits.<br>Set the DMAMD.SM[1:0] bits.<br>Set the DMAMD.DARA[4:0] bits.<br>Set the DMAMD.SARA[4:0] bits.<br>Set the DMAMD.DADR bit.<br>Set the DMAMD.SADR bit. | Set the Transfer destination address update mode bits.<br>Set the Transfer source address update mode bits.<br>Set the Transfer destination address extended repeat area bits.<br>Set the Transfer source address extended repeat area bits.<br>Set the Transfer destination address update select after reloading.<br>Set the Transfer source address update select after reloading. |
| 9 | Set the DMTMD.DCTG[1:0] bits.<br>Set the DMTMD.SZ[1:0] bits.<br>Set the DMTMD.MD[1:0] bits.<br>Set the DMTMD.TKP bit. | Set the Transfer request select bits.<br>Set the Data transfer size bits.<br>Set the Transfer mode to repeat-block transfer mode.<br>Set the transfer keeping select bit. |
| 10 | Set the DMSAR register.<br>Set the DMDAR register.<br>Set the DMSRR register.<br>Set the DMDRR register.<br>Set the DMCRA register.<br>Set the DMCRB register. | Set the transfer source start address.<br>Set the transfer destination start address.<br>Set the initial value of source start address.<br>Set the initial value of destination start address.<br>Set the number of transfer operations.<br>Set the number of block transfer operations. |
| 11 | Set the DMSBS register.<br>Set the DMDBS register. | To use the address update function with incremental, decremental or offset.<br>Set the source buffer size and access offset.<br>Set the destination buffer size and access offset. |
| 12 | Set the DMINT.DTIE bit to 1. | To use DMA transfer end interrupts.<br>Enable DMACn transfer end interrupts. |
| 13 | Set the DMCNT.DTE bit to 1. | Enable DMACn transfer. |
| 14 | Set the DMAST.DMST bit to 1. | Enable DMAC operation. *1 |
| 15 | Start the peripheral function as a DMACn request source. | To use peripheral function interrupt as a DMA activation source. |
| 16 | Enable the IRQ pin as a DMACn request source. | To use external pin interrupt as a DMA activation source. |

**Table 16.20 Register setting procedure of repeat-block transfer mode (2 of 2)**

| No. | Step name | Description |
|---|---|---|
| 17 | End of initial settings. | For activation by software.<br>On completion of the initial settings, writing 1 to the DMA software start bit (DMREQ.SWREQ) starts DMA transfer. |

Note: n: DMAC channel (n = 0 to 7)
Note 1. The DMAST.DMST bit setting does not necessarily have to follow the settings for the individual activation sources.

## 16.3.11 Starting DMA Transfer

To enable the DMA transfer, set the DMCNT.DTE bit to 1 (enable the DMA transfer), and then set the DMAST.DMST bit to 1 (enable the DMAC activation).

New activation requests are not accepted during the transfer of another DMAC channel or DTC. When the preceding transfer is complete, channel arbitration selects the DMA transfer request of the highest priority channel, and the DMA transfer of that channel starts. When the DMA transfer starts, the DMSTS.ACT flag is set to 1 (the DMAC is in the active state).

## 16.3.12 Registers during DMA Transfer

The DMAC registers are updated by a DMA transfer. The value to be updated differs according to the other settings and the transfer state. The registers to be updated are DMSAR, DMDAR, DMCRA, DMCRB, DMSBS, DMDBS, DMCNT, and DMSTS.

### DMA Source Address Register (DMSAR)

When data has been transferred in response to one transfer request, the contents of DMSAR are updated to the address to be accessed by the next transfer request.

For details on register update operation in each transfer mode, see Table 16.5 to Table 16.13.

### DMA Destination Address Register (DMDAR)

When data has been transferred in response to one transfer request, the contents of DMDAR are updated to the address to be accessed by the next transfer request.

For details on register update operation in each transfer mode, see Table 16.5 to Table 16.13.

### DMA Transfer Count Register (DMCRA)

When data has been transferred in response to one transfer request, the count value is updated. The update operation depends on the transfer mode selected.

For details on register update operation in each transfer mode, see Table 16.5 to Table 16.13.

### DMA Block Transfer Count Register (DMCRB)

When data has been transferred in response to one transfer request, the count value is updated. The update operation depends on the transfer mode selected.

For details on register update operation in each transfer mode, see Table 16.5 to Table 16.13.

### DMA Source Buffer Size Register (DMSBS)

When data has been transferred in response to one transfer request, the count value is updated. The update operation depends on the transfer mode selected.

For details on register update operation in each transfer mode, see Table 16.8 to Table 16.13.

### DMA Destination Buffer Size Register (DMDBS)

When data has been transferred in response to one transfer request, the count value is updated. The update operation depends on the transfer mode selected.

For details on register update operation in each transfer mode, see Table 16.8 to Table 16.13.

#### DMA Transfer Enable Bit (DMCNT.DTE)

Although the DMCNT.DTE bit enables or disables data transfer by the register write access, it is automatically cleared to 0 by the DMAC according to the DMA transfer state.

The conditions for clearing this bit by the DMAC are as follows:

- When the specified total volume of data transfer is completed
- When DMA transfer is stopped by the repeat size end interrupt
- When DMA transfer is stopped by the extended repeat area overflow interrupt
- When DMA transfer error occurs

Writing to the registers for the channels when the corresponding DMCNT.DTE bit is set to 1 is prohibited (except for DMCNT ). In this case, writing must be performed after the bit is cleared to 0.

#### DMAC Active Flag (DMSTS.ACT)

The DMSTS.ACT flag indicates whether the DMACn is in the idle or active state.

This flag is set to 1 when the DMAC starts data transfer, and is cleared to 0 when data transfer in response to one transfer request is completed.

Even when DMA transfer is stopped by writing 0 to the DMCNT.DTE bit during DMA transfer, this flag remains 1 until DMA transfer is completed.

#### Transfer End Interrupt Flag (DMSTS.DTIF)

The DMSTS.DTIF flag is set to 1 after DMA transfer of the total transfer size of data is completed.

When both this flag and the DMINT.DTIE bit are set to 1, a transfer end interrupt is requested.

This flag is set to 1 when the DMA transfer bus cycle is completed and the DMSTS.ACT flag is cleared to 0 indicating the DMA transfer end.

This flag is automatically cleared to 0 when the DMCNT.DTE bit is set to 1 during the interrupt handling.

#### Transfer Escape End Interrupt Flag (DMSTS.ESIF)

The DMSTS.ESIF flag is set to 1 when a repeat size end interrupt or extended repeat area overflow interrupt is requested. When this bit and the DMINT.ESIE bit are set to 1, a transfer escape end interrupt is requested.

This flag is set to 1 when the bus cycle of the DMA transfer having caused the interrupt request is completed and the DMSTS.ACT flag is cleared to 0 indicating the DMA transfer end.

This flag is automatically cleared to 0 when the DMCNT.DTE bit is set to 1 during an interrupt handling.

Before sending an interrupt request from the DMAC to the CPU or the DTC, the interrupt control register must be set.

For details, see section 13, Interrupt Controller Unit (ICU).

### 16.3.13 Channel Priority

When multiple DMA transfer requests are present, the DMAC determines the priority of channels that have DMA transfer requests.

- The channel priority is fixed as follows: Channel 0 > Channel 1 > Channel 2 > Channel 3… > Channel 7 (Channel 0: Highest).

When a DMA transfer request is generated during data transfer, channel arbitration is started after the final data has been transferred, and DMA transfer of the higher-priority channel starts.

### 16.3.14 Channel Security

The security attribute of transfer access of DMACn, security attribute of access to register of DMACn, security attribute of access to the ICU.DELSRn register are controlled by ICUSARC.SADMACn bit. For details on the ICUSARC register, see section 13, Interrupt Controller Unit (ICU).

When the ICUSARC.SADMACn bit is 0, transfer of DMACn is secure access for both read and write. At the same time, the registers of channel n and the DELSRn register are protected from a non-secure access.

When the ICUSARC.SADMACn bit is 1, transfer of DMACn is non-secure access for both read and write. At the same time, the registers of channel n and the DELSRn register are non-secure attributes.

Do not write to the ICUSARC.SADMACn bit while DMA transfer of same channel is enabled or a bus master is writing to the DMA registers of same channel.

Figure 16.23 shows security attribute about each DMAC channels.

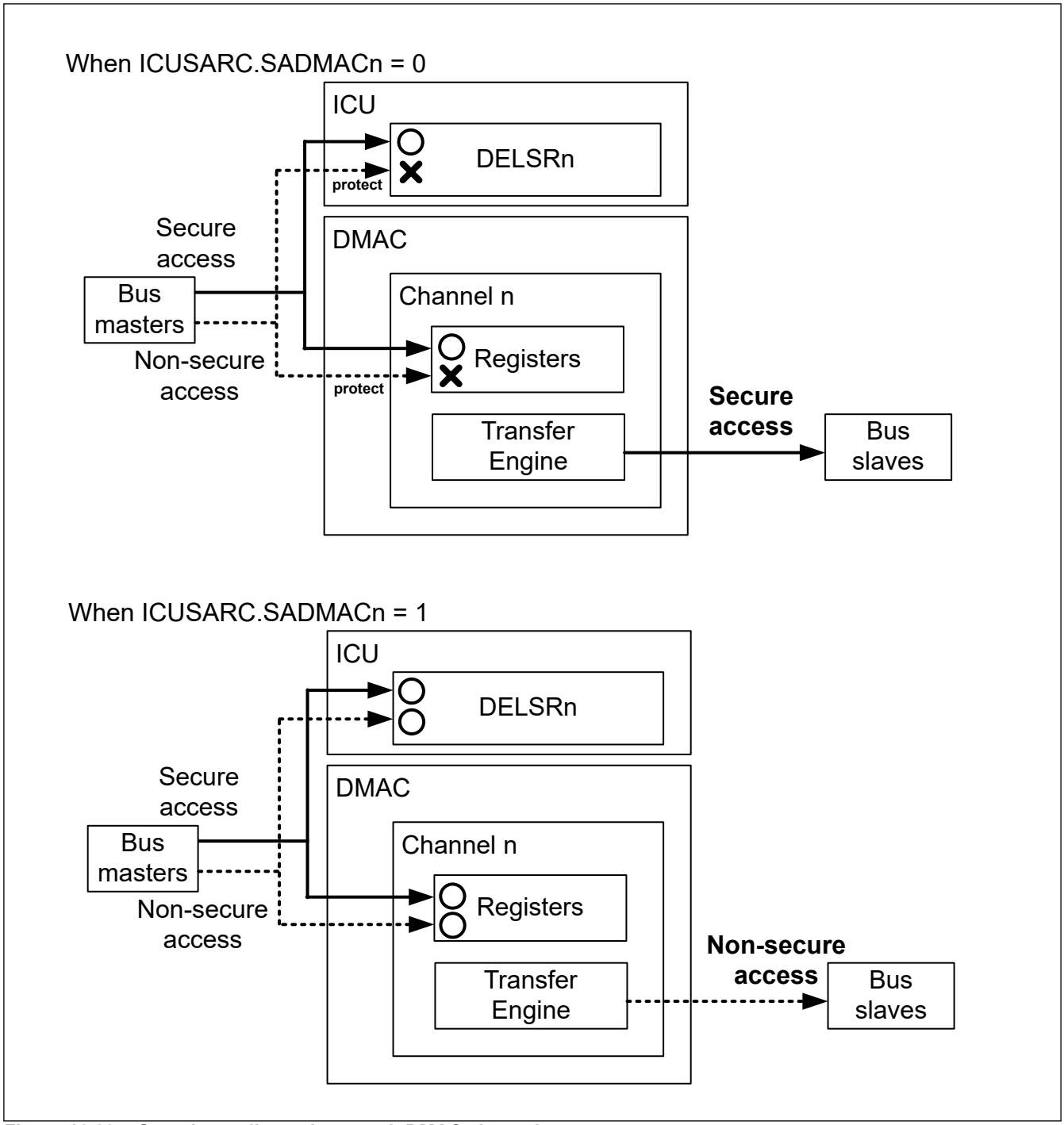



**Figure 16.23 Security attribute about each DMAC channels**

## 16.3.15 Master TrustZone Filter in DMAC

DMAC has the Master TrustZone Filter. The MasterTrustZone Filter in DMAC can detect the security areas of Flash area(code Flash and data Flash) and SRAM area(ECC / Parity RAM) defined by IDAU. When set No-secure channel

accesses those addresses, it detects the security violation. Access of violation address is not performed. Detected the error is handled as the Master TrustZone Filter error.

## 16.4 Ending DMA Transfer

The operation for ending DMA transfer depends on the transfer end conditions. When DMA transfer ends, the DMCNT.DTE bit and the DMSTS.ACT flag are changed from 1 to 0, indicating that DMA transfer has ended.

### 16.4.1 Transfer End by Completion of Specified Total Number of Transfer Operations

#### (1) In Normal Transfer Mode (DMTMD.MD[1:0] = 00b)

When the value of DMCRAL changes from 1 to 0, DMA transfer ends on the corresponding channel, and the DMCNT.DTE bit is cleared to 0 and the DMSTS.DTIF flag is set to 1 at the same time. If the DMINT.DTIE bit is 1 at this time, a transfer end interrupt request is issued to the CPU or the DTC.

#### (2) In Repeat Transfer Mode (DMTMD.MD[1:0] = 01b)

When the value of DMCRBL changes from 1 to 0, DMA transfer ends on the corresponding channel, and the DMCNT.DTE bit is cleared to 0 and the DMSTS.DTIF flag is set to 1 at the same time. If the DMINT.DTIE bit is 1 at this time, an interrupt request is issued to the CPU or the DTC.

If the DMTMD.TKP bit is 1 (in free-running function), the DMSTS.DTIF bit is set to 1, but the DMCNT.DTE bit is not cleared to 0.

#### (3) In Block Transfer Mode (DMTMD.MD[1:0] = 10b)

When the value of DMCRBL changes from 1 to 0, DMA transfer ends on the corresponding channel, and the DMCNT.DTE bit is cleared to 0 and the DMSTS.DTIF flag is set to 1 at the same time. If the DMINT.DTIE bit is 1 at this time, an interrupt request is issued to the CPU or the DTC.

Before sending an interrupt request from the DMAC to the CPU or the DTC, the interrupt control register must be set.

For details, see section 13, Interrupt Controller Unit (ICU).

If the DMTMD.TKP bit is 1 (in free-running function),the DMSTS.DTIF bit is set to 1, but the DMCNT.DTE bit is not cleared to 0.

#### (4) In Repeat-Block Transfer Mode (DMTMD.MD[1:0] = 11b)

When the value of DMCRBL changes from 1 to 0, DMA transfer ends on the corresponding channel, and the DMCNT.DTE bit is cleared to 0 and the DMSTS.DTIF flag is set to 1 at the same time. If the DMINT.DTIE bit is 1 at this time, an interrupt request is issued to the CPU or the DTC.

Before sending an interrupt request from the DMAC to the CPU or the DTC, the interrupt control register must be set. For details, see section 13, Interrupt Controller Unit (ICU).

If the DMTMD.TKP bit is 1 (in free-running function),the DMSTS.DTIF bit is set to 1, but the DMCNT.DTE bit is not cleared to 0.

### 16.4.2 Transfer End by Repeat Size End Interrupt

In repeat transfer mode, a repeat size end interrupt is requested when transfer of a 1-repeat size of data is completed while the DMINT.RPTIE bit is set to 1. When the interrupt is requested to complete DMA transfer, the DMCNT.DTE bit is cleared to 0 and the DMSTS.ESIF flag is set to 1 even if the DMTMD.TKP bit is 1 (in free-running function). If the DMINT.ESIE bit is 1 at this time, an interrupt request is issued to the CPU or the DTC. Here, the transfer can be resumed by writing 1 to the DMCNT.DTE bit.

A repeat size end interrupt can be requested also in block transfer mode. In block transfer mode, the interrupt is requested in the same way as in repeat transfer mode when transfer of a 1-block size data is completed.

Repeat size end interrupt cannot be requested in repeat-block transfer mode.

Before sending an interrupt request from the DMAC to the CPU or the DTC, the interrupt control register must be set. For details, see section 13, Interrupt Controller Unit (ICU).

### 16.4.3 Transfer End by Interrupt on Extended Repeat Area Overflow

When an overflow on the extended repeat area occurs while the extended repeat area is specified and the DMINT.SARIE or DMINT.DARIE bit is set to 1 even if the DMTMD.TKP bit is 1 (in free-running function), an interrupt by an extended repeat area overflow is requested. When the interrupt is requested, the DMA transfer is terminated, the DMCNT.DTE bit is cleared to 0, and the ESIF flag in DMSTS is set to 1. If the DMINT.ESIE bit is 1 at this time, an interrupt request is issued to the CPU or the DTC.

Even if an interrupt by an extended repeat area overflow is requested during a read cycle, the following write cycle is performed.

In block transfer mode, even if an interrupt by an extended repeat area overflow is requested during a 1-block transfer, the remaining data in the block is transferred; transfer is terminated after a block transfer.

An interrupt by an extended repeat area overflow cannot be requested in repeat-block transfer mode.

Before sending an interrupt request from the DMAC to the CPU or the DTC, the interrupt control register must be set. For details, see section 13, Interrupt Controller Unit (ICU).

## 16.5 Processing on DMA Transfer Error

DMA transfer error occurs with the Master TrustZone Filter error in DMAC, the Slave TrustZone Filter error, the Master MPU error, the Slave Bus Error or the Illegal Access Error. If the access error occurs during the DMA transfer, the DMAC immediately stops the transfer of error occurred channel. At this time, the ICU setting of the corresponding channel is also cleared. If there is a request other than the channel which caused the error, it will be re-arbitration as it is.

When the transfer error occurs, DMCNT.DTE of the error causing channel is set to 0. Also, the error response is informed to the ICU.DELSRn of the corresponding channel is cleared. Write back to each register is not performed. Furthermore, it generates the error response detection interrupt request (DMA_TRANSERR) to notify that an error has occurred by DMAC/DTC transfer.

When the Master TrustZone Filter error occurs, the Slave TrustZone error occurs or the Master MPU error occurs, it is possible to confirm the error information of DMAC by selecting NMI. The DMAC error channel register is cleared by selecting reset. Under the conditions where NMI is generated due to transfer error in DMAC, two interrupts(NMI and DMA_TRANSERR) are generated. In this case, NMI always responds first.

The error response detection interrupt request (DMA_TRANSERR) occurs when the Slave Bus error or the Illegal Access error occurs. Furthermore, it occurs after NMI when the error response detection interrupt request (DMA_TRANSERR) is not cleared in NMI handler.

section 16.5.1. Processing on NMI handler describes how to confirm the error information of the DMAC in the NMI handler.

section 16.5.2. Processing on Error response detection interrupt request (DMA_TRANSERR) handler describes how to confirm the error information of the DMAC in the DMA_TRANSERR handler.

Interrupts and the error information generated due to transfer errors are shown in section 16.6.2. Transfer Error Interrupt.

### 16.5.1 Processing on NMI handler

The cause of NMI due to the DMA transfer error is the Master TrustZone Filter error, the Slave TrustZone Filter error or the Master MPU error. When NMI occurs due to the DMAC transfer error, the error response detection interrupt request (DMA_TRANSERR) will occur after the end of NMI handler. It is possible to confirm the cause of the error and the DMAC channel in which the error occurred. When NMI occurs, perform the necessary processing according to the flow described in the ICU chapter.

Figure 16.24 shows the flow for confirm the channel that caused the Master TrustZone Filter Error in DMAC

Figure 16.25 shows the flow for confirm the channel that caused the Slave TrustZone Filter Error in DMAC

Figure 16.26 shows the flow for confirm the channel and Security Attribute that caused the Master MPU error in DMAC.

If completing all processing in NMI handler, it is possible to clear the error response detection interrupt request (DMA_TRANSERR) that occurs subsequently.

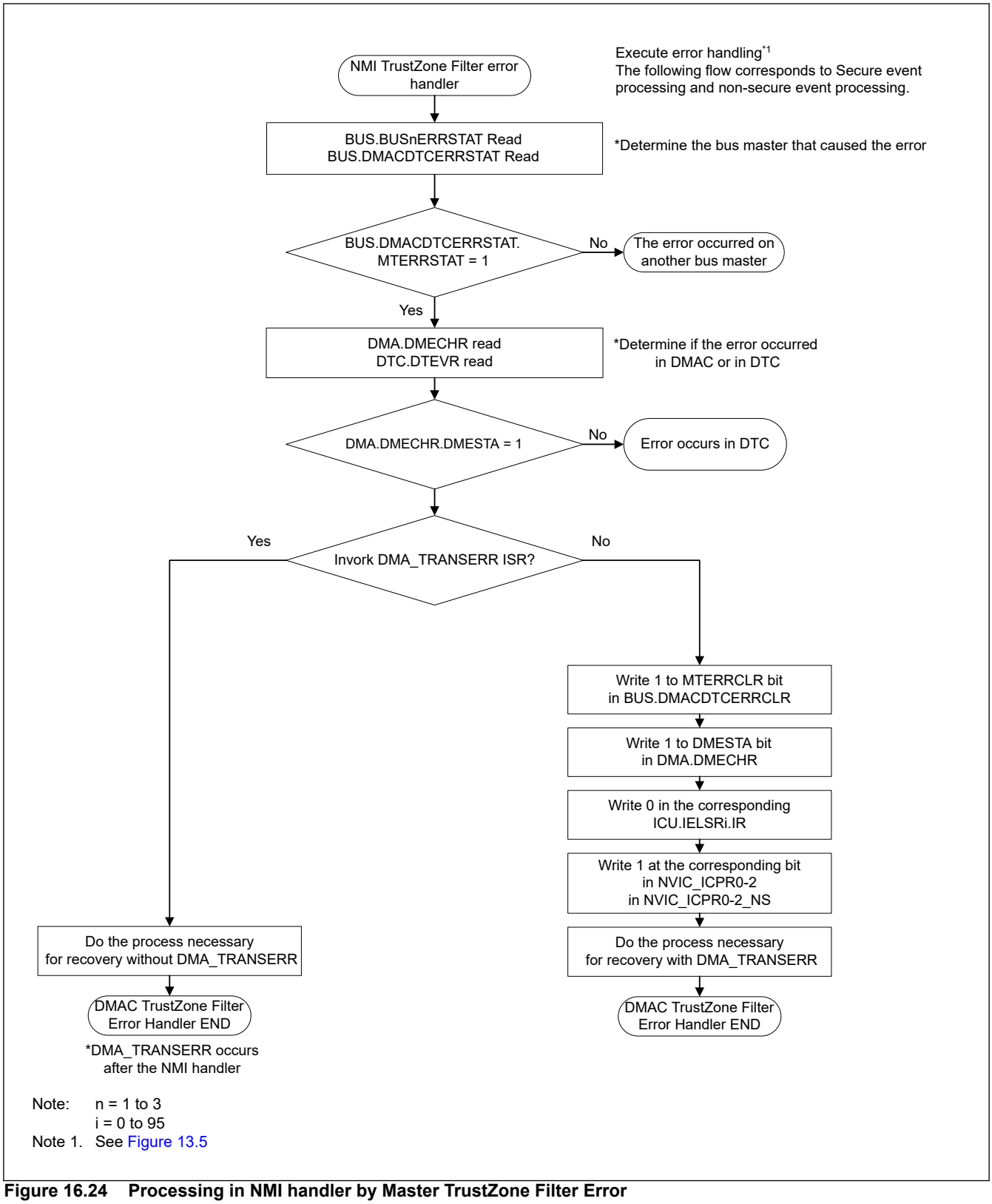
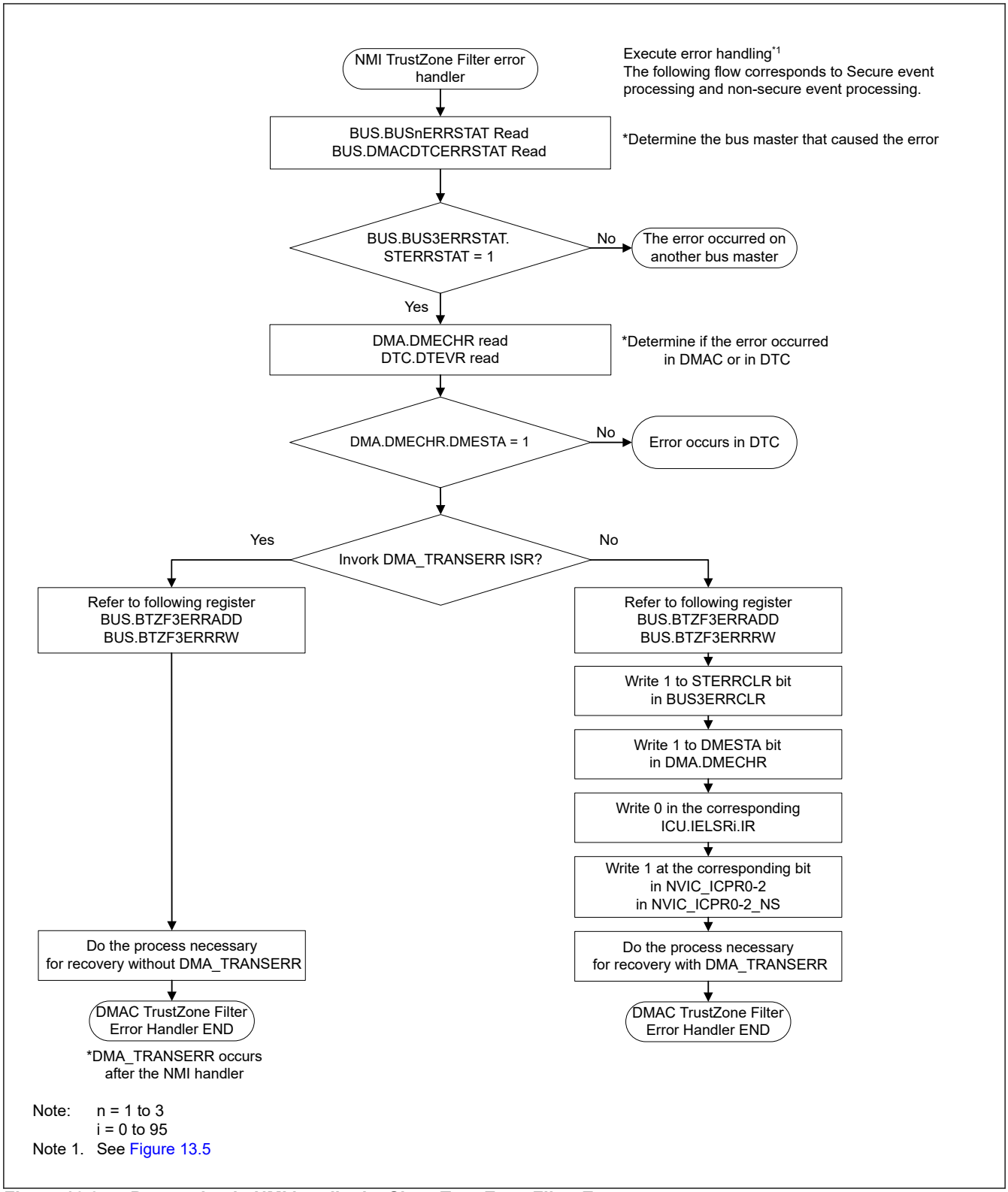
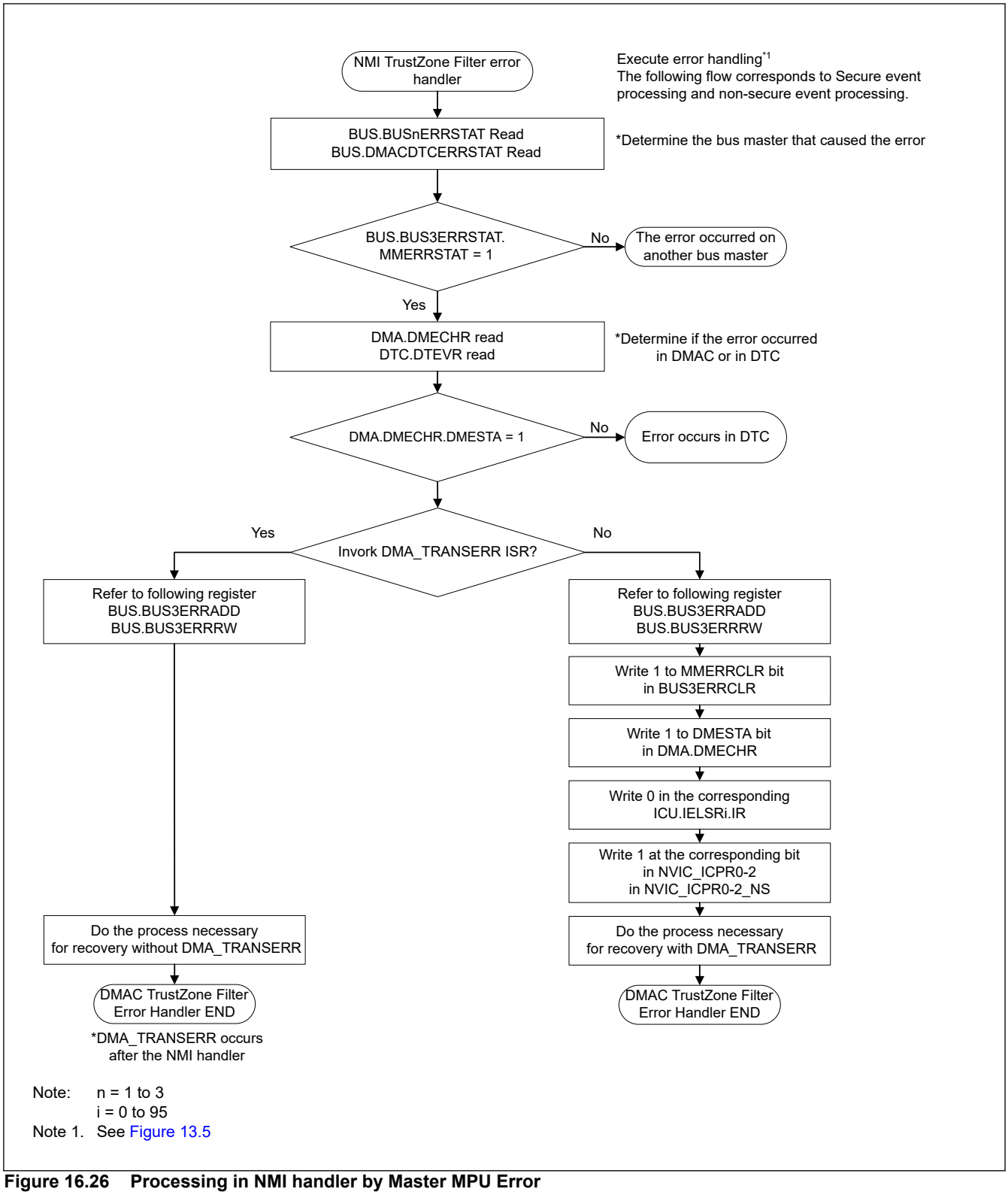



**Figure 16.24 Processing in NMI handler by Master TrustZone Filter Error**




Note: n = 1 to 3
i = 0 to 95
Note 1. See Figure 13.5

**Figure 16.25 Processing in NMI handler by Slave TrustZone Filter Error**




Note: n = 1 to 3
i = 0 to 95
Note 1. See Figure 13.5

**Figure 16.26 Processing in NMI handler by Master MPU Error**

## 16.5.2 Processing on Error response detection interrupt request (DMA_TRANSERR) handler

The cause of error response detection interrupt request (DMA_TRANSERR) due to DMA transfer error is the Slave Bus Error or Illegal Access Error. Also, it occurs after the NMI handler error response detection interrupt request (DMA_TRANSERR) is not cleared by the NMI handler.

It is possible to confirm the cause of the error and the DMAC channel in which the error occurred.

Error cause confirmation procedure is shown Figure 16.27.

Figure 16.28 shows the flow for confirm the channel that caused the Master TrustZone Filter Error in DMAC

Figure 16.29 shows the flow for confirm the channel that caused the Slave TrustZone Filter Error in DMAC

Figure 16.30 shows the flow for confirm the channel and Security Attribute that caused the Master MPU Error in DMAC

Figure 16.31 shows the flow for confirm the channel and Security Attribute that caused the Slave Bus Error in DMAC

Figure 16.32 shows the flow for confirm the channel and Security Attribute that caused the Illegal Access Error in DMAC

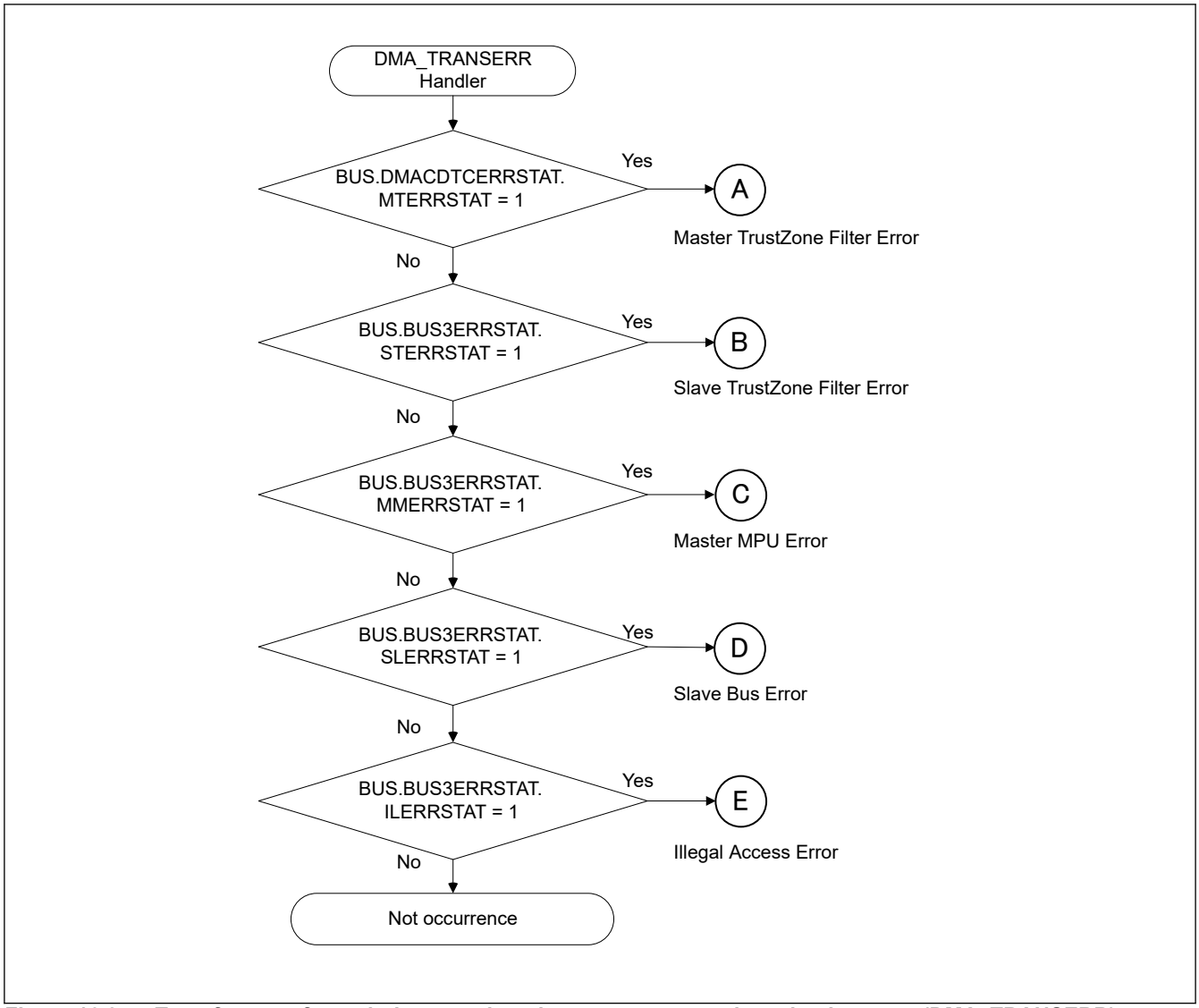



**Figure 16.27 Transfer error factor judgment when the error response detection interrupt (DMA_TRANSERR) occurs**

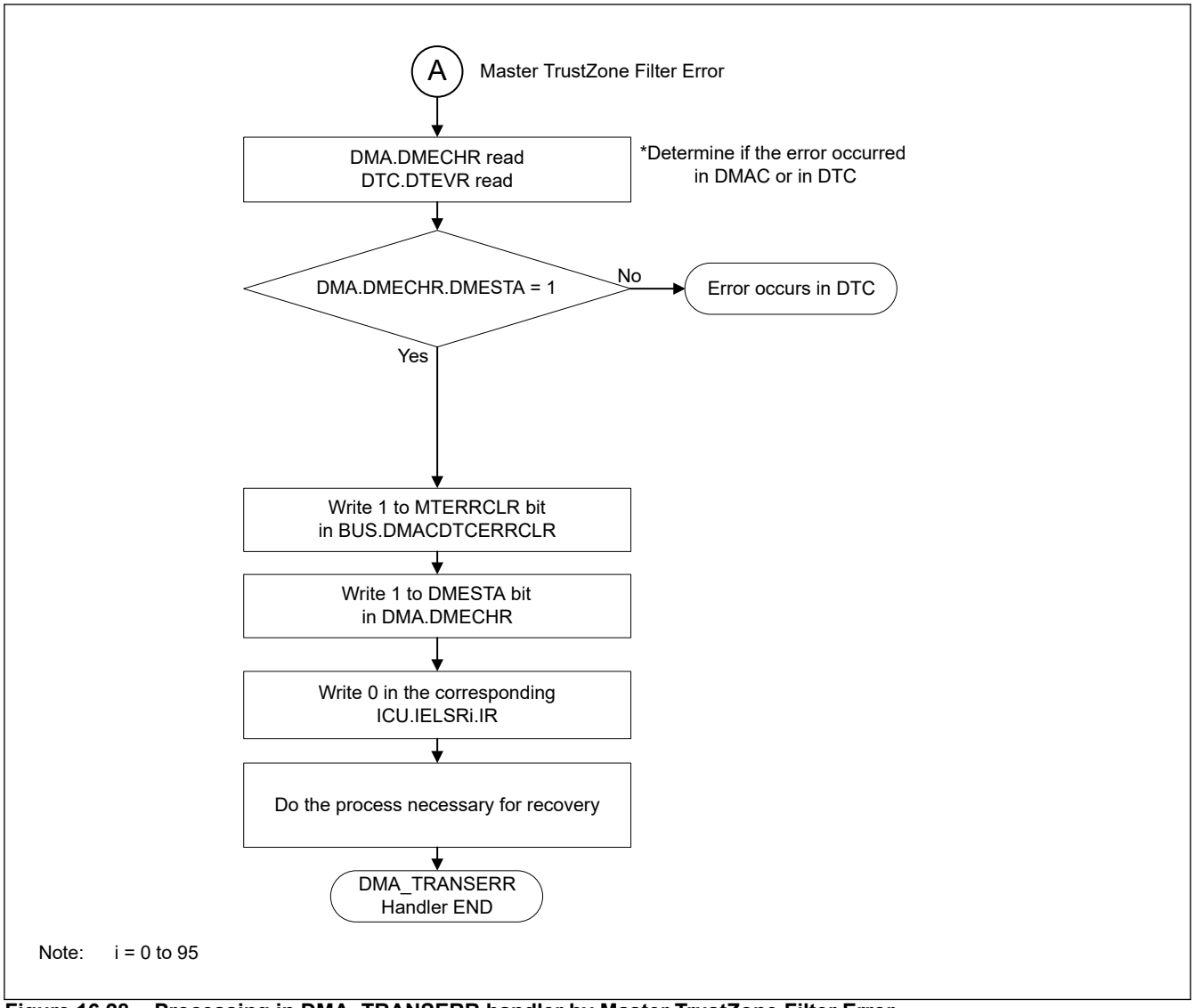



**Figure 16.28 Processing in DMA_TRANSERR handler by Master TrustZone Filter Error**

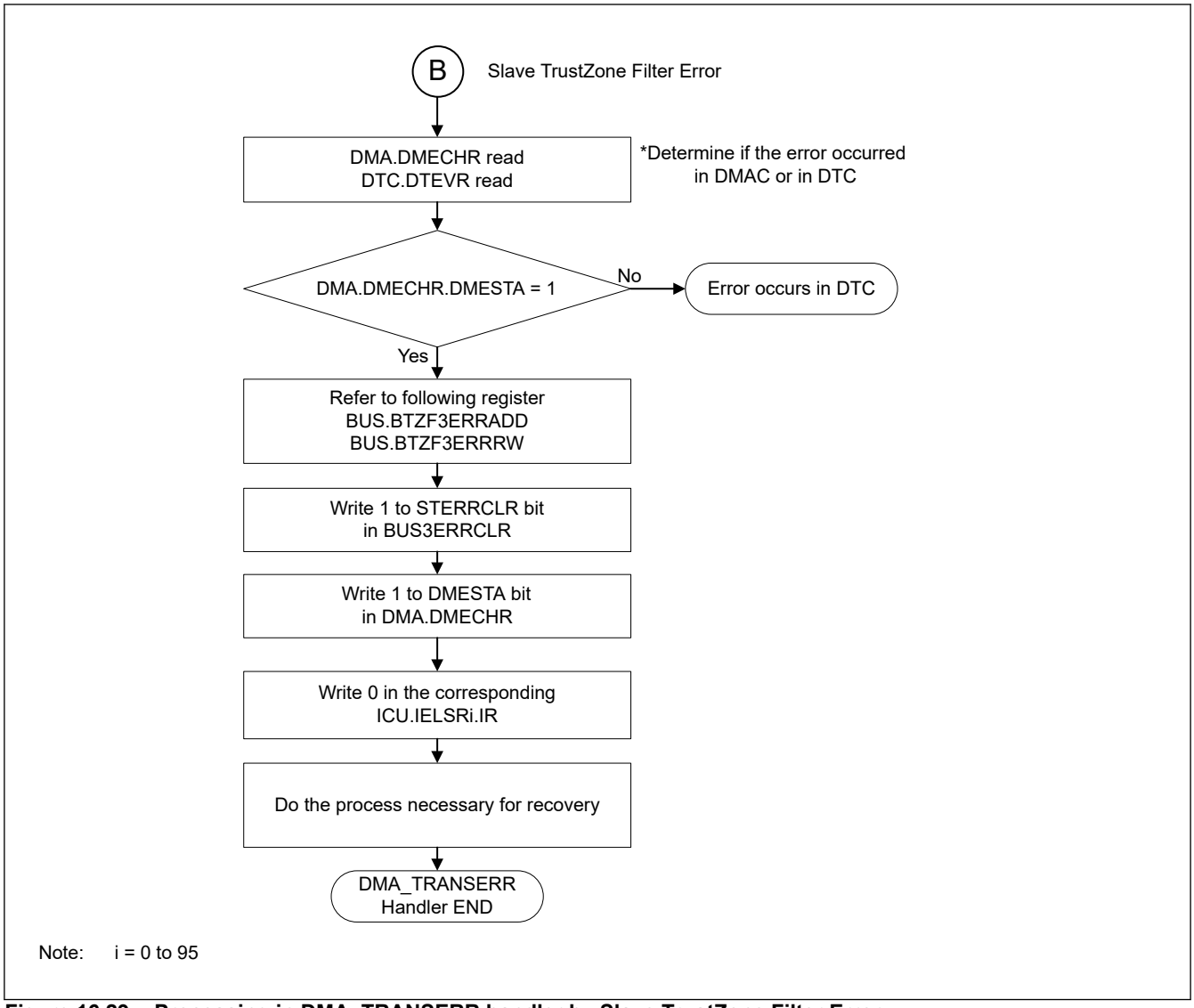
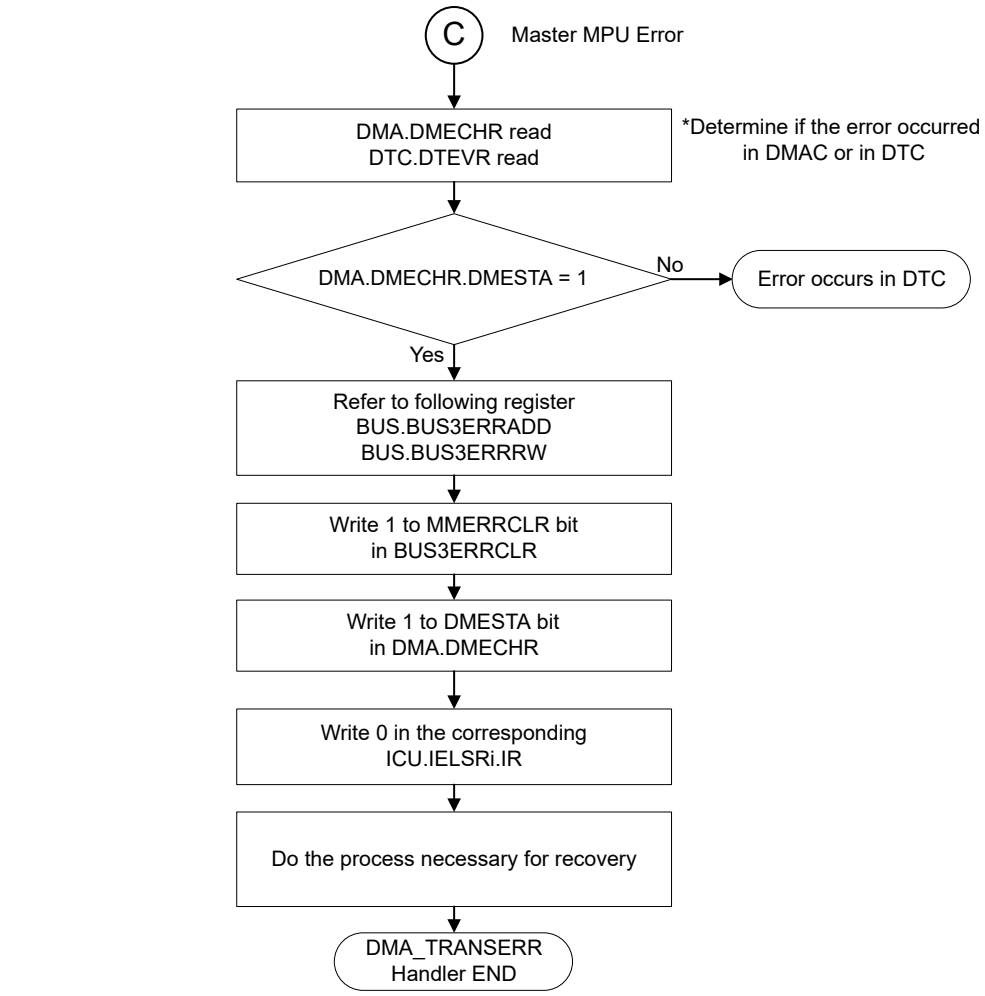



**Figure 16.29 Processing in DMA_TRANSERR handler by Slave TrustZone Filter Error**




Note: i = 0 to 95

**Figure 16.30 Processing in DMA_TRANSERR handler by Master MPU Error**

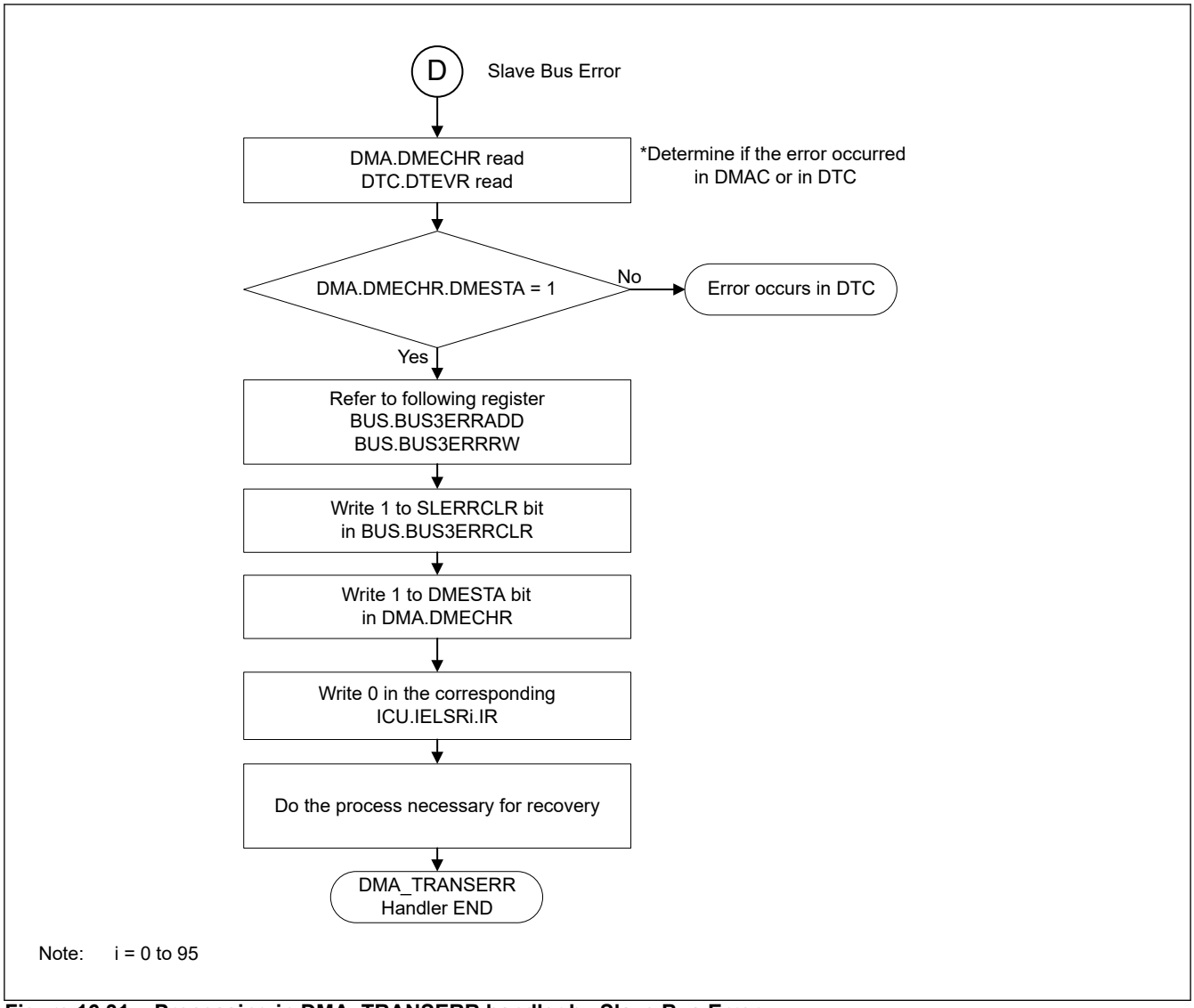



**Figure 16.31 Processing in DMA_TRANSERR handler by Slave Bus Error**

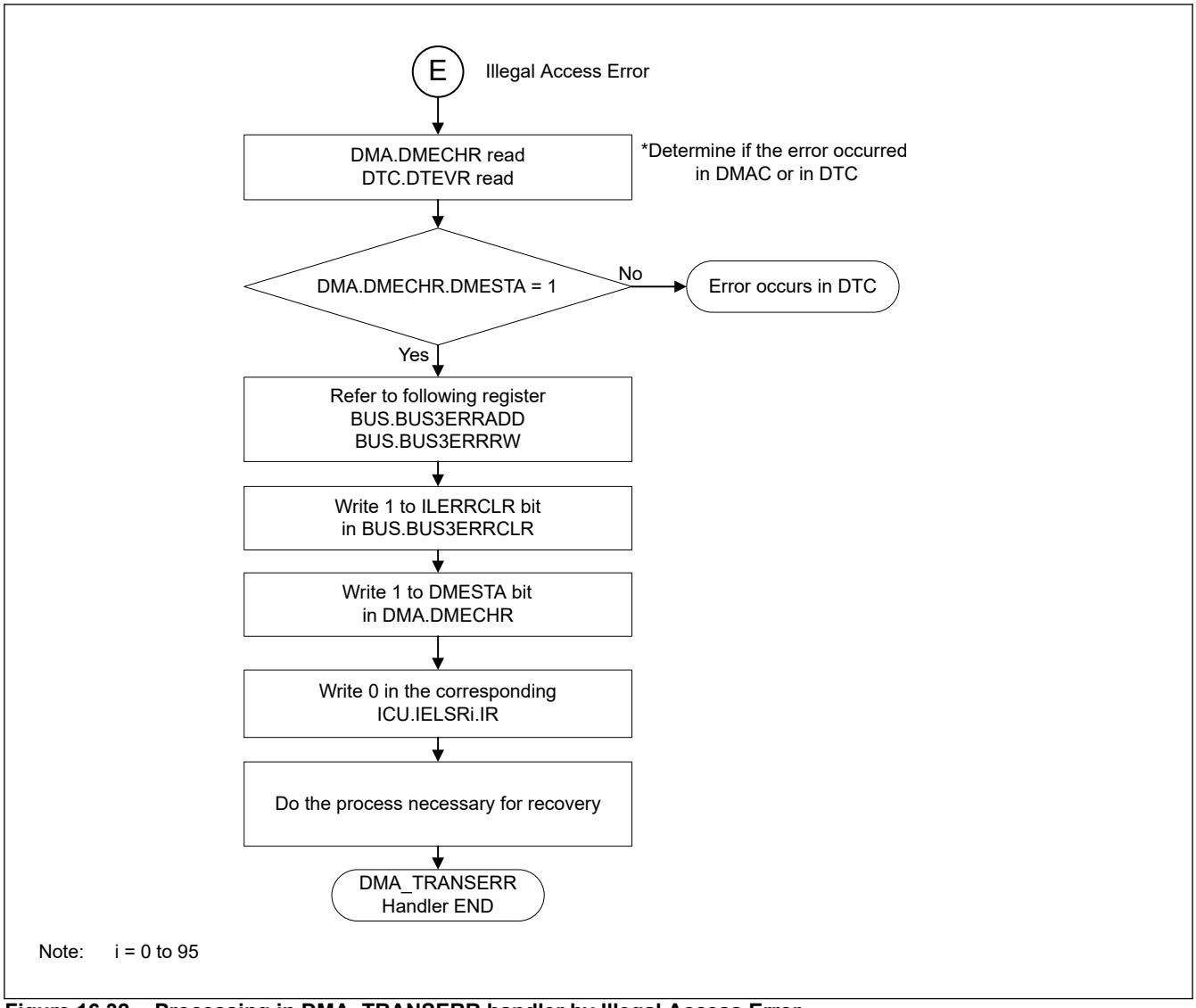



**Figure 16.32 Processing in DMA_TRANSERR handler by Illegal Access Error**

## 16.6 Interrupts

### 16.6.1 Transfer End Interrupt

Each DMAC channel can output an interrupt request (DMACn_INT) to the CPU or the DTC after transfer in response to one request is completed.

In repeat-block transfer mode, do not enable escape transfer end interrupt.

Table 16.21 lists the relation among the interrupt sources, the interrupt status flags, and the interrupt enable bits. Figure 16.33 shows the schematic logic diagram of interrupt outputs (DMACn (n = 0 to 7)). Figure 16.34 shows the DMAC interrupt handling routine to resume/terminate DMA transfer.

**Table 16.21 Relation among interrupt sources, interrupt status flags, and interrupt enable bits (1 of 2)**

| Interrupt sources | Interrupt enable bits | Interrupt status flags | Request output enable bits |
|---|---|---|---|
| Transfer end | — | DMSTS.DTIF | DMINT.DTIE |

**Table 16.21 Relation among interrupt sources, interrupt status flags, and interrupt enable bits (2 of 2)**

| Interrupt sources | | Interrupt enable bits | Interrupt status flags | Request output enable bits |
|---|---|---|---|---|
| Escape transfer end | Repeat size end | DMINT.RPTIE | DMSTS.ESIF | DMINT.ESIE |
| | Source address extended repeat area overflow | DMINT.SARIE | | |
| | Destination address extended repeat area overflow | DMINT.DARIE | | |

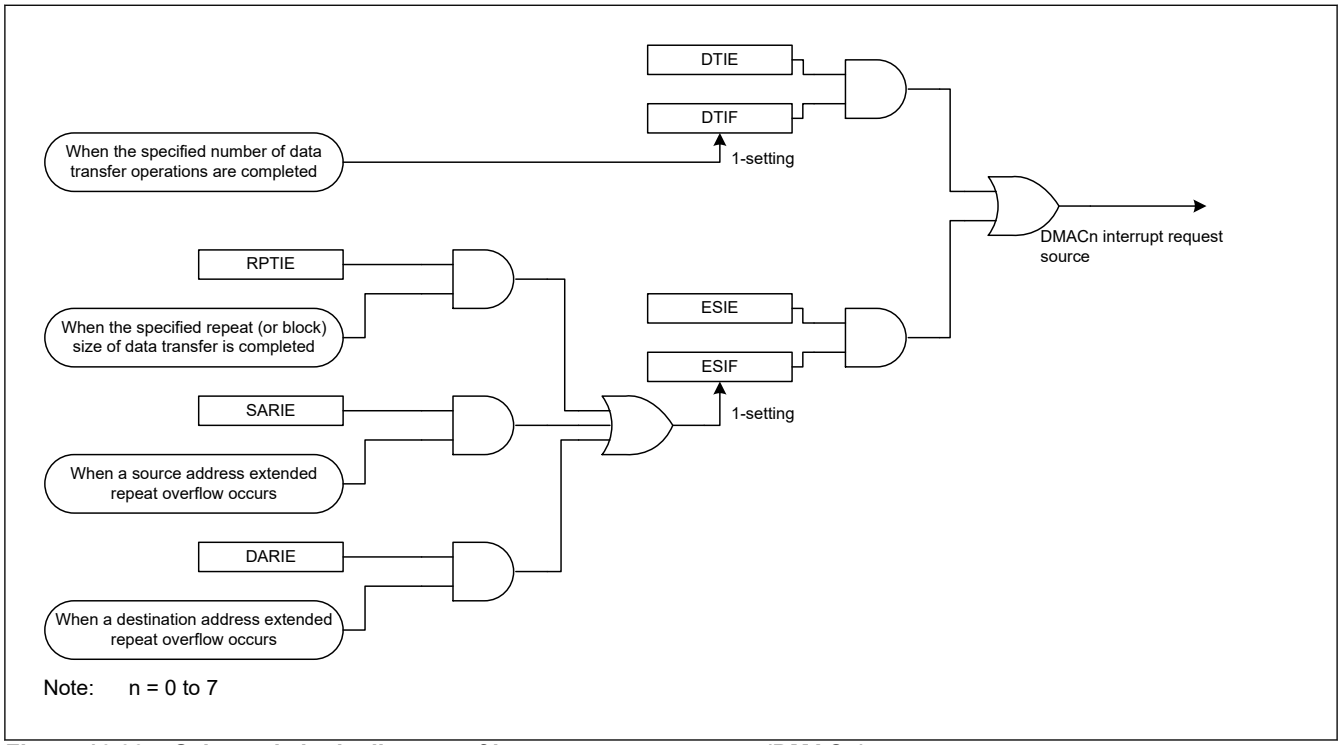



**Figure 16.33 Schematic logic diagram of interrupt output source (DMACn)**

Specifically, the different procedures are used for canceling an interrupt to restart DMA transfer in the following two cases:

- When terminating a DMA transfer
- When continuing a DMA transfer

### 16.6.1.1 When Terminating a DMA Transfer

Write 0 to the DMSTS.DTIF flag to clear a transfer end interrupt, and to the DMSTS.ESIF flag to clear a repeat size interrupt and an extended repeat area overflow interrupt. The DMACn remains in the stop state. When starting another DMA transfer after that, set the appropriate registers, and set the DMCNT.DTE bit to 1 (DMA transfer enabled).

### 16.6.1.2 When Continuing a DMA Transfer

Write 1 to the DMCNT.DTE bit. The DMSTS.ESIF flag is automatically cleared to 0 (interrupt source cleared), and DMA transfer is resumed.

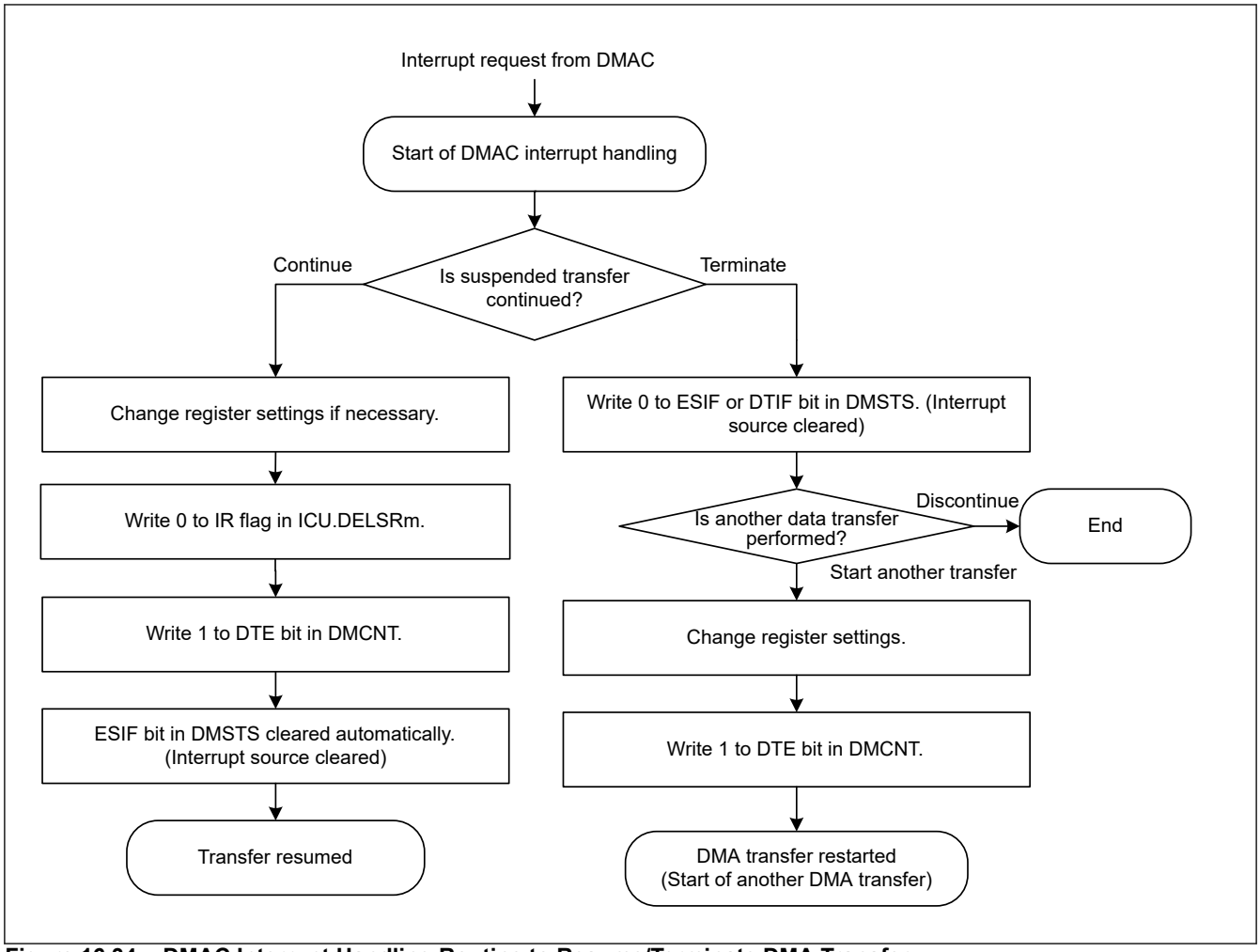



**Figure 16.34 DMAC Interrupt Handling Routine to Resume/Terminate DMA Transfer**

## 16.6.2 Transfer Error Interrupt

Error response detection interrupt request (DMA_TRANSERR) is generated from the DMAC/DTC when the transfer error is detected during DMAC transfer. The types of interrupts that occur when a DMAC transfer error occurs are listed in the Table 16.22. The Table 16.22 also shows error information stored when a transfer error occurs.

**Table 16.22 Interrupt and error information due to DMAC transfer error cause**

| Transfer error factor | NMI/RESET*1 Request | Interrupt Request | Bus Error Status | Error Address Error R/W | Error Channel Information |
|---|---|---|---|---|---|
| Master TrustZone Filter (in DMAC/DTC) | ICU.NMISR.TZFST*1 | DMA_TRANSERR | BUS.DMACDTCERRSTAT.MTERRSTAT*1 | — | DMA.DMECHR |
| Slave TrustZone Filter | ICU.NMISR.TZFST*1 | DMA_TRANSERR | BUS.BUS3ERRSTAT.STERRSTAT*1 | BUS.BTZF3ERRADD BUS.BTZF3ERRRW | DMA.DMECHR |
| Master MPU | ICU.NMISR. BUSMST | DMA_TRANSERR | BUS.BUS3ERRSTAT.MMERRSTAT | BUS.BUS3ERRADD BUS.BUS3ERRRW | DMA.DMECHR |
| Slave Bus Error | —*2 | DMA_TRANSERR | BUS.BUS3ERRSTAT.SLERRSTAT*2 | BUS.BUS3ERRADD BUS.BUS3ERRRW | DMA.DMECHR |
| Illegal Access Error | —*2 | DMA_TRANSERR | BUS.BUS3ERRSTAT.ILERRSTAT*2 | BUS.BUS3ERRADD BUS.BUS3ERRRW | DMA.DMECHR |

Note 1. Interrupt generated, when NMI request selected as the operation after detection of the Master MPU error and the TrustZone Filter error. By confirming BUS.BUS3ERRSTAT and BUS.DMACDTCERRSTAT, judge whether it is the Master or the Slave.

Note 2. If the error response detection interrupt (DMA_TRANSERR) occurs and NMI of the Master MPU or NMI of the TrustZone Filter has not occurred, treat it as the Illegal address access error or the Slave Bus Error. It can be judged also by BUS.BUS3ERRSTAT and BUS. DMACDTCERRSTAT.

Note that if the bus error occurs when writing the last data of transfer, the transfer end event and the error response detection interrupt (DMA_TRANSERR) occurs.

## 16.7 Event Link

Each DMAC channel outputs an event link request signal (DMACn_INT) every time it completes a data transfer, or a block transfer in block transfer mode.

For details, see section 18, Event Link Controller (ELC).

If a bus error occurs when writing the last data of transfer, a transfer end event and error response detection interrupt (DMA_TRANSERR) occurs.

## 16.8 Low-Power Consumption Function

Before entering the module-stop state or Software Standby mode, or Deep Software Standby mode, you must first set the DMAST.DMST bit to 0 (the DMAC module suspended) and use the settings in the sections that follow.

### (1) Module-stop function

Writing 1 to the MSTPCRA.MSTPA22 bit enables the module-stop function of the DMAC. If a DMA transfer is in progress when 1 is written to the MSTPA22 bit, the transition to the module-stop state proceeds after the DMA transfer ends. Access to the DMAC registers is prohibited while the MSTPA22 bit is 1. Writing 0 to the MSTPA22 bit releases the DMAC from the module-stop state.

### (2) Software Standby mode and Deep Software Standby mode

Use the settings described in section 10.7.1. Transitioning to Software Standby Mode, or in section 10.9.1. Transitioning to Deep Software Standby Mode .

If DMA transfer operations are in progress when the WFI instruction is executed, the DMA transfer completes before the transition to Software Standby mode or Deep Software Standby mode.

### (3) Notes on low power consumption function

For information on the WFI instruction and register settings, see section 10.10.7. Timing of WFI Instruction.

To perform a DMA transfer after returning from a low power mode, set the DMAST.DMST bit to 1 again. To use a request that is generated in Software Standby mode as an interrupt request to the CPU but not as a DMAC startup request, specify the CPU as the interrupt request destination, as described in section 13.4.1. Detecting Interrupts, and then execute the WFI instruction.

## 16.9 Usage Notes

### 16.9.1 Access to the Registers during DMA Transfer

Do not write to the following registers while the DMSTS.ACT flag of the same channel is set to 1 (DMAC active state) or the DMCNT.DTE bit of the same channel is set to 1 (DMA transfer enabled):

- DMSAR
- DMDAR
- DMCRA
- DMCRB
- DMTMD
- DMINT
- DMAMD
- DMOFR
- DMSBS
- DMDBS
- DMSRR

- DMDRR
- ICUSARC
- DMACSAR

### 16.9.2 DMA Transfer to Reserved Areas

DMA transfer to the reserved areas is prohibited. If such an access is made, transfer results are not guaranteed. For details on the reserved areas, see section 4, Address Space.

### 16.9.3 Setting of DMAC Event Link Setting Register of the Interrupt Controller Unit (ICU.DELSRn n = 0 to 7)

The DMAC event link setting register (ICU.DELSRn) should be set while the DMA transfer enable bit (DMCNT.DTE) is cleared to 0 (DMA transfer is disabled). Moreover, the DTC activation enable register (ICU.IELSRn.DTCE (n = 0 to 95)) that corresponds to the same event number that has been set by the ICU.DELSRn register should not be set to 1. For details on the ICU.IELSRn.DTCE and ICU.DELSRn, see section 13, Interrupt Controller Unit (ICU).

### 16.9.4 Suspending or Restarting DMAC Activation

To suspend a DMAC activation request, write 0x00 to the DMAC Event Link select bits (ICU.DELSRn.DELS[8:0]). To restart the DMA transfer, write the event number to the ICU.DELSRn.DELS[8:0] bits following the settings shown in section 16.3.10. Activating the DMAC.

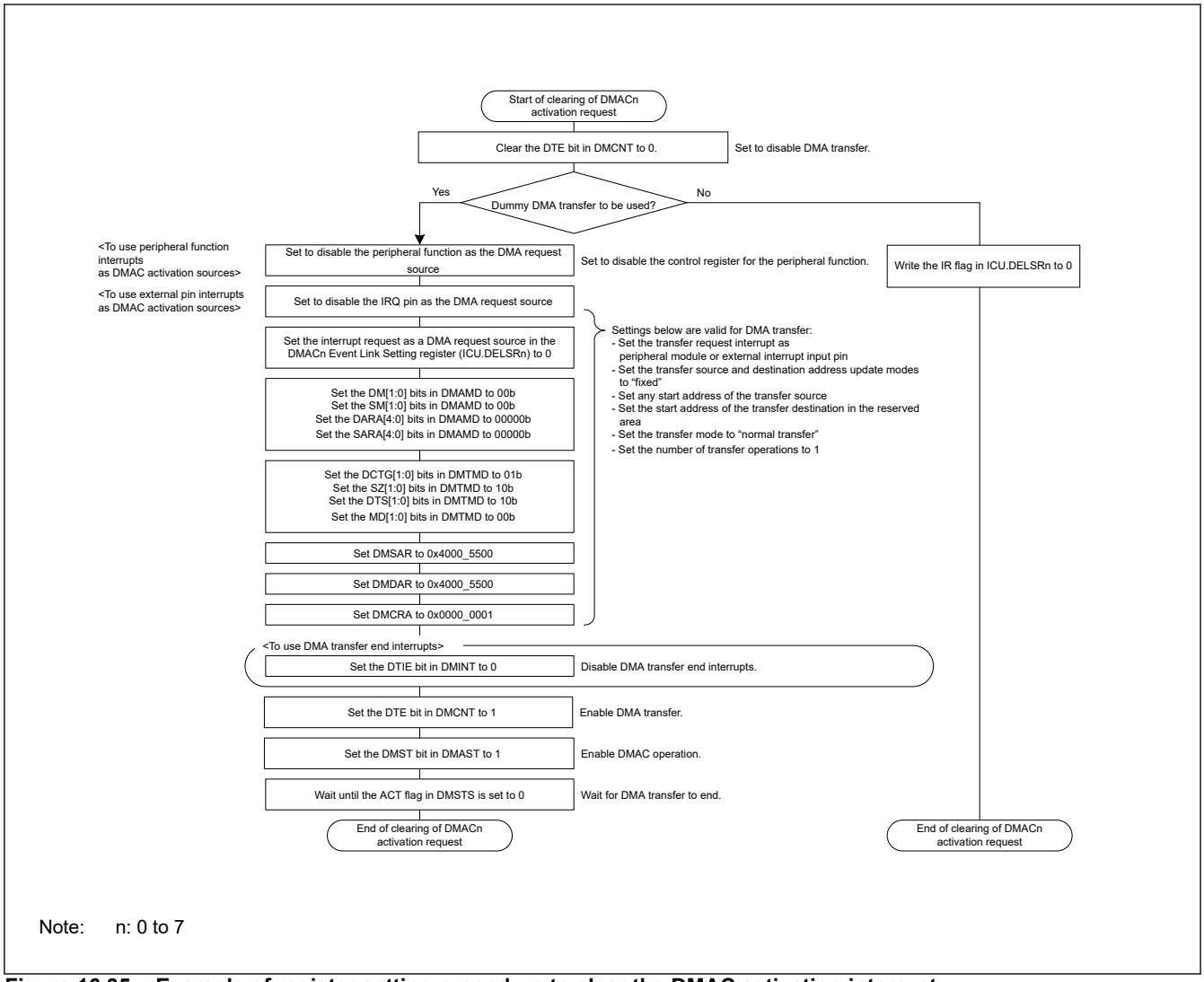
### 16.9.5 Precautions for Resuming DMA Transfer

A DMAC activation request might occur in the next request after a DMA transfer completes. If this happens, the DMA transfer starts and the DMAC activation request is held in the DMAC. To prevent this, stop the DMAC activation requests by setting the DELSRn.DELS[8:0] bits in the ICU to 0.

When a DMAC activation request occurs after the last round of the DMA transfer is generated, clear the DMAC activation request with either of the following approaches.

- Clear the DMAC activation request with a DMA dummy transfer.
- Set the DMCNT.DTE bit to 0 and then set the ICU.DELSRn.IR flag to 0.

See Figure 16.35.




Note: n: 0 to 7

**Figure 16.35 Example of register setting procedure to clear the DMAC activation interrupt**

# 17. Data Transfer Controller (DTC)

## 17.1 Overview

A Data Transfer Controller (DTC) module is provided for transferring data when activated by an interrupt request.

Table 17.1 lists the DTC specifications and Figure 17.1 shows DTC block diagram.

**Table 17.1 DTC specifications**

| Parameter | Description |
|---|---|
| Transfer modes | • Normal transfer mode<br>A single activation leads to a single data transfer.<br>• Repeat transfer mode<br>A single activation leads to a single data transfer.<br>The transfer address returns to the start address after the number of data transfers reaches the specified repeat size.<br>The maximum number of repeat transfers is 256 and the maximum data transfer size is 256 × 32 bits (1024 bytes)<br>• Block transfer mode<br>A single activation leads to a transfer of a single block.<br>The maximum block size is 256 × 32 bits = 1024 bytes. |
| Transfer channel | • Channel transfer can be associated with the interrupt source (transferred by a DTC activation request from the ICU)<br>• Multiple data units can be transferred on a single activation source (chain transfer)<br>• Chain transfers are selectable to either execute when the counter is 0, or always execute. |
| Transfer space | • 4 GB area from 0x0000_0000 to 0xFFFF_FFFF, excluding reserved areas |
| Data transfer units | • Single data unit: 1 byte (8 bits), 1 halfword (16 bits), 1 word (32 bits)<br>• Single block size: 1 to 256 data units. |
| CPU interrupt source | • An interrupt request can be generated to the CPU on a DTC activation interrupt<br>• An interrupt request can be generated to the CPU after a single data transfer<br>• An interrupt request can be generated to the CPU after a data transfer of a specified volume. |
| Processing on DTC transfer error | • When the DTC transfer error occurs, it stops the transfer that caused the error<br>• Request to clear the register for activation request of DTC error number to ICU |
| Error response detection interrupt | Generated when the DTC transfer error occurs |
| Event link function | An event link request is generated after one data transfer (for block, after one block transfer) |
| Read skip | Read of transfer information can be skipped |
| Write-back skip | When the transfer source or destination address is specified as fixed, a write-back of transfer information can be skipped |
| TrustZone | TrustZone violation area of Flash and SRAM is detected in advance before access the bus. |
| Module-stop function | Module-stop state can be set to reduce power consumption |
| TrustZone Filter | Security attribution can be set for each activation source |

Note: Security attribution Regiser of DTC is described in ICU.ICUSARG, ICU.ICUSARH and ICU.ICUSARI

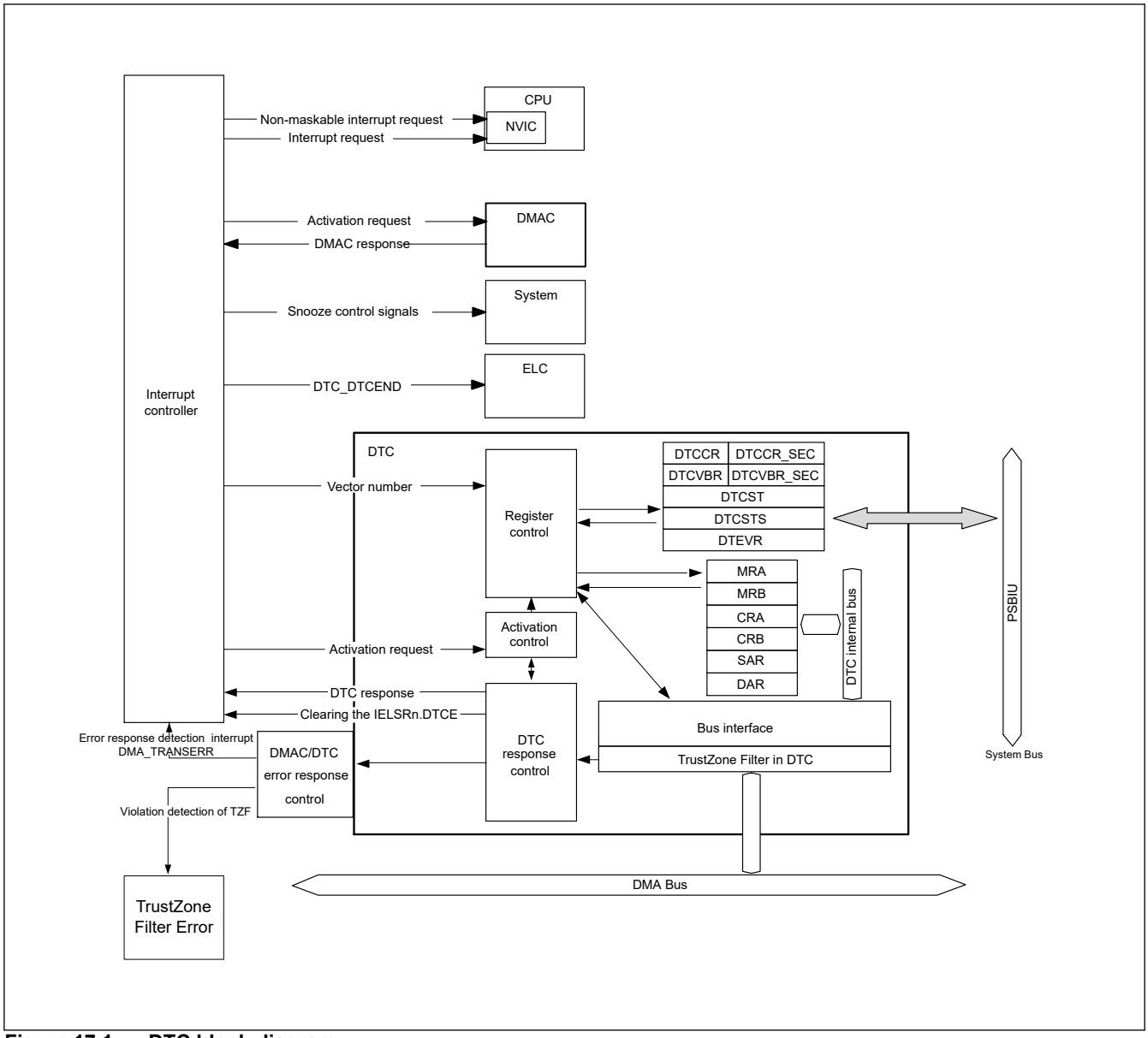



**Figure 17.1 DTC block diagram**

See section 13.1. Overview in section 13, Interrupt Controller Unit (ICU) for the connections between the DTC and NVIC in the CPU.

## 17.2 Register Descriptions

MRA, MRB, SAR, DAR, CRA, and CRB are all DTC internal registers that cannot be directly accessed from the CPU. Values to be set in these DTC internal registers are placed in the SRAM area as transfer information. When an activation request is generated, the DTC reads the transfer information from the SRAM area and sets it in its internal registers. After the data transfer ends, the internal register contents are written back to the SRAM area as transfer information.

## 17.2.1 DTCSAR : DTC Controller Security Attribution Register

Base address: CPSCU = 0x4000_8000

Offset address: 0x30

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | DTCSTSA |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DTCSTSA | DTC Security Attribution<br>0: Secure.<br>1: Non-Secure. | R/W |
| 31:1 | — | This bit is read as 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

This register only sets the DTCST security attribute.

### DTCSTSA bit (DTC Security Attribution)

Security attributes of registers for DTCST.

Do not write to the DTCSTSA bit while DTC transfer is enabled or a bus master is writing to the DTC registers.

## 17.2.2 MRA : DTC Mode Register A

Base address: DTCVBR

Offset address: 0x03 + 0x4 × Vector number
(Inaccessible directly from the CPU. See section 17.3.1. Allocating Transfer Information and DTC Vector Table)

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | MD[1:0] | | SZ[1:0] | | SM[1:0] | | — | — |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | — | The read values are undefined. The write value should be 0. | — |
| 3:2 | SM[1:0] | Transfer Source Address Addressing Mode<br>0 0: Address in the SAR register is fixed (write-back to SAR is skipped.)<br>0 1: Address in the SAR register is fixed (write-back to SAR is skipped.)<br>1 0: SAR value is incremented after data transfer:<br>+1 when SZ[1:0] = 00b<br>+2 when SZ[1:0] = 01b<br>+4 when SZ[1:0] = 10b<br>1 1: SAR value is decremented after data transfer:<br>-1 when SZ[1:0] = 00b<br>-2 when SZ[1:0] = 01b<br>-4 when SZ[1:0] = 10b | — |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:4 | SZ[1:0] | DTC Data Transfer Size<br>0 0: Byte (8-bit) transfer<br>0 1: Halfword (16-bit) transfer<br>1 0: Word (32-bit) transfer<br>1 1: Setting prohibited | — |
| 7:6 | MD[1:0] | DTC Transfer Mode Select<br>0 0: Normal transfer mode<br>0 1: Repeat transfer mode<br>1 0: Block transfer mode<br>1 1: Setting prohibited | — |

The MRA register cannot be accessed directly from the CPU, however the CPU can access the SRAM area (transfer information (n) start address + 0x03) and DTC transfers it automatically to and from the MRA register. See section 17.3.1. Allocating Transfer Information and DTC Vector Table.

### 17.2.3 MRB : DTC Mode Register B

Base address: DTCVBR

Offset address: 0x02 + 0x4 × Vector number
(Inaccessible directly from the CPU. See section 17.3.1. Allocating Transfer Information and DTC Vector Table)

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CHNE | CHNS | DISEL | DTS | DM[1:0] | | — | — |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | — | The read values are undefined. The write value should be 0. | — |
| 3:2 | DM[1:0] | Transfer Destination Address Addressing Mode<br>0 0: Address in the DAR register is fixed (write-back to DAR is skipped)<br>0 1: Address in the DAR register is fixed (write-back to DAR is skipped)<br>1 0: DAR value is incremented after data transfer:<br>+1 when MRA.SZ[1:0] = 00b<br>+2 when SZ[1:0] = 01b<br>+4 when SZ[1:0] = 10b<br>1 1: DAR value is decremented after data transfer:<br>-1 when MRA.SZ[1:0] = 00b<br>-2 when SZ[1:0] = 01b<br>-4 when SZ[1:0] = 10b | — |
| 4 | DTS | DTC Transfer Mode Select<br>0: Select transfer destination as repeat or block area.<br>1: Select transfer source as repeat or block area. | — |
| 5 | DISEL | DTC Interrupt Select<br>0: Generate an interrupt request to the CPU when specified data transfer is complete.<br>1: Generate an interrupt request to the CPU each time DTC data transfer is performed. | — |
| 6 | CHNS | DTC Chain Transfer Select<br>0: Chain transfer is continuous.<br>1: Chain transfer occurs only when the transfer counter changes from 1 to 0 or 1 to CRAH. | — |
| 7 | CHNE | DTC Chain Transfer Enable<br>0: Chain transfer is disabled.<br>1: Chain transfer is enabled. | — |

The MRB register cannot be accessed directly from the CPU, however the CPU can access the SRAM area (transfer information (n) start address + 0x02) and DTC transfers it automatically to and from the MRB register. See section 17.3.1. Allocating Transfer Information and DTC Vector Table.

#### DM[1:0] bits (Transfer Destination Address Addressing Mode)

The DM[1:0] bits are to fix the address of the DAR register or specify increment / decrement of the DAR register after transfer.

#### DTS bit (DTC Transfer Mode Select)

The DTS bit specifies whether the transfer source or destination is the repeat or block area in repeat or block transfer mode.

#### DISEL bit (DTC Interrupt Select)

The DISEL bit specifies the condition for generating an interrupt request to the CPU.

#### CHNS bit (DTC Chain Transfer Select)

The CHNS bit selects the chain transfer condition. When CHNE is 0, the CHNS setting is ignored. For details on the conditions for chain transfer, see Table 17.3.

When the next transfer is chain transfer, completion of the specified number of transfers is not determined, the activation source flag is not cleared, and an interrupt request to the CPU is not generated.

#### CHNE bit (DTC Chain Transfer Enable)

The CHNE bit enables chain transfer. The chain transfer condition is selected by the CHNS bit. For details on chain transfer, see section 17.4.6. Chain Transfer.

### 17.2.4 SAR : DTC Transfer Source Register

Base address: DTCVBR

Offset address: 0x04 + 0x4 × Vector number
(Inaccessible directly from the CPU. See section 17.3.1. Allocating Transfer Information and DTC Vector Table)

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

The SAR sets the transfer source start address and cannot be accessed directly from the CPU. However, the CPU can access the SRAM area (transfer information (n) start address + 0x04) and DTC transfers it automatically to and from the SAR register. See section 17.3.1. Allocating Transfer Information and DTC Vector Table.

Misalignment is prohibited for DTC transfers. Bit[0] must be 0 when MRA.SZ[1:0] = 01b, and bit[1] and bit[0] must be 0 when MRA.SZ[1:0] = 10b.

### 17.2.5 DAR : DTC Transfer Destination Register

Base address: DTCVBR

Offset address: 0x08 + 0x4 × Vector number
(Inaccessible directly from the CPU. See section 17.3.1. Allocating Transfer Information and DTC Vector Table)

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

The DAR sets the transfer destination start address and cannot be accessed directly from the CPU. However, the CPU can access the SRAM area (transfer information (n) start address + 0x08) and DTC transfers it automatically to and from the DAR register. See section 17.3.1. Allocating Transfer Information and DTC Vector Table.

Misalignment is prohibited for DTC transfers. Bit[0] must be 0 when MRA.SZ[1:0] = 01b, and bit[1] and bit[0] must be 0 when MRA.SZ[1:0] = 10b.

## 17.2.6 CRA : DTC Transfer Count Register A

Base address: DTCVBR

Offset address: 0x0E + 0x4 × Vector number
(Inaccessible directly from the CPU. See section 17.3.1. Allocating Transfer Information and DTC Vector Table)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | CRAL | Transfer Counter A Lower Register<br>Specify the transfer count. | — |
| 15:8 | CRAH | Transfer Counter A Upper Register<br>Specify the transfer count. | — |

Note: The function depends on the transfer mode.
Note: Set CRAH and CRAL to the same value in repeat transfer mode and block transfer mode.

The CRA register consists of 16 bits. CRAL is the lower 8 bits and CRAH is the upper 8 bits. CRA is used in normal mode.

CRAL and CRAH are used in repeat transfer mode and block transfer mode.

The CRA register cannot be accessed directly from the CPU. However, the CPU can access the SRAM area (transfer information (n) start address + 0x0E) and DTC transfers it automatically to and from the CRA register. See section 17.3.1. Allocating Transfer Information and DTC Vector Table.

### (1) Normal transfer mode (MRA.MD[1:0] = 00b)

In normal transfer mode, CRA functions as a 16-bit transfer counter. The transfer count is 1, 65535, and 65536 when the set value is 0x0001, 0xFFFF, and 0x0000, respectively. The CRA value is decremented (-1) on each data transfer.

### (2) Repeat transfer mode (MRA.MD[1:0] = 01b)

In repeat transfer mode, the CRAH register holds the transfer count and the CRAL register functions as an 8-bit transfer counter. The transfer count is 1, 255, and 256 when the set value is 0x01, 0xFF, and 0x00, respectively. The CRAL value is decremented (-1) on each data transfer. When it reaches 0x00, the CRAH value is transferred to CRAL.

### (3) Block transfer mode (MRA.MD[1:0] = 10b)

In block transfer mode, the CRAH register holds the block size and the CRAL register functions as an 8-bit block size counter. The transfer count is 1, 255, and 256 when the set value is 0x01, 0xFF, and 0x00, respectively. The CRAL value is decremented (-1) on each data transfer. When it reaches 0x00, the CRAH value is transferred to CRAL.

## 17.2.7 CRB : DTC Transfer Count Register B

Base address: DTCVBR

Offset address: 0x0C + 0x4 × Vector number
(Inaccessible directly from the CPU. See section 17.3.1. Allocating Transfer Information and DTC Vector Table)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

The CRB sets the block transfer count for block transfer mode. The transfer count is 1, 65535, and 65536 when the set value is 0x0001, 0xFFFF, and 0x0000, respectively. The CRB value is decremented (-1) when the final data of a single block size is transferred. When normal transfer mode or repeat transfer mode is selected, this register is not used, and the set value is ignored.

The CRB cannot be accessed directly from the CPU. However, the CPU can access the SRAM area (transfer information (n) start address + 0x0C) and DTC transfers it automatically to and from the CRB register. See section 17.3.1. Allocating Transfer Information and DTC Vector Table.

## 17.2.8 DTCCR : DTC Control Register

Base address: DTC = 0x4000_5400

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | RRS | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 3 | — | This bit is read as 1. The write value should be 1. | R/W |
| 4 | RRS | DTC Transfer Information Read Skip Enable<br>0: Transfer information read is not skipped<br>1: Transfer information read is skipped when vector numbers match | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

### RRS bit (DTC Transfer Information Read Skip Enable)

The RRS bit enables skipping of transfer information reads when vector numbers match. The DTC vector number is compared with the vector number in the previous activation process. When these vector numbers match and the RRS bit is set to 1, DTC data transfer is performed without reading the transfer information. However, when the previous transfer is a chain transfer, the transfer information is read regardless of the RRS bit.

When the transfer counter (CRA register) becomes 0 during the previous normal transfer and when the transfer counter (CRB register) becomes 0 during the previous block transfer, the transfer information is read regardless of the RRS bit value.

## 17.2.9 DTCCR_SEC : DTC Control Register for secure Region

Base address: DTC = 0x4000_5400

Offset address: 0x10

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | RRS | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 3 | — | This bit is read as 1. The write value should be 1. | R/W |
| 4 | RRS | DTC Transfer Information Read Skip Enable for Secure<br>0: Transfer information read is not skipped.<br>1: Transfer information read is skipped when vector numbers match. | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Secure access is allowed. Non-secure access is read-only.

### RRS bit (DTC Transfer Information Read Skip Enable for Secure)

The RRS bit enables skipping of transfer information reads when vector numbers match. The DTC vector number is compared with the vector number in the previous activation process. When these vector numbers match and the RRS bit is

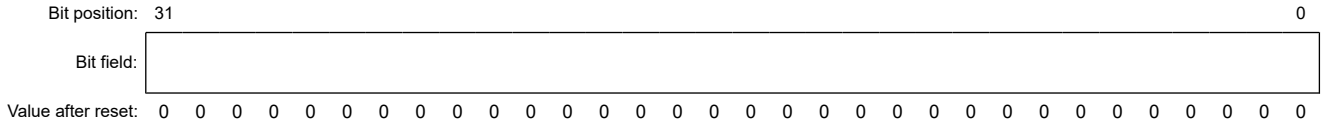
set to 1, DTC data transfer is performed without reading the transfer information. However, when the previous transfer is a chain transfer, the transfer information is read regardless of the RRS bit.

When the transfer counter (CRA register) becomes 0 during the previous normal transfer and when the transfer counter (CRB register) becomes 0 during the previous block transfer, the transfer information is read regardless of the RRS bit value.

## 17.2.10 DTCVBR : DTC Vector Base Register

Base address: DTC = 0x4000_5400

Offset address: 0x04



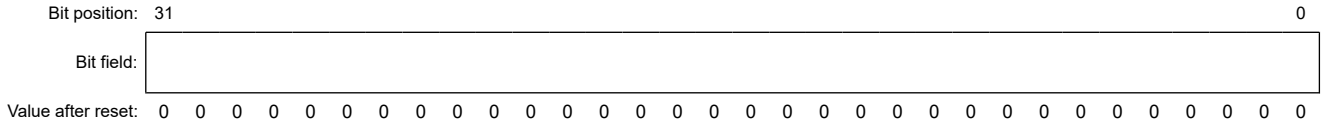

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | DTC Vector Base Address<br>Set the DTC vector base address. The lower 10 bits should be 0. | R/W |

The DTCVBR sets the base address for calculating the DTC vector table address, which can be set in the range of 0x0000_0000 to 0xFFFF_FFFF (4 GB) in 1-KB units.

## 17.2.11 DTCVBR_SEC : DTC Vector Base Register for secure Region

Base address: DTC = 0x4000_5400

Offset address: 0x14




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | DTC Vector Base Address for secure region<br>Set DTC Vector Base Address for secure region. The lower 10 bits should be 0. | R/W |

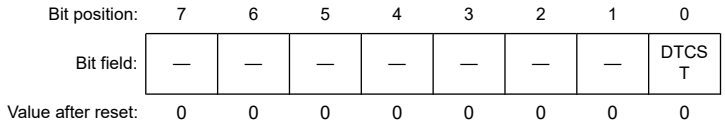
Note: Secure access is allowed. Non-secure access is read-only.

The DTCVBR_SEC sets the base address for calculating the DTC vector table address, which can be set in the range of 0x0000_0000 to 0xFFFF_FFFF (4 GB) in 1-KB units.

## 17.2.12 DTCST : DTC Module Start Register

Base address: DTC = 0x4000_5400

Offset address: 0x0C




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DTCST | DTC Module Start<br>0: DTC module stopped.<br>1: DTC module started. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### DTCST bit (DTC Module Start)

Set the DTCST bit to 1 to enable the DTC to accept transfer requests. When this bit is set to 0, transfer requests are no longer accepted. If this bit is set to 0 during a data transfer, the accepted transfer request is active until processing completes.

DTCST must be set to 0 before transitioning to one of the following state or mode:

- Module-stop state
- Software Standby mode without Snooze mode transition
- Deep Software standby mode

For details on these transitions, see section 17.10. Low Power Consumption Function and section 10, Low Power Modes.

## 17.2.13 DTCSTS : DTC Status Register

Base address: DTC = 0x4000_5400

Offset address: 0x0E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ACT | — | — | — | — | — | — | — | VECN[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | VECN[7:0] | DTC-Activating Vector Number Monitoring<br>These bits indicate the vector number for the activation source when a DTC transfer is in progress.<br>The value is only valid if a DTC transfer is in progress (ACT flag is 1). | R |
| 14:8 | — | These bits are read as 0. | R |
| 15 | ACT | DTC Active Flag<br>0: DTC transfer operation is not in progress.<br>1: DTC transfer operation is in progress. | R |

### VECN[7:0] bits (DTC-Activating Vector Number Monitoring)

While transfer by the DTC is in progress, the VECN[7:0] bits indicate the vector number associated with the activation source for the transfer. The value read from the VECN[7:0] bits is valid if the ACT flag is 1, indicating a DTC transfer in progress, and invalid if the ACT flag is 0, indicating no DTC transfer is in progress.

### ACT flag (DTC Active Flag)

The ACT flag indicates the state of the DTC transfer operation.

[Setting condition]

- When the DTC is activated by a transfer request.

[Clearing condition]

- When transfer by the DTC, in response to a transfer request, is complete.

## 17.2.14 DTEVR : DTC Error Vector Register

Base address: DTC = 0x4000_5400

Offset address: 0x20

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | DTESTA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | DTEVSAM | DTEV[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | DTEV[7:0] | DTC Error Vector Number<br>These bits represent error vector of the DTC. | R |
| 8 | DTEVSAM | DTC Error Vector Number SA Monitor<br>Indicates the SA of vector number causing the error.<br>0: Secure vector number<br>1: Non-Secure vector number | R |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R |
| 16 | DTESTA | DTC Error Status Flag<br>0: No DTC transfer error occurred<br>1: DTC transfer error occurred | R/W |
| 31:17 | — | These bits are read as 0. The write value should be 0. | R |

Note: Writing to DTESTA depends on the value of DTEVSAM

### DTEV[7:0] bits (DTC Error Vector Number)

When a transfer error due to DTC transfer occurs, the DTEV[7:0] bits store the violating DTC channel.

When reset is selected in MPU.MMPUOAD.OAD and TZF.TZFOAD.OAD, this register is also reset. Select NMI when you want to debug the program.

[Set condition]

- When the DTC transfer error occurs and DTESTA = 0.

[Clearing condition]

- When 1 is written to DTEVR.DTESTA.

### DTEVSAM bit (DTC Error Vector Number SA Monitor)

When a transfer error due to DTC transfer occurs, the DTEVSAM bit indicates the SA of the violating DTC vector number.

When reset is selected in MPU.MMPUOAD.OAD and TZF.TZFOAD.OAD, this register is also reset. Select NMI when you want to debug the program.

[Set condition]

- When the DTC transfer error occurs and DTESTA = 0.

[Clearing condition].

- When 1 is written to DTEVR.DTESTA.

### DTESTA bit (DTC Error Status Flag)

The DTESTA bit indicates whether or not a DTC transfer error occurred.

DTEV, DTEVSAM, DTESTA are cleared by writing 1 to DTESTA.

Writing 0 to DTESTA is ignored.

When reset is selected in MPU.MMPUOAD.OAD and TZF.TZFOAD.OAD, this register is also reset. Select NMI when you want to debug the program.

[Set condition]

- When the DMAC transfer error occurs.

[Clearing condition]

- When 1 is written to DTEVR.DTESTA.

Note: When DTEVSAM = 1, it can be cleared in the secure state and non-secure state. DTEVSAM = 0, it cannot be cleared in the non-secure state.

## 17.3 Activation Sources

The DTC is activated by an interrupt request. Setting the ICU.IELSRn.DTCE bit to 1 enables activation of the DTC by the associated interrupt. The selector output *n* number set in ICU.IELSRn is defined as the interrupt vector number, where n = 0 to 95. For an enabled interrupt, the specific DTC interrupt source associated with each interrupt vector number *n* is selected in ICU.IELSRn.IELS[8:0] where n = 0 to 95, as listed in section 13.3.2. Event Number in section 13, Interrupt Controller Unit (ICU). For activation by software, see section 18.2.2. ELSEGRn : Event Link Software Event Generation Register n (n = 0, 1).

The interrupt vector number is equivalent to the DTC vector table number. After the DTC accepted an activation request, it does not accept another activation request until the transfer for that single request is complete, regardless of the priority of the requests. When multiple activation requests are generated during a DTC transfer, the highest priority request is accepted on completion of the transfer. When multiple activation requests are generated while the DTC Module Start bit (DTCST.DTCST) is 0, the DTC accepts the highest priority request when DTCST.DTCST is subsequently set to 1. The smaller interrupt vector number has higher priority.

The DTC performs the following operations at the start of a single data transfer or for a chain transfer, after the last of the consecutive transfers:

- On completion of a specified round of data transfer, the ICU.IELSRn.DTCE bit is set to 0, and an interrupt request is sent to the CPU.
- If the MRB.DISEL bit is 1, an interrupt request is sent to the CPU on completion of a data transfer.
- For other transfers, the ICU.IELSRn.IR flag of the activation source is set to 0 at the start of the data transfer.

### 17.3.1 Allocating Transfer Information and DTC Vector Table

The DTC reads the start address of the transfer information associated with each activation source from the vector table and reads the transfer information starting at that address.

DTC has two vector tables, non-secure side or secure side. Because the interrupt vector number that serves as a trigger for DTC is divided into non-secure or secure. Place the vector table of the interrupt vector number of SA = 1 in DTCVBR which is the non-secure side. Place the vector table of interrupt number SA = 0 in DTCVBR_SEC which is the secure side.

The vector table must be located so that the lower 10 bits of the base address (start address) are 0. Use the DTC Vector Base Register (DTCVBR) to set the base address of the DTC vector table. Transfer information is allocated in the SRAM area. In the SRAM area, the start address of the transfer information *n* with vector number *n* must be 4n added to the base address in the vector table.

Figure 17.2 shows the relationship between the DTC vector table and transfer information. Figure 17.3 shows the allocation of transfer information in the SRAM area.

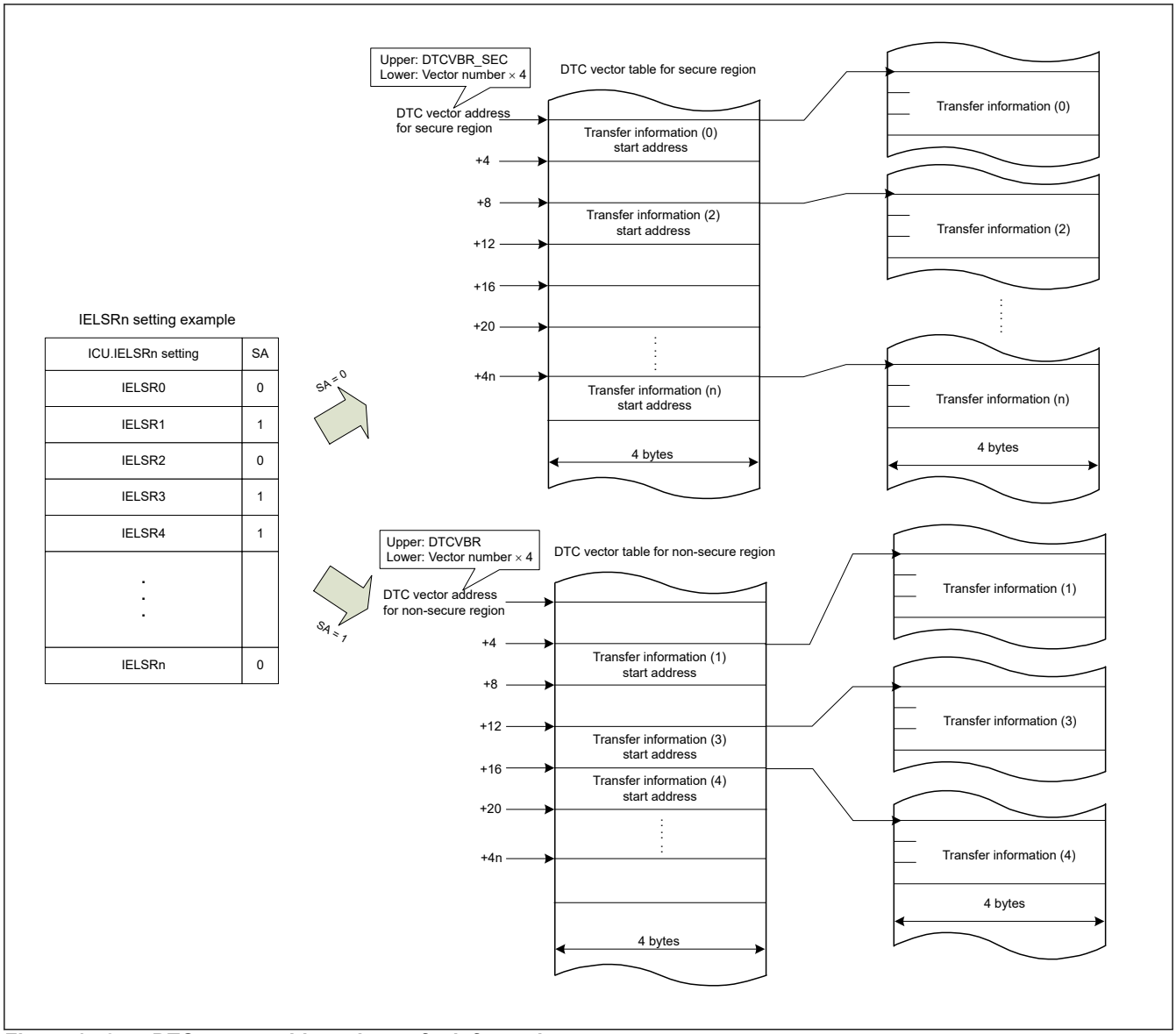



**Figure 17.2 DTC vector table and transfer information**

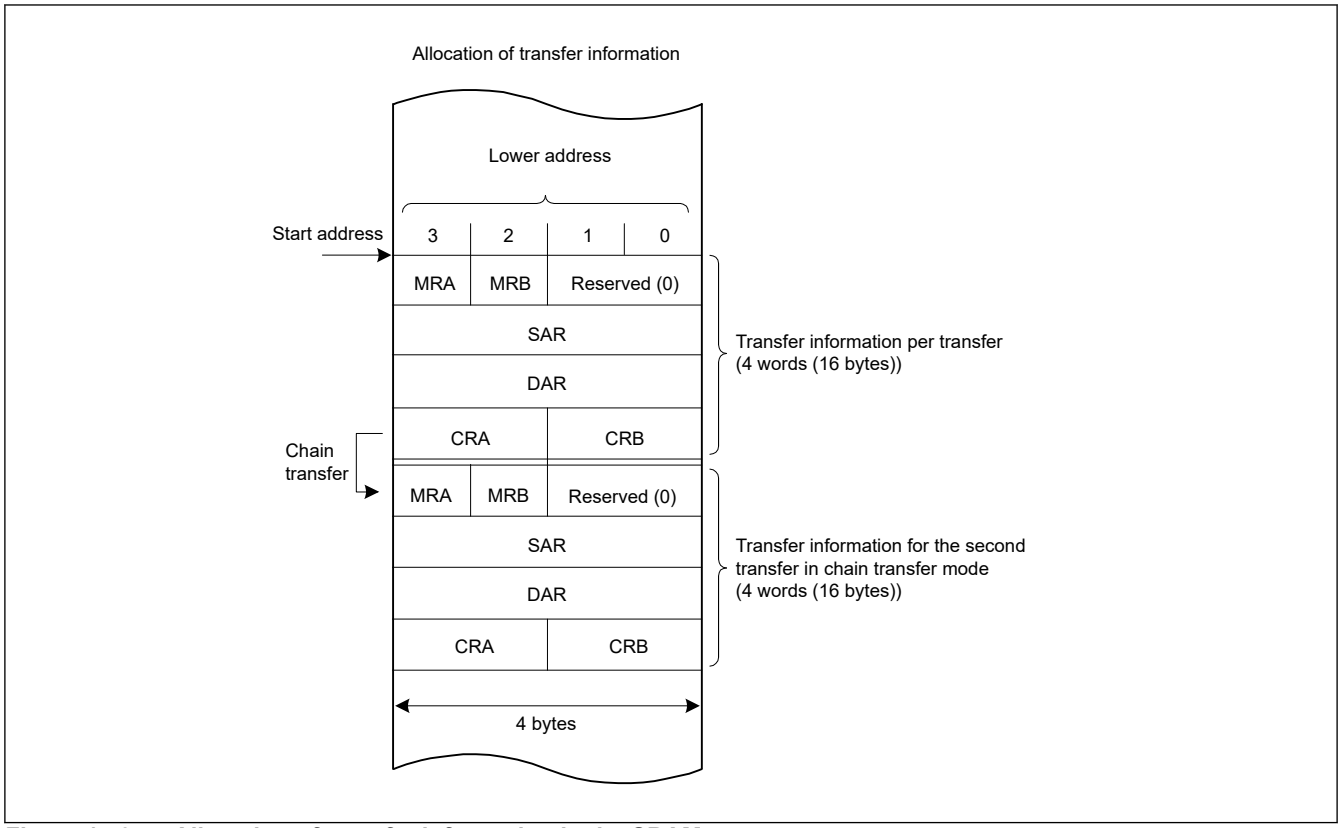



**Figure 17.3 Allocation of transfer information in the SRAM area**

## 17.4 Operation

The DTC transfers data according to the transfer information. Storage of the transfer information in the SRAM area is required before a DTC operation. When the DTC is activated, it reads the DTC vector associated with the vector number. The DTC reads the transfer information from the transfer information store address referenced by the DTC vector and transfers the data. After the data transfer, the DTC writes back the transfer information. Storing the transfer information in the SRAM area allows data transfer of any number of channels.

The transfer modes include:

- Normal transfer mode
- Repeat transfer mode
- Block transfer mode.

The DTC specifies a transfer source address in the SAR register and a transfer destination address in the DAR register. The values of these registers are incremented, decremented, or address-fixed independently after the data transfer.

Table 17.2 describes the DTC transfer modes.

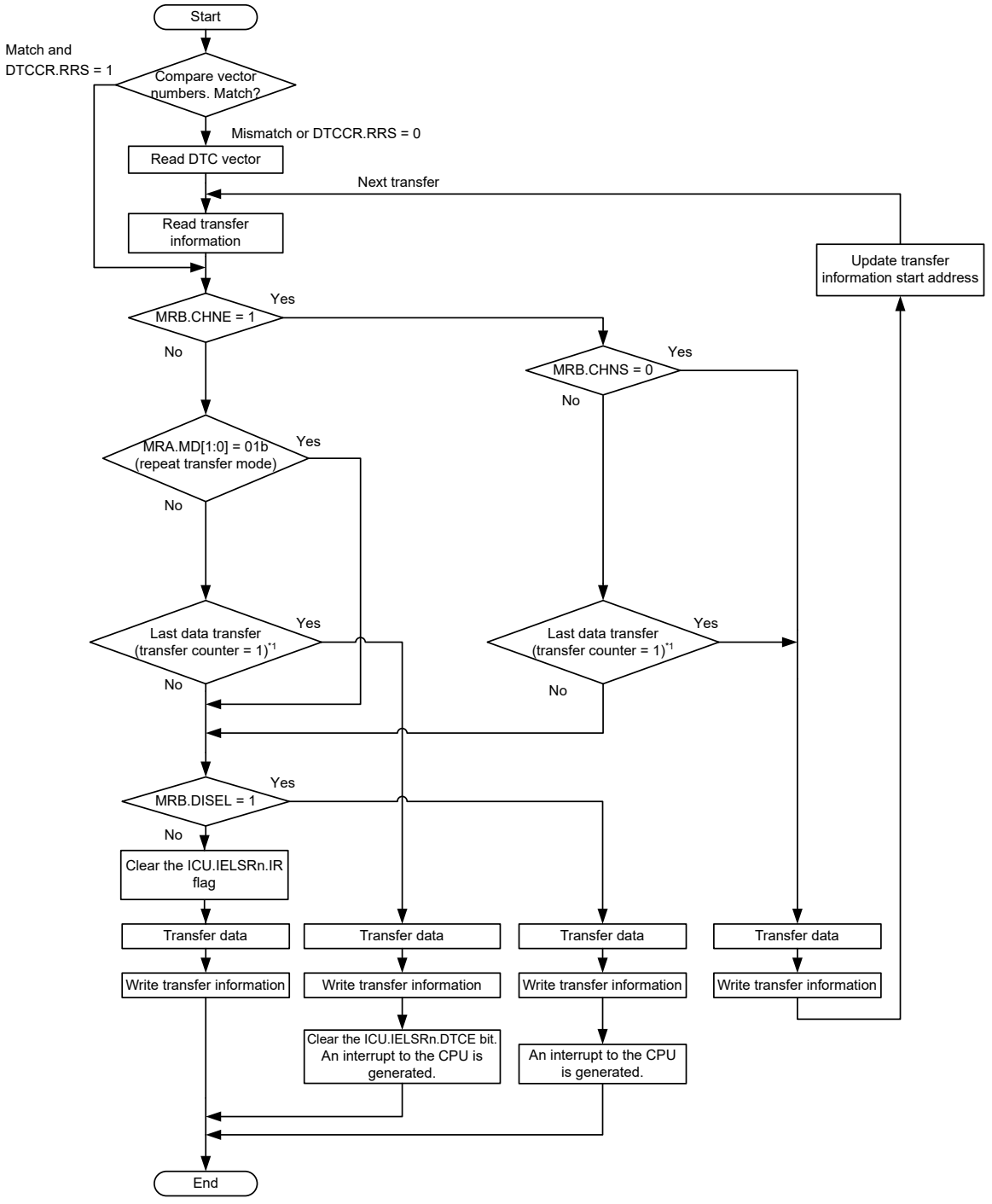
**Table 17.2 DTC transfer modes**

| Transfer mode | Data size transferred on single transfer request | Increment or decrement of memory address | Settable transfer count |
|---|---|---|---|
| Normal transfer mode | 1 byte (8 bit), 1 halfword (16 bit), 1 word (32 bit) | Incremented or decremented by 1, 2, or 4 or address-fixed | 1 to 65536 |
| Repeat transfer mode*1 | 1 byte (8 bit), 1 halfword (16 bit), 1 word (32 bit) | Incremented or decremented by 1, 2, or 4 or address-fixed | 1 to 256*3 |
| Block transfer mode*2 | Block size specified in CRAH (1 to 256 bytes, 1 to 256 halfwords (2 to 512 bytes), or 1 to 256 words (4 to 1024 bytes)) | Incremented or decremented by 1, 2, or 4 or address-fixed | 1 to 65536 |

Note 1. Set the transfer source or transfer destination as the repeat area.
Note 2. Set the transfer source or transfer destination as the block area.
Note 3. After a data transfer of the specified count, the initial state is restored and operation restarts.

Setting the MRB.CHNE bit to 1 allows multiple transfers or chain transfer on a single activation source. It also enables a chain transfer when the specified data transfer is complete.

Figure 17.4 shows the operation flow of the DTC. Table 17.3 lists the chain transfer conditions. The combination of control information for the second and subsequent transfers are omitted in this table.




Note 1. Counter value before starting data transfer.

**Figure 17.4 DTC operation flow**

**Table 17.3 Chain transfer conditions**

| First transfer | | | | Second transfer*3 | | | | |
|---|---|---|---|---|---|---|---|---|
| CHNE bit | CHNS bit | DISEL bit | Transfer counter*1 *2 | CHNE bit | CHNS bit | DISEL bit | Transfer counter*1 *2 | DTC transfer |
| 0 | — | 0 | Other than (1 → 0) | — | — | — | — | Ends after the first transfer |
| 0 | — | 0 | (1 → 0) | — | — | — | — | Ends after the first transfer with an interrupt request to the CPU |
| 0 | — | 1 | — | — | — | — | — | |
| 1 | 0 | — | — | 0 | — | 0 | Other than (1 → 0) | Ends after the second transfer |
| | | | | 0 | — | 0 | (1 → 0) | Ends after the second transfer with an interrupt request to the CPU |
| | | | | 0 | — | 1 | — | |
| 1 | 1 | 0 | Other than (1 → *) | — | — | — | — | Ends after the first transfer |
| 1 | 1 | — | (1 → *) | 0 | — | 0 | Other than (1 → 0) | Ends after the second transfer |
| | | | | 0 | — | 0 | (1 → 0) | Ends after the second transfer with an interrupt request to the CPU |
| | | | | 0 | — | 1 | — | |
| 1 | 1 | 1 | Other than (1 → *) | — | — | — | — | Ends after the first transfer with an interrupt request to the CPU |

Note 1. The transfer counter used depends on the transfer modes as follows:
Normal transfer mode — CRA register
Repeat transfer mode — CRAL register
Block transfer mode — CRB register

Note 2. On completion of a data transfer, the counters operate as follows:
1 → 0 in normal and block transfer modes
1 → CRAH in repeat transfer mode
(1 → *) in the table indicates both of these two operations, depending on the mode.

Note 3. Chain transfer can be selected for the second or subsequent transfers. The conditions for the combination of the second transfer and CHNE = 1 is omitted.

### 17.4.1 Transfer Information Read Skip Function

Reading of vector addresses and transfer information can be skipped by setting the DTCCR.RRS bit. When a DTC activation request is generated, the current DTC vector number is compared with the DTC vector number in the previous activation process. When these vector numbers match and the RRS bit is set to 1, the DTC data transfer is performed without reading the vector address and transfer information. However, when the previous transfer is a chain transfer, the vector address and transfer information are read. Additionally, when the transfer counter (CRA register) becomes 0 during the previous normal transfer, and when the transfer counter (CRB register) becomes 0 during the previous block transfer, transfer information is read regardless of the RRS bit. Figure 17.12 shows an example when reading the transfer information is skipped.

To update the vector table and transfer information, set the RRS bit to 0, update the vector table and transfer information, then set the RRS bit to 1. The stored vector number is discarded by setting the RRS bit to 0. The updated DTC vector table and transfer information are read in the next activation process.

### 17.4.2 Transfer Information Write-Back Skip Function

When the MRA.SM[1:0] bits or the MRB.DM[1:0] bits are set to address fixed, a part of the transfer information is not written back. Table 17.4 lists the transfer information write-back skip conditions and the associated registers. The CRA and CRB registers are written back, and the write-back of the MRA and MRB registers is skipped.

**Table 17.4 Transfer information write-back skip conditions and applicable registers**

| MRA.SM[1:0] bits | | MRB.DM[1:0] bits | | | |
|---|---|---|---|---|---|
| b3 | b2 | b3 | b2 | SAR register | DAR register |
| 0 | 0 | 0 | 0 | Skip | Skip |
| 0 | 0 | 0 | 1 | | |
| 0 | 1 | 0 | 0 | | |
| 0 | 1 | 0 | 1 | | |
| 0 | 0 | 1 | 0 | Skip | Write-back |
| 0 | 0 | 1 | 1 | | |
| 0 | 1 | 1 | 0 | | |
| 0 | 1 | 1 | 1 | | |
| 1 | 0 | 0 | 0 | Write-back | Skip |
| 1 | 0 | 0 | 1 | | |
| 1 | 1 | 0 | 0 | | |
| 1 | 1 | 0 | 1 | | |
| 1 | 0 | 1 | 0 | Write-back | Write-back |
| 1 | 0 | 1 | 1 | | |
| 1 | 1 | 1 | 0 | | |
| 1 | 1 | 1 | 1 | | |

### 17.4.3 Normal Transfer Mode

The normal transfer mode allows a 1-byte (8 bit), 1-halfword (16 bit), 1-word (32 bit) data transfer on a single activation source. The transfer count can be set from 1 to 65536. Transfer source and destination addresses can be independently set to increment, decrement, or fixed. This mode enables an interrupt request to the CPU to be generated at the end of a specified-count transfer.

Table 17.5 lists register functions in normal transfer mode, and Figure 17.5 shows the memory map of normal transfer mode.

**Table 17.5 Register functions in normal transfer mode**

| Register | Description | Value written back by writing transfer information |
|---|---|---|
| SAR | Transfer source address | Increment, decrement, or fixed*1 |
| DAR | Transfer destination address | Increment, decrement, fixed*1 |
| CRA | Transfer counter A | CRA - 1 |
| CRB | Transfer counter B | Not updated |

Note 1. Write-back operation is skipped in address-fixed mode.

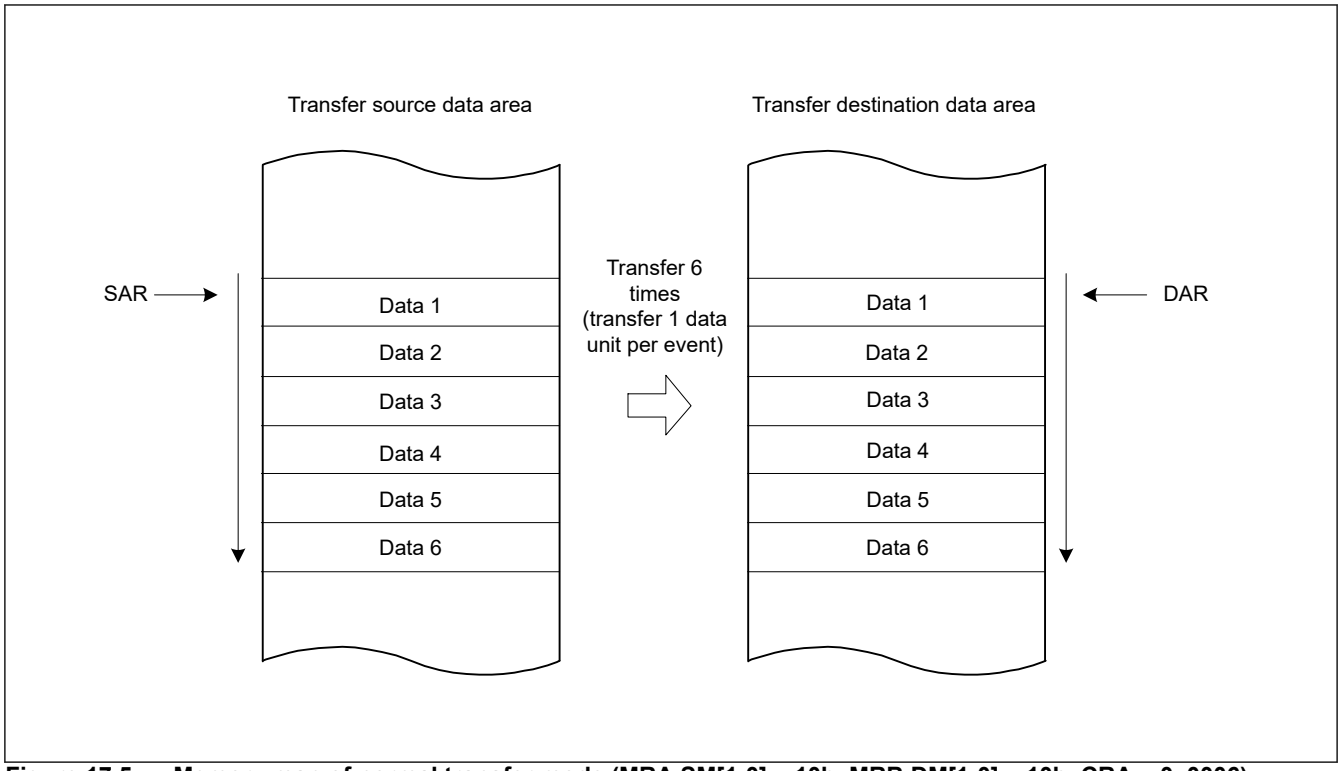



**Figure 17.5 Memory map of normal transfer mode (MRA.SM[1:0] = 10b, MRB.DM[1:0] = 10b, CRA = 0x0006)**

## 17.4.4 Repeat Transfer Mode

The repeat transfer mode allows a 1-byte (8-bit), 1-halfword (16-bit), or 1-word (32-bit) data transfer on a single activation source. Transfer source or transfer destination for the repeat area must be specified in the MRB.DTS bit. The transfer count can be set from 1 to 256. When the specified transfer count is complete, the initial value of the address register specified in the repeat area is restored, the initial value of the transfer counter is restored, and transfer is repeated. The other address register is incremented or decremented continuously or remains unchanged.

When the transfer counter CRAL decrements to 0x00 in repeat transfer mode, the CRAL value is updated to the value set in the CRAH register. As a result, the transfer counter does not clear to 0x00, which disables interrupt requests to the CPU when the MRB.DISEL bit is set to 0. An interrupt request to the CPU is generated when the specified data transfer completes.

Table 17.6 lists the register functions in repeat transfer mode, and Figure 17.6 shows the memory map of repeat transfer mode.

**Table 17.6 Register functions in repeat transfer mode**

| Register | Description | Value written back by writing transfer information | |
|---|---|---|---|
| | | When CRAL is not 1 | When CRAL is 1 |
| SAR | Transfer source address | Increment, decrement, fixed*1 | • When the MRB.DTS bit is 0<br>Increment, decrement, or fixed*1<br>• When the MRB.DTS bit is 1<br>SAR register initial value |
| DAR | Transfer destination address | Increment, decrement, or fixed*1 | • When the MRB.DTS bit is 0<br>DAR register initial value<br>• When the MRB.DTS bit is 1<br>Increment, decrement, or fixed*1 |
| CRAH | Retains transfer counter | CRAH | CRAH |
| CRAL | Transfer counter A | CRAL - 1 | CRAH |
| CRB | Transfer counter B | Not updated | Not updated |

Note 1. Write-back is skipped in address-fixed mode.

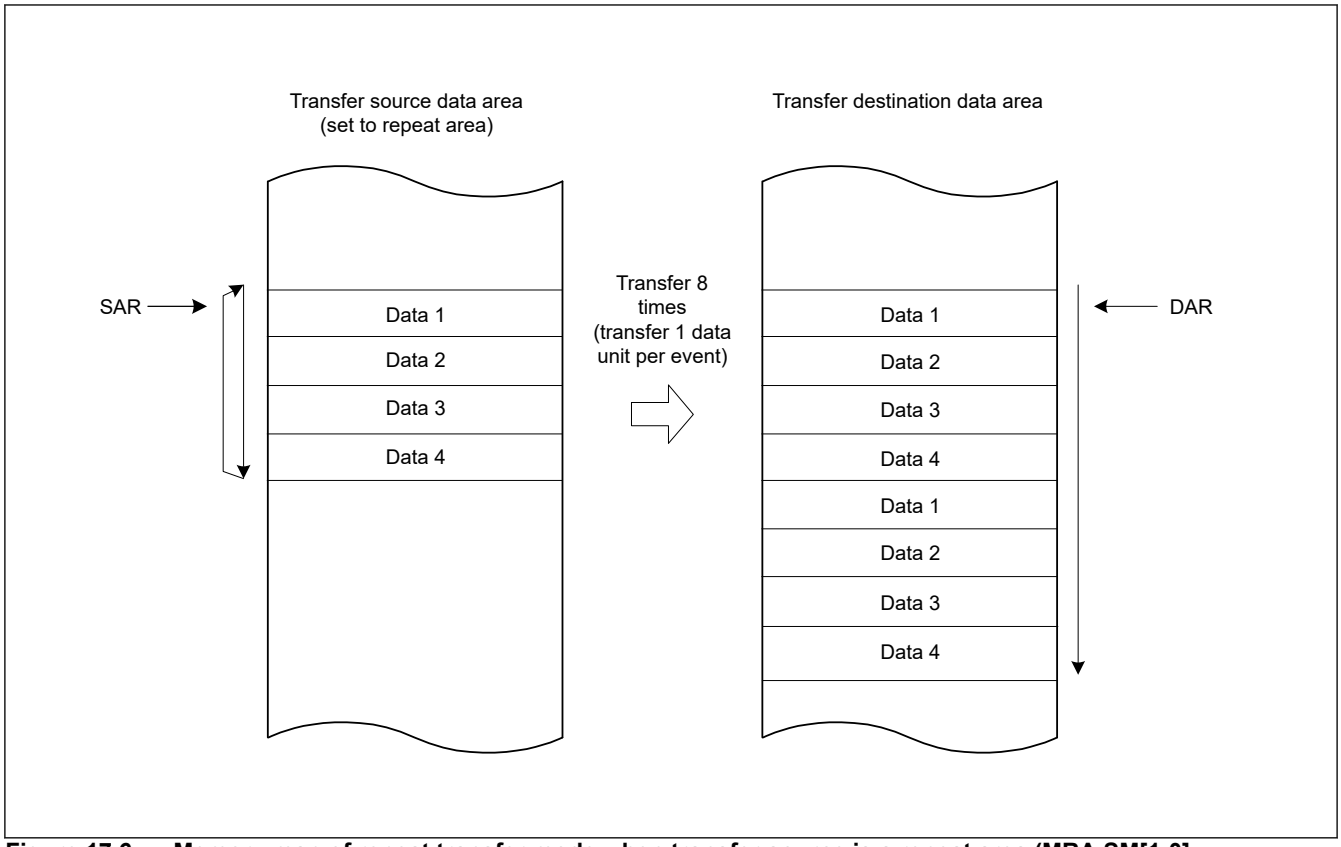



**Figure 17.6 Memory map of repeat transfer mode when transfer source is a repeat area (MRA.SM[1:0] = 10b, MRB.DM[1:0] = 10b, CRAH = 0x04)**

## 17.4.5 Block Transfer Mode

The block transfer mode allows single-block data transfer on a single activation source. Transfer source or transfer destination for the block area must be specified in the MRB.DTS bit. The block size can be set from 1 to 256 bytes, 1 to 256 halfwords (2 to 512 bytes), or 1 to 256 words (4 to 1024 bytes). When transfer of the specified block completes, the initial values of the block size counter CRAL and the address register (the SAR register when the MRB.DTS = 1 or the DAR register when the DTS = 0) specified in the block area are restored. The other address register is incremented or decremented continuously or remains unchanged.

The transfer count (block count) can be set from 1 to 65536. This mode enables an interrupt request to the CPU to be generated at the end of the specified-count block transfer.

Table 17.7 lists the register functions in block transfer mode, and Figure 17.7 shows the memory map for block transfer mode.

**Table 17.7 Register functions in block transfer mode**

| Register | Description | Value written back by writing transfer information |
|---|---|---|
| SAR | Transfer source address | • When MRB.DTS bit is 0<br>Increment, decrement, or fixed*1<br>• When MRB.DTS bit is 1<br>SAR register initial value. |
| DAR | Transfer destination address | • When MRB.DTS bit is 0<br>DAR register initial value<br>• When MRB.DTS bit is 1<br>Increment, decrement, or fixed*1. |
| CRAH | Holds block size | CRAH |
| CRAL | Block size counter | CRAH |
| CRB | Block transfer counter | CRB - 1 |

Note 1. Write-back is skipped in address-fixed mode.

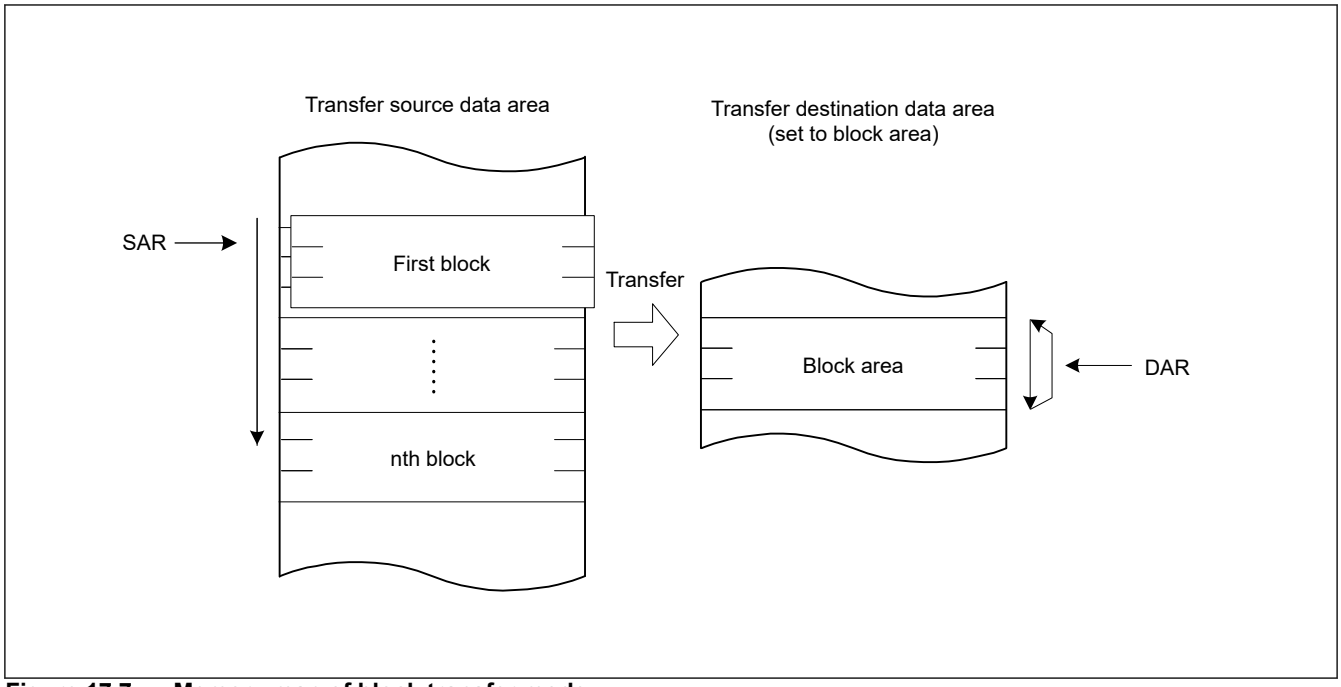



**Figure 17.7 Memory map of block transfer mode**

## 17.4.6 Chain Transfer

Setting the MRB.CHNE bit to 1 allows chain transfer to be performed continuously on a single activation source. If the MRB.CHNE is set to 1 and CHNS to 0, an interrupt request to the CPU is not generated on completion of the specified number of rounds of transfer or by setting the MRB.DISEL bit to 1. An interrupt request is sent to the CPU each time DTC data transfer is performed. Data transfer has no effect on the ICU.IELSRn.IR flag of the activation source.

The SAR, DAR, CRA, CRB, MRA, and MRB registers can be set independently of each other to define the data transfer. Figure 17.8 shows a chain transfer operation.

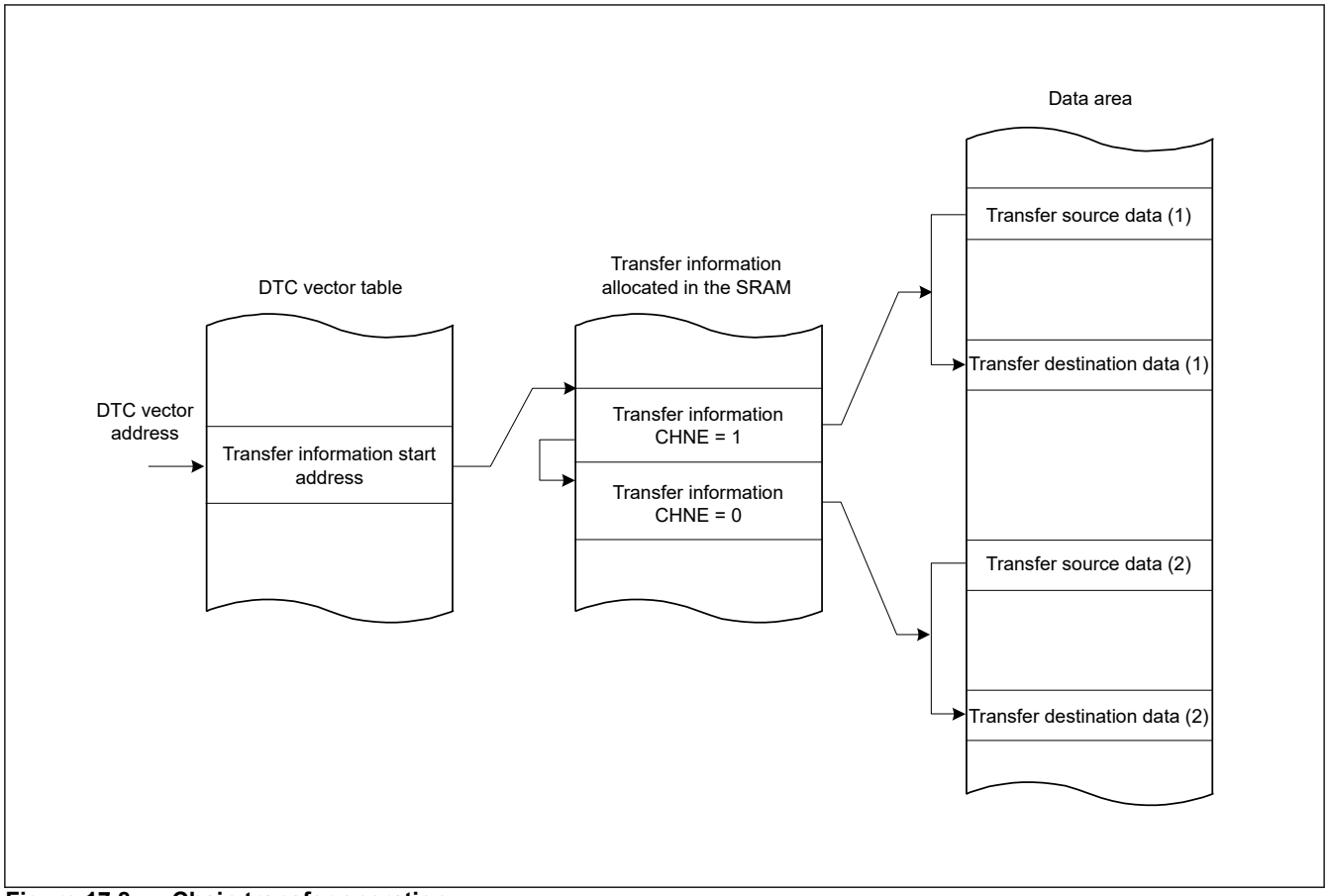



**Figure 17.8 Chain transfer operation**

Writing 1 to the MRB.CHNE and CHNS bits enables chain transfer to be performed only after completion of the specified data transfer. In repeat transfer mode, chain transfer is performed after completion of the specified data transfer. For details on chain transfer conditions, see Table 17.3.

### 17.4.7 Operation Timing

Figure 17.9 to Figure 17.12 are timing diagrams that show the minimum number of execution cycles.

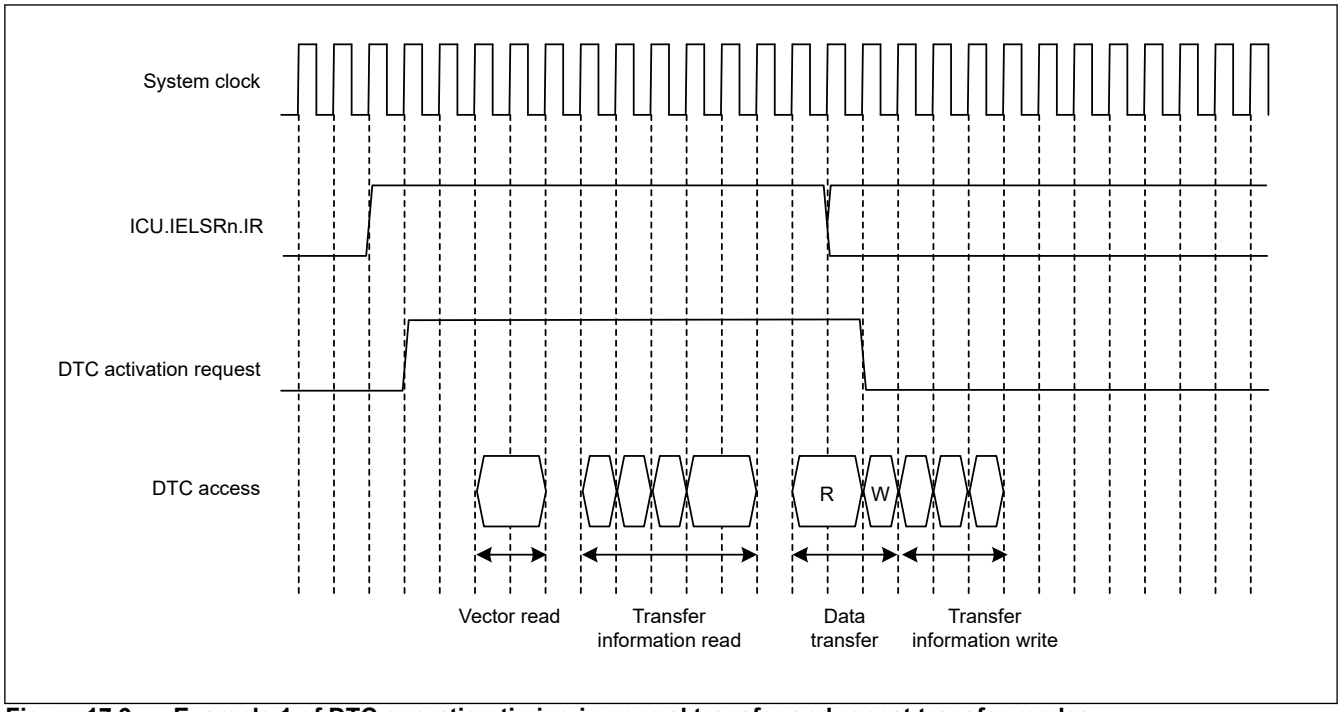



**Figure 17.9 Example 1 of DTC operation timing in normal transfer and repeat transfer modes**

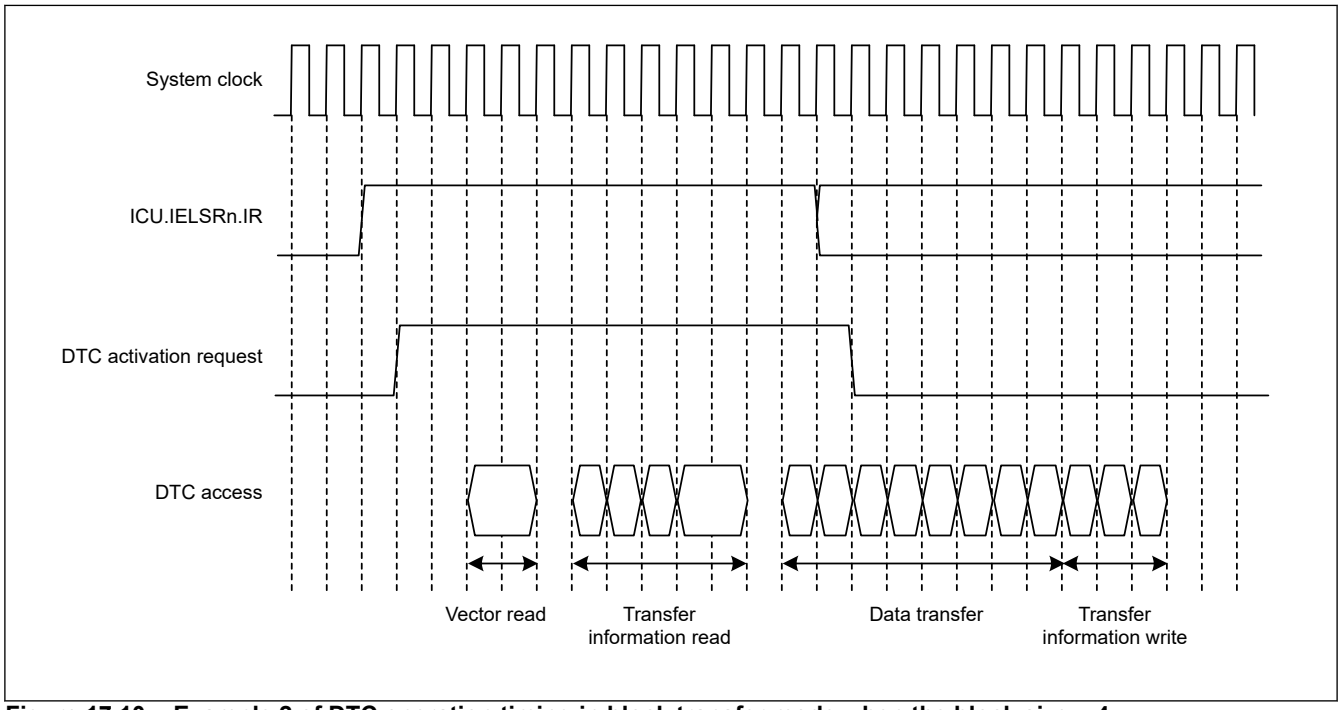



**Figure 17.10 Example 2 of DTC operation timing in block transfer mode when the block size = 4**

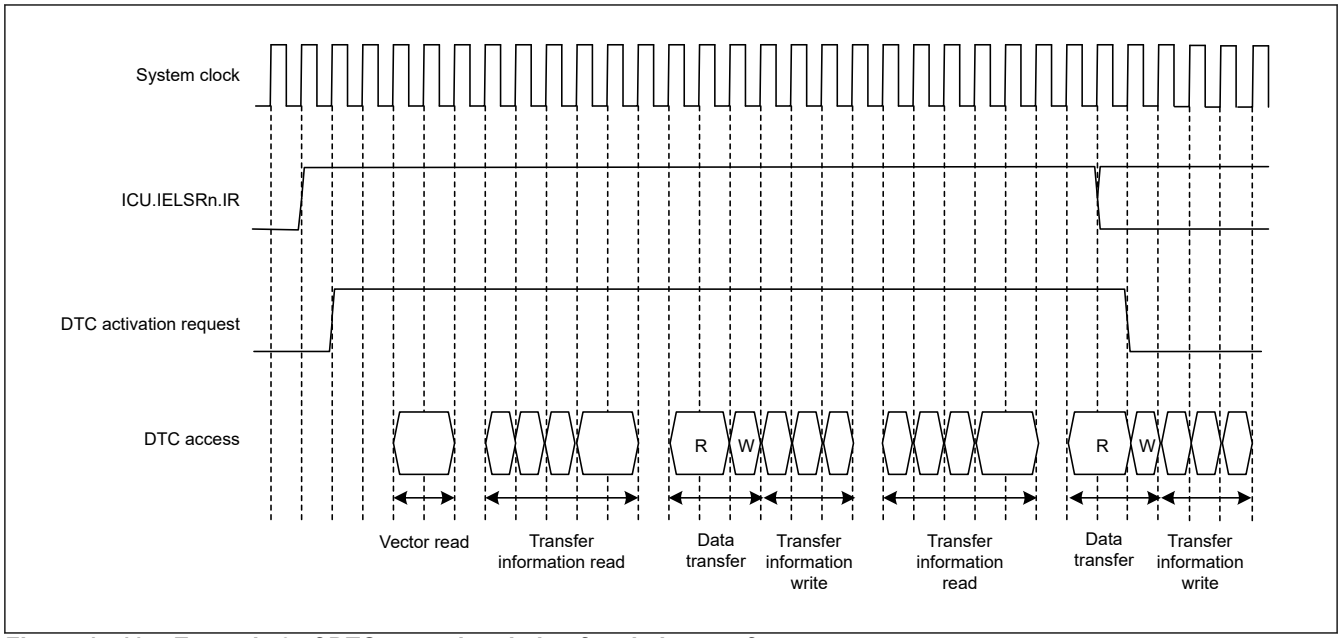
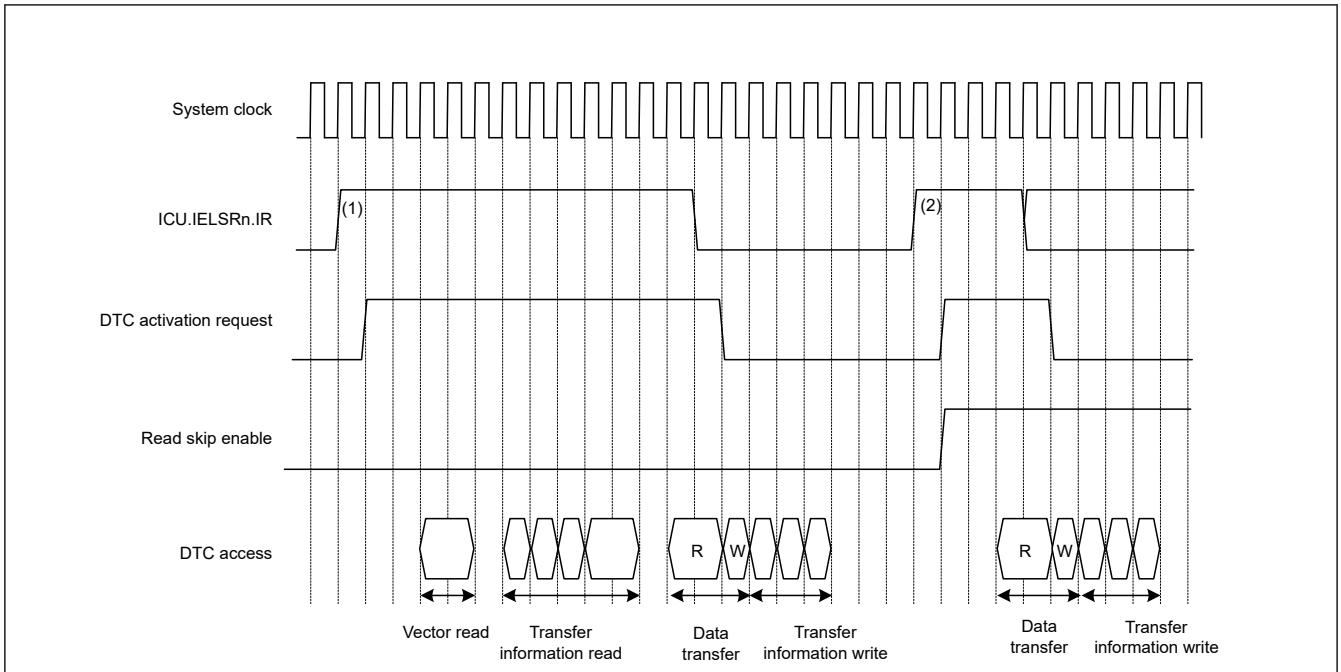



**Figure 17.11 Example 3 of DTC operation timing for chain transfer**




Note: When activation sources (vector numbers) of (1) and (2) are the same and the RRS = 1, the transfer information read for request (2) is skipped.

**Figure 17.12 Example of operation when a transfer information read is skipped with the vector, transfer information, and transfer destination data on the SRAM, and the transfer source data on the peripheral module**

## 17.4.8 Execution Cycles of DTC

Table 17.8 lists the execution cycles of single data transfer of the DTC. For the order of the execution states, see section 17.4.7. Operation Timing.

**Table 17.8 Execution cycles of DTC**

P: Block size (initial settings of CRAH and CRAL)
Cv: Cycles for access to vector transfer information storage destination
Ci: Cycles for access to transfer information storage destination address
Cr: Cycles for access to data read destination
Cw: Cycles for access to data write destination
The unit is for system clocks (ICLK) + 1 in the Vector read, Transfer information read, and Data transfer read columns and 2 in the Internal operation column.
Cv, Ci, Cr, and Cw vary depending on the corresponding access destination. For the number of cycles for respective access destinations, see section 42, SRAM, section 44, Flash Memory, and section 14, Buses.
The frequency ratio of the system clock and peripheral clock is also taken into consideration.
The DTC response time is the time from when the DTC activation source is detected until DTC transfer starts.
Table 17.8 does not include the time until DTC data transfer starts after the DTC activation source becomes active.

| Transfer mode | Vector read | | Transfer information read | | Transfer information write | | | Data transfer | | Internal operation | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Read | Write | | |
| Normal | Cv + 1 | 0*1 | 4 × Ci + 1 | 0*1 | 3 × Ci + 1*2 | 2 × Ci + 1*3 | Ci*4 | Cr + 1 | Cw + 1 | 2 | 0*1 |
| Repeat | | | | | | | | Cr + 1 | Cw + 1 | | |
| Block*5 | | | | | | | | P × Cr | P × Cw | | |

Note 1. When transfer information read is skipped.
Note 2. When neither SAR nor DAR is set to address-fixed mode.
Note 3. When SAR or DAR is set to address-fixed mode.
Note 4. When SAR and DAR are set to address-fixed mode.
Note 5. When the block size is 2 or more. If the block size is 1, the cycle number for normal transfer applies.

### 17.4.9 DTC Bus Mastership Release Timing

The DTC does not release the bus mastership during transfer information reads. Before the transfer information is read or written, the bus is arbitrated according to the priority determined by the bus master arbitrator. For bus arbitration, see section 14, Buses.

### 17.4.10 Vector Security

The security attribute of transfer access of DTC vector n and security attribute of access to the IELSRn(n = 0 to 95) register of ICU are controlled by SAIELSRn bit of ICUSARx(x = G,H or I) registers in CPSCU. For details on the CPSCU.ICUSARx registers, see section 13, Interrupt Controller Unit (ICU).

When the CPSCU.ICUSARx.SAIELSRn bit is 0, transfer of DTC vector n is secure access for both read and write. At the same time, the IELSRn register are protected from a non-secure access.

When the CPSCU.ICUSARx.SAIELSRn bit is 1, transfer of DTC vector n is non-secure access for both read and write. At the same time, the IELSRn register are non-secure attributes.

Do not write to the CPSCU.ICUSARx.SAIELSRn bit while DTC transfer is enabled or a bus master is writing to the DTC registers of same channel.

section 17.3.1. Allocating Transfer Information and DTC Vector Table shows security attribute about each DTC vectors.

### 17.4.11 Master TrustZone Filter in DTC

DTC has the Master TrustZone Filter. The Master TrustZone Filter in DTC can detect the security areas of Flash area(code Flash and data Flash) and SRAM area(ECC / Parity RAM) defined by IDAU. When no-secure accesses those addresses, it detects the security violation. Access of violation address is not performed. Detected the error is handled as the Master TrustZone Filter error.

## 17.5 DTC Setting Procedure

Before using the DTC, set the DTC Vector Base Register (DTCVBR). Set the ICU.IELSRn.IELS[8:0] bits to 0 to disable the interrupt in the NVIC and follow the procedure in Table 17.9 to set the DTC.

**Table 17.9 DTC setting procedure**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set the DTCCR*1.RRS bit to 0 | Set the DTCCR*1.RRS bit to 0 to reset the transfer information read skip flag. After that, the transfer information read is not skipped while the DTC is activated. Be sure to specify this setting when the transfer information is updated. |
| 2 | Set transfer information (MRA, MRB, SAR, DAR, CRA, and CRB) | Allocate transfer information (MRA, MRB, SAR, DAR, CRA, and CRB) in the data area. To set transfer information, see section 17.2. Register Descriptions. To allocate transfer information, see section 17.3.1. Allocating Transfer Information and DTC Vector Table. |
| 3 | Set transfer information start addresses in the DTC vector table | Set the transfer information start addresses in the DTC vector table. To set the DTC vector table, see section 17.3.1. Allocating Transfer Information and DTC Vector Table. |
| 4 | Set the DTCCR*1.RRS bit to 1 | Set the DTCCR*1.RRS bit to 1 to enable skipping of the second and subsequent transfer information read cycles for continuous DTC activation from the same interrupt source. The RRS bit can be set to 1, but if this is set during DTC transfer, it becomes valid from the next transfer. |
| 5 | Set the ICU.IELSRn.DTCE bit to 1.<br>Set the ICU.IELSRn.IELS[8:0] as interrupt source. The interrupt should be enabled in the NVIC. | Set the ICU.IELSRn.DTCE bit to 1. Set ICU.IELSRn.IELS[8:0] as interrupt sources that trigger DTC. The interrupt must be enabled in the NVIC. See section 13.3.2. Event Number in section 13, Interrupt Controller Unit (ICU). |
| 6 | Set the enable bit for an activation source interrupt | Set the enable bit for the activation source interrupts to 1. When a source interrupt is generated, the DTC is activated. To set the interrupt source enable bit, see the settings for the modules that are to be the activation sources. |
| 7 | Set the DTCST.DTCST bit to 1 | Set the DTC Module Start bit (DTCST.DTCST) to 1. |

Note: The DTCST.DTCST bit can be set even if the setting for each activation source is not completed.
Note: When used in non-secure state, DTCSAR.DTCSTSA = 1 or DTCST.DTCST = 1 must be set.
Note 1. When used in secure state, access DTCCR_SEC instead of DTCCR.

## 17.6 Examples of DTC Usage

### 17.6.1 Normal Transfer

This section provides an example of DTC usage and its application when receiving 128 bytes of data from an SCI.

#### (1) Transfer information settings

In the MRA register, select a fixed source address (MRA.SM[1:0] = 00b), normal transfer mode (MRA.MD[1:0] = 00b), and byte-sized transfer (MRA.SZ[1:0] = 00b). In the MRB register, specify incrementation of the destination address (MRB.DM[1:0] = 10b) and single data transfer by a single interrupt (MRB.CHNE = 0 and MRB.DISEL = 0). The MRB.DTS bit can be set to any value. Set the RDR register address of the SCI in the SAR register, the start address of the SRAM area for data storage in the DAR register, and 128 (0x0080) in the CRA register. The CRB register can be set to any value.

#### (2) DTC vector table settings

The start address of the transfer information for the RXI interrupt is set in the vector table for the DTC.

#### (3) ICU settings and DTC module activation

Set the ICU.IELSRn.DTCE bit to 1 and set ICU.IELSRn.IELS[8:0] as the SCI interrupt. The interrupt must be enabled in the NVIC. Set the DTCST.DTCST bit to 1.

#### (4) SCI settings

Enable the SCIn_RXI (n = 0 to 4, 9) interrupt by setting the SCR.RIE bit in the SCI to 1. If a reception error occurs during the SCI receive operation, reception stops. To manage this, use settings that allow the CPU to accept receive error interrupts.

#### (5) DTC transfer

Each time a reception of 1 byte by the SCI is complete, an SCIn_RXI interrupt is generated to activate the DTC. The DTC transfers the received byte from the RDR of the SCI to the SRAM, after which the DAR register is incremented and the CRA register is decremented.

#### (6) Interrupt handling

After 128 rounds of data transfer are complete and the value in the CRA register becomes 0, an SCIn_RXI interrupt request is generated for the CPU. Complete the process in the handling routine for this interrupt.

### 17.6.2 Chain transfer

This section provides an example of chain transfer by the DTC and describes its use in the output of pulses by the General PWM Timer (GPT). You can use chain transfer to transfer PWM timer compare data and change the period of the PWM timer for the GPT.

For the first of the chain transfers, normal transfer mode is specified for transfer to the GPTm.GTCCRC register (m = 320 to 323, 164 to 167). For the second transfer, normal transfer mode is specified for transfer to the GPTm.GTCCRE register (m = 320 to 323, 164 to 167). For the third transfer of the chained transfer, normal transfer mode for transfer to the GPTm.GTPBR register (m = 320 to 323, 164 to 167) is specified. This is because clearing of the activation source and generation of an interrupt on completion of the specified number of transfers are restricted to the third of the chain transfers, that is, transfer while MRB.CHNE = 0.

The following example shows how to use the counter overflow interrupt with the GPT320.GTPR register as an activating source for the DTC.

#### (1) First transfer information setting

Set up transfer to the GPT320.GTCCRC register.

1. In the MRA register, select incrementation of the source address (MRA.SM[1:0] = 10b).
2. Set the transfer to normal transfer mode (MRA.MD[1:0] = 00b) and word-sized transfer (MRA.SZ[1:0] = 10b).
3. In the MRB register, select the destination address as fixed (MRB.DM[1:0] = 00b) and set up chain transfer (MRB.CHNE = 1 and MRB.CHNS = 0).
4. Set the SAR register to the first address of the data table.
5. Set the DAR register to the address of the GPT320.GTCCRC register.
6. Set the CRAH and CRAL registers to the size of the data table. The CRB register can be set to any value.

#### (2) Second transfer information setting

Set up for transfer to the GPT320.GTCCRE register.

1. In the MRA register, select incrementation of the source address (MRA.SM[1:0] = 10b).
2. Set the transfer to normal transfer mode (MRA.MD[1:0] = 00b) and word-sized transfer (MRA.SZ[1:0] = 10b).
3. In the MRB register, select the destination address as fixed (MRB.DM[1:0] = 00b) and set up chain transfer (MRB.CHNE = 1, MRB.CHNS = 0).
4. Set the SAR register to the first address of the data table.
5. Set the DAR register to the address of the GPT320.GTCCRE register.
6. Set the CRAH and CRAL registers to the size of the data table. The CRB register can be set to any value.

#### (3) Third transfer information set

Set up transfer to the GPT320.GTPBR register.

1. In the MRA register, select incrementation of the source address (MRA.SM[1:0] = 10b).
2. Set the transfer to normal transfer mode (MRA.MD[1:0] = 00b) and word-sized transfer (MRA.SZ[1:0] = 10b).
3. In the MRB register, select the destination address as fixed (MRB.DM[1:0] = 00b) and set up single data transfer per interrupt (MRB.CHNE = 0, MRB.DISEL = 0). The MRB.DTS bit can be set to any value.
4. Set the SAR register to the first address of the data table.
5. Set the DAR register to the address of the GPT320.GTPBR register.
6. Set the CRA register to the size of the data table. The CRB register can be set to any value.

#### (4) Transfer information assignment

Place the transfer information for use in the transfer to the GPT320.GTPBR immediately after the transfer control information for use in the GPT320.GTCCRC and GPT320.GTCCRE registers.

#### (5) DTC vector table

In the DTC vector table, set the address where the transfer control information for use in transfer to the GPT320.GTCCRC and GPT320.GTCCRE registers starts.

#### (6) ICU setting and DTC module activation

1. Set the ICU.IELSRn.DTCE bit associated with the GPT320 counter overflow interrupt.
2. Set the ICU.IELSRn.IELS[8:0] bits and specify the GPT320 counter overflow.
3. Set the DTCST.DTCST bit to 1.

#### (7) GPT settings

1. Set the GPT320.GTIOR register so that the GTCCRA and GTCCRB registers operate as output compare registers.
2. Set the default PWM timer compare values in the GPT320.GTCCRA and GPT320.GTCCRB registers and the next PWM timer compare values in the GPT320.GTCCRC and GPT320.GTCCRE registers.
3. Set the default PWM timer period values in the GPT320.GTPR register and the next PWM timer period values in the GPT320.GTPBR register.
4. Set 1 to the output bit in PmnPFS.PDR, and set 00011b to the Peripheral Select bits in PmnPFS.PSEL[4:0].

#### (8) GPT activation

Set the GPT320.GTSTR.CSTRT bits to 1 to start the GPT320.GTCNT counter.

#### (9) DTC transfer

Each time a GPT320 counter overflow is generated with the GPT320.GTPR register, the next PWM timer compare values are transferred to the GPT320.GTCCRC and GPT320.GTCCRE registers. The setting for the next PWM timer period is transferred to the GPT320.GTPBR register.

#### (10) Interrupt handling

After the specified rounds of data transfer are complete, for example when the value in the CRA register for GPT transfer becomes 0, a GPT320 counter overflow interrupt request is issued for the CPU. Complete the process for this interrupt in the handling routine.

### 17.6.3 Chain Transfer when Counter = 0

The second data transfer is performed only when the transfer counter is set to 0 in the first data transfer, and the first data transfer information is repeatedly changed in the second transfer. Chain transfer enables transfers to be repeated 256 times or more.

The following procedure shows an example of configuring a 1-KB input buffer, where the input buffer is set so that its lower address starts with 0x00. Figure 17.13 shows a chain transfer when the counter = 0.

1. Set the normal transfer mode to input data for the first data transfer. Set the following:
   (a) Transfer source address = fixed.
   (b) CRA register = 0x0200 (512) times.
   (c) MRB.CHNE bit = 1 (chain transfer is enabled).
   (d) MRB.CHNS bit = 1 (chain transfer is performed only when the transfer counter is 0).
   (e) MRB.DISEL bit = 0 (an interrupt request to the CPU is generated when the specified data transfer completes).
2. Prepare the upper 8-bit address of the start address at every 512 times of the transfer destination address for the first data transfer in different area such as the flash. For example, when setting the input buffer to 0x8000 to 0x83FF, prepare 0x82 and 0x80.

3. For the second data transfer:
   (a) Set the repeat transfer mode (with transfer source and destination address = fixed.) to reset the transfer counter of the first data transfer.
   (b) Specify the CRA register in the first transfer information area for the transfer destination.
   (c) Set the MRB.CHNE bit = 1 (chain transfer is enabled).
   (d) Set the MRB.CHNS bit = 0 (select continuous chain transfer).
   (e) Set the MRB.DISEL bit = 0 (an interrupt request to the CPU is generated when the specified data transfer completes).
   (f) CRA register = 0x0101 (The transfer count is 1).
4. For the third data transfer:
   (a) Set the repeat transfer mode (with the source as the repeat area) to reset the transfer destination address of the first data transfer.
   (b) Specify the upper 8 bits of the DAR register in the first transfer information area for the transfer destination.
   (c) Set the MRB.CHNE bit = 0 (chain transfer is disabled).
   (d) Set the MRB.DISEL bit = 0 (an interrupt request to the CPU is generated when the specified data transfer completes).
   (e) When setting the input buffer to 0x8000 to 0x83FF, also set the transfer counter to 2.
5. The first data transfer is performed by an interrupt 512 times. When the transfer counter of the first data transfer becomes 0, the second data transfer starts. Set the transfer counter of the first data transfer to 0x0200. The lower 8 bits of the transfer destination address and the transfer counter of the first data transfer becomes 0x0200.
6. The second data transfer is performed by an interrupt 1 times. When the transfer counter of the first data transfer becomes 0, the third data transfer starts. Set the upper 8 bits of the transfer destination address of the first data transfer to 0x82. The lower 8 bits of the transfer destination address becomes 0x00 and the transfer counter of the first data transfer becomes 0x0200.
7. In succession, the first data transfer is performed by an interrupt 512 times as specified for the first data transfer. When the transfer counter of the first data transfer becomes 0, the second data transfer starts. Set the transfer counter of the first data transfer to 0x0200. The lower 8 bits of the transfer destination address and the transfer counter of the first data transfer becomes 0x0200.
8. The second data transfer is performed by an interrupt 1 times. When the transfer counter of the first data transfer becomes 0, the third data transfer starts. Set the upper 8 bits of the transfer destination address of the first data transfer to 0x80. The lower 8 bits of the transfer destination address becomes 0x00 and the transfer counter of the first data transfer becomes 0x0200.
9. Steps 5 to 8 are repeated indefinitely. Because the second data transfer is in repeat transfer mode, no interrupt request to the CPU is generated.

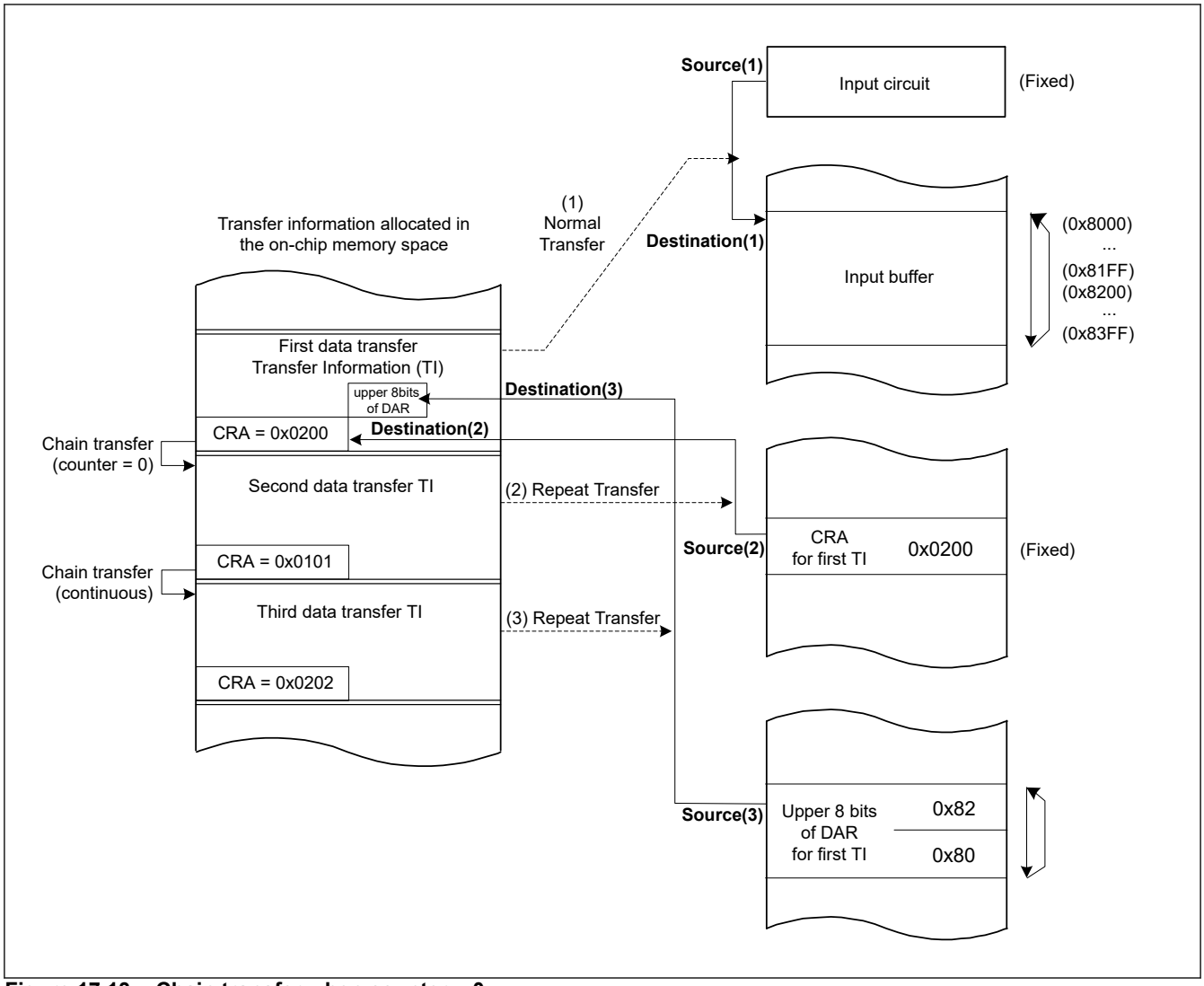



**Figure 17.13 Chain transfer when counter = 0**

## 17.7 Processing on DTC Transfer Error

If the access error occurs during DTC transfer, the DTC immediately stops access during transfer. To stop only the vector number that caused the error, inform the vector number that caused the error to the ICU and clear the corresponding ICU setting. After that, if there is a request other than the vector number which caused the error, it will be re-arbitration as it is. The condition under which the transfer error occurs is indicated when TrustZone Filter in DTC detects a violation.

The error response is informed to ICU when the transfer error occurs. ICU clears the ICU.IELSRn of the corresponding vector number which caused the transfer error. Furthermore, it generates an error response detection interrupt to notify that an error has occurred by DMAC/DTC transfer. (section 17.8.2. Interrupt Request of Transfer Error). Write back to SRAM is not performed.

When the Master TrustZone Filter error occurs, the Slave TrustZone error occurs or the Master MPU error occurs, it is possible to confirm the error information of DTC by selecting NMI. The DTC error vector register is cleared by selecting reset. Under the conditions where NMI is generated due to transfer error in DTC, two interrupts(NMI and DMA_TRANSERR) are generated. In this case, NMI always responds first.

The error response detection interrupt request (DMA_TRANSERR) occurs when the Slave Bus error or the Illegal Access error occurs. Furthermore, it occurs after NMI when the error response detection interrupt request (DMA_TRANSERR) is not cleared in NMI handler.

section 17.7.1. Processing on NMI handler describes how to confirm the error information of the DTC in the NMI handler. section 17.7.2. Processing on Error response detection interrupt request (DMA_TRANSERR) handler describes how to confirm the error information of the DTC in the DMA_TRANSERR handler.

Interrupts and the error information generated due to transfer errors are shown in section 17.8.2. Interrupt Request of Transfer Error.

### 17.7.1 Processing on NMI handler

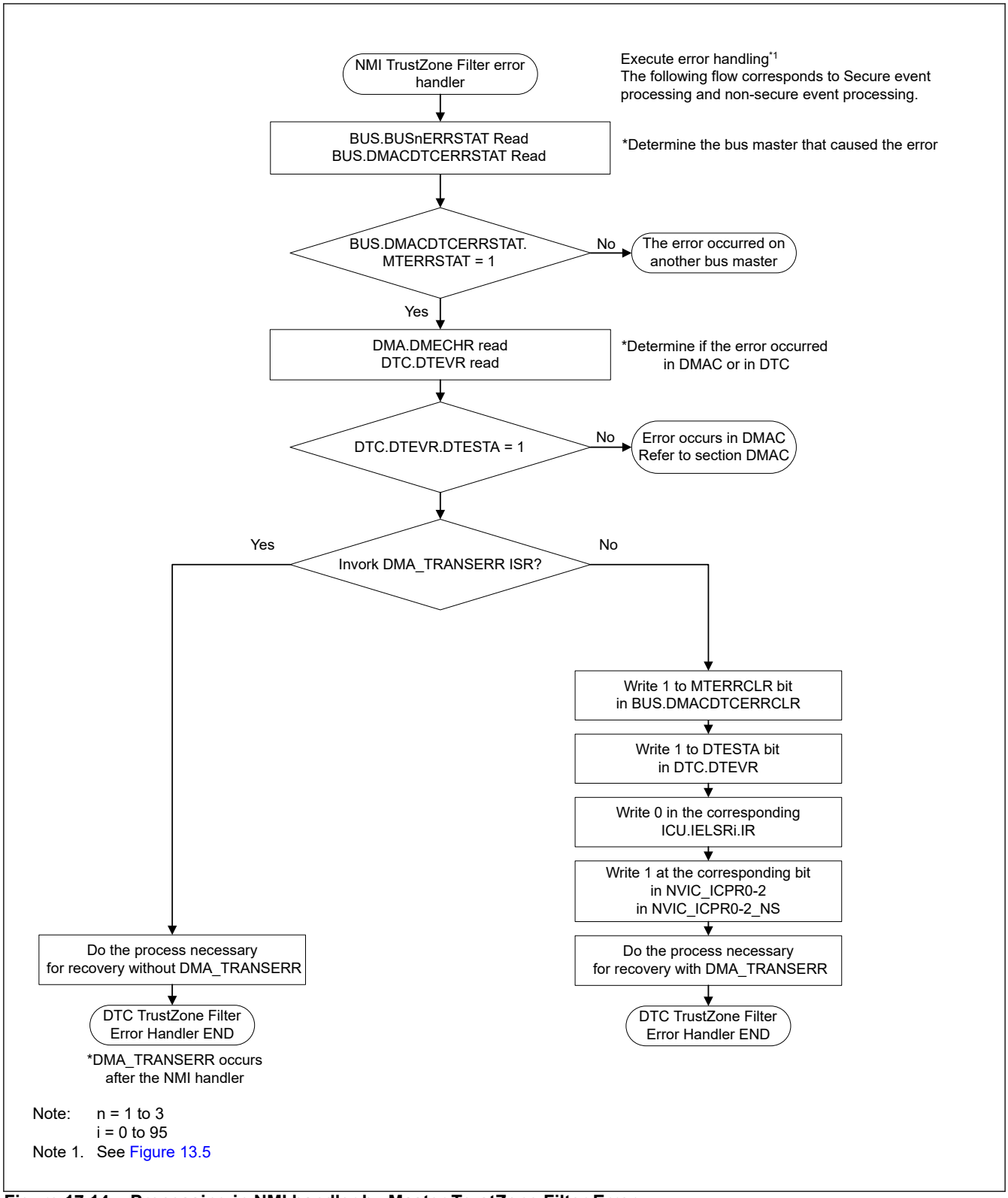
The cause of NMI due to the DMA transfer error is the Master TrustZone Filter error, the Slave TrustZone Filter error or the Master MPU error. When NMI occurs due to the DTC transfer error, the error response detection interrupt request (DMA_TRANSERR) will occur after the end of NMI handler. It is possible to confirm the cause of the error and the DTC vector number in which the error occurred. When NMI occurs, perform the necessary processing according to the flow described in the ICU chapter.

Figure 17.14 shows the flow for confirm the vector number that caused the Master TrustZone Filter Error in DTC

Figure 17.15 shows the flow for confirm the vector number that caused the Slave TrustZone Filter Error in DTC

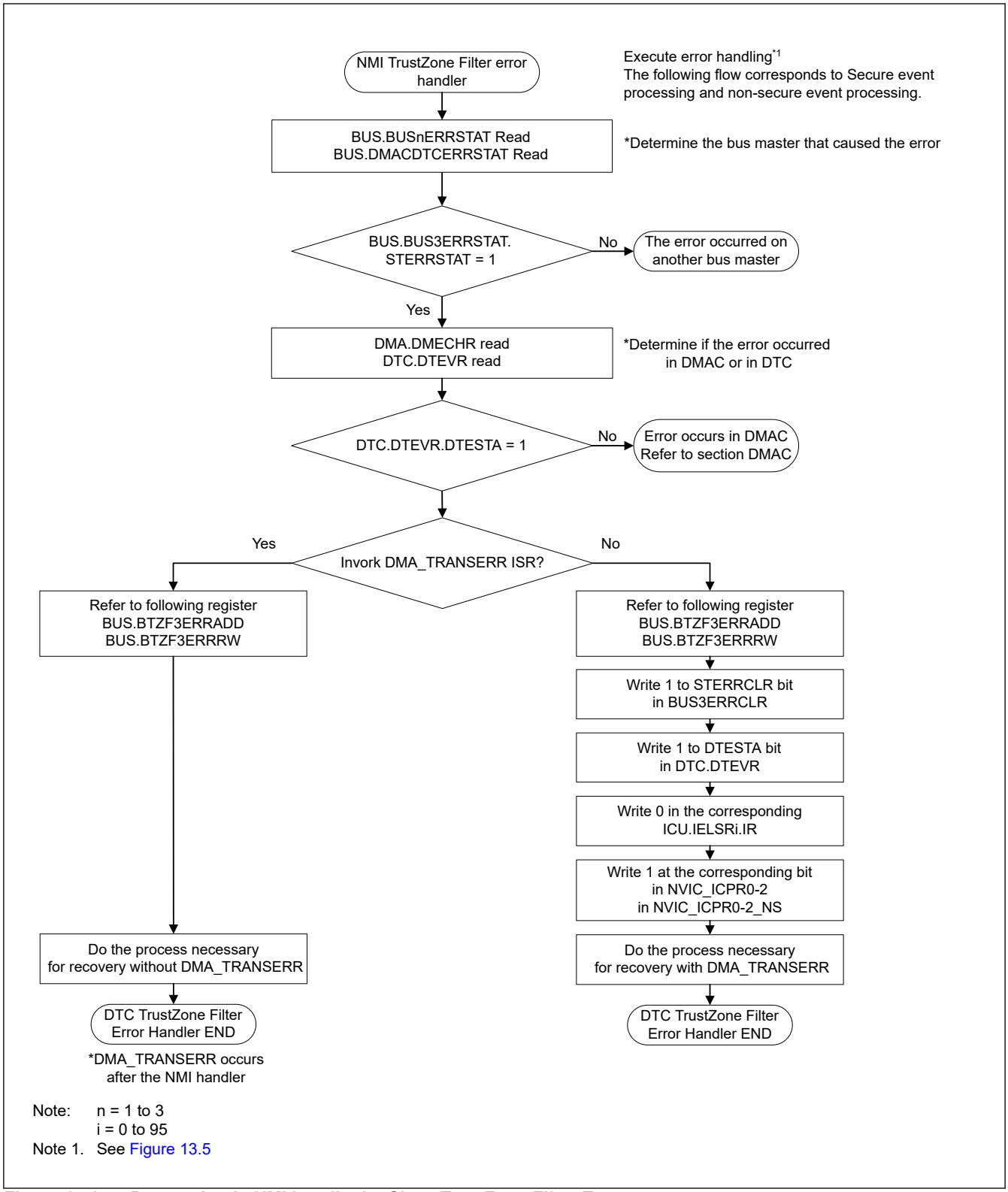
Figure 17.16 shows the flow for confirm the vector number and Security Attribute that caused the Master MPU error in DTC

If completing all processing in NMI handler, it is possible to clear the error response detection interrupt request (DMA_TRANSERR) that occurs subsequently.




Note: n = 1 to 3
i = 0 to 95
Note 1. See Figure 13.5

**Figure 17.14 Processing in NMI handler by Master TrustZone Filter Error**




Note: n = 1 to 3
i = 0 to 95
Note 1. See Figure 13.5

**Figure 17.15 Processing in NMI handler by Slave TrustZone Filter Error**

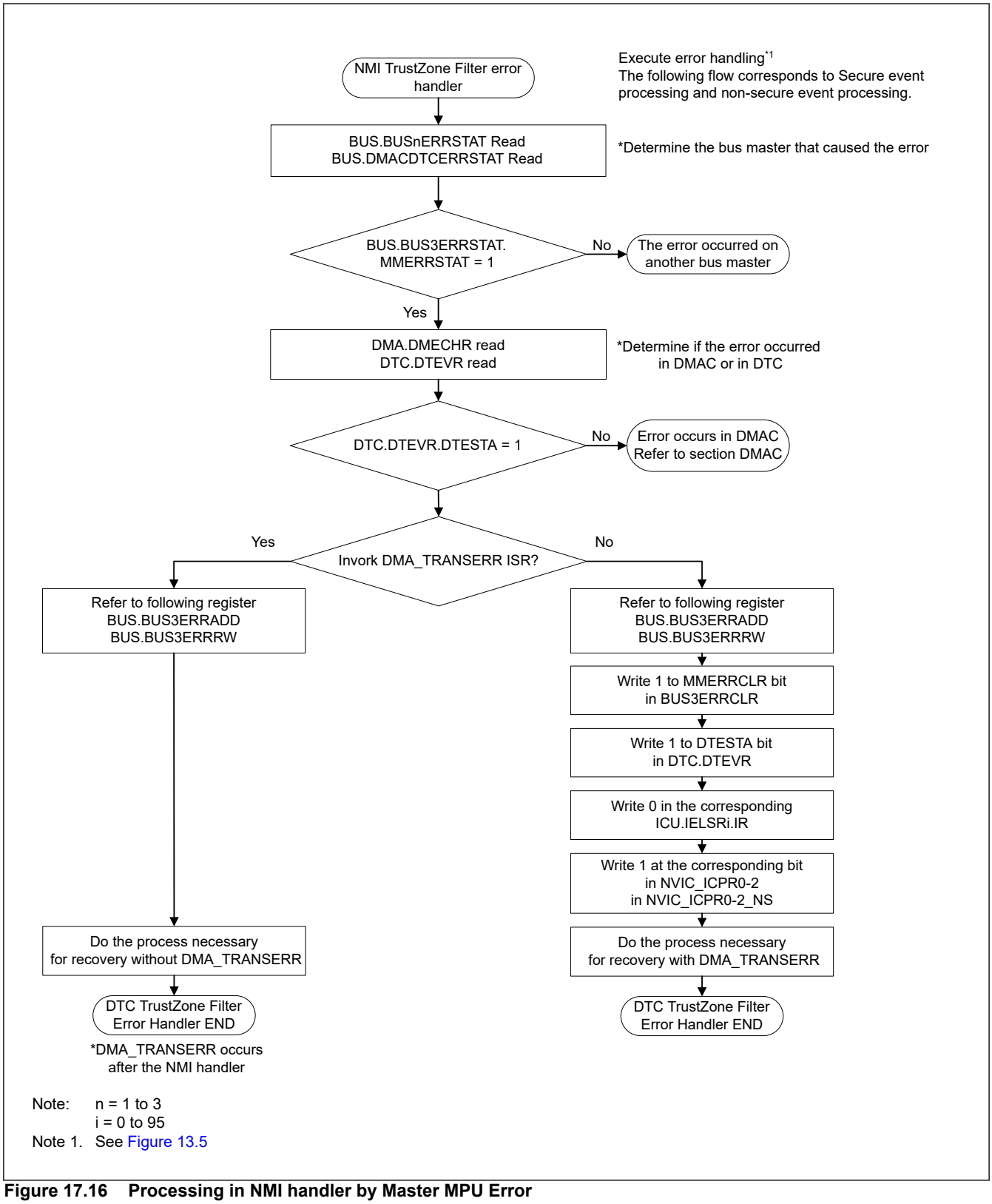



**Figure 17.16 Processing in NMI handler by Master MPU Error**

## 17.7.2 Processing on Error response detection interrupt request (DMA_TRANSERR) handler

The cause of error response detection interrupt request (DMA_TRANSERR) due to DMA transfer error is the Slave Bus Error or Illegal Access Error. Also, it occurs after the NMI handler error response detection interrupt request (DMA_TRANSERR) is not cleared by the NMI handler.

It is possible to confirm the cause of the error and the vector number of DTC in which the error occurred.

Error cause confirmation procedure is shown Figure 17.17.

Figure 17.18 shows the flow for confirm the vector number that caused the Master TrustZone Filter Error in DTC

Figure 17.19 shows the flow for confirm the vector number that caused the Slave TrustZone Filter Error in DTC

Figure 17.20 shows the flow for confirm the vector number and Security Attribute that caused the Master MPU Error in DTC

Figure 17.21 shows the flow for confirm the vector number and Security Attribute that caused the Slave Bus Error in DTC

Figure 17.22 shows the flow for confirm the vector number and Security Attribute that caused the Illegal Access Error in DTC

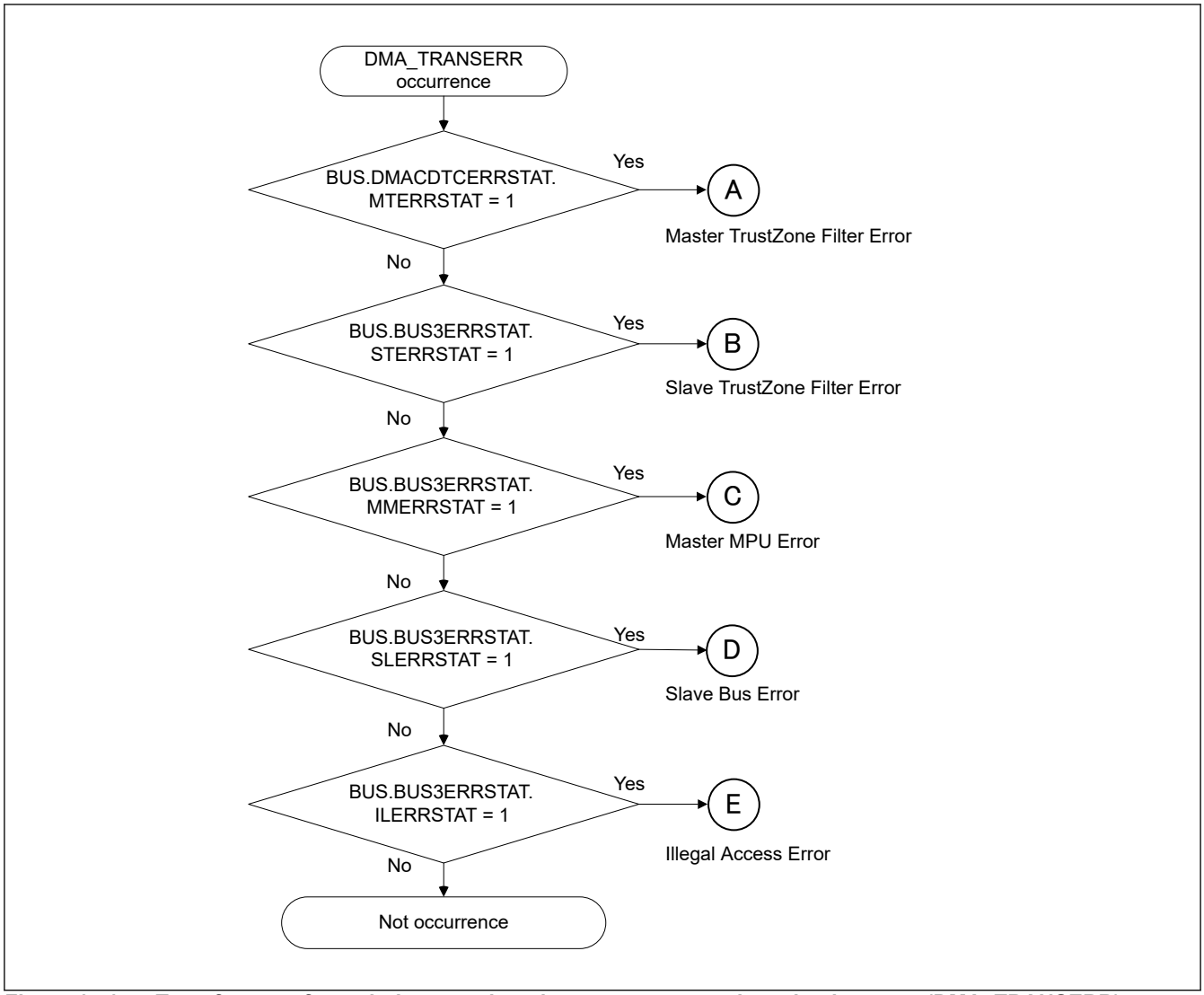



**Figure 17.17 Transfer error factor judgment when the error response detection interrupt (DMA_TRANSERR) occurs**

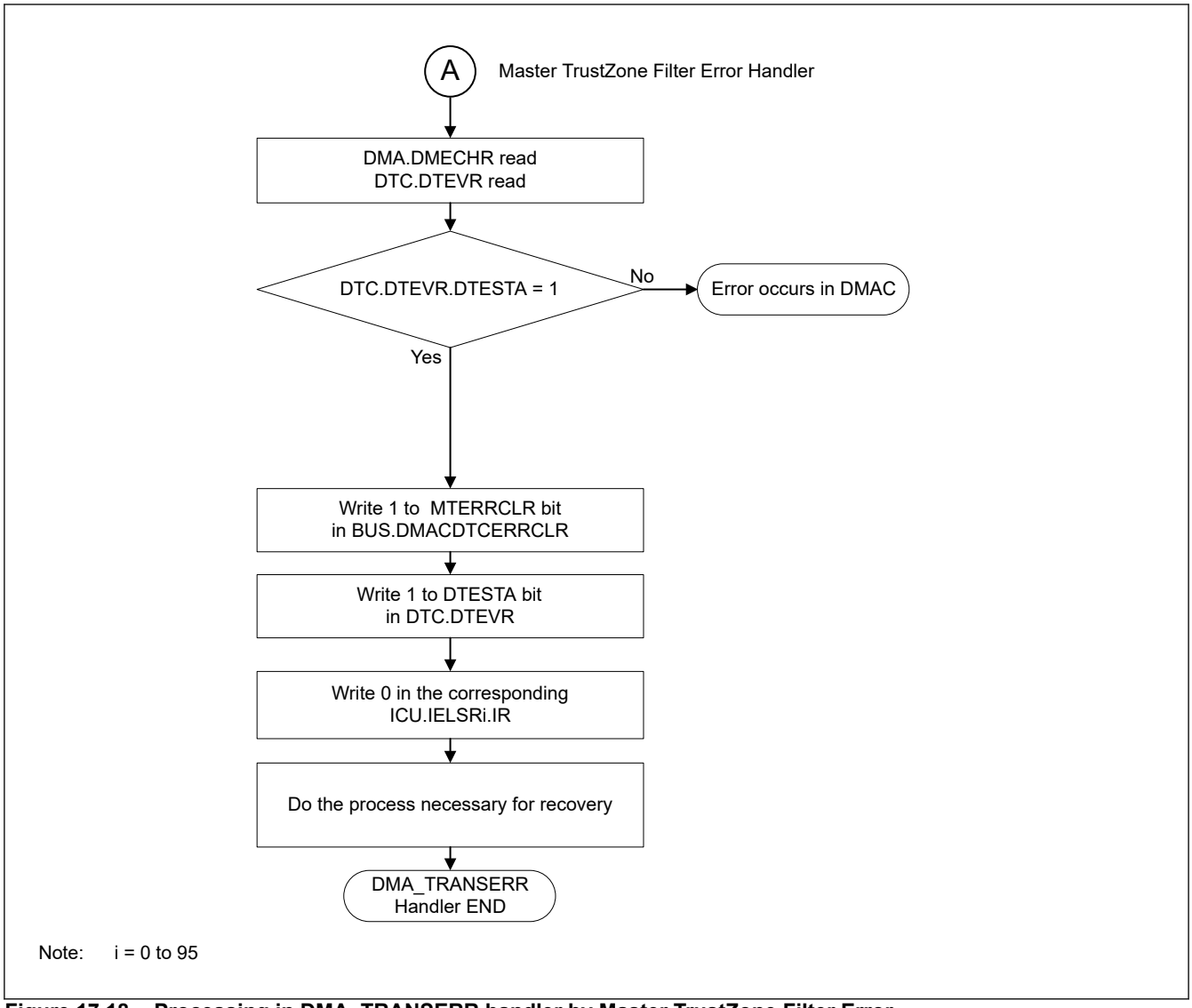



**Figure 17.18 Processing in DMA_TRANSERR handler by Master TrustZone Filter Error**

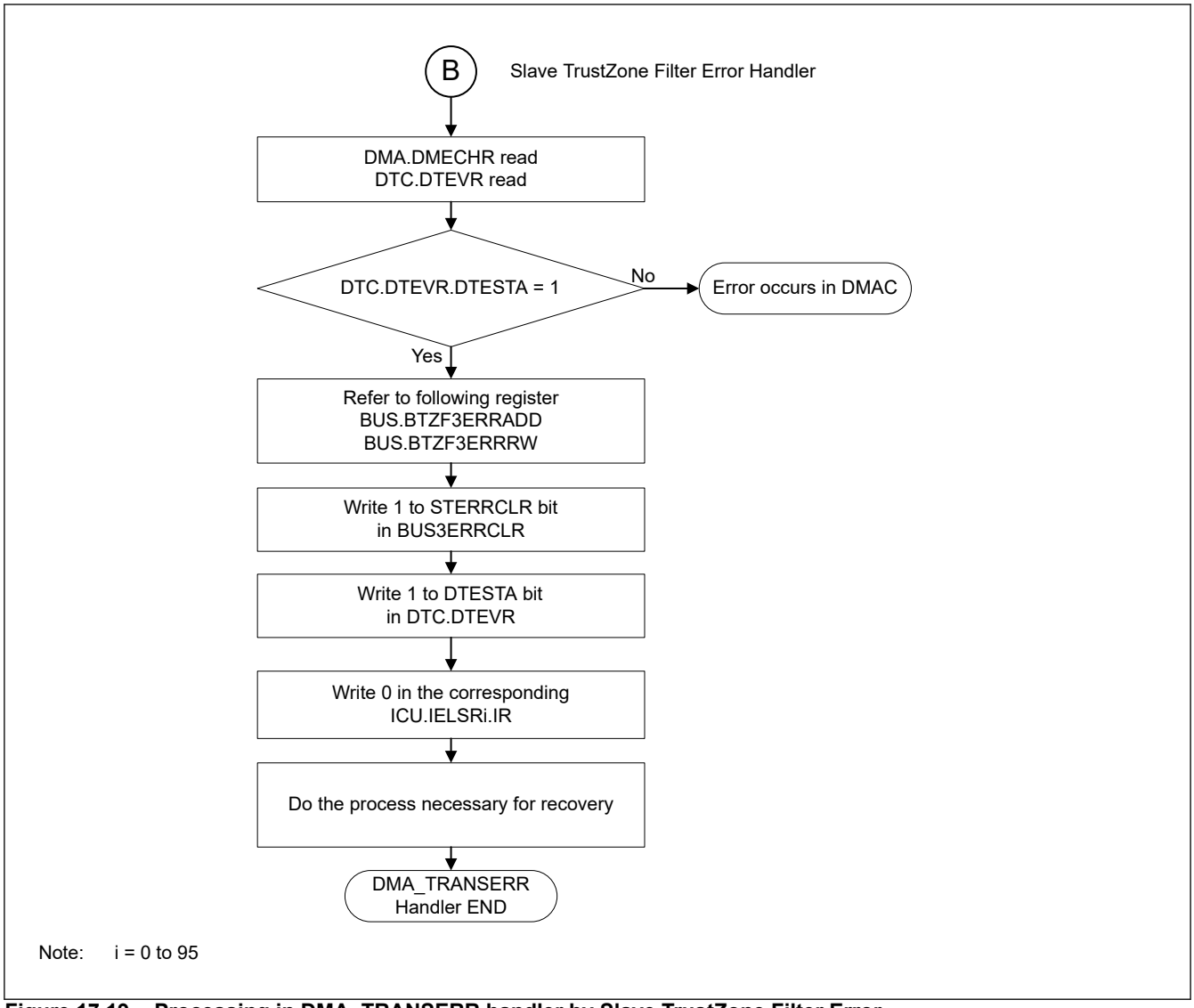
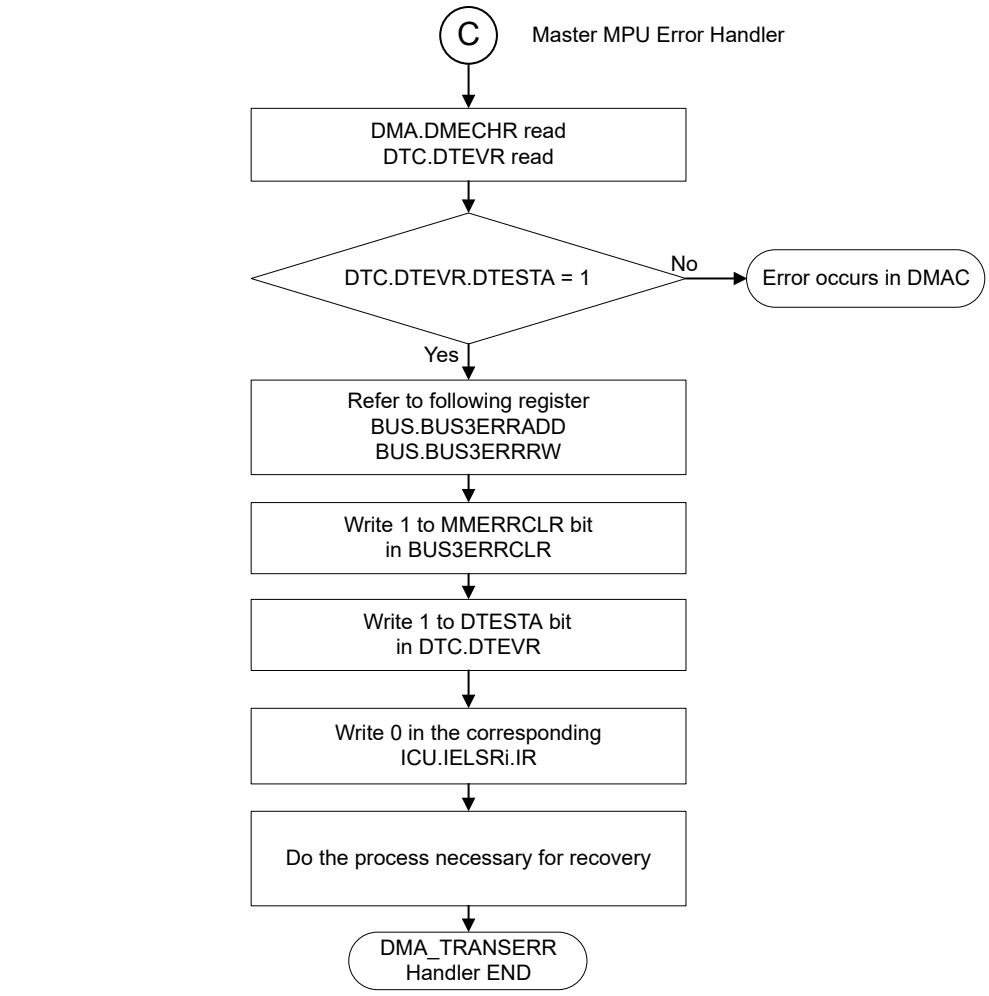
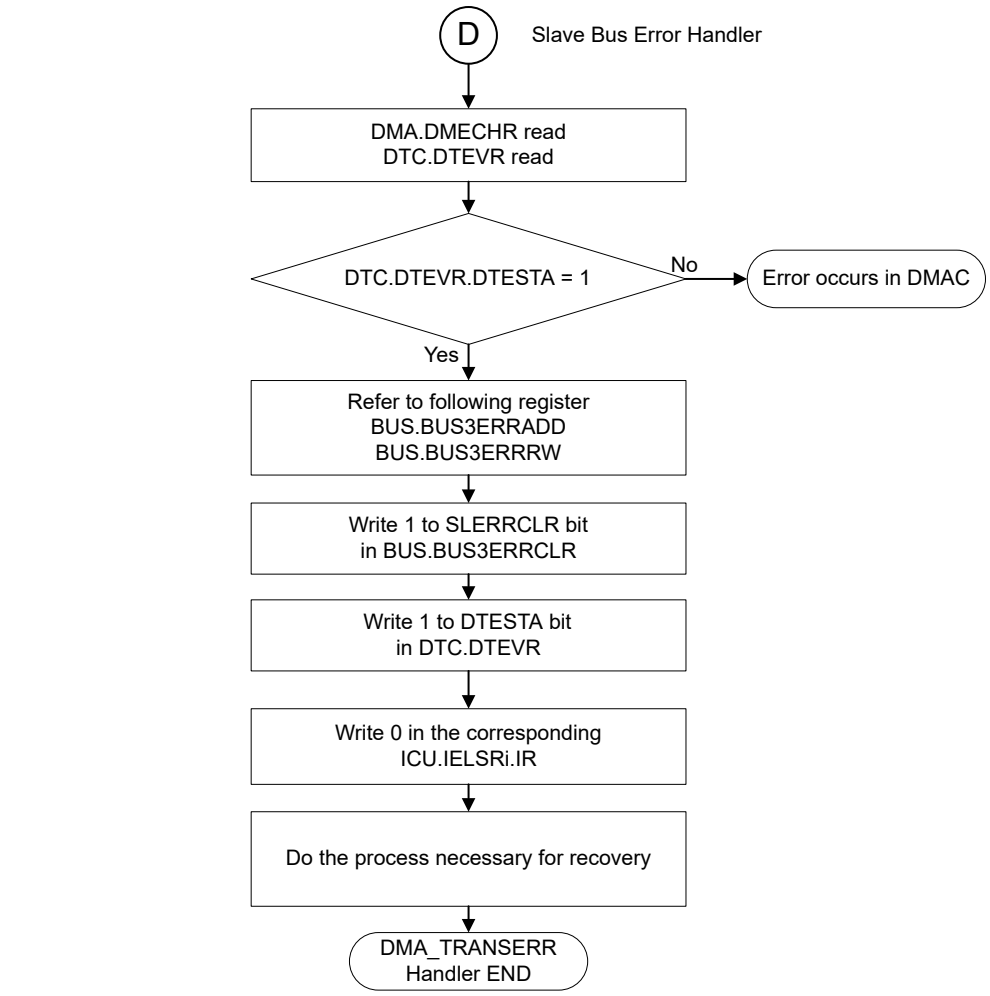



**Figure 17.19 Processing in DMA_TRANSERR handler by Slave TrustZone Filter Error**




Note: i = 0 to 95

**Figure 17.20 Processing in DMA_TRANSERR handler by Master MPU Error**




Note: i = 0 to 95

**Figure 17.21 Processing in DMA_TRANSERR handler by Slave Bus Error**

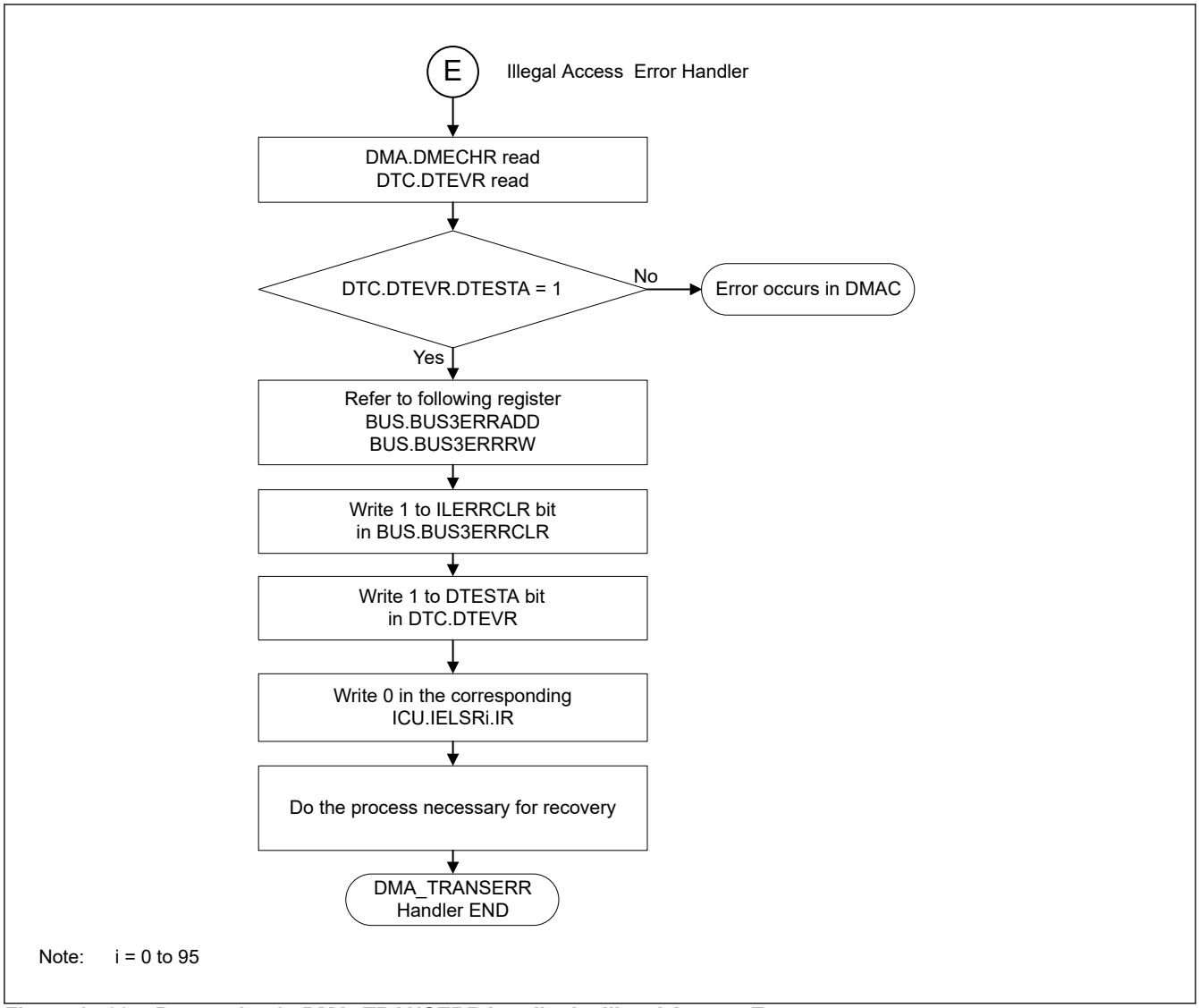



**Figure 17.22 Processing in DMA_TRANSERR handler by Illegal Access Error**

## 17.8 Interrupt

### 17.8.1 Interrupt Request of Transfer End

When the DTC completes data transfer of the specified count or when data transfer with MRB.DISEL set to 1 is complete, a DTC activation source generates an interrupt to the CPU. Two types of interrupt are available: interrupts triggered by a DTC activation (per channel) and an interrupt triggered by the event signal DTC_COMPLETE (common to all channels). Interrupts to the CPU are controlled according to the settings in the NVIC and the ICU.IELSRn.IELS[8:0] bits. See section 13, Interrupt Controller Unit (ICU). The DTC prioritizes activation sources by granting the smaller interrupt vector numbers higher priority. The priority of interrupts to the CPU is determined by the NVIC priority.

### 17.8.2 Interrupt Request of Transfer Error

The error response detection interrupt request (DMA_TRANSERR) is generated from the DMAC/DTC when the transfer error is detected during DTC transfer. The types of interrupts that occur when the DTC transfer error occurs are listed in the Table 17.10. The Table 17.10 also shows error information stored when a transfer error occurs.

**Table 17.10 Interrupt and error information due to DMAC transfer error cause**

| Transfer error factor | NMI/RESET*1 Request | Interrupt Request | Bus Error Status | Error Address Error R/W | Error Channel Information |
|---|---|---|---|---|---|
| Master TrustZone Filter (in DMAC/DTC) | ICU.NMISR.TZFST*1 | DMA_TRANSERR | BUS.DMACDTCERRSTAT.MTERRSTAT*1 | — | DTC.DTEVR |
| Slave TrustZone Filter | ICU.NMISR.TZFST*1 | DMA_TRANSERR | BUS.BUS3ERRSTAT.STERRSTAT*1 | BUS.BTZF3ERRADD BUS.BTZF3ERRRW | DTC.DTEVR |
| Master MPU | ICU.NMISR. BUSMST | DMA_TRANSERR | BUS.BUS3ERRSTAT.MMERRSTAT | BUS.BUS3ERRADD BUS.BUS3ERRRW | DTC.DTEVR |
| Slave Bus Error | —*2 | DMA_TRANSERR | BUS.BUS3ERRSTAT.SLERRSTAT*2 | BUS.BUS3ERRADD BUS.BUS3ERRRW | DTC.DTEVR |
| Illegal Access Error | —*2 | DMA_TRANSERR | BUS.BUS3ERRSTAT.ILERRSTAT*2 | BUS.BUS3ERRADD BUS.BUS3ERRRW | DTC.DTEVR |

Note 1. Interrupt generated, when NMI request selected as the operation after detection of the Master MPU error and The TrustZone Filter error. By confirming BUS.BUS3ERRSTAT and BUS.DMACDTCERRSTAT, judge whether it is the Master or the Slave.
Note 2. If the error response detection interrupt (DMA_TRANSERR) occurs and NMI of the Master MPU or NMI of the TrustZone Filter has not occurred, treat it as the Illegal address access error or the Slave Bus Error. It can be judged also by BUS.BUS3ERRSTAT and BUS. DMACDTCERRSTAT.

Note that if the bus error occurs when writing the last data of transfer, the transfer end event and the error response detection interrupt (DMA_TRANSERR) occurs.

## 17.9 Event Link

The DTC can produce an event link request on completion of one transfer request.

## 17.10 Low Power Consumption Function

Before transitioning to the module-stop state, Software Standby mode without Snooze mode transition, Deep Software Standby mode, set the DTCST.DTCST bit to 0, and then perform the operations described in the following sections. The DTC is available in Snooze mode by setting the SYSTEM.SNZCR.SNZDTCEN bit to 1. See section 10, Low Power Modes.

### (1) Module-Stop Function

Writing 1 to the MSTPCRA.MSTPA22 bit enables the module-stop function of the DTC. If a DTC transfer is in progress when 1 is written to the MSTPCRA.MSTPA22 bit, the transition to the module-stop state proceeds after the DTC transfer ends. While the MSTPCRA.MSTPA22 bit is 1, accessing the DTC registers is prohibited. Writing 0 to the MSTPCRA.MSTPA22 bit releases the DTC from the module-stop state.

### (2) Software Standby Mode and Deep Software Standby Mode

Use the settings described in section 10.7.1. Transitioning to Software Standby Mode or section 10.9.1. Transitioning to Deep Software Standby Mode.

If DTC transfer operations are in progress when the WFI instruction is executed, the transition to Software Standby mode or Deep Software Standby mode is executed after the completion of the DTC transfer.

### (3) Snooze Mode

When the snooze control circuit receives a snooze request in Software Standby mode, the MCU transitions to Snooze mode. See section 10.8.1. Transition to Snooze Mode. DTC operation in Snooze mode can be selected in the SYSTEM.SNZCR.SNZDTCEN bit. If DTC operation is enabled in Snooze mode, before transitioning to Software Standby mode, set the DTCST.DTCST bit to 1. To return to Software Standby mode through DTC, set SYSTEM.SNZEDCR0.DTCZRED or SYSTEM.SNZEDCR0.DTCNZRED to 1. See section 10.8.3. Returning from Snooze Mode to Software Standby Mode. SYSTEM.SNZEDCR0.DTCZRED enables or disables a snooze end request on completion of the last DTC transmission, detected on DTC transmission completion when CRA and CRB are 0. SYSTEM.SNZEDCR0.DTCNZRED enables or disables a snooze end request on a not last DTC transmission completion (CRA and CRB are not 0), detected on DTC transmission completion when CRA and CRB are not 0. The DTC activation request from the ICU is stopped during Software Standby mode but not stopped during Snooze mode.

#### (4) Notes on Low Power Consumption Function

For the WFI instruction and the register setting procedure, see section 10, Low Power Modes.

To perform a DTC transfer after returning from a low power mode without a Snooze mode transition, set the DTCST.DTCST bit to 1 again.

To use a request that is generated in Software Standby mode as an interrupt request to the CPU but not as a DTC activation request, specify the CPU as the interrupt request destination as described in section 13.4.1. Detecting Interrupts, then execute the WFI instruction. If DTC operation is enabled in Snooze mode, do not use the module-stop function of the DTC.

## 17.11 Usage Notes

### 17.11.1 Transfer Information Start Address

You must set multiples of 4 for the transfer information start addresses in the vector table. Otherwise, such addresses are accessed with their lowest 2 bits regarded as 00b.

# 18. Event Link Controller (ELC)

## 18.1 Overview

The Event Link Controller (ELC) uses the event requests generated by various peripheral modules as source signals to connect them to different modules, allowing direct link between the modules without CPU intervention.

Table 18.1 lists the ELC specifications, and Figure 18.1 shows a block diagram.

**Table 18.1 ELC Specifications**

| Item | Description |
|---|---|
| Event link function | 171 types of event signals can be directly connected to modules. The ELC generates the ELC event signal, and events that activate the DTC. |
| Module-stop function | Module-stop state can be set. |
| TrustZone Filter | Security attribution can be set for each registers |

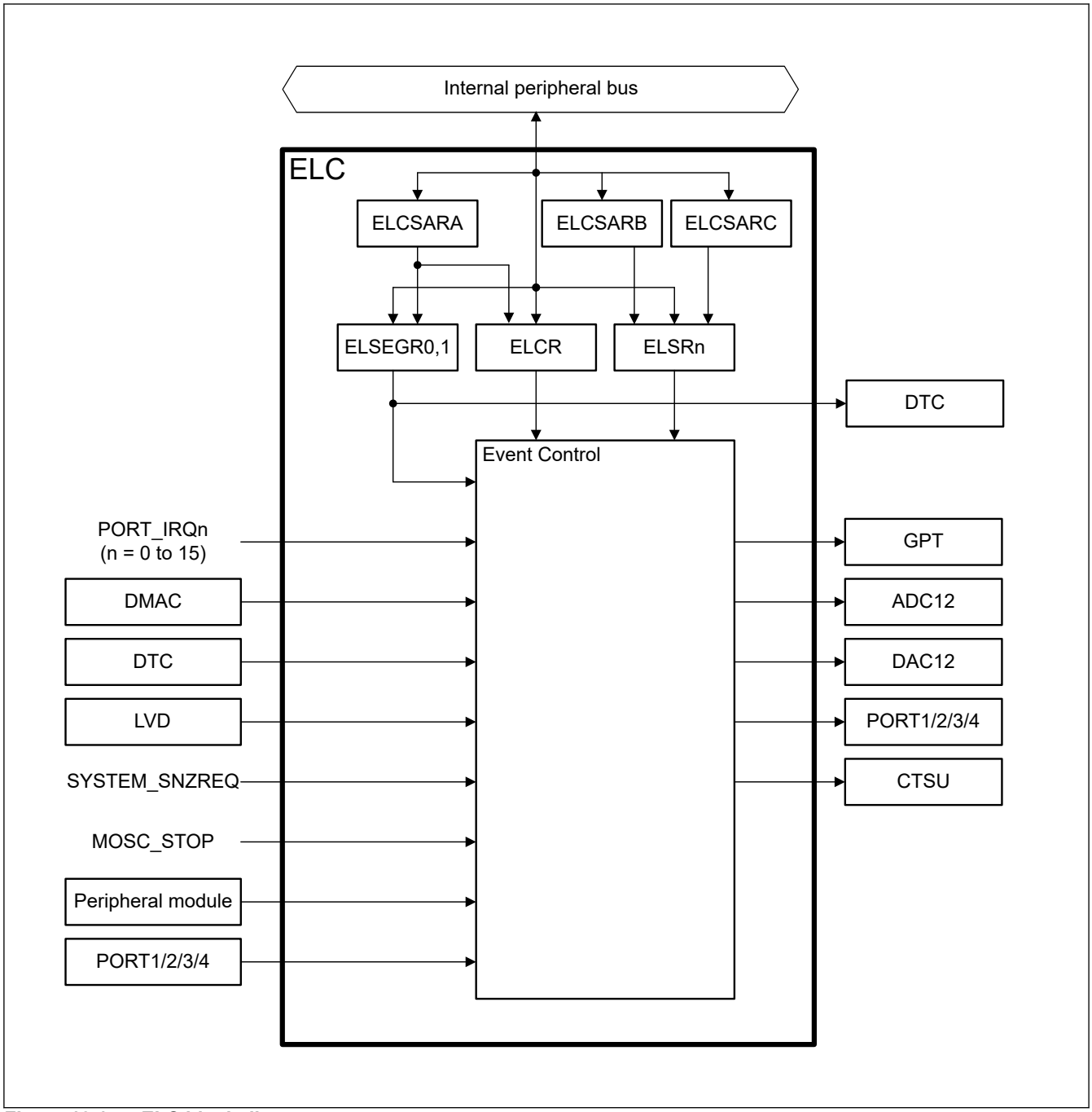



**Figure 18.1 ELC block diagram**

## 18.2 Register Descriptions

### 18.2.1 ELCR : Event Link Controller Register

Base address: ELC = 0x4008_2000

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ELCON | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | ELCON | All Event Link Enable<br>0: ELC function is disabled.<br>1: ELC function is enabled. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The ELCR register controls the ELC operation.

## 18.2.2 ELSEGRn : Event Link Software Event Generation Register n (n = 0, 1)

Base address: ELC = 0x4008_2000

Offset address: 0x02 + 0x02 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | WI | WE | — | — | — | — | — | SEG |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SEG | Software Event Generation<br>0: Normal operation<br>1: Software event is generated. | W |
| 5:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | WE | SEG Bit Write Enable<br>0: Write to SEG bit disabled.<br>1: Write to SEG bit enabled. | R/W |
| 7 | WI | ELSEGR Register Write Disable<br>0: Write to ELSEGR register enabled.<br>1: Write to ELSEGR register disabled. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### SEG bit (Software Event Generation)

When 1 is written to the SEG bit while the WE bit is 1, a software event is generated. This bit is read as 0. Even when 1 is written to this bit, data is not stored. The WE bit must be set to 1 before writing to this bit.

A software event can trigger a linked DTC event.

### WE bit (SEG Bit Write Enable)

The SEG bit can only be written to when the WE bit is 1. Clear the WI bit to 0 before writing to this bit.

[Setting condition]

- If 1 is written to this bit while the WI bit is 0, this bit becomes 1.

[Clearing condition]

- If 0 is written to this bit while the WI bit is 0, this bit becomes 0.

### WI bit (ELSEGR Register Write Disable)

The ELSEGR register can only be written to when the write value to the WI bit is 0. This bit is read as 1. Before setting the WE or SEG bit, the WI bit must be set to 0.

### 18.2.3 ELSRn : Event Link Setting Register n (n = 0 to 9, 12 to 18)

Base address: ELC = 0x4008_2000

Offset address: 0x10 + 0x04 × n

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | ELS[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | ELS[8:0] | Event Link Select<br>0x000: Event output disabled for the associated peripheral module<br>0x001: Number setting for the event signal to be linked<br>⋮<br>0x1DB: Number setting for the event signal to be linked<br>Others: Settings prohibited | R/W |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The ELSRn register specifies an event signal to be linked to each peripheral module. Table 18.2 shows the association between the ELSRn register and the peripheral modules. Table 18.3 shows the association between the event signal names set in the ELSRn register and the signal numbers.

**Table 18.2 Association between the ELSRn registers and peripheral functions**

| Register name | Peripheral function (module) | Event name |
|---|---|---|
| ELSR0 | GPT (A) | ELC_GPTA |
| ELSR1 | GPT (B) | ELC_GPTB |
| ELSR2 | GPT (C) | ELC_GPTC |
| ELSR3 | GPT (D) | ELC_GPTD |
| ELSR4 | GPT (E) | ELC_GPTE |
| ELSR5 | GPT (F) | ELC_GPTF |
| ELSR6 | GPT (G) | ELC_GPTG |
| ELSR7 | GPT (H) | ELC_GPTH |
| ELSR8 | ADC12A0 | ELC_AD00 |
| ELSR9 | ADC12B0 | ELC_AD01 |
| ELSR12 | DAC12 channel 0 | ELC_DA0 |
| ELSR13 | DAC12 channel 1 | ELC_DA1 |
| ELSR14 | PORT1 | ELC_PORT1 |
| ELSR15 | PORT2 | ELC_PORT2 |
| ELSR16 | PORT3 | ELC_PORT3 |
| ELSR17 | PORT4 | ELC_PORT4 |
| ELSR18 | CTSU | ELC_CTSU |

**Table 18.3 Association between event signal names set in ELSRn.ELS[8:0] bits and signal numbers (1 of 5)**

| Event number | Interrupt request source | Name | Description |
|---|---|---|---|
| 0x001 | Port | PORT_IRQ0*1 | External pin interrupt 0 |
| 0x002 | | PORT_IRQ1*1 | External pin interrupt 1 |
| 0x003 | | PORT_IRQ2*1 | External pin interrupt 2 |
| 0x004 | | PORT_IRQ3*1 | External pin interrupt 3 |
| 0x005 | | PORT_IRQ4*1 | External pin interrupt 4 |
| 0x006 | | PORT_IRQ5*1 | External pin interrupt 5 |
| 0x007 | | PORT_IRQ6*1 | External pin interrupt 6 |
| 0x008 | | PORT_IRQ7*1 | External pin interrupt 7 |
| 0x009 | | PORT_IRQ8*1 | External pin interrupt 8 |
| 0x00A | | PORT_IRQ9*1 | External pin interrupt 9 |
| 0x00B | | PORT_IRQ10*1 | External pin interrupt 10 |
| 0x00C | | PORT_IRQ11*1 | External pin interrupt 11 |
| 0x00D | | PORT_IRQ12*1 | External pin interrupt 12 |
| 0x00E | | PORT_IRQ13*1 | External pin interrupt 13 |
| 0x00F | | PORT_IRQ14*1 | External pin interrupt 14 |
| 0x010 | | PORT_IRQ15*1 | External pin interrupt 15 |
| 0x020 | DMAC | DMAC0_INT | DMAC transfer end 0 |
| 0x021 | | DMAC1_INT | DMAC transfer end 1 |
| 0x022 | | DMAC2_INT | DMAC transfer end 2 |
| 0x023 | | DMAC3_INT | DMAC transfer end 3 |
| 0x024 | | DMAC4_INT | DMAC transfer end 4 |
| 0x025 | | DMAC5_INT | DMAC transfer end 5 |
| 0x026 | | DMAC6_INT | DMAC transfer end 6 |
| 0x027 | | DMAC7_INT | DMAC transfer end 7 |
| 0x02A | DTC | DTC_DTCEND*4 | DTC transfer end |
| 0x038 | LVD | LVD_LVD1 | Voltage monitor 1 interrupt |
| 0x039 | | LVD_LVD2 | Voltage monitor 2 interrupt |
| 0x03B | MOSC | MOSC_STOP | Mail Clock oscillation stop |
| 0x03C | LPW | SYSTEM_SNZREQ*3*4 | Snooze enrty |
| 0x040 | AGT0 | AGT0_AGTI | AGT interrupt |
| 0x041 | | AGT0_AGTCMAI | Compare match A |
| 0x042 | | AGT0_AGTCMBI | Compare match B |
| 0x043 | AGT1 | AGT1_AGTI | AGT interrupt |
| 0x044 | | AGT1_AGTCMAI | Compare match A |
| 0x045 | | AGT1_AGTCMBI | Compare match B |
| 0x046 | AGT2 | AGT2_AGTI | AGT interrupt |
| 0x047 | | AGT2_AGTCMAI | Compare match A |
| 0x048 | | AGT2_AGTCMBI | Compare match B |

**Table 18.3 Association between event signal names set in ELSRn.ELS[8:0] bits and signal numbers (2 of 5)**

| Event number | Interrupt request source | Name | Description |
|---|---|---|---|
| 0x049 | AGT3 | AGT3_AGTI | AGT interrupt |
| 0x04A | | AGT3_AGTCMAI | Compare match A |
| 0x04B | | AGT3_AGTCMBI | Compare match B |
| 0x04C | AGT4 | AGT4_AGTI | AGT interrupt |
| 0x04D | | AGT4_AGTCMAI | Compare match A |
| 0x04E | | AGT4_AGTCMBI | Compare match B |
| 0x04F | AGT5 | AGT5_AGTI | AGT interrupt |
| 0x050 | | AGT5_AGTCMAI | Compare match A |
| 0x051 | | AGT5_AGTCMBI | Compare match B |
| 0x052 | IWDT | IWDT_NMIUNDF | IWDT underflow |
| 0x053 | CWDT | WDT_NMIUNDF | WDT underflow |
| 0x055 | RTC | RTC_PRD | Periodic interrupt |
| 0x073 | IIC0 | IIC0_RXI | Receive data full |
| 0x074 | | IIC0_TXI | Transmit data empty |
| 0x075 | | IIC0_TEI | Transmit end |
| 0x076 | | IIC0_EEI | Transfer error |
| 0x078 | IIC1 | IIC1_RXI | Receive data full |
| 0x079 | | IIC1_TXI | Transmit data empty |
| 0x07A | | IIC1_TEI | Transmit end |
| 0x07B | | IIC1_EEI | Transfer error |
| 0x0B1 | I/O Port | IOPORT_GROUP1 | Port 1 event |
| 0x0B2 | | IOPORT_GROUP2 | Port 2 event |
| 0x0B3 | | IOPORT_GROUP3 | Port 3 event |
| 0x0B4 | | IOPORT_GROUP4 | Port 4 event |
| 0x0B5 | ELC | ELC_SWEVT0 | Software event 0 |
| 0x0B6 | | ELC_SWEVT1 | Software event 1 |
| 0x0C0 | GPT0 | GPT0_CCMPA | Compare match A |
| 0x0C1 | | GPT0_CCMPB | Compare match B |
| 0x0C2 | | GPT0_CMPC | Compare match C |
| 0x0C3 | | GPT0_CMPD | Compare match D |
| 0x0C4 | | GPT0_CMPE | Compare match E |
| 0x0C5 | | GPT0_CMPF | Compare match F |
| 0x0C6 | | GPT0_OVF | Overflow |
| 0x0C7 | | GPT0_UDF | Underflow |
| 0x0C8 | | GPT0_PC | Cycle count function end |

**Table 18.3 Association between event signal names set in ELSRn.ELS[8:0] bits and signal numbers (3 of 5)**

| Event number | Interrupt request source | Name | Description |
|---|---|---|---|
| 0x0C9 | GPT1 | GPT1_CCMPA | Compare match A |
| 0x0CA | | GPT1_CCMPB | Compare match B |
| 0x0CB | | GPT1_CMPC | Compare match C |
| 0x0CC | | GPT1_CMPD | Compare match D |
| 0x0CD | | GPT1_CMPE | Compare match E |
| 0x0CE | | GPT1_CMPF | Compare match F |
| 0x0CF | | GPT1_OVF | Overflow |
| 0x0D0 | | GPT1_UDF | Underflow |
| 0x0D1 | | GPT1_PC | Cycle count function end |
| 0x0D2 | GPT2 | GPT2_CCMPA | Compare match A |
| 0x0D3 | | GPT2_CCMPB | Compare match B |
| 0x0D4 | | GPT2_CMPC | Compare match C |
| 0x0D5 | | GPT2_CMPD | Compare match D |
| 0x0D6 | | GPT2_CMPE | Compare match E |
| 0x0D7 | | GPT2_CMPF | Compare match F |
| 0x0D8 | | GPT2_OVF | Overflow |
| 0x0D9 | | GPT2_UDF | Underflow |
| 0x0DB | GPT3 | GPT3_CCMPA | Compare match A |
| 0x0DC | | GPT3_CCMPB | Compare match B |
| 0x0DD | | GPT3_CMPC | Compare match C |
| 0x0DE | | GPT3_CMPD | Compare match D |
| 0x0DF | | GPT3_CMPE | Compare match E |
| 0x0E0 | | GPT3_CMPF | Compare match F |
| 0x0E1 | | GPT3_OVF | Overflow |
| 0x0E2 | | GPT3_UDF | Underflow |
| 0x0E4 | GPT4 | GPT4_CCMPA | Compare match A |
| 0x0E5 | | GPT4_CCMPB | Compare match B |
| 0x0E6 | | GPT4_CMPC | Compare match C |
| 0x0E7 | | GPT4_CMPD | Compare match D |
| 0x0E8 | | GPT4_CMPE | Compare match E |
| 0x0E9 | | GPT4_CMPF | Compare match F |
| 0x0EA | | GPT4_OVF | Overflow |
| 0x0EB | | GPT4_UDF | Underflow |
| 0x0EC | | GPT4_PC | Cycle count function end |

**Table 18.3 Association between event signal names set in ELSRn.ELS[8:0] bits and signal numbers (4 of 5)**

| Event number | Interrupt request source | Name | Description |
|---|---|---|---|
| 0x0ED | GPT5 | GPT5_CCMPA | Compare match A |
| 0x0EE | | GPT5_CCMPB | Compare match B |
| 0x0EF | | GPT5_CMPC | Compare match C |
| 0x0F0 | | GPT5_CMPD | Compare match D |
| 0x0F1 | | GPT5_CMPE | Compare match E |
| 0x0F2 | | GPT5_CMPF | Compare match F |
| 0x0F3 | | GPT5_OVF | Overflow |
| 0x0F4 | | GPT5_UDF | Underflow |
| 0x0F5 | | GPT5_PC | Cycle count function end |
| 0x0F6 | GPT6 | GPT6_CCMPA | Compare match A |
| 0x0F7 | | GPT6_CCMPB | Compare match B |
| 0x0F8 | | GPT6_CMPC | Compare match C |
| 0x0F9 | | GPT6_CMPD | Compare match D |
| 0x0FA | | GPT6_CMPE | Compare match E |
| 0x0FB | | GPT6_CMPF | Compare match F |
| 0x0FC | | GPT6_OVF | Overflow |
| 0x0FD | | GPT6_UDF | Underflow |
| 0x0FE | | GPT6_PC | Cycle count function end |
| 0x0FF | GPT7 | GPT7_CCMPA | Compare match A |
| 0x100 | | GPT7_CCMPB | Compare match B |
| 0x101 | | GPT7_CMPC | Compare match C |
| 0x102 | | GPT7_CMPD | Compare match D |
| 0x103 | | GPT7_CMPE | Compare match E |
| 0x104 | | GPT7_CMPF | Compare match F |
| 0x105 | | GPT7_OVF | Overflow |
| 0x106 | | GPT7_UDF | Underflow |
| 0x150 | GPT | GPT_UVWEDGE | UVW edge event |
| 0x160 | ADC12 | ADC120_ADI | A/D scan end interrupt |
| 0x164 | | ADC120_WCMPM*4 | Compare match |
| 0x165 | | ADC120_WCMPUM*4 | Compare mismatch |
| 0x180 | SCI0 | SCI0_RXI*2 | Receive data full |
| 0x181 | | SCI0_TXI*2 | Transmit data empty |
| 0x182 | | SCI0_TEI*2 | Transmit end |
| 0x183 | | SCI0_ERI | Receive error |
| 0x184 | | SCI0_AM | Address match event |
| 0x186 | SCI1 | SCI1_RXI*2 | Received data full |
| 0x187 | | SCI1_TXI*2 | Transmit data empty |
| 0x188 | | SCI1_TEI*2 | Transmit end |
| 0x189 | | SCI1_ERI | Receive error |

**Table 18.3 Association between event signal names set in ELSRn.ELS[8:0] bits and signal numbers (5 of 5)**

| Event number | Interrupt request source | Name | Description |
|---|---|---|---|
| 0x18C | SCI2 | SCI2_RXI*2 | Received data full |
| 0x18D | | SCI2_TXI*2 | Transmit data empty |
| 0x18E | | SCI2_TEI*2 | Transmit end |
| 0x18F | | SCI2_ERI | Receive error |
| 0x192 | SCI3 | SCI3_RXI*2 | Received data full |
| 0x193 | | SCI3_TXI*2 | Transmit data empty |
| 0x194 | | SCI3_TEI*2 | Transmit end |
| 0x195 | | SCI3_ERI | Receive error |
| 0x196 | | SCI3_AM | Address match event |
| 0x198 | SCI4 | SCI4_RXI*2 | Received data full |
| 0x199 | | SCI4_TXI*2 | Transmit data empty |
| 0x19A | | SCI4_TEI*2 | Transmit end |
| 0x19B | | SCI4_ERI | Receive error |
| 0x19C | | SCI4_AM | Address match event |
| 0x1B6 | SCI9 | SCI9_RXI*2 | Received data full |
| 0x1B7 | | SCI9_TXI*2 | Transmit data empty |
| 0x1B8 | | SCI9_TEI*2 | Transmit end |
| 0x1B9 | | SCI9_ERI | Receive error |
| 0x1BA | | SCI9_AM | Address match event |
| 0x1C4 | SPI0 | SPI0_SPRI | Receive buffer full |
| 0x1C5 | | SPI0_SPTI | Transmit buffer empty |
| 0x1C6 | | SPI0_SPII | Idle |
| 0x1C7 | | SPI0_SPEI | Error |
| 0x1C8 | | SPI0_SPCEND | Communication complete event |
| 0x1DB | DOC | DOC_DOPCI*4 | Data operation circuit interrupt |

Note 1. Only pulse (edge detection) is supported.
Note 2. This event is not supported in FIFO mode.
Note 3. ELSR8, ELSR9, ELSR14 to ELSR17, and ELSR18 can select this event.
Note 4. This event can occur in Snooze mode.

## 18.2.4 ELCSARA : Event Link Controller Security Attribution Register A

Base address: ELC = 0x4008_2000

Offset address: 0x74

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | ELSEGR1 | ELSEGR0 | ELCR |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ELCR | Event Link Controller Register Security Attribution<br>Target register: ELCR<br>0: Secure<br>1: Non-secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | ELSEGR0 | Event Link Software Event Generation Register 0 Security Attribution<br>0: Secure<br>1: Non-secure | R/W |
| 2 | ELSEGR1 | Event Link Software Event Generation Register 1 Security Attribution<br>0: Secure<br>1: Non-secure | R/W |
| 15:3 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

The ELCR register controls operation of the ELC.

## 18.2.5 ELCSARB : Event Link Controller Security Attribution Register B

Base address: ELC = 0x4008_2000

Offset address: 0x78

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ELSR[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ELSR[15:0] | Event Link Setting Register n Security Attribution<br>Target register: ELSRn (n = 0 to 15)<br>0: Secure<br>1: Non-secure | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

This register specifies the security attribution for the Register ELSRn (n = 0 to 15).

## 18.2.6 ELCSARC : Event Link Controller Security Attribution Register C

Base address: ELC = 0x4008_2000

Offset address: 0x7C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | ELSR[18:16] | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | ELSR[18:16] | Event Link Setting Register n Security Attribution (n = 16 to 18)<br>Target register: ELSRn (n = 16 to 18)<br>0: Secure<br>1: Non-secure | R/W |
| 15:3 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

This register specifies the security attribution for the Register ELSRn (n = 16 to 18)

## 18.3 Operation

### 18.3.1 Relation between Interrupt Handling and Event Linking

Event number for an event link is the same as that for the associated interrupt source. For information on generating event signals, see the explanation in the chapter for each event source module.

### 18.3.2 Linking Events

When an event occurs and that event is already set as a trigger in the Event Link Setting Register (ELSRn), the associated module is activated. The operation of the module must be set up in advance. Table 18.4 lists the operations of modules when an event occurs.

**Table 18.4 Module operations when event occurs**

| Module | Operations When Event is Input |
|---|---|
| GPT | • Start counting<br>• Stop counting<br>• Clear counting<br>• Up counting<br>• Down counting<br>• Input capture |
| DAC12 | Start D/A conversion |
| I/O Ports | • Change pin output based on the EORR (reset) or EOSR (set)<br>• Latch pin state to EIDR<br>• The following ports can be used for the ELC:<br>Port 1<br>Port 2<br>Port 3<br>Port 4 |
| CTSU | Start measurement operation |
| ADC12 | Start A/D conversion |
| DTC | Start DTC data transfer |

### 18.3.3 Example of Procedure for Linking Events

To link events:

1. Set the operation of the module for which an event is to be linked.
2. Set the appropriate ELSRn.ELS[8:0] bits for the module to be linked.
3. Set the ELCR.ELCON bit to 1 to enable linkage of all events.
4. Configure the module from which an event is output and activate the module. The link between the two modules is now active.
5. To stop event linkage of modules individually, set 0 to the ELSRn.ELS[8:0] bit associated with the modules. To stop linkage of all the events, set the ELCR.ELCON bit to 0.

If event link output from the RTC is to be used, set the ELC after the RTC settings, for example, for initialization and time setting. Unintended events may be generated if RTC settings are made after the ELC settings.

If event link output from the LVD is to be used, set the ELC after setting the LVD. To disable the LVD, do so after setting 0x00 to the associated ELSRn register.

## 18.4 Usage Notes

### 18.4.1 Linking DMAC/DTC Transfer End Signals as Events

When linking the DMAC/DTC transfer end signals as events, do not set the same peripheral module as the DMAC/DTC transfer destination and event link destination. If set, the peripheral module might be started before DMAC/DTC transfer to the peripheral module is complete.

### 18.4.2 Setting Clocks

To link events, you must enable the ELC and the related modules. The modules cannot operate if the related modules are in the module-stop state or in low power mode in which the module is stopped (Software Standby mode or Deep Software Standby mode).

Some modules can perform in Snooze mode. For more information, see Table 18.3 and section 10, Low Power Modes.

### 18.4.3 Module-Stop Function Setting

The Module Stop Control Register C (MSTPCRC) can enable or disable ELC operation. The ELC is initially stopped after reset. Releasing the module-stop state enables access to the registers. The ELCON bit must be set to 0 before disabling ELC operation using the Module Stop Control Register. For more information, see Table 18.3 and section 10, Low Power Modes.

### 18.4.4 ELC Delay Time

In Figure 18.2, module A accesses module B through the ELC. There is a delay time in the ELC between module A and module B. Table 18.5 shows the ELC delay time.

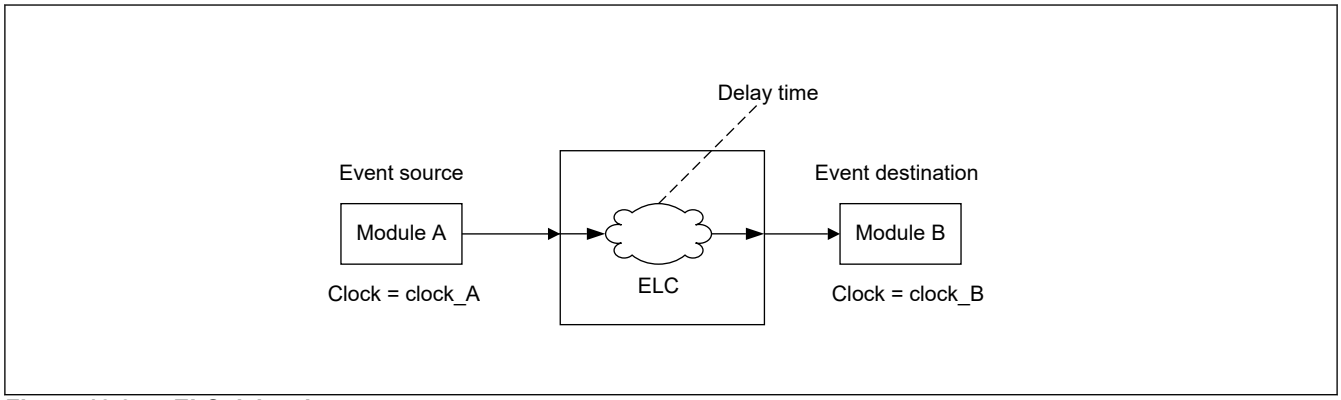



**Figure 18.2 ELC delay time**

**Table 18.5 ELC delay time**

| Clock domain | Clock frequency | ELC delay time |
|---|---|---|
| clock_A = clock_B | clock_A = clock_B | 0 cycle |
| clock_A ≠ clock_B | clock_A = clock_B | 1 cycle to 2 cycles |
| | clock_A > clock_B | 1 cycle to 2 cycles of clock_B |
| | clock_A < clock_B | 1 cycle to 2 cycles of clock_A |

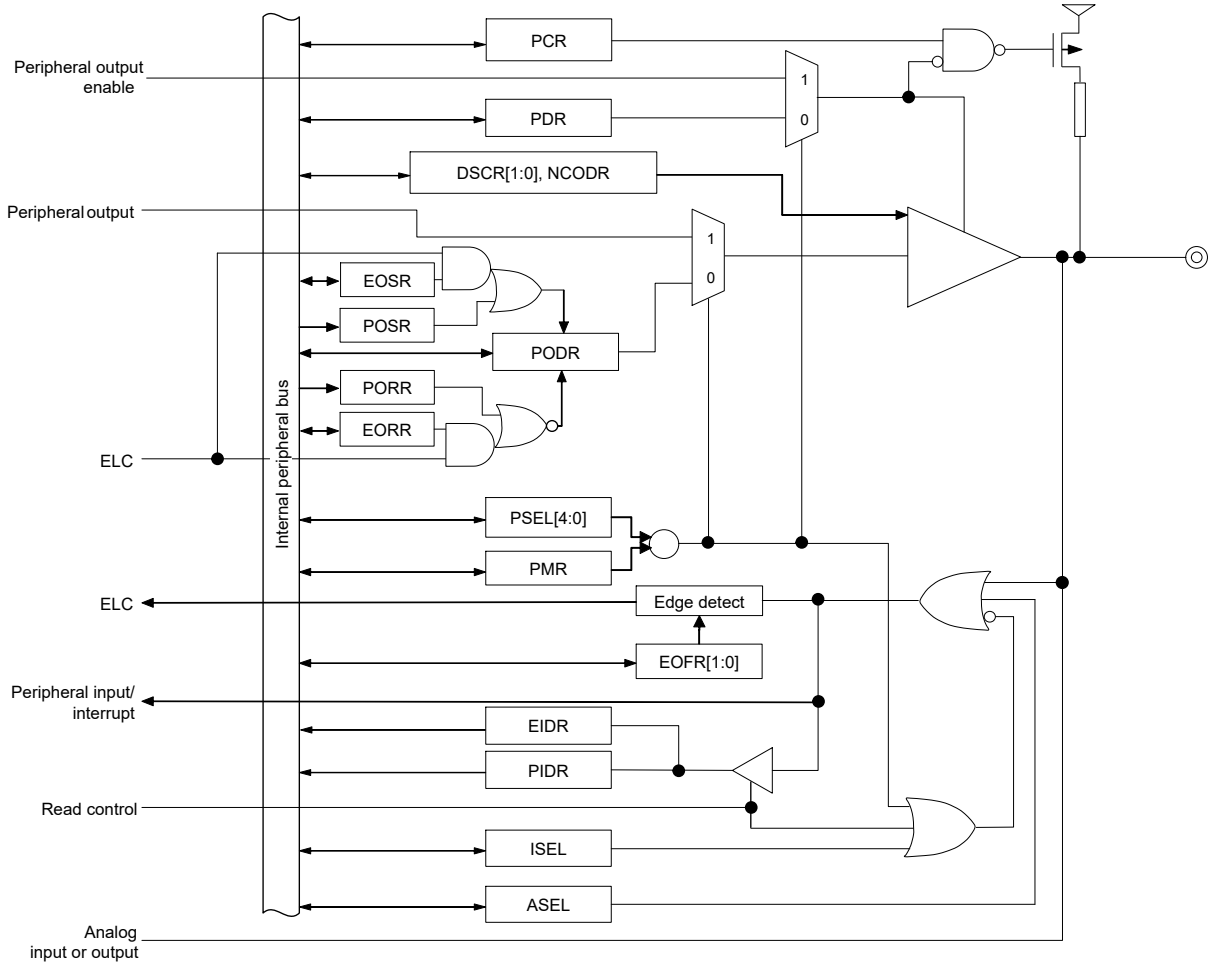
# 19. I/O Ports

## 19.1 Overview

The I/O port pins operate as general I/O port pins, I/O pins for peripheral modules, interrupt input pins, analog I/O, port group function for the ELC.

All pins except P109 ( as TDO of JTAG ports ) operate as input pins immediately after a reset, and pin functions are switched by register settings. The I/O ports and peripheral modules for each pin are specified in the associated registers.

Figure 19.1 shows a connection diagram for the I/O port registers. The configuration of the I/O ports differs for different packages. Table 19.1 lists the I/O port specifications by package, and Table 19.2 lists the port functions.




Note: This figure shows a basic port configuration. The configuration differs depending on the ports.

**Figure 19.1 Connection diagram for I/O port registers**

**Table 19.1 I/O port specifications (1 of 2)**

| Port | Package | Number of pins | Package | Number of pins | Package | Number of pins |
|---|---|---|---|---|---|---|
| | 100 pins | | 64 pins | | 48 pins | |
| PORT0 | P000 to P008, P013 to P015 | 12 | P000 to P004, P013 to P015 | 8 | P000 to P002, P013 to P015 | 6 |
| PORT1 | P100 to P115 | 16 | P100 to P113 | 14 | P100 to P104, P108 to P112 | 10 |
| PORT2 | P200, P201, P205 to P214 | 12 | P200, P201, P205 to P208, P212, P213 | 8 | P200,P201, P206, P207, P212, P213 | 6 |

**Table 19.1 I/O port specifications (2 of 2)**

| Port | Package 100 pins | Number of pins | Package 64 pins | Number of pins | Package 48 pins | Number of pins |
|---|---|---|---|---|---|---|
| PORT3 | P300 to P307 | 8 | P300 to P304 | 5 | P300 to P302 | 3 |
| PORT4 | P400 to P415 | 16 | P400 to P402, P407 to P411 | 8 | P402, P407 to P409 | 4 |
| PORT5 | P500 to P505 | 6 | P500 | 1 | P500 | 1 |
| PORT6 | P600 to P603, P608 to P610 | 7 | — | 0 | — | 0 |
| PORT7 | P708 | 1 | — | 0 | — | 0 |

**Table 19.2 I/O port functions**

| Port | Port name | Input pull-up | Open-drain output | Drive capacity switching | 5V tolerant | I/O |
|---|---|---|---|---|---|---|
| PORT0 | P000 to P008, P013 to P015 | ✓ | ✓ | Low | — | Input / Output |
| PORT1 | P100 to P115 | ✓ | ✓ | Low, middle, high | — | Input / Output |
| PORT2 | P200 | ✓ | — | — | — | Input |
| | P201 | ✓ | ✓ | Low | — | Input / Output |
| | P207 to P214 | ✓ | ✓ | Low, middle, high | — | Input / Output |
| | P205, P206 | ✓ | ✓ | Low, middle, high | ✓ | Input / Output |
| PORT3 | P300 to P307 | ✓ | ✓ | Low, middle, high | — | Input / Output |
| PORT4 | P400, P401, P407 to P415 | ✓ | ✓ | Low, middle, high | ✓ | Input / Output |
| | P402 to P406 | ✓ | ✓ | Low, middle, high | — | Input / Output |
| PORT5 | P500 to P505 | ✓ | ✓ | Low, middle, high | — | Input / Output |
| PORT6 | P600 to P603, P608 to P610 | ✓ | ✓ | Low, middle, high | — | Input / Output |
| PORT7 | P708 | ✓ | ✓ | Low, middle, high | ✓ | Input / Output |

Note: ✓: Available
—: Setting prohibited

## 19.2 Register Descriptions

### 19.2.1 PCNTR1/PODR/PDR : Port Control Register 1

Base address: PORTm = 0x4008_0000 + 0x0020 × m (m = 0 to 7)

Offset address: 0x000 (PCNTR1/PODR)
0x002 (PDR)

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PODR15 | PODR14 | PODR13 | PODR12 | PODR11 | PODR10 | PODR09 | PODR08 | PODR07 | PODR06 | PODR05 | PODR04 | PODR03 | PODR02 | PODR01 | PODR00 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PDR15 | PDR14 | PDR13 | PDR12 | PDR11 | PDR10 | PDR09 | PDR08 | PDR07 | PDR06 | PDR05 | PDR04 | PDR03 | PDR02 | PDR01 | PDR00 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | PDR15 to PDR00 | Pmn Direction<br>0: Input (functions as an input pin)<br>1: Output (functions as an output pin) | R/W*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:16 | PODR15 to PODR00 | Pmn Output Data<br>0: Low output<br>1: High output | R/W*2 |

Note: m = 0 to 7, n = 00 to 15
Note 1. If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 2. If the security attribution is configured as Secure:
- Secure access is allowed
- Non-secure read value is 0 and TrustZone access error is not generated
- Non-secure write access is ignored and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The Port Control Register 1 (PCNTR1/PODR/PDR) is a 32-bit or 16-bit read/write register that controls port direction and port output data. The PCNTR1 specifies the port direction and output data, and is accessed in 32-bit units. The PDRn (bits [15:0] in PCNTR1) and PODRn (bits [31:16] in PCNTR1) respectively, are accessed in 16-bit units.

### PDRn bits (Pmn Direction)

The PDRn bits select the input or output direction for individual pins on the associated port when the pins are configured as general I/O pins. Each pin on port m is associated with a PORTm.PCNTR1.PDRn bit. The I/O direction can be specified in 1-bit unit. Bits associated with non-existent pins are reserved. Reserved bits are read as 0. The write value should be 0. In the case of input only ports, PDRn bits are reserved. See Table 19.2. The PDRn bit in the PORTm.PCNTR1 register serves the same function as the PDR bit in the PFS.PmnPFS register.

### PODRn bits (Pmn Output Data)

The PODRn bits hold data to be output from the general I/O pins. Bits of non-existent port m are reserved. Reserved bits are read as 0. The write value should be 0. In the case of input only ports, PODRn bits are reserved. See Table 19.2. The PODRn bit in the PORTm.PCNTR1 register serves the same function as the PODR bit in the PFS.PmnPFS register.

## 19.2.2 PCNTR2/EIDR/PIDR : Port Control Register 2

Base address: PORTm = 0x4008_0000 + 0x0020 × m (m = 0 to 7)

Offset address: 0x004 (PCNTR2/EIDR)
0x006 (PIDR)

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | EIDR15 | EIDR14 | EIDR13 | EIDR12 | EIDR11 | EIDR10 | EIDR09 | EIDR08 | EIDR07 | EIDR06 | EIDR05 | EIDR04 | EIDR03 | EIDR02 | EIDR01 | EIDR00 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PIDR15 | PIDR14 | PIDR13 | PIDR12 | PIDR11 | PIDR10 | PIDR09 | PIDR08 | PIDR07 | PIDR06 | PIDR05 | PIDR04 | PIDR03 | PIDR02 | PIDR01 | PIDR00 |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | PIDR15 to PIDR00 | Pmn State<br>0: Low level<br>1: High level | R |
| 31:16 | EIDR15 to EIDR00 *2 | Port Event Input Data*1<br>When an ELC_PORTx signal occurs<br>0: Low input<br>1: High input | R |

Note: If the security attribution is configured as Secure:
- Secure read access is allowed
- Non-secure read value is 0 and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:

- Secure and Non-secure read access are allowed.

Note: m = 0 to 7, n = 00 to 15
Note 1. x = 1, 2, 3 or 4 for EIDR only
Note 2. Supported by ports 1, 2, 3 or 4.

The Port Control Register 2 (PCNTR2/EIDR/PIDR) allows read access to the Pmn state and the port event input data using 32-bit or 16-bit access.

The PCNTR2 represents the Pmn state and the port event input data, and is accessed in 32-bit units.

The PIDRn (bits [15:0] in PCNTR2) and EIDRn (bits [31:16] in PCNTR2) respectively, are accessed in 16-bit units. Bits associated with non-existent pins are reserved. Reserved bits are read as undefined.

### PIDRn bits (Pmn State)

The PIDRn bits reflect the individual pin states of the port, regardless of the values set in PmnPFS.PMR and PORTm.PCNTR1.PDRn. The PIDRn bit in the PORTm.PCNTR2 register serves the same function as the PIDR bit in the PFS.PmnPFS register.

A pin state cannot be reflected in PIDRn when one of the following functions is enabled:

- RTC Time Capture input (RTCIC)
- Analog function (ASEL = 1)
- Capacitive Touch Sensing Unit (CTSU)

### EIDRn bits (Port Event Input Data)

The EIDRn bits latch a pin state when an ELC_PORTx signal occurs. Pin states can only be input to EIDRn when PmnPFS.PMR and PORTm.PCNTR1.PDRn are 0. When the PmnPFS.ASEL bit is set to 1, the associated pin state is not reflected in EIDRn.

## 19.2.3 PCNTR3/PORR/POSR : Port Control Register 3

Base address: PORTm = 0x4008_0000 + 0x0020 × m (m = 0 to 7)

Offset address: 0x008 (PCNTR3/PORR)
0x00A (POSR)

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PORR 15 | PORR 14 | PORR 13 | PORR 12 | PORR 11 | PORR 10 | PORR 09 | PORR 08 | PORR 07 | PORR 06 | PORR 05 | PORR 04 | PORR 03 | PORR 02 | PORR 01 | PORR 00 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | POSR 15 | POSR 14 | POSR 13 | POSR 12 | POSR 11 | POSR 10 | POSR 09 | POSR 08 | POSR 07 | POSR 06 | POSR 05 | POSR 04 | POSR 03 | POSR 02 | POSR 01 | POSR 00 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | POSR15 to POSR00 | Pmn Output Set<br>0: No effect on output<br>1: High output | W |
| 31:16 | PORR15 to PORR00 | Pmn Output Reset<br>0: No effect on output<br>1: Low output | W |

Note: If the security attribution is configured as Secure:
- Secure write access is allowed
- Non-secure write access is ignored and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure write access are allowed.

Note: m = 0 to 7, n = 00 to 15

The Port Control Register 3 (PCNTR3/PORR/POSR) is a 32-bit or 16-bit write register that controls the setting or resetting of the port output data.

The PCNTR3 controls the setting or resetting of the port output data, and is accessed in 32-bit units.

The POSRn (bits [15:0] in PCNTR3) and the PORRn (bits [31:16] in PCNTR3) respectively, are accessed in 16-bit units.

### POSRn bits (Pmn Output Set)

POSR changes PODR when set by a software write. For example, for P100, when PORT1.PCNTR3.POSR00 = 1, PORT1.PCNTR1.PODR00 outputs 1. Bits associated with non-existent pins are reserved. The write value should always be 0. In the case of input only ports, POSRn bits are reserved. See Table 19.2.

### PORRn bits (Pmn Output Reset)

PORR changes PODR when reset by a software write. For example, for P100, when PORT1.PCNTR3.PORR00 = 1, PORT1.PCNTR1.PODR00 outputs 0. Bits associated with non-existent pins are reserved. The write value should always be 0. In the case of input only ports, PORRn bits are reserved. See Table 19.2.

Note: When EORRn or EOSRn is set, writing is prohibited to PODRn, PORRn, and POSRn.
Note: PORRn and POSRn should not be set at the same time.

## 19.2.4 PCNTR4/EORR/EOSR : Port Control Register 4

Base address: PORTm = 0x4008_0000 + 0x0020 × m (m = 1 to 4)

Offset address: 0x00C (PCNTR4/EORR)
0x00E (EOSR)

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | EORR 15 | EORR 14 | EORR 13 | EORR 12 | EORR 11 | EORR 10 | EORR 09 | EORR 08 | EORR 07 | EORR 06 | EORR 05 | EORR 04 | EORR 03 | EORR 02 | EORR 01 | EORR 00 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | EOSR 15 | EOSR 14 | EOSR 13 | EOSR 12 | EOSR 11 | EOSR 10 | EOSR 09 | EOSR 08 | EOSR 07 | EOSR 06 | EOSR 05 | EOSR 04 | EOSR 03 | EOSR 02 | EOSR 01 | EOSR 00 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | EOSR15 to EOSR00 | Pmn Event Output Set<br>When an ELC_PORTx signal occurs<br>0: No effect on output<br>1: High output | R/W |
| 31:16 | EORR15 to EORR0 | Pmn Event Output Reset<br>When an ELC_PORTx signal occurs<br>0: No effect on output<br>1: Low output | R/W |

Note: If the security attribution is configured as Secure:
- Secure access is allowed
- Non-secure read value is 0 and TrustZone access error is not generated
- Non-secure write access is ignored and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note: m = 1 to 4, n = 00 to 15, x = 1 to 4

The Port Control Register 4 (PCNTR4/EORR/EOSR) is a 32-bit or 16-bit read/write register that controls the setting or resetting of the port output data by an event input from the ELC.

The PCNTR4 controls the setting or resetting of the port output data by an event input from the ELC, and is accessed in 32-bit units.

The EOSRn (bits [15:0] in PCNTR4) and EORRn (bits [31:16] in PCNTR4) respectively, are accessed in 16-bit units.

### EOSRn bits (Pmn Event Output Set)

EOSR changes PODR when set because an ELC_PORTx signal occurs. For example, for P100 if PORT1.PCNTR4.EOSR00 is set to 1 when the ELC_PORTx occurs, PORT1.PCNTR1.PODR00 outputs 1. Bits associated with non-existent pins are reserved. The write value should always be 0. For input only ports, EOSRn bits are reserved. See Table 19.2.

### EORRn bits (Pmn Event Output Reset)

EORR changes PODR when reset because an ELC_PORTx signal occurs. For example, for P100 if PORT1.PCNTR4.EORR00 = 1 when the ELC_PORTx occurs, PORT1.PCNTR1.PODR00 outputs 0. Bits associated with non-existent pins are reserved. The write value should always be 0. For input only ports, EORRn bits are reserved. See Table 19.2.

Note: When EORRn or EOSRn is set, writing is prohibited to PODRn, PORRn, and POSRn.

Note: EORRn and EOSRn should not be set at the same time.

## 19.2.5 PmnPFS/PmnPFS_HA/PmnPFS_BY : Port mn Pin Function Select Register (m = 0 to 7, n = 00 to 15)

Base address: PFS = 0x4008_0800

Offset address: 0x000 + 0x040 × m + 0x004 × n (PmnPFS)
0x002 + 0x040 × m + 0x004 × n (PmnPFS_HA)
0x003 + 0x040 × m + 0x004 × n (PmnPFS_BY)

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | PSEL[4:0] | | | | | — | — | — | — | — | — | — | PMR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0*1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ASEL | ISEL | EOFR[1:0] | | DSCR[1:0] | | — | — | — | NCODR | — | PCR | — | PDR | PIDR | PODR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0*1 | 0 | 0 | 0 | 0 | 0 | 0*1 | 0 | 0 | x | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PODR | Port Output Data<br>0: Low output<br>1: High output | R/W*3 |
| 1 | PIDR | Pmn State<br>0: Low level<br>1: High level | R*4 |
| 2 | PDR | Port Direction<br>0: Input (functions as an input pin)<br>1: Output (functions as an output pin) | R/W*5 |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | PCR | Pull-up Control<br>0: Disable input pull-up<br>1: Enable input pull-up | R/W*5 |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | NCODR | N-Channel Open-Drain Control<br>0: CMOS output<br>1: NMOS open-drain output | R/W*5 |
| 9:7 | — | These bits are read as 0. The write value should be 0. | R/W |
| 11:10 | DSCR[1:0] | Port Drive Capability<br>0 0: Low drive<br>0 1: Middle drive<br>1 0: Setting prohibited<br>1 1: High drive | R/W*5 |
| 13:12 | EOFR[1:0] | Event on Falling/Event on Rising*2<br>0 0: Don't care<br>0 1: Detect rising edge<br>1 0: Detect falling edge<br>1 1: Detect both edges | R/W*5 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14 | ISEL | IRQ Input Enable<br>0: Not used as an IRQn input pin<br>1: Used as an IRQn input pin | R/W*5 |
| 15 | ASEL | Analog Input Enable<br>0: Not used as an analog pin<br>1: Used as an analog pin | R/W*5 |
| 16 | PMR | Port Mode Control<br>0: Used as a general I/O pin<br>1: Used as an I/O port for peripheral functions | R/W*5 |
| 23:17 | — | These bits are read as 0. The write value should be 0. | R/W |
| 28:24 | PSEL[4:0] | Peripheral Select<br>These bits select the peripheral function. For individual pin functions, see the associated tables in this chapter. | R/W*5 |
| 31:29 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The initial value of P108, P109, P110, P201 and P300 is not 0x0000_0000. P108 is 0x0001_0410, P109 is 0x0001_0400, P110 is 0x0001_0010, P201 is 0x0000_0010, and P300 is 0x0001_0010.
Note 2. Supported by PORTn (n = 1 to 4).
Note 3. If the security attribution is configured as Secure:
- Secure access is allowed
- Non-secure read value is 0 and TrustZone access error is not generated
- Non-secure write access is ignored and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 4. If the security attribution is configured as Secure:
- Secure read access is allowed
- Non-secure read value is 0 and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure read access are allowed.

Note 5. If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Port mn Pin Function Select Register (PmnPFS/PmnPFS_HA/PmnPFS_BY) is a 32-bit, 16-bit, or 8-bit read/write control register that selects the port mn pin function, and is accessed in 32-bit units. PmnPFS_HA (PmnPFS [15:0] bits) is accessed in 16-bit units. PmnPFS_BY (PmnPFS[7:0] bits) is accessed in 8-bit units.

The available Port mn pin depends on the product. For details, see Table 19.1

### PODR bit (Port Output Data), PIDR bit (Port State), PDR bit (Port Direction)

The PDR, PIDR, and PODR bits serve the same function as the PCNTR. When these bits are read, the PCNTR value is read.

### PCR bit (Pull-up Control)

The PCR bit enables or disables an input pull-up resistor on the individual port pins. When a pin is in the input state with the associated bit in PmnPFS.PCR set to 1, the pull-up resistor connected to the pin is enabled. When a pin is set as a general port output pin, or a peripheral function output pin, the pull-up resistor for the pin is disabled regardless of the PCR setting. The pull-up resistor is also disabled in the reset state. Bits associated with non-existent pins are reserved. Reserved bits are read as 0. The write value should be 0.

### NCODR bit (N-Channel Open-Drain Control)

The NCODR bit specifies the output type for the port pins. Bits associated with non-existent pins are reserved. Reserved bits are read as 0. The write value should be 0.

### DSCR[1:0] bits (Port Drive Capability)

The DSCR[1:0] bits switches the drive capacity of the port. If the drive capacity of a pin is fixed, the associated bit is a read/write bit, but the drive capacity cannot be changed. Bits associated with non-existent pins are reserved. Reserved bits are read as 0. The write value should be 0.

### EOFR[1:0] bits (Event on Falling/Event on Rising)

The EOFR[1:0] bits select the edge detection method for the port group input signal. These bits support rising, falling, or both edge detections. When the EOFR[1:0] bits are set to 01b, 10b, or 11b, the input enable of the I/O cell is asserted. Following that, the event pulse is input from the external pin, and the GPIO outputs the event pulse to the ELC. Bits associated with non-existent pins are reserved. Reserved bits are read as 0. The write value should be 0.

### ISEL bit (IRQ Input Enable)

The ISEL bit specifies IRQ input pins. This setting can be used in combination with the peripheral functions, although an IRQn (external pin interrupt) of the same number must only be enabled for one pin. The ISEL bit for an unspecified IRQn is reserved.

### ASEL bit (Analog Input Enable)

The ASEL bit specifies analog pins. When a pin is set as an analog pin by this bit:

1. Specify it as a general I/O port in the Port Mode Control bit (PmnPFS.PMR).
2. Disable the pull-up resistor in the Pull-up Control bit (PmnPFS.PCR).
3. Specify the input in the Port Direction bit (PmnPFS.PDR). The pin state cannot be read at this point. The PmnPFS register is protected by the Write-Protect Register (PWPR). Release write-protect before modifying the register.

The ASEL bit for an unspecified analog I/O pin is reserved.

### PMR bit (Port Mode Control)

The PMR bit specifies the port pin function. Bits associated with non-existent pins are reserved. The write value should be 0.

### PSEL[4:0] bits (Peripheral Select)

The PSEL[4:0] bits assign the peripheral function.

## 19.2.6 PWPR : Write-Protect Register

Base address: PFS = 0x4008_0800

Offset address: 0x503

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | B0WI | PFSWE | — | — | — | — | — | — |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | PFSWE | PmnPFS Register Write Enable<br>0: Writing to the PmnPFS register is disabled<br>1: Writing to the PmnPFS register is enabled | R/W |
| 7 | B0WI | PFSWE Bit Write Disable<br>0: Writing to the PFSWE bit is enabled<br>1: Writing to the PFSWE bit is disabled | R/W |

### PFSWE bit (PmnPFS Register Write Enable)

Writing to the PmnPFS register is enabled only when the PFSWE bit is set to 1. You must first write 0 to the B0WI bit before setting PFSWE to 1.

### B0WI bit (PFSWE Bit Write Disable)

Writing to the PFSWE bit is enabled only when the B0WI bit is set to 0.

## 19.2.7 PWPRS : Write-Protect Register for Secure

Base address: PFS = 0x4008_0800

Offset address: 0x505

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | B0WI | PFSWE | — | — | — | — | — | — |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | PFSWE | PmnPFS Register Write Enable<br>0: Disable writes to the PmnPFS register<br>1: Enable writes to the PmnPFS register | R/W |
| 7 | B0WI | PFSWE Bit Write Disable<br>0: Enable writes the PFSWE bit<br>1: Disable writes to the PFSWE bit | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

### PFSWE bit (PmnPFS Register Write Enable)

Writing to the PmnPFS register of the IO port pin set as secure by the PmSAR register is enabled only when the PFSWE bit is set to 1. You must first write 0 to the B0WI bit before setting PFSWE to 1.

### B0WI bit (PFSWE Bit Write Disable)

Writing to the PFSWE bit is enabled only when the B0WI bit is set to 0.

## 19.2.8 PmSAR : Port Security Attribution register (m = 0 to 7)

Base address: PFS = 0x4008_0800

Offset address: 0x510 + 0x002 × m

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PMNSA[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | PMNSA[15:0] | Pmn Security Attribution<br>Target I/O port pin : Pmn<br>0: Secure<br>1: Non Secure | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

Note: m = 0 to 7, n = 00 to 15

Port Security Attribution Register is a 16-bit register that setting the Security Attribution of the each port, the registers are accessed only in 16-bit units.

### PMNSA[15:0] bits (Pmn Security Attribution)

The PmnSA bit specifies the Security Attribution of Pmn.

## 19.3 Operation

### 19.3.1 General I/O Ports

All pins except P108 to P110, and P300 operate as general I/O ports after reset. General I/O ports are organized as 16 bits per port and can be accessed by port with the Port Control Registers (PCNTRn, where n = 1 to 4), or by individual pins with the Port mn Pin Function Select register. For details on these registers, see section 19.2. Register Descriptions.

Each port has the following bits:

- Port Security Attribution register (PmSAR)(m = 0 to 7), which indicates the security attribution.
- Port Direction bit (PDRn), which selects input or output direction
- Port Output Data bit (PODRn), which holds data for output
- Port Input Data bit (PIDRn), which indicates the pin states
- Event Input Data bit (EIDRn), which indicates the pin state when an ELC_PORTn (n = 1, 2, 3 or 4) signal occurs
- Port Output Set bit (POSRn), which indicates the output value when a software write occurs
- Port Output Reset bit (PORRn), which indicates the output value when a software write occurs
- Event Output Set bit (EOSRn), which indicates the output value when an ELC_PORTn (n = 1, 2, 3 or 4) signal occurs
- Event Output Reset bit (EORRn), which indicates the output value when an ELC_PORTn (n = 1, 2, 3 or 4) signal occurs.

### 19.3.2 Port Function Select

The following port functions are available for configuring each pin:

- Security function: Security attribution for each pins
- I/O configuration: CMOS output or NMOS open-drain output, pull-up control, and drive capacity
- General I/O port: Port direction, output data setting, and read input data
- Alternate function: Configured function mapping to the pin.

Each pin is associated with a Port mn Pin Function Select register (PmnPFS), which includes the associated PODR, PIDR, and PDR bits. In addition, the PmnPFS register includes the following:

- PCR: Pull-up resistor control bit that turns the input pull-up MOS on or off
- NCODR: N-channel open-drain control bit that selects the output type for each pin
- DSCR[1:0]: Drive capacity control bit that selects the drive capacity
- EOFR[1:0]: For selecting the edge of the event that input from the port group
- ISEL: IRQ input enable bit to specify an IRQ input pin
- ASEL: Analog input enable bit to specify an analog pin
- PMR: Port mode bit to specify the pin function of each port
- PSEL[4:0]: Port function select bits to select the associated peripheral function.

These configurations can be made by a single-register access to the Port mn Pin Function Select register. For details, see section 19.2.5. PmnPFS/PmnPFS_HA/PmnPFS_BY : Port mn Pin Function Select Register (m = 0 to 7, n = 00 to 15).

### 19.3.3 Port Group Function for ELC

In the MCU, Port 1 to Port 4 are assigned for the ELC port group function.

#### 19.3.3.1 Behavior When ELC_PORTn (n = 1, 2, 3 or 4) is Input from ELC

The MCU supports the two functions described in this section when an ELC_PORTn (n = 1, 2, 3 or 4) signal comes from the ELC.

### (1) Input to EIDR

For the GPI function (PDR = 0 and PMR = 0 in the PmnPFS register), when an ELC_PORTn (n = 1, 2, 3 or 4) signal comes from the ELC, the input enable of the I/O cell is asserted, and data from the external pins is read into the EIDR bit. See Figure 19.2

For the GPO function (PDR = 1) or the peripheral mode (PMR = 1), 0 is input into the EIDR bit from the external pins.

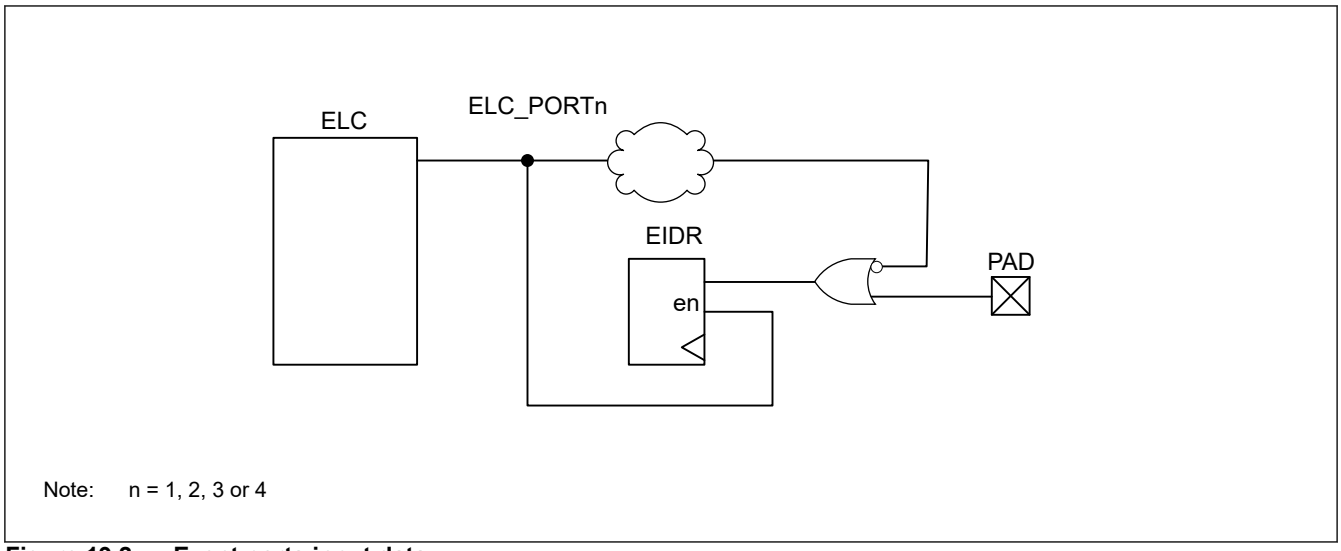



**Figure 19.2 Event ports input data**

### (2) Output from PODR by EOSR and EORR

When an ELC_PORTn (n = 1, 2, 3 or 4) signal occurs, the data is output from the PODR to the external pin based on the settings in the EOSR and EORR registers.

- If EOSR is set to 1, when an ELC_PORTn (n = 1, 2, 3 or 4) signal occurs, the PODR register outputs 1 to the external pin. Otherwise, when EOSR = 0, the PODR value is retained.
- If EORR is set to 1, when ELC_PORTn (n = 1, 2, 3 or 4) signal occurs, the PODR register outputs 0 to the external pin. Otherwise, when EORR = 0, the PODR value is retained.

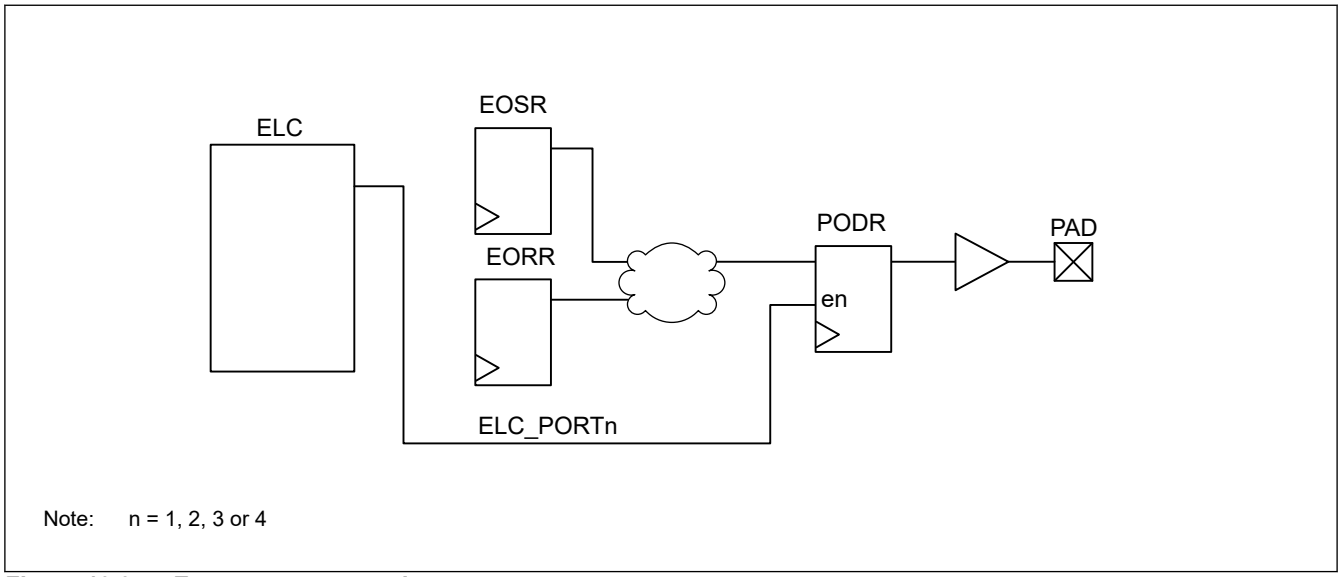



**Figure 19.3 Event ports output data**

#### 19.3.3.2 Behavior When an Event Pulse is Output to ELC

To output the event pulse from the external pins to the ELC, set the EOFR[1:0] bits in the PmnPFS register. For details, see section 19.2.5. PmnPFS/PmnPFS_HA/PmnPFS_BY : Port mn Pin Function Select Register (m = 0 to 7, n = 00 to 15). When the EOFR[1:0] bits are set, the input enable of the I/O cell is asserted.

Data from the external pin is the input. For example, for Port 1, when the data is input from P100 to P115, the data of those 16 pins is organized by OR logic. This data is formed into a one-shot pulse that goes to the ELC. The operation of Port n (n = 2 to 4) is also the same as Port 1. See Figure 19.4.

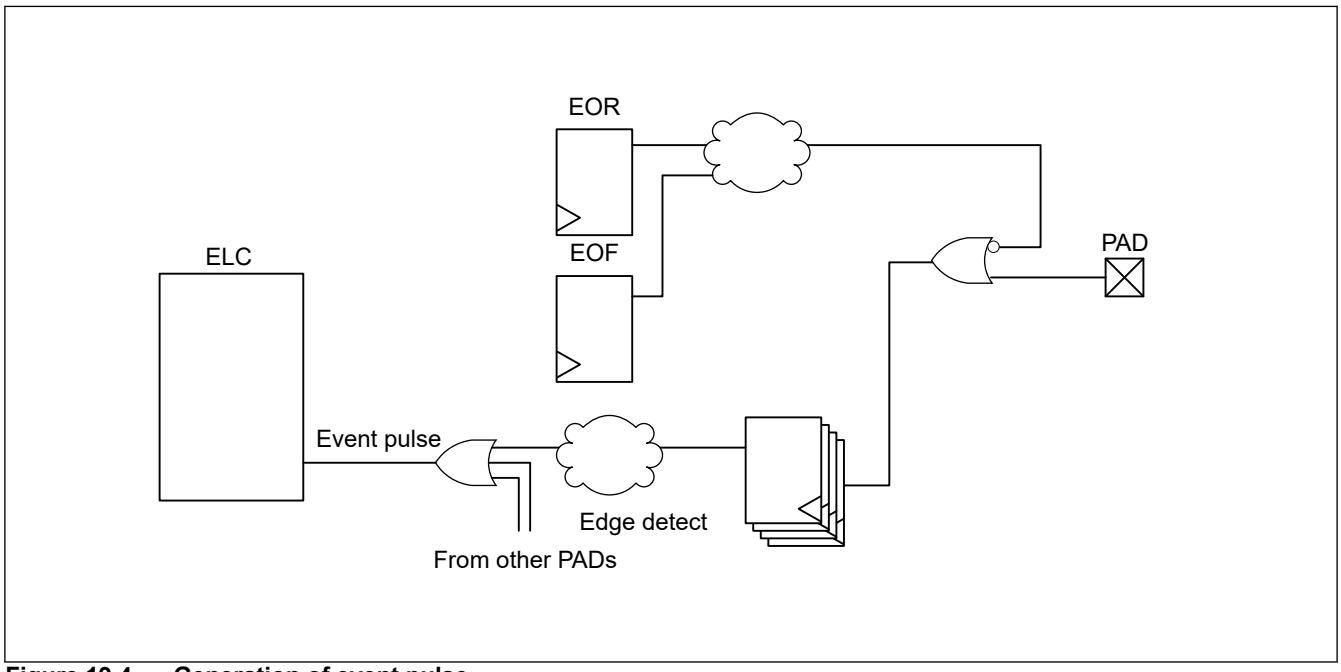



**Figure 19.4 Generation of event pulse**

## 19.4 Handling of Unused Pins

Table 19.3 shows how to handle unused pins.

**Table 19.3 Handling of unused pins (1 of 2)**

| Pin name | Description |
|---|---|
| MD | Use as a mode selection pin |
| RES | Connect to VCC through a resistor (pulling up) |
| USB_DP | Keep pin open |
| USB_DM | Keep pin open |
| P200/NMI | Connect to VCC through a resistor (pulling up) |
| EXTAL | When the main clock oscillator is not used, set the MOSCCR.MOSTP bit to 1 (general port P212). When this pin is not used as port P212, configure it in the same way as ports 1 to 7. |
| XTAL | When the main clock oscillator is not used, set the MOSCCR.MOSTP bit to 1 (general port P213). When the external clock is input to the EXTAL pin, the XTAL pin functions as P213. When this pin is not used as port P213, configure it in the same way as ports 1 to 7. |
| XCIN | Connect to VSS through a resistor (pulling down) |
| XCOUT | Keep pin open |
| P000 to P015 | • If the direction is set to input (PCNTR1.PDRn = 0), connect the associated pin to AVCC0 (pulled up) through a resistor or to AVSS0 (pulled down) through a resistor *1<br>• If the direction is set to output (PCNTR1.PDRn = 1), keep pin open. *1 |

**Table 19.3 Handling of unused pins (2 of 2)**

| Pin name | Description |
|---|---|
| P1x to P7x | • If the direction is set to input (PCNTR1.PDRn = 0), connect the associated pin to VCC (pulled up) through a resistor or to VSS (pulled down) through a resistor. *1 *2<br>• If the direction is set to output (PCNTR1.PDRn = 1), keep pin open. *1 *3 |
| VREFH0, VREFH | Connect to AVCC0 |
| VREFL0 | Connect to AVSS0 |
| VBATT | Connect to VCC or VSS. |

Note 1. Clear the PmnPFS.PMR, PmnPFS.ISEL, PmnPFS.PCR, and PmnPFS.ASEL bits to 0.
Note 2. P108, P110, and P300 should be enabled for input pull-up from the initial value (PmnPFS.PCR = 1).
Note 3. P109 is recommended for setting the direction to output (PCNTR1.PDRn = 1), because this pin is output from the initial value.

## 19.5 Usage Notes

### 19.5.1 Procedure for Specifying the Pin Functions

To specify the I/O pin functions:

1. Clear the B0WI bit in the PWPR register. This enables writing to the PFSWE bit in the PWPR register.*1
2. Set 1 to the PFSWE bit in the PWPR register. This enables writing to the PmnPFS register.*1
3. Clear the Port Mode Control bit in the PMR to 0 for the target pin to select the general I/O port.
4. Specify the I/O function for the pin through the PSEL[4:0] bits settings in the PmnPFS register.
5. Set the PMR bit to 1 as required to switch to the selected I/O function for the pin.
6. Clear the PFSWE bit in the PWPR register. This disables writing to the PmnPFS register.*1
7. Set 1 to the B0WI bit in the PWPR register. This disables writing to the PFSWE bit in the PWPR register.*1

Note 1. When the security attribution of Pmn is set to 0, set the PWPRS register to write to the PmnPFS register.

### 19.5.2 Procedure for Using Port Group Input

To use the port group input (port n (n = 1 to 4)):

1. Set the ELSRx.ELS[8:0] bits to all 0 to ignore unexpected pulses. For more information, see section 18, Event Link Controller (ELC).
2. Set the EOFR[1:0] bits of the PmnPFS register to specify the rising, falling, or both edge detections.
3. Execute a dummy read or wait for a short time, for example 100 ns. Ignoring of unexpected pulses depends on the initial value of the external pin.
4. Set the ELSRx.ELS[8:0] bits to enable the event signals.

### 19.5.3 Port Output Data Register (PODR) Summary

This register outputs data as follows:

1. Outputs 0 if PCNTR4.EORR is set to 1 when ELC_PORTn (n = 1, 2, 3 or 4) signal occurs.
2. Outputs 1 if PCNTR4.EOSR is set to 1 when ELC_PORTn (n = 1, 2, 3 or 4) signal occurs.
3. Outputs 0 if PCNTR3.PORR is set to 1.
4. Outputs 1 if PCNTR3.POSR is set to 1.
5. Outputs 0 or 1 because PCNTR1.PODRn is set.
6. Outputs 0 or 1 because PmnPFS.PODRn is set.

Numbers in this list correspond to the priority for writing to the PODRn. For example, if 1. and 3. from the list occur at the same time, the higher priority event 1. is executed.

### 19.5.4 Notes on Using Analog Functions

To use an analog function, set the Port Mode Control bit (PMR) and the Port Direction bit (PDRn) to 0 so that the pin acts as a general input port. Next, set the Analog Input Enable bit (ASEL) in the Port mn Pin Function Select Register (PmnPFS.ASEL) to 1.

### 19.5.5 I/O Buffer Specification

The P402, P403, and P404 can be used as the RTC input, AGT input and other peripheral functions. Table 19.4 lists the P402, P403, P404 specifications.

**Table 19.4 P402, P403, P404 specifications**

| I/O port | RTC and AGT | | | Other peripheral | |
|---|---|---|---|---|---|
| | RTC and AGT input enable register | RTC | AGT | other peripheral enable register | CAC, GPT, CAN, SCI, SSIE and interrupt |
| P402 | VBTICTLR.VCH0INEN | RTCIC0 | AGTIO0<br>AGTIO1<br>AGTIO2<br>AGTIO3 | P402PFS.PSEL and PMR | For details, see section 19.6. Peripheral Select Settings for Each Product. |
| P403 | VBTICTLR.VCH1INEN | RTCIC1 | AGTIO0<br>AGTIO1<br>AGTIO2<br>AGTIO3 | P403PFS.PSEL and PMR | |
| P404 | VBTICTLR .VCH2INEN | RTCIC2 | AGTIO0<br>AGTIO1<br>AGTIO2<br>AGTIO3 | P404PFS.PSEL and PMR | |

These RTC and AGT inputs are controlled by the VBTICTLR register and this register has the highest priority for selecting the functions.

P402, P403, and P404 can be used as IRQn-DS (n = 4, 14, 15) whether RTC and AGT inputs are selected or not. When using these interrupts, set the interrupt procedure after setting the VBTICTLR register. (see section 11.2.6. VBTICTLR : VBATT Input Control Register.)

See Figure 19.5.

The VBTICTLR register is not initialized on reset. Therefore, when not using the RTC or AGT inputs, the associated bit of VBTICTLR register must be set to 0 after reset.

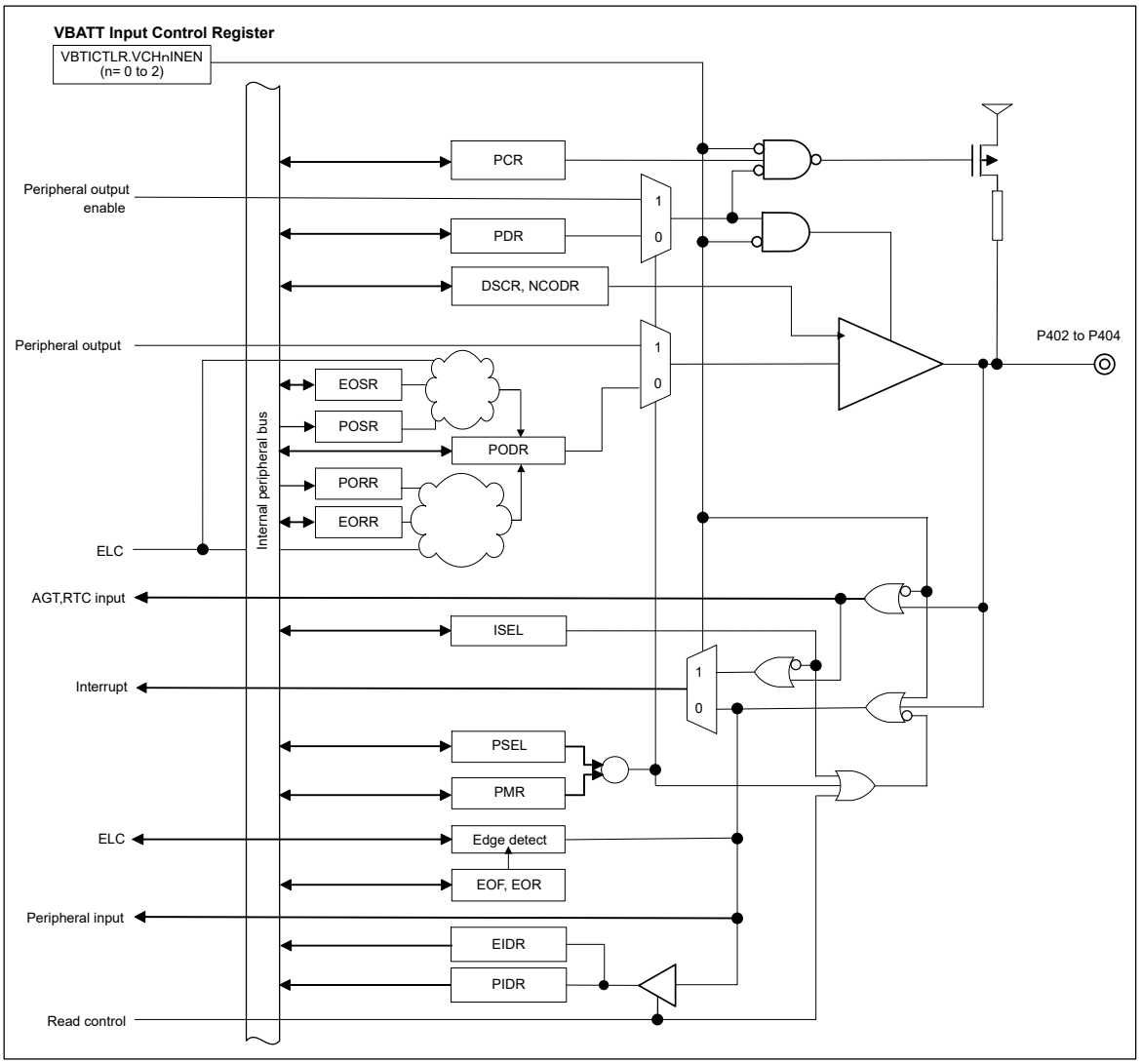



**Figure 19.5 P402, P403, P404 diagram**

## 19.6 Peripheral Select Settings for Each Product

This section describes the pin function select configuration using the PmnPFS register. Some pin names have added _A, _B, or _C suffixes. When assigning IIC, SSIE, and SDHI functionality, select the functional pins having the same suffix. The other pins can be selected regardless of the suffix. Assigning the same function to two or more pins simultaneously is prohibited.

1. In Pmn pin function select register(PmnPFS), the PSEL bits have to be set when the PMR bit of the target pin is 0. If the PSEL bits are set when the PMR bit is 1, the unexpected edges may be input at the input function or the unexpected pulses may be output to the external pin at the output function.
2. Only the allowed values (functions) should be specified in the PSEL bits of PmnPFS register. If a value which is not allowed for the register is specified, the correct operation is not guaranteed.
3. The single function should not be assigned to the multiple pins by PmnPFS register. When the GPT1, GPT5, SCI3, IIC0 or SPI0 are configured as secure and these pin function is being assigned to the pin which security attribution is set as secure by the PmSAR register, the write access to the PSEL bits for setting same function as secure pin in other pins is ignored when the security attribution of that pin is non-secure. For example, if the PSARE.PSARE30 bit is 0 (GPT1 is secure) and the P109PFS.PSEL bits is 00011b (pin function is GTIOC1A) and the P1SAR.109SA bit is 0 (P109 is secure), the write 00011b to the P405PFS.PSEL bits is ignored when the P4SAR.405SA bit is 1 (P405 is non-secure).

4. The PORT0 and 5 have the analog functions such as A/D converter. When these pins are used as an analog function, for avoiding the loss of resolution, the PMR bit should be set to 0 and PDR bit should be set to 0. After that, ASEL bit should be set to 1

**Table 19.5 Register settings for input/output pin function (PORT0)**

| PSEL[4:0] settings | Function | pin | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P000 | P001 | P002 | P003 | P004 | P005 | P006 | P007 | P008 | P013 | P014 | P015 |
| 00000b (value after reset) | Hi-Z/ JTAG/SWD | Hi-Z | | | | | | | | | | | |
| ASEL bit | | AN000 | AN001 | AN002 | AN003 | AN004 | AN005 | AN006 | AN007 | AN008 | AN011 | AN012/DA0 | AN013/DA1 |
| ISEL bit | | IRQ6-DS | IRQ7-DS | IRQ8-DS | — | IRQ9-DS | IRQ10-DS | IRQ11-DS | — | IRQ12-DS | — | — | IRQ13 |
| DSCR[1:0] bits | Drive capacity control*1 | L | L | L | L | L | L | L | L | L | L | L | L |
| NCODR bit | N-ch open-drain | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| PCR bit | Pull-up | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 100 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 64 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — | — | ✓ | ✓ | ✓ |
| 48 pins product | | ✓ | ✓ | ✓ | — | — | — | — | — | — | ✓ | ✓ | ✓ |

✓: Available
—: Setting prohibited

Note 1. The drive strength of this port cannot be controlled by PmnPFS.DSCR[1:0] bits.

**Table 19.6 Register settings for input/output pin function (PORT1) (1 of 2)**

| PSEL[4:0] settings | Function | pin | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P100 | P101 | P102 | P103 | P104 | P105 | P106 | P107 | P108 | P109 | P110 | P111 | P112 | P113 | P114 | P115 |
| 00000b (value after reset) | Hi-Z/ JTAG/SWD | Hi-Z | | | | | | | | TMS/ SWDIO | TDO/SWO | TDI | Hi-Z | | | | |
| 00001b | AGT | AGTIO0 | AGTEE0 | AGTO0 | AGTIO2 | AGTEE2 | AGTO2 | AGTOB0 | AGTOA0 | AGTOA3 | AGTOB3 | AGTEE3 | AGTOA5 | AGTOB5 | AGTEE5 | AGTIO5 | — |
| 00010b | GPT*2 | GTETRGA | GTETRGB | GTOWLO | GTOWUP | GTETRGB | GTETRGA | — | — | GTOULO | GTOVUP | GTOVLO | — | — | — | — | — |
| 00011b | GPT*2 | GTIOC5B | GTIOC5A | GTIOC2B | GTIOC2A | GTIOC1B | GTIOC1A | — | — | GTIOC0B | GTIOC1A | GTIOC1B | GTIOC3A | GTIOC3B | GTIOC2A | GTIOC2B | GTIOC4A |
| 00100b | SCI | RXD0/ MISO0/ SCL0 | TXD0/ MOSI0/ SDA0 | SCK0 | CTS0_ RTS0/ SS0 | — | — | — | — | — | — | CTS2_ RTS2/ SS2 | SCK2 | TXD2/ MOSI2/ SDA2/ TXDX2/ SIOX2 | RXD2/ MISO2/ SCL2/ RXDX2 | — | — |
| 00101b | SCI | SCK1 | CTS1_ RTS1/ SS1 | — | — | — | — | — | — | CTS9_ RTS9/ SS9 | TXD9/ MOSI9/ SDA9 | RXD9/ MISO9/ SCL9 | SCK9 | SCK1 | — | CTS9 | — |
| 00110b | SPI | — | — | — | — | — | — | — | — | SSLA0 | MOSIA | MISOA | RSPCKA | SSLA0 | — | — | — |
| 01001b | CLKOUT/ RTC | — | — | — | — | — | — | — | — | — | CLKOUT | — | — | — | — | — | — |
| 01010b | CAC/ ADC12 | — | — | ADTRG0 | — | — | — | — | — | — | — | — | — | — | — | — | — |
| 10000b | CAN | — | — | CRX0 | CTX0 | — | — | — | — | — | — | — | — | — | — | — | — |
| 10001b | QSPI | QSPCLK | QIO1 | QIO0 | QIO3 | QIO2 | — | — | — | — | — | — | — | QSSL | — | — | — |
| 10010b | SSIE*1 | — | — | — | — | — | — | — | — | — | — | — | — | SSIBCK0_B | SSILRCK0/SSIFS0_B | SSIRXD0_B | SSITXD0_B |
| ASEL bit | | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| ISEL bit | | IRQ2 | IRQ1 | — | — | IRQ1 | IRQ0 | — | — | — | — | IRQ3 | IRQ4 | — | — | — | — |
| DSCR[1:0] bits | Drive capacity control | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H |
| NCODR bit | N-ch open-drain | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| PCR bit | Pull-up | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 100 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 64 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | — | — |

**Table 19.6 Register settings for input/output pin function (PORT1) (2 of 2)**

| PSEL[4:0] settings | Function | pin | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P100 | P101 | P102 | P103 | P104 | P105 | P106 | P107 | P108 | P109 | P110 | P111 | P112 | P113 | P114 | P115 |
| 48 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — |

✓: Available
—: Setting prohibited

Note 1. Recommend using pins that have a letter appended to their names, for instance "_A" or "_B", to indicate group membership. For the interface, the AC portion of the electrical characteristics is measured for each group.

Note 2. There are 2 types output buffer which are middle drive and high drive. Recommend using same drive buffer for output skew spec ($t_{GTISK}$)

**Table 19.7 Register settings for input/output pin function (PORT2)**

| PSEL[4:0] settings | Function | Pin | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P200*4 | P201 | P205 | P206 | P207 | P208 | P209 | P210 | P211 | P212 | P213 | P214 |
| 00000b (value after reset) | Hi-Z/ JTAG/SWD | Hi-Z | | | | | | | | | | | |
| 00001b | AGT | — | — | AGTO1 | — | — | — | AGTEE5 | AGTOB5 | AGTOA5 | AGTEE1 | AGTEE2 | AGTO5 |
| 00010b | GPT*2 | — | — | GTIV | GTIU | — | GTOVLO | GTOVUP | GTIW | GTIV | GTETRGD | GTETRGC | GTIU |
| 00011b | GPT*2 | — | — | GTIOC4A | — | — | — | — | — | — | GTIOC0B | GTIOC0A | — |
| 00100b | SCI | — | — | TXD4/ MOSI4/ SDA4 | RXD4/ MISO4/ SCL4 | TXD4/ MOSI4/ SDA4 | — | — | — | — | — | — | — |
| 00101b | SCI | — | — | CTS9_RTS9/ SS9 | CTS9 | — | — | — | — | — | RXD1/ MISO1/ SCL1/ RXDX1 | TXD1/ MOSI1/ SDA1/ TXDX1/ SIOX1 | — |
| 00111b | IIC*1 | — | — | SCL1_B | SDA1_B | — | — | — | — | — | — | — | — |
| 01001b | CLKOUT/ RTC | — | — | CLKOUT | — | — | — | — | — | — | — | — | — |
| 01100b | CTSU | — | — | TS01 | TS02 | TSCAP | — | — | — | — | — | — | — |
| 10001b | QSPI | — | — | — | — | QSSL | QIO3 | QIO2 | QIO1 | QIO0 | — | — | QSPCLK |
| 10011b | USBFS | — | — | USB_OVRCUR A-DS | USB_VBUSEN | — | — | — | — | — | — | — | — |
| 10101b | SDHI*1 | — | — | SD0DAT3_A | SD0DAT2_A | — | SD0DAT0_B | SD0WP | SD0CD | SD0CMD_B | — | — | SD0CLK_B |
| 11010b | Trace (Debug) | — | — | — | — | — | TDATA3 | TDATA2 | TDATA1 | TDATA0 | — | — | TCLK |
| ASEL bit | | — | — | — | — | — | — | — | — | — | — | — | — |
| ISEL bit | | — | — | IRQ1-DS | IRQ0-DS | — | — | — | — | — | IRQ3 | IRQ2 | — |
| DSCR[1:0] bits | Drive capacity control | — | L*3 | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H |
| NCODR bit | N-ch open-drain | — | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| PCR bit | Pull-up | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 100 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 64 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — | ✓ | ✓ | — |
| 48 pins product | | ✓ | ✓ | — | ✓ | ✓ | — | — | — | — | ✓ | ✓ | — |

✓: Available
—: Setting prohibited

Note 1. Recommend using pins that have a letter appended to their names, for instance "_A" or "_B", to indicate group membership. For the interface, the AC portion of the electrical characteristics is measured for each group.

Note 2. There are 2 types output buffer which are middle drive and high drive. Recommend using same drive buffer for output skew spec ($t_{GTISK}$)

Note 3. The driver strength of this port can not be controlled by PmnPFS.DSCR[1:0] bits.

Note 4. When using NMI pin interrupt, Port related registers setting are not required.

**Table 19.8 Register settings for input/output pin function (PORT3) (1 of 2)**

| PSEL[4:0] settings | Function | Pin | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | P300 | P301 | P302 | P303 | P304 | P305 | P306 | P307 |
| 00000b (value after reset) | Hi-Z/JTAG/SWD | TCK/SWCLK | Hi-Z | | | | | | |
| 00001b | AGT | — | AGTIO0 | — | — | AGTEE2 | AGTOB2 | AGTOA2 | AGTEE4 |
| 00010b | GPT*1 | GTOUUP | GTOULO | GTOUUP | — | GTOWLO | GTOWUP | GTOULO | GTOUUP |

**Table 19.8 Register settings for input/output pin function (PORT3) (2 of 2)**

| PSEL[4:0] settings | Function | Pin | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | P300 | P301 | P302 | P303 | P304 | P305 | P306 | P307 |
| 00011b | GPT*1 | GTIOC0A | GTIOC4B | GTIOC4A | GTIOC7B | GTIOC7A | — | — | — |
| 00100b | SCI | — | RXD2/MISO2/ SCL2/RXDX2 | TXD2/MOSI2/ SDA2/TXDX2/ SIOX2 | — | — | — | — | — |
| 00101b | SCI | — | CTS9_RTS9/ SS9 | — | CTS9 | — | — | — | — |
| 00110b | SPI | SSLA1 | SSLA2 | SSLA3 | — | — | — | — | — |
| 10001b | QSPI | — | — | — | — | — | QSPCLK | QSSL | QIO0 |
| ASEL bit | | — | — | — | — | — | — | — | — |
| ISEL bit | | — | IRQ6 | IRQ5 | — | IRQ9 | IRQ8 | — | — |
| DSCR[1:0] bits | Drive capacity control | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H |
| NCODR bit | N-ch open-drain | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| PCR bit | Pull-up | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 100 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 64 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — |
| 48 pins product | | ✓ | ✓ | ✓ | — | — | — | — | — |

✓: Available
—: Setting prohibited

Note 1. There are 2 types output buffer which are middle drive and high drive. Recommend using same drive buffer for output skew spec ($t_{GTISK}$)

**Table 19.9 Register settings for input/output pin function (PORT4) (1 of 2)**

| PSEL[4:0] settings | Function | pin | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P400 | P401 | P402 | P403 | P404 | P405 | P406 | P407 | P408 | P409 | P410 | P411 | P412 | P413 | P414 | P415 |
| 00000b (value after reset) | Hi-Z/ JTAG/SW D | Hi-Z | | | | | | | | | | | | | | | |
| 00001b | AGT | AGTIO 1 | — | — | — | — | — | AGTO5 | AGTIO 0 | AGTOB 2 | AGTOA 2 | AGTOB 1 | AGTOA 1 | AGTEE 1 | AGTEE 3 | AGTIO 5 | AGTIO 4 |
| 00010b | GPT*3 | — | GTETR GA | — | — | — | — | — | — | GTOW LO | GTOW UP | GTOVL O | GTOVU P | GTOUL O | GTOUU P | — | — |
| 00011b | GPT*3 | GTIOC 6A | GTIOC 6B | — | GTIOC 3A | GTIOC 3B | GTIOC 1A | GTIOC 1B | GTIOC 6A | GTIOC 6B | — | — | — | — | — | GTIOC 0B | GTIOC 0A |
| 00100b | SCI | SCK4 | CTS4_ RTS4/ SS4 | CTS4 | — | — | — | — | CTS4_ RTS4/ SS4 | CTS4 | — | RXD0/ MISO0/ SCL0 | TXD0/ MOSI0/ SDA0 | SCK0 | CTS0_ RTS0/ SS0 | CTS0 | — |
| 00101b | SCI | — | — | — | — | — | — | — | — | RXD3/ MISO3/ SCL3 | TXD3/ MOSI3/ SDA3 | SCK3 | CTS3_ RTS3/ SS3 | CTS3 | — | — | — |
| 00111b | IIC*2 | SCL0_ A | SDA0_ A | — | — | — | — | — | SDA0_ B | SCL0_ B | — | — | — | — | — | — | — |
| 01001b | CLKOUT/ RTC | — | — | — | — | — | — | — | RTCOU T | — | — | — | — | — | — | — | — |
| 01010b | CAC/ ADC12 | — | — | CACRE F | — | — | — | — | ADTRG 0 | — | — | — | — | — | — | — | — |
| 01100b | CTSU | — | — | — | — | — | — | — | TS03 | TS04 | TS05 | TS06 | TS07 | TS08 | TS09 | TS10 | TS11 |
| 10000b | CAN | — | CTX0 | CRX0 | — | — | — | — | — | — | — | — | — | — | — | — | — |
| 10010b | SSIE*2 | AUDIO _CLK | — | AUDIO _CLK | SSIBC K0_A | SSILRC K0/SSI FS0_A | SSITXD 0_A | SSIRX D0_A | — | — | — | — | — | — | — | — | — |
| 10011b | USBFS | — | — | — | — | — | — | — | USB_V BUS | USB_ID | USB_E XICEN | — | — | — | — | — | USB_V BUSEN |
| 10101b | SDHI*2 | — | — | — | — | — | — | — | — | — | — | SD0DA T1_A | SD0DA T0_A | SD0CM D_A | SD0CL K_A | SD0WP | SD0CD |
| Don't-care | AGT, RTC | — | — | AGTIO 0*1/ AGTIO 1*1/ AGTIO 2*1/ AGTIO 3*1/ RTCIC0 *1 | AGTIO 0*1/ AGTIO 1*1/ AGTIO 2*1/ AGTIO 3*1/ RTCIC1 *1 | AGTIO 0*1/ AGTIO 1*1/ AGTIO 2*1/ AGTIO 3*1/ RTCIC2 *1 | — | — | — | — | — | — | — | — | — | — | — |

**Table 19.9 Register settings for input/output pin function (PORT4) (2 of 2)**

| PSEL[4:0] settings | Function | pin | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P400 | P401 | P402 | P403 | P404 | P405 | P406 | P407 | P408 | P409 | P410 | P411 | P412 | P413 | P414 | P415 |
| ASEL bit | | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| ISEL bit | | IRQ0 | IRQ5-DS | IRQ4-DS | IRQ14-DS | IRQ15-DS | — | — | — | IRQ7 | IRQ6 | IRQ5 | IRQ4 | — | — | IRQ9 | IRQ8 |
| DSCR[1:0] bits | Drive capacity control | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H |
| NCODR bit | N-ch open-drain | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| PCR bit | Pull-up | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 100 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 64 pins product | | ✓ | ✓ | ✓ | — | — | — | — | ✓ | ✓ | ✓ | ✓ | ✓ | — | — | — | — |
| 48 pins product | | — | — | ✓ | — | — | — | — | ✓ | ✓ | ✓ | — | — | — | — | — | — |

✓: Available
—: Setting prohibited

Note 1. To use this pin function, set the associated pin as a general input (set the PmnPFS.PDR and PmnPFS.PMR bits to 0).
Note 2. Recommend using pins that have a letter appended to their names, for instance "_A" or "_B", to indicate group membership. For the interface, the AC portion of the electrical characteristics is measured for each group.
Note 3. There are 2 types output buffer which are middle drive and high drive. Recommend using same drive buffer for output skew spec ($t_{GTISK}$)

**Table 19.10 Register settings for input/output pin function (PORT5)**

| PSEL[4:0] settings | Function | pin | | | | | |
|---|---|---|---|---|---|---|---|
| | | P500 | P501 | P502 | P503 | P504 | P505 |
| 00000b (value after reset) | Hi-Z/JTAG/SWD | Hi-Z | | | | | |
| 00001b | AGT | AGTOA0 | AGTOB0 | AGTOA2 | AGTOB2 | AGTOA3 | AGTOB3 |
| 00010b | GPT*1 | GTIU | GTIV | GTIW | GTETRGC | GTETRGD | — |
| 01010b | CAC/ADC12 | CACREF | — | — | — | — | — |
| 10001b | QSPI | QSPCLK | QSSL | QIO0 | QIO1 | QIO2 | QIO3 |
| 10011b | USBFS | USB_VBUSEN | USB_OVRCURA | USB_OVRCURB | USB_EXICEN | USB_ID | — |
| ASEL bit | | AN016 | — | — | — | — | — |
| ISEL bit | | — | IRQ11 | IRQ12 | — | — | IRQ14 |
| DSCR[1:0] bits | Drive capacity control | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H |
| NCODR bit | N-ch open-drain | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| PCR bit | Pull-up | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 100 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 64 pins product | | ✓ | — | — | — | — | — |
| 48 pins product | | ✓ | — | — | — | — | — |

✓: Available
—: Setting prohibited

Note 1. There are 2 types output buffer which are middle drive and high drive. Recommend using same drive buffer for output skew spec ($t_{GTISK}$)

**Table 19.11 Register settings for input/output pin function (PORT6) (1 of 2)**

| PSEL[4:0] settings | Function | Pin | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | P600 | P601 | P602 | P603 | P608 | P609 | P610 |
| 00000b (value after reset) | Hi-Z/JTAG/SWD | Hi-Z | | | | | | |
| 00001b | AGT | AGTIO3 | AGTEE3 | AGTO3 | AGTIO4 | — | AGTO5 | AGTO4 |
| 00011b | GPT | GTIOC6B | GTIOC6A | GTIOC7B | GTIOC7A | GTIOC4B | GTIOC5A | GTIOC5B |
| 00101b | SCI | SCK9 | RXD9/MISO9/SCL9 | TXD9/MOSI9/SDA9 | CTS9_RTS9/SS9 | — | — | — |
| 01001b | CLKOUT/RTC | CLKOUT | — | — | — | — | — | — |
| 01010b | CAC/ADC12 | CACREF | — | — | — | — | — | — |
| ASEL bit | | — | — | — | — | — | — | — |
| ISEL bit | | — | — | — | — | — | — | — |
| DSCR[1:0] bits | Drive capacity control | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H | L/M/H |

**Table 19.11 Register settings for input/output pin function (PORT6) (2 of 2)**

| PSEL[4:0] settings | Function | Pin | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | P600 | P601 | P602 | P603 | P608 | P609 | P610 |
| NCODR bit | N-ch open-drain | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| PCR bit | Pull-up | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 100 pins product | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 64 pins product | | — | — | — | — | — | — | — |
| 48 pins product | | — | — | — | — | — | — | — |

✓: Available
—: Setting prohibited

**Table 19.12 Register settings for input/output pin function (PORT7)**

| PSEL[4:0] settings | Function | Pin |
|---|---|---|
| | | P708 |
| 00000b (value after reset) | Hi-Z/JTAG/SWD | Hi-Z |
| 00101b | SCI | RXD1/MISO1/SCL1/RXDX1 |
| 01010b | CAC/ADC12 | CACREF |
| 01100b | CTSU | TS12 |
| 10010b | SSIE | AUDIO_CLK |
| ASEL bit | | — |
| ISEL bit | | IRQ11 |
| DSCR[1:0] bits | Drive capacity control | L/M/H |
| NCODR bit | N-ch open-drain | ✓ |
| PCR bit | Pull-up | ✓ |
| 100 pins product | | ✓ |
| 64 pins product | | — |
| 48 pins product | | — |

✓: Available
—: Setting prohibited

# 20. Port Output Enable for GPT (POEG)

## 20.1 Overview

The Port Output Enable (POEG) function can place the General PWM Timer (GPT) output pins in the output disable state in one of the following ways:

- Input level detection of the GTETRGn (n = A to D) pins
- Output-disable request from the GPT
- Oscillation stop detection of the clock generation circuit
- Register settings

The GTETRGn (n = A to D) pins can be used as GPT external trigger input pins.

Table 20.1 lists the POEG specifications, Figure 20.1 shows a block diagram, and Table 20.2 lists the input pins.

**Table 20.1 POEG specifications**

| Parameter | Specifications |
|---|---|
| Output-disable control through input level detection | • The GPT output pins can be disabled when a GTETRGn rising edge or high level is sampled after polarity and filter selection. |
| Output-disable request from the GPT | • When the GTIOCxA pin and the GTIOCxB pin are driven to an active level simultaneously, the GPT generates an output-disable request to the POEG. Through reception of these requests, the POEG can control whether the GTIOCxA and GTIOCxB pins are output-disabled. |
| Output-disable control through oscillation stop detection | • The GPT output pins can be disabled when oscillation of the clock generation circuit stops. |
| Output-disable control by software (registers) | • The GPT output pins can be disabled by modifying the register settings. |
| Interrupt | • Interrupts can be generated by detecting the input level of external trigger input pins (GTETRGn pins).<br>• Interrupts can be generated when all GPT output pins are driven to an active level simultaneously. |
| External trigger output to the GPT | • The GTETRGn signals can be output to the GPT after polarity and filter selection. (count start, count stop, count clear, up-count, down-count, or input capture function) |
| Noise filtering | • For input from the GTETRGn pins, PCLKB/1, PCLKB/8, PCLKB/32, or PCLKB/128 can be selected as the noise filtering clock. (Filtering is performed by sampling the input signals three times using the selected clock.)<br>• Positive or negative polarity can be selected for any of the GTETRGn input pins.<br>• Signal state after polarity and filter selection can be monitored. |
| TrustZone Filter | • Security attribution can be set for each groups. |

Note: n = A to D, x = 0 to 7

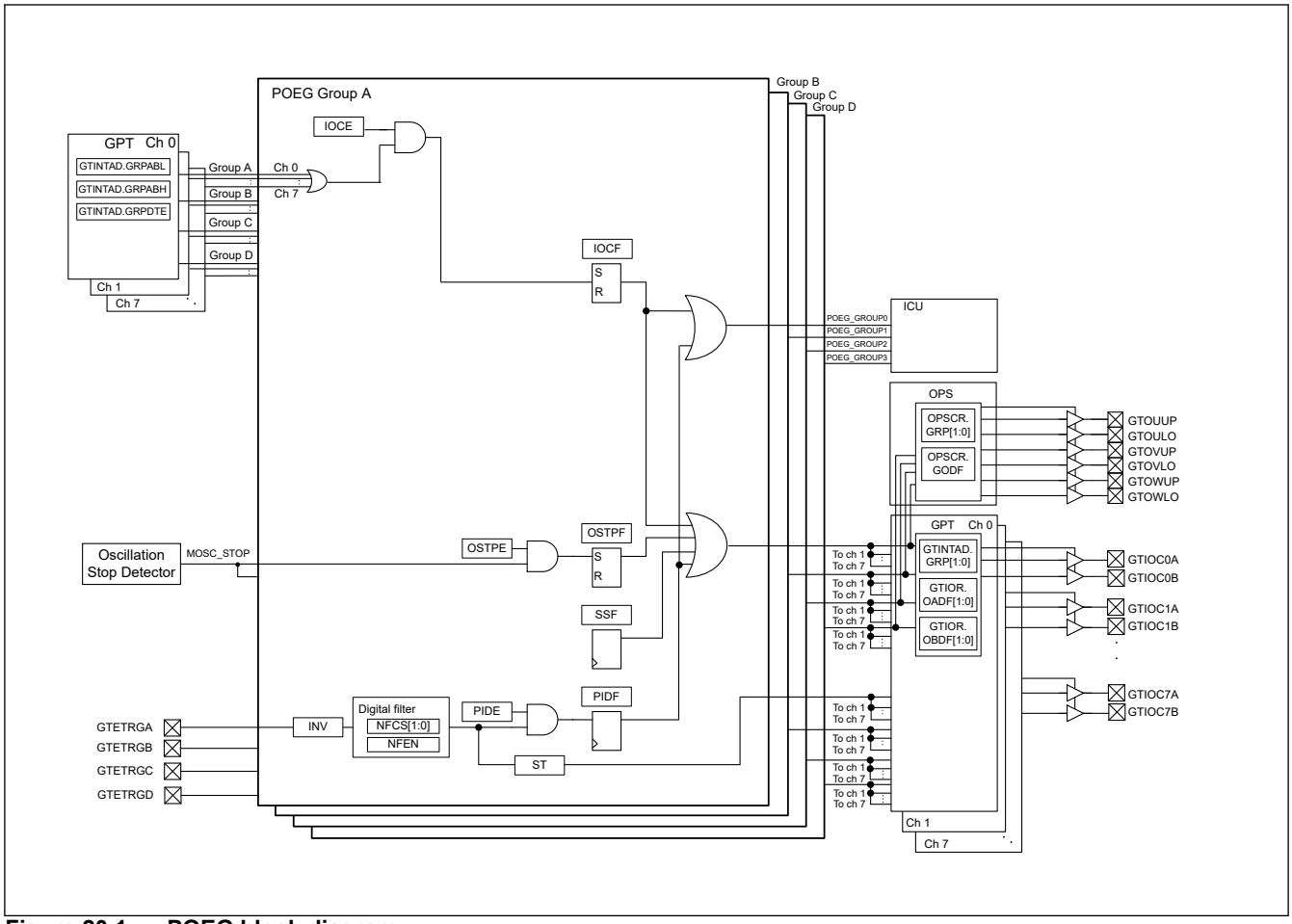



**Figure 20.1 POEG block diagram**

**Table 20.2 POEG input pins**

| Pin name | I/O | Description |
|---|---|---|
| GTETRGA | Input | GPT output pin output-disable request signal or GPT external trigger input pin A |
| GTETRGB | Input | GPT output pin output-disable request signal or GPT external trigger input pin B |
| GTETRGC | Input | GPT output pin output-disable request signal or GPT external trigger input pin C |
| GTETRGD | Input | GPT output pin output-disable request signal or GPT external trigger input pin D |

## 20.2 Register Descriptions

### 20.2.1 POEGGn : POEG Group n Setting Register (n = A to D)

Base address: POEG = 0x4008_A000

Offset address: 0x000 (POEGGA)
0x100 (POEGGB)
0x200 (POEGGC)
0x300 (POEGGD)

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | NFCS[1:0] | | NFEN | INV | — | — | — | — | — | — | — | — | — | — | — | ST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | OSTPE | IOCE | PIDE | SSF | OSTPF | IOCF | PIDF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PIDF | Port Input Detection Flag<br>0: No output-disable request from the GTETRGn pin occurred<br>1: Output-disable request from the GTETRGn pin occurred. | R/W*1 |
| 1 | IOCF | Detection Flag for GPT Output-Disable Request<br>0: No output-disable request from GPT occurred.<br>1: Output-disable request from GPT occurred. | R/W*1 |
| 2 | OSTPF | Oscillation Stop Detection Flag<br>0: No output-disable request from oscillation stop detection occurred<br>1: Output-disable request from oscillation stop detection occurred | R/W*1 |
| 3 | SSF | Software Stop Flag<br>0: No output-disable request from software occurred<br>1: Output-disable request from software occurred | R/W |
| 4 | PIDE | Port Input Detection Enable<br>0: Disable output-disable requests from the GTETRGn pins<br>1: Enable output-disable requests from the GTETRGn pins | R/W*2 |
| 5 | IOCE | Enable for GPT Output-Disable Request<br>0: Disable output-disable requests from GPT<br>1: Enable output-disable requests from GPT | R/W*2 |
| 6 | OSTPE | Oscillation Stop Detection Enable<br>0: Disable output-disable requests from oscillation stop detection<br>1: Enable output-disable requests from oscillation stop detection | R/W*2 |
| 15:7 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | ST | GTETRGn Input Status Flag<br>0: GTETRGn input after filtering was 0<br>1: GTETRGn input after filtering was 1 | R |
| 27:17 | — | These bits are read as 0. The write value should be 0. | R/W |
| 28 | INV | GTETRGn Input Reverse<br>0: Input GTETRGn as-is<br>1: Input GTETRGn in reverse | R/W |
| 29 | NFEN | Noise Filter Enable<br>0: Disable noise filtering<br>1: Enable noise filtering | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:30 | NFCS[1:0] | Noise Filter Clock Select<br>0 0: Sample GTETRGn pin input level three times every PCLKB<br>0 1: Sample GTETRGn pin input level three times every PCLKB/8<br>1 0: Sample GTETRGn pin input level three times every PCLKB/32<br>1 1: Sample GTETRGn pin input level three times every PCLKB/128 | R/W |

Note 1. Only 0 can be written to clear the flag.
Note 2. Can be modified only once after a reset.

The POEGGn (n = A to D) registers control the output-disable state of the GPT pins, interrupts, and the external trigger input to the GPT.

In the descriptions, POEGGn represents the POEGGn (n = A to D) registers.

## 20.3 Output-Disable Control Operation

If any of the following conditions is satisfied, the GTIOCxA, GTIOCxB, and the 3-phase PWM output for BLDC motor control pins can be set to output-disable:

- Input level or edge detection of the GTETRGn pins
  When POEGGn.PIDE is 1, the POEGGn.PIDF flag is set to 1.
- Output-disable request from the GPT
  When POEGGn.IOCE is 1, the POEGGn.IOCF flag is set to 1 if the disable request is enabled by GTINTAD. The GTINTAD.GRPABH and GTINTAD.GRPABL settings apply to the group selected by the GPT register GTINTAD.GRP[1:0] or OPSCR.GRP[1:0].
- Oscillation stop detection for the clock generation circuit
  While POEGGn.OSTPE is 1, the halt status of the main clock oscillator is detected and the POEGGn. OSTPF flag is set to 1.
- SSF bit setting
  When POEGGn.SSF is set to 1, the GPT and PWM output are disabled.

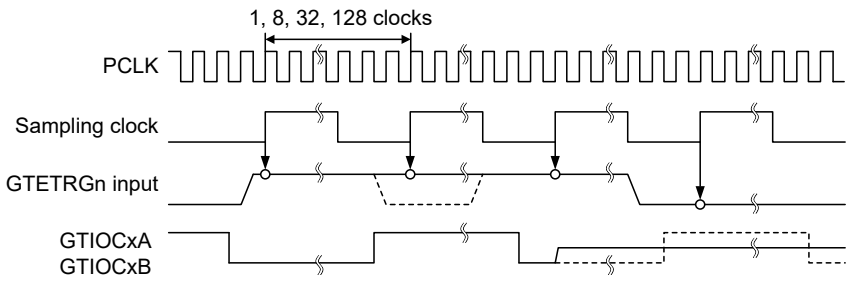
The output-disable state is controlled in the GPT module. The output-disable of the GTIOCxA and GTIOCxB pins is set in the GTINTAD.GRP[1:0], GTIOR.OADF[1:0], and GTIOR.OBDF[1:0] bits in GPTx. The output-disable of the 3-phase PWM output for BLDC motor control pins is set in the OPSCR.GRP[1:0] bits and OPSCR.GODF bit in GPT_OPS.

### 20.3.1 Pin Input Level Detection Operation

If the input conditions set in POEGGn.PIDE, POEGGn.NFCS[1:0], POEGGn.NFEN, and POEGGn.INV occur on the GTETRGn pins, the GPT output pins are output-disabled.

#### 20.3.1.1 Digital Filter

Figure 20.2 shows high-level detection by the digital filter. When a high level associated with the POEGGn.INV polarity setting is detected three times consecutively with the sampling clock selected in POEGGn.NFCS[1:0], the detected level is recognized as high, and the GPT output pins are output-disabled. If even one low level is detected during this interval, the detected level is not recognized as high. In addition, in an interval where the sampling clock is not output, changes of the levels on the GTETRGn pins are ignored.




| | | | | |
|---|---|---|---|---|
| When high level is sampled at all points | [1] | [2] | [3] | Flag set (GTETRGn received) |
| When low level is sampled at least once | [1] | [0] | [1] | Flag not set |

Note: PCLK = PCLKB
Note: Each channel output can be set in the GPT setting. Low level sampling can be set in the POEGGn.INV setting.

**Figure 20.2 Example of digital filter operation**

### 20.3.2 Output-Disable Requests from the GPT

For details on the operation, see the description for GTIOC Pin Output Negate Control in section 21, General PWM Timer (GPT).

### 20.3.3 Output-Disable Control Using Detection of Stopped Oscillation

When the oscillation stop detection function in the clock generation circuit detects stopped oscillation while POEGGn.OSTPE is 1, the GPT output pins are output-disabled for each group.

### 20.3.4 Output-Disable Control Using Registers

The GPT output pins can be directly controlled by writing 1 to the Software Stop flag, POEGGn.SSF.

### 20.3.5 Release from Output-Disable

To release the GPT output pins placed in the output-disable state, either return them to their initial state with a reset or clear all of the following flags:

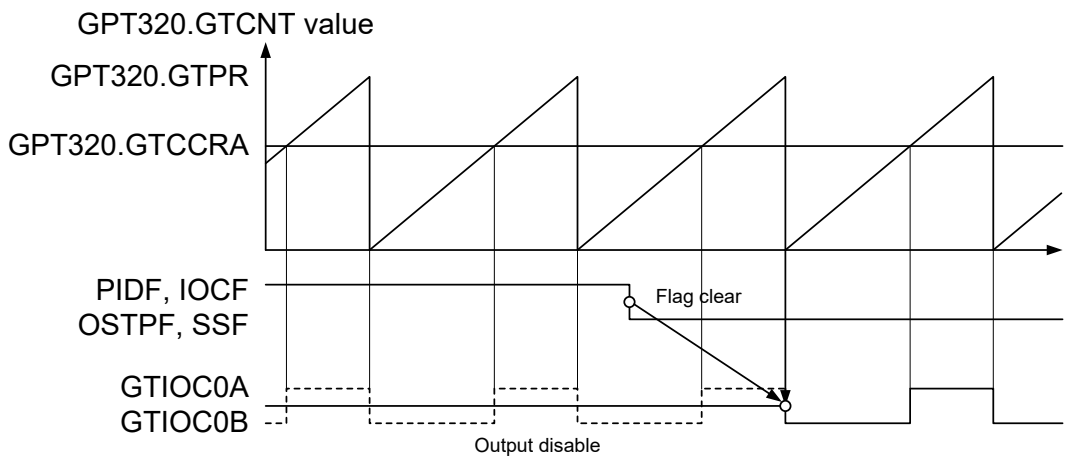
- POEGGn.PIDF
- POEGGn.IOCF
- POEGGn.OSTPF
- POEGGn.SSF

Writing 0 to the POEGGn.PIDF flag is ignored (the flag is not cleared) if the external input pins, GTETRGn are not disabled and the POEGGn.ST bit is not set to 0.

Writing 0 to the POEGGn.IOCF flag is valid (the flag is cleared) only if all of the GTST.OABHF and GTST.OABLF flags in the GPT are set to 0.

Writing 0 to the POEGGn.OSTPF flag is ignored (the flag is not cleared) if the OSTDSR.OSTDF flag in the clock generation circuit is not set to 0. In addition, when the flag set and release occur at the same time, the flag set takes precedence.

Figure 20.3 shows the release timing for output-disable. The output-disable is released at the beginning of the next count cycle of the GPT after the flag is cleared.




Note: The base address of the registers are the same.

**Figure 20.3 Output-disable release timing for GPT pin outputs**

## 20.4 Interrupt Sources

The POEG generates an interrupt request for the following factors:

- Output-disable control by the input level detection
- Output-disable request from the GPT

Table 20.3 lists the conditions for interrupt requests.

**Table 20.3 Interrupt sources and conditions**

| Interrupt source | Symbol | Associated flag | Trigger conditions |
|---|---|---|---|
| POEG group A interrupt | POEG_GROUPA | POEGGA.IOCF | An output-disable request from a GPT disable request occurred |
| | | POEGGA.PIDF | An output-disable request from the GTETRGA pin occurred |
| POEG group B interrupt | POEG_GROUPB | POEGGB.IOCF | An output-disable request from a GPT disable request occurred |
| | | POEGGB.PIDF | An output-disable request from the GTETRGB pin occurred |
| POEG group C interrupt | POEG_GROUPC | POEGGC.IOCF | An output-disable request from a GPT disable request occurred |
| | | POEGGC.PIDF | An output-disable request from the GTETRGC pin occurred |
| POEG group D interrupt | POEG_GROUPD | POEGGD.IOCF | An output-disable request from a GPT disable request occurred |
| | | POEGGD.PIDF | An output-disable request from the GTETRGD pin occurred |

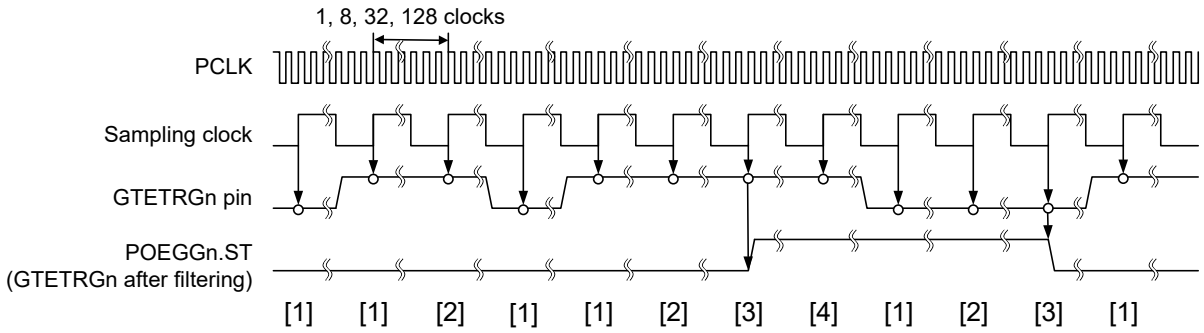
## 20.5 External Trigger Output to the GPT

The POEG outputs signals generated by filtering and level detection of GTETRGn pins input signals as the GPT operation trigger signal for the following:

- Count start
- Count stop
- Count clear
- Up-count
- Down-count
- Input capture

For the POEGGn.INV polarity setting signal, when the same level is input three times continuously with the sampling clock selected in POEGGn.NFCS[1:0], that value is output. Set the control registers the same as for the input level detection operation described in section 20.3.1. Pin Input Level Detection Operation The state after filtering can be monitored in POEGGn.ST.

Figure 20.4 shows the output timing of an external trigger to the GPT.




Note: PCLK = PCLKB
Note: Each channel output can be set in the GPT settings. Polarity can be reversed in POEGGn.INV.

**Figure 20.4 Output timing of external trigger to the GPT**

## 20.6 Usage Notes

### 20.6.1 Transition to Software Standby Mode

When using the POEG, do not invoke Software Standby mode. In this mode, the POEG stops and therefore output disable of the pins cannot be controlled.

### 20.6.2 Specifying Pins Associated with the GPT

The POEG controls output-disable only when a pin is associated with the GPT in the PmnPFS.PMR and PmnPFS.PSEL settings. When the pin is specified as a general I/O pin, the POEG does not perform output-disable control.

# 21. General PWM Timer (GPT)

## 21.1 Overview

The General PWM Timer (GPT) is a 32-bit timer with GPT32 × 4 channels and a 16-bit timer with GPT16 × 4 channels. PWM waveforms can be generated by controlling the up-counter, down-counter, or the up- and down-counter. In addition, PWM waveforms can be generated for controlling brushless DC motors. The GPT can also be used as a general-purpose timer.

Table 21.1 lists the GPT specifications, Table 21.2 shows the GPT functions, and Figure 21.1 shows a block diagram.

**Table 21.1 GPT specifications**

| Item | Description |
|---|---|
| Functions | • 32 bits × 4 channels (GPT32n (n = 0 to 3))<br>• 16 bits × 4 channels (GPT16m (m = 4 to 7))<br>• Up-counting or down-counting (saw waves) or up/down-counting (triangle waves) for each counter<br>• Clock sources independently selectable for each channel<br>• Two input/output pins per channel<br>• Two output compare/input capture registers per channel<br>• For the two output compare/input capture registers of each channel, four registers are provided as buffer registers and are capable of operating as comparison registers when buffering is not in use<br>• In output compare operation, buffer switching can be at crests or troughs, enabling the generation of laterally asymmetric PWM waveforms<br>• Registers for setting up frame cycles in each channel with capability for generating interrupts at overflow or underflow<br>• Generation of dead times in PWM operation<br>• Synchronous starting, stopping and clearing counters for arbitrary channels<br>• Count start, count stop, count clear, up-count, down-count, or input capture operation in response to a maximum of 8 ELC events<br>• Count start, count stop, count clear, up-count, down-count, or input capture operation in response to the status of two input pins<br>• Count start, count stop, count clear, up-count, down-count, or input capture operation in response to a maximum of 4 external triggers<br>• Output pin disable function by detected short-circuits between output pins<br>• PWM waveform for controlling brushless DC motors can be generated<br>• Compare match A to F event, overflow/underflow event, and input UVW edge event can be output to the ELC<br>• Enables the noise filter for input capture and input UVW<br>• Period count function<br>• Logical operation between the channel output<br>• Bus clock: PCLKA, Core clock: PCLKD<br>• Frequency ratio: PCLKA:PCLKD = 1:N (N = 1/2/4/8/16/32/64) |

**Table 21.2 GPT functions (1 of 2)**

| Parameter | Description |
|---|---|
| Count clock | PCLKD<br>PCLKD/2<br>PCLKD/4<br>PCLKD/8<br>PCLKD/16<br>PCLKD/32<br>PCLKD/64<br>PCLKD/256<br>PCLKD/1024<br>GTETRGA, GTETRGB, GTETRGC, GTETRGD |
| Output compare/input capture registers (GTCCR) | GTCCRA<br>GTCCRB |
| Compare/buffer registers | GTCCRC<br>GTCCRD<br>GTCCRE<br>GTCCRF |
| Cycle setting register | GTPR |
| Cycle setting buffer register | GTPBR |

**Table 21.2 GPT functions (2 of 2)**

| Parameter | | Description |
|---|---|---|
| I/O pins | | GTIOCnA<br>GTIOCnB<br>(n = 0 to 7) |
| External trigger input pin*1 | | GTETRGA<br>GTETRGB<br>GTETRGC<br>GTETRGD |
| Counter clear sources | | GTPR register compare match<br>Input capture<br>Input pin status<br>ELC event input<br>GTETRGn (n = A to D) pin input |
| Period count function | | Available (GPT32n (n = 0, 1), GPT16m (m = 4 to 6)) |
| Compare match output | Low output | Available |
| | High output | Available |
| | Toggle output | Available |
| Input capture function | | Available |
| Automatic addition of dead time | | Available<br>(no dead time buffer) |
| PWM mode | | Available |
| Phase count function | | Available |
| Buffer operation | | Double buffer<br>Simultaneous operation disable control for multiple channels |
| One-shot operation | | Available |
| DMAC/DTC activation | | All the interrupt sources |
| Brushless DC motor control function | | Available |
| Interrupt sources | | 9 sources<br>• GTCCRA comare match/input capture(GPTn_CCMPA)<br>• GTCCRB comare match/input capture(GPTn_CCMPB)<br>• GTCCRC comare match(GPTn_CMPC)<br>• GTCCRD comare match(GPTn_CMPD)<br>• GTCCRE comare match(GPTn_CMPE)<br>• GTCCRF comare match(GPTn_CMPF)<br>• GTCNT overflow (GTPR compare match) (GPTn_OVF)<br>• GTCNT underflow (GPTn_UDF)<br>• GTPC count stop(GPTx_PC) (x = 0,1,4-6) |
| Event linking (ELC) function | | Available*2 |
| Noise filtering function | | Available |
| Logical operation between the channel output | | Available |
| TrustZone Filter | | Available |

Note 1. GTETRGn connects to GPT through the POEG module. Therefore, to use the GPT function, supply the POEG clock by clearing the MSTPCRD.MSTPDn (n = 11 to 14) bit.
Note 2. See section 21.5. Operations Linked by ELC.

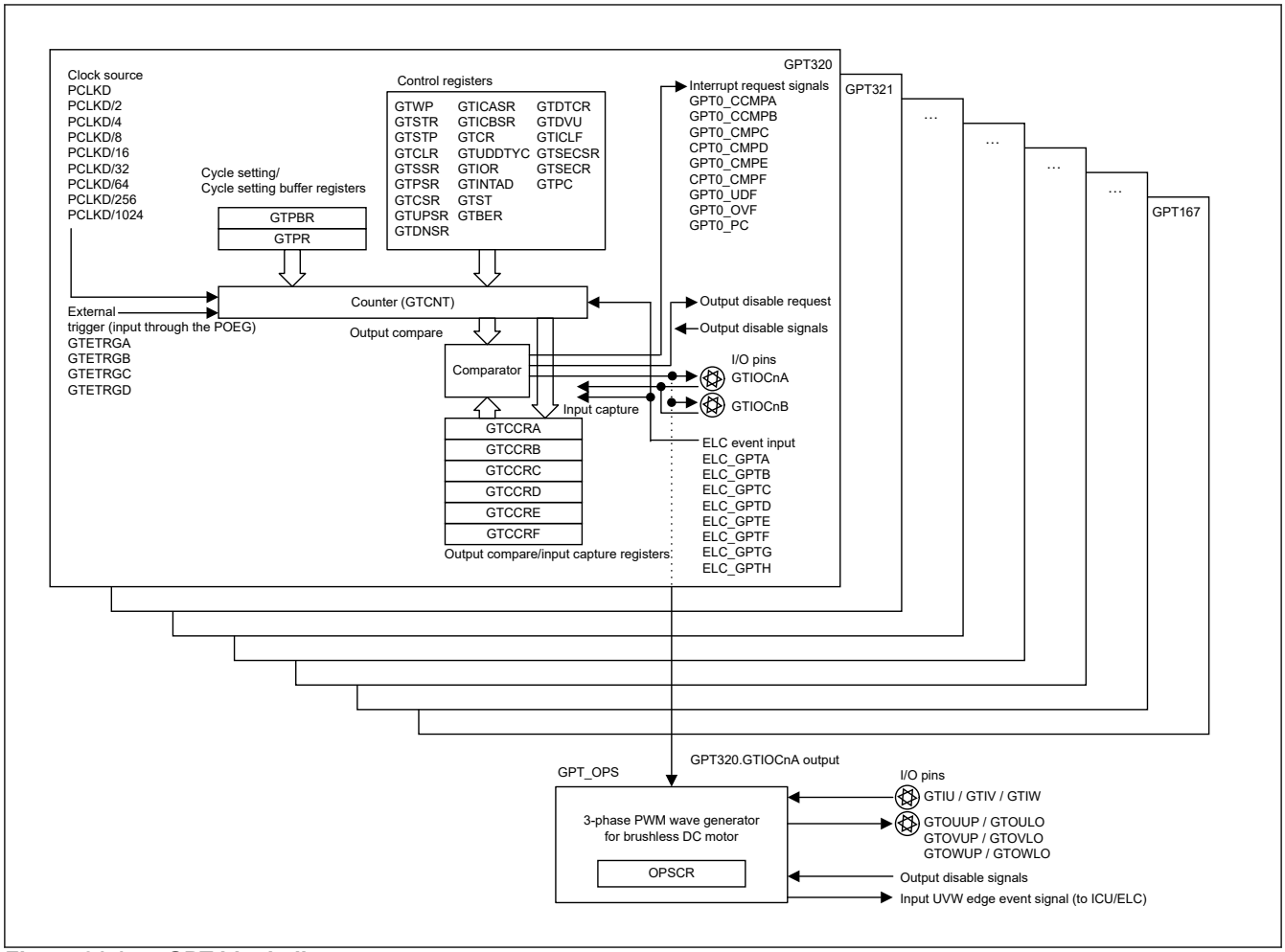



**Figure 21.1 GPT block diagram**

Figure 21.2 shows an example using multiple GPTs.

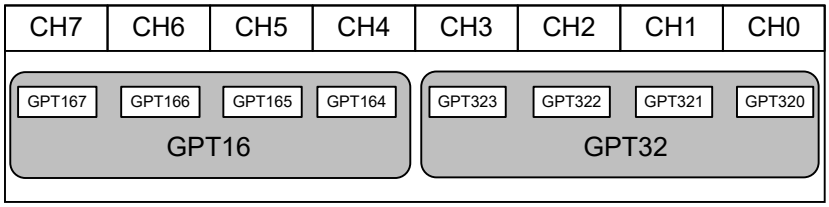



**Figure 21.2 Association between GPT channels and module names**

Table 21.3 lists the I/O pins.

**Table 21.3 GPT I/O pins (1 of 2)**

| Channel | Pin name | I/O | Function |
|---|---|---|---|
| Common | GTETRGx | Input | External trigger input pin x (input through the POEG) |
| GPT32n | GTIOCnA | I/O | GTCCRA register input capture input/output compare output/PWM output pin |
| | GTIOCnB | I/O | GTCCRB register input capture input/output compare output/PWM output pin |
| GPT16m | GTIOCmA | I/O | GTCCRA register input capture input/output compare output/PWM output pin |
| | GTIOCmB | I/O | GTCCRB register input capture input/output compare output/PWM output pin |

**Table 21.3 GPT I/O pins (2 of 2)**

| Channel | Pin name | I/O | Function |
|---|---|---|---|
| GPT_OPS | GTIU | Input | Hall sensor input pin U |
| | GTIV | Input | Hall sensor input pin V |
| | GTIW | Input | Hall sensor input pin W |
| | GTOUUP | Output | 3-phase PWM output for BLDC motor control (positive U-phase) |
| | GTOULO | Output | 3-phase PWM output for BLDC motor control (negative U-phase) |
| | GTOVUP | Output | 3-phase PWM output for BLDC motor control (positive V-phase) |
| | GTOVLO | Output | 3-phase PWM output for BLDC motor control (negative V-phase) |
| | GTOWUP | Output | 3-phase PWM output for BLDC motor control (positive W-phase) |
| | GTOWLO | Output | 3-phase PWM output for BLDC motor control (negative W-phase) |

Note: x: A to D
n: 0 to 3
m: 4 to 7

## 21.2 Register Descriptions

### 21.2.1 GTWP : General PWM Timer Write-Protection Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x00

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PRKEY[7:0] | | | | | | | | — | — | — | CMNWP | CLRWP | STPWP | STRWP | WP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | WP | Register Write Disable<br>0: Write to the register enabled<br>1: Write to the register disabled | R/W |
| 1 | STRWP | GTSTR.CSTRT Bit Write Disable<br>0: Write to the bit is enabled<br>1: Write to the bit is disabled | R/W |
| 2 | STPWP | GTSTP.CSTOP Bit Write Disable<br>0: Write to the bit is enabled<br>1: Write to the bit is disabled | R/W |
| 3 | CLRWP | GTCLR.CCLR Bit Write Disable<br>0: Write to the bit is enabled<br>1: Write to the bit is disabled | R/W |
| 4 | CMNWP | Common Register Write Disabled<br>0: Write to the register is enabled<br>1: Write to the register is disabled | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | PRKEY[7:0] | GTWP Key Code<br>When 0xA5 is written to these bits, writing to the WP, STRWP, STPWP, CLRWP, and CMNWP bits are permitted. These bits are read as 0. | W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

GTWP enables or disables writing to registers to prevent accidental modification. Protection by the GTWP register is only for the writes by the CPU. GTWP does not protect registers from updates that occur in association with CPU writes.

#### WP bit (Register Write Disable)

The following is a list of write enabled or disabled registers:

GTSSR, GTPSR, GTCSR, GTUPSR, GTDNSR, GTICASR, GTICBSR, GTCR, GTUDDTYC, GTIOR, GTINTAD, GTST, GTBER, GTCNT, GTCCRA, GTCCRB, GTCCRC, GTCCRD, GTCCRE, GTCCRF, GTPR, GTPBR, GTDTCR, GTDVU, GTICLF, GTPC.

#### STRWP bit (GTSTR.CSTRT Bit Write Disable)

The STRWP bit enables or disables starting the updating of counter values by writing to the CSTRTn bit (n = 0 to 7) corresponding to a channel number in the GTSTR register.

The bit position of each CSTRTn bit in the GTSTR register is allocated to the channel with the corresponding number, and writing to the GTSTR register for any channel results in writing to the registers of all channels. The STRWP bit for each channel does not control writing but only controls updating of the CSTRT bit for the corresponding channel when simultaneously writing to all channels.

Therefore, when writing to the CSTRT bits of a channel for which the setting of the STRWP bit is 1(disabling writing), the CSTRT bit for the given channel is not updated, but the CSTRT bits corresponding to channel for which the setting of the STRWP bit is 0 (enabling writing) are updated. For example, when the setting of the GPT320.GTWP.STRWP bit is 0 (enabling writing), writing 1 to the GPT321.GTSTR.CSTRT0 bit when its current setting is 0 causes the value to be updated, and the GPT320.GTCNT counter starts to run. When the setting of the GPT320.GTWP.STRWP bit is 1 (disabling writing), writing 1 to the GPT321.GTSTR.CSTRT0 bit when its current setting is 0 leaves the bit with the value 0, and the GPT320.GTCNT counter does not run.

If you want to protect all bits in the GTSTR register from being updated, set the STRWP bits of all channels to 1.

#### STPWP bit (GTSTP.CSTOP Bit Write Disable)

The STPWP bit enables or disables starting the updating of counter values by writing to the CSTOPn bit (n = 0 to 7) corresponding to a channel number in the GTSTP register.

The bit position of each CSTOPn bit in the GTSTP registers is allocated to the channel with the corresponding number, and the writing to the GTSTP register for any channel results in writing to the registers of all channels. The STPWP bit for each channel does not control writing but only controls updating of the CSTOP bit for the corresponding channel when simultaneously writing to all channels.

Therefore, when writing to the CSTOP bits of a channel for which the setting of the STPWP bit is 1(disabling writing), the CSTOP bit for the given channel is not updated, but the CSTOP bits corresponding to channel for which the setting of the STPWP bit is 0 (enabling writing) are updated. For example, when the setting of the GPT320.GTWP.STPWP bit is 0 (enabling writing), writing 1 to the GPT321.GTSTP.CSTOP0 bit when its current setting is 0 causes the value to be updated, and the GPT320.GTCNT counter is stopped. When the setting of the GPT320.GTWP.STPWP bit is 1 (disabling writing), writing 1 to the GPT321.GTSTP.CSTOP0 bit when its current setting is 0 leaves the bit with the value 0, and the GPT320.GTCNT counter is not stopped.

If you want to protect all bits in the GTSTP register from being updated, set the STPWP bits of all channels to 1.

#### CLRWP bit (GTCLR.CCLR Bit Write Disable)

CLRWP bit enables or disables starting the updating of counter values by writing to the CCLRn bit (n = 0 to 7) corresponding to a channel number in the GTCLR register.

The bit position of each CCLRn bit in the GTCLR registers is allocated to the channel with the corresponding number, and the writing to the GTCLR register for any channel results in writing to the registers of all channels. The CLRWP bit for each channel does not control writing but only controls updating of the CCLR bit for the corresponding channel when simultaneously writing to all channels.

Therefore, when writing to the CCLR bits of a channel for which the setting of the CLRWP bit is 1(disabling writing), the CCLR bit for the given channel is not updated, but the CCLR bits corresponding to channel for which the setting of the

CLRWP bit is 0 (enabling writing) are updated. For example, when the setting of the GPT320.GTWP.CLRWP bit is 0 (enabling writing), writing 1 to the GPT321.GTCLR.CCLR0 bit when its current setting is 0 causes the value to be updated, and the GPT320.GTCNT counter is cleared. When the setting of the GPT320.GTWP.CLRWP bit is 1 (disabling writing), writing 1 to the GPT321.GTCLR.CCLR0 bit when its current setting is 0 leaves the bit with the value 0, and the GPT320.GTCNT counter is not cleared.

If you want to protect all bits in the GTCLR register from being updated, set the CLRWP bits of all channels to 1.

#### CMNWP bit (Common Register Write Disabled)

CMNWP bit enables or disables starting the updating of counter values by writing to the SECSELn bit (n = 0 to 7) corresponding to a channel number in the GTSECSR register or to the GTSECR register.

The bit position of each SECSEL bit in the GTSECSR registers is allocated to the channel with the corresponding number, and the writing to the GTSECSR register for any channel results in writing to the registers of all channels. Writing to the GTSECR register of any channel leads to writing to the registers of all channels. The CMNWP bit for each channel does not control writing but only controls updating of the SECSEL bit and the GTSECR register value for the corresponding channel when simultaneously writing to all channels.

Therefore, when writing to the SECSEL bit and the GTSECR register value of a channel for which the setting of the CMNWP bit is 1(disabling writing), the SECSEL bit and the GTSECR register value for the given channel is not updated, but the SECSEL bit and the GTSECR register value corresponding to channel for which the setting of the CMNWP bit is 0 (enabling writing) are updated.

For example, when the setting of the GPT320.GTWP.CMNWP bit is 0 (enabling writing), writing to the GPT321.GTSECSR.SECSEL0 bit causes the value of the GPT320.GTSECSR.SECSEL0 bit to be updated. In the same way, writing to the GPT321.GTSECR register updates the value of the GPT320.GTSECR register. When the setting of the GPT320.GTWP.CMNWP bit is 1 (disabling writing), writing to the GPT321.GTSECSR.SECSEL0 bit does not cause the value of the GPT320.GTSECSR.SECSEL0 bit to be updated. In the same way, writing to the GPT321.GTSECR register does not update the value of the GPT320.GTSECR register.

If you want to protect all bits in the GTSECSR and GTSECR registers from being updated, set the CMNWP bits of all channels to 1.

#### PRKEY[7:0] bit (GTWP Key Code)

This bit controls whether the WP, STRWP, STPWP, CLRWP, and CMNWP bits can be overwritten.

### 21.2.2 GTSTR : General PWM Timer Software Start Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x04

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CSTRT31 to CSTRT0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | CSTRT0 to CSTRT31*1 | Channel n GTCNT Count Start (n is the same as the bit position value)<br>0: GTCNT counter is not started<br>1: GTCNT counter is started | R/W |

Note 1. The bits that can be used vary depending on the product. The n in CSTRTn is the same as the GPT channel number. For this product, n is 0 to 7.

The GTSTR starts the GTCNT counter operation for each channel n, where n = 0 to 7.

The GTSTR bit number represents the channel number. The GTSTR register of each channel is shared by all of the channels. The GTCNT counter starts for the channel associated with the GTSTR bit number where 1 is written. Writing 0 has no effect on the status of GTCNT counter and the value of GTSTR register.

The bit corresponding to channel which security attribution is configured as secure can be read by non-secure access but cannot be written by non-secure access. For example, if GPT channel 0 is configured as secure and other GPTs are

configured as non-secure, the CSTRT0 bit cannot be written by non-secure access to GPT321.GTSTR register, and the GTCNT counter operation status of GPT channel 0 is not changed. When the GTSTR register in GPT channel 1 is read by non-secure access in the same security configuratin as the previous example, the GTCNT counter operation status of GPT channel 0 (CSTRT0 bit) can be read.

For the association between module names and channel numbers, see Figure 21.2.

#### CSTRTn bits (Channel n GTCNT Count Start (n = 0 to 7))

The CSTRTn bits start channel n of the GTCNT counter operation. Writing to the GTSTR.CSTRTn bit (n = 0 to 7) has no effect unless the GTSSR.CSTRT bit is set to 1.

The read data shows the counter status of each channel (GTCR.CST bit). A value of 0 means the counter is stopped and 1 means the counter is running.

### 21.2.3 GTSTP : General PWM Timer Software Stop Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x08

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CSTOP31 to CSTOP0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | CSTOP0 to CSTOP31*1 | Channel n GTCNT Count Stop (n is the same as the bit position value)<br>0: GTCNT counter is not stopped<br>1: GTCNT counter stopped | R/W |

Note 1. The bits that can be used vary depending on the product. The n in CSTOPn is the same as the GPT channel number. For this product, n is 0 to 7.

The GTSTP stops the GTCNT counter operation for each channel n, where n = 0 to 7.

The GTSTP bit number represents the channel number. The GTSTP register of each channel is shared by all the channels. The GTCNT counter stops for the channel associated with the GTSTP bit number where 1 is written. Writing 0 has no effect on the status of the GTCNT counter and the value of GTSTP register.

The bit corresponding to channel which security attribution is configured as secure can be read by non-secure access but cannot be written by non-secure access. For example, if GPT channel 0 is configured as secure and other GPTs are configured as non-secure, the CSTOP0 bit cannot be written by non-secure access to GTSTP register in GPT channel 1, and the GTCNT counter operation status of GPT channel 0 is not changed. When the GTSTP register in GPT channel 1 is read by non-secure access in the same security configuratin as the previous example, the GTCNT counter operation status of GPT channel 0 (CSTOP0 bit) can be read.

For the association between module names and channel numbers, see Figure 21.2.

#### CSTOPn bits (Channel n GTCNT Count Stop (n = 0 to 7))

The CSTOPn bits stop channel n of the GTCNT counter operation. Writing to the GTSTP.CSTOPn bit (n = 0 to 7) has no effect unless the GTPSR.CSTOP bit is set to 1. The read data shows the counter status of each channel (invert of GTCR.CST bit). A value of 0 means the counter is running and 1 means the counter is stopped.

## 21.2.4 GTCLR : General PWM Timer Software Clear Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x0C

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CCLR31 to CCLR0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | CCLR0 to CCLR31*1 | Channel n GTCNT Count Clear (n : the same as bit position value)<br>0: GTCNT counter is not cleared<br>1: GTCNT counter is cleared | W |

Note 1. The bits that can be used vary depending on the product. The n of CCLRn is the same as the GPT channel number. For this product, n is 0 to 7.

The GTCLR is a write-only register that clears the GTCNT counter operation for each channel n, where n = 0 to 7.

The GTCLR bit number represents the channel number. The GTCLR register of each channel is shared by all the channels. The GTCNT counter is cleared for the channel associated with the GTCLR bit number where 1 is written. Writing 0 has no effect on the status of GTCNT counter.

The bit corresponding to channel which security attribution is configured as secure can not be written by non-secure access. For example, if GPT channel 0 is configured as secure and other GPTs are configured as non-secure, the CCLR0 bit cannot be written by non-secure access to GTCLR register in GPT channel 1, and the GTCNT counter of GPT channel 0 is not cleared.

For the association between module names and channel numbers, see Figure 21.2.

### CCLRn bits (Channel n GTCNT Count Clear (n = 0 to 7))

When the counting direction flag is set for decrement (GTST.TUCF flag = 0) with saw-wave mode selected in the GTCR.MD[2:0] bits, the value of the GTCNT counter becomes that of the corresponding GTPR register in response to writing 1 to the CCLRn bit. The value of the counter becomes 0x0000 0000 with other settings. These bits are read as 0.

## 21.2.5 GTSSR : General PWM Timer Start Source Select Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x10

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CSTRT | — | — | — | — | — | — | — | SSELCH | SSELCG | SSELCF | SSELCE | SSELCD | SSELCC | SSELCB | SSELCA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SSCBFAH | SSCBFAL | SSCBRAH | SSCBRAL | SSCAFBH | SSCAFBL | SSCARBH | SSCARBL | SSGTRGDF | SSGTRGDR | SSGTRGCF | SSGTRGCR | SSGTRGBF | SSGTRGBR | SSGTRGAF | SSGTRGAR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SSGTRGAR | GTETRGA Pin Rising Input Source Counter Start Enable<br>0: Counter start disabled on the rising edge of GTETRGA input<br>1: Counter start enabled on the rising edge of GTETRGA input | R/W |
| 1 | SSGTRGAF | GTETRGA Pin Falling Input Source Counter Start Enable<br>0: Counter start disabled on the falling edge of GTETRGA input<br>1: Counter start enabled on the falling edge of GTETRGA input | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2 | SSGTRGBR | GTETRGB Pin Rising Input Source Counter Start Enable<br>0: Counter start disabled on the rising edge of GTETRGB input<br>1: Counter start enabled on the rising edge of GTETRGB input | R/W |
| 3 | SSGTRGBF | GTETRGB Pin Falling Input Source Counter Start Enable<br>0: Counter start disabled on the falling edge of GTETRGB input<br>1: Counter start enabled on the falling edge of GTETRGB input | R/W |
| 4 | SSGTRGCR | GTETRGC Pin Rising Input Source Counter Start Enable<br>0: Counter start disabled on the rising edge of GTETRGC input<br>1: Counter start enabled on the rising edge of GTETRGC input | R/W |
| 5 | SSGTRGCF | GTETRGC Pin Falling Input Source Counter Start Enable<br>0: Counter start disabled on the falling edge of GTETRGC input<br>1: Counter start enabled on the falling edge of GTETRGC input | R/W |
| 6 | SSGTRGDR | GTETRGD Pin Rising Input Source Counter Start Enable<br>0: Counter start disabled on the rising edge of GTETRGD input<br>1: Counter start enabled on the rising edge of GTETRGD input | R/W |
| 7 | SSGTRGDF | GTETRGD Pin Falling Input Source Counter Start Enable<br>0: Counter start disabled on the falling edge of GTETRGD input<br>1: Counter start enabled on the falling edge of GTETRGD input | R/W |
| 8 | SSCARBL | GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Start Enable<br>0: Counter start disabled on the rising edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter start enabled on the rising edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 9 | SSCARBH | GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Start Enable<br>0: Counter start disabled on the rising edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter start enabled on the rising edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 10 | SSCAFBL | GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Start Enable<br>0: Counter start disabled on the falling edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter start enabled on the falling edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 11 | SSCAFBH | GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Start Enable<br>0: Counter start disabled on the falling edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter start enabled on the falling edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 12 | SSCBRAL | GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Start Enable<br>0: Counter start disabled on the rising edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter start enabled on the rising edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 13 | SSCBRAH | GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Start Enable<br>0: Counter start disabled on the rising edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter start enabled on the rising edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 14 | SSCBFAL | GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Start Enable<br>0: Counter start disabled on the falling edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter start enabled on the falling edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 15 | SSCBFAH | GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Start Enable<br>0: Counter start disabled on the falling edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter start enabled on the falling edge of GTIOCnB input when GTIOCnA input is 1 | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 16 | SSELCA | ELC_GPTA Event Source Counter Start Enable<br>0: Counter start disabled at the ELC_GPTA input<br>1: Counter start enabled at the ELC_GPTA input | R/W |
| 17 | SSELCB | ELC_GPTB Event Source Counter Start Enable<br>0: Counter start disabled at the ELC_GPTB input<br>1: Counter start enabled at the ELC_GPTB input | R/W |
| 18 | SSELCC | ELC_GPTC Event Source Counter Start Enable<br>0: Counter start disabled at the ELC_GPTC input<br>1: Counter start enabled at the ELC_GPTC input | R/W |
| 19 | SSELCD | ELC_GPTD Event Source Counter Start Enable<br>0: Counter start disabled at the ELC_GPTD input<br>1: Counter start enabled at the ELC_GPTD input | R/W |
| 20 | SSELCE | ELC_GPTE Event Source Counter Start Enable<br>0: Counter start disabled at the ELC_GPTE input<br>1: Counter start enabled at the ELC_GPTE input | R/W |
| 21 | SSELCF | ELC_GPTF Event Source Counter Start Enable<br>0: Counter start disabled at the ELC_GPTF input<br>1: Counter start enabled at the ELC_GPTF input | R/W |
| 22 | SSELCG | ELC_GPTG Event Source Counter Start Enable<br>0: Counter start disabled at the ELC_GPTG input<br>1: Counter start enabled at the ELC_GPTG input | R/W |
| 23 | SSELCH | ELC_GPTH Event Source Counter Start Enable<br>0: Counter start disabled at the ELC_GPTH input<br>1: Counter start enabled at the ELC_GPTH input | R/W |
| 30:24 | — | These bits are read as 0. The write value should be 0. | R/W |
| 31 | CSTRT | Software Source Counter Start Enable<br>0: Counter start disabled by the GTSTR register<br>1: Counter start enabled by the GTSTR register | R/W |

The GTSSR sets the source to start the GTCNT counter.

Input from GTETRGn (n = A to D) pins are input to the GPT through the POEG. Set the polarity of these signals with the POEG.

#### SSGTRGAR bit (GTETRGA Pin Rising Input Source Counter Start Enable)

The SSGTRGAR bit enables or disables the GTCNT counter start on the rising edge of the GTETRGA pin input.

#### SSGTRGAF bit (GTETRGA Pin Falling Input Source Counter Start Enable)

The SSGTRGAF bit enables or disables the GTCNT counter start on the falling edge of the GTETRGA pin input.

#### SSGTRGBR bit (GTETRGB Pin Rising Input Source Counter Start Enable)

The SSGTRGBR bit enables or disables the GTCNT counter start on the rising edge of the GTETRGB pin input.

#### SSGTRGBF bit (GTETRGB Pin Falling Input Source Counter Start Enable)

The SSGTRGBF bit enables or disables the GTCNT counter start on the falling edge of the GTETRGB pin input.

#### SSGTRGCR bit (GTETRGC Pin Rising Input Source Counter Start Enable)

The SSGTRGCR bit enables or disables the GTCNT counter start on the rising edge of the GTETRGC pin input.

#### SSGTRGCF bit (GTETRGC Pin Falling Input Source Counter Start Enable)

The SSGTRGCF bit enables or disables the GTCNT counter start on the falling edge of the GTETRGC pin input.

#### SSGTRGDR bit (GTETRGD Pin Rising Input Source Counter Start Enable)

The SSGTRGDR bit enables or disables the GTCNT counter start on the rising edge of the GTETRGD pin input.

#### SSGTRGDF bit (GTETRGD Pin Falling Input Source Counter Start Enable)

The SSGTRGDF bit enables or disables the GTCNT counter start on the falling edge of the GTETRGD pin input.

#### SSCARBL bit (GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Start Enable)

The SSCARBL bit enables or disables the GTCNT counter start on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 0.

#### SSCARBH bit (GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Start Enable)

The SSCARBH bit enables or disables the GTCNT counter start on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 1.

#### SSCAFBL bit (GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Start Enable)

The SSCAFBL bit enables or disables the GTCNT counter start on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 0.

#### SSCAFBH bit (GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Start Enable)

The SSCAFBH bit enables or disables the GTCNT counter start on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 1.

#### SSCBRAL bit (GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Start Enable)

The SSCBRAL bit enables or disables the GTCNT counter start on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 0.

#### SSCBRAH bit (GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Start Enable)

The SSCBRAH bit enables or disables the GTCNT counter start on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 1.

#### SSCBFAL bit (GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Start Enable)

The SSCBFAL bit enables or disables the GTCNT counter start on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 0.

#### SSCBFAH bit (GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Start Enable)

The SSCBFAH bit enables or disables the GTCNT counter start on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 1.

#### SSELCm bit (ELC_GPTm Event Source Counter Start Enable) (m = A to H)

The SSELCm bit enables or disables the GTCNT counter start at the ELC_GPTm event input.

#### CSTRT bit (Software Source Counter Start Enable)

The CSTRT bit enables or disables the GTCNT counter start by GTSTR register.

### 21.2.6 GTPSR : General PWM Timer Stop Source Select Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x14

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CSTOP | — | — | — | — | — | — | — | PSELCH | PSELCG | PSELCF | PSELCE | PSELCD | PSELCC | PSELCB | PSELCA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PSCBFAH | PSCBFAL | PSCBRAH | PSCBRAL | PSCAFBH | PSCAFBL | PSCARBH | PSCARBL | PSGTRGDF | PSGTRGDR | PSGTRGCF | PSGTRGCR | PSGTRGBF | PSGTRGBR | PSGTRGAF | PSGTRGAR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PSGTRGAR | GTETRGA Pin Rising Input Source Counter Stop Enable<br>0: Counter stop disabled on the rising edge of GTETRGA input<br>1: Counter stop enabled on the rising edge of GTETRGA input | R/W |
| 1 | PSGTRGAF | GTETRGA Pin Falling Input Source Counter Stop Enable<br>0: Counter stop disabled on the falling edge of GTETRGA input<br>1: Counter stop enabled on the falling edge of GTETRGA input | R/W |
| 2 | PSGTRGBR | GTETRGB Pin Rising Input Source Counter Stop Enable<br>0: Counter stop disabled on the rising edge of GTETRGB input<br>1: Counter stop enabled on the rising edge of GTETRGB input | R/W |
| 3 | PSGTRGBF | GTETRGB Pin Falling Input Source Counter Stop Enable<br>0: Counter stop disabled on the falling edge of GTETRGB input<br>1: Counter stop enabled on the falling edge of GTETRGB input | R/W |
| 4 | PSGTRGCR | GTETRGC Pin Rising Input Source Counter Stop Enable<br>0: Counter stop disabled on the rising edge of GTETRGC input<br>1: Counter stop enabled on the rising edge of GTETRGC input | R/W |
| 5 | PSGTRGCF | GTETRGC Pin Falling Input Source Counter Stop Enable<br>0: Counter stop disabled on the falling edge of GTETRGC input<br>1: Counter stop enabled on the falling edge of GTETRGC input | R/W |
| 6 | PSGTRGDR | GTETRGD Pin Rising Input Source Counter Stop Enable<br>0: Counter stop disabled on the rising edge of GTETRGD input<br>1: Counter stop enabled on the rising edge of GTETRGD input | R/W |
| 7 | PSGTRGDF | GTETRGD Pin Falling Input Source Counter Stop Enable<br>0: Counter stop disabled on the falling edge of GTETRGD input<br>1: Counter stop enabled on the falling edge of GTETRGD input | R/W |
| 8 | PSCARBL | GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Stop Enable<br>0: Counter stop disabled on the rising edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter stop enabled on the rising edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 9 | PSCARBH | GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Stop Enable<br>0: Counter stop disabled on the rising edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter stop enabled on the rising edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 10 | PSCAFBL | GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Stop Enable<br>0: Counter stop disabled on the falling edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter stop enabled on the falling edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 11 | PSCAFBH | GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Stop Enable<br>0: Counter stop disabled on the falling edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter stop enabled on the falling edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 12 | PSCBRAL | GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Stop Enable<br>0: Counter stop disabled on the rising edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter stop enabled on the rising edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 13 | PSCBRAH | GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Stop Enable<br>0: Counter stop disabled on the rising edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter stop enabled on the rising edge of GTIOCnB input when GTIOCnA input is 1 | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14 | PSCBFAL | GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Stop Enable<br>0: Counter stop disabled on the falling edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter stop enabled on the falling edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 15 | PSCBFAH | GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Stop Enable<br>0: Counter stop disabled on the falling edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter stop enabled on the falling edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 16 | PSELCA | ELC_GPTA Event Source Counter Stop Enable<br>0: Counter stop disabled at the ELC_GPTA input<br>1: Counter stop enabled at the ELC_GPTA input | R/W |
| 17 | PSELCB | ELC_GPTB Event Source Counter Stop Enable<br>0: Counter stop disabled at the ELC_GPTB input<br>1: Counter stop enabled at the ELC_GPTB input | R/W |
| 18 | PSELCC | ELC_GPTC Event Source Counter Stop Enable<br>0: Counter stop disabled at the ELC_GPTC input<br>1: Counter stop enabled at the ELC_GPTC input | R/W |
| 19 | PSELCD | ELC_GPTD Event Source Counter Stop Enable<br>0: Counter stop disabled at the ELC_GPTD input<br>1: Counter stop enabled at the ELC_GPTD input | R/W |
| 20 | PSELCE | ELC_GPTE Event Source Counter Stop Enable<br>0: Counter stop disabled at the ELC_GPTE input<br>1: Counter stop enabled at the ELC_GPTE input | R/W |
| 21 | PSELCF | ELC_GPTF Event Source Counter Stop Enable<br>0: Counter stop disabled at the ELC_GPTF input<br>1: Counter stop enabled at the ELC_GPTF input | R/W |
| 22 | PSELCG | ELC_GPTG Event Source Counter Stop Enable<br>0: Counter stop disabled at the ELC_GPTG input<br>1: Counter stop enabled at the ELC_GPTG input | R/W |
| 23 | PSELCH | ELC_GPTH Event Source Counter Stop Enable<br>0: Counter stop disabled at the ELC_GPTH input<br>1: Counter stop enabled at the ELC_GPTH input | R/W |
| 30:24 | — | These bits are read as 0. The write value should be 0. | R/W |
| 31 | CSTOP | Software Source Counter Stop Enable<br>0: Counter stop disabled by the GTSTP register<br>1: Counter stop enabled by the GTSTP register | R/W |

The GTPSR sets the source to stop the GTCNT counter.

Inputs from GTETRGn (n = A to D) pins are input to the GPT through the POEG. Set the polarity of these signals with the POEG.

#### PSGTRGAR bit (GTETRGA Pin Rising Input Source Counter Stop Enable)

The PSGTRGAR bit enables or disables the GTCNT counter stop on the rising edge of the GTETRGA pin input.

#### PSGTRGAF bit (GTETRGA Pin Falling Input Source Counter Stop Enable)

The PSGTRGAF bit enables or disables the GTCNT counter stop on the falling edge of the GTETRGA pin input.

#### PSGTRGBR bit (GTETRGB Pin Rising Input Source Counter Stop Enable)

PSGTRGBR bit enables or disables the GTCNT counter stop on the rising edge of the GTETRGB pin input.

#### PSGTRGBF bit (GTETRGB Pin Falling Input Source Counter Stop Enable)

The PSGTRGBF bit enables or disables the GTCNT counter stop on the falling edge of the GTETRGB pin input.

**PSGTRGCR bit (GTETRGC Pin Rising Input Source Counter Stop Enable)**

PSGTRGCR bit enables or disables the GTCNT counter stop on the rising edge of the GTETRGC pin input.

**PSGTRGCF bit (GTETRGC Pin Falling Input Source Counter Stop Enable)**

The PSGTRGCF bit enables or disables the GTCNT counter stop on the falling edge of the GTETRGC pin input.

**PSGTRGDR bit (GTETRGD Pin Rising Input Source Counter Stop Enable)**

PSGTRGDR bit enables or disables the GTCNT counter stop on the rising edge of the GTETRGD pin input.

**PSGTRGDF bit (GTETRGD Pin Falling Input Source Counter Stop Enable)**

The PSGTRGDF bit enables or disables the GTCNT counter stop on the falling edge of the GTETRGD pin input.

**PSCARBL bit (GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Stop Enable)**

The PSCARBL bit enables or disables the GTCNT counter stop on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 0.

**PSCARBH bit (GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Stop Enable)**

The PSCARBH bit enables or disables the GTCNT counter stop on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**PSCAFBL bit (GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Stop Enable)**

The PSCAFBL bit enables or disables the GTCNT counter stop on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 0.

**PSCAFBH bit (GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Stop Enable)**

The PSCAFBH bit enables or disables the GTCNT counter stop on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**PSCBRAL bit (GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Stop Enable)**

The PSCBRAL bit enables or disables the GTCNT counter stop on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 0.

**PSCBRAH bit (GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Stop Enable)**

The PSCBRAH bit enables or disables the GTCNT counter stop on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 1.

**PSCBFAL bit (GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Stop Enable)**

The PSCBFAL bit enables or disables the GTCNT counter stop on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 0.

**PSCBFAH bit (GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Stop Enable)**

The PSCBFAH bit enables or disables the GTCNT counter stop on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 1.

**PSELCm bit (ELCm Event Source Counter Stop Enable) (m = A to H)**

The PSELCm bit enables or disables the GTCNT counter stop at the ELC_GPTm event input.

**CSTOP bit (Software Source Counter Stop Enable)**

The CSTOP bit enables or disables the GTCNT counter stop by the GTSTP register.

## 21.2.7 GTCSR : General PWM Timer Clear Source Select Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x18

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CCLR | — | — | — | — | — | — | — | CSELCH | CSELCG | CSELCF | CSELCE | CSELCD | CSELCC | CSELCB | CSELCA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CSCBFAH | CSCBFAL | CSCBRAH | CSCBRAL | CSCAFBH | CSCAFBL | CSCARBH | CSCARBL | CSGTRGDF | CSGTRGDR | CSGTRGCF | CSGTRGCR | CSGTRGBF | CSGTRGBR | CSGTRGAF | CSGTRGAR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CSGTRGAR | GTETRGA Pin Rising Input Source Counter Clear Enable<br>0: Counter clear disabled on the rising edge of GTETRGA input<br>1: Counter clear enabled on the rising edge of GTETRGA input | R/W |
| 1 | CSGTRGAF | GTETRGA Pin Falling Input Source Counter Clear Enable<br>0: Counter clear disabled on the falling edge of GTETRGA input<br>1: Counter clear enabled on the falling edge of GTETRGA input | R/W |
| 2 | CSGTRGBR | GTETRGB Pin Rising Input Source Counter Clear Enable<br>0: Disable counter clear on the rising edge of GTETRGB input<br>1: Enable counter clear on the rising edge of GTETRGB input | R/W |
| 3 | CSGTRGBF | GTETRGB Pin Falling Input Source Counter Clear Enable<br>0: Counter clear disabled on the falling edge of GTETRGB input<br>1: Counter clear enabled on the falling edge of GTETRGB input | R/W |
| 4 | CSGTRGCR | GTETRGC Pin Rising Input Source Counter Clear Enable<br>0: Disable counter clear on the rising edge of GTETRGC input<br>1: Enable counter clear on the rising edge of GTETRGC input | R/W |
| 5 | CSGTRGCF | GTETRGC Pin Falling Input Source Counter Clear Enable<br>0: Counter clear disabled on the falling edge of GTETRGC input<br>1: Counter clear enabled on the falling edge of GTETRGC input | R/W |
| 6 | CSGTRGDR | GTETRGD Pin Rising Input Source Counter Clear Enable<br>0: Disable counter clear on the rising edge of GTETRGD input<br>1: Enable counter clear on the rising edge of GTETRGD input | R/W |
| 7 | CSGTRGDF | GTETRGD Pin Falling Input Source Counter Clear Enable<br>0: Counter clear disabled on the falling edge of GTETRGD input<br>1: Counter clear enabled on the falling edge of GTETRGD input | R/W |
| 8 | CSCARBL | GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Clear Enable<br>0: Counter clear disabled on the rising edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter clear enabled on the rising edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 9 | CSCARBH | GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Clear Enable<br>0: Counter clear disabled on the rising edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter clear enabled on the rising edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 10 | CSCAFBL | GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Clear Enable<br>0: Counter clear disabled on the falling edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter clear enabled on the falling edge of GTIOCnA input when GTIOCnB input is 0 | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 11 | CSCAFBH | GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Clear Enable<br>0: Counter clear disabled on the falling edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter clear enabled on the falling edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 12 | CSCBRAL | GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Clear Enable<br>0: Counter clear disabled on the rising edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter clear enabled on the rising edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 13 | CSCBRAH | GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Clear Enable<br>0: Counter clear disabled on the rising edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter clear enabled on the rising edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 14 | CSCBFAL | GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Clear Enable<br>0: Counter clear disabled on the falling edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter clear enabled on the falling edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 15 | CSCBFAH | GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Clear Enable<br>0: Counter clear disabled on the falling edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter clear enabled on the falling edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 16 | CSELCA | ELC_GPTA Event Source Counter Clear Enable<br>0: Counter clear disabled at the ELC_GPTA input<br>1: Counter clear enabled at the ELC_GPTA input | R/W |
| 17 | CSELCB | ELC_GPTB Event Source Counter Clear Enable<br>0: Counter clear disabled at the ELC_GPTB input<br>1: Counter clear enabled at the ELC_GPTB input | R/W |
| 18 | CSELCC | ELC_GPTC Event Source Counter Clear Enable<br>0: Counter clear disabled at the ELC_GPTC input<br>1: Counter clear enabled at the ELC_GPTC input | R/W |
| 19 | CSELCD | ELC_GPTD Event Source Counter Clear Enable<br>0: Counter clear disabled at the ELC_GPTD input<br>1: Counter clear enabled at the ELC_GPTD input | R/W |
| 20 | CSELCE | ELC_GPTE Event Source Counter Clear Enable<br>0: Counter clear disabled at the ELC_GPTE input<br>1: Counter clear enabled at the ELC_GPTE input | R/W |
| 21 | CSELCF | ELC_GPTF Event Source Counter Clear Enable<br>0: Counter clear disabled at the ELC_GPTF input<br>1: Counter clear enabled at the ELC_GPTF input | R/W |
| 22 | CSELCG | ELC_GPTG Event Source Counter Clear Enable<br>0: Counter clear disabled at the ELC_GPTG input<br>1: Counter clear enabled at the ELC_GPTG input | R/W |
| 23 | CSELCH | ELC_GPTH Event Source Counter Clear Enable<br>0: Counter clear disabled at the ELC_GPTH input<br>1: Counter clear enabled at the ELC_GPTH input | R/W |
| 30:24 | — | These bits are read as 0. The write value should be 0. | R/W |
| 31 | CCLR | Software Source Counter Clear Enable<br>0: Counter clear disabled by the GTCLR register<br>1: Counter clear enabled by the GTCLR register | R/W |

The GTCSR sets the source to clear the GTCNT counter.

Counter clearing can be executed whether the counter is running (GTCR.CST=1) or stopped (GTCR.CST=0).

Inputs from GTETRGn (n = A to D) pins are input to the GPT through the POEG. Set the polarity of these signals with the POEG.

**CSGTRGAR bit (GTETRGA Pin Rising Input Source Counter Clear Enable)**

The CSGTRGAR bit enables or disables the GTCNT counter clear on the rising edge of the GTETRGA pin input.

**CSGTRGAF bit (GTETRGA Pin Falling Input Source Counter Clear Enable)**

The CSGTRGAF bit enables or disables the GTCNT counter clear on the falling edge of the GTETRGA pin input.

**CSGTRGBR bit (GTETRGB Pin Rising Input Source Counter Clear Enable)**

The CSGTRGBR bit enables or disables the GTCNT counter clear on the rising edge of the GTETRGB pin input.

**CSGTRGBF bit (GTETRGB Pin Falling Input Source Counter Clear Enable)**

The CSGTRGBF bit enables or disables the GTCNT counter clear on the falling edge of the GTETRGB pin input.

**CSGTRGCR bit (GTETRGC Pin Rising Input Source Counter Clear Enable)**

The CSGTRGCR bit enables or disables the GTCNT counter clear on the rising edge of the GTETRGC pin input.

**CSGTRGCF bit (GTETRGC Pin Falling Input Source Counter Clear Enable)**

The CSGTRGCF bit enables or disables the GTCNT counter clear on the falling edge of the GTETRGC pin input.

**CSGTRGDR bit (GTETRGD Pin Rising Input Source Counter Clear Enable)**

The CSGTRGDR bit enables or disables the GTCNT counter clear on the rising edge of the GTETRGD pin input.

**CSGTRGDF bit (GTETRGD Pin Falling Input Source Counter Clear Enable)**

The CSGTRGDF bit enables or disables the GTCNT counter clear on the falling edge of the GTETRGD pin input.

**CSCARBL bit (GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Clear Enable)**

The CSCARBL bit enables or disables the GTCNT counter clear on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 0.

**CSCARBH bit (GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Clear Enable)**

The CSCARBH bit enables or disables the GTCNT counter clear on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**CSCAFBL bit (GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Clear Enable)**

The CSCAFBL bit enables or disables the GTCNT counter clear on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 0.

**CSCAFBH bit (GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Clear Enable)**

The CSCAFBH bit enables or disables the GTCNT counter clear on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**CSCBRAL bit (GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Clear Enable)**

The CSCBRAL bit enables or disables the GTCNT counter clear on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 0.

**CSCBRAH bit (GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Clear Enable)**

The CSCBRAH bit enables or disables the GTCNT counter clear on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 1.

**CSCBFAL bit (GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Clear Enable)**

The CSCBFAL bit enables or disables the GTCNT counter clear on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 0.

### CSCBFAH bit (GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Clear Enable)

The CSCBFAH bit enables or disables the GTCNT counter clear on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 1.

### CSELCm bit (ELCm Event Source Counter Clear Enable) (m = A to H)

The CSELCm bit enables or disables the GTCNT counter clear at the ELC_GPTm event input.

### CCLR bit (Software Source Counter Clear Enable)

The CCLR bit enables or disables the GTCNT counter clear by the GTCLR register.

## 21.2.8 GTUPSR : General PWM Timer Up Count Source Select Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x1C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | USELCH | USELCG | USELCF | USELCE | USELCD | USELCC | USELCB | USELCA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | USCBFAH | USCBFAL | USCBRAH | USCBRAL | USCAFBH | USCAFBL | USCARBH | USCARBL | USGTRGDF | USGTRGDR | USGTRGCF | USGTRGCR | USGTRGBF | USGTRGBR | USGTRGAF | USGTRGAR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | USGTRGAR | GTETRGA Pin Rising Input Source Counter Count Up Enable<br>0: Counter count up disabled on the rising edge of GTETRGA input<br>1: Counter count up enabled on the rising edge of GTETRGA input | R/W |
| 1 | USGTRGAF | GTETRGA Pin Falling Input Source Counter Count Up Enable<br>0: Counter count up disabled on the falling edge of GTETRGA input<br>1: Counter count up enabled on the falling edge of GTETRGA input | R/W |
| 2 | USGTRGBR | GTETRGB Pin Rising Input Source Counter Count Up Enable<br>0: Counter count up disabled on the rising edge of GTETRGB input<br>1: Counter count up enabled on the rising edge of GTETRGB input | R/W |
| 3 | USGTRGBF | GTETRGB Pin Falling Input Source Counter Count Up Enable<br>0: Counter count up disabled on the falling edge of GTETRGB input<br>1: Counter count up enabled on the falling edge of GTETRGB input | R/W |
| 4 | USGTRGCR | GTETRGC Pin Rising Input Source Counter Count Up Enable<br>0: Counter count up disabled on the rising edge of GTETRGC input<br>1: Counter count up enabled on the rising edge of GTETRGC input | R/W |
| 5 | USGTRGCF | GTETRGC Pin Falling Input Source Counter Count Up Enable<br>0: Counter count up disabled on the falling edge of GTETRGC input<br>1: Counter count up enabled on the falling edge of GTETRGC input | R/W |
| 6 | USGTRGDR | GTETRGD Pin Rising Input Source Counter Count Up Enable<br>0: Counter count up disabled on the rising edge of GTETRGD input<br>1: Counter count up enabled on the rising edge of GTETRGD input | R/W |
| 7 | USGTRGDF | GTETRGD Pin Falling Input Source Counter Count Up Enable<br>0: Counter count up disabled on the falling edge of GTETRGD input<br>1: Counter count up enabled on the falling edge of GTETRGD input | R/W |
| 8 | USCARBL | GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Count Up Enable<br>0: Counter count up disabled on the rising edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter count up enabled on the rising edge of GTIOCnA input when GTIOCnB input is 0 | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9 | USCARBH | GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Count Up Enable<br>0: Counter count up disabled on the rising edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter count up enabled on the rising edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 10 | USCAFBL | GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Count Up Enable<br>0: Counter count up disabled on the falling edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter count up enabled on the falling edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 11 | USCAFBH | GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Count Up Enable<br>0: Counter count up disabled on the falling edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter count up enabled on the falling edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 12 | USCBRAL | GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Count Up Enable<br>0: Counter count up disabled on the rising edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter count up enabled on the rising edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 13 | USCBRAH | GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Count Up Enable<br>0: Counter count up disabled on the rising edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter count up enabled on the rising edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 14 | USCBFAL | GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Count Up Enable<br>0: Counter count up disabled on the falling edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter count up enabled on the falling edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 15 | USCBFAH | GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Count Up Enable<br>0: Counter count up disabled on the falling edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter count up enabled on the falling edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 16 | USELCA | ELC_GPTA Event Source Counter Count Up Enable<br>0: Counter count up disabled at the ELC_GPTA input<br>1: Counter count up enabled at the ELC_GPTA input | R/W |
| 17 | USELCB | ELC_GPTB Event Source Counter Count Up Enable<br>0: Counter count up disabled at the ELC_GPTB input<br>1: Counter count up enabled at the ELC_GPTB input | R/W |
| 18 | USELCC | ELC_GPTC Event Source Counter Count Up Enable<br>0: Counter count up disabled at the ELC_GPTC input<br>1: Counter count up enabled at the ELC_GPTC input | R/W |
| 19 | USELCD | ELC_GPTD Event Source Counter Count Up Enable<br>0: Counter count up disabled at the ELC_GPTD input<br>1: Counter count up enabled at the ELC_GPTD input | R/W |
| 20 | USELCE | ELC_GPTE Event Source Counter Count Up Enable<br>0: Counter count up disabled at the ELC_GPTE input<br>1: Counter count up enabled at the ELC_GPTE input | R/W |
| 21 | USELCF | ELC_GPTF Event Source Counter Count Up Enable<br>0: Counter count up disabled at the ELC_GPTF input<br>1: Counter count up enabled at the ELC_GPTF input | R/W |
| 22 | USELCG | ELC_GPTG Event Source Counter Count Up Enable<br>0: Counter count up disabled at the ELC_GPTG input<br>1: Counter count up enabled at the ELC_GPTG input | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 23 | USELCH | ELC_GPTH Event Source Counter Count Up Enable<br>0: Counter count up disabled at the ELC_GPTH input<br>1: Counter count up enabled at the ELC_GPTH input | R/W |
| 31:24 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTUPSR sets the source to count up the GTCNT counter.

When at least one bit in the GTUPSR register is set to 1, the GTCNT counter is counted up by the source that is set to 1 in this register. In this case, GTCR.TPCS has no effect.

Number of increment in counting is one even when multiple sources are generated simultaneously.

Inputs from GTETRGn (n = A to D) pins are input to the GPT through the POEG. Set the polarity of these signals with the POEG.

#### USGTRGAR bit (GTETRGA Pin Rising Input Source Counter Count Up Enable)

The USGTRGAR bit enables or disables the GTCNT counter count up on the rising edge of the GTETRGA pin input.

#### USGTRGAF bit (GTETRGA Pin Falling Input Source Counter Count Up Enable)

The USGTRGAF bit enables or disables the GTCNT counter count up on the falling edge of the GTETRGA pin input.

#### USGTRGBR bit (GTETRGB Pin Rising Input Source Counter Count Up Enable)

The USGTRGBR bit enables or disables the GTCNT counter count up on the rising edge of the GTETRGB pin input.

#### USGTRGBF bit (GTETRGB Pin Falling Input Source Counter Count Up Enable)

The USGTRGBF bit enables or disables the GTCNT counter count up on the falling edge of the GTETRGB pin input.

#### USGTRGCR bit (GTETRGC Pin Rising Input Source Counter Count Up Enable)

The USGTRGCR bit enables or disables the GTCNT counter count up on the rising edge of the GTETRGC pin input.

#### USGTRGCF bit (GTETRGC Pin Falling Input Source Counter Count Up Enable)

The USGTRGCF bit enables or disables the GTCNT counter count up on the falling edge of the GTETRGC pin input.

#### USGTRGDR bit (GTETRGD Pin Rising Input Source Counter Count Up Enable)

The USGTRGDR bit enables or disables the GTCNT counter count up on the rising edge of the GTETRGD pin input.

#### USGTRGDF bit (GTETRGD Pin Falling Input Source Counter Count Up Enable)

The USGTRGDF bit enables or disables the GTCNT counter count up on the falling edge of the GTETRGD pin input.

#### USCARBL bit (GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Count Up Enable)

The USCARBL bit enables or disables GTCNT counter count up on the rising edge of GTIOCnA pin input, when GTIOCnB input is 0.

#### USCARBH bit (GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Count Up Enable)

The USCARBH bit enables or disables the GTCNT counter count up on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 1.

#### USCAFBL bit (GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Count Up Enable)

The USCAFBL bit enables or disables the GTCNT counter count up on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 0.

#### USCAFBH bit (GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Count Up Enable)

The USCAFBH bit enables or disables the GTCNT counter count up on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 1.

### USCBRAL bit (GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Count Up Enable)

The USCBRAL bit enables or disables the GTCNT counter count up on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 0.

### USCBRAH bit (GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Count Up Enable)

The USCBRAH bit enables or disables the GTCNT counter count up on the rising edge of the GTIOCnB pin input, when the GTIOCnA input is 1.

### USCBFAL bit (GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Count Up Enable)

The USCBFAL bit enables or disables the GTCNT counter count up on the falling edge of the GTIOCnB pin input, when the GTIOCnA input is 0.

### USCBFAH bit (GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Count Up Enable)

The USCBFAH bit enables or disables the GTCNT counter count up on the falling edge of the GTIOCnB pin input, when the GTIOCnA input is 1.

### USELCm bit (ELC_GPTm Event Source Counter Count Up Enable) (m = A to H)

The USELCm bit enables or disables the GTCNT counter count up at the ELC_GPTm event input.

## 21.2.9 GTDNSR : General PWM Timer Down Count Source Select Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x20

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | DSELCH | DSELCG | DSELCF | DSELCE | DSELCD | DSELCC | DSELCB | DSELCA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DSCBFAH | DSCBFAL | DSCBRAH | DSCBRAL | DSCAFBH | DSCAFBL | DSCARBH | DSCARBL | DSGTRGDF | DSGTRGDR | DSGTRGCF | DSGTRGCR | DSGTRGBF | DSGTRGBR | DSGTRGAF | DSGTRGAR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DSGTRGAR | GTETRGA Pin Rising Input Source Counter Count Down Enable<br>0: Counter count down disabled on the rising edge of GTETRGA input<br>1: Counter count down enabled on the rising edge of GTETRGA input | R/W |
| 1 | DSGTRGAF | GTETRGA Pin Falling Input Source Counter Count Down Enable<br>0: Counter count down disabled on the falling edge of GTETRGA input<br>1: Counter count down enabled on the falling edge of GTETRGA input | R/W |
| 2 | DSGTRGBR | GTETRGB Pin Rising Input Source Counter Count Down Enable<br>0: Counter count down disabled on the rising edge of GTETRGB input<br>1: Counter count down enabled on the rising edge of GTETRGB input | R/W |
| 3 | DSGTRGBF | GTETRGB Pin Falling Input Source Counter Count Down Enable<br>0: Counter count down disabled on the falling edge of GTETRGB input<br>1: Counter count down enabled on the falling edge of GTETRGB input | R/W |
| 4 | DSGTRGCR | GTETRGC Pin Rising Input Source Counter Count Down Enable<br>0: Counter count down disabled on the rising edge of GTETRGC input<br>1: Counter count down enabled on the rising edge of GTETRGC input | R/W |
| 5 | DSGTRGCF | GTETRGC Pin Falling Input Source Counter Count Down Enable<br>0: Counter count down disabled on the falling edge of GTETRGC input<br>1: Counter count down enabled on the falling edge of GTETRGC input | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6 | DSGTRGDR | GTETRGD Pin Rising Input Source Counter Count Down Enable<br>0: Counter count down disabled on the rising edge of GTETRGD input<br>1: Counter count down enabled on the rising edge of GTETRGD input | R/W |
| 7 | DSGTRGDF | GTETRGD Pin Falling Input Source Counter Count Down Enable<br>0: Counter count down disabled on the falling edge of GTETRGD input<br>1: Counter count down enabled on the falling edge of GTETRGD input | R/W |
| 8 | DSCARBL | GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Count Down Enable<br>0: Counter count down disabled on the rising edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter count down enabled on the rising edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 9 | DSCARBH | GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Count Down Enable<br>0: Counter count down disabled on the rising edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter count down enabled on the rising edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 10 | DSCAFBL | GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Count Down Enable<br>0: Counter count down disabled on the falling edge of GTIOCnA input when GTIOCnB input is 0<br>1: Counter count down enabled on the falling edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 11 | DSCAFBH | GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Count Down Enable<br>0: Counter count down disabled on the falling edge of GTIOCnA input when GTIOCnB input is 1<br>1: Counter count down enabled on the falling edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 12 | DSCBRAL | GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Count Down Enable<br>0: Counter count down disabled on the rising edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter count down enabled on the rising edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 13 | DSCBRAH | GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Count Down Enable<br>0: Counter count down disabled on the rising edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter count down enabled on the rising edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 14 | DSCBFAL | GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Count Down Enable<br>0: Counter count down disabled on the falling edge of GTIOCnB input when GTIOCnA input is 0<br>1: Counter count down enabled on the falling edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 15 | DSCBFAH | GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Count Down Enable<br>0: Counter count down disabled on the falling edge of GTIOCnB input when GTIOCnA input is 1<br>1: Counter count down enabled on the falling edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 16 | DSELCA | ELC_GPTA Event Source Counter Count Down Enable<br>0: Counter count down disabled at the ELC_GPTA input<br>1: Counter count down enabled at the ELC_GPTA input | R/W |
| 17 | DSELCB | ELC_GPTB Event Source Counter Count Down Enable<br>0: Counter count down disabled at the ELC_GPTB input<br>1: Counter count down enabled at the ELC_GPTB input | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 18 | DSELCC | ELC_GPTC Event Source Counter Count Down Enable<br>0: Counter count down disabled at the ELC_GPTC input<br>1: Counter count down enabled at the ELC_GPTC input | R/W |
| 19 | DSELCD | ELC_GPTD Event Source Counter Count Down Enable<br>0: Counter count down disabled at the ELC_GPTD input<br>1: Counter count down enabled at the ELC_GPTD input | R/W |
| 20 | DSELCE | ELC_GPTE Event Source Counter Count Down Enable<br>0: Counter count down disabled at the ELC_GPTE input<br>1: Counter count down enabled at the ELC_GPTE input | R/W |
| 21 | DSELCF | ELC_GPTF Event Source Counter Count Down Enable<br>0: Counter count down disabled at the ELC_GPTF input<br>1: Counter count down enabled at the ELC_GPTF input | R/W |
| 22 | DSELCG | ELC_GPTG Event Source Counter Count Down Enable<br>0: Counter count down disabled at the ELC_GPTG input<br>1: Counter count down enabled at the ELC_GPTG input | R/W |
| 23 | DSELCH | ELC_GPTH Event Source Counter Count Down Enable<br>0: Counter count down disabled at the ELC_GPTH input<br>1: Counter count down enabled at the ELC_GPTH input | R/W |
| 31:24 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTDNSR sets the source to count down the GTCNT counter.

When at least one bit in the GTDNSR register is set to 1, the GTCNT counter is counted down by the source that is set to 1 in this register. In this case, GTCR.TPCS has no effect.

Number of decrement in counting is one even when multiple sources are generated simultaneously.

Inputs from GTETRGn (n = A to D) pins are input to the GPT through the POEG. Set the polarity of these signals with the POEG.

#### DSGTRGAR bit (GTETRGA Pin Rising Input Source Counter Count Down Enable)

The DSGTRGAR bit enables or disables the GTCNT counter count down on the rising edge of the GTETRGA pin input.

#### DSGTRGAF bit (GTETRGA Pin Falling Input Source Counter Count Down Enable)

The DSGTRGAF bit enables or disables the GTCNT counter count down on the falling edge of the GTETRGA pin input.

#### DSGTRGBR bit (GTETRGB Pin Rising Input Source Counter Count Down Enable)

The DSGTRGBR bit enables or disables the GTCNT counter count down on the rising edge of the GTETRGB pin input.

#### DSGTRGBF bit (GTETRGB Pin Falling Input Source Counter Count Down Enable)

The DSGTRGBF bit enables or disables the GTCNT counter count down on the falling edge of the GTETRGB pin input.

#### DSGTRGCR bit (GTETRGC Pin Rising Input Source Counter Count Down Enable)

The DSGTRGCR bit enables or disables the GTCNT counter count down on the rising edge of the GTETRGC pin input.

#### DSGTRGCF bit (GTETRGC Pin Falling Input Source Counter Count Down Enable)

The DSGTRGCF bit enables or disables the GTCNT counter count down on the falling edge of the GTETRGC pin input.

#### DSGTRGDR bit (GTETRGD Pin Rising Input Source Counter Count Down Enable)

The DSGTRGDR bit enables or disables the GTCNT counter count down on the rising edge of the GTETRGD pin input.

#### DSGTRGDF bit (GTETRGD Pin Falling Input Source Counter Count Down Enable)

The DSGTRGDF bit enables or disables the GTCNT counter count down on the falling edge of the GTETRGD pin input.

**DSCARBL bit (GTIOCnA Pin Rising Input during GTIOCnB Value Low Source Counter Count Down Enable)**

The DSCARBL bit enables or disables the GTCNT counter count down on the rising edge of the GTIOCnA pin input, when the GTIOCnB input is 0.

**DSCARBH bit (GTIOCnA Pin Rising Input during GTIOCnB Value High Source Counter Count Down Enable)**

The DSCARBH bit enables or disables the GTCNT counter count down on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**DSCAFBL bit (GTIOCnA Pin Falling Input during GTIOCnB Value Low Source Counter Count Down Enable)**

The DSCAFBL bit enables or disables the GTCNT counter count down on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 0.

**DSCAFBH bit (GTIOCnA Pin Falling Input during GTIOCnB Value High Source Counter Count Down Enable)**

The DSCAFBH bit enables or disables the GTCNT counter count down on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**DSCBRAL bit (GTIOCnB Pin Rising Input during GTIOCnA Value Low Source Counter Count Down Enable)**

The DSCBRAL bit enables or disables the GTCNT counter count down on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 0.

**DSCBRAH bit (GTIOCnB Pin Rising Input during GTIOCnA Value High Source Counter Count Down Enable)**

The DSCBRAH bit enables or disables the GTCNT counter count down on the rising edge of GTIOCnB pin input, when GTIOCnA input is 1.

**DSCBFAL bit (GTIOCnB Pin Falling Input during GTIOCnA Value Low Source Counter Count Down Enable)**

The DSCBFAL bit enables or disables the GTCNT counter count down on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 0.

**DSCBFAH bit (GTIOCnB Pin Falling Input during GTIOCnA Value High Source Counter Count Down Enable)**

The DSCBFAH bit enables or disables the GTCNT counter count down on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 1.

**DSELCm bit (ELC_GPTm Event Source Counter Count Down Enable) (m = A to H)**

The DSELCm bit enables or disables the GTCNT counter count down at the ELC_GPTm event input.

## 21.2.10 GTICASR : General PWM Timer Input Capture Source Select Register A

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x24

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | ASELCH | ASELCG | ASELCF | ASELCE | ASELCD | ASELCC | ASELCB | ASELCA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ASCBFAH | ASCBFAL | ASCBRAH | ASCBRAL | ASCAFBH | ASCAFBL | ASCARBH | ASCARBL | ASGTRGDF | ASGTRGDR | ASGTRGCF | ASGTRGCR | ASGTRGBF | ASGTRGBR | ASGTRGAF | ASGTRGAR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ASGTRGAR | GTETRGA Pin Rising Input Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the rising edge of GTETRGA input<br>1: GTCCRA input capture enabled on the rising edge of GTETRGA input | R/W |
| 1 | ASGTRGAF | GTETRGA Pin Falling Input Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the falling edge of GTETRGA input<br>1: GTCCRA input capture enabled on the falling edge of GTETRGA input | R/W |
| 2 | ASGTRGBR | GTETRGB Pin Rising Input Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the rising edge of GTETRGB input<br>1: GTCCRA input capture enabled on the rising edge of GTETRGB input | R/W |
| 3 | ASGTRGBF | GTETRGB Pin Falling Input Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the falling edge of GTETRGB input<br>1: GTCCRA input capture enabled on the falling edge of GTETRGB input | R/W |
| 4 | ASGTRGCR | GTETRGC Pin Rising Input Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the rising edge of GTETRGC input<br>1: GTCCRA input capture enabled on the rising edge of GTETRGC input | R/W |
| 5 | ASGTRGCF | GTETRGC Pin Falling Input Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the falling edge of GTETRGC input<br>1: GTCCRA input capture enabled on the falling edge of GTETRGC input | R/W |
| 6 | ASGTRGDR | GTETRGD Pin Rising Input Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the rising edge of GTETRGD input<br>1: GTCCRA input capture enabled on the rising edge of GTETRGD input | R/W |
| 7 | ASGTRGDF | GTETRGD Pin Falling Input Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the falling edge of GTETRGD input<br>1: GTCCRA input capture enabled on the falling edge of GTETRGD input | R/W |
| 8 | ASCARBL | GTIOCnA Pin Rising Input during GTIOCnB Value Low Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the rising edge of GTIOCnA input when GTIOCnB input is 0<br>1: GTCCRA input capture enabled on the rising edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 9 | ASCARBH | GTIOCnA Pin Rising Input during GTIOCnB Value High Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the rising edge of GTIOCnA input when GTIOCnB input is 1<br>1: GTCCRA input capture enabled on the rising edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 10 | ASCAFBL | GTIOCnA Pin Falling Input during GTIOCnB Value Low Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the falling edge of GTIOCnA input when GTIOCnB input is 0<br>1: GTCCRA input capture enabled on the falling edge of GTIOCnA input when GTIOCnB input is 0 | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 11 | ASCAFBH | GTIOCnA Pin Falling Input during GTIOCnB Value High Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the falling edge of GTIOCnA input when GTIOCnB input is 1<br>1: GTCCRA input capture enabled on the falling edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 12 | ASCBRAL | GTIOCnB Pin Rising Input during GTIOCnA Value Low Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the rising edge of GTIOCnB input when GTIOCnA input is 0<br>1: GTCCRA input capture enabled on the rising edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 13 | ASCBRAH | GTIOCnB Pin Rising Input during GTIOCnA Value High Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the rising edge of GTIOCnB input when GTIOCnA input is 1<br>1: GTCCRA input capture enabled on the rising edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 14 | ASCBFAL | GTIOCnB Pin Falling Input during GTIOCnA Value Low Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the falling edge of GTIOCnB input when GTIOCnA input is 0<br>1: GTCCRA input capture enabled on the falling edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 15 | ASCBFAH | GTIOCnB Pin Falling Input during GTIOCnA Value High Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled on the falling edge of GTIOCnB input when GTIOCnA input is 1<br>1: GTCCRA input capture enabled on the falling edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 16 | ASELCA | ELC_GPTA Event Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled at the ELC_GPTA input<br>1: GTCCRA input capture enabled at the ELC_GPTA input | R/W |
| 17 | ASELCB | ELC_GPTB Event Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled at the ELC_GPTB input<br>1: GTCCRA input capture enabled at the ELC_GPTB input | R/W |
| 18 | ASELCC | ELC_GPTC Event Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled at the ELC_GPTC input<br>1: GTCCRA input capture enabled at the ELC_GPTC input | R/W |
| 19 | ASELCD | ELC_GPTD Event Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled at the ELC_GPTD input<br>1: GTCCRA input capture enabled at the ELC_GPTD input | R/W |
| 20 | ASELCE | ELC_GPTE Event Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled at the ELC_GPTE input<br>1: GTCCRA input capture enabled at the ELC_GPTE input | R/W |
| 21 | ASELCF | ELC_GPTF Event Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled at the ELC_GPTF input<br>1: GTCCRA input capture enabled at the ELC_GPTF input | R/W |
| 22 | ASELCG | ELC_GPTG Event Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled at the ELC_GPTG input<br>1: GTCCRA input capture enabled at the ELC_GPTG input | R/W |
| 23 | ASELCH | ELC_GPTH Event Source GTCCRA Input Capture Enable<br>0: GTCCRA input capture disabled at the ELC_GPTH input<br>1: GTCCRA input capture enabled at the ELC_GPTH input | R/W |
| 31:24 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTICASR sets the source of input capture for GTCCRA.

When at least one bit among bits in the GTICASR register is set to 1, input capture operation making the GTCCRA register as an input capture register is performed.

Inputs from GTETRGn (n = A to D) pins are input to the GPT through the POEG. Set the polarity of these signals with the POEG.

**ASGTRGAR bit (GTETRGA Pin Rising Input Source GTCCRA Input Capture Enable)**

The ASGTRGAR bit enables or disables the input capture for GTCCRA on the rising edge of the GTETRGA pin input.

**ASGTRGAF bit (GTETRGA Pin Falling Input Source GTCCRA Input Capture Enable)**

The ASGTRGAF bit enables or disables the input capture for GTCCRA on the falling edge of the GTETRGA pin input.

**ASGTRGBR bit (GTETRGB Pin Rising Input Source GTCCRA Input Capture Enable)**

The ASGTRGBR bit enables or disables the input capture for GTCCRA on the rising edge of the GTETRGB pin input.

**ASGTRGBF bit (GTETRGB Pin Falling Input Source GTCCRA Input Capture Enable)**

The ASGTRGBF bit enables or disables the input capture for GTCCRA on the falling edge of the GTETRGB pin input.

**ASGTRGCR bit (GTETRGC Pin Rising Input Source GTCCRA Input Capture Enable)**

The ASGTRGCR bit enables or disables the input capture for GTCCRA on the rising edge of the GTETRGC pin input.

**ASGTRGCF bit (GTETRGC Pin Falling Input Source GTCCRA Input Capture Enable)**

The ASGTRGCF bit enables or disables the input capture for GTCCRA on the falling edge of the GTETRGC pin input.

**ASGTRGDR bit (GTETRGD Pin Rising Input Source GTCCRA Input Capture Enable)**

The ASGTRGDR bit enables or disables the input capture for GTCCRA on the rising edge of the GTETRGD pin input.

**ASGTRGDF bit (GTETRGD Pin Falling Input Source GTCCRA Input Capture Enable)**

The ASGTRGDF bit enables or disables the input capture for GTCCRA on the falling edge of the GTETRGD pin input.

**ASCARBL bit (GTIOCnA Pin Rising Input during GTIOCnB Value Low Source GTCCRA Input Capture Enable)**

The ASCARBL bit enables or disables the input capture for GTCCRA on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 0.

**ASCARBH bit (GTIOCnA Pin Rising Input during GTIOCnB Value High Source GTCCRA Input Capture Enable)**

The ASCARBH bit enables or disables the input capture for GTCCRA on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**ASCAFBL bit (GTIOCnA Pin Falling Input during GTIOCnB Value Low Source GTCCRA Input Capture Enable)**

The ASCAFBL bit enables or disables the input capture for GTCCRA on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 0.

**ASCAFBH bit (GTIOCnA Pin Falling Input during GTIOCnB Value High Source GTCCRA Input Capture Enable)**

The ASCAFBH bit enables or disables the input capture for GTCCRA on the falling edge of the GTIOCnA pin input, when the GTIOCnB input is 1.

**ASCBRAL bit (GTIOCnB Pin Rising Input during GTIOCnA Value Low Source GTCCRA Input Capture Enable)**

The ASCBRAL bit enables or disables the input capture for GTCCRA on the rising edge of the GTIOCnB pin input, when the GTIOCnA input is 0.

**ASCBRAH bit (GTIOCnB Pin Rising Input during GTIOCnA Value High Source GTCCRA Input Capture Enable)**

The ASCBRAH bit enables or disables the input capture for GTCCRA on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 1.

### ASCBFAL bit (GTIOCnB Pin Falling Input during GTIOCnA Value Low Source GTCCRA Input Capture Enable)

The ASCBFAL bit enables or disables the input capture for GTCCRA on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 0.

### ASCBFAH bit (GTIOCnB Pin Falling Input during GTIOCnA Value High Source GTCCRA Input Capture Enable)

The ASCBFAH bit enables or disables the input capture for GTCCRA on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 1.

### ASELCm bit (ELC_GPTm Event Source Counter GTCCRA Input Capture Enable) (m = A to H)

The ASELCm bit enables or disables the input capture for GTCCRA at the ELC_GPTm event input.

## 21.2.11 GTICBSR : General PWM Timer Input Capture Source Select Register B

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x28

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | BSELCH | BSELCG | BSELCF | BSELCE | BSELCD | BSELCC | BSELCB | BSELCA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BSCBFAH | BSCBFAL | BSCBRAH | BSCBRAL | BSCAFBH | BSCAFBL | BSCARBH | BSCARBL | BSGTRGDF | BSGTRGDR | BSGTRGCF | BSGTRGCR | BSGTRGBF | BSGTRGBR | BSGTRGAF | BSGTRGAR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BSGTRGAR | GTETRGA Pin Rising Input Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the rising edge of GTETRGA input<br>1: GTCCRB input capture enabled on the rising edge of GTETRGA input | R/W |
| 1 | BSGTRGAF | GTETRGA Pin Falling Input Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the falling edge of GTETRGA input<br>1: GTCCRB input capture enabled on the falling edge of GTETRGA input | R/W |
| 2 | BSGTRGBR | GTETRGB Pin Rising Input Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the rising edge of GTETRGB input<br>1: GTCCRB input capture enabled on the rising edge of GTETRGB input | R/W |
| 3 | BSGTRGBF | GTETRGB Pin Falling Input Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the falling edge of GTETRGB input<br>1: GTCCRB input capture enabled on the falling edge of GTETRGB input | R/W |
| 4 | BSGTRGCR | GTETRGC Pin Rising Input Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the rising edge of GTETRGC input<br>1: GTCCRB input capture enabled on the rising edge of GTETRGC input | R/W |
| 5 | BSGTRGCF | GTETRGC Pin Falling Input Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the falling edge of GTETRGC input<br>1: GTCCRB input capture enabled on the falling edge of GTETRGC input | R/W |
| 6 | BSGTRGDR | GTETRGD Pin Rising Input Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the rising edge of GTETRGD input<br>1: GTCCRB input capture enabled on the rising edge of GTETRGD input | R/W |
| 7 | BSGTRGDF | GTETRGD Pin Falling Input Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the falling edge of GTETRGD input<br>1: GTCCRB input capture enabled on the falling edge of GTETRGD input | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8 | BSCARBL | GTIOCnA Pin Rising Input during GTIOCnB Value Low Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the rising edge of GTIOCnA input when GTIOCnB input is 0<br>1: GTCCRB input capture enabled on the rising edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 9 | BSCARBH | GTIOCnA Pin Rising Input during GTIOCnB Value High Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the rising edge of GTIOCnA input when GTIOCnB input is 1<br>1: GTCCRB input capture enabled on the rising edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 10 | BSCAFBL | GTIOCnA Pin Falling Input during GTIOCnB Value Low Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the falling edge of GTIOCnA input when GTIOCnB input is 0<br>1: GTCCRB input capture enabled on the falling edge of GTIOCnA input when GTIOCnB input is 0 | R/W |
| 11 | BSCAFBH | GTIOCnA Pin Falling Input during GTIOCnB Value High Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the falling edge of GTIOCnA input when GTIOCnB input is 1<br>1: GTCCRB input capture enabled on the falling edge of GTIOCnA input when GTIOCnB input is 1 | R/W |
| 12 | BSCBRAL | GTIOCnB Pin Rising Input during GTIOCnA Value Low Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the rising edge of GTIOCnB input when GTIOCnA input is 0<br>1: GTCCRB input capture enabled on the rising edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 13 | BSCBRAH | GTIOCnB Pin Rising Input during GTIOCnA Value High Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the rising edge of GTIOCnB input when GTIOCnA input is 1<br>1: GTCCRB input capture enabled on the rising edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 14 | BSCBFAL | GTIOCnB Pin Falling Input during GTIOCnA Value Low Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the falling edge of GTIOCnB input when GTIOCnA input is 0<br>1: GTCCRB input capture enabled on the falling edge of GTIOCnB input when GTIOCnA input is 0 | R/W |
| 15 | BSCBFAH | GTIOCnB Pin Falling Input during GTIOCnA Value High Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled on the falling edge of GTIOCnB input when GTIOCnA input is 1<br>1: GTCCRB input capture enabled on the falling edge of GTIOCnB input when GTIOCnA input is 1 | R/W |
| 16 | BSELCA | ELC_GPTA Event Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled at the ELC_GPTA input<br>1: GTCCRB input capture enabled at the ELC_GPTA input | R/W |
| 17 | BSELCB | ELC_GPTB Event Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled at the ELC_GPTB input<br>1: GTCCRB input capture enabled at the ELC_GPTB input | R/W |
| 18 | BSELCC | ELC_GPTC Event Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled at the ELC_GPTC input<br>1: GTCCRB input capture enabled at the ELC_GPTC input | R/W |
| 19 | BSELCD | ELC_GPTD Event Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled at the ELC_GPTD input<br>1: GTCCRB input capture enabled at the ELC_GPTD input | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 20 | BSELCE | ELC_GPTE Event Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled at the ELC_GPTE input<br>1: GTCCRB input capture enabled at the ELC_GPTE input | R/W |
| 21 | BSELCF | ELC_GPTF Event Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled at the ELC_GPTF input<br>1: GTCCRB input capture enabled at the ELC_GPTF input | R/W |
| 22 | BSELCG | ELC_GPTG Event Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled at the ELC_GPTG input<br>1: GTCCRB input capture enabled at the ELC_GPTG input | R/W |
| 23 | BSELCH | ELC_GPTH Event Source GTCCRB Input Capture Enable<br>0: GTCCRB input capture disabled at the ELC_GPTH input<br>1: GTCCRB input capture enabled at the ELC_GPTH input | R/W |
| 31:24 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTICBSR sets the source of input capture for GTCCRB.

When at least one bit among bits in the GTICBSR register is set to 1, input capture operation making the GTCCRB register as an input capture register is performed.

Inputs from GTETRGn (n = A to D) pins are input to the GPT through the POEG. Set the polarity of these signals with the POEG.

#### BSGTRGAR bit (GTETRGA Pin Rising Input Source GTCCRB Input Capture Enable)

The BSGTRGAR bit enables or disables the input capture for GTCCRB on the rising edge of the GTETRGA pin input.

#### BSGTRGAF bit (GTETRGA Pin Falling Input Source GTCCRB Input Capture Enable)

The BSGTRGAF bit enables or disables the input capture for GTCCRB on the falling edge of the GTETRGA pin input.

#### BSGTRGBR bit (GTETRGB Pin Rising Input Source GTCCRB Input Capture Enable)

The BSGTRGBR bit enables or disables the input capture for GTCCRB on the rising edge of GTETRGB pin input.

#### BSGTRGBF bit (GTETRGB Pin Falling Input Source GTCCRB Input Capture Enable)

The BSGTRGBF bit enables or disables the input capture for GTCCRB on the falling edge of the GTETRGB pin input.

#### BSGTRGCR bit (GTETRGC Pin Rising Input Source GTCCRB Input Capture Enable)

The BSGTRGCR bit enables or disables the input capture for GTCCRB on the rising edge of GTETRGC pin input.

#### BSGTRGCF bit (GTETRGC Pin Falling Input Source GTCCRB Input Capture Enable)

The BSGTRGCF bit enables or disables the input capture for GTCCRB on the falling edge of the GTETRGC pin input.

#### BSGTRGDR bit (GTETRGD Pin Rising Input Source GTCCRB Input Capture Enable)

The BSGTRGDR bit enables or disables the input capture for GTCCRB on the rising edge of GTETRGD pin input.

#### BSGTRGDF bit (GTETRGD Pin Falling Input Source GTCCRB Input Capture Enable)

The BSGTRGDF bit enables or disables the input capture for GTCCRB on the falling edge of the GTETRGD pin input.

#### BSCARBL bit (GTIOCnA Pin Rising Input during GTIOCnB Value Low Source GTCCRB Input Capture Enable)

The BSCARBL bit enables or disables the input capture for GTCCRB on the rising edge of the GTIOCnA pin input, when the GTIOCnB input is 0.

#### BSCARBH bit (GTIOCnA Pin Rising Input during GTIOCnB Value High Source GTCCRB Input Capture Enable)

The BSCARBH bit enables or disables the input capture for GTCCRB on the rising edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**BSCAFBL bit (GTIOCnA Pin Falling Input during GTIOCnB Value Low Source GTCCRB Input Capture Enable)**

The BSCAFBL bit enables or disables the input capture for GTCCRB on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 0.

**BSCAFBH bit (GTIOCnA Pin Falling Input during GTIOCnB Value High Source GTCCRB Input Capture Enable)**

The BSCAFBH bit enables or disables the input capture for GTCCRB on the falling edge of the GTIOCnA pin input, when GTIOCnB input is 1.

**BSCBRAL bit (GTIOCnB Pin Rising Input during GTIOCnA Value Low Source GTCCRB Input Capture Enable)**

The BSCBRAL bit enables or disables the input capture for GTCCRB on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 0.

**BSCBRAH bit (GTIOCnB Pin Rising Input during GTIOCnA Value High Source GTCCRB Input Capture Enable)**

The BSCBRAH bit enables or disables the input capture for GTCCRB on the rising edge of the GTIOCnB pin input, when GTIOCnA input is 1.

**BSCBFAL bit (GTIOCnB Pin Falling Input during GTIOCnA Value Low Source GTCCRB Input Capture Enable)**

The BSCBFAL bit enables or disables the input capture for GTCCRB on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 0.

**BSCBFAH bit (GTIOCnB Pin Falling Input during GTIOCnA Value High Source GTCCRB Input Capture Enable)**

The BSCBFAH bit enables or disables the input capture for GTCCRB on the falling edge of the GTIOCnB pin input, when GTIOCnA input is 1.

**BSELCm bit (ELC_GPTm Event Source Counter GTCCRB Input Capture Enable) (m = A to H)**

The BSELCm bit enables or disables the input capture for GTCCRB at the ELC_GPTm event input.

## 21.2.12 GTCR : General PWM Timer Control Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x2C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | TPCS[3:0] | | | | — | — | — | — | MD[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | CST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CST | Count Start<br>0: Count operation is stopped<br>1: Count operation is performed | R/W |
| 15:1 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 18:16 | MD[2:0] | Mode Select<br>0 0 0: Saw-wave PWM mode (single buffer or double buffer possible)<br>0 0 1: Saw-wave one-shot pulse mode (fixed buffer operation)<br>0 1 0: Setting prohibited<br>0 1 1: Setting prohibited<br>1 0 0: Triangle-wave PWM mode 1 (32-bit transfer at trough) (single buffer or double buffer is possible)<br>1 0 1: Triangle-wave PWM mode 2 (32-bit transfer at crest and trough) (single buffer or double buffer is possible)<br>1 1 0: Triangle-wave PWM mode 3 (64-bit transfer at trough) (fixed buffer operation)<br>1 1 1: Setting prohibited | R/W |
| 22:19 | — | These bits are read as 0. The write value should be 0. | R/W |
| 26:23 | TPCS[3:0] | Timer Prescaler Select<br>0 0 0 0: PCLKD/1<br>0 0 0 1: PCLKD/2<br>0 0 1 0: PCLKD/4<br>0 0 1 1: PCLKD/8<br>0 1 0 0: PCLKD/16<br>0 1 0 1: PCLKD/32<br>0 1 1 0: PCLKD/64<br>0 1 1 1: Setting prohibited<br>1 0 0 0: PCLKD/256<br>1 0 0 1: Setting prohibited<br>1 0 1 0: PCLKD/1024<br>1 0 1 1: Setting prohibited<br>1 1 0 0: GTETRGA (Via the POEG)<br>1 1 0 1: GTETRGB (Via the POEG)<br>1 1 1 0: GTETRGC (Via the POEG)<br>1 1 1 1: GTETRGD (Via the POEG) | R/W |
| 31:27 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTCR controls GTCNT.

### CST bit (Count Start)

The CST bit controls the GTCNT counter start and stop.

[Setting conditions]

- The GTSTR value where the channel number associated with the bit number is set to 1 with the GTSSR.CSTRT bit at 1
- The ELC event input, the external trigger, or the GTIOCnA/GTIOCnB input that are enabled by GTSSR for the starting counter source, occurs (n = 0 to 7)
- 1 is written by software directly.

[Clearing conditions]

- The GTSTP value where the channel number associated with the bit number is set to 1 with the GTPSR.CSTOP bit at 1
- The ELC event input, the external trigger, or the GTIOCnA/GTIOCnB input enabled by GTPSR as the counter stop source, occurs (n = 0 to 7)
- 0 is written by software directly.
- When the period count function is finished while the GTPC.ASTP bit is 1.

### MD[2:0] bits (Mode Select)

The MD[2:0] bits select the GPT operating mode. The MD[2:0] bits must be set while the GTCNT operation is stopped.

### TPCS[3:0] bits (Timer Prescaler Select)

The TPCS[3:0] bits select the clock for GTCNT. A clock prescaler can be selected independently for each channel. The TPCS[3:0] bits must be set while the GTCNT operation is stopped.

## 21.2.13 GTUDDTYC : General PWM Timer Count Direction and Duty Setting Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x30

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | OBDTYR | OBDTYF | OBDTY[1:0] | | — | — | — | — | OADTYR | OADTYF | OADTY[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | UDF | UD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | UD | Count Direction Setting<br>0: GTCNT counts down<br>1: GTCNT counts up | R/W |
| 1 | UDF | Forcible Count Direction Setting<br>0: Not forcibly set<br>1: Forcibly set | R/W |
| 15:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 17:16 | OADTY[1:0] | GTIOCnA Output Duty Setting<br>0 0: GTIOCnA pin duty depends on the compare match<br>0 1: GTIOCnA pin duty depends on the compare match<br>1 0: GTIOCnA pin duty 0%<br>1 1: GTIOCnA pin duty 100% | R/W |
| 18 | OADTYF | Forcible GTIOCnA Output Duty Setting<br>0: Not forcibly set<br>1: Forcibly set | R/W |
| 19 | OADTYR | GTIOCnA Output Value Selecting after Releasing 0%/100% Duty Setting<br>0: The function selected by the GTIOA[3:2] bits is applied to the output value when the duty cycle is set after release from the 0 or 100% duty-cycle setting.<br>1: The function selected by the GTIOA[3:2] bits is applied to the compare match output value which is masked after release from the 0 or 100% duty-cycle setting. | R/W |
| 23:20 | — | These bits are read as 0. The write value should be 0. | R/W |
| 25:24 | OBDTY[1:0] | GTIOCnB Output Duty Setting<br>0 0: GTIOCnB pin duty depends on the compare match<br>0 1: GTIOCnB pin duty depends on the compare match<br>1 0: GTIOCnB pin duty 0%<br>1 1: GTIOCnB pin duty 100% | R/W |
| 26 | OBDTYF | Forcible GTIOCnB Output Duty Setting<br>0: Not forcibly set<br>1: Forcibly set | R/W |
| 27 | OBDTYR | GTIOCnB Output Value Selecting after Releasing 0%/100% Duty Setting<br>0: The function selected by the GTIOB[3:2] bits is applied to the output value when the duty cycle is set after release from the 0 or 100% duty-cycle setting.<br>1: The function selected by the GTIOB[3:2] bits is applied to the compare match output value which is masked after release from the 0 or 100% duty-cycle setting. | R/W |
| 31:28 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: n = 0 to 7

The GTUDDTYC sets the direction in which the GTCNT counts (up-counting or down-counting), and sets the duty of the GTIOCnA/GTIOCnB pin output.

The setting is invalid during the event count operation.

Count Direction:

- In saw-wave mode.
  When the UD value is set to 0 during up-counting, the count direction changes at an overflow (the timing synchronous with count clock after the GTCNT value becomes the GTPR value). When the UD value is set to 1 during down-counting, the count direction changes at an underflow (the timing synchronous with count clock after the GTCNT value becomes 0).
  When the UD value changes from 1 to 0 with the UDF bit being 0 and while counting stops, the counter starts up-counting and the count direction changes at an overflow (the timing synchronous with count clock after the GTCNT value becomes the GTPR value). When the UD value changes from 0 to 1 with the UDF bit being 0 and while counting stops, the counter starts down-counting and the count direction changes at an underflow (the timing synchronous with count clock after the GTCNT value becomes 0).
  When the UDF bit is set to 1 while counting stops, the UD bit value is reflected in the count direction when counting starts.
- In triangle-wave mode.
  When the UD value changes during counting, the count direction does not change. When the UD value changes while the UDF bit is 0 and counting stops, the change is not reflected in the count direction when counting starts.
  When the UDF bit is set to 1 while counting is stopped, the UD value is reflected in the count direction when counting starts.

#### UD bit (Count Direction Setting)

The UD bit sets the count direction (up-counting or down-counting) for GTCNT.

#### UDF bit (Forcible Count Direction Setting)

The UDF bit forcibly sets the count direction when GTCNT starts operation as the UD value. Only 0 should be written to this bit during counter operation. When 1 is written to this bit while counting stops, return this bit to 0 before counting starts.

Output duty

- In saw-wave mode.
  When the OADTY/OBDTY value changes during up-counting, the duty is reflected at an overflow (GTCNT = GTPR).
  When the OADTY/OBDTY value is changed during down-counting, the duty is reflected at an underflow (GTCNT = 0).
  When the OADTY/OBDTY value is changed with the OADTYF/OBDTYF bit being 0 and while counting stops the output duty is not reflected at the starting counter operation. When the count direction is up, the output duty is reflected at an overflow (GTCNT = GTPR). When the count direction is down, the output duty is reflected at an underflow (GTCNT = 0).
  When the OADTY/OBDTY value is changed with the OADTYF/OBDTYF bit being 1 and while counting stops, the output duty is reflected at starting counter operation.
- In triangle-wave mode.
  When the OADTY/OBDTY value changes during counting, the duty is reflected at an underflow.
  When the OADTY/OBDTY value is changed with the OADTYF/OBDTYF bit being 0 and while counting stops, the output duty is not reflected at the starting counter operation. The output duty is reflected at an underflow.
  When the OADTY/OBDTY value is changed with the OADTYF/OBDTYF bit being 1 and while counting stops, the output duty is reflected at starting counter operation.

In both saw-wave mode and triangle-wave mode, when the OADTYF/OBDTYF bit is set back to 0 and the OADTY[1:0]/OBDTY[1:0] bits are set after setting the OADTYF/OBDTYF bit to 1 and setting the OADTY[1:0]/OBDTY[1:0] bits for the duty of first cycle while count operation is stopped, these duty-cycle set during stopping count operation are reflected in the first cycle and the second cycle after starting count operation.

#### OmDTY[1:0] bits (GTIOCnm Output Duty Setting) (m = A, B)

The OmDTY[1:0] bits set the output duty (0%, 100% or compare match control) of the GTIOCnm pin.

#### OmDTYF bit (Forcible GTIOCnm Output Duty Setting) (m = A, B)

The OmDTYF bit forcibly sets the output duty cycle to the OmDTY setting. Set this bit to 0 during counter operation.

**OmDTYR bit (GTIOCnm Output Value Selecting after Releasing 0%/100% Duty Setting) (m = A, B)**

The OmDTYR bit selects the value that is the object of output retained or toggled at cycle end, when the control changes from 0% or 100% duty setting to compare match for the GTIOCnm pin and GTIOR.GTIOm[3:2] bits are set to 00b (output retained at cycle end) or the GTIOR.GTIOm[3:2] bits are set to 11b (output toggled at cycle end).

The GPT internally continues to perform compare match operation during duty-cycle 0% or 100% operation. When the OmDTYR bit is 1, the value after the period has elapsed due this compare match operation is target for the GTIOm[3:2] bits.

## 21.2.14 GTIOR : General PWM Timer I/O Control Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x34

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | NFCSB[1:0] | | NFBEN | — | — | OBDF[1:0] | | OBE | OBHLD | OBDFLT | — | GTIOB[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | NFCSA[1:0] | | NFAEN | — | — | OADF[1:0] | | OAE | OAHLD | OADFLT | — | GTIOA[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | GTIOA[4:0] | GTIOCnA Pin Function Select<br>See Table 21.4. | R/W |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | OADFLT | GTIOCnA Pin Output Value Setting at the Count Stop<br>0: The GTIOCnA pin outputs low when counting stops<br>1: The GTIOCnA pin outputs high when counting stops | R/W |
| 7 | OAHLD | GTIOCnA Pin Output Setting at the Start/Stop Count<br>0: The GTIOCnA pin output level at the start or stop of counting depends on the register setting<br>1: The GTIOCnA pin output level is retained at the start or stop of counting | R/W |
| 8 | OAE | GTIOCnA Pin Output Enable<br>0: Output is disabled<br>1: Output is enabled | R/W |
| 10:9 | OADF[1:0] | GTIOCnA Pin Disable Value Setting<br>0 0: None of the below options are specified<br>0 1: GTIOCnA pin is set to Hi-Z in response to controlling the output negation<br>1 0: GTIOCnA pin is set to 0 in response to controlling the output negation<br>1 1: GTIOCnA pin is set to 1 in response to controlling the output negation | R/W |
| 12:11 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13 | NFAEN | Noise Filter A Enable<br>0: The noise filter for the GTIOCnA pin is disabled<br>1: The noise filter for the GTIOCnA pin is enabled | R/W |
| 15:14 | NFCSA[1:0] | Noise Filter A Sampling Clock Select<br>0 0: PCLKD/1<br>0 1: PCLKD/4<br>1 0: PCLKD/16<br>1 1: PCLKD/64 | R/W |
| 20:16 | GTIOB[4:0] | GTIOCnB Pin Function Select<br>See Table 21.4. | R/W |
| 21 | — | This bit is read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 22 | OBDFLT | GTIOCnB Pin Output Value Setting at the Count Stop<br>0: The GTIOCnB pin outputs low when counting stops<br>1: The GTIOCnB pin outputs high when counting stops | R/W |
| 23 | OBHLD | GTIOCnB Pin Output Setting at the Start/Stop Count<br>0: The GTIOCnB pin output level at the start/stop of counting depends on the register setting<br>1: The GTIOCnB pin output level is retained at the start/stop of counting | R/W |
| 24 | OBE | GTIOCnB Pin Output Enable<br>0: Output is disabled<br>1: Output is enabled | R/W |
| 26:25 | OBDF[1:0] | GTIOCnB Pin Disable Value Setting<br>0 0: None of the below options are specified<br>0 1: GTIOCnB pin is set to Hi-Z in response to controlling the output negation<br>1 0: GTIOCnB pin is set to 0 in response to controlling the output negation<br>1 1: GTIOCnB pin is set to 1 in response to controlling the output negation | R/W |
| 28:27 | — | These bits are read as 0. The write value should be 0. | R/W |
| 29 | NFBEN | Noise Filter B Enable<br>0: The noise filter for the GTIOCnB pin is disabled<br>1: The noise filter for the GTIOCnB pin is enabled | R/W |
| 31:30 | NFCSB[1:0] | Noise Filter B Sampling Clock Select<br>0 0: PCLKD/1<br>0 1: PCLKD/4<br>1 0: PCLKD/16<br>1 1: PCLKD/64 | R/W |

Note: n = 0 to 7

The GTIOR sets the functions of the GTIOCnA and GTIOCnB pins. (n = 0 to 7)

### GTIOA[4:0] bits (GTIOCnA Pin Function Select)

The GTIOA[4:0] bits select the GTIOCnA pin function. For details, see Table 21.4.

### OADFLT bit (GTIOCnA Pin Output Value Setting at the Count Stop)

The OADFLT bit sets whether the GTIOCnA pin outputs high or low when counting stops.

### OAHLD bit (GTIOCnA Pin Output Setting at the Start/Stop Count)

The OAHLD bit specifies whether the GTIOCnA pin output level is retained or the level at the start or stop of counting depends on the register setting.

When the OAHLD bit is set to 0:

- The value specified in bit [4] of the GTIOA[4:0] bits is output when counting starts
- The value specified in the OADFLT bit is output when counting stops
- If the OADFLT bit is modified while counting stops, the new value is immediately reflected in the output.

When the OAHLD bit is set to 1:

- The output is retained when counting starts or stops.

### OAE bit (GTIOCnA Pin Output Enable)

The OAE bit disables or enables the GTIOCnA pin output.

When GTCCRA register is used as the input capture register (at least one bit in the GTICASR register is set to 1), the GTIOCnA pin does not output regardless of the OAE bit value.

### OADF[1:0] bits (GTIOCnA Pin Disable Value Setting)

The OADF[1:0] bits select the output value of the GTIOCnA pin in response to a request to disable output from the POEG.

#### NFAEN bit (Noise Filter A Enable)

The NFAEN bit disables or enables the noise filter for input from the GTIOCnA pin. Because changing the value of the bit might lead to the internal generation of an unexpected edge, select the output compare function for the relevant pin in the GTIOR register before doing so.

#### NFCSA[1:0] bits (Noise Filter A Sampling Clock Select)

The NFCSA[1:0] bits set the sampling interval for the noise filter of the GTIOCnA pin. When setting these bits, wait for 2 cycles of the selected sampling interval before setting the input capture function.

#### GTIOB[4:0] bits (GTIOCnB Pin Function Select)

The GTIOB[4:0] bits select the GTIOCnB pin function. For details, see Table 21.4.

#### OBDFLT bit (GTIOCnB Pin Output Value Setting at the Count Stop)

The OBDFLT bit sets whether the GTIOCnB pin outputs high or low when counting stops.

#### OBHLD bit (GTIOCnB Pin Output Setting at the Start/Stop Count)

The OBHLD bit specifies whether theGTIOCnB pin output level is retained or the level at the start or stop of counting depends on the register setting.

When the OBHLD bit is set to 0:

- The value specified in bit [4] of the GTIOB[4:0] bits is output when counting starts
- The value specified in the OBDFLT bit is output when counting stops
- If the OBDFLT bit is modified while counting stops, the new value is immediately reflected in the output.

When the OBHLD bit is set to 1:

- The output is retained when counting starts or stops.

#### OBE bit (GTIOCnB Pin Output Enable)

The OBE bit disables or enables the GTIOCnB pin output.

When GTCCRB register is used as the input capture register (at least one bit in the GTICBSR register is set to 1), the GTIOCnB pin does not output regardless of the OBE bit value.

#### OBDF[1:0] bits (GTIOCnB Pin Disable Value Setting)

The OBDF[1:0] bits select the output value of the GTIOCnB pin in response to a request to disable output from the POEG.

#### NFBEN bit (Noise Filter B Enable)

The NFBEN bit disables or enables the noise filter for input from the GTIOCnB pin. Because changing the value of the bit might lead to the internal generation of an unexpected edge, select the output compare function for the relevant pin in the GTIOR register before doing so.

#### NFCSB[1:0] bits (Noise Filter B Sampling Clock Select)

The NFCSB[1:0] bits set the sampling interval for the noise filter of the GTIOCnB pin. When setting these bits, wait for 2 cycles of the selected sampling interval before setting the input capture function.

**Table 21.4 Settings of GTIOA[4:0] and GTIOB[4:0] bits**

| GTIOA/GTIOB[4:0] bits | | | | | Function | | |
|---|---|---|---|---|---|---|---|
| b4 | b3 | b2 | b1 | b0 | b4 | b3, b2*1 *2 *3 | b1, b0*2 |
| 0 | 0 | 0 | 0 | 0 | Initial output is low | Output retained at cycle end | Output retained at GTCCRA/GTCCRB compare match |
| 0 | 0 | 0 | 0 | 1 | | | Low output at GTCCRA/GTCCRB compare match |
| 0 | 0 | 0 | 1 | 0 | | | High output at GTCCRA/GTCCRB compare match |
| 0 | 0 | 0 | 1 | 1 | | | Output toggled at GTCCRA/GTCCRB compare match |
| 0 | 0 | 1 | 0 | 0 | | Low output at cycle end | Output retained at GTCCRA/GTCCRB compare match |
| 0 | 0 | 1 | 0 | 1 | | | Low output at GTCCRA/GTCCRB compare match |
| 0 | 0 | 1 | 1 | 0 | | | High output at GTCCRA/GTCCRB compare match |
| 0 | 0 | 1 | 1 | 1 | | | Output toggled at GTCCRA/GTCCRB compare match |
| 0 | 1 | 0 | 0 | 0 | | High output at cycle end | Output retained at GTCCRA/GTCCRB compare match |
| 0 | 1 | 0 | 0 | 1 | | | Low output at GTCCRA/GTCCRB compare match |
| 0 | 1 | 0 | 1 | 0 | | | High output at GTCCRA/GTCCRB compare match |
| 0 | 1 | 0 | 1 | 1 | | | Output toggled at GTCCRA/GTCCRB compare match |
| 0 | 1 | 1 | 0 | 0 | | Output toggled at cycle end | Output retained at GTCCRA/GTCCRB compare match |
| 0 | 1 | 1 | 0 | 1 | | | Low output at GTCCRA/GTCCRB compare match |
| 0 | 1 | 1 | 1 | 0 | | | High output at GTCCRA/GTCCRB compare match |
| 0 | 1 | 1 | 1 | 1 | | | Output toggled at GTCCRA/GTCCRB compare match |
| 1 | 0 | 0 | 0 | 0 | Initial output is high | Output retained at cycle end | Output retained at GTCCRA/GTCCRB compare match |
| 1 | 0 | 0 | 0 | 1 | | | Low output at GTCCRA/GTCCRB compare match |
| 1 | 0 | 0 | 1 | 0 | | | High output at GTCCRA/GTCCRB compare match |
| 1 | 0 | 0 | 1 | 1 | | | Output toggled at GTCCRA/GTCCRB compare match |
| 1 | 0 | 1 | 0 | 0 | | Low output at cycle end | Output retained at GTCCRA/GTCCRB compare match |
| 1 | 0 | 1 | 0 | 1 | | | Low output at GTCCRA/GTCCRB compare match |
| 1 | 0 | 1 | 1 | 0 | | | High output at GTCCRA/GTCCRB compare match |
| 1 | 0 | 1 | 1 | 1 | | | Output toggled at GTCCRA/GTCCRB compare match |
| 1 | 1 | 0 | 0 | 0 | | High output at cycle end | Output retained at GTCCRA/GTCCRB compare match |
| 1 | 1 | 0 | 0 | 1 | | | Low output at GTCCRA/GTCCRB compare match |
| 1 | 1 | 0 | 1 | 0 | | | High output at GTCCRA/GTCCRB compare match |
| 1 | 1 | 0 | 1 | 1 | | | Output toggled at GTCCRA/GTCCRB compare match |
| 1 | 1 | 1 | 0 | 0 | | Output toggled at cycle end | Output retained at GTCCRA/GTCCRB compare match |
| 1 | 1 | 1 | 0 | 1 | | | Low output at GTCCRA/GTCCRB compare match |
| 1 | 1 | 1 | 1 | 0 | | | High output at GTCCRA/GTCCRB compare match |
| 1 | 1 | 1 | 1 | 1 | | | Output toggled at GTCCRA/GTCCRB compare match |

Note 1. The cycle end means an overflow (GTCNT changes from GTPR to 0 in up-counting), an underflow (GTCNT changes from 0 to GTPR in down-counting), or counter clearing for saw-wave mode, and means a trough (GTCNT changes from 0 to 1) for triangle-wave mode.

Note 2. When the timing of a cycle end and the timing of a GTCCRA/GTCCRB compare match are the same in a compare-match operation, the b3 and b2 settings are given priority in saw-wave PWM mode, and the b1 and b0 settings are given priority in any other mode.

Note 3. In event count operation where at least one bit in GTUPSR or GTDNSR is set to 1, the setting of b3 and b2 is ignored.

## 21.2.15 GTINTAD : General PWM Timer Interrupt Output Setting Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x38

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | GRPABL | GRPABH | — | — | — | GRP[1:0] | | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 23:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 25:24 | GRP[1:0] | Output Disable Source Select<br>0 0: Group A output disable source is selected<br>0 1: Group B output disable source is selected<br>1 0: Group C output disable source is selected<br>1 1: Group D output disable source is selected | R/W |
| 28:26 | — | These bits are read as 0. The write value should be 0. | R/W |
| 29 | GRPABH | Same Time Output Level High Disable Request Enable<br>0: Same time output level high disable request disabled<br>1: Same time output level high disable request enabled | R/W |
| 30 | GRPABL | Same Time Output Level Low Disable Request Enable<br>0: Same time output level low disable request disabled<br>1: Same time output level low disable request enabled | R/W |
| 31 | — | This bit is read as 0. The write value should be 0. | R/W |

The GTINTAD enables or disables interrupt requests and output disable requests.

### GRP[1:0] bits (Output Disable Source Select)

These bits select the group of output disable request from GPT to POEG and the group of output disable for GTIOCnA pin and GTIOCnB pin from POEG to GPT.

The output disable request to POEG is output to the group selected in the GRP[1:0] bit, with dead-time errors, simultaneous high output, and simultaneous low output factors following their respective disable request enable bits.

GTST.ODF shows the request of the output disable source group that is selected with the GRP[1:0] bits. Set the GRP[1:0] bits when both GTIOR.OAE and GTIOR.OBE bits are 0.

### GRPABH bit (Same Time Output Level High Disable Request Enable)

The GRPABH bit enables or disables the output disable request when the GTIOCnA pin and GTIOCnB pin output 1 at the same time.

### GRPABL bit (Same Time Output Level Low Disable Request Enable)

The GRPABL bit enables or disables the output disable request when the GTIOCnA pin and GTIOCnB pin output 0 at the same time.

## 21.2.16 GTST : General PWM Timer Status Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x3C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PCF | OABLF | OABHF | — | — | — | — | ODF | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | TUCF | — | — | — | — | — | — | — | TCFPU | TCFPO | TCFF | TCFE | TCFD | TCFC | TCFB | TCFA |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TCFA | Input Capture/Compare Match Flag A<br>0: No input capture/compare match of GTCCRA is generated<br>1: An input capture/compare match of GTCCRA is generated | R/W*1 |
| 1 | TCFB | Input Capture/Compare Match Flag B<br>0: No input capture/compare match of GTCCRB is generated<br>1: An input capture/compare match of GTCCRB is generated | R/W*1 |
| 2 | TCFC | Input Compare Match Flag C<br>0: No compare match of GTCCRC is generated<br>1: A compare match of GTCCRC is generated | R/W*1 |
| 3 | TCFD | Input Compare Match Flag D<br>0: No compare match of GTCCRD is generated<br>1: A compare match of GTCCRD is generated | R/W*1 |
| 4 | TCFE | Input Compare Match Flag E<br>0: No compare match of GTCCRE is generated<br>1: A compare match of GTCCRE is generated | R/W*1 |
| 5 | TCFF | Input Compare Match Flag F<br>0: No compare match of GTCCRF is generated<br>1: A compare match of GTCCRF is generated | R/W*1 |
| 6 | TCFPO | Overflow Flag<br>0: No overflow (crest) occurred<br>1: An overflow (crest) occurred | R/W*1 |
| 7 | TCFPU | Underflow Flag<br>0: No underflow (trough) occurred<br>1: An underflow (trough) occurred | R/W*1 |
| 14:8 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15 | TUCF | Count Direction Flag<br>0: GTCNT counter counts downward<br>1: GTCNT counter counts upward | R |
| 23:16 | — | These bits are read as 0. The write value should be 0. | R/W |
| 24 | ODF | Output Disable Flag<br>0: No output disable request is generated<br>1: An output disable request is generated | R |
| 28:25 | — | These bits are read as 0. The write value should be 0. | R/W |
| 29 | OABHF | Same Time Output Level High Flag<br>0: No simultaneous generation of 1 both for the GTIOCA and GTIOCB pins has occurred.<br>1: A simultaneous generation of 1 both for the GTIOCA and GTIOCB pins has occurred. | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 30 | OABLF | Same Time Output Level Low Flag<br>0: No simultaneous generation of 0 both for the GTIOCA and GTIOCB pins has occurred.<br>1: A simultaneous generation of 0 both for the GTIOCA and GTIOCB pins has occurred. | R |
| 31 | PCF | Period Count Function Finish Flag<br>0: No period count function finish has occurred<br>1: A period count function finish has occurred | R/W*1 |

Note 1. Only 0 can be written to this bit. Do not write 1.

The GTST indicates the status of the GPT.

### TCFA flag (Input Capture/Compare Match Flag A)

The TCFA flag indicates the status for the input capture or compare match of GTCCRA.

[Setting conditions]

- GTCNT = GTCCRA, when the GTCCRA register functions as a compare match register
- GTCNT counter value is transferred to GTCCRA by the input capture signal when the GTCCRA register functions as an input capture register.

[Clearing condition]

- 0 is written to this flag.

### TCFB flag (Input Capture/Compare Match Flag B)

The TCFB flag indicates the status for the input capture or compare match of GTCCRB.

[Setting conditions]

- GTCNT = GTCCRB, when the GTCCRB register functions as a compare match register
- GTCNT counter value is transferred to GTCCRB by the input capture signal when the GTCCRB register functions as an input capture register.

[Clearing condition]

- 0 is written to this flag.

### TCFC flag (Input Compare Match Flag C)

The TCFC flag indicates the status for the compare match of GTCCRC.

When GTCCRC performs buffer operation, GTCCRC does not perform compare match.

[Setting condition]

- GTCNT = GTCCRC.

[Clearing condition]

- 0 is written to this flag.

[Not comparing condition]

- GTCR.MD[2:0] = 001b (saw-wave one-shot pulse mode)
- GTCR.MD[2:0] = 110b (triangle-wave PWM mode 3)
- GTBER.CCRA[1:0] = 01b, 10b, 11b (GTCCRC performs buffer operation).

### TCFD flag (Input Compare Match Flag D)

The TCFD flag indicates the status for the compare match of GTCCRD.

When GTCCRD performs buffer operation, GTCCRD does not perform compare match.

[Setting condition]

- GTCNT = GTCCRD.

[Clearing condition]

- 0 is written to this flag.

[Not comparing condition]

- GTCR.MD[2:0] = 001b (saw-wave one-shot pulse mode)
- GTCR.MD[2:0] = 110b (Triangle-wave PWM mode 3)
- GTBER.CCRA[1:0] = 10b, 11b (GTCCRD performs buffer operation).

#### TCFE flag (Input Compare Match Flag E)

The TCFE flag indicates the status for the compare match of GTCCRE.

When GTCCRE performs buffer operation, GTCCRE does not perform compare match.

[Setting condition]

- GTCNT = GTCCRE.

[Clearing condition]

- 0 is written to this flag.

[Not comparing condition]

- GTCR.MD[2:0] = 001b (saw-wave one-shot pulse mode)
- GTCR.MD[2:0] = 110b (Triangle-wave PWM mode 3)
- GTBER.CCRB[1:0] = 01b, 10b, 11b (GTCCRE performs buffer operation).

#### TCFF flag (Input Compare Match Flag F)

The TCFF flag indicates the status for the compare match of GTCCRF.

When GTCCRF performs buffer operation, GTCCRF does not perform compare match.

[Setting condition]

- GTCNT = GTCCRF.

[Clearing condition]

- 0 is written to this flag.

[Not comparing condition]

- GTCR.MD[2:0] = 001b (saw-wave one-shot pulse mode)
- GTCR.MD[2:0] = 110b (Triangle-wave PWM mode 3)
- GTBER.CCRB[1:0] = 10b, 11b (GTCCRF performs buffer operation).

#### TCFPO flag (Overflow Flag)

The TCFPO flag indicates when an overflow or crest has occurred.

[Setting conditions]

- In saw-wave mode, an overflow (GTCNT changes from GTPR to 0 in up-counting) has occurred
- In triangle-wave mode, a crest (GTCNT changes from GTPR to GTPR - 1) has occurred
- In counting by hardware sources, an overflow (GTCNT changes from GTPR to 0 in up-counting) has occurred.

[Clearing condition]

- 0 is written to this flag.

### TCFPU flag (Underflow Flag)

The TCFPU flag indicates when an underflow or trough has occurred.

[Setting conditions]

- In saw-wave mode, an underflow (GTCNT changes from 0 to GTPR in down-counting) has occurred
- In triangle-wave mode, a trough (GTCNT changes from 0 to 1) has occurred
- In counting by hardware sources, an underflow (GTCNT changes from 0 to GTPR in down-counting) has occurred.

[Clearing condition]

- 0 is written to this bit.

### TUCF flag (Count Direction Flag)

The TUCF flag indicates the count direction of GTCNT. In event count operation, this flag is set to 1 in up-counting and to 0 in down-counting.

### ODF flag (Output Disable Flag)

The ODF flag shows the request of the output disable source group that is selected in the GRP[1:0] bits.

When output is disabled, an output disable control is not released within the same cycle in which an output disable request is negated. It is released in the next cycle.

### OABHF flag (Same Time Output Level High Flag)

The OABHF flag indicates that the GTIOCnA pin and GTIOCnB pin output 1 at the same time.

When the GTIOCnA or GTIOCnB pin outputs 0, this flag returns to 0. This flag is read only. Writing 0 to clear the flag is prohibited.

When the output disable request by the OABHF flag is enabled (GTINTAD.GRPABH = 1), the OABHF flag is output to POEG as an output disable request. The GPT does not have an interrupt to indicate that outputs have been simultaneous driven to the high level. Use the interrupt function in the POEG if this is necessary.

[Setting condition]

- The GTIOCnA and GTIOCnB pins output 1 at the same time when both OAE and OBE bits are set to 1.

[Clearing conditions]

- The GTIOCnA pin output value is different from the GTIOCnB pin output value when both OAE and OBE bits are set to 1
- The GTIOCnA and GTIOCnB pins output 0 at the same time when both OAE and OBE bits are set to 1
- Either the OAE bit or OBE bit is set to 0.

### OABLF flag (Same Time Output Level Low Flag)

The OABLF flag indicates that the GTIOCnA and GTIOCnB pins output 0 at the same time.

When the GTIOCnA pin or GTIOCnB pin outputs 1, this flag returns to 0. This flag is read only. Writing 0 to clear the flag is prohibited.

When the output disable request by the OABLF flag is enabled (GTINTAD.GRPABL = 1), the OABLF flag is output to POEG as an output disable request. The GPT does not have an interrupt to indicate that outputs have been simultaneous driven to the low level. Use the interrupt function in the POEG if this is necessary.

[Setting condition]

- The GTIOCnA and GTIOCnB pins output 0 at the same time when both OAE and OBE bits are set to 1.

[Clearing conditions]

- The GTIOCnA pin output value is different from the GTIOCnB pin output value when both OAE and OBE bits are set to 1
- The GTIOCnA and GTIOCnB pins output 1 at the same time when both OAE and OBE bits are set to 1
- Either the OAE bit or the OBE bit is set to 0.

The compare-target signals to generate the OABHF/OABLF flag are the compare match outputs (PWM outputs) signals before they are masked by the output disable function. Even during the output disable condition, compare match operation continues internally, where the OABHF or OABLF flag is updated based on the operation results.

### PCF flag (Period Count Function Finish Flag)

This bit is status flag of period count function finish.

[Setting condition]

- The GTPC.PCEN bit is 1 and the GTPC.PCNT counter is 1 at the end of cycle.
- The GTPC.PCEN bit is 1 and the GTPC.PCNT counter is 0 at the count clock.

[Clearing condition]

- 0 is written to this bit.

## 21.2.17 GTBER : General PWM Timer Buffer Enable Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x40

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | CCRSWT | PR[1:0] | | CCRB[1:0] | | CCRA[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | BD1 | BD0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BD0 | GTCCR Buffer Operation Disable<br>0: Buffer operation is enabled<br>1: Buffer operation is disabled | R/W |
| 1 | BD1 | GTPR Buffer Operation Disable<br>0: Buffer operation is enabled<br>1: Buffer operation is disabled | R/W |
| 15:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 17:16 | CCRA[1:0] | GTCCRA Buffer Operation<br>0 0: No buffer operation<br>0 1: Single buffer operation (GTCCRA ↔GTCCRC)<br>Others: Double buffer operation (GTCCRA ↔ GTCCRC ↔ GTCCRD) | R/W |
| 19:18 | CCRB[1:0] | GTCCRB Buffer Operation<br>0 0: No buffer operation<br>0 1: Single buffer operation (GTCCRB ↔ GTCCRE)<br>Others: Double buffer operation (GTCCRB ↔ GTCCRE ↔ GTCCRF) | R/W |
| 21:20 | PR[1:0] | GTPR Buffer Operation<br>0 0: No buffer operation<br>0 1: Single buffer operation (GTPBR → GTPR)<br>Others: Setting prohibited | R/W |
| 22 | CCRSWT | GTCCRA and GTCCRB Forcible Buffer Operation<br>Writing 1 to this bit forces a buffer transfer of GTCCRA and GTCCRB. This bit automatically returns to 0 after 1 is written. This bit is read as 0. | W |
| 31:23 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTBER register provides settings for the buffer operation. Set the GTBER register while the GTCNT counter is stopped.

#### BD0 bit (GTCCR Buffer Operation Disable)

The BD0 bit disables the buffer operation using GTCCRA, GTCCRB, GTCCRC, GTCCRD, GTCCRE, and GTCCRF combined.

When GTDTCR.TDE is 1 and when BD0 is set to 0, GTCCRB does not perform buffer operation. The GTCCRB register is automatically set to a compare match value for negative-phase waveform with dead time.

A value for the BD0 bit in the channel related to the position of the bit written with 1 by the GTSECSR register can be set when 1 is written to the GTSECR.SBDCE or GTSECR.SBDCD.

#### BD1 bit (GTPR Buffer Operation Disable)

The BD1 bit disables the buffer operation using GTPR and GTPBR combined.

A value for the BD1 bit in the channel related to the position of the bit written with 1 by the GTSECSR register can be set when 1 is written to the GTSECR.SBDPE or GTSECR.SBDPD.

#### CCRA[1:0] bits (GTCCRA Buffer Operation)

The CCRA[1:0] bits set the buffer operation with GTCCRA, GTCCRC, and GTCCRD combined. When the buffer operation is restricted by the operating mode set in GTCR, the GTCR setting is given priority.

The buffer operation mode is fixed in saw-wave one-shot pulse mode or triangle-wave PWM mode 3 (64-bit transfer at trough).

#### CCRB[1:0] bits (GTCCRB Buffer Operation)

The CCRB[1:0] bits set the buffer operation using GTCCRB, GTCCRE, and GTCCRF combined. When the buffer operation is restricted by the operating mode set in GTCR, the GTCR setting is given priority.

The buffer operation mode is fixed in saw-wave one-shot pulse mode or triangle-wave PWM mode 3 (64-bit transfer at trough).

#### PR[1:0] bits (GTPR Buffer Operation)

The PR[1:0] bits set the buffer operation with GTPR and GTPBR combined.

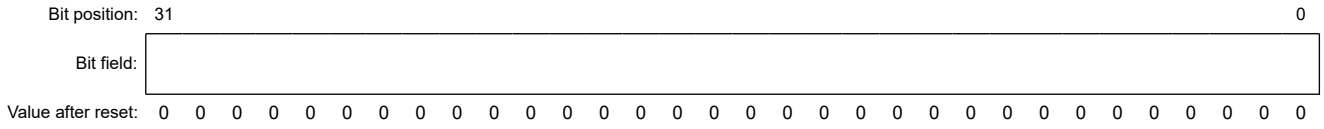
#### CCRSWT bit (GTCCRA and GTCCRB Forcible Buffer Operation)

Writing 1 to the CCRSWT bit forces a buffer transfer of GTCCRA and GTCCRB. This bit automatically returns to 0 after the 1 is written. This bit is read as 0, and is valid only when counting is stopped with a compare match operation specified.

### 21.2.18 GTCNT : General PWM Timer Counter

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x48

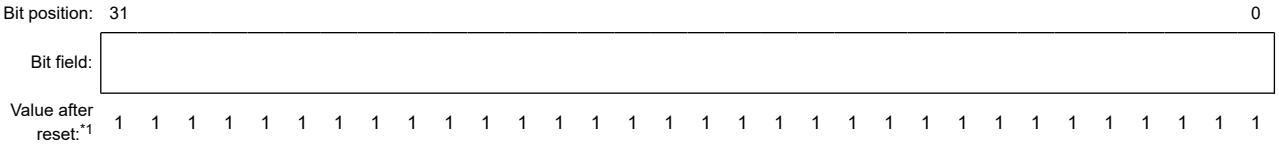



| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | GTCNT is a 32-bit read/write counter for GPT32n (n = 0 to 3). For GPT16m (m = 4 to 7), GTCNT is a 16-bit register. GTCNT can only be written to after counting stops.<br>For GPT16m (m = 4 to 7), the upper 16 bits for access in a 32-bit unit are always read as 0x0000, and writing to these bits is ignored.<br>GTCNT must be set within the range of 0 ≤ GTCNT ≤ GTPR. | R/W |

## 21.2.19 GTCCRk : General PWM Timer Compare Capture Register k (k = A to F)

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x4C (GTCCRA)
0x50 (GTCCRB)
0x54 (GTCCRC)
0x58 (GTCCRE)
0x5C (GTCCRD)
0x60 (GTCCRF)



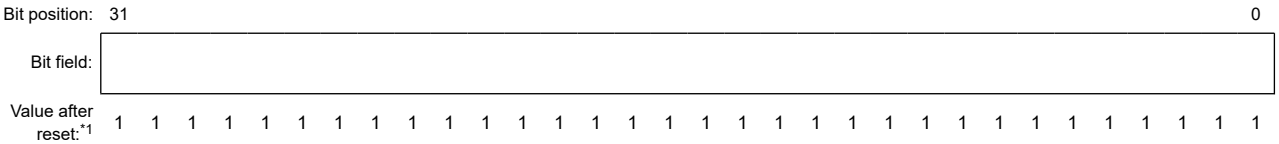

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | GTCCRk registers are read/write registers. The effective size of GTCCRk is the same as GTCNT (16- or 32-bit). If the effective size of GTCCRk is 16 bits, the upper 16 bits for access in a 32-bit unit are always read as 0x0000, and writing to these bits is ignored.<br>GTCCRA and GTCCRB are registers used for both output compare and input capture.<br>GTCCRC and GTCCRE are compare match registers, and can also function as buffer registers for GTCCRA and GTCCRB.<br>GTCCRD and GTCCRF are compare match registers, and can also function as buffer registers for GTCCRC and GTCCRE (double-buffer registers for GTCCRA and GTCCRB). | R/W |

Note 1. For GPT16m (m = 4 to 7), the value of the upper 16 bits after reset is 0x0000.

## 21.2.20 GTPR : General PWM Timer Cycle Setting Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x64



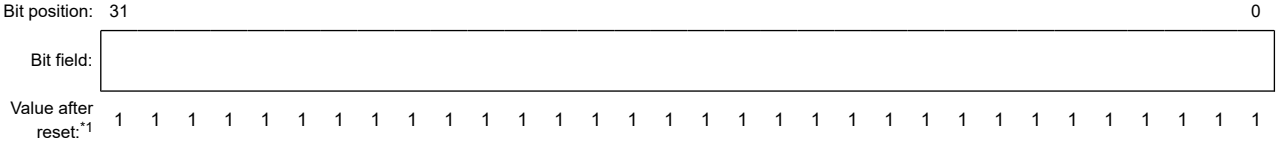

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | GTPR is a read/write register that sets the maximum count value of GTCNT. The effective size of GTPR is the same as GTCNT (16- or 32-bit). If the effective size of GTPR is 16 bits, the upper 16 bits for access in a 32-bit unit are always read as 0x0000, and writing to these bits is ignored.<br>For saw waves, the value of (GTPR + 1) is the cycle. For triangle waves, the value of (GTPR value × 2) is the cycle. | R/W |

Note 1. For GPT16m (m = 4 to 7), the value of the upper 16 bits after reset is 0x0000.

## 21.2.21 GTPBR : General PWM Timer Cycle Setting Buffer Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x68



| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | GTPBR is a read/write register that functions as a buffer register for GTPR. The effective size of GTPBR is the same as GTCNT (16- or 32-bit). If the effective size of GTPBR is 16 bits, the upper 16 bits for access in a 32-bit unit are always read as 0x0000, and writing to these bits is ignored. | R/W |

Note 1. For GPT16m (m = 4 to 7), the value of the upper 16 bits after reset is 0x0000.

## 21.2.22 GTDTCR : General PWM Timer Dead Time Control Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x88

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | TDE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TDE | Negative-Phase Waveform Setting<br>0: GTCCRB is set without using GTDVU<br>1: GTDVU is used to set the compare match value for negative-phase waveform with dead time automatically in GTCCRB | R/W |
| 31:1 | — | These bits are read as 0. The write value should be 0. | R/W |

GTDTCR enables automatic setting of a compare match value for negative-phase waveform with dead time. GPT has a dead time control function and the GTDVU register is used for setting dead time value.

### TDE bit (Negative-Phase Waveform Setting)

The TDE bit specifies whether to use GTDVU. When GTDVU is used, the compare match value for a negative-phase waveform with dead time obtained by the compare match value of a positive-phase waveform (GTCCRA) and the dead time value (GTDVU) is automatically set in GTCCRB.

The TDE bit setting is ignored in saw-wave PWM mode, and the GTCCRB is not automatic setting.

The GTCCRB value is automatically set and has the following upper and lower limit values. If the obtained GTCCRB value is not within the upper or lower limit, the following limit value is set in GTCCRB.

- Triangle waves:
  Upper limit value: GTPR − 1
  Lower limit value: 1 in up-counting, 0 in down-counting
- Saw-wave one-shot pulse mode:
  Upper limit value: GTPR
  Lower limit value: 0.

## 21.2.23 GTDVU : General PWM Timer Dead Time Value Register U

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0x8C

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset:*1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | GTDVU is a read/write register that sets the dead time for generating PWM waveforms with dead time. The effective size of GTDVU is the same as GTCNT (16 or 32 bits). If the effective size of GTDVU is 16 bits, the upper 16 bits for access in a 32-bit unit are always read as 0x0000, and writing to these bits is ignored.<br>Setting a GTDVU value greater than or equal to GTPR is prohibited. When using the automatic dead time setting function, do not set a value that makes a change point of the waveform exceeding the count period. The set value can be confirmed by reading from GTCCRB. When GTDVU is used, writing to GTCCRB is prohibited. When this register is set to 0, waveforms without dead time are output.<br>While GPT is running, changing the GTDVU values is prohibited. To change GTDVU to a new value, stop the GPT with the CST bit in the GTCR register. | R/W |

Note 1. For GPT16m (m = 4 to 7), the value of the upper 16 bits after reset is 0x0000.

## 21.2.24 GTICLF : General PWM Timer Inter Channel Logical Operation Function Setting Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0xB8

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | ICLFSELD[5:0] | | | | | | — | ICLFB[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | ICLFSELC[5:0] | | | | | | — | ICLFA[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | ICLFA[2:0] | GTIOCnA Output Logical Operation Function Select<br>0 0 0: A (no delay)<br>0 0 1: NOT A (no delay)<br>0 1 0: C (1PCLKD delay)<br>0 1 1: NOT C (1PCLKD delay)<br>1 0 0: A AND C (1PCLKD delay)*2<br>1 0 1: A OR C (1PCLKD delay)*2<br>1 1 0: A EXOR C (1PCLKD delay)*2<br>1 1 1: A NOR C (1PCLKD delay)*2 | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9:4 | ICLFSELC[5:0] | Inter Channel Signal C Select*1*2<br>0x00: GTIOC0A<br>0x01: GTIOC0B<br>0x02: GTIOC1A<br>0x03: GTIOC1B<br>0x04: GTIOC2A<br>0x05: GTIOC2B<br>0x06: GTIOC3A<br>0x07: GTIOC3B<br>⋮ ⋮<br>0x3E: GTIOC31A<br>0x3F: GTIOC31B | R/W |
| 15:10 | — | These bits are read as 0. The write value should be 0. | R/W |
| 18:16 | ICLFB[2:0] | GTIOCnB Output Logical Operation Function Select<br>0 0 0: B (no delay)<br>0 0 1: NOT B (no delay)<br>0 1 0: D (1PCLKD delay)<br>0 1 1: NOT D (1PCLKD delay)<br>1 0 0: B AND D (1PCLKD delay)*3<br>1 0 1: B OR D (1PCLKD delay)*3<br>1 1 0: B EXOR D (1PCLKD delay)*3<br>1 1 1: B NOR D (1PCLKD delay)*3 | R/W |
| 19 | — | This bit is read as 0. The write value should be 0. | R/W |
| 25:20 | ICLFSELD[5:0] | Inter Channel Signal D Select*1*3<br>0x00: GTIOC0A<br>0x01: GTIOC0B<br>0x02: GTIOC1A<br>0x03: GTIOC1B<br>0x04: GTIOC2A<br>0x05: GTIOC2B<br>0x06: GTIOC3A<br>0x07: GTIOC3B<br>⋮ ⋮<br>0x3E: GTIOC31A<br>0x3F: GTIOC31B | R/W |
| 31:26 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The signal before performing output disable control is selected.
Note 2. When channel's own GTIOCnA is selected, C is treated as "1".
Note 3. When channel's own GTIOCnB is selected, D is treated as "1".

The GTICLF register sets the logical operation function between compare match outputs. The logical operation is performed with the signals that the duty 0%/100% control is performed after compare match control. (The output disable control is performed with the signal after logical operation.)

Access in 8-bit units to GTICLF is prohibited.

### ICLFm[2:0] bit (GTIOCnm Output Logical Operation Function Select) (m = A, B)

These bits select the logical operation function between signals before performing output disable control for GTIOCnm. To prevent hazard to the GPT output, the signal after logical operation is latched with PCLKD. After latching, the output disable control is performed. When the logical operation function which causes the delay of 1 PCLKD is selected, the output enable signal is also delayed with 1 PCLKD and input to the output disable control.

When the same signal to operate logical function AND, OR, EXOR and NOR is selected, one signal is treated as "1".

### ICLFSELk[5:0] bit (Inter Channel Signal k Select) (k = C, D)

These bits select the signal k that the logical operation is performed with the signal before performing output disable control for GTIOCnm.

## 21.2.25 GTPC : General PWM Timer Period Count Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0, 1)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 6)

Offset address: 0xBC

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | PCNT[11:0] | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | ASTP | — | — | — | — | — | — | — | PCEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PCEN | Period Count Function Enable<br>0: Period count function is disabled<br>1: Period count function is enabled | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | ASTP | Automatic Stop Function Enable<br>0: Automatic stop function is disabled<br>1: Automatic stop function is enabled | R/W |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 27:16 | PCNT[11:0] | Period Counter<br>Counter for the number of period | R/W |
| 31:28 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTPC register counts the number of period.

### PCEN bit (Period Count Function Enable)

This bit enables or disables period count function.

Writing is available when counting is both in progress and stopped.

When 1 is written to either the GTSECR.SPCE bit or the GTSECR.SPCD bit, the value is simultaneously set to the PCEN bit in the channels set to 1 by the GTSECSR register.

### ASTP bit (Automatic Stop Function Enable)

This bit enables or disables the GTCNT counter automatic stopping after finishing counting the number of period.

When the PCEN bis is 0, writing is available.

When the PCEN bit is 1, writing is disabled.

When the PCEN bit is 1, the ASTP bit is 1, and the PCNT counter is stopped at PCNT = 0, the GTCNT counter is also stopped. When the ASTP bit is 0, the GTCNT counter continues to count.

### PCNT[11:0] bit (Period Counter)

This counter counts the number of period.

When the PCEN bis is 0, writing the number of period is available.

When the PCEN bit is 1, writing is disabled, and down-counting is performed at the end of period. In saw-wave mode, the end of period refers to overflow, underflow, or counter clearing. In triangle-wave mode, it refers to trough.

When the PCNT counter is 1 at the end of period, it becomes 0 and counting is stopped.

When the GTCNT counter is stopped while period count function is enabled, the PCNT counter keeps its value. When the GTCNT counter restarts counting and the PCEN bit is 1, the PCNT counter restarts down-counting from the hold value.

When the PCEN bit is changed from 0 to 1 while the PCNT counter is 0 and the ASTP bit is 1, the GTCNT counter is stopped at the count clock immediately after that.

## 21.2.26 GTSECSR : General PWM Timer Operation Enable Bit Simultaneous Control Channel Select Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0xD0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | SECSEL7 | SECSEL6 | SECSEL5 | SECSEL4 | SECSEL3 | SECSEL2 | SECSEL1 | SECSEL0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SECSEL0 | Channel 0 Operation Enable Bit Simultaneous Control Channel Select<br>0: Disable simultaneous control<br>1: Enable simultaneous control | R/W |
| 1 | SECSEL1 | Channel 1 Operation Enable Bit Simultaneous Control Channel Select<br>0: Disable simultaneous control<br>1: Enable simultaneous control | R/W |
| 2 | SECSEL2 | Channel 2 Operation Enable Bit Simultaneous Control Channel Select<br>0: Disable simultaneous control<br>1: Enable simultaneous control | R/W |
| 3 | SECSEL3 | Channel 3 Operation Enable Bit Simultaneous Control Channel Select<br>0: Disable simultaneous control<br>1: Enable simultaneous control | R/W |
| 4 | SECSEL4 | Channel 4 Operation Enable Bit Simultaneous Control Channel Select<br>0: Disable simultaneous control<br>1: Enable simultaneous control | R/W |
| 5 | SECSEL5 | Channel 5 Operation Enable Bit Simultaneous Control Channel Select<br>0: Disable simultaneous control<br>1: Enable simultaneous control | R/W |
| 6 | SECSEL6 | Channel 6 Operation Enable Bit Simultaneous Control Channel Select<br>0: Disable simultaneous control<br>1: Enable simultaneous control | R/W |
| 7 | SECSEL7 | Channel 7 Operation Enable Bit Simultaneous Control Channel Select<br>0: Disable simultaneous control<br>1: Enable simultaneous control | R/W |
| 9:8 | — | These bits are read as 0. The write value should be 0. | R/W |
| 31:10 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTSECSR register selects an intended channel n (n = 0 to 7) for updating an operation enable bit by the GTSECR register. A bit position for the GTSECSR register indicates a channel number. The GTSECSR register of each channel is a common register, and writing 1 to a bit in the GTSECSR register in any channel and updating it changes a channel, related to the position of the bit written with 1 by the GTSECSR register, to be simultaneously controlled of the operation enable bit by the GTSECR register.

The bit corresponding to channel which security attribution is configured as secure can be read by non-secure access but cannot be written by non-secure access. For example, if GPT channel 0 is configured as secure and other GPTs are configured as non-secure, the SECSEL0 bit cannot be written by non-secure access to GPT321.GTSECSR register, and the simultaneous control status of GPT channel 0 is not changed. When the GPT321.GTSECSR register is read by non-secure

access in the same security configuratin as the previous example, the simultaneous control status of GPT channel 0 (SECSEL0 bit) can be read.

Access in 8-bit or 16-bit units to GTSECSR is prohibited, and it should be accessed in 32-bit units.

**SECSELn bit (Operation Enable Bit Simultaneous Control Channel Select) (n = 0 to 7)**

This bit enables or disables the simultaneous control of operation enable in channel n.

When the bit is set to 1, the simultaneous control is enabled, and disabled when the bit is 0.

## 21.2.27 GTSECR : General PWM Timer Operation Enable Bit Simultaneous Control Register

Base address: GPT32n = 0x4016_9000 + 0x0100 × n (n = 0 to 3)
GPT16m = 0x4016_9000 + 0x0100 × m (m = 4 to 7)

Offset address: 0xD4

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | SPCD | — | — | — | — | — | — | — | SPCE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | SBDPD | SBDCD | — | — | — | — | — | — | SBDPE | SBDCE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SBDCE | GTCCR Register Buffer Operation Simultaneous Enable<br>0: Disable simultaneous enabling GTCCR buffer operations<br>1: Enable GTCCR register buffer operations simultaneously | R/W |
| 1 | SBDPE | GTPR Register Buffer Operation Simultaneous Enable<br>0: Disable simultaneous enabling GTPR buffer operations<br>1: Enable GTPR register buffer operations simultaneously | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | SBDCD | GTCCR Register Buffer Operation Simultaneous Disable<br>0: Disable simultaneous disabling GTCCR buffer operations<br>1: Disable GTCCR register buffer operations simultaneously | R/W |
| 9 | SBDPD | GTPR Register Buffer Operation Simultaneous Disable<br>0: Disable simultaneous disabling GTPR buffer operations<br>1: Disable GTPR register buffer operations simultaneously | R/W |
| 15:10 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | SPCE | Period Count Function Simultaneous Enable<br>0: Disable simultaneous enabling period count function<br>1: Enable period count function simultaneously | R/W |
| 23:17 | — | These bits are read as 0. The write value should be 0. | R/W |
| 24 | SPCD | Period Count Function Simultaneous Disable<br>0: Disable simultaneous disabling period count function<br>1: Disable period count function simultaneously | R/W |
| 31:25 | — | These bits are read as 0. The write value should be 0. | R/W |

The GTSECR register simultaneously updates the value for operation enable bits of a channel set by the GTSECSR register.

Writing 1 to a bit in the GTSECR register in any channel and updating it updates an operation enable bit for all channels, related to the position of the bit written with 1 by the all GTSECSR registers.

The GTSECR register of channel which security attribution is configured as secure can not be written by non-secure access. For example, if GPTn is configured as secure and other GPTs are configured as non-secure, the GPTn.GTSECR register can

not be written by non-secure access to GPTn+1.GTSECR register even if the simultaneous control of GPTn is enabled, and the simultaneous control status of GPTn is not changed.

Setting enable and disable bits for the same operation enable bit to 1 in the GTSECR is prohibited.

A bit written to 1 is automatically cleared. When the GTSECR is read, 0 is read.

Access in 8-bit or 16-bit units to the GTSECR register is prohibited, and it should be accessed in 32-bit units.

#### SBDCE bit (GTCCR Register Buffer Operation Simultaneous Enable)

When 1 is written to this bit, 0 is simultaneously set to a GTBER.BD[0] bit in the channels set to 1 by the GTSECSR register, and buffer operations using the GTCCRA, GTCCRC, and GTCCRD registers and using the GTCCRB, GTCCRE, and GTCCRF registers are enabled.

Simultaneous setting of SBDCE and SBDCD bits to 1 is prohibited.

#### SBDPE bit (GTPR Register Buffer Operation Simultaneous Enable)

When 1 is written to this bit, 0 is simultaneously set to a GTBER.BD[1] bit in the channels set to 1 by the GTSECSR register, and buffer operations using the GTPR and GTPBR registers are enabled.

Simultaneous setting of SBDPE and SBDPD bits to 1 is prohibited.

#### SBDCD bit (GTCCR Register Buffer Operation Simultaneous Disable)

When 1 is written to this bit, 1 is simultaneously set to a GTBER.BD[0] bit in the channels set to 1 by the GTSECSR register, and buffer operations using the GTCCRA, GTCCRC, and GTCCRD registers and using the GTCCRB, GTCCRE, and GTCCRF registers are disabled.

Simultaneous setting of SBDCE and SBDCD bits to 1 is prohibited.

#### SBDPD bit (GTPR Register Buffer Operation Simultaneous Disable)

When 1 is written to this bit, 1 is simultaneously set to a GTBER.BD[1] bit in the channels set to 1 by the GTSECSR register, and buffer operations using the GTPR and GTPBR registers are disabled.

Simultaneous setting of SBDPE and SBDPD bits to 1 is prohibited.

#### SPCE bit (Period Count Function Simultaneous Enable)

When 1 is written to this bit, 1 is simultaneously set to GTPC.PCEN bit in the channels set to 1 by the GTSECSR register, and period count function is enabled.

Simultaneous setting of SPCE and SPCD bits to 1 is prohibited.

#### SPCD bit (Period Count Function Simultaneous Disable)

When 1 is written to this bit, 0 is simultaneously set to GTPC.PCEN bit in the channels set to 1 by the GTSECSR register, and period count function is disabled.

Simultaneous setting of SPCE and SPCD bits to 1 is prohibited.

### 21.2.28 OPSCR : Output Phase Switching Control Register

Base address: GPT_OPS = 0x4016_9A00

Offset address: 0x00

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | NFCS[1:0] | | NFEN | — | — | GODF | GRP[1:0] | | — | — | ALIGN | RV | INV | N | P | FB |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | EN | — | W | V | U | — | WF | VF | UF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | UF | Input Phase Soft Setting<br>These bits set the input phase from software settings.<br>Setting these bits is valid when OPSCR.FB = 1. | R/W |
| 1 | VF | | R/W |
| 2 | WF | | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | U | Input U-Phase Monitor<br>This bit monitors the state of the input phase.<br>OPSCR.FB = 0 : External input that are synchronized by PCLKD<br>OPSCR.FB = 1 : Software settings (UF) | R |
| 5 | V | Input V-Phase Monitor<br>This bit monitors the state of the input phase.<br>OPSCR.FB = 0 : External input that are synchronized by PCLKD<br>OPSCR.FB = 1 : Software settings (VF) | R |
| 6 | W | Input W-Phase Monitor<br>This bit monitors the state of the input phase.<br>OPSCR.FB = 0 : External input that are synchronized by PCLKD<br>OPSCR.FB = 1 : Software settings (WF) | R |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |
| 8 | EN | Output Phase Enable<br>0: Do not output (Hi-Z external pin)<br>1: Output*1 | R/W |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | FB | External Feedback Signal Enable<br>This bit selects the input phase from software settings and external input.<br>0: Select the external input<br>1: Select the soft setting (OPSCR.UF, VF, WF) | R/W |
| 17 | P | Positive-Phase Output (P) Control<br>0: Level signal output<br>1: PWM signal output | R/W |
| 18 | N | Negative-Phase Output (N) Control<br>0: Level signal output<br>1: PWM signal output | R/W |
| 19 | INV | Output Phase Invert Control<br>0: Positive logic (active-high) output<br>1: Negative logic (active-low) output | R/W |
| 20 | RV | Output Phase Rotation Direction Reversal Control<br>0: Positive rotation<br>1: Reverse rotation | R/W |
| 21 | ALIGN | Input Phase Alignment<br>0: Input phase aligned to PCLKD<br>1: Input phase aligned to the falling edge of PWM | R/W |
| 23:22 | — | These bits are read as 0. The write value should be 0. | R/W |
| 25:24 | GRP[1:0] | Output Disabled Source Selection<br>0 0: Select group A output disable source<br>0 1: Select group B output disable source<br>1 0: Select group C output disable source<br>1 1: Select group D output disable source | R/W |
| 26 | GODF | Group Output Disable Function<br>0: This bit function is ignored<br>1: Group disable clears the OPSCR.EN bit*1 | R/W |
| 28:27 | — | These bits are read as 0. The write value should be 0. | R/W |
| 29 | NFEN | External Input Noise Filter Enable<br>0: Do not use a noise filter on the external input<br>1: Use a noise filter on the external input | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:30 | NFCS[1:0] | External Input Noise Filter Clock Selection<br>Noise filter sampling clock setting of the external input.<br>0 0: PCLKD/1<br>0 1: PCLKD/4<br>1 0: PCLKD/16<br>1 1: PCLKD/64 | R/W |

Note 1. When OPSCR.GODF = 1 and the signal value selected by the OPSCR.GRP[1:0] bit is high, the OPSCR.EN bit is set to 0.

The OPSCR register sets the output of the signal waveform required for brushless DC motor control.

### UF , VF , WF bits (Input Phase Soft Setting)

The UF , VF , WF bits set the input phase from the software settings. When OPSCR.FB bit is 1, these bits are valid. The set value of the UF /VF /WF takes the place of the U/V/W external input.

### U, V, W bits (Input Phase Monitor)

When the OPSCR.FB bit is 0, external inputs that are synchronized by PCLKD are monitored by these bits. When the OPSCR.FB bit is 1, the OPSCR.U, OPSCR.V, and OPSCR.W bits can read the OPSCR.UF , OPSCR.VF , and OPSCR.WF bits.

### EN bit (Output Phase Enable)

The EN bit controls the output enable signal of output phase (positive phase/negative phase).

When the OPSCR.EN bit is 1, the signal waveform is output.

When the OPSCR.EN bit is 0, first set OPSCR.FB, OPSCR.UF /VF /WF (software setting is selected), OPSCR.P/N, OPSCR.INV, OPSCR.RV, OPSCR.ALIGN, OPSCR.GRP[1:0], OPSCR.GODF, OPSCR.NFEN, OPSCR.NFCS. Then, set the EN bit to 1. The EN bit should be set when output disable request does not occur from POEG. Also when OPSCR.GODF is 1 and the signal value selected in the OPSCR.GRP[1:0] bit is high, the OPSCR.EN bit is set to 0. Even if 1 is written by software, the EN bit remains at 0.

For the return, after clearing the Output Disable Request by software, set the EN bit to 1.

Priority order of the EN bit is as follows (when the conflict occurs).

When writing 1 by software and clearing to 0 by the Output Disable Request conflict for the EN bit, clearing to 0 by the Output Disable Request is enabled.

### FB bit (External Feedback Signal Enable)

The FB bit selects the input phase from the software settings (OPSCR.UF, VF, WF) and external input such as a Hall element.

### P bit (Positive-Phase Output (P) Control)

The P bit selects one of the level signal output or PWM signal output for the positive-phase output (GTOUUP, GTOVUP and GTOWUP pins).

### N bit (Negative-Phase Output (N) Control)

The N bit selects one of the level signal output or PWM signal output for the negative-phase output (GTOULO, GTOVLO and GTOWLO pins).

### INV bit (Output Phase Invert Control)

The INV bit selects one of the positive logic (active-high) output or negative logic (active-low) output for the output phase.

### RV bit (Output Phase Rotation Direction Reversal Control)

The RV bit reverses the direction of rotation of the motor by inverting the input phase.

### ALIGN bit (Input Phase Alignment)

The ALIGN bit selects the PCLKD or PWM for the sampling of the input phase (input phase is specified in the OPSCR.FB bit).

When OPSCR.ALIGN bit is 0, input phase is aligned to PCLKD.

Note: When the chopping is performed, there are cases where the PWM width of output is shorter than the width of the PWM used to chop just before or after switching of output phase, depending on the phase difference between the phase output switch timing and the phase of PWM.

When OPSCR.ALIGN bit is 1, input phase is aligned with the falling edge of PWM.

#### GRP[1:0] bit (Output Disabled Source Selection)

The GRP[1:0] bit selects the output disable source.

The GRP bits should be set when GODF bit is 0. If GRP bits select a POEG except for the connected groups, the status of output pin never change to disable.

#### GODF bit (Group Output Disable Function)

When OPSCR.GODF is 1 and the signal value selected by the OPSCR.GRP[1:0] bit is high, the OPSCR.EN bit is set to 0.

When OPSCR.GODF bit is 0, this bit is ignored.

The GODF bit should be set when output disable request does not occur from POEG.

#### NFEN bit (External Input Noise Filter Enable)

The NFEN bit selects the noise filter for external input. When OPSCR.NFEN bit is 0, a noise filter for the external input is not used.

Note: Set this bit during the EN bit is 0 to avoid generation of unintentional internal edge caused by switching this bit.

#### NFCS[1:0] bits (External Input Noise Filter Clock Selection)

The NFCS[1:0] bits select the clock for the external input noise filter. When the OPSCR.NFEN bit is 1, noise filter sampling clock setting of the external input is enabled.

1. Set the NFCS[1:0].
2. Wait for 2 cycles.
3. Set the OPSCR.EN bit to 1.

## 21.3 Operation

### 21.3.1 Basic Operation

Each channel has a 32-bit and 16-bit timer that performs a periodic count operation using the count clock and hardware sources. The count function provides both up-counting and down-counting. The GTPR controls the count cycle.

When the GTCNT counter value matches the value in GTCCRA or GTCCRB, the output from the associated GTIOCnA or GTIOCnB can be changed (n = 0 to 7). GTCCRA or GTCCRB can be used as an input capture register with hardware resources.

GTCCRC and GTCCRD can function as buffer registers for GTCCRA. GTCCRE and GTCCRF can function as buffer registers for GTCCRB.

#### 21.3.1.1 Counter operation

##### (1) Counter start and stop

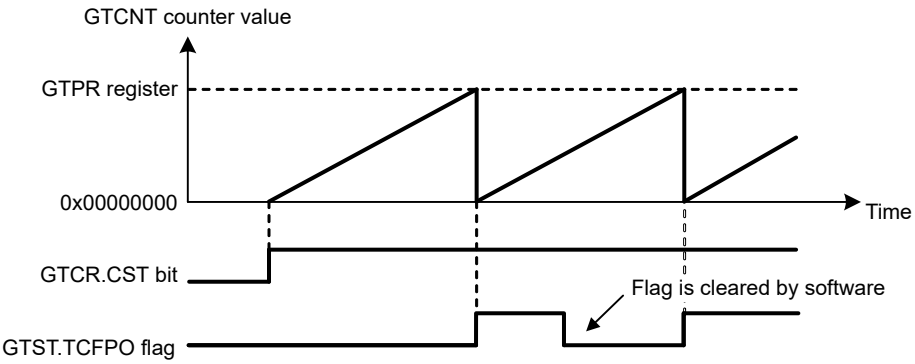
The counter of each channel starts the count operation when GTCR.CST is set to 1, and stops counting when the bit is set to 0. The GTCR.CST bit value is changed by the following sources:

- Writing to GTCR register
- Writing 1 to the bit in GTSTR associated with the GPT channel number when the GTSSR.CSTRT bit set to 1
- Writing 1 to the bit in GTSTP associated with the GPT channel number when the GTPSR.CSTOP bit set to 1
- The hardware source selected in the GTSSR register
- The hardware source selected in the GTPSR register
- Completion of the period count function while the GTPC.ASTP bit is 1

### (2) Periodic count operation in up-counting by count clock

The GTCNT counter in each channel starts up-counting when the associated GTCR.CST bit is set to 1 with GTUPSR and GTDNSR registers set to 0x00000000. When the GTCNT value changes from the GTPR value to 0 (overflow), the GTST.TCFPO flag is set to 1, and the overflow interrupt(GPTn_OVF) is also generated. After GTCNT overflows, up-counting resumes from 0x00000000.

Figure 21.3 shows an example of a periodic count operation in up-counting by the count clock.




Note: The base address of registers are the same.

**Figure 21.3 Example of periodic count operation in up-counting by the count clock**

Table 21.5 shows an example for setting periodic count operation in up-counting by the count clock.

**Table 21.5 Example for setting a periodic count operation in up-counting by the count clock**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.3, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.3, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. In Figure 21.3, 0x00000000 is set. |
| 6 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |

### (3) Periodic count operation in down-counting by count clock

The GTCNT counter in each channel can perform down-counting by setting GTUDDTYC.UD with GTUPSR and GTDNSR registers set to 0x00000000. When GTCNT changes from 0 to the GTPR value (underflow), GTST.TCFPU is set to 1, and the underflow interrupt(GPTn_UDF) is also generated. After the GTCNT counter underflows, down-counting resumes from the GTPR value.

Figure 21.4 shows an example of periodic count operation in down-counting by the count clock.

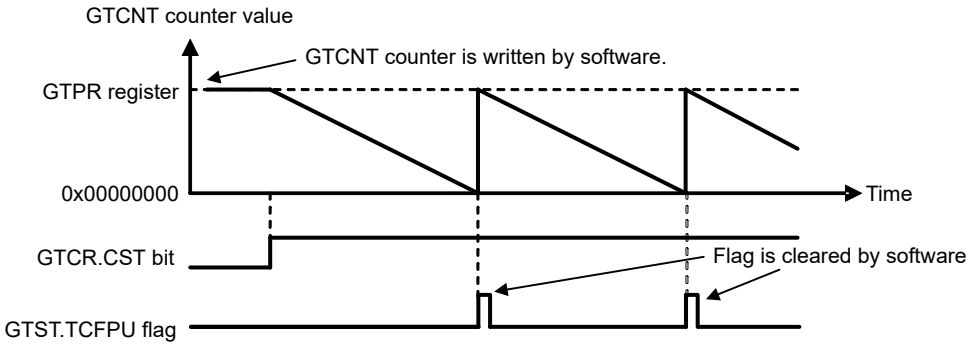



**Figure 21.4 Example of periodic count operation in down-counting by the count clock**

Table 21.6 shows an example for setting periodic count operation in down-counting by the count clock.

**Table 21.6 Example for setting periodic count operation in down-counting by count clock**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.4, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction with the GTUDDTYC register.<br>In Figure 21.4, after 10b is set in the GTUDDTYC[1:0] bits, 00b is set in the GTUDDTYC[1:0] bits (down-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter.<br>In Figure 21.4, the GTPR register value is set. |
| 6 | Start count operation | Set the GTCR.CST bit to 1 to start count operation.<br>In Figure 21.4, 1 is set in the CST bit. |

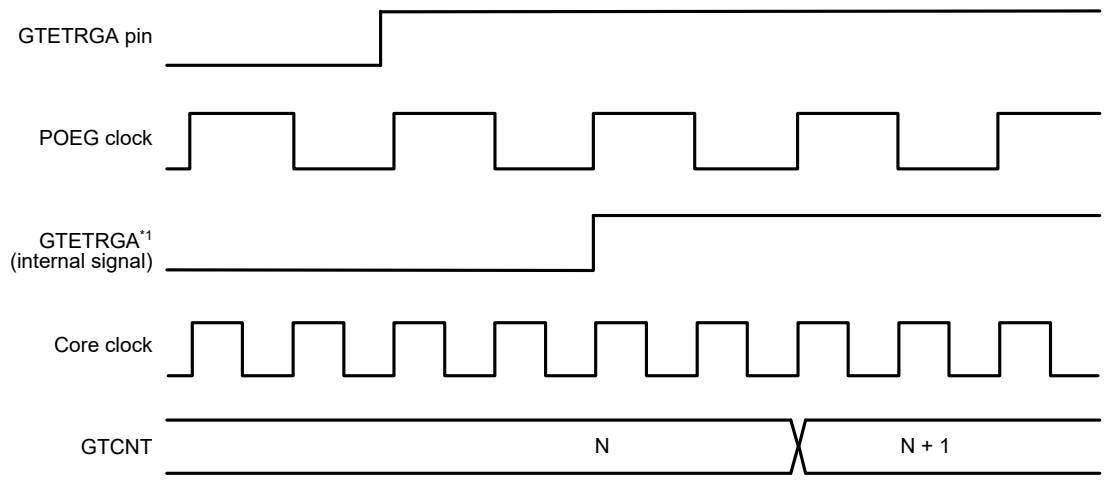
### (4) Event count operation in up-counting using hardware sources

The GTCNT counter in each channel can perform up-counting using hardware sources as set in GTUPSR.

When GTUPSR is set to enable, the count clock selected in GTCR.TPCS[3:0] and the count direction selected in GTUDDTYC.UD are ignored. If up-counting and down-counting using hardware sources occur at the same time, the GTCNT counter value does not change. The overflow behavior when up-counting using hardware sources is the same as when up-counting by the count clock.

If you are using a hardware source to count up, set the GTCR.CST bit to 1 to enable the counting operation. After GTCR.CST is set to 1, the counter cannot count up for 1 clock cycle as specified in GTCR.TPCS[3:0] because the count operation is synchronized by the count clock selected in GTCR.TPCS[3:0]. Set GTCR.TPCS[3:0] to 000b to count up with a 1 PCLKD delay after GTCR.CST is set to 1.

Figure 21.5 shows an example of an event count operation in up-counting by a hardware resource (the rising edge of GTETRGA pin input).




Note: POEG clock :PCLKB, Core clock : PCLKD
Note 1. The GTETRGA internal signal reflects the signal being input to the GPT when the digital noise filter for the GTETRGA pin in the POEG is not in use.

**Figure 21.5 Example of event count operation in up-counting using hardware sources**

Table 21.7 shows an example for setting event count operation in up-counting by a hardware source.

**Table 21.7 Example for setting an event count operation in up-counting using hardware sources**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set count source | Select the counting-up hardware source with the GTUPSR register. |
| 2 | Set cycle | Set the cycle in the GTPR register. |
| 3 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 4 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |

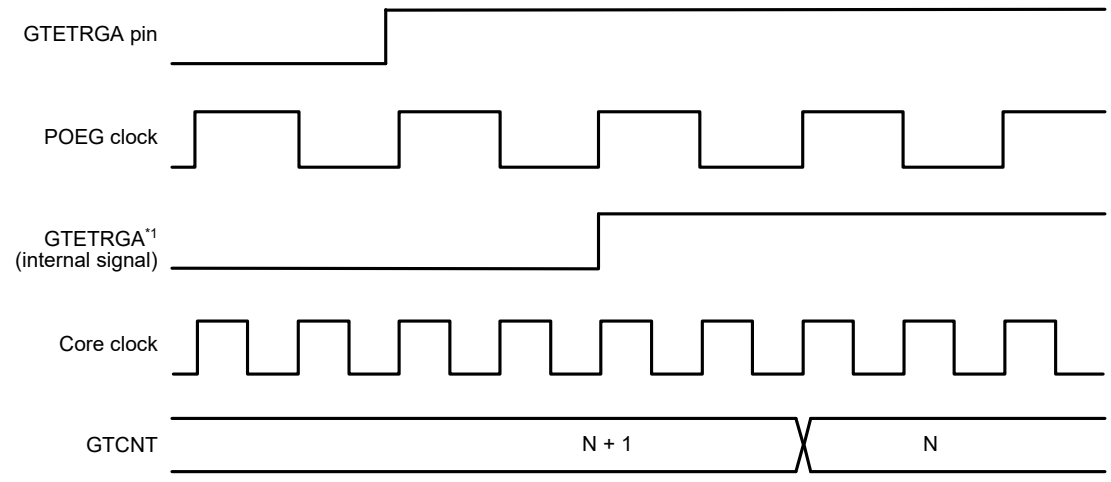
## (5) Event count operation in down-counting using hardware sources

The GTCNT counter in each channel can perform down-counting using hardware sources set in the GTDNSR.

When GTDNSR is set to enable, the count clock selected in GTCR.TPCS[3:0] and the count direction selected in GTUDDTYC.UD are ignored. If up-counting and down-counting using hardware sources occur at the same time, the GTCNT counter value does not change. The underflow behavior when down-counting using hardware sources is the same as when down-counting by the count clock.

When GTCR.CST bit is set to 1 to count down using hardware sources, the count operation is enabled. After GTCR.CST is set to 1, the counter cannot count down for 1 clock cycle as specified in GTCR.TPCS[3:0] because the count operation is synchronized with the count clock selected in GTCR.TPCS[3:0]. Set GTCR.TPCS[3:0] to 000b to count down with a 1 PCLKD delay after GTCR.CST is set to 1.

Figure 21.6 shows an example of a event count operation in down-counting by a hardware resource (rising edge of GTETRGA pin).




Note: POEG clock: PCLKB, Core clock: PCLKD
Note 1. The GTETRGA internal signal reflects the signal being input to the GPT when the digital noise filter for the GTETRGA pin in the POEG is not in use.

**Figure 21.6 Example of event count operation in down-counting using hardware sources**

Table 21.8 shows an example for setting a periodic count operation in down-counting using a hardware resource.

**Table 21.8 Example for setting an event count operation in down-counting using hardware sources**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set count source | Select the counting-down hardware source with the GTDNSR register. |
| 2 | Set cycle | Set the cycle in the GTPR register. |
| 3 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 4 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |

### (6) Counter clear operation

The counter of each channel is cleared by following sources:

- Writing 0 to GTCNT register
- Writing 1 to the bit in GTCLR associated with the GPT channel number when the GTCSR.CCLR bit set to 1
- The hardware source selected in GTCSR register.

Writing to the GTCNT register is prohibited during count operation. The GTCNT counter can be cleared both by writing 1 to the GTCLR and by the clear request of hardware sources, whether GTCNT is counting (GTCR.CST is 1) or not (GTCR.CST is 0).

When the count direction flag is set as decrement (GTST.TCUF flag = 0) in saw-wave mode selected with GTCR.MD[2:0] bits, the GTCNT register is set to the value of the GTPR register when writing 1 to the GTCLR register and when clearing by hardware sources are performed.

When not in saw-waves mode and down-counting, the GTCNT register is set to 0 when writing 1 to the GTCLR register and when clearing by hardware sources are performed.

In event count operation when at least 1 bit in the GTUPSR or GTDNSR is set to 1, after clear sources occur, both writing to GTCLR register and clearing by hardware sources are performed immediately to synchronize with PCLKD. If other settings are used, clear is synchronized with the counter clock selected in GTCR.TPCS[3:0].

### 21.3.1.2 Waveform output by compare match

Compare match means that the GTCNT counter value matches the value of GTCCRA or GTCCRB. When a compare match occurs, the compare match flag is generated synchronously with the count clock, including the event count. At the same time, the GPT can output low, high, or toggled output from the associated GTIOCnA or GTIOCnB output pin (n = 0 to 7). In addition, the GTIOCnA or GTIOCnB pin output can be low, high, or toggled at the cycle end which is determined by GTPR.

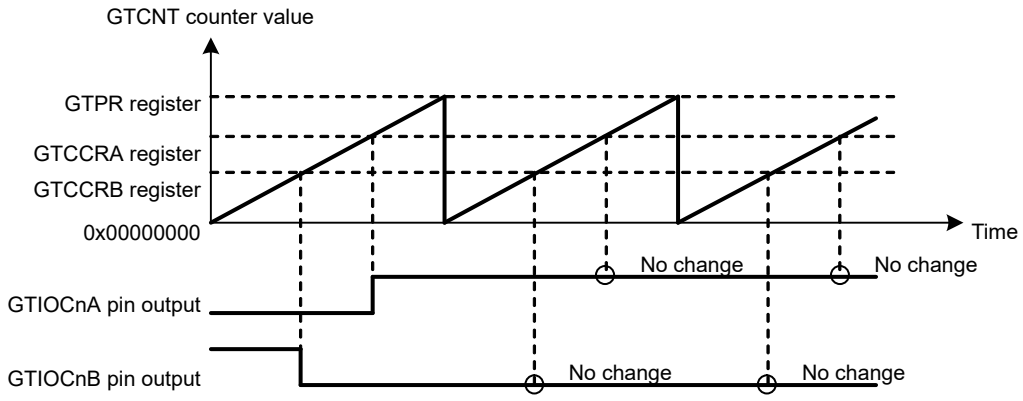
The cycle end is:

- For saw waves in up-counting – when GTCNT changes from the GTPR value to 0 (overflow)
- For saw waves in down-counting – when GTCNT changes from 0 to GTPR value (underflow)
- For saw waves – when the GTCNT counter is cleared
- For triangle waves – when the GTCNT changes from 0 to 1 (trough).

#### (1) Low output and high output

Figure 21.7 shows an example of low output and high output operation by a compare match of GTCCRA and GTCCRB.

In this example, the GTCNT counter performs up-counting, and settings are made so that high is output from the GTIOCnA pin by a GTCCRA compare match, and low is output from the GTIOCnB pin by a GTCCRB compare match. The pin level does not change when the specified level and pin level match.




[Setting examples]
GTIOR.GTIOA[4:0] bits: Initial output is low, high output at compare match, output retained at cycle end
GTIOR.GTIOB[4:0] bits: Initial output is high, low output at compare match, output retained at cycle end

Note: The base address of registers are the same.

**Figure 21.7 Example of low output and high output operation**

Table 21.9 shows an example for setting low output and high output operation.

**Table 21.9 Example for setting low output and high output operation (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.7, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.7, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |

**Table 21.9 Example for setting low output and high output operation (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 6 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.7, GTIOA[4:0] = 00010b, GTIOB[4:0] = 10001b. |
| 7 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 8 | Set compare match value | Set compare match values in the GTCCRA and GTCCRB registers. |
| 9 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |

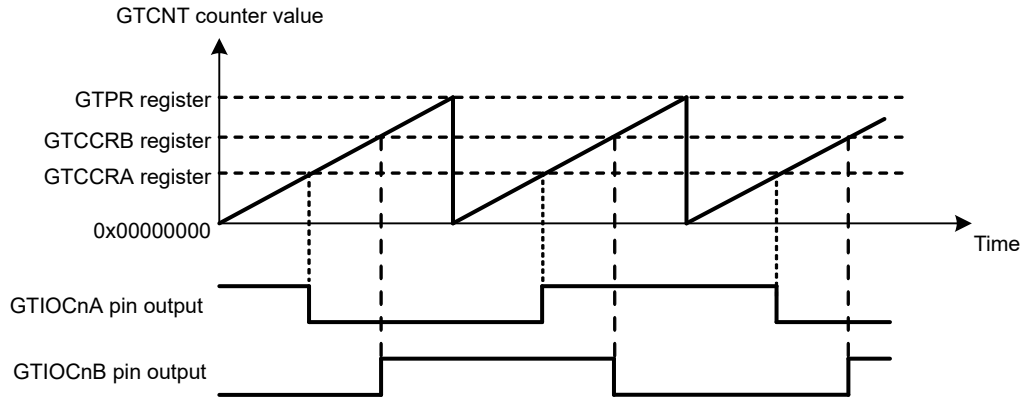
Note: n: 0 to 7
m: A, B

## (2) Toggled output

Figure 21.8 and Figure 21.9 show examples of toggled output operation by compare matches of GTCCRA and GTCCRB.

In Figure 21.8, the GTCNT counter performs up-counting, and settings are made so that the GTIOCnA pin output by a GTCCRA compare match and GTIOCnB pin output by a GTCCRB compare match are toggled.

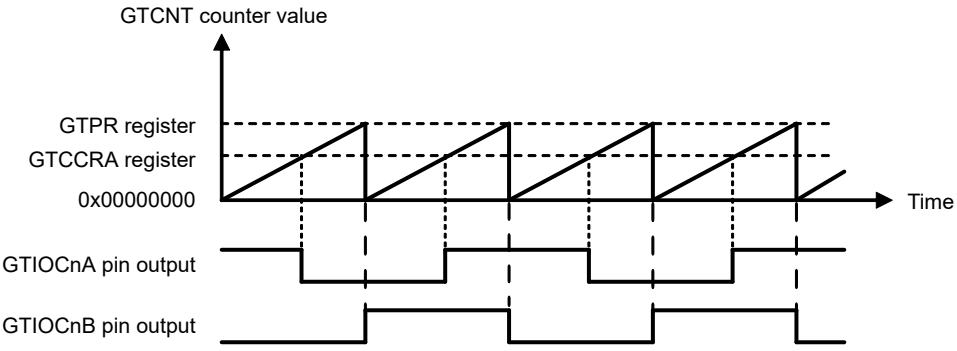
In Figure 21.9, the GTCNT counter performs up-counting, and settings are made so that a GTCCRA compare match toggles the GTIOCnA pin output level and a cycle end toggles the GTIOCnB pin output level.




[Setting examples]
GTIOR.GTIOA[4:0] bits: Initial output is high, output toggled at compare match, output retained at cycle end
GTIOR.GTIOB[4:0] bits: Initial output is low, output toggled at compare match, output retained at cycle end

Note: The base address of registers are the same.

**Figure 21.8 Example of toggled output operation (1)**




[Setting examples]
GTIOR.GTIOA[4:0] bits: Initial output is high, output toggled at compare match, output retained at cycle end
GTIOR.GTIOB[4:0] bits: Initial output is low, output retained at compare match, output toggled at cycle end

Note: The base address of registers are the same.

**Figure 21.9 Example of toggled output operation (2)**

Table 21.10 shows an example for setting toggled output operation.

**Table 21.10 Example for setting toggled output operation**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.8 and Figure 21.9, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register. In Figure 21.8 and Figure 21.9, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 6 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.8, GTIOA[4:0] = 10011b, GTIOB[4:0] = 00011b, and in Figure 21.9, GTIOA[4:0] = 10011b, GTIOB[4:0] = 01100b. |
| 7 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 8 | Set compare match value | Set compare match values in the GTCCRA and GTCCRB registers. |
| 9 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |

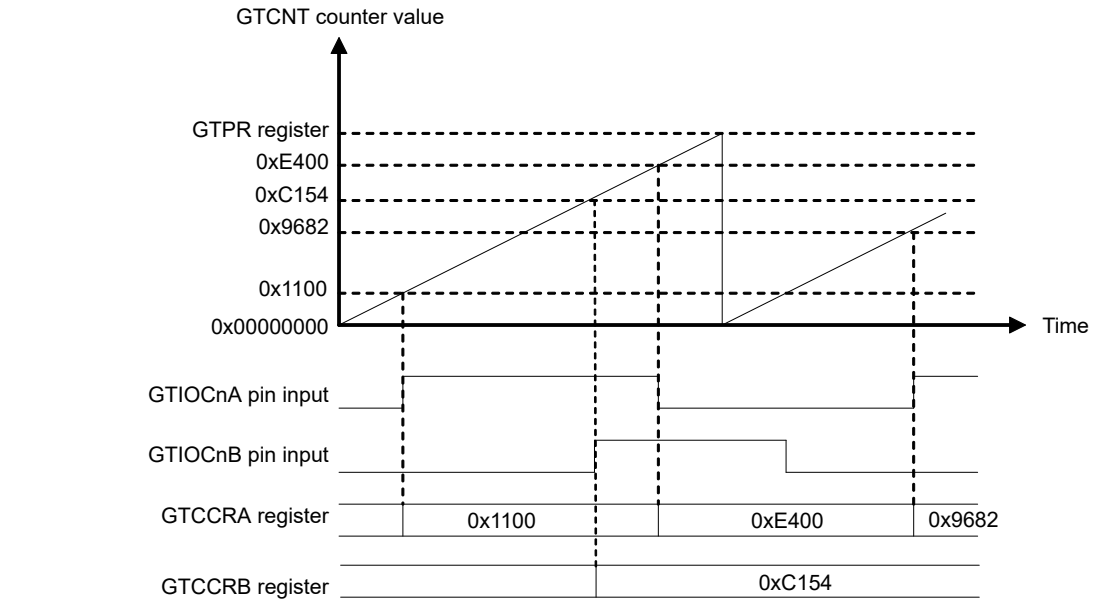
Note: n: 0 to 7
m: A, B

### 21.3.1.3 Input Capture Function

The GTCNT counter value can be transferred to either GTCCRA or GTCCRB on detection of the hardware source that is set in GTICASR and GTICBSR.

Figure 21.10 shows an example of the input capture function.

In this example, the GTCNT counter performs up-counting by the count clock, and settings are made so that an input capture is performed to GTCCRA at both edges of the GTIOCnA input pin and to GTCCRB on the rising edge of the GTIOCnB input pin.




Note: The base address of registers are the same.
Note: [Setting examples]
GTICASR setting: input capture at both edges
GTICBSR setting: input capture at the rising edge

**Figure 21.10 Example of input capture operation**

Table 21.11 and Table 21.14 show the example for setting an input capture operation with count operation by the count clock.

**Table 21.11 Example for setting input capture operation**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.10, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.10, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 6 | Select input capture source | Select the input capture source in the GTICASR and GTICBSR registers.<br>In Figure 21.10, GTICASR = 0x00000F00, GTICBSR = 0x00003000. |
| 7 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |

## 21.3.2 Buffer Operation

The following buffer operations can be set with GTBER:

- GTPR and GTPBR
- GTCCRA, GTCCRC, and GTCCRD
- GTCCRB, GTCCRE, and GTCCRF

### 21.3.2.1 GTPR Register Buffer Operation

GTPBR can function as a buffer register for GTPR.

The buffer transfer is performed at an overflow (during up-counting) or an underflow (during down-counting) in saw-wave mode or in event count, and at a trough in triangle-wave mode.

In saw-wave mode or in event count, the buffer transfer is performed when the following counter clear operations occur during counting:

- Clear by hardware sources (the clear source is selected in GTCSR register)
- Clear by software (when GTCSR.CCLR bit is 1 and GTCLR.CCLRn bit is set to 1, n = 0 to 7).

To set GTPR to function as a buffer, set the GTBER.PR bit to 1. To set GTPR not to function as a buffer, set the GTBER.PR bit to 0.

Figure 21.11 to Figure 21.13 show examples of GTPR buffer operation and Table 21.12 shows an example for setting GTPR buffer operation.

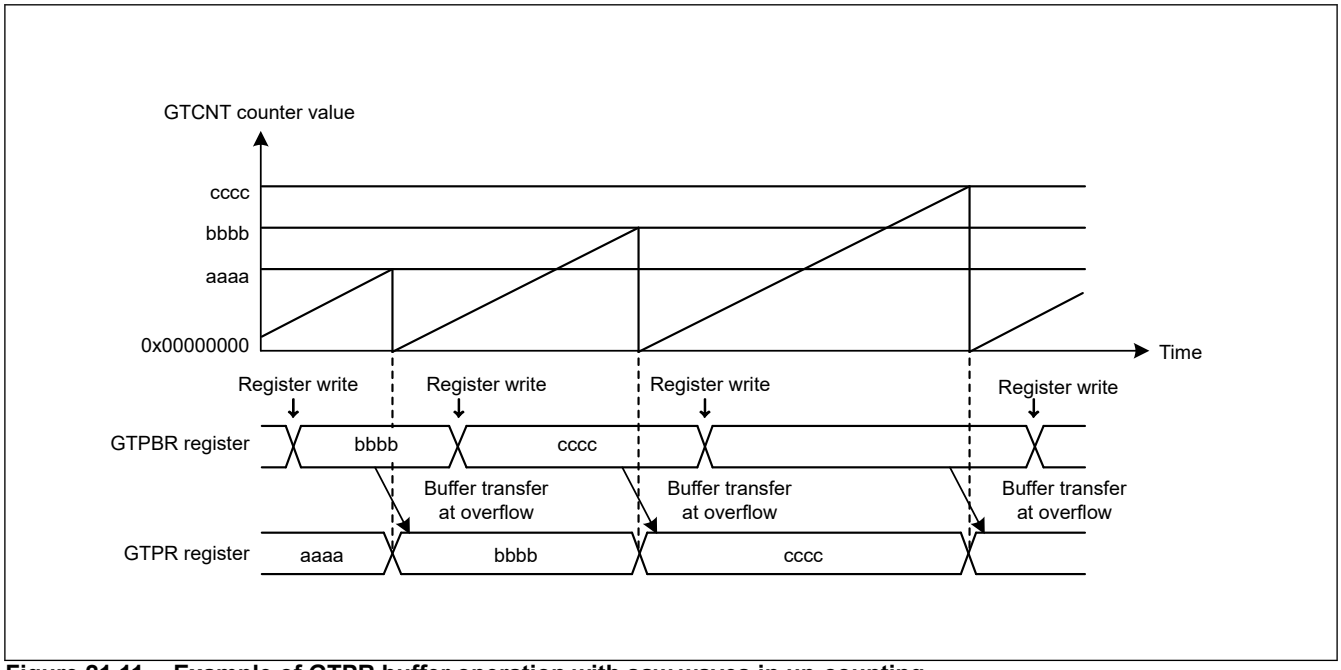



**Figure 21.11 Example of GTPR buffer operation with saw waves in up-counting**

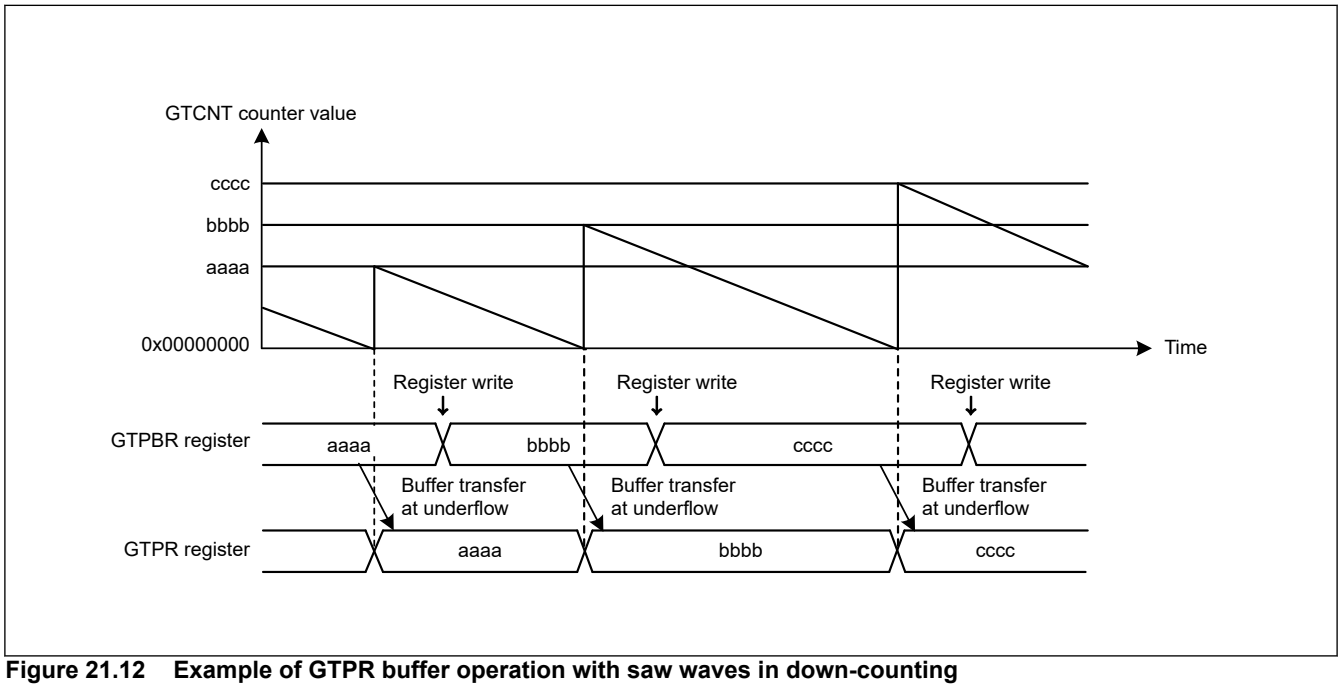



**Figure 21.12 Example of GTPR buffer operation with saw waves in down-counting**

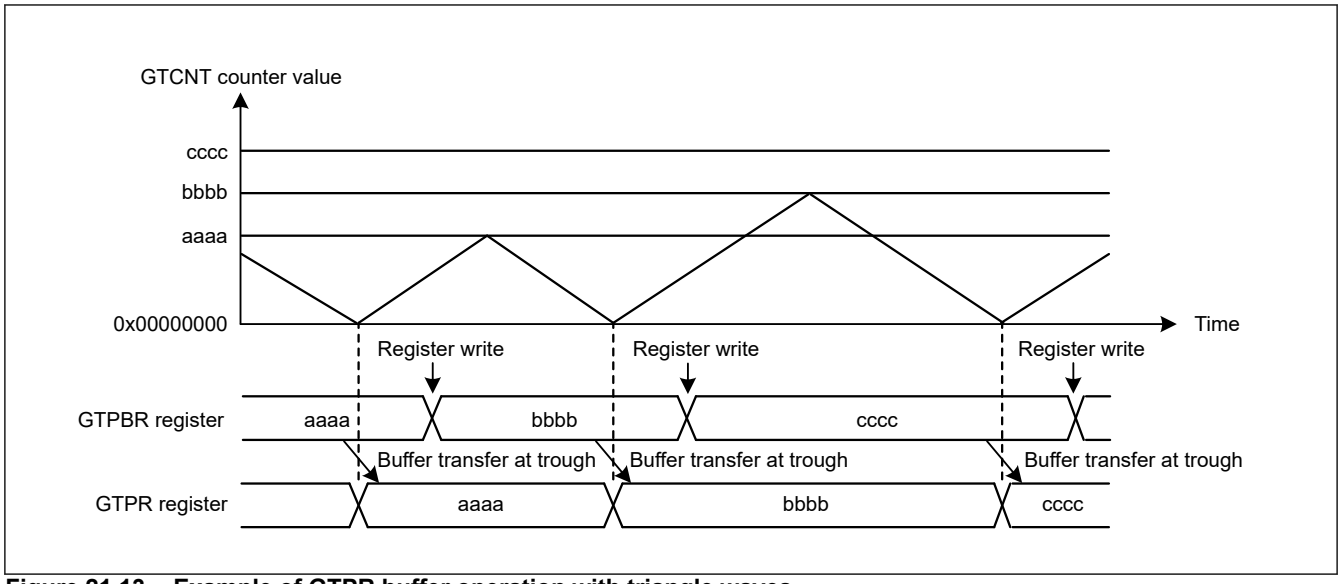



**Figure 21.13 Example of GTPR buffer operation with triangle waves**

**Table 21.12 Example for setting GTPR register buffer operation**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.11 and Figure 21.12, 000b (saw-wave PWM mode) is set, and in Figure 21.13, 100b (triangle-wave PWM mode 1) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.11, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). In Figure 21.12, after 10b is set in the GTUDDTYC[1:0] bits, 00b is set in the GTUDDTYC[1:0] bits (down-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 6 | Set buffer operation | Set buffer operation with the GTBER.PR[1:0] bits. In Figure 21.11, Figure 21.12, and Figure 21.13, PR[1:0] = 01b. |
| 7 | Set buffer value | For buffer operation, set a value in one cycle after the current cycle in the GTPBR register. |
| 8 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 9 | Set buffer value for each cycle | For buffer operation, set a value in one cycle after the current cycle in the GTPBR register. |

### 21.3.2.2 Buffer Operation for GTCCRA and GTCCRB Registers

GTCCRC can function as the GTCCRA buffer register and GTCCRD can function as the GTCCRC buffer register (double-buffer register for GTCCRA). Similarly, GTCCRE can function as the GTCCRB buffer register and GTCCRF can function as the GTCCRE buffer register (double-buffer register for GTCCRB).

To set GTCCRA or GTCCRB to function as a double buffer, set GTBER.CCRA[1:0] or GTBER.CCRB[1:0] to 10b or 11b. For single buffer operation, set 01b. To set GTCCRA or GTCCRB to not function as a buffer, set 00b.

#### (1) When GTCCRA or GTCCRB Functions as Output Compare Register

Buffer transfer occurs in the following situations:

- Buffer transfer by overflow or underflow
  Buffer transfer is performed at an overflow (during up-counting) or an underflow (during down-counting) in saw-wave mode or in event count operation. In triangle-wave mode, buffer transfer is performed at a trough (triangle-wave PWM mode 1) or a crest and trough (triangle-wave PWM mode 2).
- Buffer transfer by counter clear

In saw-wave mode or in event count operation, during counting, buffer transfer (which is the same as an overflow during up-counting or an underflow during down-counting) is performed by the counter clear sources similar to the case shown in section 21.3.2.1. GTPR Register Buffer Operation.
In triangle-wave mode, buffer transfer is not performed by the counter clear.

- Forcible buffer transfer
  When GTBER.CCRSWT bit is set to 1 while the count operation is stopped, the GTCCRA and the GTCCRB register buffer transfer are performed forcibly in saw-wave mode, in event count operation and in triangle-wave mode. Additionally buffer transfer from the GTCCRD register to temporary register A and from the GTCCRF register to temporary register B are performed in saw-wave one-shot pulse mode or triangle-wave PWM mode 3.

Figure 21.14 to Figure 21.16 show examples of GTCCRA and GTCCRB buffer operation and Table 21.13 shows an example for setting GTCCRA and GTCCRB buffer operation.

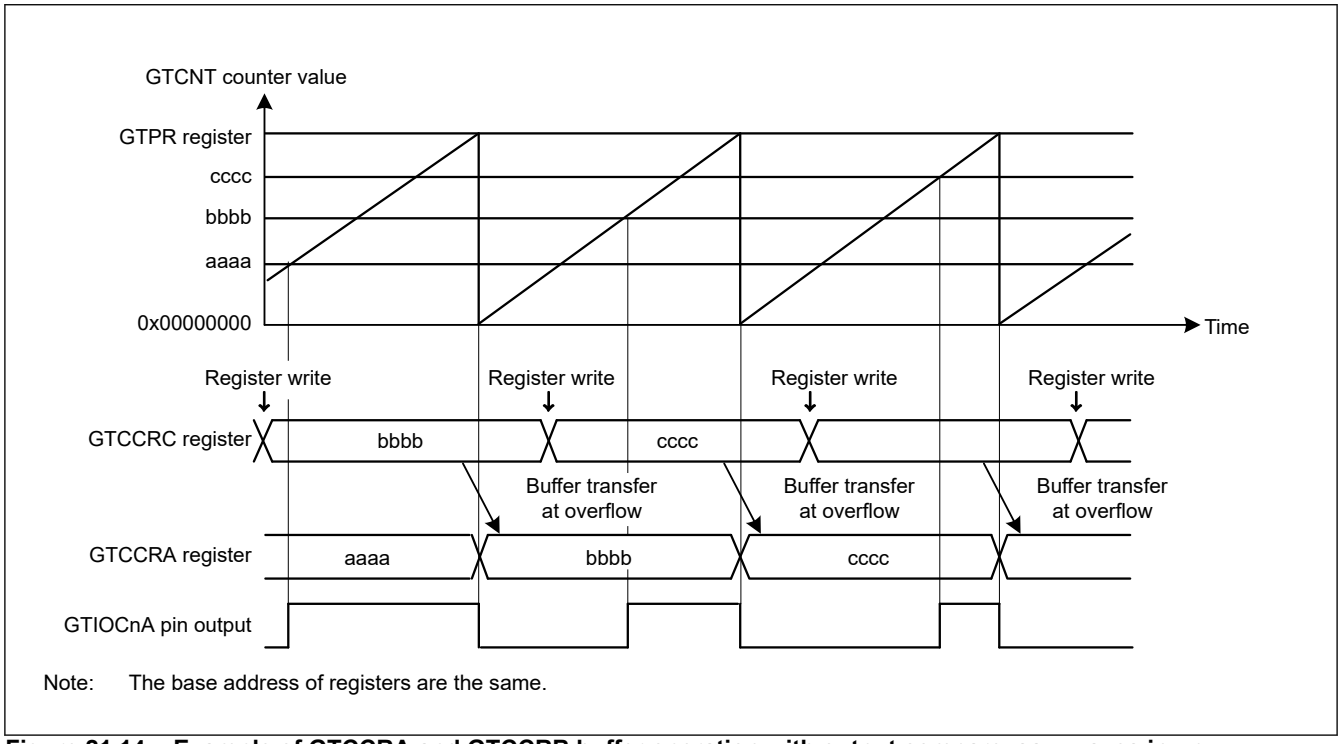
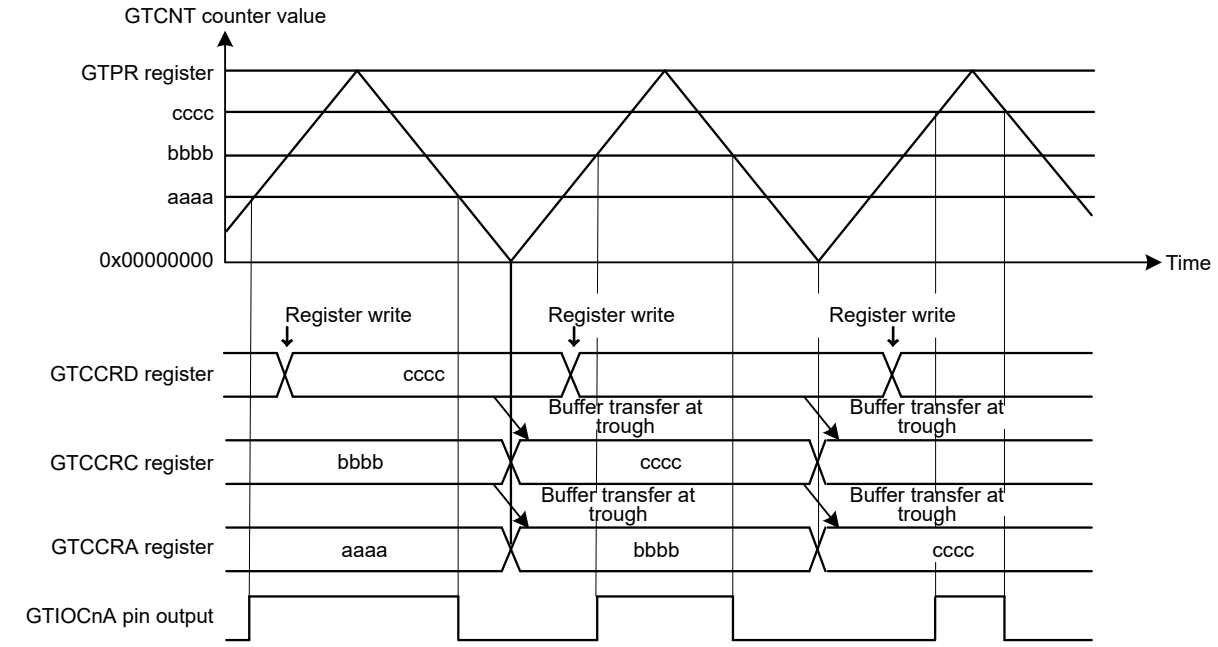
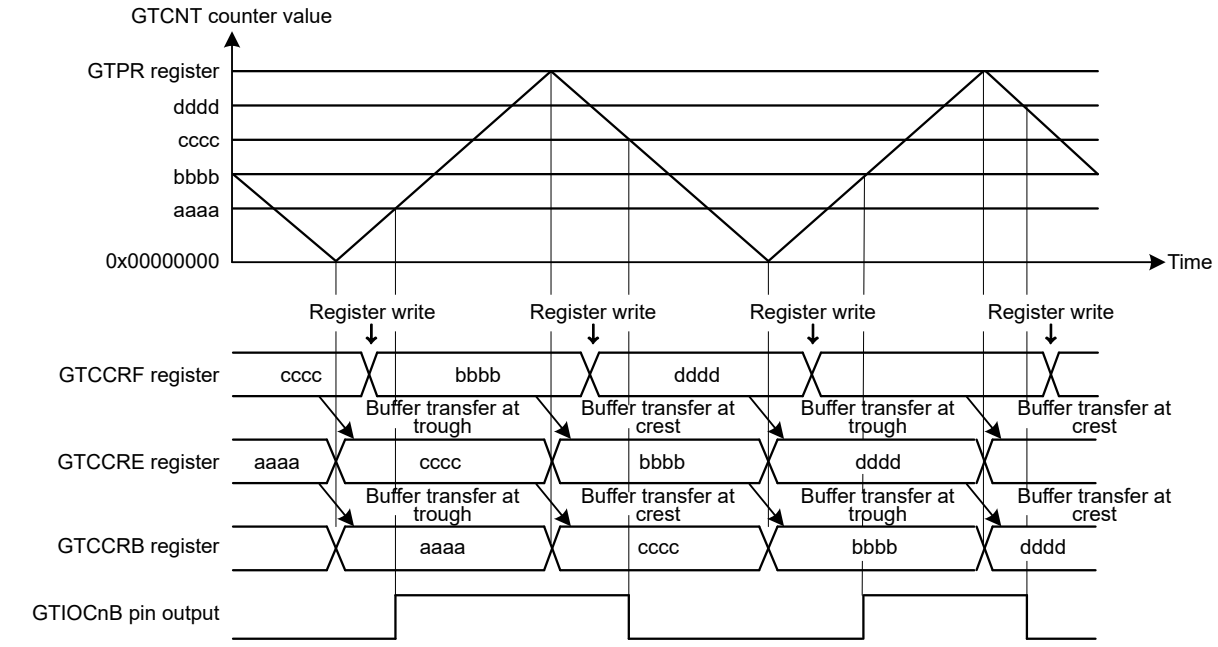



**Figure 21.14 Example of GTCCRA and GTCCRB buffer operation with output compare, saw waves in up-counting, high output at GTCCRA compare match, and low output at cycle end**




Note: The base address of registers are the same.

**Figure 21.15 Example of GTCCRA and GTCCRB double buffer operation with output compare, triangle waves, buffer operation at trough, output toggled at GTCCRA compare match, and output retained at cycle end**




Note: The base address of registers are the same.

**Figure 21.16 Example of GTCCRA and GTCCRB double buffer operation with output compare, triangle waves, buffer operation at both troughs and crests, output toggled at GTCCRB compare match, and output retained at cycle end**

**Table 21.13 Example for setting GTCCRA and GTCCRB buffer operation for output compare**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.14, 000b (saw-wave PWM mode) is set, in Figure 21.15, 100b (triangle-wave PWM mode 1) is set, and in Figure 21.16, 101b (triangle-wave PWM mode 2) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.14, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 6 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.14, GTIOA[4:0] = 00110b, in Figure 21.15, GTIOA[4:0] = 00011b, and in Figure 21.16, GTIOB[4:0] = 00011b. |
| 7 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 8 | Set buffer operation | Set buffer operation with the CCRA[1:0] and CCRB[1:0] bits in the GTBER register.<br>In Figure 21.14, CCRA[1:0] = 01b, in Figure 21.15, CCRA[1:0] = 1xb, and in Figure 21.16, CCRB[1:0] = 1xb. |
| 9 | Set compare match value | Set the GTIOCnA pin transition in the GTCCRA register and the GTIOCnB pin transition in the GTCCRB register. |
| 10 | Set buffer value | For buffer operation, set the GTIOCnA and GTIOCnB pins transitions in 1 cycle after the current cycle (in saw-wave mode or triangle-wave mode with buffer transfer at trough or crest) or half cycle after the current cycle (in triangle-wave mode with buffer transfer at both trough and crest) in the GTCCRC and GTCCRE registers, respectively.<br>For double buffer operation, also set the GTIOCnA and GTIOCnB pins transitions in 2 cycles after the current cycle (in saw-wave mode or triangle-wave mode with buffer transfer at trough or crest) or 1 cycle after the current cycle (in triangle-wave mode with buffer transfer at both trough and crest) in the GTCCRD and GTCCRF registers, respectively. |
| 11 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 12 | Set buffer value for each cycle | For buffer operation, set the GTIOCnA and GTIOCnB pins transitions in 1 cycle after the current cycle (in saw-wave mode or triangle-wave mode with buffer transfer at trough or crest) or half cycle after the current cycle (in triangle-wave mode with buffer transfer at both trough and crest) in the GTCCRC and GTCCRE registers, respectively.<br>For double buffer operation, also set the GTIOCnA and GTIOCnB pins transitions in 2 cycles after the current cycle (in saw-wave mode or triangle-wave mode with buffer transfer at trough or crest) or 1 cycle after the current cycle (in triangle-wave mode with buffer transfer at both trough and crest) in the GTCCRD and GTCCRF registers, respectively. |

Note: n: 0 to 7
m: A, B

## (2) When GTCCRA or GTCCRB Functions as Input Capture Register

When an input capture is generated, the GTCNT counter value is transferred to GTCCRA and GTCCRB and the stored GTCCRA and GTCCRB register values are transferred to the buffer registers. In input capture operation, the buffer transfer is not performed by the counter clear.

Figure 21.17 and Figure 21.18 show examples of GTCCRA and GTCCRB buffer operation and Table 21.14 shows an example for setting GTCCRA and GTCCRB buffer operation.

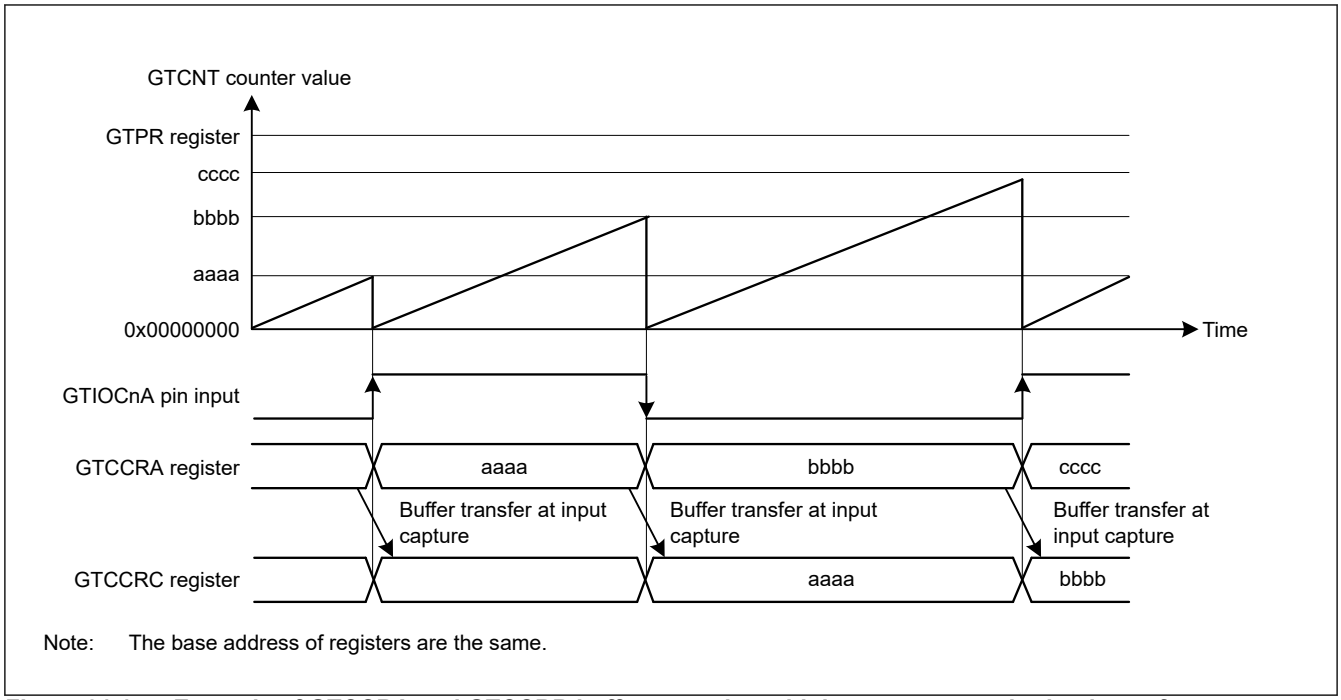



**Figure 21.17 Example of GTCCRA and GTCCRB buffer operation with input capture at both edges of GTIOCnA input, saw waves in up-counting, and GTCNT counter cleared at both edges of GTIOCnA input**

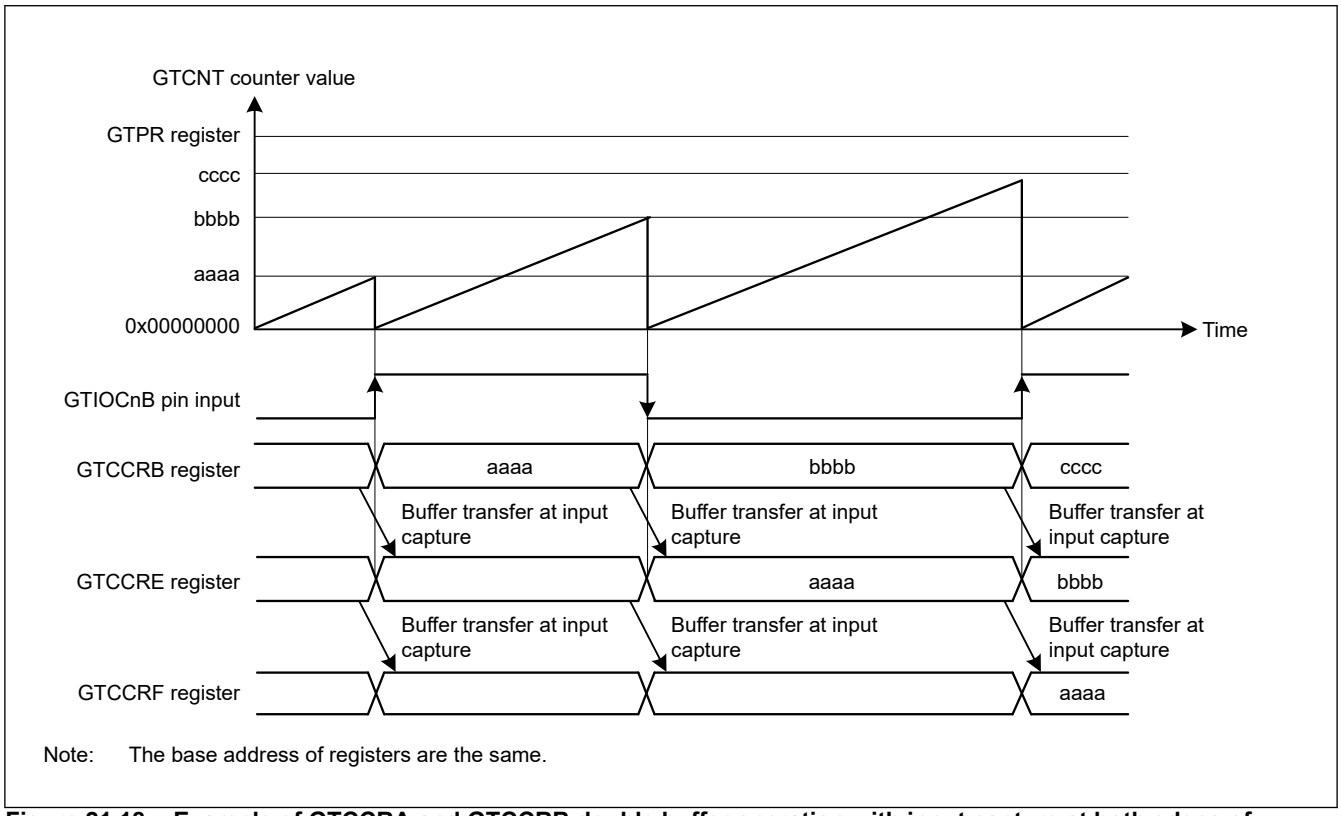



**Figure 21.18 Example of GTCCRA and GTCCRB double buffer operation with input capture at both edges of GTIOCnB input, saw waves in up-counting, and GTCNT counter cleared at both edges of GTIOCnB input**

**Table 21.14 Example for setting GTCCRA and GTCCRB buffer operation for input capture**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode and counter clear sources | Set the operating mode with the GTCR.MD[2:0] bits and count clear source with the GTCSR register.<br>In Figure 21.17, MD[2:0] = 000b (saw-wave PWM mode) and GTCSR = 0x00000F00, and in Figure 21.18, MD[2:0] = 000b (saw-wave PWM mode) and GTCSR = 0x0000F000. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.17, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 6 | Select input capture source | Select input capture source in the GTICASR register and GTICBSR register.<br>In Figure 21.17, GTICASR = 0x00000F00, and in Figure 21.18, GTICBSR = 0x0000F000. |
| 7 | Set buffer operation | Set buffer operation with the CCRA and CCRB bits in the GTBER register.<br>In Figure 21.17, CCRA[1:0] = 01b, and in Figure 21.18, CCRB[1:0] = 1xb. |
| 8 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |

### 21.3.3 PWM Output Operating Mode

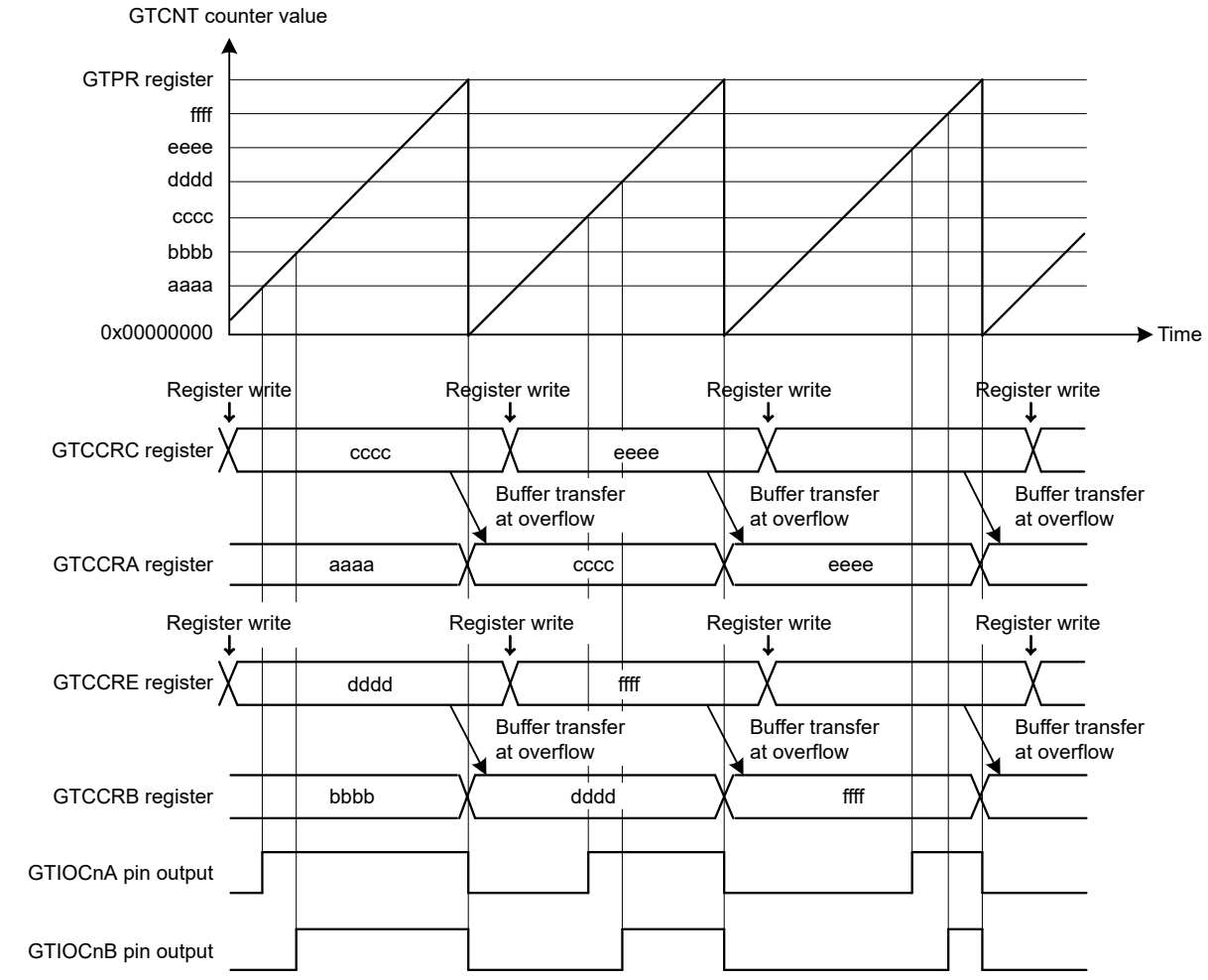
The GPT can output PWM waveforms to the GTIOCnA or GTIOCnB pin (n = 0 to 7) by a compare match between the GTCNT counter and GTCCRA or GTCCRB.

By setting GTDTCR and GTDVU, the compare match value for a negative-phase waveform with dead time can automatically be set to GTCCRB.

#### 21.3.3.1 Saw-Wave PWM Mode

In saw-wave PWM mode, GTCNT performs saw-wave (half-wave) operation by setting the cycle in GTPR and a PWM waveform is output to the GTIOCnA or GTIOCnB pin (n = 0 to 7) when a GTCCRA or GTCCRB compare match occurs. The pin output value can be selected from low output, high output, or toggle output separately for a compare match and for the cycle end according to the GTIOR setting.

Figure 21.19 shows an example of saw-wave PWM mode operation, and Table 21.15 shows an example for setting saw-wave PWM mode.




Note: The base address of registers are the same.

**Figure 21.19 Example of saw-wave PWM mode operation with up-counting, buffer operation, high output at GTCCRA/GTCCRB compare match, and low output at cycle end**

**Table 21.15 Example for setting saw-wave PWM mode (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits. In Figure 21.19, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.19, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 6 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.19, GTIOA[4:0] = 00110b and GTIOB[4:0] = 00110b. |
| 7 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 8 | Set buffer operation | Set buffer operation with the CCRA[1:0] and CCRB[1:0] bits in the GTBER register. In Figure 21.19, CCRA[1:0] = 01b and CCRB[1:0] = 01b. |
| 9 | Set compare match value | Set the GTIOCnA pin transition in the GTCCRA register and the GTIOCnB pin transition in the GTCCRB register. |

**Table 21.15 Example for setting saw-wave PWM mode (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 10 | Set buffer value | For buffer operation, set the GTIOCnA and GTIOCnB pins transitions in 1 cycle after the current cycle in the GTCCRC and GTCCRE registers, respectively.<br>For double buffer operation, also set the GTIOCnA and GTIOCnB pins transitions in 2 cycles after the current cycle in the GTCCRD and GTCCRF registers, respectively. |
| 11 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 12 | Set buffer value for each cycle | For buffer operation, set the GTIOCnA and GTIOCnB pins transitions in 1 cycle after the current cycle in the GTCCRC and GTCCRE registers, respectively.<br>For double buffer operation, also set the GTIOCnA and GTIOCnB pins transitions in 2 cycles after the current cycle in the GTCCRD and GTCCRF registers, respectively. |

Note: n: 0 to 7
m: A, B

### 21.3.3.2 Saw-Wave One-Shot Pulse Mode

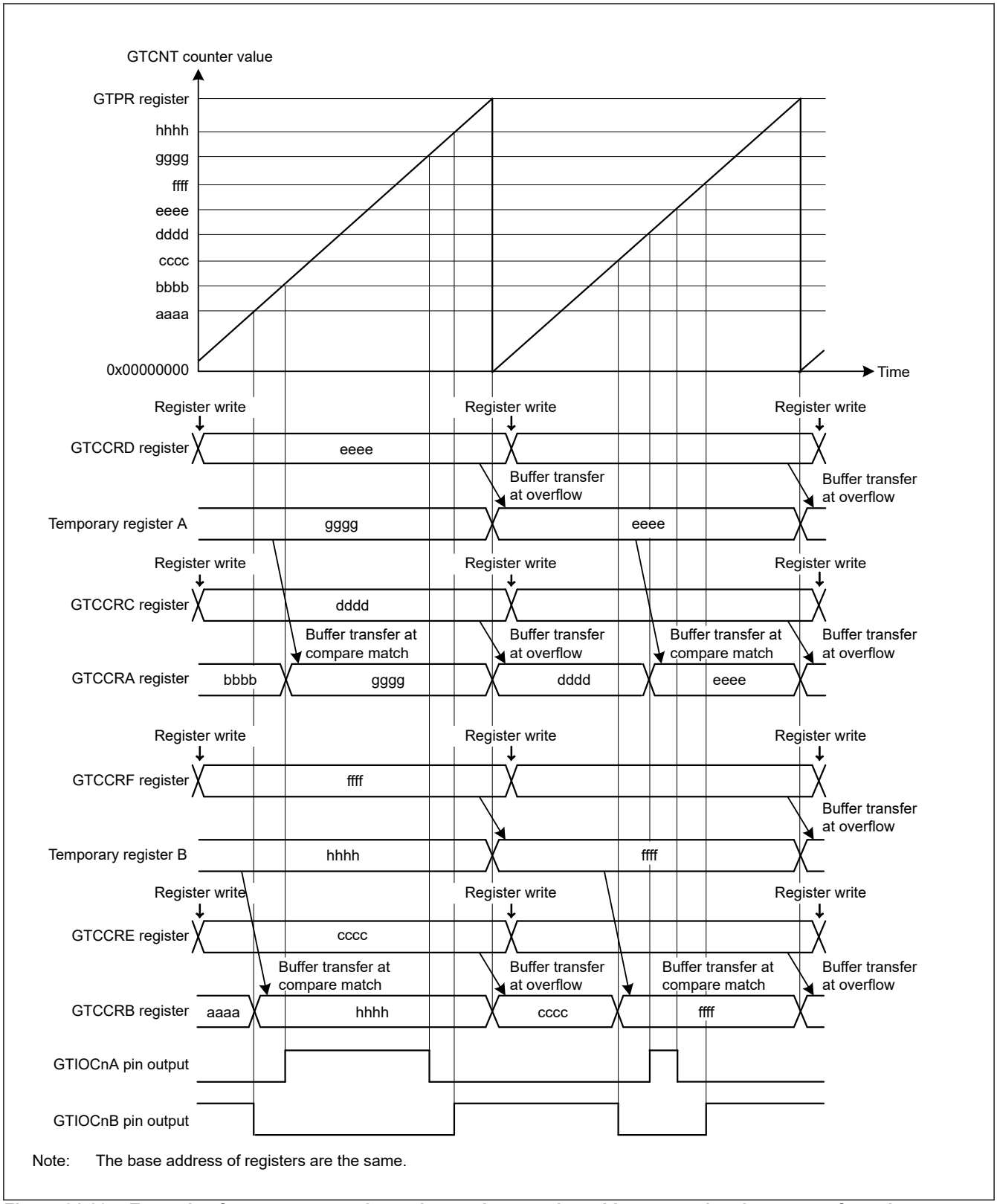
The saw-wave one-shot pulse mode is a mode in which the cycle is set in GTPR, the GTCNT counter performs saw-wave (half-wave) operation and a PWM waveform is output to the GTIOCnA or GTIOCnB pin (n = 0 to 7) at a compare match of GTCCRA or GTCCRB with buffer operation fixed.

Buffer operation in saw-wave one-shot pulse mode is different from the usual buffer operation. Buffer transfer is performed from:

- GTCCRC to GTCCRA at the cycle end
- GTCCRE to GTCCRB at the cycle end
- GTCCRD to temporary register A at the cycle end
- GTCCRF to temporary register B at the cycle end
- Temporary register A to GTCCRA at a GTCCRA compare match
- Temporary register B to GTCCRB at a GTCCRB compare match.

The pin output value can be selected from low output, high output, or toggled output separately for a compare match and the cycle end according to the GTIOR setting. When the GTBER.CCRSWT bit is set to 1 while count operation is stopped, the buffer is transferred forcibly from the GTCCRD register to temporary register A and from the GTCCRF register to temporary register B. By setting GTDTCR, GTDVU, a compare match value for a negative-phase waveform with dead time can automatically be set to GTCCRB.

Figure 21.20 shows an example of saw-wave one-shot pulse mode operation, and Table 21.16 shows an example for setting saw-wave one-shot pulse mode.




Note: The base address of registers are the same.

**Figure 21.20 Example of saw-wave one-shot pulse mode operation with up-counting, low output from the GTIOCnA pin and high output from the GTIOCnB pin at count start, output toggled at GTCCRA/ GTCCRB compare match, and output retained at cycle end**

**Table 21.16 Example setting for saw-wave one-shot pulse mode**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.20, 001b (saw-wave one-shot pulse mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.20, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 6 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.20, GTIOA[4:0] = 00011b and GTIOB[4:0] = 10011b. |
| 7 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 8 | Set compare match value | Set the GTIOCnA pin transition immediately after the count start in the GTCCRC and GTCCRD registers and the GTIOCnB pin transition in the GTCCRE and GTCCRF registers. |
| 9 | Set forcible buffer transfer | Set the GTBER.CCRSWT bit to 1 to transfer buffer register data forcibly. |
| 10 | Set buffer value | For buffer operation, set the GTIOCnA pin transition in one cycle after the current cycle in the GTCCRC and GTCCRD registers and the GTIOCnB pin transition in the GTCCRE and GTCCRF registers. |
| 11 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 12 | Set buffer value for each cycle | For buffer operation, set the GTIOCnA pin transition in one cycle after the current cycle in the GTCCRC and GTCCRD registers and the GTIOCnB pin transition in the GTCCRE and GTCCRF registers. |

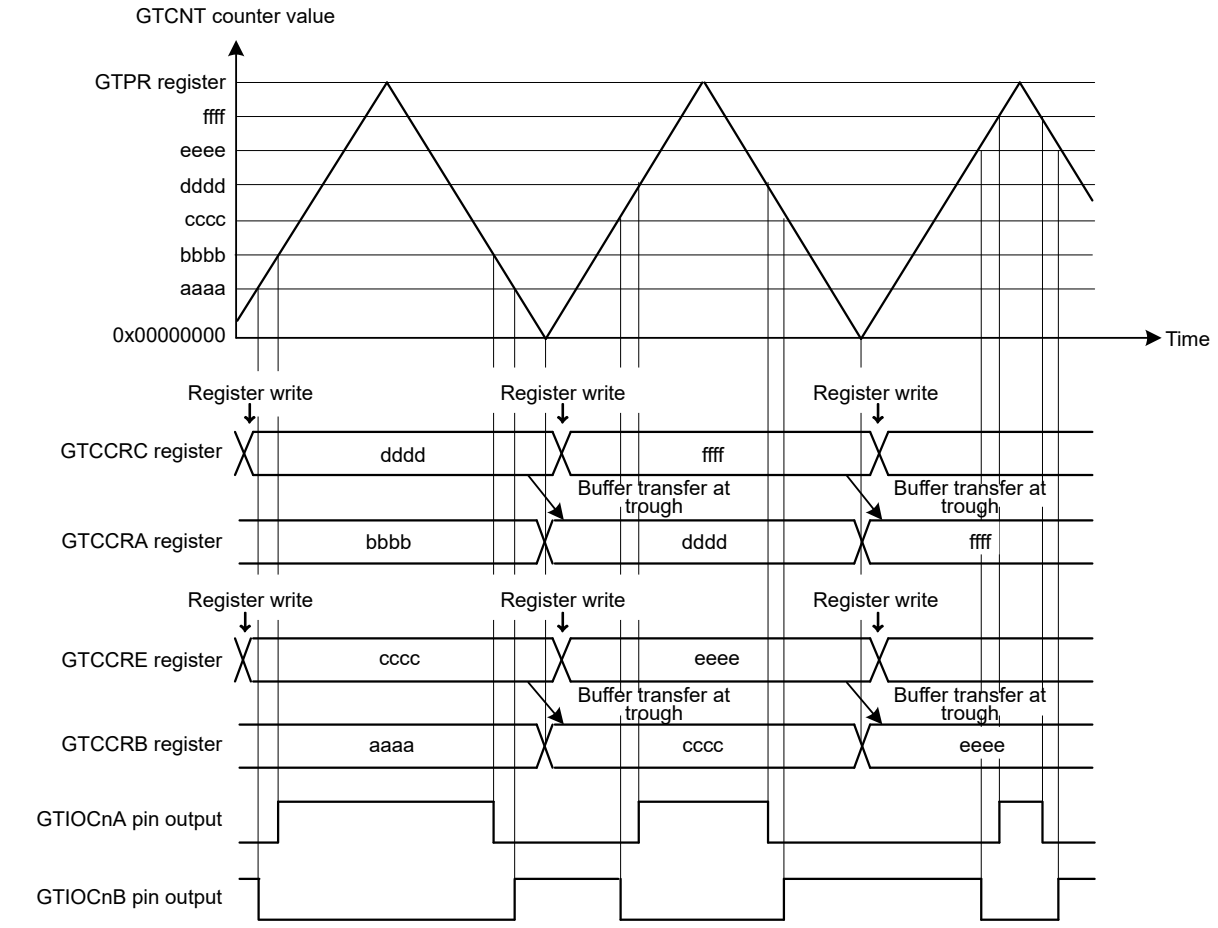
Note: n: 0 to 7
m: A, B

### 21.3.3.3 Triangle-Wave PWM Mode 1 (32-Bit Transfer at Trough)

The triangle-wave PWM mode 1 is a mode in which the cycle is set in GTPR. The GTCNT counter performs triangle-wave (full-wave) operation, and a PWM waveform is output to the GTIOCnA or GTIOCnB pin (n = 0 to 7) when a GTCCRA or GTCCRB compare match occurs. Buffer transfer is performed at the trough. The pin output value can be selected from low output, high output, or toggled output separately for a compare match and for the cycle end according to the GTIOR setting.

By setting GTDTCR, GTDVU, a compare match value for a negative-phase waveform with dead time can automatically be set to GTCCRB.

Figure 21.21 shows an example of a triangle-wave PWM mode 1 operation, and Table 21.17 shows an example for setting a triangle-wave PWM mode 1.




Note: The base address of registers are the same.

**Figure 21.21 Example of triangle-wave PWM mode 1 operation with buffer operation, low output from the GTIOCnA pin and high output from the GTIOCnB pin at count start, output toggled at GTCCRA/ GTCCRB register compare match, and output retained at cycle end**

**Table 21.17 Example setting for triangle-wave PWM mode 1 (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.21, 100b (triangle-wave PWM mode 1) is set. |
| 2 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 3 | Set cycle | Set the cycle in the GTPR register. |
| 4 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 5 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.21, GTIOA[4:0] = 00011b and GTIOB[4:0] = 10011b. |
| 6 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 7 | Set buffer operation | Set buffer operation with the CCRA[1:0] and CCRB[1:0] bits in the GTBER register.<br>In Figure 21.21, CCRA[1:0] = 01b and CCRB[1:0] = 01b. |
| 8 | Set compare match value | Set the GTIOCnA and GTIOCnB pins transitions in the GTCCRA and GTCCRB registers, respectively. |
| 9 | Set buffer value | For buffer operation, set the GTIOCnA and GTIOCnB pins transitions in 1 cycle after the current cycle in the GTCCRC and GTCCRE registers, respectively.<br>For double buffer operation, also set the GTIOCnA and GTIOCnB pins transitions in 2 cycles after the current cycle in the GTCCRD and GTCCRF registers, respectively. |

**Table 21.17 Example setting for triangle-wave PWM mode 1 (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 10 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 11 | Set buffer value for each cycle | For buffer operation, set the GTIOCnA and GTIOCnB pins transitions in 1 cycle after the current cycle in the GTCCRC and GTCCRE registers, respectively.<br>For double buffer operation, also set the GTIOCnA and GTIOCnB pins transitions in 2 cycles after the current cycle in the GTCCRD and GTCCRF registers, respectively. |

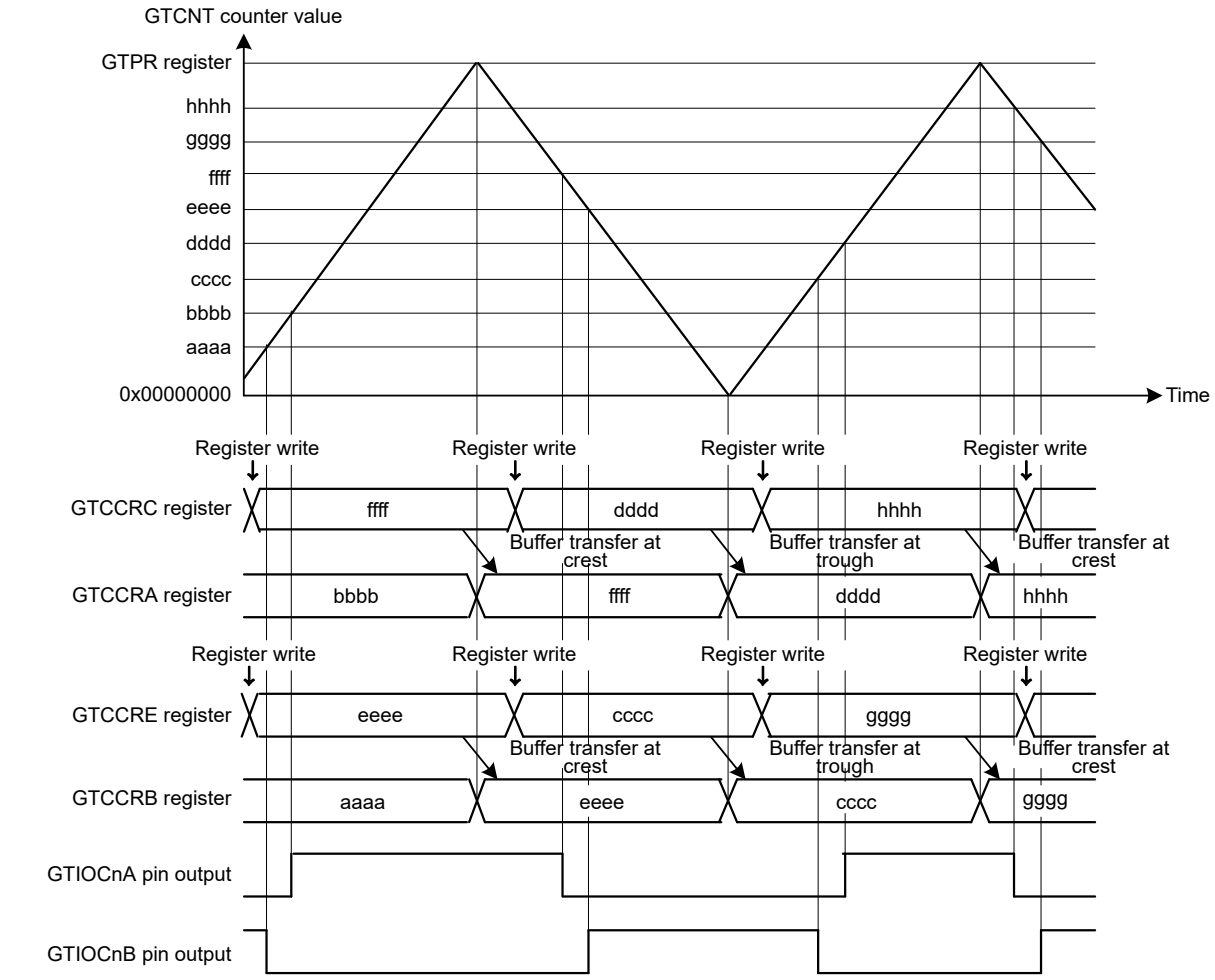
Note: n: 0 to 7
m: A, B

### 21.3.3.4 Triangle-Wave PWM Mode 2 (32-Bit Transfer at Crest and Trough)

Similarly to triangle-wave PWM mode 1, in triangle-wave PWM mode 2 the cycle is set in GTPR. The GTCNT counter performs triangle-wave (full-wave) operation, and a PWM waveform is output to the GTIOCnA or GTIOCnB pin (n = 0 to 7) when a GTCCRA or GTCCRB compare match occurs. The buffer transfer is performed at both crests and troughs. The pin output value can be selected from low output, high output, or toggle output separately for a compare match and for the cycle end according to the GTIOR setting.

By setting GTDTCR, GTDVU, a compare match value for a negative-phase waveform with dead time can automatically be set to GTCCRB.

Figure 21.22 shows an example of triangle-wave PWM mode 2 operation, and Table 21.18 shows an example for setting triangle-wave PWM mode 2.




Note: The base address of registers are the same.

**Figure 21.22 Example of triangle-wave PWM mode 2 operation with buffer operation, low output from the GTIOCnA pin and high output from the GTIOCnB pin at count start, output toggled at GTCCRA/ GTCCRB compare match, and output retained at cycle end**

**Table 21.18 Example for setting triangle-wave PWM mode 2 (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.22, 101b (triangle-wave PWM mode 2) is set. |
| 2 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 3 | Set cycle | Set the cycle in the GTPR register. |
| 4 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 5 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.22, GTIOA[4:0] = 00011b and GTIOB[4:0] = 10011b. |
| 6 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 7 | Set buffer operation | Set buffer operation with the CCRA[1:0] and CCRB[1:0] bits in the GTBER register.<br>In Figure 21.22, CCRA[1:0] = 01b and CCRB[1:0] = 01b. |
| 8 | Set compare match value | Set the GTIOCnA and GTIOCnB pins transitions in the GTCCRA and GTCCRB registers, respectively. |

**Table 21.18 Example for setting triangle-wave PWM mode 2 (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 9 | Set buffer value | For buffer operation, set the GTIOCnA and GTIOCnB pins transitions in half cycle after the current cycle in the GTCCRC and GTCCRE registers, respectively.<br>For double buffer operation, also set the GTIOCnA and GTIOCnB pins transitions in 1 cycle after the current cycle in the GTCCRD and GTCCRF registers, respectively. |
| 10 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 11 | Set buffer value for each half cycle | For buffer operation, set the GTIOCnA and GTIOCnB pins transitions in half cycle after the current cycle in the GTCCRC and GTCCRE registers, respectively.<br>For double buffer operation, also set the GTIOCnA and GTIOCnB pins transitions in 1 cycle after the current cycle in GTCCRD and GTCCRF registers, respectively. |

Note: n: 0 to 7
m: A, B

### 21.3.3.5 Triangle-Wave PWM Mode 3 (64-Bit Transfer at Trough)

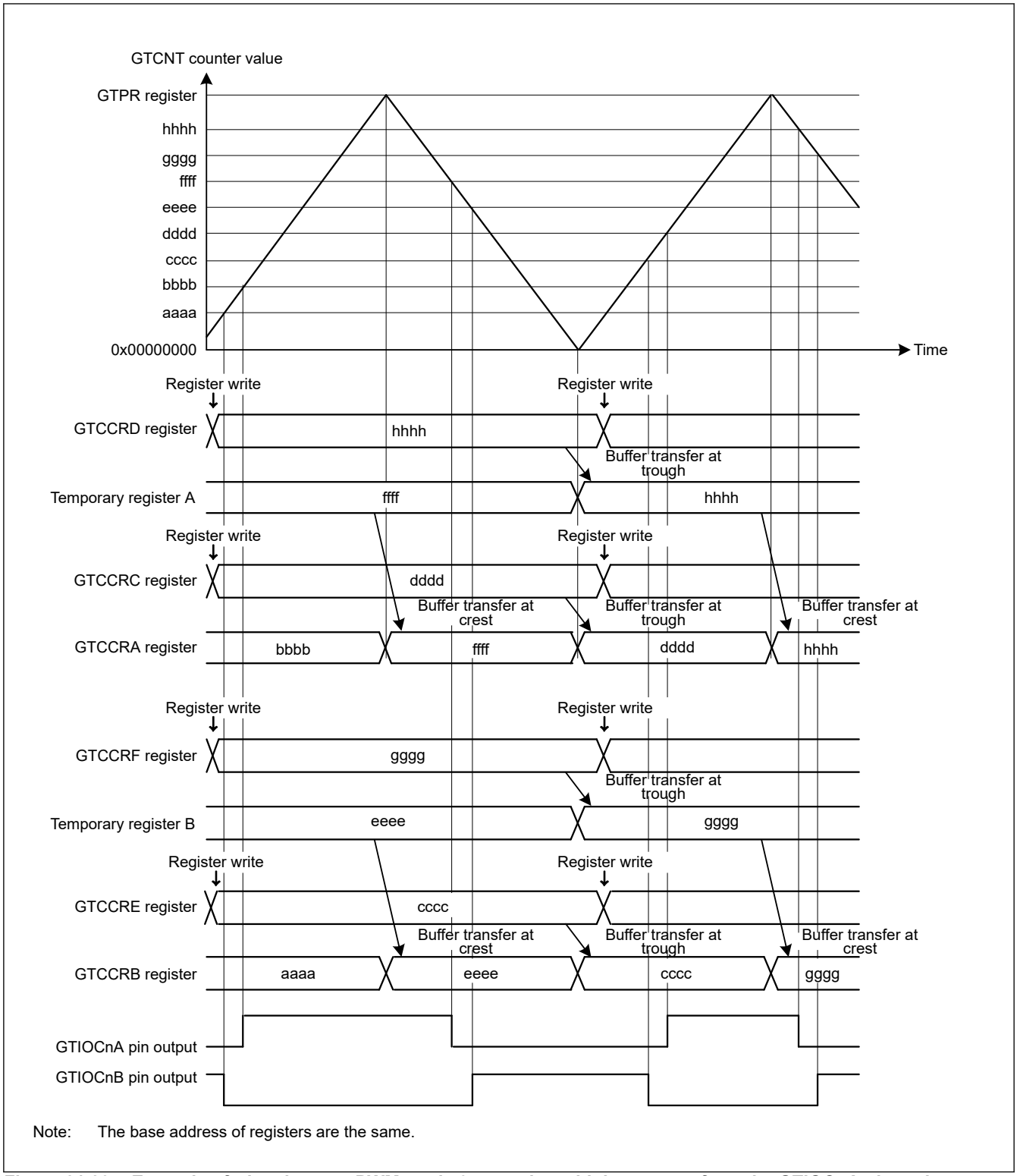
The triangle-wave PWM mode 3 is a mode in which the cycle is set in GTPR. The GTCNT counter performs triangle-wave (full-wave) operation and a PWM waveform is output to the GTIOCnA or GTIOCnB pin (n = 0 to 7) at a compare match of GTCCRA or GTCCRB with buffer operation fixed. Buffer operation in triangle-wave PWM mode 3 is different from the usual buffer operation. Buffer transfer is performed from the following:

- GTCCRC to GTCCRA at the trough
- GTCCRE to GTCCRB at the trough
- GTCCRD to temporary register A at the trough
- GTCCRF to temporary register B at the trough
- Temporary register A to GTCCRA at the crest
- Temporary register B to GTCCRB at the crest.

The pin output value can be selected from low output, high output, or toggled output separately for a compare match and for the cycle end according to the GTIOR setting.

By setting GTDTCR, GTDVU, a compare match value for a negative-phase waveform with dead time can automatically be set to GTCCRB.

Figure 21.23 shows an example of triangle-wave PWM mode 3 operation, and Table 21.19 shows an example for setting triangle-wave PWM mode 3.




Note: The base address of registers are the same.

**Figure 21.23 Example of triangle-wave PWM mode 3 operation with low output from the GTIOCnA pin and high output from the GTIOCnB pin at count start, output toggled at GTCCRA/GTCCRB compare match, and output retained at cycle end**

**Table 21.19 Example setting for triangle-wave PWM mode 3 (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.23, 110b (triangle-wave PWM mode 3) is set. |

**Table 21.19 Example setting for triangle-wave PWM mode 3 (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 2 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 3 | Set cycle | Set the cycle in the GTPR register. |
| 4 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 5 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register. In Figure 21.23, GTIOA[4:0] = 00011b and GTIOB[4:0] = 10011b. |
| 6 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 7 | Set compare match value | Set the GTIOCnA pin transition immediately after the count start in the GTCCRC and GTCCRD registers and the GTIOCnB pin transition in the GTCCRE and GTCCRF registers. |
| 8 | Set forcible buffer transfer | Set the GTBER.CCRSWT bit to 1 to transfer buffer register data forcibly. |
| 9 | Set buffer value | Set the GTIOCnA pin transition in 1 cycle after the current cycle in the GTCCRC and GTCCRD registers and the GTIOCnB pin transition in the GTCCRE and GTCCRF registers. |
| 10 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 11 | Set buffer value for each cycle | Set the GTIOCnA pin transition in 1 cycle after the current cycle in the GTCCRC and GTCCRD registers and the GTIOCnB pin transition in the GTCCRE and GTCCRF registers. |

Note: n: 0 to 7
m: A, B

### 21.3.4 Automatic Dead Time Setting Function

By setting GTDTCR, a compare match value for a negative waveform with dead time obtained by a compare match value for a positive waveform (GTCCRA value) and specified dead time value (GTDVU value) can automatically be set to GTCCRB. The automatic dead time setting function can be used in saw-wave one-shot pulse mode and all the triangle PWM modes.

Writing to GTCCRB is prohibited when the automatic dead time setting function is used. Dead time setting beyond the cycle is also prohibited. Values for automatic dead time setting can be read from GTCCRB. In triangle-wave mode, when the dead time is beyond the cycle by setting the value GTCCRA = 0 or GTCCRA ≥ GTPR for GTCCRA, the output protection function keeps the level of output. For details, see section 21.7.3. GTIOCnm Pin Output Negate Control (n = 0 to 7, m = A, B). The automatic dead time value setting to GTCCRB is performed at the next clock cycle count when registers that are used for calculating the automatic dead time value are updated.

When a dead time error occurs, the compare match values for positive and negative waveforms are adjusted to generate the waveforms with the dead time as shown in Table 21.20.

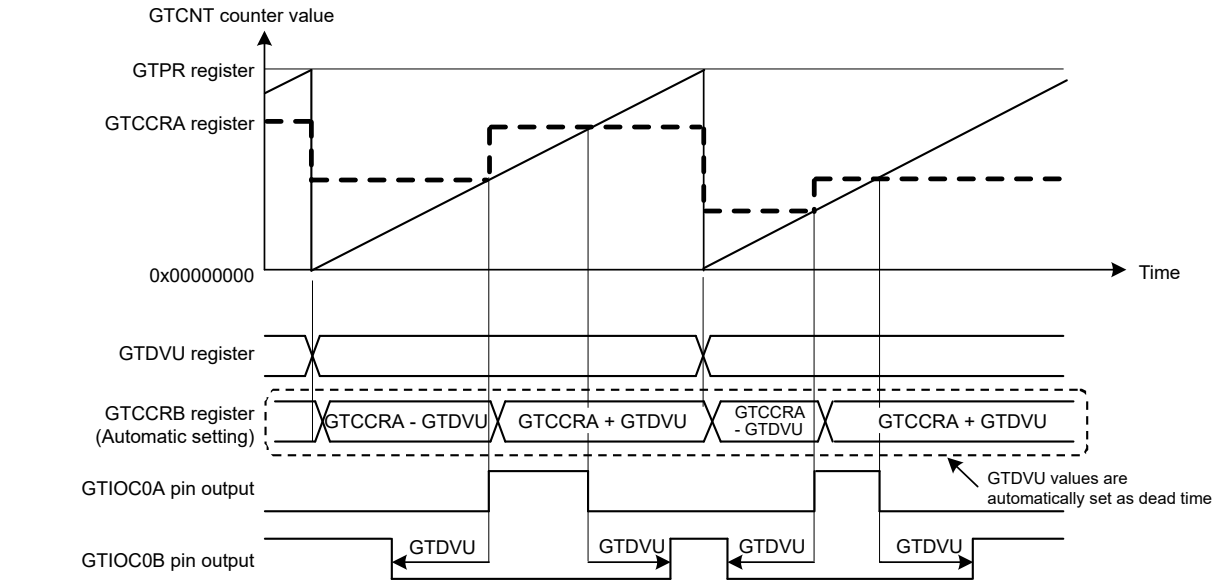
The adjusted value for the negative waveform is set for GTCCRB automatically.

The adjusted value for the positive waveform is used as internal signal and not set for GTCCRA.

**Table 21.20 Adjustment of the Waveform Change Point When a Dead-Time Error Occurs**

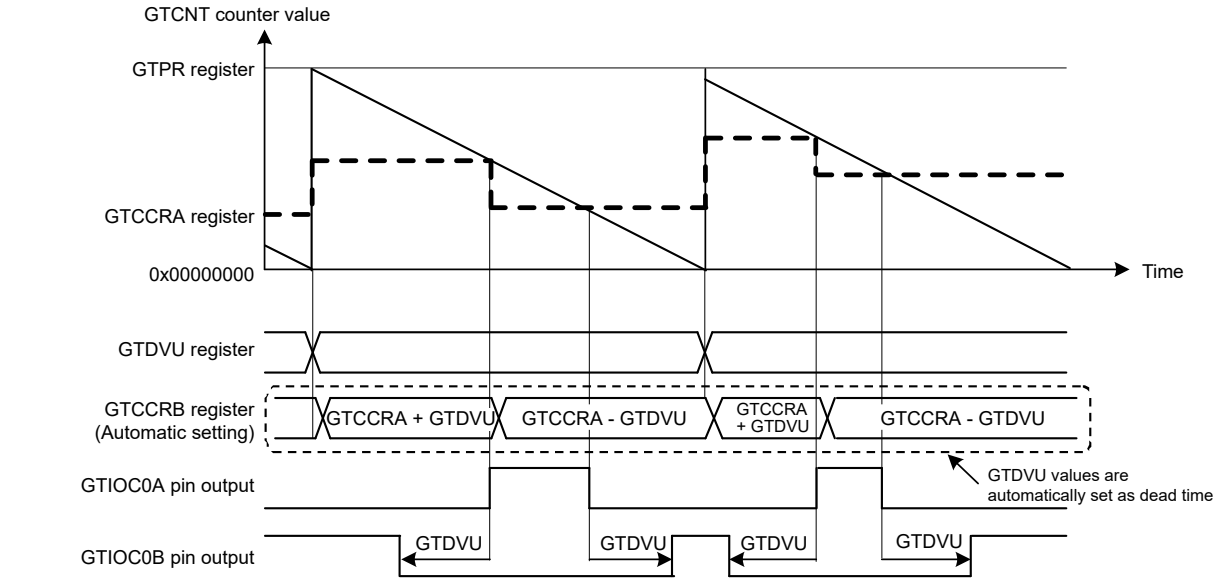
| Mode | Count Direction | Period | Condition for Dead Time Error | Change Point of the Positive-Phase Waveform after Adjustment | Change Point of the Negative-Phase Waveform after Adjustment |
|---|---|---|---|---|---|
| Sawtooth-wave one-shot pulse mode | Up-counting | First half | GTCCRA - GTDVU < 0 | GTDVU | 0 |
| | | Second half | GTCCRA + GTDVU > GTPR | GTPR - GTDVU | GTPR |
| | Down-counting | First half | GTCCRA + GTDVU > GTPR | GTPR - GTDVU | GTPR |
| | | Second half | GTCCRA - GTDVU < 0 | GTDVU | 0 |
| Triangle-wave PWM mode 1/2/3 | Up-counting | (First half) | GTCCRA - GTDVU ≤ 0 | GTDVU + 1 | 1 |
| | Down-counting | (Second half) | GTCCRA - GTDVU < 0 | GTDVU | 0 |

Figure 21.24 to Figure 21.27 show examples of automatic dead time setting function operation. Table 21.21 and Table 21.22 show the setting examples.



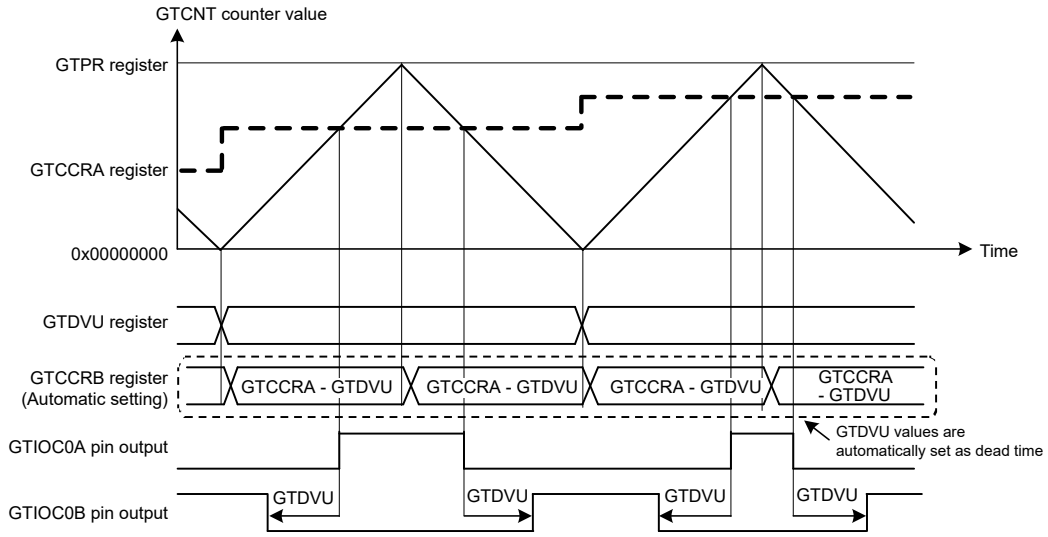

Note: The base address of registers are the same.

**Figure 21.24 Example of automatic dead time setting function operation in saw-wave one-shot pulse mode, up-counting, and active-high**



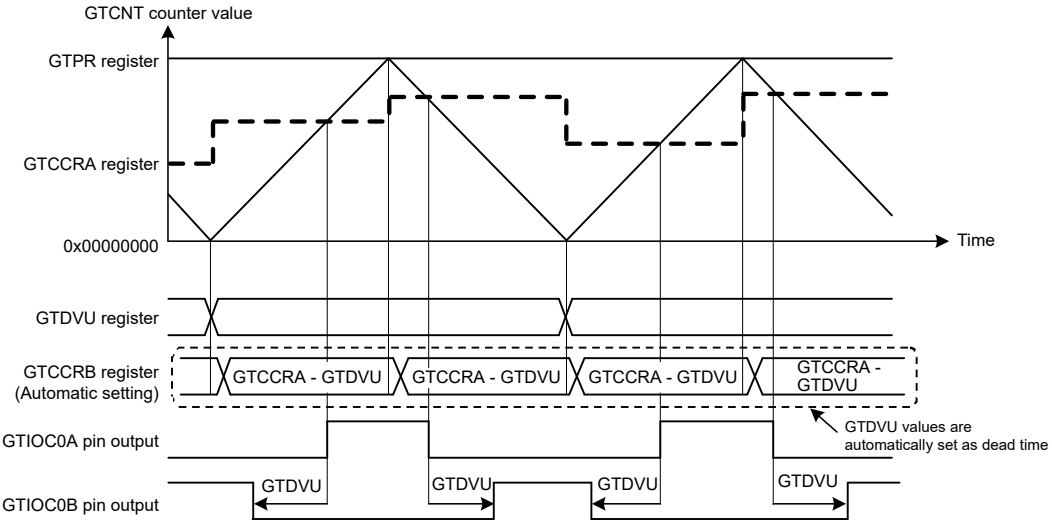

Note: The base address of registers are the same.

**Figure 21.25 Example of automatic dead time setting function operation in saw-wave one-shot pulse mode, down-counting, and active-high**




Note: The base address of registers are the same.

**Figure 21.26 Example of automatic compare-match value setting function with dead time in triangle-wave PWM mode 1, and active-high**




Note: The base address of registers are the same.

**Figure 21.27 Example of automatic compare-match value setting function with dead time in triangle-wave PWM mode 2 or 3, and active-high**

**Table 21.21 Example setting for automatic dead time setting function in saw-wave one-shot pulse mode, and triangle-wave PWM mode 3 (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits. In Figure 21.24 and Figure 21.25, 001b (saw-wave one-shot pulse mode) is set. In Figure 21.27, 110b (triangle-wave PWM mode 3) is set. |

**Table 21.21 Example setting for automatic dead time setting function in saw-wave one-shot pulse mode, and triangle-wave PWM mode 3 (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.24, 01b is set after 11b is set in the GTUDDTYC[1:0] bits (up count). In Figure 21.25, 00b is set after 10b is set in the GTUDDTYC[1:0] bits (down count). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 6 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.24, Figure 21.25, and Figure 21.27, GTIOA[4:0] = 00011b and GTIOB[4:0] = 10011b. |
| 7 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 8 | Set buffer value for compare match | Set the GTIOCnA pin transition immediately after the count start in the GTCCRC and GTCCRD registers. |
| 9 | Set forcible buffer transfer for compare match | Set the GTBER.CCRSWT bit to 1 to transfer buffer register data forcibly to the GTCCRA register. |
| 10 | Set buffer value for compare match | Set the GTIOCnA pin transition in 1 cycle after the current cycle in the GTCCRC and GTCCRD registers. |
| 11 | Set automatic dead time setting function | Set the GTDTCR.TDE bit to 1 to enable the automatic dead time setting function. |
| 12 | Set dead time value | Set the dead time value in the GTDVU register. |
| 13 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 14 | Set buffer value for each cycle | For buffer operation, set the GTIOCnA pin transition in 1 cycle after the current cycle in the GTCCRC and GTCCRD registers. |

Note: n: 0 to 7
m: A, B

**Table 21.22 Example setting for automatic dead time setting function in triangle-wave PWM mode 1 or 2**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits. In Figure 21.26, 100b (triangle-wave PWM mode 1) is set. In Figure 21.27, 101b (triangle-wave PWM mode 2) is set. |
| 2 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 3 | Set cycle | Set the cycle in the GTPR register. |
| 4 | Set initial value for counter | Set the initial value in the GTCNT counter. |
| 5 | Set GTIOCnm pin function | Set the GTIOCnm pin function with the GTIOA[4:0] and GTIOB[4:0] bits in the GTIOR register.<br>In Figure 21.26 and Figure 21.27, GTIOA[4:0] = 00011b and GTIOB[4:0] = 10011b. |
| 6 | Enable GTIOCnm pin output | Set to enable the GTIOCnm pin output with the OAE and OBE bits in the GTIOR register. |
| 7 | Set buffer operation for compare match | Set buffer operation with the CCRA[1:0] bits in the GTBER register. |
| 8 | Set compare match value | Set the GTIOCnA pin transition in the GTCCRA register. |
| 9 | Set buffer value for compare match | For buffer operation, set the GTIOCnA pin transition in 1 cycle after the current cycle (in triangle-wave PWM mode 1) or half cycle after the current cycle (in triangle-wave PWM mode 2) in the GTCCRC register. For double buffer operation, also set the GTIOCnA pin transition in 2 cycles after the current cycle (in triangle-wave PWM mode 1) or 1 cycle after the current cycle (in triangle-wave PWM mode 2) in the GTCCRD registers. |
| 10 | Set automatic dead time setting function | Set the GTDTCR.TDE bit to 1 to enable the automatic dead time setting function. |
| 11 | Set dead time value | Set the dead time value in the GTDVU register. |
| 12 | Start count operation | Set the GTCR.CST bit to 1 to start count operation. |
| 13 | Set buffer value for each cycle | When the compare match register is used for buffer operation, set the GTIOCnA pin transition in 1 cycle after the current cycle (in triangle-wave PWM mode 1) or half cycle after the current cycle (in triangle-wave PWM mode 2) in GTCCRC. |

Note: n: 0 to 7
m: A, B

### 21.3.5 Count Direction Changing Function

The count direction of the GTCNT counter can be changed by modifying the UD bit in GTUDDTYC.

In saw-wave mode, if the UD bit in GTUDDTYC is modified during count operation, the count direction is changed at an overflow (when modified during up-counting) or an underflow (when modified during down-counting). If the GTUDDTYC.UD bit is modified while the count operation stops and the GTUDDTYC.UDF bit is 0, the GTUDDTYC.UD bit modification is not reflected at the start of counting and the count direction is changed at an overflow or an underflow. If the UDF bit is set to 1 while the count operation stops, the GTUDDTYC.UD bit value at that time is reflected at the start of counting.

In triangle-wave mode, the count direction does not change even though the UD bit in GTUDDTYC is modified during the count operation. Similarly, even though the GTUDDTYC.UD bit is modified while the count operation stops and GTUDDTYC.UDF bit is 0, the GTUDDTYC.UD bit value is not reflected to the count operation. If the GTUDDTYC.UDF bit is set to 1 while the count operation is stopped, the GTUDDTYC.UD bit value at that time is reflected at the start of counting.

If the count direction changes during a saw-wave count operation, the GTPR value after the start of up-counting is reflected in the count cycle during up-counting and the GTPR value after the start of down-counting is reflected in the count cycle during down-counting.

Figure 21.28 shows an example of count direction changing function operation.

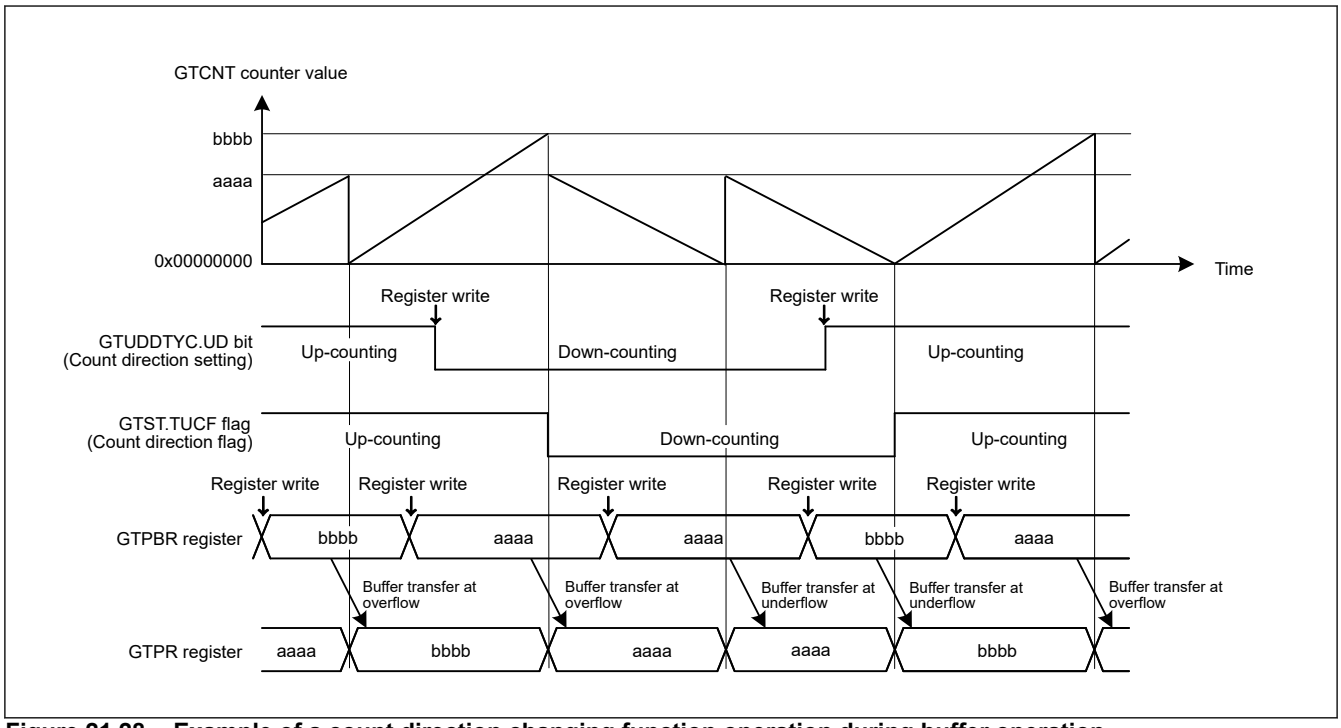



**Figure 21.28 Example of a count direction changing function operation during buffer operation**

### 21.3.6 Function of Output Duty 0% and 100%

The output duty of the GTIOCnA pin and the GTIOCnB pin (n = 0 to 7) are set to 0% or 100% by changing the GTUDDTYC.OADTY bit or GTUDDTYC.OBDTY bit.

In saw-wave mode, if the GTUDDTYC.OADTY bit or the GTUDDTYC.OBDTY bit is modified during the count operation, the output duty setting is reflected at an overflow (when modified during up-counting) or an underflow (when modified during down-counting). If the GTUDDTYC.OADTY bit or the GTUDDTYC.OBDTY bit is modified while the count operation is stopped and the GTUDDTYC.OADTYF or the GTUDDTYC.OBDTYF bit is 0, the output duty modification is not reflected at the start of counting. The output duty changes at an overflow or an underflow. If the GTUDDTYC.OADTY bit or the GTUDDTYC.OBDTY bit is modified while the count operation is stopped and the

GTUDDTYC.OADTYF or the GTUDDTYC.OBDTYF bit is 1, the GTUDDTYC.OADTY bit or the GTUDDTYC.OBDTY bit value at that time is reflected at the start of counting.

In triangle-wave mode, if the GTUDDTYC.OADTY bit or the GTUDDTYC.OBDTY bit is modified during the count operation, the output duty setting is reflected an underflow.

If the GTUDDTYC.OADTY bit or the GTUDDTYC.OBDTY bit is modified while the count operation is stopped and the GTUDDTYC.OADTYF or the GTUDDTYC.OBDTYF bit is 0, the output duty modification is not reflected at the start of counting. The output duty changes at an underflow. If the GTUDDTYC.OADTY bit or the GTUDDTYC.OBDTY bit is modified while the count operation stops and the GTUDDTYC.OADTYF or the GTUDDTYC.OBDTYF bit is 1, the output duty modification is reflected at the start of counting.

In performing 0% or 100% duty operation, GPT internally continues to:

- Perform compare match operation
- Set compare match flag
- Output interrupt
- Perform buffer operation.

When the control is changed from 0% or 100% duty setting to compare match, the output value of GTIOCnA pin at cycle end is decided by GTIOR.GTIOA[3:2] and GTUDDTYC.OADTYR. The output value of GTIOCnB pin at cycle end is decided by GTIOR.GTIOB[3:2] and GTUDDTYC.OBDTYR.

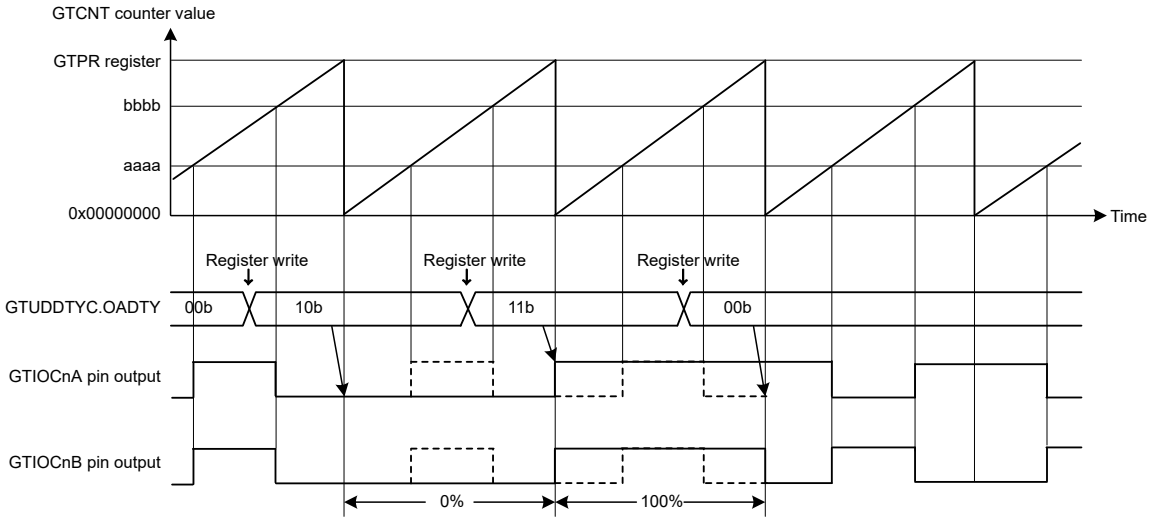
When GTIOR.GTIOA[3:2] and GTIOR.GTIOB[3:2] are set to 01b, the output pins output low at cycle end. When GTIOR.GTIOA[3:2] and GTIOR.GTIOB[3:2] are set to 10b, the output pins output high at cycle end.

GTUDDTYC.OADTYR selects the value that is the object of output retained/toggled at cycle end, when GTIOR.GTIOm[3:2] are set to 00b (output retained at cycle end) or when GTIOR.GTIOm[3:2] are set to 11b (output toggled at cycle end). Table 21.23 shows the values of GTIOCnA and GTIOCnB pin output at cycle end.

**Table 21.23 Output values after releasing 0% or 100% duty setting (m = A, B)**

| GTIOR.GTIOm[3:2] | Compare match value at cycle end masked by 0% or 100% duty setting | GTUDDTYC.OmDTYR in duty 0% setting | | GTUDDTYC.OmDTYR in duty 100% setting | |
|---|---|---|---|---|---|
| | | 0 | 1 | 0 | 1 |
| 00 (output retained at cycle end) | 0 | 0 | 0 | 1 | 0 |
| | 1 | 0 | 1 | 1 | 1 |
| 01 (low output at cycle end) | — | 0 | 0 | 0 | 0 |
| 10 (high output at cycle end) | — | 1 | 1 | 1 | 1 |
| 11 (output toggled at cycle end) | 0 | 1 | 1 | 0 | 1 |
| | 1 | 1 | 0 | 0 | 0 |

Figure 21.29 shows an example of output duty 0% and 100% function.




Note: The base address of registers are the same.

Note: [Setting examples]

| | |
|---|---|
| GTIOR.GTIOA[4:0] bits: | 00011b<br>initial low output, output toggled at compare match, output retained at cycle end |
| GTUDDTYC.OADTYR bit: | 0b<br>The function selected by GTIOA[3:2] is applied to the output value when the duty cycle is set after release from 0 or 100% duty-cycle setting. |
| GTIOR.GTIOB[4:0] bits: | 00011b<br>initial low output, output toggled at compare match, output retained at cycle end |
| GTUDDTYC.OBDTYR bit: | 1<br>The function of GTIOB[3:2] is applied to the compare match output value which is masked after 0 or 100% duty setting release. |

**Figure 21.29 Example of output duty 0% and 100% function**

## 21.3.7 Hardware Count Start/Count Stop and Clear Operation

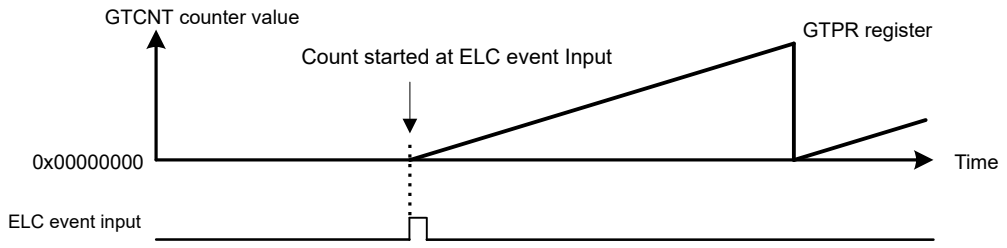
The GTCNT counter can be started, stopped, or cleared by the following hardware sources:

- External trigger input
- ELC event input
- GTIOCnA and GTIOCnB pin input (n = 0 to 7).

### 21.3.7.1 Hardware Start Operation

The GTCNT counter can be started by selecting a hardware source using GTSSR.

Figure 21.30 shows an example of a count start operation by a hardware source. Table 21.24 shows the setting example.




Note: ELC event input: ELC_GPTA event input

**Figure 21.30 Example of count start operation by a hardware source started at the input of the signal from the ELC_GPTA event**

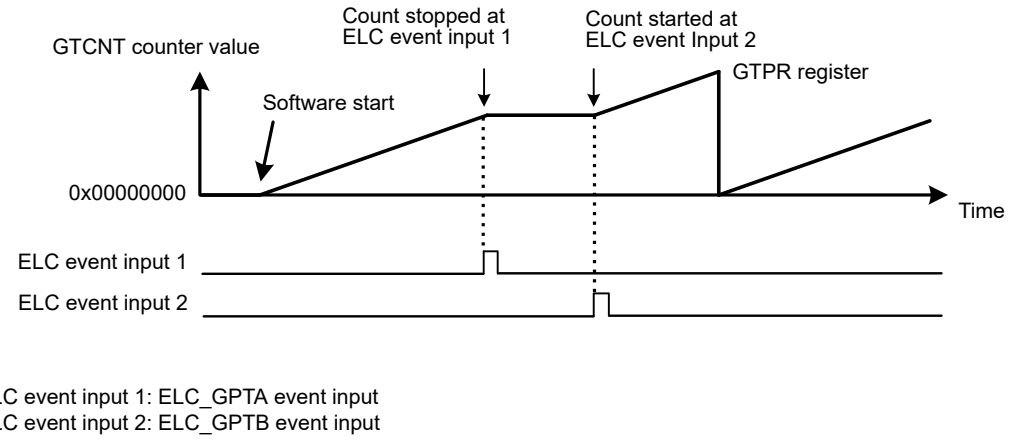
**Table 21.24 Example setting for count start operation by a hardware source**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.30, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.30, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter.<br>In Figure 21.30, 0x00000000 is set. |
| 6 | Set hardware count start | Select a hardware source for starting count operation in the GTSSR register.<br>In Figure 21.30, GTSSR.SSELCA = 1 |
| 7 | Set hardware source operation | Set operation of the hardware source selected by the GTSSR register and start counting.<br>In Figure 21.30, the ELC_GPTA event input operation is set. |

### 21.3.7.2 Hardware Stop Operation

The GTCNT counter can be stopped by selecting a hardware source using GTPSR.

Figure 21.31 shows an example of a count stop operation by a hardware source. Table 21.25 shows the setting example. In this example, the count operation stops at the ELC_GPTA event input and restarts at the ELC_GPTB event input.




Note: ELC event input 1: ELC_GPTA event input
ELC event input 2: ELC_GPTB event input

**Figure 21.31 Example of count stop operation by hardware source started by software, stopped at ELC_GPTA input, and restarted at ELC_GPTB input**

**Table 21.25 Example setting for count stop operation by a hardware source**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with GTCR.MD[2:0] bits.<br>In Figure 21.31, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.31, after 11b is set in GTUDDTYC[1:0], 01b is set in GTUDDTYC[1:0] (up-counting). |
| 3 | Select count clock | Select the count clock with GTCR.TPCS[3:0]. |
| 4 | Set cycle | Set the cycle in GTPR. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter.<br>In Figure 21.31, 0x00000000 is set. |
| 6 | Set hardware count start | Select a hardware source for starting count operation in GTSSR register, and wait for count start by the hardware source. In Figure 21.31, GTSSR.SSELCB = 1. |
| 7 | Set hardware count stop | Select a hardware source for stopping count operation in GTPSR register and wait for count stop by the hardware source. In Figure 21.31, GTPSR.PSELCA = 1. |
| 8 | Set hardware source operation | Set operation of the hardware source selected in GTSSR register or GTPSR register, and start or stop counting. In Figure 21.31, ELC_GPTA input operation and ELC_GPTB input operation are set. |

Figure 21.32 shows an example of a count start/stop operation by a hardware source. Table 21.26 shows the setting example. In this example, the counter operates during the high-level periods of the external trigger input GTETRGA.

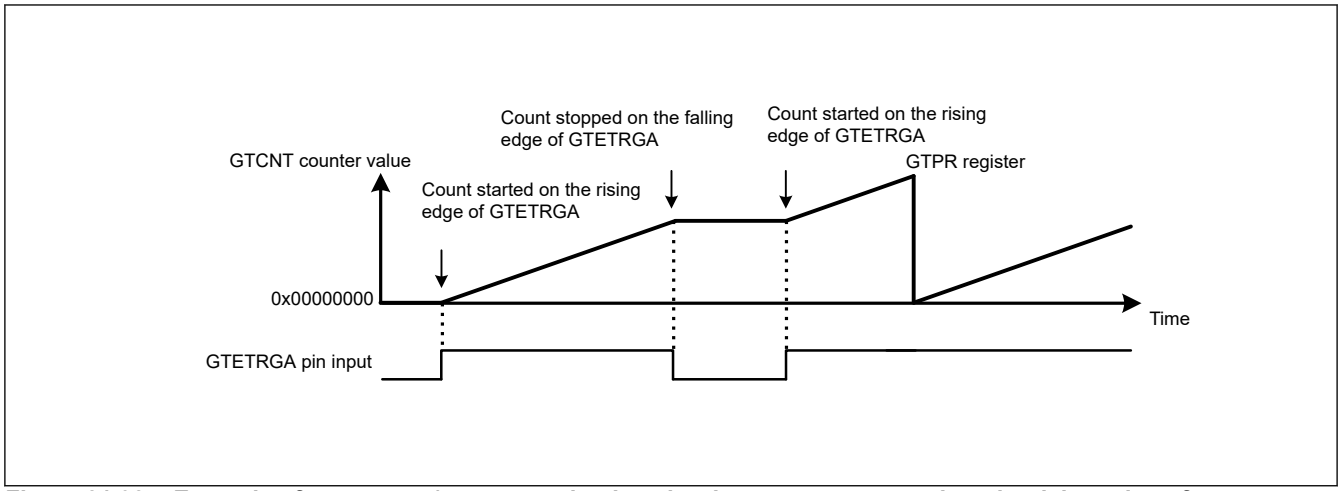



**Figure 21.32 Example of count start/stop operation by a hardware source started on the rising edge of GTETRGA pin input, and stopped on the falling edge of GTETRGA pin input**

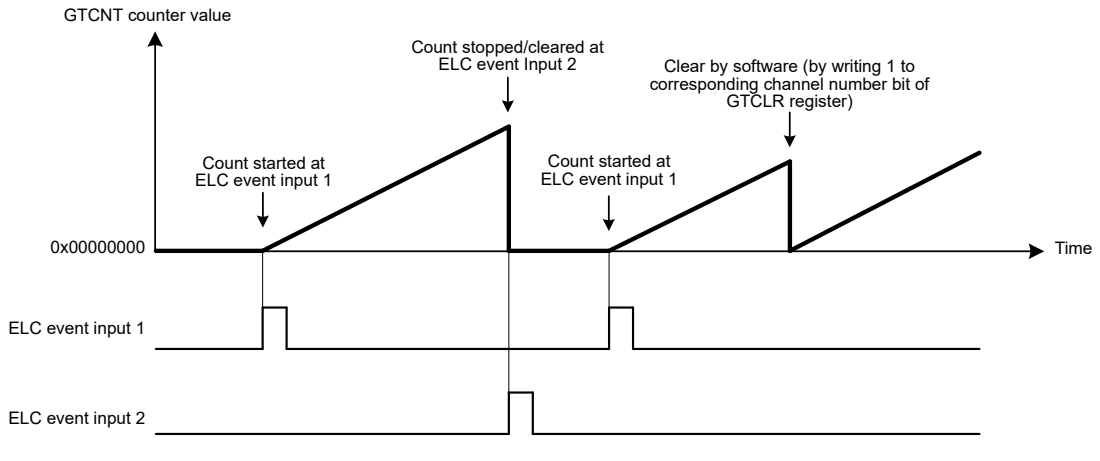
**Table 21.26 Example setting for count start/stop operation by a hardware source**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.32, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.32, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter.<br>In Figure 21.32, 0x00000000 is set. |
| 6 | Set hardware count start | Select a hardware source for starting count operation with the GTSSR register, and wait for count start by the hardware source.<br>In Figure 21.32, GTSSR.SSGTRGAR = 1. |
| 7 | Set hardware count stop | Select a hardware source for stopping count operation with the GTPSR register, and wait for count stop by the hardware source.<br>In Figure 21.32, GTPSR.PSGTRGAF = 1. |
| 8 | Set hardware source operation | Set operation of the hardware source selected in the GTSSR register or GTPSR register and start or stop counting.<br>In Figure 21.32, the GTETRGA pin operation is set. |

### 21.3.7.3 Hardware Clear Operation

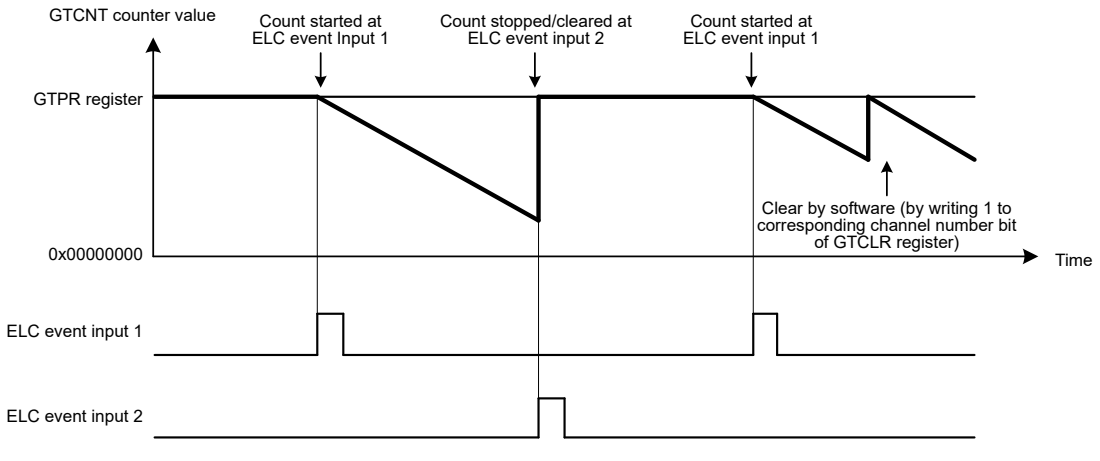
The GTCNT counter can be cleared by selecting a hardware source using GTCSR. The GPTn_OVF/GPTn_UDF (n = 0 to 7) interrupt (overflow/underflow interrupt) is not generated when the GTCNT counter is cleared by a hardware source or by software.

Figure 21.33 and Figure 21.34 show examples of the GTCNT counter clearing operation by a hardware source. Table 21.27 shows the setting example. In this example, the GTCNT counter starts at the ELC_GPTA input, and the counter stops and clears at the ELC_GPTB input.




Note: ELC event input 1: ELC_GPTA event input
ELC event input 2: ELC_GPTB event input

**Figure 21.33 Examples of count clearing operation by hardware source in saw wave up-counting, started at ELC_GPTA input, and stopped/cleared at ELC_GPTB input**




Note: ELC event input 1: ELC_GPTA event input
ELC event input 2: ELC_GPTB event input

**Figure 21.34 Examples of count clearing operation by hardware source in saw wave down-counting, started at ELC_GPTA input, and stopped/cleared at ELC_GPTB input**

**Table 21.27 Example setting for count clearing operation by a hardware source (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.33 and Figure 21.34, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.33, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting).<br>In Figure 21.34, after 10b is set in the GTUDDTYC[1:0] bits, 00b is set in the GTUDDTYC[1:0] bits (down-counting). |

**Table 21.27 Example setting for count clearing operation by a hardware source (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter.<br>In Figure 21.33, 0x00000000 is set. In Figure 21.34, the GTPR register value is set. |
| 6 | Set hardware count start | Select a hardware source for starting count operation in the GTSSR register, and wait for count start by the hardware source. In Figure 21.33 and Figure 21.34, GTSSR.SSELCA = 1. |
| 7 | Set hardware count stop | Select a hardware source for stopping count operation in the GTPSR register, and wait for count stop by the hardware source. In Figure 21.33 and Figure 21.34, GTPSR.PSELCB = 1. |
| 8 | Set hardware count clear | Select a hardware source for clearing count operation in the GTCSR register, and wait for count clear by the hardware source. In Figure 21.33 and Figure 21.34, GTCSR.CSELCB = 1. |
| 9 | Set hardware source operation | Set operation of the hardware source selected in the GTSSR register, GTPSR register or GTCSR register and start, stop or clear counting.<br>In Figure 21.33 and Figure 21.34, the ELC_GPTA input and ELC_GPTB input are set. |

The GPTn_OVF/GPTn_UDF (n = 0 to 7) interrupt (overflow/underflow interrupt) is not generated when the counter is cleared by a hardware source or by software.

Figure 21.35 shows the relationship between the counter clearing by a hardware source and the GPTn_OVF (n = 0 to 7) interrupt.

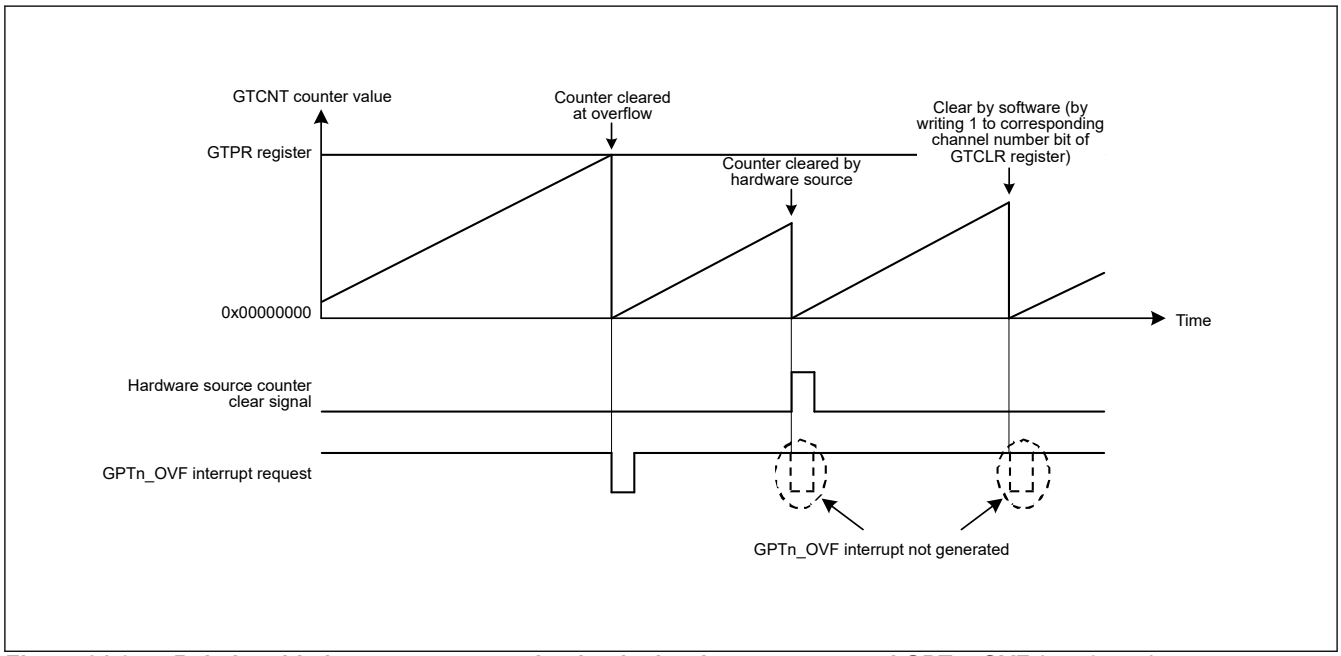



**Figure 21.35 Relationship between counter clearing by hardware source and GPTn_OVF (n = 0 to 7) interrupt**

## 21.3.8 Synchronized Operation

Synchronized operation on channels such as a synchronized start, stop, and clear operation can be performed.

### 21.3.8.1 Synchronized Operation by Software

The GTCNT counters can be started, stopped, and cleared on multiple channels by setting the associated GTSTR, GTSTP, or GTCLR bits simultaneously to 1.

Count start with a phase difference is possible by setting the initial value in the GTCNT counter and setting the associated GTSTR bits simultaneously to 1.

Figure 21.36 shows an example of a simultaneous start, stop, and clear by software. Figure 21.37 shows an example of phase start operation by software.

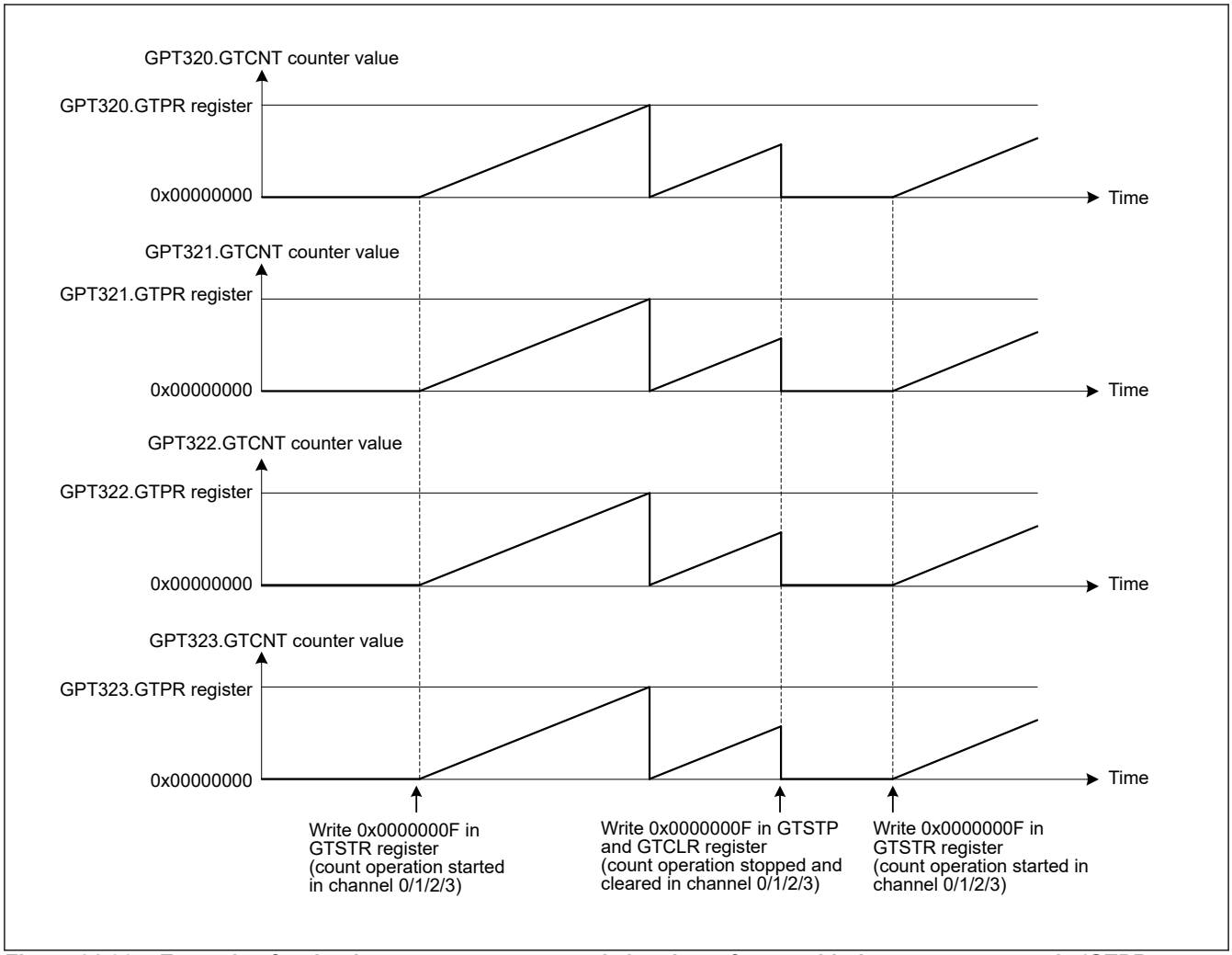



**Figure 21.36 Example of a simultaneous start, stop, and clear by software with the same count cycle (GTPR register value)**

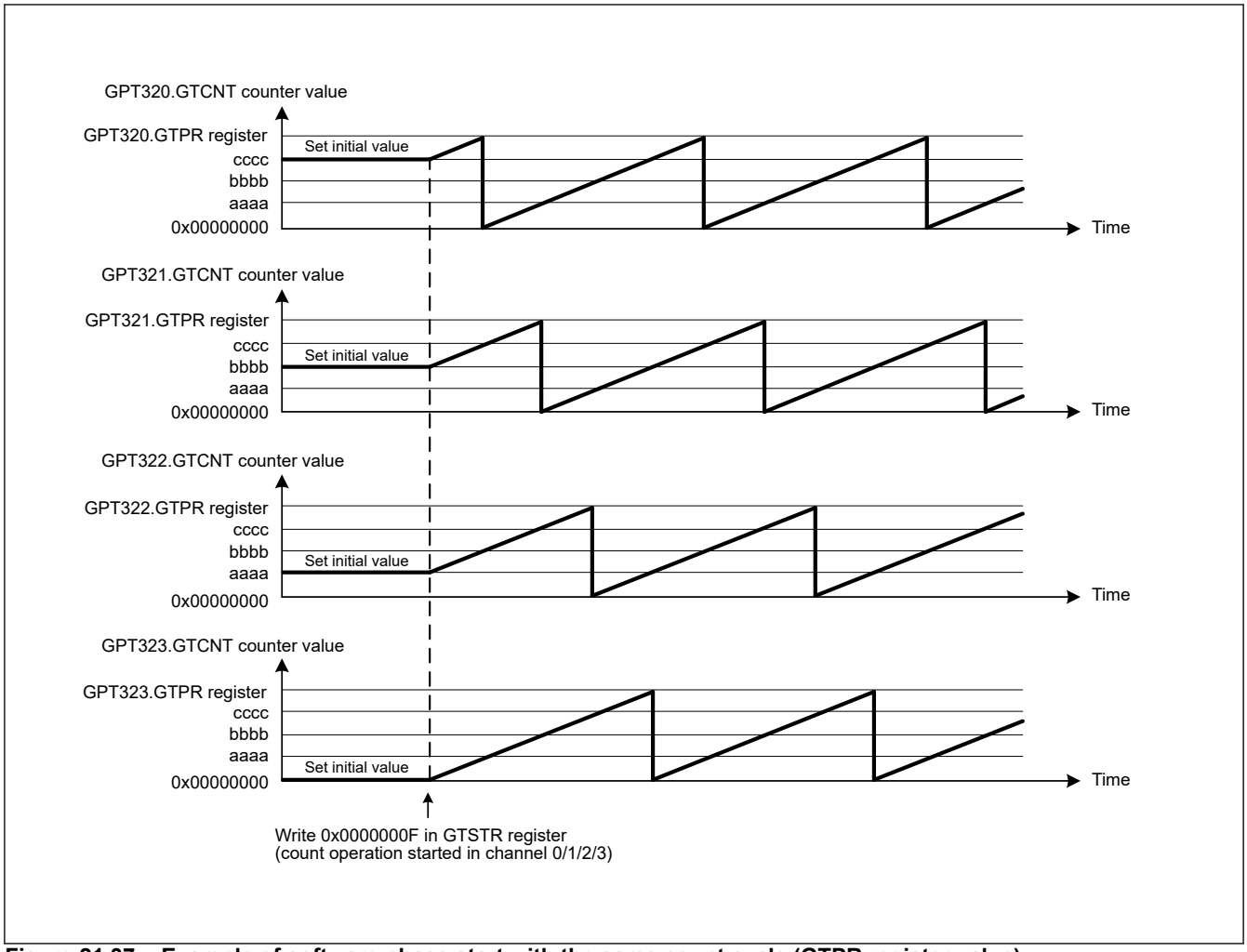
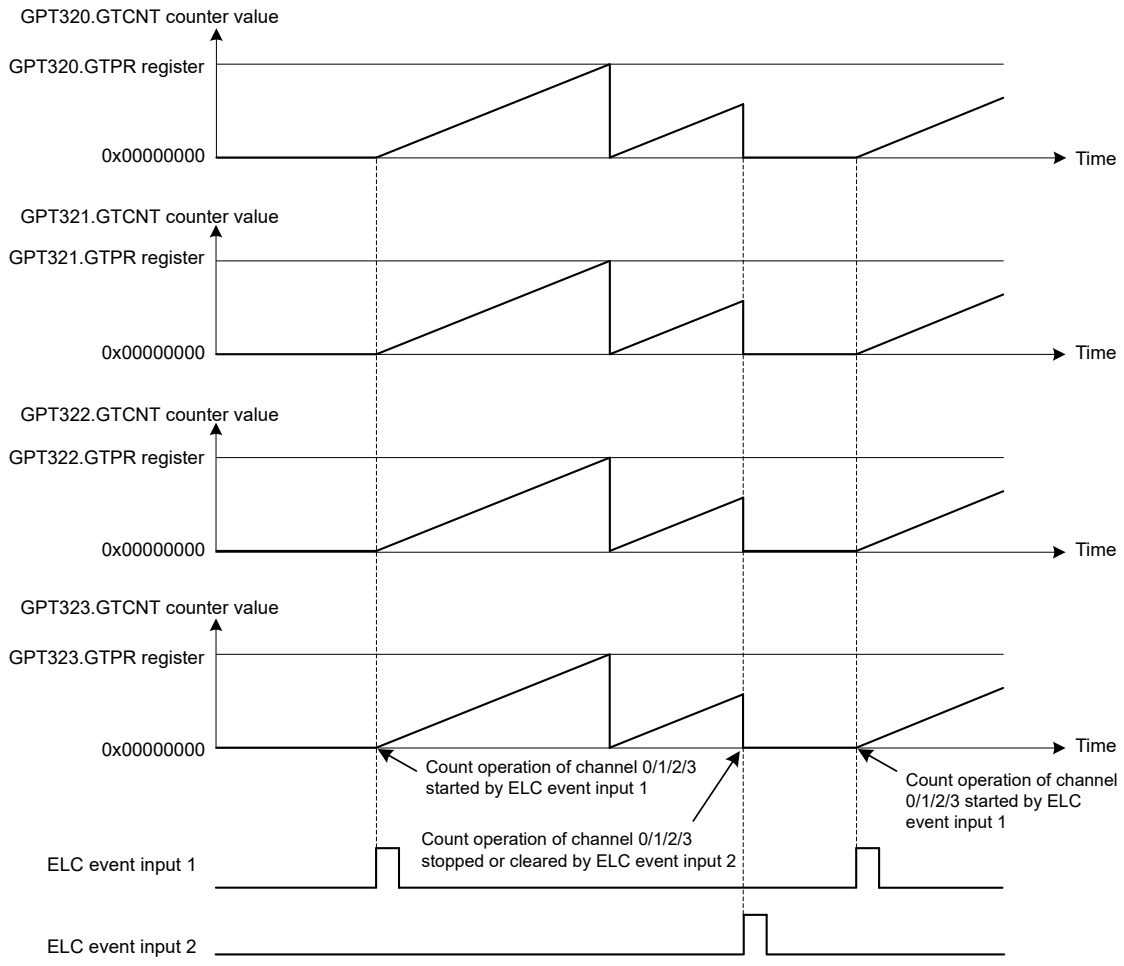



**Figure 21.37 Example of software phase start with the same count cycle (GTPR register value)**

### 21.3.8.2 Synchronized Operation by Hardware

The counters for multiple channels can be started, stopped, and cleared simultaneously by the following hardware sources. Hardware sources that can cause a synchronized operation are external trigger input and ELC event input. Synchronized operation through the GTIOCnA and GTIOCnB pin inputs is possible by setting an ELC event due to input capture as a hardware source (n = 0 to 7).

Figure 21.38 shows an example of a simultaneous start, stop, and clear operation by a hardware source. Table 21.28 shows the setting example.




Note: ELC event input 1: ELC_GPTA event input
ELC event input 2: ELC_GPTB event input

**Figure 21.38 Example of a simultaneous start, stop, and clear by a hardware source with the same count cycle (GTPR register value)**

**Table 21.28 Example setting for simultaneous start by a hardware source (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Set operating mode | Set the operating mode with the GTCR.MD[2:0] bits.<br>In Figure 21.38, 000b (saw-wave PWM mode) is set. |
| 2 | Set count direction | Select the count direction (up or down) with the GTUDDTYC register.<br>In Figure 21.38, after 11b is set in the GTUDDTYC[1:0] bits, 01b is set in the GTUDDTYC[1:0] bits (up-counting). |
| 3 | Select count clock | Select the count clock with the GTCR.TPCS[3:0] bits. |
| 4 | Set cycle | Set the cycle in the GTPR register. |
| 5 | Set initial value for counter | Set the initial value in the GTCNT counter.<br>In Figure 21.38, 0x00000000 is set. |
| 6 | Set hardware count start | Select a hardware source for starting count operation with the GTSSR register, and wait for count start by the hardware source.<br>In Figure 21.38, GTSSR.SSELCA = 1. |

**Table 21.28 Example setting for simultaneous start by a hardware source (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 7 | Set hardware count stop | Select a hardware source for stopping count operation with the GTPSR register, and wait for count stop by the hardware source.<br>In Figure 21.38, GTPSR.PSELCB = 1. |
| 8 | Set hardware count clear | Select a hardware source for clearing count operation with the GTCSR register, and wait for count clear by the hardware source.<br>In Figure 21.38, GTCSR.CSELCB = 1. |
| 9 | Set hardware source operation | Set operation of the hardware source selected in the GTSSR, GTPSR, or GTCSR registers, and start, stop, or clear counting.<br>In Figure 21.38, ELC_GPTA input and ELC_GPTB input are set. |

## 21.3.9 PWM Output Operation Examples

### (1) Synchronized PWM output

The GPT outputs 8 × 2 phases of linked PWM waveforms for a maximum of GPT × 8 channels.

Figure 21.39 shows an example in which four channels perform synchronized operation in saw-wave PWM mode and eight phases of PWM waveforms are output. The GTIOCnA is set so that it outputs low as the initial value, high at a GTCCRA compare match, and low at the cycle end. The GTIOCnB is set so that it outputs low as the initial value, high at a GTCCRB compare match, and low at the cycle end.

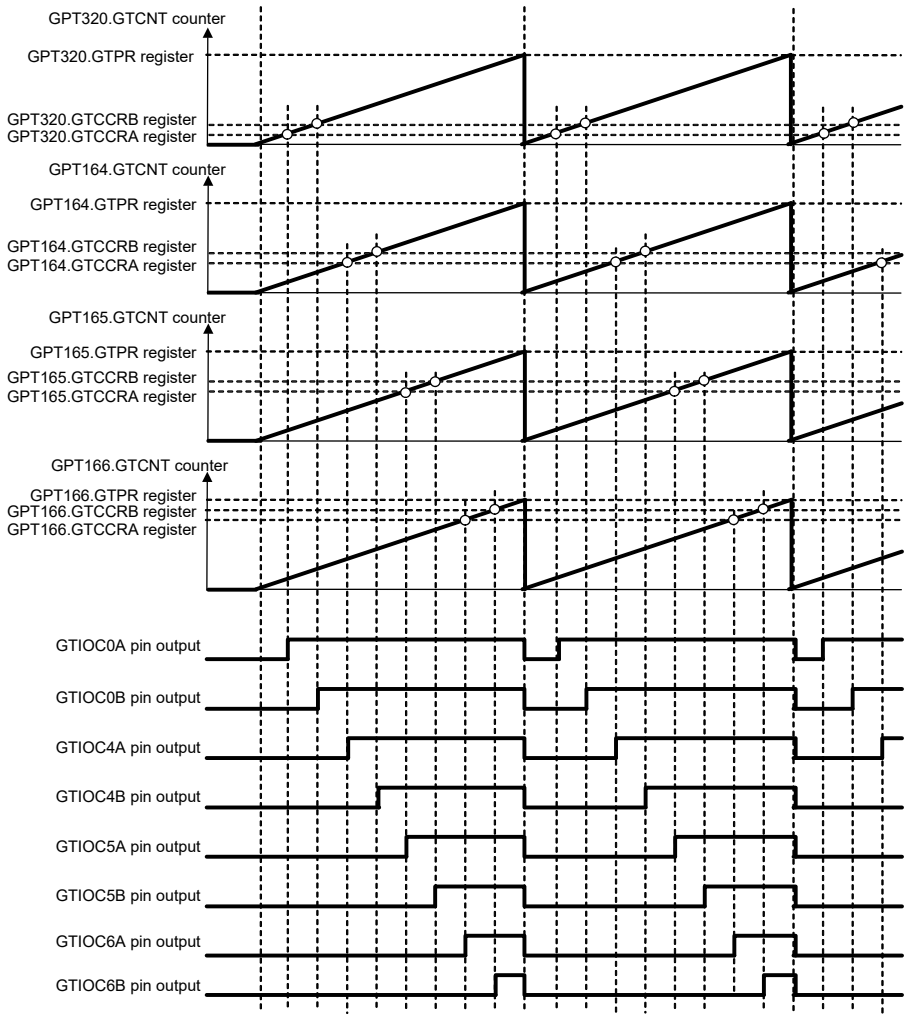



**Figure 21.39 Example of synchronized PWM output**

## (2) 3-phase saw-wave complementary PWM output

Figure 21.40 shows an example in which three channels perform synchronized operation in saw-wave PWM mode and 3-phase complementary PWM waveforms are output. The GTIOCnA pin is set so that it outputs low as the initial value, high at a GTCCRA compare match, and low at the cycle end. The GTIOCnB pin is set so that it outputs high as the initial value, low at a GTCCRB compare match, and high at the cycle end.

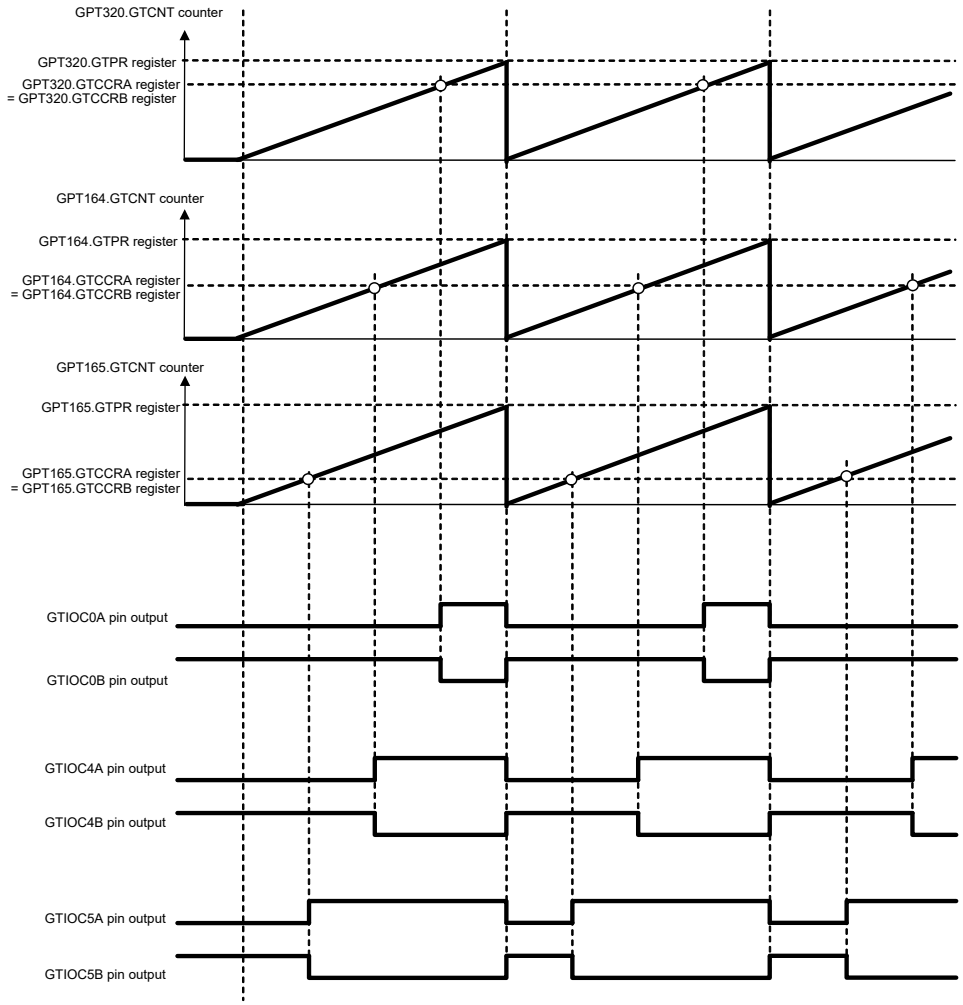



**Figure 21.40 Example of 3-phase saw-wave complementary PWM output**

### (3) 3-phase saw-wave complementary PWM output with automatic dead time setting

Figure 21.41 shows an example in which three channels perform synchronized operation in saw-wave one-shot pulse mode with automatic dead time setting and 3-phase complementary PWM waveforms are output. The GTIOCnA pin is set so that it outputs low as the initial value, toggles the output at a GTCCRA compare match, and retains the output at the cycle end. The GTIOCnB pin is set so that it outputs high as the initial value, toggles the output at a GTIOCnB compare match, and retains the output at the cycle end.

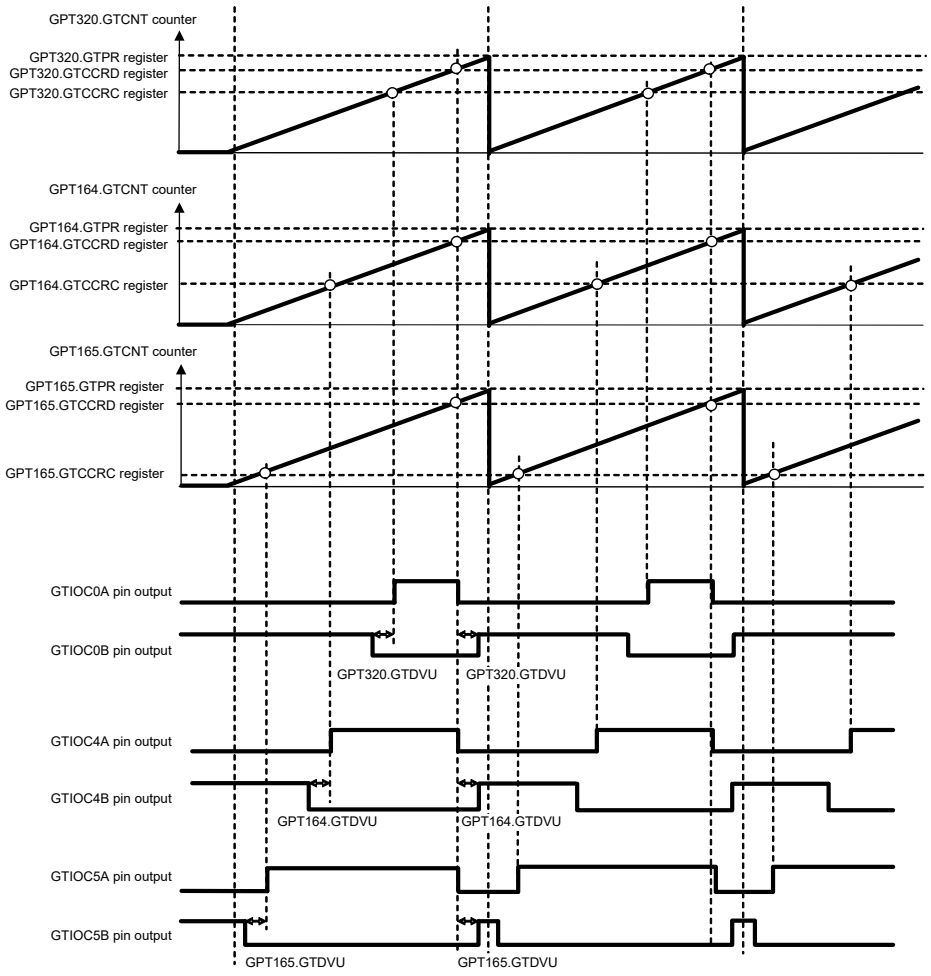



**Figure 21.41 Example of 3-phase saw-wave complementary PWM output with automatic dead time setting**

### (4) 3-phase triangle-wave complementary PWM output

Figure 21.42 shows an example in which three channels perform synchronized operation in triangle-wave PWM mode 1 and 3-phase complementary PWM waveforms are output. The GTIOCnA pin is set so that it outputs low as the initial value, toggles the output at a GTCCRA compare match, and retains the output at the cycle end. The GTIOCnB pin is set so that it outputs high as the initial value, toggles the output at a GTCCRB compare match, and retains the output at the cycle end.

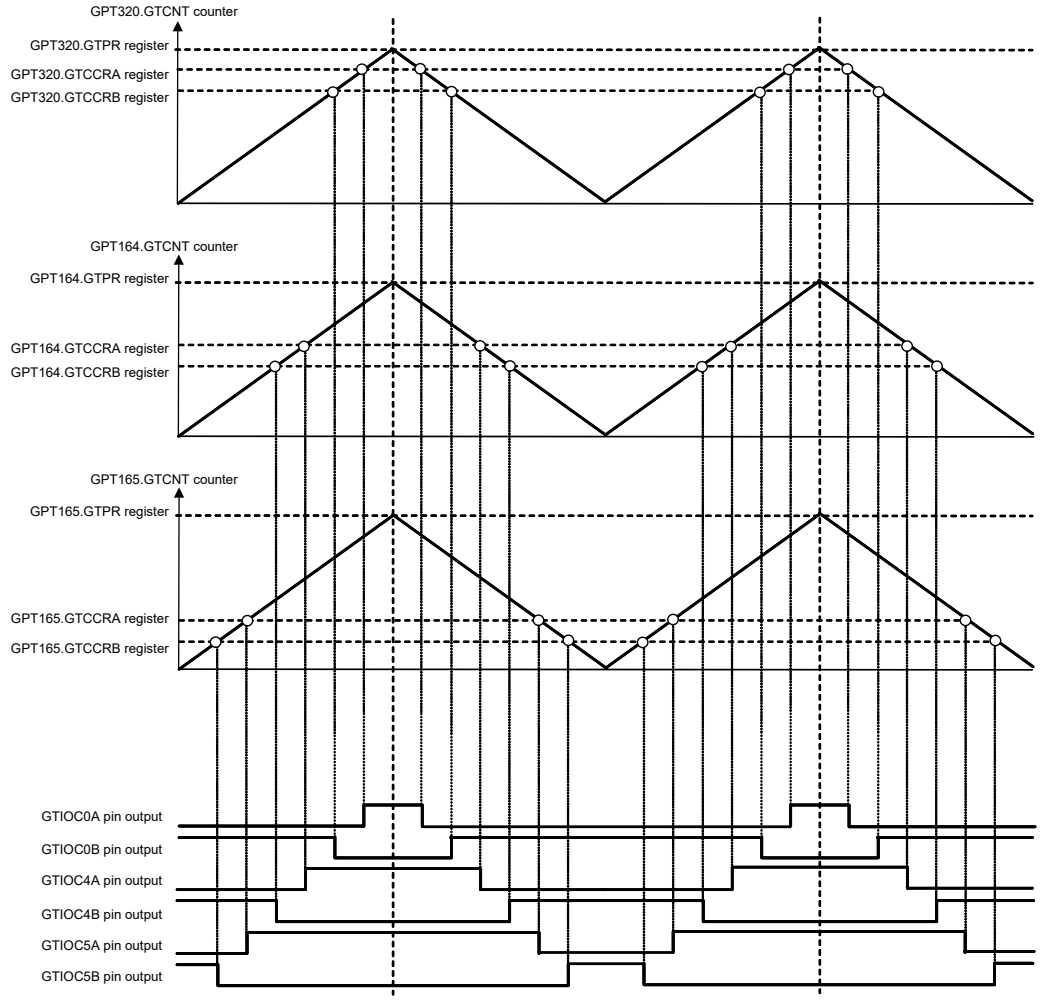



**Figure 21.42 Example of 3-phase triangle-wave complementary PWM output**

### (5) 3-phase triangle-wave complementary PWM output with automatic dead time setting

Figure 21.43 shows an example in which three channels perform synchronized operation in triangle-wave PWM mode 1 with automatic dead time setting and 3-phase complementary PWM waveforms are output. The GTIOCnA pin is set so that it outputs low as the initial value, toggles the output at a GTCCRA compare match, and retains the output at the cycle end. The GTIOCnB pin is set so that it outputs high as the initial value, toggles the output at a GTCCRB compare match, and retains the output at the cycle end.

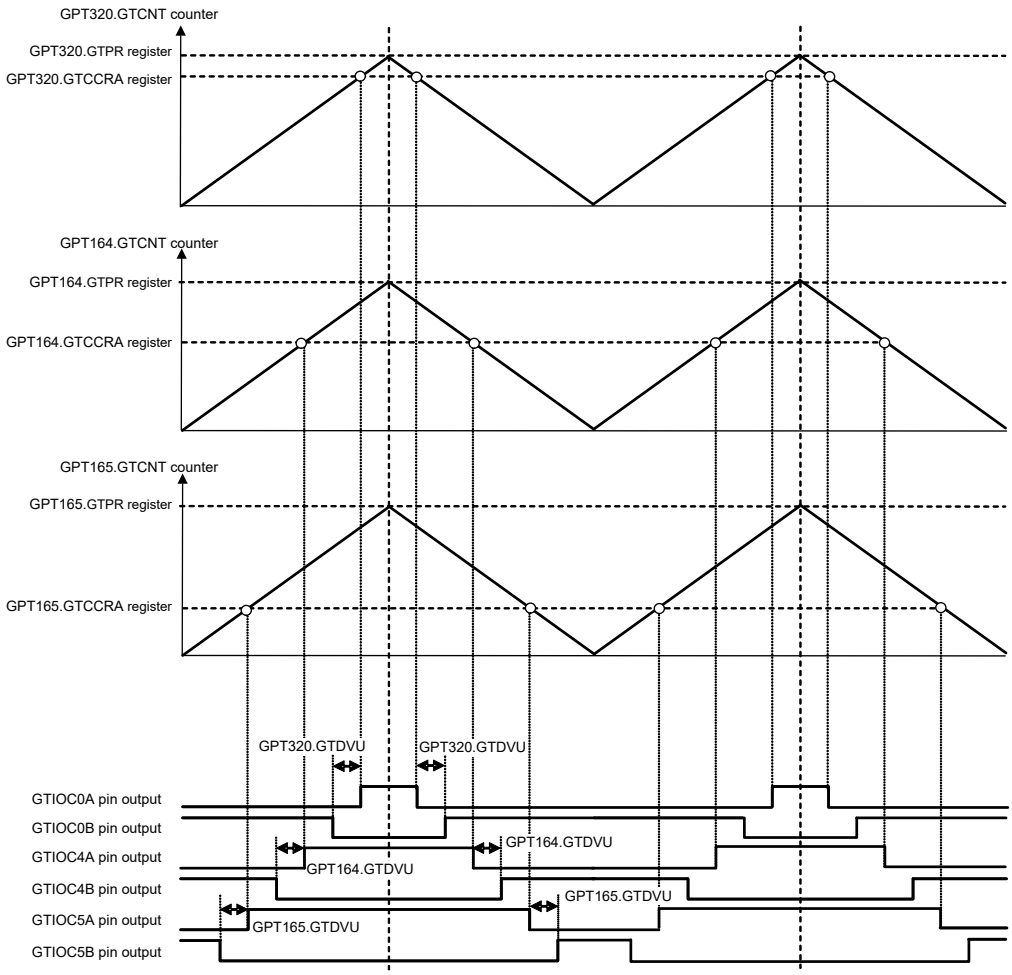



**Figure 21.43 Example of 3-phase triangle-wave complementary PWM output with automatic dead time setting**

## (6) 3-phase asymmetric triangle-wave complementary PWM output with automatic dead time setting

Figure 21.44 shows an example in which three channels perform synchronized operation in triangle-wave PWM mode 3 with automatic dead time setting and 3-phase complementary PWM waveforms are output. The GTIOCnA is set so that it outputs low as the initial value, toggles the output at a GTCCRA compare match, and retains the output at the cycle end. The GTIOCnB is set so that it outputs high as the initial value, toggles the output at a GTCCRB compare match, and retains the output at the cycle end.

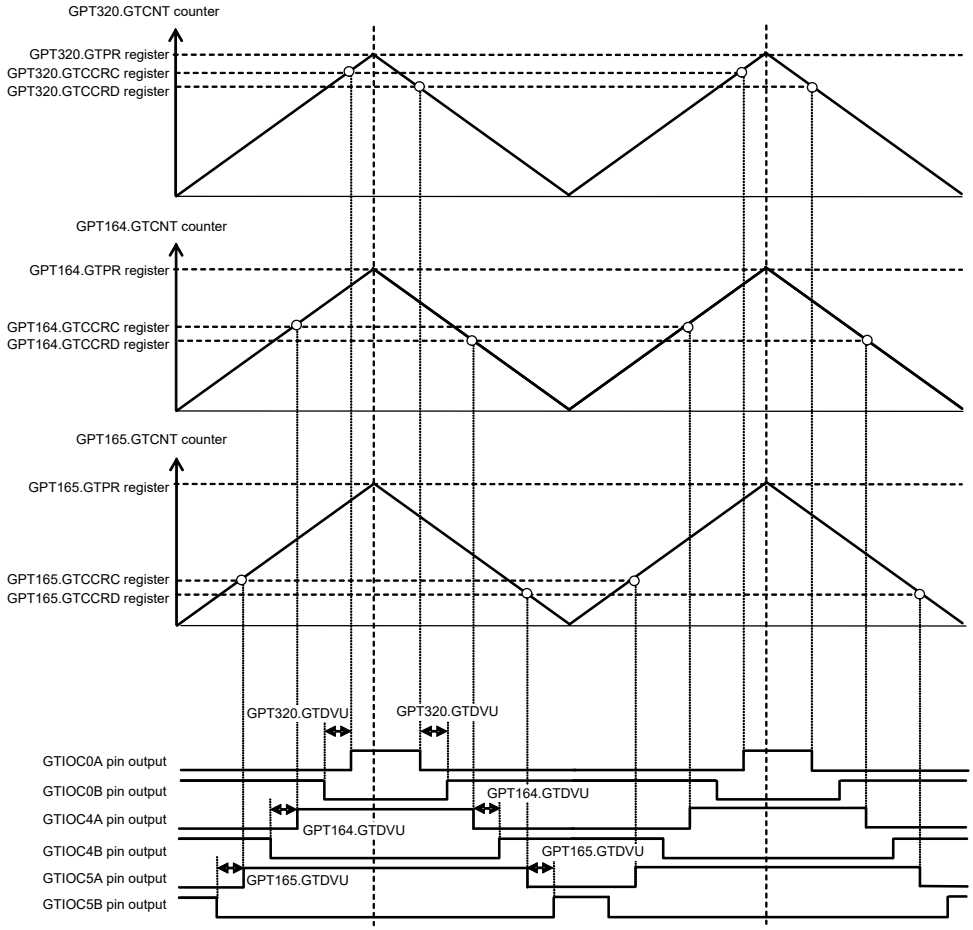



**Figure 21.44 Example of 3-phase asymmetric triangle-wave complementary PWM output with automatic dead time setting**

## 21.3.10 Period Count Function

By setting the GTPC register, the end of period can be counted.

The number of period to be counted should be set into the GTPC.PCNT counter when the GTPC.PCEN bit is 0. When the PCEN bit is 1, the PCNT counter can be read, but writing is disabled. When the PCEN bit is 1, down-counting is performed at the end of period. When the PCNT counter is 1 at the end of period, it becomes 0 and counting is stopped to finish the period count function. At that time, the GTST.PCF flag is set, and the period count function finish interrupt request GPTn_PC is generated. When the GTPC.ASTP bit is 1, the GTCNT counter is also stopped at the same time that the period count function is finished.

When the GTCNT counter is stopped while period count function is enabled, the PCNT counter keeps its value. When the GTCNT counter restarts counting and the PCEN bit is 1, the PCNT counter restarts down-counting from the hold value.

When the PCEN bit is changed from 0 to 1 while the PCNT counter is 0 and the ASTP bit is 1, the GTCNT counter is stopped at the count clock immediately after that.

When either GTSECR.SPCE bit or GTSECR.SPCD bit is set to 1, the PCEN bit in the channels set to 1 by the GTSECSR register is simultaneously set the value to enable or disable the period count function for multiple channels.

Figure 21.45 and Figure 21.46 show examples of PWM cycle count function.

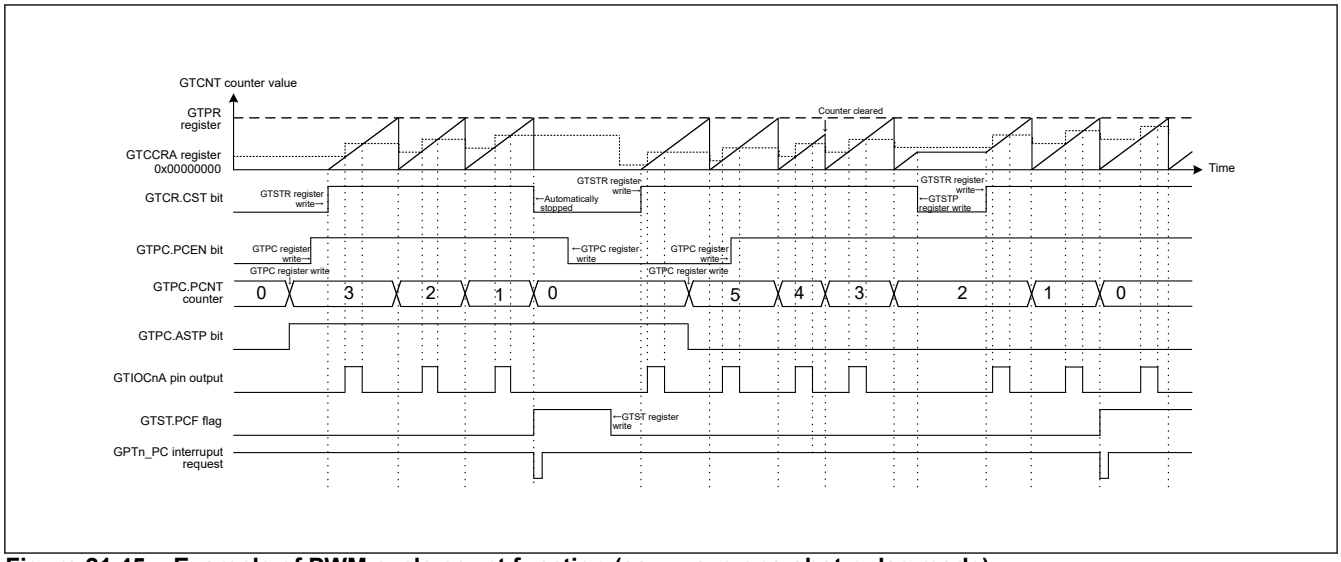
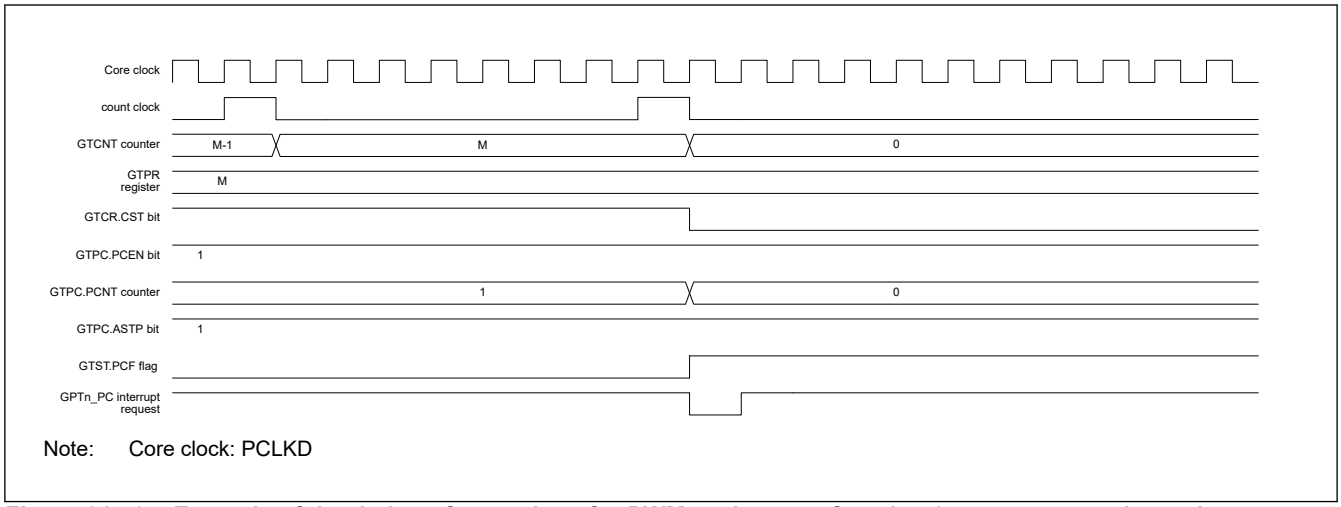



**Figure 21.45 Example of PWM cycle count function (saw-wave one-shot pulse mode)**




Note: Core clock: PCLKD

**Figure 21.46 Example of the timing of operations for PWM cycle count function (saw-wave one-shot pulse mode, up-counting)**

## 21.3.11 Phase Counting Function

The phase difference between the GTIOCnA and GTIOCnB pin (n = 0 to 7) inputs is detected and the associated GTCNT counts up or counts down. The detectable phase difference is available in any combination with the relationship between the edge and the level of GTIOCnA and GTIOCnB pin inputs being set in the GTUPSR and GTDNSR registers. For details on count operation, see section 21.3.1.1. Counter operation.

Figure 21.47 to Figure 21.56 show an example of phase counting modes 1 to 5 operation when the GTIOCnA, GTIOCnB pins are used. Table 21.29 to Table 21.38 show conditions of up-counting or down-counting and list settings for the GTUPSR and GTDNSR registers which is corresponding to Figure 21.47 to Figure 21.56.

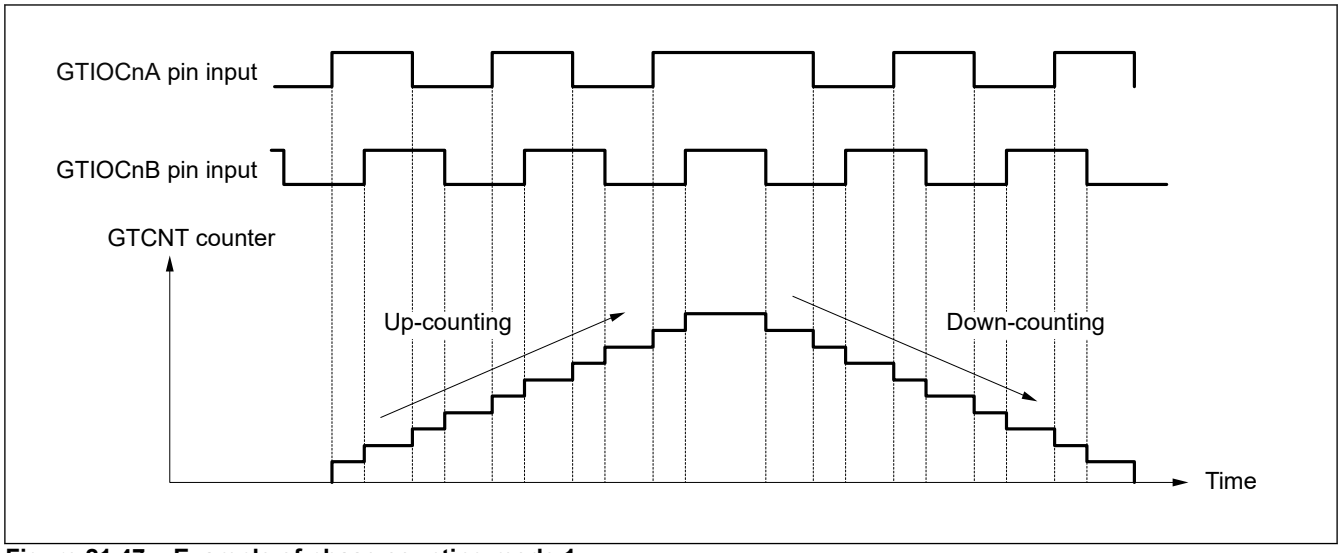



**Figure 21.47 Example of phase counting mode 1**

**Table 21.29 Conditions of up-counting/down-counting in phase counting mode 1**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | ↑ | Up-counting | GTUPSR = 0x00006900<br>GTDNSR = 0x00009600 |
| Low | ↓ | | |
| ↑ | Low | | |
| ↓ | High | | |
| High | ↓ | Down-counting | |
| Low | ↑ | | |
| ↑ | High | | |
| ↓ | Low | | |

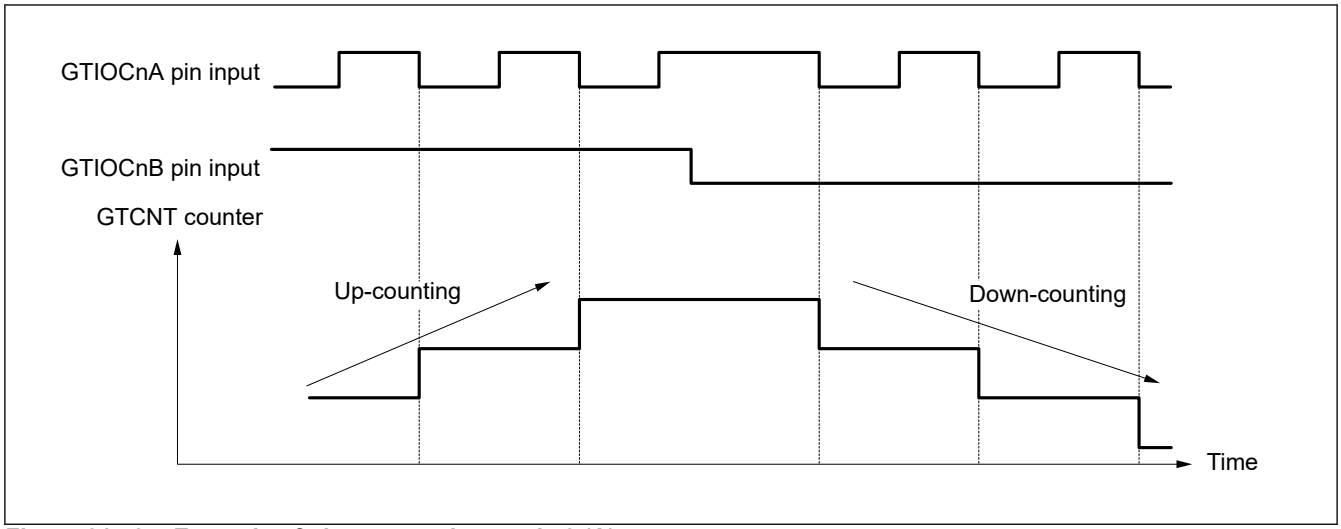



**Figure 21.48 Example of phase counting mode 2 (A)**

**Table 21.30 Conditions of up-counting/down-counting in phase counting mode 2 (A)**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | | Not counting | GTUPSR = 0x00000800<br>GTDNSR = 0x00000400 |
| Low | | | |
| | Low | | |
| | High | Up-counting | |
| High | | Not counting | |
| Low | | | |
| | High | | |
| | Low | Down-counting | |

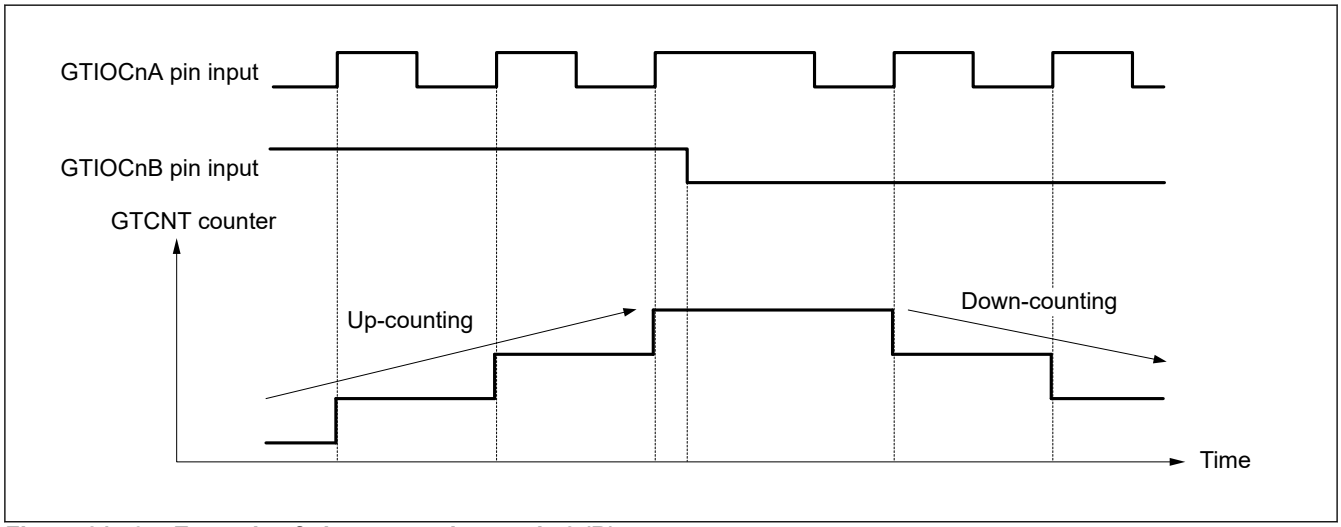



**Figure 21.49 Example of phase counting mode 2 (B)**

**Table 21.31 Conditions of up-counting/down-counting in phase counting mode 2 (B)**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | | Not counting | GTUPSR = 0x00000200<br>GTDNSR = 0x00000100 |
| Low | | | |
| | Low | Down-counting | |
| | High | Not counting | |
| High | | | |
| Low | | | |
| | High | Up-counting | |
| | Low | Not counting | |

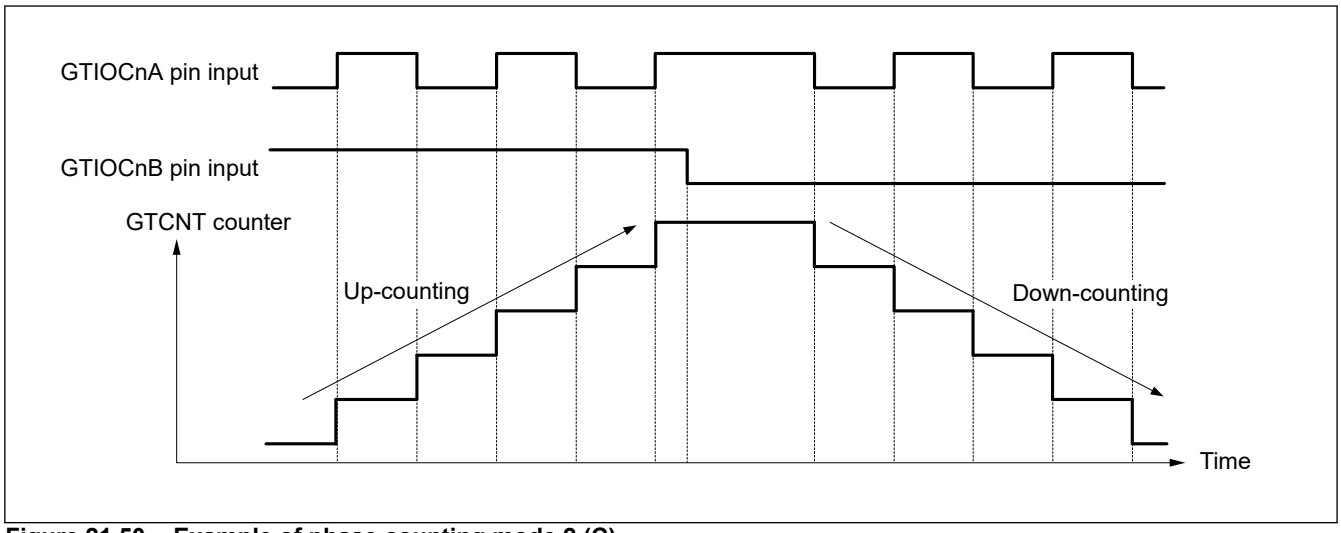



**Figure 21.50 Example of phase counting mode 2 (C)**

**Table 21.32 Conditions of up-counting/down-counting in phase counting mode 2 (C)**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | | Not counting | GTUPSR = 0x00000A00<br>GTDNSR = 0x00000500 |
| Low | | | |
| | Low | Down-counting | |
| | High | Up-counting | |
| High | | Not counting | |
| Low | | | |
| | High | Up-counting | |
| | Low | Down-counting | |

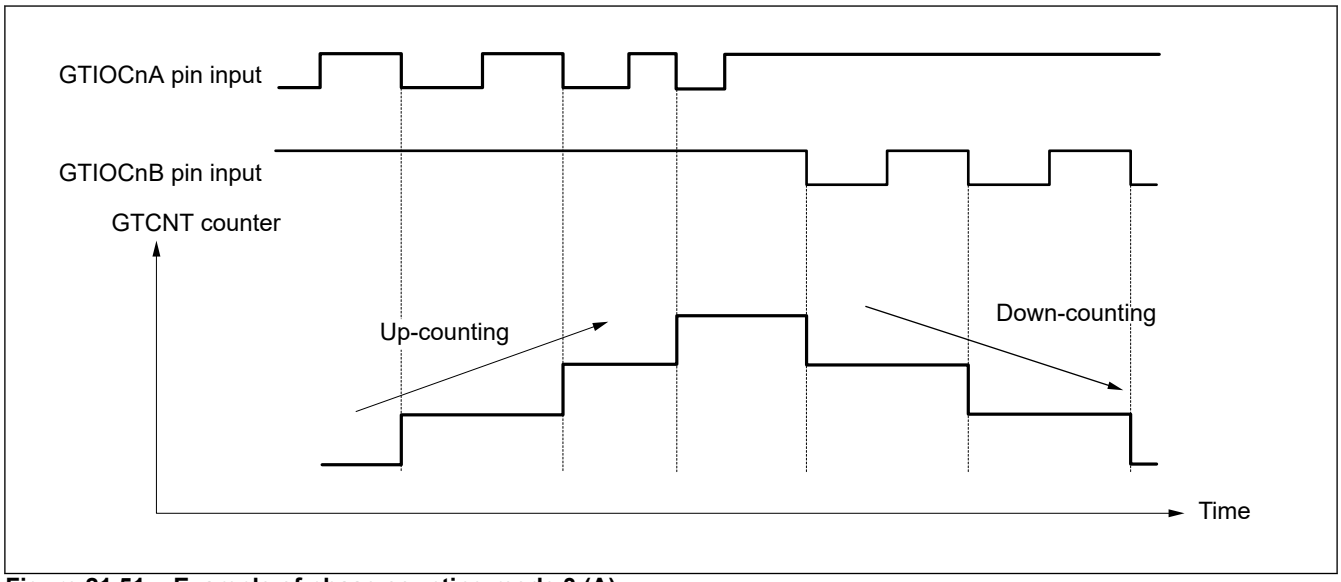



**Figure 21.51 Example of phase counting mode 3 (A)**

**Table 21.33 Conditions of up-counting/down-counting in phase counting mode 3 (A)**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | | Not counting | GTUPSR = 0x00000800<br>GTDNSR = 0x00008000 |
| Low | | | |
| | Low | | |
| | High | Up-counting | |
| High | | Down-counting | |
| Low | | Not counting | |
| | High | | |
| | Low | | |

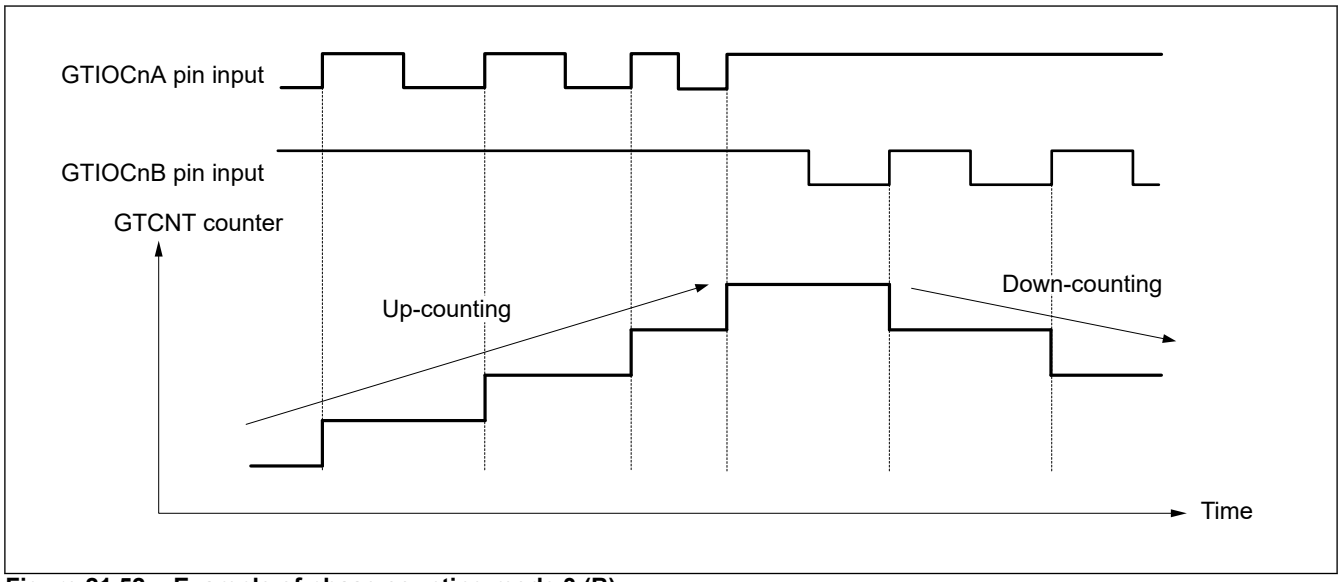



**Figure 21.52 Example of phase counting mode 3 (B)**

**Table 21.34 Conditions of up-counting/down-counting in phase counting mode 3 (B)**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | ↑ | Down-counting | GTUPSR = 0x00000200<br>GTDNSR = 0x00002000 |
| Low | ↓ | Not counting | |
| ↑ | Low | | |
| ↓ | High | | |
| High | ↓ | | |
| Low | ↑ | | |
| ↑ | High | Up-counting | |
| ↓ | Low | Not counting | |

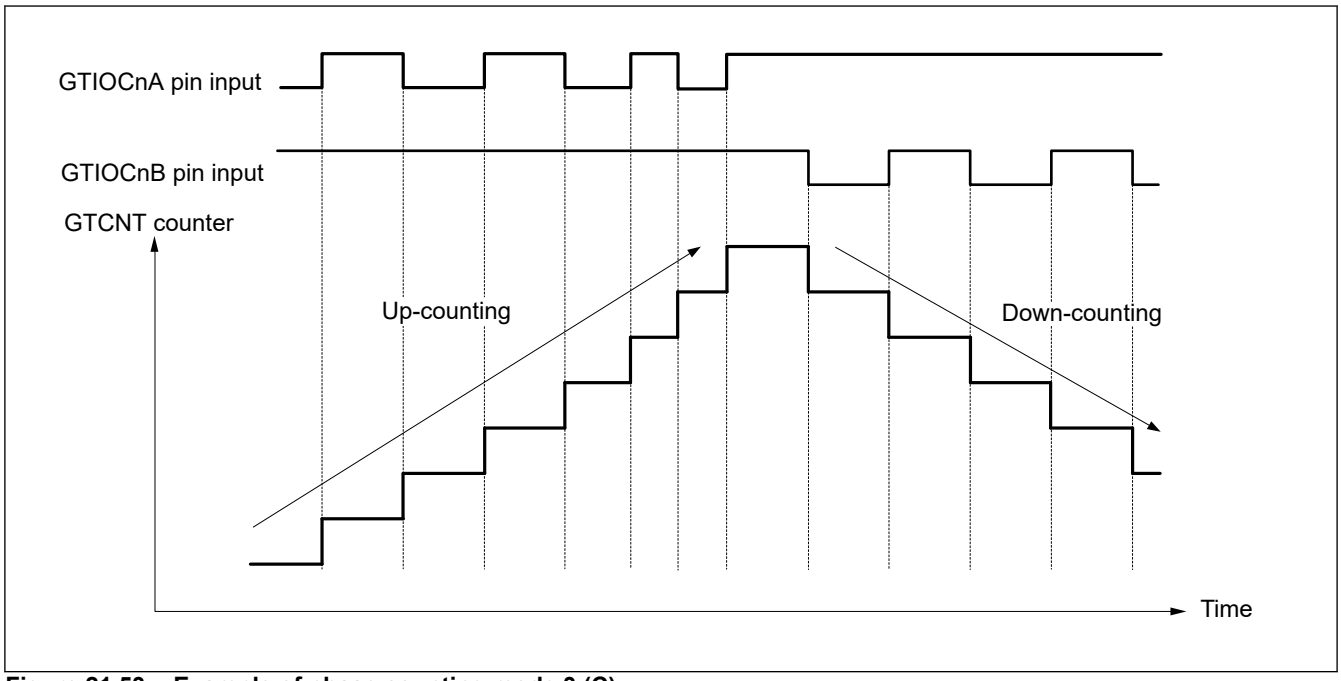



**Figure 21.53 Example of phase counting mode 3 (C)**

**Table 21.35 Conditions of up-counting/down-counting in phase counting mode 3 (C)**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | ↑ | Down-counting | GTUPSR = 0x00000A00<br>GTDNSR = 0x0000A000 |
| Low | ↓ | Not counting | |
| ↑ | Low | | |
| ↓ | High | Up-counting | |
| High | ↓ | Down-counting | |
| Low | ↑ | Not counting | |
| ↑ | High | Up-counting | |
| ↓ | Low | Not counting | |

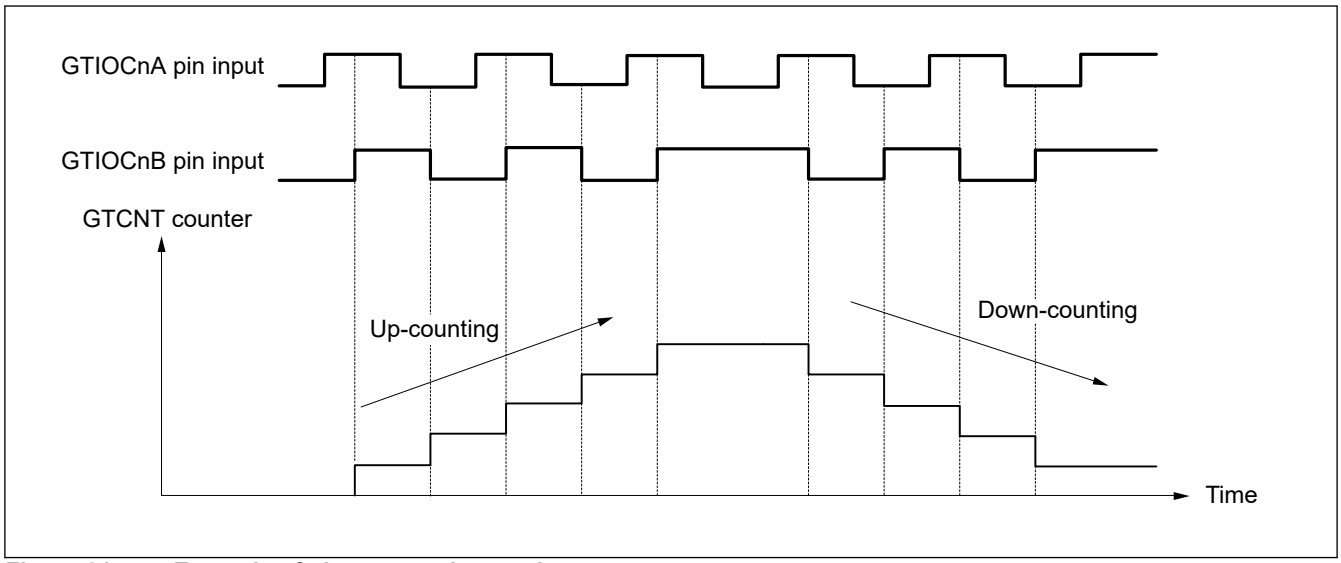



**Figure 21.54 Example of phase counting mode 4**

**Table 21.36 Conditions of up-counting/down-counting in phase counting mode 4**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | ↑ | Up-counting | GTUPSR = 0x00006000<br>GTDNSR = 0x00009000 |
| Low | ↓ | | |
| ↑ | Low | Not counting | |
| ↓ | High | | |
| High | ↓ | Down-counting | |
| Low | ↑ | | |
| ↑ | High | Not counting | |
| ↓ | Low | | |

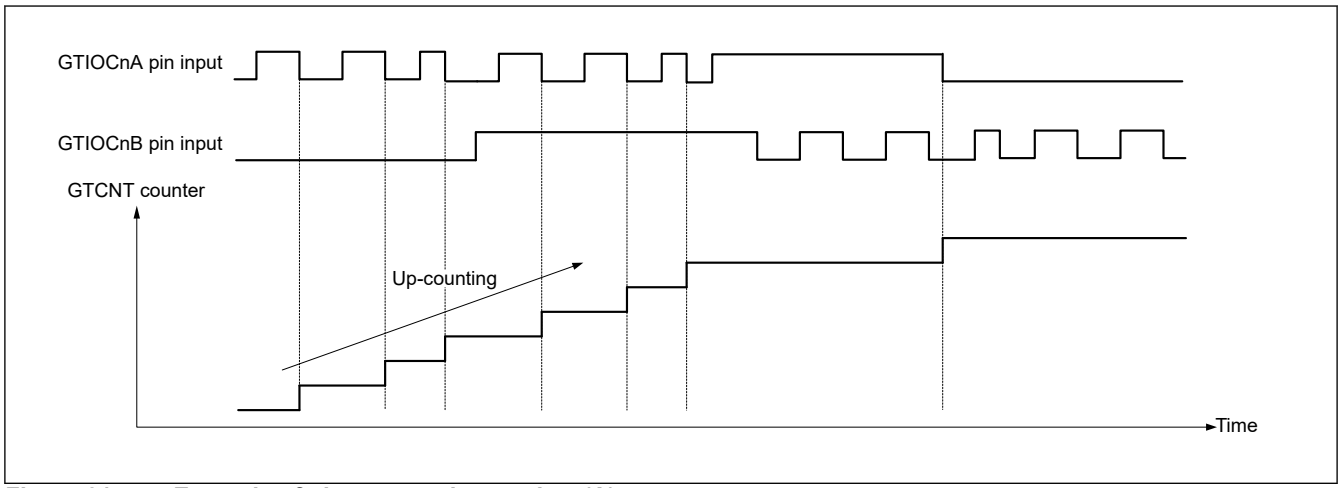



**Figure 21.55 Example of phase counting mode 5 (A)**

**Table 21.37 Conditions of up-counting/down-counting in phase counting mode 5 (A)**

: Rising edge

: Falling edge

| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | | Not counting | GTUPSR = 0x00000C00<br>GTDNSR = 0x00000000 |
| Low | | | |
| | Low | | |
| | High | Up-counting | |
| High | | Not counting | |
| Low | | | |
| | High | | |
| | Low | Up-counting | |

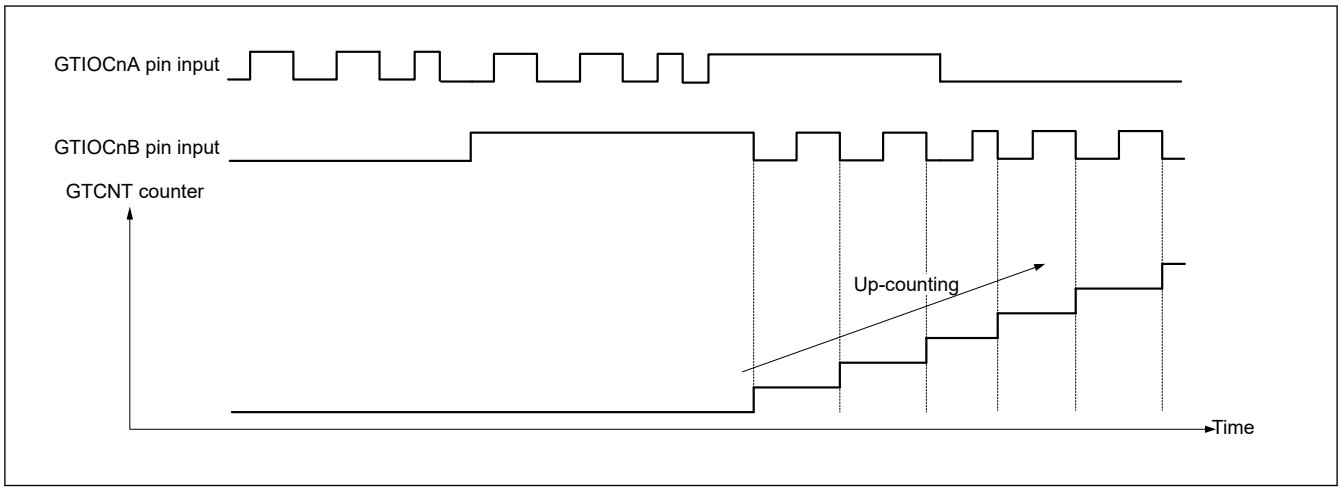



**Figure 21.56 Example of phase counting mode 5 (B)**

**Table 21.38 Conditions of up-counting/down-counting in phase counting mode 5 (B)**

: Rising edge

: Falling edge

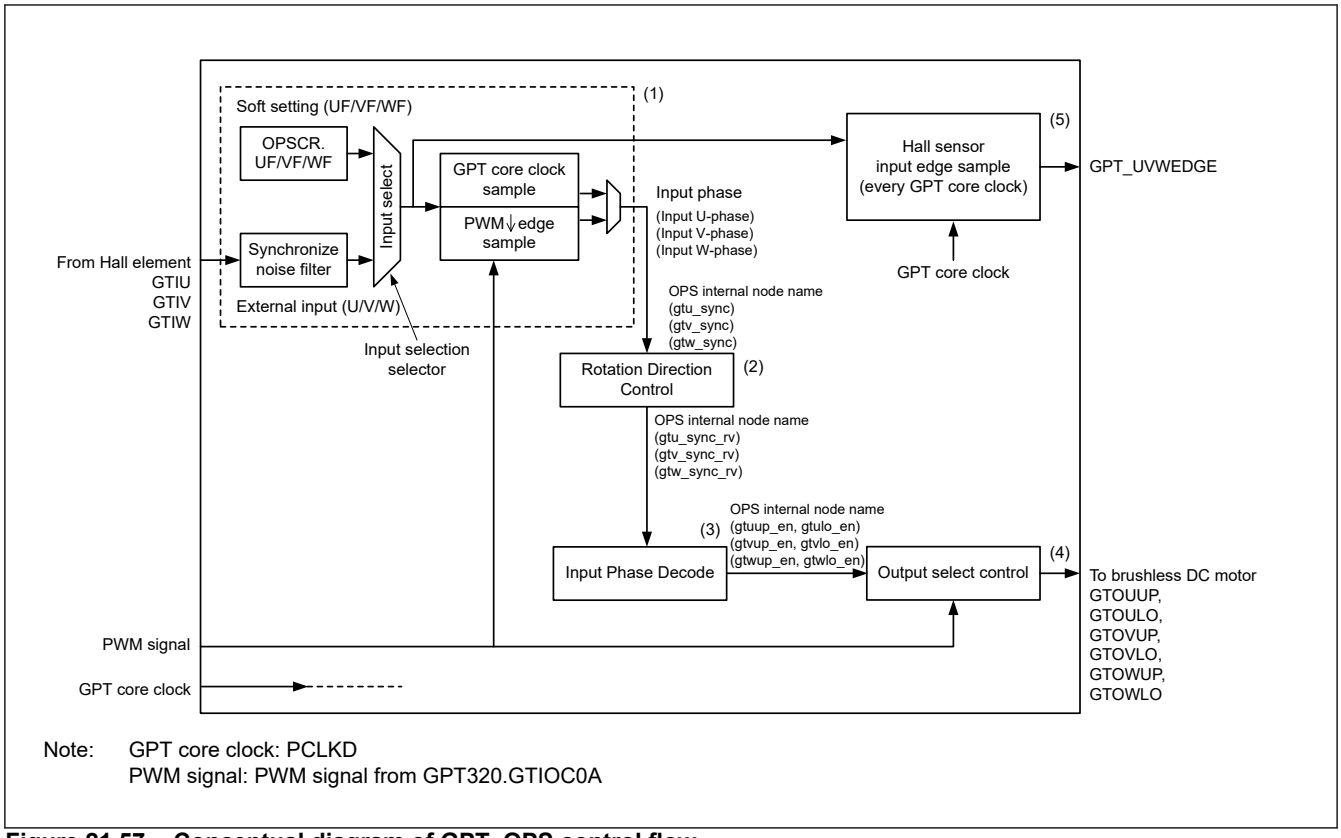
| GTIOCnA pin input | GTIOCnB pin input | Operation | Register setting |
|---|---|---|---|
| High | | Not counting | GTUPSR = 0x0000C000<br>GTDNSR = 0x00000000 |
| Low | | Up-counting | |
| | Low | Not counting | |
| | High | | |
| High | | Up-counting | |
| Low | | Not counting | |
| | High | | |
| | Low | | |

## 21.3.12 Output Phase Switching (GPT_OPS)

GPT_OPS provides a function for easy control of brushless DC motor operation using the Output Phase Switching Control Register (OPSCR).

GPT_OPS outputs a PWM signal to be used for chopper control or level signal for each phase (U-positive phase/negative phase, V-positive phase/negative phase, W-positive phase/negative phase) of the 6-phase motor control. This function uses a soft setting value (OPSCR.UF, VF, WF) set by software or external signals detected by the Hall element, a PWM waveform of GPT320.GTIOC0A.

Figure 21.57 shows the conceptual diagram of GPT_OPS control flow.

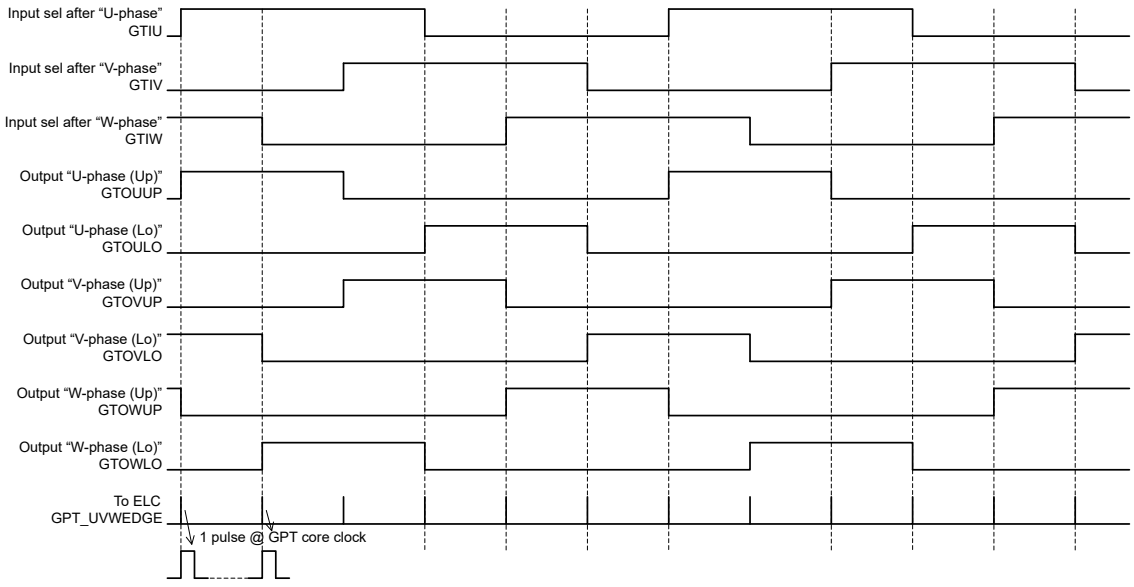



Note: GPT core clock: PCLKD
PWM signal: PWM signal from GPT320.GTIOC0A

**Figure 21.57 Conceptual diagram of GPT_OPS control flow**

Figure 21.58 shows a 6-phase level signals output example of a GPT_OPS operation.

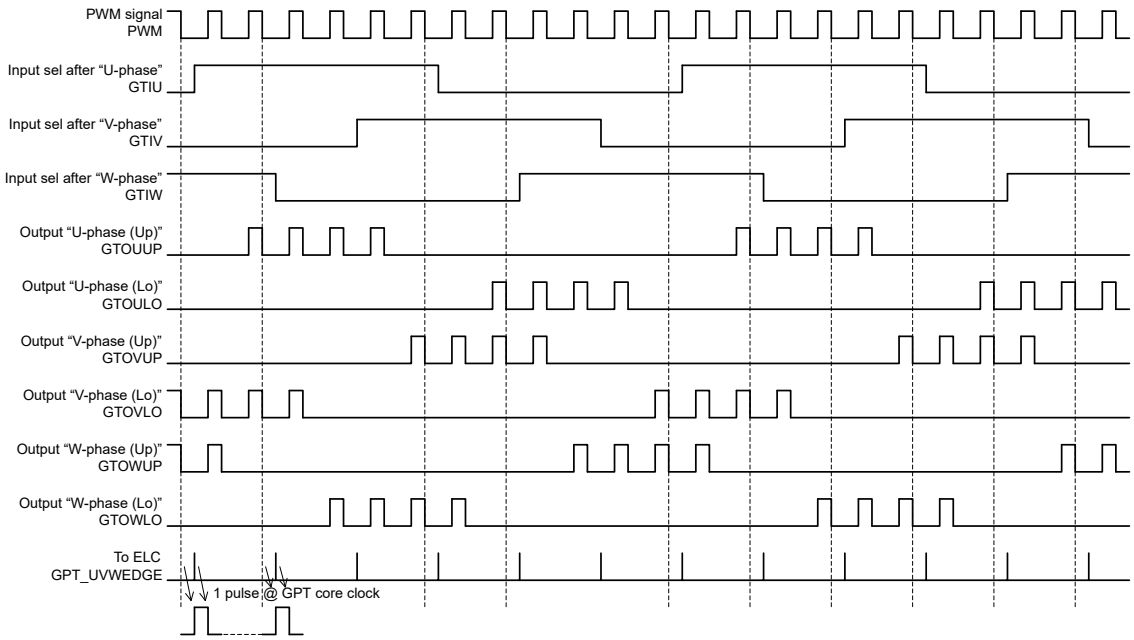
The GPT_UVWEDGE signal in Figure 21.58 is the Hall sensor input edge that outputs to the ELC.




Note: Register settings : OPSCR.ALIGN = 0, OPSCR.EN = 1, OPSCR.P = 0, OPSCR.N = 0, OPSCR.INV = 0, OPSCR.RV = 0
Note: GPT core clock: PCLKD

**Figure 21.58 Example of 6-phase level output operation**

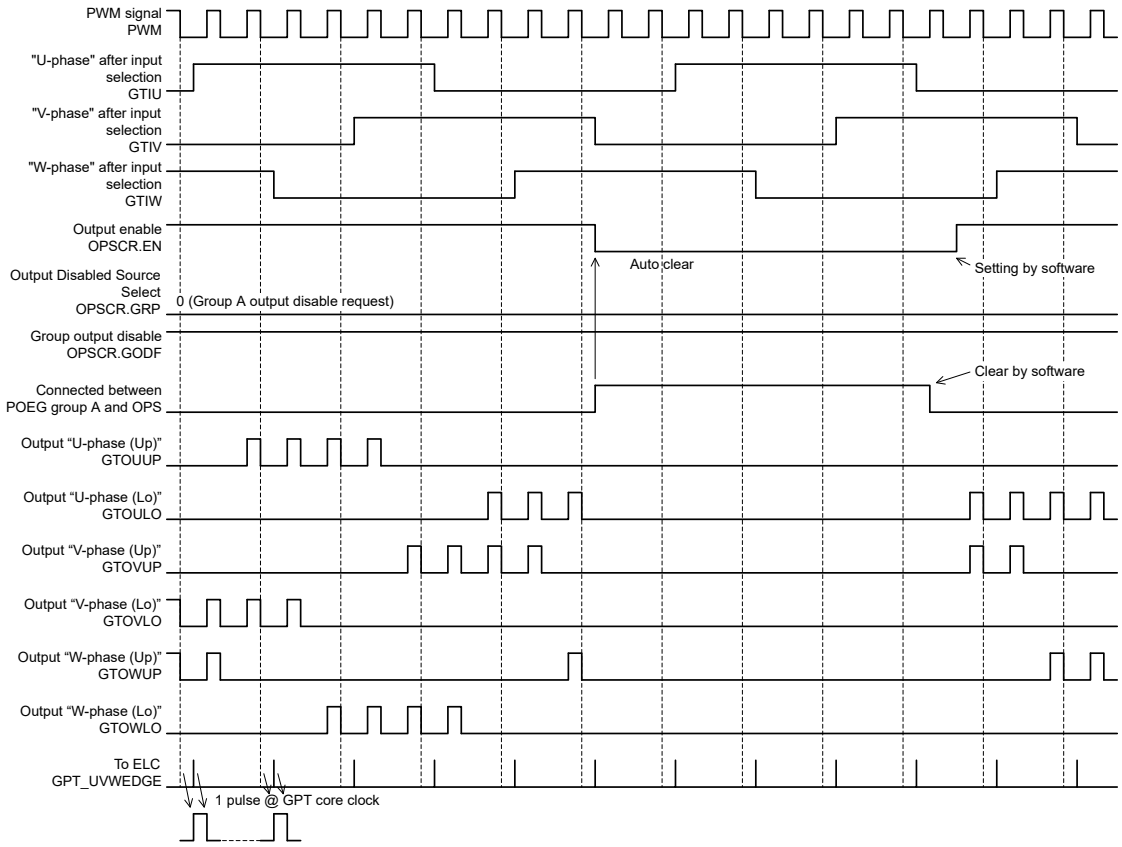
Figure 21.59 shows a 6-phase PWM output example of a GPT_OPS operation with chopper control.




Note: Register settings : OPSCR.ALIGN = 1, OPSCR.EN = 1, OPSCR.P = 1, OPSCR.N = 1, OPSCR.INV = 0, OPSCR.RV = 0
Note: GPT core clock: PCLKD
PWM signal: PWM signal from GPT320.GTIOC0A

**Figure 21.59 Example of 6-phase PWM output operation with chopper control**

Figure 21.60 shows a 6-phase PWM output example of an output disable control operation.




Note: Register settings : OPSCR.P = 1, OPSCR.N = 1, OPSCR.INV = 0, OPSCR.RV = 0
Note: GPT core clock: PCLKD
PWM signal: PWM signal from GPT320.GTIOC0A

**Figure 21.60 Example of group output disable control operation**

### 21.3.12.1 Input Selection and Synchronization of External Input Signal

In the GPT_OPS control flow conceptual diagram shown in Figure 21.57, (1) is a selection of input phase from the software settings and external input by the OPSCR.FB bit.

When OPSCR.FB bit is 0, select the external input. Enable the input signal after synchronization with the GPT core clock (PCLKD). After carrying out noise filtering (optional), set the external input to the input phase of PWM (PWM of GPT320.GTIOC0A) using falling edge sampling with OPSCR.ALIGN bit set to 1.

When OPSCR.FB bit is 1, select the soft setting (OPSCR.UF, VF, WF) with the value of the input phase of PWM (PWM of GPT320.GTIOC0A) using falling edge sampling with OPSCR.ALIGN bit set to 1.

When OPSCR.ALIGN bit is 0, GPT_OPS operates with the input phase of PCLKD synchronization with either OPSCR.FB bit set to 0 or OPSCR.FB bit set to 1. However, there are cases where the PWM pulse width of the output U/V/W phases (PWM output mode) of switch timing (just before/just after) is shortened.

Table 21.39 shows the input selection process and setting of associated OPSCR bits.

**Table 21.39 Input selection processing method**

| Register OPSCR | | Selection of input phase sampling method (U/V/W-phase) | Synchronization input/output selection process (GPT_OPS internal node name) |
|---|---|---|---|
| FB bit | ALIGN bit | | |
| 0 | 1 | External Input at PWM Falling Edge Sampling (PCLKD synchronization + falling edge sample) | Input Phase<br>Input U-Phase (gtu_sync)<br>Input V-Phase (gtv_sync)<br>Input W-Phase (gtw_sync) |
| | 0 | External Input at PCLKD Synchronization Output (PCLKD synchronization + through mode) | |
| 1 | 1 | Software Settings at PWM Falling Edge Sampling (OPSCR.UF, VF, WF of falling edge sample) | |
| | 0 | Software Setting Value Selection (= OPSCR.UF/VF/WF value) (= PCLKD synchronization) | |

### 21.3.12.2 Input Sampling

The OPSCR.U, V, W bits indicate the PCLKD sampling results of the input selected in the OPSCR.FB bit.

When OPSCR.FB bit is 0 and after synchronization with the GPT core clock (PCLKD) and noise filtering (optional), OPSCR.U, V, W bits indicate the sampling results of the external input. When OPSCR.FB bit is 1, OPSCR.U, V, W bits are the value (OPSCR.UF, VF, WF) of the soft setting.

### 21.3.12.3 Input Phase Decode

In the GPT_OPS control flow conceptual diagram shown in Figure 21.57, (3) enables the 6-phase signals by decoding the input phase selected in the OPSCR.FB bit.

Table 21.40 shows the decode table of input phase when OPSCR.RV bit is 0.

**Table 21.40 Decode table of input phase (OPSCR.RV = 0)**

| Input phase (U/V/W) (GPT_OPS internal node name) | | | 6-phase enable {U/V/W (Up/Lo)} by decoding input phase (GPT_OPS internal node name) | | | | | |
|---|---|---|---|---|---|---|---|---|
| Input U-Phase | Input V-Phase | Input W-Phase | U-phase (Up) | U-phase (Lo) | V-phase (Up) | V-phase (Lo) | W-phase (Up) | W-phase (Lo) |
| (gtu_sync) | (gtv_sync) | (gtw_sync) | (gtuup_en) | (gtulo_en) | (gtvup_en) | (gtvlo_en) | (gtwup_en) | (gtwlo_en) |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 21.41 Decode table of input phase (OPSCR.RV = 1) (1 of 2)**

| Input phase (U/V/W) (GPT_OPS internal node name) | | | 6-phase enable {U/V/W (Up/Lo)} by decoding input phase (GPT_OPS internal node name) | | | | | |
|---|---|---|---|---|---|---|---|---|
| Input U-Phase | Input V-Phase | Input W-Phase | U-phase (Up) | U-phase (Lo) | V-phase (Up) | V-phase (Lo) | W-phase (Up) | W-phase (Lo) |
| (gtu_sync) | (gtv_sync) | (gtw_sync) | (gtuup_en) | (gtulo_en) | (gtvup_en) | (gtvlo_en) | (gtwup_en) | (gtwlo_en) |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

**Table 21.41 Decode table of input phase (OPSCR.RV = 1) (2 of 2)**

| Input phase (U/V/W) (GPT_OPS internal node name) | | | 6-phase enable {U/V/W (Up/Lo)} by decoding input phase (GPT_OPS internal node name) | | | | | |
|---|---|---|---|---|---|---|---|---|
| Input U-Phase | Input V-Phase | Input W-Phase | U-phase (Up) | U-phase (Lo) | V-phase (Up) | V-phase (Lo) | W-phase (Up) | W-phase (Lo) |
| (gtu_sync) | (gtv_sync) | (gtw_sync) | (gtuup_en) | (gtulo_en) | (gtvup_en) | (gtvlo_en) | (gtwup_en) | (gtwlo_en) |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

### 21.3.12.4 Rotation Direction Control

In the GPT_OPS control flow conceptual diagram shown in Figure 21.57, (2) controls the direction of rotation of a 3-phase motor using the OPSCR.RV bit.

When the rotation direction is reverse (RV bit = 1), the input phase is inverted.

Table 21.42 shows the assigned output phases based on the OPSCR.RV bit setting (before and after rotation direction control).

**Table 21.42 Rotation Direction Control Method**

| Reversal of Direction of Rotation Using Output Phases as Specified in OPSCR Register | Output of Rotation Direction Control [U/V/W (Positive/Negative)] (GPT_OPS Internal Node Name after Control) | | | | | |
|---|---|---|---|---|---|---|
| OPSCR.RV bit | (gtuup_ren) | (gtulo_ren) | (gtvup_ren) | (gtvlo_ren) | (gtwup_ren) | (gtwlo_ren) |
| 0 | U-phase (Up) (gtuup_en) | U-phase (Lo) (gtulo_en) | V-phase (Up) (gtvup_en) | V-phase (Lo) (gtvlo_en) | W-phase (Up) (gtwup_en) | W-phase (Lo) (gtwlo_en) |
| 1 | U-phase (Up) (gtuup_en) | U-phase (Lo) (gtulo_en) | W-phase (Up) (gtwup_en) | W-phase (Lo) (gtwlo_en) | V-phase (Up) (gtvup_en) | V-phase (Lo) (gtvlo_en) |

### 21.3.12.5 Output Selection Control

In the GPT_OPS control flow conceptual diagram in Figure 21.57, (4) represents the selection of the output waveform by setting the OPSCR register bit.

For output selection, the following bits are relevant:

- The OPSCR.EN bit controls whether to output the 6-phase output, or to stop
- The OPSCR.P and OPSCR.N bits can select from the level signal or PWM signal (chopper output) for the output phase
- The polarity of the output phase can be set to a positive logic or negative logic by the OPSCR. INV bit.

Table 21.43 and Table 21.44 show the output selection control method using the OPSCR register bit.

**Table 21.43 Output selection control method (positive phase) (1 of 2)**

| Enable-phase output control | Positive-phase output (P) control | Invert-phase output control | Output port name (positive phase = up) (output selection internal node allocation) | |
|---|---|---|---|---|
| OPSCR.EN | OPSCR.P | OPSCR.INV | GTOUUP GTOVUP GTOWUP | Mode |
| 0 | x | x | 0 | Output Stop (External pin: Hi-Z) GPT_OPS → 0 output |

**Table 21.43 Output selection control method (positive phase) (2 of 2)**

| Enable-phase output control | Positive-phase output (P) control | Invert-phase output control | Output port name (positive phase = up) (output selection internal node allocation) | |
|---|---|---|---|---|
| OPSCR.EN | OPSCR.P | OPSCR.INV | GTOUUP<br>GTOVUP<br>GTOWUP | Mode |
| 1 | 0 | 0 | Level signal<br>(gtuup_ren)<br>(gtvup_ren)<br>(gtwup_ren) | Level Output Mode<br>(Positive phase)<br>(Positive logic) |
| 1 | 0 | 1 | Level signal<br>(~gtuup_ren)<br>(~gtvup_ren)<br>(~gtwup_ren) | Level Output Mode<br>(Positive phase)<br>(Negative logic) |
| 1 | 1 | 0 | PWM signal<br>(PWM & gtuup_ren)<br>(PWM & gtvup_ren)<br>(PWM & gtwup_ren) | PWM Output Mode<br>(Positive phase)<br>(Positive logic) |
| 1 | 1 | 1 | PWM signal<br>(~(PWM & gtuup_ren))<br>(~(PWM & gtvup_ren))<br>(~(PWM & gtwup_ren)) | PWM Output Mode<br>(Positive phase)<br>(Negative logic) |

**Table 21.44 Output selection control method (negative phase)**

| Enable-phase output control | Positive-phase output (N) control | Invert-phase output control | Output port name (negative phase = Lo) (output selection internal node allocation) | |
|---|---|---|---|---|
| OPSCR.EN | OPSCR.N | OPSCR.INV | GTOULO<br>GTOVLO<br>GTOWLO | Mode |
| 0 | x | x | 0 | Output Stop<br>(External pin: Hi-Z)<br>GPT_OPS → 0 output |
| 1 | 0 | 0 | Level signal<br>(gtulo_ren)<br>(gtvlo_ren)<br>(gtwlo_ren) | Level Output Mode<br>(Negative phase)<br>(Positive logic) |
| 1 | 0 | 1 | Level signal<br>(~gtulo_ren)<br>(~gtvlo_ren)<br>(~gtwlo_ren) | Level Output Mode<br>(Negative phase)<br>(Negative logic) |
| 1 | 1 | 0 | PWM signal<br>(PWM & gtulo_ren)<br>(PWM & gtvlo_ren)<br>(PWM & gtwlo_ren) | PWM Output Mode<br>(Negative phase)<br>(Positive logic) |
| 1 | 1 | 1 | PWM signal<br>(~(PWM & gtulo_ren))<br>(~(PWM & gtvlo_ren))<br>(~(PWM & gtwlo_ren)) | PWM Output Mode<br>(Negative phase)<br>(Negative logic) |

### 21.3.12.6 Output Selection Control (Group Output Disable Function)

When OPSCR.GODF is 1 and the signal value selected by the OPSCR.GRP bit is high (output disable request), the group output-disable function asynchronously sets the output to Hi-Z. When an output-disable request is generated, the OPSCR.EN bit is cleared to 0. For the return, set the OPSCR.EN bit to 1 after clearing the output disable request by software.

To ensure output-disable control, use the POEG_GROUPn (n = A to D) interrupt to clear the flag in the POE or check that the OPSCR.EN bit is 0 and then clear the flag. For an example of the operation for group output disable control, see Figure 21.60.

### 21.3.12.7 Event Link Controller (ELC) Output

In the GPT_OPS control flow conceptual diagram shown in Figure 21.57, (5) outputs the Hall sensor input signal edge to the ELC.

The Hall sensor input edge signal is the logical OR of the rising and falling edge signals of each U-phase/V-phase/W-phase input sampled at PCLKD. That is, if the high period of each of the U-phase/V-phase/W-phase of the input phase is short in duration, the Hall sensor edge input signal is not output at that time.

When the OPSCR.FB bit is 0, the Hall sensor input edge signal is the logical OR of the edge signals of the external input phase sampled at PCLKD.

When OPSCR.FB bit is 1, the Hall sensor input edge signal is the logical OR of the edge of the soft setting (OPSCR.UF, VF, WF) sampled at PCLKD.

See Figure 21.58 to Figure 21.60 for examples of the output signal to the ELC.

### 21.3.12.8 GPT_OPS Start Operation Setting Flow

**Table 21.45 Example setting of GPT_OPS start operation**

| No. | Step Name | Description |
|---|---|---|
| 1 | GPT320 operation mode setting | GPT320.GTIOC0A set the PWM output operation mode of the saw-wave or triangle-wave.<br>For details, see section 21.3.3. PWM Output Operating Mode. |
| 2 | Counting of GPT320 | Start the count operation of GPT320, and outputs a PWM waveform. |
| 3 | GPT_OPS input data set (only software setting is selected) | Set software setting to OPSCR.UF, VF, and WF bits. |
| 4 | Noise filter settings of GPT_OPS external input (only external input is selected) | When using a noise filter, set the sampling clock of the noise filter by OPSCR.NFCS[1:0] bits.<br>Then the noise filter is enabled if OPSCR.NFEN = 1. |
| 5 | GPT_OPS input phase selection setting/input phase alignment setting | Select the input phase from the external input or software setting by OPSCR.FB bit.<br>Select the alignment of the input phase by OPSCR.ALIGN bit. |
| 6 | Setting the GPT_OPS output phase | Set the level output/PWM output of the positive/negative phase output by OPSCR.P/OPSCR.N bit.<br>Set the positive logic/negative logic of the output phase by OPSCR.INV bit. Set the rotation direction by OPSCR.RV bit |
| 7 | GPT_OPS setting the group output disable function | Set the selection of output disable source by OPSCR.GRP bit.<br>Perform the setting of on/off of the group output disable function by OPSCR.GODF bit. |
| 8 | GPT_OPS Working | Setting the OPSCR.EN = 1 outputs the 6-phase output to drive the brushless DC motor from the GPT_OPS. |

## 21.3.13 Inter Channel Logical Operation Function

The logical operation function between compare match outputs can be performed.

Figure 21.61 shows the block diagram of inter channel logical operation.

To prevent hazard to the GPT output, the signal after logical operation is latched with PCLKD. After latching, the output disable control is performed.

When the logical operation function which causes the delay of 1 PCLKD is selected, the output enable signal is also delayed with 1 PCLKD and input to the output disable control.

When the same signal (C = A or D = B) to operate logical function AND, OR, EXOR and NOR is selected, C or D is treated as 1. In the case of GTIOCnA pin output, when A of same channel is selected for C, the result of AND is A, the result of OR is 1, the result of EXOR is NOT A, and the result of NOR is 0.

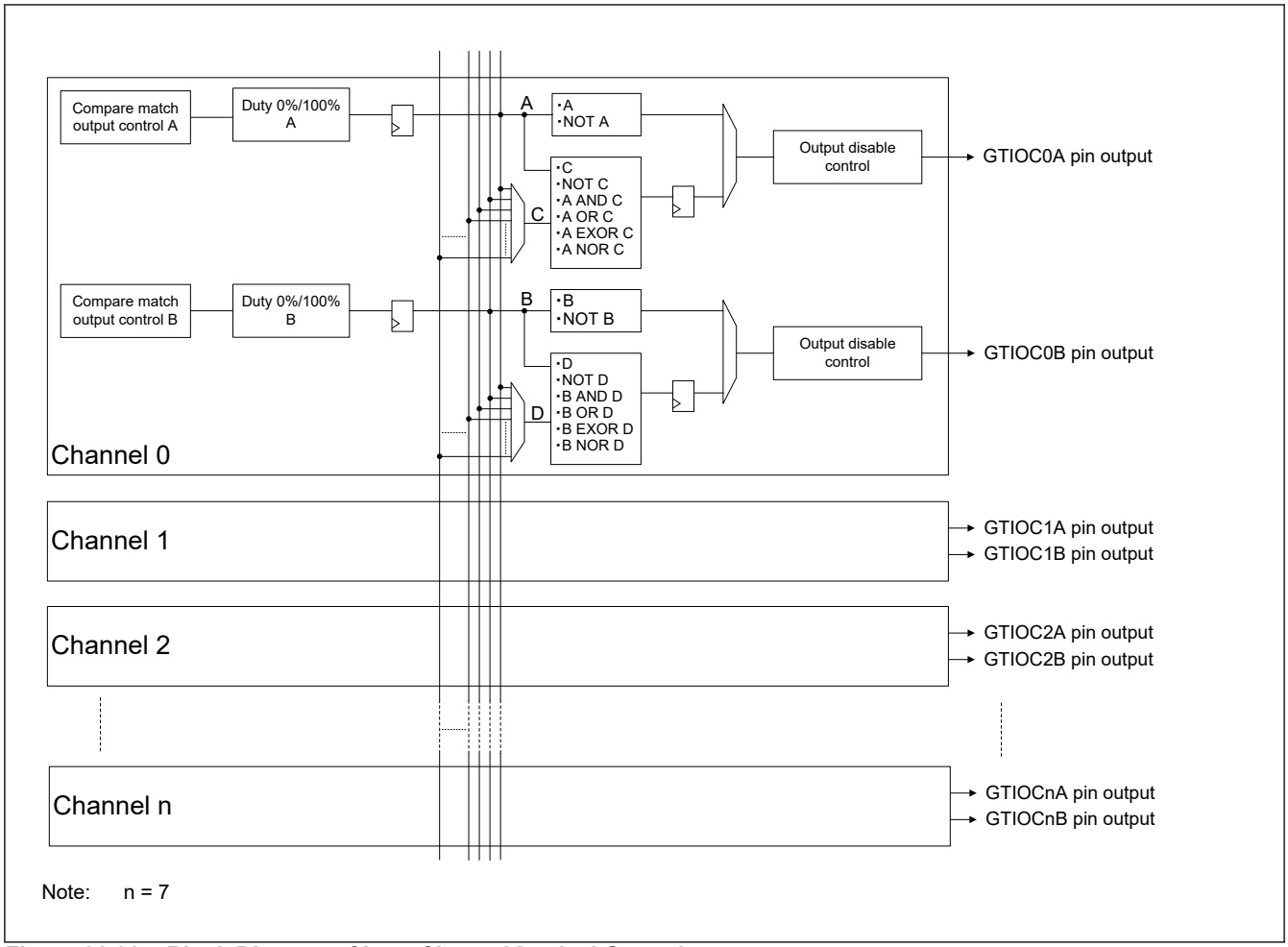



**Figure 21.61 Block Diagram of Inter Channel Logical Operation**

Figure 21.62 shows an example of inter channel logical operation.

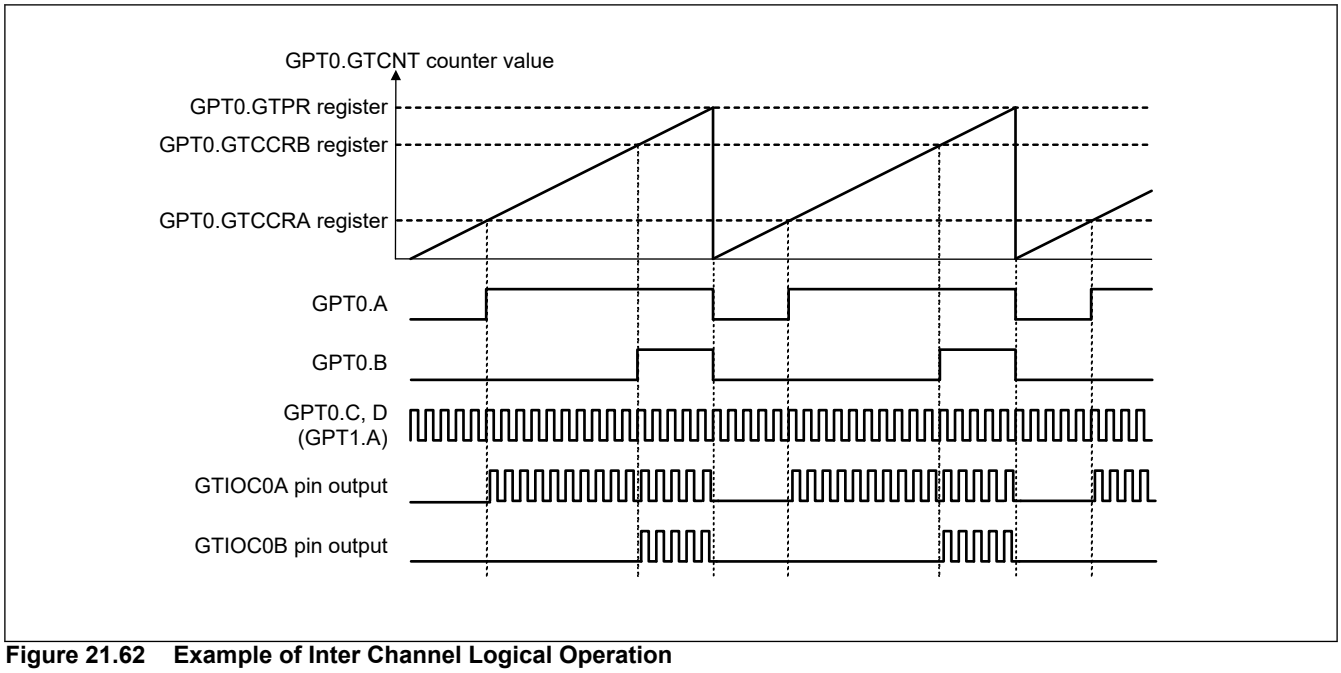



**Figure 21.62 Example of Inter Channel Logical Operation**

## 21.4 Interrupt Sources

### 21.4.1 Interrupt Sources

The GPT provides the following interrupt sources:

- GTCCR input capture/compare match
- GTCNT counter overflow (GTPR compare match)/underflow.
- period count function finish

Each interrupt source has its own status flag. When an interrupt source signal is generated, the associated status flag in GTST is set to 1. The associated status flag in GTST can be cleared by writing 0. If flag set and flag clear occur at the same time, flag clear takes priority over flag set. These flags are automatically updated by the internal state. The Interrupt Controller Unit can change the relative channel priorities. However, the priority within a channel is fixed. For details, see section 13, Interrupt Controller Unit (ICU).

Table 21.46 lists the GPT interrupt sources.

**Table 21.46 Interrupt sources**

| Channel | Name | Interrupt source | Interrupt flag | DTC activation |
|---|---|---|---|---|
| n = 0 to 3 | GPTn_CCMPA | GPT32n.GTCCRA input capture/compare match | GTST[0] (TCFA) | Possible |
| | GPTn_CCMPB | GPT32n.GTCCRB input capture/compare match | GTST[1] (TCFB) | Possible |
| | GPTn_CMPC | GPT32n.GTCCRC compare match | GTST[2] (TCFC) | Possible |
| | GPTn_CMPD | GPT32n.GTCCRD compare match | GTST[3] (TCFD) | Possible |
| | GPTn_CMPE | GPT32n.GTCCRE compare match | GTST[4] (TCFE) | Possible |
| | GPTn_CMPF | GPT32n.GTCCRF compare match | GTST[5] (TCFF) | Possible |
| | GPTn_OVF | GPT32n.GTCNT overflow (GPT32m.GTPR compare match) | GTST[6] (TCFPO) | Possible |
| | GPTn_UDF | GPT32n.GTCNT underflow | GTST[7] (TCFPU) | Possible |
| | GPTn_PC | Period count function finish (n = 0, 1) | GTST[31] (PCF) | Possible |
| n = 4 to 7 | GPTn_CCMPA | GPT16n.GTCCRA input capture/compare match | GTST[0] (TCFA) | Possible |
| | GPTn_CCMPB | GPT16n.GTCCRB input capture/compare match | GTST[1] (TCFB) | Possible |
| | GPTn_CMPC | GPT16n.GTCCRC compare match | GTST[2] (TCFC) | Possible |
| | GPTn_CMPD | GPT16n.GTCCRD compare match | GTST[3] (TCFD) | Possible |
| | GPTn_CMPE | GPT16n.GTCCRE compare match | GTST[4] (TCFE) | Possible |
| | GPTn_CMPF | GPT16n.GTCCRF compare match | GTST[5] (TCFF) | Possible |
| | GPTn_OVF | GPT16n.GTCNT overflow (GPT320.GTPR compare match) | GTST[6] (TCFPO) | Possible |
| | GPTn_UDF | GPT16n.GTCNT underflow | GTST[7] (TCFPU) | Possible |
| | GPTn_PC | Period count function finish (n = 4 to 6) | GTST[31] (PCF) | Possible |

#### (1) GPTn_CCMPA interrupt (n = 0 to 7)

An interrupt request is generated under the following conditions:

- When the GTCCRA register functions as a compare match register, the GTCNT counter value matches with the GTCCRA register
- When the GTCCRA register functions as an input capture register, the input-capture signal causes transfer of the GTCNT counter value to the GTCCRA register.

#### (2) GPTn_CCMPB interrupt (n = 0 to 7)

An interrupt request is generated under the following conditions:

- When the GTCCRB register functions as a compare match register, the GTCNT counter value matches with the GTCCRB register

- When the GTCCRB register functions as an input capture register, the input-capture signal causes transfer of the GTCNT counter value to the GTCCRB register.

### (3) GPTn_CMPC interrupt (n = 0 to 7)

An interrupt request is generated under the following condition:

- When the GTCCRC register functions as a compare match register, the GTCNT counter value matches with the GTCCRC register.

A compare match is not performed and therefore, an interrupt is not requested in the following conditions:

- GTCR.MD[2:0] = 001b (saw-wave one-shot pulse mode)
- GTCR.MD[2:0] = 110b (triangle-wave PWM mode 3)
- GTBER.CCRA[1:0] = 01b, 10b, 11b (buffer operation with the GTCCRC register).

### (4) GPTn_CMPD interrupt (n = 0 to 7)

An interrupt request is generated under the following condition:

- When the GTCCRD register functions as a compare match register, the GTCNT counter value matches with the GTCCRD register.

A compare match is not performed and therefore, an interrupt is not requested in the following conditions:

- GTCR.MD[2:0] = 001b (saw-wave one-shot pulse mode)
- GTCR.MD[2:0] = 110b (triangle-wave PWM mode 3)
- GTBER.CCRA[1:0] = 10b, 11b (buffer operation with the GTCCRD register).

### (5) GPTn_CMPE interrupt (n = 0 to 7)

An interrupt request is generated under the following condition:

- When the GTCCRE register functions as a compare match register, the GTCNT counter value matches with the GTCCRE register.

A compare match is not performed and therefore, an interrupt is not requested in the following conditions:

- GTCR.MD[2:0] = 001b (saw-wave one-shot pulse mode)
- GTCR.MD[2:0] = 110b (triangle-wave PWM mode 3)
- GTBER.CCRB[1:0] = 01b, 10b, 11b (buffer operation with the GTCCRE register).

### (6) GPTn_CMPF interrupt (n = 0 to 7)

An interrupt request is generated under the following condition:

- When the GTCCRF register functions as a compare match register, the GTCNT counter value matches with the GTCCRF register.

A compare match is not performed and therefore, an interrupt is not requested in the following conditions:

- GTCR.MD[2:0] = 001b (saw-wave one-shot pulse mode)
- GTCR.MD[2:0] = 110b (triangle-wave PWM mode 3)
- GTBER.CCRB[1:0] = 10b, 11b (buffer operation with the GTCCRF register).

### (7) GPTn_OVF interrupt (n = 0 to 7)

An interrupt request is generated in the following conditions:

- In saw-wave mode, interrupt requests are enabled at overflows (when the GTCNT counter value changes from GTPR to 0 during up-counting)
- In triangle-wave mode, interrupt requests are enabled at crests (the GTCNT changes from GTPR to GTPR-1)

- In counting by hardware sources, an overflow (GTCNT changes from GTPR to 0 in up count) has occurred.

#### (8) GPTn_UDF interrupt (n = 0 to 7)

An interrupt request is generated in the following conditions.

- In saw-wave mode, interrupt requests are enabled at underflows (when the GTCNT counter value changes from 0 to GTPR during down-counting)
- In triangle-wave mode, interrupt requests are enabled at troughs (the GTCNT changes from 0 to 1)
- In counting by hardware sources, underflow (GTCNT changes from 0 to GTPR in down count) has occurred.

About Interrupt signals and interrupt status flags, see section 21.2.16. GTST : General PWM Timer Status Register.

#### (9) GPTn_PC Interrupt (n = 0, 1, 4 to 6)

When the GTPC.PCEN bit is 1 and the GTPC.PCNT counter is 1, an interrupt request is generated at the end of cycle.

### 21.4.2 DMAC and DTC Activation

The DMAC and DTC can be activated by the interrupt in each channel. For details, see section 13, Interrupt Controller Unit (ICU), section 16, DMA Controller (DMAC), and section 17, Data Transfer Controller (DTC).

## 21.5 Operations Linked by ELC

### 21.5.1 Event Signal Output to ELC

The GPT can perform operation linked with another module set in advance when its interrupt request signal is used as an event signal by the Event Link Controller (ELC).

The GPT has the following ELC event signals:

- Generation of compare match and input capture A interrupt (GPTn_CCMPA)
- Generation of compare match and input capture B interrupt (GPTn_CCMPB)
- Generation of compare match C interrupt (GPTn_CMPC)
- Generation of compare match D interrupt (GPTn_CMPD)
- Generation of compare match E interrupt (GPTn_CMPE)
- Generation of compare match F interrupt (GPTn_CMPF)
- Generation of overflow interrupt (GPTn_OVF)
- Generation of underflow interrupt (GPTn_UDF)
- Finish of period count function (GPTm_PC)

Note: n = 0 to 7
m = 0, 1, 4 to 6

### 21.5.2 Event Signal Inputs from ELC

The GPT can perform the following operations in response to a maximum of 8 events from the ELC:

- Start counting, stop counting, clear counting
- Up-counting, down-counting
- Input capture.

See section 18, Event Link Controller (ELC) for the connection between the ELC and the event signal input.

## 21.6 Noise Filter Function

Each pin for use in input capture and Hall sensor input to the GPT is equipped with a noise filter. The noise filter samples input signals at the sampling clock and removes the pulses whose length is less than 3 sampling cycles.

The noise filter functionality includes enabling and disabling the noise filter for each pin and setting of the sampling clock for each channel.

Figure 21.63 shows the timing of noise filtering.

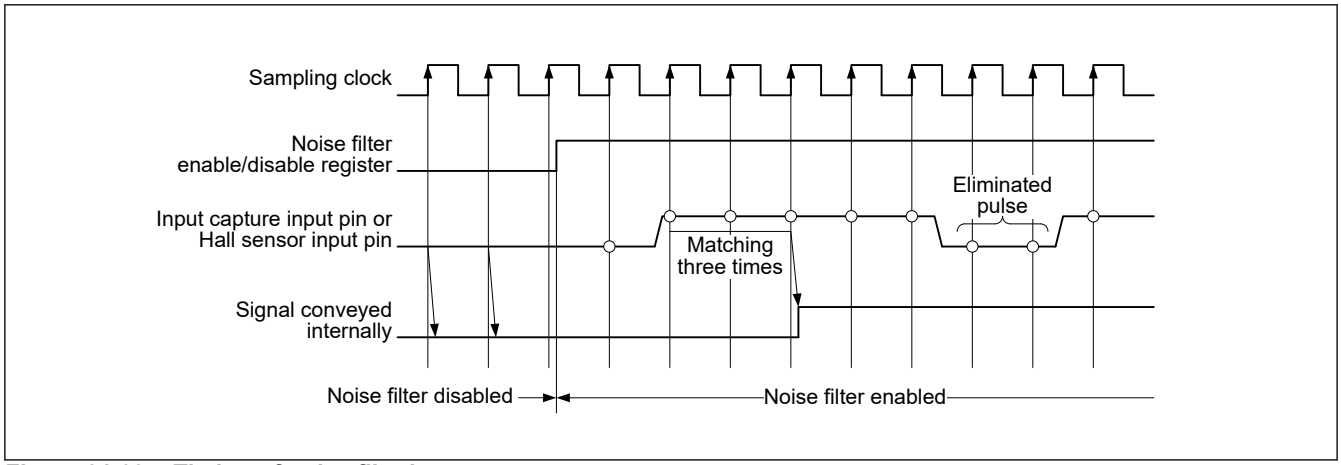



**Figure 21.63 Timing of noise filtering**

If noise filtering is enabled, the input capture operation or hall sensor input operation is performed on the edges of the noise filtered signal after a delay of (sampling interval × 2 + PCLKD) at the shortest. This is due to the noise filtering for the input capture input or hall sensor input.

## 21.7 Protection Function

### 21.7.1 Write-Protection for Registers

To prevent registers from being accidentally modified, registers can be write-protected in channel units by setting GTWP.WP. Write-protection can be set for the following registers:

GTSSR, GTPSR, GTCSR, GTUPSR, GTDNSR, GTICASR, GTICBSR, GTCR, GTUDDTYC, GTIOR, GTINTAD, GTST, GTBER, GTCNT, GTCCRA, GTCCRB, GTCCRC, GTCCRD, GTCCRE, GTCCRF, GTPR, GTPBR, GTDTCR, GTDVU, GTICLF, GTPC.

Every bit in registers GTSTR, GTSTP and GTCLR which can update the corresponding registers in other channels and can be updated by any of the corresponding registers in other channels conversely, can be protected by setting the GTWP.STRWP, STPWP, and CLRWP bits, respectively, per channel.

Likewise, writing to the GTSECSR and GTSECR registers, which can control all channels by writing to the GTSECSR and GTSECR registers of a given channel, can by enabled or disabled by the setting of the GTWP.CMNWP bit.

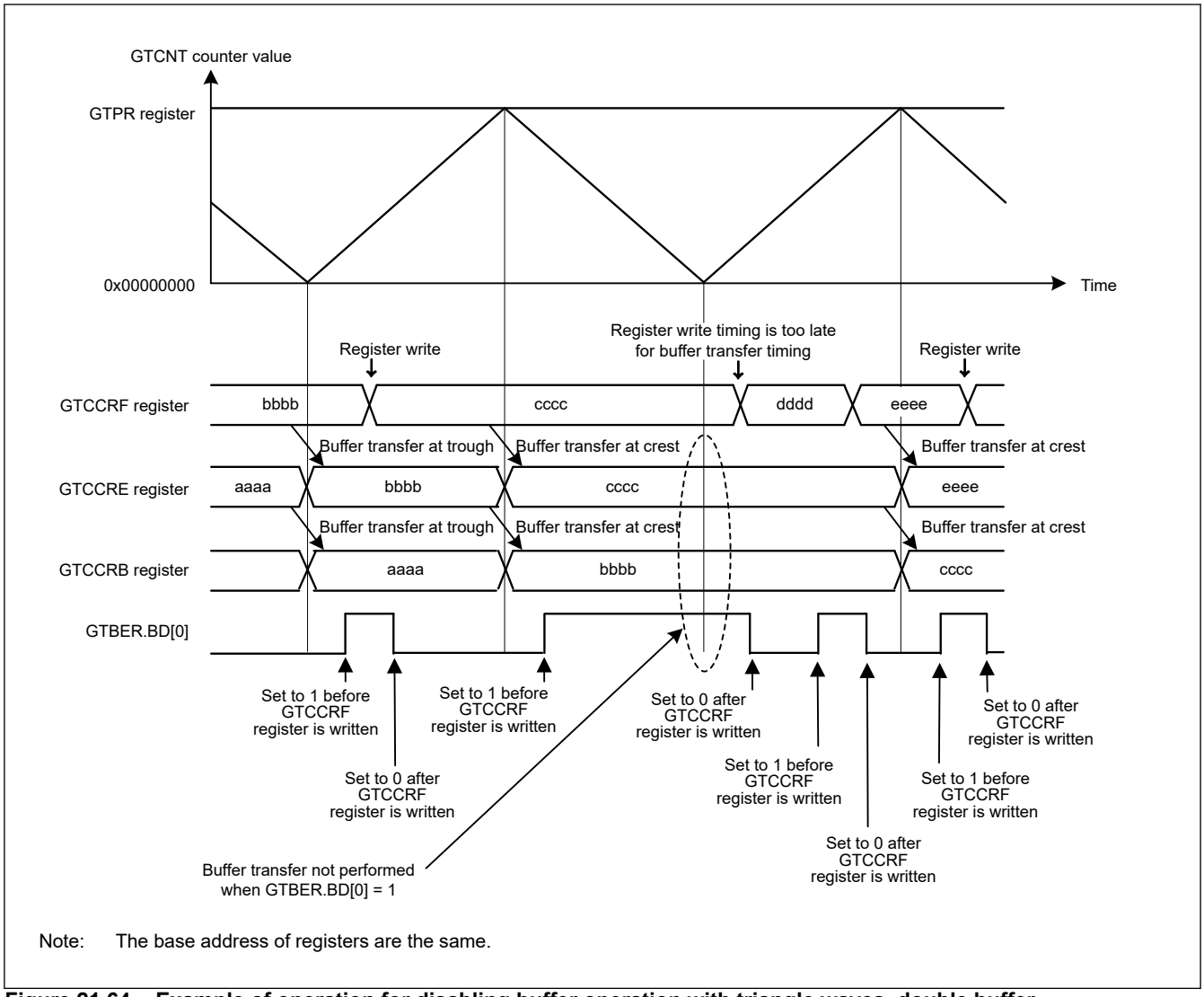
Protection using the GTWP register is only for write operations by the CPU. This protection does not cover updates to registers that occur in association with CPU writes.

### 21.7.2 Disabling of Buffer Operation

If the timing of the buffer register write is delayed relative to the timing for the buffer transfer, buffer operation can be suspended with the GTBER.BD[1] and BD[0] bits settings. Specifically, buffer transfer can be temporarily disabled even though a buffer transfer condition is generated during buffer register write, by setting the BD[1] and BD[0] bits to 1 (buffer operation disabled) before buffer register write, and setting the bits to 0 (buffer operation enabled) after completion of writing to all the buffer registers.

The BD[1] and BD[0] bits can be set on channel basis by writing directly to the GTBER register or it can be set to 0 simultaneously by setting the GTSECR register for multiple channels which were set by the GTSECSR register.

Figure 21.64 shows an example of operation for disabling buffer operation by writing to the GTBER register.




Note: The base address of registers are the same.

**Figure 21.64 Example of operation for disabling buffer operation with triangle waves, double buffer operation, and buffer transfer at both troughs and crests**

### 21.7.2.1 Simultaneous Control of Buffer Operations of Multiple Channels

The GTBER.BD bit can be set by writing directly to the GTBER register per channel or by making settings in the GTSECR register for multiple channels that have already set in the GTSECSR register.

Follow the procedure below to simultaneously set the GTBER.BD bits of multiple channels.

1. Select the channels for simultaneously setting by the GTSECSR register
   Set the GTSECSR register so that the values at the bit positions for the corresponding channels for simultaneously setting of the GTBER.BD bits become 1. All GTSECSR registers can be updated by writing to the GTSECSR register of any channel.
2. Simultaneously set the GTBER.BD bits by updating the GTSECR register
   In the GTSECR register, set the operation of the GTBER.BD bits (enabling or disabling of buffer operation) which are to be simultaneously set. Writing to a GTSECR register from any channel updates the GTBER.BD bits in all channels corresponding to the bits set as 1 in the GTSECSR register, in accordance with the value of the GTSECR register.

Figure 21.65 show examples of simultaneously controlling the enabling or disabling of buffer operation for multiple channels.

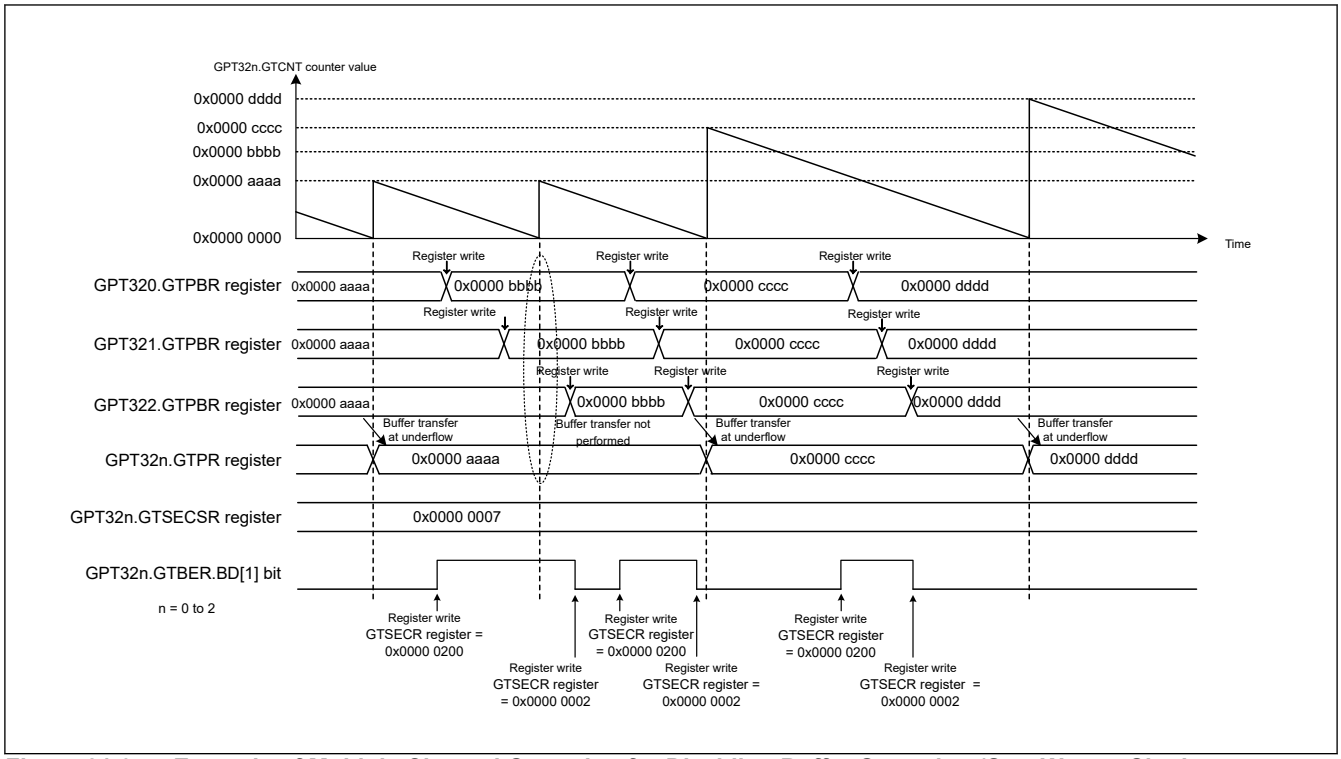



**Figure 21.65 Example of Multiple Channel Operation for Disabling Buffer Operation (Saw Waves, Single Buffer Operation)**

## 21.7.3 GTIOCnm Pin Output Negate Control (n = 0 to 7, m = A, B)

For protection from system failure, the output disable control that changes the GTIOCnm pin output value forcibly is provided for GTIOCnm pin output by the request of output disable from POEG. Output protection is required when the same output level being on the GTIOCnA and GTIOCnB pins is detected. GPT detects this condition and generates output disable requests to POEG according to the setting of the output disable request permission bits, such as GTINTAD.GRPABH, GTINTAD.GRPABL. After the POEG performs the logical OR of the output disable request from each channel and the output disable request from the external input, the POEG generates output disable requests to GPT.

One output disable signal (representing the shared output disable request signal of the GTIOCnA pin and the GTIOCnB pin) out of 4 output disable requests generated by the POEG is selected by setting GTINTAD.GRP[1:0]. The status of the selected disable output request is monitored by reading the GTST.ODF bit. The output level during output disable is set based on the GTIOR.OADF[1:0] bits for the GTIOCnA pin and the GTIOR.OBDF[1:0] setting for the GTIOCnB pin.

The change to the output disable state is performed asynchronously by generating the output disable request from the POEG. The release of the output disable state is performed at end of cycle by terminating the output disable request. It is after 3 PCLKD at shortest when the output disable condition is released after the output disable request becomes no longer satisfied. To reliably control output disabling, clear the flag of POEG for which the condition for the request to disable the output is no longer satisfied after 4 cycles of PCLKD.

When event count is performed or when the output disable state should be released immediately without waiting for end of cycle, GTIOR.OADF[1:0] should be set to 00b (for GTIOCnA pin) or GTIOR.OBDF[1:0] should be set to 00b (for the GTIOCnB pin).

Figure 21.66 shows an example of the GTIOCnm pin output disable control operation. (n = 0 to 7, m = A, B)

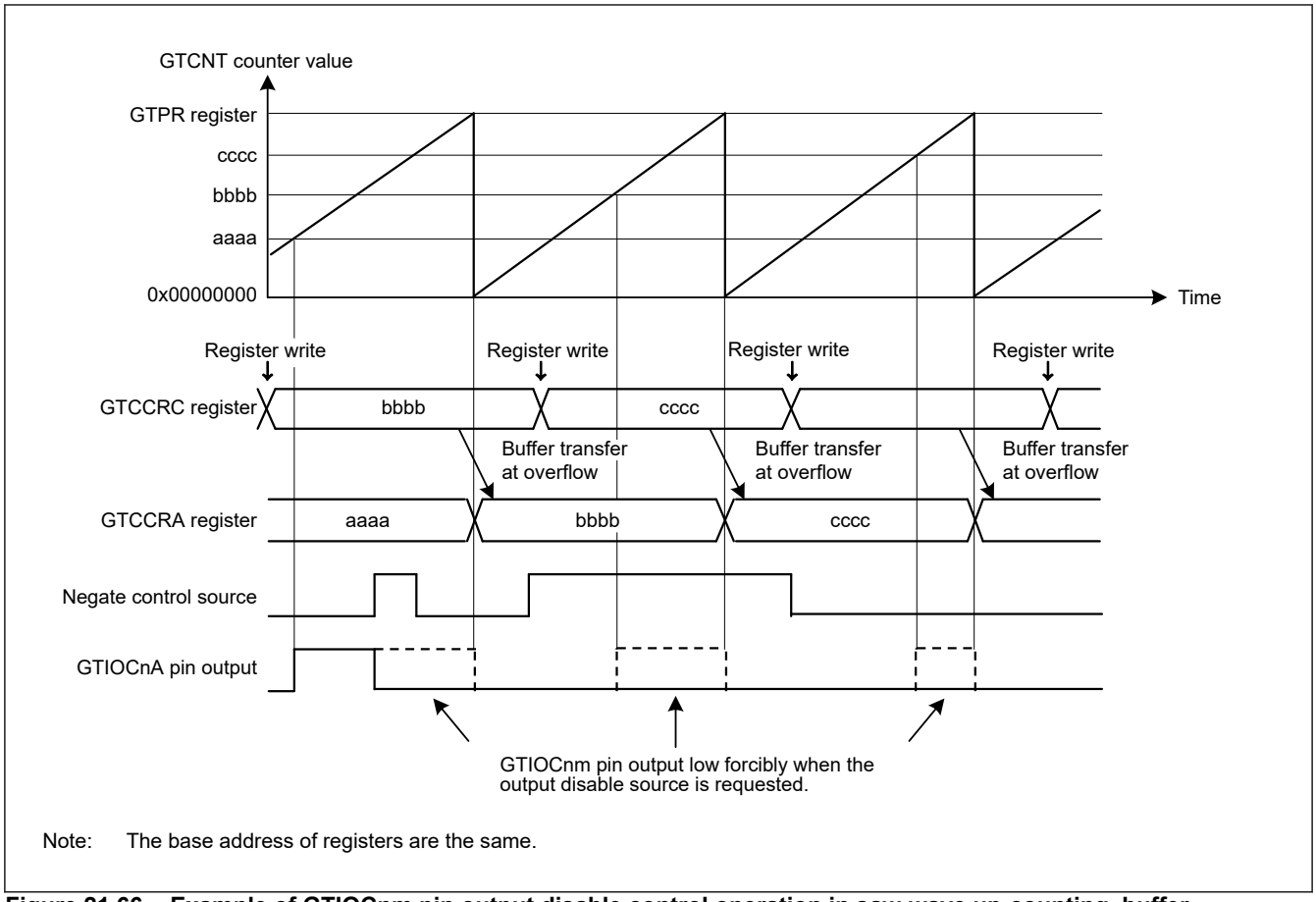
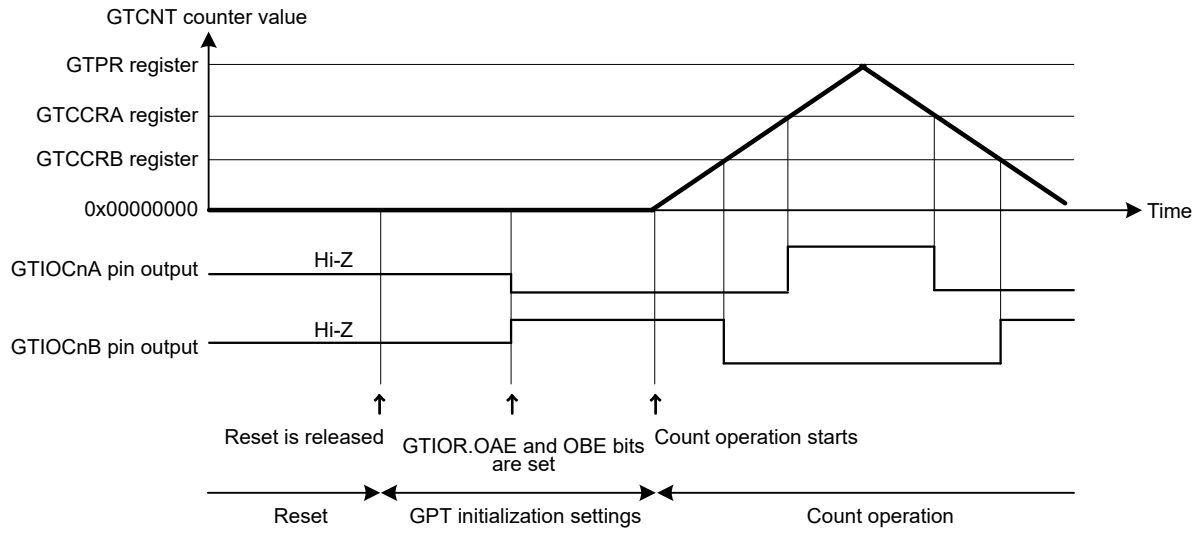



**Figure 21.66 Example of GTIOCnm pin output disable control operation in saw-wave up-counting, buffer operation, active level 1, high output at GTCCRA compare match, low output at cycle end, and low output at output disable (n = 0 to 7, m = A, B)**

## 21.8 Initialization Method of Output Pins

### 21.8.1 Pin Settings after Reset

The GPT registers are initialized at a reset. Start counting after selecting the port pin function with the PmnPFS register, setting GTIOR.OAE and GTIOR.OBE bits, and outputting the GPT function to external pins.




[Setting examples]
GTIOR.GTIOA[4:0] bits: Initial low output, output retained at cycle end, output toggled at compare match
GTIOR.GTIOB[4:0] bits: Initial high output, output retained at cycle end, output toggled at compare match

Note: The base address of registers are the same.

**Figure 21.67 Example of pin settings after reset**

### 21.8.2 Pin Initialization Due to Error during Operation

If an error occurs during GPT operation, the following four types of pin control can be performed before pin initialization:

- Set the OAHLD and OBHLD bits in GTIOR to 1 and retain the outputs at count stop
- Set the OAHLD and OBHLD bits in GTIOR to 0, specify arbitrary output values at OADFLT and OBDFLT in GTIOR, and output the arbitrary values at count stop
- Set the pin to output an arbitrary value as a general output port by setting the PDR, PODR registers and PmnPFS.PMR bit of the I/O port in advance. Set the OAE and OBE bits in GTIOR to 0, and the control bit associated with the pin in the PMR to 0 to allow arbitrary values to be output from the pin set as a general output port when an error occurs.
- Drive the output to a high impedance state using the POEG function.

If the automatic dead time setting is made, clear the GTDTCR.TDE bit to 0 after counting stops. When counting stops, only the values of registers that are changed by a GPT external source change. If counting is resumed, operation continues from where it stopped. If counting is stopped, the registers must be initialized before counting starts.

## 21.9 Usage Notes

### 21.9.1 Module-Stop Function Setting

The Module Stop Control Register can enable or disable GPT operation. The GPT is initially stopped after a reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

### 21.9.2 GTCCRn Settings during Compare Match Operation (n = A to F)

#### (1) When automatic dead time setting is made in triangle-wave PWM mode

The GTCCRA register must satisfy all of the following conditions:

- GTDVU < GTCCRA
- 0 < GTCCRA < GTPR

#### (2) When automatic dead time setting is not made in triangle-wave PWM mode

The GTCCRA register must be set within the range of 0 < GTCCRA < GTPR. If GTCCRA = 0 or GTCCRA = GTPR is set, a compare match occurs within the cycle only when GTCCRA = 0 or GTCCRA = GTPR is satisfied. When GTCCRA > GTPR, no compare match occurs.

Similarly, GTCCRB must be set within the range of 0 < GTCCRB < GTPR. If GTCCRB = 0 or GTCCRB = GTPR is set, a compare match occurs within the cycle only when GTCCRB is 0 or GTCCRB = GTPR is satisfied. When GTCCRB > GTPR, no compare match occurs.

#### (3) When automatic dead time setting is made in saw-wave one-shot pulse mode

The GTCCRC and GTCCRD registers must be set to satisfy the following restrictions. If the restrictions are not satisfied, the correct output waveforms with secured dead time may not be obtained.

- In up-counting: GTCCRC < GTCCRD, GTCCRC > GTDVU, GTCCRD < GTPR - GTDVU
- In down-counting: GTCCRC > GTCCRD, GTCCRC < GTPR - GTDVU, GTCCRD > GTDVU

#### (4) When automatic dead time setting is not made in saw-wave one-shot pulse mode

The GTCCRC and GTCCRD registers must be set to satisfy the following restrictions. If the restrictions are not satisfied, two compare matches do not occur and pulse output cannot be performed.

- In up-counting: 0 < GTCCRC < GTCCRD < GTPR
- In down-counting: GTPR > GTCCRC > GTCCRD > 0

Similarly, GTCCRE and GTCCRF must be set to satisfy the following restrictions. If the restrictions are not satisfied, two compare matches do not occur and pulse output cannot be performed.

- In up-counting: 0 < GTCCRE < GTCCRF < GTPR
- In down-counting: GTPR > GTCCRE > GTCCRF > 0.

#### (5) In saw-wave PWM mode

The GTCCRA register must be set with the range of 0 < GTCCRA < GTPR. If GTCCRA = 0 or GTCCRA = GTPR is set, a compare match occurs within the cycle only when GTCCRA = 0 or GTCCRA = GTPR is satisfied. If GTCCRA > GTPR is set, no compare match occurs.

Similarly, GTCCRB must be set with the range of 0 < GTCCRB < GTPR. If GTCCRB = 0 or GTCCRB = GTPR is set, a compare match occurs within the cycle only when GTCCRB = 0 or GTCCRB = GTPR is satisfied. If GTCCRB > GTPR is set, no compare match occurs.

### 21.9.3 Setting Range for GTCNT Counter

The GTCNT counter register must be set with the range of 0 ≤ GTCNT ≤ GTPR.

### 21.9.4 Starting and Stopping the GTCNT Counter

The control timing of starting and stopping the GTCNT counter by the GTCR.CST bit synchronizes the count clock that is selected in GTCR.TPCS[3:0]. When GTCR.CST is updated, the GTCNT counter starts/stops after a count clock that is selected in GTCR.TPCS[3:0]. Therefore, an event generated before the GTCNT counter actually starts is ignored, resulting in situations in which an event is accepted or an interrupt occurs after GTCR.CST is set to 0.

### 21.9.5 Priority Order of Each Event

#### (1) GTCNT register

Table 21.47 shows a priority order of events updating the GTCNT register.

**Table 21.47 Priority order of sources updating GTCNT**

| Source updating GTCNT | Priority order |
|---|---|
| Writing by CPU (writing to GTCNT/GTCLR) | High |
| Clear by hardware sources set in GTCSR | ↑ |
| Count up or down by hardware sources set in GTUPSR/GTDNSR | ↑ |
| Count operation | Low |

If up-counting and down-counting by hardware sources occur at the same time, the GTCNT counter value does not change. When there is a conflict between updating the GTCNT register and reading by the CPU, pre-update data is read.

### (2) GTCR.CST bit

When there is a conflict between starting/stopping by hardware sources set in the GTSSR/GTPSR registers and writing by the CPU (writing to GTCR/GTSTR/GTSTP registers), the writing by CPU has priority over the starting/stopping by hardware sources.

In case that stop by the period count function conflicts with start by the CPU writing (GTCR register writing/GTSTR register writing), the period count function is finished with setting the GTST.PCF flag. The CST bit is not changed and the GTCNT continues to count.

When there is a conflict between starting by hardware sources set in the GTSSR register and stopping by hardware sources set in GTPSR register, the GTCR.CST bit value does not change. When there is a conflict between updating the GTCR.CST bit and reading by the CPU (reading from GTCR/GTSTR/GTSTP registers), pre-update data is read.

### (3) GTCCRm registers (m = A to F)

When there is a conflict between input capture/buffer transfer operation and writing to the GTCCRm registers, the writing to GTCCRm registers has priority over input capture/buffer transfer operation. When there is a conflict between input capture and writing to the counter register by the CPU or updating the counter register by hardware sources, the pre-update counter value is captured. When there is a conflict between updating the GTCCRm registers and reading by the CPU, pre-update data is read.

### (4) GTPR register

When there is a conflict between buffer transfer operation and writing to the GTPR register, writing to GTPR register has priority over buffer transfer operation. When there is a conflict between updating GTPR register and reading by the CPU, pre-update data is read.

# 22. Low Power Asynchronous General Purpose Timer (AGT)

## 22.1 Overview

The Low Power Asynchronous General Purpose Timer (AGT) is a 16-bit timer that can be used for pulse output, external pulse width or period measurement, and counting external events. This timer consists of a reload register and a down counter. The reload register and the down counter are allocated to the same address, and can be accessed with the AGT register.

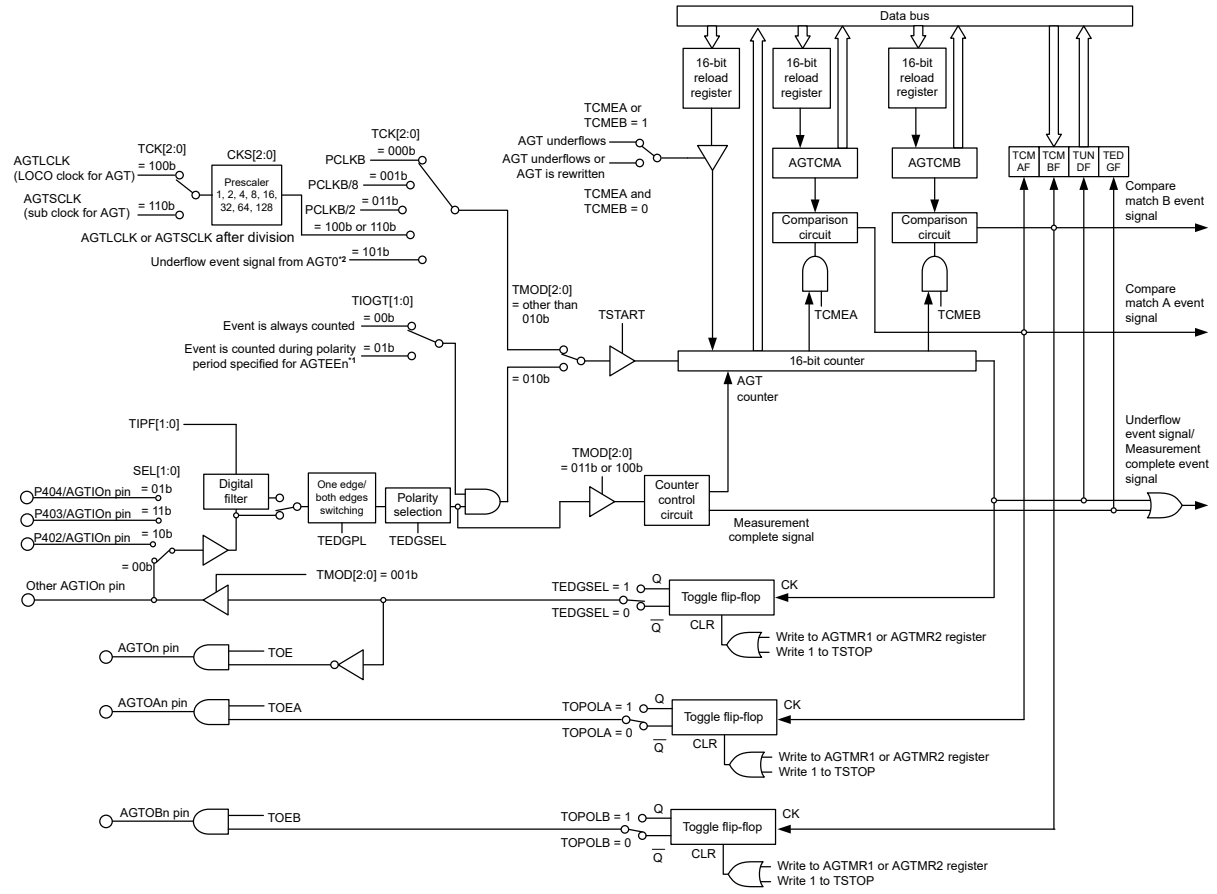
Table 22.1 lists the AGT specifications, Figure 22.1 shows a block diagram, and Table 22.2 lists the I/O pins.

The n of AGTOAn and AGTOBn in this document is defined as n = 0 to 3, 5.

**Table 22.1 AGT specifications**

| Parameter | | Description |
|---|---|---|
| Operating modes | Timer mode | The count source is counted |
| | Pulse output mode | The count source is counted and the output is inverted at each timer underflow |
| | Event counter mode | An external event is counted |
| | Pulse width measurement mode | An external pulse width is measured |
| | Pulse period measurement mode | An external pulse period is measured |
| Number of Channels | | 16 bits × 6 channels (AGTn (n = 0 to 5)) |
| Count source (operating clock)*2 | Timer mode | PCLKB, PCLKB/2, PCLKB/8, AGTLCLK/d, AGTSCLK/d (d = 1, 2, 4, 8, 16, 32, 64, or 128), or underflow signal of AGTn (n = 0, 2, 4) selectable.*1 |
| | Pulse output mode | |
| | Pulse width measurement mode | |
| | Pulse period measurement mode | |
| | Event counting mode | External event input |
| Interrupt and Event Link function | | • Underflow event signal or measurement complete event signal<br>– When the counter underflows<br>– When the measurement of the active width of the external input pin (AGTIOn) completes in pulse width measurement mode<br>– When the set edge of the external input pin (AGTIOn) is input in pulse period measurement mode.<br>• Compare match A event signal<br>– When the values of AGT register and AGTCMA register matched (compare match A function enabled).<br>• Compare match B event signal<br>– When the values of AGT and AGTCMB registers matched (compare match B function enabled).<br>• Return from Snooze mode or Software Standby mode can be performed with AGTn_AGTI, AGTn_AGTCMAI, or AGTn_AGTCMBI (n = 1, 3)*3 |
| Selectable functions | | • Compare match function<br>One or two of the AGT Compare Match A register and AGT Compare Match B register is selectable. |
| TrustZone Filter | | Security attribution can be set for each channels |

Note 1. AGTn (n = 0, 2, 4) cannot use underflow signal. AGTn (n = 1, 3, 5) connects directly with the underflow event signal from the AGTn (n = 0, 2, 4) timer.
Note 2. Satisfy the frequency of the peripheral module clock (PCLKB) ≥ the frequency of the count source clock.
Note 3. For details, see section 10, Low Power Modes.




Note 1. The polarity can be selected by the EEPS bit in the AGTISR register.
Note 2. AGTn (n = 0, 2, 4) cannot use AGTn (n = 0, 2, 4) underflow event. AGTn (n = 1, 3, 5) uses the underflow of AGTn (n = 0, 2, 4).

**Figure 22.1 AGT block diagram**

**Table 22.2 AGT I/O pins**

| Pin name | I/O | Function |
|---|---|---|
| AGTEEn | Input | External event input enable for AGT |
| AGTIOn | Input/output | External event input and pulse output for AGT<br>AGTIOm (m=0 to 3) from P402, P403, P404 can be controlled by the VBTICTRLR register.<br>For details, see section 19.5.5. I/O Buffer Specification. |
| AGTOn | Output | Pulse output for AGT |
| AGTOAn | Output | Compare match A output for AGT |
| AGTOBn | Output | Compare match B output for AGT |

Note: Channel number: n = 0 to 5
Note: P402, P403, P404 can only be used as input.

## 22.2 Register Descriptions

### 22.2.1 AGT : AGT Counter Register

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x00

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | 16-bit counter and reload register<br>Setting range : 0x0000 to 0xFFFF | R/W |

AGTn.AGT is a 16-bit register. The write value is written to the reload register and the read value is read from the counter.

The states of the reload register and the counter change according to the TSTART bit in the AGTCR register and TCMEA/TCMEB bit in the AGTCMSR register. For details, see section 22.3.1. Reload Register and Counter Rewrite Operation.

When 1 is written to the TSTOP bit in the AGTCR register, AGT counter is forcibly stopped and set to 0xFFFF.

When the TCK[2:0] bits setting in the AGTMR1 register are a value other than 001b (PCLKB/8) or 011b (PCLKB/2), if the AGT register is set to 0x0000, a request signal to the ICU, the DTC, the DMAC, and the ELC is generated once immediately after the count starts. The AGTOn, AGTIOn pin output are toggled.

When the AGT register is set to 0x0000 in event counter mode, regardless of the value of TCK[2:0] bits, a request signal to the ICU, the DTC, the DMAC, and the ELC is generated once immediately after the count starts.

In addition, the AGTOn pin output is toggled even during a period other than the specified count period. When the AGT register is set to 0x0001 or more, a request signal is generated each time AGT underflows.

### 22.2.2 AGTCMA : AGT Compare Match A Register

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x02

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | 16-bit compare match A data is stored.*1<br>Setting range : 0x0000 to 0xFFFF | R/W |

Note 1. Set the AGTCMA register to 0xFFFF when compare match A is not used.

The AGTCMA register is a read/write register to set a value for compare match with the AGT counter. The states of the reload register and compare register A change according to the TSTART bit in the AGTCR register. For details, see section 22.3.2. Reload Register and AGT Compare Match A/B Register Rewrite Operation.

### 22.2.3 AGTCMB : AGT Compare Match B Register

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x04

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | 16-bit compare match B data is stored.*1<br>Setting range : 0x0000 to 0xFFFF | R/W |

Note 1. Set the AGTCMB register to 0xFFFF when compare match B is not used.

The AGTCMB register is a read/write register to set a value for compare match with the AGT counter. The states of the reload register and compare register B change according to the TSTART bit in the AGTCR register. For details, see section 22.3.2. Reload Register and AGT Compare Match A/B Register Rewrite Operation.

### 22.2.4 AGTCR : AGT Control Register

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x08

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TCMBF | TCMAF | TUNDF | TEDGF | — | TSTOP | TCSTF | TSTART |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TSTART | AGT Count Start*2<br>0: Count stops<br>1: Count starts | R/W |
| 1 | TCSTF | AGT Count Status Flag*2<br>0: Count stopped<br>1: Count in progress | R |
| 2 | TSTOP | AGT Count Forced Stop*1<br>0: Writing is invalid<br>1: The count is forcibly stopped | W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | TEDGF | Active Edge Judgment Flag<br>0: No active edge received<br>1: Active edge received | R/(W)*3 |
| 5 | TUNDF | Underflow Flag<br>0: No underflow<br>1: Underflow | R/(W)*3 |
| 6 | TCMAF | Compare Match A Flag<br>0: No match<br>1: Match | R/(W)*3 |
| 7 | TCMBF | Compare Match B Flag<br>0: No match<br>1: Match | R/(W)*3 |

Note 1. When 1 (count is forcibly stopped) is written to the TSTOP bit, the TSTART bit and TCSTF flag are initialized at the same time. The pulse output level is also initialized. The read value is 0.
Note 2. For information on using the TSTART bit and TCSTF flag, see section 22.4.1. Count Operation Start and Stop Control.
Note 3. Only 0 can be written to clear the flag.

### TSTART bit (AGT Count Start)

The count operation is started by writing 1 to the TSTART bit and stopped by writing 0. When the TSTART bit is set to 1 (count starts), the TCSTF flag is set to 1 (count in progress) in synchronization with the count source. Also, after 0 is written to the TSTART bit, the TCSTF flag is set to 0 (count stops) in synchronization with the count source. For details, see section 22.4.1. Count Operation Start and Stop Control.

### TCSTF flag (AGT Count Status Flag)

The TCSTF flag indicates the AGT count status.

[Setting condition]

- When 1 is written to the TSTART bit (the TCSTF flag is set to 1 in synchronization with the count source).

[Clearing conditions]

- When 0 is written to the TSTART bit (the TCSTF flag is set to 0 in synchronization with the count source)
- When 1 is written to the TSTOP bit.

### TSTOP bit (AGT Count Forced Stop)

When 1 is written to the TSTOP bit, the count is forcibly stopped. The read value is 0.

### TEDGF flag (Active Edge Judgment Flag)

The TEDGF flag indicates that an active edge was detected.

[Setting condition]

- When the measurement of the active width of the external input pin (AGTIOn) is complete in pulse width measurement mode
- When the set edge of the external input pin (AGTIOn) is input in pulse period measurement mode.

[Clearing condition]

- When 0 is written to this flag by software.

### TUNDF flag (Underflow Flag)

The TUNDF flag indicates that the counter underflowed.

[Setting condition]

- When the counter underflows.

[Clearing condition]

- When 0 is written to this flag by software.

### TCMAF flag (Compare Match A Flag)

The TCMAF flag indicates that compare match A was detected.

[Setting condition]

- When the value in the AGT register matches the value in the AGTCMA register.

[Clearing condition]

- When 0 is written to this flag by software.

### TCMBF flag (Compare Match B Flag)

The TCMBF flag indicates that compare match B was detected.

[Setting condition]

- When the value in the AGT register matches the value in the AGTCMB register.

[Clearing condition]

- When 0 is written to this flag by software.

## 22.2.5 AGTMR1 : AGT Mode Register 1

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x09

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | TCK[2:0] | | | TEDGPL | TMOD[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | TMOD[2:0] | Operating Mode*3<br>0 0 0: Timer mode<br>0 0 1: Pulse output mode<br>0 1 0: Event counter mode<br>0 1 1: Pulse width measurement mode<br>1 0 0: Pulse period measurement mode<br>Others: Setting prohibited | R/W |
| 3 | TEDGPL | Edge Polarity*4<br>0: Single-edge<br>1: Both-edge | R/W |
| 6:4 | TCK[2:0] | Count Source*1 *2 *5 *7<br>0 0 0: PCLKB<br>0 0 1: PCLKB/8<br>0 1 1: PCLKB/2<br>1 0 0: Divided clock AGTLCLK specified by CKS[2:0] bits in the AGTMR2 register<br>1 0 1: Underflow event signal from AGTn (n = 0, 2, 4)*6<br>1 1 0: Divided clock AGTSCLK specified by CKS[2:0] bits in the AGTMR2 register<br>Others: Setting prohibited | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |

Note: Write access to the AGTMR1 register initializes the output from the AGTOn, AGTIOn, AGTOAn, and AGTOBn pins. For details on the output level at initialization, see section 22.2.7. AGTIOC : AGT I/O Control Register.

Note 1. When event counter mode is selected, the external input pin (AGTIOn) is selected as the count source regardless of the setting of TCK[2:0] bits.

Note 2. Do not switch count sources during count operation. Only switch count sources when both the TSTART bit and TCSTF flag in the AGTCR register are set to 0 (count stops).

Note 3. The operating mode can only be changed when the count is stopped while both the TSTART bit and TCSTF flag in the AGTCR register are set to 0 (count is stopped). Do not change the operating mode during count operation.

Note 4. The TEDGPL bit is enabled only in event counter mode.

Note 5. To run AGT in Software Standby mode, Snooze mode, or Deep Software Standby mode, select AGTLCLK or AGTSCLK (TCK[2:0] = 100b, 110b).

Note 6. AGTn (n = 0, 2, 4) cannot use AGTn (n = 0, 2, 4) underflow (setting prohibited). AGTn (n = 1, 3, 5) uses the AGTn (n = 0, 2, 4) underflow.

Note 7. Do not change the TCK[2:0] bits when the CKS[2:0] bits in the AGTMR2 register is not 000b. First, change the CKS[2:0] bits in the AGTMR2 register to 000b. Then change the TCK[2:0] bits and wait for one cycle of the count source.

## 22.2.6 AGTMR2 : AGT Mode Register 2

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x0A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | LPM | — | — | — | — | CKS[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | CKS[2:0] | AGTLCLK or AGTSCLK Count Source Clock Frequency Division Ratio*1 *2 *3<br>0 0 0: 1/1<br>0 0 1: 1/2<br>0 1 0: 1/4<br>0 1 1: 1/8<br>1 0 0: 1/16<br>1 0 1: 1/32<br>1 1 0: 1/64<br>1 1 1: 1/128 | R/W |
| 6:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | LPM | Low Power Mode<br>0: Normal mode<br>1: Low power mode | R/W |

Note 1. Do not rewrite the CKS[2:0] bits during count operation. Only rewrite the CKS[2:0] bits when both the TSTART bit and TCSTF flag in the AGTCR register are set to 0 (count stops).
Note 2. When count source is AGTLCLK or AGTSCLK, the switch of CKS[2:0] bits is valid.
Note 3. Do not switch the TCK[2:0] bits in the AGTMR1 register when CKS[2:0] bits are not 000b. Switch the TCK[2:0] bits in the AGTMR1 register after CKS[2:0] bits are set to 000b, and wait for 1 cycle of the count source.

### CKS[2:0] bit (AGTLCLK or AGTSCLK Count Source Clock Frequency Division Ratio)

CKS[2:0] bits select the Count Source Clock Frequency Division Ratio for AGTLCLK or AGTSCLK.

### LPM bit (Low Power Mode)

The LPM bit sets the low power operation, which impacts access to certain AGT registers. Set this bit to 1 to operate in low power.

When this bit is 1, access to the following registers is prohibited:

- AGT/AGTCMA/AGTCMB/AGTCR.

After this bit is switched from 1 to 0, the first access to the register is constrained as follows:

- When reading from the AGT register, read AGT register twice. Only the second reading of data is valid.
- When writing to the AGT, AGTCMA, AGTCMB, and AGTCR register, allow at least 2 cycles of the count source clock when writing to the register.
- When confirm the value written to the AGT, AGTCMA, AGTCMB, and AGTCR registers.
  - When the count operation is stopped; after writing data, it can be read in the next cycle.
  - When the count operation is operating; after writing data, it can be read 4 cycles after the count source clock.

Figure 22.2 shows the flow of how to write LPM bit

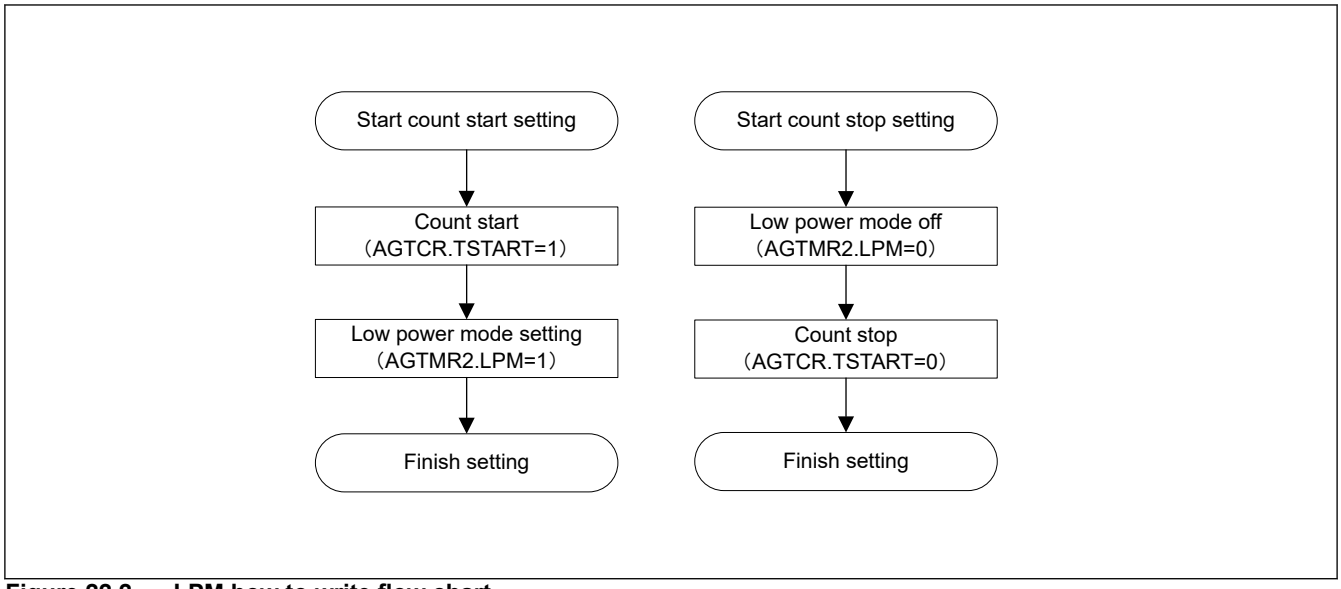



**Figure 22.2 LPM how to write flow chart**

## 22.2.7 AGTIOC : AGT I/O Control Register

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x0C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TIOGT[1:0] | | TIPF[1:0] | | — | TOE | — | TEDG SEL |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TEDGSEL | I/O Polarity Switch<br>Function varies depending on the operating mode (see Table 22.3 and Table 22.4). | R/W |
| 1 | — | This bit is read as 0. The write value should be 0. | R/W |
| 2 | TOE | AGTOn pin Output Enable<br>0: AGTOn pin output disabled<br>1: AGTOn pin output enabled | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 5:4 | TIPF[1:0] | Input Filter*3<br>These bits specifies the sampling frequency of the filter for the AGTIOn input. If the input to the AGTIOn pin is sampled and the value matches three successive times, that value is taken as the input value.<br>0 0: No filter<br>0 1: Filter sampled at PCLKB<br>1 0: Filter sampled at PCLKB/8<br>1 1: Filter sampled at PCLKB/32 | R/W |
| 7:6 | TIOGT[1:0] | Count Control*1 *2<br>0 0: Event is always counted<br>0 1: Event is counted during polarity period specified for AGTEEn pin<br>Others: Setting prohibited | R/W |

Note 1. When AGTEEn pin is used, the polarity to count an event can be selected with the EEPS bit in the AGTISR register.
Note 2. TIOGT[1:0] bits are enabled only in event counter mode.
Note 3. When event counter mode operation is performed during Software Standby and Deep Software Standby mode, the digital filter function cannot be used.

### TEDGSEL bit (I/O Polarity Switch)

The TEDGSEL bit switches the AGTOn pin output polarity and the AGTIOn pin input/output edge and polarity.

In pulse output mode, it only controls polarity of the AGTOn pin output and AGTIOn pin output. AGTOn pin output and AGTIOn pin output are initialized when the AGTMR1 register is written or the TSTOP bit in the AGTCR register is written with 1.

### TOE bit (AGTOn pin Output Enable)

The TOE bit selects whether the AGTOn pin output is disabled or enabled.

### TIPF[1:0] bits (Input Filter)

The TIPF[1:0] bits specify the sampling frequency of the AGTIOn pin input filter. When the input to the AGTIOn pin is sampled and the values match three times in succession, the value is regarded as the input value.

### TIOGT[1:0] bits (Count Control)

The TIOGT[1:0] bits control the event count.

**Table 22.3 AGTIOn pin I/O edge and polarity switching**

| Operating mode | Function |
|---|---|
| Timer mode | Not used |
| Pulse output mode | 0: Output is started at high (initialization level: high) i.e. inverted output<br>1: Output is started at low (initialization level: low). i.e. normal output |
| Event counter mode | 0: Count on rising edge<br>1: Count on falling edge. |
| Pulse width measurement mode | 0: Low-level width is measured<br>1: High-level width is measured. |
| Pulse period measurement mode | 0: Measure from one rising edge to the next rising edge<br>1: Measure from one falling edge to the next falling edge. |

**Table 22.4 AGTOn pin output polarity switching**

| Operating mode | Function |
|---|---|
| All modes | 0: Output is started at low (initial level: low): Normal output<br>1: Output is started at high (initial level: high): Inverted output |

## 22.2.8 AGTISR : AGT Event Pin Select Register

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x0D

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | EEPS | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 2 | EEPS | AGTEEn Polarity Selection<br>0: An event is counted during the low-level period<br>1: An event is counted during the high-level period | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

### EEPS bit (AGTEEn Polarity Selection)

The EEPS bit selects the polarity of events to be counted.

## 22.2.9 AGTCMSR : AGT Compare Match Function Select Register

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x0E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | TOPOLB | TOEB | TCMEB | — | TOPOLA | TOEA | TCMEA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TCMEA | AGT Compare Match A Register Enable*1 *2<br>0: AGT Compare match A register disabled<br>1: AGT Compare match A register enabled | R/W |
| 1 | TOEA | AGTOAn Pin Output Enable*1 *2<br>0: AGTOAn pin output disabled<br>1: AGTOAn pin output enabled | R/W |
| 2 | TOPOLA | AGTOAn Pin Polarity Select*1 *2<br>0: AGTOAn pin output is started on low. i.e. normal output<br>1: AGTOAn pin output is started on high. i.e. inverted output | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | TCMEB | AGT Compare Match B Register Enable*1 *2<br>0: Compare match B register disabled<br>1: Compare match B register enabled | R/W |
| 5 | TOEB | AGTOBn Pin Output Enable*1 *2<br>0: AGTOBn pin output disabled<br>1: AGTOBn pin output enabled | R/W |
| 6 | TOPOLB | AGTOBn Pin Polarity Select*1 *2<br>0: AGTOBn pin output is started on low. i.e. normal output<br>1: AGTOBn pin output is started on high. i.e. inverted output | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |

Note 1. Do not rewrite the AGTCMSR register during a count operation. Only rewrite the AGTCMSR register when both the TSTART bit and TCSTF flag in the AGTCR register are set to 0 (count stops).

Note 2. Do not set 1 when in pulse width measurement mode or pulse period measurement mode.

## 22.2.10 AGTIOSEL : AGT Pin Select Register

Base address: AGTn = 0x400E_8000 + 0x0100 × n (n = 0 to 5)

Offset address: 0x0F

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | TIES | — | — | SEL[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | SEL[1:0] | AGTIOn Pin Select*1<br>0 0: Select Pm/AGTIO as AGTIO. Pm/AGTIO can not be used as AGTIO input pin in Deep Software Standby mode. (m = 100, 301, and 407 (AGT0), m = 400 (AGT1), m = 103 (AGT2), m = 600(AGT3).)<br>0 1: Select P404/AGTIO as AGTIO P404/AGTIO can be used as AGTIO input pin in Deep Software Standby mode. P404/AGTIOn is input only. It cannot be used for output.<br>1 0: Select P402/AGTIO as AGTIO P402/AGTIO can be used as AGTIO input pin in Deep Software Standby mode. P402/AGTIOn is input only. It cannot be used for output.<br>1 1: Select P403/AGTIO as AGTIO. P403/AGTIO can be used as AGTIO input pin in Deep Software Standby mode. P403/AGTIOn is input only. It cannot be used for output. | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | TIES | AGTIOn Pin Input Enable<br>0: External event input is disabled during Software Standby mode<br>1: External event input is enabled during Software Standby mode | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. You must set the Pin Function Select Register. See section 19, I/O Ports.

The AGTIOSEL register sets the AGTIOn pin when using the AGTIOn pin in Deep Software Standby mode and Software Standby mode.

#### SEL[1:0] bits (AGTIOn Pin Select)

The SEL[1:0] bits select the AGTIOn pin function.

#### TIES bit (AGTIOn Pin Input Enable)

The TIES bit enables or disables an external event input.

## 22.3 Operation

### 22.3.1 Reload Register and Counter Rewrite Operation

Regardless of the operating mode, the timing of the rewrite operation to the reload register and the counter differs depending on the value of the TSTART bit in the AGTCR register and of the TCMEA or TCMEB bit in the AGTCMSR register. When the TSTART bit is 0 (count stops), the count value is directly written to the reload register and the counter. When the TSTART bit is 1 (count starts) and the TCMEA bit and TCMEB bit are 0 (AGT compare match A/B register are invalid), the value is written to the reload register in synchronization with the count source, and then to the counter in synchronization with the next count source. When the TSTART bit is 1 (count starts) and the TCMEA bit or the TCMEB bit is 1 (AGT compare match A register or compare match B register is valid), the value is written to the reload register in synchronization with the count source, and then to the counter in synchronization with the underflow of the counter.

Figure 22.3 and Figure 22.4 show the timing of rewrite operation with TSTART bit value and TCMEA/TCMEB bit value.

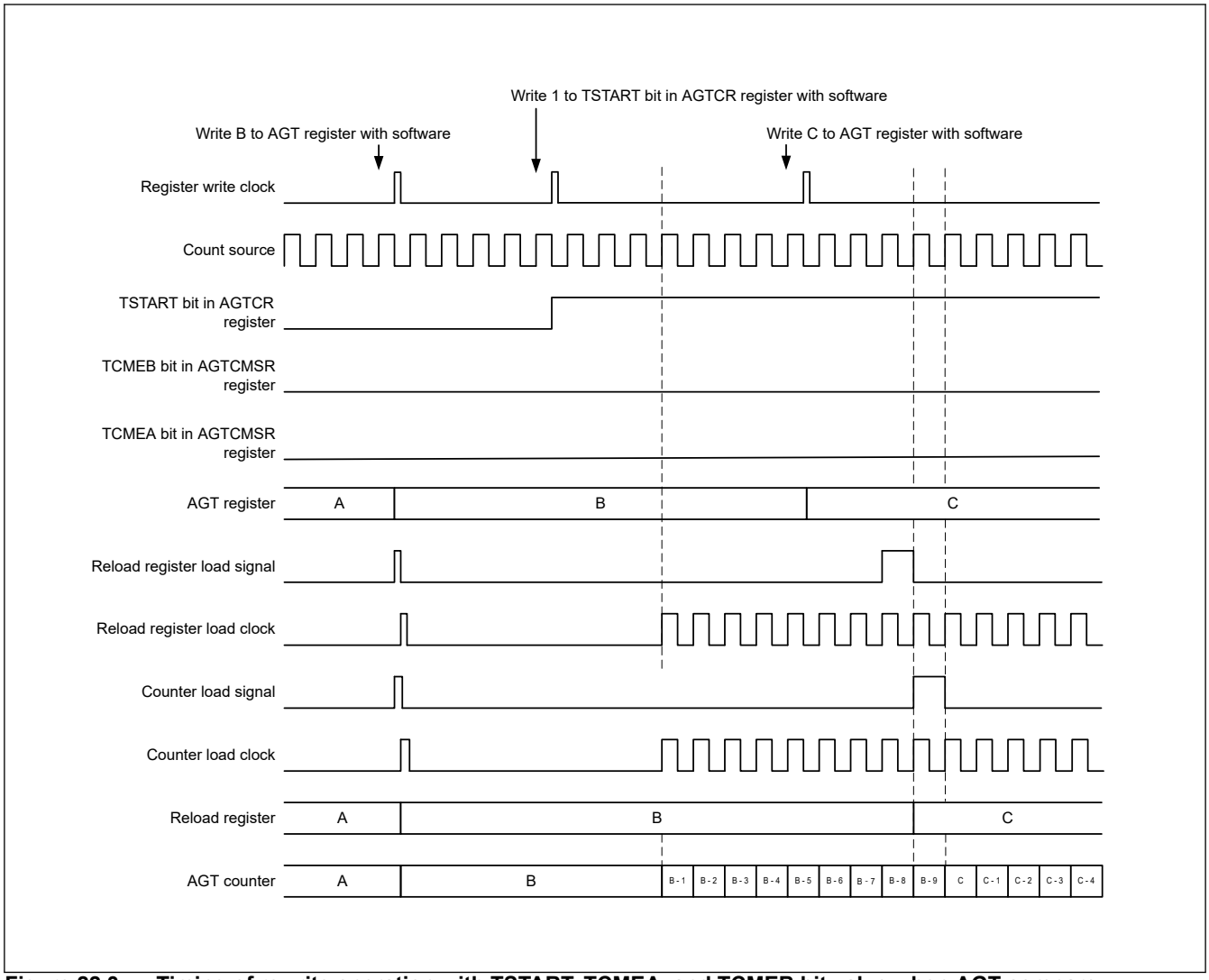



**Figure 22.3 Timing of rewrite operation with TSTART, TCMEA, and TCMEB bit value when AGT compare match A register and AGT compare match B register is invalid**

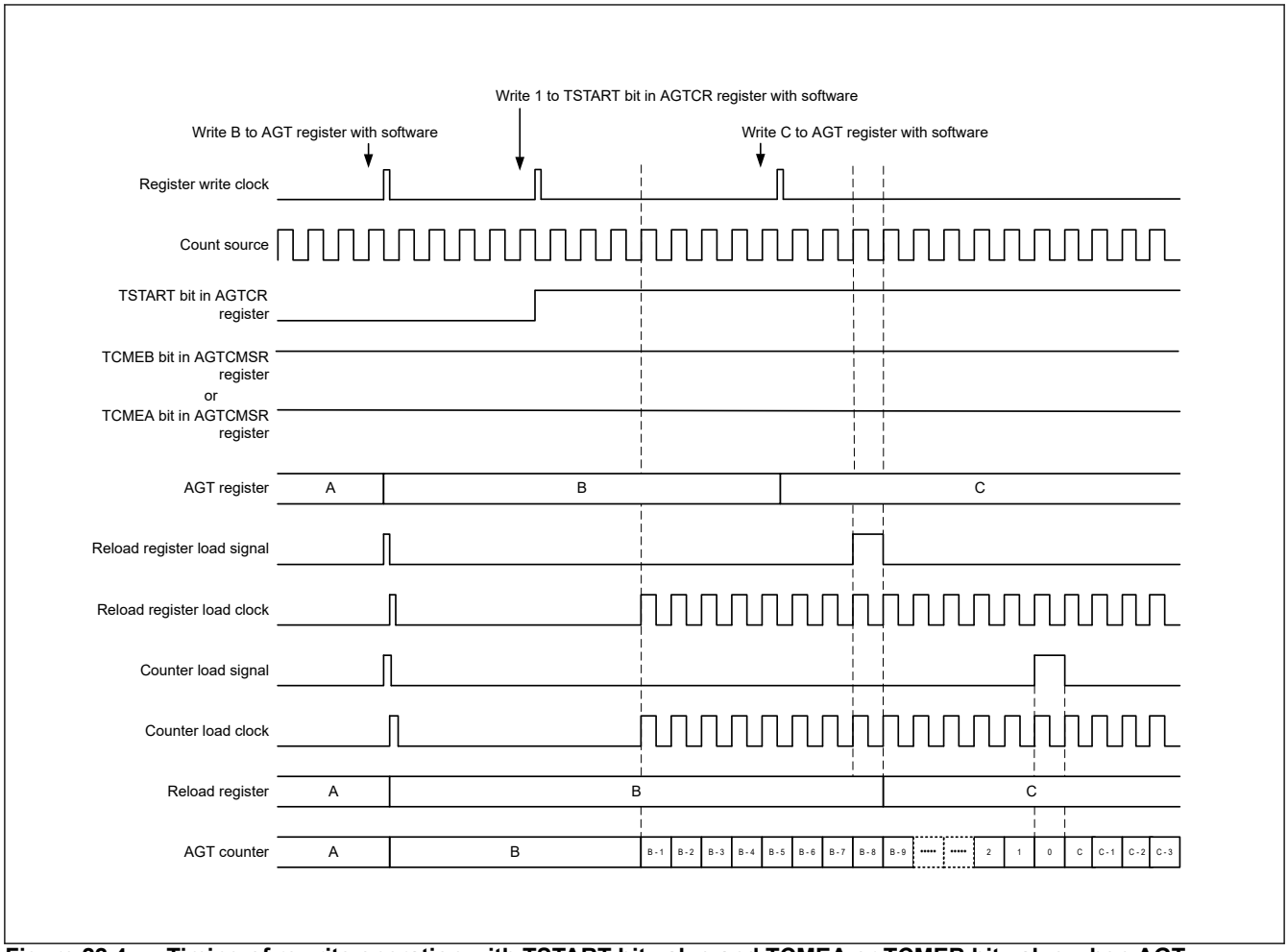



**Figure 22.4 Timing of rewrite operation with TSTART bit value and TCMEA or TCMEB bit value when AGT compare match A register or AGT compare match B register is valid**

## 22.3.2 Reload Register and AGT Compare Match A/B Register Rewrite Operation

Regardless of the operating mode, the timing of the rewrite operation to the reload register and AGT compare register A/B depends on the value of the TSTART bit in the AGTCR register. When the TSTART bit is 0 (count stops), the count value is directly written to the reload register and AGT compare register A/B. When the TSTART bit is 1 (count starts), the value is written to the reload register in synchronization with the count source, and then to the compare register in synchronization with the underflow of the counter.

Figure 22.5 shows the timing of rewrite operation with TSTART bit value for compare register A. AGT Compare register B is of the same timing as AGT compare register A.

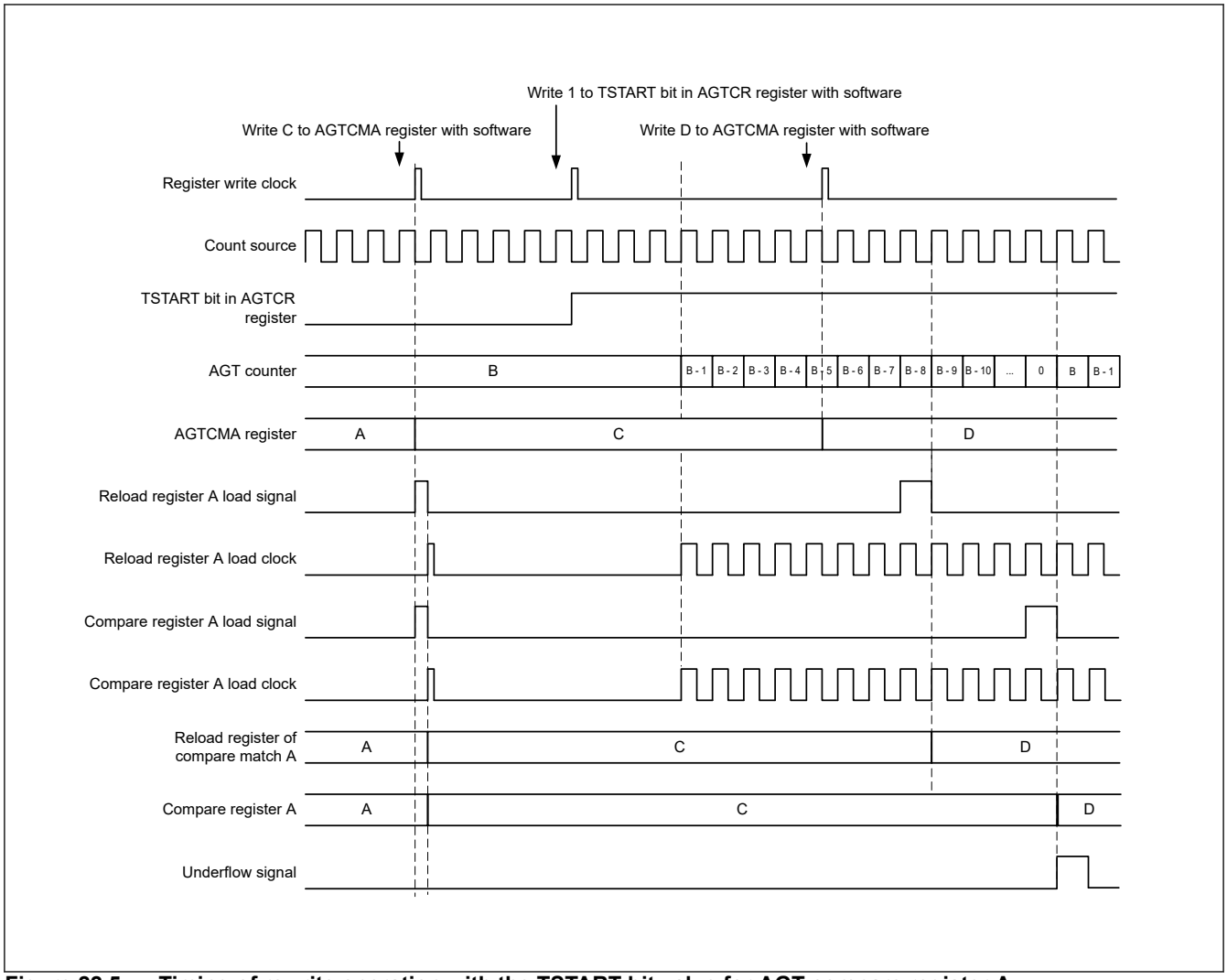



**Figure 22.5 Timing of rewrite operation with the TSTART bit value for AGT compare register A**

## 22.3.3 Timer Mode

In this mode, the AGT counter is decremented by the count source selected with the TCK[2:0] bits in the AGTMR1 register. In timer mode, the count value is decremented by 1 on each rising edge of the count source. When the count value reaches 0x0000 and the next count source is input, an underflow occurs and an interrupt request is generated.

Figure 22.6 shows the operation example in timer mode.

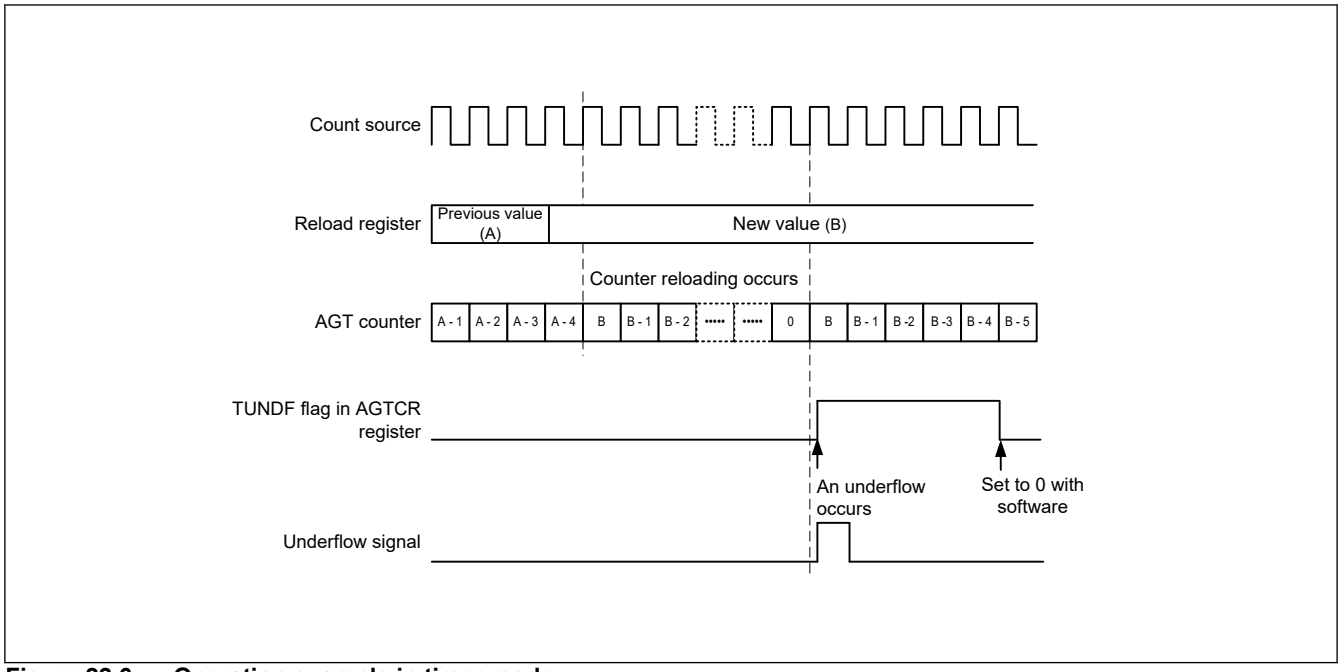



**Figure 22.6 Operation example in timer mode**

## 22.3.4 Pulse Output Mode

In pulse output mode, the counter is decremented by the count source selected with the TCK[2:0] bits in the AGTMR1 register, and the output level of the AGTIOn and AGTOn pins inverted each time an underflow occurs.

In pulse output mode, the count value is decremented by 1 on each rising edge of the count source. When the count value reaches 0x0000 and the next count source is input, an underflow occurs and an interrupt request is generated. In addition, a pulse can be output from the AGTIOn and AGTOn pins. The output level is inverted each time an underflow occurs. The pulse output from the AGTOn pin can be stopped with the TOE bit in the AGTIOC register. The output level can be selected with the TEDGSEL bit in the AGTIOC register.

Figure 22.7 shows the operation example in pulse output mode.

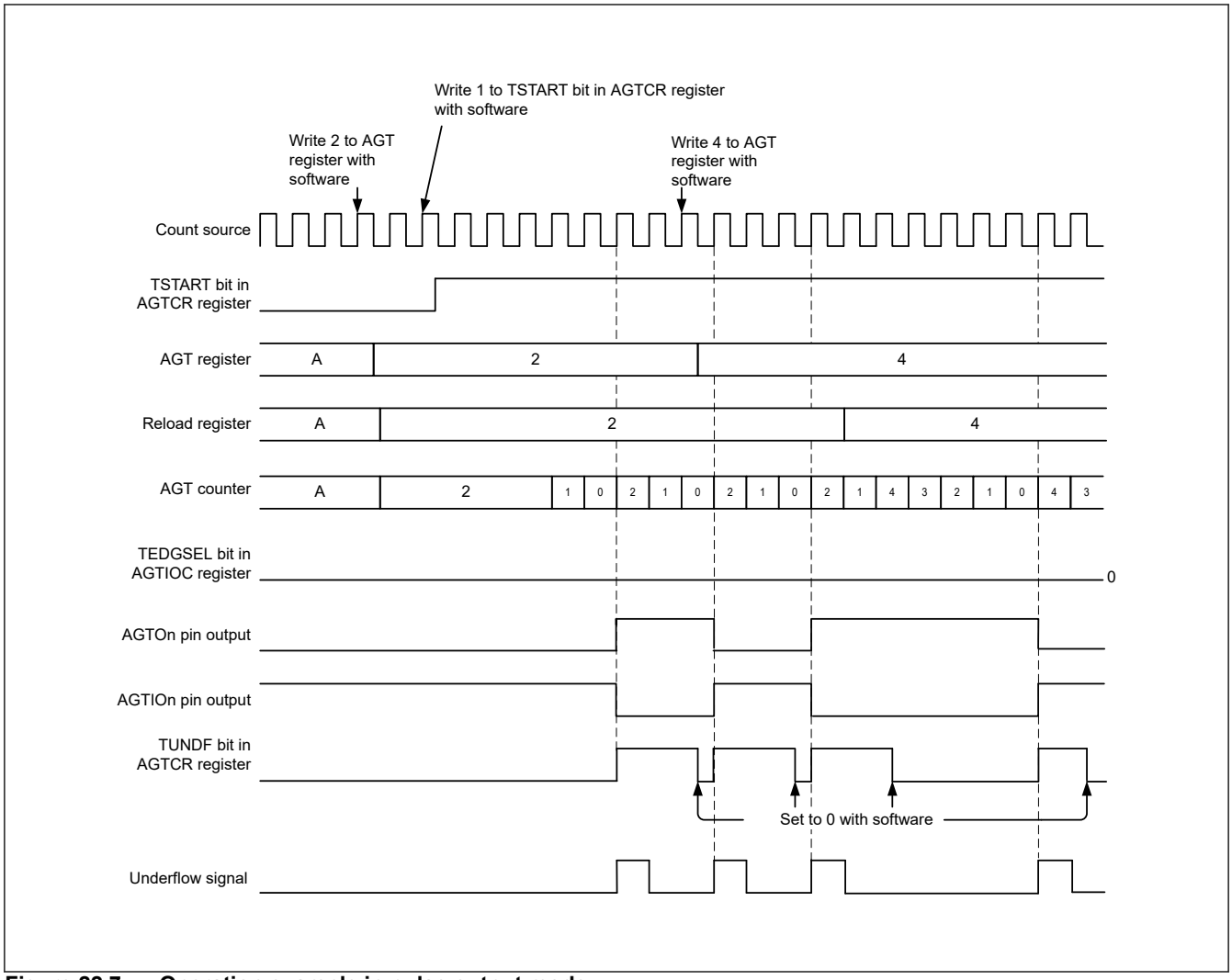



**Figure 22.7 Operation example in pulse output mode**

## 22.3.5 Event Counter Mode

In event counter mode, the counter is decremented by an external event signal (count source) input to the AGTIOn pin. Various periods for counting events can be set with the TIOGT[1:0] bits in the AGTIOC register and AGTISR registers. In addition, the filter function for the AGTIOn pin input can be specified with bits TIPF[1:0] in the AGTIOC register. The output from the AGTOn pin can be toggled even in event counter mode.

Figure 22.8 shows the operation example in event counter mode.

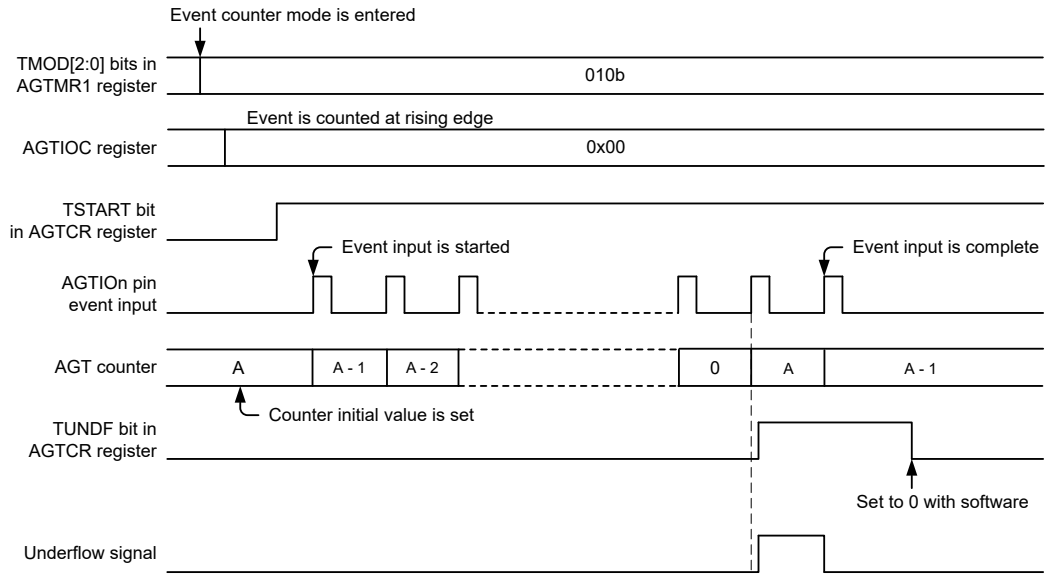
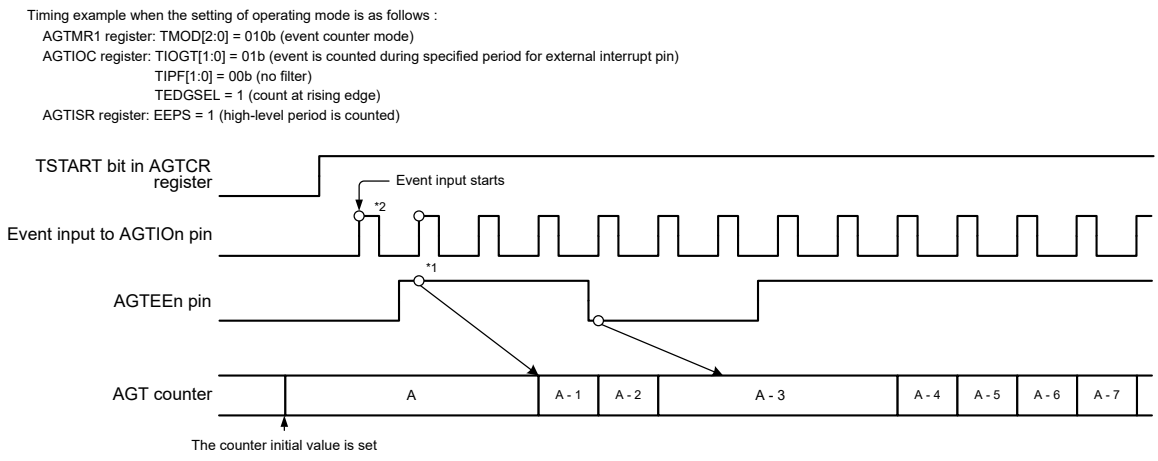



**Figure 22.8 Operation example 1 in event counter mode**

Figure 22.9 shows an operation example for counting during the specified period in event counter mode (TIOGT[1:0] bits in the AGTIOC register are set to 01b).




Note 1. To control synchronization, there is a delay of 2 cycles of the count source until the count operation is affected. It is also possible that the count start timing is shifted by 1 cycle because of the phase difference between the AGTEEn pin and the sampling clock.

Note 2. Count operation can be performed for 2 cycles of the count source immediately after the count starts, depending on the previous state before the count stops.
To disable the count for 2 cycles immediately after the count starts, write 1 to the TSTOP bit in the AGTCR register to initialize the internal circuit, and then complete the operation settings before starting the count operation.

**Figure 22.9 Operation example 2 in event counter mode**

## 22.3.6 Pulse Width Measurement Mode

In pulse width measurement mode, the pulse width of an external signal input to the AGTIOn pin is measured. When the level specified by the TEDGSEL bit in the AGTIOC register is input to the AGTIOn pin, the counter is decremented by the

count source selected with the TCK[2:0] bits in the AGTMR1 register. When the specified level on the AGTIOn pin ends, the counter is stopped, the TEDGF flag in the AGTCR register is set to 1 (active edge received), and an interrupt request is generated. The measurement of pulse width data is performed by reading the count value while the counter is stopped. Also, when the counter underflows during measurement, the TUNDF flag in the AGTCR register is set to 1 and an interrupt request is generated.

Figure 22.10 shows the operation example in pulse width measurement mode.

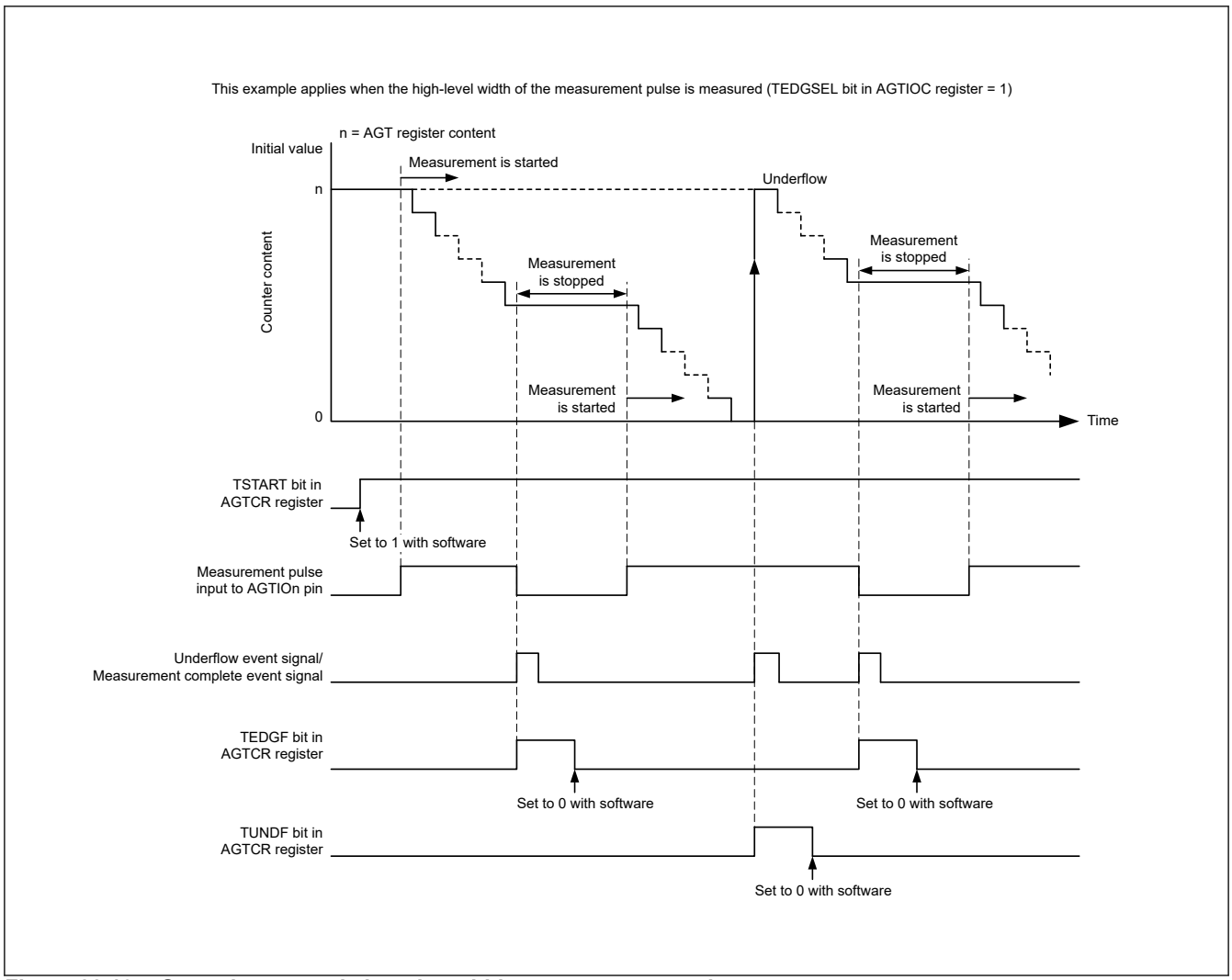


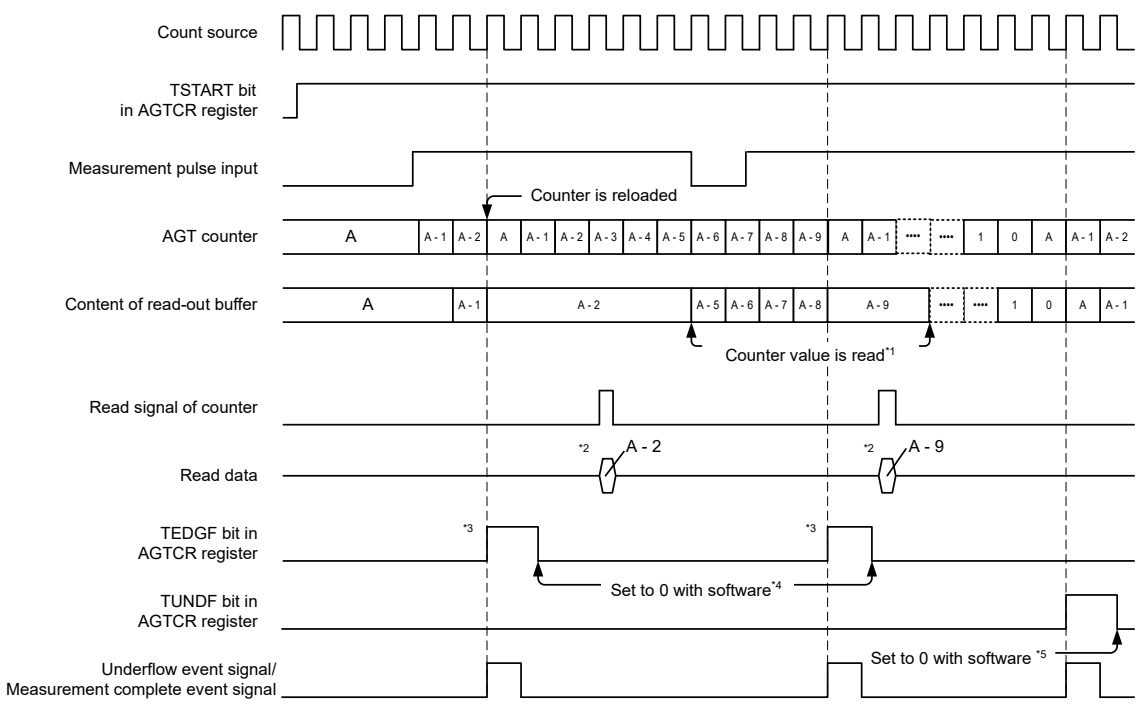

**Figure 22.10 Operation example in pulse width measurement mode**

## 22.3.7 Pulse Period Measurement Mode

In pulse period measurement mode, the pulse period of an external signal input to the AGTIOn pin is measured. The counter is decremented by the count source selected with TCK[2:0] bits in the AGTMR1 register. When a pulse with the period specified by the TEDGSEL bit in the AGTIOC register is input to the AGTIOn pin, the count value is transferred to the read-out buffer on the rising edge of the count source. The value in the reload register is loaded to the counter at the next rising edge. Simultaneously, the TEDGF flag in the AGTCR register is set to 1 (active edge received) and an interrupt request is generated. The read-out buffer (AGT register) is read at this time and the difference from the reload value (see section 22.4.6. How to Calculate Event Number, Pulse Width, and Pulse Period) is the period data of the input pulse. The period data is retained until the read-out buffer is read. When the counter underflows, the TUNDF flag in the AGTCR register is set to 1 (underflow) and an interrupt request is generated.

Figure 22.11 shows the operation example in pulse period measurement mode.

Only input pulses with a period longer than twice the period of the count source are measured. Also, the low-level and high-level widths must both be longer than the period of the count source. If a pulse period shorter than these conditions is input, the input might be ignored.




This example applies when the initial value of the AGT register is set to A, the TEDGSEL bit in the AGTIOC register is set to 0, and the period from one rising edge to the next edge of the measurement pulse is measured.

Note 1. Reading from the AGT register must be performed during the period from when the TEDGF flag is set to 1 (active edge received) until the next active edge is input. The content of the read-out buffer is retained until the AGT register is read. If it is not read before the active edge is input, the measurement result of the previous period is retained.
Note 2. When the AGT register is read in pulse period measurement mode, the content of the read-out buffer is read.
Note 3. When the active edge of the measurement pulse is input and then the set edge of an external pulse is input, the TEDGF flag in the AGTCR register is set to 1 (active edge received).
Note 4. To set to 0 with software, write 0 to the TEDGF flag in the AGTCR register with an 8-bit memory manipulation instruction.
Note 5. To set to 0 with software, write 0 to the TUNDF flag in the AGTCR register with an 8-bit memory manipulation instruction.

**Figure 22.11 Operation example in pulse period measurement mode**

## 22.3.8 Compare Match function

The compare match function detects matches (compare match) between the content of the AGTCMA or AGTCMB register and the content of the AGT register. This function is enabled when the TCMEA or TCMEB bit in the AGTCMSR register is 1 (compare match A register or compare match B register is valid). The counter is decremented by the count source selected with the TCK[2:0] bits in the AGTMR1 register, and when the values of AGT and AGTCMA or AGTCMB match, the TCMAF/TCMBF flag in the AGTCR register is set to 1 (match), and an interrupt request is generated.

When the compare match function is enabled, the timing of the rewrite operation to the reload register and the counter differs. See section 22.3.1. Reload Register and Counter Rewrite Operation for details. In addition, the output level of the AGTOAn, AGTOBn pins is inverted by the match and by the underflow. The output level can be selected with the TOPOLA or TOPOLB bit in the AGTCMSR register.

Figure 22.12 shows the operation example in compare match function.

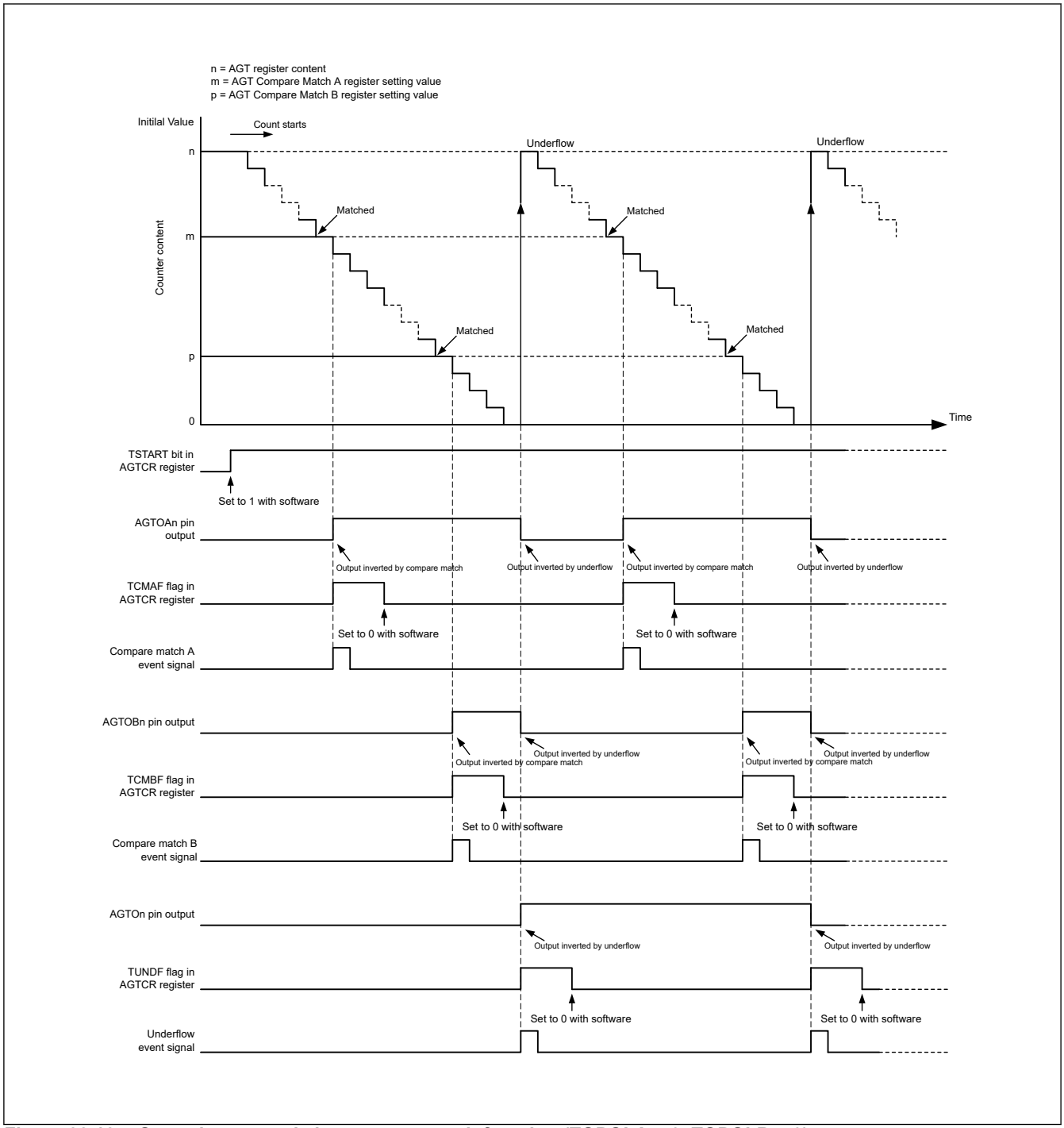



**Figure 22.12 Operation example in compare match function (TOPOLA = 0, TOPOLB = 0)**

## 22.3.9 Output Settings for Each Mode

Table 22.5 to Table 22.8 list the states of pins AGTOn, AGTIOn, AGTOAn, and AGTOBn pins in each mode.

**Table 22.5 AGTOn pin setting**

| Operating mode | AGTIOC register | | AGTOn pin output |
|---|---|---|---|
| | TOE bit | TEDGSEL bit | |
| All modes | 1 | 1 | Inverted output |
| | | 0 | Normal output |
| | 0 | 0 or 1 | Output disabled |

**Table 22.6 AGTIOn pin setting**

| Operating mode | AGTIOC register TEDGSEL bit | AGTIOn pin I/O |
|---|---|---|
| Timer mode | 0 or 1 | Input (not used) |
| Pulse output mode | 1 | Normal output |
| | 0 | Inverted output |
| Event counter mode | 0 or 1 | Input |
| Pulse width measurement mode | | |
| Pulse period measurement mode | | |

**Table 22.7 AGTOAn pin setting**

| Operating mode | AGTCMSR register | | AGTOAn pin output |
|---|---|---|---|
| | TOEA bit | TOPOLA bit | |
| Timer mode | 1 | 1 | Inverted output |
| | | 0 | Normal output |
| | 0 | 0 or 1 | Output disabled (not used) |
| Pulse output mode | 1 | 1 | Inverted output |
| | | 0 | Normal output |
| | 0 | 0 or 1 | Output disabled (not used) |
| Event counter mode | 1 | 1 | Inverted output |
| | | 0 | Normal output |
| | 0 | 0 or 1 | Output disabled (not used) |
| Pulse width measurement mode | 0 | 0 | Prohibited |
| Pulse period measurement mode | | | |

**Table 22.8 AGTOBn pin setting**

| Operating mode | AGTCMSR register | | AGTOBn pin output |
|---|---|---|---|
| | TOEB bit | TOPOLB bit | |
| Timer mode | 1 | 1 | Inverted output |
| | | 0 | Normal output |
| | 0 | 0 or 1 | Output disabled (not used) |
| Pulse output mode | 1 | 1 | Inverted output |
| | | 0 | Normal output |
| | 0 | 0 or 1 | Output disabled (not used) |
| Event counter mode | 1 | 1 | Inverted output |
| | | 0 | Normal output |
| | 0 | 0 or 1 | Output disabled (not used) |
| Pulse width measurement mode | 0 | 0 | Prohibited |
| Pulse period measurement mode | | | |

## 22.3.10 Standby Mode

The AGT can operate in Software Standby and Deep Software Standby mode. Set it to Software Standby or Deep Software Standby mode with count operation start (TSTART = 1, TCSTF = 1).

Table 22.9 and Table 22.10 show the setting that can be used in Software Standby and Deep Software Standby mode.

**Table 22.9 Usable settings in Software Standby and Deep Software Standby mode (AGTn (n = 0, 2, 4)*2)**

| Operating mode | AGTMR1.TCK[2:0] | Operating clock | Resurgence factor of CPU |
|---|---|---|---|
| Timer mode | 100b or 110b | AGTLCLK or AGTSCLK | — |
| Pulse output mode | 100b or 110b | AGTLCLK or AGTSCLK | — |
| Event counter mode | — | AGTIOn (n = 0, 2, 4)*1 | — |
| Pulse width measurement mode | 100b or 110b | AGTLCLK or AGTSCLK | — |
| Pulse period measurement mode | 100b or 110b | AGTLCLK or AGTSCLK | — |

Note: —: invalid
Note 1. When using the AGTIOn pin for external event input in Software Standby mode, set AGTIOSEL.TIES = 1.
Note 2. AGT4 cannot operate in Deep Software Standby mode.

**Table 22.10 Usable settings in Software Standby and Deep Software Standby mode (AGTn (n = 1, 3, 5)*3)**

| Operating mode | AGTMR1.TCK[2:0] | Operating clock | Resurgence factor of CPU |
|---|---|---|---|
| Timer mode | 100b or 110b or 101b*1 | AGTLCLK or AGTSCLK or AGTn (n = 0, 2, 4) underflow | • Underflow<br>• Compare match A/B |
| Pulse output mode | 100b or 110b or 101b*1 | AGTLCLK or AGTSCLK or AGTn (n = 0, 2, 4) underflow | • Underflow<br>• Compare match A/B |
| Event counter mode | — | AGTIOn (n = 1, 3, 5)*2 | • Underflow<br>• Compare match A/B |
| Pulse width measurement mode | 100b or 110b or 101b*1 | AGTLCLK or AGTSCLK or AGTn (n = 0, 2, 4) underflow | • Underflow<br>• Active edge |
| Pulse period measurement mode | 100b or 110b or 101b*1 | AGTLCLK or AGTSCLK or AGTn (n = 0, 2, 4) underflow | • Underflow<br>• Active edge |

Note: —: invalid
Note: Release of Software Standby or Deep Software Standby mode is only AGT1.
Note: Compare match A/B is resurgence factor of CPU from Software Standby mode.
Note 1. Only when AGTn (n = 0, 2, 4) operates in Table 22.9
Note 2. When using the AGTIOn pin for external event input in Software Standby mode, set AGTIOSEL.TIES = 1.
Note 3. AGT5 cannot operate in Deep Software Standby mode.

## 22.3.11 Interrupt Sources

The AGTn has three interrupt sources as listed in Table 22.11.

**Table 22.11 AGT interrupt sources**

| Name | Interrupt source | DMAC/DTC activation |
|---|---|---|
| AGTn_AGTI | • When the counter underflows<br>• When measurement of the active width of the external input pin (AGTIOn) is complete in pulse width measurement mode<br>• When the set edge of the external input pin (AGTIOn) is input in pulse period measurement mode. | Possible |
| AGTn_AGTCMAI | • When the values of AGT register and AGTCMA register match | Possible |
| AGTn_AGTCMBI | • When the values of AGT register and AGTCMB register match | Possible |

Note: Channel number (n = 0 to 5)

## 22.3.12 Event Signal Output to ELC

The AGT uses the Event Link Controller (ELC) to perform a link operation to a specified module using the interrupt request signal as the event signal. The AGT outputs compare match A, compare match B, and underflow/measurement complete signals as event signals. For details, see section 18, Event Link Controller (ELC).

## 22.4 Usage Notes

### 22.4.1 Count Operation Start and Stop Control

- When the operating mode (see Table 22.1) is set to other than the event counter mode, or the count source is set to other than AGTn underflow event signal (TCK[2:0] = 101b):
  - After 1 (count starts) is written to the TSTART bit in the AGTCR register while the count is stopped, the TCSTF flag in the AGTCR register remains 0 (count stops) for 3 cycles of the count source. Do not access the registers associated with AGT other than the TCSTF flag until this flag is set to 1 (count in progress).
  - After 0 (count stops) is written to the TSTART bit during a count operation, the TCSTF flag remains 1 for 3 cycles of the count source. When the TCSTF flag is set to 0, the count is stopped. Do not access the registers associated with AGT other than the TCSTF flag until this flag is set to 0.
- When the operating mode (see Table 22.1) is set to event counter mode, or the count source is set to AGT1 underflow event signal (TCK[2:0] = 101b):
  - After 1 (count starts) is written to the TSTART bit in the AGTCR register while the count is stopped, the TCSTF flag in the AGTCR register remains 0 (count stops) for 2 PCLKB cycles. Do not access the registers associated with AGT other than the TCSTF flag until this flag is set to 1 (count in progress).
  - After 0 (count stops) is written to the TSTART bit during a count operation, the TCSTF flag remains 1 for 2 PCLKB cycles. When the TCSTF flag is set to 0, the count is stopped. Do not access the registers associated with AGT other than the TCSTF flag until this flag is set to 0.

### 22.4.2 Access to Counter Register

When the TSTART bit and TCSTF flag in the AGTCR register are both 1 (count starts), allow at least 3 cycles of the count source clock between writes when writing to the AGT register successively.

### 22.4.3 When Changing Mode

The registers associated with AGT operating mode (AGTMR1, AGTMR2, AGTIOC, AGTISR, and AGTCMSR) can be changed only when the count is stopped with both the TSTART bit and TCSTF flag set to 0 (count stops). Do not change these registers during count operation.

When the registers associated with AGT operating mode are changed, the values of TEDGF, TUNDF, TCMAF, and TCMBF flags are undefined. Before starting the count, write 0 to the following flags:

- TEDGF (no active edge received)
- TUNDF (no underflow)
- TCMAF (no match)
- TCMBF (no match).

### 22.4.4 Output pin setting

When using the AGTOn, AGTIOn, AGTOAn, or AGTOBn as an output pin, set up the Operation and determine the initial output values. Then set an output mode in the port register.

When using the AGTIOn as an input pin in pulse width measurement mode or pulse period measurement mode, set up the Operation and start count operation. Then start to enter external events from the AGTIOn pin. Invalidate the first measurement and validate the second and later completed measurements.

### 22.4.5 Digital Filter

When using the digital filter, do not start the timer operation for 5 cycles of the digital filter clock after setting TIPF[1:0] bits and when the TEDGSEL bit in the AGTIOC register changes.

### 22.4.6 How to Calculate Event Number, Pulse Width, and Pulse Period

- In event counter mode, event number is expressed mathematically as follows:
  Event number = initial value of counter [AGT register] - counter value of active event end

- In pulse width measurement mode, pulse width is expressed mathematically as follows:
  Pulse width = counter value of stopping measurement - counter value of next stopping measurement
- In pulse period measurement mode, input pulse period is expressed mathematically as follows:
  Period of input pulse = (initial value of counter [AGT register] - reading value of the read-out buffer) + 1.

### 22.4.7 When Count is Forcibly Stopped by TSTOP Bit

After the counter is forcibly stopped by the TSTOP bit in the AGTCR register, do not access the following I/O registers for 1 cycle of the count source:

- AGT
- AGTCMA
- AGTCMB
- AGTCR
- AGTMR1
- AGTMR2.

### 22.4.8 When Selecting AGTn (n = 0, 2, 4) Underflow as the Count Source

Operate according to the following procedures described in this section when selecting the underflow event signal as the count source.

#### (1) Procedure for starting operation

1. Set AGT.
2. Start the count operation of AGTn (n = 1, 3, 5).
3. Start the count operation of AGTn (n = 0, 2, 4).

#### (2) Procedure for stopping operation

1. Stop the count operation of AGTn (n = 0, 2, 4).
2. Stop the count operation of AGTn (n = 1, 3, 5).
3. Stop the count source clock of AGTn (n = 1, 3, 5) (write 000b in the AGTMR1.TCK[2:0] bits).

### 22.4.9 Module-stop function

AGT operation can be disabled or enabled using Module Stop Control Register D (MSTPCRD) and Module Stop Control Register E (MSTPCRE). The AGT module is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes

### 22.4.10 When Switching Source Clock

When switching a clock source by changing SCKSCR.CKSEL[2:0], the clock output from the selector stops for 4 cycles of the switched clock. Therefore, when using the AGTIOn, AGTEEn, or both input as external event input, the clock source should not be switched. If switching the clock source while using the external event input, extend the input pulse width by 4 clock cycles of the switched source clock cycles.

# 23. Realtime Clock (RTC)

## 23.1 Overview

The realtime clock (RTC) has two counting modes, calendar count mode and binary count mode, that are used by switching register settings. For calendar count mode, the RTC has a 100-year calendar from 2000 to 2099 and automatically adjusts dates for leap years. For binary count mode, the RTC counts seconds and retains the information as a serial value. Binary count mode can be used for calendars other than the Gregorian (Western) calendar.

The sub-clock oscillator or LOCO can be selected as the count source of the time counters. The RTC uses a 128-Hz clock acquired by dividing the count source by a prescaler. Year, month, date, day-of-week, a.m. /p.m. (in 12-hour mode), hour, minute, second, or 32-bit binary is counted by 1/128 second.

Table 23.1 lists the RTC specifications, Figure 23.1 shows a block diagram, and Table 23.2 lists the I/O pins.

**Table 23.1 RTC specifications**

| Parameter | Specifications |
|---|---|
| Count mode | Calendar count mode/binary count mode |
| Count source*1 | Sub-clock (XCIN) or LOCO |
| Clock and calendar functions | ● Calendar count mode<br>– Year, month, date, day of week, hour, minute, second are counted, BCD display<br>– 12 hours/24 hours mode switching function<br>– 30 seconds adjustment function (a number less than 30 is rounded down to 00 seconds, and 30 seconds or more are rounded up to 1 minute)<br>– Automatic adjustment function for leap years<br>● Binary count mode<br>– Count seconds in 32 bits, binary display<br>● Shared by both modes<br>– Start/stop function<br>– The sub-second digit is displayed in binary units (1 Hz, 2 Hz, 4 Hz, 8 Hz, 16 Hz, 32 Hz, or 64 Hz)<br>– Clock error correction function<br>– Clock (1-Hz/64-Hz) output |
| Interrupts | ● Alarm interrupt (RTC_ALM)<br>– As an alarm interrupt condition, selectable for comparison with the following:<br>– Calendar count mode: Year, month, date, day-of-week, hour, minute, or second can be selected<br>– Binary count mode: Each bit of the 32-bit binary counter<br>● Periodic interrupt (RTC_PRD)<br>– 2 seconds, 1 second, 1/2 second, 1/4 second, 1/8 second, 1/16 second, 1/32 second, 1/64 second, 1/128 second, or 1/256 second can be selected as an interrupt period.<br>● Carry interrupt (RTC_CUP)<br>– An interrupt is generated at either of the following conditions:<br>• When a carry from the 64-Hz counter to the second counter is generated.<br>• When the 64-Hz counter is changed and the R64CNT register is read at the same time. (32-KHz count mode is only for 64-Hz counter reading)<br>● Return from Software Standby or Deep Software Standby mode can be performed by the alarm interrupt or periodic interrupt |
| Time capture function | ● Times can be captured when the edge of the time capture event input pin is detected. For every event input, month, date, hour, minute, and second are captured or the 32-bit binary counter value is captured.<br>● Interrupt can be generated when the edge of the time capture event input is detected. The time capture event input pin and IRQ are shared. |
| Event link function | Periodic event output (RTC_PRD) |
| TrustZone filter | Security attribution can be set |

Note 1. The frequency of the peripheral module clock (PCLKB) ≥ the frequency of the count source should be satisfied.

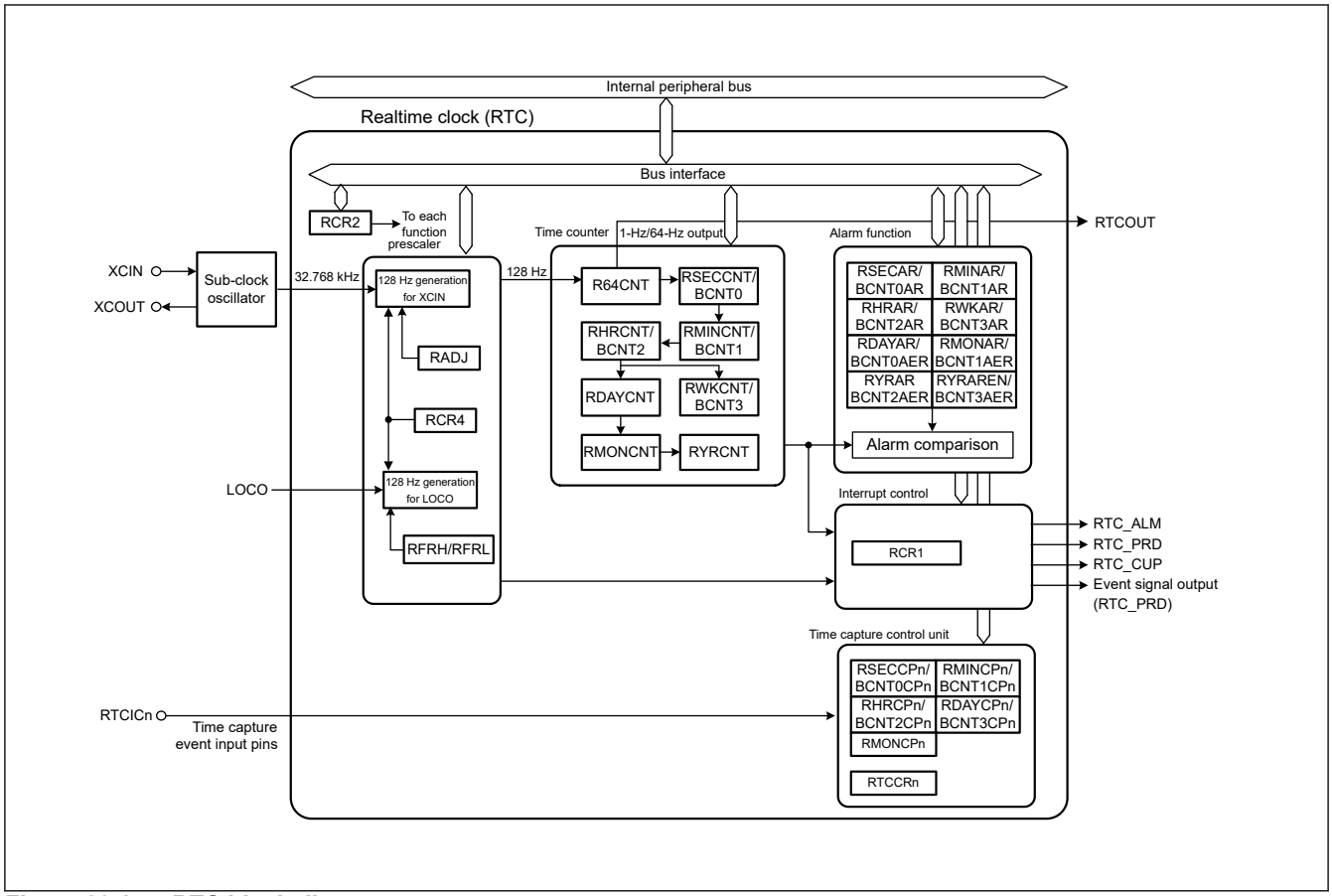



**Figure 23.1 RTC block diagram**

**Table 23.2 RTC I/O pins**

| Pin name | I/O | Description |
|---|---|---|
| XCIN | Input | Connect a 32.768-kHz crystal to these pins |
| XCOUT | Output | |
| RTCOUT | Output | This pin is used to output a 1-Hz/64-Hz waveform, but not in Deep Software Standby mode |
| RTCICn (n = 0 to 2) | Input | Time capture event input pins<br>RTCICn can be controlled by the VBTICTLR register. For more information, see section 11, Battery Backup Function and section 19, I/O Ports. |

## 23.2 Register Descriptions

Write or read from the RTC registers as described in section 23.6.5. Notes on Writing to and Reading from Registers.

If the value in an RTC register after a reset is given as x (undefined bits) in the list, it is not initialized by a reset. When RTC enters the reset state or a low power state during counting operations, for example, while the RCR2.START bit is 1, the year, month, day of the week, date, hours, minutes, seconds, and 64-Hz counters continue to operate.

Note: A reset generated while writing to a register might destroy the register value. In addition, do not allow the MCU to enter Software Standby mode, Deep Software Standby mode, or battery backup state immediately after setting any of these registers. For details, see section 23.6.4. Transitions to Low Power Modes after Setting Registers.

### 23.2.1 R64CNT : 64-Hz Counter

Base address: RTC = 0x4008_3000

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | F1HZ | F2HZ | F4HZ | F8HZ | F16HZ | F32HZ | F64HZ |
| Value after reset: | 0 | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | F64HZ | 64-Hz Flag<br>This bit indicates the 64-Hz state of the sub-second digit. | R |
| 1 | F32HZ | 32-Hz Flag<br>This bit indicates the 32-Hz state of the sub-second digit. | R |
| 2 | F16HZ | 16-Hz Flag<br>This bit indicates the 16-Hz state of the sub-second digit. | R |
| 3 | F8HZ | 8-Hz Flag<br>This bit indicates the 8-Hz state of the sub-second digit. | R |
| 4 | F4HZ | 4-Hz Flag<br>This bit indicates the 4-Hz state of the sub-second digit. | R |
| 5 | F2HZ | 2-Hz Flag<br>This bit indicates the 2-Hz state of the sub-second digit. | R |
| 6 | F1HZ | 1-Hz Flag<br>This bit indicates the 1-Hz state of the sub-second digit. | R |
| 7 | — | This bit is read as 0. | R |

The R64CNT counter is used in both calendar count mode and binary count mode The 64-Hz counter (R64CNT) generates the period for a second by counting up periods of the 128-Hz clock. The state in the sub-second range can be confirmed by reading this counter.

This counter is set to 0x00 by an RTC software reset or an execution of a 30-second adjustment. To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

### 23.2.2 RSECCNT : Second Counter (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x02

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | SEC10[2:0] | | | SEC1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | SEC1[3:0] | 1-Second Count<br>Counts from 0 to 9 every second. When a carry is generated, 1 is added to the tens place. | R/W |
| 6:4 | SEC10[2:0] | 10-Second Count<br>Counts from 0 to 5 for 60-second counting. | R/W |
| 7 | — | The read value is undefined. The write value should be 0. | R/W |

The RSECCNT counter sets and counts the BCD-coded second value. It counts the carries generated once per second in the 64-Hz counter.

The setting range is decimal 00 to 59. The RTC does not operate normally if any other value is set. Before writing to this register, you must stop the count operation using the START bit in RCR2.

To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

### 23.2.3 RMINCNT : Minute Counter (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | MIN10[2:0] | | | MIN1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | MIN1[3:0] | 1-Minute Count<br>Counts from 0 to 9 every minute. When a carry is generated, 1 is added to the tens place. | R/W |
| 6:4 | MIN10[2:0] | 10-Minute Count<br>Counts from 0 to 5 for 60-minute counting. | R/W |
| 7 | — | The read value is undefined. The write value should be 0. | R/W |

The RMINCNT counter sets and counts the BCD-coded minute value. It counts the carries generated once every minute in the second counter.

A value from 00 through 59 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. Before writing to this register, you must stop the count operation using the START bit in RCR2. To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

### 23.2.4 RHRCNT : Hour Counter (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x06

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | PM | HR10[1:0] | | HR1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | HR1[3:0] | 1-Hour Count<br>Counts from 0 to 9 once per hour. When a carry is generated, 1 is added to the tens place. | R/W |
| 5:4 | HR10[1:0] | 10-Hour Count<br>Counts from 0 to 2 once per carry from the ones place. | R/W |
| 6 | PM | AM/PM select for time counter setting.<br>0: AM<br>1: PM | R/W |
| 7 | — | The read value is undefined. The write value should be 0. | R/W |

The RHRCNT counter sets and counts the BCD-coded hour value. It counts the carries generated once per hour in the minute counter. The specifiable time differs based on the setting in the hours mode bit (RCR2.HR24):

- When the RCR2.HR24 bit is 0 – from 00 to 11 (in BCD).
- When the RCR2.HR24 bit is 1 – from 00 to 23 (in BCD).

If a value outside of this range is specified, the RTC does not operate correctly. Before writing to this register, you must stop the count operation using the START bit in RCR2. The PM bit is only enabled when the RCR2.HR24 bit is 0.

Otherwise, the setting in the PM bit has no effect. To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

### 23.2.5 RWKCNT : Day-of-Week Counter (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x08

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | DAYW[2:0] | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | DAYW[2:0] | Day-of-Week Counting<br>0 0 0: Sunday<br>0 0 1: Monday<br>0 1 0: Tuesday<br>0 1 1: Wednesday<br>1 0 0: Thursday<br>1 0 1: Friday<br>1 1 0: Saturday<br>1 1 1: Setting prohibited | R/W |
| 7:3 | — | The read values are undefined. The write value should be 0. | R/W |

The RWKCNT counter sets and counts in the coded day-of-week value. It counts the carries generated once per day in the hour counter. A value from 0 through 6 can be specified. If a value outside of this range is specified, the RTC does not operate correctly. Before writing to this register, you must stop the count operation using the START bit in RCR2. To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

### 23.2.6 BCNTn : Binary Counter n (n = 0 to 3) (in Binary Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x02 + 0x02 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | BCNT[7:0] | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | BCNT[7:0] | Binary Counter | R/W |

BCNTn is a read/write 8-bit register to access BCNT[31:0] that is a 32-bit binary counter. BCNT3 is assigned to the BCNT[31:24] bits, BCNT2 is assigned to the BCNT[23:16] bits, BCNT1 is assigned to the BCNT[15:8] bits, and BCNT0 is assigned to the BCNT[7:0] bits. BCNTn performs count operation by a carry generated for each second of the 64-Hz counter. Before writing to this register, you must stop the count operation using the START bit in RCR2. To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

### 23.2.7 RDAYCNT : Day Counter

Base address: RTC = 0x4008_3000

Offset address: 0x0A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | DATE10[1:0] | | DATE1[3:0] | | | |
| Value after reset: | 0 | 0 | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | DATE1[3:0] | 1-Day Count<br>Counts from 0 to 9 once per day. When a carry is generated, 1 is added to the tens place. | R/W |
| 5:4 | DATE10[1:0] | 10-Day Count<br>Counts from 0 to 3 once per carry from the ones place. | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |

The RDAYCNT counter is used in calendar count mode to set and count the BCD-coded date value. It counts the carries generated once per day in the hour counter. The count operation depends on the month and whether the year is a leap year. Leap years are determined according to whether the year counter (RYRCNT) value is divisible by 400, 100, and 4.

A value from 01 through 31 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. When specifying a value, the range of specifiable days depends on the month and whether the year is a leap year. Before writing to this register, you must stop the count operation using the START bit in RCR2. To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

## 23.2.8 RMONCNT : Month Counter

Base address: RTC = 0x4008_3000

Offset address: 0x0C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | MON10 | MON1[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | MON1[3:0] | 1-Month Count<br>Counts from 0 to 9 once per month. When a carry is generated, 1 is added to the tens place. | R/W |
| 4 | MON10 | 10-Month Count<br>Counts from 0 to 1 once per carry from the ones place. | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

The RMONCNT counter is used in calendar count mode to set and count the BCD-coded month value. It counts the carries generated once per month in the date counter.

A value from 01 through 12 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. Before writing to this register, you must stop the count operation using the START bit in RCR2. To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

## 23.2.9 RYRCNT : Year Counter

Base address: RTC = 0x4008_3000

Offset address: 0x0E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | YR10[3:0] | | | | YR1[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | YR1[3:0] | 1-Year Count<br>Counts from 0 to 9 once per year. When a carry is generated, 1 is added to the tens place. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:4 | YR10[3:0] | 10-Year Count<br>Counts from 0 to 9 once per carry from ones place. When a carry is generated in the tens place, 1 is added to the hundreds place. | R/W |
| 15:8 | — | These bits are read as 0. The write value should be 0. | R/W |

The RYRCNT counter is used in calendar count mode to set and count the BCD-coded year value. It counts the carries generated once per year in the month counter.

A value from 00 through 99 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. Before writing to this register, you must stop the count operation using the START bit in RCR2. To read this counter, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.

## 23.2.10 RSECAR : Second Alarm Register (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x10

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB | SEC10[2:0] | | | SEC1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | SEC1[3:0] | 1 Second<br>Value for the ones place of seconds. | R/W |
| 6:4 | SEC10[2:0] | 10 Seconds<br>Value for the tens place of seconds. | R/W |
| 7 | ENB | ENB<br>0: Do not compare register value with RSECCNT counter value<br>1: Compare register value with RSECCNT counter value | R/W |

RSECAR is an alarm register associated with the BCD-coded second counter RSECCNT. When the ENB bit is set to 1, the RSECAR value is compared with the RSECCNT value. From the following alarm registers, only those selected with the ENB bits set to 1 are compared with the associated counters:

- RSECAR
- RMINAR
- RHRAR
- RWKAR
- RDAYAR
- RMONAR
- RYRAREN

When all the respective values match, the IR flag associated with the RTC_ALM interrupt is set to 1. RSECAR values from 00 through 59 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. This register is set to 0x00 by an RTC software reset.

## 23.2.11 RMINAR : Minute Alarm Register (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x12

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB | MIN10[2:0] | | | MIN1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | MIN1[3:0] | 1 Minute<br>Value for the ones place of minutes. | R/W |
| 6:4 | MIN10[2:0] | 10 Minutes<br>Value for the tens place of minutes. | R/W |
| 7 | ENB | ENB<br>0: Do not compare register value with RMINCNT counter value<br>1: Compare register value with RMINCNT counter value | R/W |

RMINAR is an alarm register associated with the BCD-coded minute counter RMINCNT. When the ENB bit is set to 1, the RMINAR value is compared with the RMINCNT value. From the following alarm registers, only those selected with the ENB bits set to 1 are compared with the associated counters:

- RSECAR
- RMINAR
- RHRAR
- RWKAR
- RDAYAR
- RMONAR
- RYRAREN

When all the respective values match, the IR flag associated with the RTC_ALM interrupt is set to 1. RMINAR values from 00 through 59 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. This register is set to 0x00 by an RTC software reset.

## 23.2.12 RHRAR : Hour Alarm Register (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x14

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB | PM | HR10[1:0] | | HR1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | HR1[3:0] | 1 Hour<br>Value for the ones place of hours. | R/W |
| 5:4 | HR10[1:0] | 10 Hours<br>Value for the tens place of hours. | R/W |
| 6 | PM | AM/PM select for alarm setting.<br>0: AM<br>1: PM | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | ENB | ENB<br>0: Do not compare register value with RHRCNT counter value<br>1: Compare register value with RHRCNT counter value | R/W |

RHRAR is an alarm register associated with the BCD-coded hour counter RHRCNT. When the ENB bit is set to 1, the RHRAR value is compared with the RHRCNT value. From the following alarm registers, only those selected with the ENB bits set to 1 are compared with the associated counters:

- RSECAR
- RMINAR
- RHRAR
- RWKAR
- RDAYAR
- RMONAR
- RYRAREN

When all the respective values match, the IR flag associated with the RTC_ALM interrupt is set to 1. The specifiable time differs according to the setting in the hours mode bit (RCR2.HR24):

- When the RCR2.HR24 bit is 0 – From 00 to 11 (in BCD).
- When the RCR2.HR24 bit is 1 – From 00 to 23 (in BCD).

If a value outside of this range is specified, the RTC does not operate correctly. When the RCR2.HR24 bit is 0, you must set the PM bit. When the RCR2.HR24 bit is 1, the setting in the PM bit has no effect. This register is set to 0x00 by an RTC software reset.

## 23.2.13 RWKAR : Day-of-Week Alarm Register (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x16

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB | — | — | — | — | DAYW[2:0] | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | DAYW[2:0] | Day-of-Week Setting<br>0 0 0: Sunday<br>0 0 1: Monday<br>0 1 0: Tuesday<br>0 1 1: Wednesday<br>1 0 0: Thursday<br>1 0 1: Friday<br>1 1 0: Saturday<br>1 1 1: Setting prohibited | R/W |
| 6:3 | — | The read values are undefined. The write value should be 0. | R/W |
| 7 | ENB | ENB<br>0: Do not compare register value with RWKCNT counter value<br>1: Compare register value with RWKCNT counter value | R/W |

RWKAR is an alarm register associated with the coded day-of-week counter RWKCNT. When the ENB bit is set to 1, the RWKAR value is compared with the RWKCNT value. From the following alarm registers, only those selected with the ENB bits set to 1 are compared with the associated counters:

- RSECAR

- RMINAR
- RHRAR
- RWKAR
- RDAYAR
- RMONAR
- RYRAREN

When all the respective values match, the IR flag associated with the RTC_ALM interrupt is set to 1. RWKAR values from 0 through 6 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. This register is set to 0x00 by an RTC software reset.

## 23.2.14 BCNTnAR : Binary Counter n Alarm Register (n = 0 to 3) (in Binary Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x10 + 0x02 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | BCNTAR | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | BCNTAR | Alarm register associated with the 32-bit binary counter | R/W |

BCNTnAR is a read/write alarm register associated with the 32-bit binary counter. BCNT3AR is assigned to the BCNTAR[31:24] bits, BCNT2AR is assigned to the BCNTAR[23:16] bits, BCNT1AR is assigned to the BCNTAR[15:8] bits, and BCNT0AR is assigned to the BCNTAR[7:0]. This register is set to 0x00 by an RTC software reset.

## 23.2.15 RDAYAR : Date Alarm Register (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x18

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB | — | DATE10[1:0] | | DATE1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | DATE1[3:0] | 1 Day<br>Value for the ones place of days. | R/W |
| 5:4 | DATE10[1:0] | 10 Days<br>Value for the tens place of days. | R/W |
| 6 | — | The read value is undefined. The write value should be 0. | R/W |
| 7 | ENB | ENB<br>0: Do not compare register value with RDAYCNT counter value<br>1: Compare register value with RDAYCNT counter value | R/W |

RDAYAR is an alarm register associated with the BCD-coded date counter RDAYCNT. When the ENB bit is set to 1, the RDAYAR value is compared with the RDAYCNT value. From the following alarm registers, only those selected with the ENB bits set to 1 are compared with the associated counters:

- RSECAR
- RMINAR
- RHRAR

- RWKAR
- RDAYAR
- RMONAR
- RYRAREN

When all the respective values match, the IR flag associated with the RTC_ALM interrupt is set to 1. The RDAYAR values from 01 through 31 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. This register is set to 0x00 by an RTC software reset.

## 23.2.16 RMONAR : Month Alarm Register (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x1A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB | — | — | MON10 | MON1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | MON1[3:0] | 1 Month<br>Value for the ones place of months. | R/W |
| 4 | MON10 | 10 Months<br>Value for the tens place of months. | R/W |
| 6:5 | — | The read values are undefined. The write value should be 0. | R/W |
| 7 | ENB | ENB<br>0: Do not compare register value with RMONCNT counter value<br>1: Compare register value with RMONCNT counter value | R/W |

RMONAR is an alarm register associated with the BCD-coded month counter RMONCNT. When the ENB bit is set to 1, the RMONAR value is compared with the RMONCNT value. From the following alarm registers, only those selected with the ENB bits set to 1 are compared with the associated counters:

- RSECAR
- RMINAR
- RHRAR
- RWKAR
- RDAYAR
- RMONAR
- RYRAREN

When all the respective values match, the IR flag associated with the RTC_ALM interrupt is set to 1. The RMONAR values from 01 through 12 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. This register is set to 0x00 by an RTC software reset.

## 23.2.17 RYRAR : Year Alarm Register (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x1C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | YR10[3:0] | | | | YR1[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | YR1[3:0] | 1 Year<br>Value for the ones place of years. | R/W |
| 7:4 | YR10[3:0] | 10 Years<br>Value for the tens place of years. | R/W |
| 15:8 | — | These bits are read as 0. The write value should be 0. | R/W |

RYRAR is an alarm register associated with the BCD-coded year counter RYRCNT. The RYRAR values from 00 through 99 (in BCD) can be specified. If a value outside of this range is specified, the RTC does not operate correctly. This register is set to 0x0000 by an RTC software reset.

## 23.2.18 RYRAREN : Year Alarm Enable Register (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x1E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB | — | — | — | — | — | — | — |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:0 | — | The read values are undefined. The write value should be 0. | R/W |
| 7 | ENB | ENB<br>0: Do not compare register value with the RYRCNT counter value<br>1: Compare register value with the RYRCNT counter value | R/W |

When the ENB bit in the RYRAREN register is set to 1, the RYRAR value is compared with the RYRCNT value. From the following alarm registers, only those selected with the ENB bits set to 1 are compared with the associated counters:

- RSECAR
- RMINAR
- RHRAR
- RWKAR
- RDAYAR
- RMONAR
- RYRAREN

When all the respective values match, the IR flag associated with the RTC_ALM interrupt is set to 1. This register is set to 0x00 by an RTC software reset.

## 23.2.19 BCNTnAER : Binary Counter n Alarm Enable Register (n = 0, 1) (in Binary Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x18 + 0x02 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB[7:0] | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | ENB[7:0] | Setting the alarm enable associated with the 32-bit binary counter | R/W |

BCNTnAER is a read/write register for setting the alarm enable (BCNTAER) associated with the 32-bit binary counter. BCNT3AER is assigned to the BCNTAER.ENB[31:24] bits, BCNT2AER register is assigned to the BCNTAER.ENB[23:16] bits, BCNT1AER is assigned to the BCNTAER.ENB[15:8] bits, and BCNT0AER is assigned to the BCNTAER.ENB[7:0] bits. The binary counter (BCNT[31:0]) associated with the BCNTAER.ENB[31:0] bits that are set to 1 is compared with the binary alarm register (BCNTAR) and, when all match, the IR flag associated with the RTC_ALM interrupt is set to 1. This register is set to 0x00 by an RTC software reset.

## 23.2.20 BCNT2AER : Binary Counter 2 Alarm Enable Register (in Binary Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x1C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | ENB[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | ENB[7:0] | Setting the alarm enable associated with the 32-bit binary counter | R/W |
| 15:8 | — | These bits are read as 0. The write value should be 0. | R/W |

BCNT2AER is a read/write register for setting the alarm enable (BCNTAER) associated with the 32-bit binary counter. BCNT3AER is assigned to the BCNTAER.ENB[31:24] bits, BCNT2AER register is assigned to the BCNTAER.ENB[23:16] bits, BCNT1AER is assigned to the BCNTAER.ENB[15:8] bits, and BCNT0AER is assigned to the BCNTAER.ENB[7:0] bits. The binary counter (BCNT[31:0]) associated with the BCNTAER.ENB[31:0] bits that are set to 1 is compared with the binary alarm register (BCNTAR) and, when all match, the IR flag associated with the RTC_ALM interrupt is set to 1. This register is set to 0x00 by an RTC software reset.

## 23.2.21 BCNT3AER : Binary Counter 3 Alarm Enable Register (in Binary Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x1E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ENB[7:0] | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | ENB[7:0] | Setting the alarm enable associated with the 32-bit binary counter | R/W |

BCNT3AER is a read/write register for setting the alarm enable (BCNTAER) associated with the 32-bit binary counter. BCNT3AER is assigned to the BCNTAER.ENB[31:24] bits, BCNT2AER register is assigned to the BCNTAER.ENB[23:16] bits, BCNT1AER is assigned to the BCNTAER.ENB[15:8] bits, and BCNT0AER is assigned to the BCNTAER.ENB[7:0] bits. The binary counter (BCNT[31:0]) associated with the BCNTAER.ENB[31:0] bits that are set to 1 is compared with the binary alarm register (BCNTAR) and, when all match, the IR flag associated with the RTC_ALM interrupt is set to 1. This register is set to 0x00 by an RTC software reset.

## 23.2.22 RCR1 : RTC Control Register 1

Base address: RTC = 0x4008_3000

Offset address: 0x22

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | PES[3:0] | | | | RTCOS | PIE | CIE | AIE |
| Value after reset: | x | x | x | x | 0 | x | 0 | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | AIE | Alarm Interrupt Enable<br>0: Disable alarm interrupt requests<br>1: Enable alarm interrupt requests | R/W |
| 1 | CIE | Carry Interrupt Enable<br>0: Disable carry interrupt requests<br>1: Enable carry interrupt requests | R/W |
| 2 | PIE | Periodic Interrupt Enable<br>0: Disable periodic interrupt requests<br>1: Enable periodic interrupt requests | R/W |
| 3 | RTCOS | RTCOUT Output Select<br>0: Outputs 1 Hz on RTCOUT<br>1: Outputs 64 Hz RTCOUT | R/W |
| 7:4 | PES[3:0] | Periodic Interrupt Select<br>0x6: Generate periodic interrupt every 1/256 second*1<br>0x7: Generate periodic interrupt every 1/128 second<br>0x8: Generate periodic interrupt every 1/64 second<br>0x9: Generate periodic interrupt every 1/32 second<br>0xA: Generate periodic interrupt every 1/16 second<br>0xB: Generate periodic interrupt every 1/8 second<br>0xC: Generate periodic interrupt every 1/4 second<br>0xD: Generate periodic interrupt every 1/2 second<br>0xE: Generate periodic interrupt every 1 second<br>0xF: Generate periodic interrupt every 2 seconds<br>Others: Do not generate periodic interrupts | R/W |

Note 1. When LOCO is selected (RCR4.RCKSEL = 1) while PES[3:0] = 0x6, a periodic interrupt is generated every 1/128 second.

The RCR1 register is used in both calendar count mode and binary count mode. Bits AIE, PIE, and PES[3:0] are updated synchronously with the count source. When the RCR1 register is modified, check that all the bits are updated before proceeding.

### AIE bit (Alarm Interrupt Enable)

The AIE bit enables or disables alarm interrupt requests.

If the times indicated in the counters and alarm settings match in Deep Software Standby mode, the MCU returns from Deep Software Standby mode regardless of the AIE bit value.

### CIE bit (Carry Interrupt Enable)

The CIE bit enables or disables interrupt requests when a carry to the RSECCNT/BCNT0 register occurs, or when a carry to the 64-Hz counter (R64CNT) occurs while reading the 64-Hz counter.

### PIE bit (Periodic Interrupt Enable)

The PIE bit enables or disabled a periodic interrupt.

If the periods indicated in the counters and PES[3:0] settings match in Deep Software Standby mode, the MCU returns from Deep Software Standby mode regardless of the PIE bit value.

### RTCOS bit (RTCOUT Output Select)

The RTCOS bit selects the RTCOUT output period. The RTCOS bit must be rewritten while the count operation is stopped (RCR2.START = 0) and the RTCOUT output is disabled (RCR2.RTCOE = 0). When RTCOUT is output to an external pin, the RCR2.RTCOE bit must be enabled.

### PES[3:0] bits (Periodic Interrupt Select)

The PES[3:0] bits specify the period for the periodic interrupt. A periodic interrupt is generated with the period specified in these bits.

## 23.2.23 RCR2 : RTC Control Register 2 (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x24

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CNTMD | HR24 | AADJP | AADJE | RTCOE | ADJ30 | RESET | START |
| Value after reset: | x | x | x | x | 0 | 0 | 0 | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | START | Start<br>0: Stop prescaler and time counter<br>1: Operate prescaler and time counter normally | R/W |
| 1 | RESET | RTC Software Reset<br>0: In writing: Invalid (writing 0 has no effect).<br>In reading: Normal time operation in progress, or an RTC software reset has completed.<br>1: In writing: Initialize the prescaler and target registers for RTC software reset*1.<br>In reading: RTC software reset in progress. | R/W |
| 2 | ADJ30 | 30-Second Adjustment<br>0: In writing: Invalid (writing 0 has no effect).<br>In reading: Normal time operation in progress, or 30-second adjustment has completed.<br>1: In writing: Execute 30-second adjustment.<br>In reading: 30-second adjustment in progress. | R/W |
| 3 | RTCOE | RTCOUT Output Enable<br>0: Disable RTCOUT output<br>1: Enable RTCOUT output | R/W |
| 4 | AADJE | Automatic Adjustment Enable*2*3<br>0: Disable automatic adjustment<br>1: Enable automatic adjustment | R/W |
| 5 | AADJP | Automatic Adjustment Period Select*2*3<br>0: The RADJ.ADJ[5:0] setting from the count value of the prescaler every minute.<br>1: The RADJ.ADJ[5:0] setting value is adjusted from the coun tvalue of the prescaler every 10 seconds. | R/W |
| 6 | HR24 | Hours Mode*3<br>0: Operate RTC in 12-hour mode<br>1: Operate RTC in 24-hour mode | R/W |
| 7 | CNTMD | Count Mode Select*4<br>0: Calendar count mode<br>1: Binary count mode | R/W |

Note 1. R64CNT, RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, RMONAR, RYRAR, RYRAREN, RADJ, RTCCRn, RSECCPn, RMINCPn, RHRCPn, RDAYCPn, RMONCPn, RCR2.ADJ30, RCR2.AADJE, RCR2.AADJP.

Note 2. When LOCO is selected, the setting of this bit is disabled.

Note 3. When rewriting this bit, confirm that the value has been rewritten before performing the following processing. See section 23.6.5. Notes on Writing to and Reading from Registers for notes on register writing/reading.

Note 4. When rewriting this bit, confirm that the value has been rewritten before performing the following processing.

The RCR2 register is related to hours mode, automatic adjustment function, enabling RTCOUT output, 30-second adjustment, RTC software reset, and controlling count operation.

#### START bit (Start)

The START bit stops or restarts the prescaler or time counter operation. This bit is updated in synchronization with the next cycle of the count source. When the START bit is modified, check that the bit is updated before proceeding.

#### RESET bit (RTC Software Reset)

The RESET bit initializes the prescaler and registers to be reset by RTC software. When 1 is written to this bit, initialization starts in synchronization with the count source. When the initialization is complete, the RESET bit is automatically set to 0. Check that this bit is 0 before proceeding.

#### ADJ30 bit (30-Second Adjustment)

The ADJ30 bit is for 30-second adjustment.

When 1 is written to the ADJ30 bit, the RSECCNT value of less than 30 seconds is rounded down to 00 second and the value of 30 seconds or more is rounded up to 1 minute.

The 30-second adjustment is performed in synchronization with the count source. When 1 is written to this bit, the ADJ30 bit is automatically set to 0 after the 30-second adjustment completes. If 1 is written to the ADJ30 bit, check that the bit is 0 before proceeding. When the 30-second adjustment is performed, the prescaler and R64CNT are also reset. The ADJ30 bit is set to 0 by an RTC software reset.

#### RTCOE bit (RTCOUT Output Enable)

The RTCOE bit enables output of a 1-Hz/64-Hz clock signal from the RTCOUT pin.

Use the START bit to stop counting before changing the value of the RTCOE bit. Do not stop counting (write 0 to the START bit) and change the value of the RTCOE bit at the same time.

When RTCOUT is to be output from an external pin, enable the RTCOE bit and set up the port control for the pin.

#### AADJE bit (Automatic Adjustment Enable)

The AADJE bit controls (enables or disables) automatic adjustment.

Set the plus-minus bits (RADJ.PMADJ[1:0]) to 00b (adjustment is not performed) before changing the value of the AADJE bit.

The AADJE bit is set to 0 by an RTC software reset.

#### AADJP bit (Automatic Adjustment Period Select)

The AADJP bit selects the automatic-adjustment period.

Set the plus-minus bits (RADJ.PMADJ[1:0]) to 00b (adjustment is not performed) before changing the value of the AADJP bit.

The AADJP bit is set to 0 by an RTC software reset.

#### HR24 bit (Hours Mode)

The HR24 bit specifies whether the RTC operates in 12- or 24-hour mode.

Use the START bit to stop counting before changing the value of the HR24 bit. Do not stop counting (write 0 to the START bit) and change the value of the HR24 bit at the same time.

#### CNTMD bit (Count Mode Select)

The CNTMD bit specifies whether the RTC count mode operates in calendar count mode or in binary count mode.

When setting the count mode, execute an RTC software reset and start again from the initial settings. This bit is updated in synchronization with the count source. However, the count mode switches only after the RTC software reset. (Bit switches before RTC reset, mode switches after RTC reset.)

For details on initial settings, see section 23.3.1. Outline of Initial Settings of Registers after Power On.

## 23.2.24 RCR2 : RTC Control Register 2 (in Binary Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x24

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CNTMD | — | AADJP | AADJE | RTCOE | — | RESET | START |
| Value after reset: | x | x | x | x | 0 | 0 | 0 | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | START | Start<br>0: Stop the 32-bit binary counter, 64-Hz counter, and prescaler<br>1: Operate the 32-bit binary counter, 64-Hz counter, and prescaler normally | R/W |
| 1 | RESET | RTC Software Reset<br>0: In writing: Invalid (writing 0 has no effect).<br>In reading: Normal time operation in progress, or an RTC software reset has completed.<br>1: In writing: Initialize the prescaler and target registers for RTC software reset*1.<br>In reading: RTC software reset in progress. | R/W |
| 2 | — | This bit is read as 0. The write value should be 0. | R/W |
| 3 | RTCOE | RTCOUT Output Enable<br>0: Disable RTCOUT output<br>1: Enable RTCOUT output | R/W |
| 4 | AADJE | Automatic Adjustment Enable*2*3<br>0: Disable automatic adjustment<br>1: Enable automatic adjustment | R/W |
| 5 | AADJP | Automatic Adjustment Period Select*2*3<br>0: Add or subtract RADJ.ADJ [5:0] bits from prescaler count value every 32 seconds<br>1: Add or subtract RADJ.ADJ [5:0] bits from prescaler countvalue every 8 seconds. | R/W |
| 6 | — | The read value is undefined. The write value should be 0. | R/W |
| 7 | CNTMD | Count Mode Select*4<br>0: Calendar count mode<br>1: Binary count mode | R/W |

Note 1. R64CNT, BCNTnAR, BCNTnAER, RADJ, RTCCRn, BCNTnCPm, RCR2.ADJ30, RCR2.AADJE, RCR2.AADJP.
Note 2. When LOCO is selected, the setting of this bit is disabled.
Note 3. When rewriting this bit, confirm that the value has been rewritten before performing the following processing. See section 23.6.5. Notes on Writing to and Reading from Registers for notes on register writing/reading.
Note 4. When rewriting this bit, confirm that the value has been rewritten before performing the following processing.

RCR2 in the binary count mode is a register related to the automatic correction function, RTCOUT output enable, RTC software reset, and count mode control.

### START bit (Start)

The START bit stops or restarts the prescaler or counter (clock) operation. This bit is updated in synchronization with the count source. When the START bit is modified, check that the bit is updated before proceeding.

### RESET bit (RTC Software Reset)

The RESET bit initializes the prescaler and registers to be reset by RTC software. When 1 is written to this bit, initialization starts in synchronization with the count source. When the initialization is complete, the RESET bit is automatically set to 0. When 1 is written to the RESET bit, check that the bit is 0 before proceeding.

### RTCOE bit (RTCOUT Output Enable)

The RTCOE bit enables output of a 1-Hz/64-Hz clock signal from the RTCOUT pin.

Use the START bit to stop counting before changing the value of the RTCOE bit. Do not stop counting (write 0 to the START bit) and change the value of the RTCOE bit at the same time. When an RTCOUT signal is to be output from an external pin, enable the port control in addition to setting this bit.

### AADJE bit (Automatic Adjustment Enable)

The AADJE bit controls (enables or disables) automatic adjustment.

Set the plus-minus bits (RADJ.PMADJ[1:0]) to 00b (adjustment is not performed) before changing the value of the AADJE bit. The AADJE bit is set to 0 by an RTC software reset.

### AADJP bit (Automatic Adjustment Period Select)

The AADJP bit selects the automatic-adjustment period.

Correction period can be selected from 32 second units or 8 second units in binary count mode.

Set the plus-minus bits (RADJ.PMADJ[1:0]) to 00b (adjustment is not performed) before changing the value of the AADJP bit. The AADJP bit is set to 0 by an RTC software reset.

### CNTMD bit (Count Mode Select)

The CNTMD bit specifies whether the RTC count mode operates in calendar count mode or in binary count mode.

When setting the count mode, execute an RTC software reset and start again from the initial settings. This bit is updated in synchronization with the count source. However, the count mode switches only after the RTC software reset. (Bit switches before RTC reset, mode switches after RTC reset.)

For details on initial settings, see section 23.3.1. Outline of Initial Settings of Registers after Power On .

## 23.2.25 RCR4 : RTC Control Register 4

Base address: RTC = 0x4008_3000

Offset address: 0x28

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | RCKSEL |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RCKSEL | Count Source Select<br>0: Sub-clock oscillator is selected<br>1: LOCO is selected | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

The RCR4 register is used in both calendar count mode and binary count mode.

### RCKSEL bit (Count Source Select )

The RCKSEL bit selects the count source from the sub-clock oscillator and LOCO.

The RCKSEL bit is only used in normal operation mode. When the RCKSEL bit is set to 0, the time is counted with the sub-clock oscillator. When the bit is set to 1, the time is counted with LOCO.

For details on count source setting, see section 23.3.1. Outline of Initial Settings of Registers after Power On and section 23.3.2. Clock and Count Mode Setting Procedure. The count source must be selected only once before specifying the initial settings of the RTC registers at power on.

## 23.2.26 RFRL : Frequency Register L

Base address: RTC = 0x4008_3000

Offset address: 0x2C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | RFC[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | RFC[15:0] | Frequency Comparison Value<br>Write 0x00FF to this register when using the LOCO. | R/W |

RFRL is a register for controlling the prescaler when LOCO is selected.

The RTC time counter operates on a 128-Hz clock signal as the base clock. Therefore, when LOCO is selected, LOCO is divided by the prescaler to generate a 128-Hz clock signal. Set the frequency comparison value in the RFC[15:0] bits to generate a 128-Hz clock from the LOCO frequency. Before writing to RFC[15:0] after a cold start, write 0x0000 to the RFRH register.

A value from 0x0007 through 0x01FF can be specified as the frequency comparison value. If a value outside of this range is specified, the RTC does not operate correctly. Before writing to this register, be sure to stop the count operation through the setting of the START bit in RCR2. The operating frequency of the peripheral module clock and the LOCO should be such that the peripheral module clock is ≥ LOCO.

Calculation method of frequency comparison value:

RFC[15:0] = (LOCO clock frequency) / 128 - 1

When the LOCO frequency is 32.768 kHz, the RFRL register should be set to 0x00FF.

## 23.2.27 RFRH : Frequency Register H

Base address: RTC = 0x4008_3000

Offset address: 0x2A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | RFC16 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RFC16 | Write 0 before writing to the RFRL register after a cold start. | R/W |
| 15:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Before writing to RFRHL.RFC[15:0] after a cold start, write 0x0000 to the RFRH register.

## 23.2.28 RADJ : Time Error Adjustment Register

Base address: RTC = 0x4008_3000

Offset address: 0x2E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | PMADJ[1:0] | | ADJ[5:0] | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | ADJ[5:0] | Adjustment Value<br>These bits specify the adjustment value from the prescaler. | R/W |
| 7:6 | PMADJ[1:0] | Plus-Minus<br>0 0: Do not perform adjustment.<br>0 1: Adjustment is performed by the addition to the prescaler<br>1 0: Adjustment is performed by the subtraction from the prescaler<br>1 1: Setting prohibited. | R/W |

The RADJ register is used in both calendar count mode and binary count mode. Adjustment is performed by the addition to or subtraction from the prescaler or 64-Hz counter. If the Automatic Adjustment Enable (RCR2.AADJE) bit is 0, adjustment

is performed when writing to the RADJ. If the RCR2.AADJE bit is 1, adjustment is performed in the interval specified in the Automatic Adjustment Period Select (RCR2.AADJP) bit.

The current adjustment by software (disabling automatic adjustment) may be invalid if the following adjustment value is specified within 320 cycles of the count source after the register setting. To perform adjustment consecutively, wait for 320 cycles or more of the count source after the register setting, then specify the next adjustment value.

RADJ is updated in synchronization with the count source. When RADJ is modified, check that all the bits are updated before continuing with more processing. This register is set to 0x00 by an RTC software reset. The setting of this register is enabled only when the sub-clock oscillator is selected. When LOCO is selected, adjustment is not performed.

#### ADJ[5:0] bits (Adjustment Value)

The ADJ[5:0] bits specify the adjustment value (number of sub-clock cycles) from the prescaler.

#### PMADJ[1:0] bits (Plus-Minus)

The PMADJ[1:0] bits select whether the clock is set ahead or back depending on the error-adjustment value set in the ADJ[5:0] bits.

### 23.2.29 RTCCRn : Time Capture Control Register n (n = 0 to 2)

Base address: RTC = 0x4008_3000

Offset address: 0x40 + 0x02 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TCEN | — | TCNF[1:0] | | — | TCST | TCCT[1:0] | |
| Value after reset: | x | 0 | x | x | 0 | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | TCCT[1:0] | Time Capture Control<br>0 0: Do not detect events<br>0 1: Detect rising edge<br>1 0: Detect falling edge<br>1 1: Detect both edges | R/W |
| 2 | TCST | Time Capture Status<br>0: No event detected<br>1: Event detected*1 | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 5:4 | TCNF[1:0] | Time Capture Noise Filter Control<br>0 0: Turn noise filter off<br>0 1: Setting prohibited<br>1 0: Turn noise filter on (count source)<br>1 1: Turn noise filter on (count source by divided by 32) | R/W |
| 6 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | TCEN | Time Capture Event Input Pin Enable<br>0: Disable the RTCICn pin as the time capture event input pin<br>1: Enable the RTCICn pin as the time capture event input pin | R/W |

Note 1. Indicates that an event is detected. Writing 1 to this bit has no effect. Writing 0 sets this bit to 0.

The RTCCRn register is used both in calendar count mode and in binary count mode. RTCCR0, RTCCR1, and RTCCR2 control the RTCIC0, RTCIC1, and RTCIC2 pins respectively.

RTCCRn is updated in synchronization with the count source. When RTCCRn is modified, check that all the bits except the TCST bit are updated before continuing with additional processing. This register is cleared to 0x00 by an RTC software reset. When RTCICm is used as the time capture pin, VBTICTLR.VCHnIEN (n = 0 to 2) must be set to 1.

#### TCCT[1:0] bits (Time Capture Control)

The TCCT[1:0] bits control the edge detection of the time capture event input pins, RTCIC0, RTCIC1, and RTCIC2. The detection edge is selectable. The TCCT[1:0] bits must be set while the VBTICTLR.VCHnIEN bit is 1.

### TCST bit (Time Capture Status)

The TCST bit indicates that an event on the time capture event input pins, RTCIC0, RTCIC1, and RTCIC2,was detected. When the TCST bit is 0, no event is detected. When the TCST bit is 1, this bit indicates that an event was detected on the associated pin and the capture register is valid. When multiple events are detected, the capture time for the first event is retained.

The event is detected only during count operation (RCR2.START bit = 1). Before reading the capture register, make sure that this bit is set to 1.

Set the TCST bit while the TCCT[1:0] bits are 00b (no event is detected). The TCST bit is set to 0 in synchronization with the count source. When the TCST bit is set to 0, check that the bit is updated before continuing with additional processing.

### TCNF[1:0] bits (Time Capture Noise Filter Control)

The TCNF[1:0] bits control the noise filter of the time capture event input pins (RTCIC0, RTCIC1, and RTCIC2).

When the noise filter is on, the count source divided by 1 or divided by 32 is selectable. In this case, when the input level on the time capture event input pin matches three consecutive times at the set sampling period, the input level is determined.

Set the TCNF[1:0] bits while the TCCT[1:0] bits are 00b (no event is detected). When the noise filter is used, set the TCNF[1:0] bits, wait for 3 cycles of the specified sampling period, then set the TCCT[1:0] bits. Set the TCNF[1:0] bits when the VBTICTLR.VCHnIEN bit is 1.

### TCEN bit (Time Capture Event Input Pin Enable)

The TCEN bit enables or disables the time capture event input pins RTCIC0, RTCIC1, and RTCIC2.When the functions of the time capture event input pins are multiplexed, set VBTICTLR first. If the TCEN bit is set to 0, also set the TCCT[1:0] bits to 00b.

Before setting this bit to 1, be sure to set the count source setting bit (RCR4.RCKSEL), RTC time capture event enable bit (RCPE.RTCEN), port control setting bits (PmnPFS.PDR, and PmnPFS.PMR). For details on the port control setting bits (PmnPFS.PDR and PmnPFS.PMR), see section 19, I/O Ports.

## 23.2.30 RSECCPn : Second Capture Register n (n = 0 to 2) (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x52 + 0x10 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | SEC10[2:0] | | | SEC1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | SEC1[3:0] | 1-Second Capture<br>Capture value for the ones place of seconds. | R |
| 6:4 | SEC10[2:0] | 10-Second Capture<br>Capture value for the tens place of seconds. | R |
| 7 | — | The read value is undefined. | R |

RSECCPn is a read-only register that captures the RSECCNT value when a time capture event is detected.

The event detection times detected by the RTCIC0, RTCIC1, and RTCIC2 pins ares stored in the RSECCP0, RSECCP1, and RSECCP2 registers, respectively. This register is cleared to 0x00 by an RTC software reset. Before reading from this register, the time capture event detection should be stopped using the RTCCRn.TCCT[1:0] bits.

## 23.2.31 RMINCPn : Minute Capture Register n (n = 0 to 2) (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x54 + 0x10 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | MIN10[2:0] | | | MIN1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | MIN1[3:0] | 1-Minute Capture<br>Capture value for the ones place of minutes. | R |
| 6:4 | MIN10[2:0] | 10-Minute Capture<br>Capture value for the tens place of minutes. | R |
| 7 | — | The read value is undefined. | R |

RMINCPn is a read-only register that captures the RMINCNT value when a time capture event is detected.

The event detection times detected by the RTCIC0, RTCIC1, and RTCIC2 pins are stored in the RMINCP0, RMINCP1, and RMINCP2 registers, respectively.

This register is cleared to 0x00 by an RTC software reset. Before reading from this register, the time capture event detection should be stopped using the RTCCRn.TCCT[1:0] bits.

## 23.2.32 RHRCPn : Hour Capture Register n (n = 0 to 2) (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x56 + 0x10 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | PM | HR10[1:0] | | HR1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | HR1[3:0] | 1-Hour Capture<br>Capture value for the ones place of hours | R |
| 5:4 | HR10[1:0] | 10-Hour Capture<br>Capture value for the tens place of hours | R |
| 6 | PM | PM<br>0: AM<br>1: PM | R |
| 7 | — | The read value is undefined. | R |

RHRCPn is a read-only register that captures the RHRCNT value when a time capture event is detected.

The event detection times detected by the RTCIC0, RTCIC1, and RTCIC2 pins are stored in the RHRCP0, RHRCP1, and RHRCP2 registers, respectively.

The PM bit is only enabled when the RCR2.HR24 bit is 0 (in 12-hour mode).

This register is cleared to 0x00 by an RTC software reset. Before reading from this register, you must stop the time capture event detection using the RTCCRn.TCCT[1:0] bits.

## 23.2.33 RDAYCPn : Date Capture Register n (n = 0 to 2) (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x5A + 0x10 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | DATE10[1:0] | | DATE1[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | DATE1[3:0] | 1-Day Capture<br>Capture value for the ones place of days. | R |
| 5:4 | DATE10[1:0] | 10-Day Capture<br>Capture value for the tens place of days. | R |
| 7:6 | — | The read value is undefined. | R |

RDAYCPn is a read-only register that captures the RDAYCNT value when a time capture event is detected.

The event detection times detected by the RTCIC0, RTCIC1, and RTCIC2 pins are stored in the RDAYCP0, RDAYCP1, and RDAYCP2 registers, respectively.

This register is cleared to 0x00 by an RTC software reset. Before reading from this register, the time capture event detection should be stopped using the RTCCRn.TCCT[1:0] bits.

## 23.2.34 RMONCPn : Month Capture Register n (n = 0 to 2) (in Calendar Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x5C + 0x10 × n

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | MON10 | MON1[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | MON1[3:0] | 1-Month Capture<br>Capture value for the ones place of months. | R |
| 4 | MON10 | 10-Month Capture<br>Capture value for the tens place of months. | R |
| 7:5 | — | These bits are read as 0. | R |

RMONCPn is a read-only register that captures the RMONCNT value when a time capture event is detected.

The event detection times detected by the RTCIC0, RTCIC1, and RTCIC2 pins are stored in the RMONCP0, RMONCP1, and RMONCP2 registers, respectively.

This register is cleared to 0x00 by an RTC software reset. Before reading from this register, the time capture event detection should be stopped using the RTCCRn.TCCT[1:0] bits.

### 23.2.35 BCNTnCPm : BCNTn Capture Register m (n= 0 to 3, m = 0 to 2) (in Binary Count Mode)

Base address: RTC = 0x4008_3000

Offset address: 0x52 + 0x10 × m (BCNT0CPm)
0x54 + 0x10 × m (BCNT1CPm)
0x56 + 0x10 × m (BCNT2CPm)
0x5A + 0x10 × m (BCNT3CPm)

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

BCNTnCPm is a read-only register that captures the BCNTn value when a time capture event is detected. BCNT3CPm is assigned to the BCNTCPm[31:24] bits, BCNT2CPm is assigned to the BCNTCPm[23:16] bits, BCNT1CPm is assigned to the BCNTCPm[15:8] bits and BCNT0CPm is assigned to the BCNTCPm[7:0] bits. The event detection times detected by the RTCIC0, RTCIC1, and RTCIC2 pins are stored in the BCNTnCP0, BCNTnCP1, and BCNTnCP2 registers, respectively.

This register is cleared to 0x00 by an RTC software reset. Before reading from this register, you must stop the time capture event detection using the RTCCRn.TCCT[1:0] bits.

## 23.3 Operation

### 23.3.1 Outline of Initial Settings of Registers after Power On

After the power is turned on, perform the initial settings for the clock, count mode, time error adjustment, time, alarm, interrupts, and time capture.

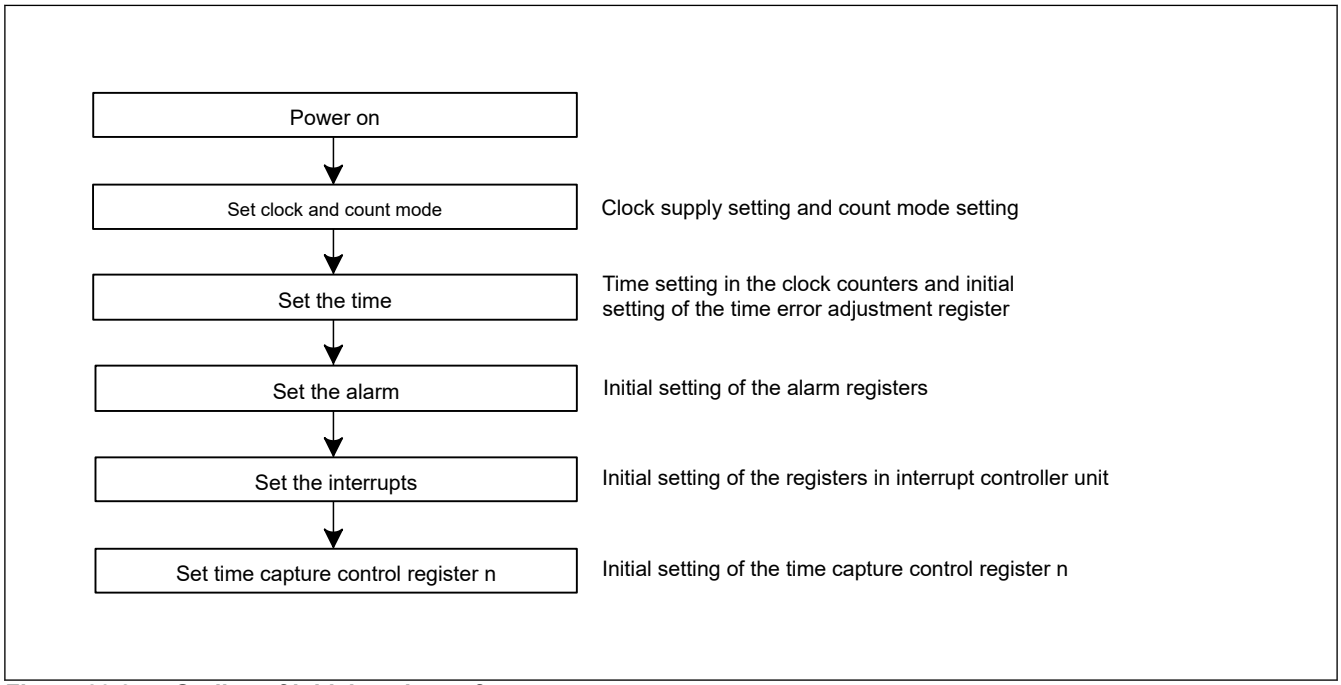
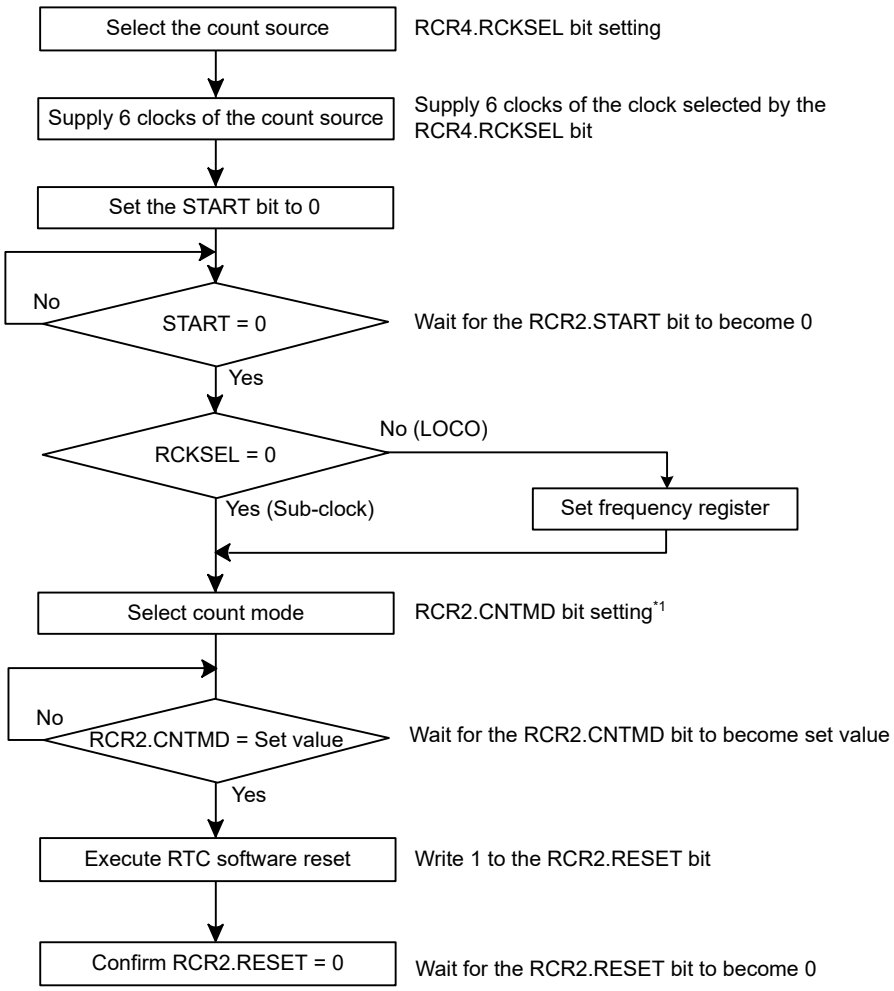



**Figure 23.2 Outline of initial settings after a power on**

### 23.3.2 Clock and Count Mode Setting Procedure

Figure 23.3 shows how to set the clock and the count mode.

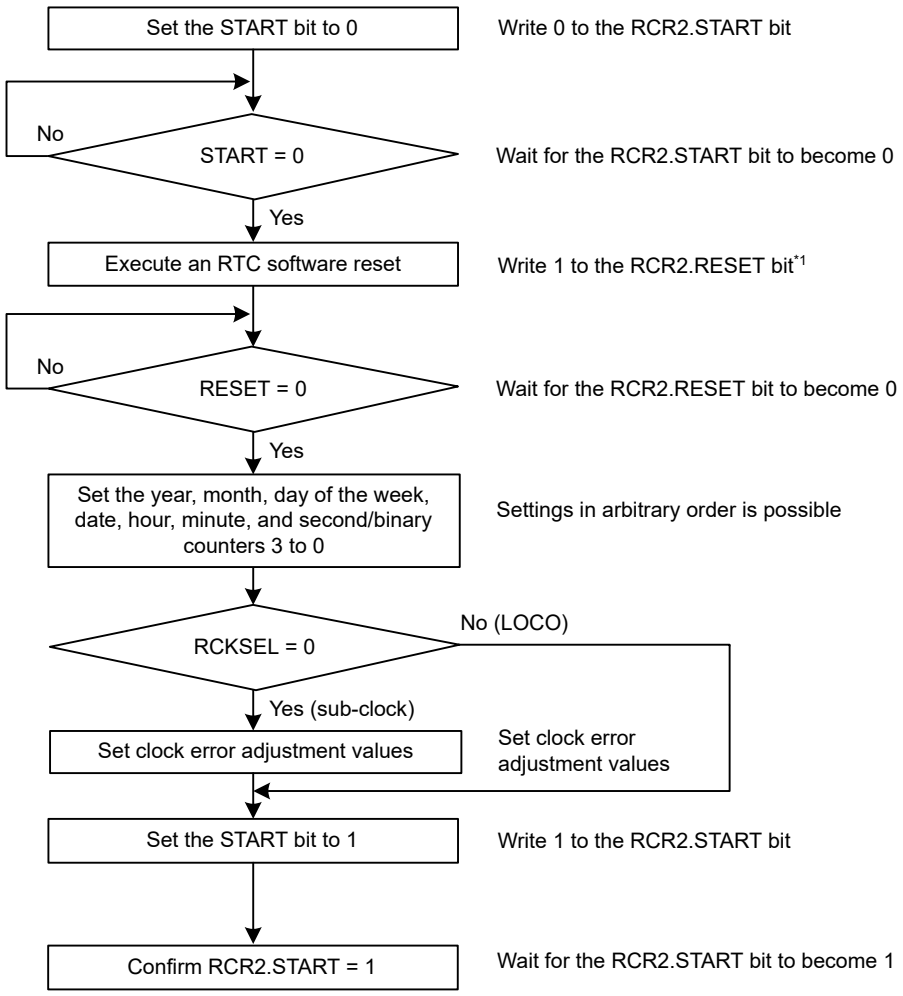



Note 1. This step is not required if the count mode is concurrently set by setting the START bit to 0. A value associated with the count mode setting must be written to the RCR2.CNTMD bit.

**Figure 23.3 Clock and count mode setting procedure**

### 23.3.3 Setting the Time

Figure 23.4 shows how to set the time.




Note 1. This step can be omitted in initial settings after power-on because an RTC software reset is executed in the clock setting procedure that is performed during initial settings at power-on.
Note 2. Set in any order.

**Figure 23.4 Setting the time**

### 23.3.4 30-Second Adjustment

Figure 23.5 shows how to execute a 30-second adjustment.

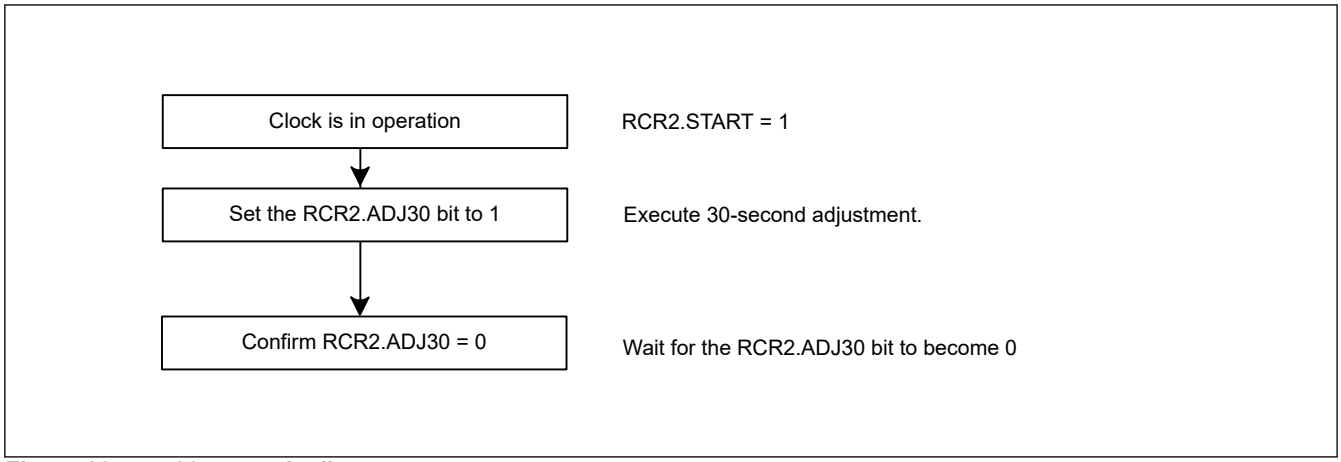
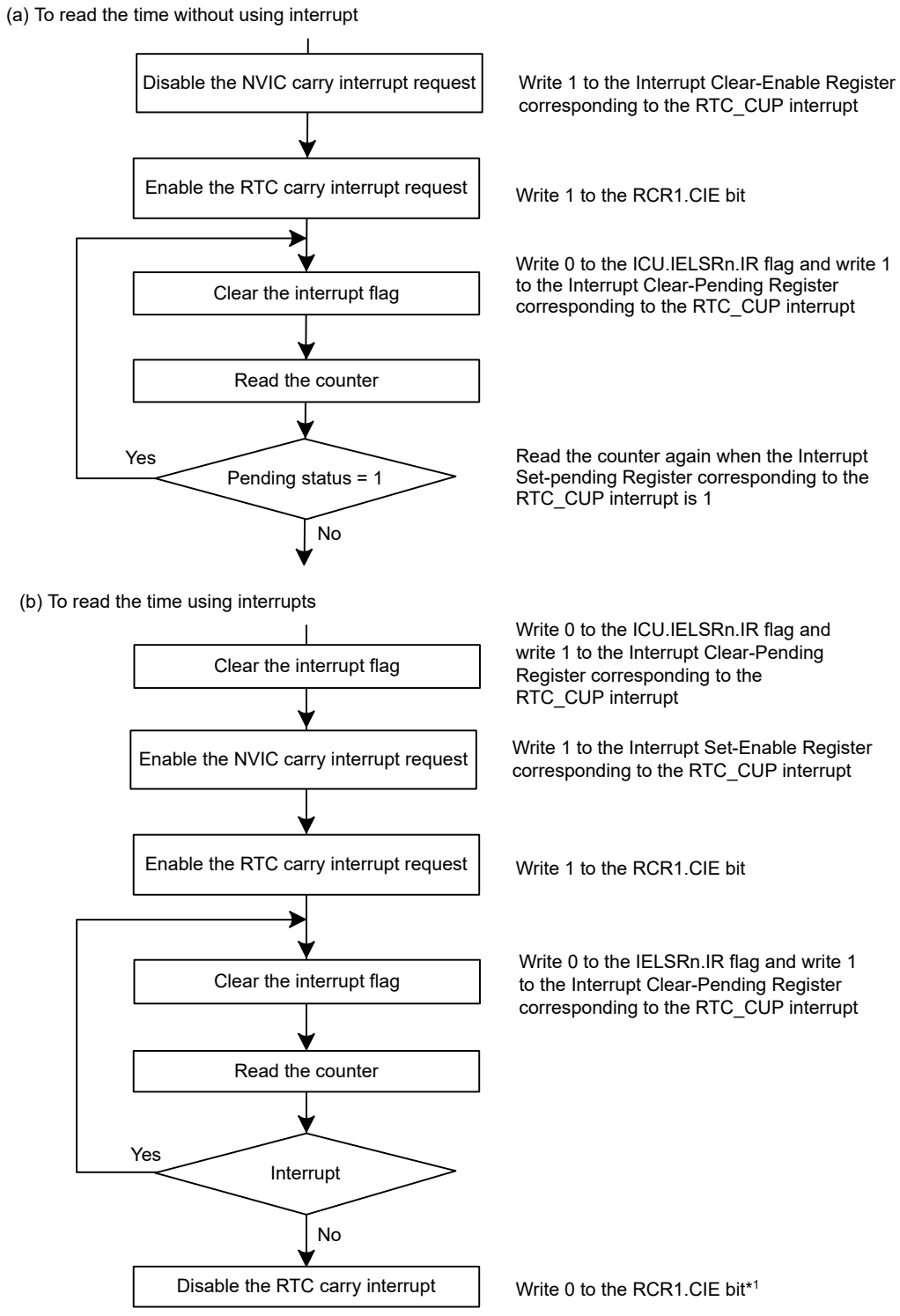



**Figure 23.5 30-second adjustment**

### 23.3.5 Reading 64-Hz Counter and Time

Figure 23.6 shows how to read a 64-Hz counter and time.




Note 1. Disable interrupts if required.

**Figure 23.6 Reading time**

If a carry occurs while the 64-Hz counter and time are read, the correct time is not obtained, therefore they must be read again. The procedure for reading the time without using interrupts is shown in (a) in Figure 23.6, and the procedure using carry interrupts is shown in (b). To keep the program simple, method (a) should be used in most cases.

### 23.3.6 Alarm Function

Figure 23.7 shows how to use the alarm function.

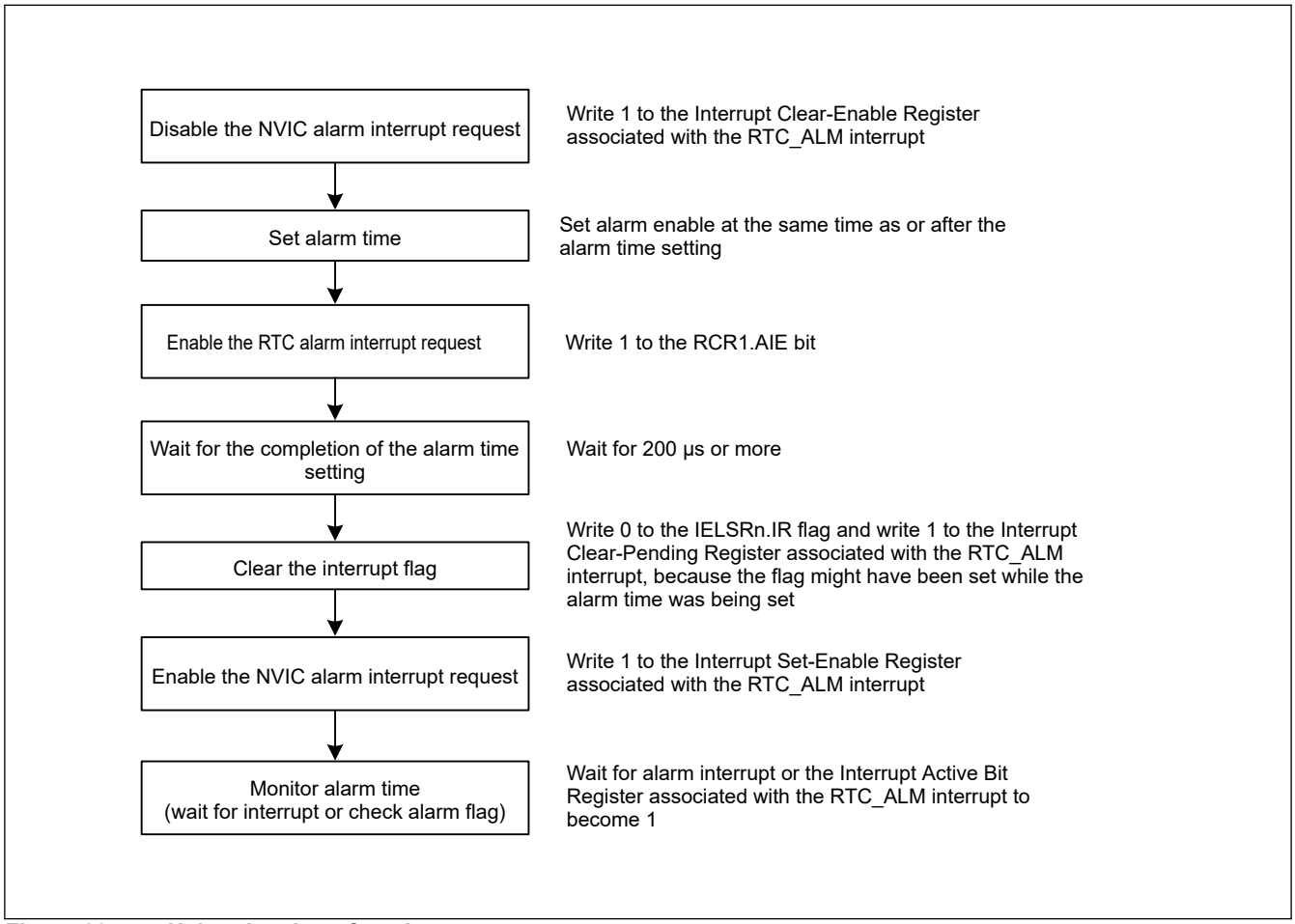



**Figure 23.7 Using the alarm function**

In calendar count mode, an alarm can be generated by any one of year, month, date, day-of-week, hour, minute or second, or any combination of those. Write 1 to the ENB bit in the alarm registers involved in the alarm setting, and set the alarm time in the lower bits. Write 0 to the ENB bit in registers not involved in the alarm setting.

In binary count mode, an alarm can be generated in any bit combination of 32 bits. Write 1 to the ENB bit of the Alarm Enable register associated with the target bit of the alarm, and set the alarm time in the alarm register. For bits that are not the target of the alarm, write 0 to the ENB bit of the Alarm Enable register.*1

For any of the ENB[31:0] bits that are set to 1, the bits in the corresponding positions in the binary counter (BCNT[31:0]) are compared with the values of the corresponding bits in the binary alarm registers*1. When all such bits match, the IR flag associated with the RTC_ALM interrupt is set to 1 and the corresponding bits in the Interrupt Set-Pending/Clear-Pending Registers are set to 1. Alarm detection can be confirmed by reading the Interrupt Set-Pending Register associated with the RTC_ALM interrupt, but an interrupt should be used in most cases. If 1 is set in the Interrupt Set-Enable Register associated with the RTC_ALM interrupt, an alarm interrupt is generated in the event of the alarm, enabling the alarm to be detected.

Writing 0 sets the IELSRn.IR flag associated with the RTC_ALM interrupt to 0. If interrupt is enabled, the Interrupt Set-Pending/Clear-Pending Register associated with the RTC_ALM interrupt is cleared automatically after exiting the interrupt handler. Otherwise, write 1 to the Interrupt Clear-Pending Register associated with the RTC_ALM interrupt to clear it.

When the counter and the alarm time match in a low power state, the MCU returns from the low power state.

Note 1. For any bits in the ENB bits that are set to 1, the values in the corresponding positions in the alarm registers from the following registers are compared with the corresponding bits of the counted values.

Counter registers: RSECCNT, RMINCNT, RHRCNT, RWKCNT, RDAYCNT, RMONCNT, RYRCNT

Alarm registers: RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, RMONAR, RYRAREN

### 23.3.7 Procedure for Disabling Alarm Interrupt

Figure 23.8 shows the procedure for disabling the enabled alarm interrupt request.

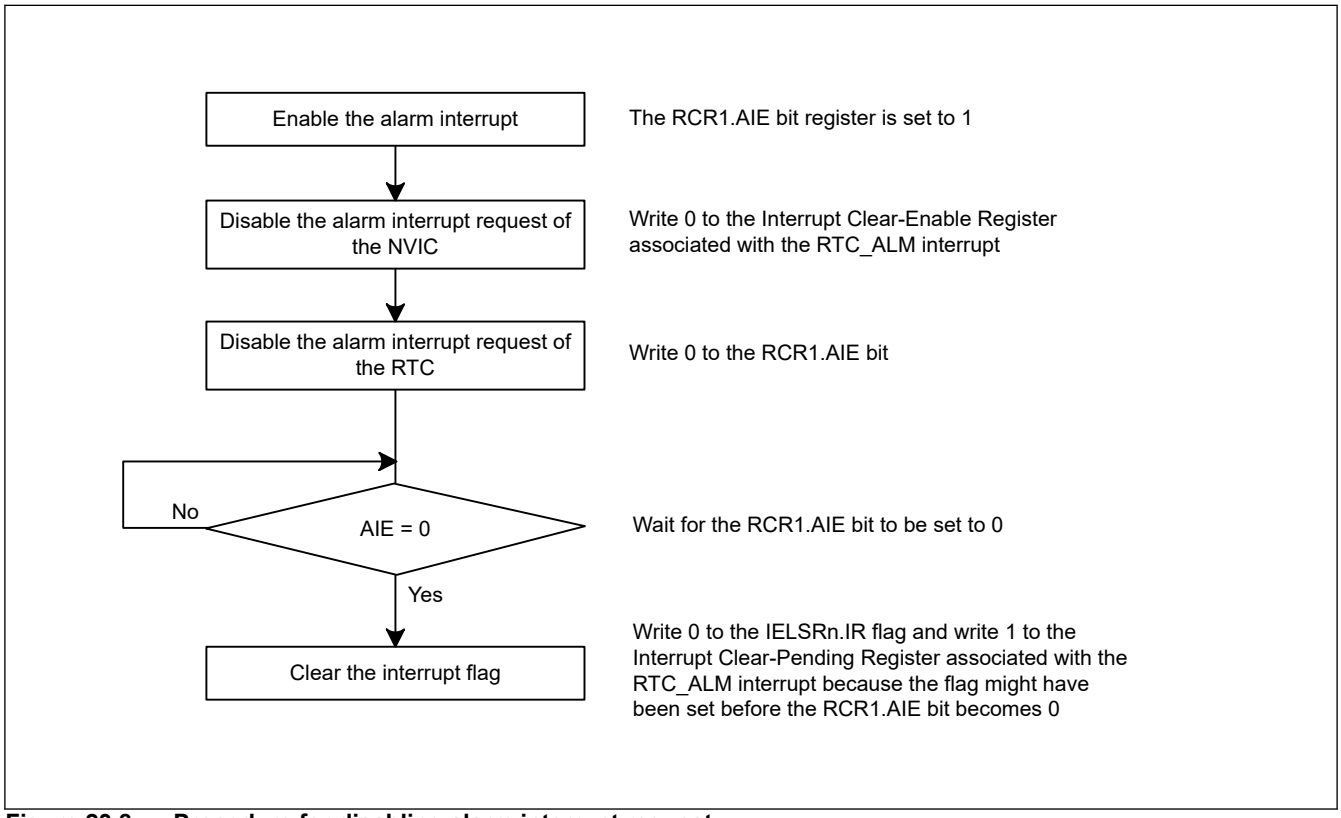



**Figure 23.8 Procedure for disabling alarm interrupt request**

### 23.3.8 Time Error Adjustment Function

The time error adjustment function is used to correct errors, running fast or slow, in the time caused by variation in the precision of oscillation by the sub-clock oscillator. Because 32768 cycles of the sub-clock oscillator constitute 1 second of operation when the sub-clock oscillator is selected, the clock runs fast if the sub-clock frequency is high and slow if the sub-clock frequency is low.

The time error adjustment functions include:

- Automatic adjustment
- Adjustment by software

Use the RCR2.AADJE bit to select automatic adjustment or adjustment by software.

#### 23.3.8.1 Automatic adjustment

Enable automatic adjustment by setting the RCR2.AADJE bit to 1.

Automatic adjustment is the addition or subtraction of the value counted by the prescaler to or from the value in the RADJ register every time the adjustment period selected by the RCR2.AADJP bit elapses.

##### (1) Example 1: Sub-clock oscillator running at 32.769 kHz

**Adjustment procedure**

When the sub-clock oscillator is running at 32.769 kHz, 1 second elapses every 32769 clock cycles. The RTC is meant to run at 32768 clock cycles, so the clock runs fast by 1 clock cycle every second. The time on the clock is fast by 60 clock cycles per minute, so adjustment can take the form of setting the clock back by 60 cycles every minute.

**Register settings when RCR2.CNTMD = 0:**

- RCR2.AADJP = 0 (adjustment every minute)
- RADJ.PMADJ[1:0] = 10b (adjustment is performed by the subtraction from the prescaler)
- RADJ.ADJ[5:0] = 60 (0x3C)

#### (2) Example 2: Sub-clock oscillator running at 32.766 kHz

**Adjustment procedure**

When the sub-clock oscillator is running at 32.766 kHz, 1 second elapses every 32766 clock cycles. The RTC is meant to run at 32768 clock cycles, so the clock runs slow by 2 clock cycles every second. The time on the clock is slow by 20 clock cycles every 10 seconds, so adjustment can take the form of setting the clock forward by 20 cycles every 10 seconds.

**Register settings when RCR2.CNTMD = 0:**

- RCR2.AADJP = 1 (adjustment every 10 seconds)
- RADJ.PMADJ[1:0] = 01b (adjustment is performed by the addition to the prescaler)
- RADJ.ADJ[5:0] = 20 (0x14)

#### (3) Example 3: Sub-clock oscillator running at 32.764 kHz

**Adjustment procedure**

When the sub-clock oscillator is running at 32.764 kHz, 1 second elapses on 32764 clock cycles. Because the RTC operates for 32768 clock cycles as 1 second, the clock is delayed for 4 clock cycles per second. In 8 seconds, the delay is 32 clock cycles, therefore correction can be made by advancing the clock 32 clock cycles every 8 seconds.

**Register settings when RCR2.CNTMD = 1:**

- RCR2.AADJP = 1 (adjustment every 8 seconds)
- RADJ.PMADJ[1:0] = 01b (adjustment is performed by the addition to the prescaler)
- RADJ.ADJ[5:0] = 32 (0x20)

### 23.3.8.2 Adjustment by software

Enable adjustment by software by setting the RCR2.AADJE bit to 0. Adjustment by software is the addition or subtraction of the value counted by the prescaler to or from the value in the RADJ register on execution of a write instruction to the RADJ register.

#### (1) Example 1: Sub-clock oscillator running at 32.769 kHz

**Adjustment procedure**

When the sub-clock oscillator is running at 32.769 kHz, 1 second elapses every 32769 clock cycles. The RTC is meant to run at 32768 clock cycles, so the clock runs fast by 1 clock cycle every second. The time on the clock is fast by 1 clock cycle per second, so adjustment can take the form of setting the clock back by 1 cycle every second.

**Register settings**

- RADJ.PMADJ[1:0] = 10b (adjustment is performed by the subtraction from the prescaler)
- RADJ.ADJ[5:0] = 1 (0x01)
  This is written to the RADJ register once per 1-second interrupt.

### 23.3.8.3 Procedure to change the mode of adjustment

When changing the mode of adjustment, change the value of the AADJE bit in RCR2 after setting the RADJ.PMADJ[1:0] bits to 00b (adjustment is not performed).

To change adjustment by software to automatic adjustment:

1. Set the RADJ.PMADJ[1:0] bits to 00b (adjustment is not performed).
2. Set the RCR2.AADJE bit to 1 (automatic adjustment is enabled).

3. Use the RCR2.AADJP bit to select the period of adjustment.
4. In RADJ, set the PMADJ[1:0] bits for addition or subtraction and the ADJ[5:0] bits to the value for use in time error adjustment.

To change automatic adjustment to adjustment by software:

1. Set the RADJ.PMADJ[1:0] bits to 00b (adjustment is not performed).
2. Set the RCR2.AADJE bit to 0 (adjustment by software is enabled).
3. Proceed with the adjustment by setting the RADJ.PMADJ[1:0] bits for addition or subtraction and the RADJ.ADJ[5:0] bits to the value for use in time error adjustment at the wanted time. After that, the time is adjusted every time a value is written to the RADJ register.

#### 23.3.8.4 Procedure to stop adjustment

Stop the adjustment by setting the RADJ.PMADJ[1:0] bits to 00b (adjustment is not performed).

### 23.3.9 Capturing the time

The RTC is capable of storing the month, date, hour, minute and second/binary counters 3 to 0 by detecting an edge of a signal on a time capture event input pin in calendar count mode or binary count mode .

A noise filter can also be used on a time capture event input pin. If the noise filter is enabled, the RTCCRn.TCST bit is set to 1 when the input level on the pin matches three times.

The noise filter can be switched on or off for each of the time capture event input pins.Set VBTICTLR.VCHnIEN (n = 0 to 2) to 1 to enable the RTCICn input. Operation when the noise filter is off is shown in Figure 23.9 and operation when the noise filter is on is shown in Figure 23.10.

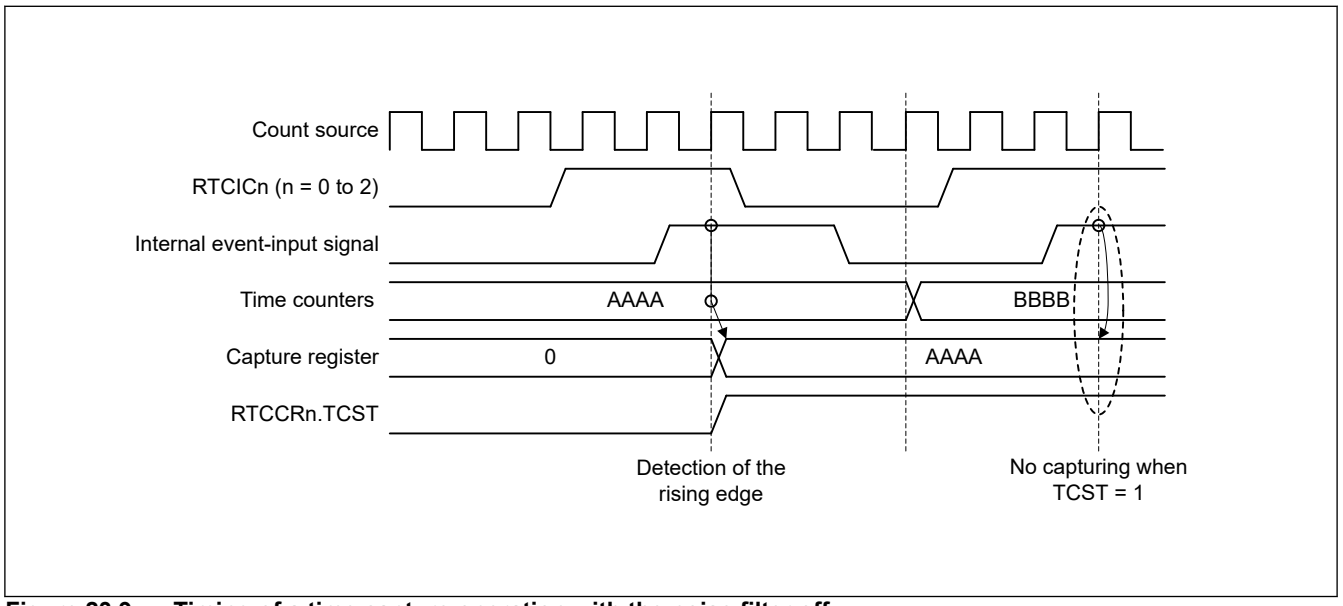



**Figure 23.9 Timing of a time capture operation with the noise filter off**

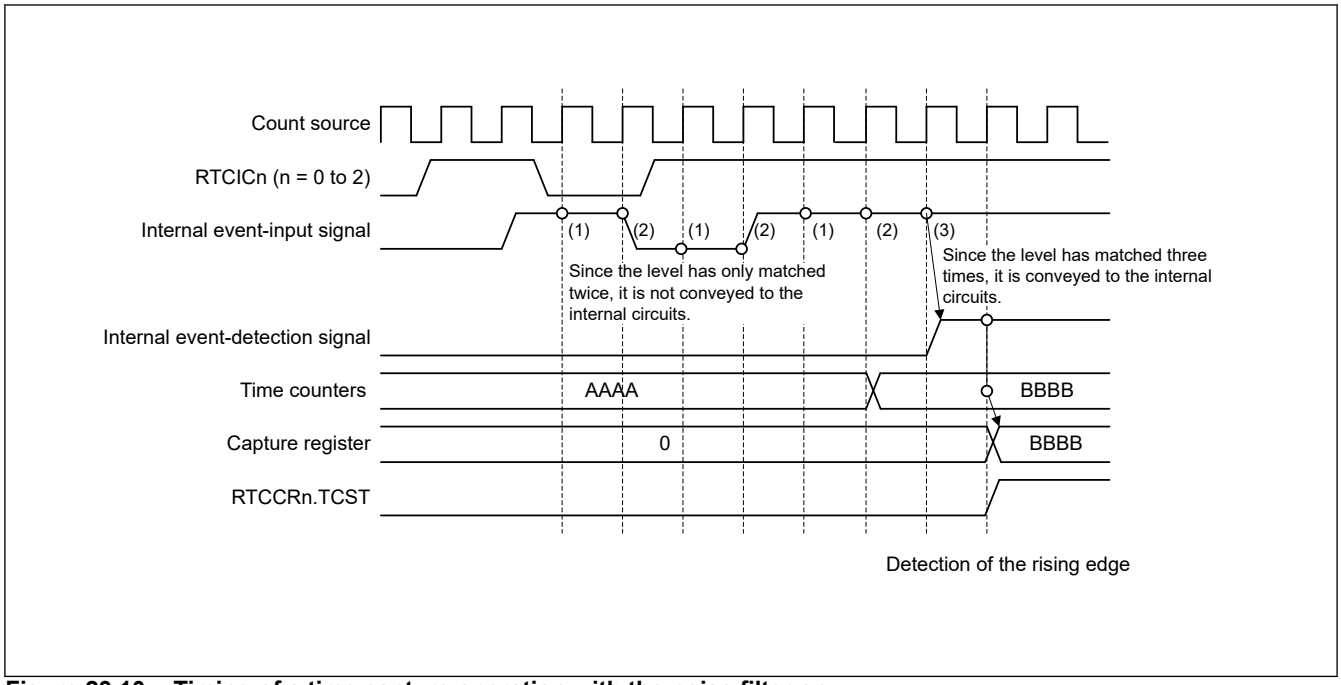



**Figure 23.10 Timing of a time capture operation with the noise filter on**

## 23.4 Interrupt Sources

The RTC has three interrupt sources, as listed in Table 23.3.

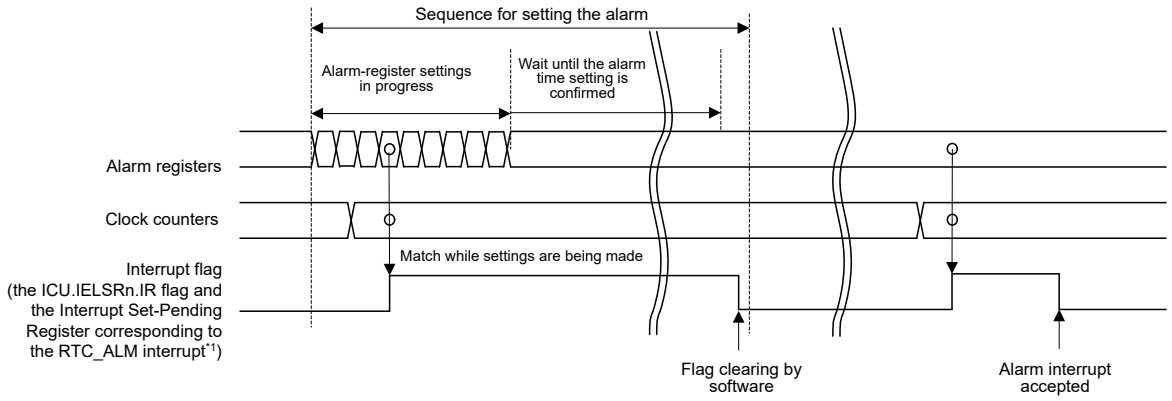
**Table 23.3 RTC interrupt sources**

| Name | Interrupt source |
|---|---|
| RTC_ALM | Alarm interrupt |
| RTC_PRD | Periodic interrupt |
| RTC_CUP | Carry interrupt |

### (1) Alarm interrupt (RTC_ALM)

This interrupt is generated based on the comparison result between the alarm registers and RTC counters. For details, see section 23.3.6. Alarm Function.

Because there is a possibility that the interrupt flag might be set to 1 when the settings of the alarm registers match the clock counters, wait for the alarm time settings to be confirmed and clear the IELSRn.IR flag and the interrupt Set-Pending Register associated with the RTC_ALM interrupt to 0 again after modifying values of the alarm registers. After the interrupt flag for the alarm interrupt is set to 1 and the state is returned to mismatching of the alarm registers and clock counters, the flag is not 1 again until there is another match or the values of the alarm registers are modified again.




Note 1. See section 13, Interrupt Controller Unit (ICU) for details on the associated interrupt vector number.

**Figure 23.11 Timing for the alarm interrupt (RTC_ALM)**

### (2) Periodic interrupt (RTC_PRD)

This interrupt is generated at intervals of 2 seconds, 1 second, 1/2 second, 1/4 second, 1/8 second, 1/16 second, 1/32 second, 1/64 second, 1/128 second, or 1/256 second. The interrupt interval can be selected in the RCR1.PES[3:0] bits.

### (3) Carry interrupt (RTC_CUP)

This interrupt is generated when a carry to the second counter/binary counter 0 occurred or a carry to the R64CNT counter occurred during read access to the 64-Hz counter.

Figure 23.12 shows the timing of the carry interrupt (RTC_CUP).

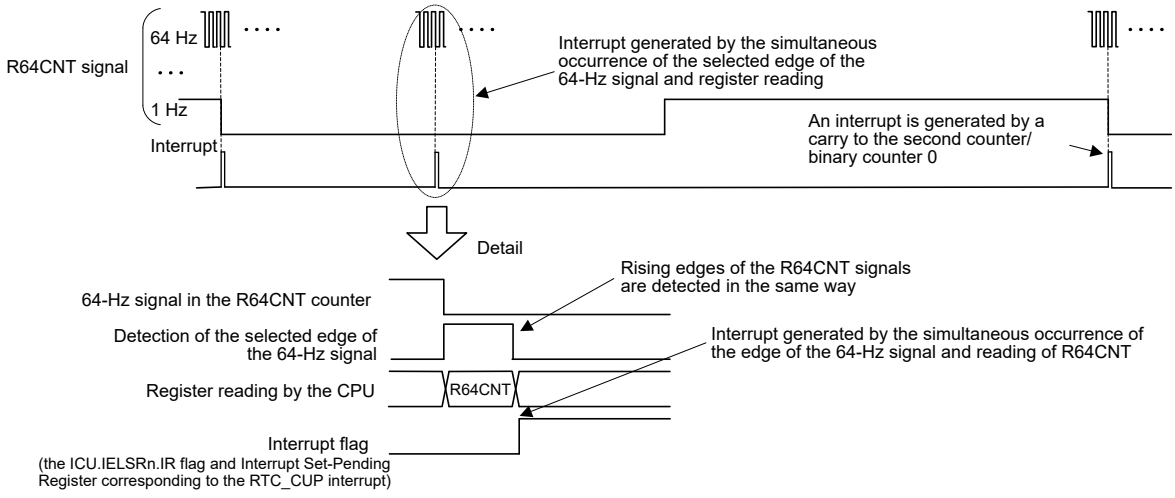



**Figure 23.12 Timing for the carry interrupt (RTC_CUP)**

## 23.5 Event Link Output

The RTC generates periodic event output (RTC_PRD) event signal for the ELC that can be used to initiate operations by other modules selected in advance.

The periodic event signal is output at the interval selected from 1/256, 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2, 1, and 2 seconds by setting the RCR1.PES[3:0] bits.

The event generation period immediately after the event generation is selected is not guaranteed.

Note: If event linking from the RTC is used, only set the ELC after setting the RTC, for example initialization and time settings. Setting the RTC after the ELC can lead to output of unexpected event signals.

### 23.5.1 Interrupt Handling and Event Linking

The RTC has a bit to enable or disable periodic interrupts. An interrupt request signal is output to the CPU when an interrupt source is generated while the associated enable bit is enabled.

In contrast, an event link output signal is sent to other modules as an event signal through the ELC when an interrupt source is generated, regardless of the setting of the associated interrupt enable bit.

Note: Although alarm and periodic interrupts can still be output during Software Standby or Deep Software Standby mode, the periodic event signals for the ELC are not output.

## 23.6 Usage Notes

### 23.6.1 Register Writing during Counting

The following registers should not be written to during counting, that is, while the RCR2.START bit is 1:

- RSECCNT/BCNT0
- RMINCNT/BCNT1
- RHRCNT/BCNT2
- RDAYCNT
- RWKCNT/BCNT3
- RMONCNT
- RYRCNT
- RCR1.RTCOS
- RCR2.RTCOE
- RCR2.HR24
- RFRL

The counter should be stopped before writing to any of these registers.

### 23.6.2 Use of Periodic Interrupts

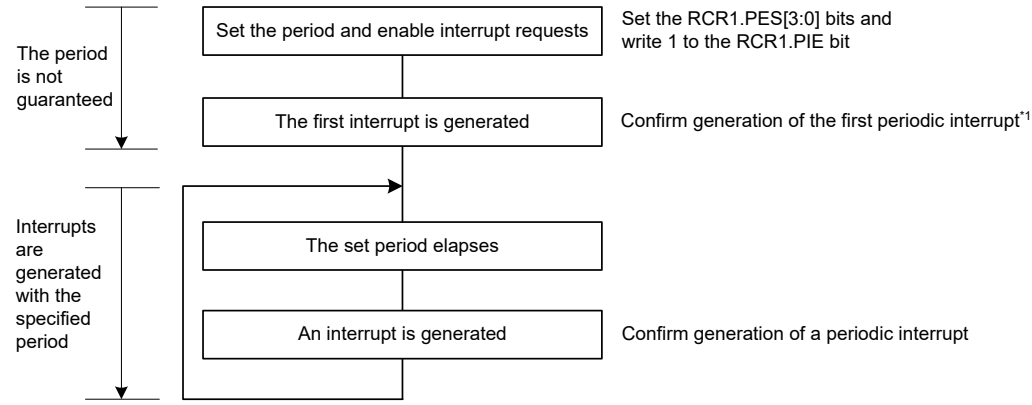
Figure 23.13 shows the procedure for using periodic interrupts.

The generation and period of the periodic interrupt can be changed by setting the RCR1.PES[3:0] bits. However, because the prescaler R64CNT and RSECCNT/BCNT0 are used to generate interrupts, the interrupt period is not guaranteed immediately after setting the RCR1.PES[3:0] bits.

In addition, any of the following operation can affect the interrupt period:

- Stopping/restarting or resetting counter operation
- Reset by RTC software
- 30-second adjustment by changing the RCR2 value

When the time error adjustment function is used, the interrupt generation period after adjustment is added or subtracted based on the adjustment value.




Note 1. When an interrupt generation period changes while the periodic interrupt is used, an interrupt might be generated at the completion of the setting. If the interrupt is generated immediately after the setting, the period is not guaranteed for two interrupts including the current interrupt.

**Figure 23.13 Using the periodic interrupt function**

### 23.6.3 RTCOUT (1-Hz/64-Hz) Clock Output

Stopping/restarting or resetting counter operation, reset by RTC software, and the 30-second adjustment by changing the RCR2 value affects the period of RTCOUT (1-Hz/64-Hz) output. When the time error adjustment function is used, the period of RTCOUT (1-Hz/64-Hz) output after adjustment is added or subtracted based on the adjustment value.

### 23.6.4 Transitions to Low Power Modes after Setting Registers

A transition to a low power state (Software Standby mode, Deep Software Standby mode, or battery backup state) during a write to an RTC register might corrupt the value of the register. After setting the register, confirm that the setting is in place before initiating a transition to a low power state.

### 23.6.5 Notes on Writing to and Reading from Registers

- When reading a counter register such as the second counter after writing to the counter register, follow the procedure in section 23.3.5. Reading 64-Hz Counter and Time.
- The value written to the count registers, alarm registers, year alarm enable register, bits RCR2.AADJE, AADJP, and HR24, RCR4 register, or frequency register is reflected when fourth read operations are performed after writing.
- The values written to the RCR1.CIE, RCR1.RTCOS, and RCR2.RTCOE bits can be read immediately after writing.
- To read the value from the timer counter after returning from a reset or a period in Software Standby mode, Deep Software Standby mode, or battery backup state, wait for 1/128 second while the clock is operating (RCR2.START bit = 1).
- After a reset is generated, write to the RTC register after 6 cycles of the count source clock have elapsed.

### 23.6.6 Changing the Count Mode

When changing the count mode (calendar count mode/binary count mode), set the RCR2.START bit to 0, stop the counting operation, then start it again from the initial setting. For details on the initial setting, see section 23.3.1. Outline of Initial Settings of Registers after Power On.

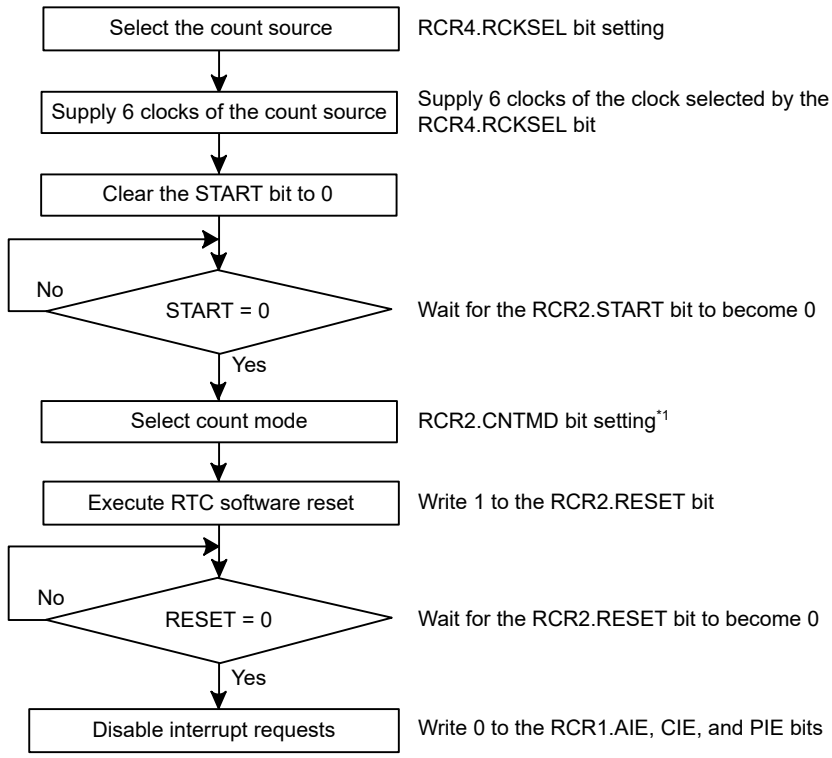
### 23.6.7 Initialization Procedure When the RTC Is Not to Be Used

Registers in the RTC are not initialized by a reset. Depending on the initial state, the generation of an unintentional interrupt request or operation of the counter might lead to increased power consumption.

For applications that do not require a realtime clock, initialize the registers by following the initialization procedure shown in Figure 23.14.

Alternatively, when the sub-clock oscillator is not used as the system clock or realtime clock, the counter can be stopped by writing 0 (sub-clock oscillator is selected) to the RCR4.RCKSEL bit and stopping the sub-clock oscillator. To stop the sub-clock oscillator, write 1 to the SOSCCR.SOSTP bit.

For details on the setting of the SOSCCR.SOSTP bit, see section 8, Clock Generation Circuit.




Note 1. This step is not necessary if the count mode is concurrently set by setting the START bit to 0.

**Figure 23.14 Initialization procedure**

## 23.6.8 When Switching Source Clock

When switching a clock source by changing SCKSCR.CKSEL[2:0], the clock output from the selector stops for 4 cycles of the switched clock. If the RTC periodical interrupt or RTC periodical event output was generated at this time, the interrupt or event is invalid.

# 24. Watchdog Timer (WDT)

## 24.1 Overview

The Watchdog Timer (WDT) is a 14-bit down counter that can be used to reset the MCU when the counter underflows because the system has run out of control and is unable to refresh the WDT. In addition, the WDT can be used to generate a non-maskable interrupt or an underflow interrupt.

Table 24.1 lists the WDT specifications and Figure 24.1 shows a block diagram.

**Table 24.1 WDT specifications**

| Parameter | Specifications |
|---|---|
| Count source*1 | Peripheral clock (PCLKB) |
| Clock division ratio | Division by 4, 64, 128, 512, 2048, or 8192 |
| Counter operation | Counting down using a 14-bit down-counter |
| Condition for starting the counter | • Auto start mode: Counting automatically starts after a reset or after an underflow or refresh error occurs<br>• Register start mode: Counting is started with a refresh by writing to the WDTRR register<br>• Only secure developer can select Auto-start mode or Register-start mode |
| Conditions for stopping the counter | • Reset (the down-counter and other registers return to their initial values)<br>• A counter underflows or a refresh error is generated |
| Window function | Window start and end positions can be specified (refresh-permitted and refresh-prohibited periods) |
| Watchdog timer reset sources | • Down-counter underflows<br>• Refreshing outside the refresh-permitted period (refresh error) |
| Non-maskable interrupt/interrupt sources | • Down-counter underflows<br>• Refreshing outside the refresh-permitted period (refresh error) |
| Reading of the counter value | The down-counter value can be read by the WDTSR register |
| Event link function (output) | • Down-counter underflow event output<br>• Refresh error event output |
| Output signal (internal signal) | • Reset output<br>• Interrupt request output<br>• Sleep-mode count stop control output |
| TrustZone Filter | Security attribution can be set |

Note 1. Satisfy the frequency of the peripheral module clock (PCLKB) ≥ 4 × (the frequency of the count clock source after division).

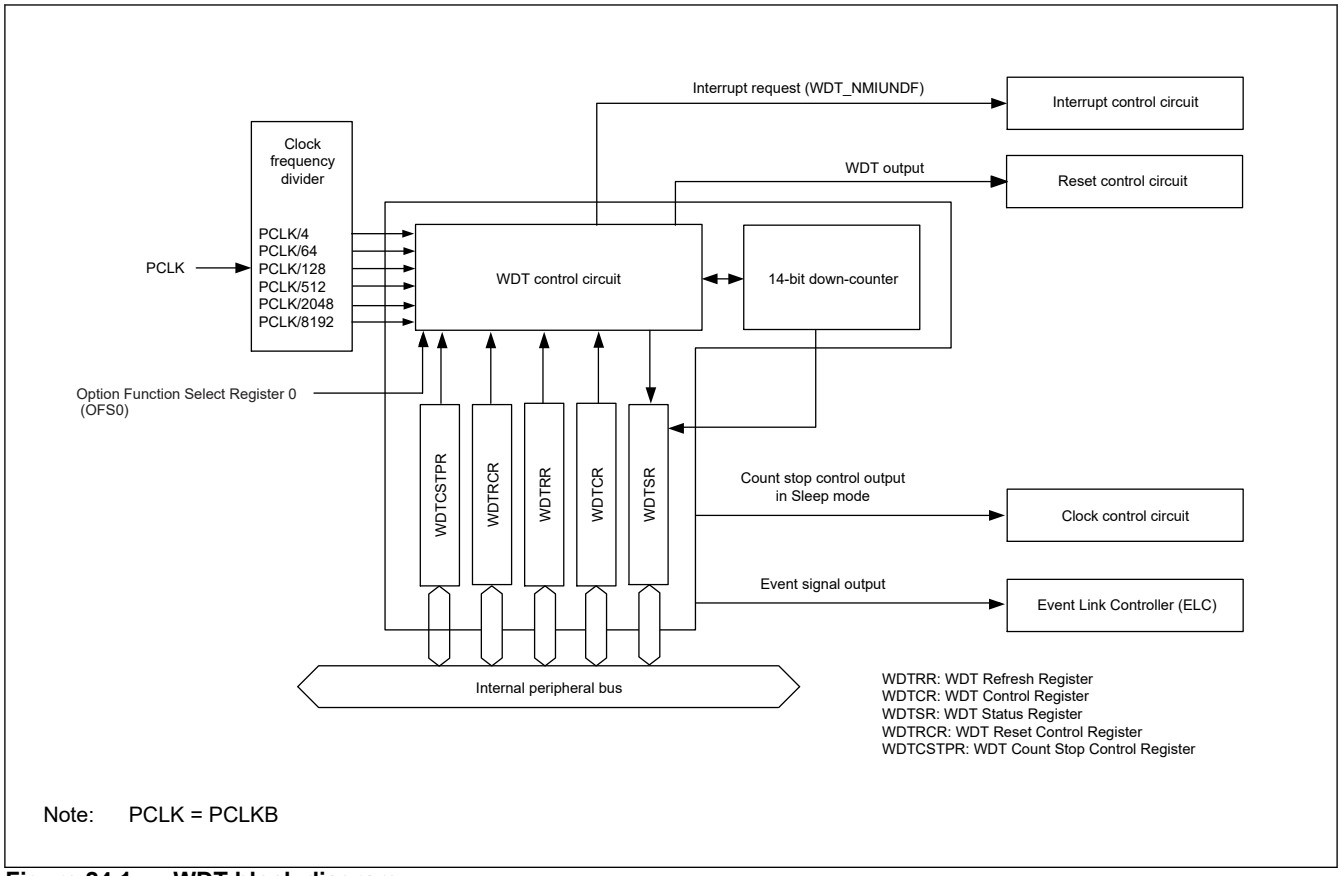



**Figure 24.1 WDT block diagram**

## 24.2 Register Descriptions

### 24.2.1 WDTRR : WDT Refresh Register

Base address: WDT = 0x4008_3400

Offset address: 0x00

| Bit position: | 7 | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | The down-counter is refreshed by writing 0x00 and then writing 0xFF to this register. | R/W |

The WDTRR register refreshes the down-counter of the WDT.

The down-counter of the WDT is refreshed by writing 0x00 and then writing 0xFF to WDTRR register (refresh operation) within the refresh-permitted period.

After the down-counter is refreshed, it starts counting down from the value selected by setting the WDT Timeout Period Select bits (OFS0.WDTTOPS[1:0]) in the Option Function Select Register 0 in auto start mode. In register start mode, counting down starts from the value selected by setting the Timeout Period Select bits (WDTCR.TOPS[1:0]) in the WDT Control Register.

When 0x00 is written, the read value is 0x00. When a value other than 0x00 is written, the read value is 0xFF. For details of the refresh operation, see section 24.3.3. Refresh Operation.

## 24.2.2 WDTCR : WDT Control Register

Base address: WDT = 0x4008_3400

Offset address: 0x02

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | RPSS[1:0] | | — | — | RPES[1:0] | | CKS[3:0] | | | | — | — | TOPS[1:0] | |
| Value after reset: | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | TOPS[1:0] | Timeout Period Select<br>0 0: 1024 cycles (0x03FF)<br>0 1: 4096 cycles (0x0FFF)<br>1 0: 8192 cycles (0x1FFF)<br>1 1: 16384 cycles (0x3FFF) | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7:4 | CKS[3:0] | Clock Division Ratio Select<br>0x1: PCLKB/4<br>0x4: PCLKB/64<br>0xF: PCLKB/128<br>0x6: PCLKB/512<br>0x7: PCLKB/2048<br>0x8: PCLKB/8192<br>Others: Setting prohibited | R/W |
| 9:8 | RPES[1:0] | Window End Position Select<br>0 0: 75%<br>0 1: 50%<br>1 0: 25%<br>1 1: 0% (do not specify window end position). | R/W |
| 11:10 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13:12 | RPSS[1:0] | Window Start Position Select<br>0 0: 25%<br>0 1: 50%<br>1 0: 75%<br>1 1: 100% (do not specify window start position). | R/W |
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |

The WDTCR register is used to set the clock division ratio, and window start and end positions for refresh, and the timeout period until the down-counter underflows in register start mode.

Some constraints apply to writes to the WDTCR register. For details, see section 24.3.2. Controlling Writes to the WDTCR, WDTRCR, and WDTCSTPR Registers.

In auto start mode, the settings in the WDTCR register are disabled, and the settings in the Option Function Select Register 0 (OFS0) are enabled. The settings for the WDTCR register can also be made in the OFS0 register. For details, see section 24.3.8. Association between Option Function Select Register 0 (OFS0) and WDT Registers.

### TOPS[1:0] bits (Timeout Period Select)

The TOPS[1:0] bits select the timeout period, the period until the down-counter underflows, from 1024, 4096, 8192, and 16384 cycles, taking the divided clock specified in the CKS[3:0] bits as 1 cycle. After the down-counter is refreshed, the combination of the CKS[3:0] and TOPS[1:0] bits determines the number of PCLKB cycles until the counter underflows.

Table 24.2 lists the relationship between the CKS[3:0] and TOPS[1:0] bit settings, the timeout period, and the number of PCLKB cycles.

**Table 24.2 Timeout period settings**

| CKS[3:0] bits | TOPS[1:0] bits | Clock division ratio | Timeout period (number of cycles) | PCLKB clock cycles |
|---|---|---|---|---|
| 0x1 | 00b | PCLKB/4 | 1024 | 4096 |
| | 01b | | 4096 | 16384 |
| | 10b | | 8192 | 32768 |
| | 11b | | 16384 | 65536 |
| 0x4 | 00b | PCLKB/64 | 1024 | 65536 |
| | 01b | | 4096 | 262144 |
| | 10b | | 8192 | 524288 |
| | 11b | | 16384 | 1048576 |
| 0xF | 00b | PCLKB/128 | 1024 | 131072 |
| | 01b | | 4096 | 524288 |
| | 10b | | 8192 | 1048576 |
| | 11b | | 16384 | 2097152 |
| 0x6 | 00b | PCLKB/512 | 1024 | 524288 |
| | 01b | | 4096 | 2097152 |
| | 10b | | 8192 | 4194304 |
| | 11b | | 16384 | 8388608 |
| 0x7 | 00b | PCLKB/2048 | 1024 | 2097152 |
| | 01b | | 4096 | 8388608 |
| | 10b | | 8192 | 16777216 |
| | 11b | | 16384 | 33554432 |
| 0x8 | 00b | PCLKB/8192 | 1024 | 8388608 |
| | 01b | | 4096 | 33554432 |
| | 10b | | 8192 | 67108864 |
| | 11b | | 16384 | 134217728 |

### CKS[3:0] bits (Clock Division Ratio Select)

The CKS[3:0] bits specify the division ratio of the clock used for the down-counter. The division ratio can be selected from the PCLKB divided by 4, 64, 128, 512, 2048, and 8192. Combined with the TOPS[1:0] bit setting, this allows the WDT to be configured to a count period between 4096 and 134217728 PCLKB clock cycles.

### RPES[1:0] bits (Window End Position Select)

The RPES[1:0] bits specify the window end position that indicates the refresh-permitted period. 75%, 50%, 25%, or 0% of the timeout period can be selected for the window end position. Set the window end position to a value less than the value for the window start position (window start position > window end position). If the window start position is set to a value less than or equal to the window end position, the window start position setting is enabled and the window end position is set to 0%.

### RPSS[1:0] bits (Window Start Position Select)

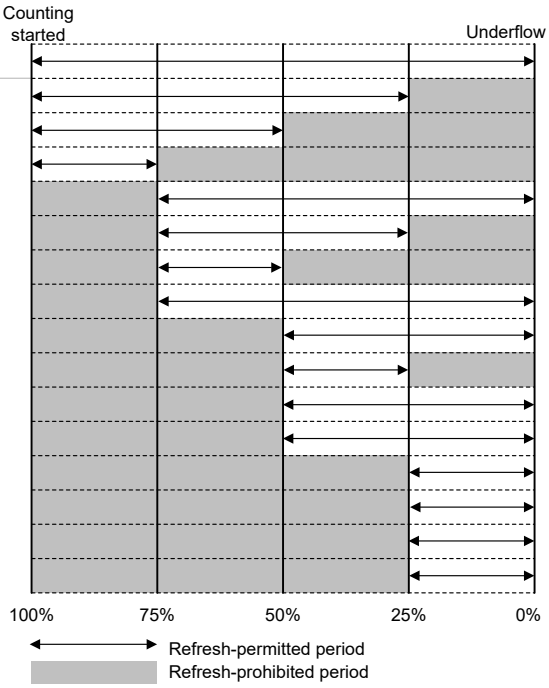
The RPSS[1:0] bits specify the window start position that indicates the refresh-permitted period. 100%, 75%, 50%, or 25% of the timeout period can be selected for the window start position. Set the window start position to a value greater than the value for the window end position. If the window start position is set to a value less than or equal to the window end position, the window start position setting is enabled and the window end position is set to 0%.

Table 24.3 lists the counter values for the window start and end positions, and Figure 24.2 shows the refresh-permitted period set in the RPSS[1:0], RPES[1:0], and TOPS[1:0] bits.

**Table 24.3 Relationship between the timeout period and window start and end counter values**

| TOPS[1:0] | Timeout period | | Window start and end counter value | | | |
|---|---|---|---|---|---|---|
| | Cycles | Counter value | 100% | 75% | 50% | 25% |
| 00b | 1024 | 0x03FF | 0x03FF | 0x02FF | 0x01FF | 0x00FF |
| 01b | 4096 | 0x0FFF | 0x0FFF | 0x0BFF | 0x07FF | 0x03FF |
| 10b | 8192 | 0x1FFF | 0x1FFF | 0x17FF | 0x0FFF | 0x07FF |
| 11b | 16384 | 0x3FFF | 0x3FFF | 0x2FFF | 0x1FFF | 0x0FFF |

| RPSS[1:0] bits | | RPES[1:0] bits | | Window | |
|---|---|---|---|---|---|
| b13 | b12 | b9 | b8 | Start (%) | End (%) |
| 1 | 1 | 1 | 1 | 100 | 0 |
| | | 1 | 0 | | 25 |
| | | 0 | 1 | | 50 |
| | | 0 | 0 | | 75 |
| 1 | 0 | 1 | 1 | 75 | 0 |
| | | 1 | 0 | | 25 |
| | | 0 | 1 | | 50 |
| | | 0 | 0 | | 75 |
| 0 | 1 | 1 | 1 | 50 | 0 |
| | | 1 | 0 | | 25 |
| | | 0 | 1 | | 50 |
| | | 0 | 0 | | 75 |
| 0 | 0 | 1 | 1 | 25 | 0 |
| | | 1 | 0 | | 25 |
| | | 0 | 1 | | 50 |
| | | 0 | 0 | | 75 |




Note: If window end position setting ≥ window start position setting, the window end position setting is set to 0%.

**Figure 24.2 RPSS[1:0] and RPES[1:0] bits setting and refresh-permitted period**

## 24.2.3 WDTSR : WDT Status Register

Base address: WDT = 0x4008_3400

Offset address: 0x04

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | REFEF | UNDF | CNTVAL[13:0] | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 13:0 | CNTVAL[13:0] | Down-Counter Value<br>Value counted by the down-counter | R |
| 14 | UNDFF | Underflow Flag<br>0: No underflow occurred<br>1: Underflow occurred | R/W*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15 | REFEF | Refresh Error Flag<br>0: No refresh error occurred<br>1: Refresh error occurred | R/W*1 |

Note 1. Only 0 can be written to clear the flag.

The WDTSR register indicates the counter value of the down-counter and the status of whether an underflow or refresh error occurred in the down-counter.

### CNTVAL[13:0] bits (Down-Counter Value)

Read the CNTVAL[13:0] bits to confirm the value of the down-counter. The read value might differ from the actual count by 1.

### UNDFF flag (Underflow Flag)

Read the UNDFF flag to confirm whether an underflow occurred in the counter. A value of 1 indicates that the down counter underflowed. Write 0 to the flag to set the value to 0. Writing 1 has no effect.

Clearing of the UNDFF flag takes (N+1) PCLKB cycles. In addition, clearing of the flag is ignored for (N+1) PCLKB cycles after an underflow. N is specified in the WDTCR.CKS[3:0] bits as follows:

- When WDTCR.CKS[3:0] = 0x1, N = 4
- When WDTCR.CKS[3:0] = 0x4, N = 64
- When WDTCR.CKS[3:0] = 0xF, N = 128
- When WDTCR.CKS[3:0] = 0x6, N = 512
- When WDTCR.CKS[3:0] = 0x7, N = 2048
- When WDTCR.CKS[3:0] = 0x8, N = 8192

### REFEF flag (Refresh Error Flag)

Read the REFEF flag to confirm whether a refresh error occurred, indicating that a refresh operation was performed during a prohibited period. A value of 1 indicates that a refresh error occurred. Write 0 to the flag to set the value to 0. Writing 1 has no effect.

Clearing of the REFEF flag takes (N+1) PCLKB cycles. In addition, clearing of the flag is ignored for (N+1) PCLKB cycles after a refresh error. N is specified in the WDTCR.CKS[3:0] bits as follows:

- When WDTCR.CKS[3:0] = 0x1, N = 4
- When WDTCR.CKS[3:0] = 0x4, N = 64
- When WDTCR.CKS[3:0] = 0xF, N = 128
- When WDTCR.CKS[3:0] = 0x6, N = 512
- When WDTCR.CKS[3:0] = 0x7, N = 2048
- When WDTCR.CKS[3:0] = 0x8, N = 8192

## 24.2.4 WDTRCR : WDT Reset Control Register

Base address: WDT = 0x4008_3400

Offset address: 0x06

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | RSTIRQS | — | — | — | — | — | — | — |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:0 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | RSTIRQS | WDT Behavior Selection<br>0: Interrupt<br>1: Reset | R/W |

The WDTRCR register controls reset output by a WDT down-counter underflow or interrupt request output.

Some constraints apply to writes to the WDTRCR register. For details, see section 24.3.2. Controlling Writes to the WDTCR, WDTRCR, and WDTCSTPR Registers.

In auto start mode, the WDTRCR register settings are disabled, and the settings in the Option Function Select register 0 (OFS0) are enabled. The settings for the WDTRCR register can also be made for the OFS0 register. For details, see section 24.3.8. Association between Option Function Select Register 0 (OFS0) and WDT Registers.

### 24.2.5 WDTCSTPR : WDT Count Stop Control Register

Base address: WDT = 0x4008_3400

Offset address: 0x08

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SLCSTP | — | — | — | — | — | — | — |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | SLCSTP | Sleep-Mode Count Stop Control Register<br>0: Disable count stop<br>1: Stop count on transition to Sleep mode | R/W |

The WDTCSTPR register controls whether to stop the WDT counter in Sleep mode. Some constraints apply to writes to the WDTCSTPR register. For details, see section 24.3.2. Controlling Writes to the WDTCR, WDTRCR, and WDTCSTPR Registers.

In auto start mode, the WDTCSTPR register settings are disabled, and the settings in the Option Function Select register 0 (OFS0) are enabled. The settings for the WDTCSTPR register can also be made for the OFS0 register. For details, see section 24.3.8. Association between Option Function Select Register 0 (OFS0) and WDT Registers.

#### SLCSTP bit (Sleep-Mode Count Stop Control Register)

The SLCSTP bit selects whether to stop counting on transition to Sleep mode.

### 24.2.6 Option Function Select Register 0 (OFS0)

For information on the OFS0 register, see section 24.3.8. Association between Option Function Select Register 0 (OFS0) and WDT Registers.

## 24.3 Operation

### 24.3.1 Count Operation in each Start Mode

The WDT has two start modes:

- Auto start mode, in which counting automatically starts after a release from the reset state
- Register start mode, in which counting starts with a refresh by writing to the register.

In auto start mode, counting automatically starts after a release from the reset state according to the settings in the Option Function Select register 0 (OFS0) in the flash.

In register start mode, counting starts with a refresh by writing to the WDTRR register after the respective registers are set after a release from the reset state.

Select auto start mode or register start mode by setting the WDT Start Mode Select bit (OFS0.WDTSTRT) in the OFS0 register.

When the auto start mode is selected, the settings in the WDT Control Register (WDTCR), WDT Reset Control Register (WDTRCR), and WDT Count Stop Control Register (WDTCSTPR) are disabled while the settings in the OFS0 register are enabled.

When the register start mode is selected, the setting for the OFS0 register is disabled while the settings for the WDT Control Register (WDTCR), WDT Reset Control Register (WDTRCR), and WDT Count Stop Control Register (WDTCSTPR) are enabled.

#### 24.3.1.1 Register start mode

When the WDT Start Mode Select bit (OFS0.WDTSTRT) is 1, register start mode is selected, the OFS0 register setting is invalid, and the WDT control register (WDTCR), WDT Reset Control Register (WDTRCR), and WDT Count Stop Control Register (WDTCSTPR) are enabled.

After the reset state is released, set the following:

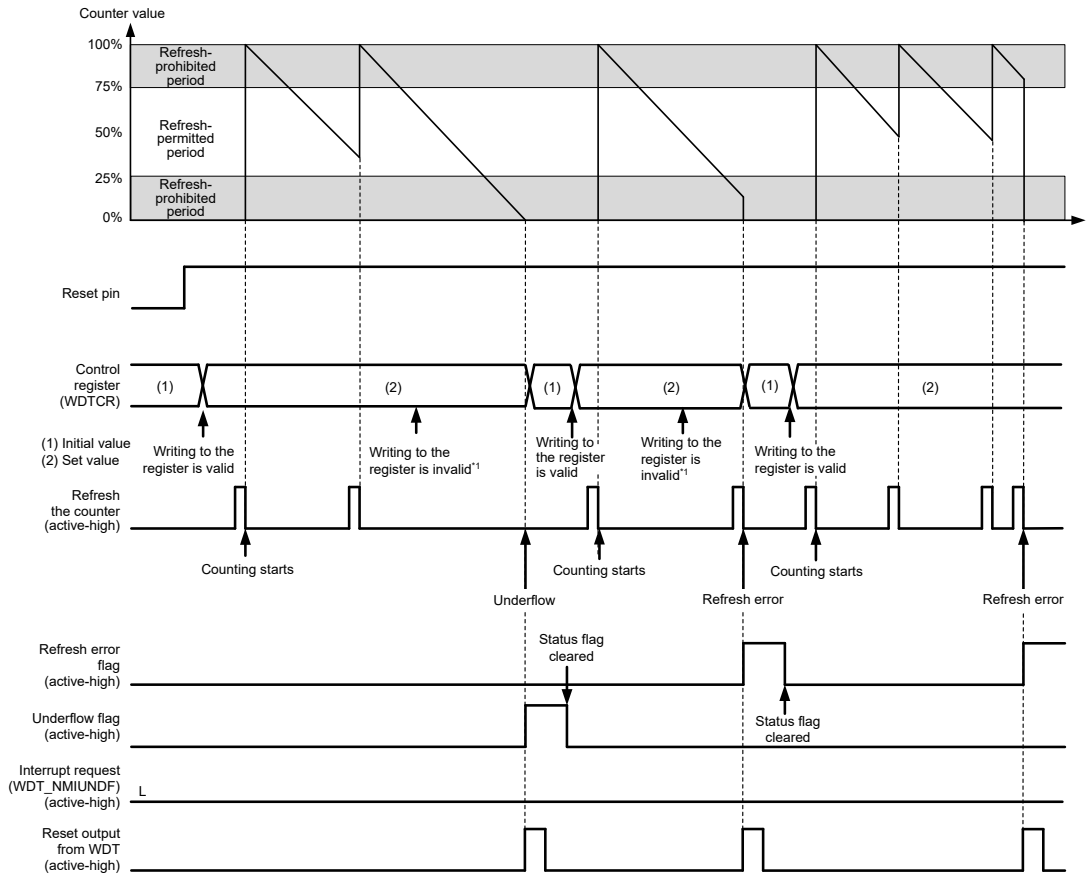
- Clock division ratio in the WDTCR register
- Window start and end positions in the WDTCR register
- Timeout period in the WDTCR register
- Reset output or interrupt request output in the WDTRCR register
- Counter stop control during transitions to Sleep mode in the WDTCSTPR register

The WDT refresh register (WDTRR) refreshes the down counter. As a result, the downcount starts at the value set by the timeout period selection bit (WDTCR.TOPS[1:0]).

Thereafter, as long as the counter is refreshed in the refresh-permitted period, the value in the counter is reset each time the counter is refreshed and counting down continues. The WDT does not output the reset signal or non-maskable interrupt request/interrupt request as long as counting continues. However, if the down-counter underflows because the down-counter cannot be refreshed due to a program runaway, or if a refresh error occurs because the counter was refreshed outside the refresh-permitted period, the WDT outputs the reset signal or a non-maskable interrupt request/interrupt request (WDT_NMIUNDF). Reset output or interrupt request output can be selected in the WDT Reset Interrupt Request Select bit (WDTRCR.RSTIRQS). The interrupt enabled for operating the NMI can be selected in the WDT Underflow/Refresh Error Interrupt Enable bit (NMIER.WDTEN).

Figure 24.3 shows an example of operation under the following conditions:

- Register start mode (OFS0.WDTSTRT = 1)
- WDT reset interrupt request selection (WDTRCR.RSTIRQS = 1)
- The window start position is 75% (WDTCR.RPSS[1:0] = 10b)
- The window end position is 25% (WDTCR.RPES[1:0] = 10b)




Note: Refreshing the counter requires up to 4 count signal cycles from refresh request to execution.
Note 1. See section 24.3.2. Controlling Writes to the WDTCR, WDTRCR, and WDTCSTPR Registers.
Reset pin = RES

**Figure 24.3 Operation example in register start mode**

### 24.3.1.2 Auto start mode

When the WDT Start Mode Select bit (OFS0.WDTSTRT) in the Option Function Select Register 0 (OFS0) is 0, auto start mode is selected, the WDT Control Register (WDTCR), WDT Reset Control Register (WDTRCR), and WDT Count Stop Control Register (WDTCSTPR) are disabled, and the settings in the OFS0 register are enabled.

Within the reset state, the setting values for the following in the Option Function Select Register 0 (OFS0) are set in the WDT registers:

- Clock division ratio
- Window start and end positions
- Timeout period
- Reset output or interrupt request
- Counter stop control during transition to Sleep mode

When the reset state is released, the down-counter automatically starts counting down from the value set in the WDT Timeout Period Select bits (OFS0.WDTTOPS[1:0]).

Thereafter, as long as the counter is refreshed in the refresh-permitted period, the value in the counter is reset each time the counter is refreshed and counting down continues. The WDT does not output the reset signal or non-maskable interrupt request/interrupt request (WDT_NMIUNDF) as long as the counting continues. However, if the down-counter underflows

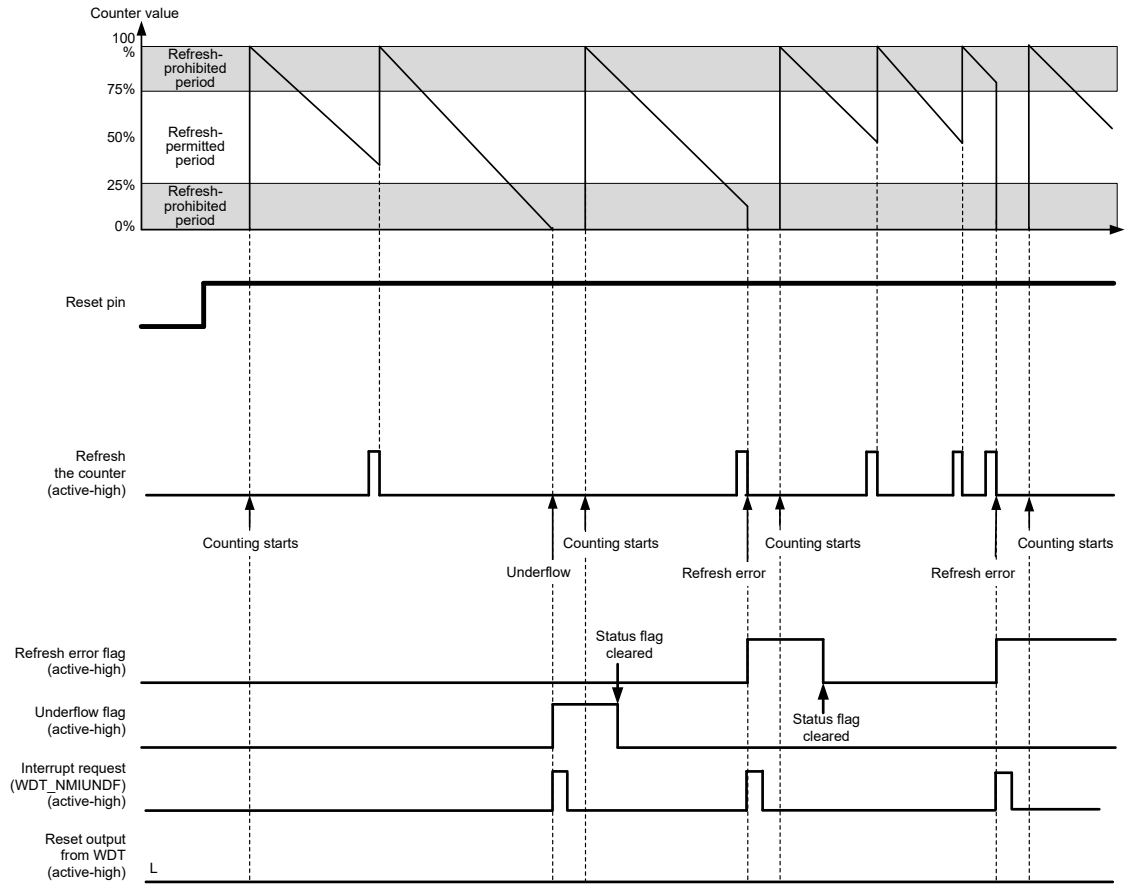
because refreshing of the down-counter is not possible due to a runaway program or if a refresh error occurs due to refreshing outside the refresh-permitted period, the WDT outputs the reset signal or non-maskable interrupt request/interrupt request (WDT_NMIUNDF).

After the reset signal or non-maskable interrupt request/interrupt request is generated, the counter reloads the timeout period after counting for 1 cycle. The value of the timeout period is set in the down-counter and counting restarts.

Reset output or interrupt request output can be selected by setting the WDT Reset Interrupt Request Select bit (OFS0.WDTRSTIRQS). Non-maskable interrupt request or interrupt request can be selected in the WDT Underflow/ Refresh Error Interrupt Enable bit (NMIER.WDTEN).

Figure 24.4 shows an example of operation (non-maskable interrupt) under the following conditions:

- Auto start mode (OFS0.WDTSTRT = 0)
- WDT behavior selection: interrupt (OFS0.WDTRSTIRQS = 0)
- Non-maskable Interrupt: IWDT Underflow/Refresh Error Interrupt Enabled (NMIER.WDTEN = 1)
- The window start position is 75% (OFS0.WDTRPSS[1:0] = 10b)
- The window end position is 25% (OFS0.WDTRPES[1:0] = 10b)




Note: Refreshing the counter requires up to 4 count signal cycles from refresh request to execution.
Note: Reset pin = RES

**Figure 24.4 Operation example in auto start mode**

### 24.3.2 Controlling Writes to the WDTCR, WDTRCR, and WDTCSTPR Registers

Writing to the WDT Control Register (WDTCR), WDT Reset Control Register (WDTRCR), or WDT Count Stop Control Register (WDTCSTPR) is possible once each between the release from the reset state and the first refresh operation.

After a refresh (counting starts) or a write to WDTCR, WDTRCR or WDTCSTPR register, the protection signal in the WDT becomes 1 to protect WDTCR, WDTRCR and WDTCSTPR register against subsequent write attempts. This protection is released by the reset source of the WDT. With other reset sources, the protection is not released.

Figure 24.5 shows control waveforms produced in response to writing to the WDTCR.

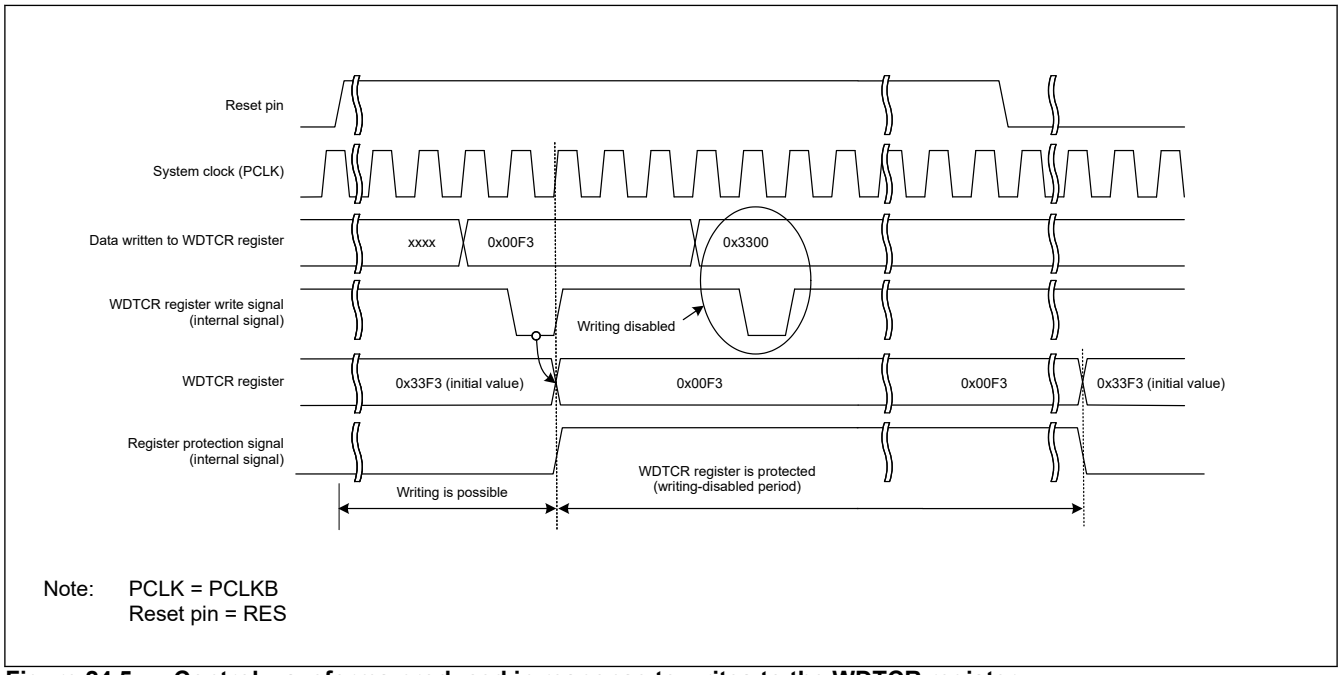



**Figure 24.5 Control waveforms produced in response to writes to the WDTCR register**

### 24.3.3 Refresh Operation

To refresh the down counter and start the counting operation, write to the WDT Refresh Register (WDTRR) in the order of values from 0x00 to 0xFF. If a value other than 0xFF is written after 0x00, the down-counter is not refreshed. If an invalid value is written, refreshing is performed normally by writing to the WDTRR register in the order of values from 0x00 to 0xFF.

Correct refreshing is also performed when a register other than WDTRR is accessed or WDTRR is read between writing 0x00 and writing 0xFF to WDTRR. Writes to refresh the counter must be made within the refresh-permitted period, and this is determined by the 0xFF write. For this reason, correct refreshing is performed even when 0x00 is written outside the refresh-permitted period.

[Example write sequences that are valid for refreshing the counter]

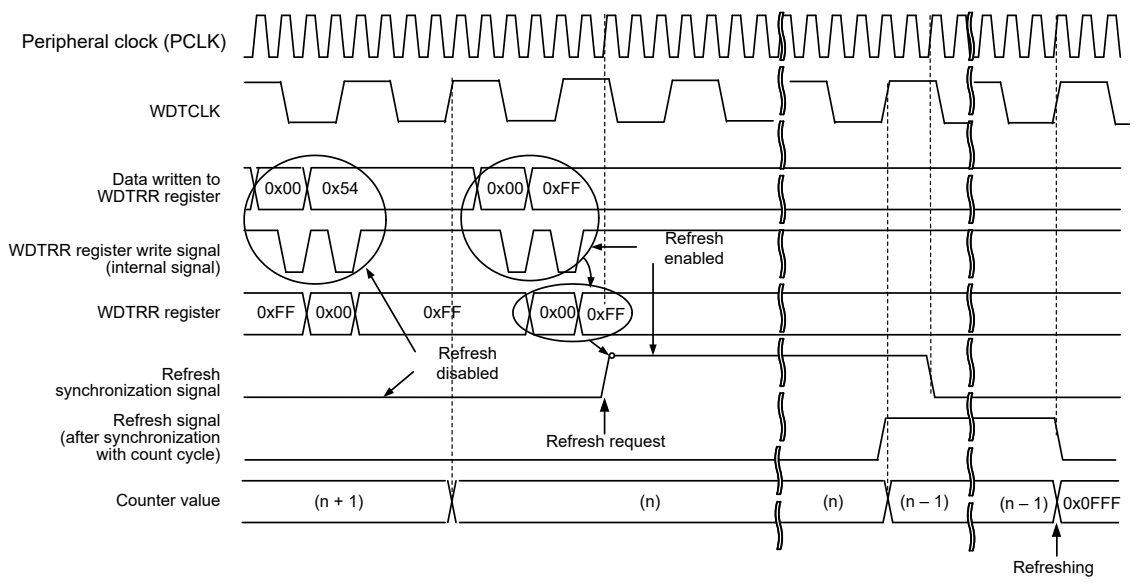
- 0x00 → 0xFF
- 0x00 ((n–1)th time) → 0x00 (nth time) → 0xFF
- 0x00 → access to another register or read from WDTRR → 0xFF

[Example write sequences that are invalid for refreshing the counter]

- 0x23 (a value other than 0x00) → 0xFF
- 0x00 → 0x54 (a value other than 0xFF)
- 0x00 → 0xAA (0x00 and a value other than 0xFF) → 0xFF

After 0xFF is written to the WDT Refresh Register (WDTRR), refreshing the down-counter requires up to 4 cycles of the signal for counting. To meet this requirement, complete writing 0xFF to WDTRR 4 count cycles before the down-counter underflows.

Figure 24.6 shows the WDT refresh-operation waveforms when the clock division ratio is PCLKB/64.




Note: PCLK = PCLKB

**Figure 24.6 WDT refresh operation waveforms when WDTCR.CKS[3:0] = 0x4 and WDTCR.TOPS[1:0] = 01b**

Note: When setting the refresh time, consider the oscillation accuracy of the clock sources of the PCLKB and WDTCLK. Set values which ensure that refreshing is possible even when the frequency varies in the range of error of the oscillation accuracy.

### 24.3.4 Status Flags

The refresh error (WDTSR.REFEF) and underflow (WDTSR.UNDFF) flags retain the source of the interrupt request from the WDT. After a release from the interrupt request generation, read the WDTSR.REFEF and WDTSR.UNDFF flags to check for the interrupt source. For each flag, writing 0 clears the bit. Writing 1 has no effect. Leaving the status flags unchanged does not affect operation. If the flags are not cleared at the next interrupt request from the WDT, the earlier interrupt source is cleared and the new interrupt source is written. For the time period between when 0 is written in each flag and when its value is reflected, see section 24.2.3. WDTSR : WDT Status Register.

### 24.3.5 Reset Output

When the Reset Interrupt Select bit (WDTRCR.RSTIRQS) is set to 1 in register start mode, or when the WDT Reset Interrupt Request Select bit (OFS0.WDTRSTIRQS) in the Option Function Select Register 0 (OFS0) is set to 1 in auto start mode, a reset signal is output for 1 cycle count when an underflow in the down-counter or a refresh error occurs.

In register start mode, the down-counter is initialized (all bits set to 0) and stopped in that state after output of a reset signal. After the reset state is released and the program is restarted, the counter is set up again and counting down starts again with a refresh. In auto start mode, counting down starts automatically after the reset state is released.

### 24.3.6 Interrupt Sources

When the Reset Interrupt Select bit (WDTRCR.RSTIRQS) is set to 0 in register start mode or when the WDT Reset Interrupt Request Select bit (OFS0.WDTRSTIRQS) in the Option Function Select Register 0 (OFS0) is set to 0 in auto start mode, an interrupt (WDT_NMIUNDF) signal is generated when an underflow in the counter or a refresh error occurs. This interrupt can be used as a non-maskable interrupt or an interrupt. For details, see section 13, Interrupt Controller Unit (ICU).

**Table 24.4 WDT interrupt source**

| Name | Interrupt source | Interrupt to CPU | Start DMAC or DTC |
|---|---|---|---|
| WDT_NMIUNDF | • Down-counter underflow<br>• Refresh error | Possible | Not possible |

## 24.3.7 Reading the Down-Counter Value

The WDT stores the counter value in the down-counter value bits (WDTSR.CNTVAL[13:0]) of the WDT Status Register. Check these bits to obtain the counter value. The read value of the down-counter might differ from the actual count by one.

Figure 24.7 shows the processing for reading the WDT down-counter value when the clock division ratio is PCLKB/64.

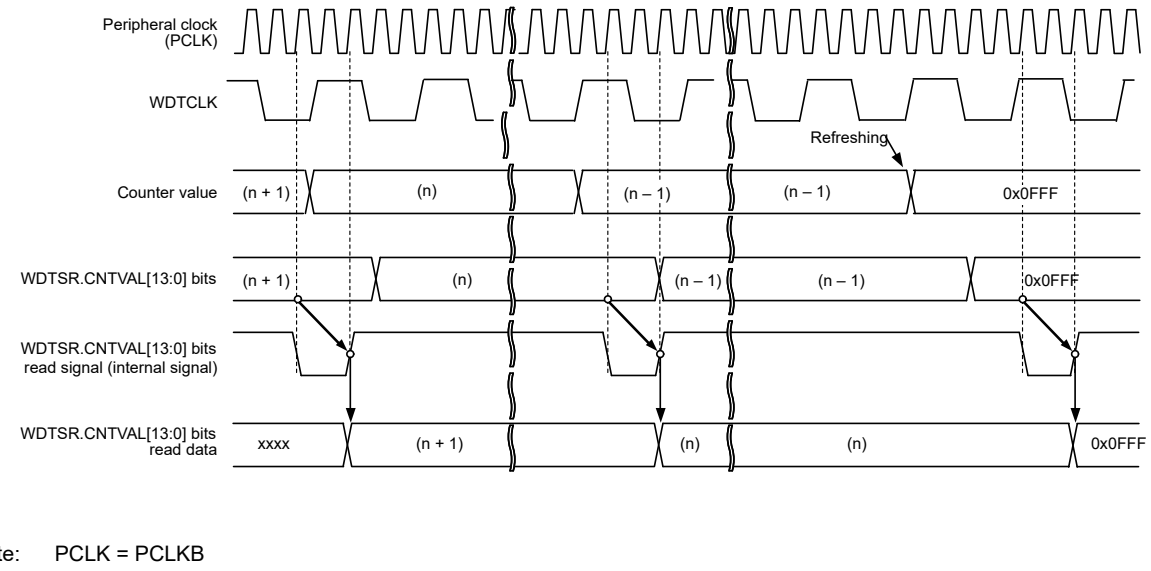



**Figure 24.7 Processing for reading WDT down-counter value when WDTCR.CKS[3:0] = 0x4 and WDTCR.TOPS[1:0] = 01b**

## 24.3.8 Association between Option Function Select Register 0 (OFS0) and WDT Registers

Table 24.5 lists the association between the Option Function Select Register 0 (OFS0) used in auto start mode, and the registers used in register start mode. For details on the Option Function Select Register 0 (OFS0), see section 6.2.1. OFS0 : Option Function Select Register 0.

**Table 24.5 Association between Option Function Select Register 0 (OFS0) and the WDT registers**

| Control target | Function | OFS0 register (enabled in auto start mode) OFS0.WDTSTRT = 0 | WDT registers (enabled in register start mode) OFS0.WDTSTRT = 1 |
|---|---|---|---|
| Down-counter | Timeout period selection | OFS0.WDTTOPS[1:0] | WDTCR.TOPS[1:0] |
| | Clock division ratio selection | OFS0.WDTCKS[3:0] | WDTCR.CKS[3:0] |
| | Window start position selection | OFS0.WDTRPSS[1:0] | WDTCR.RPSS[1:0] |
| | Window end position selection | OFS0.WDTRPES[1:0] | WDTCR.RPES[1:0] |
| Reset output or interrupt request output | Select a reset interrupt request | OFS0.WDTRSTIRQS | WDTRCR.RSTIRQS |
| Count stop | Sleep mode count stop control | OFS0.WDTSTPCTL | WDTCSTPR.SLCSTP |

## 24.4 Output to the Event Link Controller (ELC)

The WDT is capable of a link operation for the previously specified module when interrupt request signal is used as an event signal by the ELC. The event signal is output by the counter underflow and refresh error. An event signal is output regardless of the setting of the Reset Interrupt Request Select bit (WDTRCR.RSTIRQS) in register start mode or auto start mode. An event signal can also be output when the next interrupt source is generated while the Refresh Error flag (WDTSR.REFEF) or Underflow flag (WDTSR.UNDFF) is 1. For details, see section 18, Event Link Controller (ELC).

## 24.5 Usage Notes

### 24.5.1 ICU Event Link Setting Register n (IELSRn) Setting

Setting 0x53 to ICU Event Link Setting Register n (ICU.IELSRn) is prohibited when WDT reset interrupt request selection resets (OFS0.WDTRSTIRQS = 0 or WDTRCR.RSTIRQS = 0) or when enabling event link operation (ELSRn.ELS[8:0] = 0x53).

# 25. Independent Watchdog Timer (IWDT)

## 25.1 Overview

The Independent Watchdog Timer (IWDT) consists of a 14-bit down counter that must be serviced periodically to prevent counter underflow. The IWDT provides functionality to reset the MCU or to generate a non-maskable interrupt or an underflow interrupt. Because the timer operates with an independent, dedicated clock source, it is particularly useful in returning the MCU to a known state as a fail-safe mechanism when the system runs out of control. The IWDT can be triggered automatically by a reset, underflow, refresh error, or a refresh of the count value in the registers.

The IWDT functions differ from those of the WDT in the following respects:

- The divided IWDT-dedicated clock (IWDTCLK) is used as the count source (not affected by PCLKB)
- IWDT does not support register start mode

Table 25.1 lists the IWDT specifications and Figure 25.1 shows a block diagram.

**Table 25.1 IWDT specifications**

| Parameter | Description |
|---|---|
| Count source*1 | IWDT-dedicated clock (IWDTCLK) |
| Clock division ratio | Division by 1, 16, 32, 64, 128, or 256 |
| Counter operation | Counting down using a 14-bit down-counter |
| Condition for starting the counter | • Counting automatically starts after a reset<br>• Only secure developer can start the IWDT |
| Conditions for stopping the counter | • Reset (the down-counter and other registers return to their initial values)<br>• A counter underflows or a refresh error is generated (counting restarts automatically). |
| Window function | Window start and end positions can be specified (refresh-permitted and refresh-prohibited periods) |
| Reset output sources | • Down-counter underflows<br>• Refreshing outside the refresh-permitted period (refresh error). |
| Non-maskable interrupt/interrupt sources | • Down-counter underflows<br>• Refreshing outside the refresh-permitted period (refresh error). |
| Reading the counter value | The down-counter value can be read by the IWDTSR register |
| Event link function | • Down-counter underflow event output<br>• Refresh error event output. |
| Output signal (internal signal) | • Reset output<br>• Interrupt request output<br>• Sleep-mode count stop control output. |
| Auto start mode | Configurable to the following triggers:<br>• Clock frequency division ratio after a reset (OFS0.IWDTCKS[3:0] bits)<br>• Timeout period of the Independent Watchdog Timer (OFS0.IWDTTOPS[1:0] bits)<br>• Window start position in the Independent Watchdog Timer (OFS0.IWDTRPSS[1:0] bits)<br>• Window end position in the Independent Watchdog Timer (OFS0.IWDTRPES[1:0] bits)<br>• Reset output or interrupt request output (OFS0.IWDTRSTIRQS bit)<br>• Down-count stop function at transition to Sleep, Snooze, or Software Standby mode (OFS0.IWDTSTPCTL bit). |
| TrustZone Filter | Security attribution can be set |

Note 1. Satisfy the frequency of the peripheral module clock (PCLKB) ≥ 4 × (the frequency of the count clock source after division).

The bus interface and registers operate with PCLKB, and the 14-bit counter and control circuits operate with IWDTCLK.

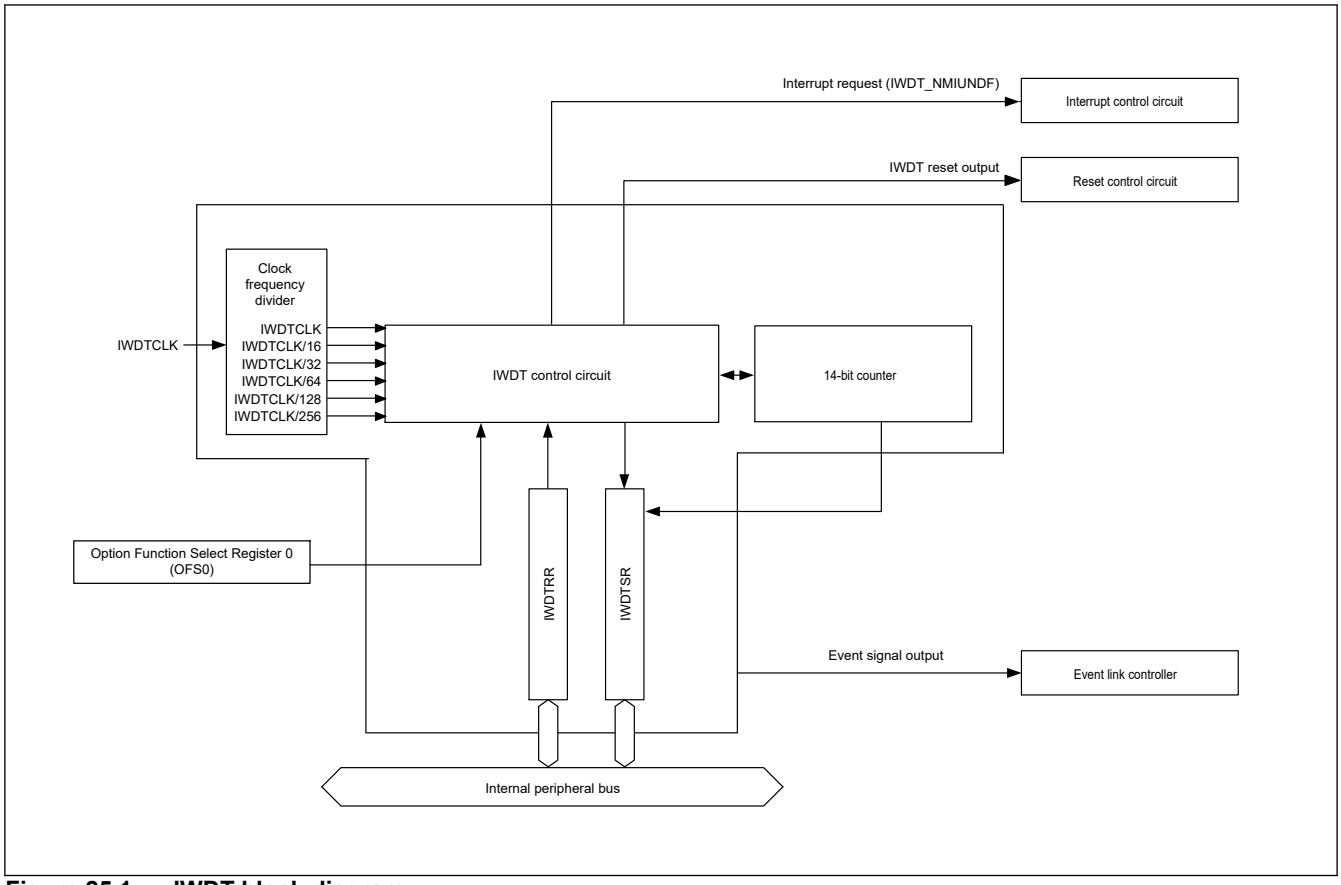



**Figure 25.1 IWDT block diagram**

## 25.2 Register Descriptions

### 25.2.1 IWDTRR : IWDT Refresh Register

Base address: IWDT = 0x4008_3200

Offset address: 0x00

| Bit position: | 7 | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | The down-counter is refreshed by writing 0x00 and then writing 0xFF to this register | R/W |

The IWDTRR register refreshes the down-counter of the IWDT. The down-counter of the IWDT is refreshed by writing 0x00 and then writing 0xFF to IWDTRR (refresh operation) within the refresh-permitted period. After the down-counter is refreshed, it starts counting down from the value selected in the IWDT Timeout Period Select bits (OFS0.IWDTTOPS[1:0]) in the Option Function Select Register 0 (OFS0).

When 0x00 is written, the read value is 0x00. When a value other than 0x00 is written, the read value is 0xFF. For details of the refresh operation, see section 25.3.2. Refresh Operation.

## 25.2.2 IWDTSR : IWDT Status Register

Base address: IWDT = 0x4008_3200

Offset address: 0x04

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | REFEF | UNDFF | CNTVAL[13:0] | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 13:0 | CNTVAL[13:0] | Down-counter Value<br>Value counted by the down-counter | R |
| 14 | UNDFF | Underflow Flag<br>0: No underflow occurred<br>1: Underflow occurred | R/W*1 |
| 15 | REFEF | Refresh Error Flag<br>0: No refresh error occurred<br>1: Refresh error occurred | R/W*1 |

Note 1. Only 0 can be written to clear the flag.

The IWDTSR register indicates the counter value of the down-counter and whether an underflow or refresh error occurred in the down-counter.

### CNTVAL[13:0] bits (Down-counter Value)

Read the CNTVAL[13:0] bits to confirm the value of the down-counter. The read value might differ from the actual count by 1.

### UNDFF flag (Underflow Flag)

Read the UNDFF flag to confirm whether an underflow occurred in the down-counter. The value 1 indicates that the down-counter underflowed. Write 0 to the UNDFF flag to set the value to 0. Writing 1 has no effect.

Clearing of the UNDFF flag takes (N + 2) IWDTCLK cycles and 2 PCLKB cycles. In addition, clearing of this flag is ignored for (N + 2) IWDTCLK cycles after an underflow. N is specified in the IWDTCKS[3:0] bits as follows:

- When OFS0.IWDTCKS[3:0] = 0x0, N = 1
- When OFS0.IWDTCKS[3:0] = 0x2, N = 16
- When OFS0.IWDTCKS[3:0] = 0x3, N = 32
- When OFS0.IWDTCKS[3:0] = 0x4, N = 64
- When OFS0.IWDTCKS[3:0] = 0xF, N = 128
- When OFS0.IWDTCKS[3:0] = 0x5, N = 256.

### REFEF flag (Refresh Error Flag)

Read the REFEF flag to confirm whether a refresh error occurred. This indicates that a refresh operation was performed during a prohibited period. The value 1 indicates that a refresh error occurred. Write 0 to the REFEF flag to set the value to 0. Writing 1 has no effect.

Clearing of the REFEF flag takes (N + 2) IWDTCLK cycles and 2 PCLKB cycles. In addition, clearing of this flag is ignored for (N + 2) IWDTCLK cycles following a refresh error. N is specified in the IWDTCKS[3:0] bits as follows:

- When OFS0.IWDTCKS[3:0] = 0x0, N = 1
- When OFS0.IWDTCKS[3:0] = 0x2, N = 16
- When OFS0.IWDTCKS[3:0] = 0x3, N = 32
- When OFS0.IWDTCKS[3:0] = 0x4, N = 64
- When OFS0.IWDTCKS[3:0] = 0xF, N = 128
- When OFS0.IWDTCKS[3:0] = 0x5, N = 256.

### 25.2.3 OFS0 : Option Function Select Register 0

For information on the Option Function Select Register 0 (OFS0), see section 6.2.1. OFS0 : Option Function Select Register 0.

#### IWDTTOPS[1:0] bits (IWDT Timeout Period Select)

The IWDTTOPS[1:0] bits select the timeout period, that is, the period until the down-counter underflows, from 128, 512, 1024, or 2048 cycles, taking the divided clock specified in the IWDTCKS[3:0] bits as 1 cycle.

After the down-counter is refreshed, the combination of the IWDTCKS[3:0] and IWDTTOPS[1:0] bits determines the number of IWDTCLK cycles until the counter underflows.

Table 25.2 lists the relationship between the IWDTCKS[3:0] and IWDTTOPS[1:0] bit settings, the timeout period, and the number of IWDTCLK cycles.

**Table 25.2 Timeout period settings**

| IWDTCKS[3:0] bits | | | | IWDTTOPS[1:0] bits | | Clock division ratio | Timeout period (number of cycles) | IWDTCLK cycles |
|---|---|---|---|---|---|---|---|---|
| b7 | b6 | b5 | b4 | b1 | b0 | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | IWDTCLK | 128 | 128 |
| | | | | 0 | 1 | | 512 | 512 |
| | | | | 1 | 0 | | 1024 | 1024 |
| | | | | 1 | 1 | | 2048 | 2048 |
| 0 | 0 | 1 | 0 | 0 | 0 | IWDTCLK/16 | 128 | 2048 |
| | | | | 0 | 1 | | 512 | 8192 |
| | | | | 1 | 0 | | 1024 | 16384 |
| | | | | 1 | 1 | | 2048 | 32768 |
| 0 | 0 | 1 | 1 | 0 | 0 | IWDTCLK/32 | 128 | 4096 |
| | | | | 0 | 1 | | 512 | 16384 |
| | | | | 1 | 0 | | 1024 | 32768 |
| | | | | 1 | 1 | | 2048 | 65536 |
| 0 | 1 | 0 | 0 | 0 | 0 | IWDTCLK/64 | 128 | 8192 |
| | | | | 0 | 1 | | 512 | 32768 |
| | | | | 1 | 0 | | 1024 | 65536 |
| | | | | 1 | 1 | | 2048 | 131072 |
| 1 | 1 | 1 | 1 | 0 | 0 | IWDTCLK/128 | 128 | 16384 |
| | | | | 0 | 1 | | 512 | 65536 |
| | | | | 1 | 0 | | 1024 | 131072 |
| | | | | 1 | 1 | | 2048 | 262144 |
| 0 | 1 | 0 | 1 | 0 | 0 | IWDTCLK/256 | 128 | 32768 |
| | | | | 0 | 1 | | 512 | 131072 |
| | | | | 1 | 0 | | 1024 | 262144 |
| | | | | 1 | 1 | | 2048 | 524288 |

#### IWDTCKS[3:0] bits (IWDT-Dedicated Clock Frequency Division Ratio Select)

The IWDTCKS[3:0] bits specify the division ratio of the clock used for the down-counter. The division ratio can be selected from the IWDT-dedicated clock (IWDTCLK) divided by 1, 16, 32, 64, 128, and 256. Combined with the IWDTTOPS[1:0] bit setting, the IWDT can be configured to a count period between 128 and 524,288 IWDTCLK cycles.

### IWDTRPES[1:0] bits (IWDT Window End Position Select)

The IWDTRPES[1:0] bits specify the window end position that indicates the refresh-permitted period. 75%, 50%, 25%, or 0% of the timeout period can be selected for the window end position. Set the window end position to a value less than the window start position (window start position > window end position). If the window start position is set to a value less than or equal to the window end position, the window start position setting is enabled and the window end position is set to 0%.

### IWDTRPSS[1:0] bits (IWDT Window Start Position Select)

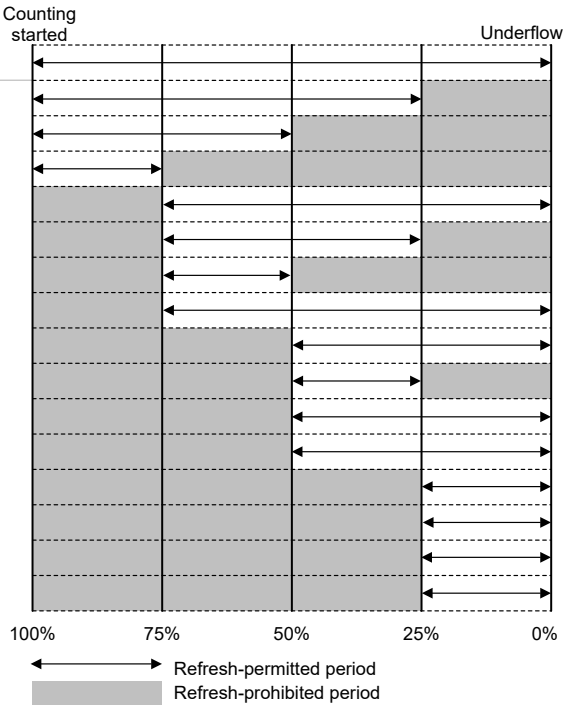
The IWDTRPSS[1:0] bits specify the window start position that indicates the refresh-permitted period. 100%, 75%, 50%, or 25% of the timeout period can be selected for the window start position. Set the window start position to a value greater than the window end position. If the window start position is set to a value less than or equal to the window end position, the window start position setting is enabled and the window end position is set to 0%.

Table 25.3 lists the counter values for the window start and end positions, and Figure 25.2 shows the refresh-permitted period set in the IWDTRPSS[1:0], IWDTRPES[1:0], and IWDTTOPS[1:0] bits.

**Table 25.3 Relationship between the timeout period and window start and end counter values**

| IWDTTOPS[1:0] bits | | Timeout period | | Window start and end counter value | | | |
|---|---|---|---|---|---|---|---|
| b1 | b0 | Cycles | Counter value | 100% | 75% | 50% | 25% |
| 0 | 0 | 128 | 0x007F | 0x007F | 0x005F | 0x003F | 0x001F |
| 0 | 1 | 512 | 0x01FF | 0x01FF | 0x017F | 0x00FF | 0x007F |
| 1 | 0 | 1024 | 0x03FF | 0x03FF | 0x02FF | 0x01FF | 0x00FF |
| 1 | 1 | 2048 | 0x07FF | 0x07FF | 0x05FF | 0x03FF | 0x01FF |

| IWDTRPSS[1:0] bits | | IWDTRPES[1:0] bits | | Window | |
|---|---|---|---|---|---|
| b13 | b12 | b9 | b8 | Start (%) | End (%) |
| 1 | 1 | 1 | 1 | 100 | 0 |
| | | 1 | 0 | | 25 |
| | | 0 | 1 | | 50 |
| | | 0 | 0 | | 75 |
| 1 | 0 | 1 | 1 | 75 | 0 |
| | | 1 | 0 | | 25 |
| | | 0 | 1 | | 50 |
| | | 0 | 0 | | 75 |
| 0 | 1 | 1 | 1 | 50 | 0 |
| | | 1 | 0 | | 25 |
| | | 0 | 1 | | 50 |
| | | 0 | 0 | | 75 |
| 0 | 0 | 1 | 1 | 25 | 0 |
| | | 1 | 0 | | 25 |
| | | 0 | 1 | | 50 |
| | | 0 | 0 | | 75 |




Note: If window end position setting ≥ window start position setting, the window end position setting is set to 0%.

**Figure 25.2 IWDTRPSS[1:0] and IWDTRPES[1:0] bit settings and refresh-permitted period**

### IWDTRSTIRQS bit (IWDT Reset Interrupt Request Select)

The IWDTRSTIRQS bit specifies the behavior when an underflow or a refresh error occurs. Setting 1 selects reset output. Setting 0 selects interrupt.

**IWDTSTPCTL bit (IWDT Stop Control)**

The IWDTSTPCTL bit selects whether to stop counting on transition to Sleep, Snooze, or Software Standby mode.

## 25.3 Operation

### 25.3.1 Auto Start Mode

When the IWDT Start Mode Select bit (OFS0.IWDTSTRT) in the Option Function Select Register 0 is 0, auto start mode is selected, otherwise the IWDT is disabled.

Within the reset state, the setting values for the following in the Option Function Select Register 0 (OFS0) are set in the IWDT registers:

- Clock division ratio (OFS0.IWDTCKS[3:0])
- Window start and end positions (OFS0.IWDTRPSS[1:0], OFS0.IWDTRPES[1:0])
- Timeout period (OFS0.IWDTTOPS[1:0])
- Reset output or interrupt request (OFS0.IWDTRSTIRQS)

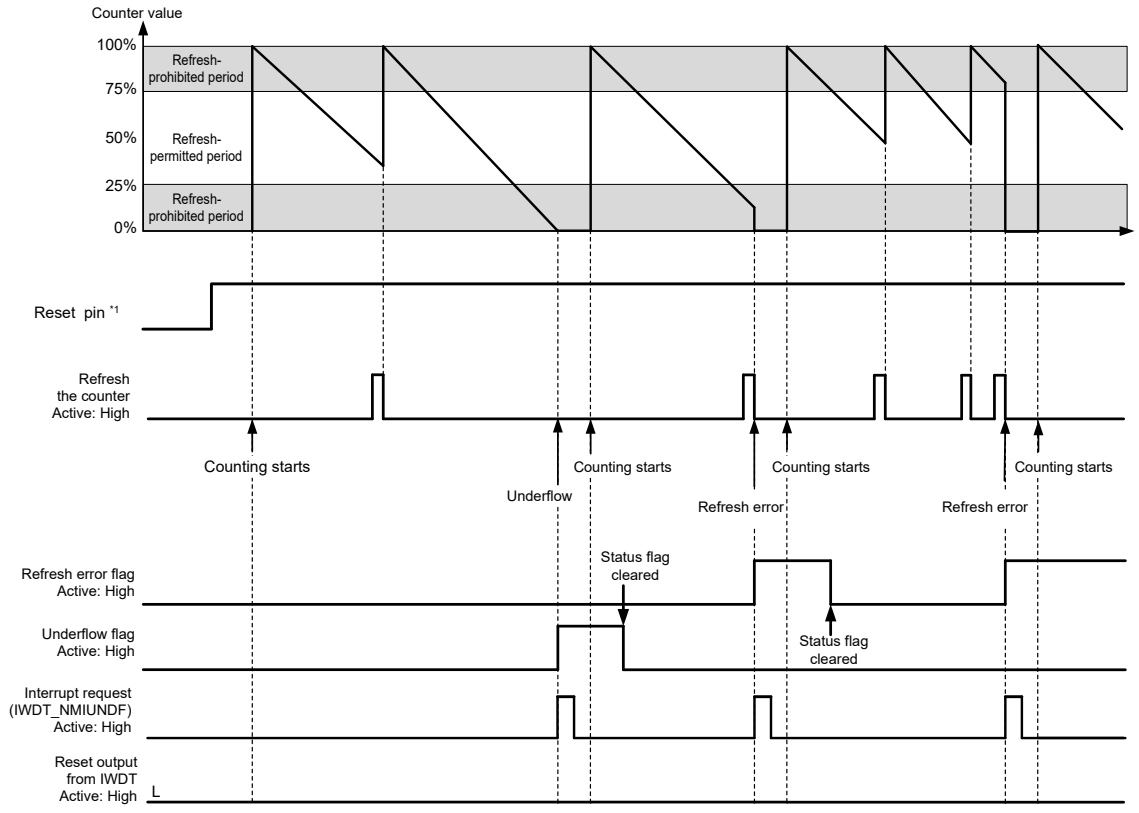
When the reset state is released, the counter automatically starts counting down from the value selected in the IWDT Timeout Period Select bits (OFS0.IWDTTOPS[1:0]).

After that, as long as the program continues normal operation and the counter is refreshed within the refresh-permitted period, the value in the counter is reset each time the counter is refreshed and down-counting continues. The IWDT does not output the reset signal as long as this procedure continues. However, if the counter underflows because the program crashed or because a refresh error occurred when an attempt is made to refresh outside the refresh-permitted period, the IWDT asserts the reset signal or non-maskable interrupt request/interrupt request (IWDT_NMIUNDF).

After the reset signal or non-maskable interrupt request/interrupt request is generated, the counter reloads the timeout period after counting for 1 cycle, the value of the timeout period is set in the down-counter and counting starts. The reset output or interrupt request output can be selected with the IWDT Reset Interrupt Request Select bit (OFS0.IWDTRSTIRQS). The interrupt enabled for operating the NMI can be selected with the IWDT Underflow/Refresh Error Interrupt Enable bit (NMIER.IWDTEN).

Figure 25.3 shows an example of operation under the following conditions:

- Auto start mode (OFS0.IWDTSTRT = 0)
- IWDT behavior selection: interrupt (OFS0.IWDTRSTIRQS = 0)
- Non-maskable Interrupt: IWDT Underflow/Refresh Error Interrupt Enabled (NMIER.IWDTEN = 1)
- The window start position is 75% (OFS0.IWDTRPSS[1:0] = 10b)
- The window end position is 25% (OFS0.IWDTRPES[1:0] = 10b).




Note 1. Reset pin = RES

**Figure 25.3 Operation example in auto start mode**

## 25.3.2 Refresh Operation

To refresh the down counter and start the counting operation, write to the IWDT Refresh Register (IWDTRR) in the order of values from 0x00 to 0xFF. If a value other than 0xFF is written after 0x00, the down-counter is not refreshed. If an invalid value is written, refreshing is performed normally by writing to the IWDTRR register in the order of values from 0x00 to 0xFF.

When writes are made in the order of 0x00 (first time) → 0x00 (second time), and if 0xFF is written after that, the writing order 0x00 → 0xFF is satisfied. Writing 0x00 ((n - 1)th time) →0x00 (nth time) → 0xFF is valid, and the refresh is performed correctly. Even when the first value written before 0x00 is not 0x00, correct refreshing is performed as long as the operation contains the write sequence of 0x00 → 0xFF.

Correct refreshing is also performed regardless of whether a register other than IWDTRR is accessed or IWDTRR is read between writing 0x00 and writing 0xFF to IWDTRR. Writes to refresh the counter must be made within the refresh-permitted period. Whether writing is done within the refresh-permitted period is determined when 0xFF is written. For this reason, correct refreshing is performed even when 0x00 is written outside the refresh-permitted period.

[Example write sequences that are valid to refresh the counter]

- 0x00 → 0xFF
- 0x00 ((n - 1)th time) → 0x00 (nth time) → 0xFF
- 0x00 → access to another register or read from IWDTRR → 0xFF.

[Example write sequences that are not valid to refresh the counter]

- 0x23 (a value other than 0x00) → 0xFF

- 0x00 → 0x54 (a value other than 0xFF)
- 0x00 → 0xAA (0x00 and a value other than 0xFF) → 0xFF.

After 0xFF is written to the IWDTRR register, refreshing the counter requires up to 4 cycles of the signal for counting (the IWDT-Dedicated Clock Frequency Division Ratio Select bits (OFS0.IWDTCKS[3:0]) to determine how many cycles of the IWDT-dedicated clock (IWDTCLK) make up 1 cycle for counting. To meet this requirement, writing 0xFF to the IWDTRR must be completed 4 count cycles before the end of the refresh-permitted period or a down-counter underflow. The value of the counter can be checked with the counter bits (IWDTSR.CNTVAL[13:0]).

[Example refreshing timings]

- When the window start position is set to 0x1FFF, even if 0x00 is written to IWDTRR before 0x1FFF is reached at (0x2002, for example), refreshing occurs if 0xFF is written to IWDTRR after the value of the IWDTSR.CNTVAL[13:0] bits reaches 0x1FFF
- When the window end position is set to 0x1FFF, refreshing occurs if 0x2003 (4 count cycles before 0x1FFF) or a greater value is read from the IWDTSR.CNTVAL[13:0] bits immediately after writing 0x00 → 0xFF to IWDTRR
- When the refresh-permitted period continues until count 0x0000, refreshing can be performed immediately before an underflow. In this case, if 0x0003 (4 count cycles before an underflow) or a greater value is read from the IWDTSR.CNTVAL[13:0] bits immediately after writing 0x00 → 0xFF to IWDTRR, no underflow occurs and refreshing is performed.

Figure 25.4 shows the IWDT refresh-operation waveforms when PCLKB > IWDTCLK and the clock division ratio is IWDTCLK.

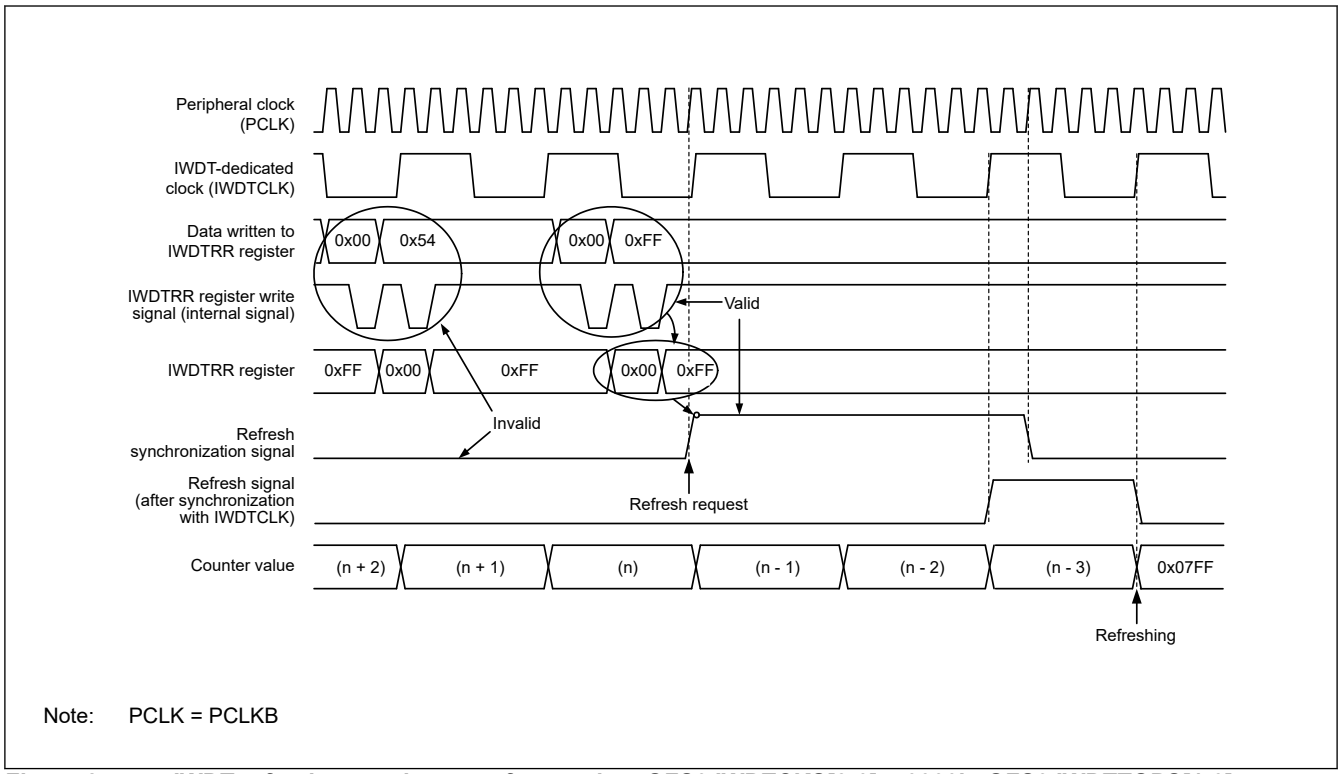



**Figure 25.4 IWDT refresh operation waveforms when OFS0.IWDTCKS[3:0] = 0000b, OFS0.IWDTTOPS[1:0] = 11b**

## 25.3.3 Status Flags

The refresh error (IWDTSR.REFEF) and underflow (IWDTSR.UNDFF) flags retain the source of the interrupt request from the IWDT. Therefore, after a release from the interrupt request generation, read the IWDTSR.REFEF and UNDFF flags to check for the interrupt source. For each flag, writing 0 clears the bit and writing 1 has no effect.

Leaving the status flags unchanged does not affect operation. If the flags are not cleared at the time of the next interrupt request from the IWDT, the earlier interrupt source is cleared and the new interrupt source is written. For the time period between when 0 is written in each flag and when its value is reflected, see section 25.2.2. IWDTSR : IWDT Status Register.

### 25.3.4 Reset Output

When the IWDT Reset Interrupt Request Select bit (OFS0.IWDTRSTIRQS) in the Option Function Select Register 0 (OFS0) is set to 1, a reset signal is output when an underflow in the counter or a refresh error occurs. Counting down automatically starts after the reset output.

### 25.3.5 Interrupt Sources

When the IWDT Reset Interrupt Request Select bit (OFS0.IWDTRSTIRQS) in the Option Function Select Register 0 (OFS0) is set to 0, an interrupt (IWDT_NMIUNDF) signal occurs when an underflow in the counter or a refresh error occurs. This interrupt can be used as a non-maskable interrupt or an interrupt. For details, see section 13, Interrupt Controller Unit (ICU).

**Table 25.4 IWDT interrupt source**

| Name | Interrupt source | Interrupt to CPU | Start DMAC or DTC |
|---|---|---|---|
| IWDT_NMIUNDF | • Down-counter underflow<br>• Refresh error | Possible | Not possible |

### 25.3.6 Reading the Down-Counter Value

As the counter is a IWDT-dedicated clock (IWDTCLK), the counter value cannot be read directly. The IWDT synchronizes the counter value with the peripheral clock (PCLKB) and stores it in the down-counter value bits (IWDTSR.CNTVAL[13:0]) of the IWDT Status Register. Check these bits to obtain the counter value indirectly.

Reading the counter value requires multiple PCLKB clock cycles (up to 4 clock cycles), and the read counter value might differ from the actual counter value by a value of one count.

Figure 25.5 shows the processing for reading the IWDT counter value when PCLKB > IWDTCLK and the clock division ratio is IWDTCLK.

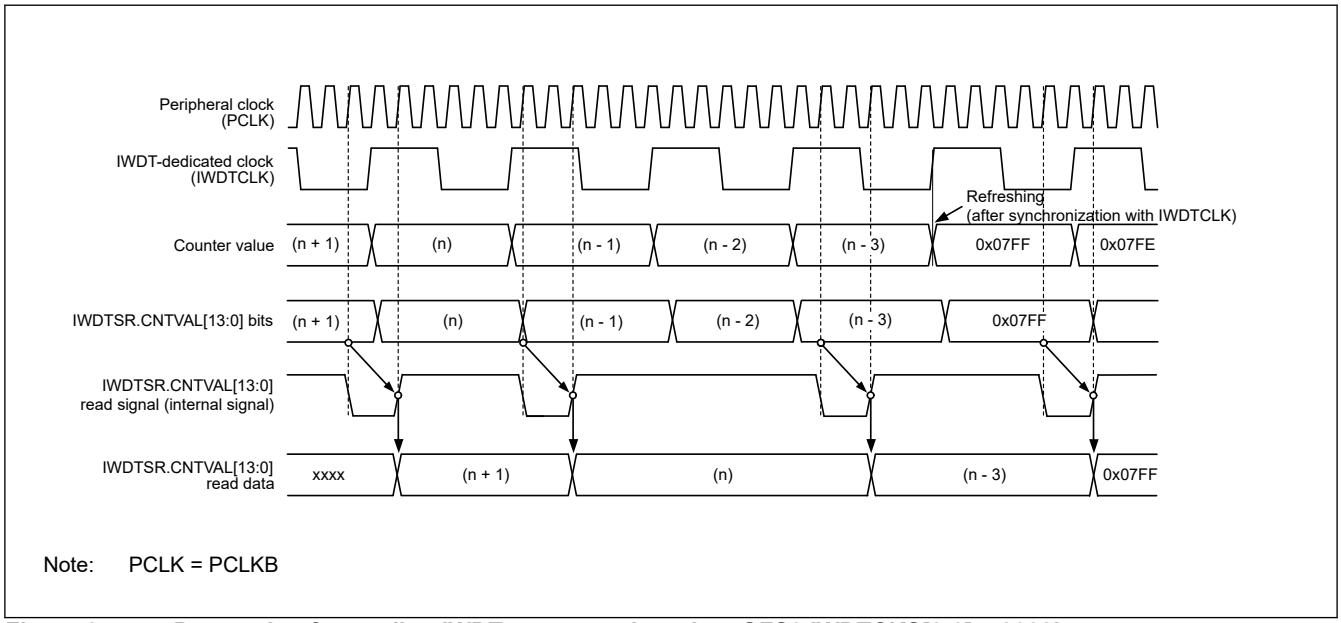



**Figure 25.5 Processing for reading IWDT counter value when OFS0.IWDTCKS[3:0] = 0000b, OFS0.IWDTTOPS[1:0] = 11b**

## 25.4 Output to the Event Link Controller (ELC)

The IWDT is capable of link operation for a specified module when the interrupt request signal is used as an event signal by the event link controller (ELC). The event signal is output by the counter underflow or refresh error.

An event signal is output regardless of the setting of the OFS0.IWDTRSTIRQS bit. An event signal can also be output at generation of the next interrupt source while the Refresh Error flag (IWDTSR.REFEF) or Underflow flag (IWDTSR.UNDFF) is 1. For details, see section 18, Event Link Controller (ELC).

## 25.5 Usage Notes

### 25.5.1 Refresh Operations

While configuring the refresh time, consider variations in the range of errors given the accuracy of PCLKB and IWDTCLK. Set values that ensure refreshing is possible.

### 25.5.2 Clock Division Ratio Setting

Satisfy the frequency of the peripheral module clock (PCLKB) ≥ 4 × (the frequency of the count clock source after division).

### 25.5.3 Constraints on the ICU Event Link Setting Register n (IELSRn) Setting

Setting 0x52 to ICU Event Link Setting Register n (IELSRn.IELS[8:0]) is prohibited when enabling the IWDT reset assertion (OFS0.IWDTRSTIRQS = 0) or when enabling event link operation (ELSRn.ELS[8:0] = 0x52).

# 26. USB 2.0 Full-Speed Module (USBFS)

## 26.1 Overview

The USB 2.0 Full-Speed module (USBFS) operates as a host or device controller compliant with the Universal Serial Bus (USB) specification revision 2.0. The host controller supports USB 2.0 full-speed and low-speed transfers, and the device controller supports USB 2.0 full-speed transfers. The USBFS has an internal USB transceiver and supports all of the transfer types defined in the USB 2.0 specification

The USBFS has FIFO buffer for data transfers, providing a maximum of 10 pipes. Any endpoint number can be assigned to pipes 1 to 9, based on the peripheral devices or the communication requirements for your system.

Table 26.1 lists the USBFS specifications, Figure 26.1 shows a block diagram, and Table 26.2 lists the I/O pins.

**Table 26.1 USBFS specifications**

| Parameter | Specifications |
|---|---|
| Features | ● USB Device Controller (UDC) and USB 2.0 transceiver supporting host controller, and device controller, and On-The-Go (OTG) functions (one channel)<br>● Host and device controller can be switched by software<br>● Self-power and bus power mode can be used<br>● Revision 1.2 of battery charging specification is supported |
| | Host controller features:<br>● Full-speed transfer (12 Mbps) and low-speed transfer (1.5 Mbps)<br>● Automatic scheduling for SOF and packet transmissions<br>● Programmable intervals for isochronous and interrupt transfers<br>● Communications with multiple peripheral devices connected through a single hub |
| | Device controller features:<br>● Full-speed transfer (12 Mbps)*1<br>● Control transfer stage control function<br>● Device state control function<br>● Auto response function for SET_ADDRESS request<br>● SOF interpolation |
| Supported transfer types | ● Control transfer<br>● Bulk transfer<br>● Interrupt transfer<br>● Isochronous transfer |
| Pipe configuration | ● FIFO buffer for USB communication<br>● Up to 10 pipes selectable, including the Default Control Pipe (DCP)<br>● Pipes 1 to 9 assignable to any endpoint number |
| | Transfer conditions specifiable for each pipe:<br>● Pipe 0: Control transfer with 64-byte single buffer<br>● Pipes 1 and 2: Selectable to bulk transfer with 64-byte double buffer or isochronous transfer with 256-byte double buffer<br>● Pipes 3 to 5: Bulk transfer with 64-byte double buffer<br>● Pipes 6 to 9: Interrupt transfer with 64-byte single buffer |
| Other features | ● Reception end function using transaction count<br>● Function that changes the BRDY interrupt event notification timing (BFRE)<br>● Automatic clearing of the FIFO buffer after the data for the pipe specified in the DnFIFO port (n = 0, 1) is read (DCLRM)<br>● NAK setting function for response PID generated on transfer end (SHTNAK)<br>● On-chip pull-up and pull-down resistors for D+ and D–<br>● Compliance with Battery Charging Class Specification Revision 1.2 |
| Module-stop function | Module-stop state can be set |
| TrustZone Filter | Security attribution can be set |

Note 1. Low-speed transfer (1.5 Mbps) is not supported.

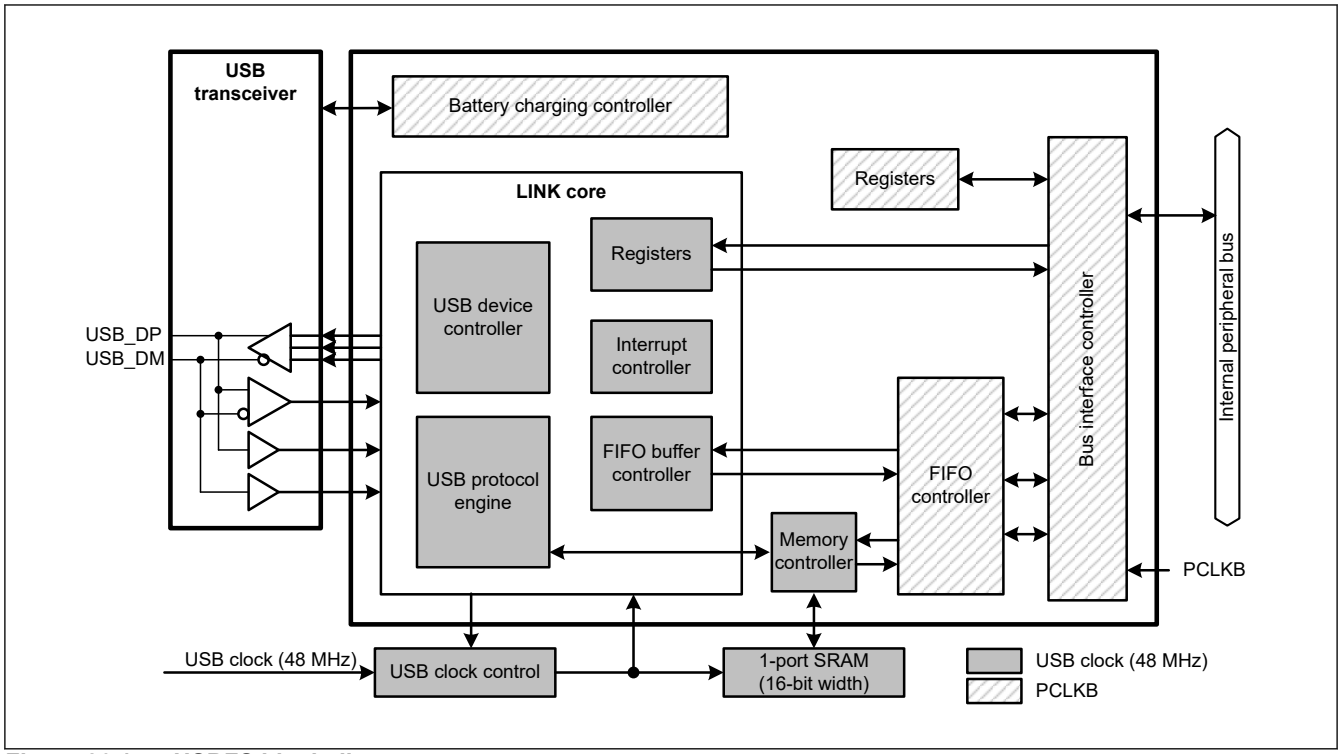



**Figure 26.1 USBFS block diagram**

**Table 26.2 USBFS pin configuration**

| Functoin | Pin name | I/O | Description |
|---|---|---|---|
| USBFS | USB_DP | I/O | D+ I/O pin of the USB on-chip transceiver. This pin should be connected to the D+ pin of the USB bus. |
| | USB_DM | I/O | D– I/O pin of the USB on-chip transceiver. This pin should be connected to the D– pin of the USB bus. |
| | USB_VBUS | Input | USB cable connection monitor pin. This pin should be connected to VBUS of the USB bus. The VBUS pin status (connected or disconnected) can be detected when the USB module is operating as a function controller. |
| | USB_EXICEN | Output | Low-power control signal for external power supply (OTG) chip. |
| | USB_VBUSEN | Output | VBUS (5 V) supply enable signal for external power supply chip. |
| | USB_OVRCURA, USB_OVRCURB | Input | External overcurrent detection signals should be connected to these pins. VBUS comparator signals should be connected to these pins when the OTG power supply chip is connected. |
| | USB_ID | Input | MicroAB connector ID input signal should be connected to this pin during operation in OTG mode. |
| | VCC_USB | Input | Power supply pins. |
| | VSS_USB | Input | Ground pins. |

## 26.2 Register Descriptions

### 26.2.1 SYSCFG : System Configuration Control Register

Base address: USBFS = 0x4009_0000

Offset address: 0x000

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | SCKE | — | — | — | DCFM | DRPD | DPRPU | — | — | — | USBE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | USBE | USBFS Operation Enable<br>0: Disable<br>1: Enable | R/W |
| 2:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | DPRPU | D+ Line Resistor Control<br>0: Disable line pull-up<br>1: Enable line pull-up | R/W |
| 5 | DRPD | D+/D– Line Resistor Control<br>0: Disable line pull-down<br>1: Enable line pull-down | R/W |
| 6 | DCFM | Controller Function Select<br>0: Select device controller<br>1: Select host controller | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |
| 9:8 | — | These bits are read as 0. The write value should be 0. | R/W |
| 10 | SCKE | USB Clock Enable<br>0: Stop clock supply to the USBFS<br>1: Enable clock supply to the USBFS | R/W |
| 15:11 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: After writing 1 to the SCKE bit, read it to confirm that it is set to 1.

### USBE bit (USBFS Operation Enable)

The USBE bit enables or disables operation of the USBFS.

Changing the USBE bit from 1 to 0 initializes the bits listed in Table 26.3. Only change this bit while the SCKE bit is 1. In host controller mode, this bit must be set to 1 after setting the DRPD bit to 1, eliminating SYSSTS0.LNST[1:0] flags chattering, and confirming that the USB bus state is stable.

**Table 26.3 Registers initialized by writing 0 to the SYSCFG.USBE bit**

| Selected function | Register | Bit | Remarks |
|---|---|---|---|
| Device controller | SYSSTS0 | LNST[1:0] | Value is saved in host controller mode |
| | DVSTCTR0 | RHST[2:0] | — |
| | INTSTS0 | DVSQ[2:0] | Value is saved in host controller mode |
| | USBADDR | USBADDR[6:0] | Value is saved in host controller mode |
| | USBREQ | BREQUEST[7:0], BMREQUESTTYPE[7:0] | Value is saved in host controller mode |
| | USBVAL | WVALUE[15:0] | Value is saved in host controller mode |
| | USBINDX | WINDEX[15:0] | Value is saved in host controller mode |
| | USBLENG | WLENTUH[15:0] | Value is saved in host controller mode |
| Host controller | DVSTCTR0 | RHST[2:0] | — |
| | FRMNUM | FRNM[10:0] | Value is saved in device controller mode |

### DPRPU bit (D+ Line Resistor Control)

The DPRPU bit enables or disables pulling up the D+ line in device controller mode.

When the DPRPU bit is set to 1 in device controller mode, the USBFS pulls up the D+ line to notify the USB host that it attached. Changing the DPRPU bit from 1 to 0 releases the pull-up, thereby notifying the USB host that it detached.

Set this bit to 1 in device controller mode and to 0 in host controller mode.

### DRPD bit (D+/D– Line Resistor Control)

TheDRPD bit enables or disables pulling down D+ and D– lines in host controller mode.

Set this bit to 1 in host controller mode and to 0 in device controller mode.

### DCFM bit (Controller Function Select)

The DCFM bit selects the host or device function of the USBFS.

Only change this bit when the DPRPU and DRPD bits are both 0.

### SCKE bit (USB Clock Enable)

The SCKE bit stops or enables the 48-MHz clock supply to the USBFS.

When this bit is 0, only SYSCFG is permitted to be read from and written to; the other registers related to the USB should not be read from or written to.

## 26.2.2 SYSSTS0 : System Configuration Status Register 0

Base address: USBFS = 0x4009_0000

Offset address: 0x004

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | OVCMON[1:0] | | — | — | — | — | — | — | — | HTACT | SOFEA | — | — | IDMON | LNST[1:0] | |
| Value after reset: | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | LNST[1:0] | USB Data Line Status Monitor<br>Indicates the status of the USB data lines, see Table 26.4 | R |
| 2 | IDMON | External ID0 Input Pin Monitor<br>0: USB_ID pin is low<br>1: USB_ID pin is high | R |
| 4:3 | — | These bits are read as 0. | R |
| 5 | SOFEA | Active Monitor When the Host Controller Is Selected<br>0: SOF output stopped<br>1: SOF output operating | R |
| 6 | HTACT | USB Host Sequencer Status Monitor<br>0: Host sequencer completely stopped<br>1: Host sequencer not completely stopped | R |
| 13:7 | — | These bits are read as 0. | R |
| 15:14 | OVCMON[1:0] | External USB_OVRCURA/ USB_OVRCURB Input Pin Monitor<br>OVCMON[1] indicates the USB_OVRCURA pin status. OVCMON[0] indicates the USB_OVRCURB pin status. | R |

Note: The values of the OVCMON[1:0] and IDMON bits depend on the status of the USB_OVRCURA, USB_OVRCURB, and USB_ID pins.

### LNST[1:0] bits (USB Data Line Status Monitor)

The LNST[1:0] bits indicate the state of the USB data lines, D+ and D–. For details, see Table 26.4.

In device controller mode, read the LNST[1:0] bits after connection processing (SYSCFG.DPRPU bit = 1). In host controller mode, read them after enabling pull-down of the lines (SYSCFG.DRPD bit = 1).

**Table 26.4 Status of the USB data bus lines (D+ and D–) (1 of 2)**

| LNST[1:0] bits | During full-speed operation | During low-speed operation |
|---|---|---|
| 00b | SE0 | SE0 |
| 01b | J-State | K-State |
| 10b | K-State | J-State |

**Table 26.4 Status of the USB data bus lines (D+ and D–) (2 of 2)**

| LNST[1:0] bits | During full-speed operation | During low-speed operation |
|---|---|---|
| 11b | SE1 | SE1 |

### SOFEA bit (Active Monitor When the Host Controller Is Selected)

The SOFEA bit is used in host controller mode to check whether the output of the last SOF is complete when the USBFS is suspended because of a 0 setting to the DVSTCTR0.UACT bit.

In host controller mode, check that both the HTACT and SOFEA bits are 0 before setting the SYSCFG.USBE bit to 0 to stop the USBFS or setting the SYSCFG.SCKE bit to 0 to stop the clock signal supply during communication.

### HTACT bit (USB Host Sequencer Status Monitor)

The HTACT bit is set to 0 when the host sequencer of the USBFS is completely stopped.

In host controller mode, check that the HTACT bit is 0 before setting the DVSTCTR0.UACT bit to 0 to place the USBFS in the suspended state or setting the SYSCFG.SCKE bit to 0 to stop the clock signal supply during communication.

### OVCMON[1:0] bits (External USB_OVRCURA/ USB_OVRCURB Input Pin Monitor)

The OVCMON[1:0] bits indicate the status of the overcurrent signals from an external power supply IC.

## 26.2.3 DVSTCTR0 : Device State Control Register 0

Base address: USBFS = 0x4009_0000

Offset address: 0x008

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | HNPBTOA | EXICEN | VBUSEN | WKUP | RWUPE | USBRST | RESUME | UACT | — | RHST[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | RHST[2:0] | USB Bus Reset Status<br>0 0 0: In host controller mode: Communication speed indeterminate (powered state or no connection)<br>In device controller mode: Communication speed indeterminate<br>0 0 1: In host controller mode: Low-speed connection<br>In device controller mode: USB bus reset in progress<br>0 1 0: In host controller mode: Full-speed connection<br>In device controller mode: USB bus reset in progress or full-speed connection<br>0 1 1: Setting prohibited<br>Others: In host controller mode: USB bus reset in progress<br>In device controller mode: Setting prohibited | R |
| 3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | UACT | USB Bus Enable<br>0: Disable downstream port (disable SOF transmission)<br>1: Enable downstream port (enable SOF transmission) | R/W |
| 5 | RESUME | Resume Output<br>0: Do not output resume signal<br>1: Output resume signal | R/W |
| 6 | USBRST | USB Bus Reset Output<br>0: Do not output USB bus reset signal<br>1: Output USB bus reset signal | R/W |
| 7 | RWUPE | Wakeup Detection Enable<br>0: Disable downstream port remote wakeup<br>1: Enable downstream port remote wakeup | R/W |
| 8 | WKUP | Wakeup Output<br>0: Do not output remote wakeup signal<br>1: Output remote wakeup signal | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9 | VBUSEN | USB_VBUSEN Output Pin Control<br>0: Output low on external USB_VBUSEN pin<br>1: Output high on external USB_VBUSEN pin | R/W |
| 10 | EXICEN | USB_EXICEN Output Pin Control<br>0: Output low on external USB_EXICEN pin<br>1: Output high on external USB_EXICEN pin | R/W |
| 11 | HNPBTOA | Host Negotiation Protocol (HNP) Control<br>Use this bit when switching from device B to device A in OTG mode. If the HNPBTOA bit is 1, the internal function control remains in the Suspend state until the HNP processing ends even if SYSCFG.DPRPU = 0 or SYSCFG.DCFM = 1. | R/W |
| 15:12 | — | These bits are read as 0. The write value should be 0. | R/W |

The USBFS controller does not support low-speed connections in device controller mode. When this value is read, abnormal connection processing must be executed in higher level application software.

### RHST[2:0] bits (USB Bus Reset Status)

RHST[2:0] bits indicate the status of the USB bus reset.

In host controller mode, writing 1 to the USBRST bit causes the RHST[2:0] bits to set to 100b. When 0 is written to the USBRST bit and the USBFS ends the SE0 state, the RHST[2:0]bits update to a new value.

In device controller mode, if the USBFS detects a USB bus reset, the RHST[2:0] bits indicate 010b if the DPRPU bit is 1, and a DVST interrupt is generated.

### UACT bit (USB Bus Enable)

When set to 1 in host controller mode, the UACT bit enables USB bus operation by controlling SOF packet transmission to the USB bus in addition to data and reception. The USBFS starts SOF packet output within one frame period after the UACT bit is set to 1. When UACT is set to 0, the USBFS enters the idle state after the SOF packet output.

The USBFS sets the UACT bit to 0 on any of the following conditions:

- A DTCH interrupt is detected during communication (when UACT = 1)
- An EOFERR interrupt is detected during communication (when UACT = 1)

Always write 1 to the UACT bit at the end of the USB bus reset processing (writing 0 to the USBRST bit) or at the end of resume processing from the suspended state (writing 0 to the RESUME bit).

In device controller mode, always set this bit to 0.

### RESUME bit (Resume Output)

The RESUME bit controls the resume signal output in host controller mode.

When this bit is set to 1, the USBFS drives the USB port to the K-state and outputs the resume signal. The USBFS sets the bit to 1 on detection of a remote wakeup signal while the RWUPE bit is 1 and in the USB Suspend state.

The USBFS continues outputting the K-state while the RESUME bit is 1, until the bit is cleared to 0 by software. The RESUME bit must be 1 (resume period) for the time defined in the USB 2.0 specification. Only set this bit to 1 while the interface is in the Suspend state. Write 1 to the UACT bit simultaneously with the end of the resume processing (writing 0 to the RESUME bit).

Always set this bit to 0 in device controller mode.

### USBRST bit (USB Bus Reset Output)

The USBRST bit controls the output of the USB bus signal in host controller mode. When this bit set to 1, the USBFS drives the USB port to the SE0 state to reset the USB bus. The USBFS continues outputting SE0 while the USBRST bit is 1, until the bit is cleared to 0 by software. The USBRST bit must be 1 (USB bus reset period) for the time defined in the USB 2.0 specification. Writing 1 to the USBRST bit during communication (UACT = 1) or during resume processing (RESUME = 1) prevents the USBFS from starting USB bus reset processing until both the UACT and RESUME bits become 0. Write 1 to the UACT bit simultaneously with the end of the USB bus reset processing (writing 0 to the USBRST bit).

Always set this bit to 0 in device controller mode.

### RWUPE bit (Wakeup Detection Enable)

The RWUPE bit enables or disables remote wakeup signals (resume signals) from downstream peripheral devices in host controller mode. When this bit is set to 1, the USBFS detects a remote wakeup signal (K-state for 2.5 µs) from a downstream peripheral device, and performs resume processing, driving the K-state. When the RWUPE bit is set to 0, the USBFS ignores remote wakeup signals (K-states) from peripheral devices connected to the USB port.

Do not stop the internal clock when the RWUPE bit is 1, even in the Suspend state (SYSCFG.SCKE bit must be set to 1).

Always set this bit to 0 in device controller mode.

### WKUP bit (Wakeup Output)

The WKUP bit enables or disables remote wakeup signals (resume signals) to the USB bus in device controller mode.

The USBFS controls the output timing of the remote wakeup signals. When this bit is set to 1, the USBFS clears it to 0 after outputting the K-state for 10 ms. The USB 2.0 specification specifies that the USB bus idle state must be kept for 5 ms or longer before a remote wakeup signal is sent. If the USBFS writes 1 to the WKUP bit immediately after detecting the Suspend state, the K-state is output after 2 ms.

Only write 1 to the WKUP bit when the device is in the Suspend state (INTSTS0.DVSQ[2:0] = 1xxb) and the USB host enables the remote wakeup signal. Do not stop the internal clock while this bit is 1, even in the Suspend state (SYSCFG.SCKE bit must be set to 1).

Always set this bit to 0 in host controller mode.

### HNPBTOA bit (Host Negotiation Protocol (HNP) Control)

TheHNPBTOA bit is used when switching from device B to device A while in OTG mode.

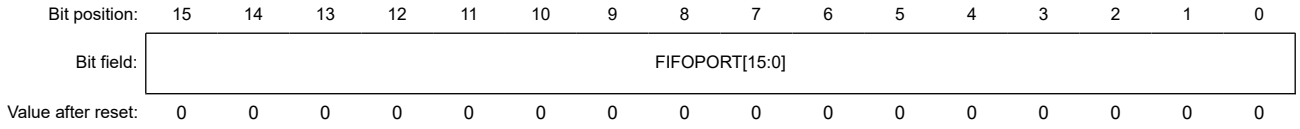
If the HNPBTOA bit is 1, the internal function control maintains the Suspend state until HNP processing ends, even if the SYSCFG.DPRPU bit is set to 0 or the SYSCFG.DCFM bit is set to 1. Resume interrupts (RESM) are not generated even if a falling edge of D+ is detected.

After this bit is set to 1, the HNP processing ends when a host attach event is detected, because of a pull-up by the initiating party, or the HNPBTOA bit is cleared to 0 by software because the HNP processing times out.

## 26.2.4 CFIFO/CFIFOL : CFIFO Port Register

Base address: USBFS = 0x4009_0000

Offset address: 0x014




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | FIFOPORT[15:0]*1 | FIFO Port<br>Read receive data from the FIFO buffer or write transmit data to the FIFO buffer by accessing these bits | R/W |

Note 1. The valid bits depend on the MBW settings (CFIFOSEL.MBW) and BIGEND settings (CFIFOSEL.BIGEND) in the associated port selection register. See Table 26.5 and Table 26.6.

Three FIFO ports are available:

- CFIFO
- D0FIFO
- D1FIFO

Each FIFO port is configured with:

- A port register (CFIFO, D0FIFO, or D1FIFO) that handles reading of data from the FIFO buffer and writing of data to the FIFO buffer
- A port selection register (CFIFOSEL, D0FIFOSEL, or D1FIFOSEL) that selects the pipe assigned to the FIFO port

- A port control register (CFIFOCTR, D0FIFOCTR, or D1FIFOCTR)

Each FIFO port has the following constraints:

- Access to the FIFO buffer for DCP control transfers is through the CFIFO port
- Access to the FIFO buffer for DMA or DTC transfers is through the D0FIO or D1FIFO port
- The D0FIFO and D1FIFO ports can also be accessed by the CPU
- When using functions specific to the FIFO port, such as the DMA or DTC transfer function, you cannot change the pipe number selected in the CURPIPE[3:0] bits of the port selection register
- Registers configuring a FIFO port do not affect other FIFO ports
- The same pipe must not be assigned to two or more FIFO ports
- There are two FIFO buffer states, one giving access rights to the CPU and the other to the serial interface engine (SIE). When the SIE has access rights, the FIFO buffer cannot be accessed by the CPU

#### FIFOPORT[15:0] bits (FIFO Port)

When the FIFOPORT[15:0] bit is accessed, the USBFS reads the received data from the FIFO buffer or writes the transmit data to the FIFO buffer. The FIFO port register can be accessed only when the FRDY bit in the associated port control register (CFIFOCTR, D0FIFOCTR, or D1FIFOCTR) is 1.

The valid bits in the FIFO port register depend on the MBW and BIGEND settings in the port selection register (CFIFOSEL, D0FIFOSEL, or D1FIFOSEL). See Table 26.5 and Table 26.6.

**Table 26.5 Endian operation in 16-bit access**

| CFIFOSEL.BIGEND bit | Bits [15:8] | Bits [7:0] |
|---|---|---|
| 0 | N + 1 data | N + 0 data |
| 1 | N + 0 data | N + 1 data |

**Table 26.6 Endian operation in 8-bit access**

| CFIFOSEL.BIGEND bit | Bits [15:8] | Bits [7:0] |
|---|---|---|
| 0 | Access prohibited*1 | N + 0 data |
| 1 | Access prohibited*1 | N + 0 data |

Note 1. Writing to or reading from these areas is not allowed.

## 26.2.5 DnFIFO/DnFIFOL : DnFIFO Port Register (n = 0, 1)

Base address: USBFS = 0x4009_0000

Offset address: 0x018 + 0x4 × n

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | FIFOPORT[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | FIFOPORT[15:0]*1 | FIFO Port<br>Read receive data from the FIFO buffer or write transmit data to the FIFO buffer by accessing these bits | R/W |

Note 1. The valid bits depend on the MBW settings (CFIFOSEL.MBW, D0FIFOSEL.MBW, and D1FIFOSEL.MBW) and BIGEND settings (CFIFOSEL.BIGEND, D0FIFOSEL.BIGEND, and D1FIFOSEL.BIGEND) in the associated port selection register. See Table 26.7 and Table 26.8.

Three FIFO ports are available:

- CFIFO
- D0FIFO

- D1FIFO

Each FIFO port is configured with:

- A port register (CFIFO, D0FIFO, or D1FIFO) that handles reading of data from the FIFO buffer and writing of data to the FIFO buffer
- A port selection register (CFIFOSEL, D0FIFOSEL, or D1FIFOSEL) that selects the pipe assigned to the FIFO port
- A port control register (CFIFOCTR, D0FIFOCTR, or D1FIFOCTR)

Each FIFO port has the following constraints:

- Access to the FIFO buffer for DCP control transfers is through the CFIFO port
- Access to the FIFO buffer for DMA or DTC transfers is through the D0FIO or D1FIFO port
- The D0FIFO and D1FIFO ports can also be accessed by the CPU
- When using functions specific to the FIFO port, such as the DMA or DTC transfer function, you cannot change the pipe number selected in the CURPIPE[3:0] bits of the port selection register
- Registers configuring a FIFO port do not affect other FIFO ports
- The same pipe must not be assigned to two or more FIFO ports
- There are two FIFO buffer states, one giving access rights to the CPU and the other to the serial interface engine (SIE). When the SIE has access rights, the FIFO buffer cannot be accessed by the CPU

#### FIFOPORT[15:0] bits (FIFO Port)

When the FIFOPORT bit is accessed, the USBFS reads the received data from the FIFO buffer or writes the transmit data to the FIFO buffer. The FIFO port register can be accessed only when the FRDY bit in the associated port control register (CFIFOCTR, D0FIFOCTR, or D1FIFOCTR) is 1.

The valid bits in the FIFO port register depend on the MBW and BIGEND settings in the port selection register (CFIFOSEL, D0FIFOSEL, or D1FIFOSEL). See Table 26.7 and Table 26.8.

**Table 26.7 Endian operation in 16-bit access**

| CFIFOSEL.BIGEND bit<br>D0FIFOSEL.BIGEND bit<br>D1FIFOSEL.BIGEND bit | Bits [15:8] | Bits [7:0] |
|---|---|---|
| 0 | N + 1 data | N + 0 data |
| 1 | N + 0 data | N + 1 data |

**Table 26.8 Endian operation in 8-bit access**

| CFIFOSEL.BIGEND bit<br>D0FIFOSEL.BIGEND bit<br>D1FIFOSEL.BIGEND bit | Bits [15:8] | Bits [7:0] |
|---|---|---|
| 0 | Access prohibited*1 | N + 0 data |
| 1 | Access prohibited*1 | N + 0 data |

Note 1. Writing to or reading from these areas is not allowed.

## 26.2.6 CFIFOSEL : CFIFO Port Select Register

Base address: USBFS = 0x4009_0000

Offset address: 0x020

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | RCNT | REW | — | — | — | MBW | — | BIGEND | — | — | ISEL | — | CURPIPE[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | CURPIPE[3:0] | CFIFO Port Access Pipe Specification<br>0x0: Default Control Pipe<br>0x1: Pipe 1<br>0x2: Pipe 2<br>0x3: Pipe 3<br>0x4: Pipe 4<br>0x5: Pipe 5<br>0x6: Pipe 6<br>0x7: Pipe 7<br>0x8: Pipe 8<br>0x9: Pipe 9<br>Others: Setting prohibited | R/W |
| 4 | — | This bit is read as 0. The write value should be 0. This bit is read as 0. The write value should be 0. | R/W |
| 5 | ISEL | CFIFO Port Access Direction When DCP Is Selected<br>0: Select reading from the FIFO buffer<br>1: Select writing to the FIFO buffer | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. These bits are read as 0. The write value should be 0. | R/W |
| 8 | BIGEND | CFIFO Port Endian Control<br>0: Little endian<br>1: Big endian | R/W |
| 9 | — | This bit is read as 0. The write value should be 0. | R/W |
| 10 | MBW | CFIFO Port Access Bit Width<br>0: 8-bit width<br>1: 16-bit width | R/W |
| 13:11 | — | These bits are read as 0. The write value should be 0. | R/W |
| 14 | REW | Buffer Pointer Rewind<br>0: Do not rewind buffer pointer<br>1: Rewind buffer pointer | W*1 |
| 15 | RCNT | Read Count Mode<br>0: The DTLN[8:0] bits (CFIFOCTR.DTLN[8:0], D0FIFOCTR.DTLN[8:0], D1FIFOCTR.DTLN[8:0]) are cleared when all receive data is read from the CFIFO. In double buffer mode, the DTLN[8:0] value is cleared when all data is read from only a single plane.<br>1: The DTLN[8:0] bits are decremented each time the receive data is read from the CFIFO. | R/W |

Note 1. Only 0 can be read.

Do not specify the same pipe number in the CURPIPE[3:0] bits in the CFIFOSEL, D0FIFOSEL, and D1FIFOSEL registers. When the CURPIPE[3:0] bits in the D0FIFOSEL and D1FIFOSEL registers are set to 0000b, no pipe is selected.

Do not change the pipe number while DMA or DTC transfer is enabled.

### CURPIPE[3:0] bits (CFIFO Port Access Pipe Specification)

The CURPIPE[3:0] bits specify the pipe number to use for reading or writing data through the CFIFO port. After writing to these bits, read them to check that the written value agrees with the read value before proceeding to the next process. Do not set the same pipe number to the CURPIPE[3:0] bits in CFIFOSEL, D0FIFOSEL, and D1FIFOSEL.

During FIFO buffer access, even when an attempt is made to change the CURPIPE[3:0] setting, the current access setting is retained until access is complete.

### ISEL bit (CFIFO Port Access Direction When DCP Is Selected)

After writing a new value to the ISEL bit with the DCP as the selected pipe, read the ISEL bit to check that the written value agrees with the read value before proceeding to the next process. Set the ISEL and CURPIPE[3:0] bits simultaneously.

### MBW bit (CFIFO Port Access Bit Width)

The MBW bit specifies the bit width for accessing the CFIFO port.

When the selected pipe is receiving, set the CURPIPE[3:0] and MBW bits simultaneously. After a write to these bits starts a data read from the FIFO buffer, do not change the MBW bit until all of the data is read.

When the selected pipe is transmitting, the bit width cannot be changed from 8-bit to 16-bit while data is being written to the FIFO buffer.

An odd number of bytes can also be written through byte-access control even when 16-bit width is selected.

#### REW bit (Buffer Pointer Rewind)

The REW bit specifies whether to rewind the buffer pointer.

When the selected pipe is receiving, setting this bit to 1 while the FIFO buffer is being read allows re-reading of the FIFO buffer from the first data. In double buffering, this setting enables re-reading of the currently-read FIFO buffer plane from the first entry.

Do not set this bit to 1 while simultaneously changing the CURPIPE[3:0] bits. Before setting the REW bit to 1, be sure to check that the FRDY bit is 1.

To rewrite to the FIFO buffer from the first data for the transmitting pipe, use the BCLR bit.

### 26.2.7 DnFIFOSEL : DnFIFO Port Select Register (n = 0, 1)

Base address: USBFS = 0x4009_0000

Offset address: 0x028 + 0x4 × n

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | RCNT | REW | DCLRM | DREQE | — | MBW | — | BIGEND | — | — | — | — | CURPIPE[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | CURPIPE[3:0] | FIFO Port Access Pipe Specification<br>0x0: Default Control Pipe<br>0x1: Pipe 1<br>0x2: Pipe 2<br>0x3: Pipe 3<br>0x4: Pipe 4<br>0x5: Pipe 5<br>0x6: Pipe 6<br>0x7: Pipe 7<br>0x8: Pipe 8<br>0x9: Pipe 9<br>Others: Setting prohibited | R/W |
| 7:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | BIGEND | FIFO Port Endian Control<br>0: Little endian<br>1: Big endian | R/W |
| 9 | — | This bit is read as 0. The write value should be 0. | R/W |
| 10 | MBW | FIFO Port Access Bit Width<br>0: 8-bit width<br>1: 16-bit width | R/W |
| 11 | — | This bit is read as 0. The write value should be 0. | R/W |
| 12 | DREQE | DMA/DTC Transfer Request Enable<br>0: Disable DMA/DTC transfer request<br>1: Enable DMA/DTC transfer request | R/W |
| 13 | DCLRM | Auto Buffer Memory Clear Mode Accessed after Specified Pipe Data is Read<br>0: Disable auto buffer clear mode<br>1: Enable auto buffer clear mode | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14 | REW | Buffer Pointer Rewind<br>0: Do not rewind buffer pointer<br>1: Rewind buffer pointer | W |
| 15 | RCNT | Read Count Mode<br>0: Clear DTLN[8:0] bits in (CFIFOCTR.DTLN[8:0], D0FIFOCTR.DTLN[8:0], D1FIFOCTR.DTLN[8:0]) when all receive data is read from DnFIFO (after read of a single plane in double buffer mode)<br>1: Decrement DTLN[8:0] bits each time receive data is read from DnFIFO | R/W |

The same pipe must not be specified in the CURPIPE[3:0] bits in the CFIFOSEL, D0FIFOSEL, and D1FIFOSEL registers. When the CURPIPE[3:0] bits in the D0FIFOSEL and D1FIFOSEL registers are set to 0000b, no pipe is selected. The pipe number must not be changed while DMA or DTC transfer is enabled.

### CURPIPE[3:0] bits (FIFO Port Access Pipe Specification)

The CURPIPE[3:0] bits specify the pipe number to use for reading or writing data through the DnFIFO port. After writing to these bits, read them to check that the written value agrees with the read value before proceeding to the next process. Do not set the same pipe number to the CURPIPE[3:0] bits in CFIFOSEL, D0FIFOSEL, and D1FIFOSEL.

During FIFO buffer access, even when an attempt is made to change the CURPIPE[3:0] setting, the current access setting is retained until access is complete.

### MBW bit (FIFO Port Access Bit Width)

The MBW bit specifies the bit width for accessing the DnFIFO port.

When the selected pipe is receiving, after a write to these bits starts a data read from the FIFO buffer, do not change the MBW bit until all of the data is read. Set the CURPIPE[3:0] and MBW bits simultaneously.

When the selected pipe is transmitting, the bit width cannot be changed from 8-bit to 16-bit while data is being written to the FIFO buffer.

An odd number of bytes can also be written through byte-access control even when 16-bit width is selected.

### DREQE bit (DMA/DTC Transfer Request Enable)

The DREQE bit enables or disables issuing of DMA or DTC transfer requests. To enable DMA or DTC transfer requests, set this bit to 1 after setting the CURPIPE[3:0] bits. To change the CURPIPE[3:0] setting, first set this bit to 0.

### DCLRM bit (Auto Buffer Memory Clear Mode Accessed after Specified Pipe Data is Read)

The DCLRM bit enables or disables automatic FIFO buffer clearing after data in the selected pipe is read.

When this bit is set to 1, on receiving a zero-length packet while the FIFO buffer assigned to the selected pipe is empty, or when reading of a received short packet is complete while the PIPECFG.BFRE bit is 1, the USBFS sets the BCLR bit in the FIFO port control register to 1.

When using the USBFS with the SOFCFG.BRDYM bit set to 1, set this bit to 0.

### REW bit (Buffer Pointer Rewind)

The REW bit specifies whether to rewind the buffer pointer.

When the selected pipe is receiving, setting this bit to 1 while the FIFO buffer is being read allows re-reading of the FIFO buffer from the first data. In double buffering, this setting enables re-reading of the currently-read FIFO buffer plane from the first entry.

Do not set this bit to 1 while simultaneously changing the CURPIPE[3:0] bits. Before setting the bit to 1, be sure to check that the FRDY bit is 1.

To rewrite to the FIFO buffer from the first data for the transmitting pipe, use the BCLR bit.

### RCNT bit (Read Count Mode)

The RCNT bit specifies the read mode for the value in the D0FIFOCTL.DTLN bit and D1FIFOCTL.DTLN bit. When accessing DnFIFO with the PIPECFG.BFRE bit set to 1, set the RCNT bit to 0.

## 26.2.8 CFIFOCTR : CFIFO Port Control Register

Base address: USBFS = 0x4009_0000

Offset address: 0x022

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BVAL | BCLR | FRDY | — | — | — | — | DTLN[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | DTLN[8:0] | Receive Data Length<br>Indicates the receive data length.<br>The meaning of the values differs depending on the RCNT bit setting in the port select register. For details, see the description of the DTLN[8:0] bits. | R |
| 12:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13 | FRDY | FIFO Port Ready<br>0: FIFO port access disabled<br>1: FIFO port access enabled | R |
| 14 | BCLR | CPU Buffer Clear<br>0: No operation<br>1: Clear FIFO buffer on the CPU side | W |
| 15 | BVAL | Buffer Memory Valid Flag<br>0: Invalid (writing 0 has no effect)<br>1: Writing ended | R/W |

The CFIFOCTR, D0FIFOCTR, and D1FIFOCTR registers correspond to the CFIFO, D0FIFO, and D1FIFO buffers.

### DTLN[8:0] bits (Receive Data Length)

The DTLN[8:0] bits indicate the length of the receive data.

While the FIFO buffer is being read, the DTLN[8:0] bits indicate different values depending on the DnFIFOSEL.RCNT bit (n = 0, 1), as follows:

- RCNT = 0

  The USBFS sets the DTLN[8:0] bits to indicate the length of the receive data until the CPU or DMA/DTC has read all of the received data from a single FIFO buffer plane.

  While the PIPECFG.BFRE bit = 1, the USBFS retains the length of the receive data until the BCLR bit is set to 1, even after all the data is read.

- RCNT = 1

  The USBFS decrements the value indicated in the DTLN[8:0] bits each time data is read from the FIFO buffer. The value is decremented by 1 when MBW = 0, and by 2 when MBW = 1.

  The USBFS sets these bits to 0 when all the data is read from one FIFO buffer plane. In double buffer mode, if data is received in one FIFO buffer plane before all of the data is read from the other plane, the USBFS sets these bits to indicate the length of the receive data in the former plane when all of the data is read from the latter plane.

### FRDY bit (FIFO Port Ready)

The FRDY bit indicates whether the FIFO port can be accessed by the CPU or DMA/DTC.

In the following cases, the USBFS sets the FRDY bit to 1 but data cannot be read through the FIFO port because there is no data to be read:

- A zero-length packet is received when the FIFO buffer assigned to the selected pipe is empty
- A short packet is received and the data is completely read while the PIPECFG.BFRE bit = 1

In these cases, set the BCLR bit to 1 to clear the FIFO buffer, and enable transmission and reception of the next data.

### BCLR bit (CPU Buffer Clear)

Set the BCLR bit to 1 to clear the FIFO buffer on the CPU side for the selected pipe.

When double buffer mode is set for the FIFO buffer assigned to the selected pipe, the USBFS clears only one plane of the FIFO buffer even when both planes are read-enabled.

When the DCP is the selected pipe, setting the BCLR bit to 1 allows the USBFS to clear the FIFO buffer regardless of whether the CPU or SIE has access rights. To clear the buffer when the SIE has access rights, set the DCPCTR.PID[1:0] bits to 00b (NAK response) before setting the BCLR bit to 1.

When the selected pipe is transmitting, if 1 is written to the BVAL flag and the BCLR bit simultaneously, the USBFS clears the data that is already written, enabling transmission of a zero-length packet.

When the selected pipe is not the DCP, only write 1 to the BCLR bit while the FRDY bit in the FIFO port control register is 1 (set by the USBFS).

### BVAL flag (Buffer Memory Valid Flag)

Set the BVAL flag to 1 when data is completely written to the FIFO buffer on the CPU side for the pipe selected in CURPIPE[3:0].

When the selected pipe is transmitting, set this flag to 1 in the following cases:

- To transmit a short packet, set this flag to 1 after data is written
- To transmit a zero-length packet, set this flag to 1 before data is written to the FIFO buffer

The USBFS then switches the FIFO buffer from the CPU side to the SIE side, enabling transmission.

When data of the maximum packet size is written for the pipe in continuous transfer mode, the USBFS sets the BVAL flag to 1 and switches the FIFO buffer from the CPU side to the SIE side, enabling transmission.

Only write 1 to the BVAL flag while the FRDY bit is 1 (set by the USBFS). When the selected pipe is receiving, do not set the BVAL flag to 1.

## 26.2.9 DnFIFOCTR : DnFIFO Port Control Register (n = 0, 1)

Base address: USBFS = 0x4009_0000

Offset address: 0x02A + 0x4 × n

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BVAL | BCLR | FRDY | — | — | — | — | DTLN[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | DTLN[8:0] | Receive Data Length<br>Indicates the receive data length.<br>The meaning of the values differs depending on the RCNT bit setting in the port select register. For details, see the description of the DTLN[8:0] bits. | R |
| 12:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13 | FRDY | FIFO Port Ready<br>0: FIFO port access disabled<br>1: FIFO port access enabled | R |
| 14 | BCLR | CPU Buffer Clear<br>0: No operation<br>1: Clear FIFO buffer on the CPU side | R/W*1 |
| 15 | BVAL | Buffer Memory Valid Flag<br>0: Invalid (writing 0 has no effect)<br>1: Writing ended | R/W |

Note 1. Only 0 can be read.

The CFIFOCTR, D0FIFOCTR, and D1FIFOCTR registers correspond to the CFIFO, D0FIFO, and D1FIFO buffers.

#### DTLN[8:0] bits (Receive Data Length)

The DTLN[8:0] bits indicate the length of the receive data.

While the FIFO buffer is being read, the DTLN[8:0] bits indicate different values depending on the DnFIFOSEL.RCNT bit (n = 0, 1), as follows:

- RCNT = 0

  The USBFS sets the DTLN[8:0] bits to indicate the length of the receive data until the CPU or DMA/DTC has read all of the received data from a single FIFO buffer plane.

  While the PIPECFG.BFRE bit = 1, the USBFS retains the length of the receive data until the BCLR bit is set to 1, even after all the data is read.

- RCNT = 1

  The USBFS decrements the value indicated in the DTLN[8:0] bits each time data is read from the FIFO buffer. The value is decremented by 1 when MBW = 0, and by 2 when MBW = 1.

  The USBFS sets these bits to 0 when all the data is read from one FIFO buffer plane. In double buffer mode, if data is received in one FIFO buffer plane before all of the data is read from the other plane, the USBFS sets these bits to indicate the length of the receive data in the former plane when all of the data is read from the latter plane.

#### FRDY bit (FIFO Port Ready)

The FRDY bit indicates whether the FIFO port can be accessed by the CPU or DMA/DTC.

In the following cases, the USBFS sets the FRDY bit to 1 but data cannot be read through the FIFO port because there is no data to be read:

- A zero-length packet is received when the FIFO buffer assigned to the selected pipe is empty
- A short packet is received and the data is completely read while the PIPECFG.BFRE bit = 1

In these cases, set the BCLR bit to 1 to clear the FIFO buffer, and enable transmission and reception of the next data.

#### BCLR bit (CPU Buffer Clear)

Set the BCLR bit to 1 to clear the FIFO buffer on the CPU side for the selected pipe.

When double buffer mode is set for the FIFO buffer assigned to the selected pipe, the USBFS clears only one plane of the FIFO buffer even when both planes are read-enabled.

When the DCP is the selected pipe, setting the BCLR bit to 1 allows the USBFS to clear the FIFO buffer regardless of whether the CPU or SIE has access rights. To clear the buffer when the SIE has access rights, set the DCPCTR.PID[1:0] bits to 00b (NAK response) before setting the BCLR bit to 1.

When the selected pipe is transmitting, if 1 is written to the BVAL flag and the BCLR bit simultaneously, the USBFS clears the data that is already written, enabling transmission of a zero-length packet.

When the selected pipe is not the DCP, only write 1 to the BCLR bit while the FRDY bit in the FIFO port control register is 1 (set by the USBFS).

#### BVAL flag (Buffer Memory Valid Flag)

Set the BVAL flag to 1 when data is completely written to the FIFO buffer on the CPU side for the pipe selected in CURPIPE[3:0].

When the selected pipe is transmitting, set this flag to 1 in the following cases:

- To transmit a short packet, set this flag to 1 after data is written
- To transmit a zero-length packet, set this flag to 1 before data is written to the FIFO buffer

The USBFS then switches the FIFO buffer from the CPU side to the SIE side, enabling transmission.

When data of the maximum packet size is written for the pipe in continuous transfer mode, the USBFS sets the BVAL flag to 1 and switches the FIFO buffer from the CPU side to the SIE side, enabling transmission.

Only write 1 to the BVAL flag while the FRDY bit is 1 (set by the USBFS). When the selected pipe is receiving, do not set the BVAL flag to 1.

## 26.2.10 INTENB0 : Interrupt Enable Register 0

Base address: USBFS = 0x4009_0000

Offset address: 0x030

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | VBSE | RSME | SOFE | DVSE | CTRE | BEMPE | NRDYE | BRDYE | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | BRDYE | Buffer Ready Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 9 | NRDYE | Buffer Not Ready Response Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 10 | BEMPE | Buffer Empty Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 11 | CTRE | Control Transfer Stage Transition Interrupt Enable *1<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 12 | DVSE | Device State Transition Interrupt Enable *1<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 13 | SOFE | Frame Number Update Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 14 | RSME | Resume Interrupt Enable*1<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 15 | VBSE | VBUS Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |

Note 1. The RSME, DVSE, and CTRE bits can only be set to 1 in device controller mode. Do not set these bits to 1 in host controller mode.

When a status flag in the INTSTS0 register sets to 1 and the associated interrupt request enable bit setting in theINTENB0 register is 1, the USBFS issues a USBFS interrupt request.

Regardless of the INTENB0 register setting, the status flag in the INTSTS0 register sets to 1 in response to a state change that satisfies the associated condition.

When an interrupt request enable bit in the INTENB0 register is switched from 0 to 1 while the associated status flag in the INTSTS0 register is set to 1, a USBFS interrupt is requested.

## 26.2.11 INTENB1 : Interrupt Enable Register 1

Base address: USBFS = 0x4009_0000

Offset address: 0x032

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | OVRCRE | BCHGE | — | DTCHE | ATTCHE | — | — | — | — | EOFERRE | SIGNE | SACKE | — | — | — | PDDETINTE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PDDETINTE | PDDETINT Detection Interrupt Request Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 3:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | SACKE | Setup Transaction Normal Response Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 5 | SIGNE | Setup Transaction Error Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 6 | EOFERRE | EOF Error Detection Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 10:7 | — | These bits are read as 0. The write value should be 0. | R/W |
| 11 | ATTCHE | Connection Detection Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 12 | DTCHE | Disconnection Detection Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 13 | — | This bit is read as 0. The write value should be 0. | R/W |
| 14 | BCHGE | USB Bus Change Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 15 | OVRCRE | Overcurrent Input Change Interrupt Enable<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |

Note: The bits in INTENB1 can only be set to 1 in host controller mode. Do not set these bits to 1 in device controller mode.

INTENB1 specifies the interrupt masks in host controller mode and for the setup transaction.

When a status flag in the INTSTS1 register sets to 1 and the associated interrupt request enable bit setting in theINTENB1 register is 1, the USBFS issues a USBFS interrupt request.

Regardless of the INTENB1 register setting, the status flag in the INTSTS1 register sets to 1 in response to a state change that satisfies the associated condition.

When an interrupt request enable bit in the INTENB1 register is switched from 0 to 1 while the associated status flag in the INTSTS1 register is set to 1, a USBFS interrupt is requested.

Do not enable interrupts in device controller mode.

## 26.2.12 BRDYENB : BRDY Interrupt Enable Register

Base address: USBFS = 0x4009_0000

Offset address: 0x036

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | PIPE9BRDYE | PIPE8BRDYE | PIPE7BRDYE | PIPE6BRDYE | PIPE5BRDYE | PIPE4BRDYE | PIPE3BRDYE | PIPE2BRDYE | PIPE1BRDYE | PIPE0BRDYE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PIPE0BRDYE | BRDY Interrupt Enable for Pipe 0<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | PIPE1BRDYE | BRDY Interrupt Enable for Pipe 1<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 2 | PIPE2BRDYE | BRDY Interrupt Enable for Pipe 2<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 3 | PIPE3BRDYE | BRDY Interrupt Enable for Pipe 3<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 4 | PIPE4BRDYE | BRDY Interrupt Enable for Pipe 4<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 5 | PIPE5BRDYE | BRDY Interrupt Enable for Pipe 5<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 6 | PIPE6BRDYE | BRDY Interrupt Enable for Pipe 6<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 7 | PIPE7BRDYE | BRDY Interrupt Enable for Pipe 7<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 8 | PIPE8BRDYE | BRDY Interrupt Enable for Pipe 8<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 9 | PIPE9BRDYE | BRDY Interrupt Enable for Pipe 9<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 15:10 | — | This bit is read as 0. The write value should be 0. | R/W |

The BRDYENB register enables or disables the INTSTS0.BRDY bit to be set to 1 when a BRDY interrupt is detected for each pipe.

When a status flag in the BRDYSTS register sets to 1 and the associated PIPEnBRDYE bit (n = 0 to 9) setting in theBRDYENB register is 1, the INTSTS0.BRDY flag sets to 1. In this case, if the BRDYE bit in INTENB0 is 1, the USBFS generates a BRDY interrupt request. While at least one PIPEnBRDY bit indicates 1, the USB generates the BRDY interrupt request when the associated interrupt request enable bit in the BRDYENB register is changed from 0 to 1 by software.

## 26.2.13 NRDYENB : NRDY Interrupt Enable Register

Base address: USBFS = 0x4009_0000

Offset address: 0x038

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | PIPE9NRDYE | PIPE8NRDYE | PIPE7NRDYE | PIPE6NRDYE | PIPE5NRDYE | PIPE4NRDYE | PIPE3NRDYE | PIPE2NRDYE | PIPE1NRDYE | PIPE0NRDYE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PIPE0NRDYE | NRDY Interrupt Enable for Pipe 0<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 1 | PIPE1NRDYE | NRDY Interrupt Enable for Pipe 1<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2 | PIPE2NRDYE | NRDY Interrupt Enable for Pipe 2<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 3 | PIPE3NRDYE | NRDY Interrupt Enable for Pipe 3<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 4 | PIPE4NRDYE | NRDY Interrupt Enable for Pipe 4<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 5 | PIPE5NRDYE | NRDY Interrupt Enable for Pipe 5<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 6 | PIPE6NRDYE | NRDY Interrupt Enable for Pipe 6<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 7 | PIPE7NRDYE | NRDY Interrupt Enable for Pipe 7<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 8 | PIPE8NRDYE | NRDY Interrupt Enable for Pipe 8<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 9 | PIPE9NRDYE | NRDY Interrupt Enable for Pipe 9<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 15:10 | — | This bit is read as 0. The write value should be 0. | R/W |

The NRDYENB register enables or disables the INTSTS0.NRDY bit to be set to 1 when a NRDY interrupt is detected for each pipe.

When a status flag in the NRDYSTS register sets to 1 and the associated PIPEnNRDYE (n = 0 to 9) bit setting in theNRDYENB register is 1, the INTSTS0.NRDY flag sets to 1. In this case, if the NRDYE bit in INTENB0 is 1, the USBFS generates a NRDY interrupt request. While at least one PIPEnNRDY bit indicates 1, the USBFS generates the NRDY interrupt request when the associated interrupt request enable bit in the NRDYENB register is changed from 0 to 1 by software.

## 26.2.14 BEMPENB : BEMP Interrupt Enable Register

Base address: USBFS = 0x4009_0000

Offset address: 0x03A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | PIPE9BEMPE | PIPE8BEMPE | PIPE7BEMPE | PIPE6BEMPE | PIPE5BEMPE | PIPE4BEMPE | PIPE3BEMPE | PIPE2BEMPE | PIPE1BEMPE | PIPE0BEMPE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PIPE0BEMPE | BEMP Interrupt Enable for Pipe 0<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 1 | PIPE1BEMPE | BEMP Interrupt Enable for Pipe 1<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 2 | PIPE2BEMPE | BEMP Interrupt Enable for Pipe 2<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | PIPE3BEMPE | BEMP Interrupt Enable for Pipe 3<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 4 | PIPE4BEMPE | BEMP Interrupt Enable for Pipe 4<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 5 | PIPE5BEMPE | BEMP Interrupt Enable for Pipe 5<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 6 | PIPE6BEMPE | BEMP Interrupt Enable for Pipe 6<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 7 | PIPE7BEMPE | BEMP Interrupt Enable for Pipe 7<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 8 | PIPE8BEMPE | BEMP Interrupt Enable for Pipe 8<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 9 | PIPE9BEMPE | BEMP Interrupt Enable for Pipe 9<br>0: Disable interrupt request<br>1: Enable interrupt request | R/W |
| 15:10 | — | This bit is read as 0. The write value should be 0. | R/W |

The BEMPENB register enables or disables the INTSTS0.BEMP bit to be set to 1 when a BEMP interrupt is detected for each pipe.

When a status flag in the BEMPSTS register sets to 1 and the associated PIPEnBEMPE (n = 0 to 9) bit setting in the BEMPENB register is 1, the INTSTS0.BEMP flag sets to 1. In this case, if the BEMPE bit in INTENB0 is 1, the USBFS generates a BEMP interrupt request. While at least one PIPEnBEMP bit indicates 1, the USBFS generates the BEMP interrupt request when the associated interrupt request enable bit in the BEMPENB register is changed from 0 to 1 by software.

## 26.2.15 SOFCFG : SOF Output Configuration Register

Base address: USBFS = 0x4009_0000

Offset address: 0x03C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | TRNENSEL | — | BRDYM | — | EDGESTS | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | EDGESTS | Edge Interrupt Output Status Monitor*1<br>Indicates 1 during the edge processing of an edge interrupt output signal. | R |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | BRDYM | BRDY Interrupt Status Clear Timing<br>0: Clear BRDY flag by software<br>1: Clear BRDY flag by the USBFS through a data read from the FIFO buffer or data write to the FIFO buffer | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |
| 8 | TRNENSEL | Transaction-Enabled Time Select*1<br>0: Not low-speed communication<br>1: Low-speed communication | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Confirm that the EDGESTS flag is 0 before stopping the clock supply to the USBFS.

### EDGESTS bit (Edge Interrupt Output Status Monitor)

The EDGESTS bit indicates 1 during the edge processing of an edge interrupt output signal. Confirm that this bit is 0 before stopping the clock supply to the USBFS.

### BRDYM bit (BRDY Interrupt Status Clear Timing)

The BRDYM bit specifies how the BRDY interrupt status flags for the pipes are cleared.

### TRNENSEL bit (Transaction-Enabled Time Select)

When the USB port is in use for full- or low-speed communications, the TRNENSEL bit specifies the timing with which the USBFS issues tokens in a frame (transaction-enabled time).

Set this bit to 1 when a low-speed device is connected. The bit is only valid in host controller mode. Set this bit to 0 in device controller mode.

## 26.2.16 INTSTS0 : Interrupt Status Register 0

Base address: USBFS = 0x4009_0000

Offset address: 0x040

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | VBINT | RESM | SOFR | DVST | CTRT | BEMP | NRDY | BRDY | VBSTS | DVSQ[2:0] | | | VALID | CTSQ[2:0] | | |
| Value after reset: | 0 | 0 | 0 | x | 0 | 0 | 0 | 0 | x | 0 | 0 | x | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | CTSQ[2:0] | Control Transfer Stage<br>0 0 0: Idle or setup stage<br>0 0 1: Control read data stage<br>0 1 0: Control read status stage<br>0 1 1: Control write data stage<br>1 0 0: Control write status stage<br>1 0 1: Control write (no data) status stage<br>1 1 0: Control transfer sequence error | R |
| 3 | VALID | USB Request Reception<br>0: Setup packet not received<br>1: Setup packet received | R/W |
| 6:4 | DVSQ[2:0] | Device State<br>Indicates the device state.<br>0 0 0: Powered state<br>0 0 1: Default state<br>0 1 0: Address state<br>0 1 1: Configured state<br>Others: Suspend state | R |
| 7 | VBSTS | VBUS Input Status<br>0: USB_VBUS pin is low<br>1: USB_VBUS pin is high | R |
| 8 | BRDY | Buffer Ready Interrupt Status<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R |
| 9 | NRDY | Buffer Not Ready Interrupt Status<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 10 | BEMP | Buffer Empty Interrupt Status<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R |
| 11 | CTRT | Control Transfer Stage Transition Interrupt Status*2<br>0: No control transfer stage transition interrupt occurred<br>1: Control transfer stage transition interrupt occurred | R/W*1 |
| 12 | DVST | Device State Transition Interrupt Status*2<br>0: No device state transition interrupt occurred<br>1: Device state transition interrupt occurred | R/W*1 |
| 13 | SOFR | Frame Number Refresh Interrupt Status<br>0: No SOF interrupt occurred<br>1: SOF interrupt occurred | R/W*1 |
| 14 | RESM | Resume Interrupt Status*2 *3<br>0: No resume interrupt occurred<br>1: Resume interrupt occurred | R/W*1 |
| 15 | VBINT | VBUS Interrupt Status*3<br>0: No VBUS interrupt occurred<br>1: VBUS interrupt occurred | R/W*1 |

Note: The value of the DVST bit is 0 when the MCU is reset and 1 after a USB bus reset.
Note: The value of the VBSTS bit is 1 when the USB_VBUS pin is high and 0 when the USB_VBUS pin is low.
Note: The value of the DVSQ[2:0] bits is 000b when the MCU is reset and 001b after a USB bus reset.
Note 1. To clear the VBINT, RESM, SOFR, DVST, CTRT, or VALID bits, write 0 only to the bits to be cleared. Write 1 to the other bits. Do not write 0 to the status bits indicating 0.
Note 2. The status of the RESM, DVST, and CTRT bits are changed only in device controller mode. Set the associated interrupt enable bits to 0 (disabled) in host controller mode.
Note 3. The USBFS detects a change in the status indicated in the VBINT andRESM bits even while the clock supply is stopped (SYSCFG.SCKE bit = 0), and it requests the interrupt when the associated interrupt request bit is 1. Enable the clock supply before clearing the status by software.

### CTSQ[2:0] bits (Control Transfer Stage)

In host controller mode, the read value of the CTSQ[2:0] bits is invalid.

### VALID bit (USB Request Reception)

In host controller mode, the read value of the VALID bit is invalid.

### DVSQ[2:0] bits (Device State)

The DVSQ[2:0] bits are initialized by a USB bus reset. In host controller mode, the read value is invalid.

### BRDY flag (Buffer Ready Interrupt Status)

The BRDY flag indicates the BRDY interrupt status.

The USBFS sets the BRDY bit to 1 when it detects a BRDY interrupt status (PIPEnBRDY = 1, n = 0 to 9) on at least one pipe for which BRDY interrupts are enabled (BRDYENB.PIPEnBRDYE = 1).

For the conditions that cause the PIPEnBRDY status to be asserted, see section 26.3.3.1. BRDY interrupt.

The USBFS sets the BRDY bit to 0 when the software writes 0 to all of the PIPEnBRDY bits associated with the PIPEnBRDYE bits that are set to 1. Writing 0 to the BRDY flag in the software does not clear the flag.

### NRDY flag (Buffer Not Ready Interrupt Status)

The NRDY flag indicates the NRDY interrupt status.

The USBFS sets the NRDY bit to 1 when it detects a NRDY interrupt status (PIPEnNRDY = 1, n = 0 to 9) on at least one pipe for which NRDY interrupts are enabled (NRDYENB.PIPEnNRDYE = 1).

For the conditions that cause the PIPEnNRDY status to be asserted, see section 26.3.3.2. NRDY interrupt.

The USBFS sets the NRDY bit to 0 when the software writes 0 to all of the PIPEnNRDY bits associated with the PIPEnNRDYE bits that are set to 1. Writing 0 to the NRDY flag in the software does not clear the flag.

### BEMP flag (Buffer Empty Interrupt Status)

The BEMP flag indicates the BEMP interrupt status.

The USBFS sets the BEMP bit to 1 when it detects a BEMP interrupt status (PIPEnBEMP = 1, n = 0 to 9) on at least one pipe for which BEMP interrupts are enabled (BEMPENB.PIPEnBEMPE = 1).

For the conditions that cause the PIPEnBEMP status to be asserted, see section 26.3.3.3. BEMP interrupt.

The USBFS sets the BEMP bit to 0 when the software writes 0 to all of the PIPEnBEMP bits associated with the PIPEnBEMPE bits that are set to 1. Writing 0 to the BEMP flag in the software does not clear the flag.

### CTRT flag (Control Transfer Stage Transition Interrupt Status)

In device controller mode, the USBFS updates the value of the CTSQ[2:0] bits and sets the CTRT flag to 1 on detecting a transition in the control transfer stage. When a control transfer stage transition interrupt occurs, clear the CTRT flag before the USBFS detects the next control transfer stage transition.

Values read from the CTRT flag in host controller mode are invalid.

### DVST flag (Device State Transition Interrupt Status)

In device controller mode, the USBFS updates the value of the DVSQ[2:0] bits and sets the DVST flag to 1 on detecting a change in the device state. When a device state transition interrupt occurs, clear theDVST flag before the USBFS detects the next device state transition.

Values read from the DVST flag in host controller mode are invalid.

### SOFR flag (Frame Number Refresh Interrupt Status)

In host controller mode, the USBFS sets the SOFR flag to 1 on updating the frame number when the DVSTCTR0.UACT bit is set to 1 by software. A SOFR interrupt is detected every 1 ms.

In device controller mode, the USBFS sets the SOFR flag to 1 on updating the frame number. A frame number refresh interrupt is detected every 1 ms.

The USBFS can detect an SOFR interrupt through the internal interpolation function even when a corrupted SOF packet is received from the USB host.

### RESM flag (Resume Interrupt Status)

In device controller mode, the USBFS sets the RESM flag to 1 on detecting the falling edge of the signal on the USB_DP pin in the Suspend state (DVSQ[2:0] = 1xxb). Values read from the RESM flag in host controller mode are invalid.

### VBINT flag (VBUS Interrupt Status)

The USBFS sets the VBINT flag to 1 on detecting a level change (high to low or low to high) in the USB_VBUS pin input value. The USBFS sets the VBSTS flag to indicate the USB_VBUS pin input value. When a VBUS interrupt occurs, eliminate transient elements by reading the VBSTS flag at least three times through software processing and check that the values read are the same.

## 26.2.17 INTSTS1 : Interrupt Status Register 1

Base address: USBFS = 0x4009_0000

Offset address: 0x042

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | OVRCR | BCHG | — | DTCH | ATTCH | — | — | — | — | EOFERR | SIGN | SACK | — | — | — | PDDETINT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PDDETINT*1 | PDDET Detection Interrupt Status Flag<br>0: No PDDET interrupt occurred<br>1: PDDET interrupt occurred | R/W*2 |
| 3:1 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4 | SACK | Setup Transaction Normal Response Interrupt Status<br>0: No SACK interrupt occurred<br>1: SACK interrupt occurred | R/W*3 |
| 5 | SIGN | Setup Transaction Error Interrupt Status<br>0: No SIGN interrupt occurred<br>1: SIGN interrupt occurred | R/W*3 |
| 6 | EOFERR | EOF Error Detection Interrupt Status<br>0: No EOFERR interrupt occurred<br>1: EOFERR interrupt occurred | R/W*3 |
| 10:7 | — | These bits are read as 0. The write value should be 0. | R/W |
| 11 | ATTCH | ATTCH Interrupt Status<br>0: No ATTCH interrupt occurred<br>1: ATTCH interrupt occurred | R/W*3 |
| 12 | DTCH | USB Disconnection Detection Interrupt Status<br>0: No DTCH interrupt occurred<br>1: DTCH interrupt occurred | R/W*3 |
| 13 | — | This bit is read as 0. The write value should be 0. | R/W |
| 14 | BCHG | USB Bus Change Interrupt Status*4<br>0: No BCHG interrupt occurred<br>1: BCHG interrupt occurred | R/W*3 |
| 15 | OVRCR | Overcurrent Input Change Interrupt Status*4<br>0: No OVRCR interrupt occurred<br>1: OVRCR interrupt occurred | R/W*3 |

Note 1. The USBFS detects a change in the status in the PDDETINT, OVRCR or BCHG bit even when the clock supply is stopped (SYSCFG.SCKE = 0), and it requests the interrupt when the associated interrupt request bit is 1. Enable the clock supply (SYSCFG.SCKE = 1) before clearing the status through software. No other interrupts can be detected while the clock supply is stopped (SYSCFG.SCKE bit = 0).
Note 2. To clear the bits in the INTSTS1 register, write 0 only to the bits to be cleared. Write 1 to the other bits.
Note 3. To clear the bits in INTSTS1, write 0 only to the bits to be cleared. Write 1 to the other bits.
Note 4. The USBFS detects a change in the status in the OVRCR or BCHG bit even when the clock supply is stopped (SYSCFG.SCKE = 0), and it requests the interrupt when the associated interrupt request bit is 1. Enable the clock supply (SYSCFG.SCKE = 1) before clearing the status through the software. No other interrupts can be detected while the clock supply is stopped (SYSCFG.SCKE bit = 0).

INTSTS1 is used to confirm the status of each interrupt in host controller mode. Only enable the status change interrupts indicated in the bits in INTSTS1 in host controller mode.

### PDDETINT flag (PDDET Detection Interrupt Status Flag)

The USBFS sets the PDDETINT flag to 1 on detecting a level change (high to low or low to high) in the PDDET pin input value. When the PDDETINT interrupt is generated, perform debouncing by reading the BCCTRL1.PDDETSTS flag at least three times through software processing and checking that the values read are the same.

### SACK flag (Setup Transaction Normal Response Interrupt Status)

The SACK flag indicates the status of the setup transaction normal response interrupt in host controller mode.

The USBFS detects the SACK interrupt and sets this flag to 1 when an ACK response is returned from a peripheral device during the setup transactions issued by the USBFS. If the associated interrupt enable bit is set to 1 by software, the USBFS generates the interrupt.

Values read from the SACK flag in device controller mode are invalid.

### SIGN flag (Setup Transaction Error Interrupt Status)

The SIGN flag indicates the status of setup transaction error interrupts in host controller mode.

The USBFS detects the SIGN interrupt and sets this flag to 1 when an ACK response is not returned from a peripheral device three consecutive times during the setup transactions issued by the USBFS. If the associated interrupt enable bit is set to 1 by software, the USBFS generates the interrupt.

The USBFS detects the SIGN interrupt when any of the following response conditions occur for three consecutive setup transactions:

- Timeout is detected by the USBFS when the peripheral device has returned no response
- A corrupted ACK packet is received
- A handshake other than ACK (NAK, NYET, or STALL) is received

Values read from the SIGN flag in device controller mode are invalid.

#### EOFERR flag (EOF Error Detection Interrupt Status)

The EOFERR flag indicates the status of EOF error detection interrupts in host controller mode.

The USBFS detects the EOFERR interrupt and sets this flag to 1 on detecting that communication did not complete at the EOF2 timing defined in the USB 2.0 specification. If the associated interrupt enable bit is set to 1 by software, the USBFS generates the interrupt.

After detecting the EOFERR interrupt, the USBFS controls the hardware as follows, regardless of the associated interrupt enable bit setting:

- Sets the DVSTCTR0.UACT bit for the port in which the EOFERR interrupt was detected to 0
- Puts the port in which the EOFERR interrupt occurred into the idle state

The software must terminate all pipes in which communications are being carried out and re-enumerate the USB port.

Values read from the EOFERR flag in device controller mode are invalid.

#### ATTCH flag (ATTCH Interrupt Status)

The ATTCH flag indicates the status of USB attach detection interrupts in host controller mode.

The USBFS detects the ATTCH interrupt and sets this flag to 1 on detecting a J- or K-state on the full- or low-speed signal level for 2.5 µs. If the associated interrupt enable bit is set to 1 by software, the USBFS generates the interrupt.

The USBFS detects the ATTCH interrupt on any of the following conditions.

- K-state, SE0, or SE1 changes to J-state, and J-state continues for 2.5 µs
- J-state, SE0, or SE1 changes to K-state, and K-state continues for 2.5 µs

Values read from the ATTCH flag in device controller mode are invalid.

#### DTCH flag (USB Disconnection Detection Interrupt Status)

The DTCH flag indicates the status of USB disconnection detection interrupts in host controller mode.

The USBFS detects the DTCH interrupt and sets this flag to 1 on detecting a USB bus detach event. If the associated interrupt enable bit is set to 1 by software, the USBFS generates the interrupt.

The USBFS detects bus detach events based on the USB 2.0 specification.

After detecting the DTCH interrupt, the USBFS controls hardware as follows, regardless of the associated interrupt enable bit setting:

- Sets the DVSTCTR0.UACT bit for the port in which the DTCH interrupt was detected to 0
- Puts the port in which the DTCH interrupt occurred into the idle state

The software must terminate all pipes in which communications are being carried out and transition to a wait state for connecting to the USB port (waiting for ATTCH interrupt generation).

Values read from the DTCH flag in device controller mode are invalid.

#### BCHG flag (USB Bus Change Interrupt Status)

The BCHG flag indicates the status of USB bus change interrupts in host controller mode.

The USBFS detects the BCHG interrupt and sets this flag to 1 when a change in the full- or low-speed signal level occurs on the USB port. This includes any change from J-state, K-state, or SE0 to J-state, K-state, or SE0. If the associated interrupt enable bit is set to 1 by software, the USBFS generates the interrupt.

The USBFS sets the LNST[1:0] flags to indicate the input state of the USB port. When a BCHG interrupt occurs, eliminate transient elements by repeat reading the LNST[1:0] flags by software until the same value is read at least three times.

Change in the USB bus state can be detected while the internal clock is stopped.

Values read from the BCHG flag in device controller mode are invalid.

### OVRCR flag (Overcurrent Input Change Interrupt Status)

The OVRCR flag indicates the status of USB_OVRCURA and USB_OVRCURB input pin change interrupts.

The USBFS detects the OVRCR interrupt and sets this flag to 1 when a change (high to low or low to high) occurs in at least one of the input values to the USB_OVRCURA and USB_OVRCURB pins. If the associated interrupt enable bit is set to 1 by software, the USBFS generates the interrupt.

## 26.2.18 BRDYSTS : BRDY Interrupt Status Register

Base address: USBFS = 0x4009_0000

Offset address: 0x046

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | PIPE9 BRDY | PIPE8 BRDY | PIPE7 BRDY | PIPE6 BRDY | PIPE5 BRDY | PIPE4 BRDY | PIPE3 BRDY | PIPE2 BRDY | PIPE1 BRDY | PIPE0 BRDY |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PIPE0BRDY | BRDY Interrupt Status for Pipe 0 *2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 1 | PIPE1BRDY | BRDY Interrupt Status for Pipe 1 *2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 2 | PIPE2BRDY | BRDY Interrupt Status for Pipe 2*2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 3 | PIPE3BRDY | BRDY Interrupt Status for Pipe 3*2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 4 | PIPE4BRDY | BRDY Interrupt Status for Pipe 4 *2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 5 | PIPE5BRDY | BRDY Interrupt Status for Pipe 5 *2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 6 | PIPE6BRDY | BRDY Interrupt Status for Pipe 6*2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 7 | PIPE7BRDY | BRDY Interrupt Status for Pipe 7 *2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 8 | PIPE8BRDY | BRDY Interrupt Status for Pipe 8 *2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 9 | PIPE9BRDY | BRDY Interrupt Status for Pipe 9 *2<br>0: No BRDY interrupt occurred<br>1: BRDY interrupt occurred | R/W*1 |
| 15:10 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. When the SOFCFG.BRDYM bit is set to 0, to clear the status indicated in the bits in BRDYSTS, write 0 only to the bits to be cleared. Write 1 to the other bits.

Note 2. When the SOFCFG.BRDYM bit is set to 0, clear BRDY interrupts before accessing the FIFO.

## 26.2.19 NRDYSTS : NRDY Interrupt Status Register

Base address: USBFS = 0x4009_0000

Offset address: 0x048

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | PIPE9 NRDY | PIPE8 NRDY | PIPE7 NRDY | PIPE6 NRDY | PIPE5 NRDY | PIPE4 NRDY | PIPE3 NRDY | PIPE2 NRDY | PIPE1 NRDY | PIPE0 NRDY |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PIPE0NRDY | NRDY Interrupt Status for Pipe 0<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 1 | PIPE1NRDY | NRDY Interrupt Status for Pipe 1<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 2 | PIPE2NRDY | NRDY Interrupt Status for Pipe 2<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 3 | PIPE3NRDY | NRDY Interrupt Status for Pipe 3<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 4 | PIPE4NRDY | NRDY Interrupt Status for Pipe 4<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 5 | PIPE5NRDY | NRDY Interrupt Status for Pipe 5<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 6 | PIPE6NRDY | NRDY Interrupt Status for Pipe 6<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 7 | PIPE7NRDY | NRDY Interrupt Status for Pipe 7<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 8 | PIPE8NRDY | NRDY Interrupt Status for Pipe 8<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 9 | PIPE9NRDY | NRDY Interrupt Status for Pipe 9<br>0: No NRDY interrupt occurred<br>1: NRDY interrupt occurred | R/W*1 |
| 15:10 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. To clear the status indicated in the bits in NRDYSTS, write 0 only to the bits to be cleared. Write 1 to the other bits.

## 26.2.20 BEMPSTS : BEMP Interrupt Status Register

Base address: USBFS = 0x4009_0000

Offset address: 0x04A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | PIPE9 BEMP | PIPE8 BEMP | PIPE7 BEMP | PIPE6 BEMP | PIPE5 BEMP | PIPE4 BEMP | PIPE3 BEMP | PIPE2 BEMP | PIPE1 BEMP | PIPE0 BEMP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PIPE0BEMP | BEMP Interrupt Status for Pipe 0<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 1 | PIPE1BEMP | BEMP Interrupt Status for Pipe 1<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 2 | PIPE2BEMP | BEMP Interrupt Status for Pipe 2<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 3 | PIPE3BEMP | BEMP Interrupt Status for Pipe 3<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 4 | PIPE4BEMP | BEMP Interrupt Status for Pipe 4<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 5 | PIPE5BEMP | BEMP Interrupt Status for Pipe 5<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 6 | PIPE6BEMP | BEMP Interrupt Status for Pipe 6<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 7 | PIPE7BEMP | BEMP Interrupt Status for Pipe 7<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 8 | PIPE8BEMP | BEMP Interrupt Status for Pipe 8<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 9 | PIPE9BEMP | BEMP Interrupt Status for Pipe 9<br>0: No BEMP interrupt occurred<br>1: BEMP interrupt occurred | R/W*1 |
| 15:10 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. To clear the status indicated in the bits in BEMPSTS, write 0 only to the bits to be cleared. Write 1 to the other bits.

## 26.2.21 FRMNUM : Frame Number Register

Base address: USBFS = 0x4009_0000

Offset address: 0x04C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | OVRN | CRCE | — | — | — | FRNM[10:0] | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 10:0 | FRNM[10:0] | Frame Number<br>Latest frame number. | R |
| 13:11 | — | These bits are read as 0. The write value should be 0. | R/W |
| 14 | CRCE | Receive Data Error<br>0: No error occurred<br>1: Error occurred | R/W*1 |
| 15 | OVRN | Overrun/Underrun Detection Status<br>0: No error occurred<br>1: Error occurred | R/W*1 |

Note 1. To clear the status, write 0 only to the bits to be cleared. Write 1 to the other bits.

### FRNM[10:0] flags (Frame Number)

The USBFS sets the FRNM[10:0] flags to indicate the latest frame number, which is updated every 1 ms, when an SOF packet is issued or received.

### CRCE flag (Receive Data Error)

The CRCE flag sets to 1 when a CRC error or bit stuffing error occurs during isochronous transfer. On detecting a CRC error in host controller mode, the USBFS generates an internal NRDY interrupt.

To clear the CRCE flag, write 0 to it while writing 1 to the other bits in the FRMNUM register.

### OVRN flag (Overrun/Underrun Detection Status)

The OVRN flag sets to 1 when an overrun or underrun error occurs during isochronous transfer. To clear the flag, write 0 to it while writing 1 to the other bits in the FRMNUM register.

In host controller mode, the OVRN flag sets to 1 on any of the following conditions:

- For a transmitting isochronous pipe, the time to issue an OUT token comes before all of the transmit data is written to the FIFO buffer
- For a receiving isochronous pipe, the time to issue an IN token comes when no FIFO buffer planes are empty

In device controller mode, the OVRN flag sets to 1 on any of the following conditions:

- For a transmitting isochronous pipe, the IN token is received before all of the transmit data is written to the FIFO buffer
- For a receiving isochronous pipe, the OUT token is received when no FIFO buffer planes are empty

## 26.2.22 DVCHGR : Device State Change Register

Base address: USBFS = 0x4009_0000

Offset address: 0x04E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DVCHG | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15 | DVCHG | Device State Change<br>0: Disable writes to the USBADDR.STSRECOV[3:0] and USBADDR.USBADDR[6:0] bits<br>1: Enable writes to the USBADDR.STSRECOV[3:0] and USBADDR.USBADDR[6:0] bits | R/W |

For details, see section 26.3.1.5. Release from Deep Software Standby mode because of USB suspend/resume interrupts.

## 26.2.23 USBADDR : USB Address Register

Base address: USBFS = 0x4009_0000

Offset address: 0x050

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | STSRECOV[3:0] | | | | — | USBADDR[6:0] | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:0 | USBADDR[6:0] | USB Address<br>In device controller mode, these bits indicate the USB address assigned by the host when the USBFS processed the SET_ADDRESS request successfully. | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |
| 11:8 | STSRECOV[3:0] | Status Recovery<br>0x4: Recovery in device controller mode: Setting prohibited<br>Recovery in host controller mode:<br>Return to the low-speed state (bits DVSTCTR0.RHST[2:0] = 001b)<br>0x8: Recovery in device controller mode: Setting prohibited<br>Recovery in host controller mode:<br>Return to the full-speed state (bits DVSTCTR0.RHST[2:0] = 010b)<br>0x9: Recovery in device controller mode:<br>Return to the full-speed state (bits DVSTCTR0.RHST[2:0] = 010b), bits INTSTS0.DVSQ[2:0] = 001b (default state)<br>Recovery in host controller mode: Setting prohibited<br>0xA: Recovery in device controller mode:<br>Return to the full-speed state (bits DVSTCTR0.RHST[2:0] = 010b), bits INTSTS0.DVSQ[2:0] = 010b (address state)<br>Recovery in host controller mode: Setting prohibited<br>0xB: Recovery in device controller mode:<br>Return to the full-speed state (bits DVSTCTR0.RHST[2:0] = 010b), bits INTSTS0.DVSQ[2:0] = 011b (configured state)<br>Recovery in host controller mode: Setting prohibited<br>Others: Setting prohibited | R/W |
| 15:12 | — | These bits are read as 0. The write value should be 0. | R/W |

### USBADDR[6:0] bits (USB Address)

In device controller mode, the USBADDR[6:0] flags indicate the USB address received when the USBFS processed a SetAddress request successfully. The USBFS sets the USBADDR[6:0] bits to 0x00 on detecting a USB bus reset.

Writing to these bits is enabled while the DVCHGR.DVCHG bit is set to 1. On recovering from a USB power shut-off, the operation can resume from the USB address set before the software shut-off.

In host controller mode, the USBADDR[6:0] bits are invalid.

### STSRECOV[3:0] bits (Status Recovery)

Use the STSRECOV[3:0] bits to resume the state of the internal sequencer on recovering from USB power shut-off. For details, see section 26.3.1.5. Release from Deep Software Standby mode because of USB suspend/resume interrupts.

Writing to these bits is enabled while the DVCHGR.DVCHG bit is set to 1.

## 26.2.24 USBREQ : USB Request Type Register

Base address: USBFS = 0x4009_0000

Offset address: 0x054

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BREQUEST[7:0] | | | | | | | | BMREQUESTTYPE[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | BMREQUESTTYPE[7:0] | Request Type<br>USB request bmRequestType value | R/W*1 |
| 15:8 | BREQUEST[7:0] | Request<br>USB request bRequest value | R/W*1 |

Note 1. In device controller mode, these bits can be read, but writing to them has no effect. In host controller mode, these bits are both read/write bits.

USBREQ stores setup requests for control transfers.

In device controller mode, the USBREQ stores the received bRequest and bmRequestType values. In host controller mode, it sets to the bRequest and bmRequestType values to be transmitted.

USBREQ is initialized by a USB bus reset.

#### BMREQUESTTYPE[7:0] bits (Request Type)

The BMREQUESTTYPE[7:0] bits hold the bmRequestType value of USB requests.

- In host controller mode:

  Set these bits to the value of the USB request data in transmission setup transactions. Do not change the value of the bits while the DCPCTR.SUREQ bit is 1.

- In device controller mode:

  These bits indicate the value of the USB request data in reception setup transactions. Writing to the bits has no effect.

#### BREQUEST[7:0] bits (Request)

The BREQUEST[7:0] bits store bRequest value of the USB request.

- In host controller mode:

  Set these bits to the value of the USB request data in setup transmission transactions. Do not change the value of the bits while the DCPCTR.SUREQ bit is 1.

- In device controller mode:

  These bits indicate the value of the USB request data in reception setup transactions. Writing to the bits has no effect.

### 26.2.25 USBVAL : USB Request Value Register

Base address: USBFS = 0x4009_0000

Offset address: 0x056

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | WVALUE[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | WVALUE[15:0] | Value<br>USB request wValue value | R/W*1 |

Note 1. In device controller mode, these bits can be read, but writing to them has no effect. In host controller mode, these bits are both read/write bits.

In device controller mode, USBVAL stores the received wValue value. In host controller mode, it sets to the wValue value to be transmitted is set.

USBVAL is initialized by a USB bus reset.

#### WVALUE[15:0] bits (Value)

The WVALUE[15:0] bits store wValue value of the USB request.

- In host controller mode:

  Set these bits to the value of the wValue field in USB requests of transmission setup transactions. Do not change the value of the bits while the DCPCTR.SUREQ bit is 1.

- In device controller mode:

  These bits indicate the wValue value of USB requests in reception setup transactions. Writing to the bits has no effect.

## 26.2.26 USBINDX : USB Request Index Register

Base address: USBFS = 0x4009_0000

Offset address: 0x058

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | WINDEX[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | WINDEX[15:0] | Index<br>USB request wIndex value | R/W*1 |

Note 1. In device controller mode, these bits can be read, but writing to them has no effect. In host controller mode, these bits are both read/write bits.

USBINDX stores setup requests for control transfers.

In device controller mode, it stores the received wIndex value. In host controller mode, it sets to the wIndex value to be transmitted.

USBINDX is initialized by a USB bus reset.

### WINDEX[15:0] bits (Index)

The WINDEX[15:0] bits hold the wIndex value of a USB request.

- In host controller mode:

  Set these bits to the wIndex value in USB requests in transmission setup transactions. Do not change the value of the bits while the DCPCTR.SUREQ bit is 1.

- In device controller mode:

  These bits indicate the wIndex value in USB requests received in reception setup transactions. Writing to the bits has no effect.

## 26.2.27 USBLENG : USB Request Length Register

Base address: USBFS = 0x4009_0000

Offset address: 0x05A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | WLENTUH[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | WLENTUH[15:0] | Length<br>USB request wLength value | R/W*1 |

Note 1. In device controller mode, these bits can be read, but writing to them has no effect. In host controller mode, these bits are both read/write bits.

USBLENG stores setup requests for control transfers.

In device controller mode, the value of wLength that is received is stored. In host controller mode, the value of wLength to be transmitted is set.

USBLENG is initialized by a USB bus reset.

### WLENTUH[15:0] bits (Length)

The WLENTUH[15:0]bits hold the wLength value of a USB request.

- In host controller mode:

Set these bits to the wLength value in USB requests in transmission setup transactions. Do not change the value of the bits while the DCPCTR.SUREQ bit is 1.

- In device controller mode:

  These bits indicate the wLength value in USB requests received in reception setup transactions. Writing to the bits has no effect.

## 26.2.28 DCPCFG : DCP Configuration Register

Base address: USBFS = 0x4009_0000

Offset address: 0x05C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | SHTNAK | — | — | DIR | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | DIR | Transfer Direction*1<br>0: Data receiving direction<br>1: Data transmitting direction | R/W |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | SHTNAK | Pipe Disabled at End of Transfer*1<br>0: Keep pipe open after transfer ends<br>1: Disable pipe after transfer ends | R/W |
| 15:8 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Only set this bit while the PID is NAK. Before setting this bit, check that the DCPCTR.PBUSY bit is 0, and then change the DCPCTR.PID[1:0] bits for the DCP from BUF to NAK. If the PID[1:0] bits are changed to NAK by the USBFS, checking the PBUSY bit through the software is not necessary.

### DIR bit (Transfer Direction)

In host controller mode, the DIR bit sets the transfer direction of the data stage and status stage for control transfers. In device controller mode, set the DIR bit to 0.

### SHTNAK bit (Pipe Disabled at End of Transfer)

The SHTNAK bit specifies whether to change PID to NAK on transfer end when the selected pipe is receiving. It is only valid when the selected pipe is receiving.

When the SHTNAK bit is 1, the USBFS changes the DCPCTR.PID[1:0] bits for the DCP to NAK on determining that a transfer has ended. The USBFS determines transfer end on the following condition:

- A short packet, including a zero-length packet, is successfully received.

## 26.2.29 DCPMAXP : DCP Maximum Packet Size Register

Base address: USBFS = 0x4009_0000

Offset address: 0x05E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DEVSEL[3:0] | | | | — | — | — | — | — | MXPS[6:0] | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:0 | MXPS[6:0] | Maximum Packet Size*1<br>Maximum data payload specification (maximum packet size) for the DCP | R/W |
| 11:7 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:12 | DEVSEL[3:0] | Device Select*2<br>0x0: Address 0000b<br>0x1: Address 0001b<br>0x2: Address 0010b<br>0x3: Address 0011b<br>0x4: Address 0100b<br>0x5: Address 0101b<br>Others: Setting prohibited | R/W |

Note 1. Only set the MXPS[6:0] bits while PID is NAK. Before setting these bits, check that the DCPCTR.PBUSY bit is 0, and then change the DCPCTR.PID[1:0] bits for the DCP from BUF to NAK. If the PID[1:0] bits are changed to NAK by the USBFS, checking the PBUSY bit through the software is not necessary. After the MXPS[6:0] bits are set and the DCP is set to the CURPIPE[3:0] bits in a port select register, clear the buffer by setting the BCLR bit in the port control register to 1.

Note 2. Only set the DEVSEL[3:0] bits while PID is NAK and the DCPCTR.SUREQ bit is 0. Before setting these bits, check that the DCPCTR.PBUSY bit is 0, and then change the DCPCTR.PID[1:0] bits for the DCP from BUF to NAK. If the PID[1:0] bits are changed to NAK by the USBFS, checking the PBUSY bit through the software is not necessary.

### MXPS[6:0] bits (Maximum Packet Size)

The MXPS[6:0] bits specify the maximum data payload (maximum packet size) for the DCP. The initial value is 0x40 (64 bytes). Set the bits to a USB 2.0-compliant value. Do not write to the FIFO buffer or set PID = BUF whileMXPS[6:0] is set to 0.

### DEVSEL[3:0] bits (Device Select)

In host controller mode, the DEVSEL[3:0] bits specify the address of the target peripheral device for a control transfer. Set up the associated DEVADDn (n = 0 to 5) register first, and then set these bits to the corresponding value. To set the DEVSEL[3:0] bits to 0010b, for example, first set the address in the DEVADD2 register.

In device controller mode, set these bits to 0000b.

## 26.2.30 DCPCTR : DCP Control Register

Base address: USBFS = 0x4009_0000

Offset address: 0x060

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BSTS | SUREQ | — | — | SUREQCLR | — | — | SQCLR | SQSET | SQMON | PBUSY | — | — | CCPL | PID[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | PID[1:0] | Response PID<br>0 0: NAK response<br>0 1: BUF response (depends on the buffer state)<br>1 0: STALL response<br>1 1: STALL response | R/W |
| 2 | CCPL | Control Transfer End Enable<br>0: Disable control transfer completion<br>1: Enable control transfer completion | R/W |
| 4:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5 | PBUSY | Pipe Busy<br>0: DCP not used for the USB bus<br>1: DCP in use for the USB bus | R |
| 6 | SQMON | Sequence Toggle Bit Monitor<br>0: DATA0<br>1: ATA1 | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | SQSET | Sequence Toggle Bit Set[*2]<br>Sets the sequence toggle bit in DCP transfers.<br>This bit is read as 0.<br>0: Invalid (writing 0 has no effect)<br>1: Set the expected value for the next transaction to DATA1 | R/W[*1] |
| 8 | SQCLR | Sequence Toggle Bit Clear[*2]<br>Clears the sequence toggle bit in DCP transfers.<br>This bit is read as 0.<br>0: Invalid (writing 0 has no effect)<br>1: Clear the expected value for the next transaction to DATA0 | R/W[*1] |
| 10:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 11 | SUREQCLR | SUREQ Bit Clear<br>Clears the SUREQ bit in host controller mode.<br>This bit is read as 0.<br>0: Invalid (writing 0 has no effect)<br>1: Clear SUREQ to 0 | R/W |
| 13:12 | — | These bits are read as 0. The write value should be 0. | R/W |
| 14 | SUREQ | Setup Token Transmission<br>Sets up token transmission in host controller mode.<br>0: Invalid (writing 0 has no effect)<br>1: Transmit setup packet | R/W |
| 15 | BSTS | Buffer Status<br>0: Buffer access disabled<br>1: Buffer access enabled | R |

Note 1. This bit is read as 0.
Note 2. Only set the SQSET and SQCLR bits while PID is NAK. Before setting these bits, check that the PBUSY bit is 0, and then change the PID[1:0] bits for the DCP from BUF to NAK. If the PID[1:0] bits are changed to NAK by the USBFS, checking the PBUSY bit through the software is not necessary.

### PID[1:0] bits (Response PID)

The PID[1:0] bits control the USB response type during control transfers.

In host controller mode, to change the PID[1:0] setting from NAK to BUF:

- When the transmitting direction is set:
  a. Write all of the transmit data to the FIFO buffer while the DVSTCTR0.UACT bit is 1 and PID is NAK.
  b. Set PID[1:0] bits to 01b (BUF).
     The USBFS then executes the OUT transaction.
- When the receiving direction is set:
  a. Check that the FIFO buffer is empty (or empty the buffer) while the DVSTCTR0.UACT bit is 1 and PID is NAK.
  b. Set PID[1:0] bits to 01b (BUF).
     The USBFS then executes the IN transaction.

The USBFS changes the PID[1:0] setting as follows:

- When the PID[1:0] bits are set to BUF (01b) by software and the USBFS has received data exceeding MaxPacketSize, the USBFS sets the PID[1:0] to STALL (11b)
- When a reception error, such as a CRC error, is detected three times consecutively, the USBFS sets thePID[1:0] bits to NAK (00b)
- On receiving the STALL handshake, the USBFS sets PID[1:0] to STALL (11b)

In device controller mode, the USBFS changes the PID[1:0] setting as follows:

- On receiving a setup packet, the USBFS sets PID[1:0] to NAK (00b). The USBFS then sets the INTSTS0.VALID flag to 1, and the PID[1:0] setting cannot be changed until the software clears the VALID flag to 0.

- When the PID[1:0] bits are set to BUF (01b) by software and the USBFS has received data exceeding MaxPacketSize, the USBFS sets PID[1:0] to STALL (11b)
- On detecting a control transfer sequence error, the USBFS sets PID[1:0] to STALL (1xb)
- On detecting a USB bus reset, the USBFS sets PID[1:0] to NAK

The USBFS does not check the PID[1:0] setting while processing a SET_ADDRESS request.

The PID[1:0] bits are initialized by a USB bus reset.

#### CCPL bit (Control Transfer End Enable)

In device controller mode, setting the CCPL bit to 1 enables the status stage of the control transfer to be completed. When the bit is set to 1 by software while the associated PID[1:0] bits are set to BUF, the USBFS completes the control transfer status stage.

During control read transfers, the USBFS transmits the ACK handshake in response to the OUT transaction from the USB host. During control write or no-data control transfers, it transmits the zero-length packet in response to the IN transaction from the USB host. On detecting a SET_ADDRESS request, the USBFS operates in auto response mode from the setup stage up to status stage completion regardless of the CCPL bit setting.

The USBFS changes the CCPL bit from 1 to 0 on receiving a new setup packet. The software cannot write 1 to the bit while the INTSTS0.VALID bit is 1. The bit is initialized by a USB bus reset.

In host controller mode, always write 0 to the CCPL bit.

#### PBUSY bit (Pipe Busy)

The PBUSY bit indicates whether DCP is used for the transaction when USBFS changes the PID[1:0] bits from BUF to NAK. The USBFS changes the PBUSY bit from 0 to 1 on start of a USB transaction for the selected pipe. It changes the PBUSY bit from 1 to 0 on completion of one transaction.

After PID is set to NAK by software, the value in the PBUSY bit indicates whether changes to pipe settings can proceed.

For details, see section 26.3.4.1. Pipe control register switching procedures.

#### SQMON bit (Sequence Toggle Bit Monitor)

The SQMON bit indicates the expected value of the sequence toggle bit for the next transaction during a DCP transfer.

The USBFS toggles the bit on normal completion of the transaction. It does not toggle the bit, however, when a DATAPID mismatch occurs during a transfer in the receiving direction.

In device controller mode, the USBFS sets the SQMON bit to 1 (specifies DATA1 as the expected value) on successful reception of the setup packet.

In device controller mode, the USBFS does not reference this bit during IN or OUT transactions at the status stage, and it does not toggle the bit on normal completion.

#### SQSET bit (Sequence Toggle Bit Set)

The SQSET bit specifies DATA1 as the expected value of the sequence toggle bit for the next transaction during a DCP transfer.

Do not set the SQCLR and SQSET bits to 1 simultaneously.

#### SQCLR bit (Sequence Toggle Bit Clear)

The SQCLR bit specifies DATA0 as the expected value of the sequence toggle bit for the next transaction during a DCP transfer. It is read as 0.

Do not set the SQCLR and SQSET bits to 1 simultaneously.

#### SUREQCLR bit (SUREQ Bit Clear)

In host controller mode, setting the SUREQCLR bit to 1 clears the SUREQ bit to 0. The bit is read as 0.

If transfer stops while the SUREQ bit is set to 1 in a setup transaction, set the SUREQCLR bit to 1 by software. This is not necessary at the end of a normal setup transaction, because the USBFS automatically clears the SUREQ bit to 0.

Only control the SUREQ bit through the SUREQCLR bit while the DVSTCTR0.UACT bit is 0. When UACT is 0, communication is halted or no transfer is occurring because a bus disconnection was detected.

In device controller mode, always write 0 to this bit.

### SUREQ bit (Setup Token Transmission)

In host controller mode, setting the SUREQ bit to 1 triggers the USBFS to transmit the setup packet. After completing the setup transaction process, the USBFS generates either the SACK or SIGN interrupt and clears the SUREQ bit to 0. The USBFS also clears the SUREQ bit to 0 when the software sets the SUREQCLR bit to 1.

Before setting the SUREQ bit to 1, set the DCPMAXP.DEVSEL[3:0] bits, USBREQ, USBVAL, USBINDX, and USBLENG appropriately to transmit the target USB request in the setup transaction. Also check that the PID[1:0] bits for the DCP are set to NAK. After setting the SUREQ bit to 1, do not change the DCPMAXP.DEVSEL[3:0] bits, USBREQ, USBVAL, USBINDX, or USBLENG until the setup transaction is complete (SUREQ bit = 1). Write 1 to the SUREQ bit only when transmitting the setup token. Otherwise, write 0.

In device controller mode, always write 0 to this bit.

### BSTS flag (Buffer Status)

The BSTS flag indicates the status of access to the DCP FIFO buffer. The meaning of this flag varies as follows depending on the CFIFOSEL.ISEL setting:

- When ISEL = 0, the bit indicates whether receive data can be read from the buffer
- When ISEL = 1, the bit indicates whether transmit data can be written to the buffer

## 26.2.31 PIPESEL : Pipe Window Select Register

Base address: USBFS = 0x4009_0000

Offset address: 0x064

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | PIPESEL[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | PIPESEL[3:0] | Pipe Window Select<br>0x0: No pipe selected<br>0x1: Pipe 1<br>0x2: Pipe 2<br>0x3: Pipe 3<br>0x4: Pipe 4<br>0x5: Pipe 5<br>0x6: Pipe 6<br>0x7: Pipe 7<br>0x8: Pipe 8<br>0x9: Pipe 9<br>Others: Setting prohibited | R/W |
| 15:4 | — | These bits are read as 0. The write value should be 0. | R/W |

Set pipes 1 to 9 using the PIPESEL, PIPECFG, PIPEMAXP, PIPEPERI, PIPEnCTR, PIPEnTRE, and PIPEnTRN registers (n = 0 to 9).

After selecting the pipe in the PIPESEL register, pipe functions must be set in the associated PIPECFG, PIPEMAXP, and PIPEPERI registers. PIPEnCTR, PIPEnTRE, and PIPEnTRN can be set independently of the pipe selection in this register.

### PIPESEL[3:0] bits (Pipe Window Select)

The PIPESEL[3:0] bits select the pipe number associated with the PIPECFG, PIPEMAXP, and PIPEPERI registers used for data writing and reading. Selecting a pipe number in the PIPESEL[3:0] bits allows writing to and reading from PIPECFG, PIPEMAXP, and PIPEPERI associated with the selected pipe number.

When PIPESEL[3:0] = 0000b, 0 is read from all of the bits in PIPECFG, PIPEMAXP, and PIPEPERI. Writing to these bits has no effect.

## 26.2.32 PIPECFG : Pipe Configuration Register

Base address: USBFS = 0x4009_0000

Offset address: 0x068

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | TYPE[1:0] | | — | — | — | BFRE | DBLB | — | SHTNAK | — | — | DIR | EPNUM[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | EPNUM[3:0] | Endpoint Number*1<br>Specifies the endpoint number for the selected pipe. Setting 0000b indicates that the pipe is not used. | R/W |
| 4 | DIR | Transfer Direction*2 *3<br>0: Receiving direction<br>1: Transmitting direction | R/W |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | SHTNAK | Pipe Disabled at End of Transfer*1<br>0: Continue pipe operation after transfer ends<br>1: Disable pipe after transfer ends | R/W |
| 8 | — | This bit is read as 0. The write value should be 0. | R/W |
| 9 | DBLB | Double Buffer Mode*2 *3<br>0: Single buffer<br>1: Double buffer | R/W |
| 10 | BFRE | BRDY Interrupt Operation Specification*2 *3<br>0: Generate BRDY interrupt on transmitting or receiving data<br>1: Generate BRDY interrupt on completion of reading data | R/W |
| 13:11 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:14 | TYPE[1:0] | Transfer Type*1<br>0 0: Pipe not used<br>0 1: Pipes 1 and 2: Bulk transfer<br>Pipes 3 to 5: Bulk transfer<br>Pipes 6 to 9: Setting prohibited<br>1 0: Pipes 1 and 2: Setting prohibited<br>Pipes 3 to 5: Setting prohibited<br>Pipes 6 to 9: Interrupt transfer<br>1 1: Pipes 1 and 2: Isochronous transfer<br>Pipes 3 to 5: Setting prohibited<br>Pipes 6 to 9: Setting prohibited | R/W |

Note 1. Only set the TYPE[1:0], SHTNAK, and EPNUM[3:0] bits while PID is NAK. Before setting these bits, check that the PIPEnCTR.PBUSY bit is 0, and then change the PIPEnCTR.PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00 (NAK) by the USBFS, checking the PBUSY bit through the software is not necessary.

Note 2. Only set the BFRE, DBLB, and DIR bits while PID is NAK and before the pipe is selected in the CURPIPE[3:0] bits in the port select register. Before setting these bits, check that the PIPEnCTR.PBUSY bit is 0, and then change the PIPEnCTR.PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00 (NAK) by the USBFS, checking the PBUSY bit through the software is not necessary.

Note 3. To change the BFRE, DBLB, or DIR bits after completing USB communication on the selected pipe, in addition to the constraints described in Note 2, write 1 and 0 to the PIPEnCTR.ACLRM bit continuously through the software and clear the FIFO buffer assigned to the pipe.

PIPECFG specifies the transfer type, FIFO buffer access direction, and endpoint numbers for pipes 1 to 9. It also selects single or double buffer mode, and whether to continue or disable pipe operation at the end of transfer.

### EPNUM[3:0] bits (Endpoint Number)

The EPNUM[3:0] bits specify the endpoint number for the selected pipe. Setting 0000b indicates the pipe not used.

Set these bits so that the combination of the DIR and EPNUM[3:0] settings is different from those for other pipes. The EPNUM[3:0] bits can be set to 0000b for all pipes.

### DIR bit (Transfer Direction)

The DIR bit specifies the transfer direction for the selected pipe.

When the software sets this bit to 0, the USBFS uses the selected pipe for receiving. When the software sets this bit to 1, the USBFS uses the selected pipe for transmitting.

### SHTNAK bit (Pipe Disabled at End of Transfer)

The SHTNAK bit specifies whether to change the PIPEnCTR.PID[1:0] bits to 00b (NAK) at the end of transfer when the selected pipe is set in the receiving direction. The bit is valid for pipes 1 to 5 in the receiving direction.

When the software sets this bit to 1 for a receiving pipe, the USBFS changes the associated PIPEnCTR.PID[1:0] bits to 00b (NAK) on determining the transfer end. The USBFS determines that the transfer has ended on the following conditions:

- A short packet data (including a zero-length packet) was successfully received
- The transaction counter is used and the number of packets specified for the transaction counter are successfully received

### DBLB bit (Double Buffer Mode)

The DBLB bit selects either single or double buffer mode for the FIFO buffer used by the selected pipe. The bit is valid for pipes 1 to 5.

### BFRE bit (BRDY Interrupt Operation Specification)

The BFRE bit specifies the BRDY interrupt generation timing from the USBFS to the CPU for the selected pipe.

When the software sets the BFRE bit to 1 and the selected pipe is in the receiving direction, the USBFS detects the transfer completion and generates the BRDY interrupt on reading the packet.

When a BRDY interrupt is generated with this setting, the software must write 1 to the BCLR bit in the port control register. The FIFO buffer assigned to the selected pipe is not enabled for reception until 1 is written to the BCLR bit.

When the BFRE bit is set to 1 by software and the selected pipe is in the transmitting direction, the USBFS does not generate the BRDY interrupt. For details, see section 26.3.3.1. BRDY interrupt.

### TYPE[1:0] bits (Transfer Type)

The TYPE[1:0] bits specify the transfer type for the pipe selected in the PIPESEL.PIPESEL[3:0] bits. Before setting PID to BUF and starting USB communication on the selected pipe, set the TYPE[1:0] bits to a value other than 00b.

## 26.2.33 PIPEMAXP : Pipe Maximum Packet Size Register

Base address: USBFS = 0x4009_0000

Offset address: 0x06C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DEVSEL[3:0] | | | | — | — | — | MXPS[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | MXPS[8:0] | Maximum Packet Size*1<br>• Pipes 1 and 2<br>1 byte (0x001) to 256 bytes (0x100) (Bit [9] not supported.)<br>• Pipes 3 to 5<br>8 bytes (0x008), 16 bytes (0x010), 32 bytes (0x020), 64 bytes (0x040) (Bits [9:7] and [2:0] not supported.)<br>• Pipes 6 to 9<br>1 byte (0x001) to 64 bytes (0x040) (Bits [9:7] not supported.) | R/W |
| 11:9 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:12 | DEVSEL[3:0] | Device Select*2<br>0x0: Address 0000b<br>0x1: Address 0001b<br>0x2: Address 0010b<br>0x3: Address 0011b<br>0x4: Address 0100b<br>0x5: Address 0101b<br>Others: Setting prohibited | R/W |

Note: The value of the MXPS[8:0] bits is 0x000 when no pipe is selected in the PIPESEL.PIPESEL[3:0] bits and 0x040 when a pipe is selected.

Note 1. Only set the MXPS[8:0] bits while PID is NAK and before the pipe is selected in the CURPIPE[3:0] bits in the port select register. Before setting these bits, check that the PIPEnCTR.PBUSY bit is 0, and then change the PIPEnCTR.PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00 (NAK) by the USBFS, checking the PBUSY bit through the software is not necessary.

Note 2. Only set the DEVSEL[3:0] bits while PID is NAK. Before setting these bits, check that the PIPEnCTR.PBUSY bit is 0, and then change the PIPEnCTR.PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00b (NAK) by the USBFS, checking the PBUSY bit through the software is not necessary.

PIPEMAXP specifies the maximum packet size for pipes 1 to 9.

#### MXPS[8:0] bits (Maximum Packet Size)

The MXPS[8:0] bits specify the maximum data payload (maximum packet size) for the selected pipe.

Set these bits to the appropriate value for each transfer type based on the USB 2.0 specification. When MXPS[8:0] = 0, do not write to the FIFO buffer or set PID to BUF. These writes have no effect.

#### DEVSEL[3:0] bits (Device Select)

In host controller mode, the DEVSEL[3:0] bits specify the address of the target device for USB communication. Set up the device address in the associated DEVADDn (n = 0 to 5) register first, and then set these bits to the corresponding value. To set the DEVSEL[3:0] bits to 0x2, for example, first set the address in the DEVADD2 register.

In device controller mode, set these bits to 0x0.

## 26.2.34 PIPEPERI : Pipe Cycle Control Register

Base address: USBFS = 0x4009_0000

Offset address: 0x06E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | IFIS | — | — | — | — | — | — | — | — | — | IITV[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | IITV[2:0]*1 | Interval Error Detection Interval<br>Specifies the interval error detection timing for the selected pipe as the n-th power of 2 of the frame timing | R/W |
| 11:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 12 | IFIS | Isochronous IN Buffer Flush<br>0: Do not flush buffer<br>1: Flush buffer | R/W |
| 15:13 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Only set the IITV[2:0] bits while PID is NAK. Before setting these bits, check that the PBUSY bit is 0, and then change the PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00 (NAK) by the USBFS, checking the PBUSY bit through the software is not necessary.

PIPEPERI selects whether the buffer is flushed or not when an interval error occurred during isochronous IN transfers, and sets the interval error detection interval for pipes 1 to 9.

### IITV[2:0] bits (Interval Error Detection Interval)

To change the IITV[2:0] bits to another value after they are set and USB communication is performed, set the PIPEnCTR.PID[1:0] bits to 00b (NAK) and then set the PIPEnCTR.ACLRM bit to 1 to initialize the interval timer.

The IITV[2:0] bits are not provided for pipes 3 to 5. Write 000b to bit positions of the IITV[2:0] bits associated with pipes 3 to 5.

### IFIS bit (Isochronous IN Buffer Flush)

The IFIS bit specifies whether to flush the buffer when the pipe selected in the PIPESEL.PIPESEL[3:0] bits is used for isochronous IN transfers.

In device controller mode when the selected pipe is for isochronous IN transfers, the USBFS automatically clears the FIFO buffer if the USBFS fails to receive the IN token from the USB host within the interval set in theIITV[2:0] bits in terms of frames.

When double buffering is specified (PIPECFG.DBLB = 1), the USBFS only clears the data in the previously used plane.

The USBFS clears the FIFO buffer on receiving the SOF packet immediately after the frame in which the USBFS expected to receive the IN token. Even if the SOF packet is corrupted, the FIFO buffer is cleared at the time the SOF packet is expected to be received by using the internal interpolation function.

In host controller mode, set the IITV[2:0] bits to 000b.

Set the IITV[2:0] bits to 000b when the selected pipe is not used for isochronous transfers.

## 26.2.35 PIPEnCTR : PIPEn Control Registers (n = 1 to 5)

Base address: USBFS = 0x4009_0000

Offset address: 0x070 + 0x2 × (n - 1)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BSTS | INBUFM | — | — | — | ATREPM | ACLRM | SQCLR | SQSET | SQMON | PBUSY | — | — | — | PID[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | PID[1:0] | Response PID<br>0 0: NAK response<br>0 1: BUF response (depends buffer state)<br>1 0: STALL response<br>1 1: STALL response | R/W |
| 4:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5 | PBUSY | Pipe Busy<br>0: Pipe n not in use for the transaction<br>1: Pipe n in use for the transaction | R |
| 6 | SQMON | Sequence Toggle Bit Confirmation<br>0: DATA0<br>1: DATA1 | R |
| 7 | SQSET | Sequence Toggle Bit Set*2<br>Sets the sequence toggle bit for pipe n.<br>This bit is read as 0.<br>0: Invalid (writing 0 has no effect)<br>1: Set the expected value for the next transaction to DATA1 | R/W*1 |
| 8 | SQCLR | Sequence Toggle Bit Clear*2<br>Clears the sequence toggle bit for pipe n.<br>This bit is read as 0.<br>0: Invalid (writing 0 has no effect)<br>1: Clear the expected value for the next transaction to DATA0 | R/W*1 |
| 9 | ACLRM | Auto Buffer Clear Mode*3<br>0: Disable<br>1: Enable (initialize all buffers) | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 10 | ATREPM | Auto Response Mode*2<br>0: Disable auto response mode<br>1: Enable auto response mode | R/W |
| 13:11 | — | These bits are read as 0. The write value should be 0. | R/W |
| 14 | INBUFM | Transmit Buffer Monitor<br>0: No data to be transmitted is in the FIFO buffer<br>1: Data to be transmitted is in the FIFO buffer | R |
| 15 | BSTS | Buffer Status<br>0: Buffer access by the CPU disabled<br>1: Buffer access by the CPU enabled | R |

Note 1. Only 0 can be read.
Note 2. Only set the ATREPM bit or write 1 to the SQCLR or SQSETbit while PID is NAK. Before setting these bits, check that the PBUSY bit is 0, and then change the PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00 (NAK) by the USBFS, checking the PBUSYbit through the software is not necessary.
Note 3. Only set the ACLRM bit while PID is NAK and before the pipe is selected in the CURPIPE[3:0] bits in the port select register. Before setting this bit, check that the PBUSY bit is 0, and then change the PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00 (NAK) by the USBFS, checking the PBUSY bit through the software is not necessary.

PIPEnCTR can be set for any pipe selection in the PIPESEL register.

### PID[1:0] bits (Response PID)

The PID[1:0]bits specify the response type for the next transaction on the selected pipe.

The default PID[1:0] setting is NAK. Change the PID[1:0] setting to BUF to use the associated pipe for USB transfer. Table 26.9 and Table 26.10 show the basic operations of the USBFS (when there are no errors in the communication packets) based on the PID[1:0] bit setting.

After changing the PID[1:0] setting from BUF to NAK through the software during USB communication on the selected pipe, check that the PBUSY bit is 1 to see if USB transfer on the pipe has actually entered the NAK state. If the USBFS changes the PID[1:0] bits to NAK, checking the PBUSY bit through the software is not necessary.

The USBFS changes the PIPEnCTR.PID[1:0] setting in the following cases:

- The USBFS sets PID to NAK on recognizing completion of the transfer when the selected pipe is in the receiving direction and the PIPECFG.SHTNAK bit for the selected pipe is set to 1 by software
- The USBFS sets PID to STALL (11b) on receiving a data packet with a payload exceeding the maximum packet size of the selected pipe
- The USBFS sets PID to NAK on detecting a USB bus reset in device controller mode
- The USBFS sets PID to NAK on detecting a reception error, such as a CRC error, three consecutive times in host controller mode
- The USBFS sets PID to STALL (11b) on receiving the STALL handshake in host controller mode

To specify the response type, set the PID[1:0] bits as follows:

- To transition from NAK (00b) to STALL, set 10b
- To transition from BUF (01b) to STALL, set 11b
- To transition from STALL (11b) to NAK, set 10b and then 00b
- To transition from STALL to BUF, set 00b (NAK) and then 01b (BUF)

**Table 26.9 Operation of the USBFS based on the PID[1:0] setting in host controller mode (1 of 2)**

| PID[1:0] value | Transfer type | Transfer direction (DIR bit) | USBFS operation |
|---|---|---|---|
| 00b (NAK) | Does not depend on the setting | Does not depend on the setting | Does not issue tokens |

**Table 26.9 Operation of the USBFS based on the PID[1:0] setting in host controller mode (2 of 2)**

| PID[1:0] value | Transfer type | Transfer direction (DIR bit) | USBFS operation |
|---|---|---|---|
| 01b (BUF) | Bulk or interrupt | Does not depend on the setting | Issues tokens when the DVSTCTR0.UACT bit is 1 and the FIFO buffer associated with the selected pipe is ready for transmission and reception.<br>Does not issue tokens when the DVSTCTR0.UACT bit is 0 or the FIFO buffer associated with the selected pipe is not ready for transmission or reception. |
| | Isochronous | Does not depend on the setting | Issues tokens regardless of the status of the FIFO buffer associated with the selected pipe. |
| 10b (STALL) or 11b (STALL) | Does not depend on the setting | Does not depend on the setting | Does not issue tokens. |

**Table 26.10 Operation of the USBFS based on the PID[1:0] setting in device controller mode**

| PID[1:0] value | Transfer type | Transfer direction (DIR bit) | USBFS operation |
|---|---|---|---|
| 00b (NAK) | Bulk or interrupt | Does not depend on the setting | Returns NAK in response to the token from the USB host |
| | Isochronous | Does not depend on the setting | Returns nothing in response to the token from the USB host |
| 01b (BUF) | Bulk | Receiving direction (DIR = 0) | Receives data and returns ACK in response to the OUT token from the USB host if the FIFO buffer associated with the selected pipe is ready for reception |
| | Interrupt | Receiving direction (DIR = 0) | Receives data and returns ACK in response to the OUT token from the USB host if the FIFO buffer associated with the selected pipe is ready for reception |
| | Bulk or interrupt | Transmitting direction (DIR = 1) | Transmits data in response to the token from the USB host if the FIFO buffer associated with the selected pipe is ready for transmission. Otherwise, returns NAK. |
| | Isochronous | Receiving direction (DIR = 0) | Receives data in response to the OUT token from the USB host if the FIFO buffer associated with the selected pipe is ready for reception. Otherwise, discards the data. |
| | Isochronous | Transmitting direction (DIR = 1) | Transmits data in response to the token from the USB host if the associated FIFO buffer is ready for transmission. Otherwise, transmits a zero-length packet. |
| 10b (STALL) or 11b (STALL) | Bulk or interrupt | Does not depend on the setting | Returns STALL in response to the token from the USB host |
| | Isochronous | Does not depend on the setting | Returns nothing in response to the token from the USB host |

### PBUSY bit (Pipe Busy)

The PBUSY bit indicates whether the selected pipe is being used for the current transaction.

The USBFS changes the PBUSY bit from 0 to 1 on start of the USB transaction for the selected pipe, and changes the PBUSY bit from 1 to 0 on completion of one transaction.

Reading the PBUSY bit by software after PID is set to NAK allows you to check whether changing the pipe setting is possible. For details, see section 26.3.4.1. Pipe control register switching procedures.

### SQMON bit (Sequence Toggle Bit Confirmation)

The SQMON bit indicates the expected value of the sequence toggle bit for the next transaction of the selected pipe.

When the selected pipe is not the isochronous transfer type, the USBFS toggles the SQMON flag on successful completion of the transaction. However, the USBFS does not toggle the SQMON flag when a DATA-PID mismatch occurs during transfer in the receiving direction.

### SQSET bit (Sequence Toggle Bit Set)

Setting the SQSET bit to 1 through the software causes the USBFS to set DATA1 as the expected value of the sequence toggle bit for the next transaction on the selected pipe. The USBFS clears the SQSET bit to 0.

### SQCLR bit (Sequence Toggle Bit Clear)

Setting the SQCLR bit to 1 through the software causes the USBFS to clear the expected value of the sequence toggle bit for the next transaction on the selected pipe to DATA0. The USBFS clears the SQCLR bit to 0.

### ACLRM bit (Auto Buffer Clear Mode)

The ACLRM bit enables or disables auto buffer clear mode for the selected pipe. To completely clear the data in the FIFO buffer allocated to the selected pipe, write 1 and then 0 to the ACLRM bit continuously.

Table 26.11 shows the data cleared by writing 1 and 0 to the ACLRM bit continuously and the cases in which this processing is required.

**Table 26.11 Data cleared by the USBFS when ACLRM = 1**

| Number | Data cleared by setting the ACLRM bit | Situations requiring data clear |
|---|---|---|
| 1 | All data in the FIFO buffer allocated to the selected pipe (two FIFO buffers in double buffer mode) | When initializing the selected pipe |
| 2 | Interval count value when the selected pipe is the isochronous transfer type | When resetting the interval count value |
| 3 | Internal flags related to the PIPECFG.BFRE bit | When changing the PIPECFG.BFRE setting |
| 4 | FIFO buffer toggle control | When changing the PIPECFG.DBLB setting |
| 5 | Internal flags related to the transaction count | When forcing the transaction count function to terminate |

### ATREPM bit (Auto Response Mode)

The ATREPM bit enables or disables auto response mode for the selected pipe.

This bit can be set to 1 in device controller mode when the selected pipe is the bulk transfer type. When the bit is set to 1, the USBFS responds to the token from the USB host as follows:

- When the selected pipe is set for bulk IN transfers (PIPECFG.TYPE[1:0] = 01b and PIPECFG.DIR = 1):
  a. When the ATREPM bit = 1 and PID = BUF, the USBFS transmits a zero-length packet in response to the IN token.
  b. The USBFS updates the sequence toggle bit (DATA-PID) each time the USBFS receives ACK from the USB host. In a single transaction, the IN token is received, a zero-length packet is transmitted, and then ACK is received. The USBFS does not generate the BRDY or BEMP interrupt.

- When the selected pipe is set for bulk OUT transfers (PIPECFG.TYPE[1:0] = 01b and PIPECFG.DIR = 0):
  When the ATREPM bit = 1 and PID = BUF, the USBFS returns NAK in response to the OUT token and generates an NRDY interrupt.

For USB communication in auto response mode, set the ATREPM bit to 1 while the FIFO buffer is empty. Do not write to the FIFO buffer during USB communication in auto response mode. When the selected pipe uses isochronous transfer, always set this bit to 0.

In host controller mode, always set the ATREPM bit to 0.

### INBUFM bit (Transmit Buffer Monitor)

The INBUFM bit indicates the FIFO buffer status for the selected pipe in the transmitting direction.

When the selected pipe is set in the transmitting direction (PIPECFG.DIR = 1), the USBFS sets this bit to 1 when the CPU or DMA/DTC completes writing data to at least one FIFO buffer plane.

The USBFS sets this bit to 0 when the USBFS completes transmission of the data from the FIFO buffer plane to which all the data is written. In double buffer mode (PIPECFG.DBLB = 1), the USBFS sets the INBUFM bit to 0 when the USBFS completes transmission of the data from the two FIFO buffer planes before the CPU or DMA/DTC completes writing data to one FIFO buffer plane.

The INBUFM bit indicates the same value as the BSTS bit when the selected pipe is in the receiving direction (PIPECFG.DIR = 0).

### BSTS bit (Buffer Status)

The BSTS bit indicates the FIFO buffer status for the selected pipe.

The meaning of the BSTS bit depends on the PIPECFG.DIR, PIPECFG.BFRE, and DnFIFOSEL.DCLRM settings, as shown in Table 26.12.

**Table 26.12 BSTS bit operation**

| DIR value | BFRE value | DCLRM value | BSTS bit function |
|---|---|---|---|
| 0 | 0 | 0 | Sets to 1 when receive data can be read from the FIFO buffer, and clears to 0 on completion of data read |
| | | 1 | Setting prohibited |
| | 1 | 0 | Sets to 1 when receive data can be read from the FIFO buffer, and clears to 0 when the software sets the BCLR bit in the port control register to 1 after the data read is complete |
| | | 1 | Sets to 1 when receive data can be read from the FIFO buffer, and clears to 0 on completion of data read |
| 1 | 0 | 0 | Sets to 1 when transmit data can be written to the FIFO buffer, and clears to 0 on completion of data write |
| | | 1 | Setting prohibited |
| | 1 | 0 | Setting prohibited |
| | | 1 | Setting prohibited |

## 26.2.36 PIPEnCTR : PIPEn Control Registers (n = 6 to 9)

Base address: USBFS = 0x4009_0000

Offset address: 0x07A + 0x2 × (n - 6)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BSTS | — | — | — | — | — | ACLRM | SQCLR | SQSET | SQMON | PBUSY | — | — | — | PID[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | PID[1:0] | Response PID<br>0 0: NAK response<br>0 1: BUF response (depends buffer state)<br>1 0: STALL response<br>1 1: STALL response | R/W |
| 4:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5 | PBUSY | Pipe Busy<br>0: Pipe n not in use for the transaction<br>1: Pipe n in use for the transaction | R |
| 6 | SQMON | Sequence Toggle Bit Confirmation<br>0: DATA0<br>1: DATA1 | R |
| 7 | SQSET | Sequence Toggle Bit Set*1<br>Sets the sequence toggle bit for pipe n.<br>0: Invalid (writing 0 has no effect)<br>1: Set the expected value for the next transaction to DATA0 | W |
| 8 | SQCLR | Sequence Toggle Bit Clear*1<br>Clears the sequence toggle bit for pipe n.<br>0: Invalid (writing 0 has no effect)<br>1: Clear the expected value for the next transaction to DATA0 | W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9 | ACLRM | Auto Buffer Clear Mode*2<br>0: Disable<br>1: Enable (all buffers initialized) | R/W |
| 14:10 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15 | BSTS | Buffer Status<br>0: Buffer access disabled<br>1: Buffer access enabled | R |

Note 1. Only write 1 to the SQCLR or SQSET bit while PID is NAK. Before setting these bits, check that the PBUSY bit is 0, and then change the PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00b (NAK) by the USBFS, checking the PBUSY bit through the software is not necessary.

Note 2. Only set the ACLRM bit while PID is NAK and before the pipe is selected in the CURPIPE[3:0] bits in the port select register. Before setting this bits, check that the PIPEnCTR.PBUSY bit is 0, and then change the PIPEnCTR.PID[1:0] bits from 01b (BUF) to 00b (NAK). If the PID[1:0] bits are changed to 00b (NAK) by the USBFS, checking the PBUSY bit through the software is not necessary.

### PID[1:0] bits (Response PID)

The PID[1:0]bits specify the response type for the next transaction of the selected pipe.

The default PID[1:0] setting is NAK. Change the PID[1:0] setting to BUF to use the associated pipe for USB transfer. Table 26.9 and Table 26.10 show the basic operation (when there are no errors in the transmitted and received packets) of the USBFS depending on the PID[1:0] setting.

After changing the PID[1:0] setting from BUF to NAK through the software during USB communication on the selected pipe, check that the PBUSY bit is 1 to see if USB transfer on the selected pipe has actually entered the NAK state. If the USBFS changes the PID[1:0] bits to NAK, checking the PBUSY bit through the software is not necessary.

The USBFS changes the PIPEnCTR.PID[1:0] setting in the following cases:

- The USBFS sets PID to STALL (11b) on receiving a data packet with a payload exceeding the maximum packet size of the selected pipe
- The USBFS sets PID to NAK on detecting a USB bus reset in device controller mode
- The USBFS sets PID to NAK on detecting a reception error, such as a CRC error, three consecutive times in host controller mode
- The USBFS sets PID to STALL (11b) on receiving the STALL handshake in host controller mode

To specify each response type, set the PID[1:0] bits as follows:

- To transition from NAK (00b) to STALL, set 10b
- To transition from BUF (01b) to STALL, set 11b
- To transition from STALL (11b) to NAK, set 10b and then 00b
- To transition from STALL to BUF, set 00b (NAK) and then 01b (BUF)

### PBUSY bit (Pipe Busy)

The PBUSY bit indicates whether the selected pipe is being used for the current transaction.

The USBFS changes the PBUSY bit from 0 to 1 on start of the USB transaction for the selected pipe, and changes the PBUSY bit from 1 to 0 on completion of one transaction.

Reading the PBUSY bit by software after PID is set to NAK allows you to check whether changing the pipe setting is possible.

### SQMON bit (Sequence Toggle Bit Confirmation)

The SQMON bit indicates the expected value of the sequence toggle bit for the next transaction of the selected pipe.

The USBFS toggles the SQMON bit on successful completion of the transaction. However, the USBFS does not toggle the SQMON bit when a DATA-PID mismatch occurs during transfer in the receiving direction.

### SQSET bit (Sequence Toggle Bit Set)

Setting the SQSET bit to 1 through the software causes the USBFS to set DATA1 as the expected value of the sequence toggle bit for the next transaction on the selected pipe. The USBFS sets the SQSET bit to 0.

### SQCLR bit (Sequence Toggle Bit Clear)

Setting the SQCLR bit to 1 through the software causes the USBFS to clear the expected value of the sequence toggle bit for the next transaction on the selected pipe to DATA0. The USBFS sets the SQCLR bit to 0.

### ACLRM bit (Auto Buffer Clear Mode)

The ACLRM bit enables or disables auto buffer clear mode for the selected pipe. To completely clear the data in the FIFO buffer allocated to the selected pipe, write 1 and then 0 to the ACLRM bit continuously.

Table 26.13 shows the data cleared by writing 1 and 0 continuously to the ACLRM bit and the cases in which this processing is required.

**Table 26.13 Data cleared by the USBFS when ACLRM = 1**

| Number | Data cleared by setting the ACLRM bit | Situations requiring data clear |
|---|---|---|
| 1 | All data in the FIFO buffer allocated to the selected pipe | When initializing the selected pipe |
| 2 | Interval count value when the selected pipe is the isochronous transfer type | When resetting the interval count value |
| 3 | Internal flags related to the PIPECFG.BFRE bit | When changing the PIPECFG.BFRE setting |
| 4 | Internal flags related to the transaction count | When forcing the transaction count function to terminate |

### BSTS bit (Buffer Status)

The BSTS bit indicates the FIFO buffer status for the selected pipe.

The meaning of the BSTS bit depends on the PIPECFG.DIR, PIPECFG.BFRE, and DnFIFOSEL.DCLRM settings, as shown in Table 26.12.

## 26.2.37 PIPEnTRE : PIPEn Transaction Counter Enable Register (n = 1 to 5)

Base address: USBFS = 0x4009_0000

Offset address: 0x090 + 0x4 × (n - 1)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | TRENB | TRCLR | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | TRCLR | Transaction Counter Clear<br>0: Invalid (writing 0 has no effect)<br>1: Clear counter value | R/W |
| 9 | TRENB | Transaction Counter Enable<br>0: Disable transaction counter<br>1: Enable transaction counter | R/W |
| 15:10 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: Set each bit in PIPEnTRE while PID is NAK. Before setting these bits after changing the PIPEnCTR.PID[1:0] bits for the selected pipe from BUF to NAK, check that the PIPEnCTR.PBUSY bit is 0. However, if the PID[1:0] bits are changed to NAK by the USBFS, checking the PBUSY bit through the software is not necessary.

### TRCLR bit (Transaction Counter Clear)

When the TRCLR bit sets to 1, the USBFS clears the value of the transaction counter associated with the selected pipe and then sets the TRCLR bit to 0.

### TRENB bit (Transaction Counter Enable)

The TRENB bit enables or disables the transaction counter.

For receiving pipes, setting the TRENB bit to 1 after setting the total number of the packets to be received in the PIPEnTRN.TRNCNT[15:0] bits through the software allows the USBFS to control hardware on having received the number of packets equal to the TRNCNT[15:0] setting, as follows:

- When the PIPECFG.SHTNAK bit is 1, the USBFS changes the PID bits to NAK for the associated pipe on having received the number of packets equal to the TRNCNT[15:0] setting
- When the PIPECFG.BFRE bit is 1, the USBFS asserts the BRDY interrupt on having received the number of packets equal to the TRNCNT[15:0] setting and then reading the last received data

For transmitting pipes, set the TRENB bit to 0.

When the transaction counter is not used, set this bit to 0. When the transaction counter is used, set the TRNCNT[15:0] bits before setting this bit to 1. Set this bit to 1 before receiving the first packet to be counted by the transaction counter.

## 26.2.38 PIPEnTRN : PIPEn Transaction Counter Register (n = 1 to 5)

Base address: USBFS = 0x4009_0000

Offset address: 0x092 + 0x4 × (n - 1)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | TRNCNT[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | TRNCNT[15:0] | Transaction Counter<br>When written to, this bit specifies the total packets (number of transactions) to be received by the selected pipe.<br>When read from, when PIPEnTRE.TRENB is 0, this bit indicates the specified number of transactions.<br>When PIPEnTRE.TRENB is 1, this bit indicates the current transaction count. | R/W |

The PIPEnTRN registers retain their settings during a USB bus reset.

### TRNCNT[15:0] bits (Transaction Counter)

The USBFS increments the value of the TRNCNT[15:0] bits by 1 when all of the following conditions are satisfied on receiving the packet:

- The PIPEnTRE.TRENB bit = 1
- (TRNCNT[15:0] set value ≠ current counter value + 1) on receiving the packet
- The payload of the received packet agrees with the PIPEMAXP.MXPS[9:0] setting

The USBFS clears the value of the TRNCNT[15:0] bits to 0 when any of the following conditions are satisfied:

All of the following conditions are satisfied:

- The PIPEnTRE.TRENB bit = 1
- (TRNCNT[15:0] set value = current counter value + 1) on receiving the packet
- The payload of the received packet agrees with the PIPEMAXP.MXPS[9:0] setting

Both of the following conditions are satisfied:

- The PIPEnTRE.TRENB bit = 1
- The USBFS received a short packet

Both of the following conditions are satisfied:

- The PIPEnTRE.TRENB bit = 1
- The PIPEnTRE.TRCLR bit was set to 1 by software

For transmitting pipes, set the TRNCNT[15:0] bits to 0. When the transaction counter is not used, set the TRNCNT[15:0] bits to 0.

Setting the number of transactions to be transferred to the TRNCNT[15:0] bits is only enabled when the PIPEnTRE.TRENB bit is 0. To set the number of transactions to be transferred, set the TRCLR bit to 1 to clear the current counter value before setting the PIPEnTRE.TRENB bit to 1.

## 26.2.39 BCCTRL1 : Battery Charging Control Register 1

Base address: USBFS = 0x4009_0000

Offset address: 0x0B0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | CHGDETSTS | PDDETSTS | — | — | CHGDETE | PDDETE | VDPSRCE | VDMSRCE | IDPSRCE | RPDME |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RPDME | D- Line Pull-down Control<br>0: Disable D- Line Pull-down<br>1: Enable D- Line Pull-down | R/W |
| 1 | IDPSRCE | D+ Line IDPSRC Output Control<br>0: Stopped<br>1: 10 µA output | R/W |
| 2 | VDMSRCE | D- Line VDMSRC (0.6 V) Output Control<br>0: Stopped<br>1: 0.6 V output | R/W |
| 3 | VDPSRCE | D+ Line VDPSRC (0.6 V) Output Control<br>0: Stopped<br>1: 0.6 V output | R/W |
| 4 | PDDETE | D+ Line 0.6 V Input Detection Control<br>0: Disable detection<br>1: Enable detection | R/W |
| 5 | CHGDETE | D- Line 0.6 V Input Detection Control<br>0: Disable detection<br>1: Enable detection | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | PDDETSTS | D+ Line 0.6 V Input Detection Status Flag<br>This Flag is valid when PDDETE = 1.<br>0: Not detected<br>1: Detected | R |
| 9 | CHGDETSTS | D- Line 0.6 V Input Detection Status Flag<br>This flag is valid when CHGDETE = 1.<br>0: Not detected<br>1: Detected | R |
| 31:10 | — | These bits are read as 0. The write value should be 0. | R/W |

### RPDME bit (D- Line Pull-down Control)

In device controller operation, set this bit to 1 to perform Data Contact Detect. In the Battery Charging Specification Revision 1.2, there are two methods to handle Data Contact Detect; the method realized by software wait and the method to contact the data line by hardware. The RPDME bit adopts the latter method. When RPDME = 1, the USBFS controls D- line pull-down.

#### IDPSRCE bit (D+ Line IDPSRC Output Control)

In device controller operation, set this bit to 1 to perform Data Contact Detect. In the Battery Charging Specification Revision 1.2, there are two methods to handle Data Contact Detect; the method realized by software wait and the method to contact the data line by hardware. The IDPSRCE bit adopts the latter method. When IDPSRCE = 1, the USBFS enables the IDP_SRC circuit

#### VDMSRCE bit (D- Line VDMSRC (0.6 V) Output Control)

In host controller operation, during Primary Detection, this bit controls VDMSRC (0.6 V) output from the USB_DM pin.

In device controller operation, during Secondary Detection, this bit controls VDMSRC (0.6 V) output from the USB_DM pin.

#### VDPSRCE bit (D+ Line VDPSRC (0.6 V) Output Control)

In device controller operation, if Primary Detection is executed, this bit controls VDPSRC (0.6 V) output from the USB_DP pin.

#### PDDETE bit (D+ Line 0.6 V Input Detection Control)

When the PDDETE bit is set to 1, the following states can be detected.

In host controller operation, during Primary Detection, VDPSRC (0.6 V) is input to the USB_DP pin from a peripheral device.

In device controller operation, during Secondary Detection, VDPSRC (0.6 V) output from the USBFS to the USB_DM pin is input to the USB_DP pin via the host.

#### CHGDETE bit (D- Line 0.6 V Input Detection Control)

In device controller operation, the following can be detected when the CHGDETE bit is set to 1.

During Primary Detection, VDMSRC (0.6 V) is input to the USB_DM pin from the host.

During Primary Detection, VDPSRC (0.6 V) output from the USBFS to the USB_DP pin is input to the USB_DM pin via the USB host.

#### PDDETSTS flag (D+ Line 0.6 V Input Detection Status Flag)

The PDDETSTS flag is enabled when the PDDETE bit is 1. The PDDETSTS flag becomes 1 under the following conditions.

In host controller operation, during Primary Detection, VDPSRC (0.6V) is input to the USB_DP pin from a peripheral device.

In device controller operation, during Secondary Detection, the VDMSRC (0.6V) output from the USBFS to the USB_DM pin is input to the USB_DP pin via the host.

#### CHGDETSTS flag (D- Line 0.6 V Input Detection Status Flag)

In device controller operation, this flag is enabled when the CHGDETE bit is 1. The CHGDETSTS flag becomes 1 under the following conditions.

During Primary Detection, VDMSRC (0.6V) is input to the USB_DM pin from the USB host.

During Primary Detection, the VDPSRC (0.6V) output from the USBFS to the USB_DP pin is input to the USB_DM pin via the USB host.

## 26.2.40 BCCTRL2 : Battery Charging Control Register 2

Base address: USBFS = 0x4009_0000

Offset address: 0x0B4

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | PHYDET[1:0] | | — | — | — | — | BATCHGE | DCPMODE | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | DCPMODE | Dedicated Charging Port (DCP) Mode Control<br>In host controller mode, setting this bit to 1 connects the D+ line and D- line. If USBFS is configured as DCP, this bit should be set to 1 before driving VBUS.<br>In device controller mode, this bit should be set to 0.<br>0: Disable DCP<br>1: Enable DCP | R/W |
| 7 | BATCHGE | Battery Charging Enable<br>0: Disable Battery Charging<br>1: Enable Battery Charging | R/W |
| 11:8 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13:12 | PHYDET[1:0] | Detect Sensitivity Adjustment<br>Adjusts the detect sensitivity of Portable Device and Charging D- Port<br>Initial value is 10b, but need to be set 01b. | R/W |
| 31:14 | — | These bits are read as 0. The write value should be 0. | R/W |

## 26.2.41 DEVADDn : Device Address n Configuration Register (n = 0 to 5)

Base address: USBFS = 0x4009_0000

Offset address: 0x0D0 + 0x2 × n

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | USBSPD[1:0] | | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7:6 | USBSPD[1:0] | Transfer Speed of Communication Target Device<br>0 0: Do not use DEVADDn<br>0 1: Low-speed<br>1 0: Full-speed<br>1 1: Setting prohibited | R/W |
| 15:8 | — | These bits are read as 0. The write value should be 0. | R/W |

The DEVADDn register specifies the transfer speed of the peripheral device that is the communication target for pipes 0 to 9.

In host controller mode, set all DEVADDn bits before starting communication to any pipes. Only change the bits in DEVADDn when no valid pipes are using the bit settings. A valid pipe is defined as one that satisfies both of the following conditions:

- The target device of the DEVADDn register is selected in the DEVSEL[3:0] bits
- The PID[1:0] bits are set to BUF for the selected pipe, or the selected pipe is the DCP with the DCPCTR.SUREQ bit set to 1

In device controller mode, set all bits in this register to 0.

**USBSPD[1:0] bits (Transfer Speed of Communication Target Device)**

TheUSBSPD[1:0] bits specify the USB transfer speed of the target peripheral device. Set these bits to 10b when a full-speed device is connected through the hub. In host controller mode, the USBFS generates packets based on the USBSPD[1:0] setting. In device controller mode, set these bits to 00b.

## 26.2.42 PHYSECTRL : PHY Single-ended Receiver Control Register

Base address: USBFS = 0x4009_0000

Offset address: 0x0F4

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | 4 | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | CNEN | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | CNEN | Single-ended Receiver Enable<br>0: Single-ended receiver operation is disabled<br>1: Single-ended receiver operation is enabled | R/W |
| 31:5 | — | These bits are read as 0. The write value should be 0. | R/W |

**CNEN bit (Single-ended Receiver Enable)**

Setting the CNEN bit to 1 enables single-ended receiver operation. Set this bit to 1 when perform the hardware-based Data Contact Detection using the battery charging function in device controller mode.

## 26.2.43 DPUSR0R : Deep Software Standby USB Transceiver Control/Pin Monitor Register

Base address: USBFS = 0x4009_0000

Offset address: 0x400

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | DVBSTS0 | — | — | DOVCA0 | — | — | DM0 | DP0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | 0 | 0 | x | 0 | 0 | x | x |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | FIXPHY0 | DRPD0 | — | RPUE0 | SRPC0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SRPC0*1 | USB Single-ended Receiver Control<br>0: Disable input through DP and DM inputs<br>1: Enable input through DP and DM inputs | R/W |
| 1 | RPUE0*1 | DP Pull-Up Resistor Control<br>0: Disable DP pull-up resistor<br>1: Enable DP pull-up resistor | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 3 | DRPD0*1 | D+/D- Pull-Down Resistor Control<br>0: Disable DP/DM pull-down resistor<br>1: Enable DP/DM pull-down resistor | R/W |
| 4 | FIXPHY0 | USB Transceiver Output Fix<br>0: Fix outputs in Normal mode and on return from Deep Software Standby mode<br>1: Fix outputs on transition to Deep Software Standby mode | R/W |
| 15:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | DP0 | USB D+ Input<br>Indicates D+ input signal on the USBFS side | R |
| 17 | DM0 | USB D- Input<br>Indicates D– input signal on the USBFS side | R |
| 19:18 | — | These bits are read as 0. The write value should be 0. | R/W |
| 20 | DOVCA0 | USB OVRCURA Input<br>Indicates OVRCURA input signal on the USBFS side | R |
| 21 | — | This bit is read as 0. The write value should be 0. | R/W |
| 22 | — | The read value is undefined. The write value should be 0. | R/W |
| 23 | DVBSTS0 | USB VBUS Input<br>Indicates VBUS input signal on the USBFS side | R |
| 31:24 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Use this bit during operation in Deep Software Standby mode. For details, see section 26.3.1.5. Release from Deep Software Standby mode because of USB suspend/resume interrupts.

### SRPC0 bit (USB Single-ended Receiver Control)

The SRPC0 bit controls the D+ and D- inputs of the USB transceiver. In host controller mode, set this bit to 1. In device controller mode, set this bit to 0 when disconnected, set to 1 when suspended. This bit is only valid when the FIXPHY0 bit is 1.

### FIXPHY0 bit (USB Transceiver Output Fix)

The FIXPHY0 bit keeps the outputs of the USB transceiver disabled.

## 26.2.44 DPUSR1R : Deep Software Standby USB Suspend/Resume Interrupt Register

Base address: USBFS = 0x4009_0000

Offset address: 0x404

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | DVBINT0 | — | — | DOVRCRA0 | — | — | DMINT0 | DPINT0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | DVBSE0 | — | — | DOVRCRAE0 | — | — | DMINTE0 | DPINTE0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DPINTE0 | USB DP Interrupt Enable/Clear<br>0: Disable recovery from Deep Software Standby mode by DP input<br>1: Enable recovery from Deep Software Standby mode by DP input | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | DMINTE0 | USB DM Interrupt Enable/Clear<br>0: Disable recovery from Deep Software Standby mode by DM input<br>1: Enable recovery from Deep Software Standby mode by DM input | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | DOVRCRAE0 | USB OVRCURA Interrupt Enable/Clear<br>0: Disable recovery from Deep Software Standby mode by OVRCURA input<br>1: Enable recovery from Deep Software Standby mode by OVRCURA input | R/W |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | — | This bit is read as 0. The write value should be 0. | R/W |
| 7 | DVBSE0 | USB VBUS Interrupt Enable/Clear<br>0: Disable recovery from Deep Software Standby mode by VBUS input<br>1: Enable recovery from Deep Software Standby mode by VBUS input | R/W |
| 15:8 | — | T These bits are read as 0. The write value should be 0. . | R/W |
| 16 | DPINT0 | USB DP Interrupt Source Recovery<br>0: System has not recovered from Deep Software Standby mode<br>1: System recovered from Deep Software Standby mode because of DP | R |
| 17 | DMINT0 | USB DM Interrupt Source Recovery<br>0: System has not recovered from Deep Software Standby mode<br>1: System recovered from Deep Software Standby mode because of DM input | R |
| 19:18 | — | This bit is read as 0. The write value should be 0. | R/W |
| 20 | DOVRCRA0 | USB OVRCURA Interrupt Source Recovery<br>0: System has not recovered from Deep Software Standby mode<br>1: System recovered from Deep Software Standby mode because of OVRCURA input | R |
| 21 | — | This bit is read as 0. The write value should be 0. | R/W |
| 22 | — | This bit is read as 0. The write value should be 0. | R/W |
| 23 | DVBINT0 | USB VBUS Interrupt Source Recovery<br>0: System has not recovered from Deep Software Standby mode<br>1: System recovered from Deep Software Standby mode because of VBUS input | R |
| 31:24 | — | These bits are read as 0. The write value should be 0. | R/W |

### DPINTE0 bit (USB DP Interrupt Enable/Clear)

The DPINTE0 bit enables or disables triggering of recovery from Deep Software Standby mode by the DP input of the USBFS. Writing 0 to this bit while the DPINT0 bit is 1 sets the DPINT0 bit to 0.

### DMINTE0 bit (USB DM Interrupt Enable/Clear)

The DMINTE0 bit enables or disables triggering of recovery from Deep Software Standby mode by the DM input of the USBFS. Writing 0 to this bit while the DMINT0 bit is 1 clears the DMINT0 bit to 0.

### DOVRCRAE0 bit (USB OVRCURA Interrupt Enable/Clear)

The DOVRCRAE0 bit enables or disables triggering of recovery from Deep Software Standby mode by the OVRCURA input of the USBFS. Writing 0 to this bit while the DOVRCRA0 bit is 1 clears the DOVRCRA0 bit to 0.

### DVBSE0 bit (USB VBUS Interrupt Enable/Clear)

The DVBSE0 bit enables or disables triggering of recovery from Deep Software Standby mode by the VBUS input of the USBFS. Writing 0 to this bit while the DVBINT0 bit is 1 clears the DVBINT0 bit to 0.

### DPINT0 bit (USB DP Interrupt Source Recovery)

The DPINT0 bit indicates that the system has returned from Deep Software Standby mode because of the DP input of the USBFS. This recovery is only enabled when the DPINTE0 bit is 1. Writing 0 to the DPINTE0 bit while this bit is 1 clears this bit to 0.

#### DMINT0 bit (USB DM Interrupt Source Recovery)

The DMINT0 bit indicates that the system has returned from Deep Software Standby mode because of the DM input of the USBFS. This recovery is only enabled when the DMINTE0 bit is 1. Writing 0 to the DMINTE0 bit while this bit is 1 clears this bit to 0.

#### DOVRCRA0 bit (USB OVRCURA Interrupt Source Recovery)

The DOVRCRA0 bit indicates that the system has returned from Deep Software Standby mode because of the OVRCURA input of the USBFS. This recovery is only enabled when the DOVRCRAE0 bit is 1. Writing 0 to the DOVRCRAE0 bit while this bit is 1 clears this bit to 0.

#### DVBINT0 bit (USB VBUS Interrupt Source Recovery)

The DVBINT0 bit indicates that the system has returned from Deep Software Standby mode because of the VBUS input of the USBFS. This recovery is only enabled when the DVBSE0 bit is 1. Writing 0 to the DVBSE0 bit while this bit is 1 clears this bit to 0.

## 26.3 Operation

### 26.3.1 System Control

This section describes register settings required for initializing the USBFS and controlling power consumption.

#### 26.3.1.1 Setting data to the USBFS registers

Setting the SYSCFG.USBE bit to 1 after starting the clock supply (SYSCFG.SCKE bit = 1) enables and starts USBFS operation.

#### 26.3.1.2 Selecting the controller function

The USBFS can operate as either a host or device controller.

Use the SYSCFG.DCFM bit to select one of these USBFS functions. The DCFM bit must be changed in the initial settings immediately after a reset or in the D+ pull-up-disabled state (SYSCFG.DPRPU bit = 0) and D+ and D– pull-down-disabled state (SYSCFG.DRPD bit = 0).

#### 26.3.1.3 Controlling the USB data bus using resistors

The USBFS provides pull-up and pull-down resistors for the D+ and D– lines. Pull these lines up or down by setting the SYSCFG.DPRPU and DRPD bits.

In device controller mode, confirm that connection to the USB host is made, and then set the SYSCFG.DPRPU bit to 1 and pull up the D+ line (in full-speed communication).

When the SYSCFG.DPRPU bit is set to 0 during communication with a PC, the USBFS disables the pull-up resistor of the USB data line, thereby notifying the USB host of disconnection.

In host controller mode, set the SYSCFG.DRPD bit to 1 to pull down the D+ and D– lines.

**Table 26.14 USB data bus resistor control**

| SYSCFG register settings | | USB data bus control | | |
|---|---|---|---|---|
| DRPD bit | DPRPU bit | D– | D+ | Function |
| 0 | 0 | Open | Open | When resistors not used |
| 0 | 1 | Open | Pull-up | When operating as a device controller at full-speed |
| 1 | 0 | Pull-down | Pull-down | When operating as a host controller |
| 1 | 1 | — | — | Setting prohibited |

#### 26.3.1.4 Example external connection circuits

Figure 26.2 shows an example OTG connection in the self-powered system. The USBFS controls the pull-up resistor of the D+ line and the pull-down resistor of D+ and D- lines. Select pull-up and pull-down for the lines in the SYSCFG.DPRPU

and SYSCFG.DRPD bits. In device controller mode, the pull-up resistor of USB data line is disabled if SYSCFG.DPRPU bit is set to 0 while communicating with the USB host. The USBFS can use this to notify the USB host of a device disconnect.

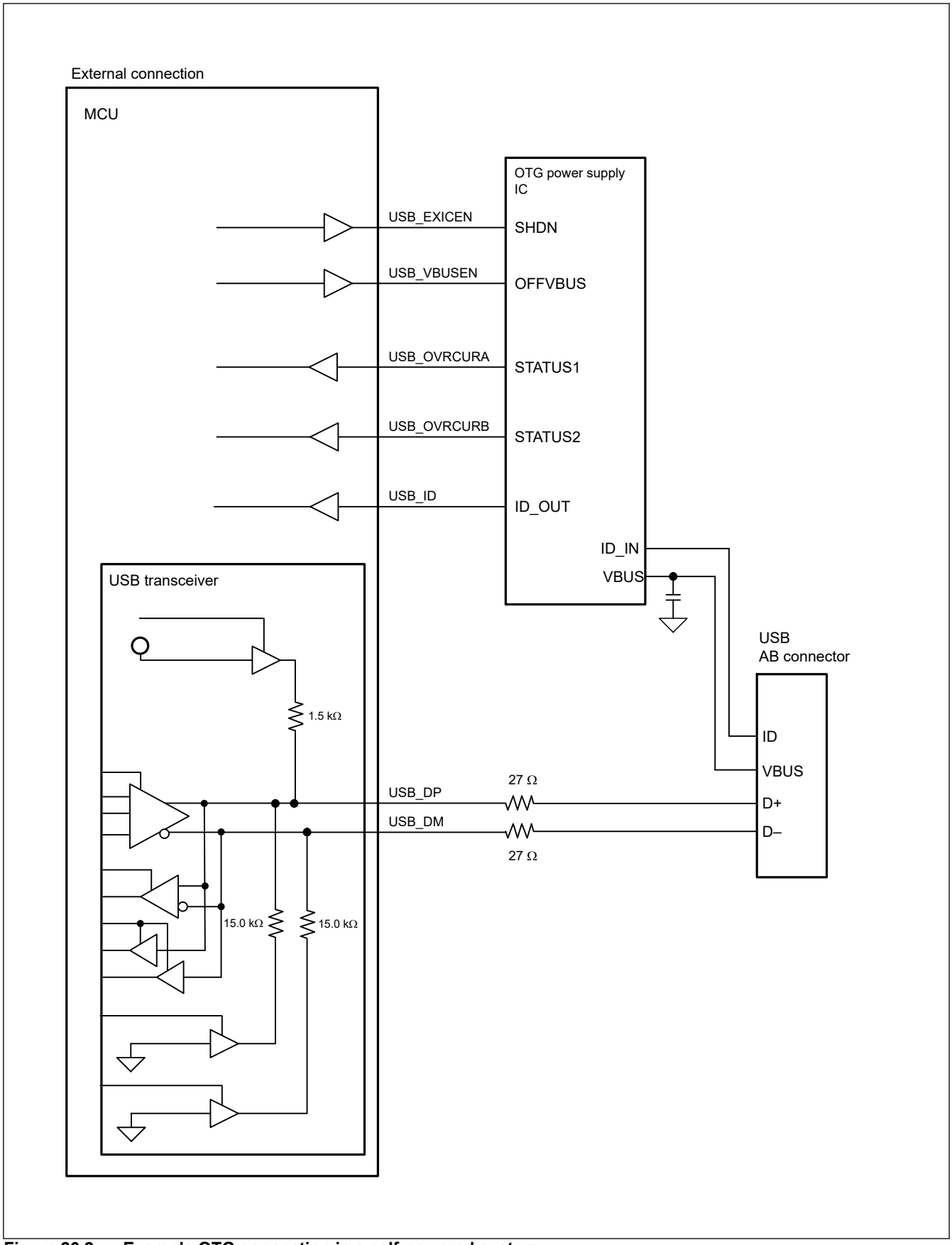



**Figure 26.2 Example OTG connection in a self-powered system**

Figure 26.3 shows an example device connection in a self-powered system.

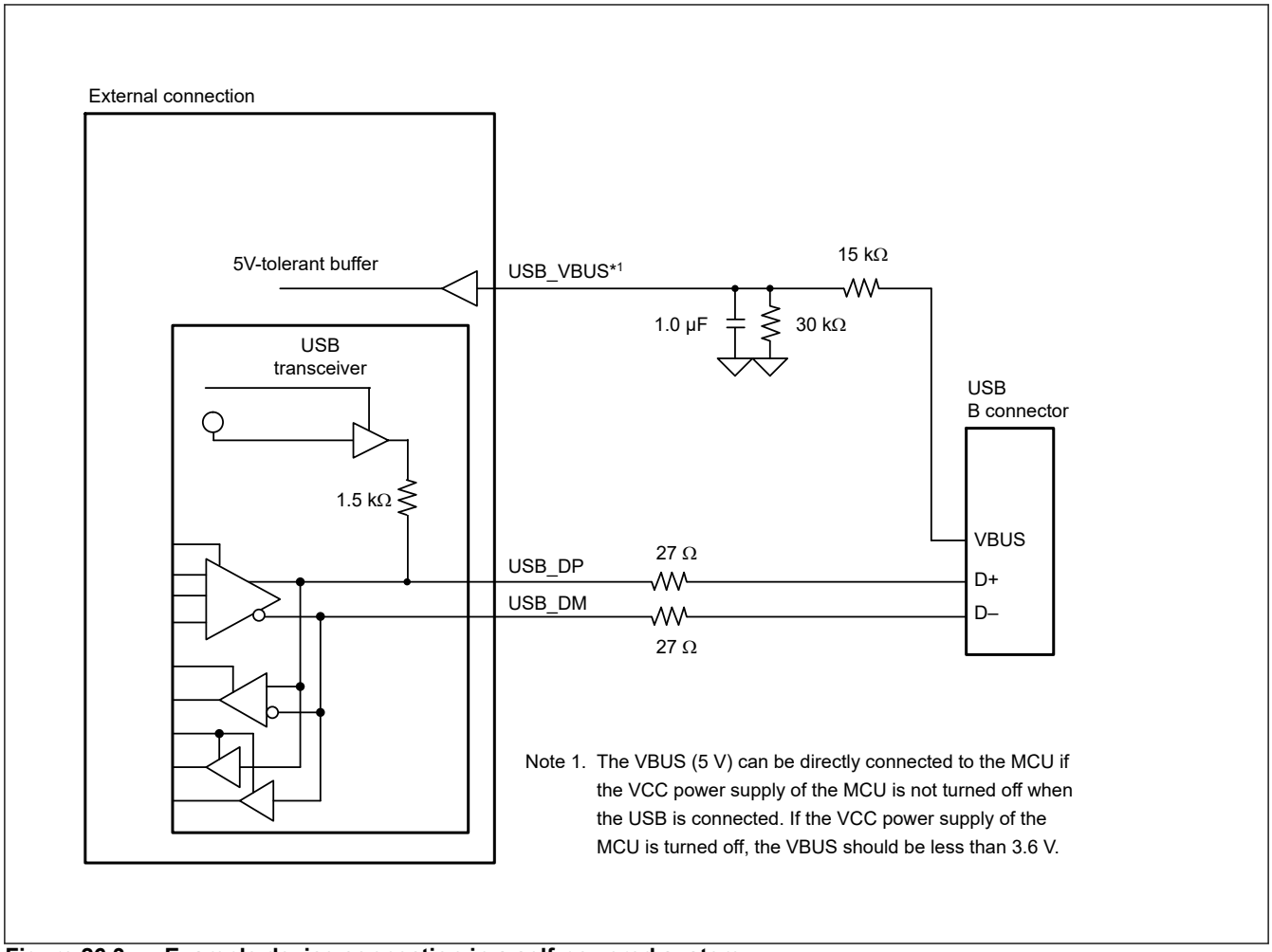



**Figure 26.3 Example device connection in a self-powered system**

Figure 26.4 shows an example host connection.

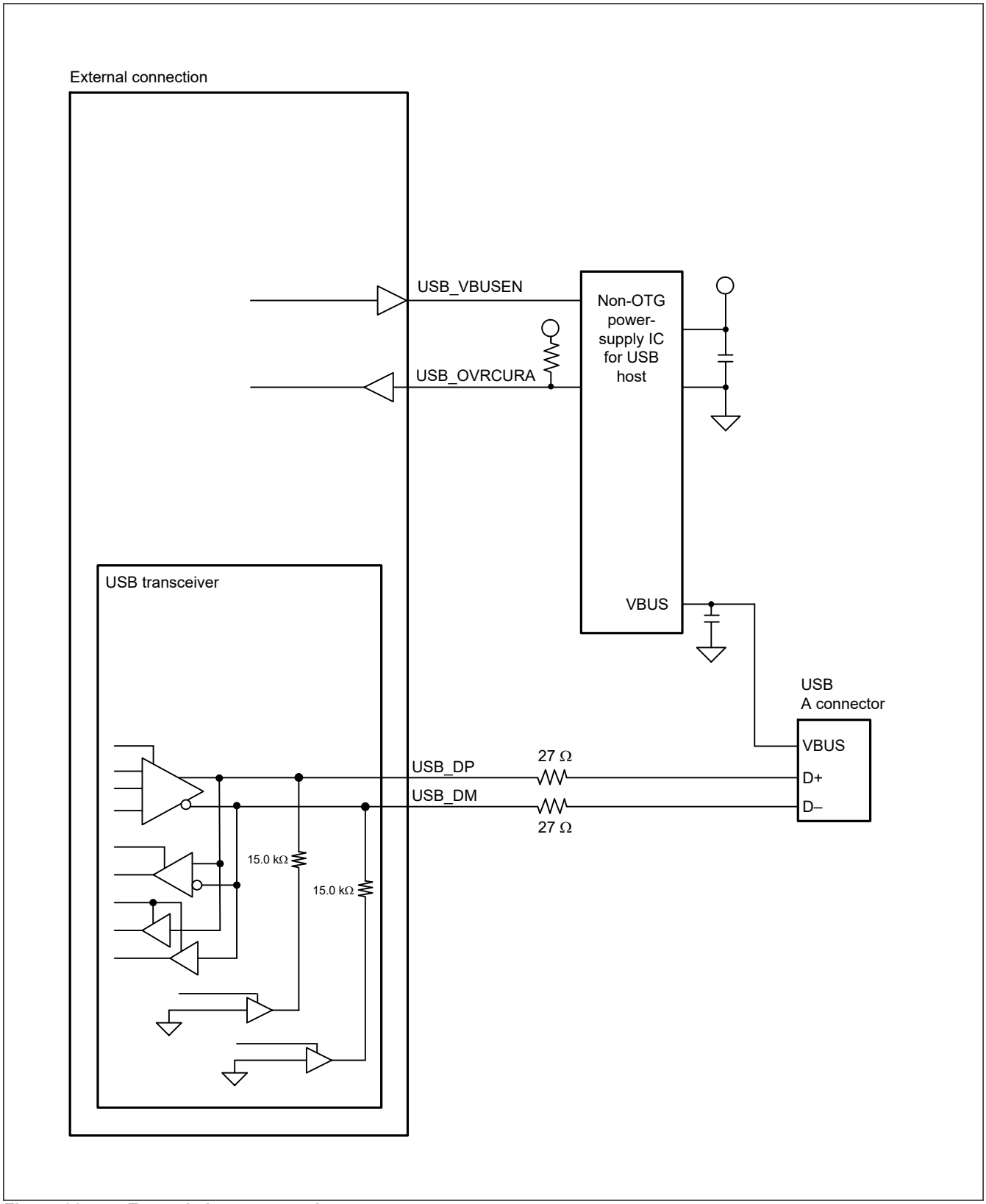



**Figure 26.4 Example host connection**

Figure 26.5 shows an example device connection in a bus-powered system.

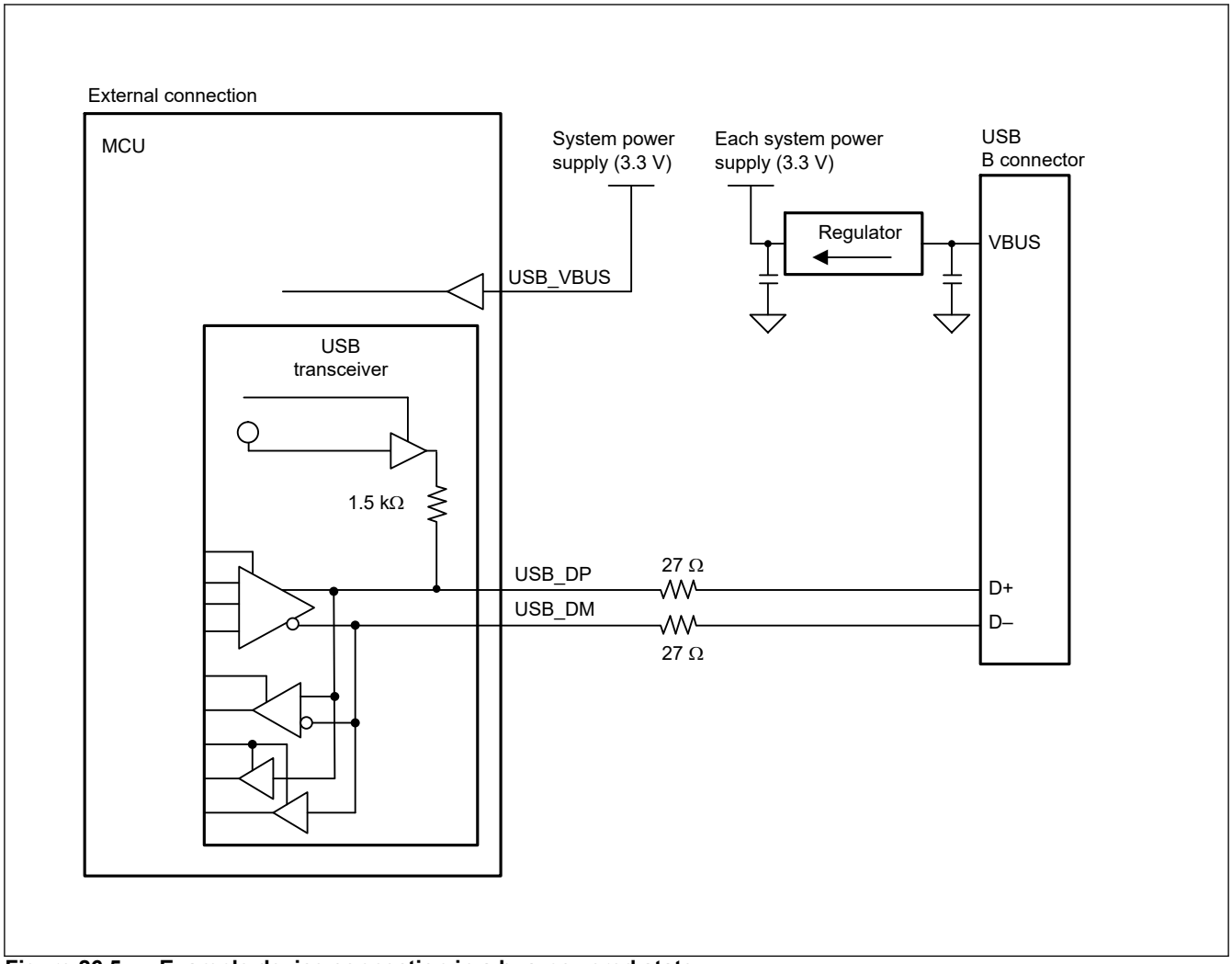



**Figure 26.5 Example device connection in a bus-powered state**

Figure 26.6 shows an example of functional connection of the USB connector with Battery Charging Rev 1.2 supported.

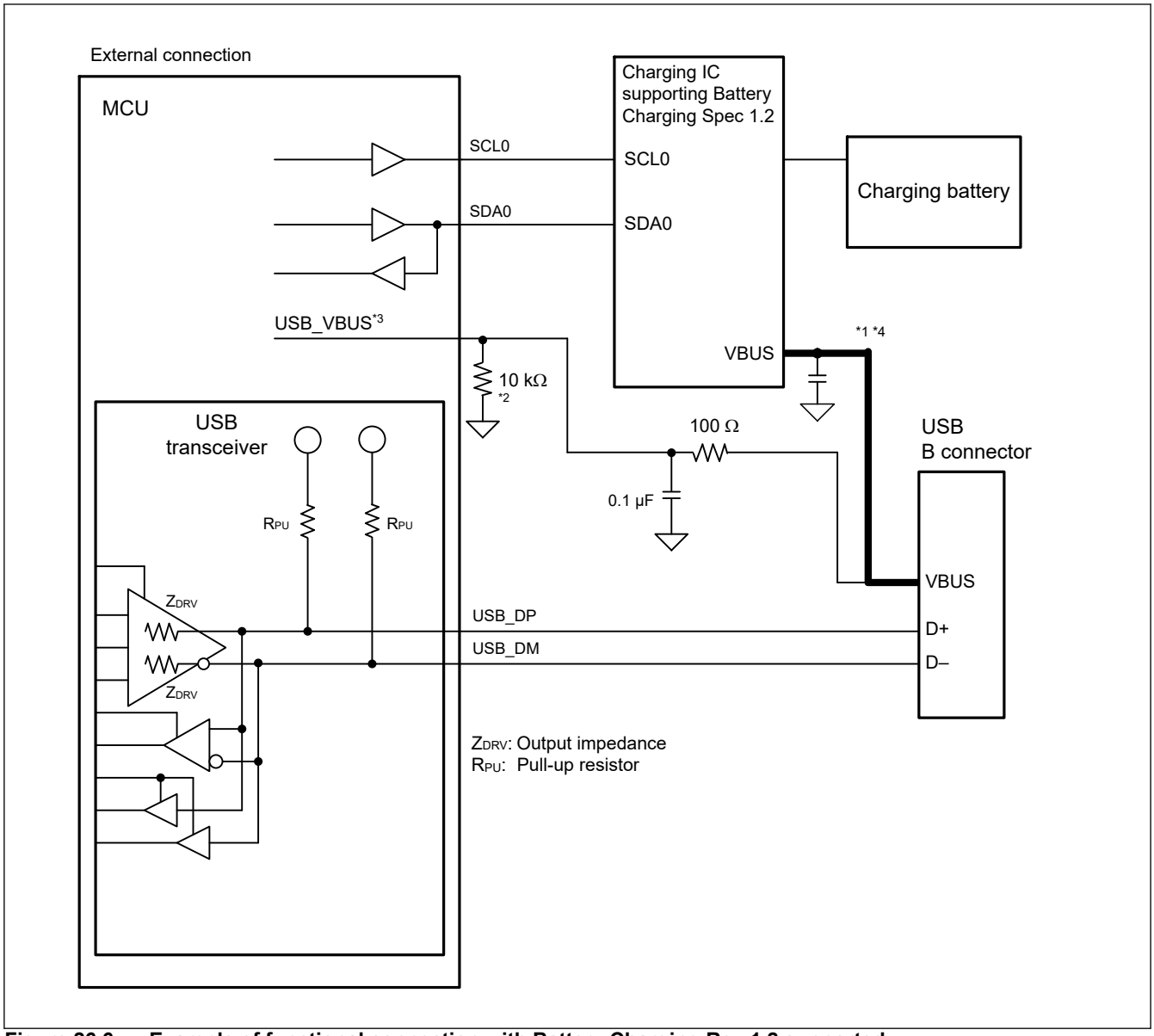



**Figure 26.6 Example of functional connection with Battery Charging Rev 1.2 supported**

The examples of external circuits given in this section are simplified circuits, and their operation in every system is not guaranteed.

### 26.3.1.5 Release from Deep Software Standby mode because of USB suspend/resume interrupts

Deep Software Standby mode can be canceled by a USB suspend/resume interrupt. USB suspend/resume interrupts are detected by the USB resume detecting unit, which controls and monitors the USB I/O pins to detect the interrupts.

Figure 26.7 shows a schematic diagram of the connection between the USB resume detecting unit and the USB I/O pins.

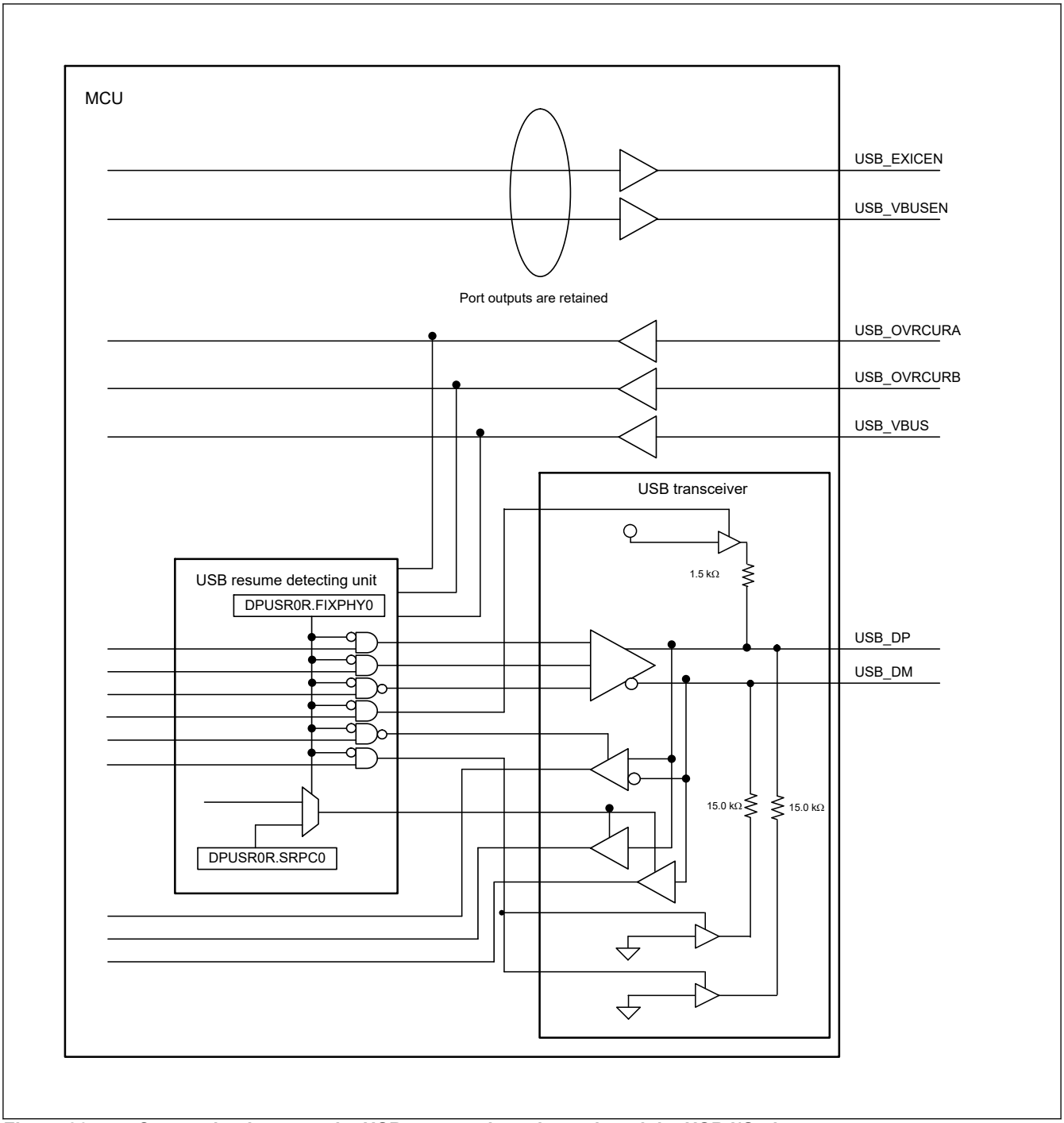



**Figure 26.7 Connection between the USB resume detecting unit and the USB I/O pins**

Table 26.15 shows the USB suspend and resume interrupt sources and their associated I/O pins.

**Table 26.15 USB suspend and resume interrupt sources and their associated I/O pins**

| USB operating mode | Source | Pin name |
|---|---|---|
| Device, OTG | Resume | USB_DP |
| Host, OTG | Attach or detach | USB_DP, USB_DM |
| Device | Attach or detach | USB_VBUS |
| Host | Overcurrent detection | USB_OVRCURA |
| OTG | Overcurrent detection | USB_OVRCURA, USB_OVRCURB |

Figure 26.8 shows the flow for setting the USBFS when entering Deep Software Standby mode from either host or device controller mode. Figure 26.9 shows the flow for setting the USBFS when canceling Deep Software Standby mode from host controller mode. Figure 26.10 shows the flow for setting the USBFS when canceling Deep Software Standby mode from device controller mode.

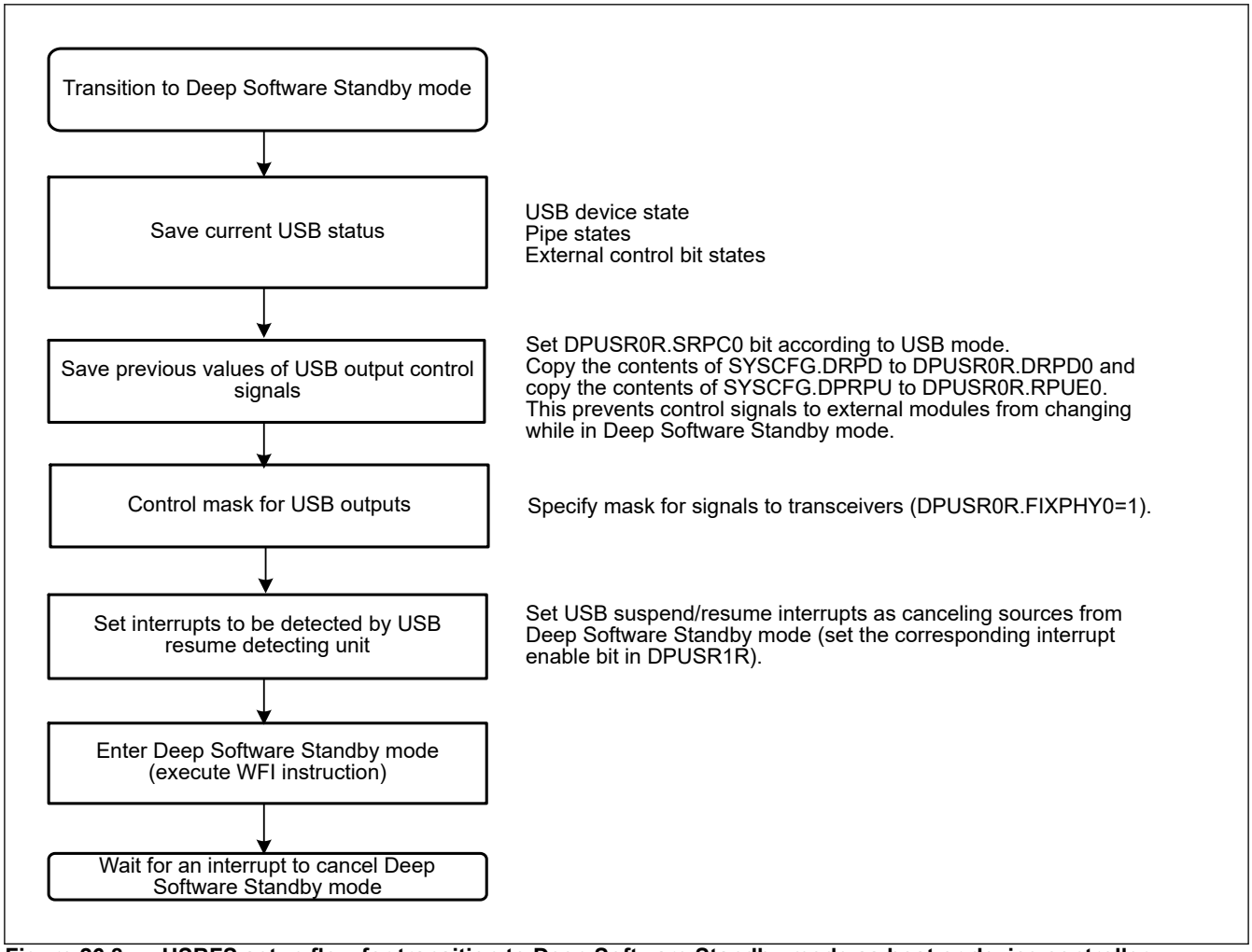



**Figure 26.8 USBFS setup flow for transition to Deep Software Standby mode as host or device controller**

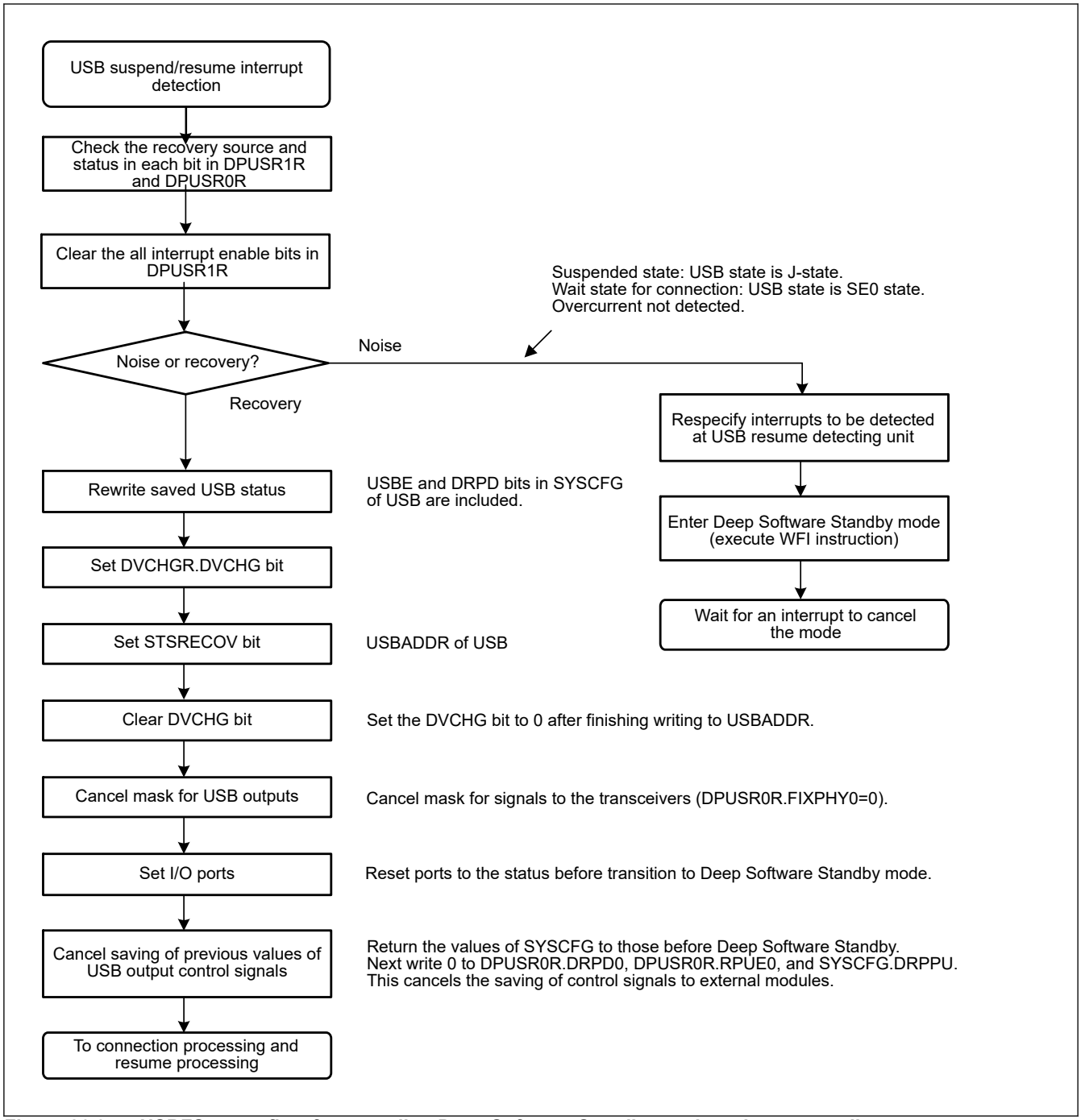



**Figure 26.9 USBFS setup flow for canceling Deep Software Standby mode as host controller**

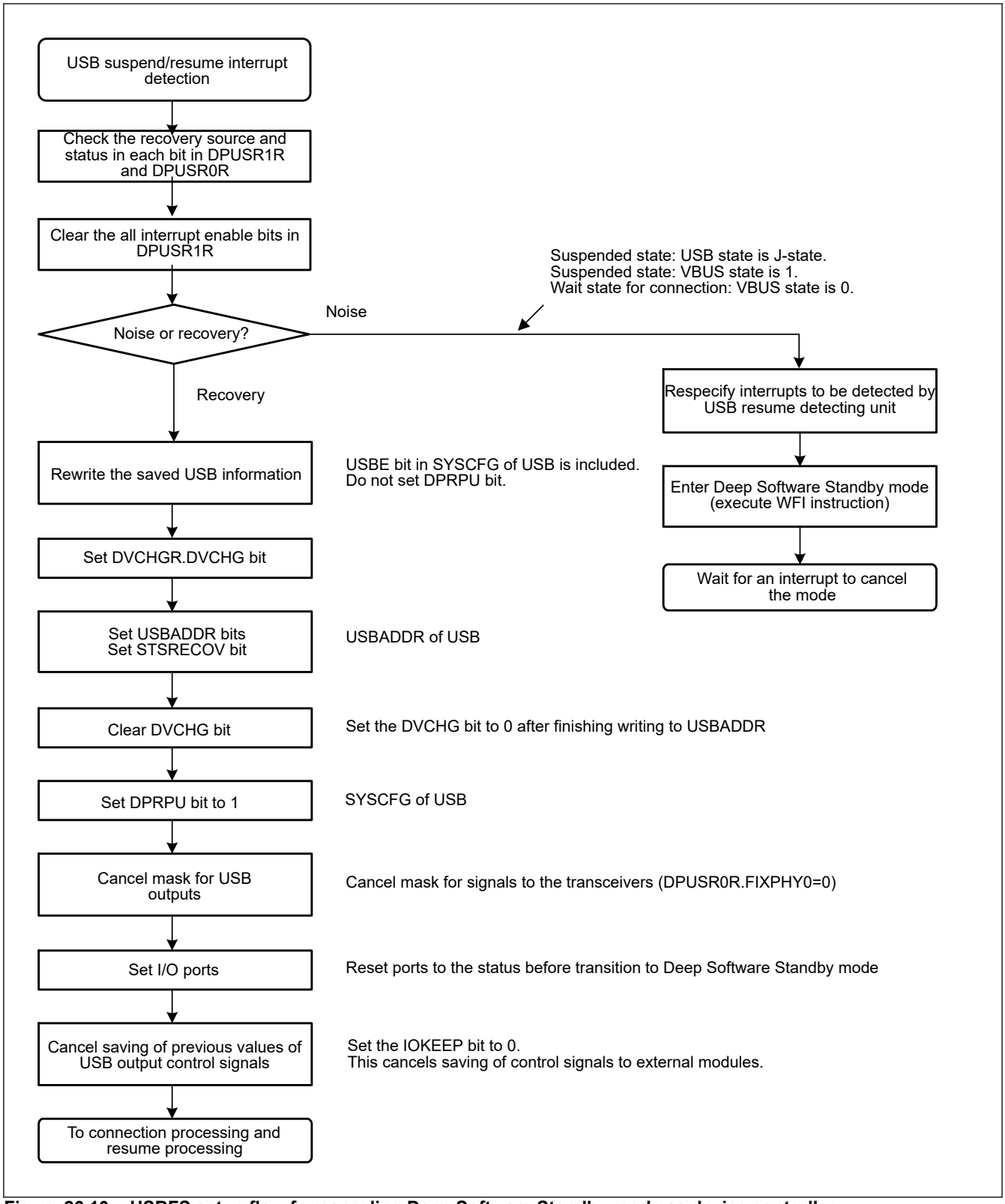



**Figure 26.10 USBFS setup flow for canceling Deep Software Standby mode as device controller**

## 26.3.2 Interrupts

Table 26.16 lists the interrupt sources in the USBFS. When an interrupt generation condition is satisfied and the interrupt output is enabled using the associated interrupt enable register, a USBFS interrupt request is issued to the Interrupt Controller Unit (ICU) and an USBFS interrupt is generated.

**Table 26.16 Interrupt sources (1 of 2)**

| Bit to be set to 1 | Name | Interrupt source | Applicable controller function | Status flag |
|---|---|---|---|---|
| VBINT | VBUS interrupt | • A change in the state of the USB_VBUS input pin was detected (low to high or high to low) | Host or device*1 | INTSTS0.VBSTS |
| RESM | Resume interrupt | • A change in the state of the USB bus was detected in the Suspend state (J-state to K-state or J-state to SE0) | Device | — |
| SOFR | Frame number update interrupt | In host controller mode:<br>• An SOF packet with a different frame number was transmitted<br>In device controller mode:<br>• An SOF packet with a different frame number was received | Host or device | — |
| DVST | Device state transition interrupt | • One of the following device state transitions was detected:<br>– USB bus reset was detected<br>– Suspend state was detected<br>– SET_ADDRESS request was received<br>– SET_CONFIGURATION request was received | Device | INTSTS0.DVSQ[2:0] |
| CTRT | Control transfer stage transition interrupt | • A control transfer stage transition was detected because of one of the following:<br>– Setup stage completed<br>– Control write transfer status stage transition occurred<br>– Control read transfer status stage transition occurred<br>– Control transfer completed<br>– Control transfer sequence error occurred | Device | INTSTS0.CTSQ[2:0] |
| BEMP | Buffer empty interrupt | • The buffer is empty after all FIFO buffer data was transmitted<br>• A packet larger than the maximum packet size was received | Host or device | BEMPSTS.PIPEnBEMP |
| NRDY | Buffer not ready interrupt | In host controller mode<br>• A STALL response was received from the peripheral device in response to the issued token<br>• The response from the peripheral device in response to the issued token was not received successfully (no response three times consecutively or packet reception error three times consecutively)<br>• An overrun or underrun error occurred during isochronous transfer<br>In device controller mode<br>• NAK was returned for an IN or OUT token while the PID[1:0] bits were set to 01b (BUF)<br>• A CRC error or bit stuffing error occurred during data reception in isochronous transfer<br>• An overrun or underrun occurred during data reception in isochronous transfer | Host or device | NRDYSTS.PIPEnNRDY |
| BRDY | Buffer ready interrupt | • The buffer is ready (readable or writable state) | Host or device | BRDYSTS.PIPEnBRDY |
| OVRCR | Overcurrent input change interrupt | • USB_OVRCURA or USB_OVRCURB input pin state change was detected (low to high or high to low) | Host | INTSTS1.OVRCR |
| BCHG | Bus change interrupt | • USB bus state change was detected | Host or device | SYSSTS0.LNST[1:0] |
| DTCH | Disconnect detection during full-speed operation | Peripheral device disconnect was detected in full-speed operation | Host | DVSTCTR0.RHST[2:0] |
| ATTCH | Device connect detection interrupt | • J-state or K-state was detected on the USB bus for 2.5 µs continuously<br>This interrupt can be used to check whether peripheral devices are connected | Host | — |

**Table 26.16 Interrupt sources (2 of 2)**

| Bit to be set to 1 | Name | Interrupt source | Applicable controller function | Status flag |
|---|---|---|---|---|
| EOFERR | EOF error detection interrupt | • An EOF error was detected for a peripheral device | Host | — |
| SACK | Setup normal interrupt | • A setup transaction normal response (ACK) was received | Host | — |
| SIGN | Setup error interrupt | • A setup transaction error (no response or ACK packet corruption) was detected three consecutive times | Host | — |
| PDDETINT | Portable Device detection interrupt | • Portable Device connection was detected | Host | BCCTRL1.PDDETSTS |

Note 1. Although this interrupt can be generated in host controller mode, it is not usually used in this mode.

Figure 26.11 shows the circuits related to the USBFS interrupts.

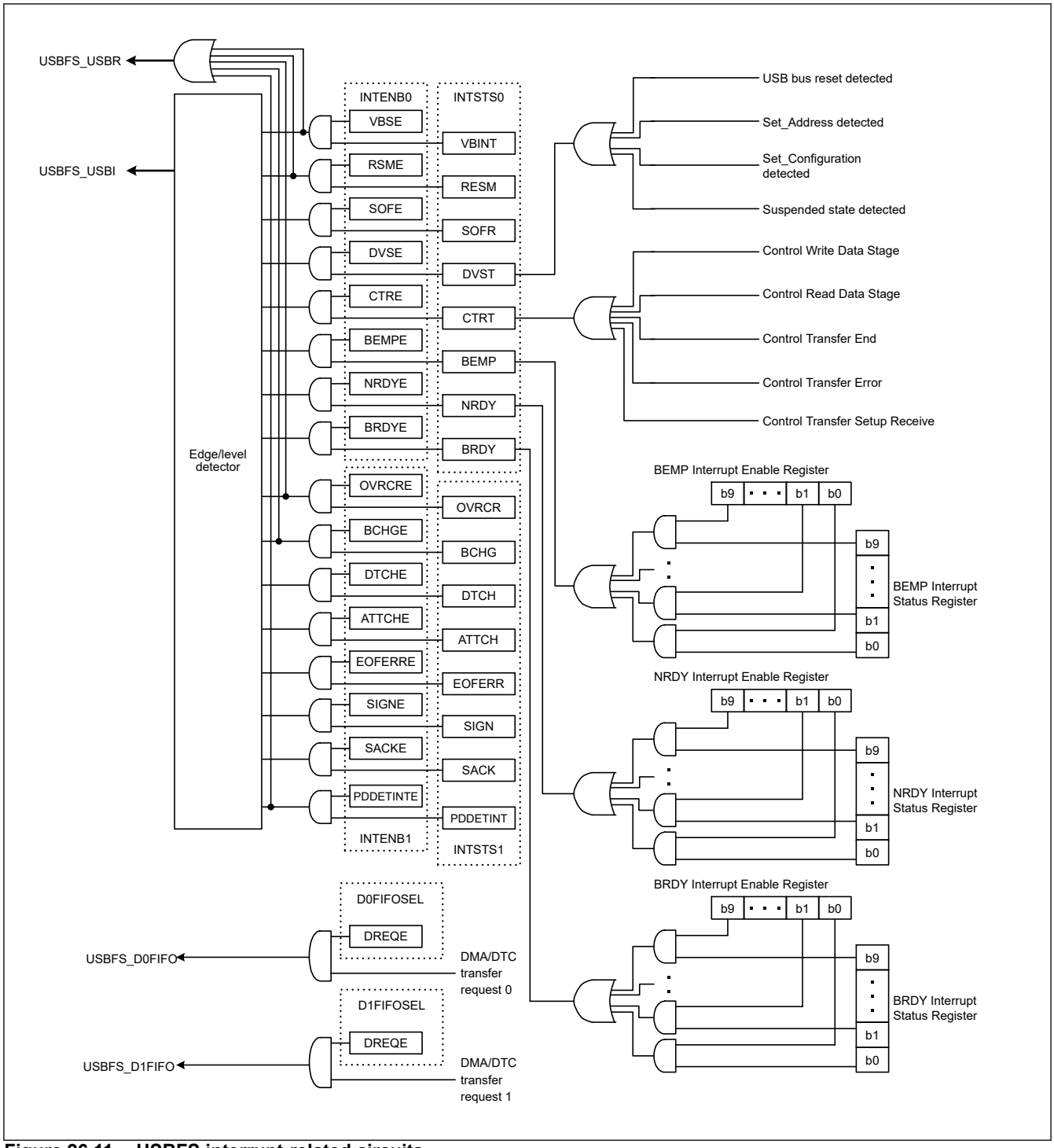



**Figure 26.11 USBFS interrupt-related circuits**

Table 26.17 shows the interrupts generated by the USBFS.

**Table 26.17 USBFS interrupts**

| Interrupt name | Interrupt status flag | DTC activation | DMAC activation | Priority |
|---|---|---|---|---|
| USBFS_D0FIFO | DMA transfer request 0 | Possible | Possible | High |
| USBFS_D1FIFO | DMA transfer request 1 | Possible | Possible | ↑ |
| USBFS_USBI | VBUS interrupt, resume interrupt, frame number update interrupt, device state transition interrupt, control transfer stage transition interrupt, buffer empty interrupt, buffer not ready interrupt, buffer ready interrupt, overcurrent input change interrupt, bus change interrupt, disconnect detection interrupt during full-speed operation, device connect detection interrupt, EOF error detection interrupt, normal setup operation interrupt, setup error interrupt, and Portable Device detection interrupt | Not possible | Not possible | Low |
| USBFS_USBR | VBUS interrupt, resume interrupt, overcurrent input change interrupt, bus change interrupt, and Portable Device detection interrupt | Not possible | Not possible | — |

## 26.3.3 Interrupt Descriptions

### 26.3.3.1 BRDY interrupt

The BRDY interrupt is generated in both host and device controller modes. This section describes the conditions in which the USBFS sets the associated bit in BRDYSTS to 1. Under these conditions, the USBFS generates a BRDY interrupt if the software has set the bit in BRDYENB associated with the given pipe to 1 and the INTENB0.BRDYE bit to 1.

The conditions for generating and clearing the BRDY interrupt depend on the SOFCFG.BRDYM and PIPECFG.BFRE settings for each pipe as follows:

#### (1) When SOFCFG.BRDYM = 0 and PIPECFG.BFRE = 0

With these settings, the BRDY interrupt indicates that the FIFO port is accessible.

On any of the following conditions, the USBFS generates an internal BRDY interrupt request trigger and sets the BRDYSTS.PIPEnBRDY bit associated with the selected pipe to 1.

**For transmitting pipes**

- When the DIR bit is changed from 0 to 1 by software
- When packet transmission is complete for a pipe while write-access from the CPU to the FIFO buffer for the pipe is disabled (when the BSTS bit is read as 0)
- When one FIFO buffer is empty on completion of writing data to the other FIFO buffer in double buffer mode
- No request trigger is generated until completion of writing data to the currently-written FIFO buffer even if transmission to the other FIFO buffer is complete
- When the hardware flushes the buffer of the pipe for isochronous transfers
- When 1 is written to the PIPEnCTR.ACLRM bit, which causes the FIFO buffer to transition from the write-disabled to write-enabled state

No request trigger is generated for the DCP, that is, during data transmission for control transfers.

**For receiving pipes**

- When packet reception is successfully complete, enabling the FIFO buffer to be read while read-access from the CPU to the FIFO buffer for the given pipe is disabled (when the BSTS bit is read as 0). No request trigger is generated for transactions in which a DATA-PID mismatch has occurred.
- When one FIFO buffer is read-enabled on completion of reading data from the other FIFO buffer in double buffer mode. No request trigger is generated until completion of reading data from the currently-read FIFO buffer, even if reception by the other FIFO buffer is complete.

In device controller mode, the BRDY interrupt is not generated in the status stage of control transfers. The PIPEBRDY interrupt status of the selected pipe can be set to 0 by writing 0 to the associated PIPEnBRDY bit through software. In this case, the other PIPEBRDY bit should be set to 1.

Clear the BRDY status before accessing the FIFO buffer.

### (2) When SOFCFG.BRDYM = 0 and PIPECFG.BFRE = 1

With these settings, the USBFS generates a BRDY interrupt on completion of reading all data for a single transfer using the receiving pipe, and sets the bit in BRDYSTS associated with the pipe to 1.

On any of the following conditions, the USBFS determines that the last data for a single transfer was received.

- When a short packet including a zero-length packet is received
- When the PIPEn transaction counter register (PIPEnTRN) is used and the number of packets specified in the PIPEnTRN.TRNCNT[15:0] bits are completely received

When the data is completely read after any of these conditions is satisfied, the USBFS determines that all data for a single transfer is completely read.

When a zero-length packet is received while the FIFO buffer is empty, the USBFS determines that all data for a single transfer is completely read when the FRDY bit in the FIFO port control register is 1 and the DTLN[8:0] bits are 0. In this case, to start the next transfer, write 1 to the BCLR bit in the associated port control register through the software. With these settings, the USBFS does not detect a BRDY interrupt for the transmitting pipe.

The PIPEBRDY interrupt status of a pipe can be set to 0 by writing 0 to the associated BRDYSTS.PIPEnBRDY bit through the software. In this case, 1s must be written to the PIPEBRDY bits for the other pipes.

In this mode, do not change the PIPECFG.BFRE bit setting until all data for a single transfer is processed. When it is necessary to change the PIPECFG.BFRE bit before completion of processing, all FIFO buffers for the pipe must be cleared using the PIPEnCTR.ACLRM bit.

### (3) When SOFCFG.BRDYM = 1 and PIPECFG.BFRE = 0

With these settings, the BRDYSTS.PIPEnBRDY values are linked to the BSTS bit setting for each pipe. In other words, the BRDY interrupt status bits (PIPEBRDY) are set to 1 or 0 by the USB depending on the FIFO buffer status.

#### For transmitting pipes

The BRDY interrupt status bits are set to 1 when the FIFO buffer is ready for write access, and are set to 0 when it is not ready. The BRDY interrupt is not generated for the DCP in the transmitting direction even when it is ready for write access.

#### For receiving pipes

The BRDY interrupt status bits set to 1 when the FIFO buffer is ready for read access, and set to 0 when all data is read (not ready for read access).

When a zero-length packet is received while the FIFO buffer is empty, the associated bit is set to 1 and the BRDY interrupt is continuously generated until the software writes 1 to BCLR. With this setting, the PIPEnBRDY bit cannot be set to 0 by software.

When the SOFCFG.BRDYM bit is set to 1, set the PIPECFG.BFRE bit for all pipes to 0.

Figure 26.12 shows the timing of BRDY interrupt generation.

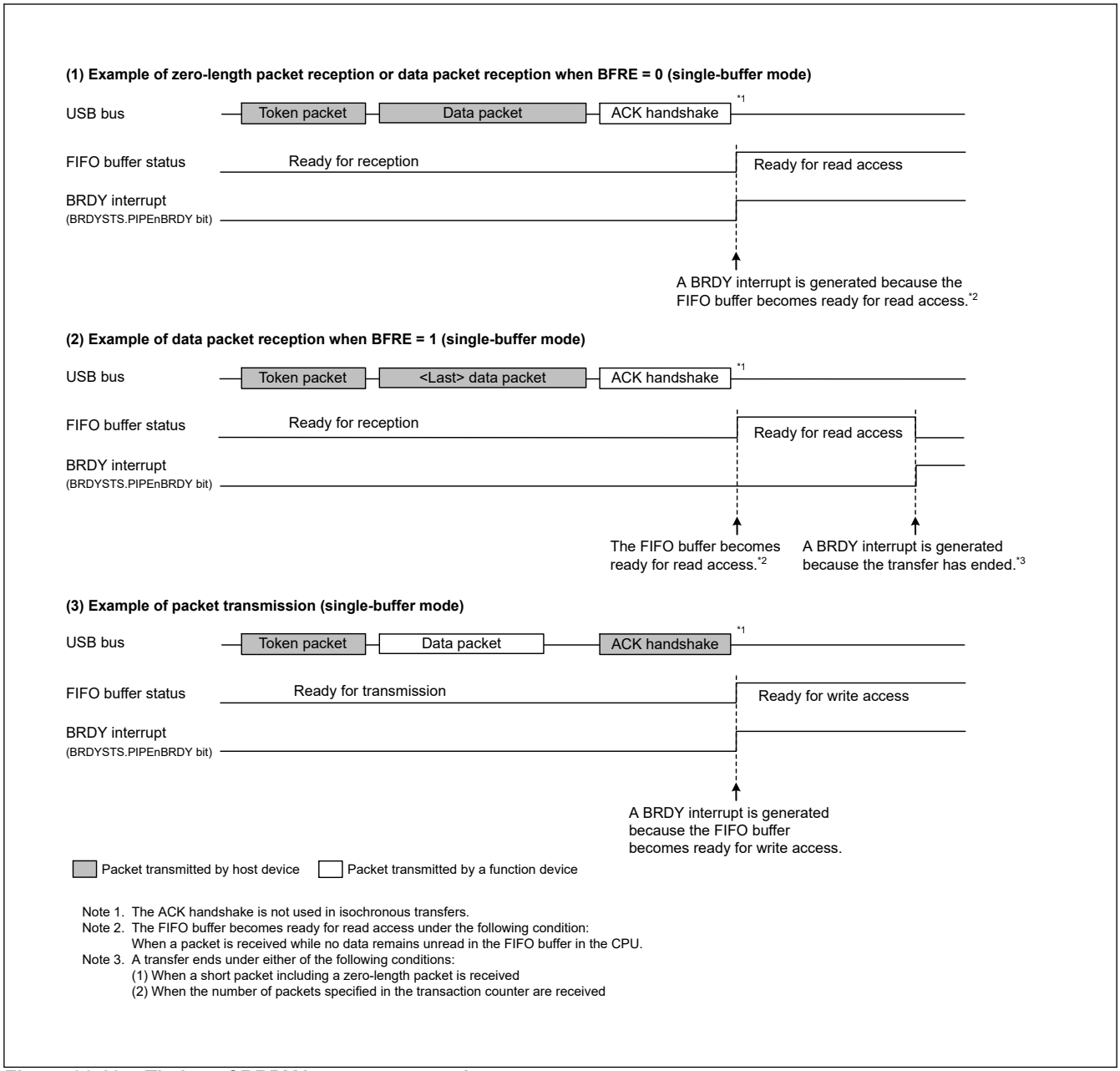



**Figure 26.12 Timing of BRDY interrupt generation**

The condition for clearing the INTSTS0.BRDY bit depends on the SOFCFG.BRDYM bit setting, as shown in Table 26.18.

**Table 26.18 Conditions for clearing the BRDY bit**

| BRDYM bit | Condition for clearing BRDY bit |
|---|---|
| 0 | The USBFS clears the BRDY bit to 0 when all bits in BRDYSTS are set to 0 by software. |
| 1 | The USBFS clears the BRDY bit to 0 when the BSTS bits for all pipes have cleared to 0. |

### 26.3.3.2 NRDY interrupt

On generating an internal NRDY interrupt request for the pipe whose PID bits are set to BUF by software, the USBFS sets the associated PIPEnNRDY bit in NRDYSTS to 1. If the associated bit in NRDYENB is set to 1 by software, the USBFS sets the INTSTS0.NRDY bit to 1 and generates a USBFS interrupt.

This section describes the conditions in which the USBFS generates the internal NRDY interrupt request for a given pipe.

The internal NRDY interrupt request is not generated during setup transaction execution in host controller mode. During setup transactions in host controller mode, the SACK or SIGN interrupt is detected.

The internal NRDY interrupt request is not generated during status stage execution of the control transfer in device controller mode.

### (1) In host controller mode

#### For transmitting pipes

On any of the following conditions, the USBFS detects an NRDY interrupt:

- For isochronous transfer pipes, when the time to issue an OUT token comes while there is no data to be transmitted in the FIFO buffer. In this case, the USBFS transmits a zero-length packet following the OUT token and sets the associated NRDYSTS.PIPEnNRDY bit and the FRMNUM.OVRN bit to 1.
- During communications other than setup transactions on pipes not used for isochronous transfers, when any combination of the following two cases occur three consecutive times:
  - No response is returned from the peripheral device (when timeout is detected before detection of the handshake packet from the peripheral device
  - An error is detected in the packet from the peripheral device. In this case, the USBFS sets the associated PIPEnNRDY bit to 1 and changes the associated PID[1:0] setting for the pipe to NAK
- During communications other than setup transactions, when the STALL handshake is received from the peripheral device. In this case, the USBFS sets the associated PIPEnNRDY bit to 1 and changes the PID[1:0] setting for the associated pipe to STALL (11b).

#### For receiving pipes

- For isochronous transfer pipes, when the time to issue an IN token comes but there is no space available in the FIFO buffer. In this case, the USBFS discards the received data for the IN token and sets the PIPEnNRDY bit associated with the pipe and the OVRN bit to 1. When a packet error is detected in the received data for the IN token, the USBFS also sets the FRMNUM.CRCE bit to 1.
- For non-isochronous transfer pipes, when any combination of the following two cases occur three consecutive times:
  - No response is returned from the peripheral device for the IN token issued by the USBFS (when timeout is detected before detection of the DATA packet from the peripheral device)
  - An error is detected in the packet from the peripheral device. In this case, the USBFS sets the associated PIPEnNRDY bit to 1 and changes the associated PID[1:0] setting for the pipe to NAK
- For isochronous transfer pipes, when no response is returned from the peripheral device for the IN token (when timeout is detected before detection of the DATA packet from the peripheral device) or an error is detected in the packet from the peripheral device. In this case, the USBFS sets the PIPEnNRDY bit associated with the pipe to 1. The PID[1:0] setting for the pipe is not changed.
- For isochronous transfer pipes, when a CRC error or a bit stuffing error is detected in the received data packet. In this case, the USBFS sets the PIPEnNRDY bit associated with the pipe and the CRCE bit to 1.
- When the STALL handshake is received. In this case, the USBFS sets the PIPEnNRDY bit associated with the pipe to 1 and changes the PID[1:0] setting for the associated pipe to STALL.

### (2) In device controller mode

#### For transmitting pipes

- When an IN token is received while there is no data to be transmitted in the FIFO buffer. In this case, the USBFS generates a NRDY interrupt request on reception of the IN token and sets the NRDYSTS.PIPEnNRDY bit to 1. For an isochronous transfer pipe in which an interrupt is generated, the USBFS transmits a zero-length packet and sets the FRMNUM.OVRN bit to 1.

#### For receiving pipes

- When an OUT token is received but there is no space available in the FIFO buffer. For an isochronous transfer pipe in which an interrupt is generated, the USBFS generates a NRDY interrupt request on reception of the OUT token and sets the PIPEnNRDY bit to 1 and OVRN bit to 1. For a non-isochronous transfer pipe in which an interrupt is generated, the USBFS generates a NRDY interrupt request when a NAK handshake is transferred after the data following the OUT token is received, and sets the PIPEnNRDY bit to 1. The NRDY interrupt request is not generated during retransmission

because of a DATA-PID mismatch. In addition, the NRDY interrupt request is not generated if an error occurs in the DATA packet.

- For isochronous transfer pipes, when a token is not received successfully within an interval frame. In this case, the USBFS generates an NRDY interrupt request when the SOF is received, and sets the PIPEnNRDY bit to 1.

Figure 26.13 shows the timing of NRDY interrupt generation in device controller mode.

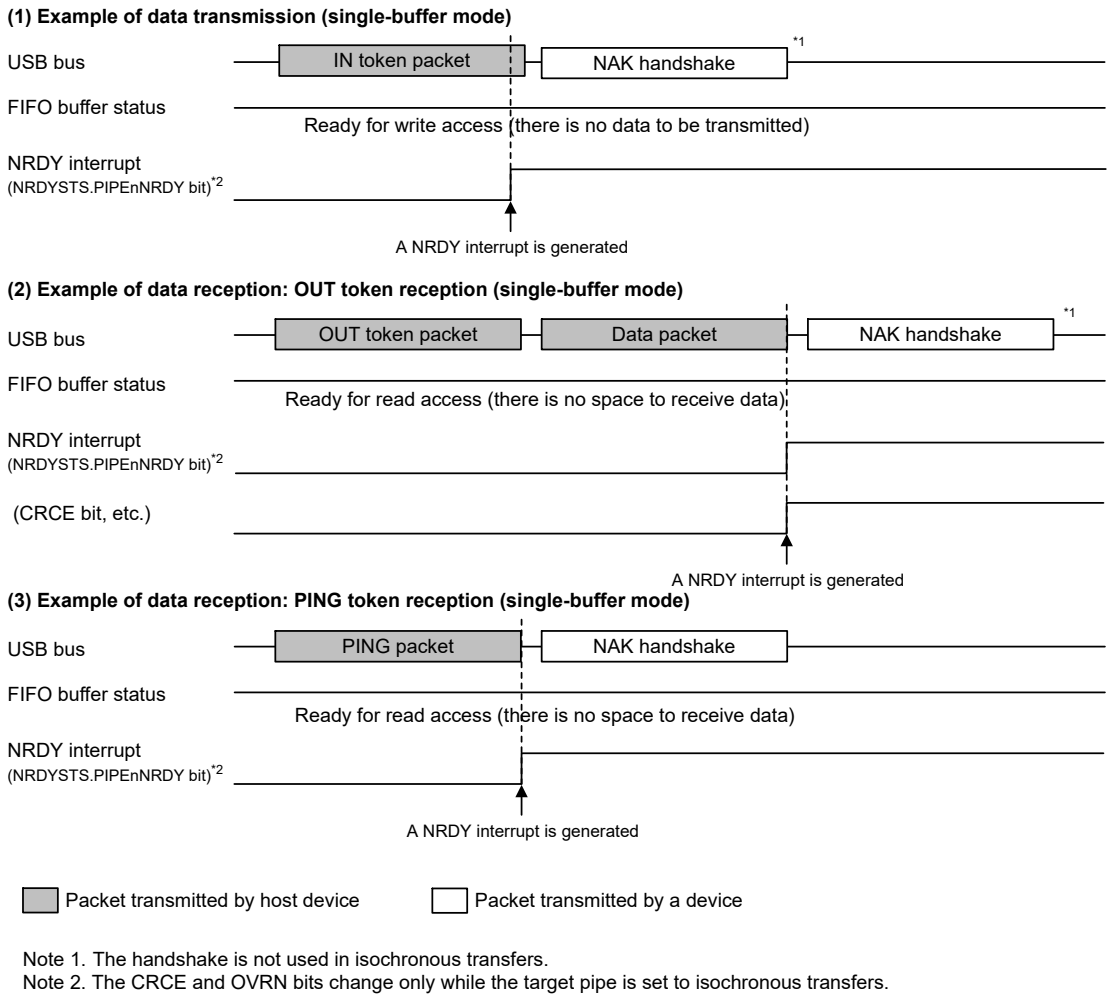



**Figure 26.13 Timing of NRDY interrupt generation in device controller mode**

### 26.3.3.3 BEMP interrupt

On detecting a BEMP interrupt for the pipe whose PID bits are set to BUF by software, the USBFS sets the associated BEMPSTS.PIPEnBEMP bit to 1. If the associated bit in BEMPENB is set to 1 by software, the USBFS sets the INTSTS0.BEMP bit to 1 and generates a USBFS interrupt. This section describes the conditions in which the USBFS generates an internal BEMP interrupt request.

#### (1) For transmitting pipes

When the FIFO buffer of the associated pipe is empty on completion of transmission, including zero-length packet transmission, and in single buffer mode, an internal BEMP interrupt request is generated simultaneously with the BRDY interrupt for a non-DCP pipe. The internal BEMP interrupt request is not generated in any of the following conditions:

- When the CPU or DMA/DTC has already started writing data to the FIFO buffer of the CPU on completion of transmitting data from one FIFO buffer in double buffer mode
- When the buffer is cleared (emptied) by setting the PIPEnCTR.ACLRM or the BCLR bit to 1 in the port control register
- When an IN transfer (zero-length packet transmission) is performed during the control transfer status stage in device controller mode

### (2) For receiving pipes

When a successfully-received data packet size exceeds the specified maximum packet size. In this case, the USBFS generates a BEMP interrupt request, sets the associated BEMPSTS.PIPEnBEMP bit to 1, discards the received data, and changes the associated PID[1:0] setting for the pipe to STALL (11b). The USBFS returns no response in host controller mode, and returns STALL response in device controller mode.

The internal BEMP interrupt request is not generated in any of the following conditions:

- When a CRC error or a bit stuffing error is detected in the received data
- When a setup transaction is being performed:
  - Writing 0 to the BEMPSTS.PIPEnBEMP bit clears the status
  - Writing 1 to the BEMPSTS.PIPEnBEMP bit has no effect

Figure 26.14 shows the timing of BEMP interrupt generation in device controller mode.

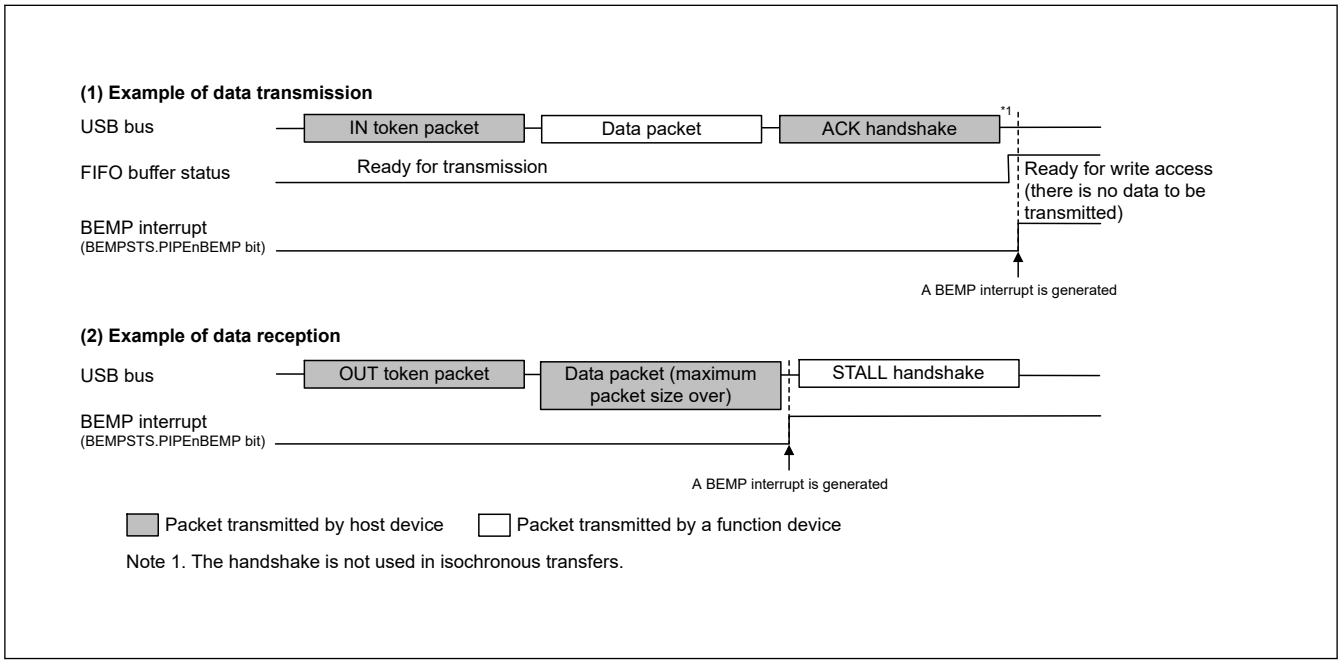



**Figure 26.14 Timing of BEMP interrupt generation in device controller mode**

## 26.3.3.4 Device state transition interrupt (device controller mode)

Figure 26.15 shows a diagram of the USBFS device state transitions. The USBFS controls device states and generates device state transition interrupts. However, recovery from the Suspend state (resume signal detection) is detected by means of the resume interrupt. Device state transition interrupts can be enabled or disabled independently in INTENB0. Devices whose states have changed can be checked in the INTSTS0.DVSQ[2:0] bits.

When a transition is made to the default state, a device state transition interrupt is generated after a USB bus reset is detected.

The USBFS controls device states, and device state transition interrupts can be generated, only in device controller mode.

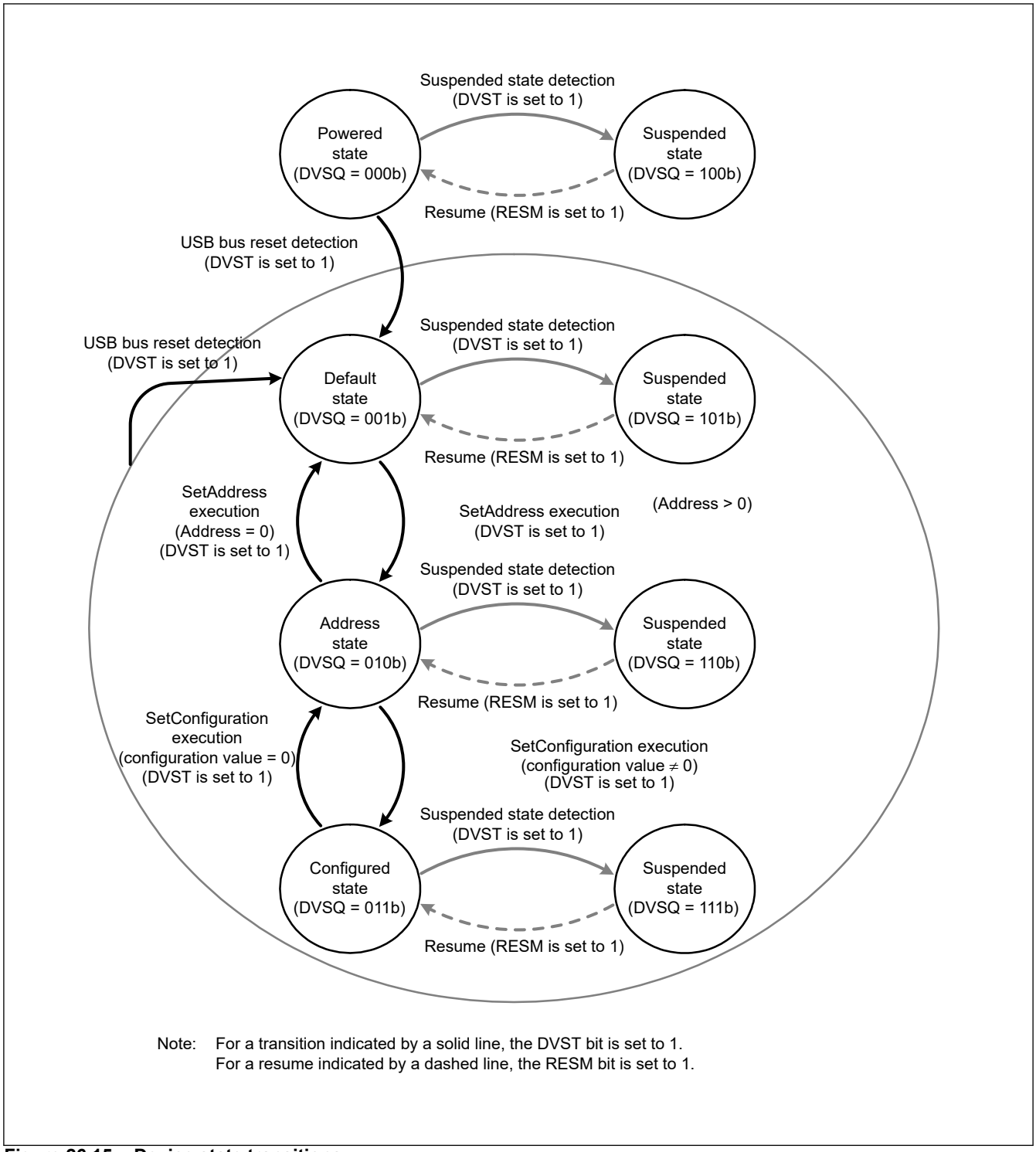



**Figure 26.15 Device state transitions**

### 26.3.3.5 Control transfer stage transition interrupt (device controller mode)

Figure 26.16 shows a diagram of the control transfer stage transitions of the USBFS. The USBFS controls the control transfer sequence and generates control transfer stage transition interrupts. Control transfer stage transition interrupts can be enabled or disabled independently in INTENB0. Transfer stages that have transitioned can be checked in the INTSTS0.CTSQ[2:0] bits.

Control transfer stage transition interrupts are generated only in device controller mode. This section describes control transfer sequence errors. If an error occurs, the DCPCTR.PID[1:0] bits are set to 1xb (STALL response).

### (1) Control read transfer errors

- An OUT token is received but no data is transferred in response to the IN token at the data stage
- An IN token is received at the status stage
- A data packet with DATAPID = DATA0 is received at the status stage

### (2) Control write transfer errors

- An IN token is received but no ACK is returned in response to the OUT token at the data stage
- A data packet with DATAPID = DATA0 is received as the first data packet at the data stage
- An OUT token is received at the status stage

### (3) Control write no data transfer errors

- An OUT token is received at the status stage

At the control write transfer data stage, if the receive data length exceeds the wLength value of the USB request, it is not recognized as a control transfer sequence error. At the control read transfer status stage, packets other than zero-length packets are received by an ACK response and the transfer ends normally.

When a CTRT interrupt occurs in response to a sequence error (INTSTS0.CTRT = 1), the CTSQ[2:0] = 110b value is saved until the CTRT bit is set to 0, clearing the interrupt status. While CTSQ[2:0] = 110b is being saved, no CTRT interrupt for ending the setup stage is generated, even if a new USB request is received. The USBFS saves the setup stage completion status, and it generates a CTRT interrupt after the interrupt status is cleared by software.

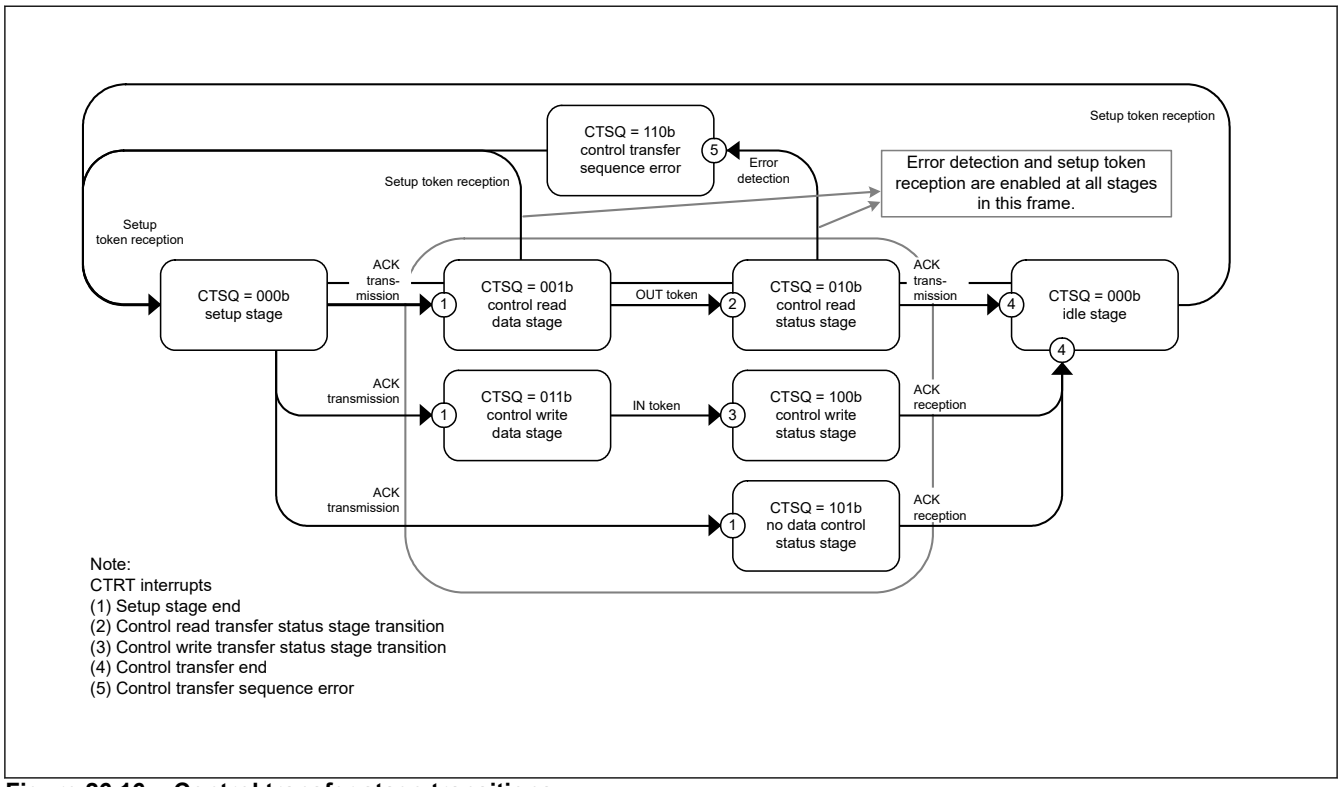



**Figure 26.16 Control transfer stage transitions**

## 26.3.3.6 Frame update interrupt

In host controller mode, an interrupt is generated when the frame number is updated.

In device controller mode, an SOFR interrupt is generated when the frame number is updated. The USBFS updates the frame number and generates an SOFR interrupt if it detects a new SOF packet during full-speed operation.

#### 26.3.3.7 VBUS interrupt

When the USB_VBUS pin level changes, a VBUS interrupt is generated. The level of the USB_VBUS pin can be checked with the INTSTS0.VBSTS bit. Whether the host controller is connected or disconnected can be confirmed using the VBUS interrupt. If the system is activated with the host controller connected, the first VBUS interrupt is not generated, because there is no change in the USB_VBUS pin level.

#### 26.3.3.8 Resume interrupt

In device controller mode, a resume interrupt is generated when the device state is the Suspend state and the USB bus state has changed (from J-state to K-state, or from J-state to SE0). Recovery from the Suspend state is detected by means of the resume interrupt.

In host controller mode, no resume interrupt is generated. Use the BCHG interrupt to detect a change in the USB bus state.

#### 26.3.3.9 OVRCR interrupt

An OVRCR interrupt is generated when the USB_OVRCURA or USB_OVRCURB pin level has changed. The levels of the USB_OVRCURA and USB_OVRCURB pins can be checked in the SYSSTS0.OVCMON[1:0] flags. The external power supply IC can check whether overcurrent is detected using the OVRCR interrupt.

For OTG connections, the OVRCR interrupt allows you to check whether a change is detected in the VBUS comparator.

#### 26.3.3.10 BCHG interrupt

A BCHG interrupt is generated when the USB bus state has changed. The BCHG interrupt can be used to detect whether a peripheral device is connected and can also be used to detect a remote wakeup in host controller mode. The BCHG interrupt is generated in both host and device controller modes.

#### 26.3.3.11 DTCH interrupt

A DTCH interrupt occurs when a USB bus disconnect is detected in host controller mode. The USBFS detects bus disconnects in compliance with the USB 2.0 specification.

On interrupt detection, all pipes in which communications are being carried out for the relevant port must be terminated by software. The pipes enter the wait state for a bus connection to the port, waiting for an ATTCH interrupt to occur. Regardless of the value set in the associated interrupt enable bit, the USBFS hardware:

- Sets the DVSTCTR0.UACT bit for the port in which the DTCH interrupt is detected to 0
- Puts the port in which the DTCH interrupt occurred into the idle state

#### 26.3.3.12 SACK interrupt

A SACK interrupt is generated when an ACK response for the transmitted setup packet is received from the peripheral device in host controller mode. The SACK interrupt can be used to confirm that the setup transaction is successfully complete.

#### 26.3.3.13 SIGN interrupt

A SIGN interrupt is generated when an ACK response for the transmitted setup packet is not correctly received from the peripheral device three consecutive times in host controller mode. The SIGN interrupt can be used to detect no ACK response transmitted from the peripheral device or corruption of an ACK packet.

#### 26.3.3.14 ATTCH interrupt

An ATTCH interrupt is generated when J-state or K-state of the full-speed signal level is detected on the USB port for 2.5 µs in host controller mode. To be more specific, an ATTCH interrupt is detected on any of the following conditions:

- When K-state, SE0, or SE1 changes to J-state, and J-state continues 2.5 µs
- When J-state, SE0, or SE1 changes to K-state, and K-state continues 2.5 µs

#### 26.3.3.15 EOFERR interrupt

An EOFERR interrupt occurs when the USBFS detects that communication is not complete at the EOF2 timing defined in the USB 2.0 specification.

On interrupt detection, all pipes in which communications are being carried out for the relevant port must be terminated by software, and the port must be re-enumerated. Regardless of the value set in the associated interrupt enable bit, the USBFS hardware:

- Sets the DVSTCTR0.UACT bit for the port in which the EOFERR interrupt is detected to 0
- Puts the port in which the EOFERR interrupt is generated into the idle state

#### 26.3.3.16 Portable Device Detection Interrupt

The USBFS sets the INTSTS1.PDDETINT flag to 1 on detecting a portable device and generates the portable device detection interrupt. When the portable device detection interrupt is generated, use software to repeat reading the BCCTRL1.PDDETSTS flag until the same value is read three or more times, and perform debounce processing.

### 26.3.4 Pipe Control

Table 26.19 lists the pipe settings for the USBFS. USB data transfer is performed through logical pipes that the software associates with endpoints. The USBFS provides 10 pipes that are used for data transfer. Set up the pipes based on your system specifications.

**Table 26.19 Pipe settings (1 of 2)**

| Register name | Bit name | Setting | Notes |
|---|---|---|---|
| DCPCFG<br>PIPECFG | TYPE | Transfer type | Pipes 1 to 9: Settable |
| | BFRE | BRDY interrupt mode | Pipes 1 to 5: Settable |
| | DBLB | Double buffer select | Pipes 1 to 5: Settable |
| | DIR | Transfer direction select | IN or OUT settable |
| | EPNUM | Endpoint number | Pipes 1 to 9: Settable<br>A value other than 0000b must be set when the pipe is used. |
| | SHTNAK | Selects disabled state for pipe when transfer ends | Pipes 1 and 2: Settable only for bulk transfers<br>Pipes 3 to 5: Settable |
| DCPMAXP<br>PIPEMAXP | DEVSEL | Device select | Referenced only in host controller mode. |
| | MXPS | Maximum packet size | Compliant with the USB 2.0 specification. |
| PIPEPERI | IFIS | Buffer flush | Pipes 1 and 2: Settable only for isochronous transfers<br>Pipes 3 to 9: Setting disabled |
| | IITV | Interval counter | Pipes 1 and 2: Settable only for isochronous transfers<br>Pipes 3 to 5: Setting disabled<br>Pipes 6 to 9: Settable only in host controller mode |
| DCPCTR<br>PIPEnCTR | BSTS | Buffer status | For the DCP, receive buffer status and transmit buffer status are switched with the ISEL bit. |
| | INBUFM | IN buffer monitor | Available only for pipes 1 to 5. |
| | SUREQ | Setup request | Settable only for the DCP and controlled in host controller mode |
| | SUREQCLR | SUREQ clear | Settable only for the DCP and controlled in host controller mode |
| | ATREPM | Auto response mode | Pipes 1 to 5: Settable only in device controller mode |
| | ACLRM | Auto buffer clear | Pipes 1 to 9: Settable |
| | SQCLR | Sequence clear | Clears the data toggle bit |
| | SQSET | Sequence set | Sets the data toggle bit |
| | SQMON | Sequence monitor | Monitors the data toggle bit |
| | PBUSY | Pipe busy status | — |
| | PID | Response PID | See section 26.3.4.6. Response PID. |

**Table 26.19 Pipe settings (2 of 2)**

| Register name | Bit name | Setting | Notes |
|---|---|---|---|
| PIPEnTRE | TRENB | Transaction counter enable | Pipes 1 to 5: Settable |
| | TRCLR | Current transaction counter clear | Pipes 1 to 5: Settable |
| PIPEnTRN | TRNCNT | Transaction counter | Pipes 1 to 5: Settable |

### 26.3.4.1 Pipe control register switching procedures

The following bits in the pipe control registers can be changed only when USB communication is prohibited (PID = NAK).

Do not change the following registers and bits when USB communication is enabled (PID = BUF):

- Bits in DCPCFG and DCPMAXP
- SQCLR and SQSET bits in DCPCTR
- Bits in PIPECFG, PIPEMAXP, and PIPEPERI
- ATREPM, ACLRM, SQCLR, and SQSET bits in PIPEnCTR
- Bits in PIPEnTRE and PIPEnTRN

To set these bits when USB communication is enabled (PID = BUF):

1. A request to change the bits in the pipe control register occurs.
2. Set the PID[1:0] bits associated with the pipe to NAK.
3. Wait until the associated PBUSY bit clears to 0.
4. Set the bits in the pipe control register.

The following bits in the pipe control registers can be changed only when the selected pipe information is not set in the CURPIPE[3:0] bits in CFIFOSEL, D0FIFOSEL, and D1FIFOSEL.

Do not set the following registers when the CURPIPE[3:0] bits are set:

- Bits in DCPCFG and DCPMAXP
- Bits in PIPECFG, PIPEMAXP and PIPEPERI

To change pipe information, you must set the CURPIPE[3:0] bits in the port select registers to a pipe other than the one to be changed. For the DCP, the buffer must be cleared using the BCLR bit in the Port Control Register after the pipe information is changed.

### 26.3.4.2 Transfer types

The PIPECFG.TYPE[1:0] bits specify the following transfer types for each pipe:

- DCP: No setting is necessary (fixed at control transfer)
- Pipes 1 and 2: Set to bulk or isochronous transfer
- Pipes 3 to 5: Set to bulk transfer
- Pipes 6 to 9: Set to interrupt transfer

### 26.3.4.3 Endpoint number

The PIPECFG.EPNUM[3:0] bits are used to set the endpoint number for each pipe. The DCP is fixed at endpoint 0. The other pipes can be set from endpoint 1 to 15.

- DCP: No setting is necessary (fixed at endpoint 0)
- Pipes 1 to 9: Select and set the endpoint numbers from 1 to 15 so that the combination of the PIPECFG.DIR and EPNUM[3:0] bits is unique

#### 26.3.4.4 Maximum packet size setting

Specify the maximum packet size for each pipe in the DCPMAXP.MXPS[6:0] and PIPEMAXP.MXPS[9:0] bits. The DCP and pipes 1 to 5 can be set to any of the maximum pipe sizes defined in the USB 2.0 specification. For pipes 6 to 9, the maximum packet size is 64 bytes. Set the maximum packet size as follows before starting a transfer (PID = BUF):

- DCP: Set to 8, 16, 32, or 64
- Pipes 1 to 5: Set to 8, 16, 32, or 64 for bulk transfers
- Pipes 1 and 2: Set between 1 and 256 for isochronous transfers
- Pipes 6 to 9: Set between 1 and 64

#### 26.3.4.5 Transaction counter for pipes 1 to 5 in the receiving direction

When the specified number of transactions is complete in the data packet receiving direction, the USBFS recognizes that the transfer ended. Two transaction counters are provided: one is the PIPEnTRN register, which specifies the number of transactions to be executed, and the other is the current counter, which internally counts the number of executed transactions. If the PIPECFG.SHTNAK bit is set to 1, when the current counter value matches the specified number of transactions, the associated PIPEnCTR.PID[1:0] bits are set to NAK and the subsequent transfer is disabled. The transactions can be counted again from the beginning by initializing the current counter of the transaction counter function through the PIPEnTRE.TRCLR bit. The data read from PIPEnTRN differs depending on the PIPEnTRE.TRENB setting as follows:

- The TRENB bit = 0: Specified transaction counter value can be read
- The TRENB bit = 1: Current counter value indicating the internally counted number of executed transactions can be read

The following constraints apply when working with the TRCLR bit:

- If the transactions are being counted and PID = BUF, the current counter cannot be cleared
- If there is any data left in the buffer, the current counter cannot be cleared

#### 26.3.4.6 Response PID

Specify the response PID for each pipe in the PID[1:0] bits in DCPCTR and PIPEnCTR. This section describes the USBFS operation with different response PID settings.

##### (1) Software response PID settings in host controller mode

Select the response PID to specify the execution of transactions as follows:

- NAK setting: Using pipes is disabled and no transactions are executed
- BUF setting: Transactions are executed based on the FIFO buffer state:
  - OUT direction: An OUT token is issued if the FIFO buffer contains transmit data.
  - IN direction: An IN token is issued if the FIFO buffer is not full and can receive data.
- STALL setting: Using pipes is disabled and no transactions are executed

Note: Use the DCPCTR.SUREQ bit to execute setup transactions for the DCP.

##### (2) Software response PID settings in device controller mode

Select the response PID to respond as follows to transactions from the host:

- NAK setting: A NAK response is returned to all generated transactions
- BUF setting: A response is returned to transactions based on the FIFO buffer
- STALL setting: A STALL response is returned to all generated transactions

Note: For setup transactions, an ACK response is always returned, regardless of the PID[1:0] bits setting, and the USB request is stored in the register.

Sections (3) and (4) describe situations in which the USBFS writes to the PID[1:0] bits because of specific transaction results.

#### (3) Hardware response PID settings in host controller mode

- NAK setting: PID = NAK is set in the following cases, and issuing of tokens is automatically stopped:
  - When a non-isochronous transfer is performed and an NRDY interrupt is generated

    (For details, see section 26.3.3.2. NRDY interrupt.)
  - If a short packet is received when the PIPECFG.SHTNAK bit is set to 1 for bulk transfers
  - If transaction counting ends when the SHTNAK bit is set to 1 for bulk transfers
- BUF setting: The USBFS does not write this setting.
- STALL setting: PID = STALL is set in the following cases, and issuing of tokens is automatically stopped:
  - When STALL is received in response to a transmitted token
  - When a received data packet exceeds the maximum packet size

#### (4) Hardware response PID settings in device controller mode

- NAK setting: PID = NAK is set in the following cases, and a NAK response is returned to transactions:
  - When the setup token is received normally (DCP only)
  - If transaction counting ends or a short packet is received when the PIPECFG.SHTNAK bit is set to 1 for bulk transfers
- BUF setting: There is no BUF writing by the USBFS.
- STALL setting: PID = STALL is set in the following cases, and a STALL response is returned to transactions:
  - When a received data packet exceeds the maximum packet size
  - When a control transfer sequence error is detected (DCP only)

### 26.3.4.7 Data PID sequence bit

The USBFS automatically toggles the sequence bit in the data PID when data is transferred successfully in the control transfer data stage, bulk transfer, and interrupt transfer. The sequence bit of the next data PID to be transmitted can be confirmed with the SQMON bit in DCPCTR and PIPEnCTR. When data is transmitted, the sequence bit toggles on ACK handshake reception. When data is received, the sequence bit toggles on ACK handshake transmission. The SQCLR bit in DCPCTR and the SQSET bit in PIPEnCTR can be used to change the data PID sequence bit.

In device controller mode when control transfers are used, the USBFS automatically sets the sequence bit for stage transitions. DATA1 is returned when the setup stage ends. The sequence bit is not referenced and PID = DATA1 is returned in the status stage. Therefore, no software settings are required. However, in host controller mode when control transfers are used, the sequence bit must be set by software for the stage transitions.

For ClearFeature requests for transmission or reception, the data PID sequence bit must be set by software in both host and device controller modes.

### 26.3.4.8 Response PID = NAK function

The USBFS provides a function for disabling pipe operation (PID response = NAK) when the final data packet of a transaction is received. The USBFS automatically distinguishes this based on reception of a short packet or the transaction counter. Enable this function by setting the PIPECFG.SHTNAK bit to 1.

When the double buffer mode is being used for the FIFO buffer, using this function enables reception of data packets in transfer units. If pipe operation is disabled, the software must enable the pipe again (PID response = BUF).

The response PID = NAK function can be used only for bulk transfers.

#### 26.3.4.9 Auto response mode

For bulk transfer pipes (1 to 5), when the PIPEnCTR.ATREPM bit is set to 1, a transition is made to auto response mode. During an OUT transfer (PIPECFG.DIR = 0), OUT-NAK mode is invoked, and during an IN transfer (DIR = 1), null auto response mode is invoked.

#### 26.3.4.10 OUT-NAK mode

For bulk OUT transfer pipes, NAK is returned in response to an OUT token, and an NRDY interrupt is output when the PIPEnCTR.ATREPM bit is set to 1. To transition from normal mode to OUT-NAK mode, specify OUT-NAK mode while pipe operation is disabled (PID[1:0] = 00b for NAK response). Next enable pipe operation (PID[1:0] = 01b for BUF response), on which OUT-NAK mode becomes valid. If an OUT token is received immediately before pipe operation is disabled, the token data is normally received, and an ACK is returned to the host.

To transition from OUT-NAK mode to normal mode, cancel OUT-NAK mode while pipe operation is disabled (NAK). Next enable pipe operation (BUF). In normal mode, reception of OUT data is enabled.

#### 26.3.4.11 Null auto response mode

For bulk IN transfer pipes, zero-length packets are continuously transmitted when the PIPEnCTR.ATREPM bit is set to 1.

To transition from normal mode to null auto response mode, specify null auto response mode while pipe operation is disabled (response PID = NAK). Next enable pipe operation (response PID = BUF) on which null auto response mode becomes valid. Before setting null auto response mode, check that PIPEnCTR.INBUFM = 0, because the mode can be set only when the buffer is empty. If the INBUFM bit is 1, empty the buffer using the PIPEnCTR.ACLRM bit. Do not write data from the FIFO port while a transition to null auto response mode is being made.

To transition from null auto response mode to normal mode, keep pipe operation disabled (response PID = NAK) for the period of the zero-length packet transmission (about 10 µs) before canceling the null auto response mode. In normal mode, data can be written from the FIFO port, so packet transmission to the host is enabled by enabling pipe operation (response PID = BUF).

### 26.3.5 FIFO Buffer

The USBFS provides a FIFO buffer for data transfers, and it manages the memory area used for each pipe. The FIFO buffer has two states depending on whether the access right is assigned to the system (CPU side) or the USBFS (SIE side).

#### (1) Buffer status

Table 26.20 and Table 26.21 show the buffer status in the USBFS. The FIFO buffer status can be confirmed using the DCPCTR.BSTS and PIPEnCTR.INBUFM bits. The transfer direction for the FIFO buffer can be specified in either the PIPECFG.DIR or CFIFOSEL.ISEL bit (when DCP is selected).

The INBUFM bit is valid for pipes 0 to 5 in the transmitting direction.

When a transmitting pipe uses double buffering, the software can read the BSTS bit to monitor the FIFO buffer status on the CPU side and the INBUFM bit to monitor the FIFO buffer status on the SIE side. When write access to the FIFO port by the CPU or DMA/DTC is slow and the buffer empty status cannot be determined using the BEMP interrupt, the software can use the INBUFM bit to confirm the end of transmission.

**Table 26.20 Buffer status indicated in the BSTS bit**

| ISEL or DIR | BSTS | FIFO buffer status |
|---|---|---|
| 0 (receiving direction) | 0 | There is no received data, or data is being received.<br>Reading from the FIFO port is disabled. |
| 0 (receiving direction) | 1 | There is received data, or a zero-length packet is received.<br>Reading from the FIFO port is allowed.<br>When a zero-length packet is received, reading is not possible and the buffer must be cleared. |
| 1 (transmitting direction) | 0 | Transmission has not completed.<br>Writing to the FIFO port is disabled. |
| 1 (transmitting direction) | 1 | Transmission is complete.<br>CPU write is allowed. |

**Table 26.21 Buffer status indicated in the INBUFM bit**

| DIR | INBUFM | FIFO buffer status |
|---|---|---|
| 0 (receiving direction) | Invalid | Invalid |
| 1 (transmitting direction) | 0 | Transmission is complete.<br>There is no data waiting to be transmitted. |
| 1 (transmitting direction) | 1 | The FIFO port has written data to the buffer.<br>There is data to be transmitted. |

## 26.3.6 FIFO Buffer Clearing

Table 26.22 shows the methods for clearing the FIFO buffer. The FIFO buffer can be cleared using BCLR bit in the port control register, DnFIFOSEL.DCLRM, or the PIPEnCTR.ACLRM bit.

Single or double buffering can be selected for pipes 1 to 5 in the PIPECFG.DBLB bit.

**Table 26.22 Buffer clearing methods**

| FIFO buffer clearing mode | Clearing FIFO buffer on the CPU side | Mode for automatically clearing the FIFO buffer after reading the specified pipe data | Auto buffer clear mode for discarding all received packets |
|---|---|---|---|
| Register used | CFIFOCTR<br>DnFIFOCTR | DnFIFOSEL | PIPEnCTR |
| Bit used | BCLR | DCLRM | ACLRM |
| Clearing condition | Cleared by writing 1 | 1: Mode valid<br>0: Mode invalid | 1: Mode valid<br>0: Mode invalid |

### (1) Auto buffer clear mode function

The USBFS discards all received data packets if the PIPEnCTR.ACLRM bit is set to 1. If a correct data packet is received, the ACK response is returned to the host controller. The auto buffer clear mode function can only be set in the FIFO buffer reading direction.

Setting the ACLRM bit to 1 and then to 0 clears the FIFO buffer of the selected pipe regardless of the access direction. An access cycle of at least 100 ns is required for the internal hardware sequence processing between ACLRM = 1 and ACLRM = 0.

## 26.3.7 FIFO Port Functions

Table 26.23 shows the settings for the FIFO port functions. In write access, writing data until the maximum packet size is reached automatically enables transmission of the data. To enable transmission before the maximum packet size is reached, set the BVAL flag in the port control register to end writing. To send a zero-length packet, use the BCLR bit to clear the buffer, and then set the BVAL flag to end writing.

In reading, reception of new packets is automatically enabled when all data is read. Data cannot be read when a zero-length packet is received (DTLN[8:0] = 0), so the buffer must be cleared with the BCLR bit. The length of the receive data can be confirmed in the DTLN[8:0] bits in the port control register.

**Table 26.23 FIFO port function settings (1 of 2)**

| Register name | Bit name | Description |
|---|---|---|
| CFIFOSEL,<br>DnFIFOSEL<br>(n = 0, 1) | RCNT | Selects DTLN[11:0] read mode |
| | REW | FIFO buffer rewind (re-read, rewrite) |
| | DCLRM | Automatically clears receive data for a specified pipe after the data is read (only for DnFIFO) |
| | DREQE | Enables DMA/DTC transfers (only for DnFIFO) |
| | MBW | FIFO port access bit width |
| | BIGEND | Selects FIFO port endian |
| | ISEL | FIFO port access direction (only for DCP) |
| | CURPIPE | Selects the current pipe |

**Table 26.23 FIFO port function settings (2 of 2)**

| Register name | Bit name | Description |
|---|---|---|
| CFIFOCTR, DnFIFOCTR (n = 0, 1) | BVAL | Ends writing to the FIFO buffer |
| | BCLR | Clears the FIFO buffer on the CPU side |
| | DTLN | Checks the length of receive data |

### (1) FIFO port selection

Table 26.24 shows the pipes that can be selected with the different FIFO ports. The pipe to be accessed must be selected in the CURPIPE[3:0] bits in the port select register. After the pipe is selected, the software must check whether the written value can be read correctly from the CURPIPE[3:0] bits. (If the previous pipe number is read, it indicates that the USBFS is modifying the pipe.) Next, the software checks that the FRDY bit in the port control register is 1.

In addition, the software must specify the bus width to be accessed in the MBW bit in the port select register. The FIFO buffer access direction conforms to the PIPECFG.DIR setting. For the DCP only, the ISEL bit in the port select register determines the direction.

**Table 26.24 FIFO port access by pipe**

| Pipe | Access method | Ports that can be used |
|---|---|---|
| DCP | CPU access | CFIFO port register |
| Pipes 1 to 9 | CPU access | • CFIFO port register<br>• D0FIFO/D1FIFO port register |
| | DMA/DTC access | D0FIFO/D1FIFO port register |

### (2) REW bit

It is possible to temporarily stop access to a pipe currently being accessed, access a different pipe, and then continue processing for the first pipe again. The REW bit in the port select register is used for this processing.

If a pipe is selected in the CURPIPE[3:0] bits in the port select register with the REW bit set to 1, the pointer used for reading from and writing to the FIFO buffer is reset, and reading or writing can be carried out from the first byte. If a pipe is selected with 0 set for the REW bit, data can be read and written in continuation from the previous selection, without the pointer being reset.

To access the FIFO port, the software must check that the FRDY bit in the port control register is 1 after selecting a pipe.

## 26.3.8 DMA Transfers (D0FIFO and D1FIFO Ports)

### (1) Overview of DMA transfers

For pipes 1 to 9, the FIFO port can be accessed using the DMAC. When buffer access for a pipe targeted for DMA transfer is enabled, a DMA transfer request is issued.

Select the unit of transfer to the FIFO port in the DnFIFOSEL.MBW bit, and select the pipe targeted for the DMA transfer in the DnFIFOSEL.CURPIPE[3:0] bits. Do not change the selected pipe during the DMA transfer.

### (2) DnFIFO auto clear mode (D0FIFO and D1FIFO port reading direction)

If 1 is set in the DnFIFOSEL.DCLRM bit, the USBFS automatically clears the FIFO buffer of the selected pipe when reading of data from the FIFO buffer is complete.

Table 26.25 shows the packet reception and FIFO buffer clearing processing by software for each of the settings. As shown in the table, the buffer clearing conditions depend on the value set in the PIPECFG.BFRE bit. Using the DnFIFOSEL.DCLRM bit eliminates the need for the buffer to be cleared by software in any situation that requires buffer clearing. This enables DMA transfers without involving software.

The DnFIFO auto clear mode can only be set in the FIFO buffer reading direction.

**Table 26.25 Packet reception and FIFO buffer clearing processing by software**

| Buffer status when packet is received | Register setting | | | |
|---|---|---|---|---|
| | DCLRM = 0 | | DCLRM = 1 | |
| | BFRE = 0 | BFRE = 1 | BFRE = 0 | BFRE = 1 |
| Buffer full | No clearing required | No clearing required | No clearing required | No clearing required |
| Zero-length packet reception | Clearing required | Clearing required | No clearing required | No clearing required |
| Normal short packet reception | No clearing required | Clearing required | No clearing required | No clearing required |
| Transaction count end | No clearing required | Clearing required | No clearing required | No clearing required |

## 26.3.9 Control Transfers Using the DCP

The DCP is used for data transfers in the control transfer data stage. The FIFO buffer of the DCP is a 64-byte single buffer with a fixed area for both control reads and control writes. The FIFO buffer can be accessed only through the CFIFO port.

### 26.3.9.1 Control transfers in host controller mode

#### (1) Setup stage

The USQREQ, USBVAL, USBINDX, and USBLENG registers are used to transmit USB requests for setup transactions. Writing the setup packet data to the registers and then writing 1 to the DCPCTR.SUREQ bit transmits the specified data for the setup transaction. On completion of the transaction, the SUREQ bit clears to 0. Do not change these USB request registers while SUREQ = 1.

When an attached function device is detected, the software must issue the first setup transaction for the device using this sequence with the DCPMAXP.DEVSEL[3:0] bits cleared to 0 and the DEVADD0.USBSPD[1:0] bits set appropriately.

When an attached function device is shifted to the Address state, the software must issue setup transactions using this sequence with the assigned USB address set in the DEVSEL[3:0] bits and the bits in DEVADDn corresponding to the specified USB address set appropriately. For example, when PIPEMAXP.DEVSEL[3:0] = 0010b, make appropriate settings in DEVADD2. When PIPEMAXP.DEVSEL[3:0] = 0101b, make appropriate settings in DEVADD5.

When the setup transaction data is sent, an interrupt request is generated based on the response from the peripheral device (SIGN or SACK bit in INTSTS1). This interrupt request allows the software to check the setup transaction result.

A DATA0 data packet (USB request) for the setup transaction is always transmitted regardless of the status of the DCPCTR.SQMON bit.

#### (2) Data stage

The data stage is used to transfer data using the DCP FIFO buffer.

Before accessing the DCP FIFO buffer, specify the access direction in the CFIFOSEL.ISEL bit. Specify the transfer direction in the DCPCFG.DIR bit.

For the first data packet of the data stage, the data PID must be transferred as DATA1. Set data PID = DATA1 in the DCPCTR.SQSET bit and set the PID bits = BUF. Completion of data transfer is detected using the BRDY or BEMP interrupt.

For control write transfers, when the number of data bytes to be sent is an integer multiple of the maximum packet size, the software must send a zero-length packet at the end.

#### (3) Status stage

The status stage is used for zero-length packet data transfers in the reverse direction of the data stage. As in the data stage, data is transfered using the DCP FIFO buffer. Transactions are executed using the same procedure as the data stage.

Data packets in the status stage must be transmitted and received with the data PID set to DATA1 using the DCPCTR.SQSET bit.

When a zero-length packet is received, check the receive-data length in the CFIFOCTR.DTLN[8:0] bits after a BRDY interrupt is generated, and then clear the FIFO buffer using the BCLR bit.

### 26.3.9.2 Control transfers in device controller mode

#### (1) Setup stage

The USBFS sends an ACK response to a normal setup packet for the USBFS. The USBFS operates in the setup stage as follows:

On receiving a new setup packet, the USBFS sets the following bits:

- Sets the INTSTS0.VALID bit to 1
- Sets the DCPCTR.PID[1:0] bits to NAK
- Sets the DCPCTR.CCPL bit to 0

When the USBFS receives a data packet following a setup packet, it stores the USB request parameters in USBREQ, USBVAL, USBINDX, and USBLENG.

Before performing the response processing for a control transfer, set the VALID flag to 0. When the VALID bit = 1, PID = BUF cannot be set, and the data stage cannot be terminated.

Using the VALID bit function, the USBFS can suspend a request being processed when it receives a new USB request during a control transfer and return a response to the latest request.

In addition, the USBFS automatically detects the direction bit (bmRequestType bit [8]) and the request data length (wLength) in the received USB request. It distinguishes between control read transfers, control write transfers, and no-data control transfers, and it controls stage transitions. For an incorrect sequence, a sequence error occurs in the control transfer stage transition interrupt, and the interrupt is reported to the software. For a diagram of the stage control by the USBFS, see Figure 26.16.

#### (2) Data stage

The DCP must be used to execute data transfers for received USB requests. Before accessing the DCP FIFO buffer, specify the access direction in the CFIFOSEL.ISEL bit.

If the transfer data is larger than the size of the DCP FIFO buffer, execute the data transfer using the BRDY interrupt for control write transfers and the BEMP interrupt for control read transfers.

#### (3) Status stage

Control transfers are terminated by setting the DCPCTR.CCPL bit to 1 while the DCPCTR.PID[1:0] bits are set to BUF.

After this setting is made, the USBFS automatically executes the status stage based on the data transfer direction determined at the setup stage. The procedure is as follows:

- For control read transfers

  The USBFS receives a zero-length packet from the USB host and transmits an ACK response.
- For control write transfers and no-data control transfers

  The USBFS transmits a zero-length packet and receives an ACK response from the USB host.

#### (4) Control transfer auto response function

The USBFS automatically responds to a correct SET_ADDRESS request. If any of the following errors occurs in the SET_ADDRESS request, a response from the software is necessary.

- bmRequestType is not 0x00: Any transfer other than a control write transfer
- wIndex is not 0x00: Request error
- wLength is not 0x00: Any transfer other than a no-data control transfer
- wValue is larger than 0x7F: Request error
- INTSTS0.DVSQ[2:0] are 011b (Configured state): Control transfer of a device state error

For all requests other than the SET_ADDRESS request, a response is required from the corresponding software.

### 26.3.10 Bulk Transfers (Pipes 1 to 5)

The FIFO buffer usage (single/double buffer setting) is configurable for bulk transfers. The USBFS provides the following functions for bulk transfers:

- BRDY interrupt function (PIPECFG.BFRE bit), see section 26.3.3.1. BRDY interrupt
- Transaction count function (PIPEnTRE.TRENB, TRCLR, and PIPEnTRN.TRNCNT[15:0] bits), see section 26.3.4.5. Transaction counter for pipes 1 to 5 in the receiving direction
- Response PID = NAK function (PIPECFG.SHTNAK bit), see section 26.3.4.8. Response PID = NAK function
- Auto response mode (PIPEnCTR.ATREPM bit), see section 26.3.4.9. Auto response mode

### 26.3.11 Interrupt Transfers (Pipes 6 to 9)

In device controller mode, the USBFS performs interrupt transfers based on the timing dictated by the host controller.

In host controller mode, the software can set the timing for issuing tokens using the interval counter.

#### 26.3.11.1 Interval counter for interrupt transfers in host controller mode

Specify the transaction interval for interrupt transfers in the PIPEPERI.IITV[2:0] bits. The USBFS issues interrupt transfer tokens based on this interval.

##### (1) Counter initialization

The USBFS initializes the interval counter under the following conditions:

- Power-on reset

  This initializes the IITV[2:0] bits.
- FIFO buffer initialization using the PIPEnCTR.ACLRM bit:

  This does not initialize the IITV[2:0] bits, but does initialize the count value. Setting the PIPEnCTR.ACLRM bit to 0 starts counting from the value set in IITV[2:0].

The interval counter is not initialized in the following case:

- USB bus reset or USB suspended

  The IITV[2:0] bits are not initialized. Setting 1 to the DVSTCTR0.UACT bit starts counting from the value saved before entering the USB bus reset state or USB suspend state.

##### (2) Operation when tokens cannot be transmitted or received even on token generation

No token is generated in the following cases even at token generation time. In these cases, the USBFS tries to execute the transaction in the next interval.

- When the PID is set to NAK or STALL
- When the FIFO buffer is full at token transmit time in the receiving (IN) direction
- When there is no data to be transmitted in the FIFO buffer at token transmit time in the transmitting (OUT) direction

### 26.3.12 Isochronous Transfers (Pipes 1 and 2)

The USBFS provides the following functions for isochronous transfers:

- Notification of isochronous transfer error
- Interval counter specified in the PIPEPERI.IITV[2:0] bits
- Isochronous IN transfer data setup control (IDLY function)
- Isochronous IN transfer buffer flush function specified in the PIPEPERI.IFIS bit

### 26.3.12.1 Error detection in isochronous transfers

The USBFS provides a function for detecting the errors described in this section, so that when errors occur in isochronous transfers, they can be controlled by software. Table 26.26 and Table 26.27 show the priority order for errors detected by the USBFS and the associated interrupts.

#### PID errors

- The PID value of the received packet is invalid.

#### CRC errors and bit stuffing errors

- A CRC error is found in a received packet or the bit stuffing is illegal.

#### Maximum packet size exceeded

- The data size of the received packet exceeds the specified maximum packet size.

#### Overrun and underrun errors

In host controller mode:

- The FIFO buffer is full at token transmit time in the IN (receiving) direction
- There is no data to be sent in the FIFO buffer at token transmit time in the OUT (transmitting) direction

In device controller mode:

- There is no data to be sent in the FIFO buffer at token receive time in the IN (transmitting) direction
- The FIFO buffer is full at token receive time in the OUT (receiving) direction

#### Interval errors

In device controller mode, the following cases are treated as an interval error:

- Failure to receive an IN token in the interval frame during an isochronous IN transfer
- Failure to receive an OUT token in the interval frame during an isochronous OUT transfer

**Table 26.26 Error detection for token transmission and reception**

| Detection priority | Error | Generated interrupt and status |
|---|---|---|
| 1 | PID error | No interrupts are generated in either host or device controller mode (ignored as a corrupted packet) |
| 2 | CRC or bit stuffing error | No interrupts are generated in either host or device controller mode (ignored as a corrupted packet) |
| 3 | Overrun or underrun error | An NRDY interrupt is generated to set the FRMNUM.OVRN bit to 1 in both host and device controller modes.<br>In device controller mode, a zero-length packet is transmitted in response to an IN token.<br>No data packets are received in response to OUT token. |
| 4 | Interval error | An NRDY interrupt is generated in device controller mode. No interrupt is generated in host controller mode. |

**Table 26.27 Error detection for data packet reception**

| Detection priority | Error | Generated interrupt and status |
|---|---|---|
| 1 | PID error | No interrupts are generated (ignored as a corrupted packet) |
| 2 | CRC or bit stuffing error | An NRDY interrupt is generated and the FRMNUM.CRCE bit sets to 1 in both host and device controller modes |
| 3 | Maximum packet size exceeded error | A BEMP interrupt is generated and the PID[1:0] bits set to STALL in both host and device controller modes |

### 26.3.12.2 DATA-PID

In device controller mode, the USBFS responds to a received PID as follows:

#### (1) IN direction

- DATA0: Transmitted as data packet PID
- DATA1: Not transmitted
- DATA2: Not transmitted
- mDATA: Not transmitted

#### (2) OUT direction

- DATA0: Received normally as data packet PID
- DATA1: Received normally as data packet PID
- DATA2: Packets ignored
- mDATA: Packets ignored

### 26.3.12.3 Interval counter

The isochronous transfer interval can be set in the PIPEPERI.IITV[2:0] bits. In device controller mode, the interval counter enables the functions as shown in Table 26.28. In host controller mode, the USBFS generates the token issuance timing, and the interval counter operation is the same as that for interrupt transfers.

**Table 26.28 Interval counter functions in device controller mode**

| Transfer direction | Function | Conditions for detection |
|---|---|---|
| IN | Transmit buffer flush | Failure to receive an IN token successfully in the interval frame during an isochronous IN transfer. |
| OUT | Notification of no reception of token | Failure to receive an OUT token successfully in the interval frame during an isochronous OUT transfer. |

The interval count is performed when an SOF is received or for interpolated SOFs, so the isochronism can be maintained even if an SOF is corrupt. The frame interval can be set to $2^{IITV}$ frames.

#### (1) Counter initialization in device controller mode

The USBFS initializes the interval counter under the following conditions:

- Power-on reset

  This initializes the PIPEPERI.IITV[2:0] bits.
- FIFO buffer initialization using the ACLRM bit

  This does not initialize the IITV[2:0] bits, but does initialize the count value.

After the interval counter is initialized, the interval count starts under one of the following conditions when a packet is transferred successfully:

- An SOF is received after data is transmitted in response to an IN token when PID = BUF
- An SOF is received after data is received in response to an OUT token when PID = BUF

The interval counter is not initialized in the following conditions:

- When the PID[1:0] bits are set to NAK or STALL

  This does not stop the interval timer. The USBFS attempts the transaction in the next interval.
- When the USB bus is reset or USBFS is suspended

  This does not initialize the IITV[2:0] bits. When an SOF is received, the interval counter starts counting from the value set before SOF was received.

### (2) Interval counting and transfer control in host controller mode

The USBFS controls the interval between token issuance operations based on the PIPEPERI.IITV[2:0] bits settings. Specifically, the USBFS issues a token for a selected pipe once every $2^{IITV}$ frames.

The USBFS starts counting the token issuance interval at the frame following the frame in which the PID[1:0] bits are set to BUF by software.

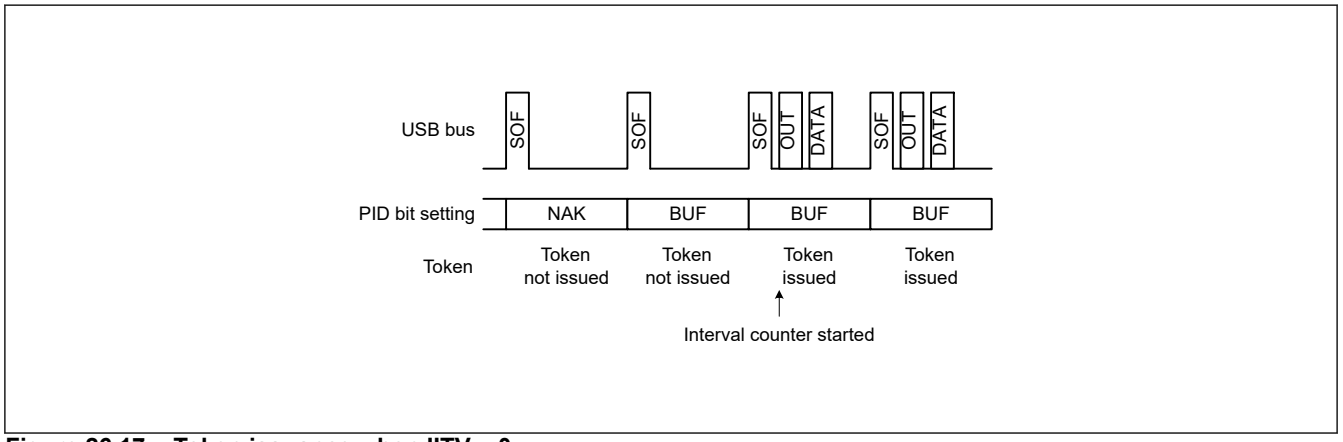



**Figure 26.17 Token issuance when IITV = 0**

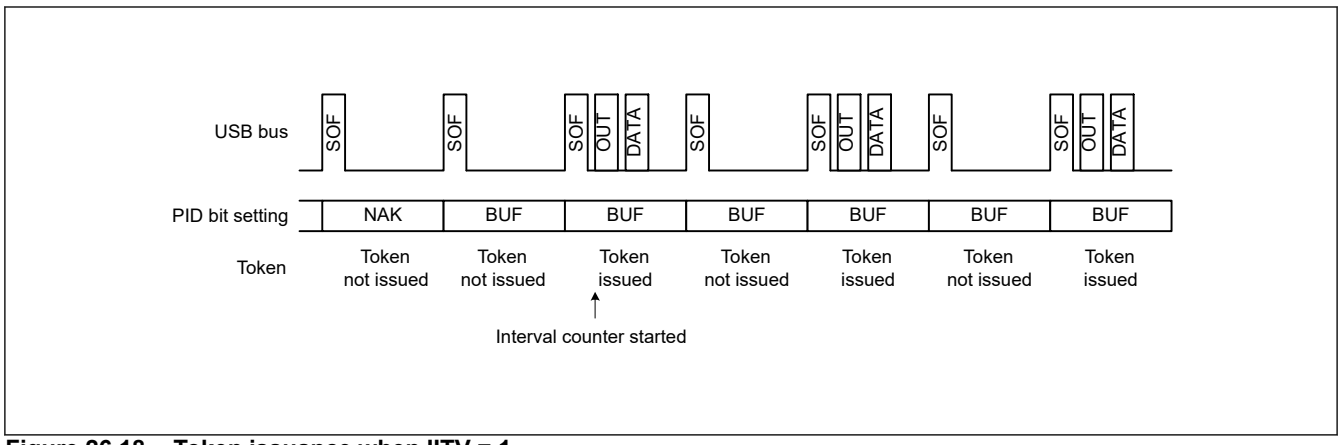



**Figure 26.18 Token issuance when IITV = 1**

When the selected pipe is set for isochronous transfers, the USBFS performs the following operation in addition to controlling the token issuance interval. The USBFS issues a token even when the NRDY interrupt generation condition is satisfied.

#### When the selected pipe is for isochronous IN transfers

The USBFS generates an NRDY interrupt when the USBFS issues an IN token but does not successfully receive a packet from a peripheral device (no response or packet error).

The USBFS sets the FRMNUM.OVRN bit to 1, generating an NRDY interrupt, when the time to issue an IN token occurs while the USBFS cannot receive data because the FIFO buffer is full, because the CPU or DMAC/DTC is too slow in reading data from the FIFO buffer.

#### When the selected pipe is for isochronous OUT transfers

The USBFS sets the OVRN bit to 1, generating an NRDY interrupt and transmitting a zero-length packet, when the time to issue an OUT token comes while there is no data to be transmitted in the FIFO buffer, or because the CPU or DMAC/DTC is too slow in writing data to the FIFO buffer.

The token issuance interval is reset on any of the following conditions:

- When the USBFS is reset through a reset pin

  This initializes the IITV[2:0] bits.

- When the PIPEnCTR.ACLRM bit is set to 1 by software

### (3) Interval counting and transfer control in device controller mode

**When the selected pipe is for isochronous OUT transfers**

The USBFS generates an NRDY interrupt when it fails to receive a data packet within the interval set in the PIPEPERI.IITV[2:0] bits.

The USBFS also generates an NRDY interrupt when it fails to receive data because of a CRC error or other errors contained in the data packet or because the FIFO buffer is full.

The NRDY interrupt is generated on SOF packet reception. Even if the SOF packet is corrupted, internal interpolation allows the interrupt to be generated when the SOF packet is received. However, when the IITV bits are set to a value other than 0, the USBFS generates an NRDY interrupt on receiving an SOF packet for every interval after interval counting starts.

When the PID[1:0] bits are set to NAK by software after starting the interval timer, the USBFS does not generate an NRDY interrupt on receiving an SOF packet.

The timing for starting interval counting depend on the IITV[2:0] setting as follows:

- When the IITV[2:0] bits = 0:

  Interval counting starts at the next frame after the software changes the PID[1:0] bits of the selected pipe to BUF.

- When the IITV[2:0] bits ≠ 0:

  Interval counting starts on completion of successful reception of the first data packet after the PID[1:0] bits for the selected pipe are changed to BUF.

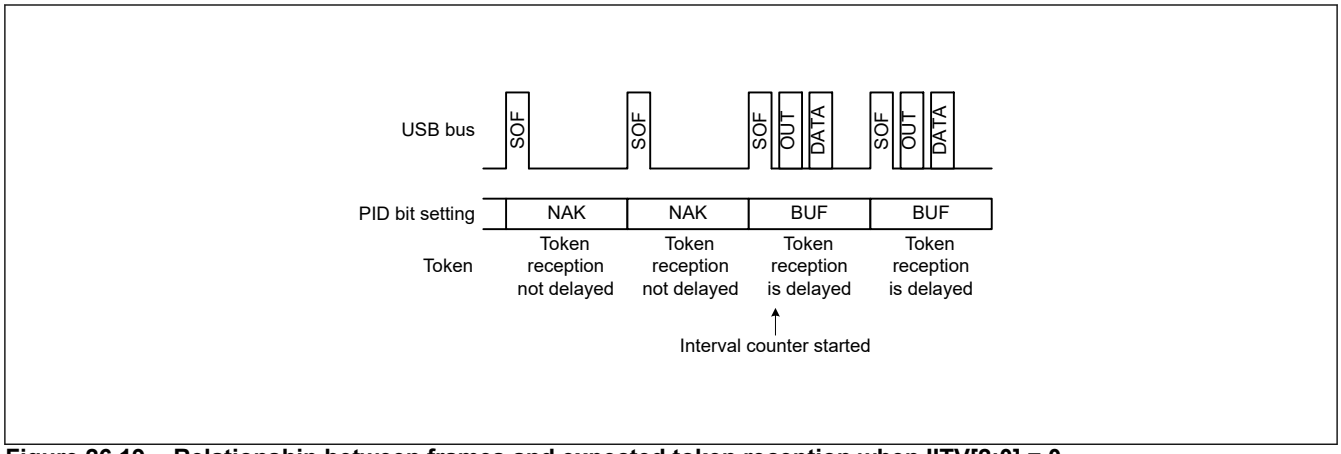



**Figure 26.19 Relationship between frames and expected token reception when IITV[2:0] = 0**

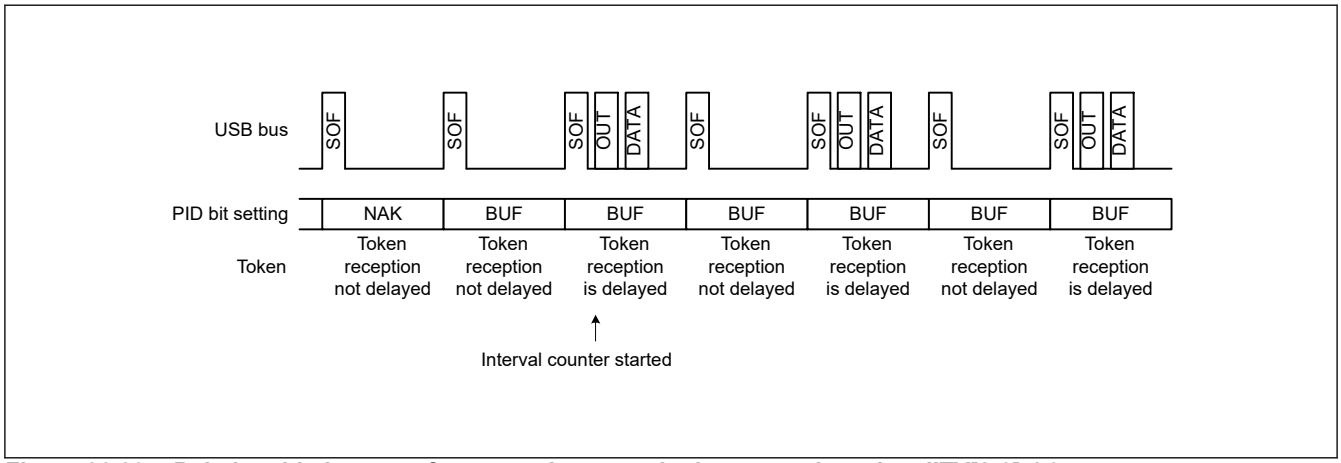



**Figure 26.20 Relationship between frames and expected token reception when IITV[2:0] ≠ 0**

**When the selected pipe is for isochronous IN transfers**

The PIPEPERI.IFIS bit must be 1 for this use case. When IFIS = 0, the USBFS transmits a data packet in response to a received IN token regardless of the PIPEPERI.IITV[2:0] setting.

When IFIS is 1 and there is data to be transmitted in the FIFO buffer, the USBFS clears the FIFO buffer when it fails to receive an IN token in the frame at the interval set in the IITV[2:0] bits.

The USBFS also clears the FIFO buffer when it fails to receive an IN token successfully because of a bus error, such as a CRC error, contained in the IN token.

The FIFO buffer is cleared on SOF packet reception. Even if the SOF packet is corrupted, the internal interpolation allows the FIFO buffer to be cleared when the SOF packet is received.

The timing to start interval counting depends on the IITV[2:0] setting, as with OUT transfers.

The interval is counted on any of the following conditions in device controller mode:

- When a hardware reset is applied to the USBFS (which also sets the IITV[2:0] bits to 000b)
- When the PIPEnCTR.ACLRM bit is set to 1 by software
- When the USBFS detects a USB bus reset

### (4) Transmit data setup for isochronous transfers in device controller mode

With isochronous data transmission using the USBFS in device controller mode, after data is written to the FIFO buffer, a data packet can be transmitted in the first frame after the SOF packet is detected. This isochronous transfer transmit data setup function can identify the frame that started transmission.

When the double buffering is used, transmission is only enabled for the buffer where data writing was completed first, even after the data write to both buffers is complete. Accordingly, even if multiple IN tokens are received, only the one packet of FIFO buffer data is transmitted.

When the FIFO buffer is ready to transmit data when an IN token is received, the data is transferred and a normal response is returned. However, if the FIFO buffer cannot transmit data, a zero-length packet is transmitted and an underrun error occurs.

Figure 26.21 shows an example transmission using the isochronous transfer transmission data setup function when IITV = 0 (every frame) is set.

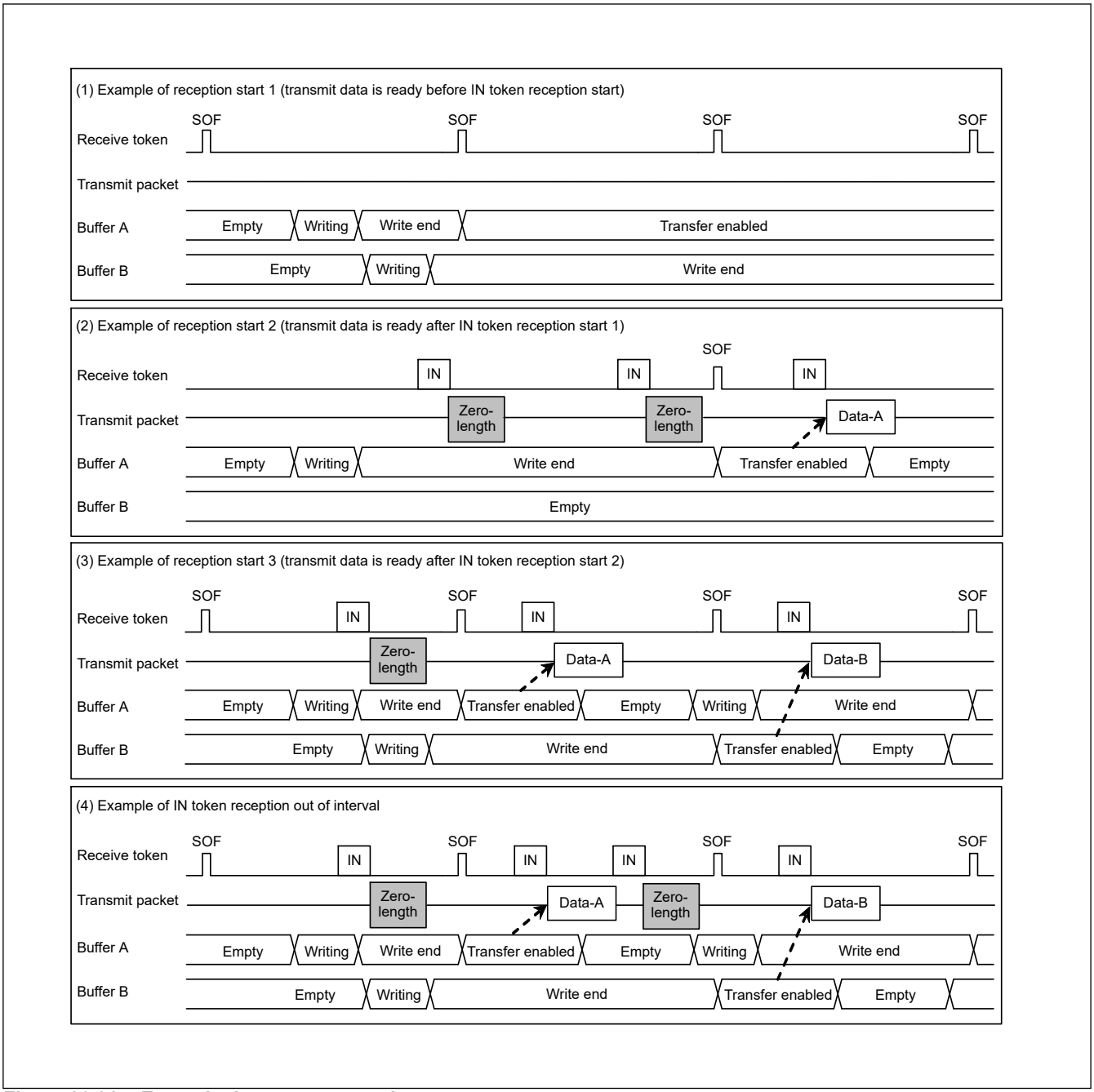



**Figure 26.21 Example data setup operation**

## (5) Transmit buffer flush for isochronous transfers in device controller mode

In device controller mode during isochronous data transmission, if the USBFS receives an SOF packet for the next frame without receiving an IN token in the interval frame, it operates as if the IN token is corrupt and clears the buffer that is enabled for transmission, putting that buffer in the writing enabled state.

When double buffering is used and writing to both buffers is complete, the cleared FIFO buffer is assumed to be the one where the data was transmitted in the interval frame, and transmission is enabled for the FIFO buffer that was not cleared on SOF packet reception.

The timing of the buffer flush function depends on the PIPEPERI.IITV[2:0] setting as follows:

- When IITV = 0:

  The buffer flush operation starts from the first frame after the pipe is enabled.

- When IITV ≠ 0:

  The buffer flush operation starts after the first normal transaction.

Figure 26.22 shows an example buffer flush. When an unanticipated token is received before the interval frame, the USBFS sends the write data or a zero-length packet as an underrun error, depending on the data setup status.

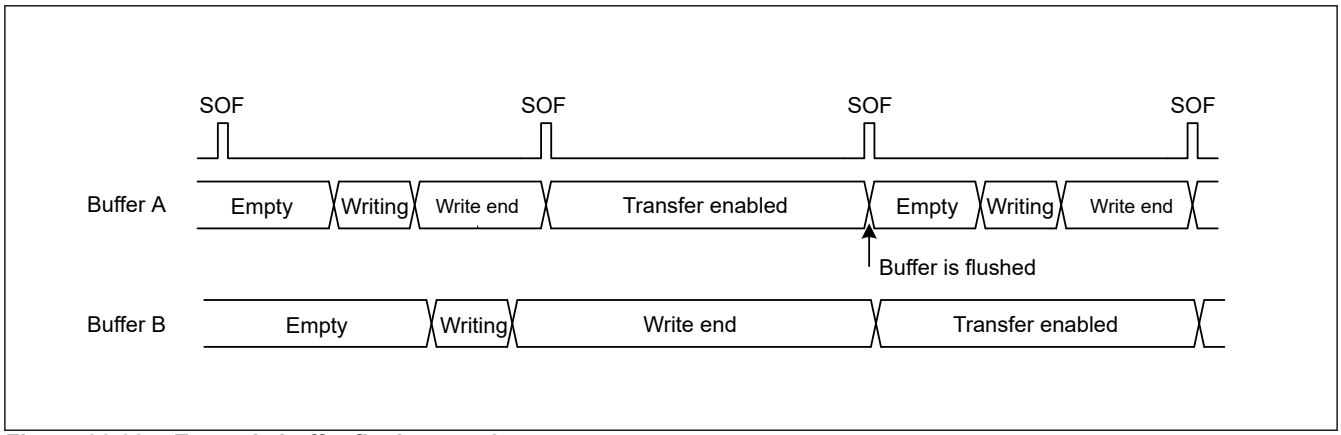



**Figure 26.22 Example buffer flush operation**

Figure 26.23 shows an example interval error occurrence. There are five types of interval errors, as shown in the figure. An interval error occurs at timing (A), and the buffer flush function is activated.

If an interval error occurs during an IN transfer, the buffer flush function is activated. If it occurs during an OUT transfer, an NRDY interrupt is generated. Use the FRMNUM.OVRN bit to distinguish between this and NRDY interrupts triggered by received packet errors and overrun errors.

For tokens that are shaded in the figure, responses are returned based on the FIFO buffer status.

- IN direction:
  - If the buffer is ready to transfer data, the data is transferred and a normal response is returned
  - If the buffer is not ready to transfer data, a zero-length packet is transmitted and an underrun error occurs
- OUT direction:
  - If the buffer is ready to receive data, the data is received and a normal response is returned
  - If the buffer is not ready to receive data, the received data is discarded and an overrun error occurs

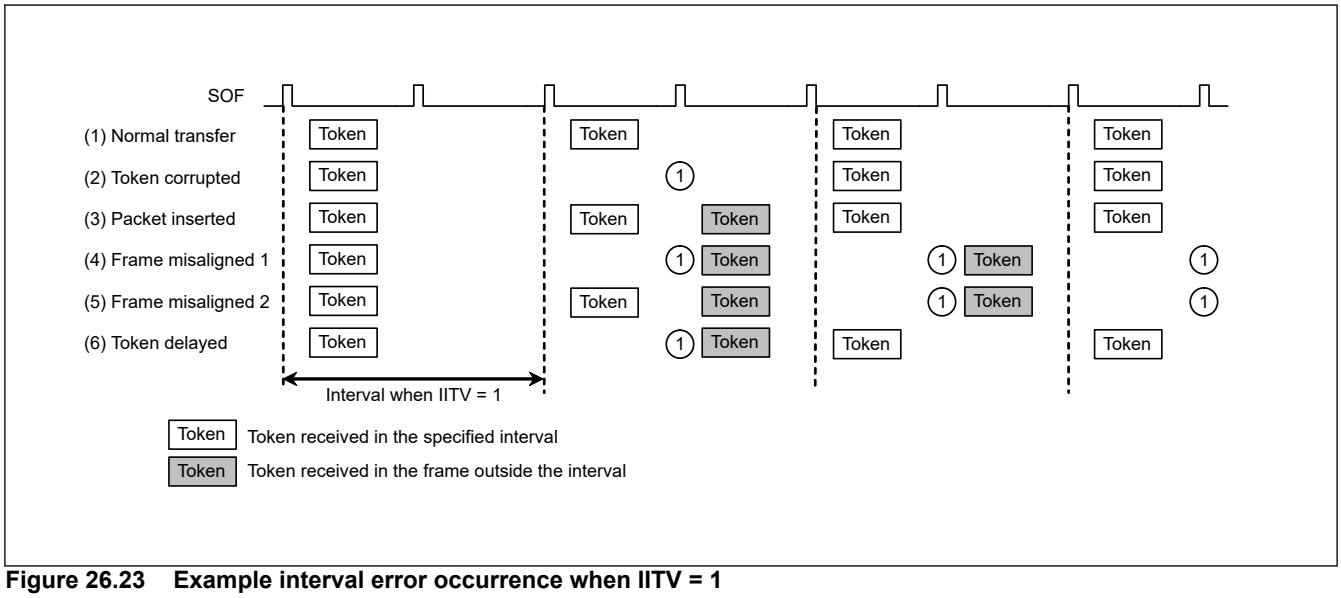



**Figure 26.23 Example interval error occurrence when IITV = 1**

## 26.3.13 SOF Interpolation Function

In device controller mode, if packet reception is disabled at intervals of 1 ms because the SOF packet is corrupted or missing, the USBFS interpolates the SOF. SOF interpolation begins when the USBE and SCKE bits in SYSCFG are set to 1 and an SOF packet is received.

The interpolation function is initialized under the following conditions:

- MCU reset
- USB bus reset
- Suspend state detection

The SOF interpolation operates as follows:

- The interpolation function is not activated until an SOF packet is received
- When the first SOF packet is received, interpolation is performed by counting 1 ms on the 48-MHz internal clock
- When the second and subsequent SOF packets are received, interpolation is performed at the previous reception interval
- Interpolation is not performed in the Suspend state or on reception of a USB bus reset

The USBFS supports the following functions controlled by SOF packet reception. These functions operate normally with SOF interpolation if the SOF packet is missing:

- Updating of the frame number
- SOFR interrupt timing
- Isochronous transfer interval count

If an SOF packet is missing during full-speed operation, the FRMNUM.FRNM[10:0] bits are not updated.

## 26.3.14 Pipe Schedule

### 26.3.14.1 Conditions for generating transactions

In host controller mode and when the DVSTCTR0.UACT bit is set to 1, the USBFS generates transactions under the conditions shown in Table 26.29.

**Table 26.29 Conditions for generating transactions**

| Transaction | Conditions for generation | | | | |
|---|---|---|---|---|---|
| | DIR | PID | IITV0 | Buffer state | SUREQ |
| Setup | —*1 | —*1 | —*1 | —*1 | 1 setting |
| Control transfer data stage, status stage, bulk transfer | IN | BUF | Invalid | Receive area exists | —*1 |
| | OUT | BUF | Invalid | Transmit data exists | —*1 |
| Interrupt transfer | IN | BUF | Valid | Receive area exists | —*1 |
| | OUT | BUF | Valid | Transmit data exists | —*1 |
| Isochronous transfer | IN | BUF | Valid | *2 | —*1 |
| | OUT | BUF | Valid | *3 | —*1 |

Note 1. An em dash (—) in the table indicates that the condition is unrelated to the generating of tokens. “Valid” indicates that, for interrupt transfers and isochronous transfers, a transaction is generated only in transfer frames that are based on the interval counter. “Invalid” indicates that a transaction is generated regardless of the interval counter.

Note 2. This indicates that a transaction is generated regardless of whether there is a receive area. If there is no receive area, however, the received data is discarded.

Note 3. This indicates that a transaction is generated regardless of whether there is any data to be transmitted. If there is no data to be transmitted, however, a zero-length packet is transmitted.

### 26.3.14.2 Transfer schedule

This section describes the transfer scheduling within a frame of the USBFS. After the USBFS sends an SOF, the transfer is carried out in the following sequence:

1. Execution of periodic transfers:

A pipe is searched for in the order of pipe 1 → pipe 2 → pipe 6 → pipe 7 → pipe 8 → pipe 9, and then if there is a pipe for which an isochronous or interrupt transfer transaction can be generated, the transaction is generated.

2. Setup transactions for control transfers:

   The DCP is checked, and if a setup transaction is possible, it is sent.

3. Execution of bulk transfers, control transfer data stages, and control transfer status stages:

   A pipe is searched for in the order of DCP → pipe 1 → pipe 2 → pipe 3 → pipe 4 → pipe 5, and then if there is a pipe for which a transaction for a bulk transfer, a control transfer data stage, or a control transfer status stage can be generated, the transaction is generated.

   When a transaction is generated, processing moves to the next pipe transaction regardless of whether the response from the peripheral device is ACK or NAK. If there is time for transfer within the frame, step 3 is repeated.

#### 26.3.14.3 Enabling USB communication

Setting the DVSTCTR0.UACT bit to 1 initiates an SOF transmission, and transaction generation is enabled. Setting the UACT bit to 0 stops SOF transmission and the Suspend state is invoked. If the UACT setting is changed from 1 to 0, processing stops after the next SOF is sent.

### 26.3.15 Battery Charging Detection Processing

The USBFS provides control over the data contact detection processing (D+ line contact checking), primary detection processing (charger detection processing), and secondary detection processing (charger determination processing) as defined in the Battery Charging Specification.

#### 26.3.15.1 Processing in device controller mode

To operate a function device as a battery charging portable device:

1. Start primary detection processing after detecting contact with the D+ and D- lines. The Battery Charging Specification describes two processing methods for Data Contact Detection. The USBFS supports both methods as follows:

   - Software processing
     After a VBINT interrupt or polling of the VBSTS flag indicates a change in the state of the USBFS_VBUS input pin, software controls a wait from 300 to 900 ms. The BCCTRL1.VDPSRCE and CHGDETE bits are then both set to 1 to start primary detection processing.
   - Hardware Processing
     Apply 7 to 13 µA of current to the D+ line to hold the D+ line at the logical high level. This is done to detect the D+ and D- lines going to the logical low level because of pull-down resistors on the host device side when the D+ and D- lines come in contact with those of the host. Monitor the SYSSTS0.LNST[1:0] flags while the PHYSECTRL.CNEN bit, BCCTRL1.RPDME and IDPSRCE bit are set to 1 to see when the level on the D+ line changes from high to low. After detecting a low level on the D+ line, clear the PHYSECTRL.CNEN bit, BCCTRL1.RPDME and IDPSRCE bit to 0 and set both the BCCTRL1.VDPSRCE and CHGDETE bits to 1 to start primary detection processing. The VDPSRCE and CHGDETE bits must be set to 1 simultaneously.

2. After the start of primary detection processing followed by a software-controlled wait of 40 ms, check the BCCTRL1.CHGDETSTS flag. A value of 1 indicates detection of a charger, and secondary detection processing starts.*1

3. To start secondary detection processing, clear the BCCTRL1.VDPSRCE bit to 0, then clear the BCCTRL1.CHGDETE bits to 0 after a software-controlled wait of 20 ms. Next, set both the BCCTRL1.VDMSRCE and PDDETE bits to 1.

4. After the start of secondary detection processing followed by a software-controlled wait of 40 ms, check the BCCTRL1.PDDETSTS flag, and determine the result of the secondary detection processing.

Note 1. In primary detection processing, detection of a voltage above the range from 0.25 to 0.4 V and below the range from 0.8 to 2.0 V on the D-Line indicates that the other device is a host device that supports battery charging (charging downstream port). The BCCTRL1.CHGDETSTS flag in the PHY block only indicates whether the voltage on the D- line is higher than the range from 0.25 to 0.4 V, so add processing as required to read the SYSSTS0.LNST[1:0] flags and confirm that the voltage on the D- line is also below the range from 0.8 to 2.0 V.

Figure 26.24 shows the process flow.

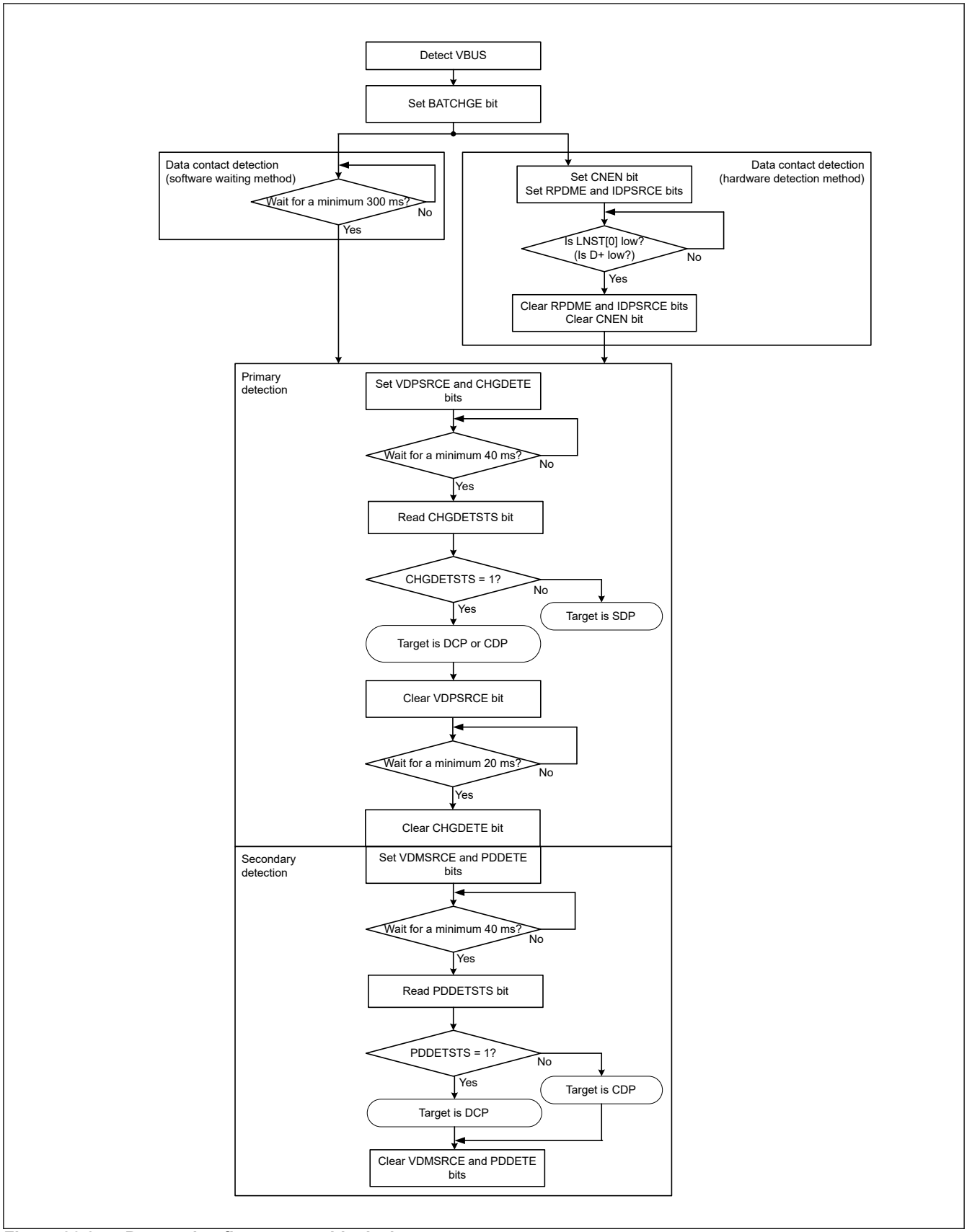



**Figure 26.24 Processing flow as portable device**

### 26.3.15.2 Processing in host controller mode

In host controller mode, driving the D- line is required for a portable device to perform primary detection. The USBFS supports the following two primary detection methods:

#### When using the Portable Device detection function

The D- line is driven when a Portable Device is detected.

1. Start driving the VBUS.
2. Set the BCCTRL1.PDDETE bit to 1 to enable the portable device detection circuit.
3. Monitor the portable device detection signal and start driving the D- line when the level of the portable device detection signal is high.*1
4. Stop driving the D- line when the portable device detection signal is at the low level.*1

Note 1. The PDDETINT interrupt indicates a change in the level of the portable device detection signal, and the current level can be obtained by reading the PDDETSTS flag.

#### When not using the Portable Device detection function

Regardless of whether or not a Portable Device was detected, drive the D- line after device disconnection is detected, and leave the D- line open after connection is detected. Software handles the timing of a and b.

a. After disconnection is detected, start driving the D- line within 200 ms.
b. After connection is detected, stop driving the D- line within 10 ms.

Figure 26.25 shows the process flow for steps 1. to 4. and Figure 26.26 shows the process flow for steps a. and b., respectively.

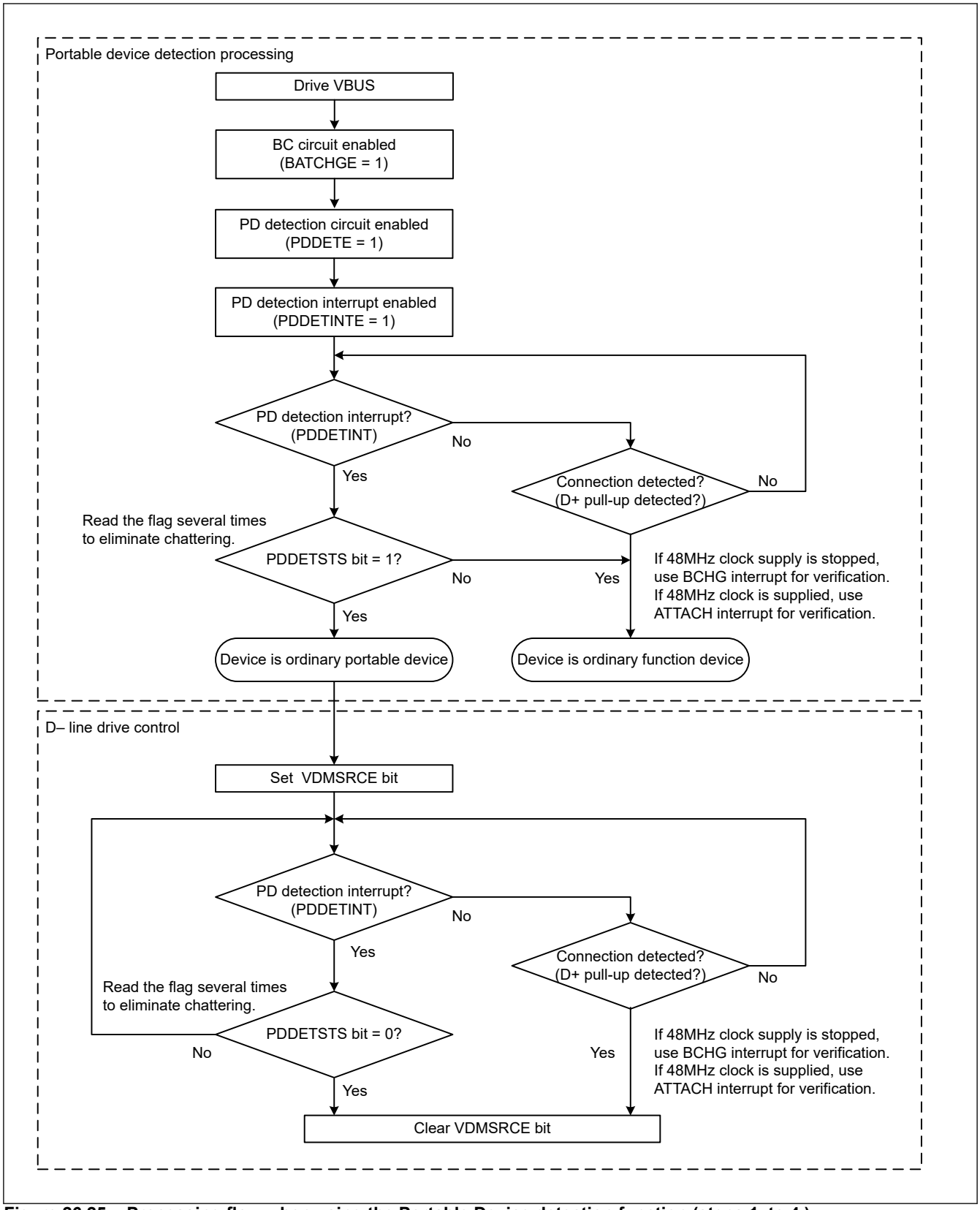



**Figure 26.25 Processing flow when using the Portable Device detection function (steps 1. to 4.)**

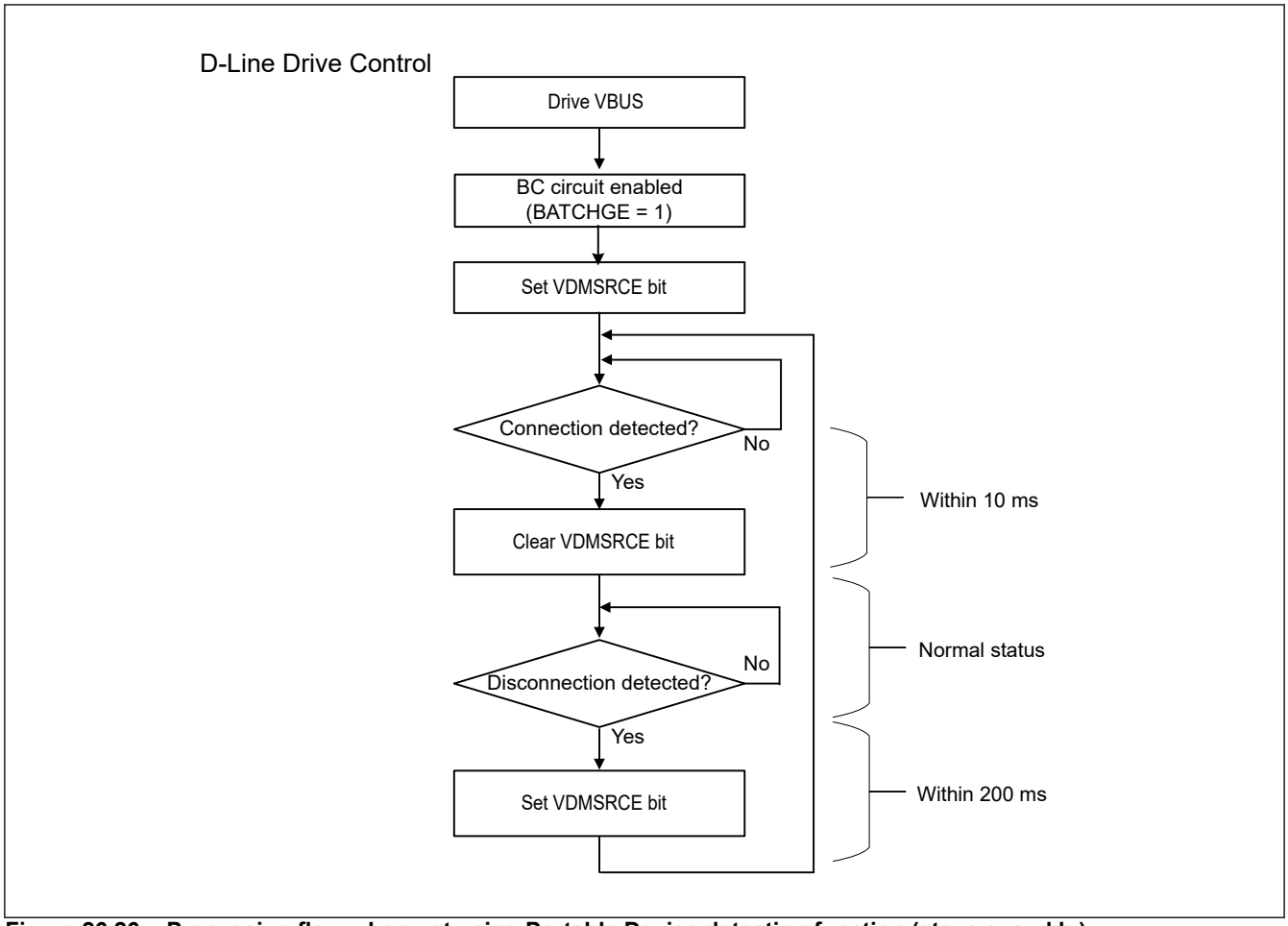



**Figure 26.26 Processing flow when not using Portable Device detection function (steps a. and b.)**

## 26.4 Usage Notes

### 26.4.1 Settings for the Module-Stop State

USBFS operation can be disabled or enabled using Module Stop Control Register B (MSTPCRB). The USBFS is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

### 26.4.2 Clearing the Interrupt Status Register on Canceling Software Standby Mode

Because the input buffer is always enabled in Software Standby mode, an unexpected interrupt might occur under the following conditions:

- When the interrupt is enabled in Normal mode
- When the interrupt is disabled in Software Standby mode
- When the input level of the pin that cancels software standby mode is changed in Software Standby mode

These conditions might cause the associated interrupt flag in the Interrupt Status Register to set unexpectedly. After the MCU cancels the Software Standby mode, the unexpected interrupt might be sent to the interrupt controller. To avoid this, always clear the INTSTS0 and INTSTS1 registers in the canceling sequence.

### 26.4.3 Clearing the Interrupt Status Register after Setting Up the Port Function

The input buffer is disabled before the PmnPFS.PSEL and PmnPFS.PMR port is set up, so the internal signal is fixed high or low. The input buffer is enabled after the port is set so that the external pin state is propagated to the MCU. An unexpected interrupt might occur at this time, causing the VBINT and OVRCR bits in INTSTS0 and INTSTS1, or other

interrupt status flags to set to 1. To avoid a malfunction, always clear the INTSTS0 and INTSTS1 registers after setting up the port.

# 27. Serial Communications Interface (SCI)

## 27.1 Overview

The Serial Communications Interface (SCI) × 6 channels have asynchronous and synchronous serial interfaces:

- Asynchronous interfaces (UART and Asynchronous Communications Interface Adapter (ACIA))
- 8-bit clock synchronous interface
- Simple IIC (master-only)
- Simple SPI
- Smart card interface
- Manchester interface
- Extended Serial interface

The smart card interface complies with the ISO/IEC 7816-3 standard for electronic signals and transmission protocol. SCIn (n = 0, 3, 4, 9) has FIFO buffers to enable continuous and full-duplex communication, and the data transfer speed can be configured independently using an on-chip baud rate generator.

In this section, PCLK refers to PCLKA.

Table 27.1 lists the SCI specifications, Figure 27.1 shows a block diagram of SCI, and Table 27.3 lists the I/O pins.

**Table 27.1 SCI specifications (1 of 3)**

| Parameter | | Specifications |
|---|---|---|
| Number of modules | | 6 (SCIn (n = 0 to 4, 9)) |
| Serial communication modes | | • Asynchronous<br>• Clock synchronous<br>• Simple IIC<br>• Simple SPI<br>• Smart card interface<br>• Manchester interface (SCIn (n = 3, 4))<br>• Extended Serial interface (SCIn (n = 1, 2)) |
| Transfer speed | | Bit rate specifiable with the on-chip baud rate generator |
| Full-duplex communications | | • Transmitter: Continuous transmission possible using double-buffering<br>• Receiver: Continuous reception possible using double-buffering |
| Data transfer | | Selectable as LSB-first or MSB-first transfer |
| Inverter for communication terminals (RXDn, TXDn) | | Selectable inverter for each terminals (RXDn, TXDn) (SCIn (n = 0, 3, 4, 9) |
| Interrupt sources | | Transmit end, transmit data empty, receive data full, receive error, receive data ready*, address match*. (* SCIn (n = 0, 3, 4, 9) support)<br>Completion of generation of a start condition, restart condition, or stop condition. (for simple IIC mode) |
| Module-stop function | | Module-stop state can be set for each channel |
| Snooze end request | | SCI0 address mismatch (SCI0_DCUF) |
| Clock synchronous mode | Data length | 8 bits |
| | Receive error detection | Overrun error |
| | Clock source | Selectable to internal clock (master mode) or external clock (slave mode) |
| | Hardware flow control | Transmission and reception controllable with CTSn_RTSn pins |
| | Transmission and reception | Selectable to 1-stage register or 16-stage FIFO (only SCIn (n = 0, 3, 4, 9) supports FIFO) |
| Asynchronous mode | Data length | 7, 8, or 9 bits |
| | Transmission stop bit | 1 or 2 bits |
| | Adjustment of receive sampling timing | Adjustable receive sampling timing before/after the default timing |

**Table 27.1 SCI specifications (2 of 3)**

| Parameter | | Specifications |
|---|---|---|
| | Adjustment of transmit timing | Adjustable edge timing of transmit waveform controlled by the setting value of registers. |
| | Parity | Even parity, odd parity, or no parity |
| | Receive error detection | • Parity error<br>• Overrun error<br>• Framing error |
| | Hardware flow control | Transmission and reception controllable with CTSn_RTSn pins |
| | Transmission and reception | Selectable to 1-stage register or 16-stage FIFO (SCIn (n = 0, 3, 4, 9) supports FIFO) |
| | Address match | Interrupt request/event output can be issued upon detecting a match between received data and the value in the compare match register (SCIn (n = 0, 3, 4, 9) support) |
| | Address mismatch (SCI0 only) receive data | Snooze end request can be issued when detecting a mismatch between the received data and the value in the compare match register |
| | Start-bit detection | Selectable to low level or falling edge detection |
| | Break detection | Breaks from framing errors detectable by read from SPTR register |
| | Clock source | Selectable to internal or external clock<br>Transfer rate clock input from the GPT can be used. (SCIn (n = 1, 2)) |
| | Double-speed mode | Baud rate generator double-speed mode is selectable |
| | Multi-processor communications function | Serial communication enabled among multiple processors |
| | Noise cancellation | Digital noise filters included on signal paths from the RXDn pin inputs |
| Smart card interface mode | Error processing | Error signal can be automatically transmitted upon detecting a parity error during reception |
| | | Data can be automatically retransmitted upon receiving an error signal during transmission |
| | Data type | Both direct and inverse convention supported |
| Manchester mode | Communication format | Manchester code with the preface and the Start Bit added |
| | Data length | 7,8, or 9 bits |
| | Transmission stop bit | 1 or 2 bits |
| | Parity function | Even parity, odd parity, or no parity |
| | Receive error detection | Parity, overrun, framing, Manchester errors |
| | Hardware flow control | CTSn and RTSn pins can be used in controlling transmission |
| | Clock source | Only internal clock can be used. |
| | Double-speed mode | Baud rate generator double-speed mode is selectable |
| | Multi-processor communication function | Serial communication among multiple processors |
| | Noise cancellation | The signal paths from input on the RXDn pins incorporate digital noise filters |
| | Preface setting / detection function | The function outputs the configured the preface pattern and detects it. |
| | Start Bit setting / detection function | The function outputs the configured the Start Bit pattern and detects it. |
| | Reception retiming function | Timing correction is performed for each bit of the received signal |
| Simple IIC mode | Transfer format | $I^2C$ bus format (MSB-first only) |
| | Operating mode | Master (single-master operation only) |
| | Transfer rate | Up to 400 kbps |

**Table 27.1 SCI specifications (3 of 3)**

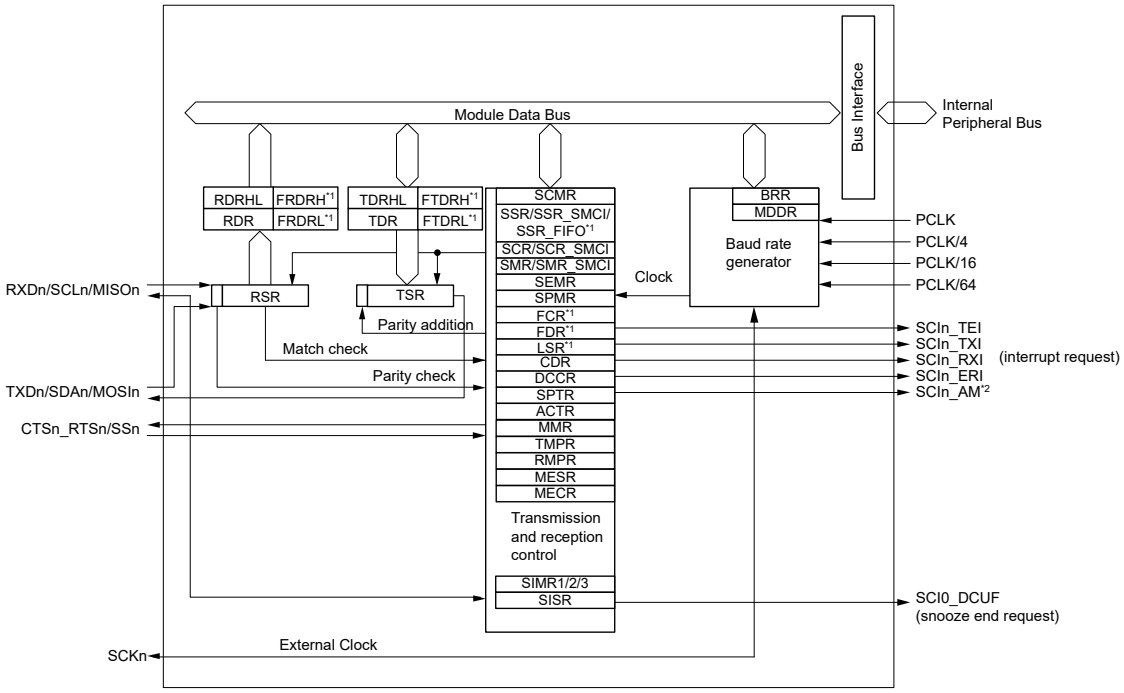
| Parameter | | Specifications |
|---|---|---|
| | Noise cancellation | The signal paths from input on the SCLn and SDAn pins incorporate digital noise filters and provide an adjustable interval for noise cancellation |
| Simple SPI mode | Data length | 8 bits |
| | Error detection | Overrun error |
| | Clock source | Selectable to internal clock (master mode) or external clock (slave mode) |
| | SSn input pin function | High impedance state can be invoked on the output pins by driving the SSn pin high. |
| | Clock settings | Configurable among four clock phase and clock polarity settings |
| Extended serial mode | Start Frame transmission | • Output of a low level as the Break Field over a specified width and generation of interrupts on completion<br>• Detection of bus collisions and the generation of interrupts on detection |
| | Start Frame reception | • Detection of the Break Field low width and generation of an interrupt on detection<br>• Comparison of Control Fields 0 and 1 and generation of an interrupt when the two match<br>• Two kinds of data for comparison (primary and secondary) can be set in Control Field 1.<br>• A priority interrupt bit can be set in Control Field 1.<br>• Handling of Start Frames that do not include a Break Field<br>• Handling of Start Frames that do not include a Control Field 0<br>• Function for measuring bit rates |
| | I/O control function | • Selectable polarity for TXDXn and RXDXn signals<br>• Selection of a digital filter for the RXDXn signal<br>• Half-duplex operation employing RXDXn and TXDXn signals multiplexed on the same pin<br>• Selectable timing for the sampling of data received through RXDXn<br>• Signals received on RXDXn can be passed though to SCIn (n = 1, 2) when the extended serial mode control section is off. |
| | Timer function | • Usable as a reloading timer |
| Bit rate modulation function | | Error reduction through correction of outputs from the on-chip baud rate generator |
| Event link function | | Error event output for receive error or error signal detection (SCIn_ERI) (n = 0 to 4, 9) |
| | | Receive data full event output (SCIn_RXI) (n = 0 to 4, 9) |
| | | Transmit data empty event output (SCIn_TXI) (n = 0 to 4, 9) |
| | | Address match event output (SCIn_AM) (n = 0, 3, 4, 9) |
| | | Transmit end event output (SCIn_TEI) (n = 0, 3, 4, 9) |
| TrustZone Filter | | Security attribution can be set for each channels |

**Table 27.2 Functions of SCI Channel (1 of 2)**

| Item | SCI0, SCI5 to SCI9 | SCI3, SCI4 | SCI1, SCI2 |
|---|---|---|---|
| Asynchronous mode | Available | Available | Available |
| Clock synchronous mode | Available | Available | Available |
| Smart card interface mode | Available | Available | Available |
| Simple I2C mode | Available | Available | Available |
| Simple SPI mode | Available | Available | Available |
| FIFO mode | Available | Available | Not Available |
| Address match | Available | Available | Not Available |
| Manchester mode | Not Available | Available | Not Available |
| Extended serial mode | Not Available | Not Available | Available |

**Table 27.2 Functions of SCI Channel (2 of 2)**

| Item | SCI0, SCI5 to SCI9 | SCI3, SCI4 | SCI1, SCI2 |
|---|---|---|---|
| GPT clock input | Not Available | Not Available | Available |




Note: n = 0 to 4, 9
Note 1. SCIn (n = 0, 3, 4, 9) only
Note 2. n = 0, 3, 4, 9

**Figure 27.1 SCI block diagram**

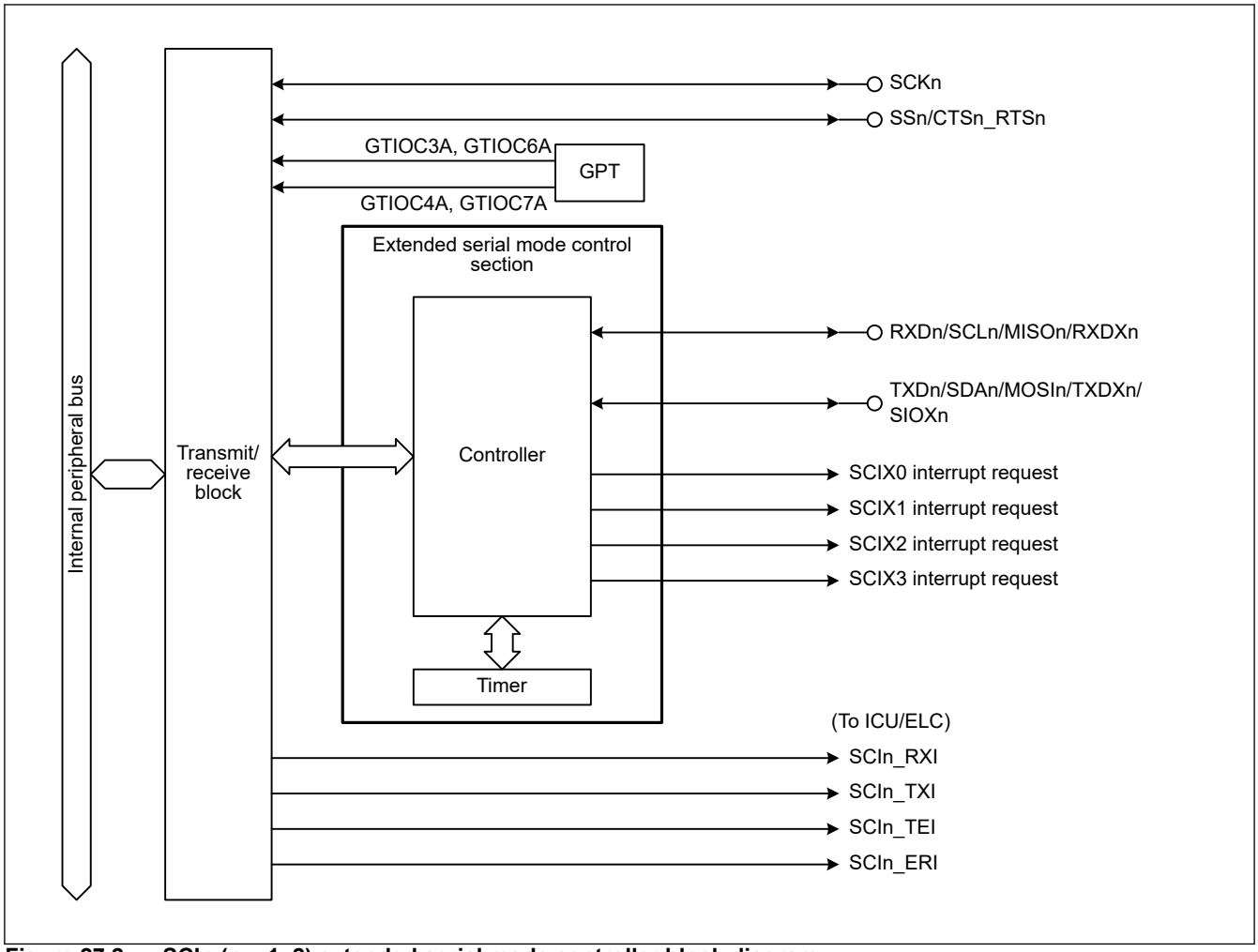



**Figure 27.2 SCIn (n = 1, 2) extended serial mode controller block diagram**

**Table 27.3 SCI I/O pins**

| Function | Pin name | Input/ Output | Description |
|---|---|---|---|
| SCIn (n = 0 to 4, 9) | RXDn/SCLn/MISOn | Input/ Output | SCIn receive data input<br>SCIn $I^2C$ clock input/output<br>SCIn slave transmit data input/output |
| | TXDn/SDAn/MOSIn | Input/ Output | SCIn transmit data output<br>SCIn $I^2C$ data input/output<br>SCIn master transmit data input/output |
| | SSn/CTSn_RTSn | Input/ Output | SCIn chip select input, active-low<br>SCIn transfer start control input/output, active-low |
| | CTSn (n = 0, 3, 4, 9) | Input | SCIn transfer start control input, active-low |
| | RXDXn (n = 1, 2) | Input/ Output | SCIn receive data input (Extended Serial Mode) |
| | TXDXn/SIOXn (n = 1, 2) | Input/ Output | SCIn transmit data output (Extended Serial Mode)<br>SCIn transmit data input/output (Extended Serial Mode) |
| | SCKn | Input/ Output | SCIn clock input/output |

## 27.2 Register Descriptions

### 27.2.1 RSR : Receive Shift Register

RSR is a shift register that receives serial data input from the RXDn pin and converts it into parallel data. When one frame of data is received, the data is automatically transferred to the RDR, RDRHL, or the receive FIFO register. The RSR register cannot be directly accessed by the CPU.

### 27.2.2 RDR : Receive Data Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x05

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

RDR is an 8-bit register that stores received data. When one frame of serial data is received, it is transferred from RSR to RDR, and the RSR register can receive more data. Because RSR and RDR function as a double buffer, continuous received operations can be performed.

Read the RDR only once after a receive data full interrupt (SCIn_RXI) occurs.

Note: If the next frame of data is received before reading the received data from RDR, an overrun error occurs. The CPU cannot write to the RDR.

### 27.2.3 RDRHL : Receive Data Register for Non-Manchester mode (MMR.MANEN = 0)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x10

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | RDAT[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | RDAT[8:0] | Serial Receive Data | R |
| 15:9 | — | These bits are read as 0. | R |

RDRHL is a 16-bit register that stores received data. Use this register when asynchronous mode and 9-bit data length are selected.

The lower 8 bits of RDRHL are the shadow register of RDR, so access to RDRHL affects the RDR register. Access to the RDRHL register is prohibited if 7-bit or 8-bit data length is selected.

After one frame of data is received, the received data is transferred from the RSR register to the RDR/RDRHL registers, allowing the RSR register to receive more data.

The RSR and RDRHL registers form a double-buffered structure to enable continuous reception. RDRHL should be read only when a receive data full interrupt (SCIn_RXI) request is issued. An overrun error occurs when the next frame of data is received before the received data is read from RDRHL. The CPU cannot write to the RDRHL register.

## 27.2.4 RDRHL_MAN : Receive Data Register for Manchester mode (MMR.MANEN = 1)

Base address: SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x10

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | RSYNC | — | — | MPB | RDAT[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | RDAT[8:0] | Serial receive data<br>It can read serial receive data | R |
| 9 | MPB | Multi-processor bit<br>It can read multi-processor bit corresponded to serial receive data (RDATA[8:0])<br>0: Data transmission cycles<br>1: ID transmission cycles | R |
| 11:10 | — | These bits are read as 0. The write value should be 0. | R |
| 12 | RSYNC | Receive SYNC data bit<br>It is valid when MMR.SBSEL = 1 in Manchester mode, 0 is read otherwise.<br>0: The received the Start Bit is DATA SYNC<br>1: The received the Start Bit is COMMAND SYNC | R |
| 15:13 | — | These bits are read as 0. The write value should be 0. | R |

RDRHL_MAN is a 16-bit register that stores received data. Use this register when asynchronous mode and 9-bit data length are selected. The lower 8 bits of RDRHL_MAN are the shadow register of RDR, so access to RDRHL_MAN affects the RDR register. Access to the RDRHL_MAN register is prohibited if 7-bit or 8-bit data length is selected.

After one frame of data is received, the received data is transferred from the RSR register to the RDR/RDRHL_MAN registers, allowing the RSR register to receive more data.

The RSR and RDRHL_MAN registers form a double-buffered structure to enable continuous reception.

RDRHL_MAN should be read only when a receive data full interrupt (SCIn_RXI) request is issued. An overrun error occurs when the next frame of data is received before the received data is read from RDRHL_MAN.

The CPU cannot write to the RDRHL_MAN register.

### RDAT[8:0] bit (Serial receive data)

It can read serial receive data.

### MPB bit (Multi-processor bit)

Holds the value of the multi-processor bit in the reception frame. This bit does not change when the SCR.RE bit is 0.

### RSYNC bit (Receive SYNC data bit)

When Manchester mode and MMR.SBSEL = 1, this bit indicates the type of SYNC of the received the Start Bit. For other settings, it is fixed to 0.

## 27.2.5 FRDRHL/FRDRH/FRDRL : Receive FIFO Data Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x10 (FRDRHL/FRDRH)
0x11 (FRDRL)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | RDF | ORER | FER | PER | DR | MPB | RDAT[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | RDAT[8:0] | Serial receive data<br>Stores the serial receive data.<br>Valid only in asynchronous mode, including multi-processor mode, and clock synchronous mode, and with FIFO selected. | R |
| 9 | MPB | Multi-Processor Bit Flag<br>Stores the value of the multi-processor bit in the serial receive data, RDAT[8:0]. Valid only in asynchronous mode with SMR.MP = 1, and with FIFO selected.<br>0: Data transmission cycle<br>1: ID transmission cycle | R |
| 10 | DR | Receive Data Ready Flag<br>This flag is the same as SSR_FIFO.DR.<br>0: Receiving is in progress, or no received data remains in the FRDRH and FRDRL registers after successfully completed reception<br>1: Next receive data is not received for a period after successfully completed reception | R*1 |
| 11 | PER | Parity Error Flag<br>0: No parity error occurred in the first data of FRDRH and FRDRL<br>1: Parity error occurred in the first data of FRDRH and FRDRL | R |
| 12 | FER | Framing Error Flag<br>0: No framing error occurred in the first data of FRDRH and FRDRL<br>1: Framing error occurred in the first data of FRDRH and FRDRL | R |
| 13 | ORER | Overrun Error Flag<br>This flag is the same as SSR_FIFO.ORER.<br>0: No overrun error occurred<br>1: Overrun error occurred | R*1 |
| 14 | RDF | Receive FIFO Data Full Flag<br>This flag is the same as SSR_FIFO.RDF.<br>0: The amount of receive data written in FRDRH and FRDRL is less than the specified receive triggering number<br>1: The amount of receive data written in FRDRH and FRDRL is equal to or greater than the specified receive triggering number | R*1 |
| 15 | — | This bit is read as 0. | R |

Note 1. If this flag is read, it indicates the same value as that read from the SSR_FIFO register. Write 0 to the SSR_FIFO register to clear the flag.

FRDRHL is a 16-bit register that consists of the 8-bit FRDRH and FRDRL registers. FRDRH is assigned to the FRDRHL[15:8] bits, and allocated to the same address as FRDRHL. FRDRL is assigned to the FRDRHL[7:0] bits, and allocated to (the address of FRDRHL + 1) address.

FRDRH and FRDRL constitute a 16-stage FIFO register that stores serial receive data and related status information readable by software. This register is only valid in asynchronous mode, including multi-processor mode, or clock synchronous mode.

The SCI completes reception of one frame of serial data by transferring the received data from the Receive Shift Register (RSR) into FRDRH and FRDRL for storage. Continuous reception is executed until 16 stages are stored. If data is read when there is no received data in FRDRH and FRDRL, the value is undefined. When FRDRH and FRDRL are full, subsequent serial receive data is lost. The CPU can read from the FRDRH and FRDRL registers but cannot write to them.

Reading 1 from the RDF, ORER, or DR flags of the FRDRH register is the same as reading from those bits in the SSR_FIFO register. When writing 0 to clear a flag in the SSR_FIFO register after reading the FRDRH register, write 0 only to the flag that is to be cleared and write 1 to the other flags.

When reading both the FRDRH and FRDRL registers, read in order from FRDRH to FRDRL. The FRDRHL register can be accessed in 16-bit units.

## 27.2.6 TDR : Transmit Data Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x03

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | Serial Transmit Data | R/W |

TDR is an 8-bit register that stores transmit data.

When the SCI detects that the TSR register is empty, it transfers the transmit data written in the TDR register to the TSR register and starts transmission.

The double-buffered structure of the TDR and TSR registers enables continuous serial transmission. If the next transmit data is already written to TDR when one frame of data is transmitted, the SCI transfers the written data to the TSR register to continue transmission.

The CPU can read from or write to TDR at any time. Only write transmit data to TDR once after each instance of the transmit data empty interrupt (SCIn_TXI).

## 27.2.7 TDRHL : Transmit Data Register for Non-Manchester mode (MMR.MANEN = 0)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x0E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | TDAT[8:0] | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | TDAT[8:0] | Serial Transmit Data | R/W |
| 15:9 | — | This bit is read as 1. The write value should be 1. | R/W |

TDRHL is a 16-bit register that stores transmit data. Use this register when asynchronous mode and 9-bit data length are selected.

The lower 8 bits of TDRHL are the shadow register of TDR, so access to TDRHL affects the TDR register. Access to the TDRHL register is prohibited if 7-bit or 8-bit data length is selected.

When empty space is detected in the TSR register, the transmit data stored in the TDRHL registers is transferred to TSR and transmission starts.

The TSR and TDRHL registers have a double-buffered structure to support continuous transmission. When the next data to be transmitted is stored in TDRHL after one frame of data is transmitted, the transmitting operation continues by transferring the data to the TSR register.

The CPU can read and write to the TDRHL register. Bits [15:9] in TDRHL are fixed to 1. These bits are read as 1. The write value should be 1.

Write transmit data to theTDRHL register only once when a transmit data empty interrupt (SCIn_TXI) request is issued.

## 27.2.8 TDRHL_MAN : Transmit Data Register for Manchester mode (MMR.MANEN = 1)

Base address: SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x0E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | TSYNC | — | — | MPBT | TDAT[8:0] | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | TDAT[8:0] | Serial transmit data<br>It can set serial transmit data | R/W |
| 9 | MPBT | Multi-processor transfer bit flag<br>Value of the multi-processor bit in the transmission frame<br>0: Data transmission cycles<br>1: ID transmission cycles | R/W |
| 11:10 | — | The write value should be 1. | R |
| 12 | TSYNC | Transmit SYNC data bit<br>It is valid when MMR.SBSEL = 1 and MMR.SYNSEL = 1 in Manchester mode.<br>0: The Start Bit is transmitted as DATA SYNC.<br>1: The Start Bit is transmitted as COMMAND SYNC. | R/W |
| 15:13 | — | The write value should be 1. | R |

TDRHL_MAN is a 16-bit register that stores transmit data. Use this register when asynchronous mode and 9-bit data length are selected.

The lower 8 bits of TDRHL_MAN are the shadow register of TDR, so access to TDRHL_MAN affects the TDR register. Access to the TDRHL_MAN register is prohibited if 7-bit or 8-bit data length is selected.

When empty space is detected in the TSR register, the transmit data stored in the TDRHL_MAN registers is transferred to TSR and transmission starts.

The TSR and TDRHL_MAN registers have a double-buffered structure to support continuous transmission. When the next data to be transmitted is stored in TDRHL_MAN after one frame of data is transmitted, the transmitting operation continues by transferring the data to the TSR register.

Write transmit data to theTDRHL_MAN register only once when a transmit data empty interrupt (SCIn_TXI) request is issued.

### TDAT[8:0] bit (Serial transmit data)

This register sets serial transmission data.

### MPBT bit (Multi-processor transfer bit flag)

Selects the multi processor bit of transmit frame.

### TSYNC bit (Transmit SYNC data bit)

When Manchester mode and MMR.SBSEL = "1" and MMR.SYNSEL = "1",the type of SYNC selected according to this bit becames the Start Bit of the transmission frame.

## 27.2.9 FTDRHL/FTDRH/FTDRL : Transmit FIFO Data Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x0E (FTDRHL/FTDRH)
0x0F (FTDRL)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | MPBT | TDAT[8:0] | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | TDAT[8:0] | Serial transmit data<br>Specifies the serial transmit data.<br>Valid only in asynchronous mode, including multi-processor mode, and clock synchronous mode, and with FIFO selected. | W |
| 9 | MPBT | Multi-Processor Transfer Bit Flag<br>Specifies the multi-processor bit in the transmission frame. Valid only in asynchronous mode and SMR.MP = 1, and with FIFO selected.Valid only in asynchronous mode, including multi-processor mode, and clock synchronous mode, and with FIFO selected.<br>0: Data transmission cycle<br>1: ID transmission cycle | W |
| 15:10 | — | The write value should be 1. | W |

FTDRHL is a 16-bit register that consists of the 8-bit FTDRH and FTDRL registers. FTDRH is assigned to the FTDRHL[15:8] bits, and allocated to the same address as FTDRHL. FTDRL is assigned to the FTDRHL[7:0] bits, and allocated to (the address of FTDRHL + 1) address.

FTDRH and FTDRL constitute a 16-stage FIFO register that stores data for serial transmission and a multi-processor transfer bit. This register is only valid in asynchronous mode, including multi-processor mode, or clock synchronous mode.

When the SCI detects that the Transmit Shift Register (TSR) is empty, it transfers data written in the FTDRH and FTDRL registers to the TSR register and starts serial transmission. Continuous serial transmission is executed until no transmit data is left in FTDRH and FTDRL. When FTDRHL is full of transmit data, no more data can be written. If writing new data is attempted, the data is ignored. The CPU can write to the FTDRH and FTDRL registers but cannot read them.

When writing to both the FTDRH and FTDRL registers, write in order from FTDRH to FTDRL.

### TDAT[8:0] bits (Serial transmit data)

The TDAT[8:0] bits set the serial transmission data. This is valid only when FIFO is selected in asynchronous mode (including multiprocessor) or clock synchronous mode.

### MPBT flag (Multi-Processor Transfer Bit Flag)

The MPBT flag specifies the value of the multi-processor bit of the transmit frame. When FCR.FM = 1, SSR.MPBT is invalid.

## 27.2.10 TSR : Transmit Shift Register

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI first automatically transfers transmit data from TDR, TDRHL, or transmit FIFO to TSR, then sends the data to the TXDn pin. The CPU cannot directly access the TSR.

## 27.2.11 SMR : Serial Mode Register for Non-Smart Card Interface Mode (SCMR.SMIF = 0)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CM | CHR | PE | PM | STOP | MP | CKS[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | CKS[1:0] | Clock Select<br>0 0: PCLK clock (n = 0)*1<br>0 1: PCLK/4 clock (n = 1)*1<br>1 0: PCLK/16 clock (n = 2)*1<br>1 1: PCLK/64 clock (n = 3)*1 | R/W*4 |
| 2 | MP | Multi-Processor Mode<br>Valid only in asynchronous mode.<br>0: Disable multi-processor communications function<br>1: Enable multi-processor communications function | R/W*4 |
| 3 | STOP | Stop Bit Length<br>Valid only in asynchronous mode.<br>0: 1 stop bit<br>1: 2 stop bits | R/W*4 |
| 4 | PM | Parity Mode<br>Valid only when the PE bit is 1.<br>0: Even parity<br>1: Odd parity | R/W*4 |
| 5 | PE | Parity Enable<br>Valid only in asynchronous mode.<br>0: When transmitting: Do not add parity bit<br>When receiving: Do not check parity bit<br>1: When transmitting: Add parity bit<br>When receiving: Check parity bit | R/W*4 |
| 6 | CHR | Character Length<br>Valid only in asynchronous mode.*2<br>Selects the transmit/receive character length in combination with the SCMR.CHR1 bit.<br>0: SCMR.CHR1 = 0: Transmit/receive in 9-bit data length<br>SCMR.CHR1 = 1: Transmit/receive in 8-bit data length (initial value)<br>1: SCMR.CHR1 = 0: Transmit/receive in 9-bit data length<br>SCMR.CHR1 = 1: Transmit/receive in 7-bit data length*3 | R/W*4 |
| 7 | CM | Communication Mode<br>0: Asynchronous mode or simple IIC mode<br>1: Clock synchronous mode or simple SPI mode | R/W*4 |

Note 1. n is the decimal notation of the value of n in the BRR register. See section 27.2.20. BRR : Bit Rate Register.
Note 2. In any mode other than asynchronous mode, this bit setting is invalid and a fixed data length of 8 bits is used.
Note 3. LSB-first is fixed and the MSB (bit [7]) in the TDR register is not transmitted in transmit mode.
Note 4. Writable only when SCR.TE = 0 and SCR.RE = 0 (both serial transmission and reception are disabled).

The SMR register sets the communication format and clock source for the on-chip baud rate generator.

### CKS[1:0] bits (Clock Select)

The CKS[1:0] bits select the clock source for the on-chip baud rate generator. For the relationship between the settings of these bits and the baud rate, see section 27.2.20. BRR : Bit Rate Register.

### MP bit (Multi-Processor Mode)

The MP bit disables or enables the multi-processor communications function. The PE and PM bit settings are invalid in multi-processor mode.

### STOP bit (Stop Bit Length)

The STOP bit selects the stop bit length in transmission.

In reception, only the first stop bit is checked regardless of this bit setting. If the second stop bit is 0, it is treated as the start bit of the next transmit frame.

### PM bit (Parity Mode)

The PM bit selects the parity mode (even or odd) for transmission and reception. The PM bit setting is invalid in multiprocessor mode.

### PE bit (Parity Enable)

When the PE bit is set to 1, the parity bit is added to transmit data, and the parity bit is checked in reception. Regardless of the PE bit setting, the parity bit is not added or checked in multi-processor format.

### CHR bit (Character Length)

The CHR bit selects the data length for transmission and reception in combination with the SCMR.CHR1 bit. In modes other than asynchronous, a fixed data length of 8 bits is used.

### CM bit (Communication Mode)

The CM bit selects the communication mode:

- Asynchronous mode or simple IIC mode
- Clock synchronous mode or simple SPI mode

## 27.2.12 SMR_SMCI : Serial Mode Register for Smart Card Interface Mode (SCMR.SMIF = 1)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | GM | BLK | PE | PM | BCP[1:0] | | CKS[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | CKS[1:0] | Clock Select<br>0 0: PCLK clock (n = 0)*1<br>0 1: PCLK/4 clock (n = 1)*1<br>1 0: PCLK/16 clock (n = 2)*1<br>1 1: PCLK/64 clock (n = 3)*1 | R/W*2 |
| 3:2 | BCP[1:0] | Base Clock Pulse<br>Selects the number of base clock cycles in combination with the SCMR.BCP2 bit. Table 27.4 lists the combinations of the SCMR.BCP2 and SMR.BCP[1:0] bits. | R/W*2 |
| 4 | PM | Parity Mode<br>Valid only when the PE bit is 1.<br>0: Even parity<br>1: Odd parity | R/W*2 |
| 5 | PE | Parity Enable<br>When this bit is set to 1, a parity bit is added to transmit data, and the parity of received data is checked. Set this bit to 1 in smart card interface mode. | R/W*2 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6 | BLK | Block Transfer Mode<br>0: Normal mode operation<br>1: Block transfer mode operation | R/W*2 |
| 7 | GM | GSM Mode<br>0: Normal mode operation<br>1: GSM mode operation | R/W*2 |

Note 1. n is the decimal notation of the value of n in the BRR register. See section 27.2.20. BRR : Bit Rate Register.
Note 2. Writable only when SCR_SMCI.TE = 0 and SCR_SMCI.RE = 0 (both serial transmission and reception are disabled).

The SMR_SMCI register sets the communication format and clock source for the on-chip baud rate generator.

### CKS[1:0] bits (Clock Select)

The CKS[1:0] bits select the clock source for the on-chip baud rate generator. For the relationship between the settings of these bits and the baud rate, see section 27.2.20. BRR : Bit Rate Register.

### BCP[1:0] bits (Base Clock Pulse)

The BCP[1:0] bits select the number of base clock cycles in a 1-bit data transfer time in smart card interface mode. Set these bits in combination with the SCMR.BCP2 bit.

For details, see section 27.7.4. Receive Data Sampling Timing and Reception Margin.

**Table 27.4 Combinations of SCMR.BCP2 and SMR_SMCI.BCP[1:0] bits**

| SCMR.BCP2 bit | SMR_SMCI.BCP[1:0] bits | Number of base clock cycles for 1-bit transfer period*1 |
|---|---|---|
| 0 | 00b | 93 clock cycles (S = 93) |
| 0 | 01b | 128 clock cycles (S = 128) |
| 0 | 10b | 186 clock cycles (S = 186) |
| 0 | 11b | 512 clock cycles (S = 512) |
| 1 | 00b | 32 clock cycles (S = 32) (initial value) |
| 1 | 01b | 64 clock cycles (S = 64) |
| 1 | 10b | 372 clock cycles (S = 372) |
| 1 | 11b | 256 clock cycles (S = 256) |

Note 1. S is the value of S in BRR (see section 27.2.20. BRR : Bit Rate Register).

### PM bit (Parity Mode)

The PM bit selects the parity mode for transmission and reception (even or odd). For details on the usage of this bit in smart card interface mode, see section 27.7.2. Data Format (Except in Block Transfer Mode).

### PE bit (Parity Enable)

Set the PE bit to 1. The parity bit is added to transmit data before transmission, and the parity bit is checked in reception.

### BLK bit (Block Transfer Mode)

Setting the BLK bit to 1 enables block transfer mode operation. For details, see section 27.7.3. Block Transfer Mode.

### GM bit (GSM Mode)

Setting the GM bit to 1 enables GSM mode operation. In GSM mode, the SSR_SMCI.TEND flag set timing is moved forward to 11.0 etus (elementary time unit = 1-bit transfer time) from the start bit, and clock output control is added. For details, see section 27.7.6. Serial Data Transmission (Except in Block Transfer Mode) and section 27.7.8. Clock Output Control.

## 27.2.13 SCR : Serial Control Register for Non-Smart Card Interface Mode (SCMR.SMIF = 0)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x02

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TIE | RIE | TE | RE | MPIE | TEIE | CKE[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | CKE[1:0] | Clock Enable<br>0 0: In asynchronous mode, the SCKn pin is available for use as an I/O port based on the I/O port settings.<br>In clock synchronous mode, the SCKn pin functions as the clock output pin.<br>0 1: In asynchronous mode, a clock with the same frequency as the bit rate is output from the SCKn pin.<br>In clock synchronous mode, the SCKn pin functions as the clock output pin.<br>Others: In asynchronous mode, input a clock with a frequency 16 times the bit rate from the SCKn pin when the SEMR.ABCS bit is 0. Input a clock signal with a frequency eight times the bit rate when the SEMR.ABCS bit is 1. The SCKn pin is available for use as an I/O port based on the I/O port settings when the GPT clock*4 is used.<br>In clock synchronous mode, the SCKn pin functions as the clock input pin. | R/W*1 |
| 2 | TEIE | Transmit End Interrupt Enable<br>0: Disable SCIn_TEI interrupt requests<br>1: Enable SCIn_TEI interrupt requests | R/W |
| 3 | MPIE | Multi-Processor Interrupt Enable<br>Valid in asynchronous mode when SMR.MP = 1.<br>0: Normal reception<br>1: When data with the multi-processor bit set to 0 is received, the data is not read, and setting the status flags RDRF, ORER, and FER in SSR to 1 and the status flags SYER, PFER, and SBER in MESR are disabled.<br>When data with the multi-processor bit set to 1 is received, the MPIE bit is automatically set to 0, and normal reception is resumed. | R/W*3 |
| 4 | RE | Receive Enable<br>0: Disable serial reception<br>1: Enable serial reception | R/W*2 |
| 5 | TE | Transmit Enable<br>0: Disable serial transmission<br>1: Enable serial transmission | R/W*2 |
| 6 | RIE | Receive Interrupt Enable<br>0: Disable SCIn_RXI and SCIn_ERI interrupt requests<br>1: Enable SCIn_RXI and SCIn_ERI interrupt requests | R/W |
| 7 | TIE | Transmit Interrupt Enable<br>0: Disable SCIn_TXI interrupt requests<br>1: Enable SCIn_TXI interrupt requests | R/W |

Note 1. Writable only when TE = 0 and RE = 0.
Note 2. 1 can be written only when TE = 0 and RE = 0, when the SMR.CM bit is 1. After setting TE or RE to 1, only 0 can be written to TE and RE. When the SMR.CM bit is 0 and the SIMR1.IICM bit is 0, writing is enabled under any condition.
Note 3. When writing a new value to a bit other than the MPIE bit of this register in multi-processor mode (SMR.MP bit = 1), write 0 to the MPIE bit using the store instruction to avoid accidentally setting the MPIE bit to 1 by a read-modify-write operation when using a bit manipulation instruction.
Note 4. GPT clock is selectable for SCI1 and SCI2.

The SCR register controls operation and clock source selection for transmission and reception.

### CKE[1:0] bits (Clock Enable)

The CKE[1:0] bits select the clock source and the SCKn pin function.

#### TEIE bit (Transmit End Interrupt Enable)

The TEIE bit enables or disables SCIn_TEI interrupt requests. Set TEIE to 0 to disable an SCIn_TEI interrupt request.

In simple IIC mode, SCIn_TEI is allocated to the interrupt on completion of issuing a start, restart, or stop condition (STIn). In this case, the TEIE bit can be used to enable or disable the STI.

#### MPIE bit (Multi-Processor Interrupt Enable)

When the MPIE bit is set to 1 and data with the multi-processor bit set to 0 is received, the data is not read and setting the status flags RDRF, ORER, FER, RDF, and DR in SSR/SSR_FIFO to 1 is disabled. When data with the multi-processor bit set to 1 is received, the MPIE bit is automatically set to 0, and normal reception resumes. For details, see section 27.4. Multi-Processor Communication Function.

When the MPB bit in the SSR register is 0, the receive data is not transferred from the RSR register to the RDR register, a receive error is not detected, and setting the flags ORER and FER to 1 is disabled.

When the MPB bit is set to 1, the MPIE bit is automatically set to 0, SCIn_RXI and SCIn_ERI interrupt requests are enabled (if the RIE bit in SCR is set to 1), and setting of the ORER and FER flags to 1 is enabled.

Set MPIE to 0 if the multi-processor communications function is not used.

#### RE bit (Receive Enable)

The RE bit enables or disables serial reception. When the RE bit is set to 1, serial reception starts by detecting the start bit in asynchronous mode or the synchronous clock input in clock synchronous mode. Set the reception format in the SMR register before setting the RE bit to 1.

In non-FIFO operation, when reception is halted by setting the RE bit to 0, the RDRF, ORER, FER, and PER flags in the SSR register are not affected, and the previous values are retained.

When FIFO operation is selected and reception is halted by setting the RE bit to 0, the RDF, ORER, FER, PER, and DR flags in SSR_FIFO are not affected and the previous values are retained.

#### TE bit (Transmit Enable)

The TE bit enables or disables serial transmission.

When the TE bit is set to 1, serial transmission is started by writing transmit data to the TDR register. Set the transmission format in the SMR register before setting the TE bit to 1.

#### RIE bit (Receive Interrupt Enable)

The RIE bit enables or disables SCIn_RXI and SCIn_ERI interrupt requests.

SCIn_RXI and SCIn_ERI interrupt requests are disabled by setting the RIE bit to 0.

An SCIn_ERI interrupt request can be canceled by reading 1 from the ORER, FER, or PER flag in SSR/SSR_FIFO then setting the flag to 0, or by setting the RIE bit to 0.

#### TIE bit (Transmit Interrupt Enable)

The TIE bit enables or disables SCIn_TXI interrupt requests. SCIn_TXI interrupt requests are disabled by setting the TIE bit to 0.

Note: To switch the TIE bit value from 0 to 1 in FIFO mode, set the TIE and TE bits to 1 simultaneously or set the TIE bit to 1 when TE = 1. When TE = 0 in FIFO mode, setting the TIE bit to 1 is prohibited.

## 27.2.14 SCR_SMCI : Serial Control Register for Smart Card Interface Mode (SCMR.SMIF = 1)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x02

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TIE | RIE | TE | RE | MPIE | TEIE | CKE[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | CKE[1:0] | Clock Enable<br>0 0: When SMR_SMCI.GM = 0: Disable output<br>The SCKn pin is available for use as an I/O port if set up in the I/O port settings<br>When SMR_SMCI.GM = 1: Fix output low<br>0 1: When SMR_SMCI.GM = 0: Output clock<br>When SMR_SMCI.GM = 1: Output clock<br>1 0: When SMR_SMCI.GM = 0: Setting prohibited<br>When SMR_SMCI.GM = 1: Fix output high<br>1 1: When SMR_SMCI.GM = 0: Setting prohibited<br>When SMR_SMCI.GM = 1: Output clock | R/W*1 |
| 2 | TEIE | Transmit End Interrupt Enable<br>Set this bit to 0 in smart card interface mode | R/W |
| 3 | MPIE | Multi-Processor Interrupt Enable<br>Set this bit to 0 in smart card interface mode | R/W |
| 4 | RE | Receive Enable<br>0: Disable serial reception<br>1: Enable serial reception | R/W*2 |
| 5 | TE | Transmit Enable<br>0: Disable serial transmission<br>1: Enable serial transmission | R/W*2 |
| 6 | RIE | Receive Interrupt Enable<br>0: Disable SCIn_RXI and SCIn_ERI interrupt requests<br>1: Enable SCIn_RXI and SCIn_ERI interrupt requests | R/W |
| 7 | TIE | Transmit Interrupt Enable<br>0: Disable SCIn_TXI interrupt requests<br>1: Enable SCIn_TXI interrupt requests | R/W |

Note 1. Writable only when TE = 0 and RE = 0.
Note 2. 1 can be written only when TE = 0 and RE = 0. After setting TE or RE to 1, only 0 can be written to TE and RE.

The SCR_SMCI register sets transmission and reception control, interrupt control, and clock source selection for transmission and reception.

For details on interrupt requests, see section 27.12. Interrupt Sources.

### CKE[1:0] bits (Clock Enable)

The CKE[1:0] bits control the clock output from the SCKn pin. In GSM mode, clock output can be dynamically switched. For details, see section 27.7.8. Clock Output Control.

### TEIE bit (Transmit End Interrupt Enable)

Set the TEIE bit to 0 in smart card interface mode.

### MPIE bit (Multi-Processor Interrupt Enable)

Set the MPIE bit to 0 in smart card interface mode.

### RE bit (Receive Enable)

The RE bit enables or disables serial reception. When the RE bit is set to 1, serial reception starts by detecting the start bit. Set the reception format in the SMR_SMCI register before setting the RE bit to 1.

If reception is halted by setting the RE bit to 0, the ORER, FER, and PER flags in SSR_SMCI are not affected and the previous values are retained.

### TE bit (Transmit Enable)

The TE bit enables or disables serial transmission. When the TE bit is set to 1, serial transmission is started by writing transmit data to TDR. Set the transmission format in the SMR_SMCI register before setting the TE bit to 1.

### RIE bit (Receive Interrupt Enable)

The RIE bit enables or disables SCIn_RXI and SCIn_ERI interrupt requests.

SCIn_RXI and SCIn_ERI interrupt requests are disabled by setting the RIE bit to 0.

An SCIn_ERI interrupt request can be canceled by reading 1 from the ORER, FER, or PER flag in the SSR_SMCI register, and then setting the flag to 0, or by setting the RIE bit to 0.

### TIE bit (Transmit Interrupt Enable)

The TIE bit enables or disables SCIn_TXI interrupt requests. SCIn_TXI interrupt requests are disabled by setting the TIE bit to 0.

## 27.2.15 SSR : Serial Status Register for Non-Smart Card Interface and Non-FIFO Mode (SCMR.SMIF = 0, FCR.FM = 0, and MMR.MANEN = 0)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MPBT | Multi-Processor Bit Transfer<br>Sets the value of the multi-processor bit in the transmission frame.<br>0: Data transmission cycle<br>1: ID transmission cycle | R/W |
| 1 | MPB | Multi-Processor<br>Value of the multi-processor bit in the reception frame.<br>0: Data transmission cycle<br>1: ID transmission cycle | R |
| 2 | TEND | Transmit End Flag<br>0: A character is being transmitted<br>1: Character transfer is complete | R |
| 3 | PER | Parity Error Flag<br>0: No parity error occurred<br>1: Parity error occurred | R/(W)*1 |
| 4 | FER | Framing Error Flag<br>0: No framing error occurred<br>1: Framing error occurred | R/(W)*1 |
| 5 | ORER | Overrun Error Flag<br>0: No overrun error occurred<br>1: Overrun error occurred | R/(W)*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6 | RDRF | Receive Data Full Flag<br>0: No received data in RDR register<br>1: Received data in RDR register | R/(W)*1 |
| 7 | TDRE | Transmit Data Empty Flag<br>0: Transmit data in TDR register<br>1: No transmit data in TDR register | R/(W)*1 |

Note 1. Only 0 can be written to clear the flag after reading 1.

The SSR register provides SCI status flags and transmission and reception multi-processor bits.

### MPBT bit (Multi-Processor Bit Transfer)

The MPBT bit sets the value of the multi-processor bit in the transmit frame.

### MPB bit (Multi-Processor)

The MPB bit holds the value of the multi-processor bit in the reception frame. This bit does not change when the SCR.RE bit is 0.

### TEND flag (Transmit End Flag)

The TEND flag indicates completion of transmission.

[Setting conditions]

- When the SCR.TE bit is set to 0 (serial transmission is disabled) and the FCR.FM bit is set to 0 (non-FIFO selected). When the SCR.TE bit is set to 1, the TEND flag is not affected and retains the value 1.
- When the TDR register is not updated on transmission of the tail-end bit of a character being transmitted.

[Clearing conditions]

- When transmit data is written to the TDR register while the SCR.TE bit is 1
- When 0 is written to TDRE after reading TDRE = 1 while the SCR.TE bit is 1

### PER flag (Parity Error Flag)

The PER flag indicates that a parity error occurred during reception in asynchronous mode and the reception ended abnormally.

[Setting condition]

- When a parity error is detected during reception in asynchronous mode when the address match function is disabled (DCCR.DCME = 0).

  Although receive data is transferred to the RDR register when the parity error occurs, no SCIn_RXI interrupt request occurs. When the PER flag is set to 1, the subsequent receive data is not transferred to the RDR register.

[Clearing condition]

- When 0 is written to PER after reading PER = 1. After writing 0 to the PER flag, read the PER flag to check that it is actually set to 0.

When the SCR.RE bit is set to 0 (serial reception is disabled), the PER flag is not affected and retains its previous value.

### FER flag (Framing Error Flag)

The FER flag indicates that a framing error occurred during reception in asynchronous mode and the reception ended abnormally.

[Setting condition]

- When 0 is sampled as the stop bit during reception in asynchronous mode when the address match function is disabled (DCCR.DCME = 0).

  In 2-stop-bit mode, only the first stop bit is checked. The second stop bit is not checked. Although receive data is transferred to the RDR register when the framing error occurs, no SCIn_RXI interrupt request occurs. When the FER flag is to 1, the subsequent receive data is not transferred to the RDR register.

[Clearing condition]

- When 0 is written to FER after reading FER = 1. After writing 0 to the FER flag, read the FER flag to check that it is actually set to 0.

When the SCR.RE bit is set to 0 (serial reception is disabled), the FER flag is not affected and retains its previous value.

#### ORER flag (Overrun Error Flag)

The ORER flag indicates that an overrun error occurred during reception and the reception ended abnormally.

[Setting condition]

- When the next data is received before receive data that does not have a parity error and a framing error is read from the RDR register.

  The data received before an overrun error occurred is saved in the RDR register, but data received after the error is lost. When the ORER flag is set to 1, receive data is not forwarded to the RDR register. In clock synchronous mode, serial transmission and reception are stopped.

[Clearing condition]

- When 0 is written to ORER after reading ORER = 1. After writing 0 to the ORER flag, read the ORER flag to check that it is actually set to 0.

When the SCR.RE bit is set to 0 (serial reception is disabled), the ORER flag is not affected and retains its previous value.

#### RDRF flag (Receive Data Full Flag)

The RDRF flag indicates the presence of receive data in the RDR register.

[Setting condition]

- When the reception ends normally, and receive data is forwarded from the RSR register to the RDR register.

[Clearing conditions]

- When 0 is written to RDRF after reading RDRF = 1
- When data is forwarded from the RDR register

#### TDRE flag (Transmit Data Empty Flag)

The TDRE flag indicates the presence of transmit data in the TDR register.

[Setting conditions]

- When the SCR.TE bit is 0
- When data is transmitted from the TDR register to the TSR register

[Clearing conditions]

- When 0 is written to TDRE after reading TDRE = 1
- When the SCR.TE bit is 1 and data is written to the TDR register

### 27.2.16 SSR_FIFO : Serial Status Register for Non-Smart Card Interface and FIFO Mode (SCMR.SMIF = 0, FCR.FM = 1, and MMR.MANEN = 0)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TDFE | RDF | ORER | FER | PER | TEND | — | DR |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | x | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DR | Receive Data Ready Flag<br>0: Receiving is in progress, or no received data remains in FRDRHL after successfully completed reception (receive FIFO empty)<br>1: Next receive data is not received for a period after normal receiving is complete, when the amount of data stored in the FIFO is equal to or less than the receive triggering number | R/W*1 |
| 1 | — | The read value is undefined. The write value should be 1. | R/W |
| 2 | TEND | Transmit End Flag<br>0: A character is being transmitted<br>1: Character transfer is complete | R/W*1 |
| 3 | PER | Parity Error Flag<br>0: No parity error occurred<br>1: Parity error occurred | R/W*1 |
| 4 | FER | Framing Error Flag<br>0: No framing error occurred<br>1: Framing error occurred | R/W*1 |
| 5 | ORER | Overrun Error Flag<br>0: No overrun error occurred<br>1: Overrun error occurred | R/W*1 |
| 6 | RDF | Receive FIFO Data Full Flag<br>0: The amount of receive data written in FRDRHL is less than the specified receive triggering number<br>1: The amount of receive data written in FRDRHL is equal to or greater than the specified receive triggering number | R/W*1 |
| 7 | TDFE | Transmit FIFO Data Empty Flag<br>0: The amount of transmit data written in FTDRHL exceeds the specified transmit triggering number<br>1: The amount of transmit data written in FTDRHL is equal to or less than the specified transmit triggering number | R/W*1 |

Note 1. Only 0 can be written, to clear the flag after reading 1.

The SSR_FIFO register provides the SCI with FIFO mode status flags.

### DR flag (Receive Data Ready Flag)

The DR flag indicates that the amount of data stored in the Receive FIFO Data Register (FRDRHL) falls below the specified receive triggering number, and that no next data is received after 15 etus (elementary time units) from the last stop bit in asynchronous mode. This flag is valid only in asynchronous mode, including multi-processor mode, when FIFO operation is selected.

In clock synchronous mode, the DR flag is not set to 1.

[Setting condition]

- When FRDRHL contains less data than the specified receive triggering number, and no next data is received after 15 etus*1 from the last stop bit, and the SSR_FIFO.FER and SSR_FIFO.PER flags are 0.

[Clearing conditions]

- When 1 is read from DR, after all received data is read
- When the FCR.FM bit is changed from 0 to 1

Note 1. This is equivalent to 1.5 frames in the 8-bit format with one stop bit.

The DR flag is only set to 1 when FIFO is selected in asynchronous mode, including multi-processor mode. It is not set to 1 in other operation modes.

### TEND flag (Transmit End Flag)

The TEND flag indicates that FTDRHL does not contain valid data when transmitting the last bit of a serial character, so the transmission is halted.

[Setting condition]

- When FTDRHL does not contain transmit data when the last bit of a 1-byte serial character is transmitted.

[Clearing conditions]

- When transmit data is written to FTDRHL[*1] while the SCR.TE bit is 1
- When 0 is written to TEND after 1 is read from TEND, when the SCR.TE bit is 1
- When the FCR.FM bit is changed from 0 to 1

Note 1. Do not use the TEND bit as a transmit end flag when the DTC writes data to FTDRHL in response to an SCIn_TXI interrupt request.

#### PER flag (Parity Error Flag)

The PER flag indicates whether there is a parity error in the data read from the FRDRHL register in asynchronous mode when the address match function is disabled (DCCR.DCME = 0).

[Setting condition]

- When data is received and a parity error is detected, when the address match function is disabled (DCCR.DCME = 0).

[Clearing condition]

- When 0 is written to PER after reading PER = 1.

The reception operation is continuous, and the receive data is stored in the FRDRHL register, even when a parity error occurs during reception.

When the SCR.RE bit is set to 0 (serial reception is disabled), the PER flag is not affected and retains its previous value.

#### FER flag (Framing Error Flag)

The FER flag indicates whether there is a framing error in the data read from the FRDRHL register in asynchronous mode when the address match function is disabled (DCCR.DCME = 0).

[Setting condition]

- When 0 is sampled as the stop bit during reception when the address match function is disabled (DCCR.DCME = 0).

[Clearing condition]

- When 0 is written to FER after reading FER = 1.

The reception operation is continuous, and the receive data is stored in the FRDRHL register, even when a framing error occurs during reception.

When the SCR.RE bit is set to 0 (serial reception is disabled), the FER flag is not affected and retains its previous value.

#### ORER flag (Overrun Error Flag)

The ORER flag indicates that the receive operation stopped abnormally because an overrun error occurred.

[Setting condition]

- When the next serial reception completes while the receive FIFO is full with 16-byte receive data.

[Clearing condition]

- When 0 is written to ORER after reading ORER = 1.

When the SCR.RE bit is set to 0 (serial reception is disabled), the ORER flag is not affected and retains its previous value.

#### RDF flag (Receive FIFO Data Full Flag)

The RDF flag indicates that receive data was transferred to the FRDRHL register, and the amount of data in FRDRHL is equal to or exceeds the specified receive triggering number. When RTRG is set to 0, the RDF flag is not set even when the amount of data in the receive FIFO is equal to 0.

[Setting condition]

- When the amount of receive data equal to or greater than the specified receive triggering number is stored in FRDRHL,[*1] and the FIFO is not empty.

[Clearing conditions]

- When 0 is written to RDF after reading RDF = 1
- When FRDRHL is read by the DTC, but only when the block transfer is the last transmission
- When the setting and clearing conditions occur at the same time, the RDF bit is set to 0. After that, when the amount of data stored in the FRDRHL register is the same as or greater than the RTRG value, RDF is set to 1 after 1 PCLK.

Note 1. Because FRDRHL is a 16-stage FIFO register, the maximum amount of data that can be read when RDF is 1 is equivalent to the specified receive triggering number. If an attempt is made to read after all the data in FRDRHL is read, the data is undefined.

#### TDFE flag (Transmit FIFO Data Empty Flag)

The TDFE flag indicates that data is transferred from the FTDRHL register into the TSR register, the amount of data in FTDRHL is below the specified transmit triggering number, and writing of transmit data to FTDRHL is enabled.

[Setting conditions]

- When the TE bit in SCR is 0
- When the amount of transmit data written in FTDRHL is equal to or less than the specified transmit triggering number*1

[Clearing conditions]

- When writing to FTDRHL is executed on the last transmission while the DTC is activated
- When 0 is written to the TDFE flag after reading TDFE = 1.*2
  The setting conditions are given priority when TE = 0. When the setting condition and clearing condition occur at the same time, the TDFE flag is set to 0. After that, when the amount of data stored in the FTDRHL register is equal to or less than the TTRG value, TDFE is set to 1 after 1 PCLK.

Note 1. Because the FTDRHL register is a 16-stage FIFO register, when the TDFE flag is 1, the maximum amount of data that can be written to the FTDRHL register is 16 minus FDR.T[4:0] bytes. If more data is written, data is discarded.
Note 2. Do not clear the TDFE flag during block transfer processing by the DTC.

### 27.2.17 SSR_SMCI : Serial Status Register for Smart Card Interface Mode (SCMR.SMIF = 1, and MMR.MANEN = 0)

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MPBT | Multi-Processor Bit Transfer<br>Set this bit to 0 in smart card interface mode | R/W |
| 1 | MPB | Multi-Processor<br>Set this bit to 0 in smart card interface mode | R |
| 2 | TEND | Transmit End Flag<br>0: A character is being transmitted<br>1: Character transfer is complete | R |
| 3 | PER | Parity Error Flag<br>0: No parity error occurred<br>1: Parity error occurred | R/W*1 |
| 4 | ERS | Error Signal Status Flag<br>0: No low error signal response<br>1: Low error signal response occurred | R/W*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5 | ORER | Overrun Error Flag<br>0: No overrun error occurred<br>1: Overrun error occurred | R/W*1 |
| 6 | RDRF | Receive Data Full Flag<br>0: No received data in RDR register<br>1: Received data in RDR register | R/W*1 |
| 7 | TDRE | Transmit Data Empty Flag<br>0: Transmit data in TDR register<br>1: No transmit data in TDR register | R/W*1 |

Note 1. Only 0 can be written, to clear the flag after reading 1.

The SSR_SMCI register provides the SCI with smart card interface mode status flags.

### TEND flag (Transmit End Flag)

When there is no error signal from the receiving side, the TEND flag is set to 1 when more data for transfer is ready to be transferred to the TDR register.

[Setting conditions]

- When the SCR_SMCI.TE bit = 0 (serial transmission is disabled).
  When the SCR_SMCI.TE bit is changed from 0 to 1, the TEND flag is not affected and retains the value 1.
- When a specified period elapses after the latest transmission of 1 byte, the ERS flag is 0, and the TDR register is not updated.

  The set timing is determined by the following register settings:
  - When SMR_SMCI.GM = 0 and SMR_SMCI.BLK = 0, 12.5 etus after the start of transmission
  - When SMR_SMCI.GM = 0 and SMR_SMCI.BLK = 1, 11.5 etus after the start of transmission
  - When SMR_SMCI.GM = 1 and SMR_SMCI.BLK = 0, 11.0 etus after the start of transmission
  - When SMR_SMCI.GM = 1 and SMR_SMCI.BLK = 1, 11.0 etus after the start of transmission

[Clearing conditions]

- When transmit data is written to the TDR register while the SCR_SMCI.TE bit is 1
- When 0 is written to TDRE after reading TDRE = 1 while the SCR_SMCI.TE bit is 1

### PER flag (Parity Error Flag)

The PER flag indicates that a parity error occurred during reception in asynchronous mode and the reception ended abnormally.

[Setting condition]

- When a parity error is detected during reception. Although receive data is transferred to RDR when a parity error occurs, no SCIn_RXI interrupt request occurs. After the PER flag is set to 1, the subsequent receive data is not transferred to RDR.

[Clearing condition]

- When 0 is written to PER after reading PER = 1. After writing 0 to the PER flag, read the flag to check that it is actually set to 0.

When the RE bit in SCR_SMCI is set to 0 (serial reception is disabled), the PER flag is not affected and retains its previous value.

### ERS flag (Error Signal Status Flag)

[Setting condition]

- When a low error signal is sampled.

[Clearing condition]

- When 0 is written to ERS after reading ERS = 1.

#### ORER flag (Overrun Error Flag)

The ORER flag indicates that an overrun error occurred during reception and the reception ended abnormally.

[Setting condition]

- When the next data is received before receive data that does not have a parity error is read from the RDR register. The data received before an overrun error occurred is saved in the RDR, but data received after the error is lost. When the ORER flag is set to 1, receive data is not forwarded to the RDR register.

[Clearing condition]

- When 0 is written to ORER after reading ORER = 1. After writing 0 to the ORER flag, read the flag to check that it is actually set to 0.

When the RE bit in SCR_SMCI is set to 0, the ORER flag is not affected and retains its previous value.

#### RDRF flag (Receive Data Full Flag)

The RDRF flag indicates the presence of receive data in the RDR register.

[Setting condition]

- When the reception ends normally, and receive data is forwarded from the RSR register to the RDR register.

[Clearing conditions]

- When 0 is written to RDRF after reading RDRF = 1
- When data is forwarded from the RDR register

#### TDRE flag (Transmit Data Empty Flag)

The TDRE flag indicates the presence of transmit data in the TDR register.

[Setting conditions]

- When the SCR_SMCI.TE bit is 0
- When data is transmitted from the TDR register to the TSR register

[Clearing conditions]

- When 0 is written to TDRE after reading TDRE = 1
- When the SCR_SMCI.TE bit is 1 and data is written to the TDR register

### 27.2.18 SSR_MANC : Serial Status Register for Manchester Mode (SCMR.SMIF = 0, and MMR.MANEN = 1)

Base address: SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MER |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MER | Manchester Error Flag<br>Valid for Manchester mode only<br>0: No Manchester error occurred<br>1: Manchester error has occurred | R/(W)*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | MPB | Multi-Processor<br>Value of the multi-processor bit in the reception frame<br>0: Data transmission cycles<br>1: ID transmission cycles | R |
| 2 | TEND | Transmit End Flag<br>0: A character is being transmitted<br>1: Character transfer has been completed. | R |
| 3 | PER | Parity Error Flag<br>0: No parity error occurred<br>1: A parity error has occurred | R/(W)*1 |
| 4 | FER | Framing Error Flag<br>0: No framing error occurred<br>1: A framing error has occurred | R/(W)*1 |
| 5 | ORER | Overrun Error Flag<br>0: No overrun error occurred<br>1: An overrun error has occurred | R/(W)*1 |
| 6 | RDRF | Receive Data Full Flag<br>0: No received data is in RDR register<br>1: Received data is in RDR register | R/(W)*1 |
| 7 | TDRE | Transmit Data Empty Flag<br>0: Transmit data is in TDR register<br>1: No transmit data is in TDR register | R/(W)*1 |

Note 1. Only 0 can be written to this bit, to clear the flag after confirmed(read) the flag is set to 1.

SSR is constructed in the status flag of SCI and reception multi processor bits.

### MER flag (Manchester Error Flag)

When data is received in Manchester mode, Manchester error is detected and it is displayed.

[Setting conditions]

- When receiving in Manchester mode and detecting Manchester code error in data area of received frame.
  Received data when an error occurs is transferred to the RDR register, but the RXI interrupt request is not generatedand the ERI interrupt request is generated.
  When the Manchester error flag is set to “1”, subsequent receive data is not transferred to the RDR register.
  For details on Manchester error, see section 27.5.11. Errors in Manchester Mode.

[Clearing conditions]

- When 0 is written to MER after reading MER = 1 (after writing 0 to it, read the MER bit to check that it has actually been set to 0.)
  Even when the RE bit in SCR is set to 0 (serial reception is disabled), the MER flag is not affected and retains its previous value.

### MPB flag (Multi-Processor)

Holds the value of the multi-processor bit in the reception frame. This bit does not change when the SCR.RE bit is 0.

### TEND flag (Transmit End Flag)

Indicates completion of transmission.

[Setting conditions]

- When the SCR.TE bit is set to 0 (serial transmission is disabled) and FCR.FM bit is set to 0 (non-FIFO selected).
  When the SCR.TE bit is changed from 0 to 1, the TEND flag is not affected and retains the value 1.
- When the TDR register is not updated at the time of transmission of the tail-end bit of a character being transmitted

[Clearing conditions]

- When transmit data are written to the TDR register while the SCR.TE bit is 1.
- When 0 is written to TDRE after reading TDRE = 1 while the SCR.TE bit is 1.

### PER flag (Parity Error Flag)

Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally.

[Setting condition]

- When a parity error is detected during reception in Asynchronous Mode and the state of Address Match function invalidity (DCCR.DCME = 0).
  Although receive data when the parity error occurs is transferred to RDR, no RXI interrupt request occurs. Note that when the PER flag is being set to 1, the subsequent receive data is not transferred to RDR.

[Clearing condition]

- When 0 is written to PER after reading PER = 1 (after writing 0 to it, read the PER bit to check that it has actually been set to 0.)

Even when the RE bit in SCR is set to 0 (serial reception is disabled), the PER flag is not affected and retains its previous value.

### FER flag (Framing Error Flag)

Indicates that a framing error has occurred during reception in asynchronous mode and the reception ends abnormally.

[Setting condition]

- When 0 is sampled as the stop bit during reception in Asynchronous Mode and the state of Address Match function invalidity (DCCR.DCME = 0).
  In 2-stop-bit mode, only the first stop bit is checked whether it is 1 but the second stop bit is not checked. Note that although receive data when the framing error occurs is transferred to RDR, no RXI interrupt request occurs. In addition, when the FER flag is being set to 1, the subsequent receive data is not transferred to RDR.

[Clearing condition]

- When 0 is written to FER after reading FER = 1. (After writing 0 to it, read the FER bit to check that it has actually been set to 0.)

Even when the RE bit in SCR is set to 0, the FER flag is not affected and retains its previous value.

### ORER flag (Overrun Error Flag)

Indicates that an overrun error has occurred during reception and the reception ends abnormally.

[Setting condition]

- When the next data is received before receive data is read from RDR which don’t have any valid reception error.
  In RDR, receive data prior to an overrun error occurrence is retained, but data received after the overrun error occurrence is lost. When the ORER flag is set to 1, reception data isn’t forwarded to RDR register. Note that, in clock synchronous mode, serial transmission and reception will be stop.

[Clearing condition]

- When 0 is written to ORER after reading ORER = 1 (after writing 0 to it, read the ORER bit to check that it has actually been set to 0.)

Even when the RE bit in SCR is set to 0, the ORER flag is not affected and retains its previous value.

### RDRF flag (Receive Data Full Flag)

Indicates the presence of receive data in the RDR register.

[Setting condition]

- When the reception ends normally, and receive data is forwarded from the RSR register to the RDR register

[Clearing conditions]

- When it’s written to “0" after the state of “1" is read
- When it’s read the data from the RDR register

**TDRE flag (Transmit Data Empty Flag)**

Indicates the presence of transmit data in the TDR register.

[Setting conditions]

- When the SCR.TE bit is 0
- When data is transmitted from the TDR register to the TSR register

[Clearing conditions]

- When it's written to 0 after the state of 1 is read
- When the SCR.TE bit is 1, it's written to the TDR register

Note: RDRF and TDRE should not be cleared by SSR register access unless communication is interrupted.

## 27.2.19 SCMR : Smart Card Mode Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x06

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | BCP2 | — | — | CHR1 | SDIR | SINV | — | SMIF |
| Value after reset: | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SMIF | Smart Card Interface Mode Select<br>0: Non-smart card interface mode (asynchronous mode, clock synchronous mode, simple SPI mode, or simple IIC mode)<br>1: Smart card interface mode | R/W*1 |
| 1 | — | This bit is read as 1. The write value should be 1. | R/W |
| 2 | SINV | Transmitted/Received Data Invert<br>Set the SINV bit to 0 for operation in simple IIC mode.<br>The level of communication terminals (RXD, TXD) are controlled by combination of this bit and SPTR.TINV/RINV. For details, see Figure 27.5.<br>The SINV bit can be used in the following modes:<br>● Smart card interface mode<br>● Asynchronous mode (including multi-processor mode)<br>● Clock synchronous mode<br>● Simple SPI mode<br>0: TDR contents are transmitted as they are. Received data is stored as received in the RDR register.<br>1: TDR register contents are inverted before transmission. Receive data is stored in inverted form in the RDR register. | R/W*1 |
| 3 | SDIR | Transmitted/Received Data Transfer Direction<br>Set the SDIR bit to 1 for operation in simple IIC mode.<br>The SDIR bit can be used in the following modes:<br>● Smart card interface mode<br>● Asynchronous mode (including multi-processor mode)<br>● Clock synchronous mode<br>● Simple SPI mode<br>0: Transfer LSB-first<br>1: Transfer MSB-first | R/W*1 |
| 4 | CHR1 | Character Length 1<br>Valid only in asynchronous mode.*2<br>Selects the transmit/receive character length in combination with the SMR.CHR bit.<br>0: SMR.CHR = 0: Transmit/receive in 9-bit data length<br>SMR.CHR = 1: Transmit/receive in 9-bit data length<br>1: SMR.CHR = 0: Transmit/receive in 8-bit data length (initial value)<br>SMR.CHR = 1: Transmit/receive in 7-bit data length*3 | R/W*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:5 | — | These bits are read as 1. The write value should be 1. | R/W |
| 7 | BCP2 | Base Clock Pulse 2<br>Selects the number of base clock cycles in combination with the SMR_SMCI.BCP[1:0] bits. Table 27.5 lists the combinations of the SCMR.BCP2 and SMR_SMCI.BCP[1:0] bits. | R/W*1 |

Note 1. Writable only when the TE and RE bits in SCR/SCR_SMCI are 0 (both serial transmission and reception are disabled).
Note 2. The setting is invalid and a fixed data length of 8 bits is used in modes other than asynchronous mode.
Note 3. LSB-first must be selected and the value of the MSB (bit [7]) in TDR cannot be transmitted.

The SCMR register selects the smart card interface and communication format.

### SMIF bit (Smart Card Interface Mode Select)

Setting the SMIF bit to 1 selects smart card interface mode. Setting it to 0 selects all other modes:

- Asynchronous mode, including multi-processor mode
- Clock synchronous mode
- Simple SPI mode
- Simple IIC mode

### SINV bit (Transmitted/Received Data Invert)

The SINV bit inverts the transmit and receive data logic level. It does not affect the logic level of the parity bit. To invert the parity bit, invert the PM bit in SMR or SMR_SMCI.

### CHR1 bit (Character Length 1)

The CHR1 bit selects the data length of transmit and receive data in combination with the CHR bit in the SMR register. A fixed data length of 8 bits is used in modes other than asynchronous mode.

### BCP2 bit (Base Clock Pulse 2)

The BCP2 bit selects the number of base clock cycles in a 1-bit data transfer time in smart card interface mode. Set this bit in combination with the SMR_SMCI.BCP[1:0] bits.

**Table 27.5 Combinations of the SCMR.BCP2 and SMR_SMCI.BCP[1:0] bits**

| SCMR.BCP2 bit | SMR_SMCI.BCP[1:0] bits | Number of base clock cycles for 1-bit transfer period |
|---|---|---|
| 0 | 00b | 93 clock cycles (S = 93)*1 |
| 0 | 01b | 128 clock cycles (S = 128)*1 |
| 0 | 10b | 186 clock cycles (S = 186)*1 |
| 0 | 11b | 512 clock cycles (S = 512)*1 |
| 1 | 00b | 32 clock cycles (S = 32) (Initial Value)*1 |
| 1 | 01b | 64 clock cycles (S = 64)*1 |
| 1 | 10b | 372 clock cycles (S = 372)*1 |
| 1 | 11b | 256 clock cycles (S = 256)*1 |

Note 1. S is the value of S in section 27.2.20. BRR : Bit Rate Register.

## 27.2.20 BRR : Bit Rate Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x01

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

BRR is an 8-bit register that adjusts the bit rate.

As each SCI channel has independent baud rate generator control, different bit rates can be set for each channel. Table 27.6 shows the relationship between the setting (N) in the BRR and the bit rate (B) for normal asynchronous mode, multi-processor transfer, clock synchronous mode, smart card interface mode, simple SPI mode, and simple IIC mode.

The initial value of the BRR register is 0xFF. The BRR register can be read by the CPU, but it can be written to only when the TE and RE bits in SCR/SCR_SMCI are 0.

**Table 27.6 Relationship between N setting in BRR and bit rate B**

| Mode | SEMR settings | | | BRR setting | Error |
|---|---|---|---|---|---|
| | BGDM bit | ABCS bit | ABCS E bit | | |
| Asynchronous, multi-processor transfer | 0 | 0 | 0 | $N = \frac{PCLK \times 10^6}{64 \times 2^{2n-1} \times B} - 1$ | $Error\ (\%) = \left\{\frac{PCLK \times 10^6}{B \times 64 \times 2^{2n-1} \times (N+1)} - 1\right\} \times 100$ |
| | 1 | 0 | 0 | $N = \frac{PCLK \times 10^6}{32 \times 2^{2n-1} \times B} - 1$ | $Error\ (\%) = \left\{\frac{PCLK \times 10^6}{B \times 32 \times 2^{2n-1} \times (N+1)} - 1\right\} \times 100$ |
| | 0 | 1 | 0 | | |
| | 1 | 1 | 0 | $N = \frac{PCLK \times 10^6}{16 \times 2^{2n-1} \times B} - 1$ | $Error\ (\%) = \left\{\frac{PCLK \times 10^6}{B \times 16 \times 2^{2n-1} \times (N+1)} - 1\right\} \times 100$ |
| | Don't care | Don't care | 1 | $N = \frac{PCLK \times 10^6}{12 \times 2^{2n-1} \times B} - 1$ | $Error\ (\%) = \left\{\frac{PCLK \times 10^6}{B \times 12 \times 2^{2n-1} \times (N+1)} - 1\right\} \times 100$ |
| Clock synchronous, simple SPI | | | | $N = \frac{PCLK \times 10^6}{8 \times 2^{2n-1} \times B} - 1$ | — |
| Smart card interface | | | | $N = \frac{PCLK \times 10^6}{S \times 2^{2n+1} \times B} - 1$ | $Error\ (\%) = \left\{\frac{PCLK \times 10^6}{B \times S \times 2^{2n+1} \times (N+1)} - 1\right\} \times 100$ |
| Simple IIC*1 | | | | $N = \frac{PCLK \times 10^6}{64 \times 2^{2n-1} \times B} - 1$ | — |

Note: B: Bit rate (bps)
N: BRR setting for on-chip baud rate generator (0 ≤ N ≤ 255)
PCLK: Operating frequency (MHz)
n and S: Determined by the SMR/SMR_SMCI and SCMR register settings as listed in Table 27.8 and Table 27.9.

Note 1. Adjust the bit rate so that the widths of high and low level of the SCLn output in simple IIC mode satisfy the $I^2C$ bus standard.

**Table 27.7 Calculating widths of SCLn high and low levels**

| Mode | SCLn | Formula (result in seconds) |
|---|---|---|
| IIC | Width at high level (minimum value) | $(N+1) \times 4 \times 2^{2n-1} \times 7 \times \frac{1}{PCLK \times 10^6}$ |
| | Width at low level (minimum value) | $(N+1) \times 4 \times 2^{2n-1} \times 8 \times \frac{1}{PCLK \times 10^6}$ |

**Table 27.8 Clock source settings**

| SMR or SMR_SMCI.CKS[1:0] bits setting | | |
|---|---|---|
| CKS[1:0] bits | Clock source | n |
| 00b | PCLK clock | 0 |
| 01b | PCLK/4 clock | 1 |
| 10b | PCLK/16 clock | 2 |
| 11b | PCLK/64 clock | 3 |

**Table 27.9 Base clock settings in smart card interface mode**

| SCMR.BCP2 bit setting | SMR_SMCI.BCP[1:0] bits setting | | |
|---|---|---|---|
| BCP2 bit | BCP[1:0] bits | Base clock cycles for 1-bit period | S |
| 0 | 00b | 93 clock cycles | 93 |
| 0 | 01b | 128 clock cycles | 128 |
| 0 | 10b | 186 clock cycles | 186 |
| 0 | 11b | 512 clock cycles | 512 |
| 1 | 00b | 32 clock cycles | 32 |
| 1 | 01b | 64 clock cycles | 64 |
| 1 | 10b | 372 clock cycles | 372 |
| 1 | 11b | 256 clock cycles | 256 |

Table 27.10 and Table 27.11 list examples of BRR (N) settings in normal asynchronous mode. Table 27.12 lists the maximum bit rate settable for each operating frequency. Table 27.16 lists examples of BRR (N) settings in smart card interface mode.

In smart card interface mode, the number of base clock cycles S in a 1-bit data transfer time can be selected. For details, see section 27.7.4. Receive Data Sampling Timing and Reception Margin. Table 27.13 and Table 27.15 list the maximum bit rates with external clock input.

When either the Asynchronous Mode Base Clock Select bit (ABCS) or the Baud Rate Generator Double-speed Mode Select bit (BGDM) in the Serial Extended Mode Register (SEMR) is set to 1 in asynchronous mode, the bit rate becomes twice the value listed in Table 27.17. When both of those registers are set to 1, the bit rate becomes four times the listed value.

**Table 27.10 Examples of BRR settings for different bit rates in asynchronous mode (1) (1 of 2)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 | | | 9.8304 | | | 10 | | | 12 | | | 12.288 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| **110** | 2 | 141 | 0.03 | 2 | 174 | -0.26 | 2 | 177 | -0.25 | 2 | 212 | 0.03 | 2 | 217 | 0.08 |
| **150** | 2 | 103 | 0.16 | 2 | 127 | 0.00 | 2 | 129 | 0.16 | 2 | 155 | 0.16 | 2 | 159 | 0.00 |
| **300** | 1 | 207 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 | 2 | 77 | 0.16 | 2 | 79 | 0.00 |
| **600** | 1 | 103 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 | 1 | 155 | 0.16 | 1 | 159 | 0.00 |
| **1200** | 0 | 207 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 | 1 | 77 | 0.16 | 1 | 79 | 0.00 |
| **2400** | 0 | 103 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 | 0 | 155 | 0.16 | 0 | 159 | 0.00 |
| **4800** | 0 | 51 | 0.16 | 0 | 63 | 0.00 | 0 | 64 | 0.16 | 0 | 77 | 0.16 | 0 | 79 | 0.00 |
| **9600** | 0 | 25 | 0.16 | 0 | 31 | 0.00 | 0 | 32 | -1.36 | 0 | 38 | 0.16 | 0 | 39 | 0.00 |
| **19200** | 0 | 12 | 0.16 | 0 | 15 | 0.00 | 0 | 15 | 1.73 | 0 | 19 | -2.34 | 0 | 19 | 0.00 |
| **31250** | 0 | 7 | 0.00 | 0 | 9 | -1.70 | 0 | 9 | 0.00 | 0 | 11 | 0.00 | 0 | 11 | 2.40 |
| **38400** | — | — | — | 0 | 7 | 0.00 | 0 | 7 | 1.73 | 0 | 9 | -2.34 | 0 | 9 | 0.00 |

**Table 27.10 Examples of BRR settings for different bit rates in asynchronous mode (1) (2 of 2)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 14 | | | 16 | | | 17.2032 | | | 18 | | | 19.6608 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| **110** | 2 | 248 | -0.17 | 3 | 70 | 0.03 | 3 | 75 | 0.48 | 3 | 79 | -0.12 | 3 | 86 | 0.31 |
| **150** | 2 | 181 | 0.16 | 2 | 207 | 0.16 | 2 | 223 | 0.00 | 2 | 233 | 0.16 | 2 | 255 | 0.00 |
| **300** | 2 | 90 | 0.16 | 2 | 103 | 0.16 | 2 | 111 | 0.00 | 2 | 116 | 0.16 | 2 | 127 | 0.00 |
| **600** | 1 | 181 | 0.16 | 1 | 207 | 0.16 | 1 | 223 | 0.00 | 1 | 233 | 0.16 | 1 | 255 | 0.00 |
| **1200** | 1 | 90 | 0.16 | 1 | 103 | 0.16 | 1 | 111 | 0.00 | 1 | 116 | 0.16 | 1 | 127 | 0.00 |
| **2400** | 0 | 181 | 0.16 | 0 | 207 | 0.16 | 0 | 223 | 0.00 | 0 | 233 | 0.16 | 0 | 255 | 0.00 |
| **4800** | 0 | 90 | 0.16 | 0 | 103 | 0.16 | 0 | 111 | 0.00 | 0 | 116 | 0.16 | 0 | 127 | 0.00 |
| **9600** | 0 | 45 | -0.93 | 0 | 51 | 0.16 | 0 | 55 | 0.00 | 0 | 58 | -0.69 | 0 | 63 | 0.00 |
| **19200** | 0 | 22 | -0.93 | 0 | 25 | 0.16 | 0 | 27 | 0.00 | 0 | 28 | 1.02 | 0 | 31 | 0.00 |
| **31250** | 0 | 13 | 0.00 | 0 | 15 | 0.00 | 0 | 16 | 1.20 | 0 | 17 | 0.00 | 0 | 19 | -1.70 |
| **38400** | — | — | — | 0 | 12 | 0.16 | 0 | 13 | 0.00 | 0 | 14 | -2.34 | 0 | 15 | 0.00 |

Note: In this example, SEMR.ABCS = 0, SEMR.ABCSE = 0, and SEMR.BGDM = 0.
When either the ABCS or BGDM bit is set to 1, the bit rate doubles.
When both ABCS and BGDM are set to 1, the bit rate increases four times.

**Table 27.11 Examples of BRR settings for different bit rates in asynchronous mode (2) (1 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 20 | | | 25 | | | 30 | | | 33 | | | 40 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 88 | -0.25 | 3 | 110 | -0.02 | 3 | 132 | 0.13 | 3 | 145 | 0.33 | 3 | 177 | -0.25 |
| 150 | 3 | 64 | 0.16 | 3 | 80 | 0.47 | 3 | 97 | -0.35 | 3 | 106 | 0.39 | 3 | 129 | 0.16 |
| 300 | 2 | 129 | 0.16 | 2 | 162 | -0.15 | 2 | 194 | 0.16 | 2 | 214 | -0.07 | 3 | 64 | 0.16 |
| 600 | 2 | 64 | 0.16 | 2 | 80 | 0.47 | 2 | 97 | -0.35 | 2 | 106 | 0.39 | 2 | 129 | 0.16 |
| 1200 | 1 | 129 | 0.16 | 1 | 162 | -0.15 | 1 | 194 | 0.16 | 1 | 214 | -0.07 | 2 | 64 | 0.16 |
| 2400 | 1 | 64 | 0.16 | 1 | 80 | 0.47 | 1 | 97 | -0.35 | 1 | 106 | 0.39 | 1 | 129 | 0.16 |
| 4800 | 0 | 129 | 0.16 | 0 | 162 | -0.15 | 0 | 194 | 0.16 | 0 | 214 | -0.07 | 1 | 64 | 0.16 |
| 9600 | 0 | 64 | 0.16 | 0 | 80 | 0.47 | 0 | 97 | -0.35 | 0 | 106 | 0.39 | 0 | 129 | 0.16 |
| 19200 | 0 | 32 | -1.36 | 0 | 40 | -0.76 | 0 | 48 | -0.35 | 0 | 53 | -0.54 | 0 | 64 | 0.16 |
| 31250 | 0 | 19 | 0.00 | 0 | 24 | 0.00 | 0 | 29 | 0.00 | 0 | 32 | 0.00 | 0 | 39 | 0.00 |
| 38400 | 0 | 15 | 1.73 | 0 | 19 | 1.73 | 0 | 23 | 1.73 | 0 | 26 | -0.54 | 0 | 32 | -1.36 |

**Table 27.11 Examples of BRR settings for different bit rates in asynchronous mode (2) (2 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 50 | | | 60 | | | 100 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 221 | -0.02 | — | — | — | 3 | 255 | — |
| 150 | 3 | 162 | -0.15 | 3 | 194 | 0.16 | 3 | 255 | — |
| 300 | 3 | 80 | 0.47 | 3 | 97 | -0.35 | 3 | 162 | -0.15 |
| 600 | 2 | 162 | -0.15 | 3 | 48 | -0.35 | 3 | 80 | 0.47 |
| 1200 | 2 | 80 | 0.47 | 2 | 97 | -0.35 | 2 | 162 | -0.15 |
| 2400 | 1 | 162 | -0.15 | 2 | 48 | -0.35 | 2 | 80 | 0.47 |

**Table 27.11 Examples of BRR settings for different bit rates in asynchronous mode (2) (3 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 50 | | | 60 | | | 100 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 4800 | 1 | 80 | 0.47 | 1 | 97 | -0.35 | 1 | 162 | -0.15 |
| 9600 | 0 | 162 | -0.15 | 1 | 48 | -0.35 | 1 | 80 | 0.47 |
| 19200 | 0 | 80 | 0.47 | 0 | 97 | -0.35 | 0 | 162 | -0.15 |
| 31250 | 0 | 49 | 0.00 | 0 | 59 | 0.00 | 1 | 24 | 0.00 |
| 38400 | 0 | 40 | -0.76 | 0 | 48 | -0.35 | 0 | 80 | 0.47 |

Note: In this example, SEMR.ABCS = 0, SEMR.ABCSE = 0, and SEMR.BGDM = 0.
When either the ABCS or BGDM bit is set to 1, the bit rate doubles.
When both ABCS = 1 and BGDM = 1, the bit rate quadruples.

**Table 27.12 Maximum bit rate for each operating frequency in asynchronous mode (1 of 2)**

| PCLK (MHz) | SEMR settings | | | | | Maximum bit rate (bps) | PCLK (MHz) | SEMR settings | | | | | Maximum bit rate (bps) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BGDM bit | ABCS bit | ABCSE bit | n | N | | | BGDM bit | ABCS bit | ABCSE bit | n | N | |
| 8 | 0 | 0 | 0 | 0 | 0 | 250,000 | 16 | 0 | 0 | 0 | 0 | 0 | 500,000 |
| | | 1 | 0 | 0 | 0 | 500,000 | | | 1 | 0 | 0 | 0 | 1,000,000 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 1,000,000 | | | 1 | 0 | 0 | 0 | 2,000,000 |
| | Don't care | Don't care | 1 | 0 | 0 | 1,333,333 | | Don't care | Don't care | 1 | 0 | 0 | 2,666,666 |
| 9.8304 | 0 | 0 | 0 | 0 | 0 | 307,200 | 17.2032 | 0 | 0 | 0 | 0 | 0 | 537,600 |
| | | 1 | 0 | 0 | 0 | 614,400 | | | 1 | 0 | 0 | 0 | 1,075,200 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 1,228,800 | | | 1 | 0 | 0 | 0 | 2,150,400 |
| | Don't care | Don't care | 1 | 0 | 0 | 1,638,400 | | Don't care | Don't care | 1 | 0 | 0 | 2,867,200 |
| 10 | 0 | 0 | 0 | 0 | 0 | 312,500 | 18 | 0 | 0 | 0 | 0 | 0 | 562,500 |
| | | 1 | 0 | 0 | 0 | 625,000 | | | 1 | 0 | 0 | 0 | 1,125,000 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 1,250,000 | | | 1 | 0 | 0 | 0 | 2,250,000 |
| | Don't care | Don't care | 1 | 0 | 0 | 1,666,666 | | Don't care | Don't care | 1 | 0 | 0 | 3,000,000 |
| 12 | 0 | 0 | 0 | 0 | 0 | 375,000 | 19.6608 | 0 | 0 | 0 | 0 | 0 | 614,400 |
| | | 1 | 0 | 0 | 0 | 750,000 | | | 1 | 0 | 0 | 0 | 1,228,800 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 1,500,000 | | | 1 | 0 | 0 | 0 | 2,457,600 |
| | Don't care | Don't care | 1 | 0 | 0 | 2,000,000 | | Don't care | Don't care | 1 | 0 | 0 | 3,276,800 |
| 12.288 | 0 | 0 | 0 | 0 | 0 | 384,000 | 20 | 0 | 0 | 0 | 0 | 0 | 625,000 |
| | | 1 | 0 | 0 | 0 | 768,000 | | | 1 | 0 | 0 | 0 | 1,250,000 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 1,536,000 | | | 1 | 0 | 0 | 0 | 2,500,000 |
| | Don't care | Don't care | 1 | 0 | 0 | 2,048,000 | | Don't care | Don't care | 1 | 0 | 0 | 3,333,333 |

**Table 27.12 Maximum bit rate for each operating frequency in asynchronous mode (2 of 2)**

| PCLK (MHz) | SEMR settings | | | | | Maximum bit rate (bps) | PCLK (MHz) | SEMR settings | | | | | Maximum bit rate (bps) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BGDM bit | ABCS bit | ABCSE bit | n | N | | | BGDM bit | ABCS bit | ABCSE bit | n | N | |
| 14 | 0 | 0 | 0 | 0 | 0 | 437,500 | 25 | 0 | 0 | 0 | 0 | 0 | 781,250 |
| | | 1 | 0 | 0 | 0 | 875,000 | | | 1 | 0 | 0 | 0 | 1,562,500 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 1,750,000 | | | 1 | 0 | 0 | 0 | 3,125,000 |
| | Don't care | Don't care | 1 | 0 | 0 | 2,333,333 | | Don't care | Don'are | 1 | 0 | 0 | 4,166,666 |
| 30 | 0 | 0 | 0 | 0 | 0 | 937,500 | 50 | 0 | 0 | 0 | 0 | 0 | 1,562,500 |
| | | 1 | 0 | 0 | 0 | 1,875,000 | | | 1 | 0 | 0 | 0 | 3,125,000 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 3,750,000 | | | 1 | 0 | 0 | 0 | 6,250,000 |
| | Don't care | Don't care | 1 | 0 | 0 | 5,000,000 | | Don't care | Don't care | 1 | 0 | 0 | 8,333,333 |
| 33 | 0 | 0 | 0 | 0 | 0 | 1,031,250 | 60 | 0 | 0 | 0 | 0 | 0 | 1,875,000 |
| | | 1 | 0 | 0 | 0 | 2,062,500 | | | 1 | 0 | 0 | 0 | 3,750,000 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 4,125,000 | | | 1 | 0 | 0 | 0 | 7,500,000 |
| | Don't care | Don't care | 1 | 0 | 0 | 5,500,000 | | Don't care | Don't care | 1 | 0 | 0 | 10,000,000 |
| 40 | 0 | 0 | 0 | 0 | 0 | 1,250,000 | 100 | 0 | 0 | 0 | 0 | 0 | 3,125,000 |
| | | 1 | 0 | 0 | 0 | 2,500,000 | | | 1 | 0 | 0 | 0 | 6,250,000 |
| | 1 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | 0 | 5,000,000 | | | 1 | 0 | 0 | 0 | 12,500,000 |
| | Don't care | Don't care | 1 | 0 | 0 | 6,666,666 | | Don't care | Don't care | 1 | 0 | 0 | 16,666,666 |

**Table 27.13 Maximum bit rate with external clock input in asynchronous mode (1 of 2)**

| Maximum bit rate (bps) | | | |
|---|---|---|---|
| PCLK (MHz) | External input clock (MHz) | SEMR.ABCS = 0 | SEMR.ABCS = 1 |
| 8 | 2.0000 | 125,000 | 250,000 |
| 9.8304 | 2.4576 | 153,600 | 307,200 |
| 10 | 2.5000 | 156,250 | 312,500 |
| 12 | 3.0000 | 187,500 | 375,000 |
| 12.288 | 3.0720 | 192,000 | 384,000 |
| 14 | 3.5000 | 218,750 | 437,500 |
| 16 | 4.0000 | 250,000 | 500,000 |
| 17.2032 | 4.3008 | 268,800 | 537,600 |
| 18 | 4.5000 | 281,250 | 562,500 |
| 19.6608 | 4.9152 | 307,200 | 614,400 |
| 20 | 5.0000 | 312,500 | 625,000 |
| 25 | 6.2500 | 390,625 | 781,250 |
| 30 | 7.5000 | 468,750 | 937,500 |
| 33 | 8.2500 | 515,625 | 1,031,250 |
| 40 | 10.0000 | 625,000 | 1,250,000 |
| 50 | 12.5000 | 781,250 | 1,562,500 |

**Table 27.13 Maximum bit rate with external clock input in asynchronous mode (2 of 2)**

| Maximum bit rate (bps) | | | |
|---|---|---|---|
| **PCLK (MHz)** | **External input clock (MHz)** | **SEMR.ABCS = 0** | **SEMR.ABCS = 1** |
| 60 | 15.0000 | 937,500 | 1,875,000 |
| 100 | 25.000 | 1,562,500 | 3,125,000 |

**Table 27.14 BRR settings for different bit rates in clock synchronous and simple SPI modes**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 | | 10 | | 16 | | 20 | | 25 | | 30 | | 33 | | 40 | | 50 | | 60 | | 100 | |
| | n | N | n | N | n | N | n | N | n | N | n | N | n | N | n | N | n | N | n | N | n | N |
| 110 | | | | | | | | | | | | | | | | | | | | | | |
| 250 | 3 | 124 | — | — | 3 | 249 | | | | | | | | | | | | | | | | |
| 500 | 2 | 249 | — | — | 3 | 124 | — | — | | | 3 | 233 | | | | | | | | | | |
| 1 k | 2 | 124 | — | — | 2 | 249 | — | — | 3 | 97 | 3 | 116 | 3 | 128 | 3 | 155 | 3 | 194 | 3 | 233 | | |
| 2.5 k | 1 | 199 | 1 | 249 | 2 | 99 | 2 | 124 | 2 | 155 | 2 | 187 | 2 | 205 | 2 | 249 | 3 | 77 | 3 | 93 | 3 | 155 |
| 5 k | 1 | 99 | 1 | 124 | 1 | 199 | 1 | 249 | 2 | 77 | 2 | 93 | 2 | 102 | 2 | 124 | 2 | 155 | 3 | 46 | 3 | 77 |
| 10 k | 0 | 199 | 0 | 249 | 1 | 99 | 1 | 124 | 1 | 155 | 1 | 187 | 1 | 205 | 1 | 249 | 2 | 77 | 2 | 93 | 3 | 38 |
| 25 k | 0 | 79 | 0 | 99 | 0 | 159 | 0 | 199 | 0 | 249 | 1 | 74 | 1 | 82 | 1 | 99 | 1 | 124 | 1 | 149 | 1 | 249 |
| 50 k | 0 | 39 | 0 | 49 | 0 | 79 | 0 | 99 | 0 | 124 | 0 | 149 | 0 | 164 | 1 | 49 | 1 | 61 | 1 | 74 | 1 | 124 |
| 100 k | 0 | 19 | 0 | 24 | 0 | 39 | 0 | 49 | 0 | 62 | 0 | 74 | 0 | 82 | 0 | 99 | 0 | 124 | 0 | 149 | 0 | 249 |
| 250 k | 0 | 7 | 0 | 9 | 0 | 15 | 0 | 19 | 0 | 24 | 0 | 29 | 0 | 32 | 0 | 39 | 0 | 49 | 0 | 59 | 1 | 24 |
| 500 k | 0 | 3 | 0 | 4 | 0 | 7 | 0 | 9 | — | — | 0 | 14 | — | — | 0 | 19 | 0 | 24 | 0 | 29 | 0 | 49 |
| 1 M | 0 | 1 | | | 0 | 3 | 0 | 4 | — | — | — | — | — | — | 0 | 9 | — | — | 0 | 14 | 0 | 24 |
| 2.5 M | | | 0 | 0*1 | | | 0 | 1 | — | — | 0 | 2 | — | — | 0 | 3 | 0 | 4 | 0 | 5 | 0 | 9 |
| 5 M | | | | | | | 0 | 0*1 | — | — | — | — | — | — | 0 | 1 | — | — | 0 | 2 | 0 | 4 |
| 7.5 M | | | | | | | | | | | 0 | 0*1 | | | | | | | 0 | 1 | | |
| 10 M | | | | | | | | | | | | | | | 0 | 0*1 | | | | | | |
| 15 M | | | | | | | | | | | | | | | | | | | 0 | 0*1 | | |

Note: Space: Setting prohibited.
—: Can be set, but an error occurs.

Note 1. Continuous transmission or reception is not possible. After transmitting or receiving one frame of data, a 1-bit period elapses before starting to transmit or receive the next frame of data. The output of the synchronization clock is stopped for a 1-bit period. Therefore, it takes 9 bits worth of time to transfer one frame (8 bits) of data, and the average transfer rate is 8/9 times the bit rate. When the FIFO is selected, this setting (BRR = 0x00 and SMR.CKS[1:0] = 00b) is not available.

**Table 27.15 Maximum bit rate with external clock input in clock synchronous and simple SPI modes (1 of 2)**

| PCLK (MHz) | External input clock (MHz) | Maximum bit rate (Mbps) |
|---|---|---|
| 8 | 1.3333 | 1.3333333 |
| 10 | 1.6667 | 1.6666667 |
| 12 | 2.0000 | 2.0000000 |
| 14 | 2.3333 | 2.3333333 |
| 16 | 2.6667 | 2.6666667 |
| 18 | 3.0000 | 3.0000000 |
| 20 | 3.3333 | 3.3333333 |
| 25 | 4.1667 | 4.1666667 |
| 30 | 5.0000 | 5.0000000 |

**Table 27.15 Maximum bit rate with external clock input in clock synchronous and simple SPI modes (2 of 2)**

| PCLK (MHz) | External input clock (MHz) | Maximum bit rate (Mbps) |
|---|---|---|
| 33 | 5.5000 | 5.5000000 |
| 40 | 6.6667 | 6.6666667 |
| 50 | 8.3333 | 8.3333333 |
| 60 | 10.0000 | 10.0000000 |
| 100 | 16.6667 | 16.6666667 |

**Table 27.16 BRR settings for different bit rates in smart card interface mode, n = 0, S = 372 (1 of 4)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 7.1424 | | | 10.00 | | | 10.7136 | | | 13.00 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 9600 | 0 | 0 | 0.00 | 0 | 1 | 30 | 0 | 1 | 25 | 0 | 1 | 8.99 |

**Table 27.16 BRR settings for different bit rates in smart card interface mode, n = 0, S = 372 (2 of 4)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 14.2848 | | | 16.00 | | | 18.00 | | | 20.00 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 9600 | 0 | 1 | 0.00 | 0 | 1 | 12.01 | 0 | 2 | 15.99 | 0 | 2 | 6.66 |

**Table 27.16 BRR settings for different bit rates in smart card interface mode, n = 0, S = 372 (3 of 4)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 25.00 | | | 30.00 | | | 33.00 | | | 40.00 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 9600 | 0 | 3 | 12.49 | 0 | 3 | 5.01 | 0 | 4 | 7.59 | 0 | 5 | -6.66 |

**Table 27.16 BRR settings for different bit rates in smart card interface mode, n = 0, S = 372 (4 of 4)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 50.00 | | | 60.00 | | | 100.00 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 9600 | 0 | 6 | 0.01 | 0 | 7 | 5.01 | 0 | 13 | 0.01 |

**Table 27.17 Maximum bit rate for each operating frequency in smart card interface mode (S = 32) (1 of 2)**

| PCLK (MHz) | Maximum bit rate (bps) | n | N |
|---|---|---|---|
| 10.00 | 156,250 | 0 | 0 |
| 10.7136 | 167,400 | 0 | 0 |
| 13.00 | 203,125 | 0 | 0 |
| 16.00 | 250,000 | 0 | 0 |
| 18.00 | 281,250 | 0 | 0 |
| 20.00 | 312,500 | 0 | 0 |
| 25.00 | 390,625 | 0 | 0 |
| 30.00 | 468,750 | 0 | 0 |
| 33.00 | 515,625 | 0 | 0 |
| 40.00 | 625,000 | 0 | 0 |
| 50.00 | 781,250 | 0 | 0 |

**Table 27.17 Maximum bit rate for each operating frequency in smart card interface mode (S = 32) (2 of 2)**

| PCLK (MHz) | Maximum bit rate (bps) | n | N |
|---|---|---|---|
| 60.00 | 937,500 | 0 | 0 |
| 100.00 | 1,562,500 | 0 | 0 |

**Table 27.18 BRR settings for different bit rates in simple IIC mode (1 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 | | | 10 | | | 16 | | | 20 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 10 k | 0 | 24 | 0.0 | 0 | 31 | -2.3 | 1 | 12 | -3.8 | 1 | 15 | -2.3 |
| 25 k | 0 | 9 | 0.0 | 0 | 12 | -3.8 | 1 | 4 | 0.0 | 1 | 6 | -10.7 |
| 50 k | 0 | 4 | 0.0 | 0 | 6 | -10.7 | 1 | 2 | -16.7 | 1 | 3 | -21.9 |
| 100 k | 0 | 2 | -16.7 | 0 | 3 | -21.9 | 0 | 4 | 0.0 | 0 | 6 | -10.7 |
| 250 k | 0 | 0 | 0.0 | 0 | 1 | -37.5 | 0 | 1 | 0.0 | 0 | 2 | -16.7 |
| 350 k | | | | | | | | | | 0 | 1 | -10.7 |
| 400 k | | | | | | | | | | 0 | 1 | -21.9 |

**Table 27.18 BRR settings for different bit rates in simple IIC mode (2 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 25 | | | 30 | | | 33 | | | 40 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 10 k | 1 | 19 | -2.3 | 1 | 23 | -2.3 | 1 | 25 | -0.8 | 0 | 124 | 0.00 |
| 25 k | 1 | 7 | -2.3 | 1 | 9 | -6.3 | 1 | 10 | -6.3 | 0 | 40 | 0.00 |
| 50 k | 1 | 3 | -2.3 | 1 | 4 | -6.3 | 1 | 5 | -14.1 | 0 | 24 | 0.00 |
| 100 k | 1 | 1 | -2.3 | 1 | 2 | -21.9 | 1 | 2 | -14.1 | 0 | 12 | -3.85 |
| 250 k | 0 | 3 | -21.9 | 0 | 3 | -6.3 | 0 | 4 | -17.5 | 0 | 4 | 0.00 |
| 350 k | 0 | 2 | -25.6 | 0 | 2 | -10.7 | 0 | 2 | -1.8 | 0 | 3 | -10.71 |
| 400 k | 0 | 1 | -2.3 | 0 | 1 | 17.2 | 0 | 2 | -14.1 | 0 | 2 | 4.17 |

**Table 27.18 BRR settings for different bit rates in simple IIC mode (3 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 50 | | | 60 | | | 100 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 10 k | 2 | 9 | -2.3 | 1 | 46 | -0.27 | 1 | 77 | 0.16 |
| 25 k | 2 | 3 | -2.3 | 0 | 74 | 0.00 | 0 | 124 | 0.00 |
| 50 k | 2 | 1 | -2.3 | 0 | 37 | -1.32 | 0 | 62 | -0.79 |
| 100 k | 1 | 3 | -2.3 | 0 | 18 | -1.32 | 0 | 30 | 0.81 |
| 250 k | 0 | 6 | -10.7 | 0 | 7 | -6.25 | 0 | 12 | -3.85 |
| 350 k | 0 | 4 | -10.7 | 0 | 4 | 7.14 | 0 | 8 | -0.79 |
| 400 k | 0 | 3 | -2.34 | 0 | 4 | -6.25 | 0 | 8 | -13.19 |

**Table 27.19 Minimum widths at SCL high and low levels at multiple bit rates in simple IIC mode (1 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 | | | 10 | | | 16 | | | 20 | | |
| | n | N | Min. widths at SCL high/low levels (µs) | n | N | Min. widths at SCL high/low levels (µs) | n | N | Min. widths at SCL high/low levels (µs) | n | N | Min. widths at SCL high/low levels (µs) |
| 10 k | 0 | 24 | 43.75/50.00 | 0 | 31 | 44.80/51.20 | 1 | 12 | 45.5/52.00 | 1 | 15 | 44.80/51.20 |
| 25 k | 0 | 9 | 17.50/20.00 | 0 | 12 | 18.2/20.80 | 1 | 4 | 17.50/20.00 | 1 | 6 | 19.60/22.40 |
| 50 k | 0 | 4 | 8.75/10.00 | 0 | 6 | 9.80/11.20 | 1 | 2 | 10.50/12.00 | 1 | 3 | 11.20/12.80 |
| 100 k | 0 | 2 | 5.25/6.00 | 0 | 3 | 5.60/6.40 | 0 | 4 | 4.37/5.00 | 0 | 6 | 4.90/5.60 |
| 250 k | 0 | 0 | 1.75/2.00 | 0 | 1 | 2.80/3.20 | 0 | 1 | 1.75/2.00 | 0 | 2 | 2.10/2.40 |
| 350 k | | | | | | | | | | 0 | 1 | 1.40/1.60 |
| 400 k | | | | | | | | | | 0 | 1 | 1.40/1.60 |

**Table 27.19 Minimum widths at SCL high and low levels at multiple bit rates in simple IIC mode (2 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 25 | | | 30 | | | 33 | | | 40 | | |
| | n | N | Min. widths at SCL high/low levels (µs) | n | N | Min. widths at SCL high/low levels (µs) | n | N | Min. widths at SCL high/low levels (µs) | n | N | Min. widths at SCL high/low levels (µs) |
| 10 k | 1 | 19 | 44.80/51.20 | 1 | 23 | 44.80/51.20 | 1 | 25 | 44.12/50.42 | 1 | 32 | 46.20/52.80 |
| 25 k | 1 | 7 | 17.92/20.48 | 1 | 9 | 18.66/21.33 | 1 | 10 | 18.66/21.33 | 1 | 12 | 18.20/20.80 |
| 50 k | 1 | 3 | 8.96/10.24 | 1 | 4 | 9.33/10.66 | 1 | 5 | 10.18/11.63 | 1 | 6 | 9.80/11.20 |
| 100 k | 1 | 1 | 4.48/5.12 | 1 | 2 | 5.60/6.40 | 1 | 2 | 5.09/5.81 | 0 | 13 | 4.90/5.60 |
| 250 k | 0 | 3 | 2.24/2.56 | 0 | 3 | 1.86/2.13 | 0 | 4 | 2.12/2.42 | 0 | 4 | 1.75/2.00 |
| 350 k | 0 | 2 | 1.68/1.92 | 0 | 2 | 1.40/1.60 | 0 | 2 | 1.27/1.45 | 0 | 3 | 1.40/1.60 |
| 400 k | 0 | 1 | 1.12/1.28 | 0 | 1 | 0.93/1.07 | 0 | 2 | 1.27 /1.45 | 0 | 2 | 1.05/1.20 |

**Table 27.19 Minimum widths at SCL high and low levels at multiple bit rates in simple IIC mode (3 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 50 | | | 60 | | | 100 | | |
| | n | N | Min. widths at SCL high/low levels (µs) | n | N | Min. widths at SCL high/low levels (µs) | n | N | Min. widths at SCL high/low levels (µs) |
| 10 k | 2 | 9 | 44.80/51.20 | 1 | 46 | 44.80/51.20 | 0 | 0 | 43.68/49.92 |
| 25 k | 2 | 3 | 17.92/20.48 | 0 | 74 | 17.50/20.00 | 0 | 0 | 17.50/20.00 |
| 50 k | 2 | 1 | 8.96/10.24 | 0 | 37 | 8.87/10.13 | 0 | 0 | 8.82/10.08 |
| 100 k | 1 | 3 | 4.48/5.12 | 0 | 18 | 4.43/5.07 | 0 | 0 | 4.34/4.96 |
| 250 k | 0 | 6 | 1.96/2.24 | 0 | 7 | 1.87/2.13 | 0 | 0 | 1.82/2.08 |
| 350 k | 0 | 4 | 1.40/1.60 | 0 | 4 | 1.17/1.33 | 0 | 0 | 1.26/1.44 |
| 400 k | 0 | 3 | 1.12/1.28 | 0 | 4 | 1.17/1.33 | 0 | 0 | 1.26/1.44 |

## 27.2.21 MDDR : Modulation Duty Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x12

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

MDDR corrects the bit rate adjusted by the BRR register.

When the BRME bit in SEMR is set to 1, the bit rate generated by the on-chip baud rate generator is evenly corrected using the settings in MDDR (M/256). Table 27.20 shows the relationship between the MDDR setting (M) and the bit rate (B).

The initial value of MDDR is 0xFF. Bit [7] in this register is fixed to 1.

The CPU can read the MDDR register, but this register is only writable when the TE and RE bits in SCR/SCR_SMCI are 0.

**Table 27.20 Relationship between MDDR setting (M) and bit rate (B) when bit rate modulation function is used**

B: Bit rate (bps)
M: MDDR setting (128 ≤ MDDR ≤ 256)
N: BRR setting for baud rate generator (0 ≤ N ≤ 255)
PCLK: Operating frequency (MHz)
n and S: Determined by the SMR/SMR_SMCI and SCMR register settings as listed in Table 27.8 and Table 27.9 in section 27.2.20. BRR : Bit Rate Register.

| Mode | SEMR settings | | | BRR setting | Error |
|---|---|---|---|---|---|
| | BGDM bit | ABCS bit | ABCSE bit | | |
| Asynchronous multiprocessor transfer | 0 | 0 | 0 | $N = \frac{PCLK \times 10^6}{64 \times 2^{2n-1} \times (256/M) \times B} - 1$ | $Error(\%) = \left\{\frac{PCLK \times 10^6}{B \times 64 \times 2^{2n-1} \times (256/M) \times (N+1)} - 1\right\} \times 100$ |
| | 1 | 0 | 0 | $N = \frac{PCLK \times 10^6}{32 \times 2^{2n-1} \times (256/M) \times B} - 1$ | $Error(\%) = \left\{\frac{PCLK \times 10^6}{B \times 32 \times 2^{2n-1} \times (256/M) \times (N+1)} - 1\right\} \times 100$ |
| | 0 | 1 | 0 | | |
| | 1 | 1 | 0 | $N = \frac{PCLK \times 10^6}{16 \times 2^{2n-1} \times (256/M) \times B} - 1$ | $Error(\%) = \left\{\frac{PCLK \times 10^6}{B \times 16 \times 2^{2n-1} \times (256/M) \times (N+1)} - 1\right\} \times 100$ |
| | × | × | 1 | $N = \frac{PCLK \times 10^6}{12 \times 2^{2n-1} \times (256/M) \times B} - 1$ | $Error(\%) = \left\{\frac{PCLK \times 10^6}{B \times 12 \times 2^{2n-1} \times (256/M) \times (N+1)} - 1\right\} \times 100$ |
| Clock synchronous, simple SPI*1 | | | | $N = \frac{PCLK \times 10^6}{8 \times 2^{2n-1} \times (256/M) \times B} - 1$ | — |
| Smart card interface | | | | $N = \frac{PCLK \times 10^6}{S \times 2^{2n+1} \times (256/M) \times B} - 1$ | $Error(\%) = \left\{\frac{PCLK \times 10^6}{B \times S \times 2^{2n+1} \times (256/M) \times (N+1)} - 1\right\} \times 100$ |
| Simple IIC*2 | | | | $N = \frac{PCLK \times 10^6}{64 \times 2^{2n-1} \times (256/M) \times B} - 1$ | — |

Note 1. Do not use this function in clock synchronous mode or in the highest speed settings in simple SPI mode (SMR.CKS[1:0] = 00b, SCR.CKE[1] = 0, and BRR = 0).

Note 2. Adjust the bit rate so that the widths at high and low level of the SCLn output in simple IIC mode satisfy the IIC standard.

Table 27.21 and Table 27.22 list examples of N settings in BRR and M settings in MDDR in normal asynchronous mode.

**Table 27.21 Examples of the BRR and MDDR settings for different bit rates in asynchronous mode (1) (1 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 | | | | | 9.8304 | | | | | 16 | | | | |
| | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) |
| 38400 | 0 | 5 | 236 | 0 | 0.03 | 0 | 7 | (256)*1 | 0 | 0.00 | 0 | 10 | 173 | 1 | -0.01 |
| 57600 | 0 | 3 | 236 | 0 | 0.03 | 0 | 4 | 240 | 0 | 0.00 | 0 | 4 | 236 | 0 | 0.03 |
| 115200 | 0 | 1 | 236 | 0 | 0.03 | 0 | 1 | 192 | 0 | 0.00 | 0 | 4 | 236 | 1 | 0.03 |
| 230400 | 0 | 0 | 236 | 0 | 0.03 | 0 | 0 | 192 | 0 | 0.00 | 0 | 1 | 189 | 1 | 0.14 |
| 460800 | 0 | 0 | 236 | 1 | 0.03 | 0 | 0 | 192 | 1 | 0.00 | 0 | 0 | 189 | 1 | 0.14 |

**Table 27.21 Examples of the BRR and MDDR settings for different bit rates in asynchronous mode (1) (2 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 12 | | | | | 12.288 | | | | | 14 | | | | |
| | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) |
| 38400 | 0 | 8 | 236 | 0 | 0.03 | 0 | 9 | (256)*1 | 0 | 0.00 | 0 | 16 | 191 | 1 | 0.00 |
| 57600 | 0 | 5 | 236 | 0 | 0.03 | 0 | 4 | 192 | 0 | 0.00 | 0 | 13 | 236 | 1 | 0.03 |
| 115200 | 0 | 2 | 236 | 0 | 0.03 | 0 | 4 | 192 | 1 | 0.00 | 0 | 6 | 236 | 1 | 0.03 |
| 230400 | 0 | 2 | 236 | 1 | 0.03 | 0 | 2 | 230 | 1 | -0.17 | 0 | 2 | 202 | 1 | -0.11 |
| 460800 | 0 | 0 | 157 | 1 | -0.18 | 0 | 0 | 154 | 1 | -0.26 | 0 | 0 | 135 | 1 | 0.14 |

**Table 27.21 Examples of the BRR and MDDR settings for different bit rates in asynchronous mode (1) (3 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 16 | | | | | 17.2032 | | | | | 18 | | | | |
| | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) |
| 38400 | 0 | 11 | 236 | 0 | 0.03 | 0 | 13 | (256)*1 | 0 | 0.00 | 0 | 18 | 166 | 1 | -0.01 |
| 57600 | 0 | 7 | 236 | 0 | 0.03 | 0 | 6 | 192 | 0 | 0.00 | 0 | 18 | 249 | 1 | -0.01 |
| 115200 | 0 | 3 | 236 | 0 | 0.03 | 0 | 6 | 192 | 1 | 0.00 | 0 | 8 | 236 | 1 | 0.03 |
| 230400 | 0 | 1 | 236 | 0 | 0.03 | 0 | 3 | 219 | 1 | -0.20 | 0 | 1 | 210 | 0 | 0.14 |
| 460800 | 0 | 1 | 236 | 1 | 0.03 | 0 | 1 | 219 | 1 | -0.20 | 0 | 0 | 210 | 0 | 0.14 |

Note 1. In this example, the ABCS and ABCSE bits in the SEMR register are 0. SEMR.BRME = 0 (M = 256) disables the bit rate modulation function.

**Table 27.22 Examples of BRR and MDDR settings for different bit rates in asynchronous mode (2) (1 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 19.6608 | | | | | 20 | | | | | 25 | | | | |
| | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) |
| 38400 | 0 | 15 | (256)*1 | 0 | 0.00 | 0 | 10 | 173 | 0 | -0.01 | 0 | 11 | 151 | 0 | 0.00 |
| 57600 | 0 | 9 | 240 | 0 | 0.00 | 0 | 9 | 236 | 0 | 0.03 | 0 | 7 | 151 | 0 | 0.00 |
| 115200 | 0 | 4 | 240 | 0 | 0.00 | 0 | 4 | 236 | 0 | 0.03 | 0 | 3 | 151 | 0 | 0.00 |
| 230400 | 0 | 1 | 192 | 0 | 0.00 | 0 | 4 | 236 | 1 | 0.03 | 0 | 1 | 151 | 0 | 0.00 |
| 460800 | 0 | 0 | 192 | 0 | 0.00 | 0 | 0 | 189 | 0 | 0.14 | 0 | 0 | 151 | 0 | 0.00 |

**Table 27.22 Examples of BRR and MDDR settings for different bit rates in asynchronous mode (2) (2 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 30 | | | | | 33 | | | | | 40 | | | | |
| | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) |
| 38400 | 0 | 36 | 194 | 1 | 0.01 | 0 | 14 | 143 | 0 | 0.01 | 0 | 21 | 173 | 0 | -0.01 |
| 57600 | 0 | 10 | 173 | 0 | -0.01 | 0 | 9 | 143 | 0 | 0.01 | 0 | 38 | 230 | 1 | -0.01 |
| 115200 | 0 | 10 | 173 | 1 | -0.01 | 0 | 4 | 143 | 0 | 0.01 | 0 | 9 | 236 | 0 | 0.03 |
| 230400 | 0 | 6 | 220 | 1 | -0.09 | 0 | 4 | 143 | 1 | 0.01 | 0 | 4 | 236 | 0 | 0.03 |
| 460800 | 0 | 3 | 252 | 1 | 0.14 | 0 | 1 | 229 | 0 | 0.10 | 0 | 4 | 236 | 1 | 0.03 |

**Table 27.22 Examples of BRR and MDDR settings for different bit rates in asynchronous mode (2) (3 of 3)**

| Bit rate (bps) | Operating frequency PCLK (MHz) | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 50 | | | | | 60 | | | | | 120 | | | | |
| | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) | n | N | M | BGDM bit | Error (%) |
| 38400 | 0 | 23 | 151 | 0 | 0.00 | 0 | 36 | 194 | 0 | 0.01 | 0 | 73 | 194 | 0 | 0.01 |
| 57600 | 0 | 15 | 151 | 0 | 0.00 | 0 | 21 | 173 | 0 | -0.01 | 0 | 58 | 232 | 0 | 0.01 |
| 115200 | 0 | 7 | 151 | 0 | 0.00 | 0 | 10 | 173 | 0 | -0.01 | 0 | 21 | 173 | 0 | -0.01 |
| 230400 | 0 | 3 | 151 | 0 | 0.00 | 0 | 10 | 173 | 1 | -0.01 | 0 | 10 | 173 | 0 | -0.01 |
| 460800 | 0 | 1 | 151 | 0 | 0.00 | 0 | 6 | 220 | 1 | -0.09 | 0 | 10 | 173 | 1 | -0.09 |

Note 1. In this example, the ABCS and ABCSE bits in the SEMR register are 0.
SEMR.BRME = 0 (M = 256) disables the bit rate modulation function.

## 27.2.22 SEMR : Serial Extended Mode Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x07

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | RXDESEL | BGDM | NFEN | ABCS | ABCSE | BRME | PADIS | ACS0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ACS0 | Asynchronous Mode Clock Source Select<br>Valid only in asynchronous mode<br>0: External clock input<br>1: Logical AND of compare matches output from the internal GPT.<br>These bit for the other SCI channels than SCIn (n = 1, 2) are reserved. | R/W*1 |
| 1 | PADIS | Preamble function Disable<br>Valid only in asynchronous mode<br>0: Preamble output function is enabled<br>1: Preamble output function is disabled<br>These bits for the other SCI channels than SCIn (n = 0, 3, 4, 9) are reserved. | R/W |
| 2 | BRME | Bit Rate Modulation Enable<br>0: Disable bit rate modulation function<br>1: Enable bit rate modulation function | R/W*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | ABCSE | Asynchronous Mode Extended Base Clock Select 1<br>Valid only in asynchronous mode with SCR.CKE[1] = 0.<br>0: Clock cycles for 1-bit period determined by combination of the BGDM and ABCS bits in the SEMR register<br>1: Baud rate is 6 base clock cycles for 1-bit period<br>These bits for the other SCI channels than SCIn (n = 0, 3, 4, 9) are reserved. | R/W*1 |
| 4 | ABCS | Asynchronous Mode Base Clock Select<br>Valid only in asynchronous mode.<br>0: Select 16 base clock cycles for 1-bit period<br>1: Select 8 base clock cycles for 1-bit period | R/W*1 |
| 5 | NFEN | Digital Noise Filter Function Enable<br>The NFEN bit must be 0 in all other modes.<br>0: In asynchronous mode: Disable noise cancellation function for RXDn input signal<br>In simple I2C mode: Disable noise cancellation function for SCLn and SDAn input signals<br>1: In asynchronous mode: Enable noise cancellation function for RXDn input signal<br>In simple I2C mode: Enable noise cancellation function for SCLn and SDAn input signals | R/W*1 |
| 6 | BGDM | Baud Rate Generator Double-Speed Mode Select<br>Valid only in asynchronous mode with SCR.CKE[1] = 0.<br>0: Output clock from baud rate generator with normal frequency<br>1: Output clock from baud rate generator with doubled frequency | R/W*1 |
| 7 | RXDESEL | Asynchronous Start Bit Edge Detection Select<br>Valid only in asynchronous mode.<br>0: Detect low level on RXDn pin as start bit<br>1: Detect falling edge of RXDn pin as start bit | R/W*1 |

Note 1. Writable only when the TE and RE bits in SCR/SCR_SMCI are 0 (both serial transmission and reception are disabled).

The SEMR register selects the clock source for the 1-bit period in asynchronous mode.

### ACS0 bit (Asynchronous Mode Clock Source Select)

Selects the clock source in the asynchronous mode.

The ACS0 bit is valid in asynchronous mode (SMR.CM bit = 0) and when an external clock input is selected (SCR.CKE[1:0] bits = 10b or 11b). This bit is used to select an external clock input or the logical AND of compare matches output from the internal GPT.

Set the ACS0 bit to 0 in other than asynchronous mode.

For SCI1 and SCI2, the GTIOCnA output (n = 3, 4, 6, 7) of GPT can be set as the serial transfer base clock. For details, see Table 27.23.

These bits for the other SCI channels than SCI1 and SCI2 are reserved. The write values to these bits for other than SCI1 and SCI2 should be 0.

**Table 27.23 Correspondence between SCI Channels and Compare Match Outputs**

| SCI | GPT | Compare Match Output |
|---|---|---|
| SCI1 | GPT3 | GTIOC3A |
| | GPT4 | GTIOC4A |
| SCI2 | GPT6 | GTIOC6A |
| | GPT7 | GTIOC7A |

Figure 27.3, Figure 27.4 show a setting example of when GTIOC3A and GTIOC4A are selected for output.

This figure shows an example when GPT clock is input to SCI1.

When generating 187.5 kbps of GPT average transfer rate for PCLKD = 32 MHz:

1. Generate a frequency of 4 MHz using GTIOC3A as the base clock.
2. Generate 3/4 clock enable of base clock to set an average transfer rate of 3 MHz/16 = 187.5 kbps using GTIOC4A.

Setting examples of GPT and SCI

- GPT3.GTSSR = 0x80000000, GPT4.GTSSR = 0x80000000 (Enable software counter start)
- GPT3.GTPR = 0x00000007, GPT4.GTPR = 0x0000001F (Maximum count value of GTCNT)
- GPT3.GTCCRA = 0x00000003, GPT4.GTCCRA = 0x00000007 (Compare match value)
- GPT3.GTCR = 0x00000000, GPT4.GTCR = 0x00000000 (Saw-wave PWM mode, timer prescaler is PCLKD/1)
- GPT3.GTIOR = 0x00000306, GPT4.GTIOR = 0x00000306

(Initial output low, output high at GTCCRA compare match, output low at cycle end)

- SCR.SCK[1:0] = 10b (External clock input or GPT clock input is selected)
- SEMR = 0x01 (TMR clock input is selected with 16 base clock cycles for 1-bit period)
- GPT3.GTSTR = 0x00000018 (Software start GTCNT counter)

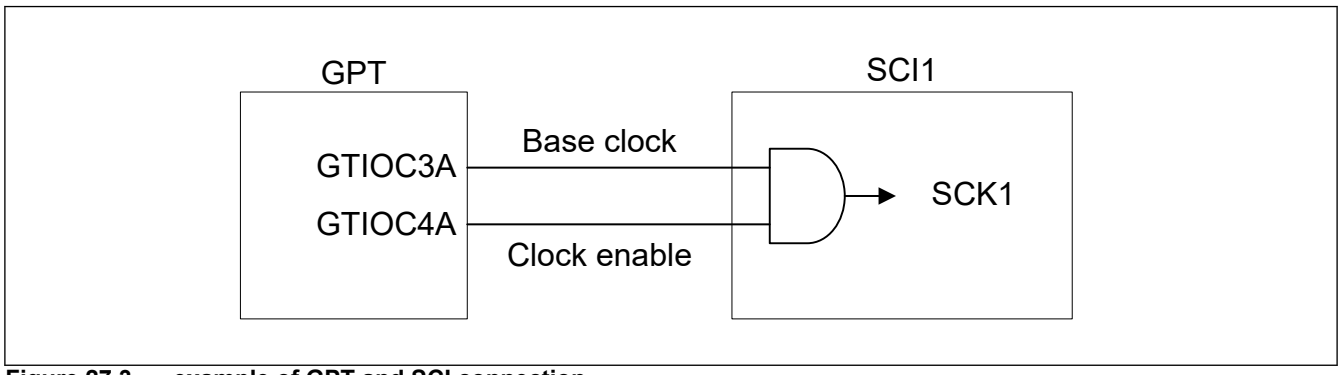



**Figure 27.3 example of GPT and SCI connection**

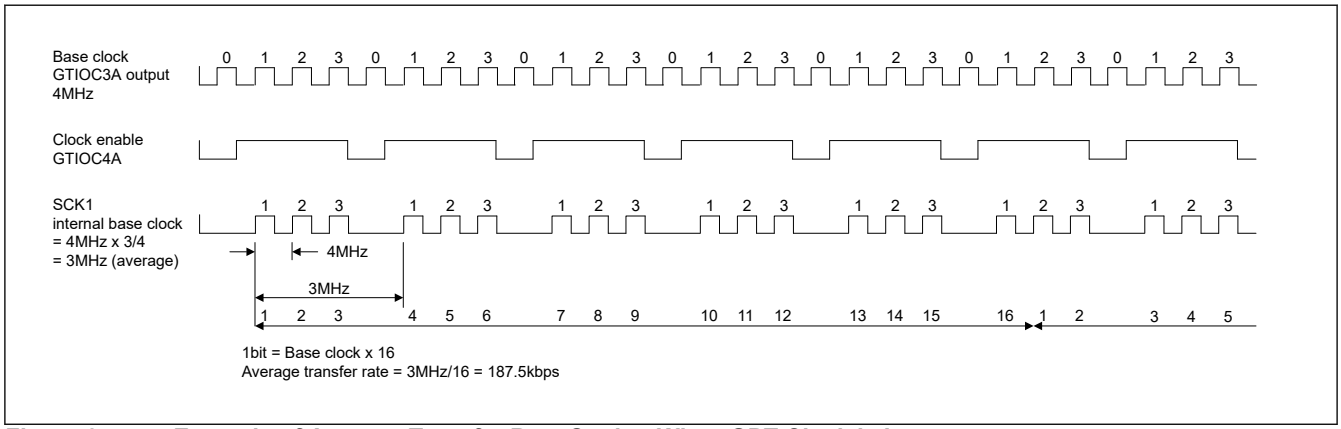



**Figure 27.4 Example of Average Transfer Rate Setting When GPT Clock is Input**

### PADIS bit (Preamble function Disable)

In asynchronous mode, select enable / disable of preamble function. In Manchester mode, preamble is not output regardless of this bit setting

### BRME bit (Bit Rate Modulation Enable)

The BRME bit enables or disables the bit rate modulation function. The bit rate generated by the on-chip baud rate generator is evenly corrected when this function is enabled. Set to 0 in Manchester mode.

### ABCSE bit (Asynchronous Mode Extended Base Clock Select 1)

The ABCSE bit sets the pulse number for the base clock in a 1-bit period to 6, and the double-frequency clock is output from the baud rate generator. When the bit rate is set to 6 while dividing the bus clock frequency, use this bit and set SMR.CKS[1:0] = 00b and BRR = 0.

Set it to 0 in modes other than asynchronous mode. Even in asynchronous mode, set it to 0 when using external clock.

### ABCS bit (Asynchronous Mode Base Clock Select)

The ABCS bit selects the number of clock cycles for a 1-bit period.

Set it to 0 in modes other than asynchronous mode and Manchester mode.

### NFEN bit (Digital Noise Filter Function Enable)

The NFEN bit enables or disables the digital noise filter function.

When the digital noise filter function is enabled:

- Noise cancellation is applied to the RXDn input signal in asynchronous mode
- Noise cancellation is applied to the SDAn and SCLn input signals in simple $I^2C$ mode

In all other modes, set the NFEN bit to 0 to disable the digital noise filter function. When the function is disabled, input signals are transferred as received.

### BGDM bit (Baud Rate Generator Double-Speed Mode Select)

The BGDM bit selects whether or not to double the base clock frequency output from the baud rate generator.

The BGDM bit is valid when the on-chip baud rate generator is selected as the clock source (SCR.CKE[1] = 0) in asynchronous mode (SMR.CM = 0) or Manchester mode (MMR.MANEN = 1). When external clock is selected (SCR.CKE[1] = 1), set it to 0. The base clock is generated by the clock output from the baud rate generator. When the BGDM bit is set to 1, the base clock cycle is halved and the bit rate is doubled.

Set this bit to 0 in modes other than asynchronous mode or Manchester mode.

### RXDESEL bit (Asynchronous Start Bit Edge Detection Select)

The RXDESEL bit selects the detection method of the start bit for reception in asynchronous mode. When a break occurs, data reception operation depends on the setting of this bit. Set this bit to 1 when reception must be stopped while a break occurs or when reception must be started without keeping the RXDn pin input at the high level for the period of one data frame or longer after completion of the break.

Set this bit to 0 in modes other than asynchronous mode.

## 27.2.23 SNFR : Noise Filter Setting Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x08

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | NFCS[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | NFCS[2:0] | Noise Filter Clock Select<br>In asynchronous mode, selects the standard setting for the base clock.<br>In simple $I^2C$ mode, selects the standard settings for the clock source of the on-chip baud rate generator selected in the SMR.CKS[1:0] bits.<br>0 0 0: In asynchronous mode: Use clock signal divided by 1 with noise filter<br>In simple $I^2C$ mode: Setting prohibited<br>0 0 1: In asynchronous mode: Setting prohibited<br>In simple $I^2C$ mode: Use clock signal divided by 1 with noise filter<br>0 1 0: In asynchronous mode: Setting prohibited<br>In simple $I^2C$ mode: Use clock signal divided by 2 with noise filter<br>0 1 1: In asynchronous mode: Setting prohibited<br>In simple $I^2C$ mode: Use clock signal divided by 4 with noise filter<br>1 0 0: In asynchronous mode: Setting prohibited<br>In simple $I^2C$ mode: Use clock signal divided by 8 with noise filter<br>Others: Setting prohibited | R/W*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Writing to these bits is only possible when the RE and TE bits in SCR/SCR_SMCI are 0 (serial reception and transmission disabled).

The SNFR register sets the digital noise filter clock.

### NFCS[2:0] bits (Noise Filter Clock Select)

The NFCS[2:0] bits select the sampling clock for the digital noise filter. To use the noise filter in asynchronous mode, set these bits to 000b. In simple $I^2C$ mode, when 32 clocks are selected as one bit period in the basic clock selection bits of the SEMR register, set the NFCS [2: 0] bits in the range from 001b to 100b. When any other value is selected for the basic clock selection bit, set the NFCS bit to 001b.

## 27.2.24 SIMR1 : IIC Mode Register 1

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x09

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | IICDL[4:0] | | | | | — | — | IICM |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IICM | Simple IIC Mode Select<br>0: SCMR.SMIF = 0: Asynchronous mode (including multi-processor mode), clock synchronous mode, or simple SPI mode<br>SCMR.SMIF = 1: Smart card interface mode<br>1: SCMR.SMIF = 0: Simple IIC mode<br>SCMR.SMIF = 1: Setting prohibited | R/W*1 |
| 2:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7:3 | IICDL[4:0] | SDAn Delay Output Select<br>SDAn signal output delay in cycles of the clock signal from the on-chip baud rate generator.<br>0x00: No output delay<br>Others: (IICDL - 1) to (IICDL) cycles | R/W*1 |

Note 1. Writing to these bits is only possible when the RE and TE bits in the SCR register are 0 (both serial transmission and reception are disabled).

SIMR1 selects simple IIC mode and the number of delay stages for the SDAn output.

### IICM bit (Simple IIC Mode Select)

In combination with the SCMR.SMIF bit, the IICM bit selects the operating mode.

### IICDL[4:0] bits (SDAn Delay Output Select)

The IICDL[4:0] bits specify an output delay on the SDAn pin relative to the falling edge of the output on the SCLn pin.

The available delay settings range from no delay to 31 cycles, with the clock signal from the on-chip baud rate generator as the base. The signal obtained by frequency-dividing PCLK by the divisor set in SMR.CKS[1:0] is supplied as the clock signal from the on-chip baud rate generator. Set 00000b to IICDL[4:0] bits unless operation is in simple IIC mode. In simple IIC mode, set the bits to a value in the range from 00001b to 11111b.

**Table 27.24 Settable value of IICDL[4: 0] bits in each communication mode (1 of 2)**

| Communication mode | ABCS | Settable value of IICDL[4:0] bits |
|---|---|---|
| Other than simple IIC mode | Don't care | 00000b |

**Table 27.24 Settable value of IICDL[4: 0] bits in each communication mode (2 of 2)**

| Communication mode | ABCS | Settable value of IICDL[4:0] bits |
|---|---|---|
| Simple IIC mode | 0 | 00001b to 11111b |
| | 1 | 00001b to 00100b |

## 27.2.25 SIMR2 : IIC Mode Register 2

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x0A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | IICACKT | — | — | — | IICCSC | IICINTM |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IICINTM | IIC Interrupt Mode Select<br>0: Use ACK/NACK interrupts<br>1: Use reception and transmission interrupts | R/W*1 |
| 1 | IICCSC | Clock Synchronization<br>0: Do not synchronize with clock signal<br>1: Synchronize with clock signal | R/W*1 |
| 4:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5 | IICACKT | ACK Transmission Data<br>0: ACK transmission<br>1: NACK transmission and ACK/NACK reception | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Writing to these bits is only possible when the RE and TE bits in the SCR register are 0 (serial reception and transmission disabled).

SIMR2 selects how reception and transmission are controlled in simple IIC mode.

### IICINTM bit (IIC Interrupt Mode Select)

The IICINTM bit selects the sources of interrupt requests in simple IIC mode.

### IICCSC bit (Clock Synchronization)

Set the IICCSC bit to 1 if the internally generated SCLn clock signal is to be synchronized when the SCLn pin is driven low because a wait was inserted by another other device.

The SCLn clock signal is not synchronized if the IICCSC bit is 0. The SCLn clock signal is generated according to the rate selected in the BRR register regardless of the level being input on the SCLn pin.

Set the IICCSC bit to 1 except during debugging.

### IICACKT bit (ACK Transmission Data)

Transmitted data contains ACK bits. Set the IICACKT bit to 1 when ACK and NACK bits are received.

## 27.2.26 SIMR3 : IIC Mode Register 3

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x0B

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | IICSCLS[1:0] | | IICSDAS[1:0] | | IICSTIF | IICSTPREQ | IICRSTAREQ | IICSTAREQ |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IICSTAREQ | Start Condition Generation<br>0: Do not generate start condition<br>1: Generate start condition*1 *3 *5 *6 | R/W |
| 1 | IICRSTAREQ | Restart Condition Generation<br>0: Do not generate restart condition<br>1: Generate restart condition*2 *3 *5 *6 | R/W |
| 2 | IICSTPREQ | Stop Condition Generation<br>0: Do not generate stop condition<br>1: Generate stop condition*2 *3 *5 *6 | R/W |
| 3 | IICSTIF | Issuing of Start, Restart, or Stop Condition Completed Flag<br>0: No requests are being made for generating conditions, or a condition is being generated<br>1: Generation of start, restart, or stop condition is complete.<br>When 0 is written to IICSTIF, it is set to 0*4 | R/W*4 |
| 5:4 | IICSDAS[1:0] | SDAn Output Select<br>0 0: Output serial data<br>0 1: Generate start, restart, or stop condition<br>1 0: Output low on SDAn pin<br>1 1: Drive SDAn pin to high-impedance state | R/W |
| 7:6 | IICSCLS[1:0] | SCLn Output Select<br>0 0: Output serial clock<br>0 1: Generate start, restart, or stop condition<br>1 0: Output low on SCLn pin<br>1 1: Drive SCLn pin to high-impedance state | R/W |

Note 1. Only generate a start condition after checking the bus state and confirming that the bus is free.
Note 2. Generate a restart or stop condition after checking the bus state and confirming that the bus is busy.
Note 3. Do not set more than one of the IICSTAREQ, IICRSTAREQ, and IICSTPREQ bits to 1 at a given time.
Note 4. Write only 0. When 1 is written, the value is ignored.
Note 5. Execute the generation of a condition after the value of the IICSTIF flag is 0.
Note 6. Do not write 0 to this bit while it is 1. Generation of a condition is suspended by writing 0 to this bit while it is 1.

The SIMR3 register is used to control the start, restart, and stop conditions in the simple $I^2C$ mode, and to hold the SSDAn and SSCLn pins at fixed levels.

### IICSTAREQ bit (Start Condition Generation)

When a start condition is to be generated, set both IICSDAS[1:0] and IICSCLS[1:0] to 01b in addition to setting the IICSTAREQ bit to 1.

[Setting condition]

- On writing 1 to the bit.

[Clearing condition]

- On completion of start condition generation.

#### IICRSTAREQ bit (Restart Condition Generation)

When a restart condition is to be generated, set both IICSDAS[1:0] and IICSCLS[1:0] to 01b in addition to setting the IICRSTAREQ bit to 1.

[Setting condition]

- On writing 1 to the bit.

[Clearing condition]

- On completion of restart condition generation.

#### IICSTPREQ bit (Stop Condition Generation)

When a stop condition is to be generated, set both IICSDAS[1:0] and IICSCLS[1:0] to 01b in addition to setting the IICSTPREQ bit to 1.

[Setting condition]

- On writing 1 to the bit.

[Clearing condition]

- On completion of stop condition generation.

#### IICSTIF flag (Issuing of Start, Restart, or Stop Condition Completed Flag)

After generating a condition, the IICSTIF flag indicates that the condition generation is complete. When using the IICSTAREQ, IICRSTAREQ, or IICSTPREQ bit to cause generation of a condition, do so after setting the IICSTIF flag to 0.

When the IICSTIF flag is 1 while an interrupt request is enabled by setting the SCR.TEIE bit, an STI request is output.

[Setting condition]

- On completion of a start, restart, or stop condition generation.

  If the setting condition conflicts with any of the clearing conditions for the flag, the clearing condition takes precedence.

[Clearing conditions]

- On writing 0 to the bit. After writing 0 to the IICSTIF bit, read the bit to check that it is actually set to 0.
- On writing 0 to the SIMR1.IICM bit when operation is not in simple IIC mode.
- On writing 0 to the SCR.TE bit.

#### IICSDAS[1:0] bits (SDAn Output Select)

The IICSDAS[1:0] bits control output from the SDAn pin. Set IICSDAS[1:0] and IICSCLS[1:0]to the same value during normal operations.

#### IICSCLS[1:0] bits (SCLn Output Select)

The IICSCLS[1:0] bits control output from the SCLn pin. Set IICSDAS[1:0] and IICSCLS[1:0] to the same value during normal operations.

### 27.2.27 SISR : IIC Status Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x0C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | IICACKR |
| Value after reset: | 0 | 0 | x | x | 0 | x | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IICACKR | ACK Reception Data Flag<br>0: ACK received<br>1: NACK received | R |
| 1 | — | This bit is read as 0. | R |
| 2 | — | The read value is undefined. | R |
| 3 | — | This bit is read as 0. | R |
| 5:4 | — | The read value is undefined. | R |
| 7:6 | — | These bits are read as 0. | R |

SISR monitors the state in simple IIC mode.

### IICACKR flag (ACK Reception Data Flag)

Received ACK and NACK bits can be read from the IICACKR flag. The IICACKR flag is updated on the rising edge of the SCLn clock for the received ACK/NACK bit.

## 27.2.28 SPMR : SPI Mode Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)
SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x0D

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CKPH | CKPOL | — | MFF | CTSPEN | MSS | CTSE | SSE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SSE | SSn Pin Function Enable<br>0: Disable SSn pin function<br>1: Enable SSn pin function | R/W*1 |
| 1 | CTSE | CTS Enable<br>0: Disable CTS function (enable RTS output function)<br>1: Enable CTS function | R/W*1 |
| 2 | MSS | Master Slave Select<br>0: Transmit through TXDn pin and receive through RXDn pin (master mode)<br>1: Receive through TXDn pin and transmit through RXDn pin (slave mode) | R/W*1 |
| 3 | CTSPEN | CTS external pin Enable<br>0: Alternate setting to use CTS and RTS functions as either one terminal<br>1: Dedicated setting for separately using CTS and RTS functions with 2 terminals<br>These bits for the other SCI channels than SCIn (n = 0, 3, 4, 9) are reserved. | R/W |
| 4 | MFF | Mode Fault Flag<br>0: No mode fault error<br>1: Mode fault error | R/W*2 |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | CKPOL | Clock Polarity Select<br>0: Do not invert clock polarity<br>1: Invert clock polarity | R/W*1 |
| 7 | CKPH | Clock Phase Select<br>0: Do not delay clock<br>1: Delay clock | R/W*1 |

Note 1. Writing to these bits is only possible when the RE and TE bits in the SCR register are 0 (both serial transmission and reception are disabled).
Note 2. Only 0 can be written to this bit, to clear the flag.

SPMR selects the extension settings in asynchronous and clock synchronous modes.

### SSE bit (SSn Pin Function Enable)

Set the SSE bit to 1 to use the SSn pin to control transmission and reception in simple SPI mode. Set this bit to 0 in all other modes. In simple SPI mode, when master mode is selected (SCR.CKE[1:0] = 00b and SPMR.MSS = 0) and there is a single master, the SSn pin on the master side is not required to control reception and transmission. In such a case, set the SSE bit to 0. Do not set both the SSE and CTSE bits to 1. If this setting is made, operation is the same as that when these bits are set to 0.

### CTSE bit (CTS Enable)

Set the CTSE bit to 1 if the SSn pin is to be used for inputting the CTS control signal to control transmission and reception. The RTS signal is output when this bit is set to 0. Set this bit to 0 in smart card interface mode, simple SPI mode, and simple IIC mode. Do not set both the CTSE and SSE bits to 1. If this setting is made, operation is the same as that when these bits are set to 0.

### MSS bit (Master Slave Select)

The MSS bit selects master or slave operation in simple SPI mode. The functions of the TXDn and RXDn pins are reversed when this bit is set to 1, so that data is received through the TXDn pin and transmitted through the RXDn pin.

Set this bit to 0 in modes other than simple SPI mode.

### CTSPEN bit (CTS external pin Enable)

Select the terminals usage method when using the CTS and RTS functions.

### MFF flag (Mode Fault Flag)

The MFF flag indicates mode fault errors. In a multi-master configuration, determine the mode fault error occurrence by reading this flag.

[Setting condition]

- When input on the SSn pin is low during master operation in simple SPI mode (SSE bit = 1 and MSS bit = 0).

[Clearing condition]

- On writing 0 to the bit after it is read as 1.

### CKPOL bit (Clock Polarity Select)

The CKPOL bit selects the polarity of the clock signal output through the SCKn pin. See Figure 27.99 for details. Set the CKPOL bit to 0 in all modes other than simple SPI mode and clock synchronous mode.

### CKPH bit (Clock Phase Select)

The CKPH bit selects the phase of the clock signal output through the SCKn pin. See Figure 27.99 for details. Set the CKPH bit to 0 in all modes other than simple SPI mode and clock synchronous mode.

## 27.2.29 FCR : FIFO Control Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x14

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | RSTRG[3:0] | | | | RTRG[3:0] | | | | TTRG[3:0] | | | | DRES | TFRST | RFRST | FM |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FM | FIFO Mode Select<br>Valid only in asynchronous mode, including multi-processor mode, or clock synchronous mode.<br>0: Non-FIFO mode.<br>Selects TDR/RDR or TDRHL/RDRHL for communication.<br>1: FIFO mode.<br>Selects FTDRHL/FRDRHL for communication. | R/W*1 |
| 1 | RFRST | Receive FIFO Data Register Reset<br>Valid only when FCR.FM = 1.<br>0: Do not reset FRDRHL<br>1: Reset FRDRHL | R/W |
| 2 | TFRST | Transmit FIFO Data Register Reset<br>Valid only when FCR.FM = 1.<br>0: Do not reset FTDRHL<br>1: Reset FTDRHL | R/W |
| 3 | DRES | Receive Data Ready Error Select<br>Selects the interrupt requested when detecting receive data ready.<br>0: Receive data full interrupt (SCIn_RXI)<br>1: Receive error interrupt (SCIn_ERI) | R/W |
| 7:4 | TTRG[3:0] | Transmit FIFO Data Trigger Number<br>Valid only in asynchronous mode, including multi-processor mode, or clock synchronous mode.<br>The trigger number is specified in the TTRG[3:0] bits. | R/W |
| 11:8 | RTRG[3:0] | Receive FIFO Data Trigger Number<br>Valid only in asynchronous mode, including multi-processor mode, or clock synchronous mode.<br>The trigger number is specified in the RTRG[3:0] bits. | R/W |
| 15:12 | RSTRG[3:0] | RTS Output Active Trigger Number Select<br>Valid only in asynchronous mode, including multi-processor mode, or clock synchronous mode, when FCR.FM = 1, SPMR.CTSE = 0, and SPMR.SSE = 0.<br>The trigger number is specified in the RSTRG[3:0] bits. | R/W |

Note 1. Writable only when TE = 0 and RE = 0.

FCR selects FIFO mode, resets FTDRHL and FRDRHL, selects the FIFO data trigger number for transmission or reception, and selects the RTS output active trigger number.

### FM bit (FIFO Mode Select)

When the FM bit is set to 1, FTDRHL and FRDRHL are selected for communication. When the FM bit is set to 0, TDR and RDR, or TDRHL and RDRHL are selected for communication.

### RFRST bit (Receive FIFO Data Register Reset)

When the RFRST bit is set to 1, the FRDRHL register is reset and the received data count resets to 0. When 1 is written to the RFRST bit, it clears to 0 after 1 PCLK.

### TFRST bit (Transmit FIFO Data Register Reset)

When the TFRST bit is set to 1, the FTDRHL register is reset and the transmit data count resets to 0. When 1 is written to the TFRST bit, it clears to 0 after 1 PCLK.

### DRES bit (Receive Data Ready Error Select)

When detecting a receive data ready error, the selection can be made from an SCIn_RXI interrupt request or an SCIn_ERI interrupt request. When starting DTC or DMAC and reading from the FRDRH and FRDRL registers, set the DRES bit to 1.

### TTRG[3:0] bits (Transmit FIFO Data Trigger Number)

The TDFE flag is set to 1 when the amount of transmit data in FTDRHL is equal to or less than the transmit triggering number specified in the TTRG[3:0] bits, and software can write data to FTDRHL. If SCR.TIE = 1, an SCIn_TXI interrupt request occurs.

### RTRG[3:0] bits (Receive FIFO Data Trigger Number)

The RDF flag is set to 1 when the amount of receive data in FRDRHL is equal to or greater than the receive triggering number specified in the RTRG[3:0] bits, and software can read data from FRDRHL. If SCR.RIE = 1, an SCIn_RXI interrupt request occurs.

When RTRG[3:0] is 0, the RDF flag is not set even when the amount of data in the receive FIFO is equal to 0, and an SCIn_RXI interrupt does not occur.

### RSTRG[3:0] bits (RTS Output Active Trigger Number Select)

When the amount of receive data stored in FRDRHL is equal to or greater than the receive triggering number specified in the RSTRG[3:0] bits, the RTS signal goes high.

When RSTRG[3:0] is 0, the RTS signal does not go high even when the amount of data in FRDRHL is equal to 0.

## 27.2.30 FDR : FIFO Data Count Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x16

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | T[4:0] | | | | | — | — | — | R[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | R[4:0] | Receive FIFO Data Count<br>Valid only in asynchronous mode, including multi-processor mode, or clock synchronous mode, when FCR.FM = 1.<br>Indicates the amount of receive data stored in FRDRHL. | R |
| 7:5 | — | These bits are read as 0. | R |
| 12:8 | T[4:0] | Transmit FIFO Data Count<br>Valid only in asynchronous mode, including multi-processor mode, or clock synchronous mode, when FCR.FM = 1.<br>Indicates the amount of non-transmitted data stored in FTDRHL. | R |
| 15:13 | — | These bits are read as 0. | R |

The FDR register indicates the amount of data stored in FRDRHL and FTDRHL.

### R[4:0] bits (Receive FIFO Data Count)

The R[4:0] bits indicate the amount of receive data stored in FRDRHL. 0x00 means no receive data, and 0x10 means that the maximum received data is stored in FRDRHL.

### T[4:0] bits (Transmit FIFO Data Count)

The T[4:0] bits indicate the amount of non-transmitted data stored in FTDRHL. 0x00 means no transmit data, and 0x10 means that all (maximum amount) of the data to be transmitted is stored in FTDRHL.

## 27.2.31 LSR : Line Status Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x18

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | PNUM[4:0] | | | | | — | FNUM[4:0] | | | | | — | ORER |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ORER | Overrun Error Flag<br>Valid only in asynchronous mode, including multi-processor mode, or clock synchronous mode, and when FIFO is selected.<br>0: No overrun error occurred<br>1: Overrun error occurred | R*1 |
| 1 | — | This bit is read as 0. | R |
| 6:2 | FNUM[4:0] | Framing Error Count<br>Indicates the amount of data with a framing error in the receive data stored in FRDRHL. | R |
| 7 | — | This bit is read as 0. | R |
| 12:8 | PNUM[4:0] | Parity Error Count<br>Indicates the amount of data with a parity error in the receive data stored in FRDRHL. | R |
| 15:13 | — | These bits are read as 0. | R |

Note 1. Write 0 to SSR_FIFO.ORER to clear the flag.

The LSR register indicates the receive error status.

### ORER flag (Overrun Error Flag)

The ORER flag reflects the value in SSR_FIFO.ORER.

### FNUM[4:0] bits (Framing Error Count)

The FNUM[4:0] value indicates the amount of data with a framing error stored in the FRDRHL register.

### PNUM[4:0] bits (Parity Error Count)

The PNUM[4:0] value indicates the amount of data with a parity error stored in the FRDRHL register.

## 27.2.32 CDR : Compare Match Data Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x1A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | CMPD[8:0] | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8:0 | CMPD[8:0] | Compare Match Data<br>Holds compare data pattern for address match wakeup function. | R/W |
| 15:9 | — | These bits are read as 0. The write value should be 0. | R/W |

The CDR register sets the compare data for the address match function.

### CMPD[8:0] bits (Compare Match Data)

The CMPD[8:0] bits set the data to be compared to receive data for the address match function, when the address match function is enabled (DCCR.DCME = 1).

Three bit lengths are available:

- CMPD[6:0] with 7-bit length
- CMPD[7:0] with 8-bit length
- CMPD[8:0] with 9-bit length

## 27.2.33 DCCR : Data Compare Match Control Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x13

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DCME | IDSEL | — | DFER | DPER | — | — | DCMF |
| Value after reset: | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DCMF | Data Compare Match Flag<br>0: Not matched<br>1: Matched | R/(W)*1 |
| 2:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 3 | DPER | Data Compare Match Parity Error Flag<br>0: No parity error occurred<br>1: Parity error occurred | R/(W)*1 |
| 4 | DFER | Data Compare Match Framing Error Flag<br>0: No framing error occurred<br>1: Framing error occurred | R/(W)*1 |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | IDSEL | ID Frame Select<br>Valid only in asynchronous mode, including multi-processor mode.<br>0: Always compare data regardless of the MPB bit value<br>1: Only compare data when MPB bit = 1 (ID frame) | R/W |
| 7 | DCME | Data Compare Match Enable<br>Valid only in asynchronous mode, including multi-processor mode.<br>0: Disable address match function<br>1: Enable address match function | R/W |

Note 1. Only 0 can be written, to clear the flag after reading 1.

The DCCR register controls the address match function.

### DCMF flag (Data Compare Match Flag)

The DCMF flag indicates that the SCI detected a receive data match with the comparison data (CDR.CMPD).

[Setting condition]

- On match of the comparison data (CDR.CMPD) with the receive data when DCCR.DCME = 1.

[Clearing condition]

- When 0 is written after 1 is read from DCMF.

Clearing the SCR.RE bit to 0 does not affect the DCMF flag, which retains its previous value.

### DPER flag (Data Compare Match Parity Error Flag)

The DPER flag indicates that a parity error occurred on address match detection (receive data match detection).

[Setting condition]

- When a parity error is detected in a frame in which an address match is detected.

[Clearing conditions]

- When 0 is written after 1 is read from DPER.

When the SCR.RE bit is set to 0 (serial reception is disabled), the DPER flag is not affected and retains its previous value.

### DFER flag (Data Compare Match Framing Error Flag)

The DFER flag indicates that a framing error occurred on address match detection (receive data match detection).

[Setting conditions]

- When a stop bit of a frame in which an address match is detected is 0.

  When in 2-stop-bit mode, only the first bit of the stop bits is checked for a value of 1 (the second stop bit is not checked).

[Clearing conditions]

- When 0 is written after 1 is read from DFER.

When the SCR.RE bit is set to 0 (serial reception is disabled), the DFER flag is not affected and retains its previous value.

### IDSEL bit (ID Frame Select)

The IDSEL bit selects whether to compare data regardless of the MPB bit value or to compare data only when MPB = 1 (ID frame), when the address match function is enabled.

### DCME bit (Data Compare Match Enable)

The DCME bit enables or disables the address match function (data compare match function).

If the SCI detects a match to the comparison data (CDR.CMPD) with the receive data, the DCME bit clears automatically, after which SCI operation mode is in normal receive mode. See section 27.3.6. Address Match (Receive Data Match Detection) Function.

The write value must be 0 for all modes other than asynchronous mode.

## 27.2.34 SPTR : Serial Port Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x1C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ATEN | ASEN | TINV | RINV | — | SPB2IO | SPB2DT | RXDMON |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RXDMON | Serial Input Data Monitor<br>Indicates the state of the RXDn pin.<br>0: When RINV = 0, RXDn terminal is the low level.<br>When RINV = 1, RXDn terminal is the High level.<br>1: When RINV = 0, RXDn terminal is the High level.<br>When RINV = 1, RXDn terminal is the Low level. | R |
| 1 | SPB2DT | Serial Port Break Data Select<br>Selects the output level of the TXDn pin when SCR.TE = 0.<br>0: When TINV = 0, Low level is output in TXDn terminal.<br>When TINV = 1, High level is output in TXDn terminal.<br>1: When TINV = 0, High level is output in TXDn terminal.<br>When TINV = 1, Low level is output in TXDn terminal. | R/W |
| 2 | SPB2IO | Serial Port Break I/O*1<br>Selects whether the value of SPB2DT is output to TXDn pin.<br>0: Do not output value of SPB2DT bit on TXDn pin<br>1: Output value of SPB2DT bit on TXDn pin | R/W |
| 3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | RINV | RXD invert bit<br>0: Received data from RXDn is not inverted and input.*2<br>1: Received data from RXDn is inverted and input. | R/W*3 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5 | TINV | TXD invert bit<br>0: Transmit data is not inverted and output to TXDn.*2<br>1: Transmit data is inverted and output to TXDn. | R/W*3 |
| 6 | ASEN | Adjust receive sampling timing enable<br>(This bit enables in asynchronous mode using internal clock)<br>This function can adjust the receive sampling timing.<br>In asynchronous mode using internal clock, see section 27.3.10. The function of adjust receive sampling timing (Asynchronous Mode) for details.<br>0: Adjust sampling timing disable.<br>1: Adjust sampling timing enable. | R/W*3 |
| 7 | ATEN | Adjust transmit timing enable<br>(This bit enables in asynchronous mode using internal clock)<br>This function can adjust the transmit edge of TXDn waveform. See section 27.3.11. The function of adjust transmit timing (Asynchronous Mode) for details.<br>0: Adjust transmit timing disable.<br>1: Adjust transmit timing enable. | R/W*3 |

Note 1. Please use this bit in asynchronous mode and manchester mode. Movement by other mode isn't guaranteed.
Note 2. RINV/TINV should be set to 0 in smart card interface mode and simple I2C mode.
Note 3. Change the value of these bits only at SCR.TE = SCR.RE = 0.

The SPTR register provides confirmation of the serial reception pin (RXDn pin) status and sets the transmission and receive pin status.

And SPTR register has enable bits for adjust functions of receive sampling timing and transmit timing.

The TXDn pin status is determined by the combination of SCR.TE, SPTR.SPB2IO, and SPTR.SPB2DT settings, as shown in Table 27.25.

The data of RDR is controlled by RINV and SCMR.SINV. And the data from TXDn terminal is controlled by TINV and SCMR.SINV. The control by RINV/TINV are done to communication terminals (RXDn/TXDn), so they can control not only data-bits but also other bits (start bit, stop bit, parity bit). For details, see Figure 27.5.

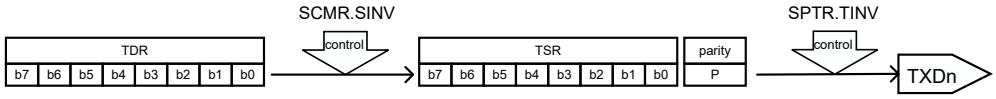
**Table 27.25 TXDn pin status**

| Value of SCR.TE | Value of SPTR.SPB2IO | Value of SPTR.SPB2DT | TXDn pin status |
|---|---|---|---|
| 0 | 0 | — | Hi-Z (initial value) |
| 0 | 1 | 0 | Low level output |
| 0 | 1 | 1 | High level output |
| 1 | — | — | Serial transmit data is output |

Note: —: Do not care.
Note: Use the SPTR register in asynchronous mode only. Using this register in any other mode is not guaranteed.

The receive/transmit data control (Data size=8bits, Even parity, MSB first)

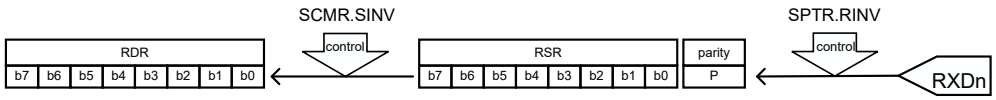
The transmit data is controlled by SPTR.TINV and SCMR.SINV.




| SCMR .SINV | SPTR. TINV | TDR | TSR | Prity (even) | TXDn waveform |
|---|---|---|---|---|---|
| 0 | 0 | 0xBE | 0xBE | 0 | |
| 0 | 1 | 0xBE | 0xBE | 0 | |
| 1 | 0 | 0xBE | 0x41 | 0 | |
| 1 | 1 | 0xBE | 0x41 | 0 | |



The received data is controlled by SPTR.RINV and SCMR.SINV.




| SCMR .SINV | SPTR. TINV | RDR | RSR | parity (even) | RXDn waveform |
|---|---|---|---|---|---|
| 0 | 0 | 0xBE | 0xBE | 0 | |
| 1 | 0 | 0x41 | 0xBE | 0 | |
| 0 | 1 | 0xBE | 0xBE | 0 | |
| 1 | 1 | 0x41 | 0xBE | 0 | |



**Figure 27.5 Example of the receive/transmit data control**

## 27.2.35 ACTR : Adjustment Communication Timing Register

Base address: SCIn = 0x4011_8000 + 0x0100 × n (n = 0, 9)
SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x1D

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | AET | ATT[2:0] | | | AJD | AST[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | AST | Adjustment value for receive Sampling Timing<br>The sampling timing of RXD terminal is adjusted from the middle of bit by the following formula.<br>Adjustment sampling timing = base clock × the setting value of AST[2:0].<br>This bit is effective only at SPTR.ASEN = 1. This setting timing is limited by setting the base clock cycles. For details, see section 27.3.10. The function of adjust receive sampling timing (Asynchronous Mode). | R/W*1 |
| 3 | AJD | Adjustment Direction for receive sampling timing<br>Adjustment direction for RXD receive sampling timing is determined by this bit.<br>0: The sampling timing is adjusted backward to the middle of bit.<br>1: The sampling timing is adjusted forward to the middle of bit.<br>This bit is effective only at SPTR.ASEN = 1. For details, see section 27.3.10. The function of adjust receive sampling timing (Asynchronous Mode). | R/W*1 |
| 6:4 | ATT | Adjustment value for Transmit timing<br>The selected edge timing of TXD is adjusted by the following formula.<br>Adjustment edge timing = base clock × the setting value of ATT[2:0].<br>This bit is effective only at SPTR.ATEN = 1. This setting timing is limited by setting the base clock cycles. For details, see section 27.3.11. The function of adjust transmit timing (Asynchronous Mode). | R/W*2 |
| 7 | AET | Adjustment edge for transmit timing<br>The adjustable edge is set by this bit.<br>When SPTR.TINV is 0,<br>0: Adjust the rising edge timing.<br>1: Adjust the falling edge timing.<br>When SPTR.TINV is 1,<br>0: Adjust the falling edge timing.<br>1: Adjust the rising edge timing.<br>This bit is effective only at SPTR.ATEN = 1. For details, see section 27.3.11. The function of adjust transmit timing (Asynchronous Mode). | R/W*2 |

Note 1. Write this bit only when SPTR.ASEN = 0.
Note 2. Write this bit only when SPTR.ATEN = 0.

This register controls adjustment of receive sampling timing and transmit timing. This register is effective only when asynchronous mode using internal clock.

For details of adjustment receive sampling timing by this register, see section 27.3.10. The function of adjust receive sampling timing (Asynchronous Mode).

For details of adjustment transmit timing by this register, see section 27.3.11. The function of adjust transmit timing (Asynchronous Mode).

Note: Sentences and a timing chart of the IP operation explanation (except section 27.1. Overview, section 27.2. Register Descriptions, section 27.3.10. The function of adjust receive sampling timing (Asynchronous Mode) and section 27.3.11. The function of adjust transmit timing (Asynchronous Mode)) are mentioned by the condition that the receive sampling timing and transmit timing adjustments are disabled (SPTR.ASEN = 0, SPTR.ATEN = 0).

## 27.2.36 MMR : Manchester Mode Register

Base address: SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x20

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | MANEN | SBSEL | SYNSEL | SYNVAL | — | ERTEN | TMPOL | RMPOL |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RMPOL | Polarity of Received Manchester Code<br>Sets the polarity of the received Manchester code<br>0: Logic 0 is coded as a zero-to-one transition in Manchester code<br>Logic 1 is coded as a one-to-zero transition in Manchester code<br>1: Logic 0 is coded as a one-to-zero transition in Manchester code<br>Logic 1 is coded as a zero-to-one transition in Manchester code | R/W*1 |
| 1 | TMPOL | Polarity of Transmit Manchester Code<br>Sets the polarity of the transmit Manchester code<br>0: Logic 0 is coded as a zero-to-one transition in Manchester code<br>Logic 1 is coded as a one-to-zero transition in Manchester code<br>1: Logic 0 is coded as a one-to-zero transition in Manchester code<br>Logic 1 is coded as a zero-to-one transition in Manchester code | R/W*1 |
| 2 | ERTEN | Manchester Edge Retiming Enable<br>Sets the receive retiming function<br>0: Disables the receive retiming function<br>1: Enables the receive retiming function | R/W*1 |
| 3 | — | This bit is read as 0. The write value should be 0. | R |
| 4 | SYNVAL | SYNC value Setting<br>Sets the SYNC type of the start bit(s) in the Manchester code<br>When the start bit area consists of one bit. (SBSEL = 0)<br>● when transmitting<br>0: The start bit is added as a zero-to-one transition.<br>1: The start bit is added as a one-to-zero transition.<br>● when receiving<br>0: Only when the start bit is a zero-to-one transition, the data is received. The other cases are judged as an error.<br>1: Only when the start bit is a one-to-zero transition, the data is received. The other cases are judged as an error.<br>When the start bit area consists of three bits. (SBSEL = 1)<br>● when transmitting<br>0: The start bits are added as a zero-to-one transition. (DATA SYNC)<br>1: The start bits are coded as a one-to-zero transition. (COMMAND SYNC)<br>● when receiving<br>When the start bit area consists of three bits, data is received regardless of the value of this bit. | R/W*1 |
| 5 | SYNSEL | SYNC Select<br>0: The start bit pattern is set with the SYNVAL bit<br>1: The start bit pattern is set with the TSYNC bit. | R/W*1 |
| 6 | SBSEL | Start Bit Select<br>0: The start bit area consists of one bit.<br>1: The start bit area consists of three bits (COMMAND SYNC or DATA SYNC) | R/W*1 |
| 7 | MANEN | Manchester Mode Enable<br>Sets the Manchester mode<br>0: Disables the Manchester mode<br>1: Enables the Manchester mode | R/W*1 |

Note: Bits 6 to 0 in this register are valid only when the Manchester mode is enabled.(MANEN = ”1”) in bit 7.
Note 1. Writing to these bits is only possible when the RE and TE bits in SCR are 0 (both serial transmission and reception are disabled).

This register is used to enable or disable the Manchester mode, set the start bit area, and set the logic polarity.

### RMPOL bit (Polarity of Received Manchester Code)

This bit sets the polarity of the received Manchester code. For details, see section 27.5.7. Serial Data Reception in Manchester Mode.

### TMPOL bit (Polarity of Transmit Manchester Code)

This bit sets the polarity of the transmit Manchester code. For details, see section 27.5.6. Serial data transmission in Manchester mode.

### ERTEN bit (Manchester Edge Retiming Enable)

This bit sets the receive retiming function in Manchester mode.

For information on the receive retiming function, see section 27.5.9. Receive Retiming.

### SYNVAL bit (SYNC value Setting)

This bit is valid when the SYNSEL bit of this register is set to 0.

The SYNC type can be set by combining this bit and the SBSEL bit.

For the start bit area determined by the combination of this bit and the SBSEL bit, see Figure 27.52 and Figure 27.53.

### SYNSEL bit (SYNC Select)

This bit is valid when the SBSEL bit of this register is set to 1. This bit determines the destination to be referred to for setting the SYNC type of the start bit area added to Manchester frames.

When this bit is set to 0, the SYNVAL bit of this register is referred to.

When this bit is set to 1, the TSYNC bit in the TDRH register is referred to.

For detail, see the bit table in section 27.2.36. MMR : Manchester Mode Register.

### SBSEL bit (Start Bit Select)

This bit sets the start bit area in Manchester frames.

When this bit is set to 1, the start bit area added to each frame consists of three bits, and the SYNSEL and SYNVAL bits in this register are valid.

When this bit is set to 0, the start bit area added to each frame consists of one bit.

### MANEN bit (Manchester Mode Enable)

This bit sets the Manchester mode.

When this bit is set to 0, the Manchester mode is disabled.

When this bit is set to 1, the Manchester mode is enabled.

## 27.2.37 TMPR : Transmit Manchester Preface Setting Register

Base address: SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x22

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | TPPAT[1:0] | | TPLEN[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | TPLEN | Transmit preface length<br>Set the preface length of the transmit data in Manchester mode<br>0x0: Disables the transmit preface generation<br>Others: Transmit preface length (bit length) | R/W*1 |
| 5:4 | TPPAT | Transmit preface pattern<br>Set the preface pattern of the transmit data<br>0 0: ALL ZERO<br>0 1: ZERO ONE<br>1 0: ONE ZERO<br>1 1: ALL ONE | R/W*1 |
| 7:6 | — | The read value is undefined. The write value should be 0. | R |

Note: This register is valid only when the Manchester mode is enabled (MMR.MANEN = 1).
Note 1. Writing to these bits is only possible when the RE and TE bits in SCR are 0 (both serial transmission and reception are disabled).

This register is used to set the preface length and preface pattern of the transmit data in Manchester mode.

### TPLEN bit (Transmit preface length)

These bits set the preface bit length of the transmit data in Manchester mode.

The settable range is 0x0 to 0xF (0 to 15). 0x0 disables the transmit preface, which is not added.

### TPPAT bit (Transmit preface pattern)

These bits set one of the four preface patterns in Manchester mode.

When these bits are set to 00b, the preface area is set to all zeros.

When these bits are set to 01b, the preface area is set to the zero-one-zero-one pattern.

When these bits are set to 10b, the preface area is set to the one-zero-one-zero pattern.

When these bits are set to 11b, the preface area is set to all ones.

Note: For the transmit and receive data when the TPPAT bits are set, see Figure 27.51.

## 27.2.38 RMPR : Receive Manchester Preface Setting Register

Base address: SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x23

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | RPPAT[1:0] | | RPLEN[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | RPLEN | Receive Preface Length<br>Set the preface length in received frames when Manchester mode is enabled<br>0: Disables the receive preface generation<br>Others: Receive preface length (bit length) | R/W*1 |
| 5:4 | RPPAT | Receive Preface Pattern<br>Set the preface pattern of received frames<br>0 0: ALL ZERO<br>0 1: ZERO ONE<br>1 0: ONE ZERO<br>1 1: ALL ONE | R/W*1 |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R |

Note: This register is valid only when the Manchester mode is enabled (MMR.MANEN = 1).
Note 1. Writing to these bits is only possible when the RE and TE bits in SCR are 0 (both serial transmission and reception are disabled).

This register is used to set the preface length and preface pattern of the received frames in Manchester mode.

### RPLEN bit (Receive Preface Length)

These bits set the preface bit length of the received frames in Manchester mode.

The settable range is 0x0 to 0xF (0 to 15). 0x0 disables the receive preface, which is not added. When 0x1 to 0xF is set, the set value is handled as the receive preface bit length.

### RPPAT bit (Receive Preface Pattern)

These bits set one of the four preface patterns in Manchester mode.

When these bits are set to 00b, the preface area is handled as all zeros.

When these bits are set to 01b, the preface area is handled as the zero-one-zero-one pattern.

When these bits are set to 10b, the preface area is handled as the one-zero-one-zero pattern.

When these bits are set to 11b, the preface area is handled as all ones.

Note: For the transmit and receive data when the RPPAT bits are set, see Figure 27.51.

## 27.2.39 MESR : Manchester Extended Error Status Register

Base address: SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x24

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | SBER | SYER | PFER |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PFER | Preface Error flag<br>This bit is set when a preface error (pattern mismatch) is detected<br>0: No preface error detected<br>1: Preface error detected | R/(W)*1 |
| 1 | SYER | SYNC Error flag<br>This bit is set when no edge is detected in the adjustable range during receive retiming<br>0: No receive SYNC error detected<br>1: Receive SYNC error detected | R/(W)*1 |
| 2 | SBER | Start Bit Error flag<br>This bit is set when a pattern mismatch in the start bit area is detected<br>0: No start bit error detected<br>1: Start bit error detected | R/(W)*1 |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R |

Note: This register is valid only when the Manchester mode is enabled (MMR.MANEN = 1).
Note 1. Only 0 can be written to this bit, to clear the flag. To clear the flag, confirm that the flag is 1 before setting it to 0.

This register indicates an error status when receiving frames in Manchester mode.

A preface error, receive SYNC error or start bit error was detected.

### PFER bit (Preface Error flag)

This bit indicates that a preface error was detected when receiving frames in Manchester mode.

[Setting condition]

- When detecting a preface error when receiving frames in Manchester mode
  The following operations are performed when a preface error occurs.
  When MECR.PFEREN = 1
  The received data is not transferred to the RDR register and no RXI interrupt request occurs. Instead, an ERI interrupt request occurs. Note that when the PFER flag is being set to 1, the subsequently received data is not transferred to the RDR register.
  When MECR.PFEREN = 0
  The received data is transferred to the RDR register and an RXI interrupt request occurs. An ERI interrupt request is not generated. The subsequent receive operations are not affected even with the PFER flag being set to 1.

[Clearing condition]

- When 0 is written to the bit after it was read as 1

Even if the SCR.RE bit is cleared, the PFER flag is not affected, and the previous state is retained.

### SYER bit (SYNC Error flag)

This bit indicates that a receive SYNC error was detected when receiving frames in Manchester mode with

MMR.ERTEN = 1 (Manchester edge retiming enabled).

[Setting condition]

- When detecting a receive SYNC error when receiving frames in Manchester mode
  The following operations are performed when a receive SYNC error occurs.
  When MECR.SYEREN = 1

Although the received data is transferred to the RDR register, no RXI interrupt request occurs. Instead, an ERI interrupt request occurs. Note that when the SYER flag is being set to 1, the subsequently received data is not transferred to the RDR register.
When MECR.SYEREN = 0
The received data is transferred to the RDR register and an RXI interrupt request occurs. An ERI interrupt request is not generated. The subsequent receive operations are not affected even with the SYER flag being set to 1.

[Clearing condition]

- When 0 is written to the bit after it was read as 1

Even if the SCR.RE bit is cleared, the SYER flag is not affected, and the previous state is retained.

#### SBER bit (Start Bit Error flag)

This bit indicates that a start bit error was detected when receiving frames in Manchester mode.

[Setting condition]

- When detecting a start bit error when receiving frames in Manchester mode
The following operations are performed when a start bit error occurs.
When MECR.SBEREN = 1
The received data is not transferred to the RDR register and no RXI interrupt request occurs. Instead, an ERI interrupt request occurs. Note that when the SBER flag is being set to 1, the subsequently received data is not transferred to the RDR register.
When MECR.SBEREN = 0
The received data is transferred to the RDR register and an RXI interrupt request occurs. An ERI interrupt request is not generated. The subsequent receive operations are not affected even with the SBER flag being set to 1.

[Clearing condition]

- When 0 is written to the bit after it was read as 1

Even if the SCR.RE bit is cleared, the SBER flag is not affected, and the previous state is retained.

### 27.2.40 MECR : Manchester Extended Error Control Register

Base address: SCIk = 0x4011_8000 + 0x0100 × k (k = 3, 4)

Offset address: 0x25

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | SBEREN | SYEREN | PFEREN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PFEREN | Preface Error Enable<br>Specifies whether to handle a preface error as an interrupt source<br>0: Does not handle a preface error as an interrupt source<br>1: Handles a preface error as an interrupt source | R/W |
| 1 | SYEREN | Receive SYNC Error Enable<br>Specifies whether to handle a receive SYNC error as an interrupt source<br>0: Does not handle a receive SYNC error as an interrupt source<br>1: Handles a receive SYNC error as an interrupt source | R/W |
| 2 | SBEREN | Start Bit Error Enable<br>Specifies whether to handle a start bit error as an interrupt source<br>0: Does not handle a start bit error as an interrupt source<br>1: Handles a start bit error as an interrupt source | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R |

Note: This register is valid only when the Manchester mode is enabled (MMR.MANEN = 1).

This register is used to specify whether to handle a preface error, receive SYNC error, or a start bit error as an interrupt source in Manchester mode. If those errors are handled as interrupt sources, interrupt requests and event requests are generated at the occurrence of each error, and the next reception is not performed until the corresponding error flag is cleared.

Set this register when MMR.MANEN = 0. However, do not change this register during communication.

### PFEREN bit (Preface Error Enable)

This bit specifies whether to handle a preface error as an interrupt source.

When it is set to 0, a preface error is not handled as an interrupt source. When it is set to 1, a preface error is handled as an interrupt source.

### SYEREN bit (Receive SYNC Error Enable)

This bit specifies whether to handle a receive SYNC error as an interrupt source.

When it is set to 0, a receive SYNC error is not handled as an interrupt source. When it is set to 1, a receive SYNC error is handled as an interrupt source.

### SBEREN bit (Start Bit Error Enable)

This bit specifies whether to handle a start bit error as an interrupt source.

When it is set to 0, a start bit error is not handled as an interrupt source. When it is set to 1, a start bit error is handled as an interrupt source.

## 27.2.41 ESMER : Extended Serial Module Enable Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x20

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | ESME |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ESME | Extended Serial Mode Enable<br>0: The extended serial mode is disabled.<br>1: The extended serial mode is enabled. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

### ESME bit (Extended Serial Mode Enable)

When the ESME bit is 1, the facilities of the extended serial mode control section are enabled.

When the ESME bit is 0, the extended serial mode control section is initialized.

**Table 27.26 Settings of the ESME Bit and Timer Operation Mode**

| ESME Bit | Timer Mode | Break Field Low Width Determination Mode | Break Field Low Width Output Mode |
|---|---|---|---|
| 0 | Available*1 | Not available | Not available |
| 1 | Available | Available | Available |

Note 1. Operation is only possible with PCLK selected.

## 27.2.42 CR0 : Control Register 0

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x21

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | BRME | RXDSF | SFSF | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | This bit is read as 0. The write value should be 0. | R/W |
| 1 | SFSF | Start Frame Status Flag<br>0: Start Frame detection function is disabled.<br>1: Start Frame detection function is enabled. | R |
| 2 | RXDSF | RXDXn Input Status Flag<br>0: RXDXn input is enabled.<br>1: RXDXn input is disabled. | R |
| 3 | BRME | Bit Rate Measurement Enable<br>0: Measurement of bit rate is disabled.<br>1: Measurement of bit rate is enabled. | R/W |
| 7:4 | — | These bits are read as 0. The write value should be 0. | R/W |

## 27.2.43 CR1 : Control Register 1

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x22

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | PIBS[2:0] | | | PIBE | CF1DS[1:0] | | CF0RE | BFE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BFE | Break Field Enable<br>0: Break Field detection is disabled.<br>1: Break Field detection is enabled. | R/W |
| 1 | CF0RE | Control Field 0 Reception Enable<br>0: Reception of Control Field 0 is disabled.<br>1: Reception of Control Field 0 is enabled. | R/W |
| 3:2 | CF1DS[1:0] | Control Field 1 Data Register Select<br>0 0: Selects comparison with the value in PCF1DR.<br>0 1: Selects comparison with the value in SCF1DR.<br>1 0: Selects comparison with the values in PCF1DR and SCF1DR.<br>1 1: Setting prohibited. | R/W |
| 4 | PIBE | Priority Interrupt Bit Enable<br>0: The priority interrupt bit is disabled.<br>1: The priority interrupt bit is enabled. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:5 | PIBS[2:0] | Priority Interrupt Bit Select<br>0 0 0: 0th bit of Control Field 1<br>0 0 1: 1st bit of Control Field 1<br>0 1 0: 2nd bit of Control Field 1<br>0 1 1: 3rd bit of Control Field 1<br>1 0 0: 4th bit of Control Field 1<br>1 0 1: 5th bit of Control Field 1<br>1 1 0: 6th bit of Control Field 1<br>1 1 1: 7th bit of Control Field 1 | R/W |

## 27.2.44 CR2 : Control Register 2

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x23

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | RTS[1:0] | | BCCS[1:0] | | — | DFCS[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | DFCS[2:0] | RXDXn Signal Digital Filter Clock Select<br>0 0 0: Filter is disabled.<br>0 0 1: Filter clock is SCI base clock*1 *2<br>0 1 0: Filter clock is PCLK/8<br>0 1 1: Filter clock is PCLK/16<br>1 0 0: Filter clock is PCLK/32<br>1 0 1: Filter clock is PCLK/64<br>1 1 0: Filter clock is PCLK/128<br>1 1 1: Setting prohibited | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 5:4 | BCCS[1:0] | Bus Collision Detection Clock Select<br>● When SEMR.BGDM = 0 or SEMR.BGDM = 1 and SMR.CKS[1:0] = a value other than 00b<br>0 0: SCI base clock<br>0 1: SCI base clock frequency divided by 2<br>1 0: SCI base clock frequency divided by 4<br>1 1: Setting prohibited<br>● When SEMR.BGDM = 1 and SMR.CKS[1:0] = 00b<br>0 0: SCI base clock frequency divided by 2<br>0 1: SCI base clock frequency divided by 4<br>1 0: Setting prohibited<br>1 1: Setting prohibited | R/W |
| 7:6 | RTS[1:0] | RXDXn Reception Sampling Timing Select<br>● When SCIn.SEMR.ABCS = 0<br>0 0: Rising edge of the 8th cycle of SCI base clock<br>0 1: Rising edge of the 10th cycle of SCI base clock<br>1 0: Rising edge of the 12th cycle of SCI base clock<br>1 1: Rising edge of the 14th cycle of SCI base clock<br>● When SCIn.SEMR.ABCS = 1<br>0 0: Rising edge of the 4th cycle of SCI base clock<br>0 1: Rising edge of the 5th cycle of SCI base clock<br>1 0: Rising edge of the 6th cycle of SCI base clock<br>1 1: Rising edge of the 7th cycle of SCI base clock | R/W |

Note: The period of the SCI base clock is 1/16 of a single bit period when the SCIn.SEMR.ABCS is 0, and 1/8 of a single bit period when the SCIn.SEMR.ABCS is 1.

Note 1. To use the SCI base clock, set the SCIn.SCR.TE bit to 1.

Note 2. The SCI base clock divided by 2 is the filter clock when the SEMR.BGDM bit is 1 and the SMR.CKS[1:0] bits are 00b.

## 27.2.45 CR3 : Control Register 3

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x24

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | SDST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SDST | Start Frame Detection Start<br>0: Detection of Start Frame is not performed.<br>1: Detection of Start Frame is performed. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

**SDST bit (Start Frame Detection Start)**

Detection of a Start Frame begins when this bit is set to 1. The bit is read as 0.

## 27.2.46 PCR : Port Control Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x25

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | SHARPS | — | — | RXDXPS | TXDXPS |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TXDXPS | TXDXn Signal Polarity Select<br>0: The polarity of TXDXn signal is not inverted for output.<br>1: The polarity of TXDXn signal is inverted for output. | R/W |
| 1 | RXDXPS | RXDXn Signal Polarity Select<br>0: The polarity of RXDXn signal is not inverted for input.<br>1: The polarity of RXDXn signal is inverted for input. | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | SHARPS | TXDXn/RXDXn Pin Multiplexing Select<br>0: The TXDXn and RXDXn pins are independent.<br>1: The TXDXn and RXDXn signals are multiplexed on the same pin. | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

**SHARPS bit (TXDXn/RXDXn Pin Multiplexing Select)**

When this bit is set to 1, the TXDXn and RXDXn signals are multiplexed on the same pin so that half-duplex communications become possible.

## 27.2.47 ICR : Interrupt Control Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x26

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | AEDIE | BCDIE | PIBDIE | CF1MIE | CF0MIE | BFDIE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BFDIE | Break Field Low Width Detected Interrupt Enable<br>0: Interrupts on detection of the low width for a Break Field are disabled.<br>1: Interrupts on detection of the low width for a Break Field are enabled. | R/W |
| 1 | CF0MIE | Control Field 0 Match Detected Interrupt Enable<br>0: Interrupts on detection of a match with Control Field 0 are disabled.<br>1: Interrupts on detection of a match with Control Field 0 are enabled. | R/W |
| 2 | CF1MIE | Control Field 1 Match Detected Interrupt Enable<br>0: Interrupts on detection of a match with Control Field 1 are disabled.<br>1: Interrupts on detection of a match with Control Field 1 are enabled. | R/W |
| 3 | PIBDIE | Priority Interrupt Bit Detected Interrupt Enable<br>0: Interrupts on detection of the priority interrupt bit are disabled.<br>1: Interrupts on detection of the priority interrupt bit are enabled. | R/W |
| 4 | BCDIE | Bus Collision Detected Interrupt Enable<br>0: Interrupts on detection of a bus collision are disabled.<br>1: Interrupts on detection of a bus collision are enabled. | R/W |
| 5 | AEDIE | Valid Edge Detected Interrupt Enable<br>0: Interrupts on detection of a valid edge are disabled.<br>1: Interrupts on detection of a valid edge are enabled. | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |

## 27.2.48 STR : Status Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x27

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | AEDF | BCDF | PIBDF | CF1MF | CF0MF | BFDF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BFDF | Break Field Low Width Detection Flag<br>[Setting conditions]<br>● Detection of the low width for a Break Field<br>● Completion of the output of the low width for a Break Field<br>● Underflow of the timer<br>[Clearing condition]<br>● Writing 1 to the BFDCL bit in STCR | R |
| 1 | CF0MF | Control Field 0 Match Flag<br>[Setting conditions]<br>● A match between the value received in Control Field 0 and the set value.<br>[Clearing condition]<br>● Writing 1 to the CF0MCL bit in STCR | R |
| 2 | CF1MF | Control Field 1 Match Flag<br>[Setting conditions]<br>● A match between the data received in Control Field 1 and the set values.<br>[Clearing condition]<br>● Writing 1 to the CF1MCL bit in STCR | R |
| 3 | PIBDF | Priority Interrupt Bit Detection Flag<br>[Setting conditions]<br>● Detection of the priority interrupt bit<br>[Clearing condition]<br>● Writing 1 to the PIBDCL bit in STCR | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4 | BCDF | Bus Collision Detected Flag<br>[Setting conditions]<br>● Detection of the bus collision<br>[Clearing condition]<br>● Writing 1 to the BCDCL bit in STCR | R |
| 5 | AEDF | Valid Edge Detection Flag<br>[Setting conditions]<br>● Detection of a valid edge<br>[Clearing condition]<br>● Writing 1 to the AEDCL bit in STCR | R |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R |

## 27.2.49 STCR : Status Clear Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x28

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | AEDCL | BCDCL | PIBDCL | CF1MCL | CF0MCL | BFDCL |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BFDCL | BFDF Clear<br>Setting this bit to 1 clears the STR.BFDF flag. This bit is read as 0. | R/W |
| 1 | CF0MCL | CF0MF Clear<br>Setting this bit to 1 clears the STR.CF0MF flag. This bit is read as 0. | R/W |
| 2 | CF1MCL | CF1MF Clear<br>Setting this bit to 1 clears the STR.CF1MF flag. This bit is read as 0. | R/W |
| 3 | PIBDCL | PIBDF Clear<br>Setting this bit to 1 clears the STR.PIBDF flag. This bit is read as 0. | R/W |
| 4 | BCDCL | BCDF Clear<br>Setting this bit to 1 clears the STR.BCDF flag. This bit is read as 0. | R/W |
| 5 | AEDCL | AEDF Clear<br>Setting this bit to 1 clears the STR.AEDF flag. This bit is read as 0. | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |

## 27.2.50 CF0DR : Control Field 0 Data Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x29

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The CF0DR register is an 8-bit readable and writable register that holds a value for comparison with Control Field 0.

## 27.2.51 CF0CR : Control Field 0 Compare Enable Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x2A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CF0CE7 | CF0CE6 | CF0CE5 | CF0CE4 | CF0CE3 | CF0CE2 | CF0CE1 | CF0CE0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CF0CE0 | Control Field 0 Bit 0 Compare Enable<br>0: Comparison with bit 0 of Control Field 0 is disabled.<br>1: Comparison with bit 0 of Control Field 0 is enabled. | R/W |
| 1 | CF0CE1 | Control Field 1 Bit 0 Compare Enable<br>0: Comparison with bit 1 of Control Field 0 is disabled.<br>1: Comparison with bit 1 of Control Field 0 is enabled. | R/W |
| 2 | CF0CE2 | Control Field 2 Bit 0 Compare Enable<br>0: Comparison with bit 2 of Control Field 0 is disabled.<br>1: Comparison with bit 2 of Control Field 0 is enabled. | R/W |
| 3 | CF0CE3 | Control Field 3 Bit 0 Compare Enable<br>0: Comparison with bit 3 of Control Field 0 is disabled.<br>1: Comparison with bit 3 of Control Field 0 is enabled. | R/W |
| 4 | CF0CE4 | Control Field 4 Bit 0 Compare Enable<br>0: Comparison with bit 4 of Control Field 0 is disabled.<br>1: Comparison with bit 4 of Control Field 0 is enabled. | R/W |
| 5 | CF0CE5 | Control Field 5 Bit 0 Compare Enable<br>0: Comparison with bit 5 of Control Field 0 is disabled.<br>1: Comparison with bit 5 of Control Field 0 is enabled. | R/W |
| 6 | CF0CE6 | Control Field 6 Bit 0 Compare Enable<br>0: Comparison with bit 6 of Control Field 0 is disabled.<br>1: Comparison with bit 6 of Control Field 0 is enabled. | R/W |
| 7 | CF0CE7 | Control Field 7 Bit 0 Compare Enable<br>0: Comparison with bit 7 of Control Field 0 is disabled.<br>1: Comparison with bit 7 of Control Field 0 is enabled. | R/W |

## 27.2.52 CF0RR : Control Field 0 Receive Data Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x2B

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

CF0RR is a readable register that holds the value received in Control Field 0.

## 27.2.53 PCF1DR : Primary Control Field 1 Data Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x2C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

PCF1DR is an 8-bit readable and writable register that holds the 8-bit primary value for comparison with Control Field 1.

## 27.2.54 SCF1DR : Secondary Control Field 1 Data Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x2D

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SCF1DR is an 8-bit readable and writable register that holds the 8-bit secondary value for comparison with Control Field 1.

## 27.2.55 CF1CR : Control Field 1 Compare Enable Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x2E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CF1CE7 | CF1CE6 | CF1CE5 | CF1CE4 | CF1CE3 | CF1CE2 | CF1CE1 | CF1CE0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CF1CE0 | Control Field 1 Bit 0 Compare Enable<br>0: Comparison with bit 0 of Control Field 1 is disabled.<br>1: Comparison with bit 0 of Control Field 1 is enabled. | R/W |
| 1 | CF1CE1 | Control Field 1 Bit 1 Compare Enable<br>0: Comparison with bit 1 of Control Field 1 is disabled.<br>1: Comparison with bit 1 of Control Field 1 is enabled. | R/W |
| 2 | CF1CE2 | Control Field 1 Bit 2 Compare Enable<br>0: Comparison with bit 2 of Control Field 1 is disabled.<br>1: Comparison with bit 2 of Control Field 1 is enabled. | R/W |
| 3 | CF1CE3 | Control Field 1 Bit 3 Compare Enable<br>0: Comparison with bit 3 of Control Field 1 is disabled.<br>1: Comparison with bit 3 of Control Field 1 is enabled. | R/W |
| 4 | CF1CE4 | Control Field 1 Bit 4 Compare Enable<br>0: Comparison with bit 4 of Control Field 1 is disabled.<br>1: Comparison with bit 4 of Control Field 1 is enabled. | R/W |
| 5 | CF1CE5 | Control Field 1 Bit 5 Compare Enable<br>0: Comparison with bit 5 of Control Field 1 is disabled.<br>1: Comparison with bit 5 of Control Field 1 is enabled. | R/W |
| 6 | CF1CE6 | Control Field 1 Bit 6 Compare Enable<br>0: Comparison with bit 6 of Control Field 1 is disabled.<br>1: Comparison with bit 6 of Control Field 1 is enabled. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | CF1CE7 | Control Field 1 Bit 7 Compare Enable<br>0: Comparison with bit 7 of Control Field 1 is disabled.<br>1: Comparison with bit 7 of Control Field 1 is enabled. | R/W |

## 27.2.56 CF1RR : Control Field 1 Receive Data Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x2F

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

CF1RR is a readable register that holds the value received in Control Field 1.

## 27.2.57 TCR : Timer Control Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x30

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | TCST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TCST | Timer Count Start<br>0: Stops the timer counting<br>1: Starts the timer counting | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

## 27.2.58 TMR : Timer Mode Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x31

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | TCSS[2:0] | | | TWRC | — | TOMS[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | TOMS[1:0] | Timer Operating Mode Select*1<br>0 0: Timer mode<br>0 1: Break Field low width determination mode<br>1 0: Break Field low width output mode<br>1 1: Setting prohibited | R/W |
| 2 | — | This bit is read as 0. The write value should be 0. | R/W |
| 3 | TWRC | Counter Write Control<br>0: Data is written to the reload register and counter<br>1: Data is written to the reload register only | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:4 | TCSS[2:0] | Timer Count Clock Source Select*1<br>0 0 0: PCLK<br>0 0 1: PCLK/2<br>0 1 0: PCLK/4<br>0 1 1: PCLK/8<br>1 0 0: PCLK/16<br>1 0 1: PCLK/32<br>1 1 0: PCLK/64<br>1 1 1: PCLK/128 | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |

Note 1. Rewrite the TOMS[1:0] and TCSS[2:0] bits only when the timer is stopped (TCST = 0).

**TWRC bit (Counter Write Control)**

This bit determines whether a value written to TPRE or TCNT is written to the reload register only or is written to both the reload register and the counter.

## 27.2.59 TPRE : Timer Prescaler Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x32

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

TPRE consists of an 8-bit reload register, a read buffer, and a counter, each of which has 0xFF as its initial value. The counter counts down in synchronization with the counter clock selected by the TMR.TCSS[2:0] bits, and is reloaded with the value in the reload register when it underflows. Underflows of this register provide the clock source to drive counting by the TCNT register. The reload register and read buffer are allocated to the same address. Data is written to the reload register in writing, and the counter value that has been transferred to the read buffer is returned in reading.

It takes one PCLK cycle to load a value from the reload register to the counter.

## 27.2.60 TCNT : Timer Count Register

Base address: SCIm = 0x4011_8000 + 0x0100 × m (m = 1, 2)

Offset address: 0x33

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

TCNT consists of an 8-bit reload register, a read buffer, and a counter, each of which has 0xFF as its initial value. This down-counter counts underflows of the TPRE register until the TCNT register underflows, and is then reloaded with the value from the reload register. The reload register and read buffer are allocated to the same address. Data is written to the reload register in writing, and the counter value that has been transferred to the read buffer is returned in reading.

It takes one PCLK cycle to load a value from the reload register to the counter.

## 27.3 Operation in Asynchronous Mode

Figure 27.6 shows the general format for asynchronous serial communications. One frame consists of a start bit (low level), transmit or receive data, a parity bit, and stop bits (high level). In asynchronous serial communications, the communications line is usually held in the mark state (high level).

The SCI monitors the communications line. When the SCI detects a low, it regards that as a start bit and starts serial communication.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communications. Both the transmitter and receiver have a double-buffered structure in addition to FIFO mode, so that data can be read or written during transmission or reception, enabling continuous data transmission and reception.

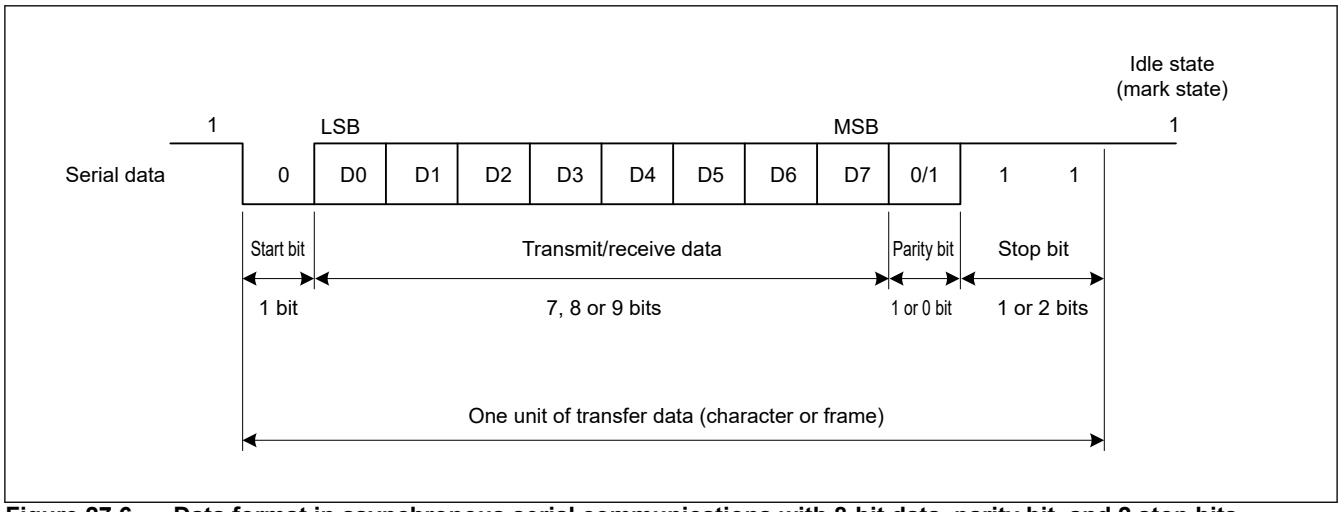



**Figure 27.6 Data format in asynchronous serial communications with 8-bit data, parity bit, and 2 stop bits**

## 27.3.1 Serial Data Transfer Format

Table 27.27 lists the serial data transfer formats that can be used in asynchronous mode. Any of 18 transfer formats can be selected with the SMR and SCMR settings. For details on the multi-processor function, see section 27.4. Multi-Processor Communication Function.

**Table 27.27 Serial transfer formats in asynchronous mode (1 of 2)**

| SCMR setting | SMR setting | | | | Serial transfer format and frame length |
|---|---|---|---|---|---|
| CHR1 | CHR | PE | MP | STOP | 1 2 3 4 5 6 7 8 9 10 11 12 13 |
| 0 | 0 | 0 | 0 | 0 | ST 9-bit data SP |
| 0 | 0 | 0 | 0 | 1 | ST 9-bit data SP SP |
| 0 | 0 | 1 | 0 | 0 | ST 9-bit data P SP |
| 0 | 0 | 1 | 0 | 1 | ST 9-bit data P SP SP |
| 1 | 0 | 0 | 0 | 0 | ST 8-bit data SP |
| 1 | 0 | 0 | 0 | 1 | ST 8-bit data SP SP |
| 1 | 0 | 1 | 0 | 0 | ST 8-bit data P SP |
| 1 | 0 | 1 | 0 | 1 | ST 8-bit data P SP SP |
| 1 | 1 | 0 | 0 | 0 | ST 7-bit data SP |
| 1 | 1 | 0 | 0 | 1 | ST 7-bit data SP SP |

**Table 27.27 Serial transfer formats in asynchronous mode (2 of 2)**

| SCMR setting | SMR setting | | | | Serial transfer format and frame length |
|---|---|---|---|---|---|
| CHR1 | CHR | PE | MP | STOP | 1 2 3 4 5 6 7 8 9 10 11 12 13 |
| 1 | 1 | 1 | 0 | 0 | ST 7-bit data P SP |
| 1 | 1 | 1 | 0 | 1 | ST 7-bit data P SP SP |
| 0 | 0 | — | 1 | 0 | ST 9-bit data MPB SP |
| 0 | 0 | — | 1 | 1 | ST 9-bit data MPB SP SP |
| 1 | 0 | — | 1 | 0 | ST 8-bit data MPB SP |
| 1 | 0 | — | 1 | 1 | ST 8-bit data MPB SP SP |
| 1 | 1 | — | 1 | 0 | ST 7-bit data MPB SP |
| 1 | 1 | — | 1 | 1 | ST 7-bit data MPB SP SP |

ST: Start bit
SP: Stop bit
P: Parity bit
MPB: Multi-processor bit

## 27.3.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI operates on a base clock with a frequency of 16 times[*1] the bit rate.

In reception, the SCI samples the falling edge of the start bit using the base clock, and performs internal synchronization.[*2]

Because receive data is sampled on the rising edge of the 8th pulse[*1] of the base clock, data is latched at the middle of each bit (when sampling timing does not adjust (SPTR.ASEN = 0) ), as shown in Figure 27.7 The reception margin in asynchronous mode is determined by the following formula (1):

$$M = \left|\left(0.5 - \frac{1}{2N}\right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F)\right| \times 100\ [\%] \quad \ldots \quad \text{Formula (1)}$$

Note: M: Reception margin
N: Ratio of bit rate to clock
(N = 16 when SEMR.ABCSE = 0 and SEMR.ABCS = 0,
N = 8 when SEMR.ABCS = 1,
N = 6 when SEMR.ABCSE = 1)
D: Duty cycle of clock (D = 0.5 to 1.0)
L: Frame length (L = 9 to 13)
F: Absolute value of clock frequency deviation

Assuming values of F = 0 and D = 0.5 in formula (1), the reception margin is determined using the following formula:
$M = \{0.5 - 1/(2 \times 16)\} \times 100\ (\%) = 46.875\ \%$

This represents the computed value. Renesas recommends a margin of 20% to 30% in system design.

Note 1. In this example, the SEMR.ABCS bit is 0 and the SEMR.ABCSE is 0. When the ABCS bit is 1 and the ABCSE bit is 0, a frequency of 8 times the bit rate is used as a base clock, and receive data is sampled on the rising edge of the 4th pulse of the base clock.

When the ABCSE bit is 1, a sextuple frequency of a bit rate is used as a base clock, and receive data is sampled on the rising edge of the 3rd pulse of the base clock.

Note 2. The determination condition of the start bit is as follows.

The function of adjust sampling timing is OFF (ASEN = 0):

The determination condition of a start bit is that Low beyond half bit length continues. It is same as the sampling timing. In Figure 27.7, Low period should be kept over 8-cycles to detect a start bit. If Low period does not keep over 8-cycles, the IP judges this as a noise. So, the IP does not start reception and wait start bit.

The function of adjust sampling timing is ON (ASEN = 1):

The determination condition of a start bit is that Low keeps up until the sampling timing. Adjusting the sampling timing forward (AJD = 1) increases the possibility of erroneously determining a noise as the start bit.

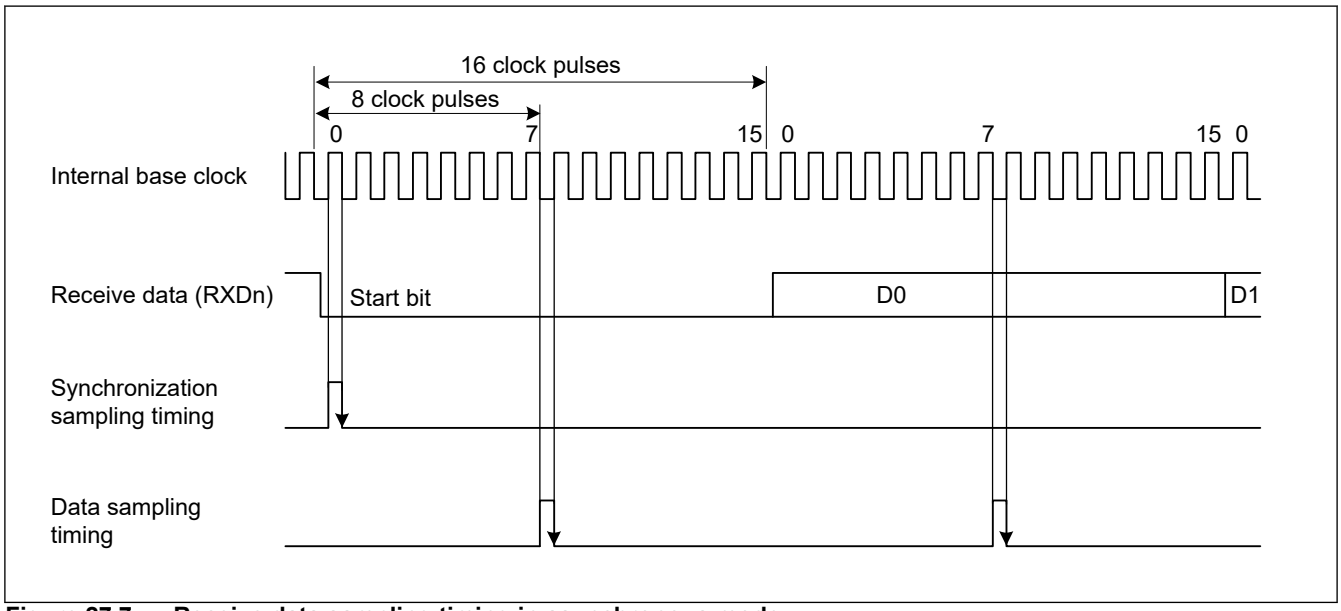



**Figure 27.7 Receive data sampling timing in asynchronous mode**

## 27.3.3 Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input to the SCKn pin can be selected as the transfer clock of the SCI, based on the SMR.CM and SCR.CKE[1:0] settings.

When an external clock is input to the SCKn pin, the clock frequency must be 16 times the bit rate (when SEMR.ABCS = 0) or 8 times the bit rate (when SEMR.ABCS = 1).

When the SCI uses its internal clock, the clock can be output from the SCKn pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is configured so that the rising edge of the clock is in the middle of the transmit data, as shown in Figure 27.8.

When clock output is enabled, the clock is output after setting the SCR.TE or SCR.RE bit to 1.

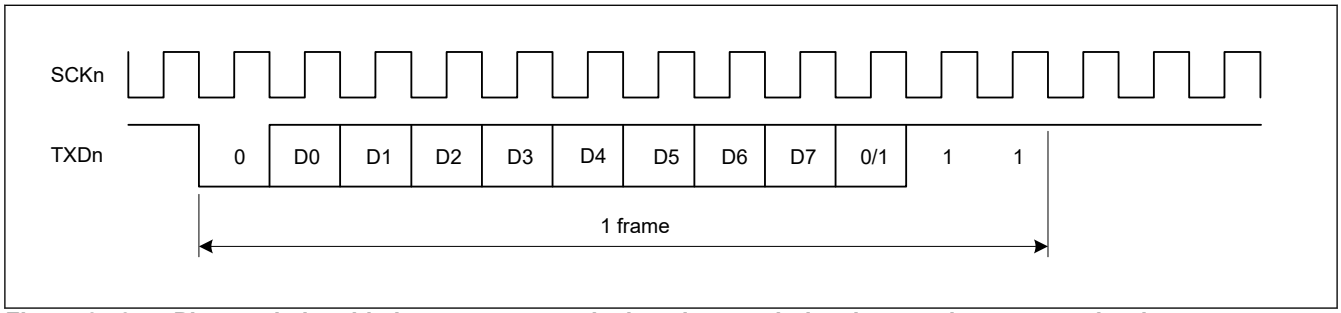



**Figure 27.8 Phase relationship between output clock and transmit data in asynchronous mode when SMR.CHR = 0, PE = 1, MP = 0, and STOP = 1**

### 27.3.4 Double-Speed Operation and Frequency of 6 Times the Bit Rate

When the SEMR.ABCS bit is set to 1 and eight pulses of the base clock for a 1-bit period is selected, the SCI operates on the bit rate twice that of when ABCS is set to 0. When the SEMR.BGDM bit is set to 1, the cycle of the base clock is half and the bit rate is double that of when BGDM is set to 0. When the SCR.CKE[1] bit is set to 0 and the on-chip baud rate generator is selected, setting the ABCS and BGDM bits to 1 allows the SCI to operate at a bit rate four times that when the ABCS and BGDM bits are set to 0.

When the SEMR.ABCSE bit is set to 1, the number of base clock pulses is 6 during a period of 1 bit, and the SCI operates at a bit rate 16/3 times that when SEMR.ABCS = 0, SEMR.BGDM = 0, and SEMR.ABCSE = 0.

As shown by Formula (1) in section 27.3.2. Receive Data Sampling Timing and Reception Margin in Asynchronous Mode, the reception margin decreases when the SEMR.ABCS or SEMR.ABCSE bit is set to 1. Therefore, if the target bit rate can be obtained with ABCS or ABCSE set to 0, it is recommended that you use the SCI with ABCS and ABCSE set to 0.

### 27.3.5 CTS and RTS Functions

The CTS function uses input on the CTSn_RTSn pin in transmission control. Setting the SPMR.CTSE bit to 1 enables the CTS function. For the functions of CTS and RTS, you can select the alternate setting that uses either function with one terminal and the dedicated setting that uses each function independently with two terminals. This setting is done with the SPMR.CTSPEN bit.

When the CTS function is enabled, placing a low level on the CTSn_RTSn pin causes transmission to start.

Driving the CTSn_RTSn pin high while transmission is in progress does not affect transmission of the current frame.

In the RTS function, which uses output on the CTSn_RTSn pin, a low level is output when reception becomes possible. Conditions for output of the low and high levels are shown in this section.

[Conditions for low level output]

Satisfaction of all conditions are listed in this section.

**Non-FIFO selected**

- The value of the SCR.RE bit is 1
- Reception is not in progress
- There is no received data yet to be read
- The ORER, FER, and PER flags in the SSR register are all 0

**FIFO selected**

- The value of the SCR.RE bit is 1
- The amount of receive data written in FRDRHL is equal to or less than the setting value of FCRH.RSTRG[3:0]
- The ORER flag in the SSR_FIFO register (ORER in FRDRH) is 0

[Condition for high level output]

- The conditions for low-level output are not satisfied

### 27.3.6 Address Match (Receive Data Match Detection) Function

The address match function can be used only in asynchronous mode.

If the DCCR.DCME bit is set to 1, when one frame of data is received, the SCI compares that received data with the data set in CDR.CMPD. If the SCI detects a match to the comparison data (CDR.CMPD[*1]) with the received data, the SCI can issue the SCIn_RXI interrupt request.

If the SMR.MP bit is set to 0, comparison occurs only for valid data in receive format. In multi-processor mode (SMR.MP bit = 1), if the DCCR.IDSEL bit is set to 1, receive data where the MPB bit is 1 is subject to comparison for address match and receive data where the MPB bit is 0 is always treated as a non-match.

If the DCCR.IDSEL bit is set to 0, SCI performs address match detection regardless of the MPB bit value of the received data.

Until SCI detects a match to the comparison data (CDR.CMPD[*1]) with receive data, received data is skipped (discarded), and the SCI cannot detect a parity error or framing error.

When SCI detects a match, the DCCR.DCME bit is automatically cleared, and the DCCR.DCMF flag is set to 1. If the DCCR.IDSEL bit is set to 1, the SCR.MPIE bit is automatically cleared. If DCCR.IDSEL is set to 0, the value of the SCR.MPIE bit is retained. If the SCR.RIE bit is set to 1, the SCI issues an SCIn_RXI interrupt request.

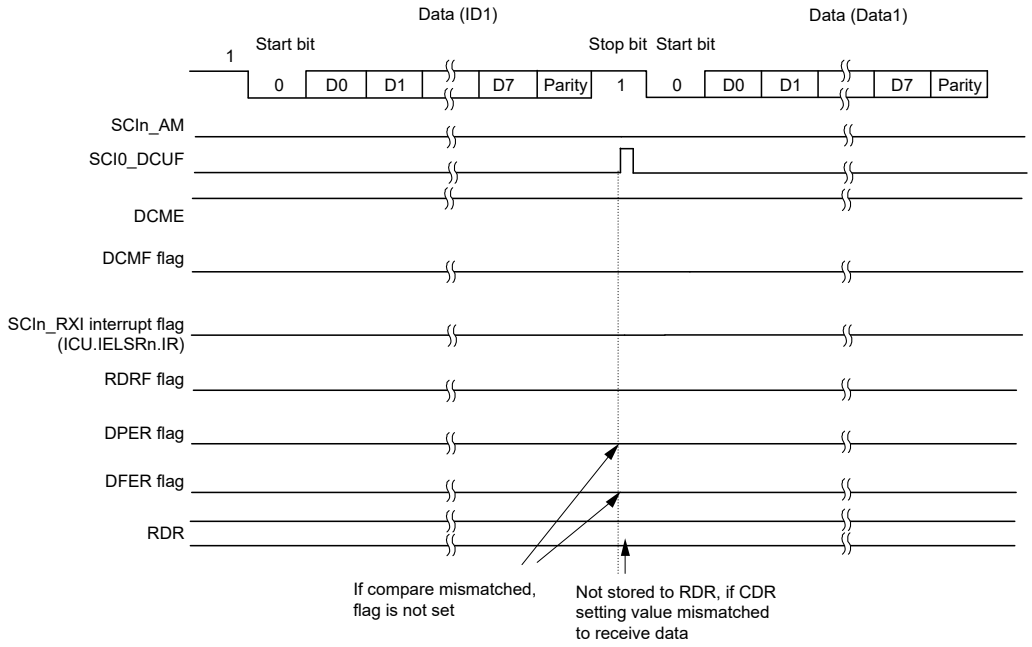
If the SCI detects a framing error in the receive data for which a match is detected, the DCCR.DFER flag is set to 1, and if the SCI detects a parity error in that frame, the DCCR.DPER flag is set to 1. The compared receive data is not stored in the RDR register, and SSR.RDRF remains 0. When FCR.FM = 1, the RDR register indicates the FRDRHL register, and the SSR.RDRF flag indicates the SSR_FIFO.RDF flag.

After the SCI detects a match, and DCCR.DCME is automatically cleared, the SCI receives the next data continuously based on the current register setting.

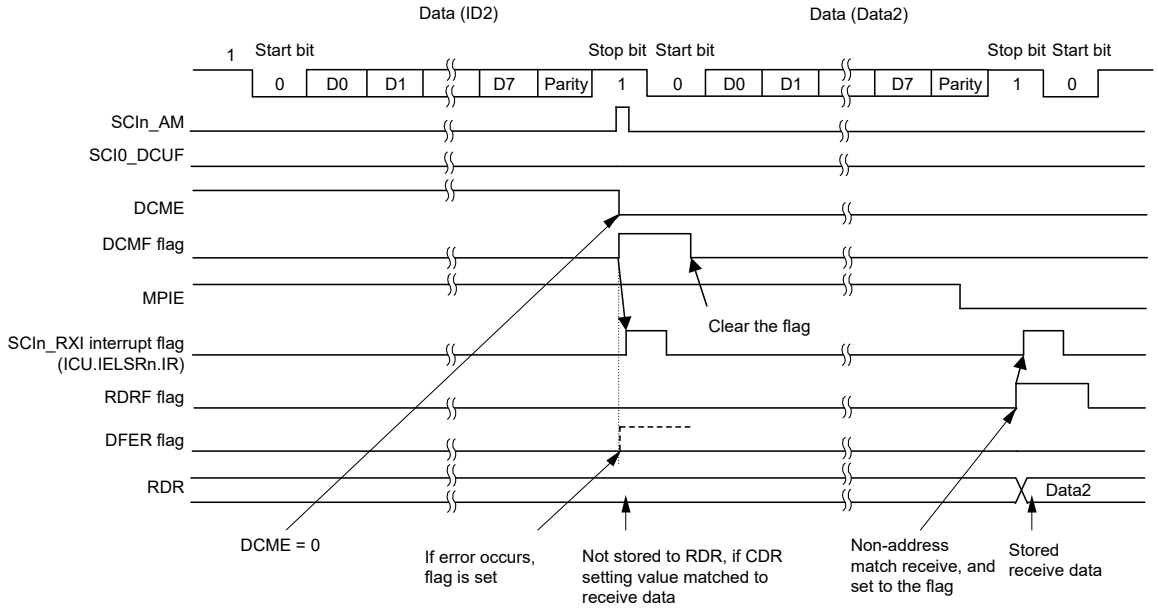
When the DCCR.DFER or DCCR.DPER flag is set, the address match is not performed. Before enabling the address match function, set the DCCR.DFER and DCCR.DPER flags to 0.

Examples of the address match function are shown in Figure 27.9 and Figure 27.10.

Note 1. This comparative target can select one length of 3 types: CMPD[6:0] with 7-bit length, CMPD[7:0] with 8-bit length, and CMPD[8:0] with 9-bit length.

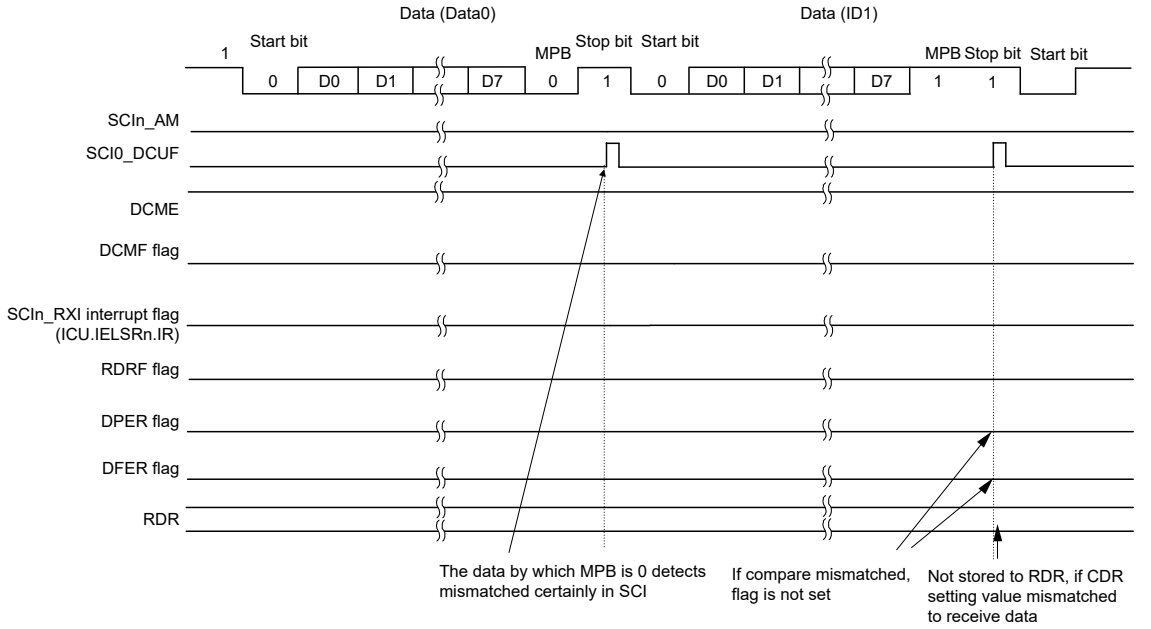



(a) Example of compare mismatched between receive data and CDR (8-bit length/parity/non multi-processor mode)

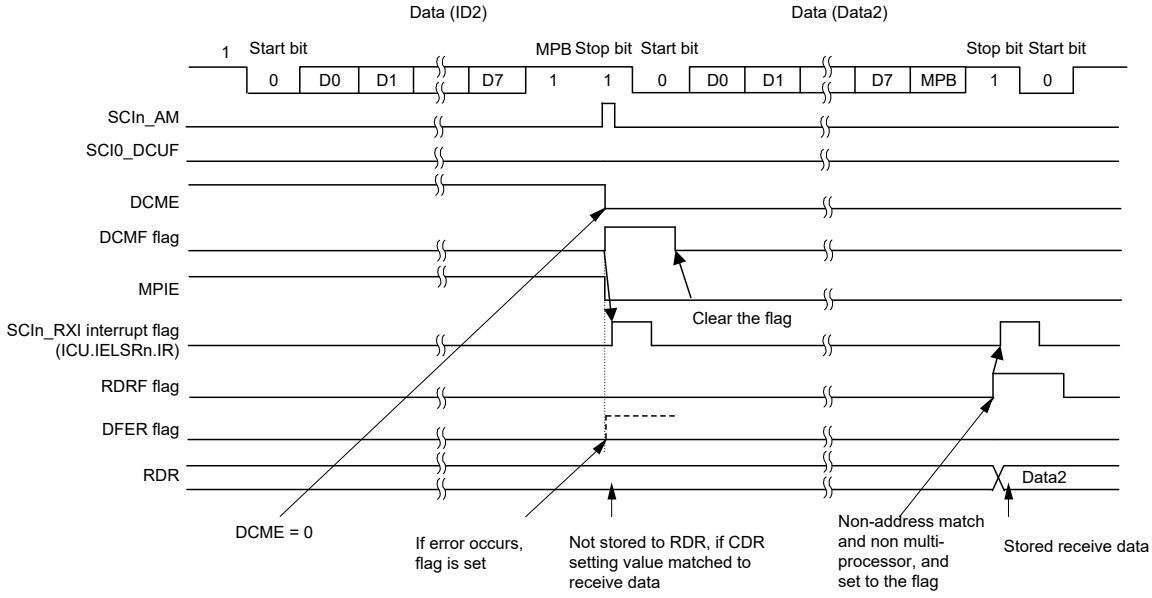



(b) Example of compare matched between receive data and CDR (8-bit length/parity/non multi-processor mode)

**Figure 27.9 Example of address match (1) normal mode**




(a) Example of compare mismatched between receive data and CDR (8-bit length/IDSEL = 1/multi-processor mode)




(b) Example of compare matched between receive data and CDR (8-bit length/IDSEL = 1/multi-processor mode)

**Figure 27.10 Example of address match (2) multi-processor mode**

## 27.3.7 SCI Initialization in Asynchronous Mode

Before transmitting and receiving data, start by writing the initial value 0x00 to the SCR register, then continue through the SCI initialization procedure (select non-FIFO or FIFO) shown in Table 27.28 and Table 27.29. Whenever the operating mode or transfer format is to be changed, the SCR register must be initialized before the change is made.

When the external clock is used in asynchronous mode, ensure that the clock signal is supplied during initialization.

Note: Setting the SCR.RE bit to 0 initializes neither the ORER, FER, RDRF, RDF, PER, and DR flags in SSR/SSR_FIFO nor RDR and RDRHL. When the TE bit is set to 0, the TEND flag for the selected FIFO buffer is not initialized.

Note: In non-FIFO mode, switching the value of the SCR.TE bit from 1 to 0 or 0 to 1 while the SCR.TIE bit is 1 leads to the generation of an SCIn_TXI interrupt request.

**Table 27.28 Example flow of SCI initialization in asynchronous mode with non-FIFO selected**

| No. | Step Name | Description |
|---|---|---|
| 1 | Start initialization | |
| 2 | Set the SCR.TIE, RIE, TE, RE, and TEIE bits to 0 | |
| 3 | Set the FCR.FM bit to 0 | Set the FCR.FM bit to 0. |
| 4 | Set the SCR.CKE[1:0] bits | Set the clock selection in SCR.<br>When the clock output is selected in asynchronous mode, the clock is output immediately after SCR settings are made. |
| 5 | Set the SIMR1.IICM bit to 0.<br>Set the SPMR.CKPH and CKPOL bits to 0. | Set the SIMR1.IICM bit to 0.<br>Set the SPMR.CKPH and CKPOL bits to 0.<br>Step 5 can be skipped if the values have not been changed from the initial values. |
| 6 | Set the data transmission/ reception format in SMR, SCMR, and SEMR | Set data transmission/reception format in SMR, SCMR, and SEMR. |
| 7 | Set a value in SPTR and ACTR | Set the communication terminals status in SPTR and adjustable sampling values in ACTR. |
| 8 | Set a value in BRR | Write a value corresponding to the bit rate to BRR.<br>This step is not necessary if an external clock is used. |
| 9 | Set a value in MDDR | Write the value obtained by correcting a bit rate error in MDDR. This step is not necessary if the BRME bit in SEMR is set to 0 or an external clock is used. |
| 10 | Set the I/O port functions | Make I/O port settings to enable input and output functions as required for TXDn, RXDn, and SCKn pins. |
| 11 | Set the SCR.TE or RE bit to 1, and set the SCR.TIE and RIE bits | Set the SCR.TE or RE bit to 1. Also set the SCR.TIE and RIE bits.<br>Setting the TE and RE bits allows TXDn and RXDn to be used. |
| 12 | Initialization completion | |

**Table 27.29 Example flow of SCI initialization in asynchronous mode with FIFO selected (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Start initialization | |
| 2 | Set the SCR.TIE, RIE, TE, RE, and TEIE bits to 0 | |
| 3 | Set the FCR.FM, TFRST, and RFRST bits to 1.<br>Set the FCR.TTRG[3:0], RTRG[3:0], and RSTRG[3:0] bits. | Set the FCR.FM, TFRST, and RFRST bits to 1 (FIFO mode enabled, transmit/receive FIFOs empty).<br>Set the FCR.TTRG[3:0], RTRG[3:0], and RSTRG[3:0] bits. |
| 4 | Set the SCR.CKE[1:0] bits | Set the clock selection in SCR.<br>When the clock output is selected in asynchronous mode, the clock is output immediately after SCR settings are made. |
| 5 | Set the SIMR1.IICM bit to 0.<br>Set the SPMR.CKPH and CKPOL bits to 0. | Set the SIMR1.IICM bit to 0.<br>Set the SPMR.CKPH and CKPOL bits to 0.<br>Step 5 can be skipped if the values have not been changed from the initial values. |
| 6 | Set the data transmission/ reception format in SMR, SCMR, and SEMR | Set data transmission/reception format in SMR, SCMR, and SEMR. |
| 7 | Set a value in SPTR and ACTR | Set the communication terminals status in SPTR and adjustable sampling values in ACTR. |

**Table 27.29 Example flow of SCI initialization in asynchronous mode with FIFO selected (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 8 | Set a value in BRR | Write a value corresponding to the bit rate to BRR.<br>This step is not necessary if an external clock is used. |
| 9 | Set a value in MDDR | Write the value obtained by correcting a bit rate error in MDDR. This step is not necessary if the BRME bit in SEMR is set to 0 or an external clock is used. |
| 10 | Set the FCR.TFRST and RFRST bits to 0 | Set the FCR.TFRST and RFRST bits to 0. |
| 11 | Set the I/O port functions | Make I/O port settings to enable input and output functions as required for TXDn, RXDn, and SCKn pins. |
| 12 | Set the SCR.TE or RE bit to 1, and set the SCR.TIE and RIE bits | Set the SCR.TE or RE bit to 1. Also set the SCR.TIE and RIE bits.<br>Setting the TE and RE bits allows TXDn and RXDn to be used. |
| 13 | Initialization completion | |

## 27.3.8 Serial Data Transmission in Asynchronous Mode

(1) Non-FIFO selected

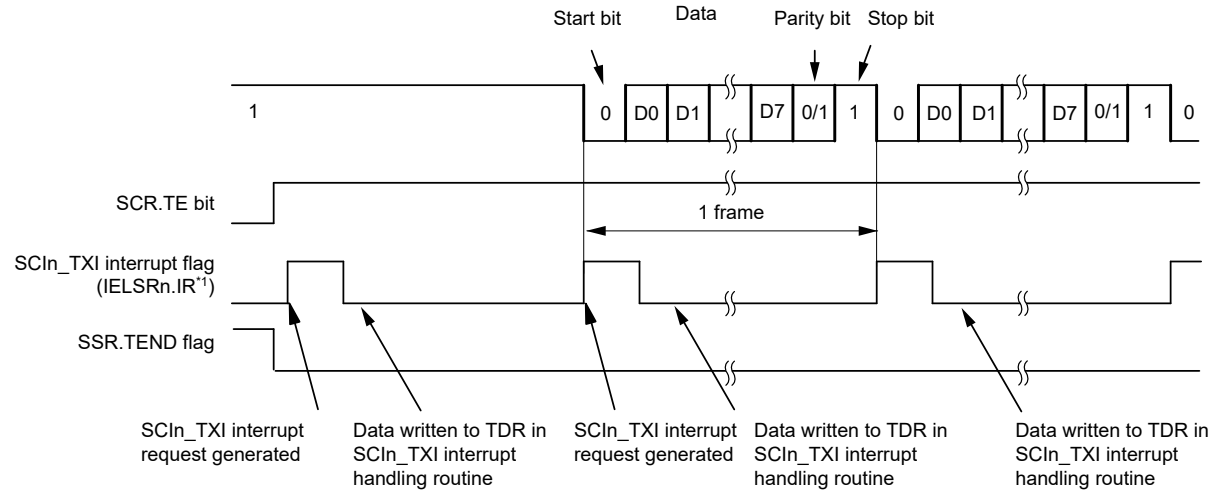
Figure 27.11, Figure 27.12, and Figure 27.13 show examples of serial transmission in asynchronous mode.

In serial transmission, the SCI operates as described in this section. When the SCR.TE bit is set to 1, the high level is output to TXDn for one frame. However, when SEMR.PADIS is set to "1", this preamble will not be output. An example of operation when preamble is not output is shown in Figure 27.14.

1. The SCI transfers data from the TDR[*1] register to the TSR register when data is written to TDR[*1] in the SCIn_TXI interrupt handling routine.
   The SCIn_TXI interrupt request at the beginning of transmission is generated when the SCR.TE and SCR.TIE bits are set to 1 simultaneously by a single instruction.
2. Transmission starts after the SPMR.CTSE bit is set to 0 (CTS function is disabled) or a low level on the CTSn_RTSn pin causes data transfer from the TDR[*1] register to the TSR register. If the SCR.TIE bit is 1, an SCIn_TXI interrupt request is generated. Continuous transmission is possible by writing the next transmit data to the TDR[*1] register in the SCIn_TXI interrupt handling routine before transmission of the current transmit data is complete. When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 (SCIn_TXI interrupt requests are disabled) and the SCR.TEIE bit to 1 (an SCIn_TEI interrupt request is enabled) after the last of the data to be transmitted is written to the TDR[*1] register from the handling routine for SCIn_TXI requests.
3. Data is sent from the TXDn pin in the following order:
   - Start bit
   - Transmit data
   - Parity bit or multi-processor bit (can be omitted depending on the format)
   - Stop bit
4. The SCI checks for update of the TDR register on output of the stop bit.
5. When the TDR register is updated, setting the SPMR.CTSE bit to 0 (CTS function is disabled) or a low level input on the CTSn_RTSn pin causes transfer of the next transmit data from the TDR[*1] register to the TSR register and transmission of the stop bit, after which serial transmission of the next frame starts.
6. If the TDR register is not updated, the SSR.TEND flag is set to 1, the stop bit is sent, and the mark state is entered, in which 1 is output. If the SCR.TEIE bit is 1, the SSR.TEND flag is set to 1 and an SCIn_TEI interrupt request is generated.

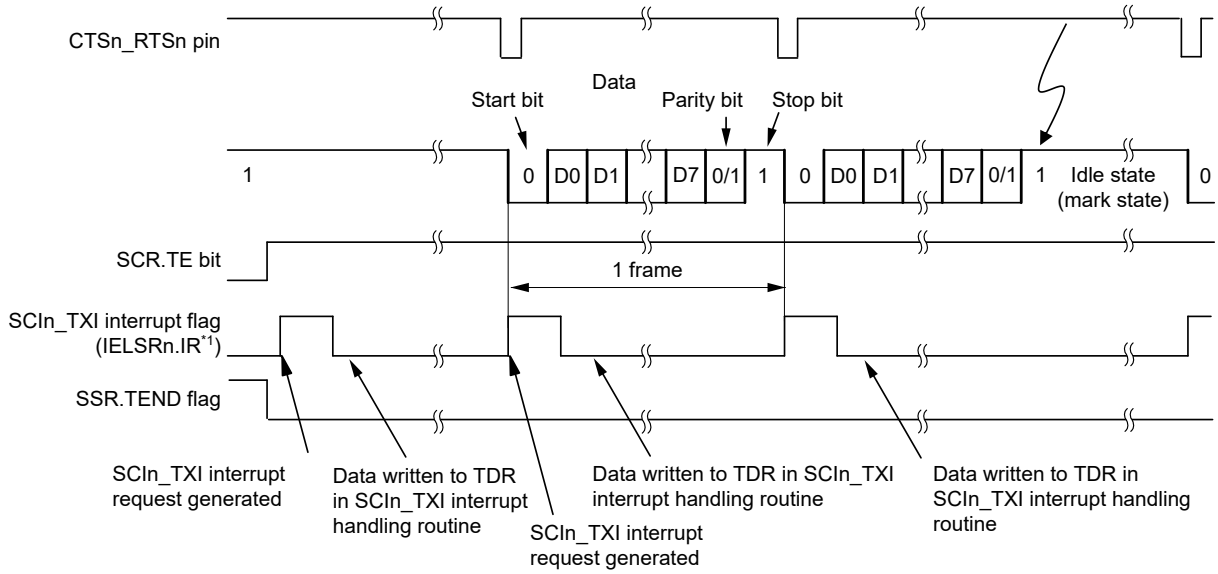
Note 1. The TDRHL register when 9-bit data length is selected.

Figure 27.11, Figure 27.12, and Figure 27.13 show examples of serial transmission in asynchronous mode.



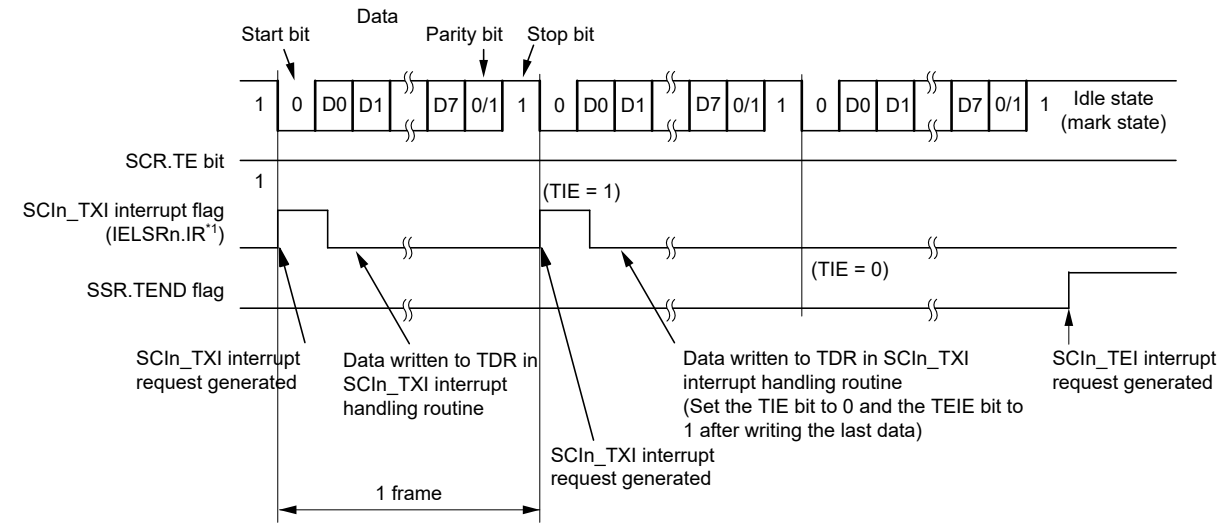

Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.11 Example operation for serial transmission in asynchronous mode (1) with 8-bit data, parity bit, 1 stop bit, CTS function not used, and at the beginning of transmission**




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.12 Example operation for serial transmission in asynchronous mode (2) with 8-bit data, parity bit, one stop bit, CTS function used, and at the beginning of transmission**




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.13 Example operation for serial transmission in asynchronous mode (3) with 8-bit data, parity bit, one stop bit, CTS function not used, and from the middle of transmission until transmission completion**

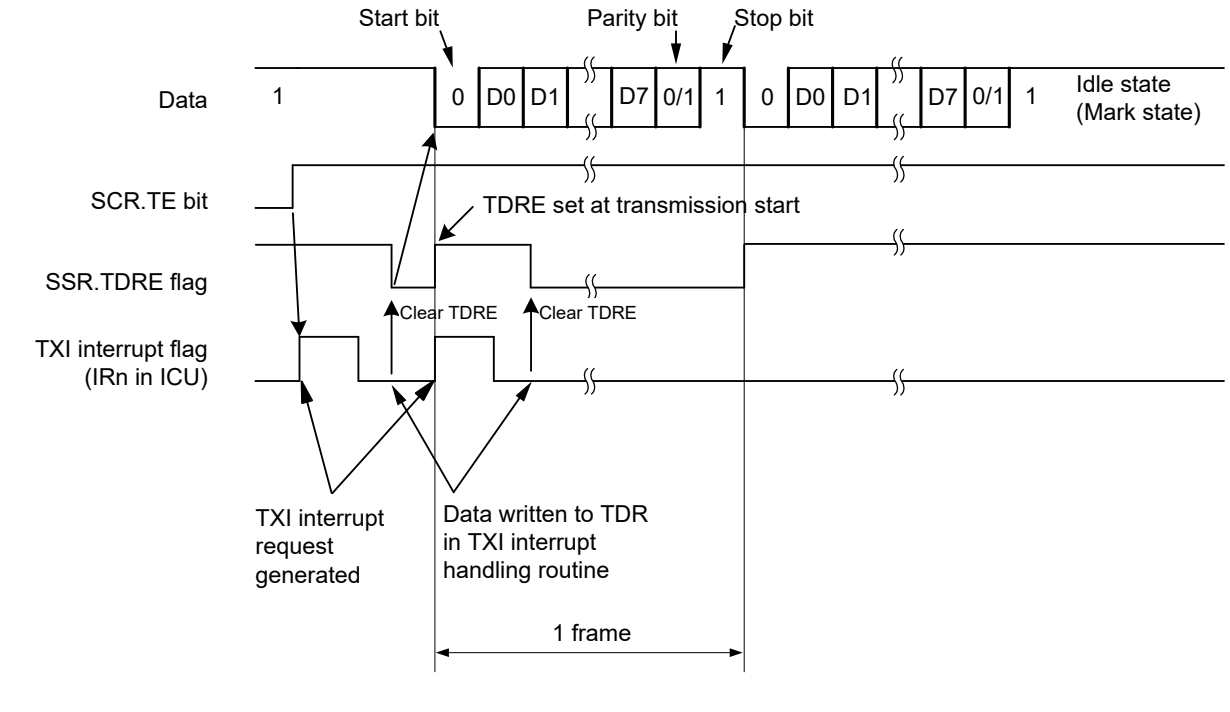
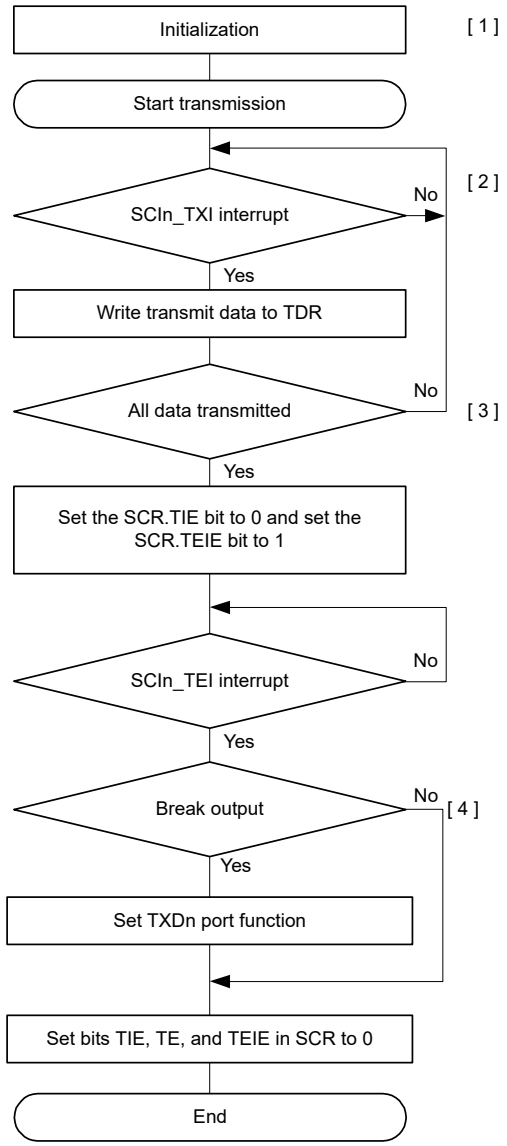



**Figure 27.14 Example of Operation for Serial Transmission in Asynchronous Mode (4)(with 8-Bit Data, Parity, 1 Stop Bit, CTS Function Not Used, from the Middle of Transmission until Transmission Completion, stop preamble)**




[1] SCI Initialization:
Set data transmission.
After the SCR.TE bit is set to 1, 1 is output for a frame, and transmission is enabled.

[2] Transmit data write to TDR by an SCIn_TXI interrupt request. When transmit data is transferred from TDR to TSR, a transmit data empty interrupt (SCIn_TXI) request is generated. Write transmit data to TDR once in the SCIn_TXI interrupt processing routine.

[3] Serial transmission continuation procedure:
To continue serial transmission write transmit data to TDR once using an SCIn_TXI interrupt request. Transmit data can also be written to TDR by activating the DMAC or DTC.
When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 and the SCR.TEIE bit to 1 after the last of the data to be transmitted is written to TDR.

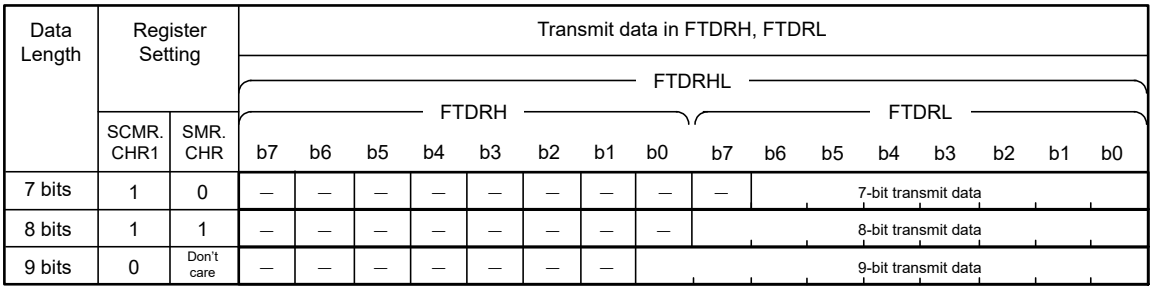
[4] Break output at the end of serial transmission:
To output a break in serial transmission, after setting the output state (LOW level output) of TXDn pin by SPTR.SPB2IO and SPTR.SPB2DT bits, set the SCR.TE bit to 0.

**Figure 27.15 Example flow of serial transmission in asynchronous mode with non-FIFO selected**

(2) FIFO selected

Figure 27.16 shows an example of a data format that is written to FTDRH and FTDRL register in asynchronous mode.

Data corresponding to the data length is set to FTDRH and FTDRL. Write 0 for unused bits. Write in order from FTDRH to FTDRL.




Note: —: Invalid. The write value should be 0.

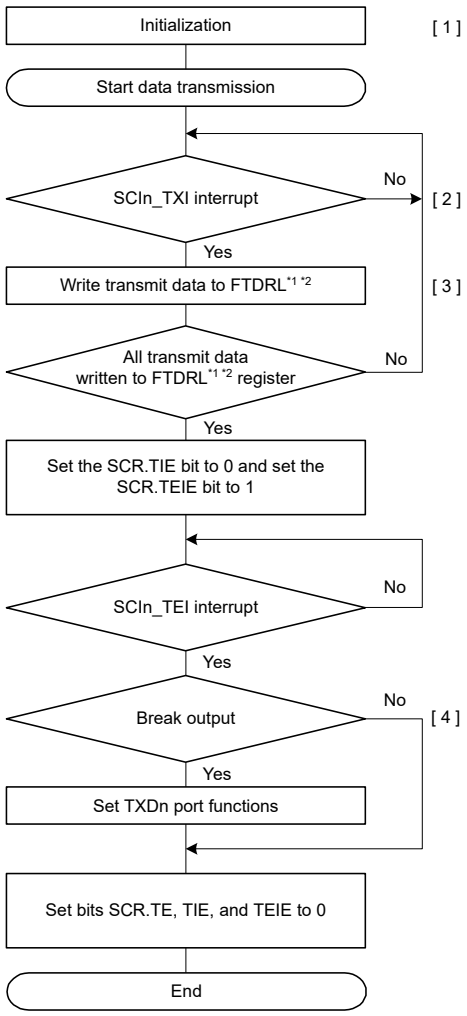
**Figure 27.16 Data format written to FTDRH and FTDRL with FIFO selected**

In serial transmission, the SCI operates as described in this section. When the TE bit is set to 1, the high level is output to TXDn for one frame (preamble).

1. The SCI transfers data from the FTDRL*1 register to the TSR register when data is written to FTDRL*1 in the SCIn_TXI interrupt handling routine. The amount of data that can be written to FTDRL is 16 minus FDR.T[4:0] bytes. The SCIn_TXI interrupt request at the beginning of transmission is generated when the SCR.TE and SCR.TIE bits are set to 1 simultaneously by a single instruction.
2. Transmission starts after the SPMR.CTSE bit is set to 0 (CTS function is disabled) or a low level on the CTSn_RTSn pin causes data transfer from the FTDRL*1 register to the TSR register. When the amount of transmit data written in FTDRL is equal to or less than the specified transmit triggering number, SSR_FIFO.TDFE is set to 1. If the SCR.TIE bit is 1, an SCIn_TXI interrupt request is generated. Continuous transmission is possible by writing the next transmit data to FTDRL*1 in the SCIn_TXI interrupt handling routine before transmission of the current transmit data is complete. When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 (SCIn_TXI interrupt requests are disabled) and the SCR.TEIE bit to 1 (an SCIn_TEI interrupt request is enabled) after the last of the data to be transmitted is written to the FTDRL*1*2 register from the handling routine for SCIn_TXI requests.
3. Data is sent from the TXDn pin in the following order:
   - Start bit
   - Transmit data
   - Parity bit or multi-processor bit (can be omitted depending on the format)
   - Stop bit
4. On output of the stop bit, the SCI checks whether non-transmitted data remains in the FTDRL*3 register.
5. When data is set to FTDRL*3, setting the SPMR.CTSE bit to 0 (CTS function is disabled) or a low level input on the CTSn_RTSn pin causes transfer of the next transmit data from FTDRL*1 to TSR and transmission of the stop bit, after which serial transmission of the next frame starts.
6. If data is not set in FTDRL*3, the TEND flag in SSR_FIFO is set to 1, the stop bit is sent, and the mark state is entered in which 1 is output. If the SCR.TEIE bit is 1, the SSR_FIFO.TEND flag is set to 1 and an SCIn_TEI interrupt request is generated.

Note 1. Write data not to FTDRL but to the FTDRH and FTDRL registers.
Note 2. Write data in order from FTDRH to FTDRL when 9-bit data length is selected.
Note 3. The SCI only checks for update to the FTDRL register and not the FTDRH register when 9-bit data length is selected.

Figure 27.17 shows an example flow of serial transmission in asynchronous mode with FIFO selected.




[ 1 ] SCI initialization:
Set data transmission.
After the SCR.TE bit is set to 1, 1 is output for a frame, and transmission is enabled.

[ 2 ] Transmit data write to FTDRL*1 by an SCIn_TXI interrupt request:
When transmit data is transferred from FTDRL to TSR, when the quantity of transmit data written in FTDRL is equal to or less than the specified transmit triggering number, a transmit data FIFO empty interrupt (SCIn_TXI) request is generated. Write transmit data to FTDRL*1 *2 once in the SCIn_TXI interrupt handling routine.

[ 3 ] Serial transmission continuation procedure:
To continue serial transmission, write all transmit data to FTDRL using an SCIn_TXI interrupt request and clear the SSR_FIFO.TDFE flag to 0. The number of transmission data it is possible to write in is 16 - (the number of stored transmit FIFO data). Transmit data can also be written to FTDRL by activating the DTC or DMAC. When writing the data to FTDRL by DTC or DMAC, the TDFE flag is cleared automatically, so do not write to the TDFE flag.
When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 and the SCR.TEIE bit to 1 after the last of the data to be transmitted is written to FTDRL.

[ 4 ] Break output at the end of serial transmission:
To output a break in serial transmission, after setting the output state (LOW level output) of TXD pin by SPTR.SPB2IO and SPTR.SPB2DT bits, set the SCR.TE bit to 0.

Note 1. When data length is 9 bits, this refers to FTDRH and FTDRL registers.
Note 2. When data length is 9 bits, write in order from FTDRH to FTDRL.

**Figure 27.17 Example flow of serial transmission in asynchronous mode with FIFO selected**

## 27.3.9 Serial Data Reception in Asynchronous Mode

(1) Non-FIFO selected

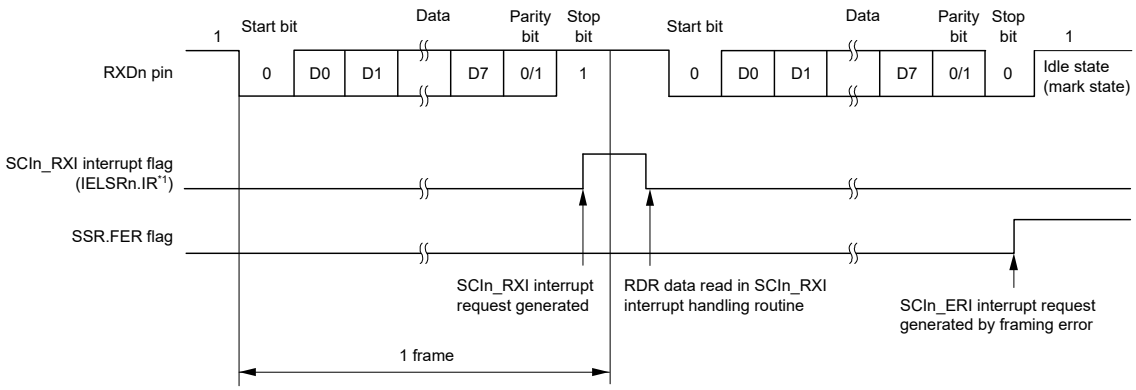
Figure 27.18 and Figure 27.19 show an example of the operation for serial data reception in asynchronous mode.

In serial data reception, the SCI operates as follows:

1. When the value of the SCR.RE bit becomes 1, the output signal on the CTSn_RTSn pin goes low.
2. The SCI monitors the communications line and when it detects a start bit, the SCI performs internal synchronization, stores receive data in RSR.
3. If the multi-processor communication function is enabled (SMR.MP = 1), see section 27.4.2. Multi-Processor Serial Data Reception. If the address match function (data compare match function) is enabled (DCCR.DCME = 1), the SCI cannot detect a parity or framing error as receive data are skipped (discarded) until the SCI detects a match between the receive data and comparison data (CDR.CMPD*1).
4. If the SCI detects an address match, the DCCR.DCME bit is automatically cleared, the DCCR.DCMF flag becomes 1, and an SCIn_AM interrupt*2 request is generated. To enable the generation of an SCIn_RXI interrupt request, set the SCR.RIE bit to 1. The compared receive data are not stored in the RDR register*3. The SSR.RDRF flag remains 0.

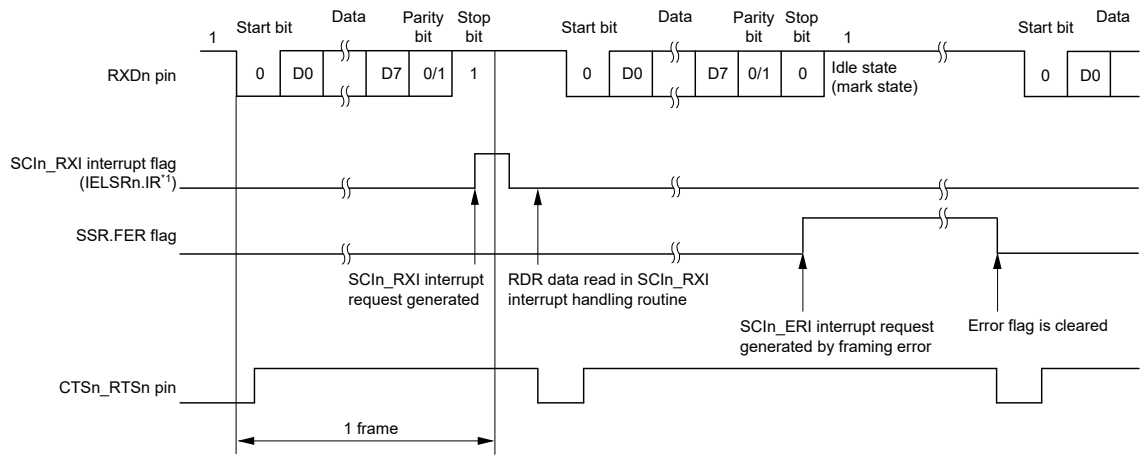
5. If the SCI detects a framing error in the receive data for which an address match is detected, the DCCR.DFER flag is set to 1, and if the SCI detects a parity error in that frame, the DCCR.DPER flag becomes 1. To enable the generation of an SCIn_ERI interrupt request, set the SCR.RIE bit to 1.
6. If a framing or a parity error is detected (the DCCR.DFER flag or DCCR.DPER flag is 1) in the SCIn_AM interrupt handling routine, set the DCCR.DFER and DCCR.DPER flags to 0 and set the DCCR.DCME bit to 1 to enable the address match function again. If neither a framing nor a parity error has been detected (the DCCR.DFER and DCCR.DPER flags are both 0), set the DCCR.DCMF flag to 0. See Figure 27.9.
7. If an overrun error occurs, the SSR.ORER flag is set to 1. If the SCR.RIE bit is 1, an SCIn_ERI interrupt request is generated. Receive data is not transferred to the RDR*3 register.
8. If a parity error is detected, the SSR.PER flag is set to 1 and receive data is transferred to the RDR*3 register. If the SCR.RIE bit is 1, an SCIn_ERI interrupt request is generated.
9. If a framing error is detected, the SSR.FER flag is set to 1 and receive data is transferred to the RDR*3 register. If the SCR.RIE bit is 1, an SCIn_ERI interrupt request is generated.
10. When reception finishes successfully, receive data is transferred to the RDR*3 register. If the SCR.RIE bit is 1, an SCIn_RXI interrupt request is generated. Continuous reception is enabled by reading the receive data transferred to the RDR register in the SCIn_RXI interrupt handling routine before reception of the next receive data is complete. Reading the received data that was transferred to the RDR register causes the CTSn_RTSn pin to output low.

Note 1. This scope of comparison is selectable as one of three lengths: CMPD[6:0] is for 7-bit length, CMPD[7:0] is for 8-bit length, and CMPD[8:0] is for 9-bit length.
Note 2. As no interrupt enable bit is assigned to the SCIn_AM interrupt, an interrupt request is generated by setting the DCCR.DCMF to 1.
Note 3. Only read data in the RDRHL register when 9-bit data length is selected.




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.18 Example of SCI operation for serial reception in asynchronous mode (1) when the RTS function is not used, and with 8-bit data, parity bit, and 1 stop bit**




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.19 Example of SCI operation for serial reception in asynchronous mode (2) when RTS function is used, and with 8-bit data, parity bit, and 1 stop bit**

Table 27.30 lists the states of the flags in the SSR register and receive data handling when a receive error is detected.

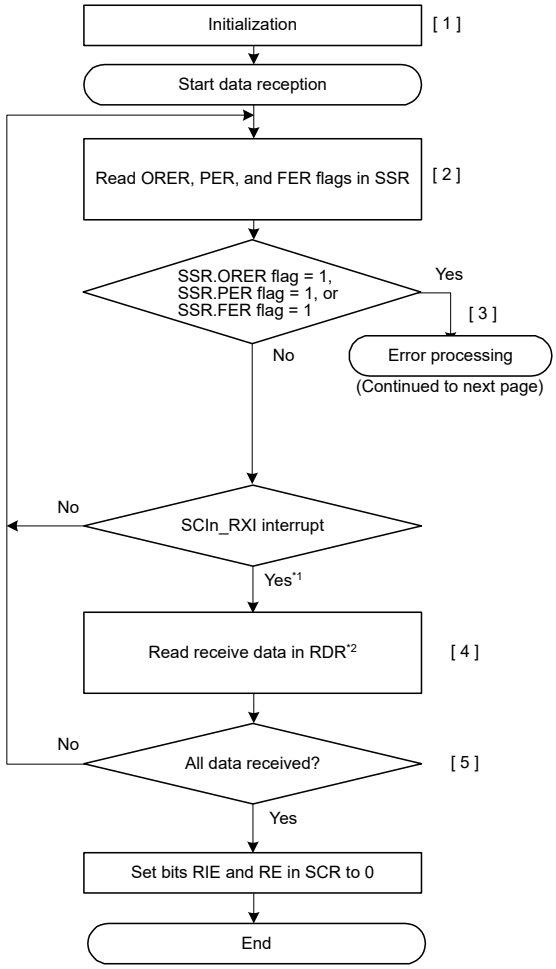
If a receive error is detected, an SCIn_ERI interrupt request is generated but an SCIn_RXI interrupt request is not generated. Data reception cannot be resumed while the receive error flag is 1. Accordingly, set the ORER, FER, and PER bits to 0 before resuming reception. In addition, be sure to read the RDR or RDRHL register during overrun error processing. When a reception is forced to terminate by setting the SCR.RE bit to 0 during operation, read the RDR or RDRHL register because received data that is not yet read might be left in the RDR or RDRHL.

Figure 27.20 and Figure 27.21 show example flows of serial data reception.

**Table 27.30 Flags in SSR Status Register and receive data handling**

| Flags in the SSR Status Register | | | Receive data | Receive error type |
|---|---|---|---|---|
| ORER | FER | PER | | |
| 1 | 0 | 0 | Lost | Overrun error |
| 0 | 1 | 0 | Transferred to RDR*1 | Framing error |
| 0 | 0 | 1 | Transferred to RDR*1 | Parity error |
| 1 | 1 | 0 | Lost | Overrun error + framing error |
| 1 | 0 | 1 | Lost | Overrun error + parity error |
| 0 | 1 | 1 | Transferred to RDR*1 | Framing error + parity error |
| 1 | 1 | 1 | Lost | Overrun error + framing error + parity error |

Note 1. Only read data in the RDRHL register when 9-bit data length is selected.




[ 1 ] SCI initialization:
Set data reception.

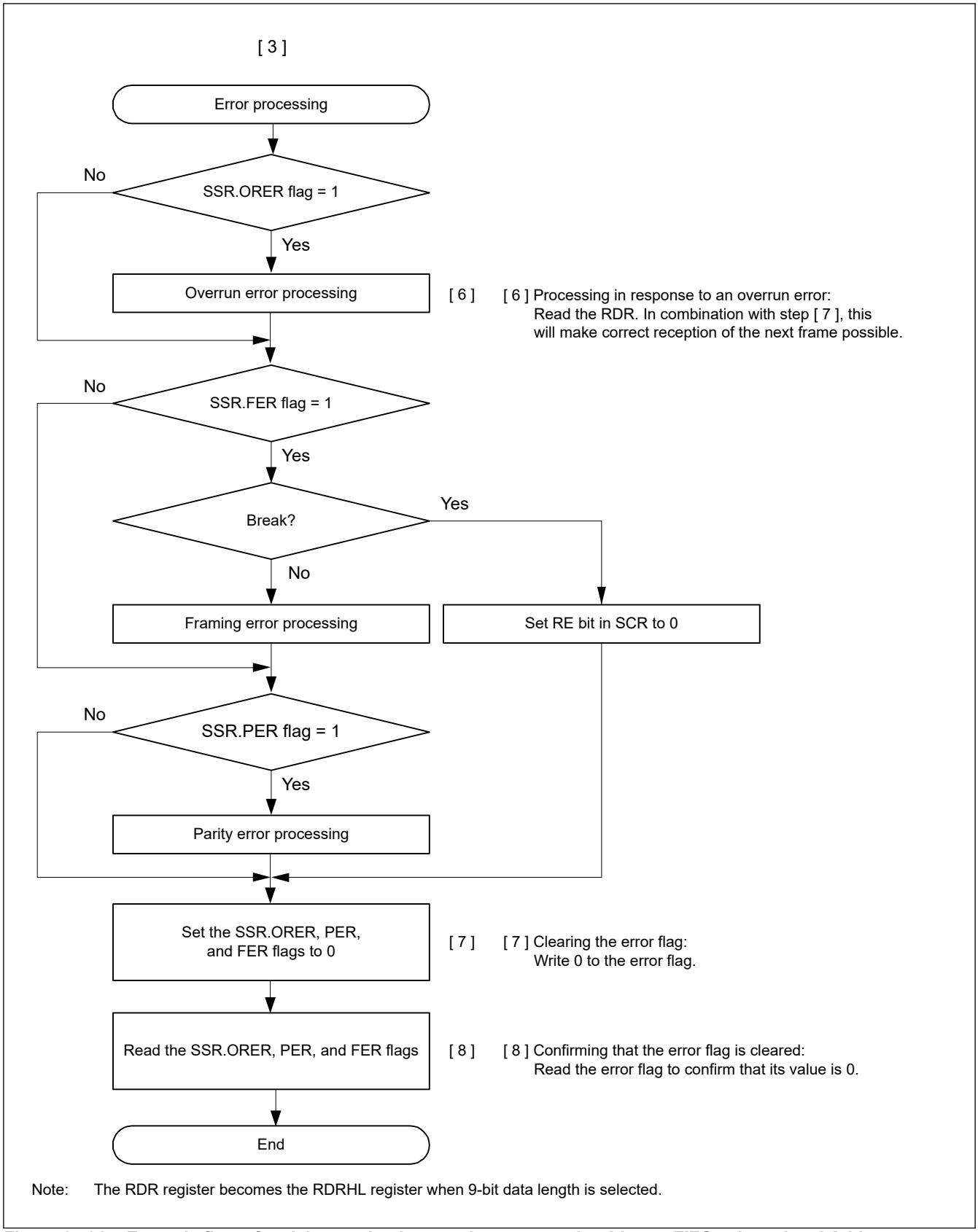
[ 2 ][ 3 ] Receive error processing and break detection:
If a receive error occurs, an SCIn_ERI interrupt is generated. The error type is identified by reading the ORER, PER, and FER flags in SSR. After performing the appropriate error processing, be sure to set the ORER, PER, and FER flags to 0. Reception cannot be resumed if any of these flags is set to 1. For a framing error, a break can be detected by reading the value of the input port associated with the RXDn pin.

[ 4 ] Read the receive data in RDR once inthe SCIn_RXI interrupt handling routine.

[ 5 ] To continue serial reception, before the stop bit of the current frame is received, read data from RDR in the SCIn_RXI interrupt processing routine. The RDR data can also be read by activating the DMAC or DTC.

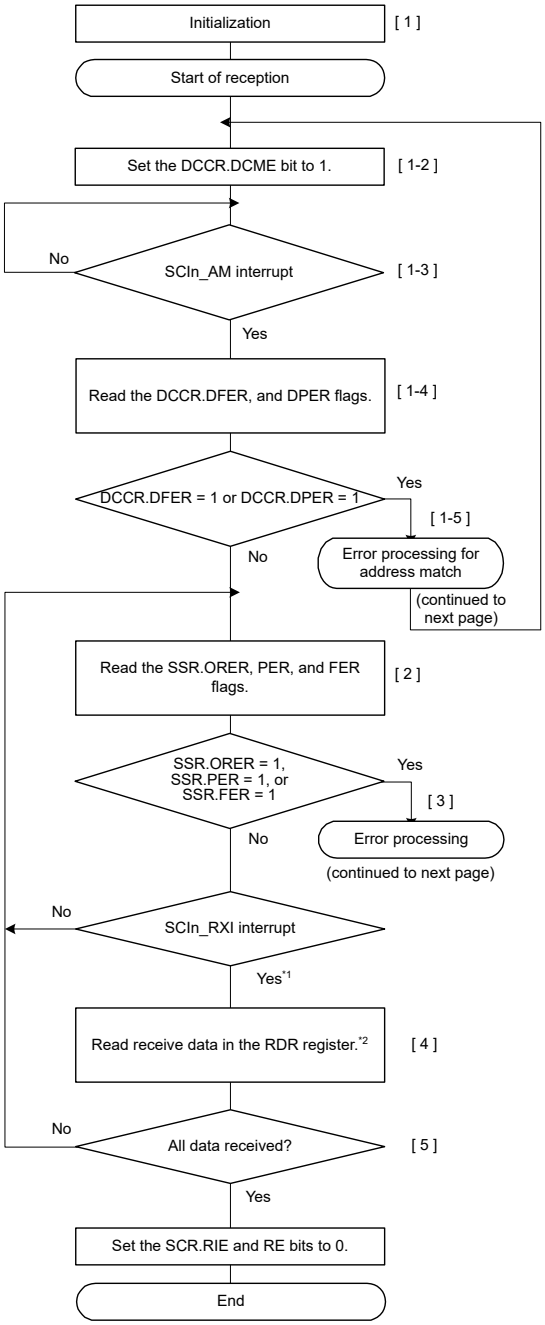
Note 1. Do not set RE to 0 before reading RDR.
Note 2. The RDR register becomes the RDRHL register when 9-bit data length is selected.

**Figure 27.20 Example flow of serial reception in asynchronous mode with non-FIFO selected and Address Matching Disabled (1)**




Note: The RDR register becomes the RDRHL register when 9-bit data length is selected.

**Figure 27.21 Example flow of serial reception in asynchronous mode with non-FIFO selected and Address Matching Disabled (2)**




[ 1 ] SCI initialization:
Set data reception.

[ 1-2 ] Address match cycle:
Set the DCCR.DCME bit to 1 to enable address matching.

[ 1-3 ] Comparing receive data and data for comparison (CDR.CMPD):
Compare receive data and data for comparison (CDR.CMPD).
The SCI cannot detect a parity or framing error as receive data are skipped (discarded) until the SCI detects a match between the receive data and data for comparison.
If the SCI detects an address match, the DCCR.DCME bit is cleared, the DCCR.DCMF flag becomes 1, and an SCIn_AM interrupt request is issued.

[ 1-4 ] [ 1-5 ] Reception error processing:
An error is identified by reading the DFER or DPER flag in DCCR in the SCIn_AM interrupt handling routine.
After appropriate handling of the error, set both the DFER and DPER flags to 0.
If either of these flags is set to 1, address matching cannot be resumed.

[ 2 ] [ 3 ] Reception error processing and break detection:
When a reception error occurs, the SCIn_ERI interrupt is generated.
The error is identified by reading the ORER, PER, and FER flags in the SSR.
After appropriate handling of the error, set all of the ORER, PER, and FER flags to 0. If any of these flags is set to 1, reception cannot be resumed.
If a framing error occurs, a break can be detected by reading the value of the input port bit corresponding to the RXDn pin.

[ 4 ] Read the receive data in the RDR register from within the SCIn_RXI interrupt handling routine.

[ 5 ] Procedure for continuing serial reception:
To continue serial reception, read the value of the RDR register in the SCIn_RXI interrupt handling routine before the stop bit is received.
The value of the RDR register can also be read by activating the DMAC or DTC.

Note 1. Be sure to set the RE bit to 0 before reading the RDR register.
Note 2. The RDRHL register is used instead of the RDR register when the 9-bit data length is selected.

**Figure 27.22 Example Flowchart of Serial Reception in Asynchronous Mode (FIFO not Selected and Address Matching Enabled) (1)**

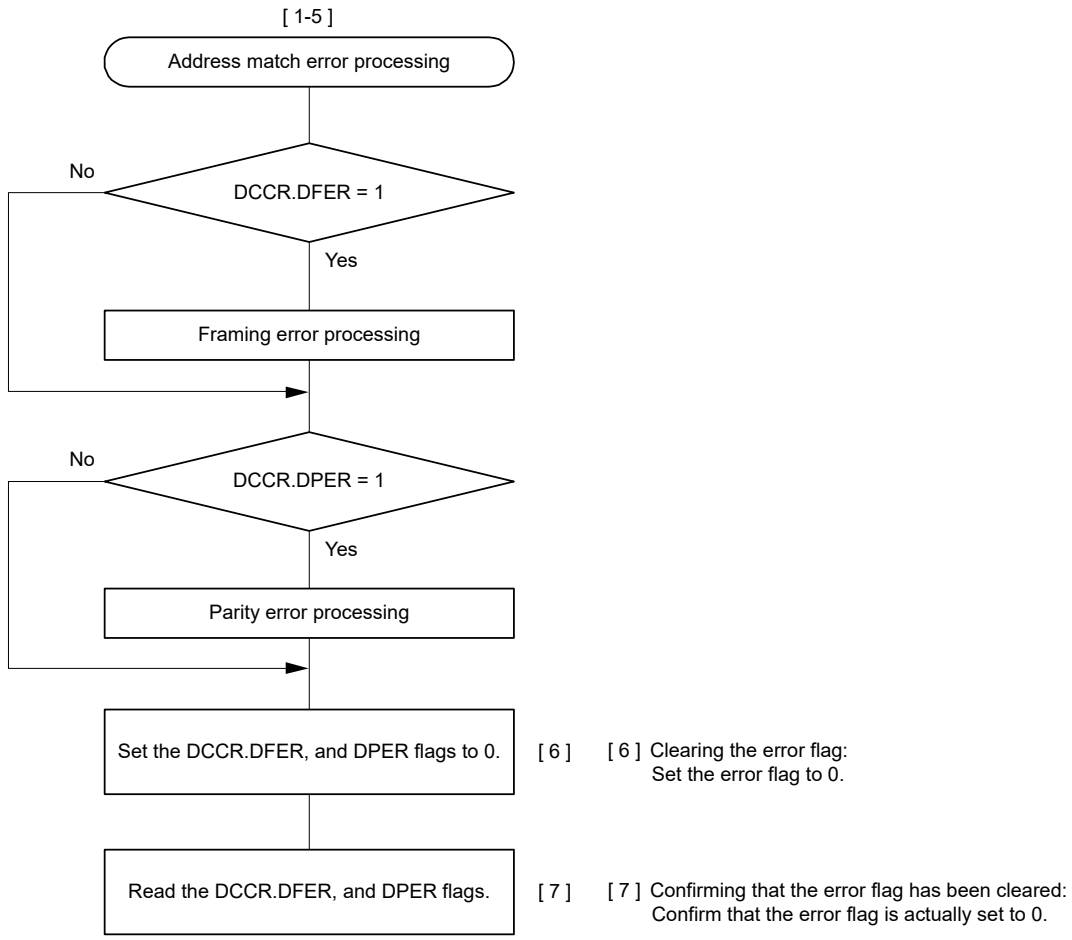
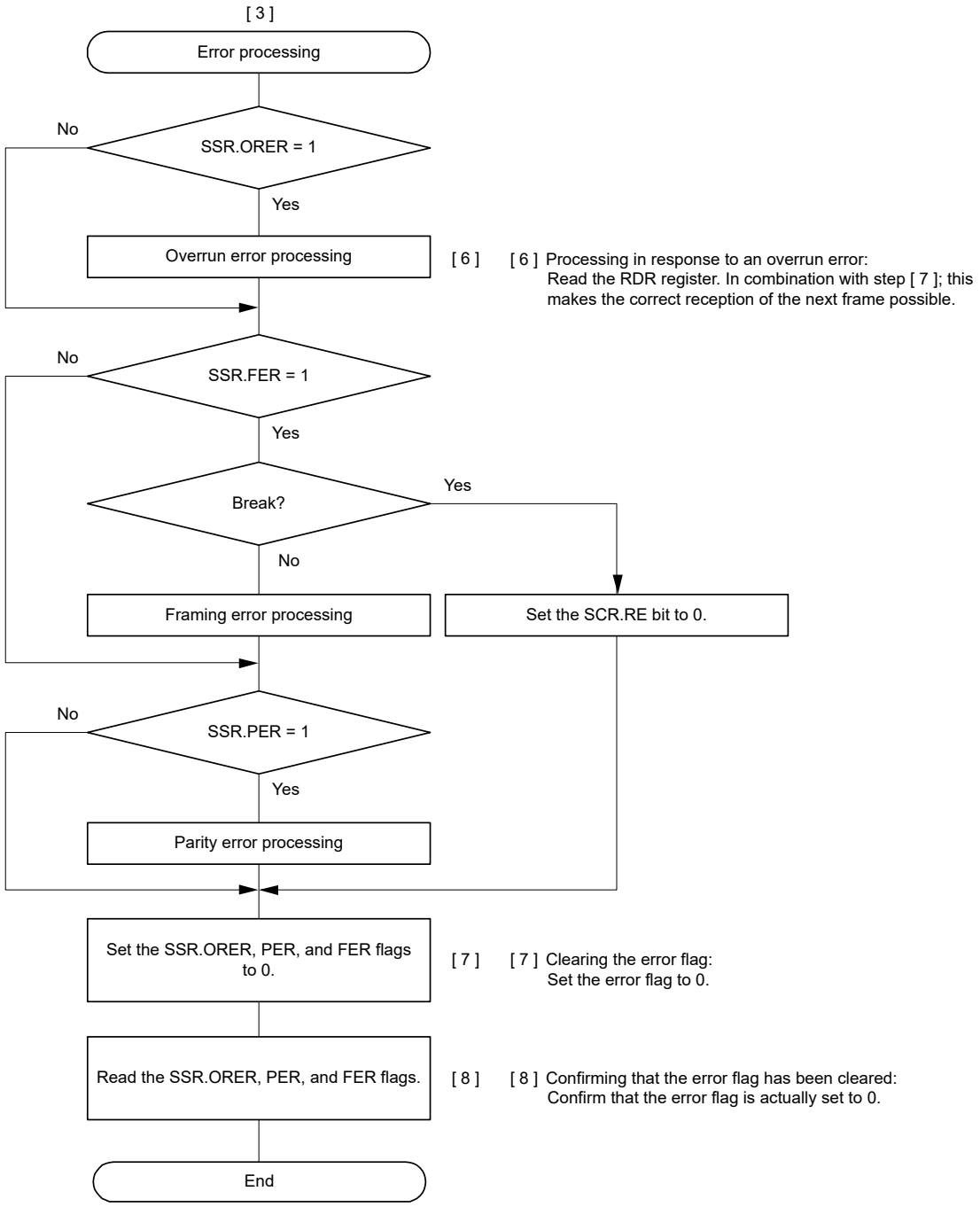



**Figure 27.23 Example Flowchart of Serial Reception in Asynchronous Mode (FIFO not Selected and Address Matching Enabled) (2)**




Note: The RDRHL register is used instead of the RDR register when the 9-bit data length is selected.

**Figure 27.24 Example Flowchart of Serial Reception in Asynchronous Mode (FIFO not Selected and Address Matching Enabled) (3)**

(2) FIFO selected

Figure 27.25 shows an example of a data format that is written to FRDRH register and FRDRL register in asynchronous mode.

In asynchronous mode, 0 is written to the MPB bit in the FRDRH register. Data that corresponds to the data length is written to FRDRH and FRDRL. Unused bits are written as 0. Read in order from FRDRH to FRDRL. If software reads FRDRL, the

SCI updates FER, PER, and receive data (RDAT[8:0]) in the FRDRL register with the next data. The flags RDF, ORER, and DR in the FRDRH register always reflect the associated flags in the SSR_FIFO register.

| Data Length | Register Setting | | Receive data in FRDRH, FRDRL | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FRDRHL | | | | | | | | | | | | | | | |
| | | | FRDRH | | | | | | | | FRDRL | | | | | | | |
| | SCMR.CHR1 | SMR.CHR | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
| 7 bits | 1 | 0 | — | RDF | ORER | FER | PER | DR | 0 | 0 | 0 | 7-bit receive data | | | | | | |
| 8 bits | 1 | 1 | — | RDF | ORER | FER | PER | DR | 0 | 0 | 8-bit receive data | | | | | | | |
| 9 bits | 0 | Don't care | — | RDF | ORER | FER | PER | DR | 0 | 9-bit receive data | | | | | | | | |

Note: 0 is always read for MPB bit (FRDRH[1])
When data length is 7 bits, 0 is always read for FRDRH[0] and FRDRL[7]
When data length is 8 bits, 0 is always read for FRDRH[0]
FRDRH[7] bit is read as an indefinite value

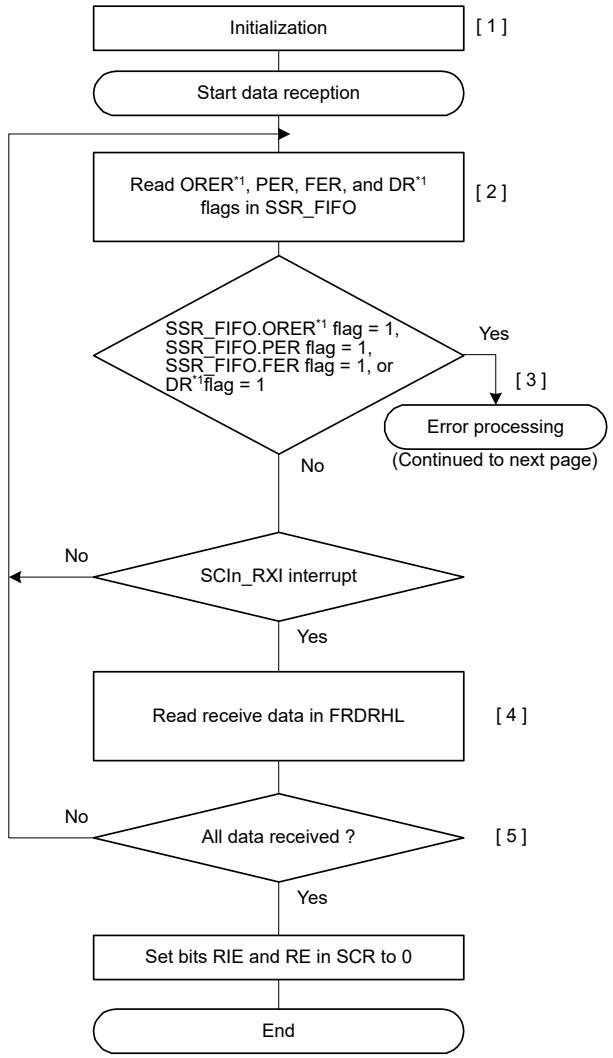
**Figure 27.25 Data format stored in FRDRH and FRDRL with FIFO selected**

In serial data reception, the SCI operates as follows:

1. When the value of the SCR.RE bit becomes 1, the output signal on the CTSn_RTSn pin goes low.
2. The SCI monitors the communications line and, when it detects a start bit, the SCI performs internal synchronization, stores receive data in the RSR register.
3. If the multi-processor communications function is enabled (SMR.MP = 1), see section 27.4.2. Multi-Processor Serial Data Reception. If the address match function (data compare match function) is enabled (DCCR.DCME = 1), the SCI cannot detect a parity or framing error as receive data are skipped (discarded) until the SCI detects a match between the receive data and the data for comparison (CDR.CMPD[*1]).
4. If the SCI detects an address match, the DCCR.DCME bit is automatically cleared, the DCCR.DCMF flag becomes 1, and an SCIn_AM interrupt[*2] request is generated. To enable the generation of an SCIn_RXI interrupt request, set the SCR.RIE bit to 1. The compared receive data are not stored in the RDR register[*3]. The SSR.RDRF flag remains 0.
5. If the SCI detects a framing error in the receive data for which an address match was detected, the DCCR.DFER flag is set to 1, and if the SCI detects a parity error in that frame, the DCCR.DPER flag becomes 1. To enable the generation of an SCIn_ERI interrupt request, set the SCR.RIE bit to 1.
6. If a framing or a parity error is detected (the DCCR.DFER flag or DCCR.DPER flag is 1) in the SCIn_AM interrupt handling routine, set the DCCR.DFER and DCCR.DPER flags to 0 and set the DCCR.DCME bit to 1 to enable the address match function again. If neither a framing nor a parity error has been detected (the DCCR.DFER and DCCR.DPER flags are 0), set the DCCR.DCMF flag to 0. See Figure 27.9.
7. If an overrun error occurs during normal communications, the SSR_FIFO.ORER flag is set to 1. If the SCR.RIE bit in SCR is 1, an SCIn_ERI interrupt request is generated. Receive data is not transferred to the FRDRL[*3] register.
8. If a parity error is detected, the PER flag and receive data are transferred to the FRDRL[*3] register. If the SCR.RIE bit is set to 1, an SCIn_ERI interrupt request is generated.
9. If a framing error is detected, the FER flag and receive data are transferred to the FRDRL[*3] register. If the SCR.RIE bit is set to 1, an SCIn_ERI interrupt request is generated.
10. After a framing error is detected and when SCI detects that the continuous receive data is zero for one frame, reception stops.
11. When the amount of data stored in the FRDRL register falls below the specified receive triggering number, and the next data is not received after 15 etus from the last stop bit in asynchronous mode, the SSR_FIFO.DR flag is set to 1. When the SCR.RIE bit is 1 and the FCR.DRES bit is 0, the SCI generates an SCIn_RXI interrupt request. When the FCR.DRES bit is 1, SCI generates an SCIn_ERI interrupt request.

12. When reception finishes successfully, receive data is transferred to the FRDRL[*3] register. The RDF bit is set to 1 when the amount of receive data written to FRDRHL is equal to or greater than the specified receive triggering number. If the SCR.RIE bit in SCR is 1, an SCIn_RXI interrupt request is generated. Continuous reception is enabled by reading the receive data transferred to the FRDRL[*4] register in the SCIn_RXI interrupt handling routine, before an overrun error occurs. If the received data that is transferred to FRDRL[*5] is less than the RTS trigger number, the CTSn_RTSn pin outputs low.

Note 1. One of three lengths is selected for the target for comparison: CMPD[6:0] is for 7-bit length, CMPD[7:0] is for 8-bit length, and CMPD[8:0] is for 9-bit length.
Note 2. As no interrupt enable bit is assigned to the SCIn_AM interrupt, an interrupt request is generated by setting the DCCR.DCMF to 1.
Note 3. Only read data in the FRDRH and FRDRL registers when 9-bit data length is selected.
Note 4. Read data in order from FRDRH to FRDRL when 9-bit data length is selected.
Note 5. The SCI only checks for update to the FRDRL register and not to the FRDRH register when 9-bit data length is selected.




[ 1 ] SCI initialization:
Set data reception.

[ 2 ] [ 3 ] Receive error processing and break detection:
If a receive error occurs, an SCIn_ERI interrupt is generated. A break can be detected by reading the SPTR.RXDMON flag. An error is identified by reading the ORER*1, PER, DR*1, and FER flags in SSR_FIFO. After performing the appropriate error processing, be sure to set the ORER*1 flag to 0. Reception cannot be resumed if ORER*1 flag is set to 1. The reception operation is continuous, even if the FER = 1 or PER = 1 or DR*1 = 1.

[ 4 ] Read the receive data in FRDRHL in the SCIn_RXI interrupt handling routine. The receive data stored in the FRDRHL register is read until the number of stored data is below the FCR.RTRG value. Confirm the number of the reception data in the FIFO by reading FDR.R.

[ 5 ] Serial reception continuation procedure:
To continue serial reception, before an overrun error occurs, read data from FRDRHL in the SCIn_RXI interrupt handling routine and clear RDF flag and DR flag to 0.
The FRDRHL data can also be read by activating the DMA or DTC. The RDF flag is cleared automatically in this case, so do not write to the RDF flag.

Note 1. Can be read in the FRDRHL.ORER and DR flags.

**Figure 27.26 Example flow of serial reception in asynchronous mode with FIFO selected and Address Matching Enabled (1)**

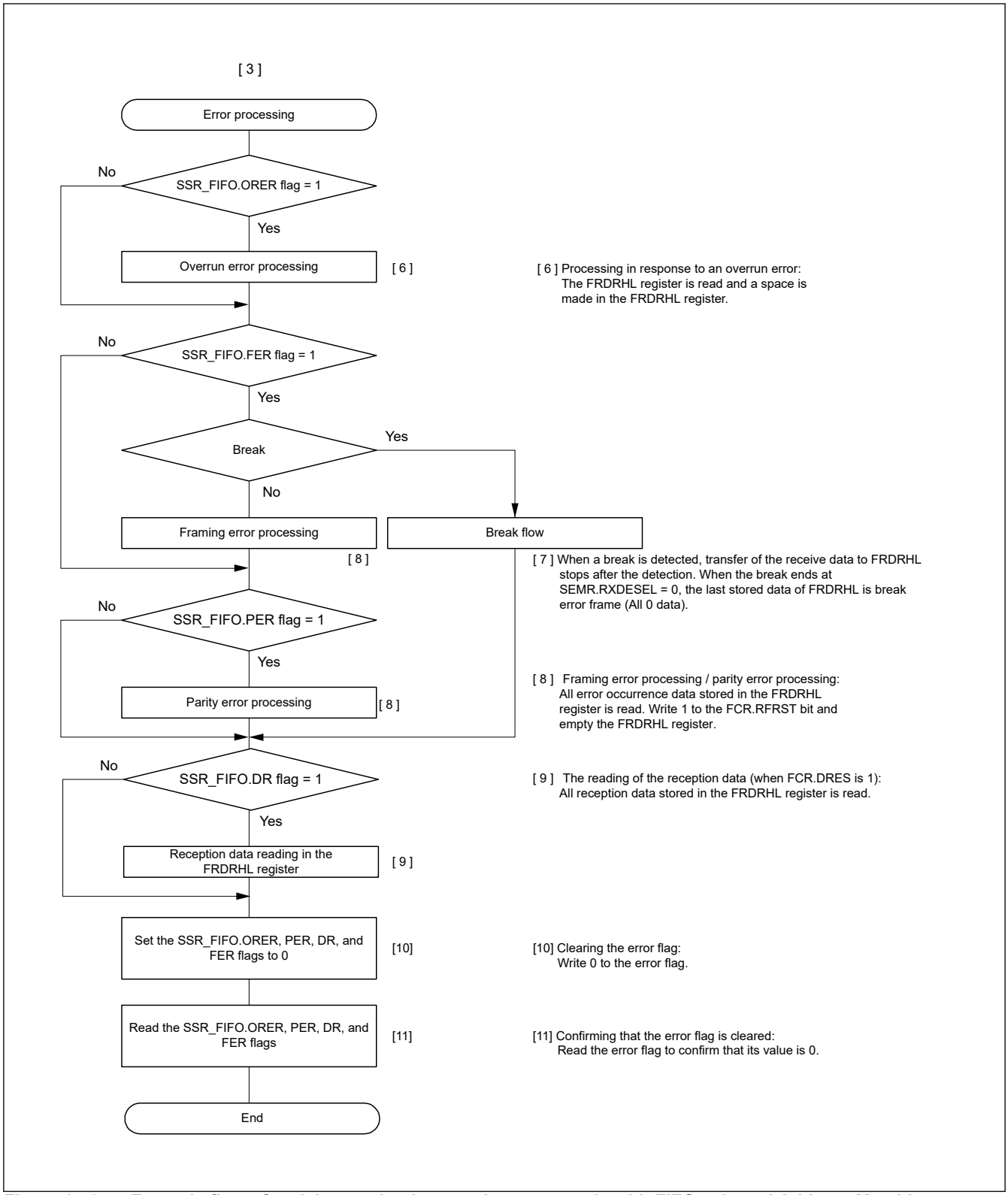
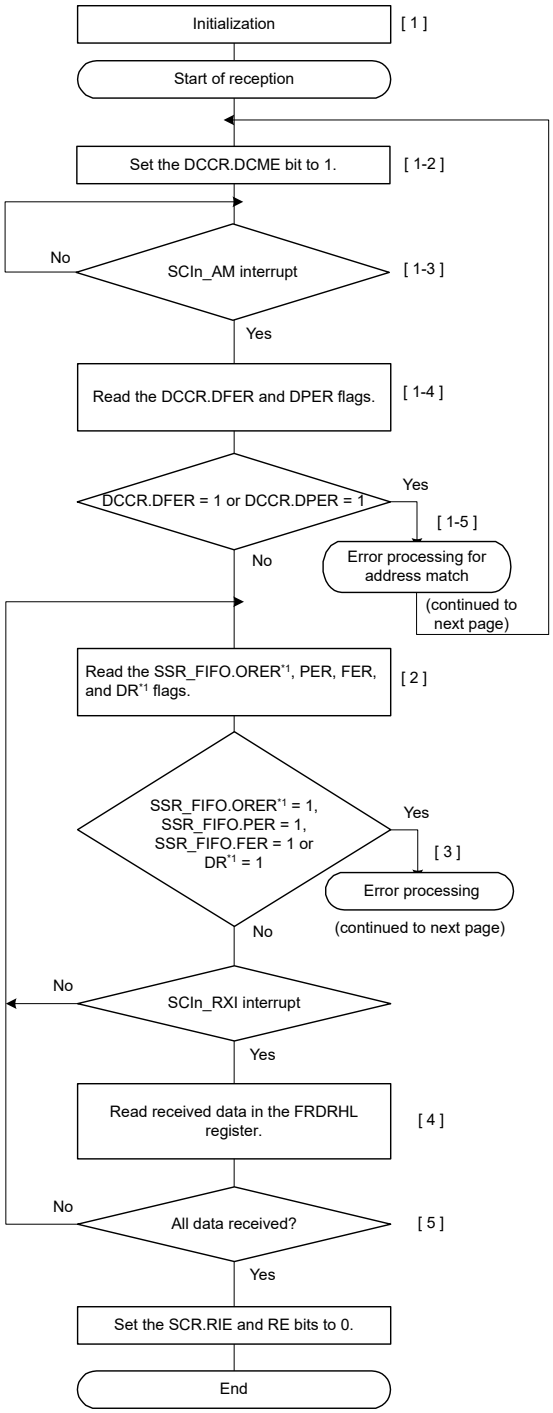



**Figure 27.27 Example flow of serial reception in asynchronous mode with FIFO selected Address Matching Disabled (2)**




[ 1 ] SCI initialization:
Set data reception.

[ 1-2 ] Address match cycle:
Set the DCCR.DCME bit to 1 to enable address matching.

[ 1-3 ] Comparing received data and data for comparison (CDR.CMPD):
Compare receive data and comparison data.
The SCI cannot detect a parity or framing error as received data are skipped (discarded) until the SCI detects a match between the received data and data for comparison.
If the SCI detects an address match, the DCCR.DCME bit is cleared, the DCCR.DCMF flag becomes 1, and an SCIn_AM interrupt request is issued.

[ 1-4 ] [ 1-5 ] Reception error processing:
An error is identified by reading the DFER or DPER flag in DCCR from within the SCIn_AM interrupt handling routine.
After appropriate handling of the error, set both the DFER and DPER flags to 0.
If either of these flags is set to 1, address matching cannot be resumed.

[ 2 ] [ 3 ] Reception error processing and break detection:
When a reception error occurs, an SCIn_ERI interrupt is generated.
A break is identified by reading SPTR.RXDMON.
The error is identified by reading each among the SSR_FIFO.ORER*1, PER, FER, and DR*1 flags.
After appropriate handling of the error, set the ORER*1 flag to 0.
If the ORER*1 flag remains 1, reception cannot be resumed.
The reception operation is continuous, even if the FER = 1, PER = 1 or DR*1 = 1.

[ 4 ] Reading the received data in the FRDRHL register from within the SCIn_RXI interrupt handling routine:
The received data stored in the FRDRHL register are read until the amount of stored data is less than the value of the FCR.RTRG[3:0] bits.
Confirm the number of received data in the FIFO with the value of the FDR.R[4:0] bits.

[ 5 ] Serial reception continuation procedure:
To continue with serial reception, read data from the FRDRHL register from within the SCIn_RXI interrupt handling routine before an overrun error occurs, and set the RDF and DR flags to 0. The data in the FRDRHL register can also be read by activating the DMAC or DTC. In this case, do not write to the RDF flag because the flag is automatically cleared.

Note 1. The FRDRHL.ORER and DR flags can also be read.

**Figure 27.28 Example Flowchart of Serial Reception in Asynchronous Mode (FIFO Selected and Address Matching Enabled) (1)**

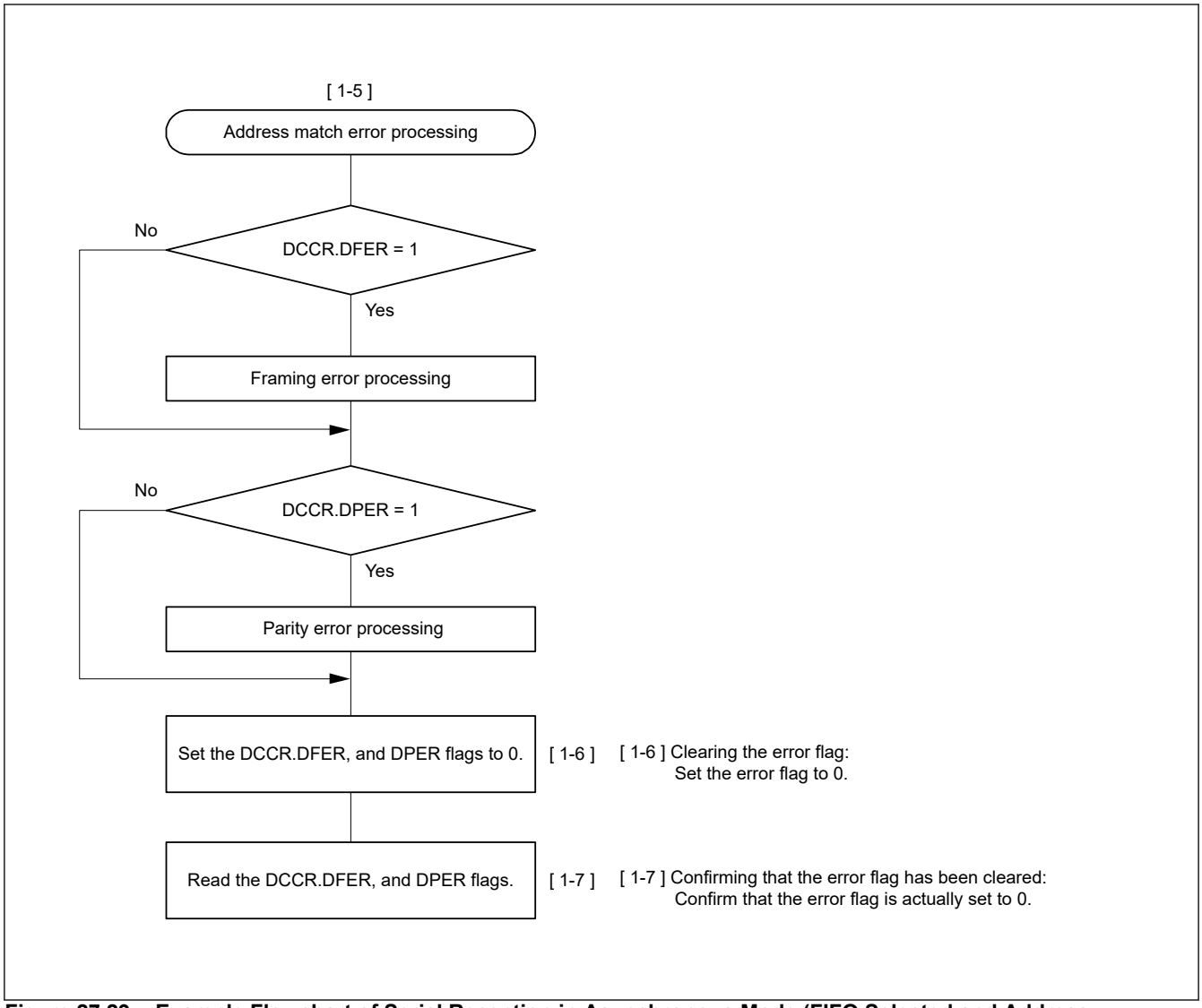



**Figure 27.29 Example Flowchart of Serial Reception in Asynchronous Mode (FIFO Selected and Address Matching Enabled) (2)**

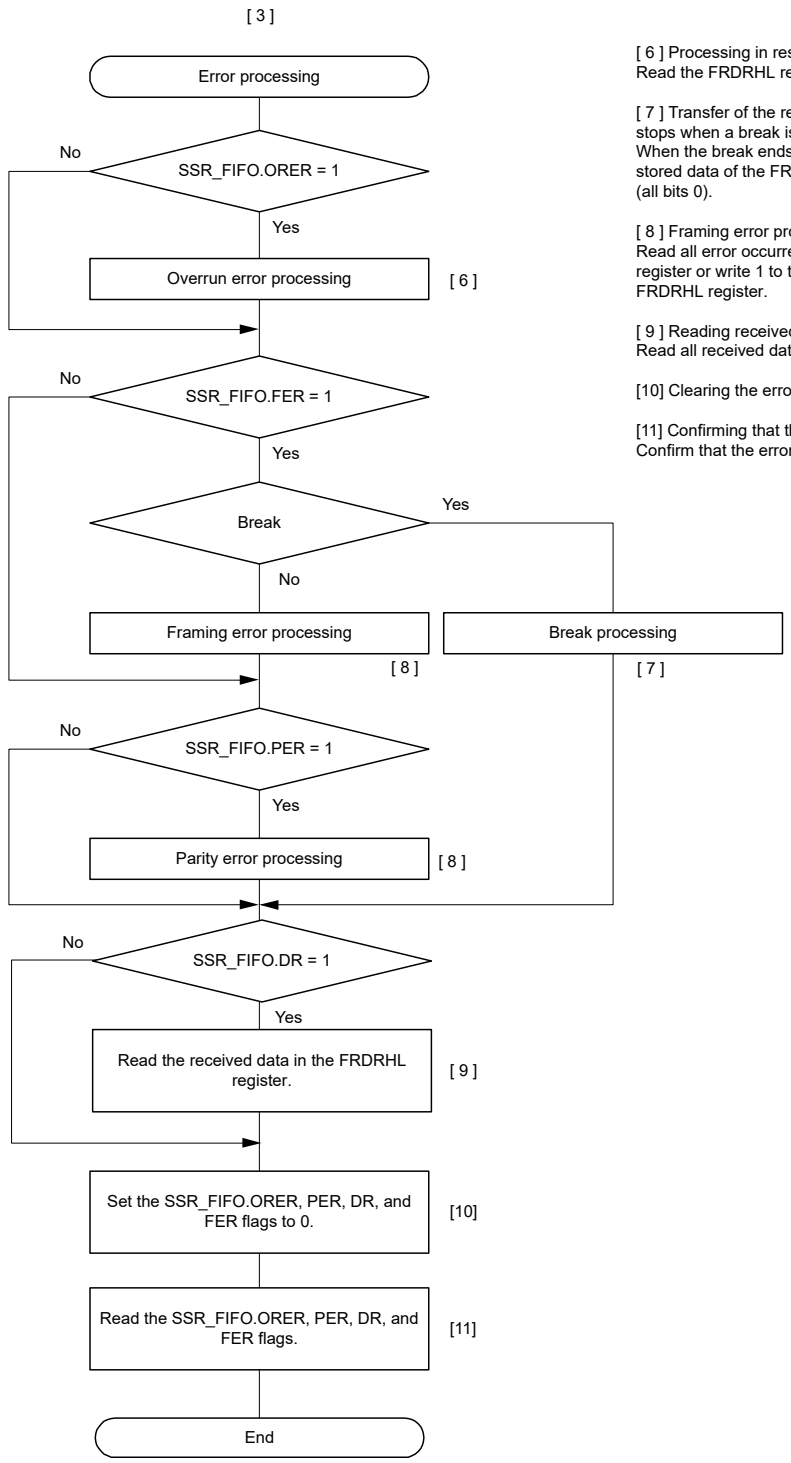



**Figure 27.30 Example Flowchart of Serial Reception in Asynchronous Mode (FIFO Selected and Address Matching Enabled) (3)**

## 27.3.10 The function of adjust receive sampling timing (Asynchronous Mode)

When there is the difference between the rising transfer time and the falling transfer time through a photo coupler, the receive sampling timing at middle of bit affects the reception margin. In this case, the receive sampling timing is able to adjust from the middle of bit to the optimum timing by using this function.

The receive sampling timing is adjusted from the middle of bit by following formula. The adjustable direction is set by ACTR.AJD. When adjusting backward (ACTR.AJD = 0), substitute AJD = +1 and substitute AJD = −1 when adjusting forward (ACTR.AJD = 1).

Adjusted sampling timing = the middle of bit + AJD × (base clock × the setting value of ACTR.AST[2:0])

The setting timing is limited by base clock cycles per 1 bit. For details, see Table 27.31.

An overview of reception operation of the communication through a photo coupler with this function is shown in Figure 27.31, Figure 27.32 and Figure 27.33, the explanation of operation with this function is shown in Figure 27.34.

Do not use this function when there is no difference between the rising transfer time and the falling transfer time, because there is a possibility of deteriorating the reception margin.

**Table 27.31 The acceptable value of ACTR register (asynchronous mode using internal clock)**

| SEMR.ABCSE | SEMR.ABCS | The number of base clock cycles/1bit | The acceptable value of ACTR | |
|---|---|---|---|---|
| | | | ACTR.AJD | ACTR.AST |
| 1 | x | 6 | 0 | 000b – 010b*1 |
| | | | 1 | |
| 0 | 1 | 8 | 0 | 000b – 011b*1 |
| | | | 1 | |
| 0 | 0 | 16 | 0 | 000b – 111b |
| | | | 1 | |

Note: x: Don't care
Note 1. When the value of ACTR.AST exceeds the acceptable value, sampling is done at default timing. (Adjustment of sampling is not done.)

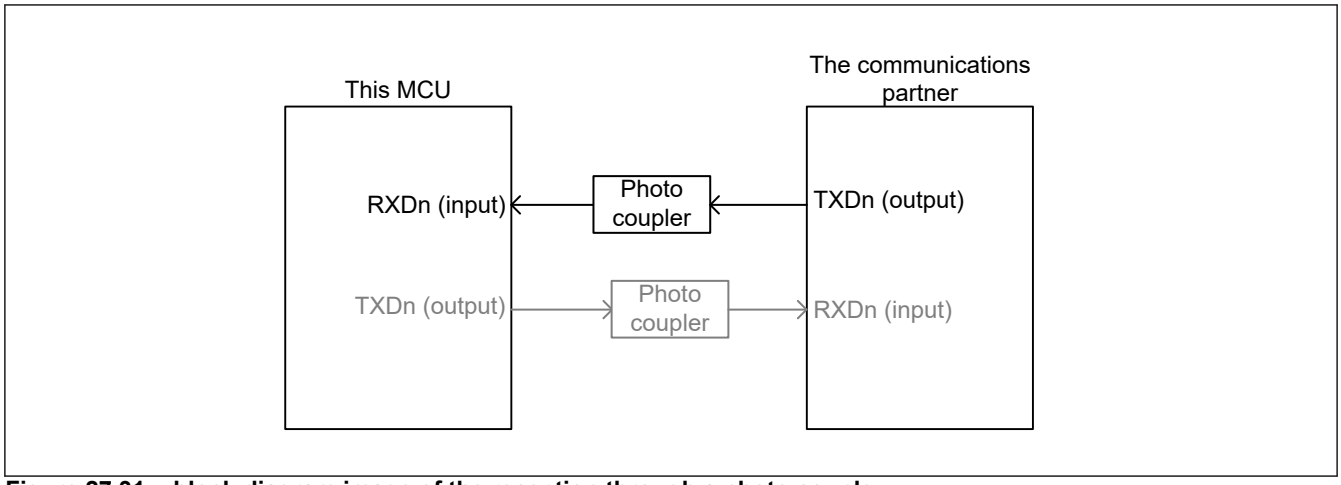
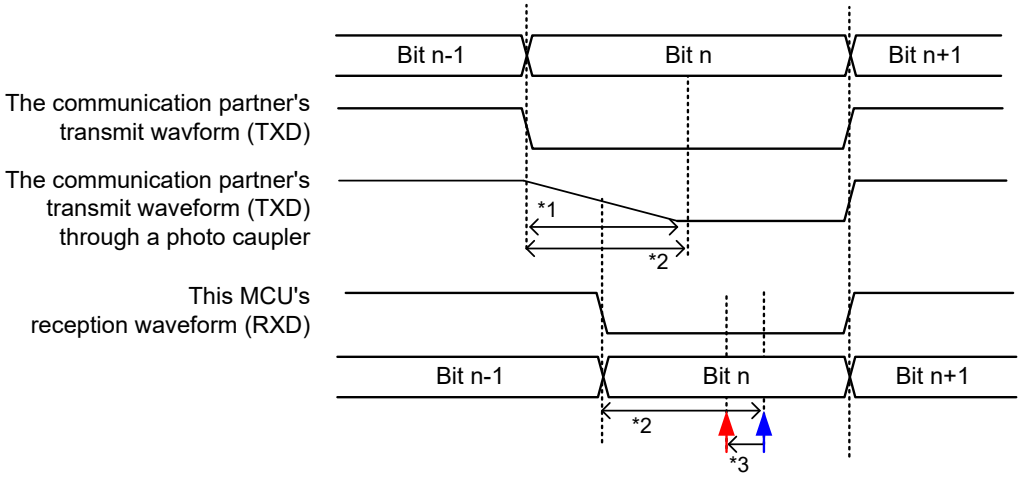



**Figure 27.31 block diagram image of the reception through a photo couple**

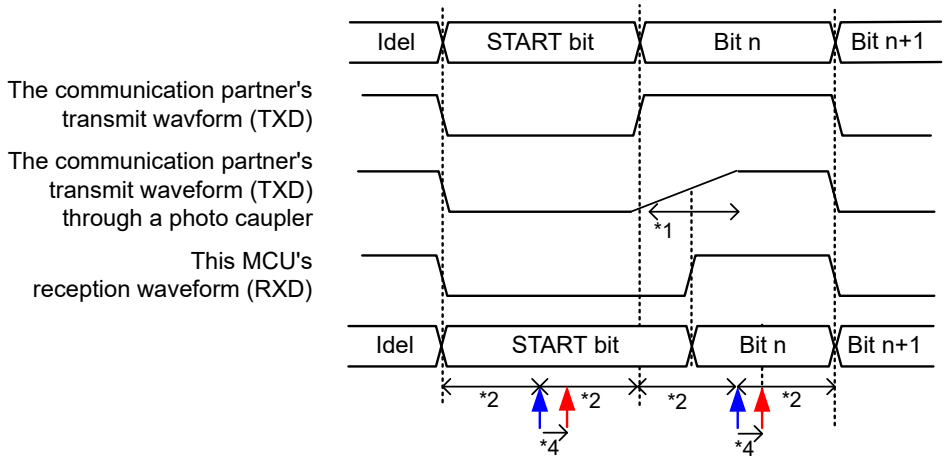
(a) In the case of the falling transfer time >> rising transfer time

The falling edge of reception waveform is made dull like following chart.
In this case, you can sampling at the middle of bit if you adjust the receive sampling timing to forward (AJD = 1).




(b) In the case of the falling transfer time << rising transfer time

The rising edge of reception waveform is made dull like following chart.
Thus, the reception margin of communications partner will be bad. In this case, you can improve the reception margin if you adjust the receive sampling timing to back.




The receive sampling timing when unadjusted (middle of bit)

The adjusted receive sampling timing

Note: This waveform shows the operation image of adjustment in reception sampling timing.
Note 1. The dull period by a photo coupler
Note 2. Bit center timing at set communication rate
Note 3. When ACTR.AJD is 1, the receive sampling timing is shifted to forward by the setting value of ACTR.AST[2:0].
Note 4. When ACTR.AJD is 0, the Receive sampling timing is shifted to backward by the setting value of ACTR.AST[2:0].

**Figure 27.32 Overview of reception operation of the communication through a photo coupler**

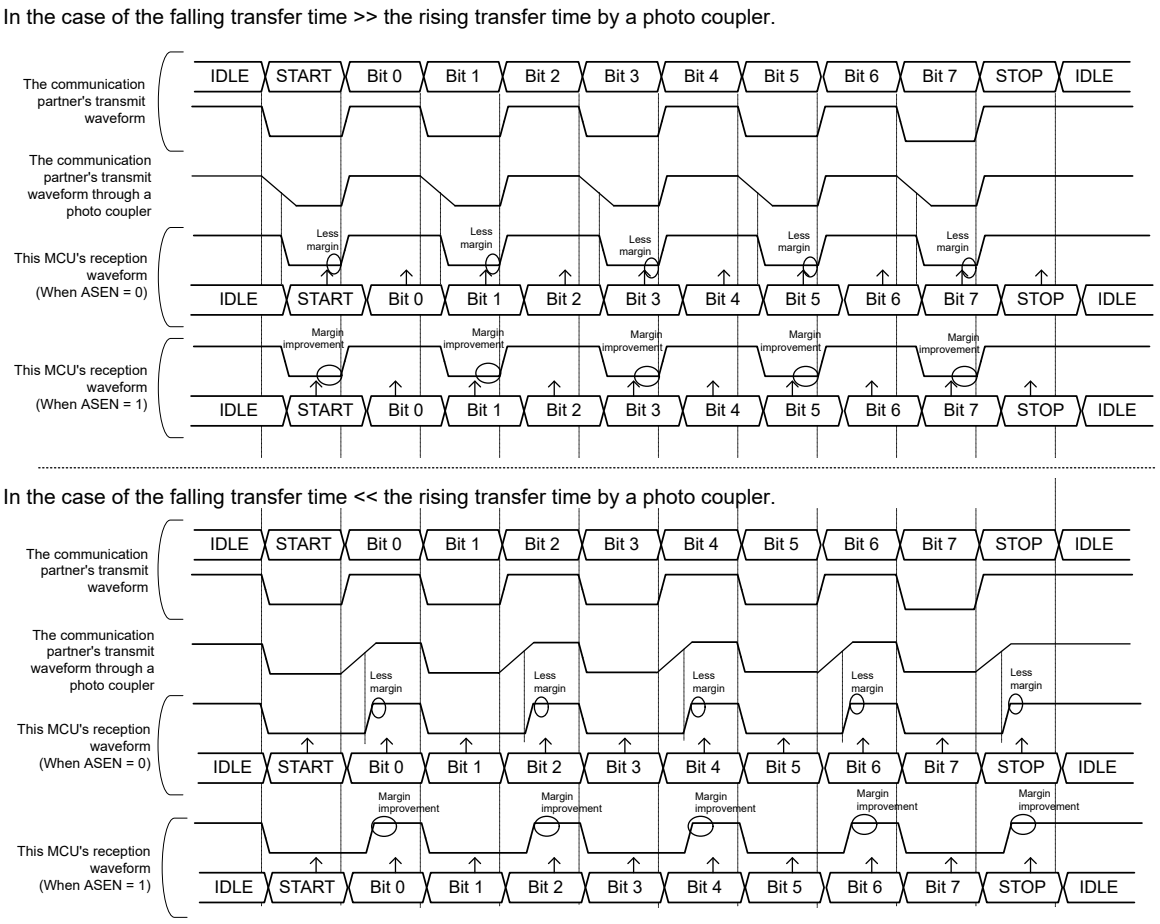



**Figure 27.33 Example of improvement in reception margin by the reception sampling timing adjustment function**

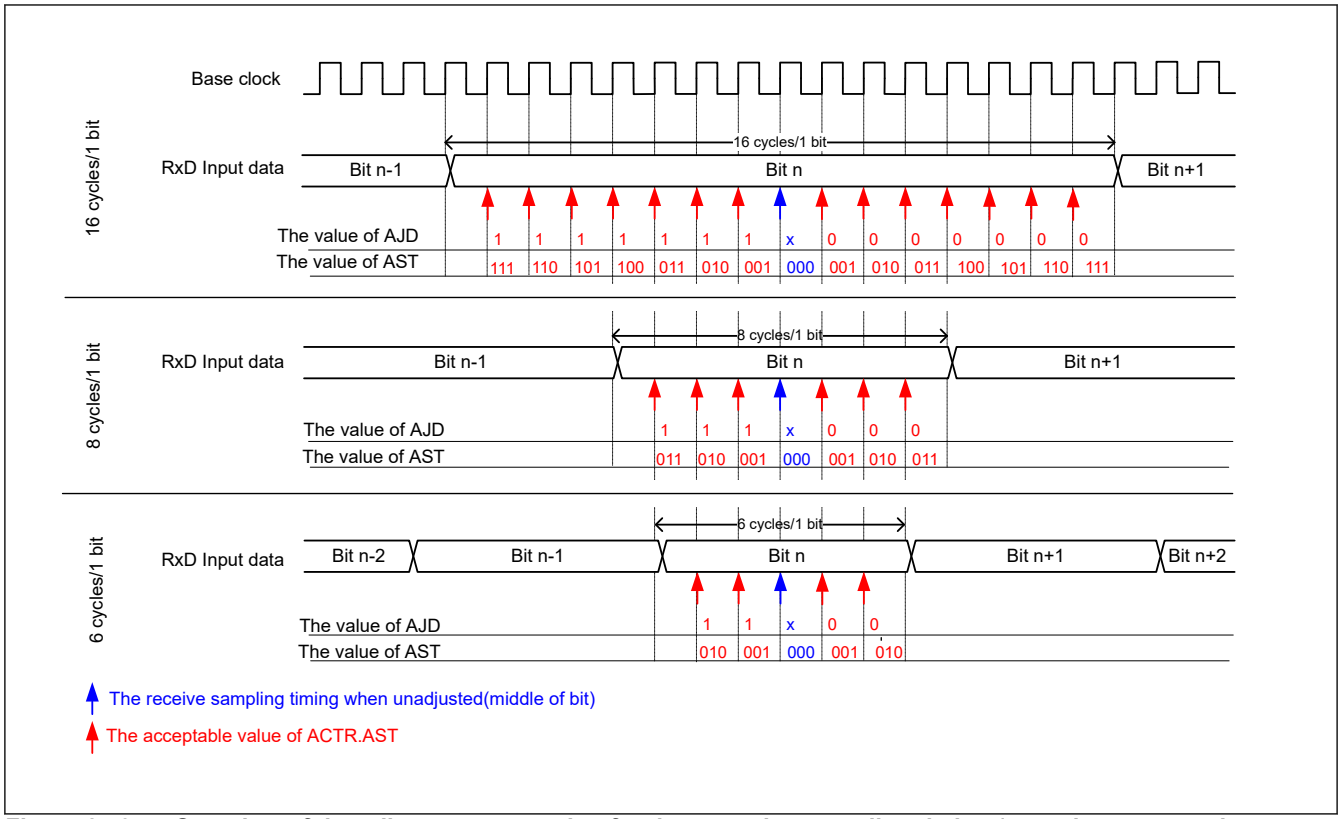



**Figure 27.34 Overview of the adjustment operation for the reception sampling timing (asynchronous mode using internal clock)**

## 27.3.11 The function of adjust transmit timing (Asynchronous Mode)

In communication via a photo coupler or the like, when either the rising or falling transition time of the TXDn output signal is long, then a communication partner receive dulled waveform. In this case, the reception margin may be affected.

In these cases, make a communication partner to be sampling at middle of bit using the function of adjust transmit timing.

When SPTR.ATEN is 1, this function can adjust the edge timing at the timing calculated by the following formula for the edge set with ACTR.AET.

The adjustment edge timing = the base clock × ACTR.ATT[2:0]

In addition, the upper limit of the adjustment edge timing is limited by setting the base clock cycles. For details, see Table 27.32.

A transmission movement image figure of the communication through a photo coupler with this function is shown in Figure 27.35, Figure 27.36 and Figure 27.37, the overview of operation with this function is shown in Figure 27.38 and Figure 27.39.

Do not use this function when there is not the difference between the rising transfer time and the falling transfer time, there is a possibility of deteriorating the reception margin of a communication partner.

**Table 27.32 The acceptable value of ACTR.AET and ACTR.ATT (asynchronous mode using internal clock) (1 of 2)**

| ABCSE | ABCS | The number of base clock cycles/1bit | The acceptable value of ACTR | |
|---|---|---|---|---|
| | | | AET | ATT[2:0] |
| 1 | x | 6 | 0 | 000b – 101b |
| | | | 1 | |
| 0 | 1 | 8 | 0 | 000b – 111b |
| | | | 1 | |

**Table 27.32 The acceptable value of ACTR.AET and ACTR.ATT (asynchronous mode using internal clock) (2 of 2)**

| ABCSE | ABCS | The number of base clock cycles/1bit | The acceptable value of ACTR | |
|---|---|---|---|---|
| | | | AET | ATT[2:0] |
| 0 | 0 | 16 | 0 | 000b – 111b |
| | | | 1 | |

Note: x: Don't care
Note: When the value of ACTR.AET/ATT is out of the acceptable value, this SCI module doesn't adjust transmit timing.

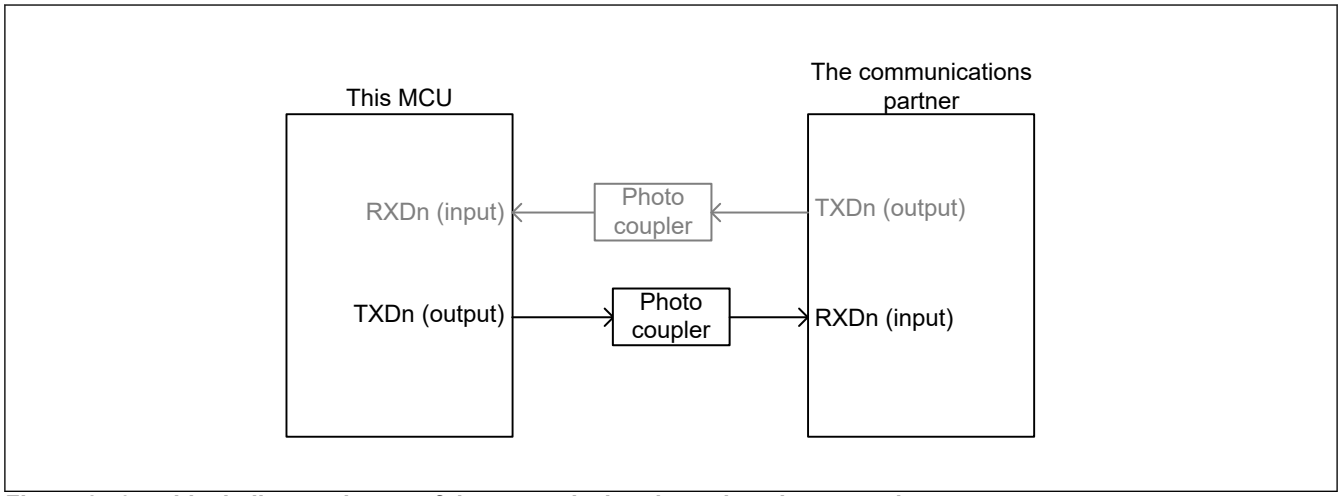
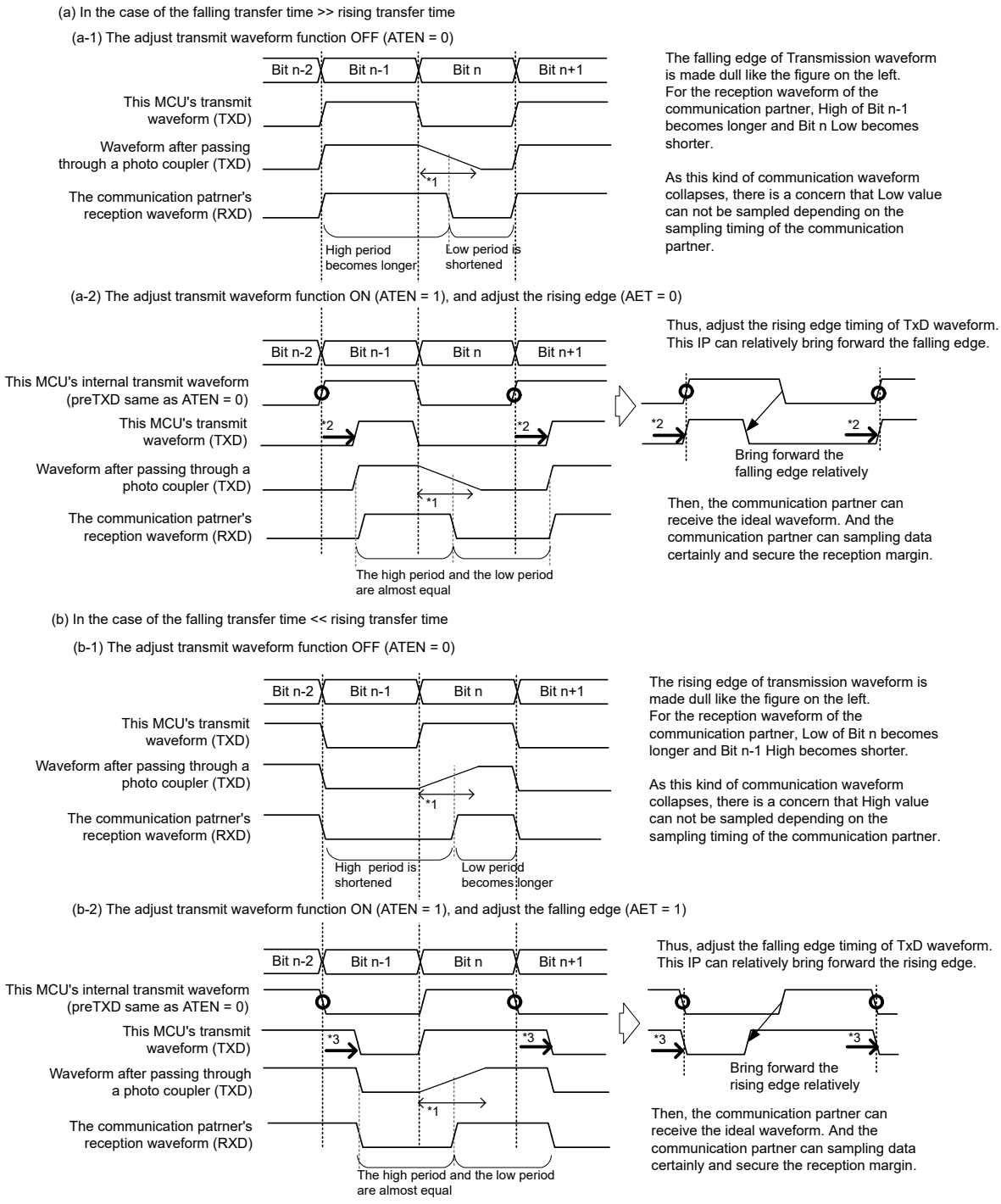



**Figure 27.35 block diagram image of the transmission through a photo coupler**




Note: This waveform shows the operation image of adjustment transmit waveform.
Note 1. The dull period by a photo coupler
Note 2. When ACTR.AET is 0, the rising edge of transmit waveform timing is shifted to back by the setting value of ACTR.ATT[2:0]. This IP transmit waveform bringing forward the falling edge relatively.
Note 3. When ACTR.AET is 1, the falling edge of transmit waveform timing is shifted to back by the setting value of ACTR.ATT[2:0]. This IP transmit waveform bringing forward the rising edge relatively.

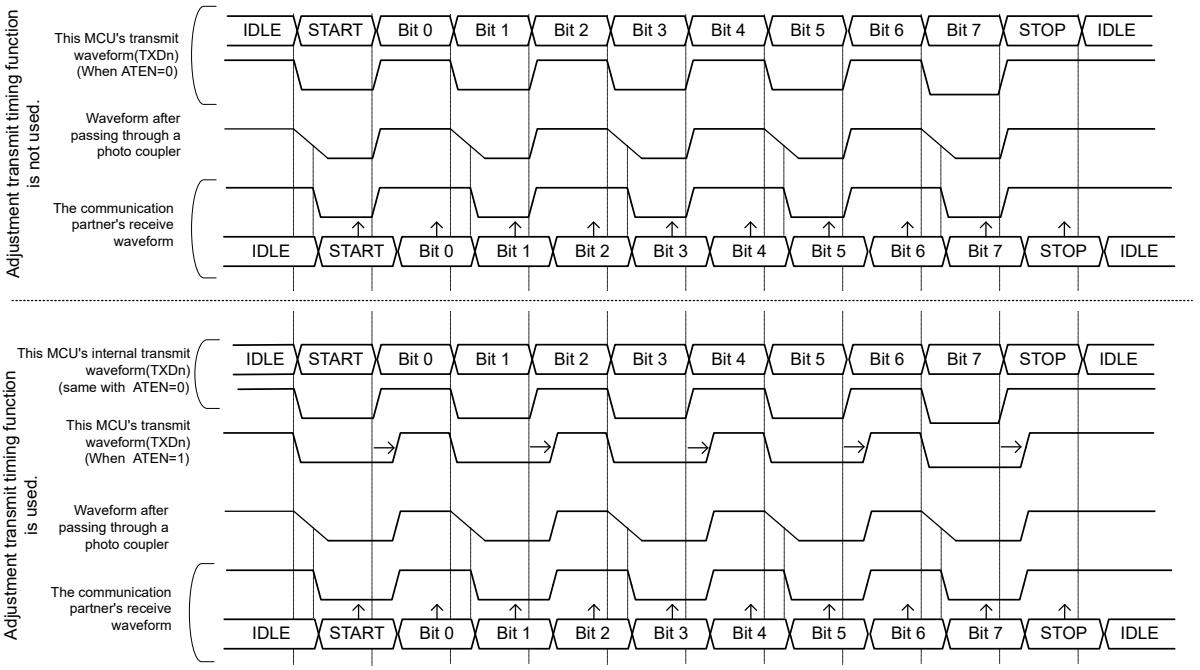
**Figure 27.36 The overview of transmission operation in the communication through a photo coupler**

The explanation of transmit waveforms of the communication through a photo coupler using adjust transmit timing function

When using the transmission timing adjustment function, adjust the edge timing of the transmission waveform and correct the reception waveform of the communication partner

The following example is 8 bit long data.

(a) In the case of the falling edge transfer time >> the rising transfer time




(b) In the case of the falling edge transfer time << the rising transfer time

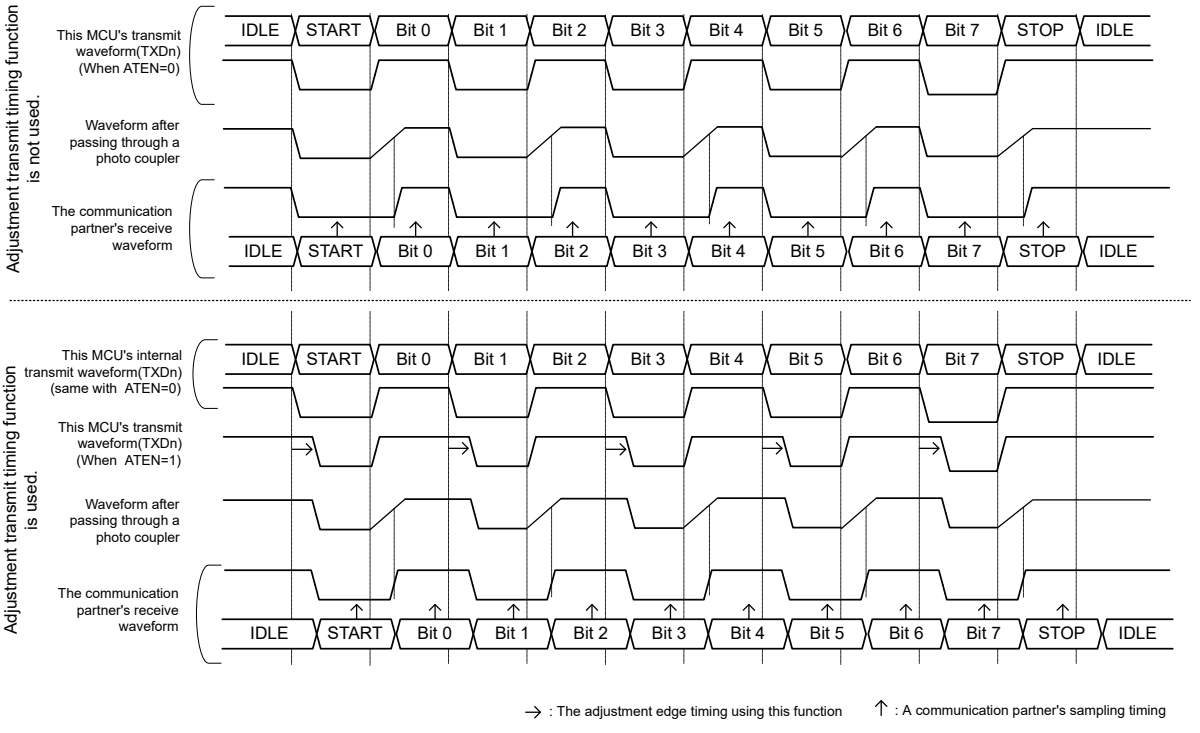
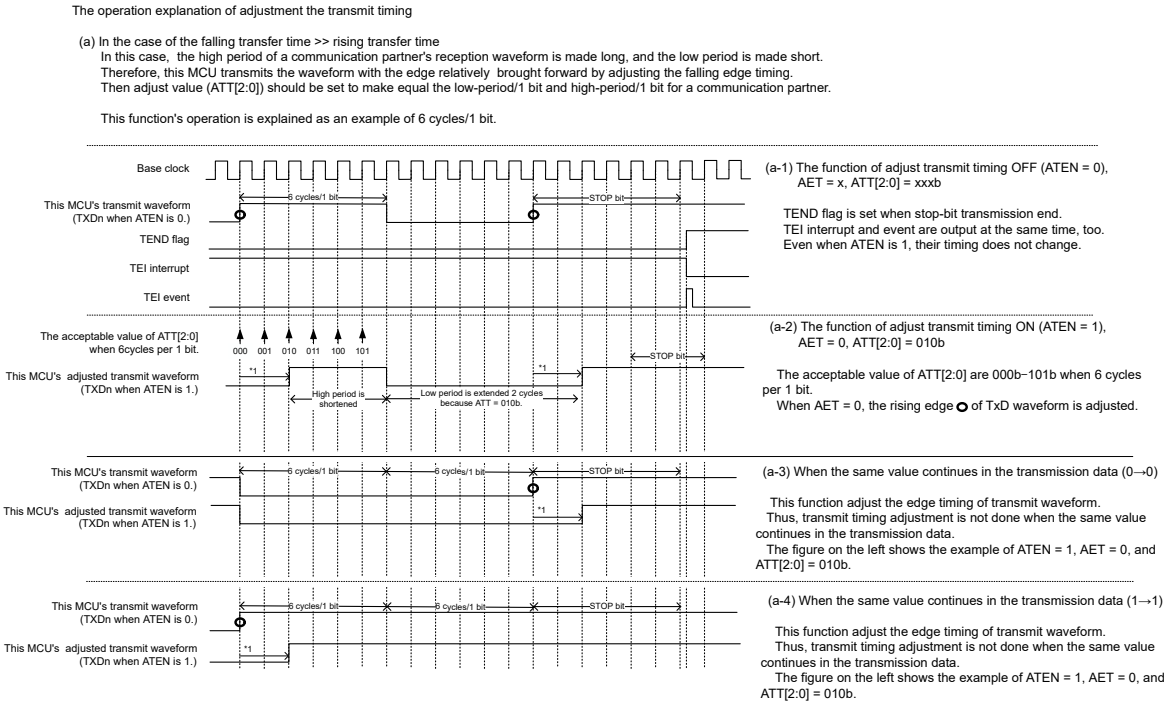
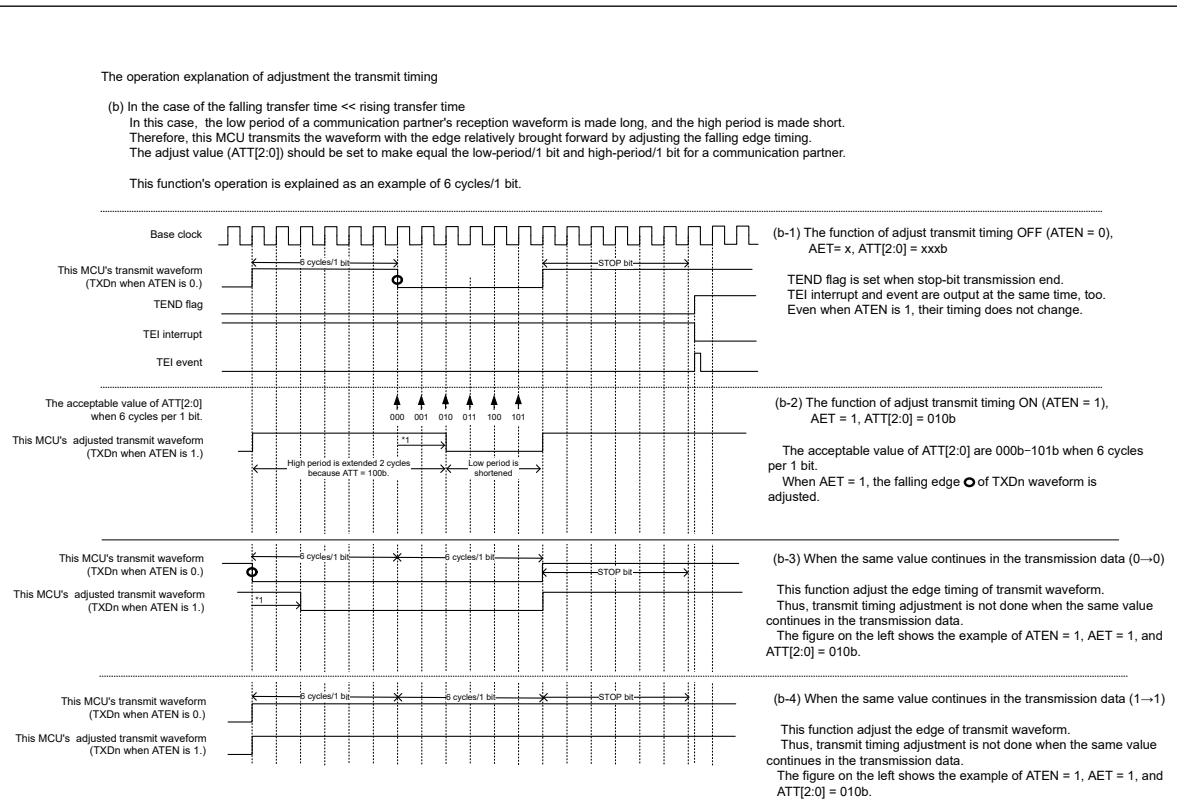



**Figure 27.37 The explanation for the transmit waveform through a photo coupler**




Note 1. The rising edge of transmit timing is shifted to back by the setting value of ACTR.ATT[2:0].

**Figure 27.38 The adjustment operation explanation for the transmit timing when AET is 0**




Note 1. The falling edge of transmit timing is shifted to back by the setting value of ACTR.ATT[2:0].

**Figure 27.39 The adjustment operation explanation for the transmit timing when AET is 1**

## 27.4 Multi-Processor Communication Function

The multi-processor communication function enables the SCI to transmit and receive data between multiple processors by sharing an asynchronous serial communication line that has an added multi-processor bit. In multi-processor communication, a unique ID code is allocated to each receiving station. Serial communication cycles consist of an ID transmission cycle to specify the receiving station and a data transmission cycle to transmit data to the specified receiving station.

The multi-processor bit is used to distinguish between the ID transmission cycle and the data transmission cycle:

- When the multi-processor bit is set to 1, the transmission cycle is the ID transmission cycle
- When the multi-processor bit is set to 0, the transmission cycle is the data transmission cycle

Figure 27.40 shows an example of communication between processors using a multi-processor format. First, a transmitting station transmits communication data in which the multi-processor bit set to 1 is added to the ID code of the receiving station. Next, the transmitting station transmits communication data in which the multi-processor bit set to 0 is added to the transmit data. After receiving communication data with the multi-processor bit set to 1, the receiving station compares the received ID with the ID of the receiving station itself. If the two match, the receiving station receives communication data that is subsequently transmitted. If the received ID does not match with the ID of the receiving station, the receiving station skips the communication data until it receives data in which the multi-processor bit is set to 1.

(1) Non-FIFO selected

To support this function, the SCI provides the SCR.MPIE bit. When the MPIE bit is set to 1, the following operations are disabled until the reception of data in which the multi-processor bit is set to 1:

- Transfer of receive data from the RSR register to the RDR register (the RDRHL register when 9-bit data length is selected)
- Detection of a receive error
- Setting of the respective RDRF, ORER, and FER status flags in the SSR register

When the SCI receives a character in which the multi-processor bit is set to 1, the SSR.MPBT bit is set to 1 and the SCR.MPIE bit is automatically cleared, returning the SCI to normal reception operation. If the SCR.RIE bit is set to 1, an SCIn_RXI interrupt is generated.

When the multi-processor format is specified, the parity bit function is disabled. Apart from this, there is no difference from operation in normal asynchronous mode. The clock used for the multi-processor communication is the same as the clock used in normal asynchronous mode.

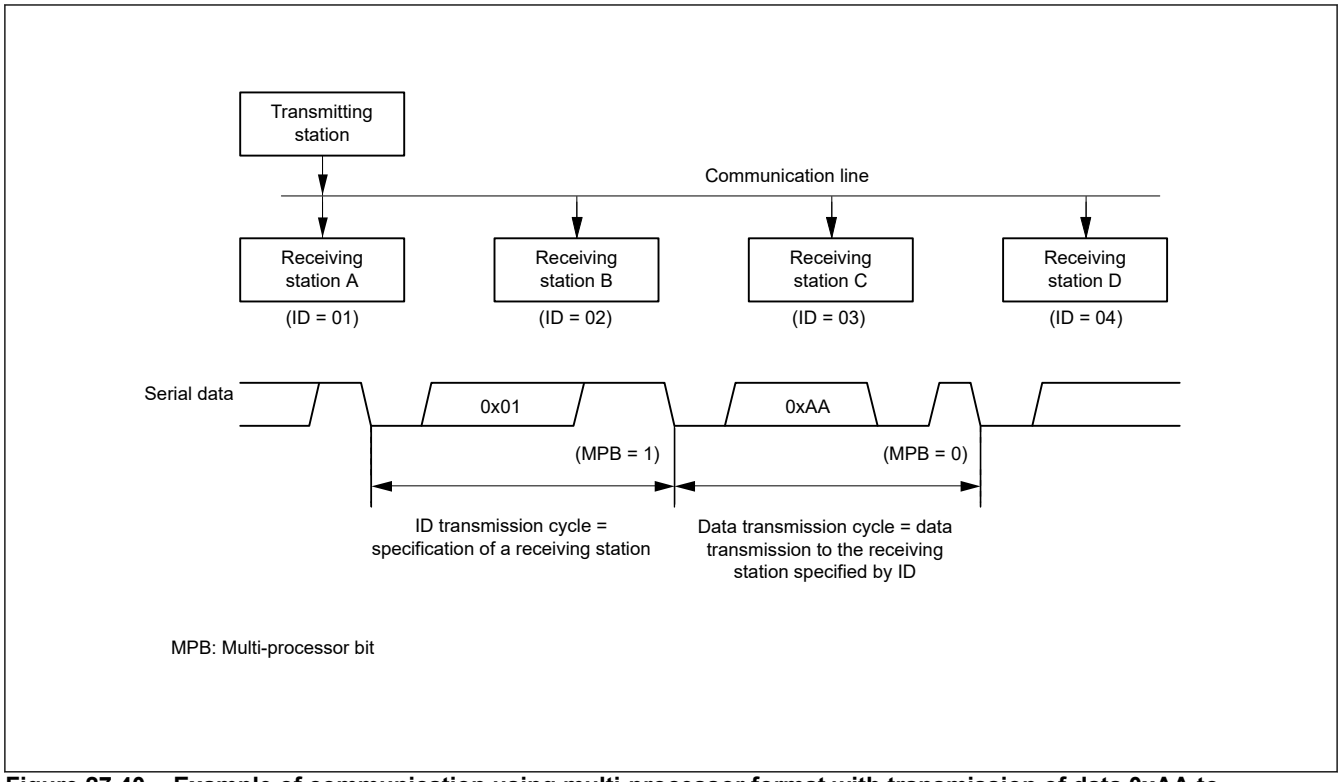



**Figure 27.40 Example of communication using multi-processor format with transmission of data 0xAA to receiving station A**

(2) FIFO selected

For data transmission, software must write data to FTDRHL.MPBT that corresponds to transmit data in FTDRHL.TDAT. For data reception, the multi-processor bit that is part of the receive data is written to FTDRHL.MPB and receive data is written to FRDRL.

When the MPIE bit is set to 1, the following operations are disabled until reception of data in which the multi-processor bit is set to 1:

- Transfer of receive data from the RSR register to the FRDRHL register
- Detection of a receive error
- Break
- Setting of the respective RDF, ORER, and FER status flags in the SSR_FIFO register

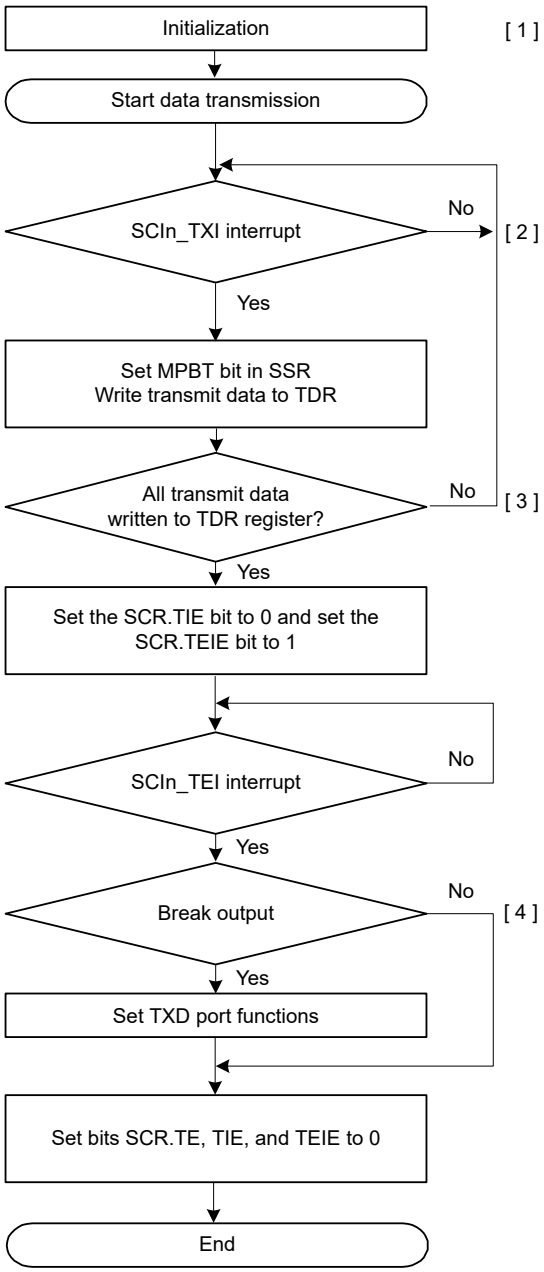
When the SCI receives an 8-bit character in which the multi-processor bit is set to 1, the FTDRHL.MPB bit is set to 1 and receive data is written to FRDRHL.RDAT. The SCR.MPIE bit is automatically cleared, returning the SCI to normal reception operation. If the SCR.RIE bit is set to 1, an SCIn_RXI interrupt is generated.

When the multi-processor format is specified, the parity bit function is disabled. Apart from this, there is no difference from operation in normal asynchronous mode with FIFO selected.

## 27.4.1 Multi-Processor Serial Data Transmission

**(1) Non-FIFO selected**

Figure 27.41 shows an example flow of multi-processor data transmission. In the ID transmission cycle, the ID must be transmitted with the SSR.MPBT bit set to 1. In the data transmission cycle, the data must be transmitted with the MPBT bit set to 0. The rest of the operations are the same as operations in asynchronous mode. Write the values in the order of the FTDRH register then the FTDRL register.




[ 1 ] SCI initialization:
Set data transmission.
After the SCR.TE bit is set to 1, 1 is output for a frame, and transmission is enabled.

[ 2 ] SCIn_TXI interrupt request:
When transmit data is transferred from TDR to TSR, a transmit data empty interrupt (SCIn_TXI) request is generated.
Set the MPBT bit in SSR to 0 or 1, and write transmit data to TDR once in the SCIn_TXI interrupt processing routine.

[ 3 ] Serial transmission continuation procedure:
To continue serial transmission, write transmit data to TDR once using an SCIn_TXI interrupt request.
Transmit data can also be written to TDR by activating the DMAC or DTC.
When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 and the SCR.TEIE bit to 1 after the last of the data to be transmitted is written to the TDR.

[ 4 ] Break output at the end of serial transmission:
To output a break in serial transmission, after setting the output state (LOW level output ) of TXD pin by SPTR.SPB2IO and SPTR.SPB2DT bits, set the SCR.TE bit to 0.

**Figure 27.41 Example flow of multi-processor serial transmission with non-FIFO selected**

### (2) FIFO selected

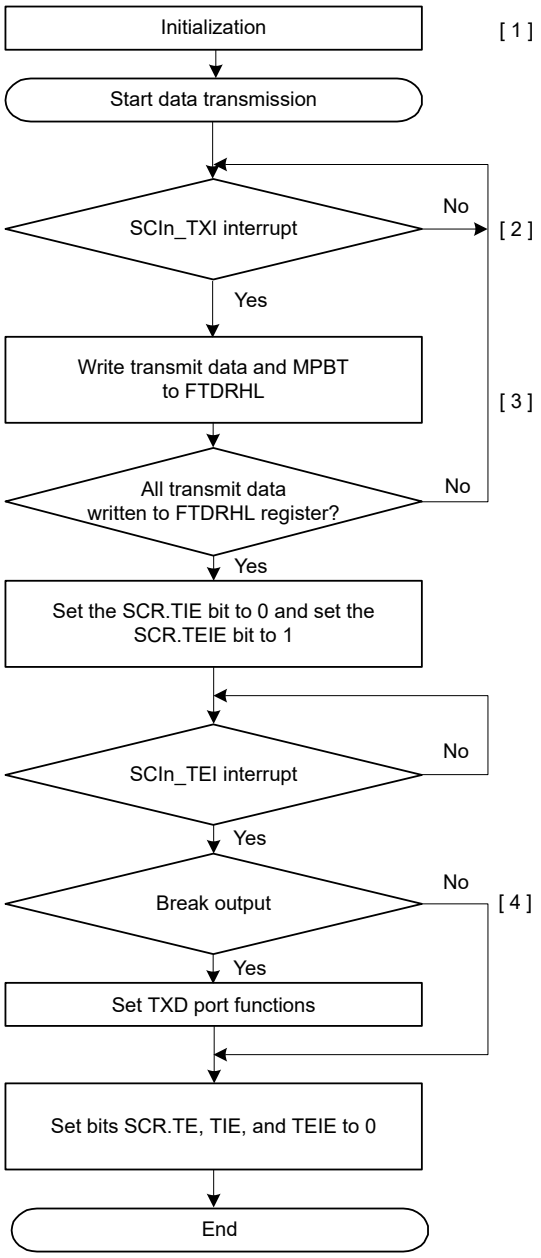
Figure 27.42 shows an example of data format that is written to FTDRH and FTDRL in multi-processor mode. The FTDRH.MPBT bit is set to 1. Data is set to FTDRH and FTDRL with the correct data length. Write 0 for unused bits. Write in order from FTDRH to FTDRL.

| Data Length | Register Setting | | Transmit data in FTDRH, FTDRL | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FTDRHL | | | | | | | | | | | | | | | |
| | | | FTDRH | | | | | | | | FTDRL | | | | | | | |
| | SCMR.CHR1 | SMR.CHR | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
| 7 bits | 1 | 0 | — | — | — | — | — | — | MPBT | — | — | 7-bit transmit data | | | | | | |
| 8 bits | 1 | 1 | — | — | — | — | — | — | MPBT | — | 8-bit transmit data | | | | | | | |
| 9 bits | 0 | Don't care | — | — | — | — | — | — | MPBT | 9-bit transmit data | | | | | | | | |

Note: —: Invalid. The write value should be 0.

**Figure 27.42 Data format written to FTDRH and FTDRL in multi-processor mode with FIFO selected**

Figure 27.43 shows an example flow of multi-processor serial transmission with FIFO selected. In the ID transmission cycle, the ID must be transmitted with the FTDRH.MPBT bit set to 1. In the data transmission cycle, the data must be transmitted with the MPBT bit set to 0. The rest of the operations are the same as operations in asynchronous mode with FIFO selected.




[ 1 ] SCI initialization:
Set data transmission.
After the SCR.TE bit is set to 1, 1 is output for a frame, and transmission is enabled.

[ 2 ] Transmit data write to FTDRHL by an SCIn_TXI interrupt request:
When transmit data is transferred from FTDRHL to TSR, when the quantity of receive data written in FTDRHL is equal to or less than the specified transmit triggering number, a transmit data FIFO empty interrupt (SCIn_TXI) request is generated.

[ 3 ] Serial transmission continuation procedure:
To continue serial transmission, write transmit data and MPBT to FTDRHL once using an SCIn_TXI interrupt request.
Transmit data can also be written to FTDRHL by activating the DMAC or DTC.
When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 and the SCR.TEIE bit to 1 after the last of the data to be transmitted is written to the FTDRHL.

[ 4 ] Break output at the end of serial transmission:
To output a break in serial transmission, after setting the output state (LOW level output) of TXD pin by SPTR.SPB2IO and SPTR.SPB2DT bits, set the SCR.TE bit to 0.

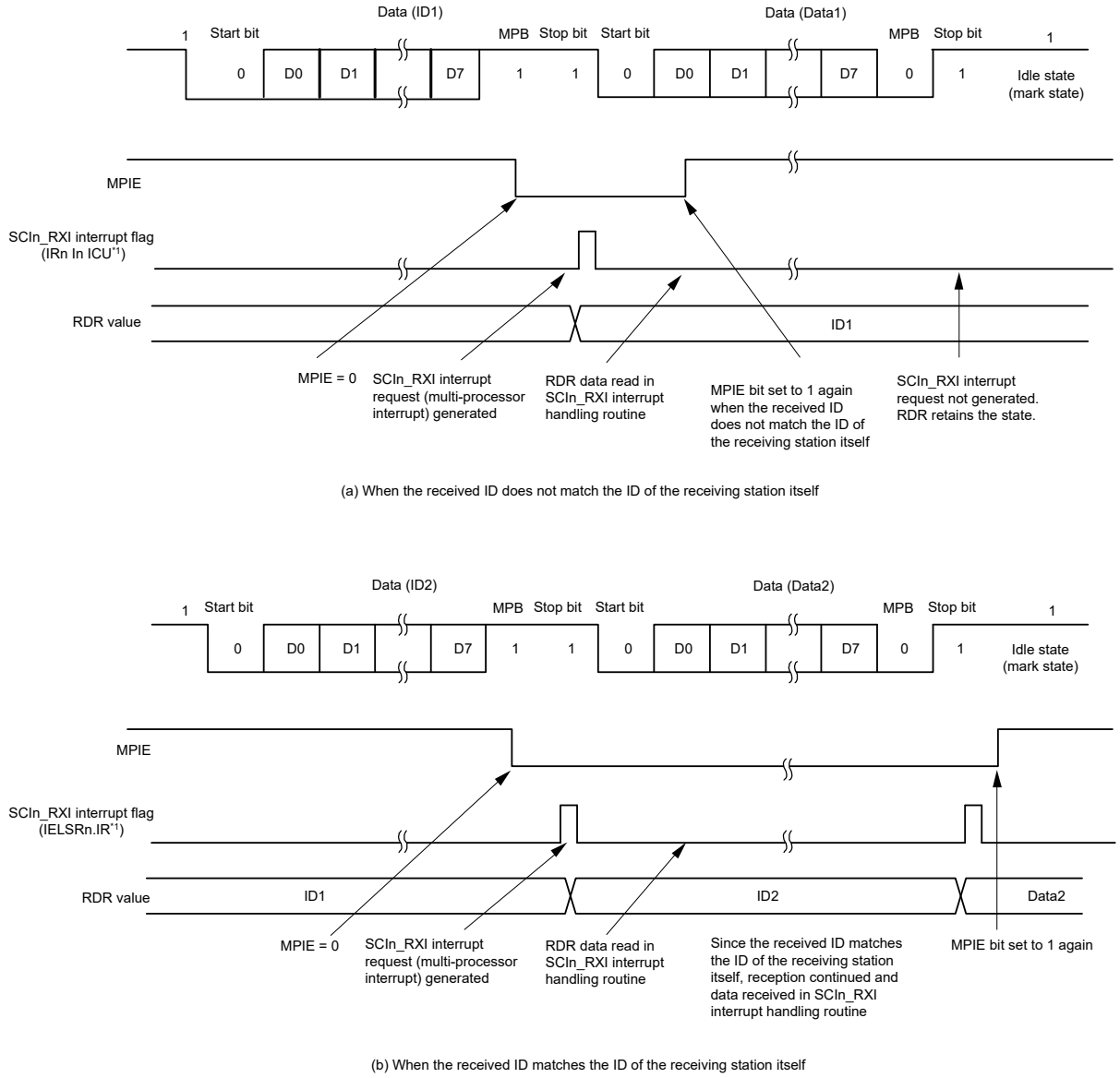
**Figure 27.43 Example flow of serial transmission in multi-processor mode with FIFO selected**

## 27.4.2 Multi-Processor Serial Data Reception

### (1) Non-FIFO selected

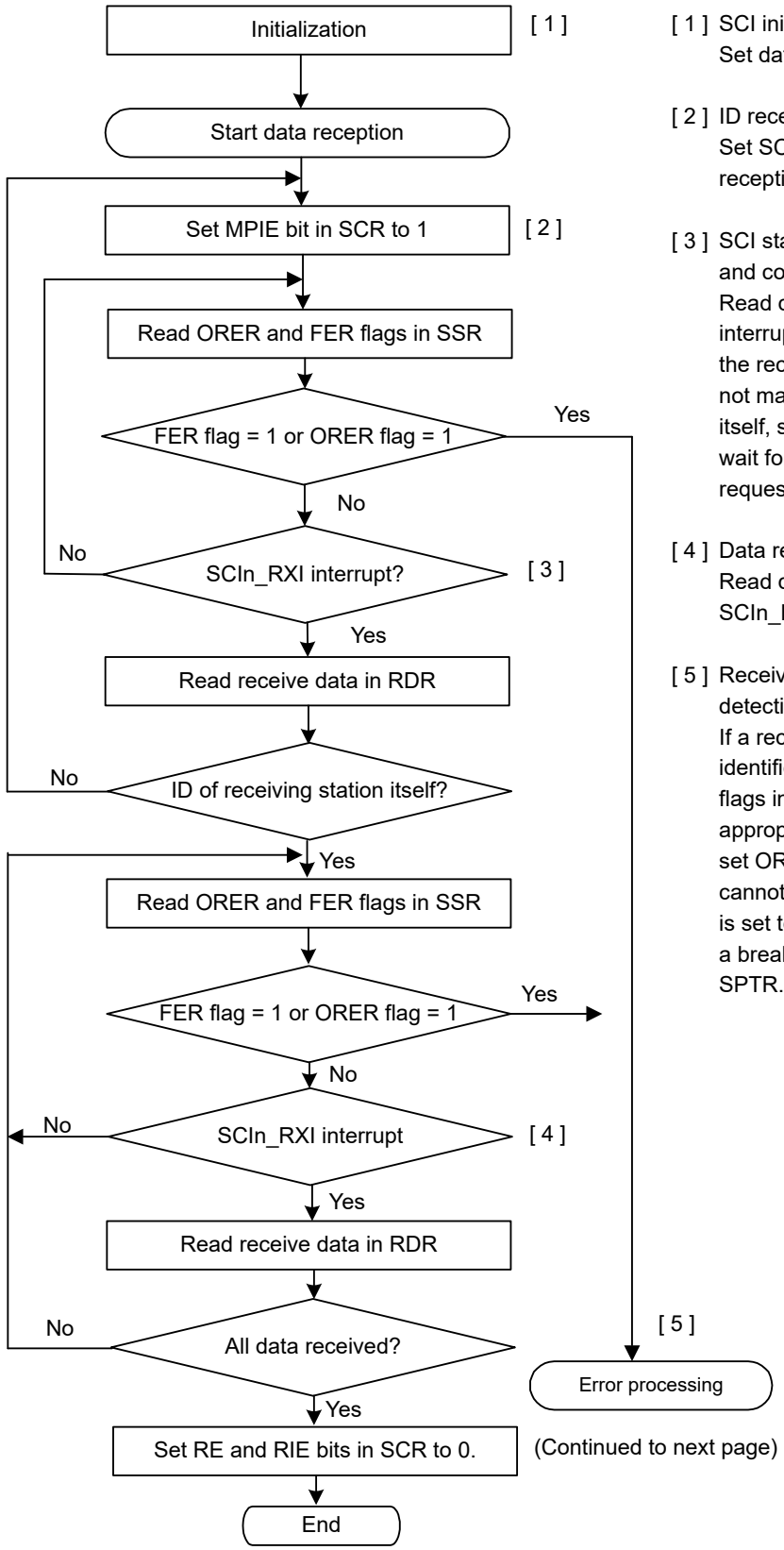
Figure 27.45 and Figure 27.46 are example flows of multi-processor serial reception. When the SCR.MPIE bit is set to 1, reading communication data is skipped until reception of communication data in which the multi-processor bit is set to 1. When communication data in which the multi-processor bit is set to 1 is received, the received data is transferred to the RDR register (the RDRHL register when 9-bit data length is selected), and the SCIn_RXI interrupt request is generated. The rest of the operations are the same as operations in asynchronous mode. Read the order from FRDRH to FRDRL.

Figure 27.44 shows an example operation for data reception.




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.44 Example of SCI reception with 8-bit data, multi-processor bit, and 1 stop bit**




[ 1 ] SCI initialization:
Set data reception.

[ 2 ] ID reception cycle:
Set SCR.MPIE bit to 1 and wait for ID reception.

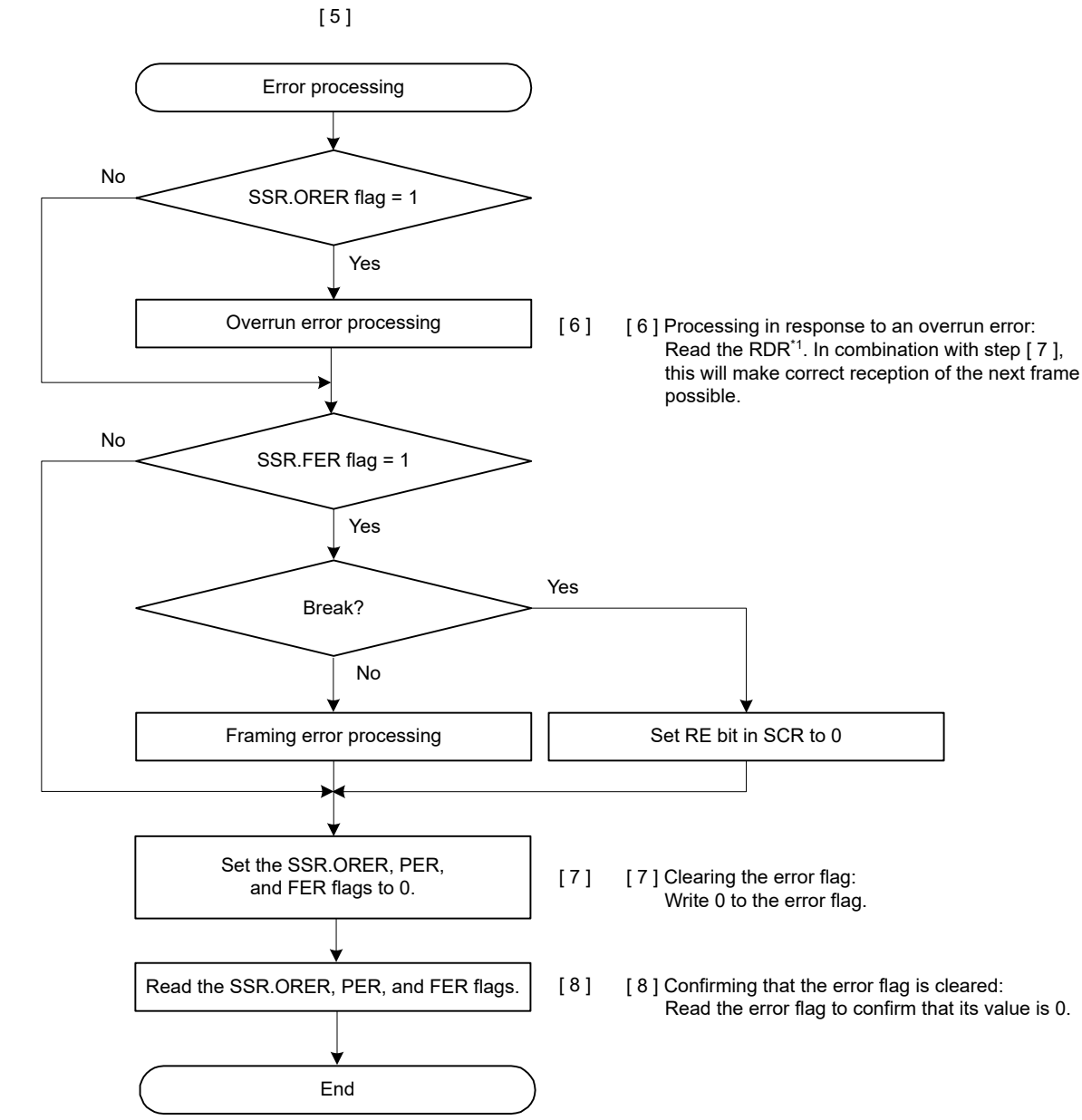
[ 3 ] SCI status confirmatIon and reception and comparison of ID:
Read data in RDR*1 at the first SCIn_RXI interrupt and compare it with the ID of the receiving statIon itself. If the ID does not match the ID of the receiving station itself, set the MPIE bit to 1 again, and wait for another SCIn_RXI interrupt request.

[ 4 ] Data reception at an SCIn_RXI interrupt:
Read data in RDR*1 once in the SCIn_RXI interrupt routine.

[ 5 ] Receive error processing and break detection:
If a receive error occurs, an error is identified by reading ORER and FER flags in SSR. After performing the appropriate error processing, be sure to set ORER and FER flags to 0. Reception cannot be resumed if any of these flags is set to 1. In the case of a framing error, a break can be detected by reading SPTR.RXDMON flag.

Note 1. The RDR register becomes the RDRHL register when 9-bit data length is selected.

**Figure 27.45 Example flow of multi-processor serial reception with non-FIFO selected (1)**




Note 1. The RDR register becomes the RDRHL register when 9-bit data length is selected.

**Figure 27.46 Example flow of multi-processor serial reception with non-FIFO selected (2)**

### (2) FIFO selected

Figure 27.47 shows an example of a data format that is written to FRDRH and FRDRL in multi-processor mode.

In multi-processor mode, the MPB value that is a part of the receive data is written to the FRDRH.MPB bit. A value of 0 is written to the FRDRH.PER flag. Data is written to FRDRH and FRDRL with the correct data length. Unused bits are written with 0. Read in order from FRDRH to FRDRL. When software reads the FRDRL register, the SCI updates FER, MPB, and receive data (RDAT[8:0]) in FRDRL with the next data. The RDF, ORER and DR flags in the FRDRH register always reflect the associated flags in the SSR_FIFO register.

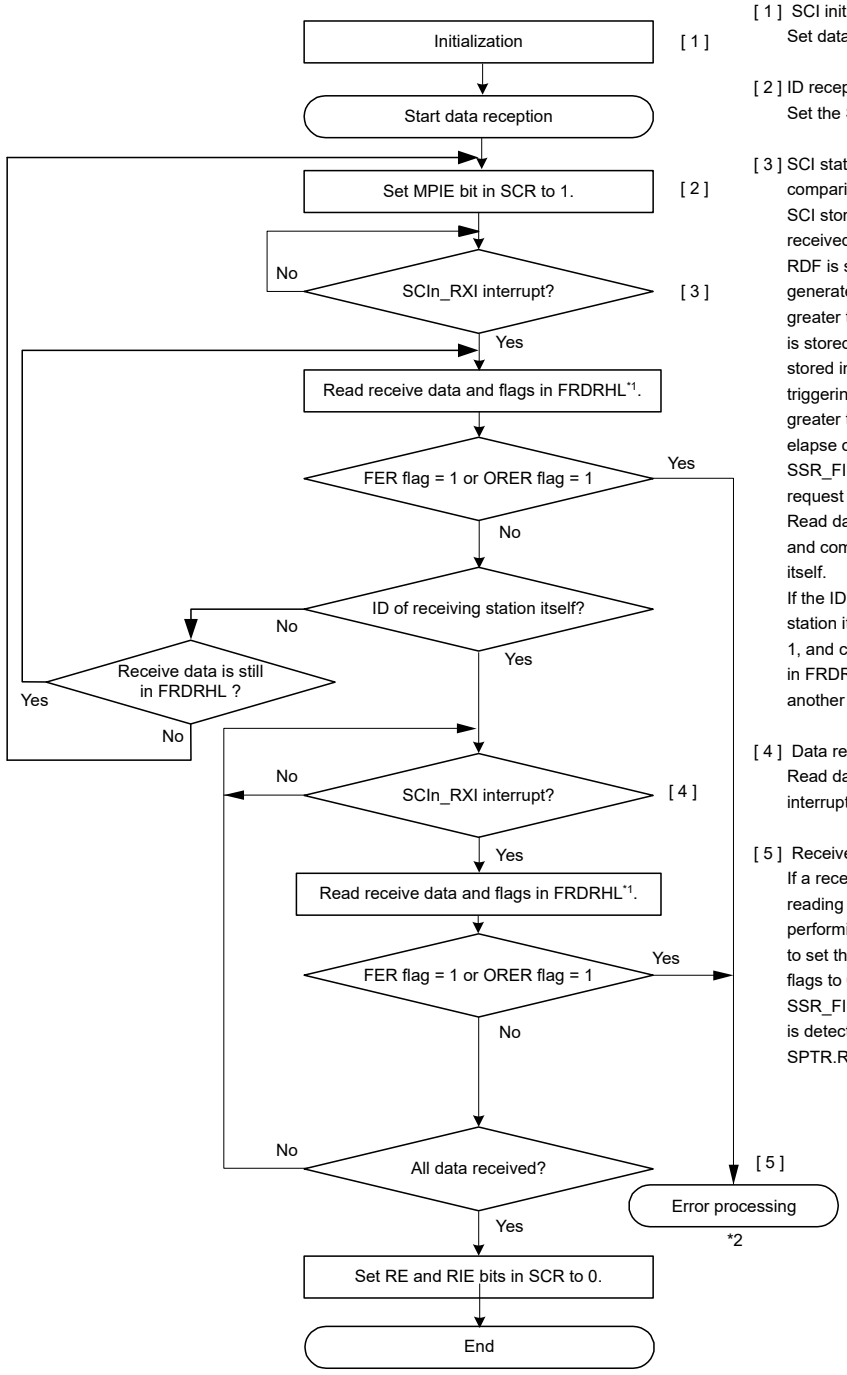
| Data Length | Register Setting | | Receive data in FRDRH, FRDRL | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FRDRHL | | | | | | | | | | | | | | | |
| | | | FRDRH | | | | | | | | FRDRL | | | | | | | |
| | SCMR.CHR1 | SMR.CHR | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
| 7 bits | 1 | 0 | — | RDF | ORER | FER | 0 | DR | MPB | 0 | 0 | 7-bit receive data | | | | | | |
| 8 bits | 1 | 1 | — | RDF | ORER | FER | 0 | DR | MPB | 0 | 8-bit receive data | | | | | | | |
| 9 bits | 0 | Don't care | — | RDF | ORER | FER | 0 | DR | MPB | 9-bit receive data | | | | | | | | |

Note: When data length is 7 bits, 0 is always read for FRDRH[0] and FRDRL[7]
When data length is 8 bits, 0 is always read for FRDRH[0]
FRDRHL[15] bit is read as an indefinite value

**Figure 27.47 Data format stored in FRDRH and FRDRL in multi-processor mode with FIFO selected**

Figure 27.48 shows an example flow of multi-processor data reception with FIFO selected. When the SCR.MPIE is set to 1, reading communication data is skipped until reception of communication data in which the multi-processor bit is set to 1. When communication data in which the multi-processor bit is set to 1 is received, the received data, MPB and associated errors are transferred to the FRDRHL register. The SCR.MPIE bit is automatically cleared and normal reception continues.

If a framing error occurs and the SSR_FIFO.FER flag is set to 1, the SCI continues data reception. The rest of the operations are the same as operations in asynchronous mode with FIFO selected.




[ 1 ] SCI initialization:
Set data reception.

[ 2 ] ID reception cycle:
Set the SCR.MPIE bit to 1 and wait for ID reception.

[ 3 ] SCI status confirmation and reception and comparison of ID:
SCI stores first data (MPB = 1) and subsequent received data in FRDRHL.
RDF is set to 1 and an SCIn_RXI interrupt request is generated when a quantity of receive data equal to or greater than the specified receive triggering number is stored in FRDRHL. When the quantity of data stored in FRDRHL falls the specified receive triggering number and received data is equal to or greater than 1, and no next data is received after the elapse of 15 etus from the last stop bit, SSR_FIFO.DR is set to 1. An SCIn_RXI interrupt request is generated when FCR.DRES bit is 0.
Read data in FRDRHL at the first SCIn_RXI interrupt, and compare it with the ID of the receiving station itself.
If the ID does not match the ID of the receiving station itself, the SCI reads until the data with MPB = 1, and compares next ID. If it is no data with MPB = 1 in FRDRHL, set the MPIE bit to 1 again and wait for another SCIn_RXI interrupt request.

[ 4 ] Data reception at an SCIn_RXI interrupt:
Read data in FRDRHL once in the SCIn_RXI interrupt routine.

[ 5 ] Receive error processing and break detection:
If a receive error occurs, an error is identified by reading the ORER and FER flags in SSR_FIFO. After performing the appropriate error processing, be sure to set the SSR_FIFO.ORER and SSR_FIFO.FER flags to 0. Reception cannot be resumed if the SSR_FIFO.ORER flag is set to 1. When framing error is detected, a break can be detected by reading the SPTR.RXDMON flag.

Note 1. If FRDRH and FRDRL are used instead of FRDRHL, read in order from FRDRH to FRDRL.
Note 2. Same as Figure 27.46

**Figure 27.48 Example flow of serial reception in multi-processor mode with FIFO selected**

## 27.5 Operation in Manchester mode

In Manchester mode, the transmit or receive serial data is coded in Manchester encoding.

Figure 27.49 shows the conceptual image of Manchester encoding.

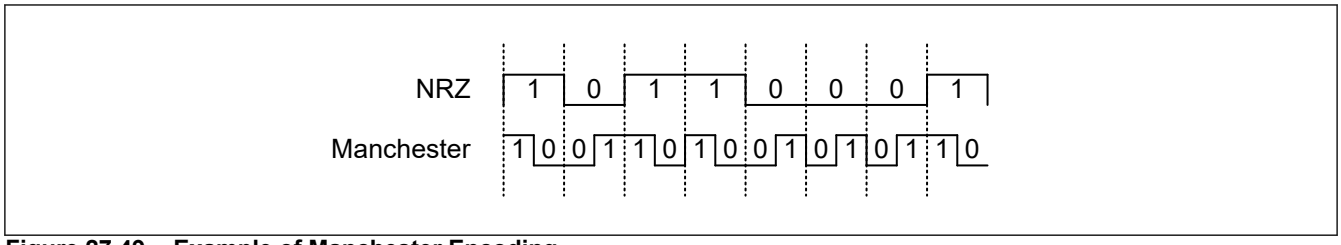



**Figure 27.49 Example of Manchester Encoding**

In Manchester mode, a preface and a start bit area are added to the transmit data in the register to configure a transmit frame. For transmission, data is encoded in Manchester encoding. When data is received, frames having the same format as transmitted frames are detected and Manchester decoding is performed.

For details on the frame format, see section 27.5.1. Frame Format.

## 27.5.1 Frame Format

Figure 27.50 shows the frame format in Manchester mode.

In the upper half of the figure, relevant setting registers are shown.

The preface area and the data area are encoded in Manchester encoding.

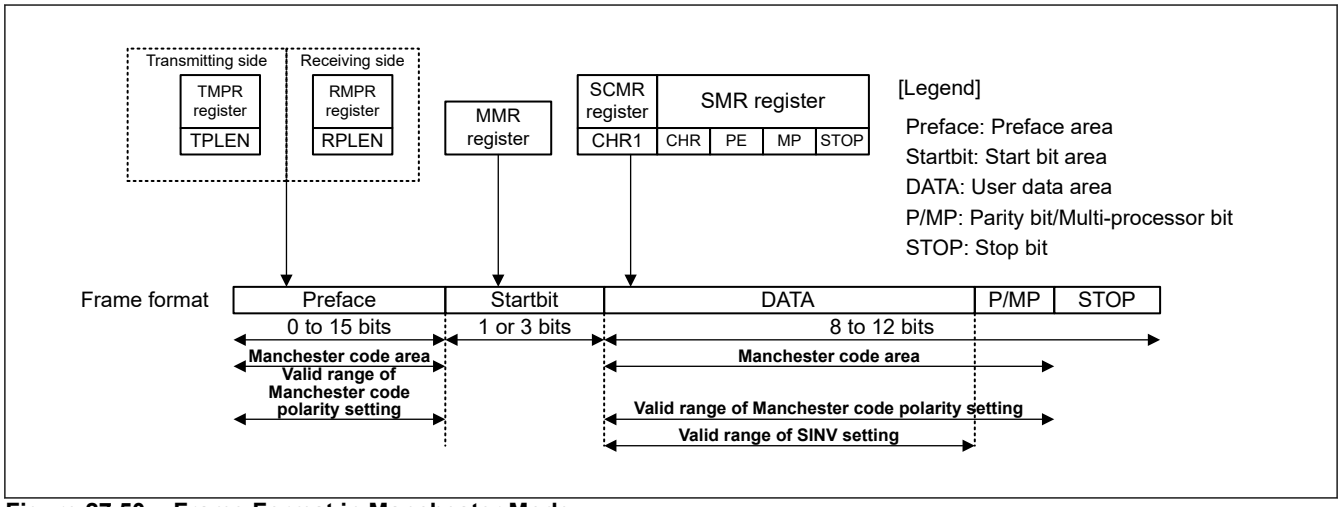



**Figure 27.50 Frame Format in Manchester Mode**

### (1) Preface area

This is a fixed pattern area located at the beginning of each frame.

Different registers are used to set the preface area for transmission and reception. The preface length is determined by setting TMPR.TPLEN[3:0] for transmission. It is determined by setting RMPR.RPLEN[3:0] for reception.

If it is set to 0, the transmit preface is disabled and is not added.

If it is set to 1d to 15d, a preface whose length is determined by this setting is added.

(For example, if it is set to 1d, a 1-bit preface is added. If it is set to 15d, a 15-bit preface is added.)

The preface pattern is set with TMPR.TPPAT[1:0] for transmission and RMPR.RPPAT[1:0] for reception, and is selected from four types of patterns.

Figure 27.51 shows how the preface pattern is set. The preface area and the start bit area are added for each communication frame.

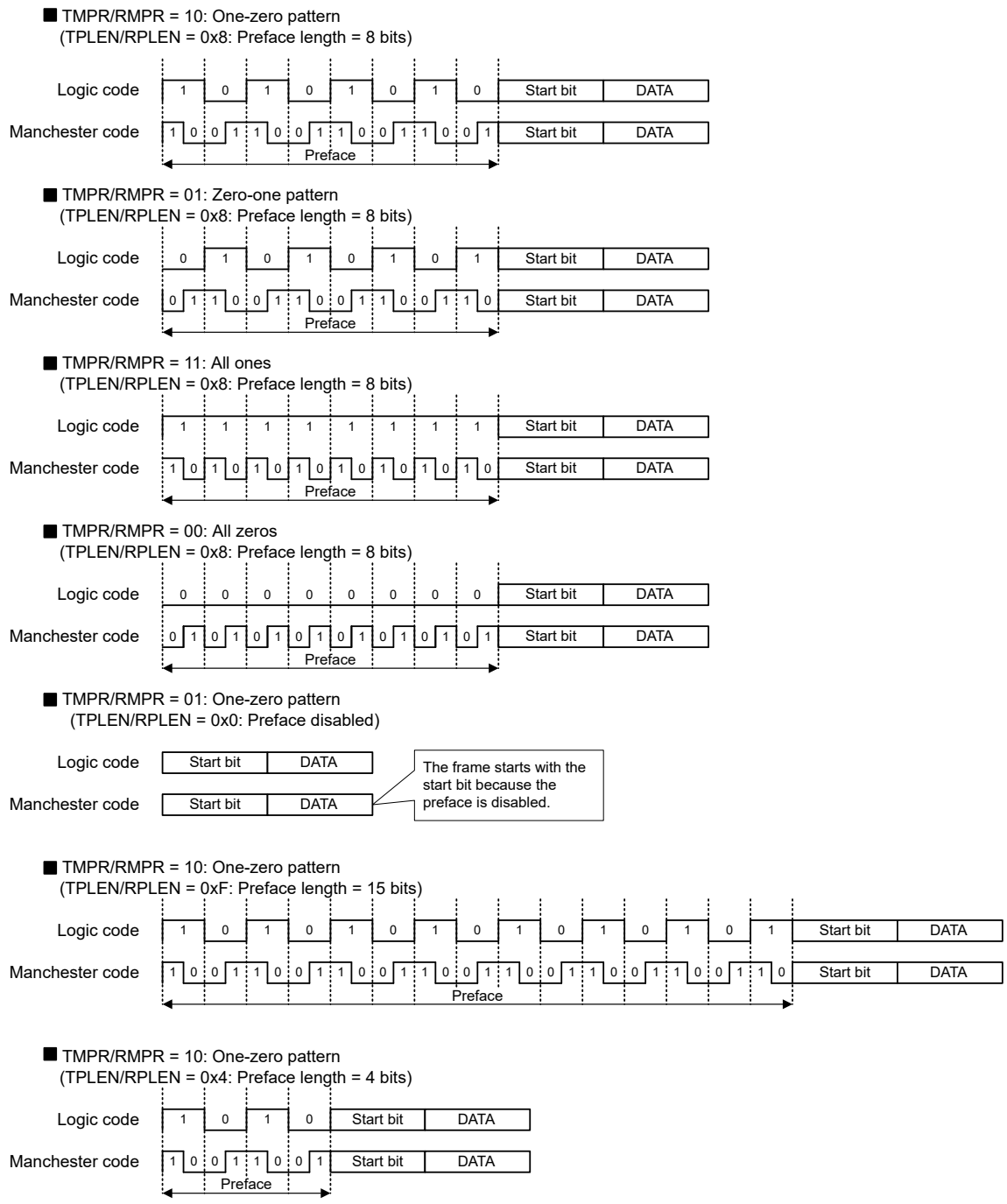



**Figure 27.51 Preface Pattern Setting Example**

## (2) Start bit area

This is an area indicating the start of valid data in a frame. It is added after the preface area.

The start bit length is determined by MMR.SBSEL setting. When MMR.SBSEL = 0, the start bit length is 1 bit.

When MMR.SBSEL = 1, the start bit length is 3 bits.

When MMR.SBSEL = 1, the SYNC type can be selected from command SYNC and data SYNC.

Command SYNC means the three start bits are added as a one-to-zero transition.

Data SYNC means the three start bits are added as a zero-to-one transition.

The SYNC type is determined by the MMR.SYNSEL, MMR.SYNVAL and TDRH_MAN.TSYNC settings.

(When receiving, the received result is applied to RDRH_MAN.RSYNC.)

When MMR.SBSEL = 0, the start bit is added as a zero-to-one or one-to-zero transition.

The selection is determined by the MMR.SYNVAL setting.

The MMR.SYNSEL bit specifies the destination to be referred to when setting for transmission.

When the MMR.SYNSEL bit is set to 1, the MMR.SYNVAL setting is referred to. When the MMR.SYNSEL bit is set to 0, the TDRH_MAN.TSYNC setting is referred to.

Figure 27.52 shows the state of the start bit area according to the settings in the MMR.SYNSEL, MMR.SYNVAL and TDRH_MAN.TSYNC registers in the case of transmission. Figure 27.53 shows that in the case of reception.

The start bit(s) is not affected by the MMR.TMPOL or MMR.RMPOL setting.

*d.c.=don't care, Preface=preface area, START=start bit area, Data=user data area,
P=parity bit, MP=multi-processor bit, STOP=stop bit

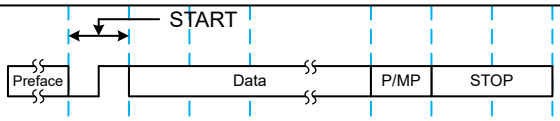
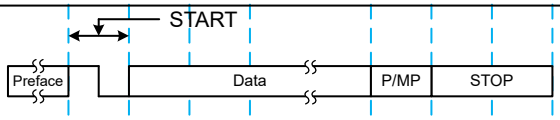
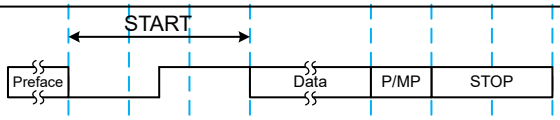
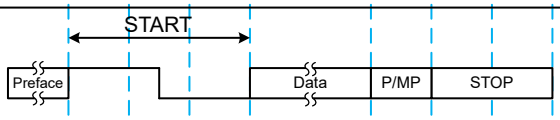
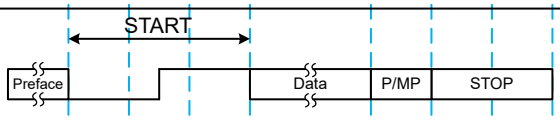
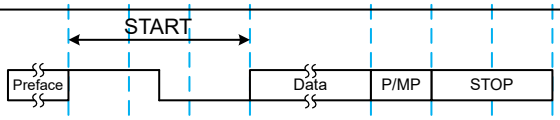
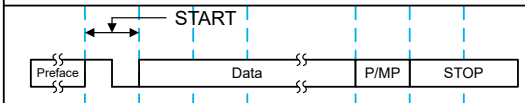
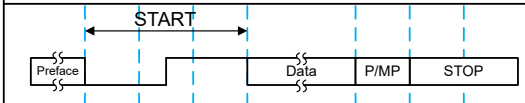
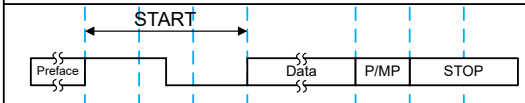
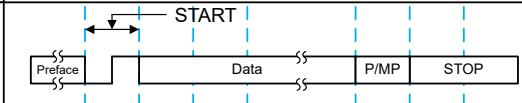
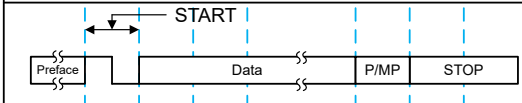
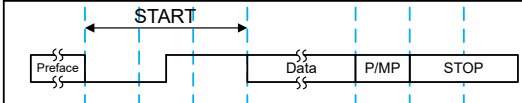
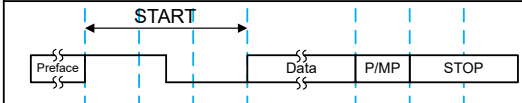
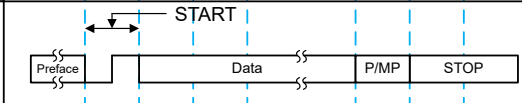
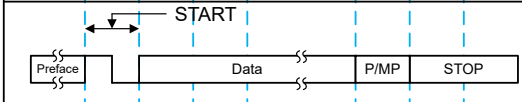
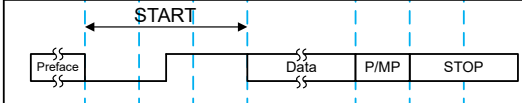
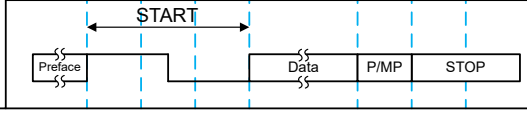
| Register setting | | | | Output signal | |
|---|---|---|---|---|---|
| MMR | | | TDRHL_MAN | Output TXDn waveform | Output start bit |
| SBSEL | SYNSEL | SYNVAL | TSYNC | | |
| 0 | d.c. | 0 | d.c. |  | 1bit<br>0 to 1 transition |
| 0 | d.c. | 1 | d.c. |  | 1bit<br>1->0transition |
| 1 | 0 | 0 | d.c. |  | 3bit<br>0to1 transition<br>(DATA SYNC) |
| 1 | 0 | 1 | d.c. |  | 3bit<br>1to0 transition<br>(COMMAND SYNC) |
| 1 | 1 | d.c. | 0 |  | 3bit<br>0to1 transition<br>(DATA SYNC) |
| 1 | 1 | d.c. | 1 |  | 3bit<br>1to0 transition<br>(COMMAND SYNC) |

**Figure 27.52 Settings Related to and Format of the Start Bit Area at Transmission**

When the start bit area is 3 bits long, SYNVAL is not referenced.

d.c. = don't care, Preface = Preface area, START = Start bit area, Data = Data area
P = Parity bit, MP = Multi-processor bit, STOP = Stop bit

| Register setting | | | | Input signal | Start bit detection result*1 | Register indication |
|---|---|---|---|---|---|---|
| MMR | | | TDRHL_MAN | RXDn input waveform | | RDRHL_MAN.RSYNC |
| SBSEL | SYNSEL | SYNVAL | TSYNC | | | |
| 0 | d.c. | 0 | d.c. | | Normal start bit (1 bit: 0-to-1 transition) | 0 |
| | | | |   | Start bit error | 0 |
| | | | |   | Start bit error | 0 |
| | | | |   | Start bit error | 0 |
| 0 | d.c. | 1 | d.c. |   | Start bit error | 0 |
| | | | |   | Normal start bit (1 bit: 1-to-0 transition) | 0 |
| | | | |   | Start bit error | 0 |
| | | | |   | Start bit error | 0 |
| 1 | d.c. | d.c. | d.c. |   | Start bit error | 0 |
| | | | |   | Start bit error | 0 |
| | | | |   | Data SYNC | 0 |
| | | | |   | Command SYNC | 1 |

Note 1. Data other than the start bit is assumed to be normal.

**Figure 27.53 Settings Related to and Judgment of the Start Bit Area at Reception**

## (3) DATA

Since the format of the data area is the same as that of the asynchronous mode, see section 27.3.1. Serial Data Transfer Format.

As shown in Figure 27.49, Frame Format in Manchester Mode, the stop bit is not included in the Manchester encoding range.

### 27.5.2 Clock

As the transfer clock in Manchester mode, the clock generated by the on-chip baud rate generator is used by setting the SMR.CKS[1:0] bit.

Also it is possible to set the oversampling (transfer rate of one-bit period) by SEMR.ABCS bit.

When the SMER.ABCS bit is set to 0, oversampling x16 is selected with the one-bit period being 16 cycles of the base clock. When the SMER.ABCS bit is set to 1, oversampling x8 is selected with the one-bit period being 8 cycles of the base clock.

### 27.5.3 Initialization of the SCI in Manchester Mode

Before transferring data, write the initial value (0x00) to the SCR register and initialize the SCI following the example of flowchart shown in Figure 27.54.

Whenever the operating mode or transfer format is changed, the SCR register must be initialized before the change is made.

Note that setting the SCR.RE bit to 0 initializes none of the ORER, FER, PER, MER, and RDRF flags in the SSR_MANC register, the SYER, PFER and SBER flags in the MESR register, and the RDR, RDRHL_MAN registers.

Note also that switching the value of SCR.TE from 0 to 1 when SCR.TIE is 1 generates a SCIn_TXI interrupt request.

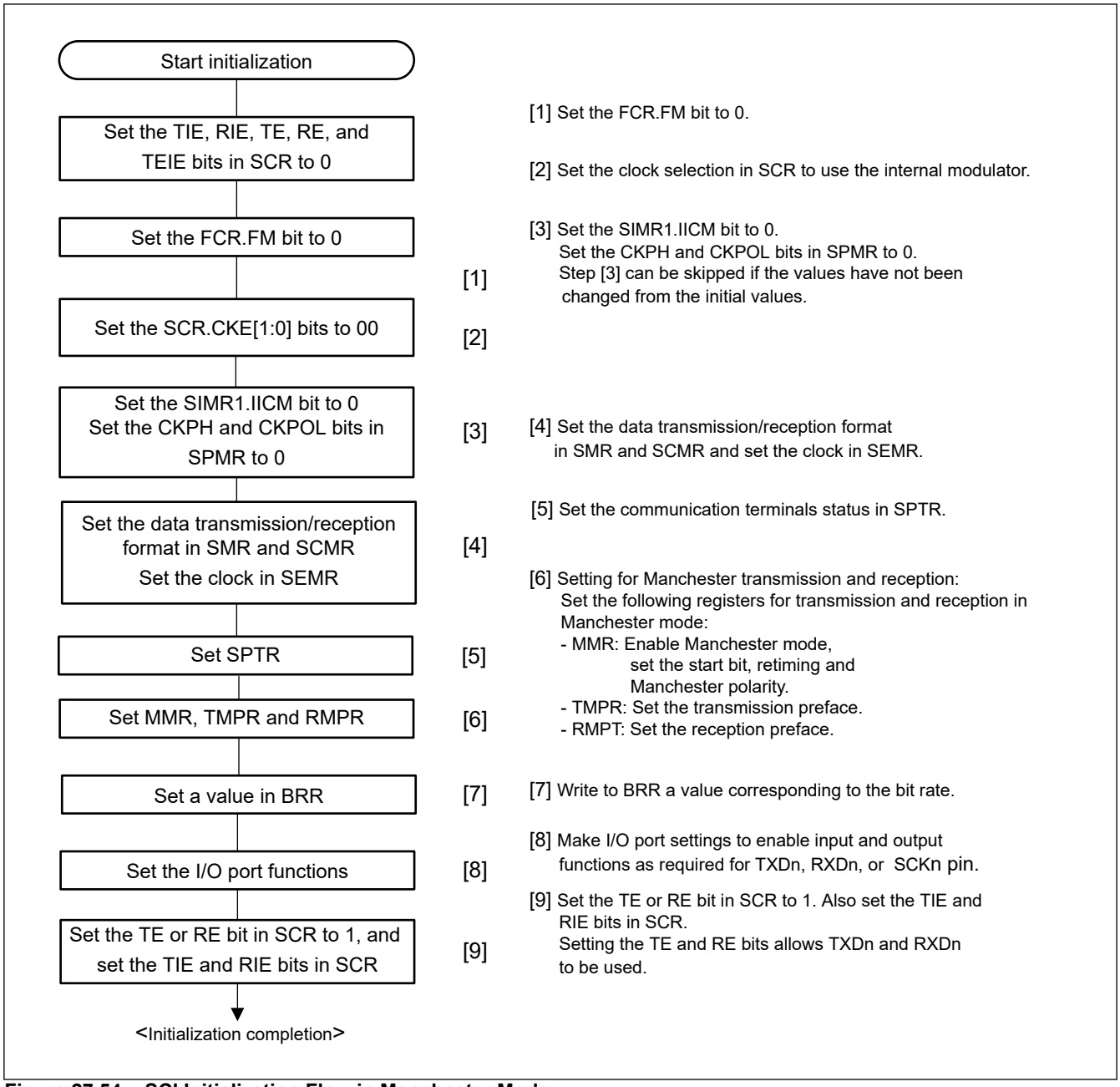



**Figure 27.54 SCI Initialization Flow in Manchester Mode**

## 27.5.4 Double-speed operation

When the ABCS bit in SEMR is set to 1 and eight pulses of the base clock for a 1-bit period is selected, the SCI operates on the bit rate twice that of when ABCS is set to 0.

When the BGDM bit in SEMR is set to 1, the cycle of the base clock is reduced to half and the SCI operates on the bit rate twice that of when ABCS is set to 0.

When the ABCS and the BGDM bits in SEMR are set to 1, the SCI operates on the bit rate four times that of when the ABCS and the BGDM bits in SEMR are set to 0.

## 27.5.5 CTS and RTS functions

The CTS function uses input on the CTSn_RTSn pin in transmission control. Setting the CTSE bit in SPMR to 1 enables the CTS function. The CTSn_RTSn pin can be set as a multiplexed pin which allows one pin to be used for either function, or as dedicated pins with each pin for a single function. Use the CTSPEN bit in SPMR for this setting.

When the CTS function is enabled, reception starts only when the CTSn_RTSn pin is at the low level.

Applying a high level to the CTSn_RTSn pin after transmission starts does not affect transmission of the current frame, which continues.

The RTS function uses output on the CTSn_RTSn pin to request transmission. When the SCI is ready to receive, it outputs a low level to the CTSn_RTSn pin, Conditions for output of the low level and high level are as follows:

[Conditions for low-level output]

When all conditions listed below are satisfied:

- The value of the RE bit in SCR is 1.
- The SCI is ready to receive.
- There is no received data yet to be read.
- All of the following flags are set to 0: SSR_MANC.ORER, FER, PER and MER flags, and MESR.SYER (when SYEREN = 1), PFER (when PFEREN = 1) and SBER flags (when SBEREN = 1).

[Conditions for high-level output]

- When the conditions for low output are not satisfied

### 27.5.6 Serial data transmission in Manchester mode

The SCI encodes data in Manchester encoding and sends the resultant data in Manchester mode.

When the polarity setting (MMR.TMPOL) set to 0, logic 0 is coded as a zero-to-one transition in Manchester code and logic 1 is coded as a one-to-zero transition in Manchester code.

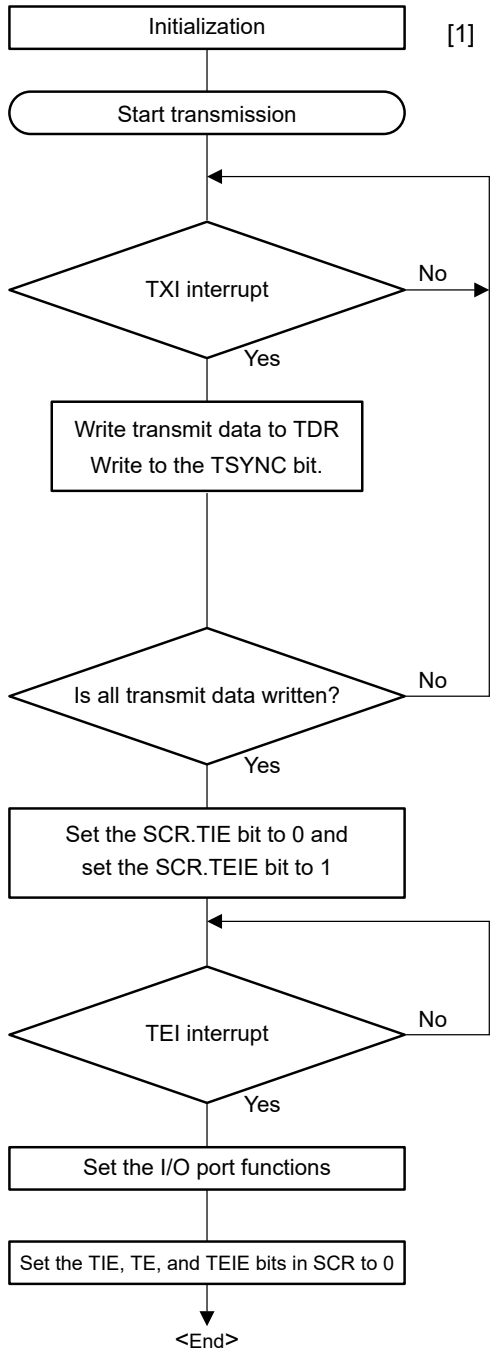
When the polarity setting (MMR.TMPOL) set to 1, logic 0 is coded as a one-to-zero transition in Manchester code and logic 1 is coded as a zero-to-one transition in Manchester code.

For this reason, a level transition occurs with the Manchester encoded data in the middle of individual logic data. (See Figure 27.49).

The transmitter constructs transmit frames in a specific format by adding a preface area to data and setting the start bit(s) according to the polarity setting and sends resultant serial data.

For details on the frame format, see section 27.5.1. Frame Format.

Figure 27.55 shows the flowchart in transmission. Figure 27.56, Figure 27.57, and Figure 27.58 show examples of the operation for serial transmission in Manchester mode.



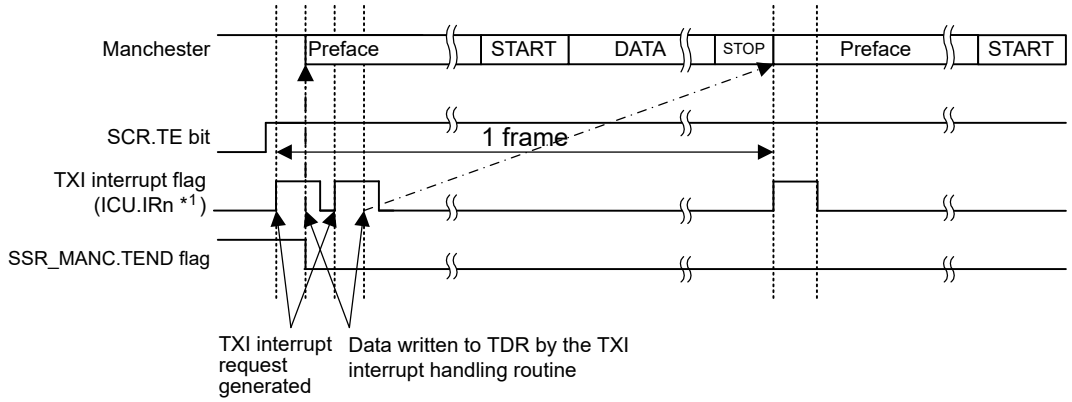

[1] SCI Initialization:
Set data transmission.
Setting the SCR.TE bit to 1 enables transmission.

[2] TXI interrupt:
When transmit data is transferred from TDR to TSR, a transmit data empty interrupt (TXI) request is generated.

[3] TDR setting:
Write the transmit data to TDR.
When TSYNC is used, also set TDRHL_MAN.TYSNC.

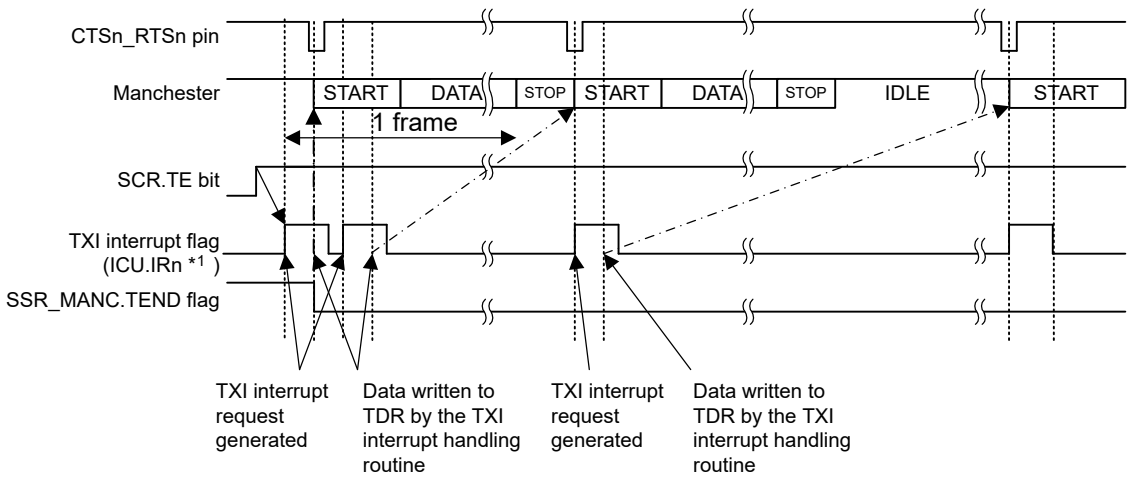
[4] Serial transmission continuation procedure:
To continue serial transmission, write the transmit data to TDR once by using an TXI interrupt.
The transmit data can also be written to TDR by activating the DMAC or DTC in the TXI interrupt request.
When TEI interrupt requests are used, set the SCR.TIE bit to 0 and the SCR.TEIE bit to 1 after the last of the data to be transmitted is written to TDR.

**Figure 27.55 Example of Serial Transmission Flowchart in Manchester Mode**



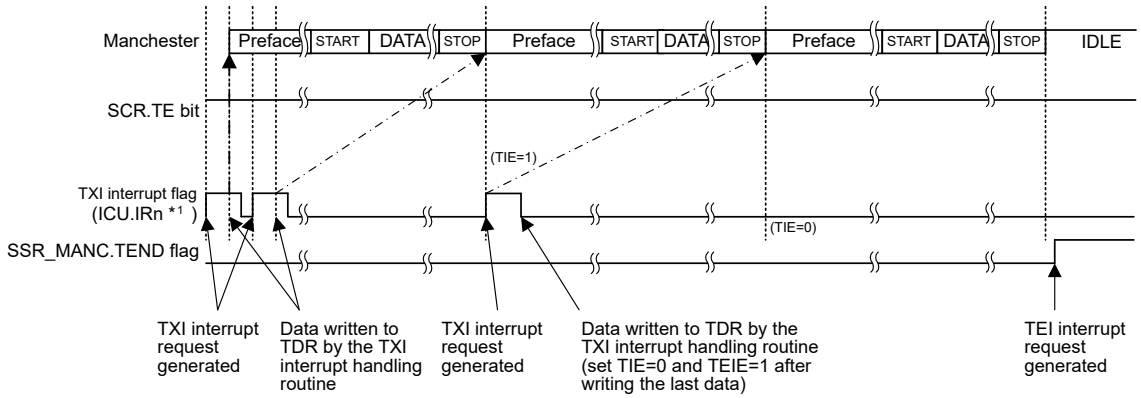

Note 1. For corresponding interrupt numbers, see section 13, Interrupt Controller Unit (ICU)

**Figure 27.56 Example of Start-of-Transmission Operation for Serial Transmission in Manchester mode (with Preface but Without the CTS Function)**




Note 1. For corresponding interrupt numbers, see section 13, Interrupt Controller Unit (ICU)

**Figure 27.57 Example of Start-of-Transmission Operation for Serial Transmission in Manchester Mode (Without Preface but with the CTS Function)**




Note 1. For corresponding interrupt numbers, see section 13, Interrupt Controller Unit (ICU)

**Figure 27.58 Example of End-of-Transmission Operation for Serial Transmission in Manchester Mode (with Preface but Without the CTS Function)**

## 27.5.7 Serial Data Reception in Manchester Mode

In Manchester mode, the SCI operates on a base clock with a frequency of 16 times*1 the bit rate. Reception starts by sampling the falling edges of received data at the base clock. As shown in Figure 27.59, reception starts at a falling edge of the received data and it continues if the received data keeps low for the duration of 1/4 bit. If the received data goes high within the duration of 1/4 bit, the SCI judges it as an error and waits for a falling edge again.

If a high level is expected in the first half of a bit in the received data, the SCI judges a low level that continues for one base clock cycle as an error and ignores the change to the low level.

Note 1. This is the case when SEMR.ABCS = 0. When SEMR.ABCS = 1, the SCI operates on a base clock with a frequency of 8 times the bit rate.

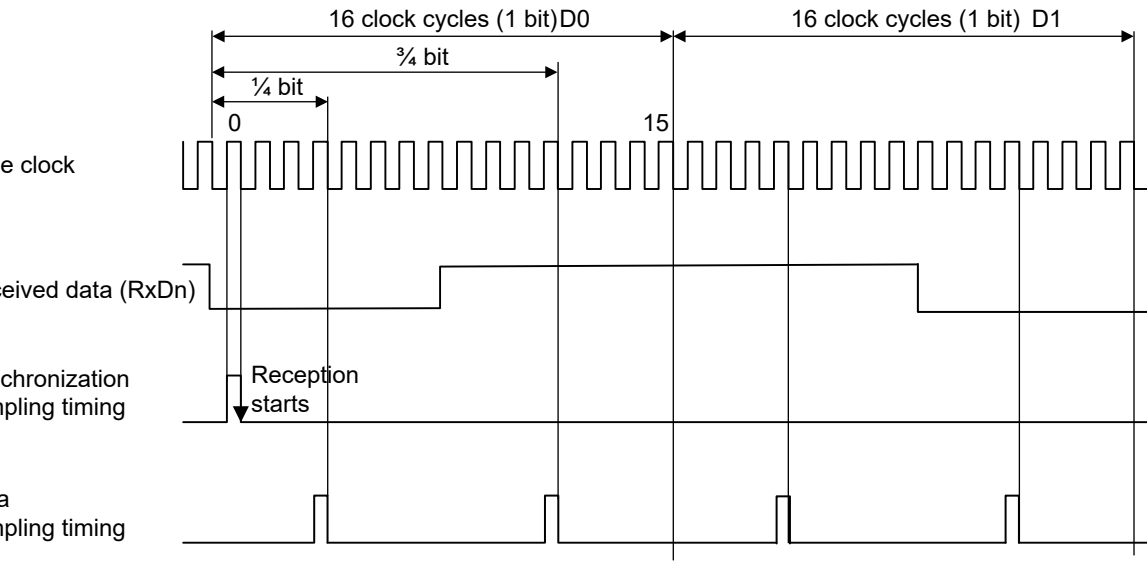



**Figure 27.59 Data Reception Sampling Timing in Manchester Mode**

In Manchester mode, data reception starts with detection of a preface and start bit area.

The SCI checks the input from the RXDn pin to see whether a preface is added based on the value of RMPR.RPLEN.

If the preface is disabled (RMPR.RPLEN = 0), it moves on to the detection of a start bit area without detecting a preface.

When a preface is enabled, it identifies a preface pattern setting according to the set value in RMPR.RPPAT, and compares it with the RXDn input for a pattern match to detect a preface pattern.

Upon detection of a preface pattern match, it judges it as a normal preface and moves on to the detection of a start bit area.

If detecting a preface pattern mismatch or a Manchester code error in the preface area, it judges it as a preface error and asserts a preface error (PFER).

For start bit detection, the SCI selects an expected value based on the register settings (MMR.SBSEL and SYNVAL), compares it with the RXDn input for a pattern match to detect a start bit area. Upon detection of a start bit pattern match, it judges it as a normal start bit area and moves on to the data processing.

Only when a preface and a start bit area are detected normally, it moves on to the next phase of data reception.

Upon detection of a start bit pattern mismatch, it asserts a start bit error flag (SBER).

In data processing, the SCI shifts the data by the expected received data length based on the register settings (SCMR.CHR1 and SMR.CHR) through the RSR register. If two sampling points in a bit of the received data are identical, the SCI judges this as a Manchester code error.

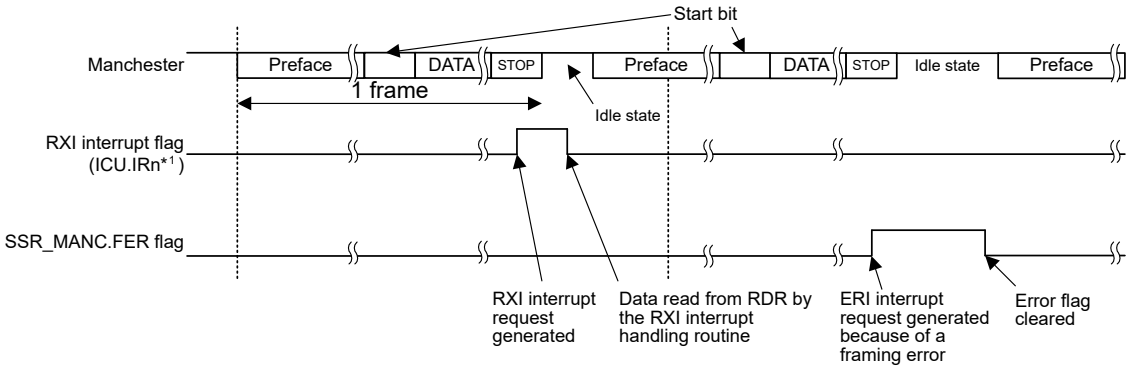
For details, see section 27.5.11. Errors in Manchester Mode (4).

When the parity function is disabled (SMR.PE = 0), the SCI moves on to the next phase of stop bit detection. When the parity function is enabled (SMR.PE = 1), the SCI performs parity checking. If detecting a parity error, it asserts a parity error flag (PER), and then moves on to stop bit detection.

In stop bit detection, the SCI checks the following in the stop bit area of the received frame:

It has two sampling points in a bit. If both points are at the high level, the bit is recognized as a normal stop bit and the data is stored in the RDR register. At least one low-level point is judged as an abnormal stop bit, causing a framing error flag (FER) to be set. Even when an error is detected, the received data is stored in the RDR register as abnormal data.

Figure 27.60 shows an example of the operation for serial data reception in Manchester mode.




Note 1. For corresponding interrupt numbers, see section 13, Interrupt Controller Unit (ICU).

**Figure 27.60 Example of Operation for Serial Data Reception in Manchester mode (with a Preface)**

For the state of each status flag in the SSR_MANC register and RXDn input processing when a receive error is detected, see section 27.5.11. Errors in Manchester Mode.

If a receive error is detected, an SCIn_ERI interrupt request is generated but an SCIn_RXI interrupt request is not generated.

Data reception cannot be resumed while the receive error flag is 1. Accordingly, set the ORER, FER, PER, MER, SYER*1, PFER*1, and SBER*1 flags to 0 before resuming reception. Also, be sure to read the RDR (or RDRHL_MAN) register during overrun error processing. When a reception is forcibly terminated by setting the SCR.RE bit to 0 during operation, read the RDR (or the RDRHL_MAN) register because received data which has not yet been read may be left in the RDR (or the RDRHL_MAN) register.

Figure 27.61 and Figure 27.62 show examples of serial data reception flowchart in Manchester mode.

Note 1. Effective when the corresponding bit is enabled.

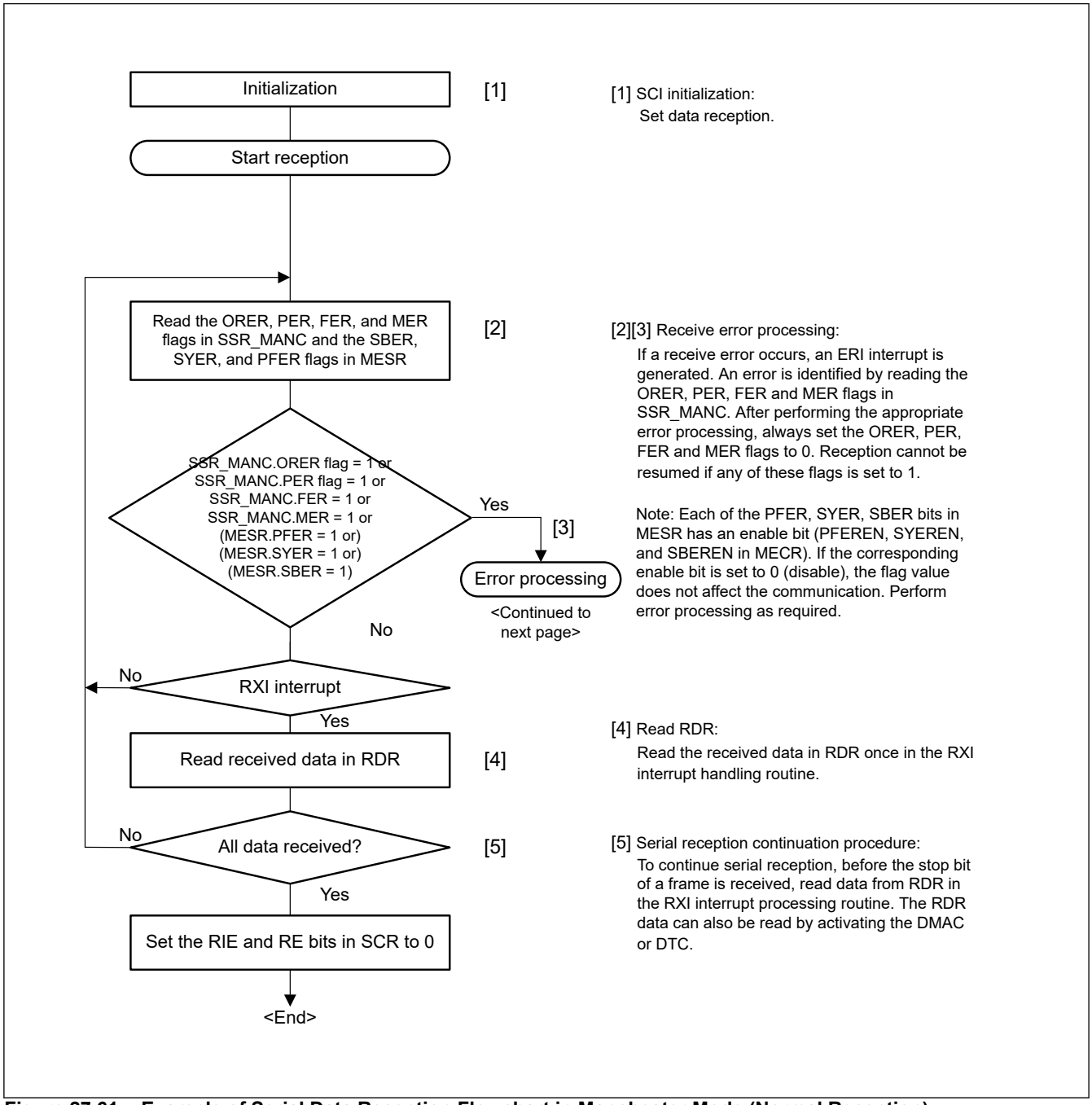



**Figure 27.61 Example of Serial Data Reception Flowchart in Manchester Mode (Normal Reception)**

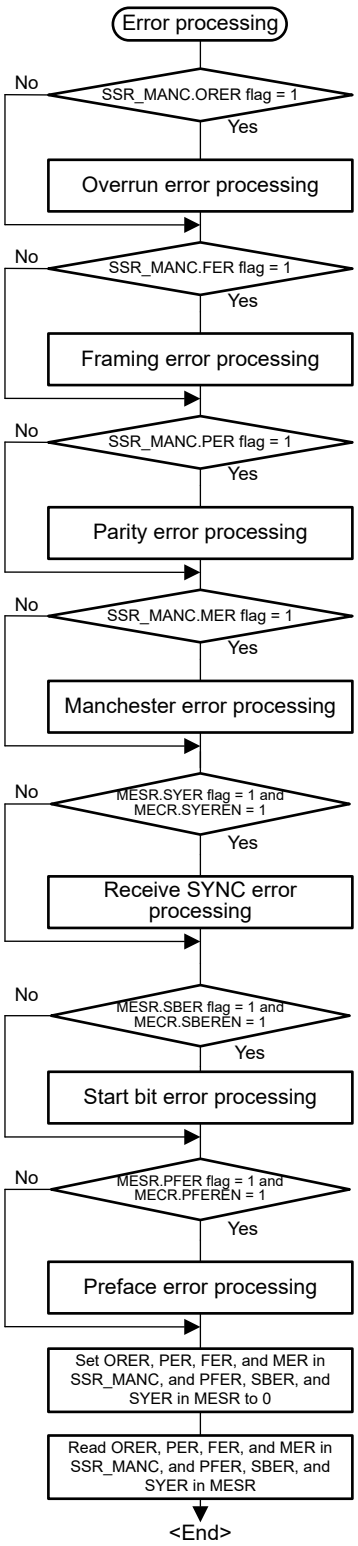



**Figure 27.62 Example of Serial Reception Flowchart in Manchester Mode (Error Processing)**

### 27.5.8 Operation When Multi-Processor Bit Is Used

See section 27.4. Multi-Processor Communication Function (1) for the operation in Manchester mode when using multi-processor mode because the operation is the same.

A preface and a start bit area are added to the frame format in Manchester mode. See Figure 27.62 for error processing in Manchester mode for the reception flowchart (Figure 27.46). See Table 27.35 for the operation status when detecting various errors.

### 27.5.9 Receive Retiming

This function corrects the timing for each central edge of the bit, taking advantage of the fact that each bit has an edge in the center in Manchester code.

The receive retiming function can be turned on or off by setting the ERTEN bit in the MMR register.

When the receive retiming function is turned off (MMR.ERTEN = 0), retiming is not performed, causing misalignment between the internal clock and the RXDn input to be accumulated and the receive margin to be reduced.

When the receive retiming function is turned on (MMR.ERTEN = 1), retiming is performed for the preface area, the start bit area*1, and the data area (excluding the stop bit).

Note 1. Retiming is not performed for the start bit area if the preface length is 0 and the start bit length is 3.

As an example, the receive retiming when oversampling x16 is selected is shown below.

When detecting an RXDn input edge two to four cycles before the expected receive cycle, the receive processing is shortened by one sampling CLK cycle.

When detecting a RXDn input edge two to three cycles after the expected receive cycle, the receive processing is extended by one sampling CLK cycle.

(Even if the clock is misaligned with the data by more than two cycles, one cycle is corrected for each bit.)

Figure 27.63 shows the conceptual image of receive retiming range.

When detecting an edge in the tolerance area in the figure, data is received as is without making correction.

When detecting an edge in the SyncJump area in the figure, data is corrected for reception.

When detecting an edge in the SyncError area in the figure, data is received as abnormal data with no correction made.

For a Manchester code error (data matches at the 1/4-phase and 3/4-phase sampling points), the SCI reports a code error.

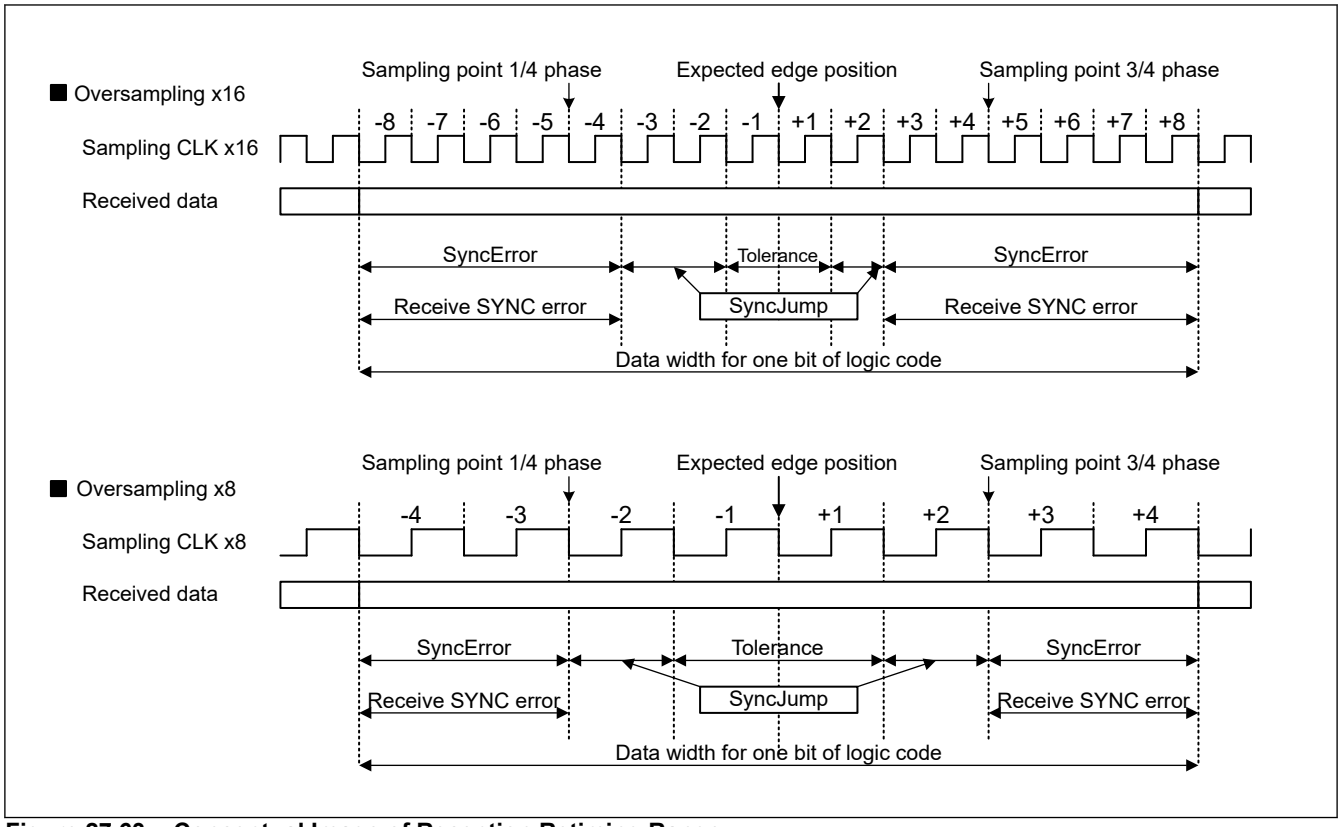



**Figure 27.63 Conceptual Image of Reception Retiming Range**

## 27.5.10 Polarity Setting for Manchester Code

The polarity of the Manchester code can be set with the Manchester Mode Register (MMR).

It can be set separately for transmission and reception. Use the MMR.TMPOL bit to set the polarity for transmission and the MMR.RMPOL bit to set the polarity for reception.

The Manchester code polarity setting is valid for the preface area, the data area, and the parity or multi-processor area.

When the initial settings (TMPOL/RMPOL = 0) are used for the polarity of Manchester code, logic 0 is encoded as a zero-to-one transition in Manchester code and logic 1 is encoded as a one-to-zero transition in Manchester code. If the settings are changed to TMPOL/RMPOL = 1, logic 0 is encoded as a one-to-zero transition in Manchester code and logic 1 is encoded as a zero-to-one transition in Manchester code. Figure 27.64 shows the conceptual image of the settings and operation.

Separately from the function above, the transmitted and received data in the data area can be inverted by the transmitted/received data inversion function (SCMR.SINV). Since the polarity of Manchester code (MMR.TMPOL/RMPOL) can be set separately from the transmitted/received data invert function (SCMR.SINV), if both are set to inversion (MMR.TMPOL/RMPOL = 1 and SCMR.SINV = 1), the transmitted and received data are set to initial state (inversion + inversion = normal).

The polarity of the start bit area can be set by a register different from the ones mentioned above.

Since a different register is used, the polarity of the start bit area is not affected by the polarity setting for Manchester code mentioned above.

For details on the setting for the start bit area, see section 27.5.1. Frame Format (2).

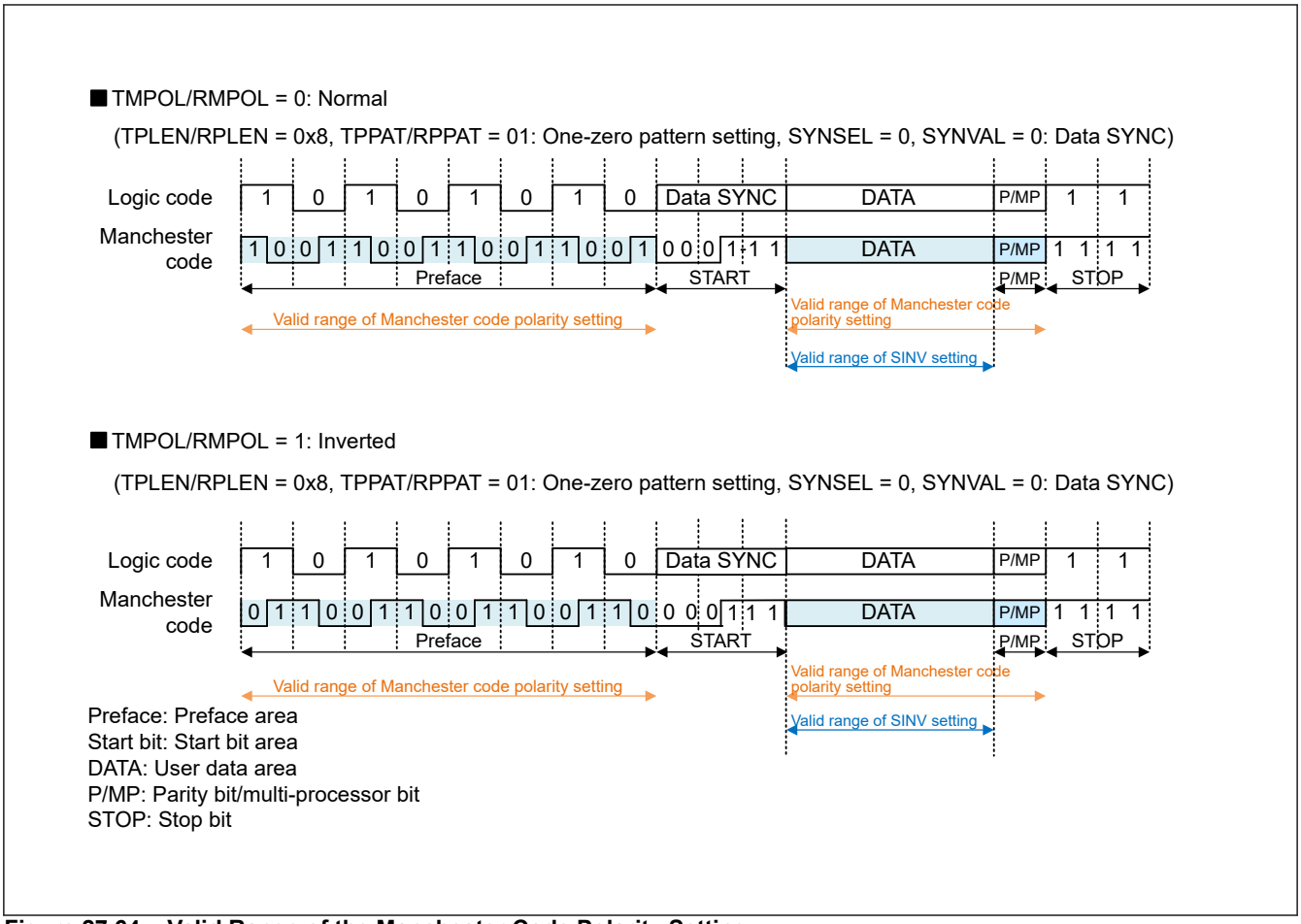



**Figure 27.64 Valid Range of the Manchester Code Polarity Setting**

## 27.5.11 Errors in Manchester Mode

There are the following errors in Manchester mode:

1. Parity error
2. Over run error
3. Framing error
4. Manchester error
5. Preface error
6. Start Bit error
7. Receive SYNC error

For errors (1) to (3), see section 27.3.9. Serial Data Reception in Asynchronous Mode (1) because they are the same as in asynchronous mode.

Each errors are judged in each area, but they are reflected on flags and operations at the timing of 3/4-bit sampling of the STOP bit area. If a preface error or start bit error is detected, subsequent data will not be received. Therefore, no other error detection is performed, and the error flag holds the previous information.

Table 27.33 lists the states of the serial status register when detecting errors and judgment about whether to store data in the RDR.

Table 27.34 lists the errors that can be detected in each area of a Manchester frame.

If a Preface error or Start bit error is detected, subsequent data will not be received. Therefore, no other error detection is performed, and the error flag holds the result of the previous frame reception. Also, if an error is detected in the previous frame, data will not be received, but errors in the pre-face area and start bit area will update that flag. Table 27.35 shows the flags and actions in this case.

(4) Manchester error

A Manchester error is generated when a Manchester code error is detected.

In Manchester code, there must be an edge (transition) in the center of the bit.

In the data area of a received frame (including the parity/multi-processor bit), the values of the 1/4-bit and 3/4-bit sampling points are checked in each received 1-bit data, and a Manchester code error is determined if these two values match.

If a Manchester code error is detected, the Manchester error flag (SSR_MANC.MER) is asserted.

If a Manchester error occurs, it is handled as an interrupt source and event source. If a Manchester error is detected, the next reception is not performed until the corresponding error flag is cleared.

(5) Preface error

A preface error is generated when the preface pattern does not match or a Manchester code error is detected in the preface area. If a preface error is detected, the preface error flag (SSR_MANC.PFER) is asserted.

It is possible to set whether to use this error flag as an interrupt source with the setting of the MECR register.

When MECR.PFEREN = 1, a preface error is handled as an interrupt source or event source. If a preface error is detected, the next reception is not performed until the corresponding error flag is cleared.

When MECR.PFEREN = 0, a preface error is not handled as an interrupt source or event source, and the next reception is not halted. However, a preface error is notified to MESR.PFER.

(6) Start bit error

A start bit error is generated when a mismatch is detected between the start bit area in the received frame and the preset start bit pattern. Upon detection of a start bit error, a start bit error flag (MESR.SBER) is asserted.

It is possible to set whether to use the start bit error as an interrupt source with the setting of the MECR register.

When MECR.SBEREN = 1, a start bit error is handled as an interrupt source or event source. If a start bit error is detected, the next reception is not performed until the corresponding error flag is cleared.

When MECR.SBEREN = 0, a start bit error is not handled as an interrupt source or event source, and the next reception is not halted. However, a start bit error is notified to MESR.SBER.

(7) Receive SYNC error

When the receive retiming function described in section 27.5.9. Receive Retiming is enabled, the receive retiming operation is performed.

If no edges are detected within the receive retiming range (SyncError area in Figure 27.63) when receive timing operation is being performed, a receive SYNC error is generated. Upon detection of a receive SYNC error, a receive SYNC error flag (MESR.SYER) is asserted. In areas not subject to retiming, receive SYNC errors are not detected.

The preface area[*1], the start bit area[*1,*2], and the data area (excluding the stop bit) for which receive retiming operation is performed are checked.

It is possible to set whether to use the receive SYNC error as an interrupt source with the setting of the MECR register.

When MECR.SYEREN = 1, a receive SYNC error is handled as an interrupt source or event source. If a receive SYNC error is detected, the next reception is not performed until the corresponding error flag is cleared.

When MECR.SYEREN = 0, a receive SYNC error is not handled as an interrupt source or event source, and the next reception is not halted. However, a receive SYNC error is notified to MESR.SYER.

Note 1. In the case of a frame that starts with a pattern that expects the first half of the bit to be High, it is excluded from retiming.

Note 2. In the start bit area, when there is no preface length and 3 bit start bit is set, it is not subject to retiming.

Also, the 1st bit and the 2nd bit in the start bit area when 3 bit start bit is set are not subject to retiming.

**Table 27.33 Flags in the SSR_MANC Register and Receive Data Handling in Manchester Mode (1 of 2)**

| Flag in the SSR_MANC register | | | | Flag in the MESR register | | | received data | Received error status (ERI interrupt / event generation) |
|---|---|---|---|---|---|---|---|---|
| ORER | FER | PER | MER | SBER[*1] | PFER[*1] | SYER | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | transfer to RDR | No error |

**Table 27.33 Flags in the SSR_MANC Register and Receive Data Handling in Manchester Mode (2 of 2)**

| Flag in the SSR_MANC register | | | | Flag in the MESR register | | | received data | Received error status (ERI interrupt / event generation) |
|---|---|---|---|---|---|---|---|---|
| ORER | FER | PER | MER | SBER*1 | PFER*1 | SYER | | |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | transfer to RDR | Framin error |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | transfer to RDR | Parity error |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | transfer to RDR | Framing error + Parity error |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | transfer to RDR | Manchester error |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | transfer to RDR | Framing error + Manchester error |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | transfer to RDR | Parity error + Manchester error |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | transfer to RDR | Framing error + Parity error + Manchester error |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | Lost | Overrun error |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | Lost | Overrun error + Framing error |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | Lost | Overrun error + Parity error |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | Lost | Overrun error + Framing error + Parity error |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | Lost | Overrun error + Manchester error |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | Lost | Overrun error + Framing error + Manchester error |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | Lost | Overrun error + Parity error + Manchester error |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | Lost | Overrun error + Framing error + Parity error + Manchester error |
| 0 | Combination of above | | | 0 | 0 | 0 | transfer to RDR | Errors above + Receive SYNC error*2 |
| 1 | | | | 0 | 0 | 0 | Lost | Errors above + Receive SYNC error*2 |
| hold | hold | hold | hold | 0 | 1 | 0 | Lost | Preface error*3 |
| hold | hold | hold | hold | 1 | 0 | 0 | Lost | Start bit error*3 |
| hold | hold | hold | hold | 0 | 1 | 1 | Lost | Preface error*3 + Receive SYNC error*2 |
| hold | hold | hold | hold | 1 | 0 | 1 | Lost | Start bit error*3 + Receive SYNC error*2 |

Note 1. Start bit error and Preface error never become 1 at the same time.
Note 2. When MECR.SYEREN = 1, SCIn_ERI interrupt / event is generated by SYER factor.
Note 3. If MECR.PFEREN = 1 or MECR.SBEREN = 1, an SCIn_ERI interrupt / event is generated when the corresponding flag is set.

**Table 27.34 Errors Detectable in Each Area**

| | Preface error (PFER) | Start Bit error (SBER) | Manchester error (MER) | Receive SYNC error (SYER) | Parity error (PER) | Framing error (FER) |
|---|---|---|---|---|---|---|
| Preface area | ✓ | — | —*1 | ✓*2 | — | — |
| Start Bit area | — | ✓ | — | ✓*2 | — | — |
| Data area | — | — | ✓ | ✓ | — | — |
| Parity area | — | — | ✓ | ✓ | ✓ | — |
| Multi-processor area | — | — | ✓ | ✓ | — | — |
| Stop Bit area | — | — | — | — | — | ✓ |

Note: ✓: Detected, —: Not detected
Note 1. When an Manchester code error occurs in the preface area, it is defined as a preface error.
Note 2. It may not be subject to Receive SYNC error detection. For details see the text section 27.5.11. Errors in Manchester Mode (7)

**Table 27.35 Operation status due to presence / absence of error in previous frame and operation status list in multiprocessor mode (1 of 2)**

| Previous frame | Each area of the Frame | | | | | PFEREN | SBEREN | SYEREN | received data | Error flag | Interrupt request | Event signal |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | preface | start bit | data | parity | stop | | | | | | | |
| No Error | PFER | No Error | Don't Care | Don't Care | Don't Care | 0 | Don't Care | Don't Care | Lost | set PFER*1 | not output | not output |
| | No SYER*1 | | | | | 1 | | | | | output | output |
| | No Error | SBER | Don't Care | Don't Care | Don't Care | Don't Care | 0 | Don't Care | Lost | set SBER*1 | not output | not output |
| | | No SYER*1 | | | | | 1 | | | | output | output |
| | SYER | No Error | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | 0 | transfer to RDR | set SYER | not output | not output |
| | No PFER | | | | | | | 1 | Lost | | output | output |
| | No Error | SYER | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | 0 | transfer to RDR | set SYER | not output | not output |
| | | No SBER | | | | | | 1 | Lost | | output | output |
| | No Error | No Error | SYER | | No Error | Don't Care | Don't Care | 0 | transfer to RDR | set SYER | not output | not output |
| | | | | | | | | 1 | Lost | | output | output |
| | No Error | No Error | MER | | No Error | Don't Care | Don't Care | Don't Care | transfer to RDR | set MER | output | output |
| | No Error | No Error | Don't Care | PER | No Error | Don't Care | Don't Care | Don't Care | transfer to RDR | set PER | output | output |
| | No Error | No Error | Don't Care | Don't Care | FER | Don't Care | Don't Care | Don't Care | transfer to RDR | set FER | output | output |
| | There is some error ORER | | | | | Don't Care | Don't Care | Don't Care | Lost | set some flags*2 | output | output |
| | No Error | No Error | No Error | No Error | No Error ORER | Don't Care | Don't Care | Don't Care | Lost | set ORER | output | output |

**Table 27.35 Operation status due to presence / absence of error in previous frame and operation status list in multiprocessor mode (2 of 2)**

| Previous frame | Each area of the Frame | | | | | PFERE N | SBERE N | SYERE N | received data | Error flag | Interrupt request | Event signal |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | preface | start bit | data | parity | stop | | | | | | | |
| some error*3 *6 | PFER<br>No SYER*1 | No Error | Don't Care | Don't Care | Don't Care | 0 | Don't Care | Don't Care | Lost | set PFER*1 | output*4 | not output*5 |
| | | | | | | 1 | | | | | | |
| | No Error | SBER<br>No SYER*1 | Don't Care | Don't Care | Don't Care | Don't Care | 0 | Don't Care | | set SBER*1 | | |
| | | | | | | | 1 | | | | | |
| | SYER<br>No PFER | No Error | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | 0 | | set SYER | | |
| | | | | | | | | 1 | | | | |
| | No Error | SYER<br>No SBER | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | 0 | | set SYER | | |
| | | | | | | | | 1 | | | | |
| | No Error | No Error | SYER | | No Error | Don't Care | Don't Care | 0 | | don't set any flags | | |
| | | | | | | | | 1 | | | | |
| | No Error | No Error | MER | | No Error | Don't Care | Don't Care | Don't Care | | | | |
| | No Error | No Error | Don't Care | PER | No Error | Don't Care | Don't Care | Don't Care | | | | |
| | No Error | No Error | Don't Care | Don't Care | FER | Don't Care | Don't Care | Don't Care | | | | |
| | There is some error<br>ORER | | | | | Don't Care | Don't Care | Don't Care | | | | |
| | No Error | No Error | No Error | No Error | No Error<br>ORER | Don't Care | Don't Care | Don't Care | | | | |

Note 1. If SYER is detected, the SYER flag is also set. Other operations are as shown in this table.
Note 2. Other detected error flags including ORER are also set.
Note 3. If all the error flags are cleared before the STOP bit is judged, the operation will be the same as the case where there is no error in the previous frame of this table.
Note 4. Since the SCIn_ERI interrupt request is level output, it remains active due to errors in the previous frame regardless of the presence or absence of error in the relevant frame.
Note 5. Since the error cause is continuously detected, the SCIn_ERI event is not newly output regardless of the presence or absence of errors in the relevant frame.
Note 6. For PFER, SBER, and SYER, when each enable bit is set to disable, it is treated as no error.

**Table 27.36 Operation when MPIE = 1 in multi-processor mode (MPIE = 0)**

| MPB*1 | Each area of the frame | | | | | PFERE N | SBERE N | SYERE N | received data | Error flag | Interrupt request | Event signal |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | preface | start bit | data | parity | stop | | | | | | | |
| 1 | No Error | No Error | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | transfer to RDR | set some flags | output*2 | output*2 |
| | No PFER | No SBER | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | 0 | | | | |
| | SYER*3 | SYER*3 | | | | | | 1 | Lost | don't set any flags | not output | not output |
| | PFER | No Error | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | | | | |
| | No Error | SBER | Don't Care | Don't Care | Don't Care | Don't Care | Don't Care | | | | | |

Note 1. If the received MPB bit is 0, it is not received the frame, and the operation is the same as lost of the reception data of this table.
Note 2. If no error is detected, SCIn_RXI interrupt request or event is output, and if it is detected, SCIn_ERI interrupt request or event is output.

Note 3. When SYER is detected in the preface area or the start bit area, the behavior of handling as an error depending on the SYEREN bit changes.

## 27.6 Operation in Clock Synchronous Mode

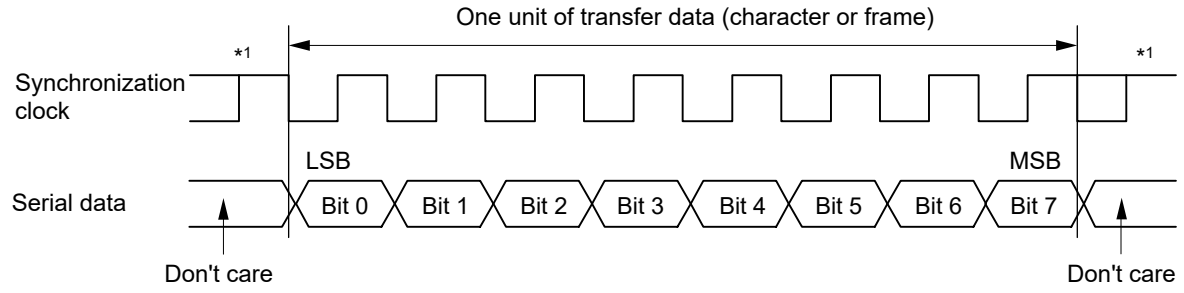
Figure 27.65 shows the data format for clock synchronous serial data communications.

In clock synchronous mode, data is transmitted or received in synchronization with clock pulses. For single-character data transfer, data consists of 8-bit. In clock synchronous mode, no parity bit can be added.

In data transmission, the SCI outputs data from one falling edge of the synchronization clock to the next falling edge. In data reception, the SCI receives data in synchronization with the rising edge of the synchronization clock. After 8-bit data is output, the transmission line holds the last bit as output state. When the SPMR.CKPH bit is 1 in slave mode, the transmission line holds the first bit output state.

Within the SCI, the transmitter and receiver are independent units, enabling full-duplex communications by using a shared clock. Both the transmitter and the receiver also have a double-buffered structure, so that the next transmit data can be written during transmission or the previous receive data can be read during reception, enabling continuous data transfer.

However, it is not possible to perform continuous transfer in the fastest bit rate setting (BRR[7:0] = 0x00 and SMR.CKS[1:0] = 00b). Therefore, when the FIFO is selected, this setting (BRR[7:0] = 0x00 and SMR.CKS[1:0] = 00b) is not available.




Note 1. Holds a high level except during continuous transfer.

**Figure 27.65 Data format in clock synchronous serial communications with LSB-first order**

### 27.6.1 Clock

Either an internal clock generated by the on-chip baud rate generator or an external synchronization clock input at the SCKn pin can be selected based on the SCR.CKE[1:0] setting.

When the SCI operates on an internal clock, the synchronization clock is output from the SCKn pin. Eight synchronization clock pulses are output in the transfer of one character. When no transfer is performed, the clock is held high. However, when only data reception is performed while the CTS function is disabled, the synchronization clock output starts when the SCR.RE bit set to 1. The synchronization clock stops when it goes high[*1] and an overrun error occurs or the SCR.RE bit is set to 0.

When only data reception is performed and the CTS function is enabled, the clock output does not start when the SCR.RE bit set to 1 and the CTSn_RTSn pin input is high. The synchronization clock output starts when the SCR.RE bit is set to 1 and the CTSn_RTSn pin input is low. Following that, when the CTSn_RTSn pin input is high on completion of the frame reception, the synchronization clock output stops when it goes high. If the CTSn_RTSn pin input continues to be low, the synchronization clock stops when it goes high[*1] and an overrun error occurs or the SCR.RE bit is set to 0.

Note 1. The signal is held high while (SPMR.CKPH = 0 and SPMR.CKPOL = 1) or (SPMR.CKPH = 1 and SPMR.CKPOL = 1). It is held low while (SPMR.CKPH = 0 and SPMR.CKPOL = 1) or (SPMR.CKPH = 1 and SPMR.CKPOL = 0).

### 27.6.2 CTS and RTS Functions

In the CTS function, the CTSn_RTSn pin input controls the start of data reception or transmission when the clock source is the internal clock. Setting the SPMR.CTSE bit to 1 enables the CTS function. When the CTS function is enabled, setting the CTSn_RTSn pin low causes data reception or transmission to start.

Setting the CTSn_RTSn pin high while the data transmission or reception is in progress does not affect transmission or reception of the current frame.

In the RTS function, the CTSn_RTSn pin output is used to request the start of data reception or transmission when the clock source is an external synchronizing clock. The CTSn_RTSn output goes low when serial communication is enabled. Conditions for output of the CTSn_RTSn low and high are shown as follows:

[Conditions for low output]

Satisfaction of all the following conditions:

**Non-FIFO selected when all of the following conditions are satisfied**

- The value of the SCR.RE bit or the SCR.TE bit is 1
- Neither transmission nor reception is in progress
- There is no received data available to be read when the SCR.RE bit is 1
- Transmit data is written when the SCR.TE bit is 1 and SCR.CKE[1] bit is 0
- Data is available for transmission in the TSR register when SCR.TE bit is 1 and SCR.CKE[1] bit is 1
- The SSR.ORER flag is 0

**FIFO selected when all of the following conditions are satisfied**

- The value of the SCR.RE bit or the SCR.TE bit is 1
- Neither transmission nor reception is in progress
- The amount of receive data written in FRDRHL is less than the setting value of FCRH.RSTRG[3:0] when SCR.RE = 1
- Data that has not been transmitted is available in FTDRHL when SCR.TE bit is 1 and SCR.CKE[1] bit is 0
- Data is available for transmission in the TSR register when SCR.TE bit is 1 and SCR.CKE[1] bit is 1
- The SSR_FIFO.ORER flag is 0

[Condition for high output]

- The conditions for low output are not satisfied

### 27.6.3 SCI Initialization in Clock Synchronous Mode

Before transmitting and receiving data, start by writing the initial value 0x00 to the SCR register, then continue through the SCI initialization procedure given in the sections describing non-FIFO and FIFO selection in section 27.6.2. CTS and RTS Functions. Anytime the operating mode or transfer format is to be changed, the SCR register must be initialized before the change can be made.

Note: Setting the SCR.RE bit to 0 initializes neither the ORER, FER, and PER flags in SSR/SSR_FIFO nor the RDR register. When the TE bit is set to 0, the TEND flag for the selected FIFO buffer is not initialized.

Note: In non-FIFO mode, switching the value of the SCR.TE bit from 1 to 0 or 0 to 1 when the SCR.TIE bit is 1 generates an SCIn_TXI interrupt request.

**Table 27.37 Example flow of SCI initialization in clock synchronous mode with non-FIFO selected (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Start initialization | |
| 2 | Set the SCR.TIE, RIE, TE, RE, and TEIE bits to 0 | |
| 3 | Set the FCR.FM bit to 0 | Set the FCR.FM bit to 0. |
| 4 | Set the SCR.CKE[1:0] bits | Set the clock selection in SCR. |
| 5 | Set the SIMR1.IICM bit to 0. Set the SPMR.CKPH and CKPOL bits. | Set the SIMR1.IICM bit to 0.<br>Set the SPMR.CKPH and CKPOL bits.<br>Step 5 can be skipped if the values have not been changed from the initial values. |

**Table 27.37 Example flow of SCI initialization in clock synchronous mode with non-FIFO selected (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 6 | Set the data transmission/ reception format in SMR, SCMR, and SEMR | Set data transmission/reception format in SMR, SCMR, and SEMR. |
| 7 | Set a value in SPTR | Set the communication terminals status in SPTR. |
| 8 | Set a value in BRR | Write a value corresponding to the bit rate to BRR.<br>This step is not necessary if an external clock is used. |
| 9 | Set a value in MDDR | Write the value obtained by correcting a bit rate error in MDDR. This step is not necessary if the BRME bit in SEMR is set to 0 or an external clock is used. |
| 10 | Set the I/O port functions | Make I/O port settings to enable input and output functions as required for TXDn, RXDn, and SCKn pins. |
| 11 | Set the SCR.TE or RE bit to 1, and set the SCR.TIE and RIE bits | Set the SCR.TE or RE bit to 1. Also set the SCR.TIE and RIE bits.<br>Setting the TE and RE bits allows TXDn and RXDn pins to be used. |
| 12 | Initialization completion | |

Note: In simultaneous transmit and receive operations, the TE and RE bits in SCR must both be set to 0 or set to 1 simultaneously

**Table 27.38 Example flow of SCI initialization in clock synchronous mode with FIFO selected**

| No. | Step Name | Description |
|---|---|---|
| 1 | Start initialization | |
| 2 | Set the SCR.TIE, RIE, TE, RE, and TEIE bits to 0 | |
| 3 | Set the FCR.FM, TFRST, and RFRST bits to 1.<br>Set the FCR.TTRG[3:0], RTRG[3:0], and RSTRG[3:0] bits. | Set the FCR.FM, TFRST, and RFRST bits to 1 (FIFO mode enabled, transmit/receive FIFOs empty).<br>Set the FCR.TTRG[3:0], RTRG[3:0], and RSTRG[3:0] bits. |
| 4 | Set the SCR.CKE[1:0] bits | Set the clock selection in SCR. |
| 5 | Set the SIMR1.IICM bit to 0.<br>Set the SPMR.CKPH and CKPOL bits. | Set the SIMR1.IICM bit to 0.<br>Set the SPMR.CKPH and CKPOL bits.<br>Step 5 can be skipped if the values have not been changed from the initial values. |
| 6 | Set the data transmission/ reception format in SMR, SCMR, and SEMR | Set data transmission/reception format in SMR, SCMR, and SEMR. |
| 7 | Set a value in SPTR | Set the communication terminals status in SPTR. |
| 8 | Set a value in BRR | Write a value corresponding to the bit rate to BRR.<br>This step is not necessary if an external clock is used. |
| 9 | Set a value in MDDR | Write the value obtained by correcting a bit rate error in MDDR. This step is not necessary if the BRME bit in SEMR is set to 0 or an external clock is used. |
| 10 | Set the FCR.TFRST and RFRST bits to 0 | Set the FCR.TFRST and RFRST bits to 0. |
| 11 | Set the I/O port functions | Make I/O port settings to enable input and output functions as required for TXDn, RXDn, and SCKn pins. |
| 12 | Set the SCR.TE or RE bit to 1, and set the SCR.TIE and RIE bits | Set the SCR.TE or RE bit to 1. Also set the SCR.TIE and RIE bits.<br>Setting the TE and RE bits allows TXDn and RXDn pins to be used. |
| 13 | Initialization completion | |

Note: In simultaneous transmit and receive operations, the TE and RE bits in SCR must both be set to 0 or set to 1 simultaneously.

## 27.6.4 Serial Data Transmission in Clock Synchronous Mode

(1) Non-FIFO selected

Figure 27.66, Figure 27.67, and Figure 27.68 show examples of serial transmission in clock synchronous mode.

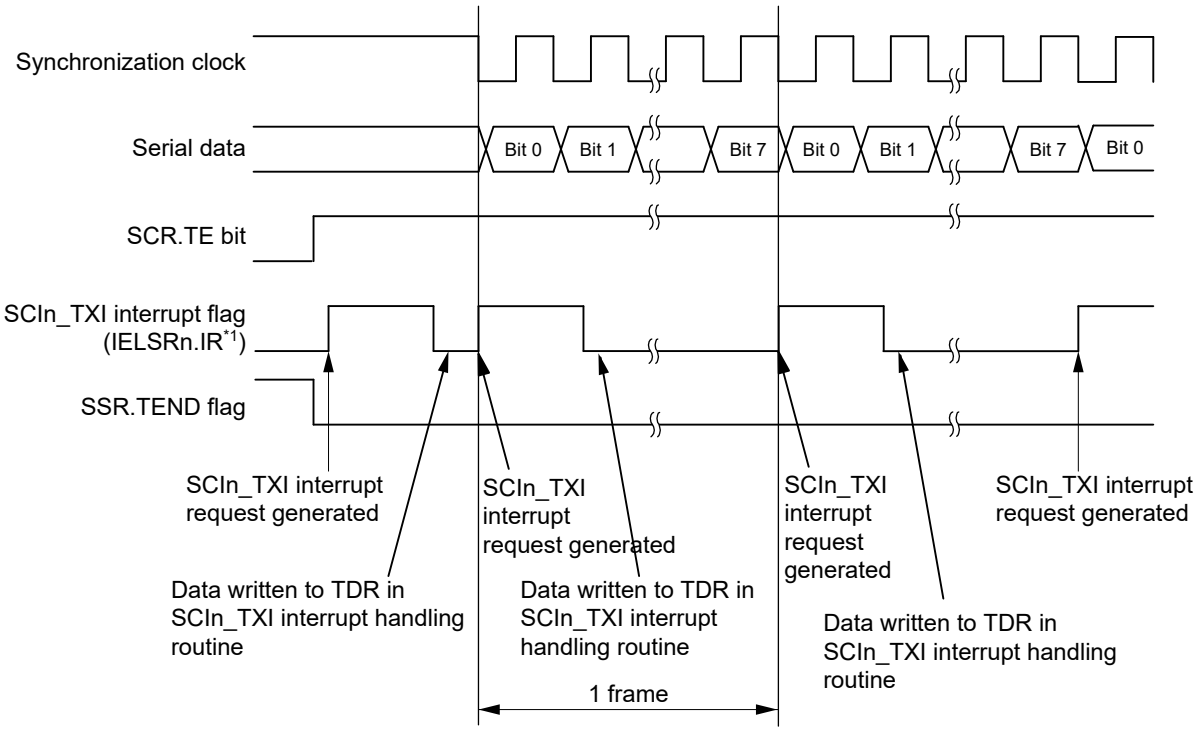
In serial data transmission, the SCI operates as follows:

1. The SCI transfers data from the TDR register to the TSR register when data is written to TDR in the SCIn_TXI interrupt handling routine. The SCIn_TXI interrupt request at the beginning of transmission is generated when the TE bit is set to 1 but only after the TIE bit in the SCR is also set to 1 or when these two bits are set to 1 simultaneously by a single instruction.
2. After transferring data from TDR to TSR, the SCI starts transmission. When the SCR.TIE bit is set to 1, an SCIn_TXI interrupt request is generated. Continuous transmission is enabled by writing the next transmit data to TDR in the SCIn_TXI interrupt handling routine before transmission of the current transmit data finishes. When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 and the SCR.TEIE bit to 1 after the last of the data to be transmitted is written to the TDR register from the handling routine for SCIn_TXI requests.
3. 8-bit data is sent from the TXDn pin in synchronization with the output clock when the clock output mode is specified and in synchronization with the input clock when the use of an external clock is specified. Output of the clock signal is suspended until the input CTS signal is low when the SPMR.CTSE bit is 1.
4. The SCI checks for update to the TDR register on output of the last bit.
5. When the TDR register is updated, the next transmit data is transferred from TDR to TSR, and serial transmission of the next frame starts.
6. If TDR is not updated, the SSR.TEND flag is set to 1. The TXDn pin retains the output state of the last bit. If the SCR.TEIE bit is 1, an SCIn_TEI interrupt request is generated and the SCKn pin is held high.

Figure 27.66, Figure 27.67, and Figure 27.68 show examples of serial data transmission.

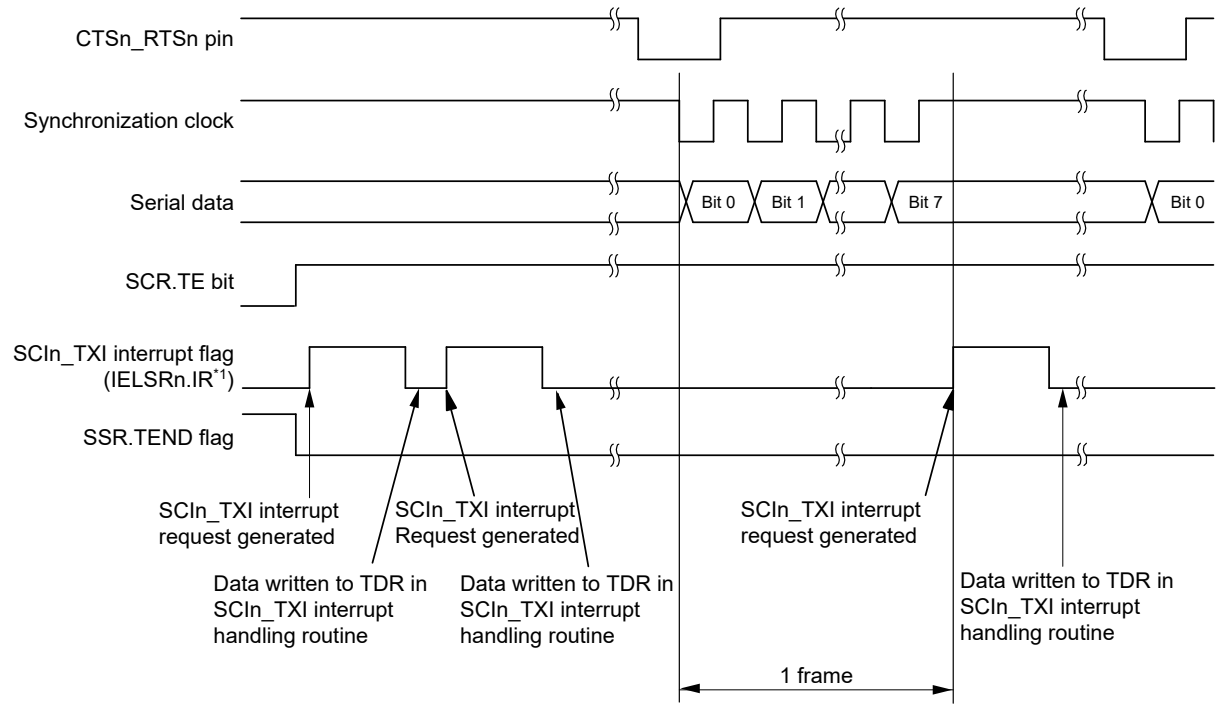
Transmission does not start while a receive error flag (ORER, FER, or PER in SSR) is set to 1. Always set the receive error flags to 0 before starting transmission.

Note: Setting the SCR.RE bit to 0 does not clear the receive error flags.



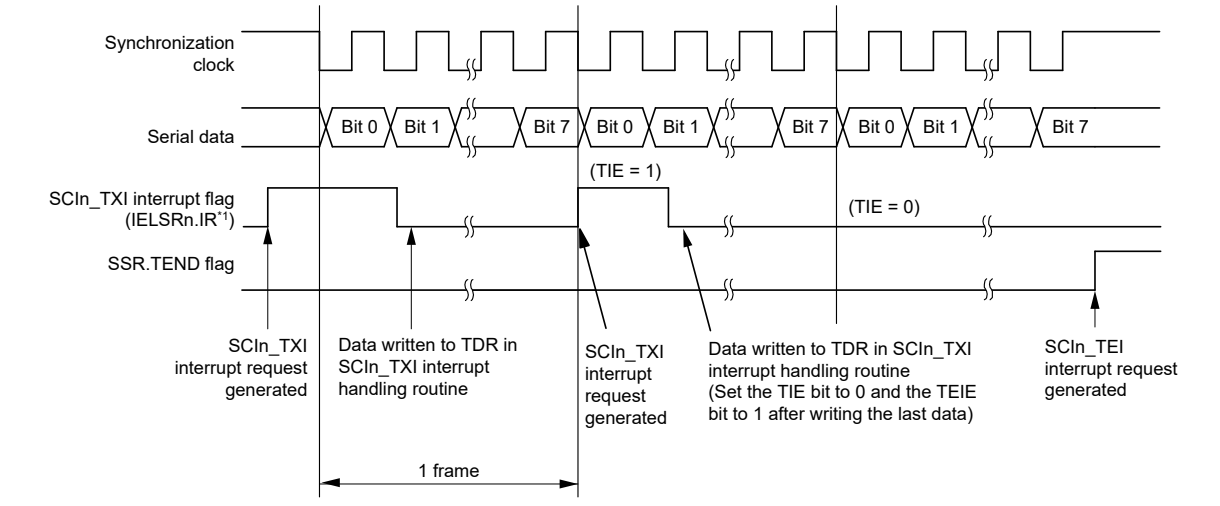

Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.66 Example of serial data transmission in clock synchronous mode when the CTS function is not used at the beginning of transmission**



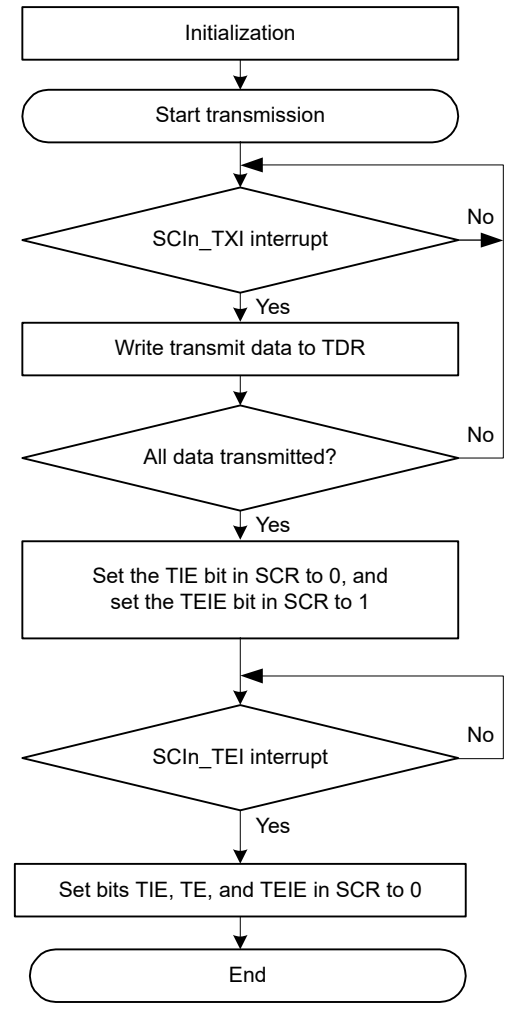

Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.67 Example of serial data transmission in clock synchronous mode when the CTS function is used at the beginning of transmission**




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.68 Example of serial data transmission in clock synchronous mode from the middle of transmission until transmission completion**




[ 1 ] [ 1 ]SCI initialization:
Set data transmission.

[ 2 ] [ 2 ]Writing transmit data write to TDR by an SCIn_TXI interrupt request:
When transmit data is transferred from TDR to TSR, a transmit data empty interrupt (SCIn_TXI) request is generated.
Transmit data is written to TDR once from the handling routine for SCIn_TXI requests.

[ 3 ] [ 3 ]Serial transmission continuation procedure:
To continue serial transmission, write transmit data to TDR upon accepting a transmit data empty interrupt (SCIn_TXI) request. Transmit data can also be written to TDR by activating the DMAC or DTC by theSCIn_TXI interrup trequest.
When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 and the SCR.TEIE bit to 1 after the last of the data to be transmitted is written to theTDR

Note: When the external clock is in use (the value of the SCR.CKE[1:0] bits is 10b or 11b), the rising edge on the SCKn pin for the last bit sets the SSR.TEND flag to 1. Setting the SCR.TE bit to 0 immediately after this might lead to insufficient received-data hold time on the receiver side.

**Figure 27.69 Example flow of serial transmission in clock synchronous mode with non-FIFO selected**

(2) FIFO selected

Figure 27.70 shows an example of serial transmission in clock synchronous mode with FIFO selected.

In serial data transmission, the SCI operates as follows:

1. The SCI transfers data from the FTDRL[*1] register to the TSR register when data is written to FTDRL[*1] in the SCIn_TXI interrupt handling routine. The amount of data that can be written to FTDRL is 16 minus FDR.T[4:0] bytes. The SCIn_TXI interrupt request at the beginning of transmission is generated when the SCR.TE bit is set to 1 but only after the SCR.TIE bit is also set to 1 or when these two bits are set to 1 simultaneously by a single instruction.
2. After transferring data from FTDRL to TSR, the SCI starts transmission. When the amount of transmit data written in FTDRL is equal to or less than the specified transmit triggering number, the SSR_FIFO.TDFE is set to 1. When the SCR.TIE bit is set to 1, an SCIn_TXI interrupt request is generated. Continuous transmission is enabled by writing the next transmit data to FTDRL in the SCIn_TXI interrupt handling routine before transmission of the current transmit data has finished. When SCIn_TEI interrupt requests are in use, set the SCR.TIE bit to 0 and the SCR.TEIE bit to 1 after the last of the data to be transmitted is written to the FTDRL from the handling routine for SCIn_TXI requests.
3. 8-bit data is sent from the TXDn pin in synchronization with the output clock when the clock output mode is specified and in synchronization with the input clock when the use of an external clock is specified. Output of the clock signal is suspended until the input CTS signal is low when the SPMR.CTSE bit is 1.
4. The SCI checks whether non-transmitted data remains in FTDRL on output of the stop bit.

5. When FTDRL is updated, the next transmit data is transferred from FTDRL to TSR and serial transmission of the next frame starts.
6. If FTDRL is not updated, the SSR_FIFO.TEND flag is set to 1. The TXDn pin retains the output state of the last bit. If the SCR.TEIE bit is 1, an SCIn_TEI interrupt request is generated and the SCKn pin is held high.

Note 1. In clock synchronous mode, FTDRH is not used.

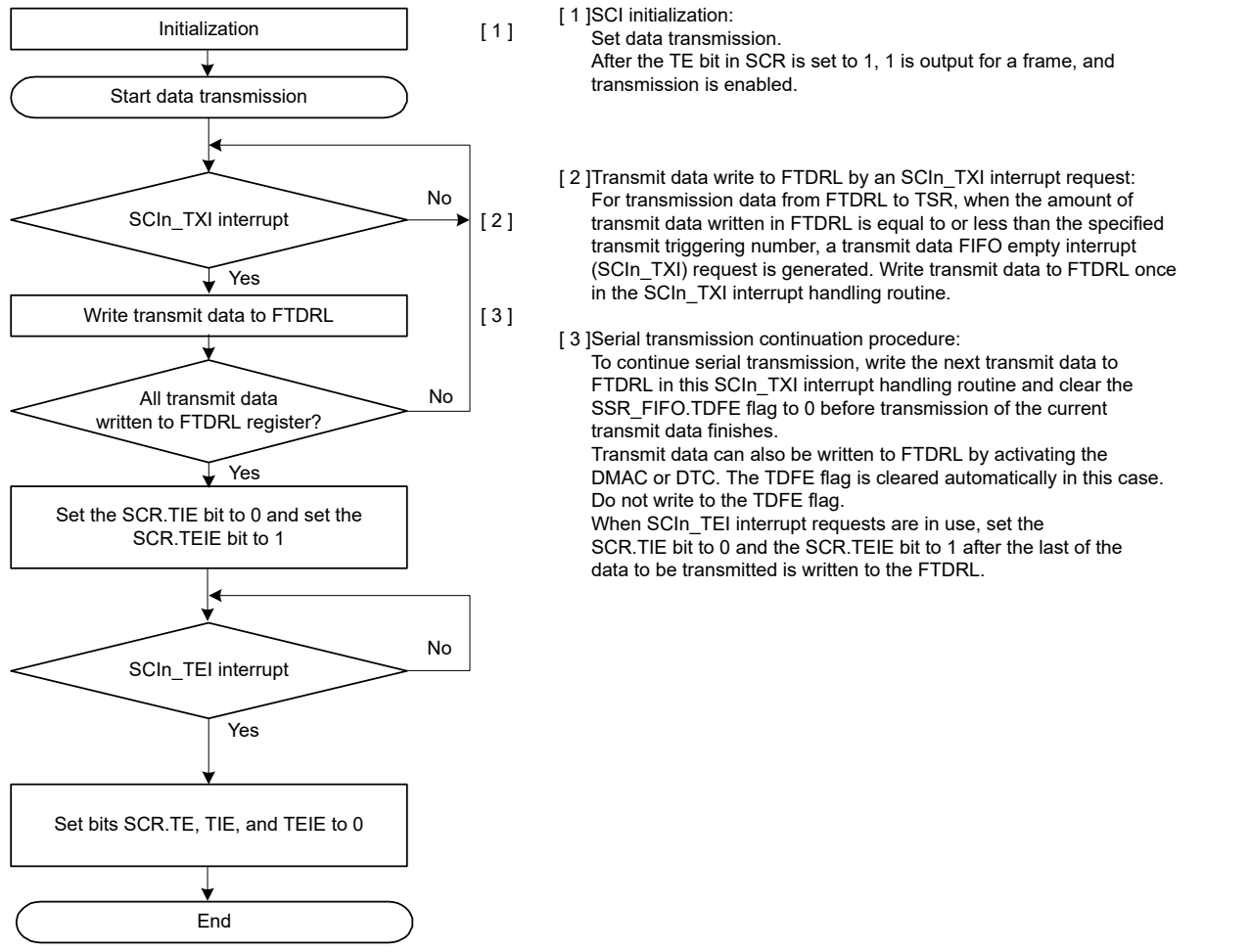



**Figure 27.70 Example flow of serial transmission in clock synchronous mode with FIFO selected**

## 27.6.5 Serial Data Reception in Clock Synchronous Mode

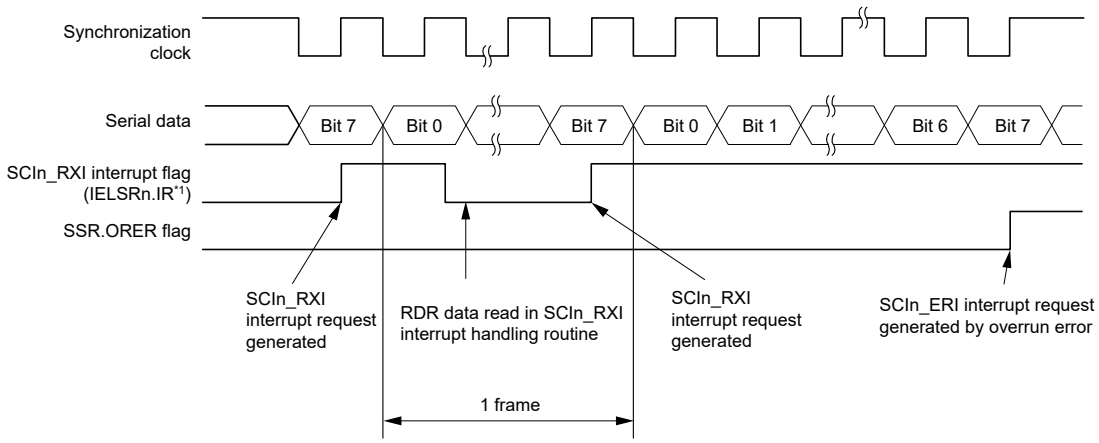
(1) Non-FIFO selected

Figure 27.71 and Figure 27.72 show examples of SCI operation for serial reception in clock synchronous mode.

In serial data reception, the SCI operates as follows:

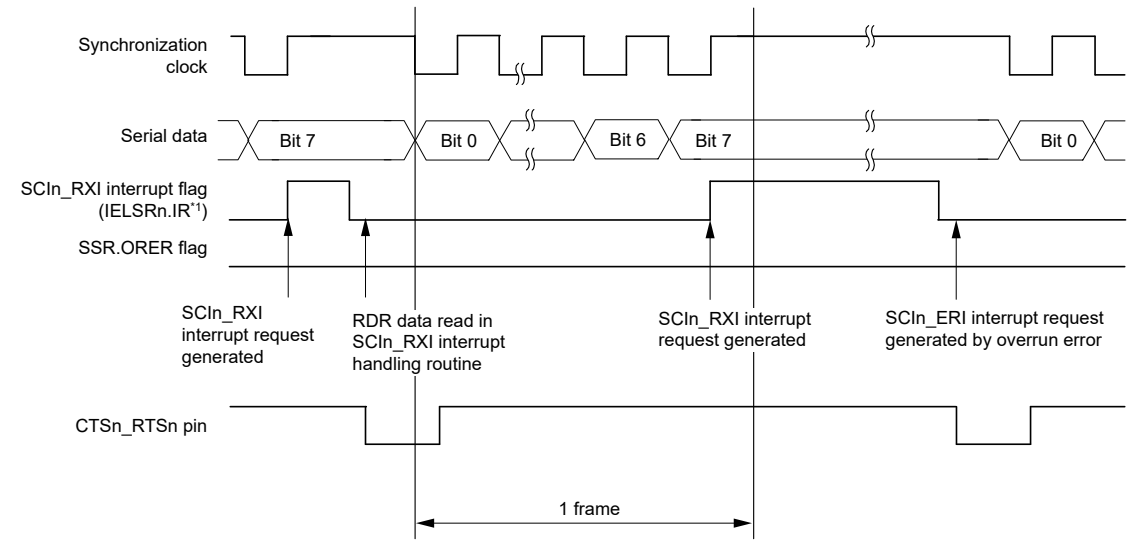
1. When the value of the SCR.RE bit becomes 1, the CTSn_RTSn pin goes low.
2. The SCI performs internal initialization and starts receiving data in synchronization with a synchronization clock input or output, and stores the receive data in the RSR register.
3. If an overrun error occurs, the SSR.ORER flag is set to 1. If the SCR.RIE bit is 1, an SCIn_ERI interrupt request is generated. Receive data is not transferred to the RDR register.

4. When reception completes successfully, receive data is transferred to the RDR register. If the SCR.RIE bit is 1, an SCIn_RXI interrupt request is generated. Continuous reception is enabled by reading the received data transferred to the RDR register in the SCIn_RXI interrupt handling routine before reception of the next receive data completes. Reading the received data that is transferred to RDR causes the CTSn_RTSn pin to output low.




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.71 Example operation for serial reception in clock synchronous mode (1) when the RTS function is not used**




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.72 Example operation for serial reception in clock synchronous mode (2) when RTS function is used**

Data transfer cannot resume while the receive error flag is 1. Therefore, clear the ORER, FER, and PER flags in the SSR register to 0 before resuming data reception. Additionally, always read the RDR register during overrun error processing. When a data reception is forced to terminate by a 0 write to the SCR.RE bit during operation, read the RDR register because received data that is not yet read might be left in the RDR register.

Figure 27.73 shows an example flow of serial data reception.

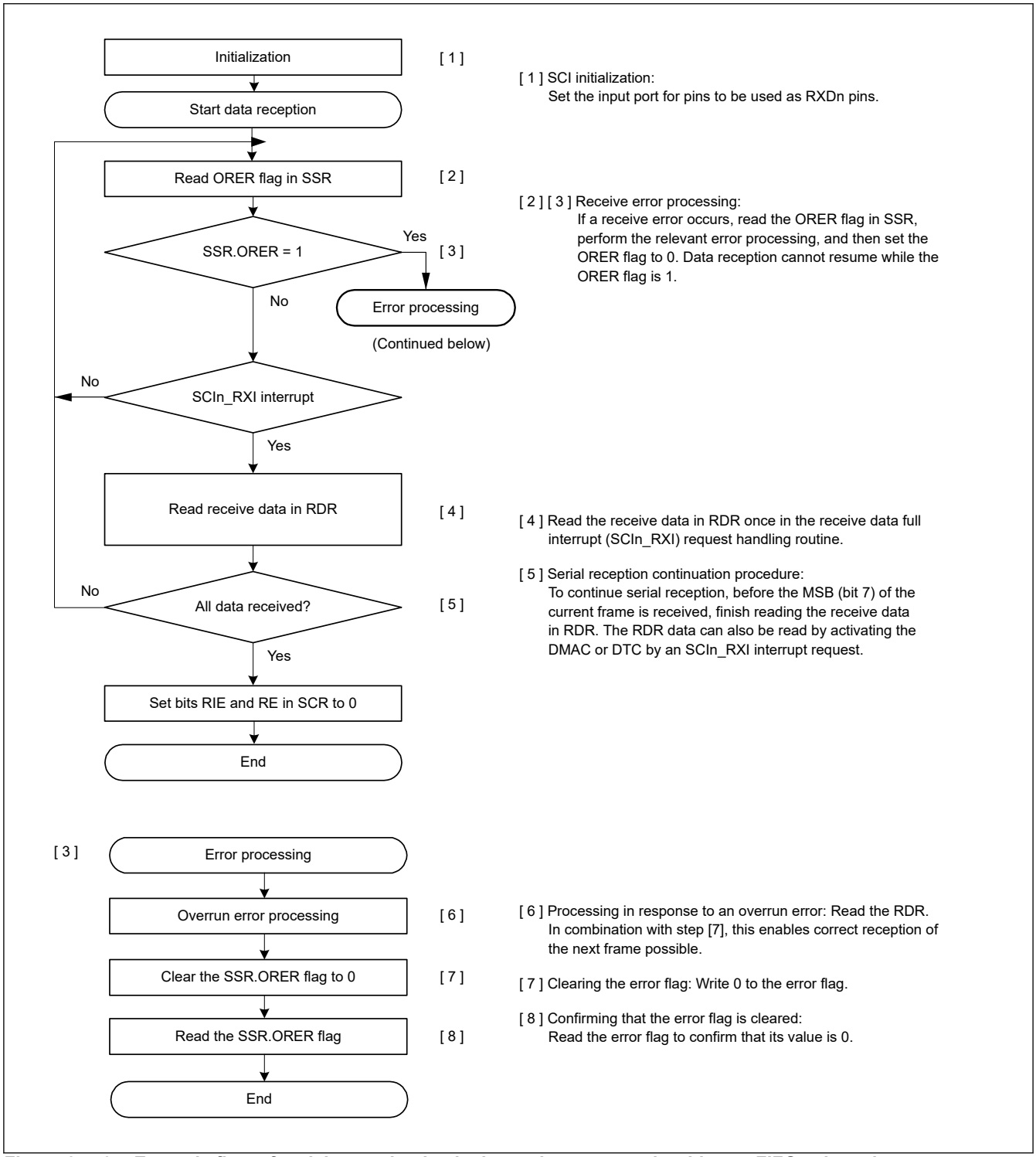



**Figure 27.73 Example flow of serial reception in clock synchronous mode with non-FIFO selected**

(2) FIFO selected

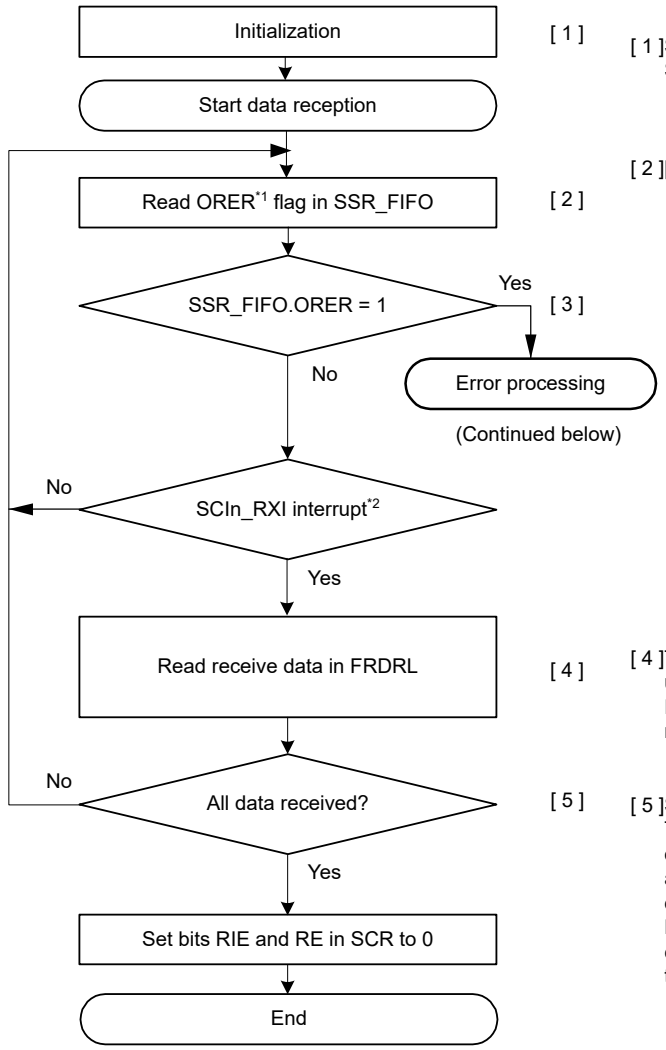
Figure 27.74 shows an example of serial reception in clock synchronous mode with FIFO selected.

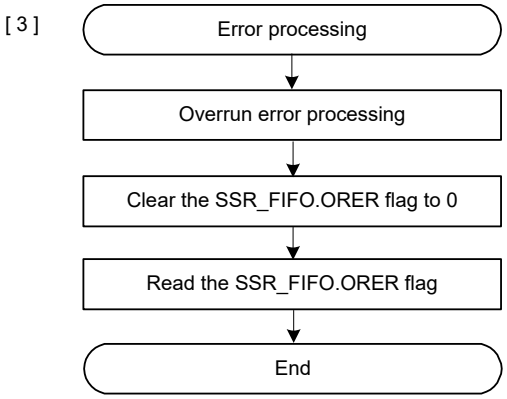
In serial data reception, the SCI operates as follows:

1. When the value of the SCR.RE bit becomes 1, the CTSn_RTSn pin goes low.
2. The SCI performs internal initialization and starts receiving data in synchronization with a synchronization clock input or output, and stores the receive data in the RSR register.

3. If an overrun error occurs, the SSR_FIFO.ORER flag is set to 1. If the SCR.RIE bit is 1, an SCIn_ERI interrupt request is generated. Received data is not transferred to the FRDRL[*1] register.
4. When data reception completes successfully, the receive data is transferred to the FRDRL[*1] register. The RDF flag is set to 1 when the amount of the receive data stored in FRDRL is equal to or greater than the specified receive triggering number. If the SCR.RIE bit is 1, an SCIn_RXI interrupt request is generated. Continuous data reception is enabled by reading the receive data transferred to FRDRL[*2] in the SCIn_RXI interrupt handling routine before an overrun error occurs. If the amount of received data that is transferred to FRDRL is less than the specified receive triggering number, the CTSn_RTSn pin goes low.

Note 1. In clock synchronous mode, FRDRH is not used.
Note 2. Read data in order from FRDRH to FRDRL when RDF and ORER are read with receive data.




[ 1 ]SCI initialization:
Set the input port for pins to be used as RXDn pins.

[ 2 ][ 3 ]Receive error processing:
If a receive error occurs, read the ORER flag in SSR_FIFO, perform the relevant error processing, then set the SSR_FIFO.ORER flag to 0. Data reception cannot resume while the ORER flag is 1.

[ 4 ]The receive data stored in FRDRL register is read until the amount of stored data is less than the FCR.RTRG[3:0] value. Software can check readable data in FDR.R[4:0].

[ 5 ]Serial reception continuation procedure:
To continue serial reception, before overrun error occurs, finish reading the receive data in FRDRL and clear the SSR_FIFO.RDF flag to 0.The FRDRL data can also be read by activating the DMAC or DTC by an SCIn_RXI interrupt request. The RDF flag is cleared automatically in this case. Do not write to the RDF flag.

[ 6 ]Processing in response to an overrun error:
Read the FRDRL. In combination with step [7], this enables correct reception of the next frame possible.

[ 7 ]Clearing the error flag:
Write 0 to the error flag.

[ 8 ]Confirming that the error flag is cleared:
Read the error flag to confirm that its value is 0.

Note 1. It can also be read from FRDRH.ORER. However, to clear the ORER flag, write 0 to the associated bit in the SSR_FIFO register.
Note 2. It should be all receive data and is an integer times the FIFO triggering number.

**Figure 27.74 Example flow of serial reception in clock synchronous mode with FIFO selected**

### 27.6.6 Simultaneous Serial Data Transmission and Reception in Clock Synchronous Mode

**(1) Non-FIFO selected**

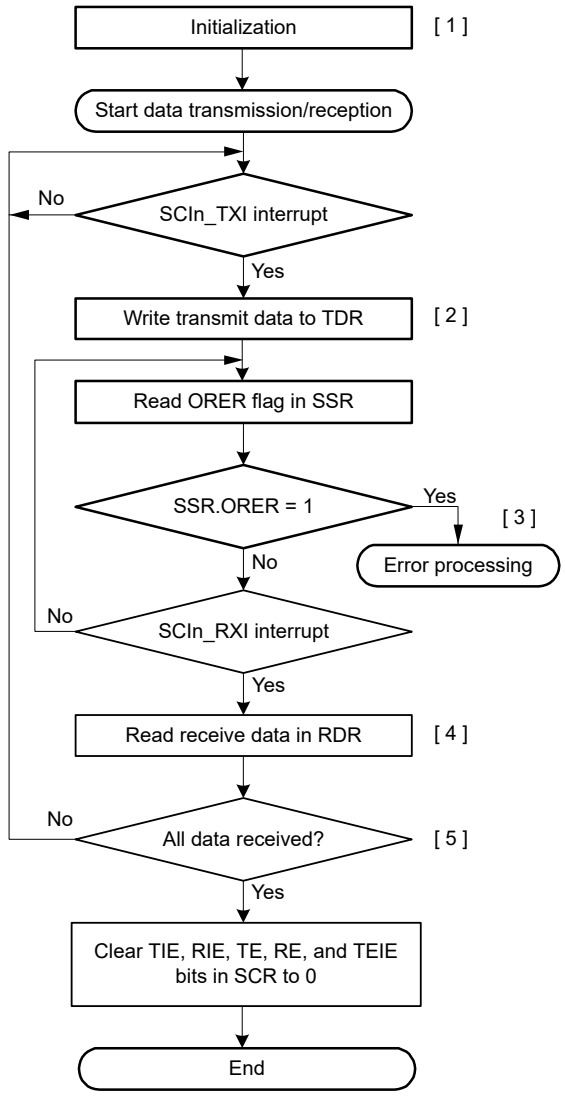
Figure 27.75 shows an example flow of simultaneous serial transmit and receive operations in clock synchronous mode. After initializing the SCI, use the following procedure for simultaneous serial data transmit and receive operations.

To switch from transmit mode to simultaneous transmit and receive mode:

1. Check that the SCI completes the data transmission by verifying that the SSR.TEND flag is set to 1.
2. Initialize the SCR register, and then set the TIE, RIE, TE, and RE bits in the SCR register to 1 simultaneously by a single instruction.

To switch from receive mode to simultaneous transmit and receive mode:

1. Check that the SCI completes the data reception.
2. Set the RIE and RE bits to 0, and then check that the receive error flag ORER in the SSR register is 0.
3. Set the TIE, RIE, TE, and RE bits in the SCR register to 1 simultaneously by a single instruction.




[ 1 ] SCI initialization:
The TXDn pin can act as the output pin for transmitted data and the RXDn pin can act as the input pin for received data at the same time.

[ 2 ] Transmit data write:
Write transmit data to TDR once in the SCIn_TXI interrupt request handling routine.

[ 3 ] Receive error processing:
If a receive error occurs, read the ORER flag in SSR, perform the relevant error processing, then set the ORER flag to 0. Data reception cannot resume while the ORER flag is 1.

[ 4 ] Reading receive data:
Read the receive data in RDR once in the SCIn_RXI interrupt request handling routine.

[ 5 ] Serial transmission/reception continuation procedure:
To continue serial transmission and reception, before the MSB bit 7 of the current frame is received, finish reading the receive data in RDR by the SCIn_RXI interrupt. Also, before the MSB bit 7 of the current frame is transmitted, write data to TDR by the SCIn_TXI interrupt.
Transmit data can also be written to TDR by activating the DMAC or DTC by a transmit data empty interrupt (SCIn_TXI) request. Similarly, the RDR data can also be read by activating the DMAC or DTC by a receive data full interrupt (SCIn_RXI) request.

Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first set the TIE, RIE, TE, RE, and TEIE bits in SCR to 0, then set TIE, RIE, TE, and RE bits to 1 simultaneously.

**Figure 27.75 Example flow of simultaneous serial transmission and reception in clock synchronous mode with non-FIFO selected**

### (2) FIFO selected

Figure 27.76 shows an example flow of simultaneous serial transmit and receive operations in clock synchronous mode with FIFO selected.

After initializing the SCI, use the following procedure for simultaneous serial data transmit and receive operations.

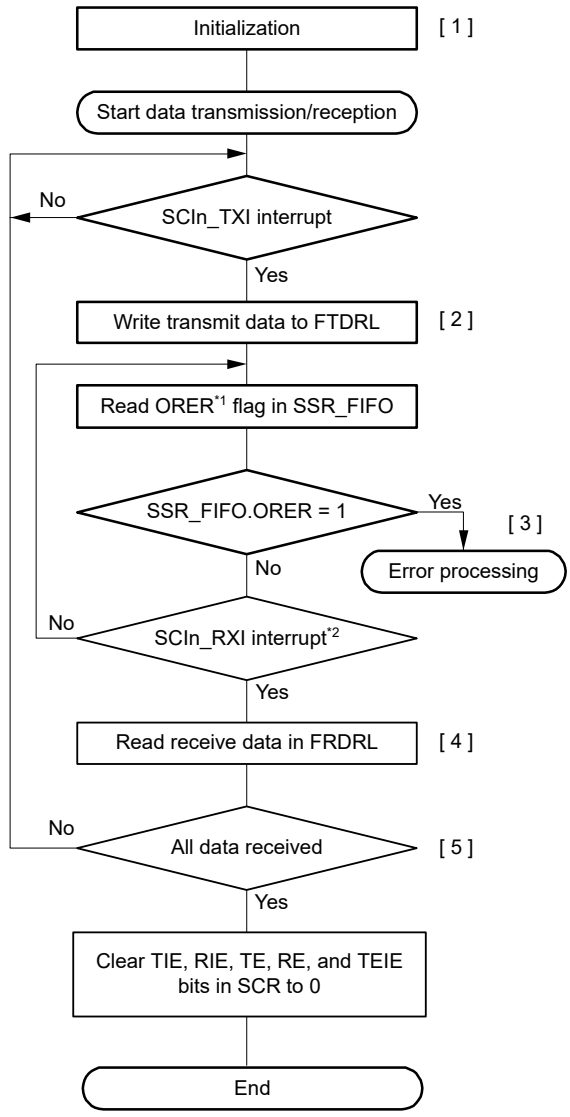
To switch from transmit mode to simultaneous transmit and receive mode:

1. Check that the SCI completes the transmission by verifying that the SSR_FIFO.TEND flag is set to 1.
2. Initialize the SCR register, then set the TIE, RIE, TE, and RE bits in the SCR register to 1 simultaneously by a single instruction.

To switch from receive mode to simultaneous transmit and receive mode:

1. Check that the SCI completes the reception.

2. Set the RIE and RE bits to 0.
3. Check that the receive error flags ORER in the SSR_FIFO register are 0, and then set the TIE, RIE, TE, and RE bits in the SCR register to 1 simultaneously by a single instruction.




[ 1 ] SCI initialization:
The TXDn pin can act as the output pin for transmitted data and the RXDn pin can act as the input pin for received data at the same time.

[ 2 ] Transmit data write:
Write transmit data to FTDRL in the SCIn_TXI interrupt request handling routine. The number of transmission data that can be written is 16 bits, which is the number of stored transmit data in the FIFO.

[ 3 ] Receive error processing:
If a receive error occurs, read the ORER flag in SSR_FIFO, performthe relevant error processing, and then set the ORER flag to 0. Data reception cannot resume while the ORER flag is 1.

[ 4 ] Reading receive data:
The reception data stored in FRDRL register is read until the number of stored data is less than the FCR.RTRG value. Software can check readable data in FDR.R[4:0].

[ 5 ] Serial transmission/reception continuation procedure:
To continue serial reception, before overrun error occurs, finish reading the receive data in FRDRL and clear the SSR_FIFO.RDF flag to 0. To continue serial transmission, before transmission of the current transmit data is finished, write the next transmit data to FTDRL in the SCIn_TXI interrupthandling routine and clear the SSR_FIFO.TDFE flag to 0.
Transmit data can also be written to FTDRL by activating the DMAC or DTC by a transmit FIFO data empty interrupt (SCIn_TXI) request. Similarly, the FRDRL data can also be read by activating the DMAC or DTC by a receive FIFO data full interrupt (SCIn_RXI) request. The RDF and TDFE flags are cleared automatically in this case. Do not write to the RDF and TDFE flags.

Note 1. It can also be read from FRDRH.ORER. To clear the ORER flag, write 0 to the associated bit in the SSR_FIFO register.
Note 2. It should be all receive data with amount an integer times FIFO triggering number.

**Figure 27.76 Example flow of simultaneous serial transmission and reception in clock synchronous mode with FIFO selected**

## 27.7 Operation in Smart Card Interface Mode

The SCI supports smart card (IC card) interfaces conforming to ISO/IEC 7816-3 (standard for Identification Cards), as an extended function of the SCI.

Smart card interface mode can be selected using the appropriate register.

### 27.7.1 Example Connection

Figure 27.77 shows an example connection between a smart card (IC card) and the MCU. As shown in Figure 27.77, because the MCU communicates with an IC card using a single transmission line, interconnect the TXDn and RXDn pins and pull up the data transmission line to VCC using a resistor.

Setting the SCR_SMCI.TE and SCR_SMCI.RE bits to 1 with an IC card disconnected enables closed-loop transmission or reception, allowing self-diagnosis. To supply an IC card with the clock pulses generated by the SCI, input the SCKn pin output to the CLK pin of an IC card.

An output port of the MCU can be used to output a reset signal.

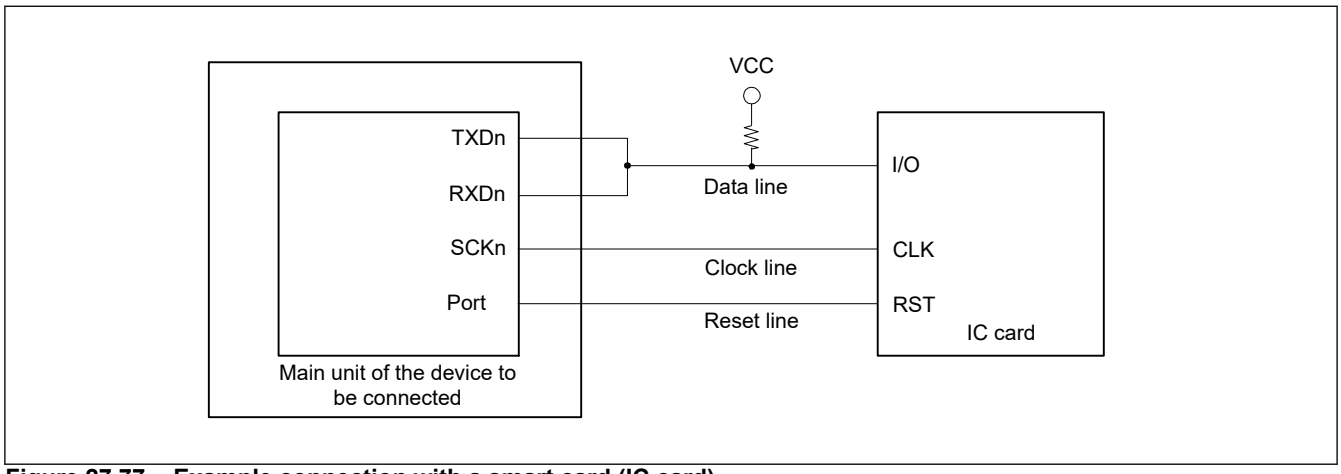



**Figure 27.77 Example connection with a smart card (IC card)**

### 27.7.2 Data Format (Except in Block Transfer Mode)

Figure 27.78 shows the data transfer formats in smart card interface mode:

- One frame consists of 8-bit data and a parity bit in asynchronous mode.
- During transmission, at least 2 etus (elementary time unit – the time required for transferring 1 bit) is set as a guard time from the end of the parity bit until the start of the next frame.
- If a parity error is detected during reception, a low error signal is output for 1 etu after 10.5 etus elapse from the start bit.
- If an error signal is sampled during transmission, the same data is automatically retransmitted after at least 2 etus.

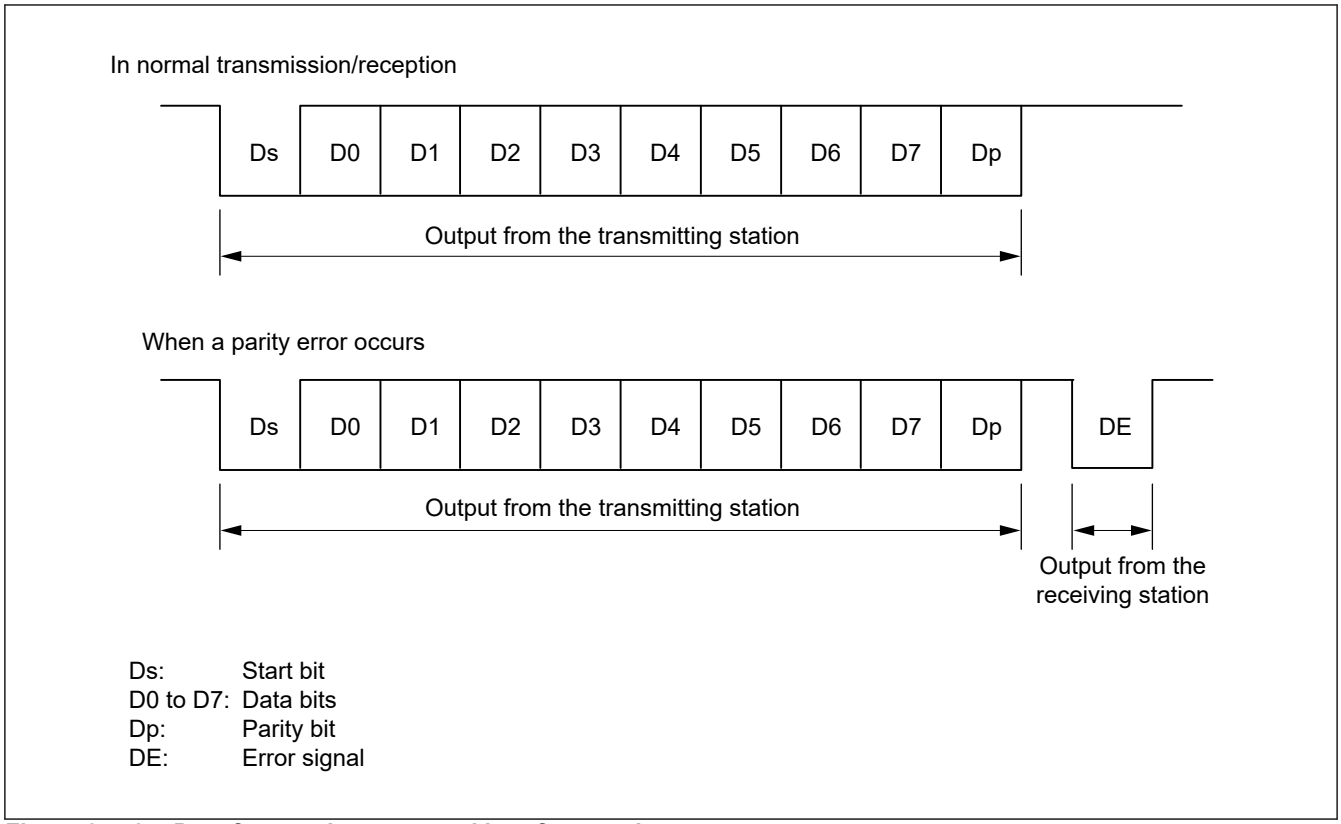



**Figure 27.78 Data formats in smart card interface mode**

For communications with IC cards of the direct convention type and inverse convention type, follow the procedures in this section.

### (1) Direct Convention Type

For the direct convention type, logic levels 1 and 0 indicate the Z and A states, respectively, and data is transferred with LSB-first for the start character, as shown in Figure 27.79. Therefore, data in the start character in the figure is 0x3B.

When using the direct convention type, write 0 to both the SCMR.SDIR and SCMR.SINV bits. Write 0 to the SMR_SMCI.PM bit to use even parity, which is prescribed by the smart card standard.

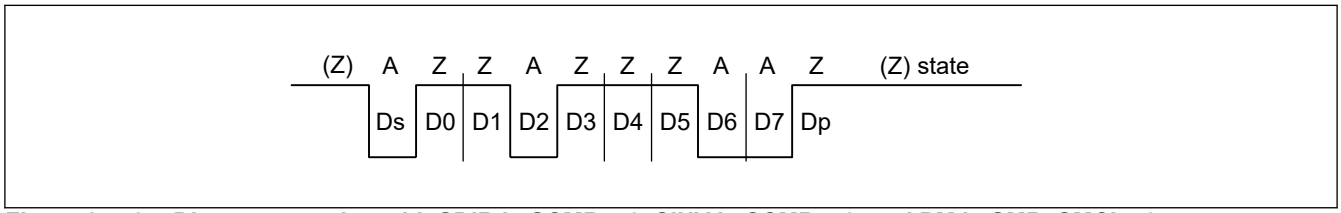



**Figure 27.79 Direct convention with SDIR in SCMR = 0, SINV in SCMR = 0, and PM in SMR_SMCI = 0**

### (2) Inverse Convention Type

For the inverse convention type, logic levels 1 and 0 indicate the A and Z states, respectively, and data is transferred with MSB-first for the start character, as shown in Figure 27.80. Therefore, data in the start character in the figure is 0x3F.

When using the inverse convention type, write 1 to both the SCMR.SDIR and SCMR.SINV bits. The parity bit is logic level 0 to produce even parity, which is prescribed by the smart card standard, and corresponds to the Z state. Because the SINV bit of the MCU only inverts data bits D7 to D0, write 1 to the PM bit in SMR_SMCI to invert the parity bit for both transmission and reception.

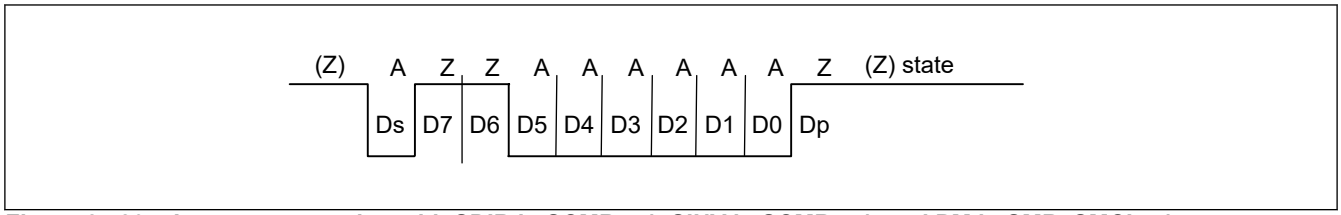



**Figure 27.80 Inverse convention with SDIR in SCMR = 1, SINV in SCMR = 1, and PM in SMR_SMCI = 1**

### 27.7.3 Block Transfer Mode

Block transfer mode differs from normal smart card interface mode as follows:

- Even if a parity error is detected during reception, no error signal is output. Because the PER flag in SSR_SMCI is set by error detection, clear the PER flag before receiving the parity bit of the next frame.
- During transmission, at least 1 etu is set as a guard time from the end of the parity bit until the start of the next frame
- Because the same data is not retransmitted, the TEND flag in SSR_SMCI is set to 11.5 etus after transmission starts
- In block transfer mode, the ERS flag in SSR_SMCI indicates the error signal status as in normal smart card interface mode, but the flag is read as 0 because no error signal is transferred

### 27.7.4 Receive Data Sampling Timing and Reception Margin

Only the internal clock generated by the on-chip baud rate generator can be used as a transfer clock in smart card interface mode.

In this mode, the SCI can operate on a base clock with a frequency of 32, 64, 372, 256, 93, 128, 186, or 512 times the bit rate set up in the SCMR.BCP2 and the SMR_SMCI.BCP[1:0] bits. The frequency is always 16 times the bit rate in normal asynchronous mode.

For data reception, the falling edge of the start bit is sampled with the base clock to perform internal synchronization.

Receive data is sampled on the 16th, 32nd, 186th, 128th, 46th, 64th, 93rd, and 256th rising edges of the base clock so that it can be latched at the middle of each bit as shown in Figure 27.81. The reception margin is determined by the following formula:

$$M = \left|\left(0.5 - \frac{1}{2N}\right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F)\right| \times 100\ [\%]$$

M: Reception margin (%)

N: Ratio of bit rate to clock ($N = 32, 64, 372, 256$)

D: Duty cycle of clock ($D = 0$ to $1.0$)

L: Frame length ($L = 10$)

F: Absolute value of clock frequency deviation

Assuming values of F = 0, D = 0.5, and N = 372 in the specified formula, the reception margin is determined using the following formula:

$$M = \{0.5 - 1/(2 \times 372)\} \times 100\ [\%] = 49.866\ \%$$

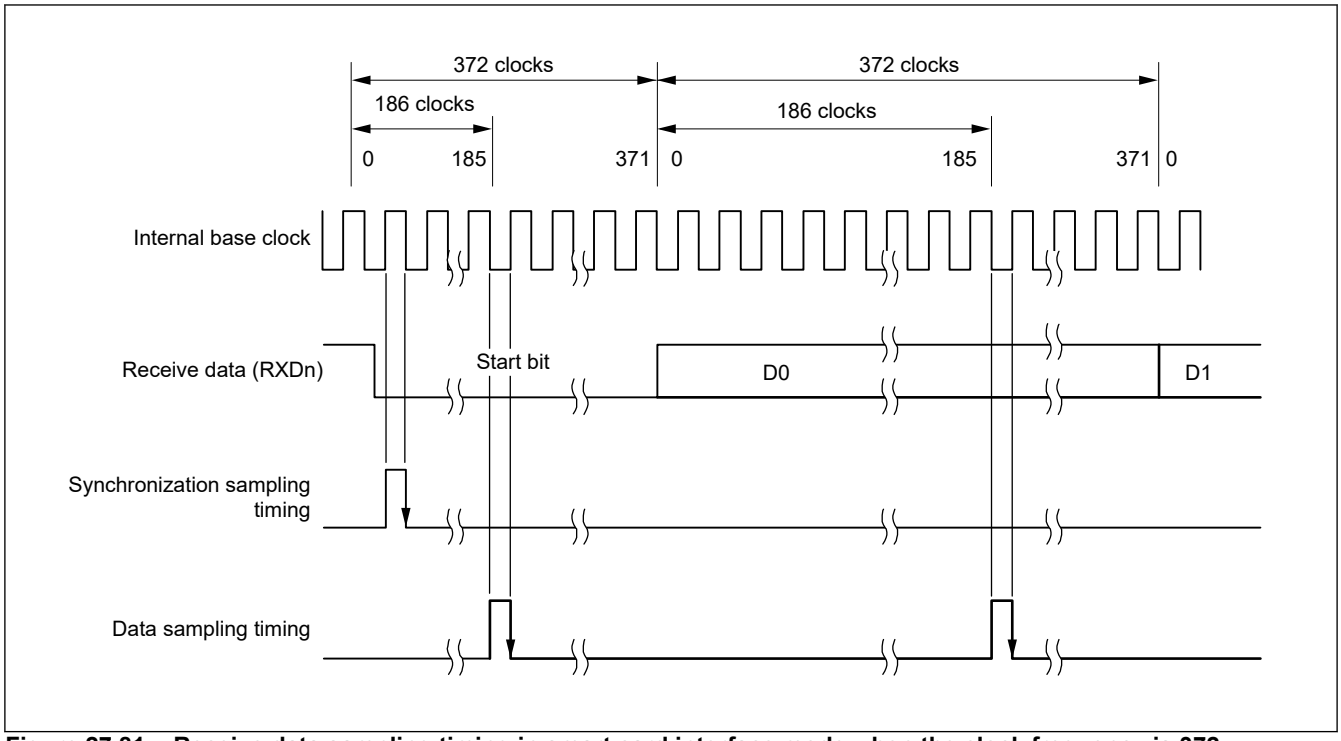



**Figure 27.81 Receive data sampling timing in smart card interface mode when the clock frequency is 372 times the bit rate**

## 27.7.5 SCI Initialization (Smart Card Interface Mode)

Before transmitting and receiving data, write the initial value 0x00 in the SCR_SMCI register and initialize the SCI following the example flow shown in Table 27.39.

Always set the initial value in the TIE, RIE, TE, RE, TEIE bits in the SCR_SMCI register before switching from transmission to reception mode or from reception to transmission mode. When SCR_SMCI.RE is set to 0, the RDR register is not initialized.

To change from reception mode to transmission mode, first check that reception has completed, then initialize the SCI. At the end of initialization, set SCR_SMCI.TE = 1 and SCR_SMCI.RE = 0. Reception completion can be verified by reading the SCIn_RXI request, ORER, or PER flag in SSR_SMCI.

To change transmission mode to reception mode, first check that transmission has completed, then initialize the SCI. At the end of initialization, set SCR_SMCI.TE = 0 and SCR_SMCI.RE = 1. Transmission completion can be verified by reading the TEND flag in SSR_SMCI.

**Table 27.39 Example flow of SCI initialization in smart card interface mode (1 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 1 | Start initialization | |
| 2 | Set SCR_SMCI.TIE, RIE, TE, RE, TEIE, and CKE[1:0] to 0 | Stop the communication and initialize SKE[1:0]. |
| 3 | Set SIMR1.IICM bit to 0.<br>Set SCMR.SMIF to 1. | Set to smart card interface mode. |
| 4 | Set SSR_SMCI.ORER, ERS, PER to 0 | Write to SSR_SMCI after reading SSR_SMCI. |
| 5 | Set SPMR.CKPH, CKPOL | Set the transmission or reception format in SPMR. |
| 6 | Set SMR_SMCI.GM, BLK, PM, BCP[1:0], CKS[1:0], and set SMR_SMCI.PE to 1 | Set the operation mode and the transmission or reception format in SMR_SMCI. |
| 7 | Set SCMR.BCP2, SDIR, SINV | Set the transmission or reception format in SCMR. |
| 8 | Set SPTR to the initial value. | Set the Initial value to SPTR. |

**Table 27.39 Example flow of SCI initialization in smart card interface mode (2 of 2)**

| No. | Step Name | Description |
|---|---|---|
| 9 | Set SEMR.BRME and SEMR.RXDESEL to 0 | Set SEMR.BRME and SEMR.RXDESEL to 0. |
| 10 | Set a value in BRR | Write the value for the bit rate in BRR. |
| 11 | Set the I/O port functions | Set the I/O port functions for TXDn, RXDn, and SCKn. |
| 12 | Set a value in SCR_SMCI.CKE[1:0] | Set the SCR_SMCI.CKE[1:0]. Even though the function depends on SMR_SMCI.GM, when the CKE[0] bit is set to 1, the clock is output from the SCKn pin. |
| 13 | Set SCR_SMCI.TE or RE to 1, and set SCR_SMCI.TIE, RIE | Set the TE or RE bit in SCR_SMCI to 1, then set the TIE and RIE bits in SCR_SMCI. Do not simultaneously set the TE and RE bits to 1 if self-diagnosis is not used. |
| 14 | Initialization completed | |

### 27.7.6 Serial Data Transmission (Except in Block Transfer Mode)

Serial data transmission in smart card interface mode (except in block transfer mode) is different from that in non-smart card interface mode, in that an error signal is sampled and data can be re-transmitted in smart card mode. Figure 27.82 shows the data re-transfer operation during transmission.

1. When an error signal from the receiver end is sampled after 1-frame data is transmitted, the SSR_SMCI.ERS flag is set to 1. If the SCR_SMCI.RIE bit is 1, an SCIn_ERI interrupt request is generated. Clear the ERS flag to 0 before the next parity bit is sampled.
2. For a frame in which an error signal is received, the SSR_SMCI.TEND flag is not set. Data is re-transferred from TDR to TSR, allowing automatic data retransmission.
3. If no error signal is returned from the receiver, the ERS flag is not set to 1.
4. In this case, the SCI determines that transmission of 1-frame data, including the re-transfer, is complete, and the TEND flag is set. If the SCR_SMCI.TIE bit is 1, an SCIn_TXI interrupt request is generated. Write transmit data to the TDR to start transmission of the next data.

Figure 27.84 shows an example flow of serial transmission. All the processing steps are automatically performed using an SCIn_TXI interrupt request to activate the DTC or DMAC.

When the SSR_SMCI.TEND flag is set to 1 in transmission and when the SCR_SMCI.TIE bit is 1, an SCIn_TXI interrupt request is generated.

The DTC or DMAC is activated by an SCIn_TXI interrupt request if the SCIn_TXI interrupt request is previously specified as a source of DTC or DMAC activation, allowing the transfer of transmit data. The TEND flag is automatically set to 0 when the DTC or DMAC transfers the data.

If an error occurs, the SCI automatically retransmits the same data. During this retransmission, the TEND flag is kept at 0 and the DTC or DMAC is not activated. Therefore, the SCI and DTC or DMAC automatically transmit the specified number of bytes, including retransmission when an error occurs. Because the ERS flag is not automatically cleared, set the RIE bit to 1 before enabling an SCIn_ERI interrupt request to be generated if an error occurs, and clear the ERS flag to 0.

When transmitting or receiving data using the DTC or DMAC, always enable the DTC or DMAC before making the SCI settings.

For DTC or DMAC settings, see section 17, Data Transfer Controller (DTC), section 16, DMA Controller (DMAC).

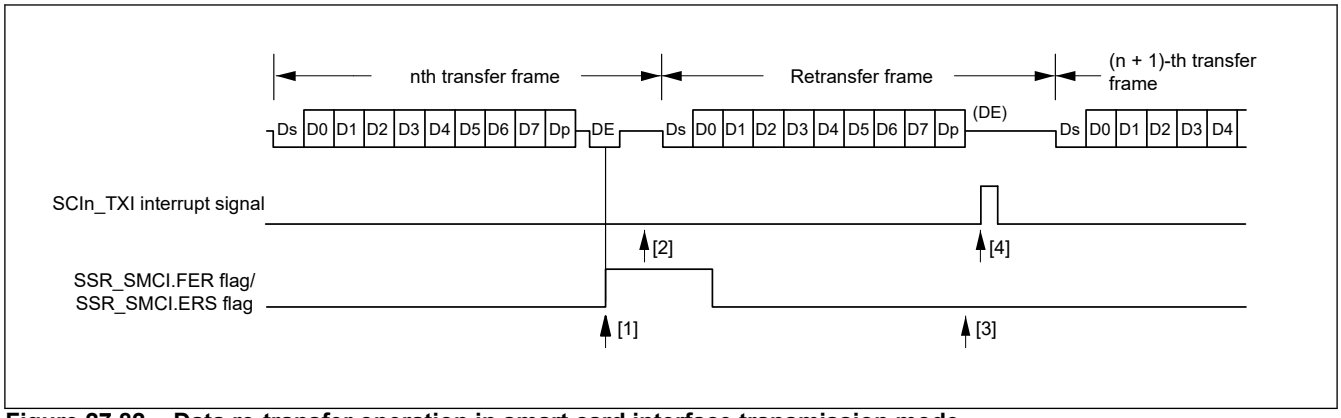



**Figure 27.82 Data re-transfer operation in smart card interface transmission mode**

The SSR_SMCI.TEND flag is set at different timings depending on the SMR_SMCI.GM bit setting. Figure 27.83 shows the TEND flag generation timing.

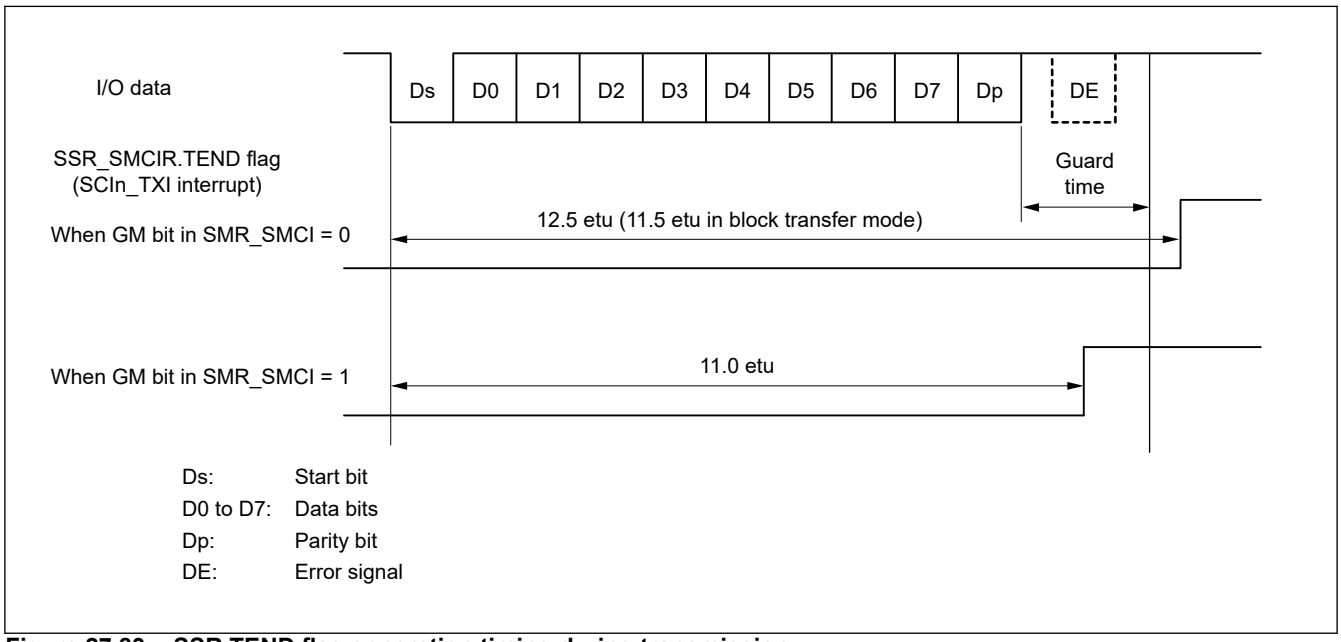



**Figure 27.83 SSR.TEND flag generation timing during transmission**

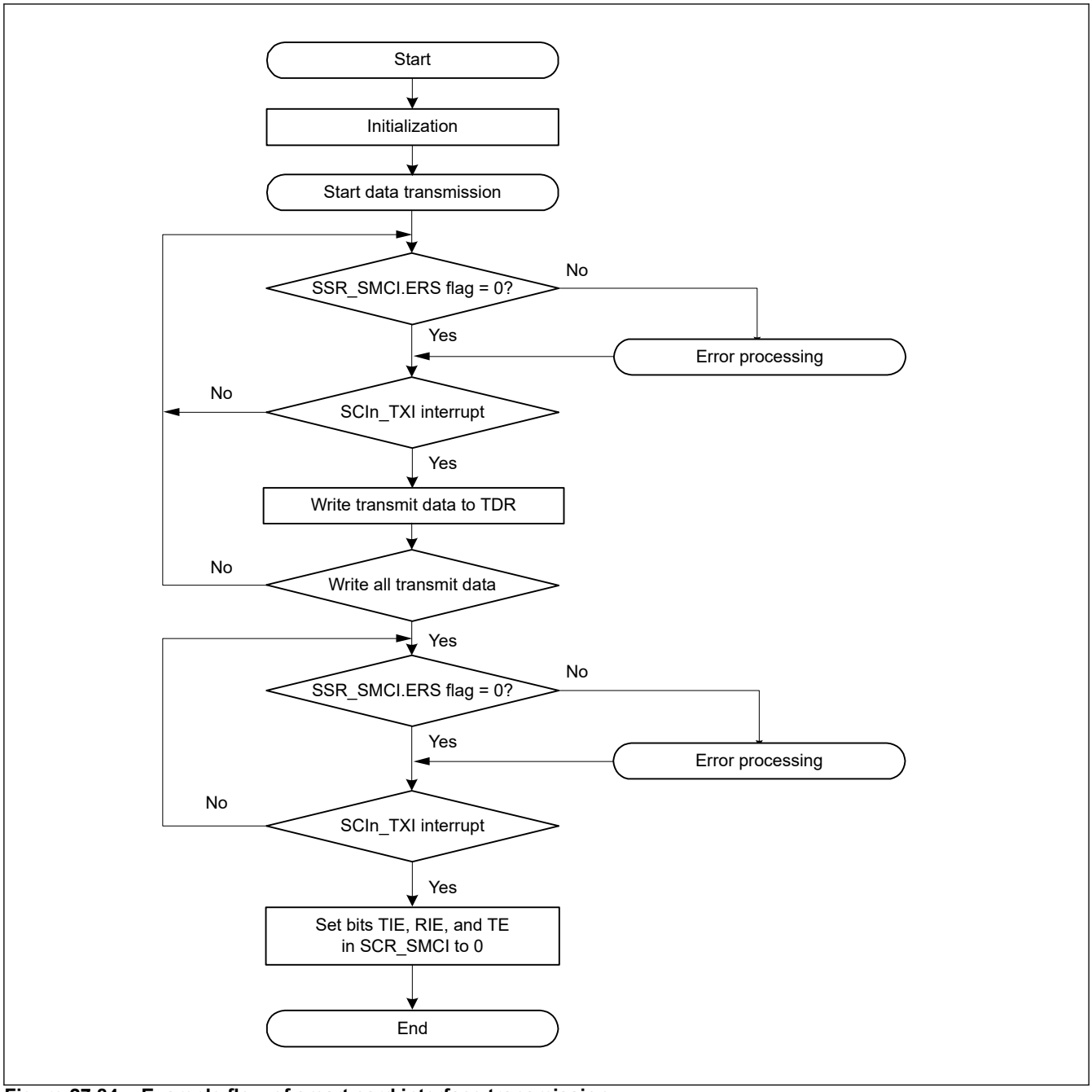



**Figure 27.84 Example flow of smart card interface transmission**

## 27.7.7 Serial Data Reception (Except in Block Transfer Mode)

Serial data reception in smart card interface mode is similar to that in non-smart card interface mode. Figure 27.85 shows the data re-transfer operation in reception mode.

1. If a parity error is detected in the receive data, the SSR_SMCI.PER flag is set to 1. When the SCR_SMCI.RIE bit is 1, an SCIn_ERI interrupt request is generated. Clear the PER flag to 0 before the next parity bit is sampled.
2. For a frame in which a parity error is detected, no SCIn_RXI interrupt is generated.
3. When no parity error is detected, the SCR_SMCI.PER flag is not set to 1.
4. In this case, data is determined to be received successfully. When the SCR_SMCI.RIE bit is 1, an SCIn_RXI interrupt request is generated.

Figure 27.86 shows an example flow of serial data reception. All the processing steps are automatically performed using an SCIn_RXI interrupt request to activate the DTC or DMAC.

In reception, setting the RIE bit to 1 allows an SCIn_RXI interrupt request to be generated. The DTC or DMAC is activated by an SCIn_RXI interrupt request if the SCIn_RXI interrupt request is previously specified as a source of DTC or DMAC activation, allowing the transfer of receive data.

If an error occurs during reception and either the ORER or PER flag in SSR_SMCI is set to 1, a receive error interrupt (SCIn_ERI) request is generated. Clear the error flag after the error occurrence. If an error occurs, the DTC or DMAC is not activated and receive data is skipped. Therefore, the number of bytes of receive data specified in the DTC or DMAC is transferred.

If a parity error occurs and the PER flag is set to 1 during reception, the receive data is transferred to RDR, allowing the data to be read.

When a reception is forced to terminate by setting SCR_SMCI.RE to 0 during operation, read the RDR register because the received data that is not yet read might be left in the RDR.

Note: For operations in block transfer mode, see section 27.3.9. Serial Data Reception in Asynchronous Mode.

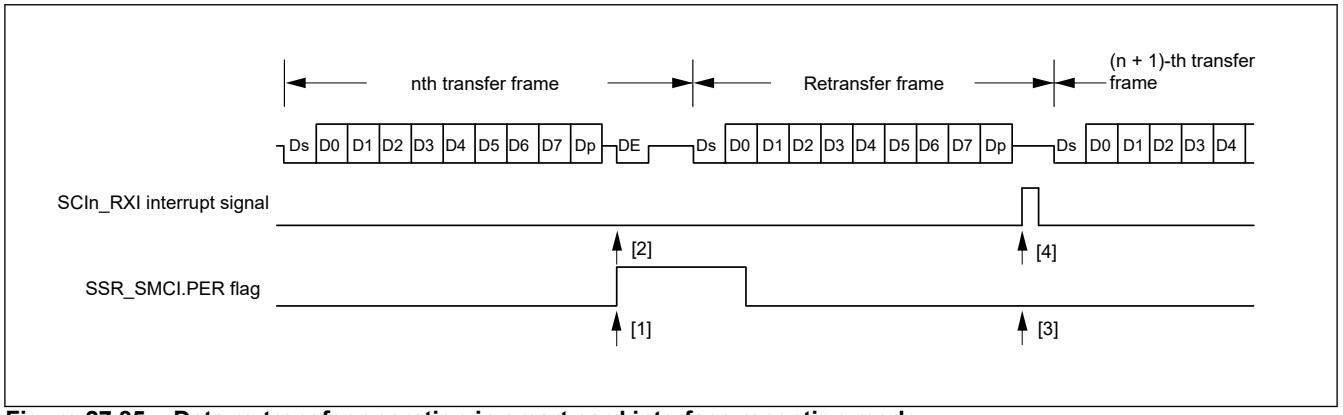



**Figure 27.85 Data re-transfer operation in smart card interface reception mode**

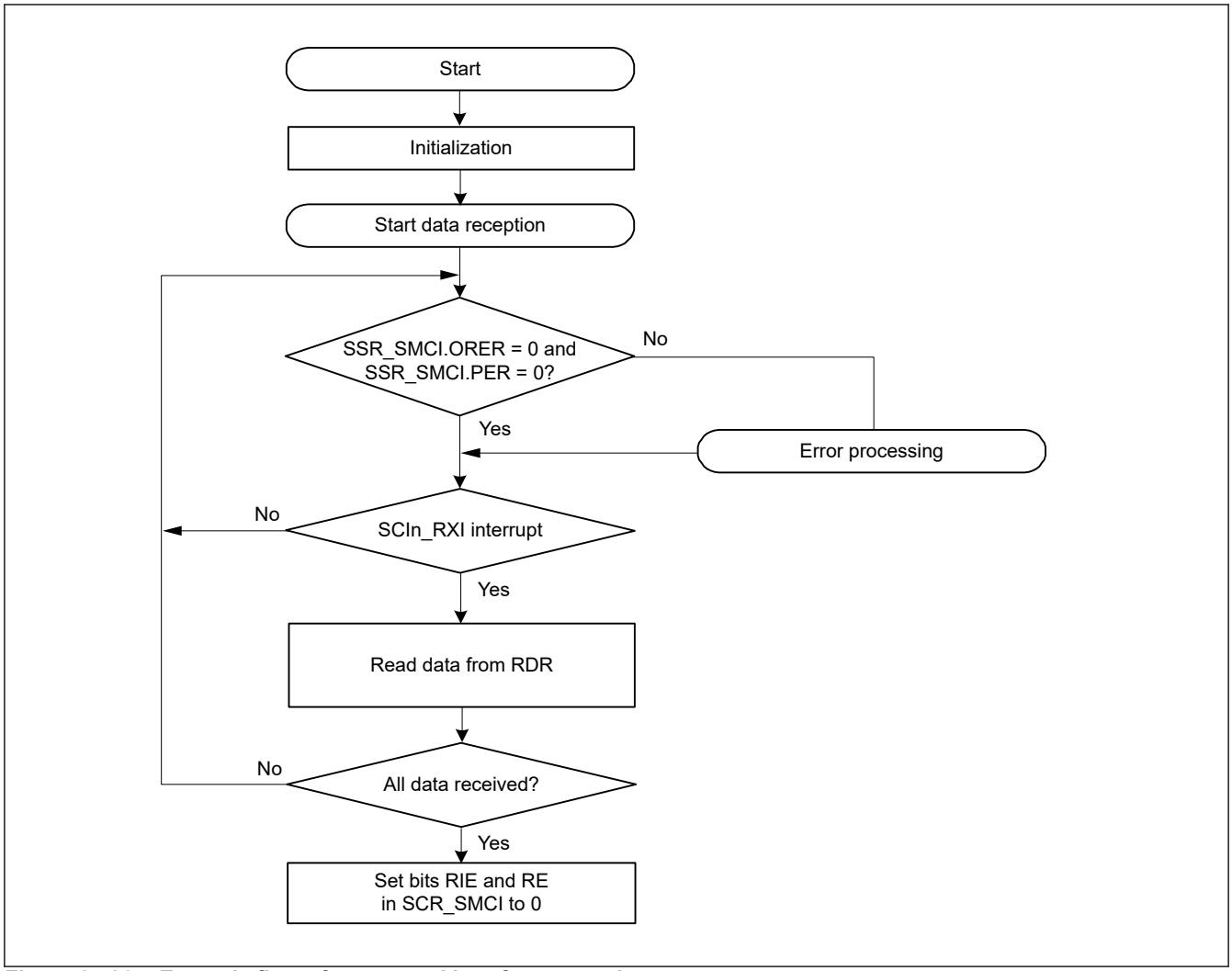



**Figure 27.86 Example flow of smart card interface reception**

## 27.7.8 Clock Output Control

When the GM bit in SMR_SMCI is set to 1, the clock output can be controlled by the CKE[1:0] bits in SCR_SMCI. For details on the CKE[1:0] bits, see section 27.2.14. SCR_SMCI : Serial Control Register for Smart Card Interface Mode (SCMR.SMIF = 1). When setting the clock output, the base clock described in section 27.7.4. Receive Data Sampling Timing and Reception Margin is applied.

Figure 27.87 shows an example timing for the clock output control when the CKE[1] bit in SCR_SMCI is set to 0 and the CKE[0] bit in SCR_SMCI is controlled.

When the GM bit in SMR_SMCI is 0, output control by the CKE[0] bit in SCR_SMCI is immediately reflected on the SCKn pin, so there is a possibility that pulses with an unintended width may be output from the SCKn pin.

When the GM bit in SMR_SMCI is 1, the clock with the same pulse width as the base clock is output even if the CKE[0] bit in SCR_SMCI is changed.

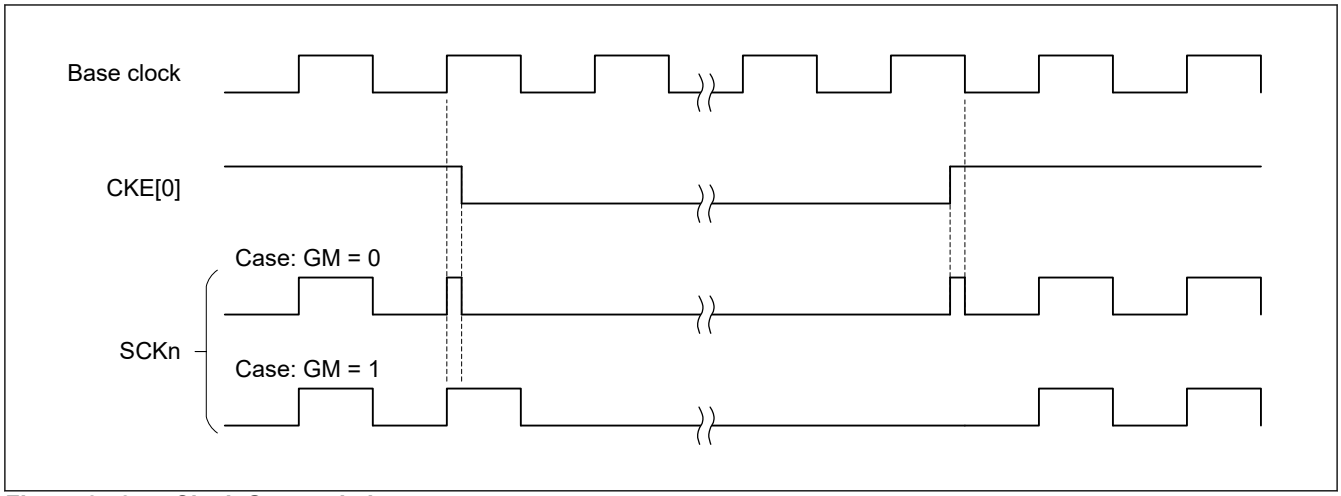



**Figure 27.87 Clock Output timing**

## 27.8 Operation in Simple IIC Mode

Simple IIC mode format is composed of 8 data bits and an acknowledge bit. By continuing into a slave-address frame after a start condition or restart condition, a master device can specify a slave device as the partner for communications. The currently specified slave device remains valid until a new slave device is specified or a stop condition is satisfied. The 8 data bits in all frames are transmitted in order from the MSB.

The $I^2C$ bus format and timing of the $I^2C$ bus are shown in Figure 27.88 and Figure 27.89.

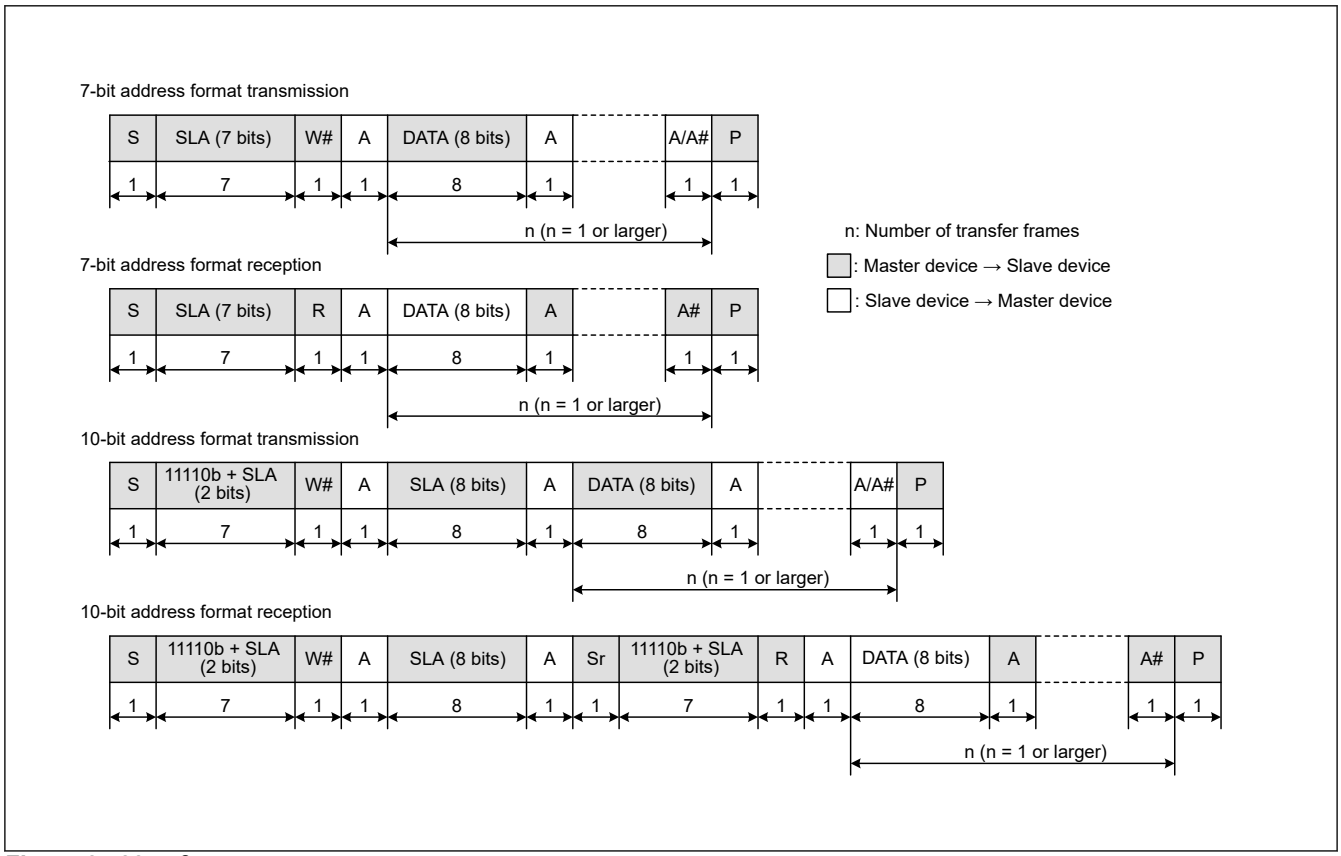



**Figure 27.88 $I^2C$ bus format**

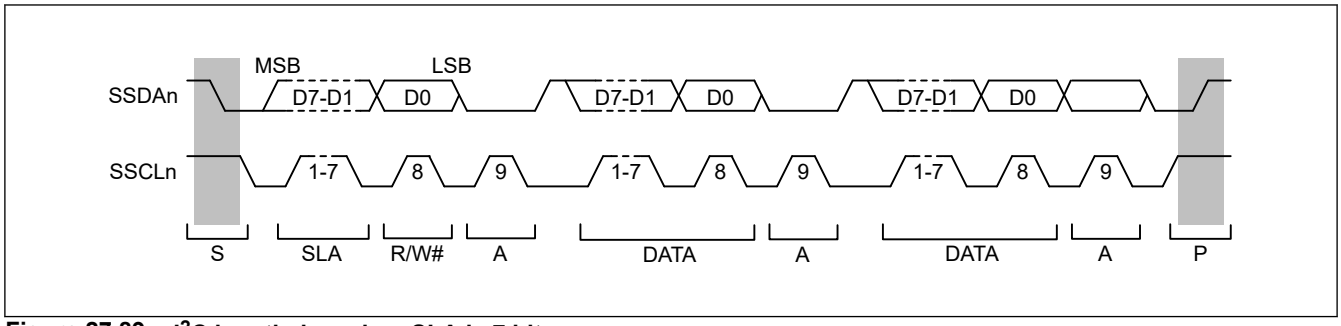



**Figure 27.89 I²C bus timing when SLA is 7 bits**

- S: Indicates a start condition, when the master device changes the level on the SDAn line from high to low while the SCLn line is high
- SLA: Indicates a slave address, by which the master device selects a slave device
- R/W#: Indicates the direction of transfer (reception or transmission). The value 1 indicates transfer from the slave device to the master device and 0 indicates transfer from the master device to the slave device.
- A/A#: Indicates an acknowledge bit. This is returned by the slave device for master transmission and by the master device for master reception. Return low indicates ACK and return high indicates NACK.
- Sr: Indicates a restart condition, when the master device changes the level on the SDAn line from high to low while the SCLn line is high and after the setup time elapses
- DATA: Indicates the data being received or transmitted
- P: Indicates a stop condition, when the master device changes the level on the SDAn line from low to high while the SCLn line is high

### 27.8.1 Generation of Start, Restart, and Stop Conditions

Writing 1 to the SIMR3.IICSTAREQ bit causes the generation of a start condition. The generation of a start condition proceeds through the following operations:

- The level on the SDAn line falls (from the high level to the low level) and the SCLn line is kept in the released state
- The hold time for the start condition is set as half of a bit period at the bit rate determined by the BRR setting
- The level on the SCLn line falls (from the high level to the low level), the IICSTAREQ bit in SIMR3 is set to 0, and a start-condition generated interrupt is output

Writing 1 to the IICRSTAREQ bit in SIMR3 causes the generation of a restart condition. The generation of a restart condition proceeds through the following operations:

- The SDAn line is released and the SCLn line is kept at the low level
- The period at low level for the SCLn line is set as half of a bit period at the bit rate determined by the BRR setting
- The SCLn line is released (transition from the low to the high level)
- When a high level is detected on the SCLn line, the setup time for the restart condition is set as half of a bit period at the bit rate determined by the BRR setting
- The level on the SDAn line falls (from the high level to the low level)
- The hold time for the restart condition is set as half of a bit period at the bit rate determined by the BRR setting
- The level on the SCLn line falls (from the high level to the low level), the SIMR3.IICRSTAREQ bit is set to 0, and a restart-condition generated interrupt is output

Writing 1 to the SIMR3.IICSTPREQ bit causes the generation of a stop condition. The generation of a stop condition proceeds through the following operations:

- The level on the SDAn line falls (from the high level to the low level) and the SCLn line is kept at the low level
- The period at low level for the SCLn line is set as half of a bit period at the bit rate determined by the BRR setting
- The SCLn line is released (transition from the low to the high level)

- When a high level is detected on the SCLn line, the setup time for the stop condition is set as half of a bit period at the bit rate determined by the BRR setting
- The SDAn line is released (transition from the low to the high level), the SIMR3.IICSTPREQ bit is set to 0, and a stop-condition generated interrupt is output

Figure 27.90 shows the timing of operations in the generation of start, restart, and stop conditions.

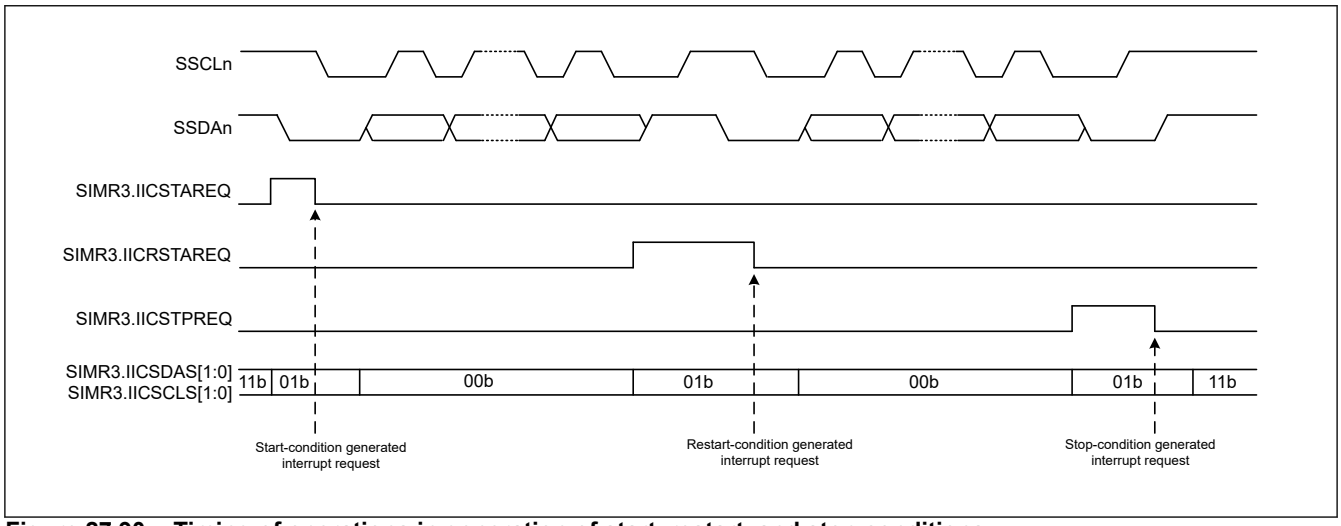



**Figure 27.90 Timing of operations in generation of start, restart, and stop conditions**

## 27.8.2 Clock Synchronization

The SCLn line can be driven low if a wait is inserted by a slave device at the other side of the transfer. Setting the SIMR2.IICCSC bit to 1 applies control to obtain synchronization when a difference arises between the levels of the internal SCLn clock signal and the level being input on the SCLn pin.

When the SIMR2.IICCSC bit is set to 1, the level of the internal SCLn clock signal changes from low to high. Counting to determine the period at a high level stops while the low level is being input on the SCLn pin. Counting to determine the period at a high level starts after the transition of the input on the SCLn pin to the high level.

The interval from this time until counting to determine the period at high level starts on the transition of the SCLn pin to the high level, is the total of the delay of SCLn output, delay for noise filtering of the input on the SCLn pin (2 or 3 cycles of sampling clock for the noise filter), and delay for internal processing (1 or 2 cycles of PCLK). The period at high level of the internal SCLn clock is extended even when other devices do not place the low level on the SCLn line.

If the SIMR2.IICCSC bit is 1, synchronization is obtained for the transmission and reception of data by taking the logical AND of the input on the SCLn pin and the internal SCLn clock. If the SIMR2.IICCSC bit is 0, synchronization with the internal SCLn clock is obtained for the transmission and reception of data.

If a slave device inserts a wait period into the interval until the transition of the internal SCLn clock signal from the low to the high level after a request for the generation of a start, restart, or stop condition is issued, the time until generation is prolonged by that period.

If a slave device inserts a wait period after the transition of the internal SCLn clock signal from the low to the high level, although the generation-completed interrupt is issued without stopping the waiting period, generation of the condition itself is not guaranteed.

Figure 27.91 shows an example operation for synchronizing the clocks.

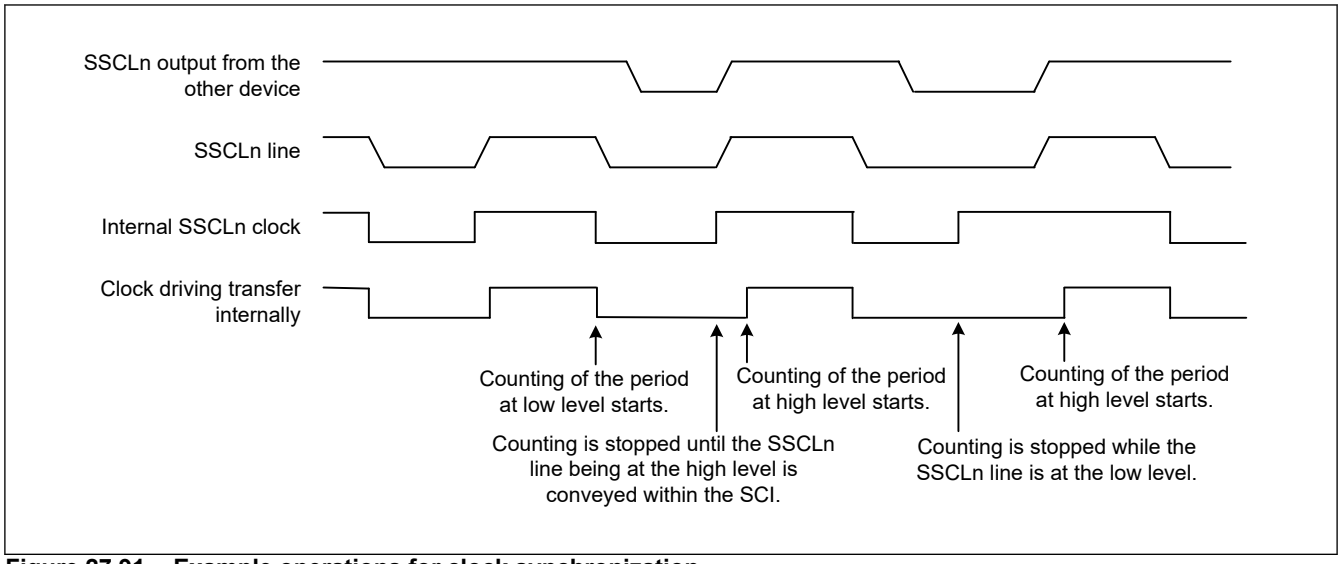



**Figure 27.91 Example operations for clock synchronization**

## 27.8.3 SDAn Output Delay

The SIMR1.IICDL[4:0] bits can be used to set a delay for output on the SDAn pin relative to falling edges of output on the SCLn pin. Delay settings from 0 to 31 are selectable, representing periods of the corresponding numbers of cycles of the clock signal from the on-chip baud rate generator (derived by frequency-dividing the base clock, PCLK, by the divisor selected in the SMR.CKS[1:0] bits). A delay for output on the SDAn pin applies to the start condition/restart condition/stop condition signal, 8-bit transmit data, and acknowledge bit.

If the SDAn output delay is shorter than the time for the level on the SCLn pin to fall, the change of the output on the SDAn pin starts while the output level on the SCLn pin is falling, creating a possibility of erroneous operation for slave devices. Ensure that settings for the delay of output on the SDAn pin specify times greater than the time output on the SCLn pin takes to fall (300 ns for IIC in normal mode and fast mode).

Figure 27.92 shows the timing of delays in SDAn output.

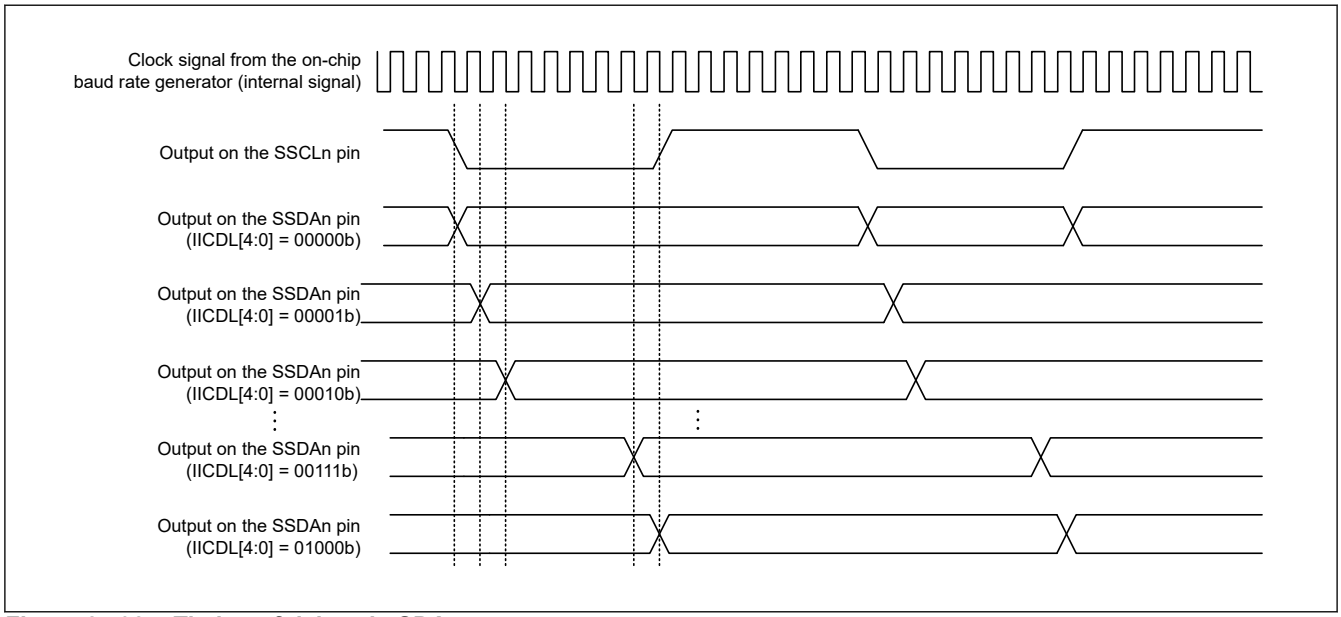



**Figure 27.92 Timing of delays in SDAn output**

## 27.8.4 SCI Initialization in Simple IIC Mode

Before transferring data, write the initial value 0x00 to SCR and initialize the interface following the example shown in Table 27.40.

Before making any changes to the operating mode or transfer format, be sure to set SCR to its initial value. In simple IIC mode, the open-drain setting for the communication ports should be made on the port side.

**Table 27.40 Example flow of SCI initialization in simple IIC mode**

| No. | Step Name | Description |
|---|---|---|
| 1 | Start of initialization | |
| 2 | Set the TIE, RIE, TE, RE, TEIE and CKE[1:0] bits in SCR to 0 | |
| 3 | Set the I/O port functions | Set the I/O port to allow use (on N-channel open-drain output pins) of the SSCLn and SSDAn pin functions. |
| 4 | Set the IICSDAS[1:0] and IICSCLS[1:0] bits in SIMR3 to 11b | Place the SSCLn and SSDAn pins in the high-impedance state until a start condition is to be generated. |
| 5 | Set up the transfer or reception format in SMR and SCMR | Set the format for transmission and reception in SMR and SCMR.<br>In SMR, set the CKS[1:0] bits to the target value and set the other bits to 0.<br>In SCMR, set the SDIR bit to 1 and the SINV and SMIF bits to 0. |
| 6 | Set the initial value to SPTR. | Set the Initial value to SPTR. |
| 7 | Set the value in BRR | Write the value for the targeted bit rate to BRR. |
| 8 | Set a value in MDDR | Write the value obtained by correcting a bit rate error in MDDR. This step is not required if the BRME bit in SEMR is set to 0. |
| 9 | Set the values in SEMR, SNFR, SIMR1, SIMR2, and SPMR | Set the values in SEMR, SNFR, SIMR1, SIMR2, and SPMR.<br>Set the NFEN and BRME bits in SEMR. In SNFR, set the NFCS[2:0] bits.<br>In SIMR1, set the IICM bit to 1 and the IICDL[4:0] bits as required.<br>In SIMR2, set the IICACKT and IICCSC bits to 1 and the IICINTM bits as required.<br>In SPMR, set all the bits to 0. |
| 10 | Set the SCR.RE and TE bit to 1 and set the SCR.TIE, RIE and TEIE bits | Set the RE and TE bits in the SCR to 1. Then, set the SCR.TIE, RIE, and TEIE bits (for transmission and when the SIMR2.IICINTM bit is 1, set the RIE bit to 0). Setting the TE and RE bits to 1 enables the SSCLn and SSDAn pin functions. |
| 11 | Start of transmission or reception | |

## 27.8.5 Operation in Master Transmission in Simple IIC Mode

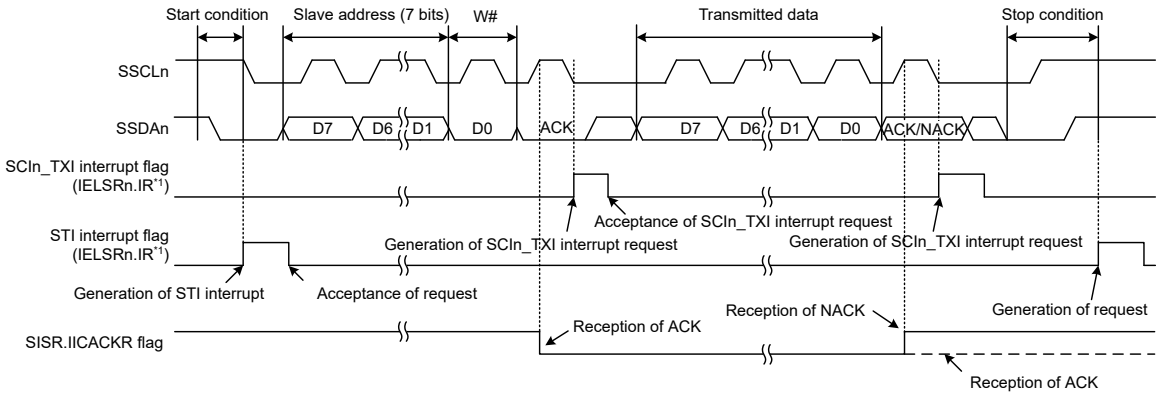
Figure 27.93 and Figure 27.94 show examples of master transmission and Figure 27.95 shows an example flow of data transmission.

Figure 27.93 shows the operation example when SIMR2.IICINTM bit is 1 (use reception and transmission interrupts) and the value of the SCR.RIE bit is assumed to be 0 (SCIn_RXI and SCIn_ERI interrupt requests are disabled).

See Table 27.45 for more information on the STI interrupt.

Figure 27.95 shows a flow chart in the case of SIMR2.IICINTM is 1 and address transmission by CPU and data transmission by DTC or DMAC. When 10-bit slave addresses are in use, steps [3] and [4] are repeated twice.

In simple IIC mode, the transmit data empty interrupt (SCIn_TXI) is generated when communication of one frame is complete, unlike the SCIn_TXI interrupt request generation timing during clock synchronous transmission.



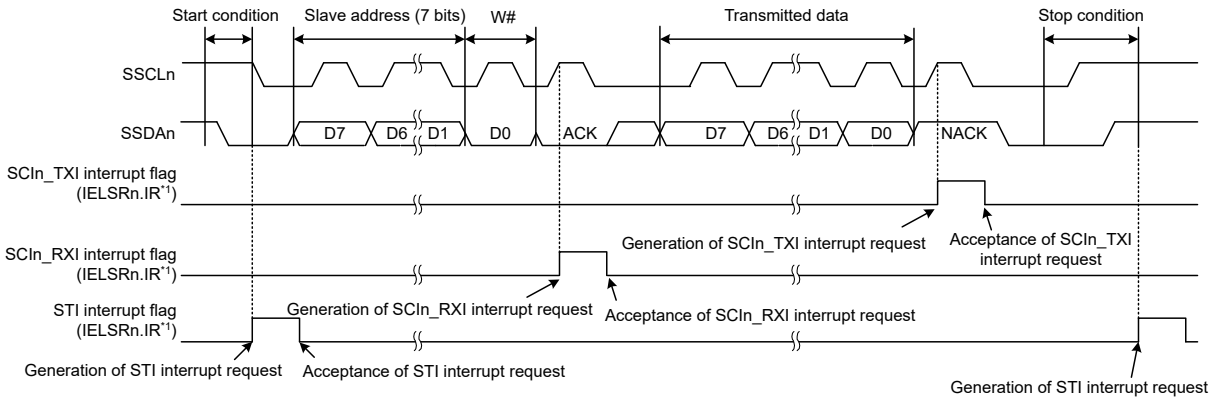

Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.93 Example 1 of operations for master transmission in simple IIC mode with 7-bit slave addresses, transmission interrupts, and reception interrupts**

When the SIMR2.IICINTM bit is set to 0 (use ACK/NACK interrupts) during master transmission, the DTC or DMAC is activated by the ACK interrupt as the trigger and required number of data bytes are transmitted. When the NACK is received, error processing such as transmission stop and retransmission is performed using the NACK interrupt as the trigger.

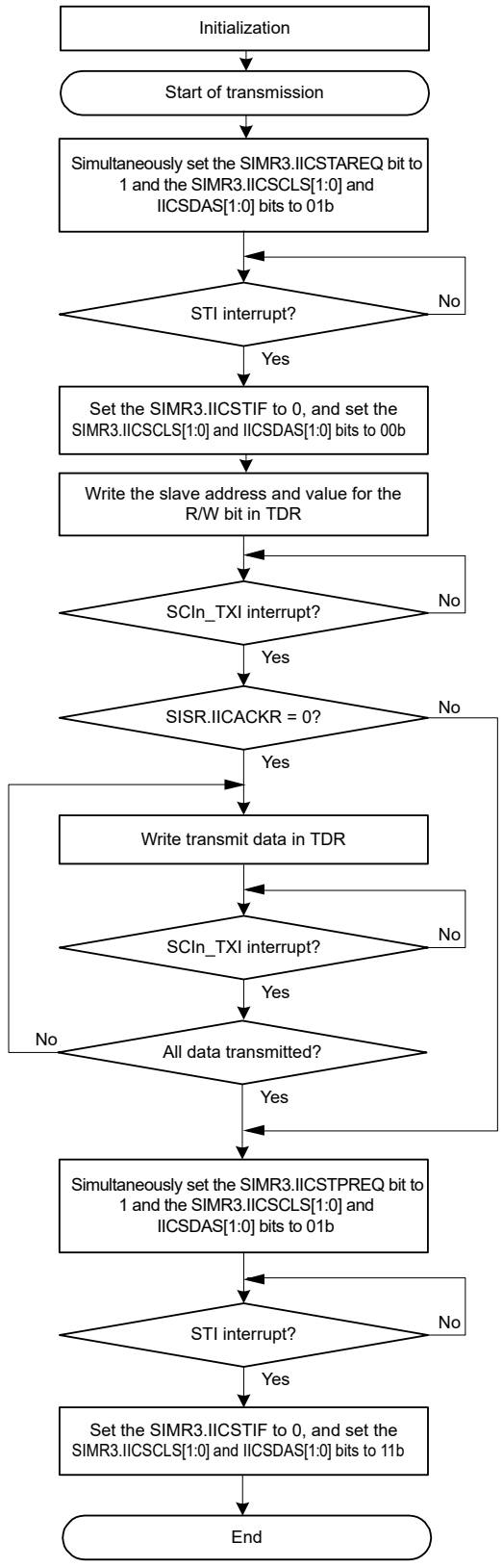
To restart communication for some reason after writing data in the TDR register, use the following procedure:

1. Set the TE and RE bits in the SCR register to 0 to stop communication.
2. Set 0xF0 in the SIMR3 register, release the $I^2C$ bus, and clear the generation of a condition.
3. If the RDRF flag in the SSR register is set to 1, clear it.
4. Set the TE and RE bits in the SCR register to 1 and start the next communication.




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.94 Example 2 of operations for master transmission in simple IIC mode with 7-bit slave addresses, ACK interrupts, and NACK interrupts**




[ 1 ] [ 1 ] Initialization in simple IIC mode:
For transmission, set the SCR.RIE to 0 to disable the SCIn_RXI and ERI interrupts.

[ 2 ] [ 2 ] Generate a start condition.

[ 3 ] [ 3 ] Writing to TDR:
Write the slave address and value for the R/W bit to TDR.

[ 4 ] [ 4 ] Confirming ACK response from the slave device:
Check the SISR.IICACKR bit. If SISR.IICACKR is 0, it indicates that the slave device responded with ACK and operations proceed. If SISR.IICACKR is 1, it indicates that there was no response from the slave device so the next transition is to generate a stop condition.

If 10-bit slave addresses are in use, processing of [ 3 ] and [ 4 ] is repeated twice.

[ 5 ] [ 5 ] Continuing with serial transmission:
When transmission is to continue, write additional transmit data to TDR. Except for the first data to be transmitted, a SCIn_TXI interrupt request can activate the DMAC or DTC to handle writing of data to TDR. But it cannot be confirmed status of ACK or NACK.

[ 6 ] [ 6 ] Generate a stop condition.

Note: In simple IIC mode, the SCIn_TXI interrupt is generated when communication completes.

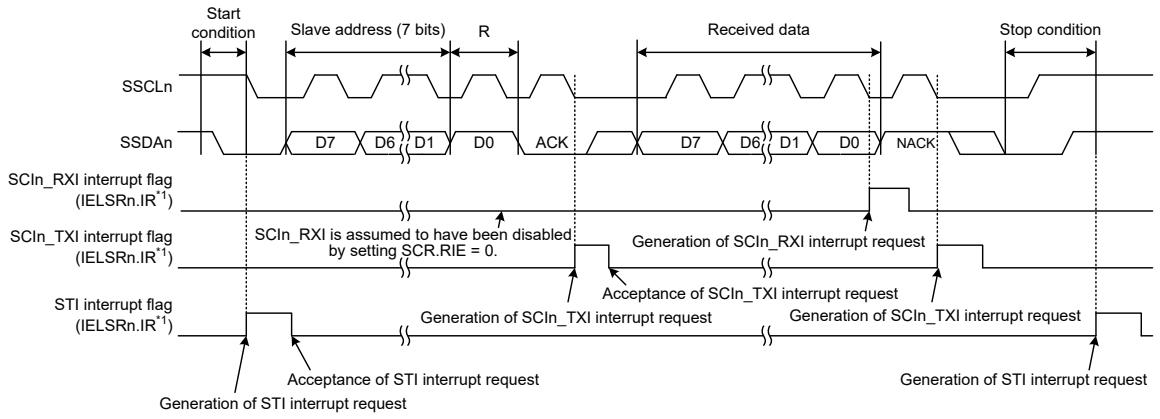
**Figure 27.95 Example flow of master transmission in simple IIC mode with transmission interrupts and reception interrupts**

## 27.8.6 Master Reception in Simple IIC Mode

Figure 27.96 shows an example operation in simple IIC mode master reception and Figure 27.97 shows an example flow of master reception.

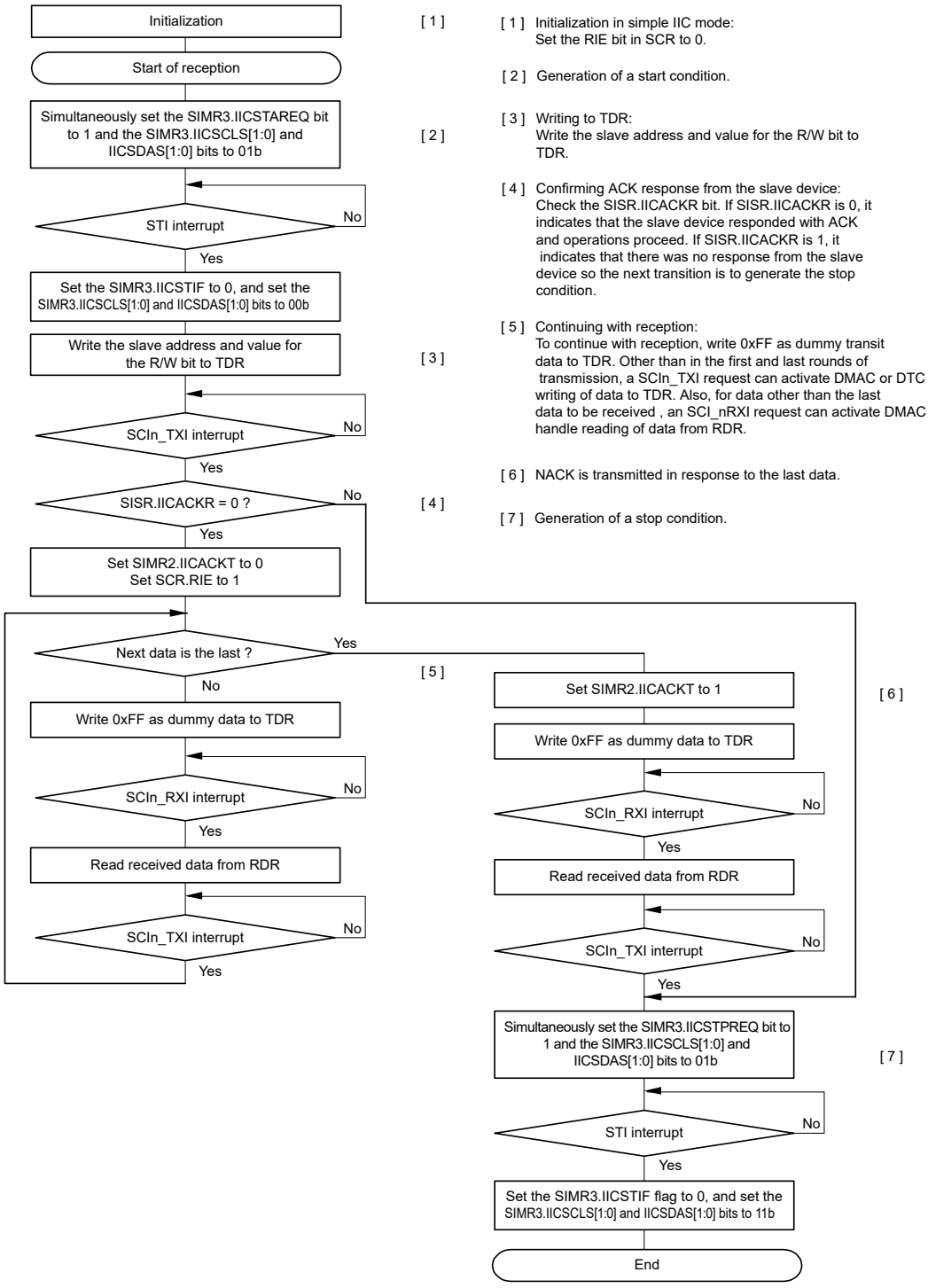
The value of the SIMR2.IICINTM bit is assumed to be 1 (use reception and transmission interrupts).

In simple IIC mode, the transmit data empty interrupt (SCIn_TXI) is generated when communication of one frame is complete, unlike the SCIn_TXI interrupt request generation timing during clock synchronous transmission.




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.96 Example operations for master reception in simple IIC mode with 7-bit slave addresses, transmission interrupts, and reception interrupts**




Note: In simple IIC mode, the SCIn_TXI interrupt request is generated when communication is complete.

**Figure 27.97 Example flow of master reception in simple IIC mode with transmission interrupts and reception interrupts**

## 27.9 Operation in Simple SPI Mode

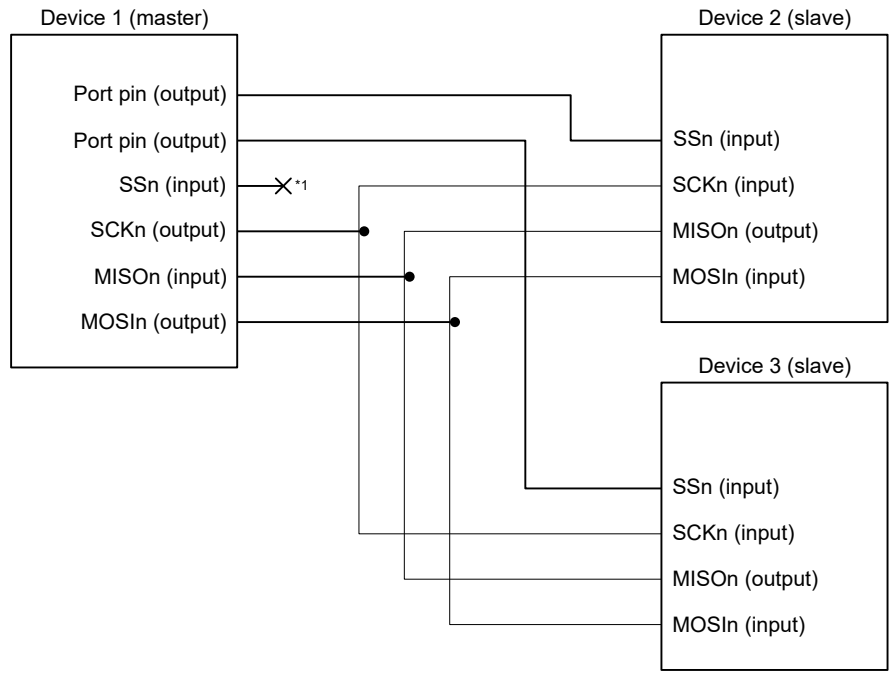
As an extended function, the SCI supports a simple SPI mode that handles transfer among one or multiple master devices and multiple slave devices.

Using the settings for clock synchronous mode (SCMR.SMIF = 0, SIMR1.IICM = 0, SMR.CM = 1) and setting the SPMR.SSE bit to 1 place the SCI in simple SPI mode. However, the SSn pin function on the master side is not required for connection of the device used as the master in simple SPI mode when the configuration only has a single master. Therefore, set the SPMR.SSE bit to 0 in such cases.

Figure 27.98 shows an example of connections for simple SPI mode. Control a general port pin to produce the SSn output signal from the master.

In simple SPI mode, data is transferred in synchronization with clock pulses in the same way as in clock synchronous mode. One character of data for transfer consists of 8 bits of data, and parity bits cannot be appended. The data can be inverted by setting the SCMR.SINV bit to 1.

Because the receiver and transmitter are independent of each other within the SCI module, full-duplex communications are possible, with a shared clock signal. Additionally, because both the transmitter and receiver have a buffered structure, writing the next transmit data while transmission is in progress and reading previously received data while reception is in progress are both possible. This enables continuous transfer.




Note 1. The SSn pin input is not required in a single-master system (the interface is used with the setting SPMR.SSE = 0).

**Figure 27.98 Example connections using simple SPI mode in single master mode with SPMR.SSE bit = 0**

## 27.9.1 States of Pins in Master and Slave Modes

The direction (input or output) of pins for the simple SPI mode interface differs according to whether the device is a master (SCR.CKE[1:0] = 00b or 01b and SPMR.MSS = 0) or slave (SCR.CKE[1:0] = 10b or 11b and SPMR.MSS = 1).

Table 27.41 lists the relationship between the pin states, mode, and level on the SSn pin.

**Table 27.41 States of pins by mode and input level on SSn pin (1 of 2)**

| Mode | Input on SSn pin | State of MOSIn pin | State of MISOn pin | State of SCKn pin |
|---|---|---|---|---|
| Master mode*1 | High level (transfer can proceed) | Output for data transmission*2 | Input for received data | Clock output*3 |
| | Low level (transfer cannot proceed) | High-impedance | Input for received data (but disabled) | High-impedance |

**Table 27.41 States of pins by mode and input level on SSn pin (2 of 2)**

| Mode | Input on SSn pin | State of MOSIn pin | State of MISOn pin | State of SCKn pin |
|---|---|---|---|---|
| Slave mode | High level (transfer cannot proceed) | Input for received data (but disabled) | High-impedance | Clock input (but disabled) |
| | Low level (transfer can proceed) | Input for received data | Output for data transmission | Clock input |

Note 1. When there is only a single master (SPMR.SSE = 0), transfer is possible regardless of the input level on the SSn pin. This is equivalent to input of a high level on the SSn pin.
Note 2. The MOSIn pin output is in the high-impedance state when serial transmission is disabled (SCR.TE bit = 0).
Note 3. The SCKn pin output is in the high-impedance state when serial transmission is disabled (SCR.TE = 0 and SCR.RE = 0) in a multi-master configuration (SPMR.SSE = 1).

### 27.9.2 SS Function in Master Mode

Setting the CKE[1:0] bits in the SCR to 00b or 01b and the MSS bit in the SPMR to 0 selects master mode operation. The SSn pin is not used in single-master configurations (SPMR.SSE = 0), so transmission or reception can proceed regardless of the value of the SSn pin.

When the level on the SSn pin is high in a multi-master configuration (SPMR.SSE = 1), a master device outputs clock signals from the SCKn pin before starting transmission or reception to indicate that there are no other masters or another master is performing reception or transmission.

When the level on the SSn pin is low in a multi-master configuration (SPMR.SSE = 1), there are other masters, and a transmission or reception is in progress. The MOSIn output and SCKn pins are placed in the high-impedance state and starting transmission or reception is not possible. In addition, the value of the SPMR.MFF bit is 1, indicating a mode fault error. In a multi-master configuration, start error processing by reading SPMR.MFF flag. If a mode fault error occurs while transmission or reception is in progress, transmission or reception does not stop, but the MOSIn and SCKn outputs are in the high-impedance state after completion of the transfer.

Use a general port pin to produce the SS output signal from the master.

### 27.9.3 SS Function in Slave Mode

Setting the SCR.CKE[1:0] bits to 10b or 11b and the SPMR.MSS bit to 1 selects slave operation. When the SSn pin is high, the MISOn output pin is in the high-impedance state and clock input through the SCKn pin is ignored. When the SSn pin is low, clock input through the SCKn pin is valid and transmission or reception can proceed.

If the input on the SSn pin changes from low to high during transmission or reception, the MISOn output pin is placed in the high-impedance state. Meanwhile, the internal processing for transmission or reception continues at the rate of the clock input through the SCKn pin until processing for the character being transmitted or received is complete, after which it stops, and the appropriate interrupt (SCIn_TXI, SCIn_RXI, or SCIn_TEI) is generated.

### 27.9.4 Relationship between Clock and Transmit/Receive Data

The CKPOL and CKPH bits in the SPMR register can be used to set up the clock for use in transmission and reception in four different ways. The relation between the clock signal and the transmission and reception of data is shown in Figure 27.99. The relation is the same for both master and slave operation. This is the same as when the level on the SSn pin is high.

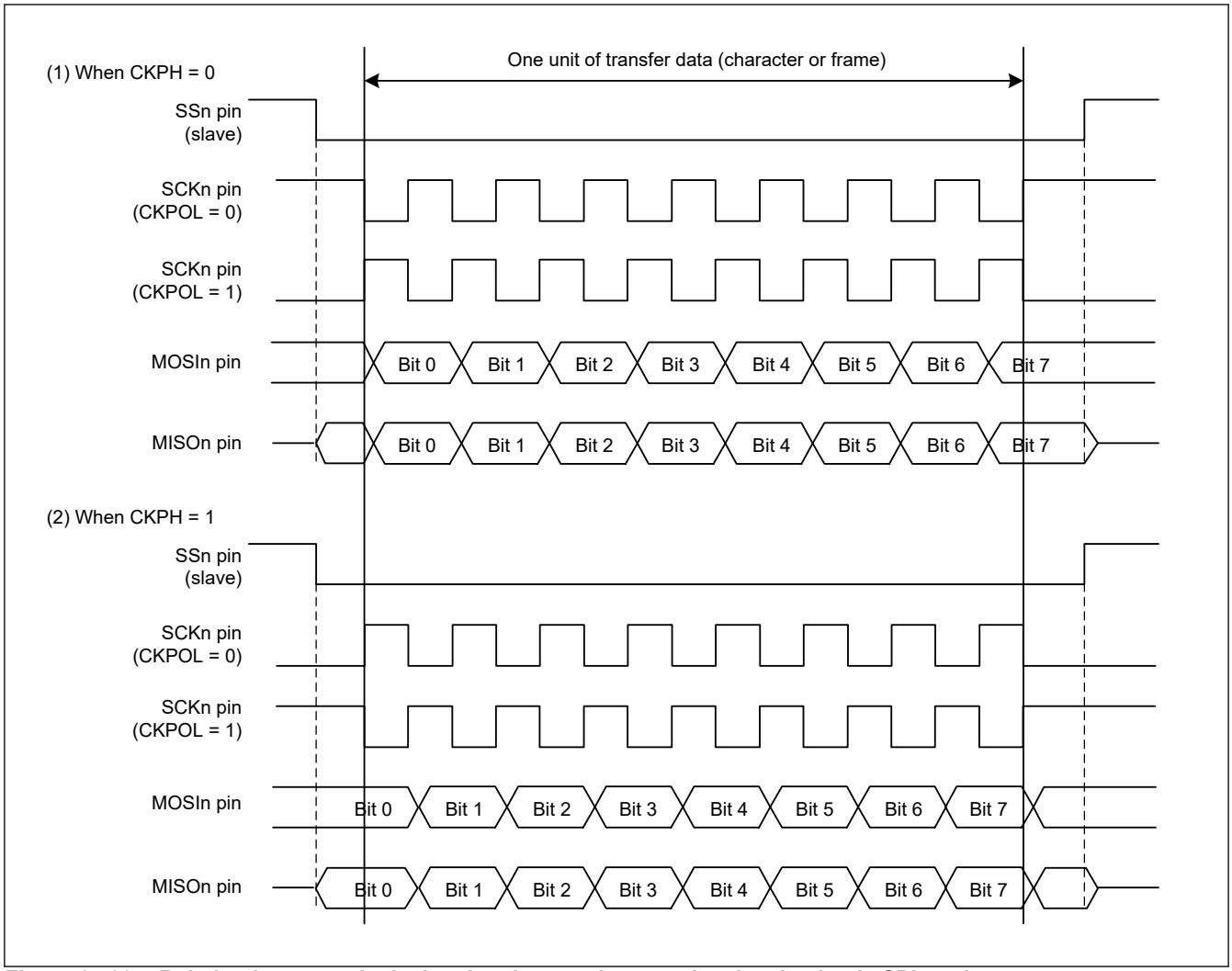



**Figure 27.99 Relation between clock signal and transmit or receive data in simple SPI mode**

## 27.9.5 SCI Initialization in Simple SPI Mode

Initialization in simple SPI mode is the same as in clock synchronous mode. See section 27.6.3. SCI Initialization in Clock Synchronous Mode for an example initialization flow. The CKPOL and CKPH bits in the SPMR register must be set to ensure that the clock signal is suitable for both master and slave devices.

Always initialize the SCR register before making any changes to the operating mode or transfer format.

Note: Only the RE bit is set to 0. The SSR.ORER, FER, PER, and RDR flags are not initialized.

Changing the value of the TE bit from 1 to 0 or from 0 to 1 when the TIE bit in the SCR register is 1 at the same time, leads to the generation of a transmit data empty interrupt (SCIn_TXI).

## 27.9.6 Transmission and Reception of Serial Data in Simple SPI Mode

In master operation, ensure that the SSn pin of the slave device on the other side of the transfer is at the low level before starting the transfer and at the high level on completion of the transfer. Otherwise, the procedures are the same as in clock synchronous mode.

## 27.10 Bit Rate Modulation Function

Using the bit rate modulation function, the bit rate can be evenly corrected using the number specified in the MDDR register among 256 clock cycles of internal clocks which is selected by the CKS[1:0] bits in SMR/SMR_SMCI.

Figure 27.100 shows an example where the PCLK is selected in the CKS[1:0] bits in SMR/SMR_SMCI, the BRR bit is set to 0, and the MDDR is set to 160 in asynchronous mode. In this example, the cycle of the base clock is evenly corrected (256/160) and the bit rate is also corrected (160/256).

Note: Enabling an internal clock causes bias, and expansion and contraction are generated in the pulse width of the internal base clock.

Do not use this function in clock synchronous mode and in the highest speed settings in simple SPI mode (SMR.CKS[1:0] = 00b, SCR.CKE[1] = 0, and BRR = 0).

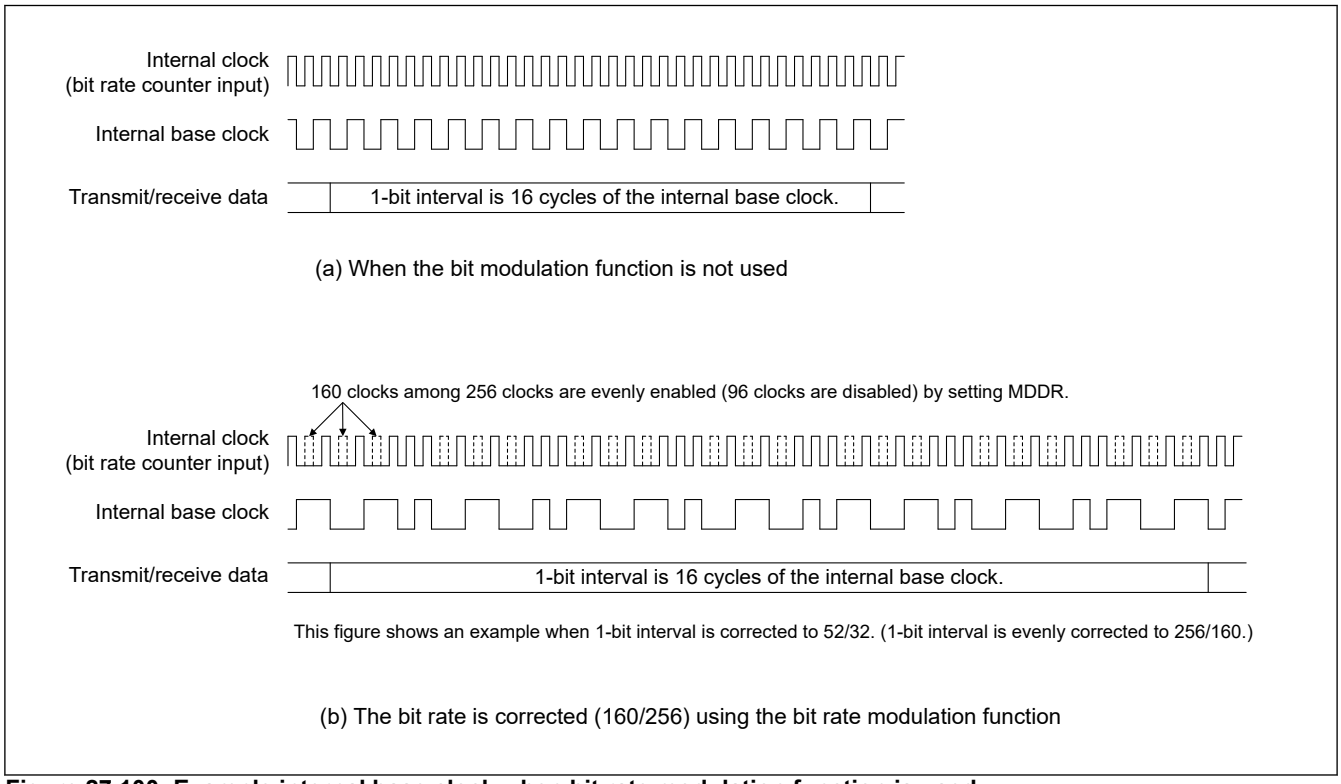



**Figure 27.100 Example internal base clock when bit rate modulation function is used**

## 27.11 Extended Serial Mode Control Section: Description of Operation

### 27.11.1 Serial Transfer Protocol

In conjunction with the SCIg module, the extended serial mode control section of the SCIh module can realize the serial transfer protocol composed of Start Frames and Information Frames that is shown in Figure 27.101.

A Start Frame is composed of a Break Field, Control Field 0, and Control Field 1. An Information Frame is composed of a number of Data Fields, a CRC16 Upper Field, and a CRC16 Lower Field.

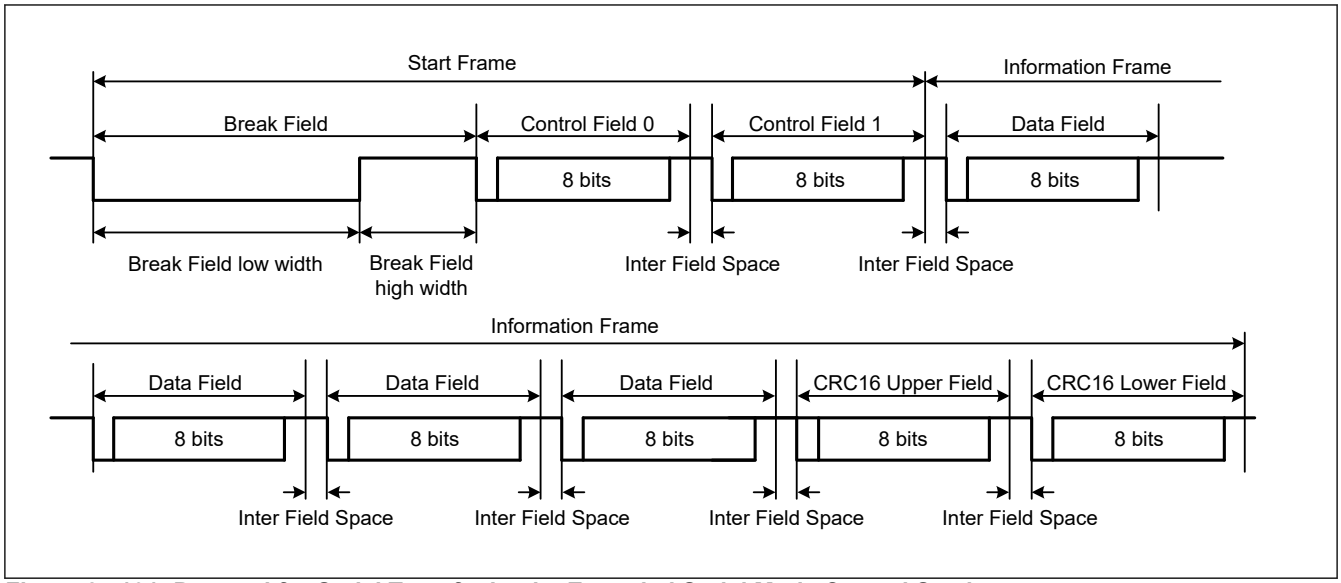



**Figure 27.101 Protocol for Serial Transfer by the Extended Serial Mode Control Section**

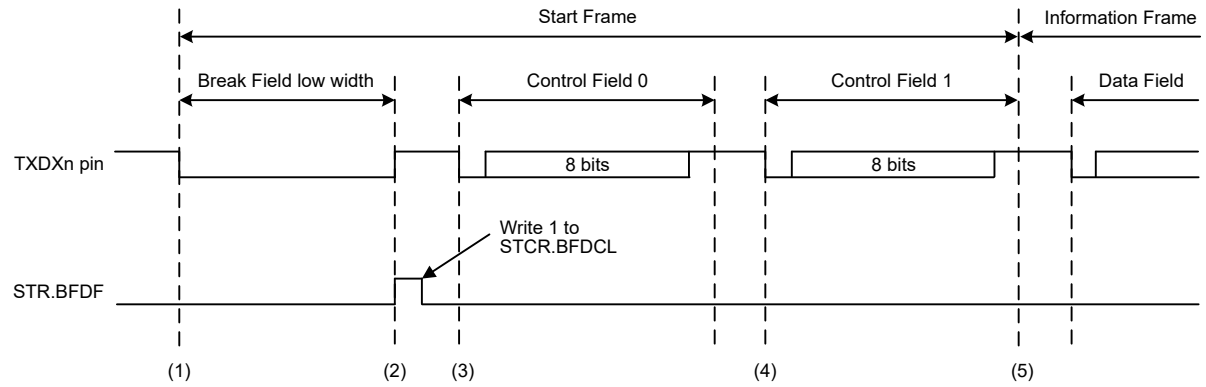
## 27.11.2 Transmitting a Start Frame

Figure 27.102 shows an example of operations to transmit a Start Frame, which is composed of the Break Field low width, Control Field 0, and Control Field 1. Figure 27.103 and Figure 27.104 are flowcharts for the transmission of a Start Frame.

Operations when the extended serial mode control section is to be used to transmit a Start Frame are as listed below. Be sure to use the SCIn (n = 1, 2) in asynchronous mode.

1. With Break Field low width output mode as the operating mode for the timer, writing 1 to the TCR.TCST bit starts counting by the timer, and the low level will be output from the TXDXn (n = 1, 2) pin over the period corresponding to registers TCNT and TPRE settings.
2. The output on the TXDXn pin is inverted when the timer counter underflows, and the STR.BFDF flag is set to 1. An SCIX0 interrupt is also generated if the value of the ICR.BFDIE bit is 1.
3. Write 0 to the TCR.TCST bit to stop counting by the timer, and send the data for Control Field 0 by using SCIn. After the Break Field low width output, stop counting before the next underflow occurs.
4. When the data for Control Field 0 have been transmitted, send the data for Control Field 1.
5. When the data for Control Field 1 have been transmitted, send an Information Frame.

Omit the Break Field and Control Field 0 to suit the structure of the Start Frame.




The above diagram assumes the following:

ESMER: ESME = 1
PCR: TXDXPS = 0
ICR: BFDIE = 1
TMR: TOMS[1:0] = 10b

Note: n = 1, 2

**Figure 27.102 Example of Operations When Transmitting a Start Frame**

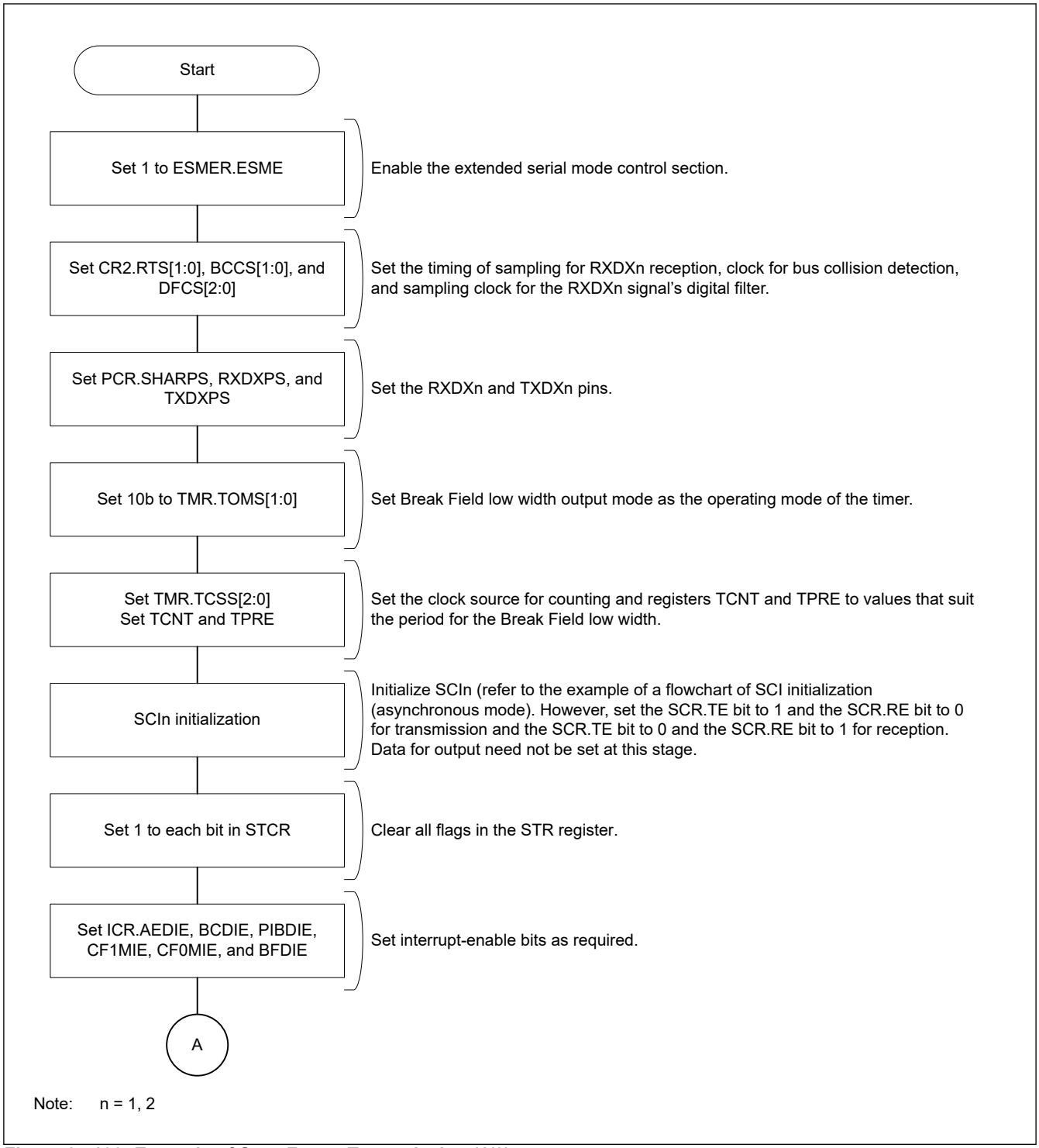



**Figure 27.103 Example of Start Frame Transmission (1/2)**

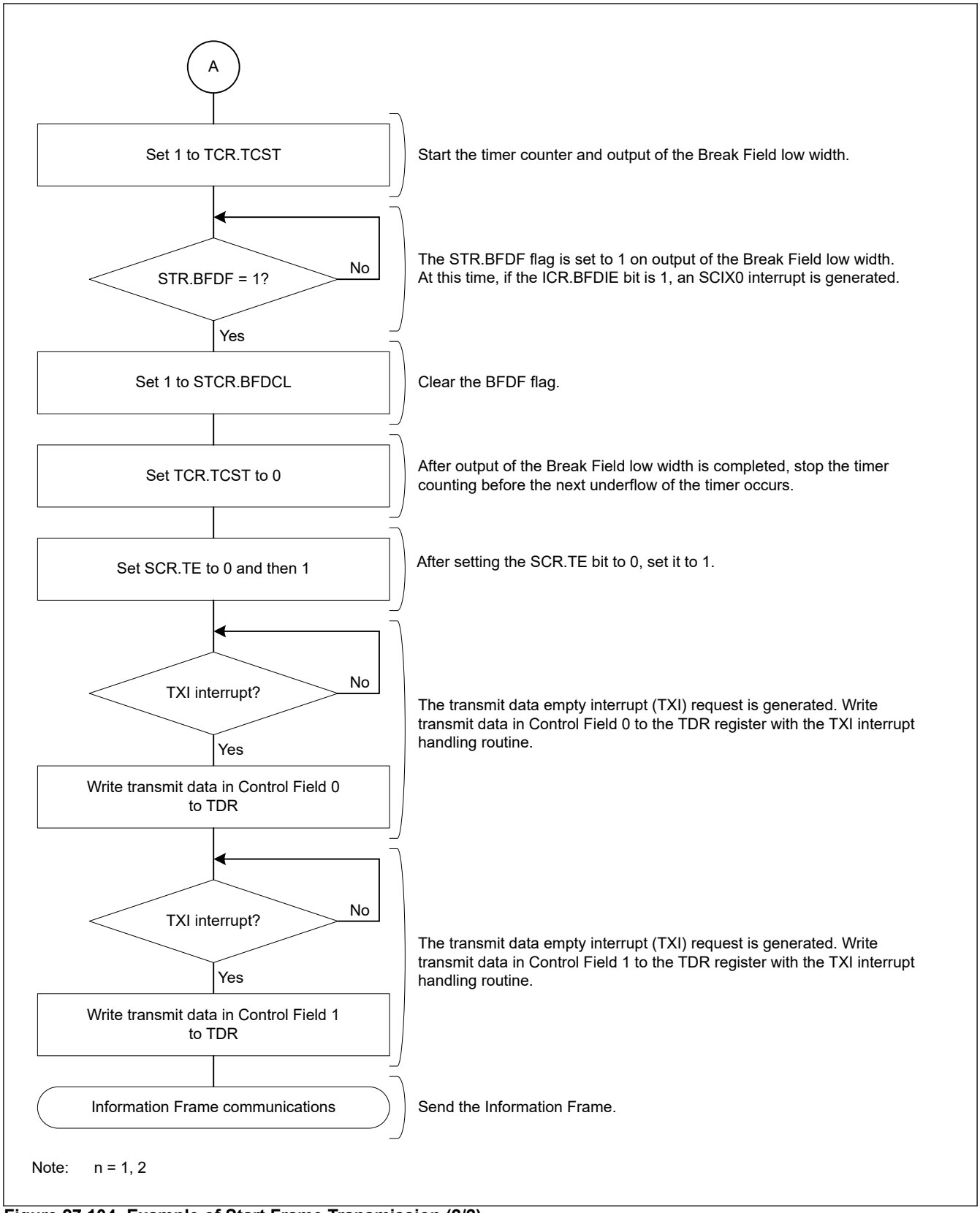



**Figure 27.104 Example of Start Frame Transmission (2/2)**

## 27.11.3 Receiving a Start Frame

The extended serial mode control section is capable of receiving Start Frames with the structures listed in Figure 27.105.

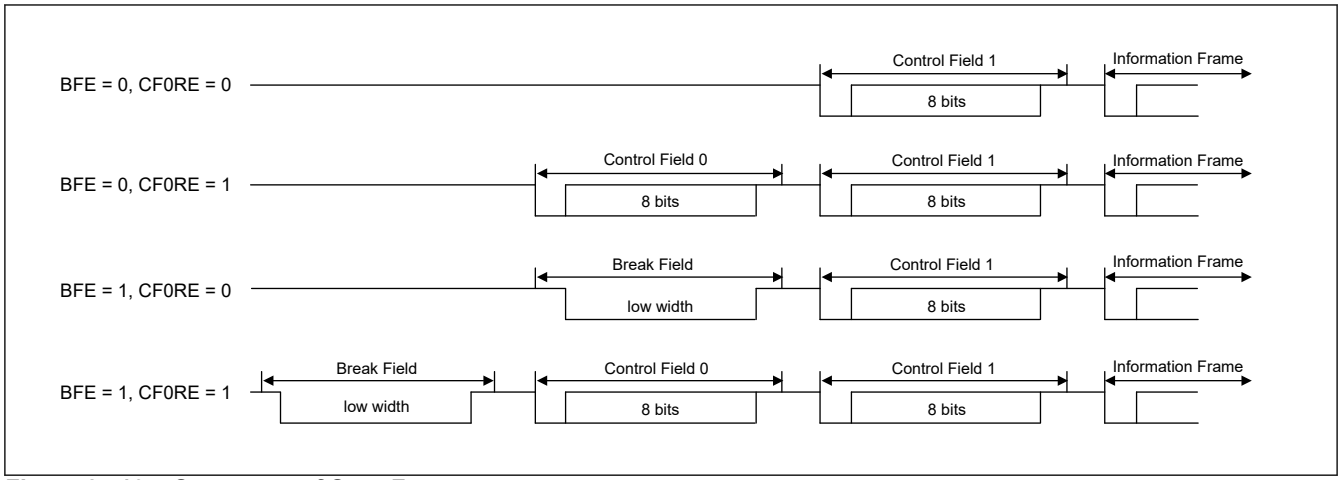


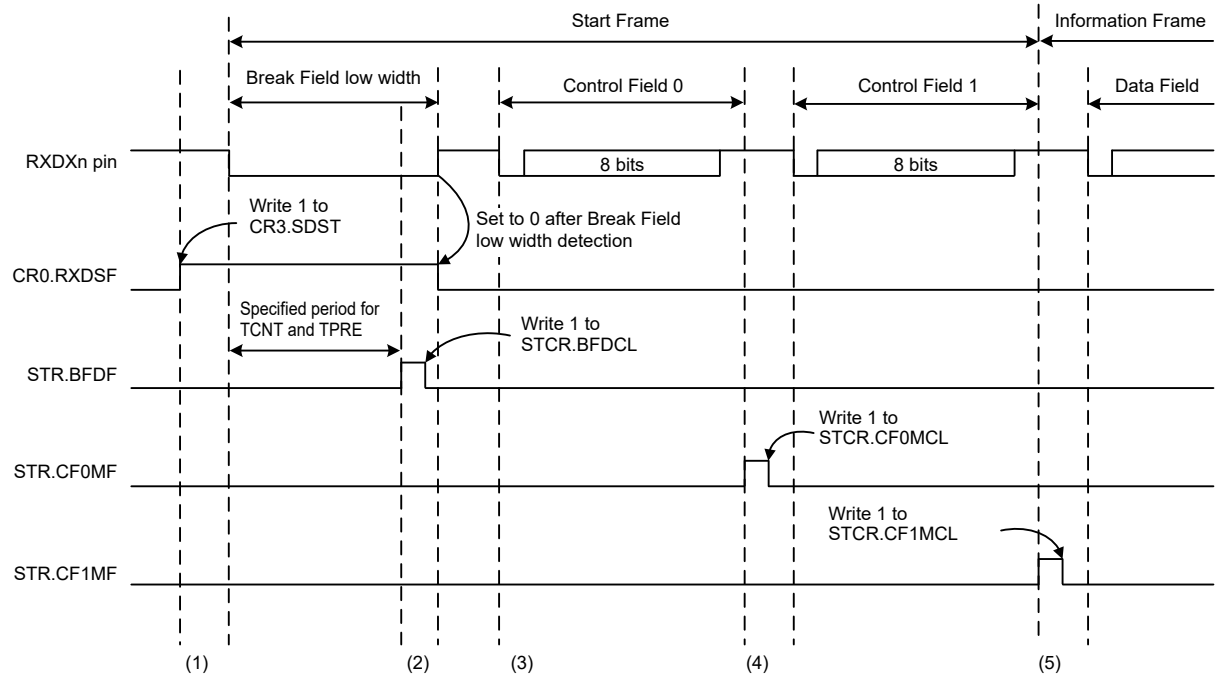

**Figure 27.105 Structures of Start Frames**

Figure 27.106 shows an example of operations to receive a Start Frame, which is composed of the Break Field low width, Control Field 0, and Control Field 1. Figure 27.107 and Figure 27.108 are flowcharts for the reception of a Start Frame, and Figure 27.109 is a state transition diagram for the extended serial mode control section.

Operations when the extended serial mode control section is to be used to receive a Start Frame are as listed below. Be sure to use the SCIn (n = 1, 2) in asynchronous mode.

1. With Break Field low width detection mode as the operating mode for the timer, writing 1 to the CR3.SDST bit enables detection of the Break Field low width. RXDXn input to the SCIn is disabled at this time.
2. Low-level input on the RXDXn pin continuing over a period longer than that corresponding to the settings of registers TCNT and TPRE is detected as the Break Field low width. At this time, the STR.BFDF flag is set to 1. An SCIX0 interrupt is also generated if the value of the ICR.BFDIE bit is 1.
3. When the input from the RXDXn pin goes high after the Break Field low width, the CR0.RXDSF flag becomes 0 and reception of Control Field 0 by the SCIn starts.
4. If the data received in Control Field 0 match the data set in the CF0DR register, the STR.CF0MF flag is set to 1. An SCIX1 interrupt is also generated if the value of the ICR.CF0MIE bit is 1. Reception of Control Field 1 by the SCIn starts after that. If the data received in Control Field 0 do not match the data set in the CF0DR register, a transition to the state prior to Break Field low width detection proceeds.
5. If the data received in Control Field 1 match the data set in registers PCF1DR and SCF1DR, the STR.CF1MF flag is set to 1. An SCIX1 interrupt is also generated if the value of the ICR.CF1MIE bit is 1. Transfer of the Information Frame by the SCIn starts after that. If the data received in Control Field 1 do not match the data set in either or both of registers PCF1DR and SCF1DR, a transition to the state prior to Break Field low width detection proceeds.

Omit the Break Field and Control Field 0 to suit the structure of the Start Frame.




The above diagram assumes the following:

ESMER: ESME = 1

CR1: BFE = 1, CF0RE = 1, CF1DS[1:0] = 10b

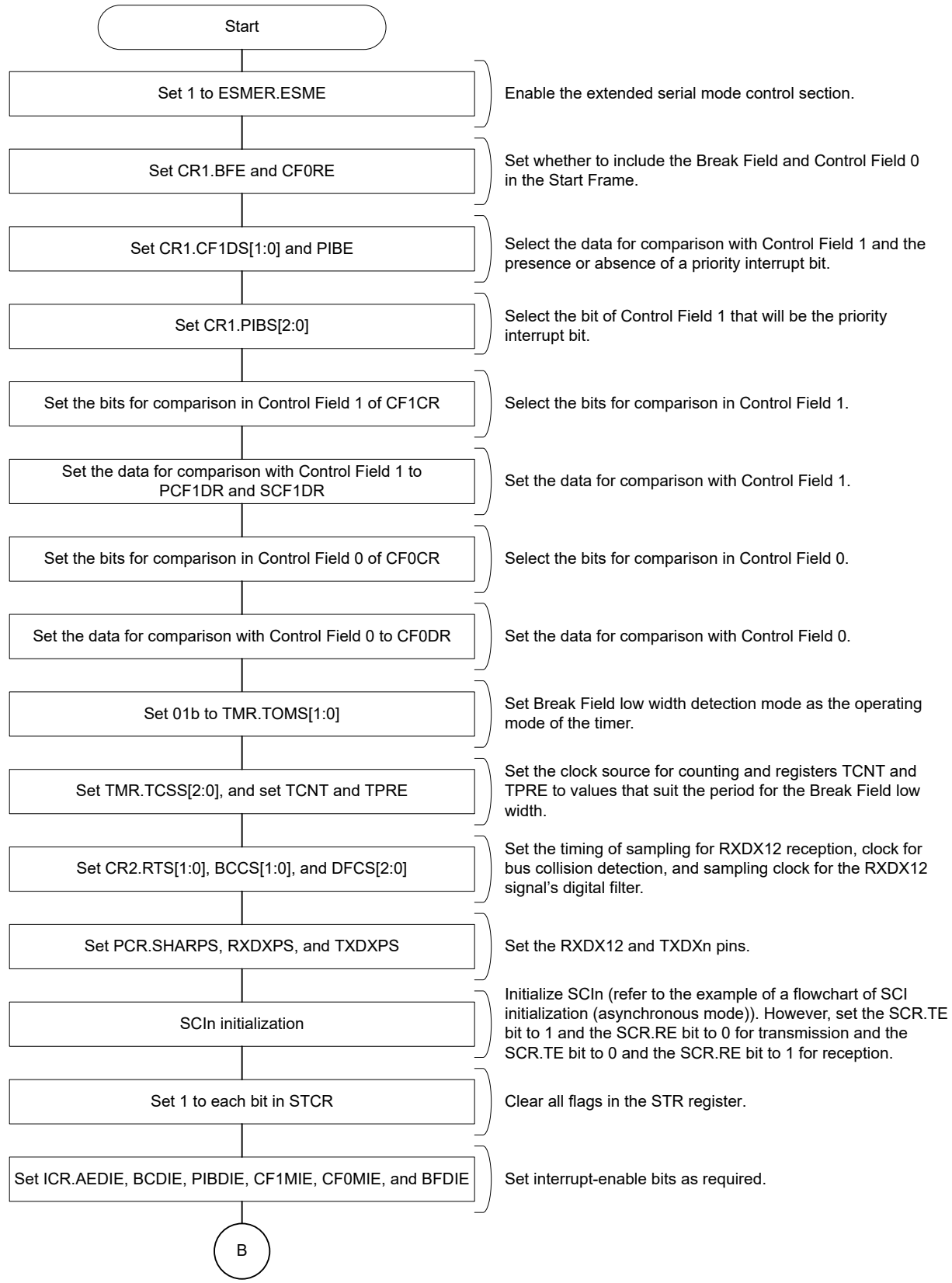
PCR: RXDXPS = 0

ICR: BFDIE = 1, CF0MIE = 1, CF1MIE = 1

TMR: TOMS[1:0] = 01b

Note: n = 1, 2

**Figure 27.106 Example of Operations at the Time of Start Frame Reception**




Note: n = 1, 2

**Figure 27.107 Sample Flowchart for Reception of a Start Frame (1)**

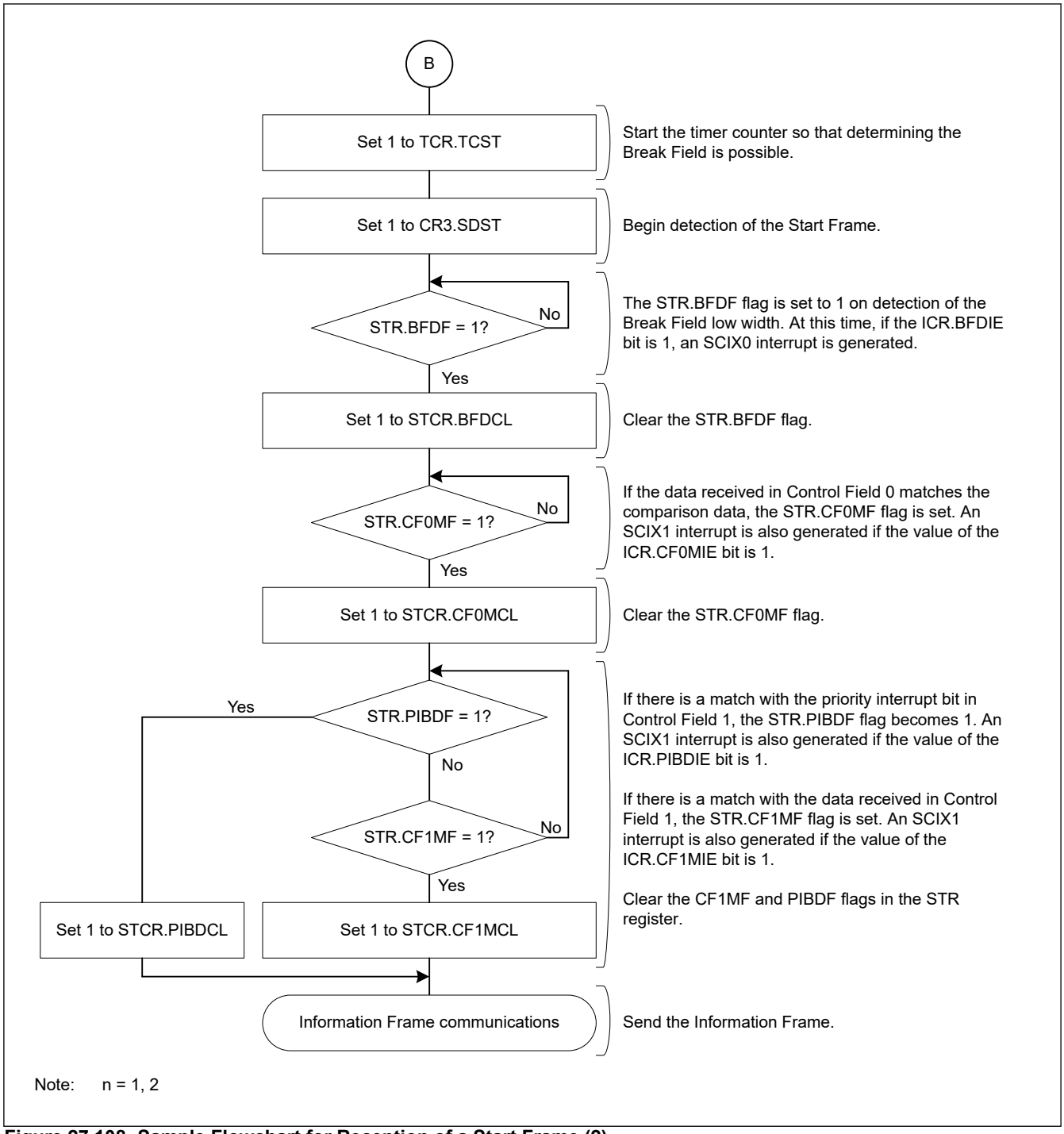



**Figure 27.108 Sample Flowchart for Reception of a Start Frame (2)**

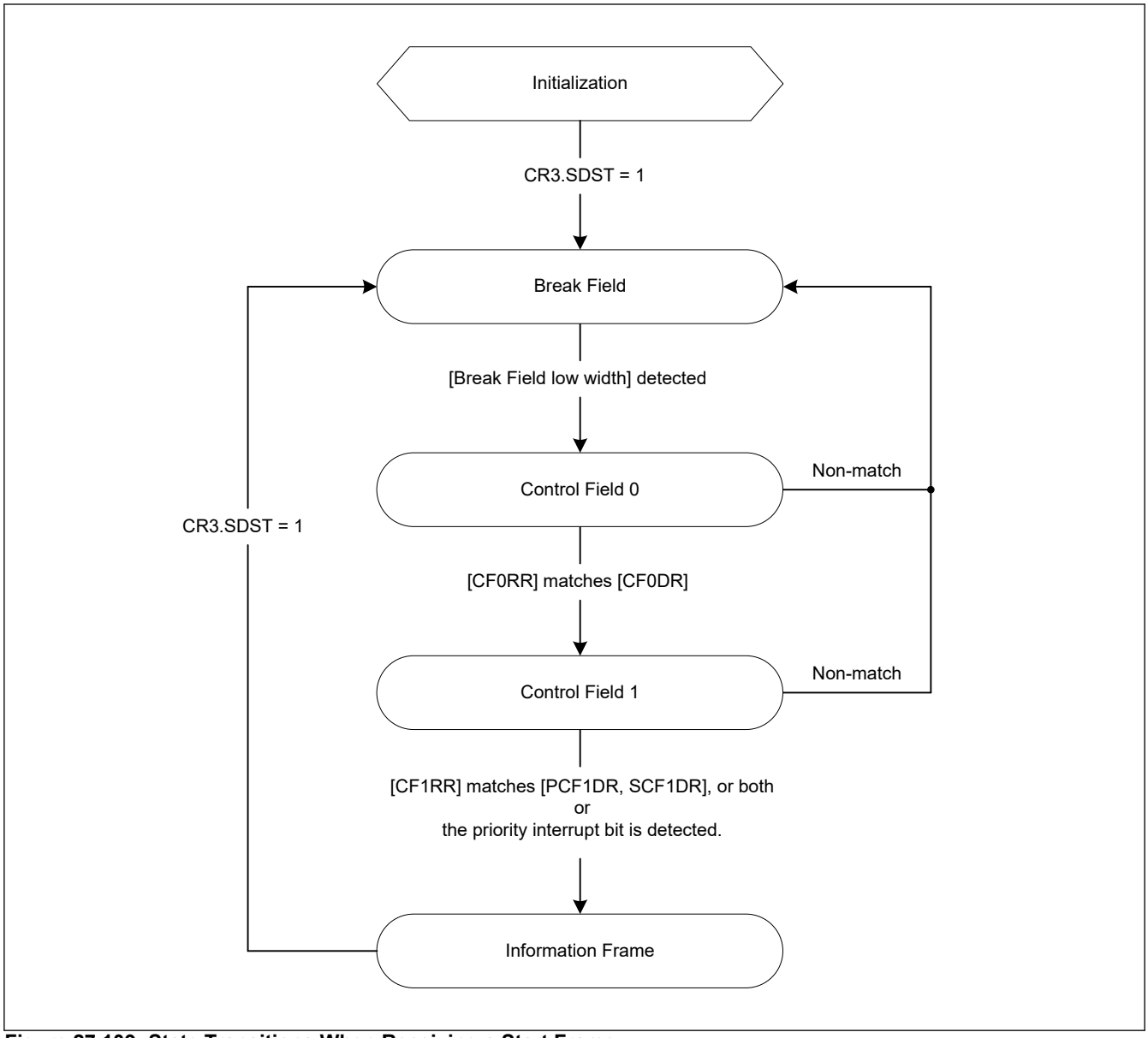



**Figure 27.109 State Transitions When Receiving a Start Frame**

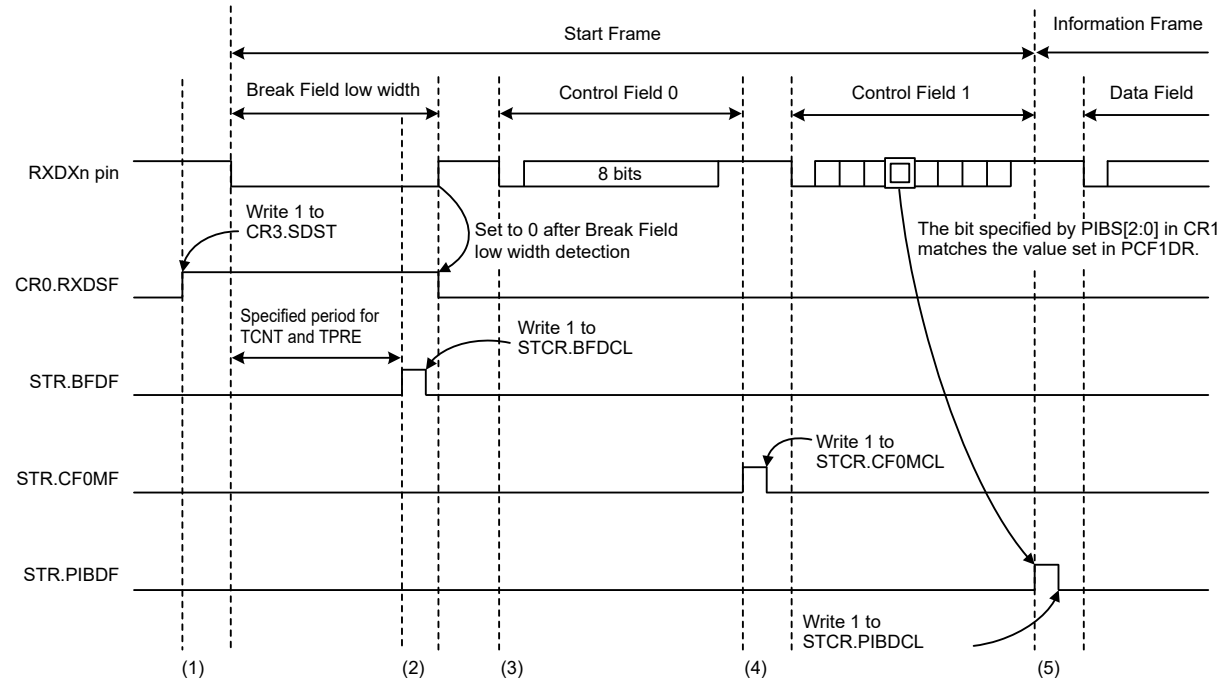
### 27.11.3.1 Priority Interrupt Bit

Figure 27.110 shows an example of operation in Start Frame reception where a priority interrupt bit is in use. Setting the CR1.PIBE bit to 1 enables the use of a priority interrupt bit.

Operations of the extended serial mode control section in start Frame reception where a priority interrupt bit is in use are as described below.

Steps (1) to (4) are the same as in Figure 27.106, for Start Frame reception.

(5) If the value of the bit selected by the CR1.PIBS[2:0] bits matches the corresponding bit in the PCF1DR register, the STR.PIBDF flag is set to 1. An SCIX1 interrupt is also generated if the value of the ICR.PIBDIE bit is 1. Transfer of the Information Frame by the SCIn (n = 1, 2) starts after that. If the data received in Control Field 1 do not match the data set in either or both of registers PCF1DR and SCF1DR and the priority interrupt bit is not detected, a transition to the state prior to Break Field low width detection proceeds.




The above diagram assumes the following:

ESMER: ESME = 1

CR1: BFE = 1, PIBE = 1, CF0RE = 0, CF1DS[1:0] = 10b, PIBS[2:0] = 011b

PCR: RXDXPS = 0

ICR: BFDIE = 1, CF0MIE = 1, PIBDIE = 1

TMR: TOMS[1:0] = 01b

Note: n = 1, 2

**Figure 27.110 Example of Operations When Receiving a Start Frame While the CR1.PIBE Bit is 1**

## 27.11.4 Detection of Bus Collisions

Detection of bus collisions operate for cases where output of the Break Field low width and transmission of data by the SCIn (n = 1, 2) are in progress when the ESMER.ESME bit and the SCIn.SCI.TE bit are set to 1.

Figure 27.111 shows an example of operations with bus collision detection. Signals output through TXDXn and input through RXDXn are sampled with the bus collision detection clock set with the CR2.BCCS[1:0] bits as the sampling clock, and the STR.BCDF flag is set to 1 if the signals fail to match three times in a row. An SCIX2 interrupt is also generated if the value of the ICR.BCDIE bit is 1.

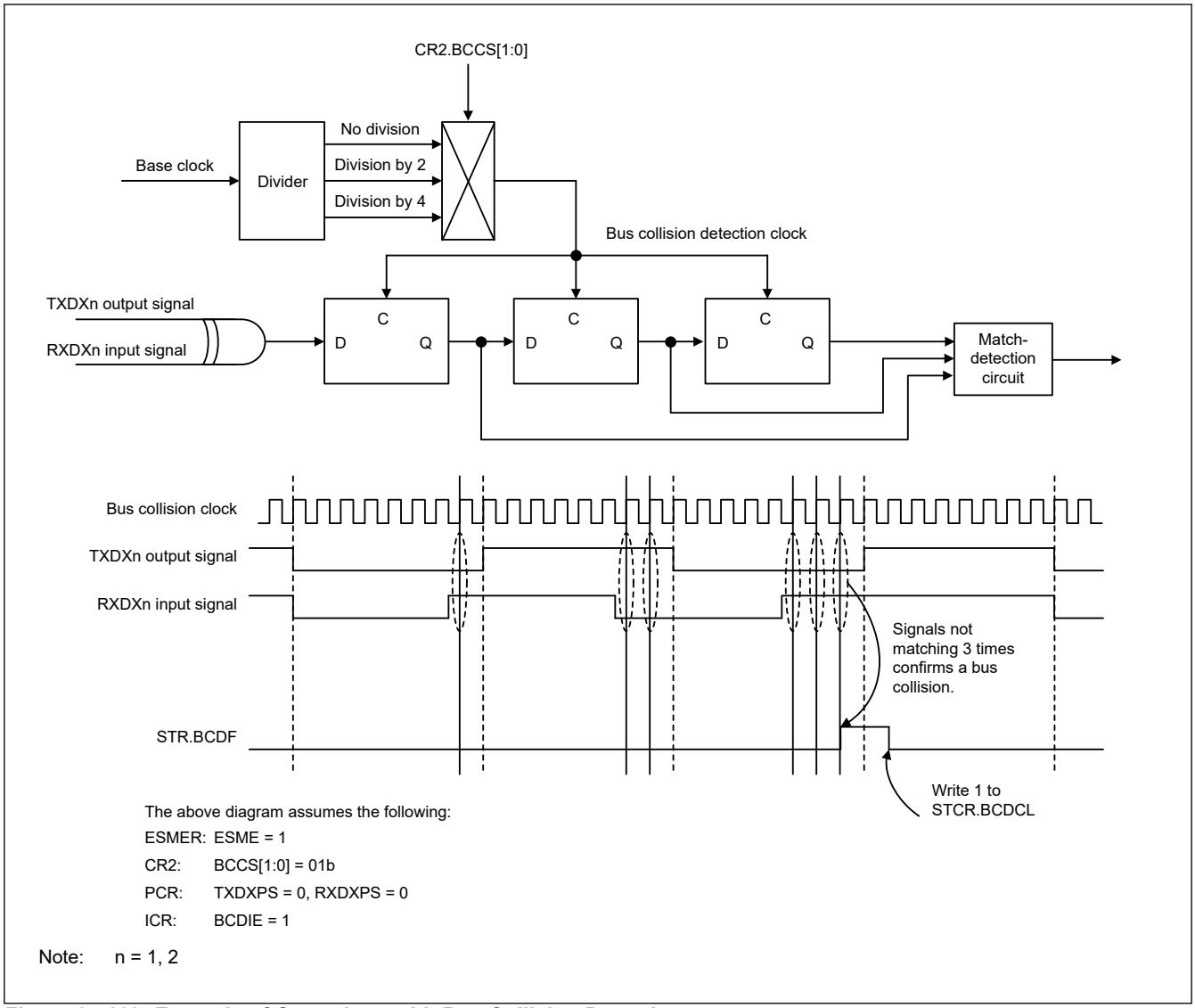



**Figure 27.111 Example of Operations with Bus Collision Detection**

## 27.11.5 Digital Filter for Input on the RXDXn (n = 1, 2) Pin

Signals input through the RXDXn pin can be passed through a digital filter before they are conveyed to the internal circuits. The digital filter consists of three flip-flop circuit stages connected in series and a match-detecting circuit. The CR2.DFCS[2:0] bits select the sampling clock for the RXDXn pin input signals. If the outputs of all three latches match, the given level is conveyed to subsequent circuits. If the levels do not match, the previous value is retained. In other words, levels are confirmed as being the signal if they are retained for at least three cycles of the sampling clock but judged to be noise rather than changes in the signal level if they change within three cycles of the sampling clock. Figure 27.112 shows an example of operations with the digital filter.

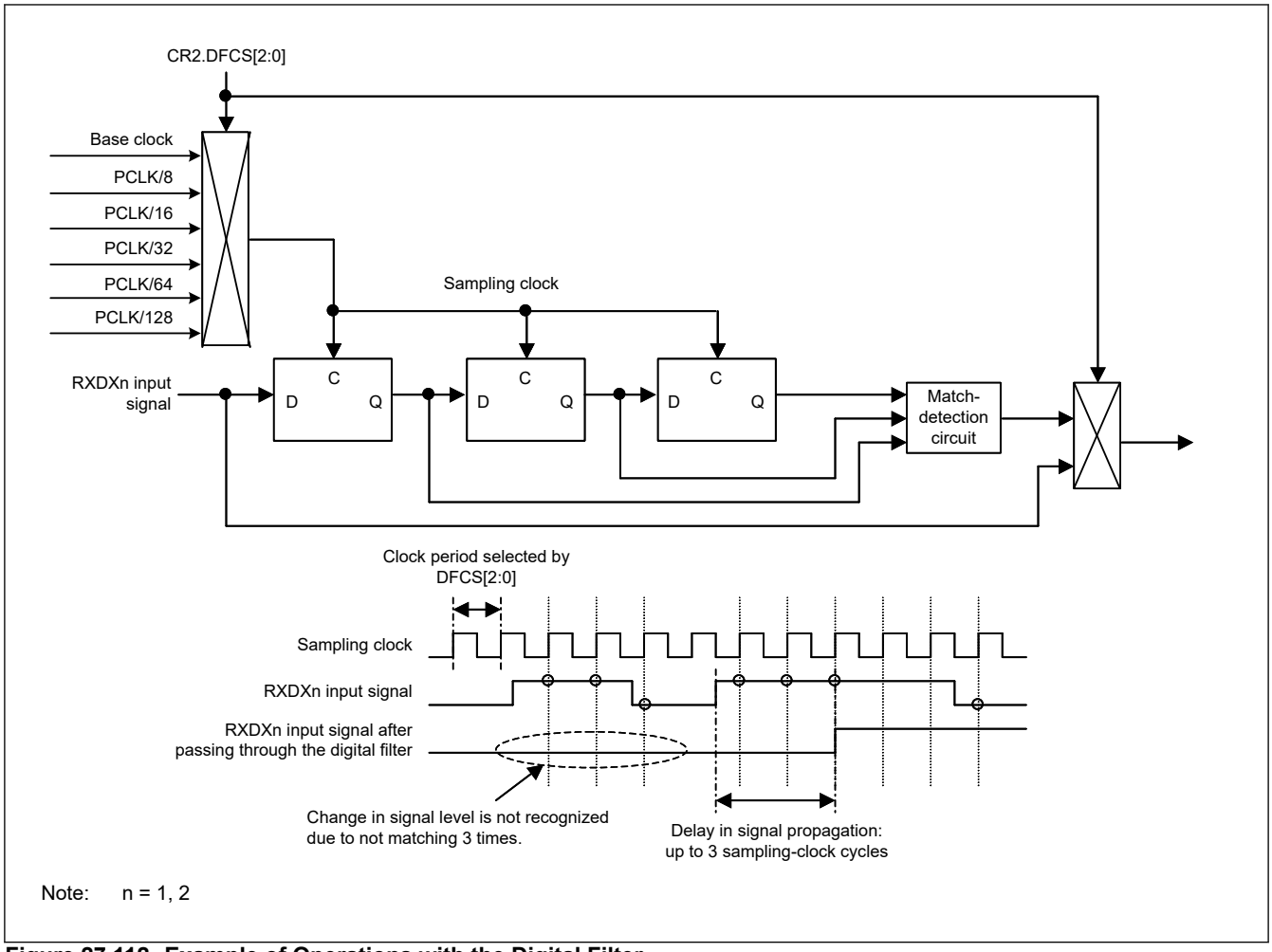
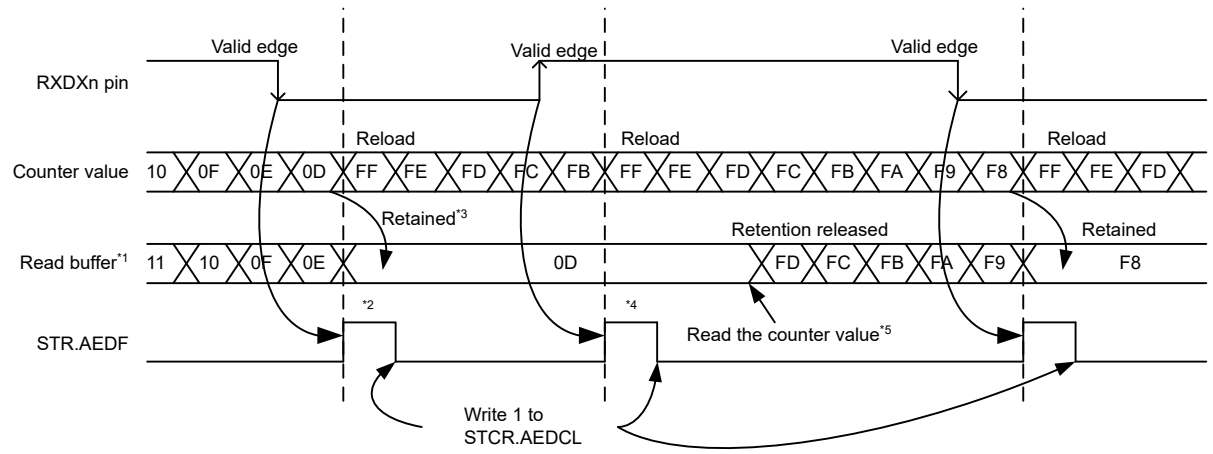



**Figure 27.112 Example of Operations with the Digital Filter**

## 27.11.6 Bit Rate Measurement

The bit rate measurement function measures the intervals between rising and falling edges and between falling and rising edges of the signal input from the RXDXn (n = 1, 2) pin. Figure 27.113 shows an example of operations for bit rate measurement.

1. Writing 1 to the CR0.BRME bit enables bit rate measurement. Only set the BRME bit to 1 when you wish to proceed with bit rate measurement. Furthermore, bit rate measurement will not proceed during a Break Field, even if the BRME bit is set to 1.
2. After detection of the Break Field low width, bit rate measurement starts when the level input on the RXDXn pin becomes high.
3. Once bit rate measurement has started, counter values from the timer are retained in the read buffers on the input of valid edges from the RXDXn pin (rising and falling edges) and the counter is reloaded. An SCIX3 interrupt is also generated if the value of the ICR.AEDIE bit is 1. Retention by registers TCNT and TPRE is released by reading these registers.
4. The bit rate as calculated from the values counted during intervals between valid edges can be used for adjusting the rate by changing the settings of the SCIn. To disable the bit rate measurement after a match with Control Field 1, write 0 to the CR0.BRME bit.

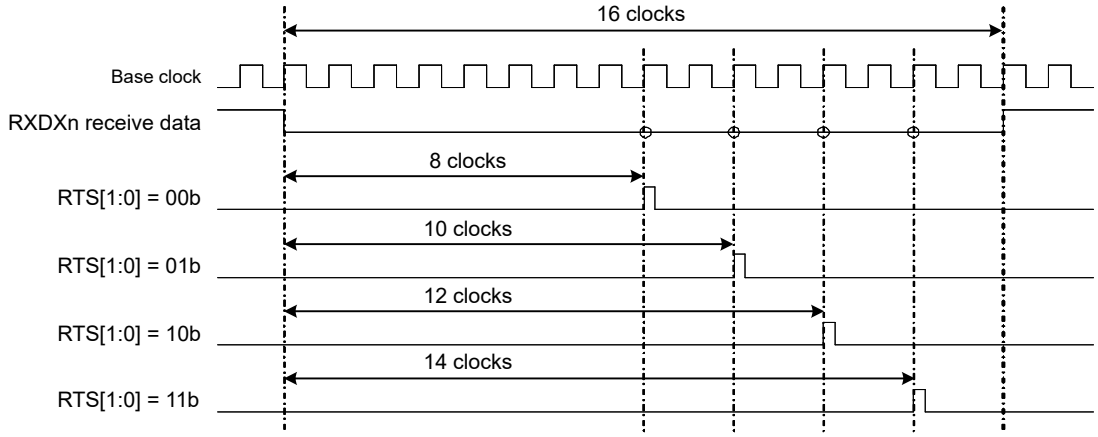



Note: n = 1, 2
Note 1. The TCNT or TPRE register indicates the contents of the read buffers.
Note 2. The STR.AEDF flag becomes 1 on detection of a valid edge.
Note 3. When an effective edge is detected, the current value of the counter is stored in the read buffer.
Note 4. The value retained in the read buffers due to one valid edge is not updated to the new value even if a further valid edge is detected.
Note 5. Retention of a value in a read buffer due to a valid edge is released by reading the TCNT or TPRE register.

**Figure 27.113 Example of Operations for Bit Rate Measurement**

## 27.11.7 Selectable Timing for Sampling Data Received through RXDXn (n = 1, 2)

The extended serial mode control section provides a way of adjusting the timing for the sampling of data received through the RXDXn pin of an SCIn by setting the CR2.RTS[1:0] bits to select the rising edges of 8th, 10th, 12th, or 14th cycle of the SCI base clock. If the value of the SEMR.ABCS bit is 1, the bits select the rising edges of 4th, 5th, 6th, or 7th cycle of the PCLK clock of the SCIn. Figure 27.114 shows timing for the sampling of data received through RXDXn.




The above diagram assumes the following:
SEMR.ABCS = 0

Note: n = 1, 2

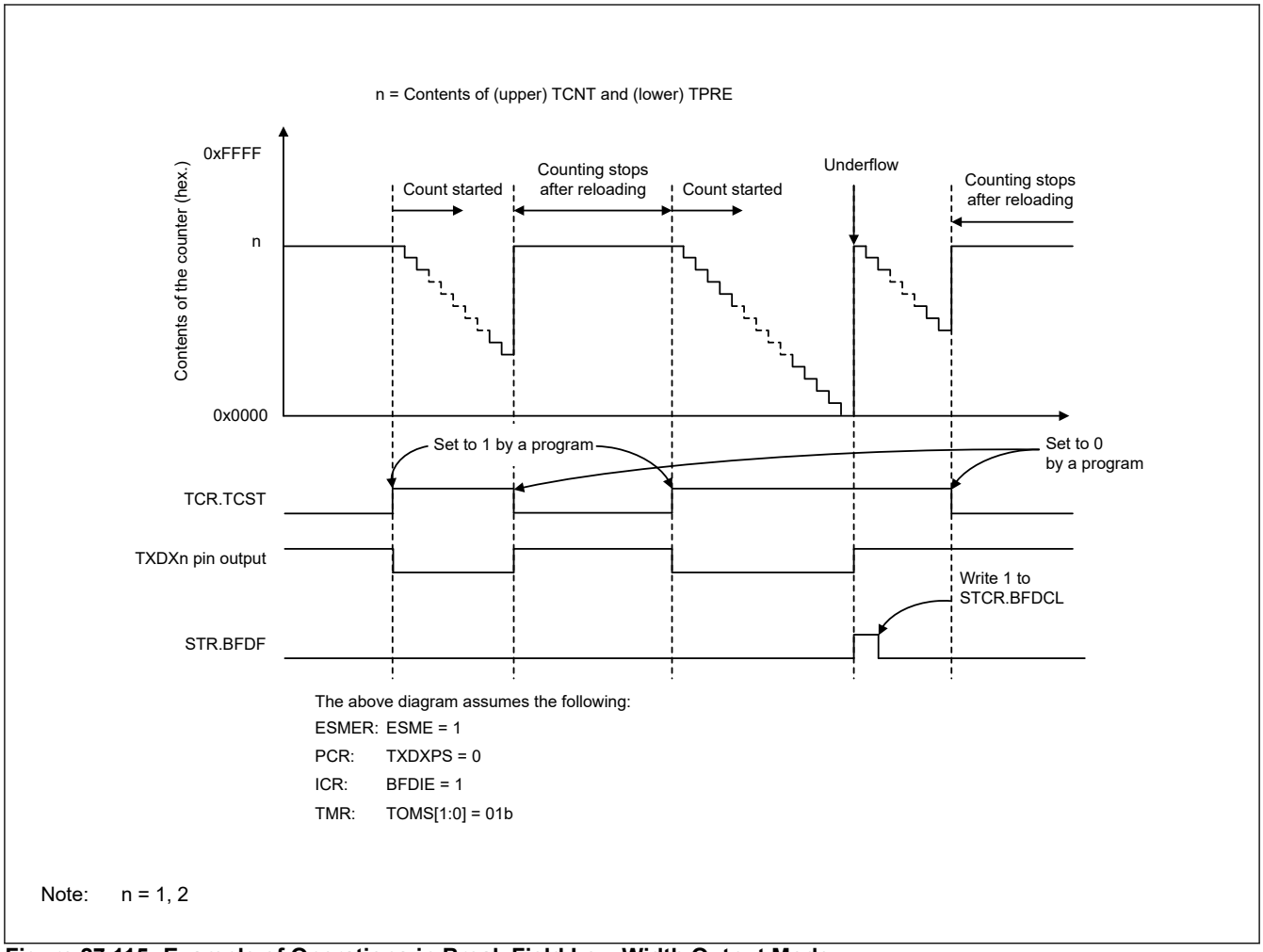
**Figure 27.114 Timing for Sampling of Data Received through RXDXn**

## 27.11.8 Timer

The timer has the following operating modes.

### (1) Break Field Low Width Output Mode

This mode is for output through the TXDXn (n = 1, 2) pin of the low level over the Break Field low width at the transmission of a Start Frame. Setting the TMR.TOMS[1:0] bits to 10b switches operation to Break Field low width output mode. The TMR.TCSS[2:0] bits select the clock source for the counter. When the TCR.TCST bit is set to 1, the output on the TXDXn pin goes to the low level and counting starts. When the timer underflows, the output on the TXDXn pin goes to the high level and the STR.BFDF flag is set to 1. An SCIX0 interrupt is also generated if the value of the ICR.BFDIE bit is 1. When 0 is written to the TCR.TCST bit, counting stops after reloading of registers TPRE and TCNT. After output of the Break Field low width is completed, stop the timer before it underflows again. Figure 27.115 shows an example of operations in Break Field low width output mode.

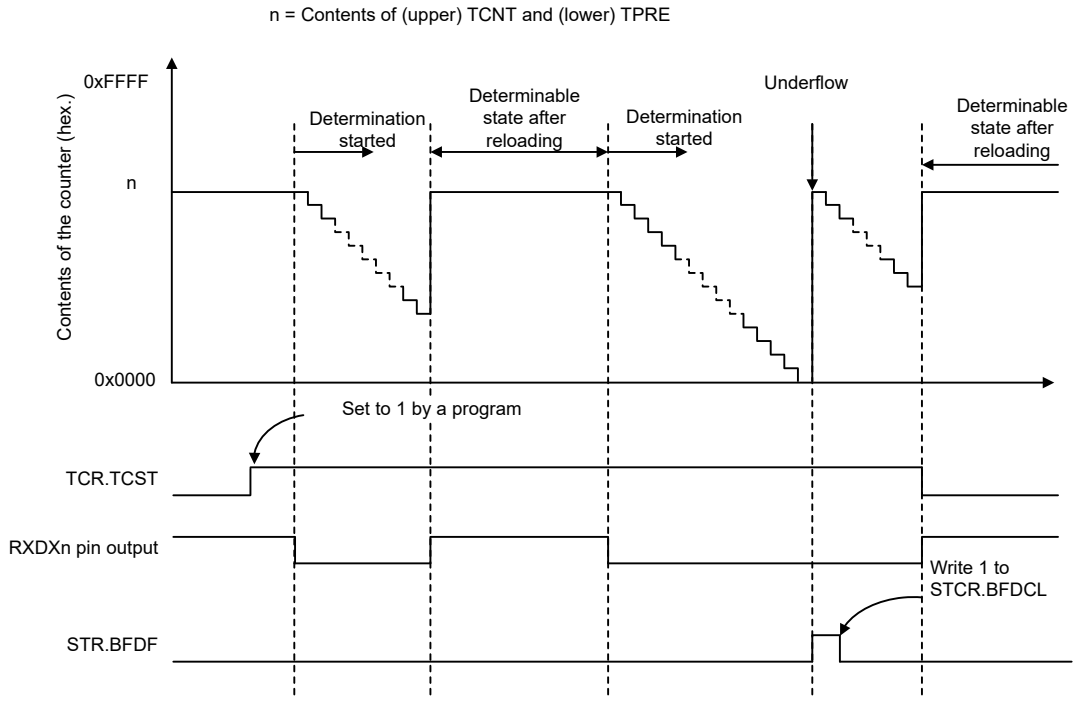



Note: n = 1, 2

**Figure 27.115 Example of Operations in Break Field Low Width Output Mode**

### (2) Break Field Low Width Determination Mode

This mode is for determining the Break Field low width in the input signal on the RXDXn (n = 1, 2) pin at the reception of a Start Frame. Setting the TMR.TOMS[1:0] bits to 01b switches operation to Break Field low width determination mode. The TMR.TCSS[2:0] bits select the clock source for the counter. When the TCR.TCST bit is set to 1, the interface enters the Break Field low width determinable state. Determination starts when a low level is input from the RXDXn pin. When a high level is then input on the RXDXn pin, registers TPRE and TCNT are reloaded and the interface enters the Break Field low width determinable state. When the timer underflows during Break Field low width determination, the STR.BFDF flag is set to 1. An SCIX0 interrupt is also generated if the value of the ICR.BFDIE bit is 1. If an underflow of the timer during data transfer cause a problem in the form of interrupt generation, stop the timer after Break Field low width determination. Figure 27.116 shows an example of operations in Break Field low width output mode.




The above diagram assumes the following:

ESMER: ESME = 1
CR1: BFE = 0
PCR: RXDXPS = 0
ICR: BFDIE = 1
TMR: TOMS[1:0] = 01b

Note: n = 1, 2

**Figure 27.116 Example of Operations in Break Field Low Width Determination Mode**

### (3) Timer Mode

This mode is for counting cycles of the internal clock as the clock source. Setting the TMR.TOMS[1:0] bits to 00b switches operation to timer mode. The TMR.TCSS[2:0] bits select the clock source for the counter. Counting starts when 1 is written to the TCR.TCST bit and stops when 0 is written to the TCST bit. Registers TPRE and TCNT both count down. The TPRE register counts cycles of the clock source for counting, and underflows of the TPRE register provide the clock source for counting by the TCNT register. When the timer underflows, the STR.BFDF flag is set to 1. An SCIX0 interrupt is also generated if the value of the ICR.BFDIE bit is 1.

## 27.12 Interrupt Sources

### 27.12.1 Buffer Operation for SCIn_TXI and SCIn_RXI Interrupts (Non-FIFO Selected)

If the conditions for an SCIn_TXI and SCIn_RXI interrupt are satisfied while the interrupt status flag in the ICU is 1, the ICU does not output the interrupt request but saves it internally with a capacity for retention of one request per source.

When the interrupt status flag in the ICU is set to 0, the interrupt request retained within the ICU is output. The internally retained interrupt request is automatically discarded when the actual interrupt is output. Clearing of the associated interrupt enable bit (the TIE or RIE bit in the SCR/SCR_SMCI) can also be used to discard an internally retained interrupt request.

### 27.12.2 Buffer Operation for SCIn_TXI and SCIn_RXI Interrupts (FIFO Selected)

When an interrupt status flag in the ICU is set to 1, the SCIn_TXI and SCIn_RXI interrupts do not output interrupt requests to the ICU. When an interrupt status flag of the ICU is set to 0, and if the conditions for an SCIn_TXI and SCIn_RXI interrupts are satisfied, an interrupt request is generated.

### 27.12.3 Interrupts in Asynchronous, Manchester, Clock Synchronous, and Simple SPI Modes

**(1) Non-FIFO selected**

Table 27.42 lists interrupt sources in asynchronous mode, Manchester mode, clock synchronous mode, and simple SPI mode.

A different interrupt vector can be assigned to each interrupt source. Individual interrupt sources can be enabled or disabled with the enable bits in the SCR register.

If the SCR.TIE bit is 1, an SCIn_TXI interrupt request is generated when transmit data is transferred from the TDR or TDRHL register*1 to the TSR register. An SCIn_TXI interrupt request can also be generated by using a single instruction to set the SCR.TE and SCR.TIE bits to 1 at the same time. An SCIn_TXI interrupt request can activate the DTC or DMAC to handle data transfer.

An SCIn_TXI interrupt request is not generated by setting the SCR.TE bit to 1 when SCR.TIE is 0 or by setting the SCR.TIE bit to 1 when the SCR.TE is 1.*2

When new data is not written by the time of transmission of the last bit of the current transmit data and SCR.TEIE is 1, the SSR.TEND flag is set to 1 and an SCIn_TEI interrupt request is generated. Additionally, when SCR.TE is 1, the SSR.TEND flag retains the value 1 until more transmit data is written to the TDR or TDRHL register*1, and setting SCR.TEIE to 1 leads to the generation of an SCIn_TEI interrupt request.

Writing data to the TDR or TDRHL register*1 leads to clearing of the SSR.TEND flag and, after a certain time, discarding of the SCIn_TEI interrupt request.

If the SCR.RIE bit is 1, an SCIn_RXI interrupt request is generated when received data is stored in the RDR register. An SCIn_RXI interrupt request can activate the DTC or DMAC to handle data transfer.

Setting any of the SSR.ORER, FER, PER and MER*3 flags to 1 when the SCR.RIE bit is 1 leads to the generation of an SCIn_ERI interrupt request.

An SCIn_RXI interrupt request is not generated in this case. Clearing all these flags (ORER, FER, PER, MER*3, SYER*3, PFER*3 and SBER*3) leads to discarding of the SCIn_ERI interrupt request.

Note 1. When asynchronous mode and 9-bit data length are selected.
Note 2. To temporarily prohibit SCIn_TXI interrupts on transmission of the last of the data when a new round of transmission is to be started, after handling the transmission-completed interrupt, control activation of the interrupt by using the interrupt request enable bit in the ICU rather than using the SCR.TIE bit. This approach can prevent the suppression of SCIn_TXI interrupt requests in the transfer of new data.
Note 3. MER, SYER, PFER, and SBER work as a factor of SCIn_ERI interrupt only in Manchester mode. SYER, PFER, and SBER also only work if its enable bits (SYEREN, PFEREN, SBEREN in MECR) are set to "1".

**(2) FIFO selected**

Table 27.43 lists interrupt sources in FIFO selected mode.

If the SCR.TIE bit is 1, an SCIn_TXI interrupt request is generated when the stored amount of data in the FTDRL register becomes the threshold value indicated in FCR.TTRG or below. An SCIn_TXI interrupt request can also be generated by using a single instruction to set the SCR.TIE and SCR.TE bits to 1 simultaneously or by setting SCR.TIE to 1 when SCR.TE is 1.

An SCIn_TXI interrupt request is not generated by setting SCR.TE to 1 when SCR.TIE is 0.

If SCR.TEIE is 1 and if the next data is not written to the FTDRL register by the time the last bit of the transmit data is sent, the SSR_FIFO.TEND flag is set to 1 and the SCIn_TEI interrupt request is generated.

If SCR.RIE is 1, the SCIn_RXI interrupt request is generated when the stored amount of data in the FRDRL register is equal to or greater than the threshold value indicated in FCR.RTRG. When RTRG is 0, an SCIn_RXI interrupt does not occur even when the amount of data in the receive FIFO is equal to 0.

If the SCR.RIE bit is 1, when the SSR_FIFO.ORER flag is set to 1 or data with a framing error or a parity error is stored in the FRDRL register, the SCIn_ERI interrupt request is generated. When the amount of data stored in the FRDRL register is at the threshold value or above, the SCIn_RXI interrupt request is also generated. The SCIn_ERI interrupt request can be canceled, in which case SSR_FIFO.ORER, FER, and PER flags are all cleared.

**Table 27.42 SCI interrupt sources with non-FIFO selected**

| Name | Interrupt source | Interrupt flag | Interrupt enable | DTC or DMAC activation |
|---|---|---|---|---|
| SCIn_ERI (n = 0 to 4, 9) | Receive error*1 | SSR.ORER, SSR.FER, SSR.PER, DCCR.DFER, DCCR.DPER, (SSR.MER, MESR.SYER, MESR.PFER, MESR.SBER)*2 | SCR.RIE | Not possible |
| SCIn_RXI (n = 0 to 4, 9) | Receive data full | SSR.RDRF | SCR.RIE | Possible |
| | Address match | DCCR.DCMF | SCR.RIE | Possible |
| SCIn_AM (n = 0, 3, 4, 9) | Address match | DCCR.DCMF | — | Not possible |
| SCIn_TXI (n = 0 to 4, 9) | Transmit data empty | SSR.TDRE | SCR.TIE | Possible |
| SCIn_TEI (n = 0 to 4, 9) | Transmit end | SSR.TEND | SCR.TEIE | Not possible |

Note 1. The interrupt flag is only ORER when in clock synchronous and simple SPI mode.
Note 2. MER, SYER, PFER, and SBER work as a factor of ERI interrupt only in Manchester mode. SYER, PFER, and SBER also only work if its enable bits (SYEREN, PFEREN, SBEREN in MECR) are set to 1.

**Table 27.43 SCI interrupt sources with FIFO selected**

| Name | Interrupt source | Interrupt flag | Interrupt enable | DTC or DMAC activation |
|---|---|---|---|---|
| SCIn_ERI (n = 0 to 4, 9) | Receive error*1 | SSR_FIFO.ORER, SSR_FIFO.FER, SSR_FIFO.PER, DCCR.DFER, DCCR.DPER | SCR.RIE | Not possible |
| | | SSR_FIFO.DR (when FCR.DRES = 1) | SCR.RIE | Not possible |
| SCIn_RXI (n = 0 to 4, 9) | Receive data full | SSR_FIFO.RDF | SCR.RIE | Possible |
| | Receive data ready | SSR_FIFO.DR (when FCR.DRES = 0) | SCR.RIE | Possible |
| | Address match | DCCR.DCMF | SCR.RIE | Possible |
| SCIn_AM (n = 0, 3, 4, 9) | Address match | DCCR.DCMF | — | Not possible |
| SCIn_TXI (n = 0 to 4, 9) | Transmit data empty | SSR_FIFO.TDFE | SCR.TIE | Possible |
| SCIn_TEI (n = 0 to 4, 9) | Transmit end | SSR_FIFO.TEND | SCR.TEIE | Not possible |

Note 1. The interrupt flag is only ORER when in clock synchronous and simple SPI mode.

## 27.12.4 Interrupts in Smart Card Interface Mode

Table 27.44 lists interrupt sources in smart card interface mode. A transmit end interrupt (SCIn_TEI) request and an address match (SCIn_AM) request cannot be used in this mode.

**Table 27.44 SCI Interrupt sources (1 of 2)**

| Name | Interrupt source | Interrupt flag | Interrupt enable | DTC or DMAC activation |
|---|---|---|---|---|
| SCIn_ERI (n = 0 to 4, 9) | Receive error or error signal detection | SSR_SMCI.ORER, SSR_SMCI.PER, SSR_SMCI.ERS | SCR_SMCI.RIE | Not possible |
| SCIn_RXI (n = 0 to 4, 9) | Receive data full | SSR_SMCI.RDRF | SCR_SMCI.RIE | Possible |

**Table 27.44 SCI Interrupt sources (2 of 2)**

| Name | Interrupt source | Interrupt flag | Interrupt enable | DTC or DMAC activation |
|---|---|---|---|---|
| SCIn_TXI (n = 0 to 4, 9) | Transmit data empty | SSR_SMCI.TEND | SCR_SMCI.TIE | Possible |

Data transmission or reception using the DTC or DMAC is also possible in smart card interface mode, similar to normal SCI mode. In transmission, when the SSR_SMCI.TEND flag is set to 1, an SCIn_TXI interrupt request is generated. This SCIn_TXI interrupt request activates the DTC or DMAC, allowing transfer of transmit data if the SCIn_TXI request is previously specified as a source of DTC or DMAC activation. The TEND flag is automatically set to 0 when the DTC or DMAC transfers the data.

If an error occurs, the SCI automatically retransmits the same data. During the retransmission, the TEND flag is kept at 0 and the DTC or DMAC is not activated. Therefore, the SCI and DTC or DMAC automatically transmit the specified number of bytes, including retransmission after an error occurrence. However, the SSR_SMCI.ERS flag is not automatically set to 0 at error occurrence. Therefore, the ERS flag must be cleared by previously setting the SCR_SMCI.RIE bit to 1 to enable an SCIn_ERI interrupt request to be generated at error occurrence.

When transmitting or receiving data using the DTC or DMAC, always enable the DTC or DMAC before making the SCI settings. For DTC or DMAC settings, see section 17, Data Transfer Controller (DTC), section 16, DMA Controller (DMAC).

In reception, an SCIn_RXI interrupt request is generated when receive data is set to the RDR register. This SCIn_RXI interrupt request activates the DTC or DMAC, allowing transfer of the receive data if the SCIn_RXI request is previously specified as a source of DTC or DMAC activation. If an error occurs, the error flag is set. Therefore, the DTC or DMAC is not activated and an SCIn_ERI interrupt request is issued to the CPU instead. The error flag must be cleared.

### 27.12.5 Interrupts in Simple IIC Mode

Table 27.45 lists the interrupt sources in simple IIC mode. The STI interrupt is allocated to the transmit end interrupt (SCIn_TEI) request. The receive error interrupt (SCIn_ERI) and the address match (SCIn_AM) request cannot be used.

The DTC or DMAC can also be used to handle transfer in simple IIC mode.

When the SIMR2.IICINTM bit is 1:

- An SCIn_RXI request is generated on the falling edge of the SCLn signal for the 8$^{th}$ bit. If SCIn_RXI is previously set up as an activation source for the DTC or DMAC, the SCIn_RXI request activates the DTC or DMAC to handle transfer of the received data.
- An SCIn_TXI request is generated on the falling edge of the SCLn signal for the 9$^{th}$ bit (acknowledge bit). If SCIn_TXI is previously set up as an activation source for the DTC or DMAC, the SCIn_TXI request activates the DTC or DMAC to handle transfer of the transmit data.

When the SIMR2.IICINTM bit is 0:

- An SCIn_RXI request (ACK detection) is generated if the input on the SDAn pin is low on the rising edge of the SCLn signal for the 9$^{th}$ bit (acknowledge bit)
- An SCIn_TXI request (NACK detection) is generated if the input on the SDAn pin is high on the rising edge of the SCLn signal for the 9$^{th}$ bit (acknowledge bit)
- If SCIn_RXI is previously set up as an activation source for the DTC or DMAC, the SCIn_RXI request activates the DTC or DMAC to handle transfer of the received data.

If the DTC or DMAC is used for data transfer in reception or transmission, always set up and enable the DTC or DMAC before setting up the SCI.

When the IICSTAREQ, IICRSTAREQ, and IICSTPREQ bits in SIMR3 are used to generate a start condition, restart condition, or stop condition, the STI request is issued when generation is complete.

**Table 27.45 SCI interrupt sources (1 of 2)**

| Name | Interrupt source | Interrupt flag | Interrupt enable | DTC or DMAC activation |
|---|---|---|---|---|
| SCIn_RXI (n = 0 to 4, 9) | Reception, ACK detection | — | SCMR.RIE | Possible*1 |

**Table 27.45 SCI interrupt sources (2 of 2)**

| Name | Interrupt source | Interrupt flag | Interrupt enable | DTC or DMAC activation |
|---|---|---|---|---|
| SCIn_TXI (n = 0 to 4, 9) | Transmission, NACK detection | — | SCMR.TIE | Possible |
| SCIn_TEI(STIn) (n = 0 to 4, 9) | Completion of generation of a start, restart, or stop condition | SIMR3.IICSTIF | SCMR.TEIE | Not possible |

Note 1. Activation of the DTC or DMAC is only possible when the SIMR2.IICINTM bit is 1 (use reception and transmission interrupts)

### 27.12.6 Interrupt Requests from the Extended Serial Mode Control Section

The extended serial mode control section has a total of six types of interrupt request for generating the SCIX0 interrupt (Break Field low width detected), SCIX1 interrupt (Control Field 0 match, Control Field 1 match, priority interrupt bit detected), SCIX2 interrupt (bus collision detected), and SCIX3 interrupt (valid edge detected). When any of the interrupt factors is generated, the corresponding status flag is set to 1. Details of all of the interrupt requests are listed in Table 27.46.

**Table 27.46 Interrupt Sources of the Extended Serial Mode Control Section**

| Interrupt Request | Status Flag | Interrupt Factors |
|---|---|---|
| SCIX0 interrupt (Break Field low width detected) | BFDF | • Detection of a Break Field low width longer than the interval corresponding to the timer setting<br>• Completion of the output of a Break Field low width over the interval corresponding to the timer setting<br>• Underflow of the timer |
| SCIX1 interrupt (Control Field 0 match) | CF0MF | The data received in Control Field 0 matching the value set in CF0DR |
| SCIX1 interrupt (Control Field 1 match) | CF1MF | The data received in Control Field 1 matching the value set in PCF1DR or SCF1DR |
| SCIX1 interrupt (priority interrupt bit detected) | PIBDF | The value of the bit specified as the priority interrupt bit matching the value set in PCF1DR |
| SCIX2 interrupt (bus collision detected) | BCDF | The output level on the RXDXn (n = 1, 2) pin and the input level on the RXDXn pin not matching on three consecutive cycles of the bus collision detection clock |
| SCIX3 interrupt (valid edge detected) | AEDF | Detection of a valid edge during bit rate measurement |

## 27.13 Event Linking

By using interrupt request signals as event signals, the SCIn can provide linked operation through the ELC for modules selected in advance.

Event signals can be output regardless of the values of the associated interrupt request enable bits.

### (1) Error event output (receive error or error signal detected) (SCIn_ERI, n = 0 to 4, 9)

- Indicates abnormal termination because of a parity error during reception in asynchronous mode
- Indicates abnormal termination because of a framing error during reception in asynchronous mode
- Indicates abnormal termination because of an overrun error during reception
- Indicates detection of the error signal during transmission in smart card interface mode
- The SSR_FIFO.FER and PER flags are 0, and receive data less than the receive FIFO data trigger number is set in a reception FIFO buffer, and it indicates that 15 etus elapse when FIFO is selected and the FCR.DRES bit is 1

### (2) Receive data full event output (SCIn_RXI, n = 0 to 4, 9)

- Indicates that ACK is detected if the SIMR2.IICINTM bit is 0 in simple IIC mode
- Indicates that the 8th-bit SCLn falling edge is detected if the SIMR2.IICINTM bit is 1 in simple IIC mode
- When the SIMR2.IICINTM bit is 1 during master transmission in simple IIC mode, set the ELC so that receive data full events are not used

**Non-FIFO selected**

- Indicates that received data is set in the Receive Data Register (RDR or RDRHL).

**FIFO selected**

- Using this event output is prohibited.

### (3) Transmit data empty event output (SCIn_TXI, n = 0 to 4, 9)

- Indicates that the SCR/SCR_SMCI.TE bit is changed from 0 to 1
- Indicates that transmission is complete in smart card interface mode
- Indicates that NACK is detected if the SIMR2.IICINTM bit is 0 in simple IIC mode
- Indicates that the 9th-bit SCLn falling edge is detected if the SIMR2.IICINTM bit is 1 in simple IIC mode

**Non-FIFO selected**

- Indicates that transmit data is transferred from the Transmit Data Register (TDR or TDRHL) to the Transmit Shift Register (TSR).

**FIFO selected**

- Using this event output is prohibited.

### (4) Transmit end event output (SCIn_TEI, n = 0 to 4, 9)

- Indicates the completion of transmission
- Indicates that the starting condition, resumption condition, or termination condition is generated in simple IIC mode

Note: When FIFO is selected, using this event output is prohibited

### (5) Address match event output (SCIn_AM, n = 0, 3, 4, 9)

- Indicates a match of the comparison data (CDR.CMPD) with one frame of receive data when DCCR.DCME is set to 1 in asynchronous mode, including multi-processor mode.

## 27.14 Address Non-match Event Output (SCI0_DCUF)

SCI0_DCUF indicates the non-match of comparison data (CDR.CMPD) with receive data that is one frame of the data that is received when DCCR.DCME is set to 1 in asynchronous mode, including multi-processor mode. This event can be used for Snooze end request only. For details, see section 10, Low Power Modes.

## 27.15 Noise Cancellation Function

Figure 27.117 shows the configuration of the noise filter used for noise cancellation. The noise filter consists of a 2-stage flip-flop circuit and a match detection circuit. When the input signals of the noise filter and the output signals of the 2- stage flip-flop circuits completely match, the matched level is conveyed as an internal signal. Unless otherwise matched, the previous value is retained. When the same level is retained for 3 cycles or longer on the sampling clock of the noise filter, it is considered as a valid receive signal. A change in pulse for 3 cycles or shorter is considered as noise, not as a receive signal.

In asynchronous mode, the noise cancellation function can be applied to the receive signal input to the RXDn pin. The receive level of the RXDn is taken in the flip-flop circuit of the noise filter on the base clock of asynchronous mode.

- When SEMR.ABCS = 0 and SEMR.ABCSE = 0, the cycle is 1/16 of a 1-bit period.
- When SEMR.ABCS = 1 and SEMR.ABCSE = 0, the cycle is 1/8 of a 1-bit period.
- When SEMR.ABCSE = 1, the cycle is 1/6 of a 1-bit period.

In simple IIC mode, this function can be used for each input on SDAn and SCLn. The sampling clock is selected from divided clock of baud rate generator settings SNFR.NFCS[2:0].

If the base clock is stopped once with the noise filter enabled and then the base clock input is restarted again, the noise filter operation resumes from the state where the clock was stopped. When SCR.TE and SCR.RE are set to 0 during base clock

input, all of the noise filter flip-flop values are initialized to 1. Accordingly, if the input data is 1 when reception operation resumes, the function determines that a level match is detected and the result is conveyed as an internal signal. When the level being input corresponds to 0, the initial output of the noise filter is retained until the level matches in three consecutive sampling cycles.

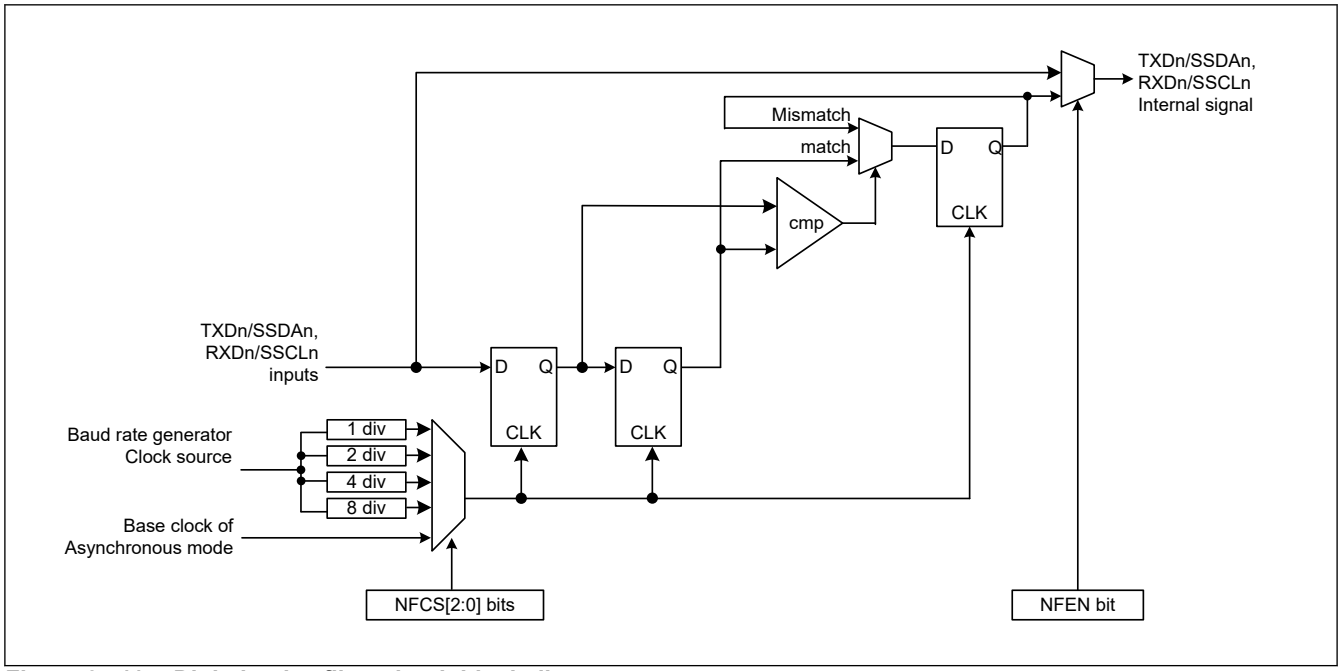



**Figure 27.117 Digital noise filter circuit block diagram**

## 27.16 Usage Notes

### 27.16.1 Settings for the Module-Stop Function

The Module Stop Control Register B (MSTPCRB) can enable or disable SCI operation. The SCI is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

### 27.16.2 SCI Operation during Low Power State

#### (1) Transmission

When setting the module to the stopped state or in transitions to Software Standby, stop operations (by setting the TIE, TE, and TEIE bits in the SCR/SCR_SMCI to 0) after switching the TXDn pin to the general I/O port pin function. When setting I/O port as an SCI connection, the SPTR register can control the state of the TXDn pin. Setting the TE bit to 0 initializes the TSR register and the TEND bit in the SSR/SSR_SMCI is initialized to 1 with non-FIFO selected, and the value is retained, with FIFO selected. Depending on the port settings and SPTR register settings, output pins might output the level before a transition to the low-power state is made after release from the module-stopped state or Software Standby mode. When transitions to these states are made during transmission, the transmitted data becomes indeterminate.

To transmit data in the same transmission mode after cancellation of the low-power state:

1. Set the TE bit to 1.
2. Read SSR/SSR_FIFO/SSR_SMCI.
3. Write data to TDR sequentially to start data transmission.

To transmit data with a different transmission mode, initialize the SCI first.

Figure 27.118 shows an example flow of transition to Software Standby mode during transmission. Figure 27.119 and Figure 27.120 show the port pin states during transition to Software Standby mode.

Before specifying the module-stop state or transitioning to Software Standby mode from the transmission mode using DTC or DMAC transfer, stop the transmit operations (TE = 0). To start transmission after cancellation using the DTC or DMAC, set the TE bit to 1. The SCIn_TXI interrupt flag is set to 1 and transmission starts using the DTC or DMAC.

### (2) Reception

#### When address match function is not used as wakeup condition

Before specifying the module-stop state or transitioning to Software Standby mode, stop the receive operations (RE = 0 in SCR/SCR_SMCI). If transition is made during data reception, the received data is invalid.

Figure 27.121 shows an example flow of transition to Software Standby mode during reception.

#### When address match function is used as wakeup condition

Before specifying the module-stop state or transitioning to Software Standby mode:

1. Set the operations after cancellation of the low power state.
2. Set CDR.CMPD and DCCR.DCME to 1.
3. Set the receive operations (RE = 1 in SCR/SCR_SMCI).
4. Set the module-stop state or Software Standby mode.

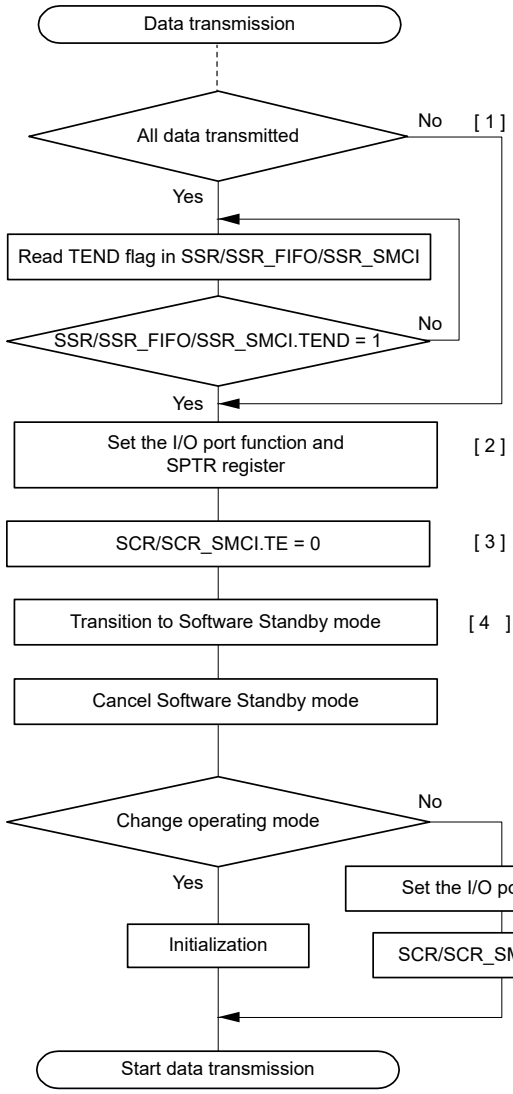
When SCI transfers to low power mode, if the receive data pin (RXD) is at the low level, set SEMR.RXDESEL = 0.

When setting SEMR.RXDESEL = 1, there is a possibility that a start bit (falling edge of RXD pin) cannot be detected on release of the low power mode.

Figure 27.122 shows an example flow of transition to Software Standby mode during reception with address match.

#### When using SCI0 in Snooze mode

When using SCI0 in Snooze mode, some restrictions apply, including maximum bit rates. For details, see section 10, Low Power Modes.




[ 1 ] Data being transmitted is lost. Data can be normally transmitted from the CPU by setting the TE bit in SCR/ SCR_SMCI to 1, reading SSR/SSR_FIFO/ SSR_SMCI, and writing in Software Standby mode. However, if the DMAC or DTC is activated, the data remaining in the DMAC or DTC is transmitted when both the TE and TIE bits in SCR/ SCR_SMCI are set to 1.

[ 2 ] Set the I/O port function and SPTR register settings to switch the TXDn pin to operate as a general I/O port.

[ 3 ] Set SCR/SCR_SMCI.TE bit to 0. If SCR/ SCR_SMCI.TIE = 1 and SCR/SCR_SMCI.TEIE = 1, these are set to 0 simultaneously with the SCR.TE bit.

[ 4 ] This includes the setting for the module-stop state.

**Figure 27.118 Example flow of transition to Software Standby mode during transmission**

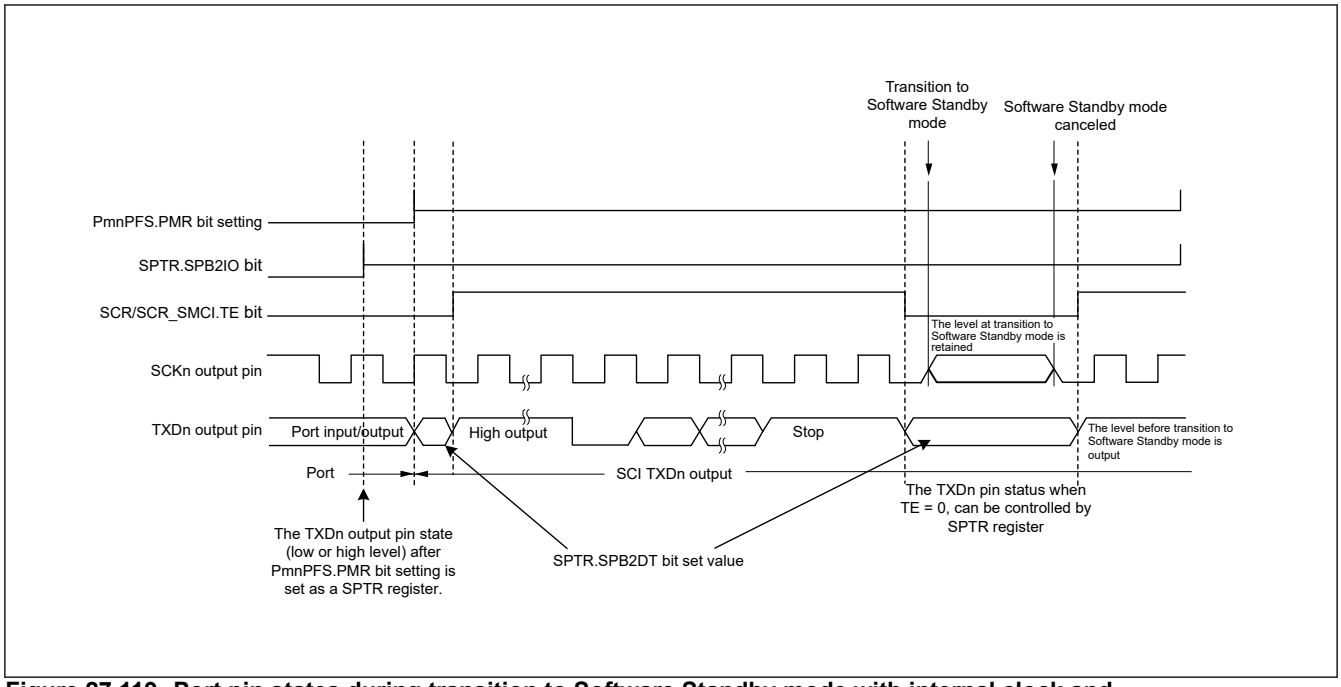



**Figure 27.119 Port pin states during transition to Software Standby mode with internal clock and asynchronous transmission**

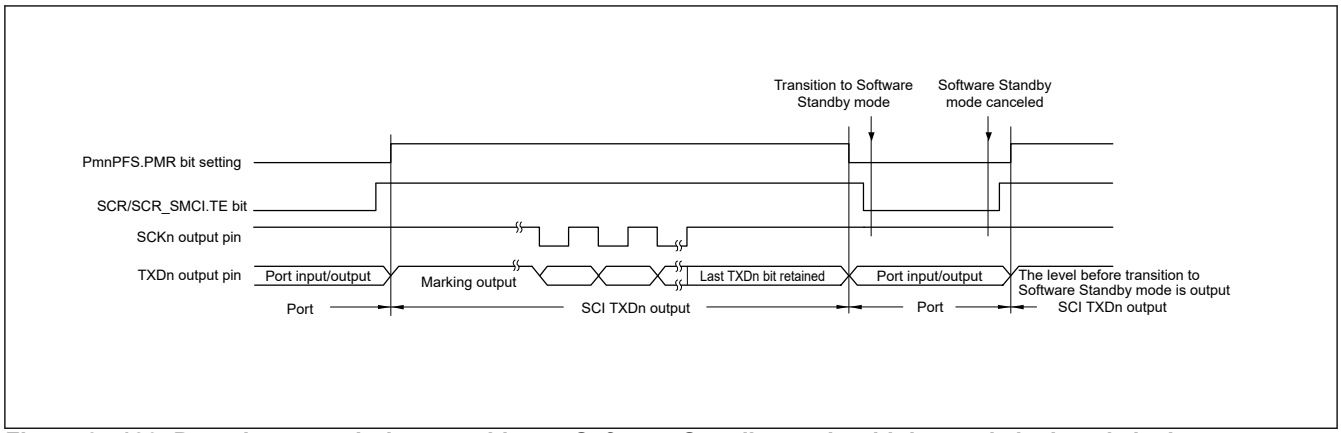



**Figure 27.120 Port pin states during transition to Software Standby mode with internal clock and clock synchronous transmission**

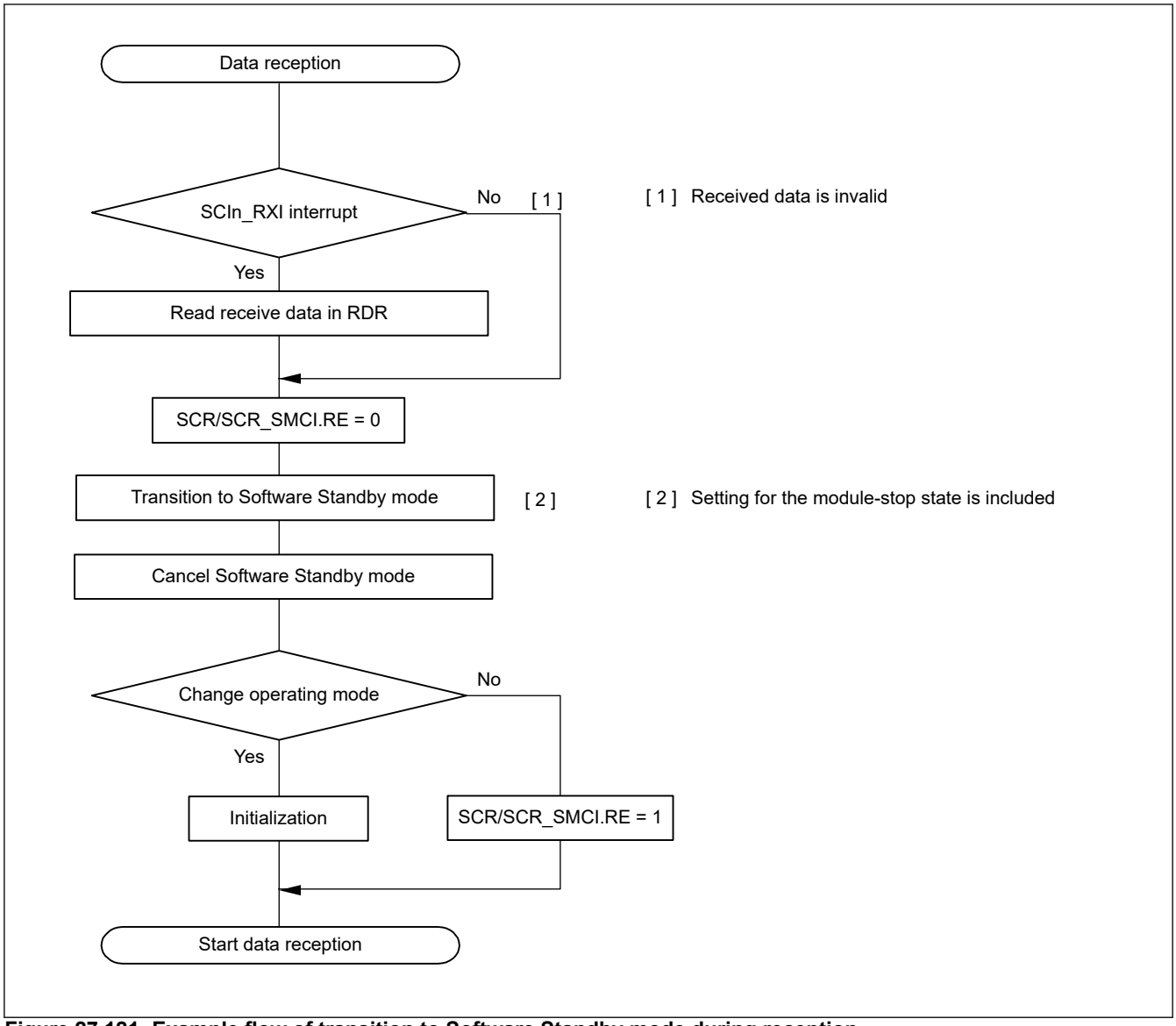



**Figure 27.121 Example flow of transition to Software Standby mode during reception**

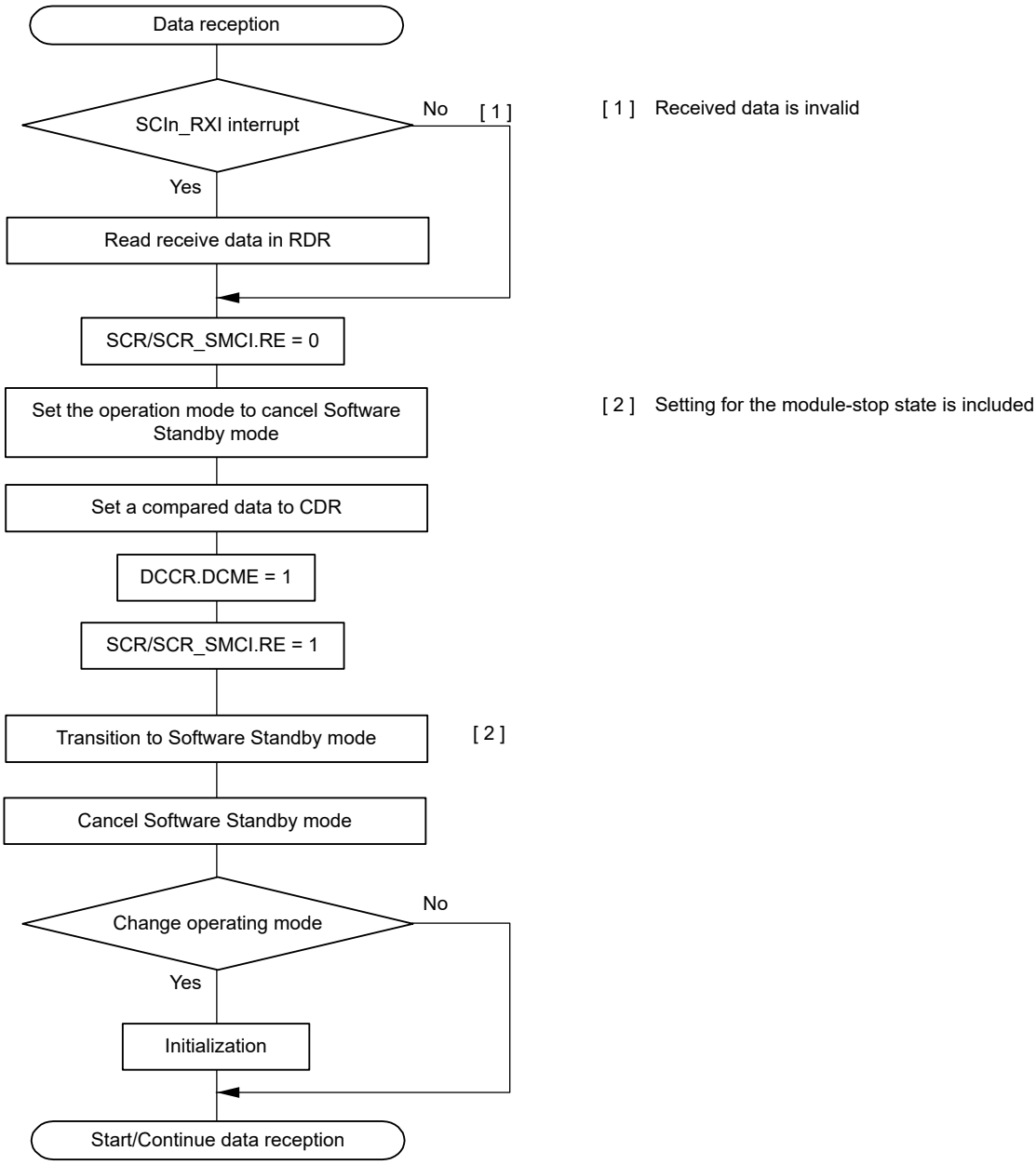



**Figure 27.122 Example flow of transition to Software Standby mode during reception with address match**

## 27.16.3 Break Detection and Processing

**(1) Non-FIFO selected**

When a framing error is detected, a break can be detected by reading the RXDn pin value directly. In a break, the input from the RXDn pin becomes all 0s, and the SSR.FER flag is set to 1 to indicate a framing error, and the SSR.PER flag might also be set to 1 to indicate a parity error. The SCI continues the receive operation even after a break is received. Therefore, even if the FER flag is 0, indicating that no framing error occurred, it is set to 1 again. When the SEMR.RXDESEL bit is 1, the SCI sets the SSR.FER flag to 1 and stops receiving operations until a start bit of the next data frame is detected. If the SSR.FER flag is set to 0, the SSR.FER flag retains 0 during the break.

When the RXDn pin is set to 1 and the break ends, detecting the beginning of the start bit on the first falling edge of the RXDn pin allows the SCI to start the receiving operation.

**(2) FIFO selected**

After a framing error is detected and when the SCI detects that continuous receive data is 0 for 1 frame, reception stops. When a framing error is detected, a break can be detected by reading the SPTR.RXDMON flag value. After the RXDn signal is in high and the break is finished, data reception to the FRDRHL register resumes.

### 27.16.4 Mark State and Production of Breaks

When the SCR/SCR_SMCI.TE bit is 0, disabling serial transmission, the state of the TXDn pin can be set using the SPTR.SPB2IO and SPTR.SPB2DT bits. With this approach, a TXDn pin can be placed in the mark state to transmit a break.

Before setting the SCR/SCR_SMCI.TE bit to 1, enabling serial transmission, set the SPB2IO and SPB2DT bits to put the communication line in the mark state (the state of 1), and change the TXDn pin using I/O port function. To output a break on data transmission, after setting the TXDn pin to output 0 by setting the SPB2IO and SPB2DT bits, change the TXDn pin using the I/O port function and set the SCR/SCR_SMCI.TE bit to 0. When the SCR/SCR_SMCI.TE bit is set to 0, the transmitter is initialized regardless of the current state of transmission.

### 27.16.5 Receive Error Flags and Transmit Operation in Clock Synchronous Mode and Simple SPI Mode

Transmission cannot start when a receive error flag (ORER) in SSR/SSR_FIFO is set to 1, even when data is written to TDR or FTDRL[*1]. Always set the receive error flags to 0 before starting transmission.

Note: The receive error flags cannot be set to 0 when the RE bit in SCR/SCR_SMCI is set to 0 (serial reception is disabled).

Note 1. Do not use the FTDRH register in simple SPI mode.

### 27.16.6 Restrictions on Clock Synchronous Transmission in Clock Synchronous Mode and Simple SPI Mode

When the external clock source is used as a synchronization clock, the following restrictions apply.

#### (1) Start of transmission

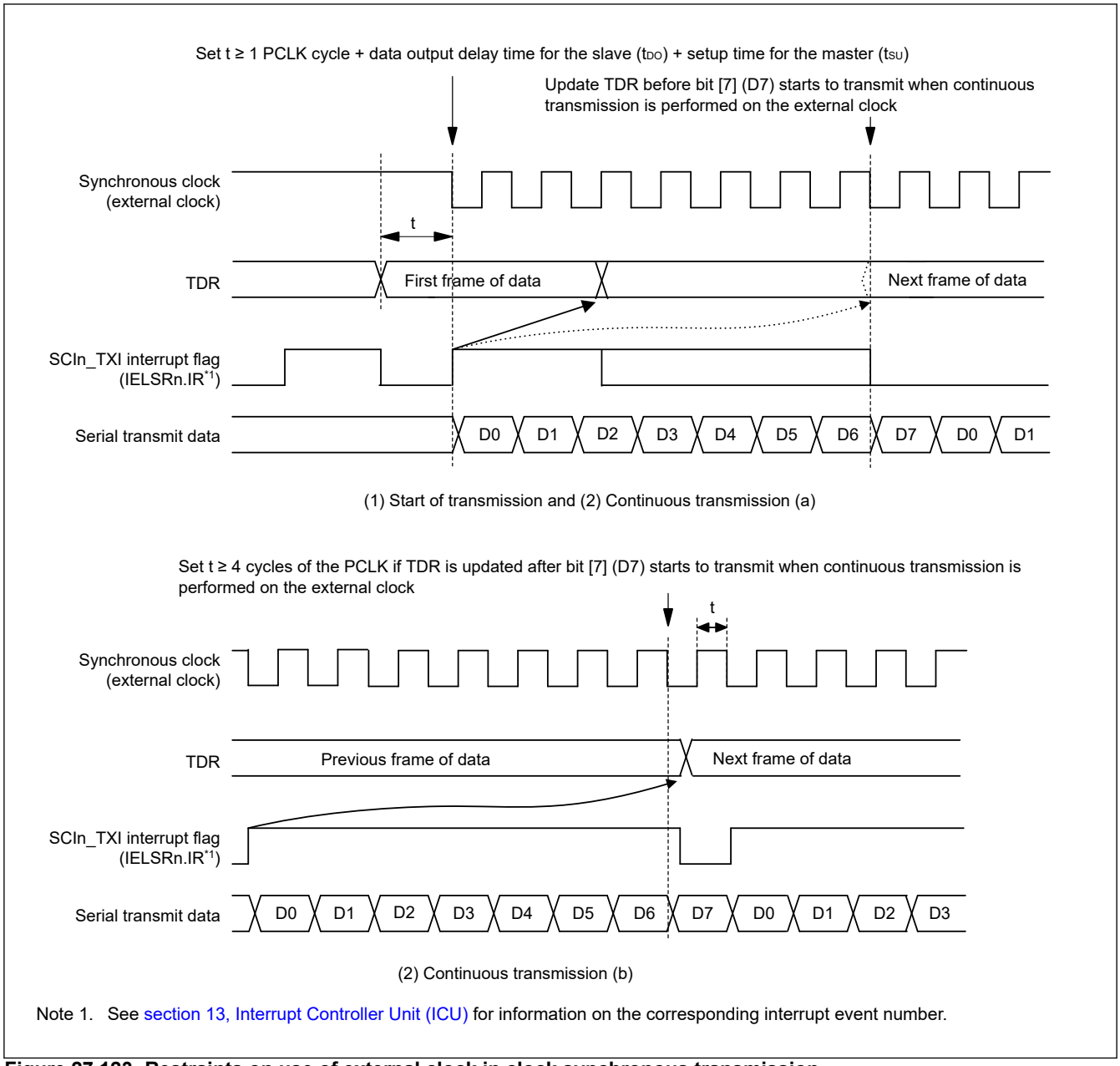
Wait at least the following time from writing transmit data to TDR to the start of the external clock input:

1 PCLK cycle + data output delay time for the slave ($t_{DO}$) + setup time for the master ($t_{SU}$). See Figure 27.123.

#### (2) Continuous transmission

Write the next transmit data to TDR or TDRHL before the falling edge of the transmit clock for bit [7] . See Figure 27.123.

When updating TDR after bit [7] has started to transmit, update TDR while the synchronization clock is in the low-level period, and set the high-level width of the transmit clock (bit [7]) to 4 PCLK cycles or longer. See Figure 27.123.




Note 1. See section 13, Interrupt Controller Unit (ICU) for information on the corresponding interrupt event number.

**Figure 27.123 Restraints on use of external clock in clock synchronous transmission**

## 27.16.7 Restrictions on Using DTC or DMAC

During transmission or reception operations using the DTC or DMAC, do not set transfer data for the DTC or DMAC.

### (1) Writing data to TDR (FTDRHL)

#### Non-FIFO selected

Data can be written to TDR and TDRHL. However, if new data is written to TDR or TDRHL when transmit data remains in TDR or TDRHL, the previous data in TDR and TDRHL is lost because it was not transferred to TSR yet. When using DTC or DMAC, always write transmit data to TDR or TDRHL in the SCIn_TXI interrupt request handling routine.

#### FIFO selected

It is possible to write data to the FTDRH and FTDRL registers when SCR.TE is 1. Confirm the amount of writable data using the FDR.T[4:0] bits.

### (2) Reading data from RDR (FRDRHL)

When using the DTC or DMAC to read RDR and RDRHL, always set the receive data full interrupt (SCIn_RXI) as the activation source of the relevant SCI.

## 27.16.8 Notes on Starting Transfer

At the point where transfer starts when the interrupt status flag (IELSRn.IR flag) in the ICU is 1, follow the procedure in this section to clear interrupt requests before permitting operations (by setting the SCR/SCR_SMCI.TE or SCR/SCR_SMCI.RE bit to 1). For details on the interrupt status flag, see section 13, Interrupt Controller Unit (ICU).

1. Confirm that transfer has stopped (the SCR/SCR_SMCI.TE or SCR/SCR_SMCI.RE bit is 0)
2. Set the associated interrupt enable bit (SCR/SCR_SMCI.TIE or SCR/SCR_SMCI.RIE bit) to 0
3. Read the associated interrupt enable bit (SCR/SCR_SMCI.TIE or SCR/SCR_SMCI.RIE bit) to check that it actually becomes 0
4. Set the interrupt status flag, IELSRn.IR, in the ICU to 0

## 27.16.9 External Clock Input in Clock Synchronous Mode and Simple SPI Mode

In clock synchronous mode and simple SPI mode, the external clock SCKn must be input as follows:

High-pulse period, low-pulse period = 2 PCLK cycles or more, period = 6 PCLK cycles or more.

## 27.16.10 Limitations on Simple SPI Mode

### (1) Master mode

- Use a resistor to pull up or pull down the clock line matching the initial settings for the transfer clock set in the SPMR.CKPH and CKPOL bits when the SPMR.SSE bit is 1.

  This prevents the clock line from being placed in the high-impedance state when the SCR.TE bit is set to 0 or unexpected edges from being generated on the clock line when the SCR.TE bit changes from 0 to 1. When the SPMR.SSE bit is 0 in single master mode, pulling up or pulling down the clock line is not required because the clock line is not placed in the high-impedance state even when the SCR.TE bit is set to 0.
- For the clock delay setting (SPMR.CKPH bit is 1), the receive data full interrupt (SCIn_RXI) is generated before the final clock edge on the SCKn pin as indicated in Figure 27.124. If the TE and RE bits in the SCR register become 0 before the final edge of the clock signal on the SCKn pin, the SCKn pin is placed in the high-impedance state, so the width of the last clock pulse of the transfer clock is shortened. Additionally, an SCIn_RXI interrupt might lead to the input signal on the SSn pin of a connected slave going to the high level before the final edge of the clock signal on the SCKn pin, leading to incorrect operation of the slave.
- In a multi-master configuration, the SCKn pin output goes to high-impedance while the input on the SSn pin is at the low level if a mode fault error occurs while a character is being transferred, stopping supply of the clock signal to the connected slave. Reset the connected slave to avoid misaligned bits when transfer is restarted.

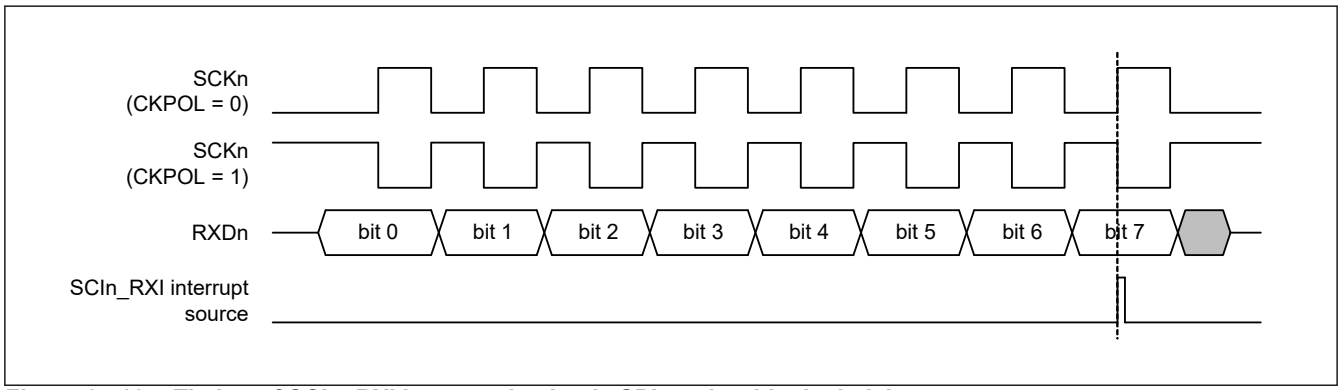



**Figure 27.124 Timing of SCIn_RXI interrupt in simple SPI mode with clock delay**

### (2) Slave mode

- Wait at least the following time from writing transmit data in the TDR register to the start of the external clock input.

  1 PCLK cycle + data output delay for the slave ($t_{DO}$) + setup time for the master ($t_{SU}$)

  Also wait at least 5 PCLK cycles from the input of the low level on the SSn pin to the start of the external clock input.
- Provide an external clock signal to the master the same as the data length for transfer
- Control the input on the SSn pin before the start and after the end of data transfer
- When the input level on the SSn pin is to be changed from low to high while a character is being transferred, set the TE and RE bits in the SCR register to 0 and, after restoring the settings, restart transfer of the first byte

## 27.16.11 Notes on Transmit Enable Bit (SCR.TE)

In initial register value, when SCR.TE = 0, the state of the TXDn pin is high impedance. The TXDn line should not be high impedance by the following one of ways.

1. The pull-up resistance is connected to the TXDn line.
2. Before setting the SCR.TE bit to 0, the function of the pin should be changed to a general-purpose output port. After that, set the SCR.TE bit to 1, and then change the function of the pin to TXDn.
3. In asynchronous mode, set SPTR and decided level of TXDn pin during SCR.TE = 0.

In the Simple SPI mode slave operation, the MISOn pin operates in the same way as the above TXDn pin. The MISOn pin, the same as TXDn pin, should not be high impedance by the above list number 1 or list number 2.

## 27.16.12 Limitation 1 on Usage of the Extended Serial Mode Control Section

When the PCR.SHARPS bit is set to 1, output on the TXDXn/RXDXn (n = 1, 2) pin is only possible when the following conditions apply.

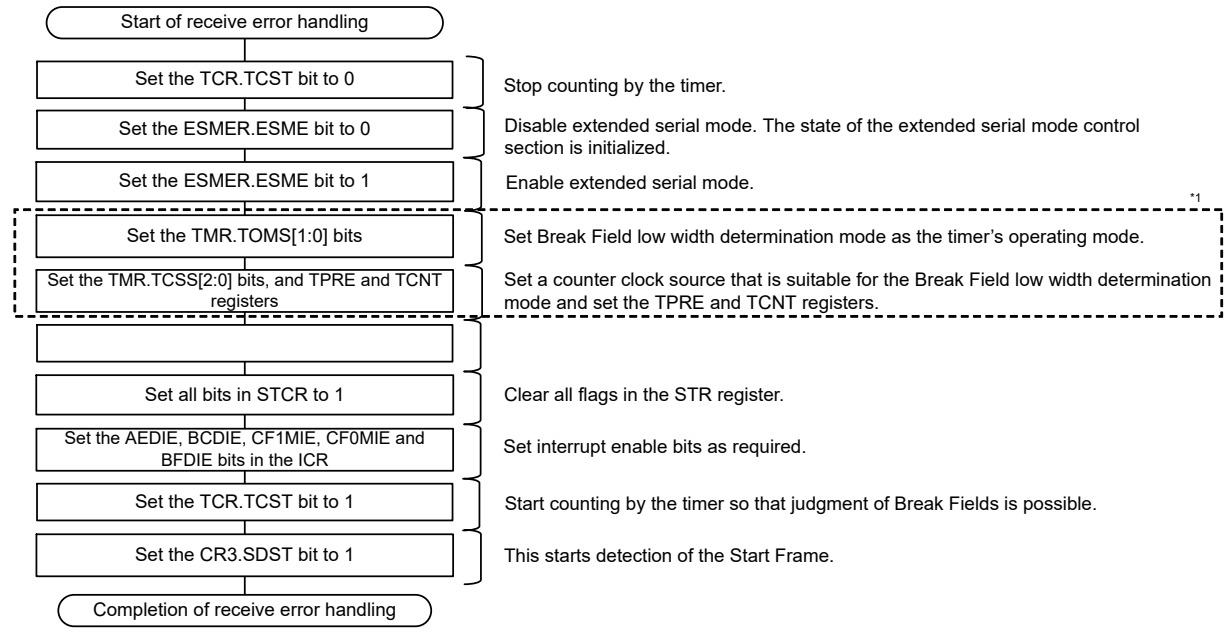
- The timer of the SCIh module is in Break Field low width output mode and the value of the TCR.TCST bit is 1 (when the TCST bit is set to 1, the high level continues to be output for up to one cycle of the clock source for counting by the timer counter before output of the low level)
- The value of the SCIn.SCR.TE bit is 1.

## 27.16.13 Limitation 2 on Usage of the Extended Serial Mode Control Section

An SCIg interrupt request is generated even if the extended serial mode is enabled. However, the SCIg interrupt should not be used during reception of a Start Frame because SCIh uses an SCIg interrupt request.

The two ways of dealing with this are described below. When a receive error is detected, clear the error flag of the SCIg and initialize the control section of the SCIh.

1. Set the SCR.RIE bit of the SCIg to 0 to disable the output of interrupt requests. Check the error flags in the SSR register for SCIg on completion of the reception of a Start Frame, because an ERI interrupt is not generated if a receive error occurs. After reception of the Start Frame is completed, set the SCR.RIE bit of the SCIg to 1 by the time the first byte of the Information Frame is received.
2. Set the SCR.RIE bit of the SCIg to 1 to disable RXI interrupts and enable ERI interrupts for ICU.
   Clear the IRn.IR flag to enable the acceptance of RXI interrupts by ICU by the time the first byte of the Information Frame is received after the completion of Start Frame reception.




Note 1. The setting in this step is not necessary if the setting conditions have not been changed.

**Figure 27.125 Example of Flowchart for Receive Error Handling (during Reception of the Start Frame)**

## 27.16.14 Note on Stopping Reception When Using the RTS Function in Asynchronous Mode

One clock cycle of PCLK is required for the time from setting the SCR.RE bit to 0 to stopping the RTS signal generator in asynchronous mode.

When reading the RDR (or RDRL) register after setting the SCR.RE bit to 0, confirm that the RE bit has been set to 0 before reading the RDR (or RDRL) register to prevent these two processes from being performed consecutively.

# 28. I2C Bus Interface (IIC)

## 28.1 Overview

The I2C bus interface (IIC) has 2 channels. The IIC module conforms with and provides a subset of the NXP I2C (Inter-Integrated Circuit) bus interface functions.

Table 28.1 lists the IIC specifications, Figure 28.1 shows a block diagram, and Figure 28.2 shows an example of I/O pin connections to external circuits, with an I2C bus configuration. Table 28.2 lists the I/O pins.

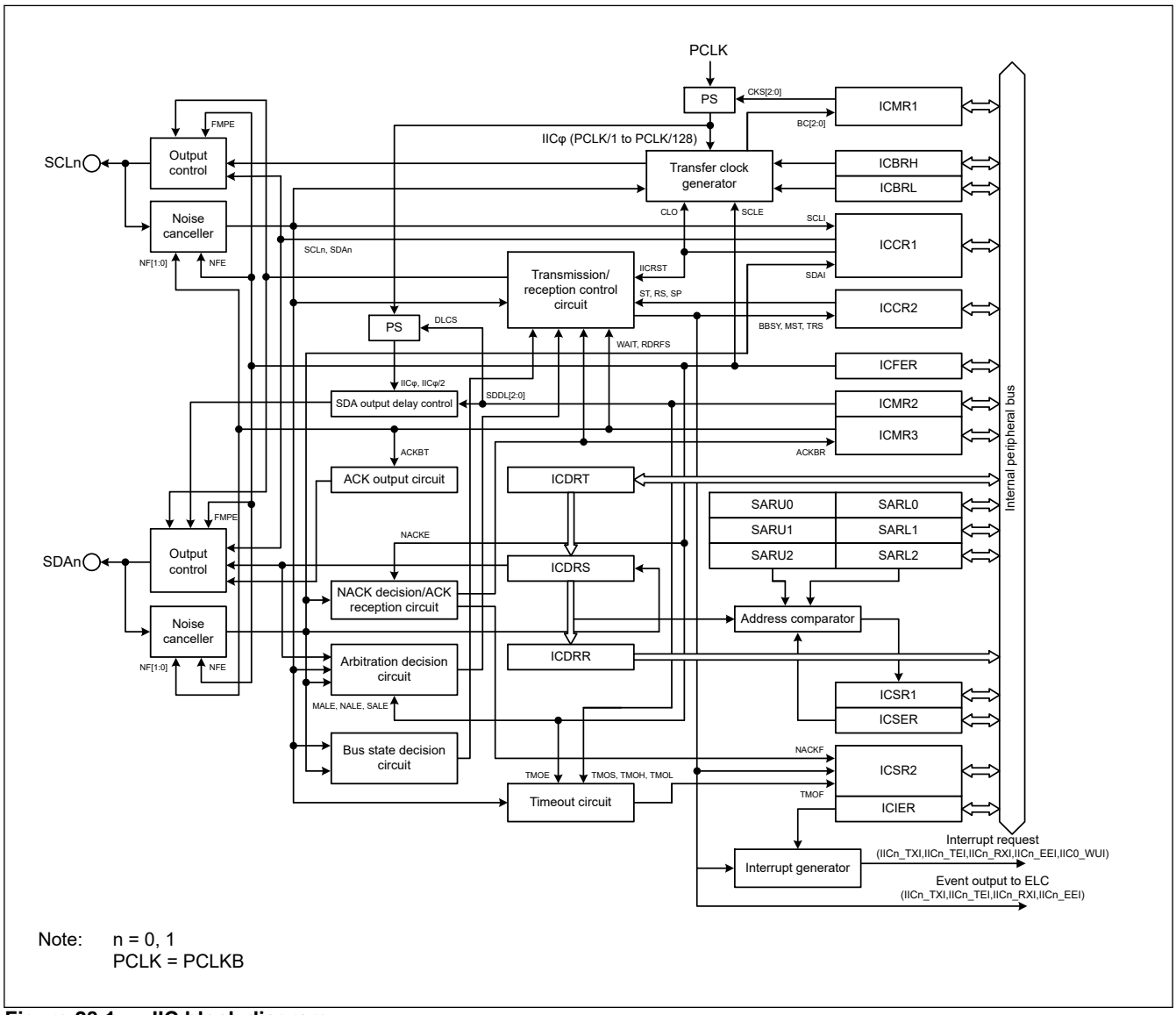
**Table 28.1 IIC specifications (1 of 2)**

| Parameter | Specifications |
|---|---|
| Communications format | • I2C-bus format or SMBus format<br>• Master or slave mode selectable<br>• Automatic securing of the setup times, hold times, and bus-free times for the transfer rate |
| Transfer rate | • Fast-mode Plus supported, up to 1 Mbps |
| SCL clock | For master operation, the duty cycle of the SCL clock is selectable in the range from 4% to 96% |
| Issuing and detecting conditions | • Start, restart, and stop conditions are automatically generated<br>• Start conditions (including restart conditions) and stop conditions are detectable |
| Slave address | • Configurable for up to three different slave addresses<br>• 7- and 10-bit address formats supported, including simultaneous use<br>• General call addresses, device ID addresses, and SMBus host addresses detectable |
| Acknowledgment | • For transmission, automatic loading of the acknowledge bit<br>Transfer of the next transmit data can be automatically suspended on detection of a not-acknowledge bit.<br>• For reception, automatic transmission of the acknowledge bit<br>If a wait between the 8th and 9th clock cycles is selected, the software can control the value in the acknowledge field in response to the received value. |
| Wait function | During reception, the following wait periods are available by holding the SCL clock low:<br>• Waiting between the eighth and ninth clock cycles<br>• Waiting between the ninth clock cycle and the 1st clock cycle of the next transfer |
| SDA output delay function | Output timing of transmitted data, including the acknowledge bit, can be delayed |
| Arbitration | • For multi-master operation:<br>– SCL clock synchronization is possible when conflict occurs with the SCL signal from another master<br>– When issuing the start condition creates conflict on the bus, loss of arbitration is detected by testing for a mismatch between the internal signal for the SDA line and the level on the SDA line<br>– In master operation, loss of arbitration is detected by testing for non-matching between the signal on the SDA line and the internal signal for the SDA line<br>• Loss of arbitration because the start condition occurs while the bus is busy is detectable, to prevent the issuing of double start conditions<br>• Loss of arbitration is detectable on transfer of a not-acknowledge bit because the internal signal for the SDA line and the level on the SDA line do not match<br>• Loss of arbitration because a mismatch of internal and line levels for data is detectable in slave transmission |
| Timeout function | Internal detection of long-interval stops of the SCL clock |
| Noise cancellation | • Digital noise filters for both the SCL and SDA signals<br>• Programmable window for noise cancellation by the filters |
| Interrupt sources | • Transfer error or event occurrence (arbitration-lost, NACK, timeout, start or restart condition, or stop condition)<br>• Receive data full, including matching with a slave address<br>• Transmit data empty, including matching with a slave address<br>• Transmit end |
| Module-stop function | Module-stop state can be set |
| IIC operating modes | • Master transmit<br>• Master receive<br>• Slave transmit<br>• Slave receive |

**Table 28.1 IIC specifications (2 of 2)**

| Parameter | Specifications |
|---|---|
| Event link function (output) | • Transfer error or event occurrence (arbitration-lost, NACK, timeout, start or restart condition, or stop condition)<br>• Receive data full, including matching with a slave address<br>• Transmit data empty, including matching with a slave address<br>• Transmit end |
| Wakeup function*1 | CPU can return from Software Standby mode using a wakeup event |
| TrustZone Filter | Security attribution can be set for each channels |

Note 1. This function is only available for IIC channel IIC0. IIC1 is not supported.




Note: n = 0, 1
PCLK = PCLKB

**Figure 28.1 IIC block diagram**

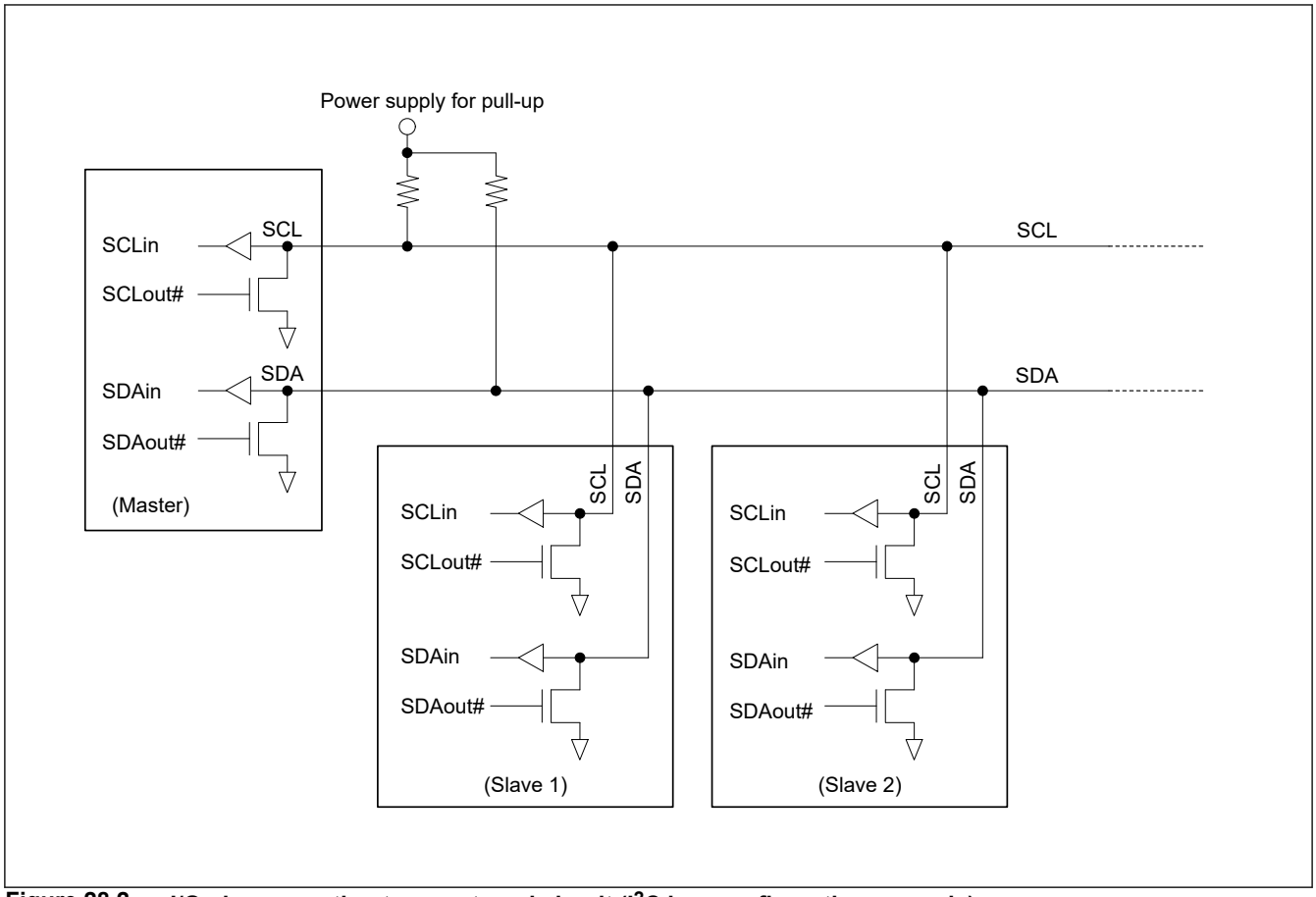



**Figure 28.2 I/O pin connection to an external circuit ($I^2C$ bus configuration example)**

The input level of the signals for IIC is CMOS when $I^2C$ bus is selected (ICMR3.SMBS = 0), or TTL when SMBus is selected (ICMR3.SMBS = 1).

**Table 28.2 IIC I/O pins**

| Channel | Pin name | I/O | Function |
|---|---|---|---|
| IICn | SCLn | I/O | IICn serial clock I/O pin |
| | SDAn | I/O | IICn serial data I/O pin |

Note: n = 0, 1

## 28.2 Register Descriptions

### 28.2.1 ICCR1 : $I^2C$ Bus Control Register 1

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | ICE | IICRST | CLO | SOWP | SCLO | SDAO | SCLI | SDAI |
| Value after reset: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SDAI | SDA Line Monitor<br>0: SDAn line is low<br>1: SDAn line is high | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | SCLI | SCL Line Monitor<br>0: SCLn line is low<br>1: SCLn line is high | R |
| 2 | SDAO | SDA Output Control/Monitor<br>0: Read: IIC drives SDAn pin low<br>Write: IIC drives SDAn pin low<br>1: Read: IIC releases SDAn pin<br>Write: IIC releases SDAn pin | R/W |
| 3 | SCLO | SCL Output Control/Monitor<br>Use an external pull-up resistor to drive the signal high.<br>0: Read: IIC drives SCLn pin low<br>Write: IIC drives SCLn pin low<br>1: Read: IIC releases SCLn pin<br>Write: IIC releases SCLn pin | R/W |
| 4 | SOWP | SCLO/SDAO Write Protect<br>This bit is read as 1.<br>0: Write enable SCLO and SDAO bits<br>1: Write protect SCLO and SDAO bits | W |
| 5 | CLO | Extra SCL Clock Cycle Output<br>This bit clears automatically after 1 clock cycle is output.<br>0: Do not output extra SCL clock cycle (default)<br>1: Output extra SCL clock cycle | R/W |
| 6 | IICRST | I2C Bus Interface Internal Reset<br>This setting clears the bit counter and the SCLn/SDAn output latch.<br>0: Release IIC reset or internal reset<br>1: Initiate IIC reset or internal reset | R/W |
| 7 | ICE | I2C Bus Interface Enable<br>Used in combination with the IICRST bit to select either IIC or internal reset.<br>0: Disable (SCLn and SDAn pins in inactive state)<br>1: Enable (SCLn and SDAn pins in active state) | R/W |

### SDAO bit (SDA Output Control/Monitor) and SCLO bit (SCL Output Control/Monitor)

The SDAO and SCLO bits directly control the SDAn and SCLn signals output from the IIC. When writing to these bits, also write 0 to the SOWP bit. Setting these bits results in input to the IIC by the input buffer. When slave mode is selected, a start condition might be detected and the bus might be released, depending on the bit settings.

Do not rewrite these bits during a start condition, stop condition, restart condition, transmission, or reception. Operation after rewriting under these conditions is not guaranteed. When reading these bits, the state of signals output from the IIC can be read.

### CLO bit (Extra SCL Clock Cycle Output)

The CLO bit allows output of an extra SCL clock cycle for debugging or error processing. Normally, set this bit to 0. Setting the bit to 1 in a normal communication state causes a communication error. For details on this function, see section 28.12.2. Extra SCL Clock Cycle Output Function.

### IICRST bit (I2C Bus Interface Internal Reset)

The IICRST bit initiates an internal state reset of the IIC. Setting this bit to 1 initiates an IIC reset or internal reset. Whether an IIC reset or internal reset is initiated is determined by the settings of this bit in combination with the ICE bit. Table 28.3 lists the IIC resets.

The IIC reset initializes all registers except ICCR1.ICE and ICCR1.IICRST bits and internal states of the IIC. In addition to the internal states of the IIC, the internal reset initializes the following:

- Bit counter (ICMR1.BC[2:0] bits)
- I2C Bus Shift Register (ICDRS)
- I2C Bus Status Registers (ICSR1 and ICSR2)
- SDAO and SCLO Output Control/Monitor (ICCR1.SDAO and ICCR1.SCLO bits)

- I²C Bus Control Register 2 (except ICCR2.BBSY bit)

For the reset conditions for each register, see section 28.15. State of Registers When Issuing Each Condition.

An internal reset initiated with the IICRST bit set to 1 during operation (with the ICE bit set to 1) resets the internal states of the IIC without initializing the port settings and the control and setting registers of the IIC. If the IIC hangs in a low-level output state, resetting the internal states cancels the low-level output state and releases the bus with the SCLn pin and SDAn pin at high impedance.

Note: If an internal reset is initiated using the IICRST bit for a bus hang-up that occurs during communication with the master device in slave mode, the slave and master devices might enter different states, because the bit counter information differs. For this reason, do not initiate an internal reset in slave mode. Initiate recovery processing from the master device. If an internal reset is required because the IIC hangs with the SCLn line in a low-level output state in slave mode, initiate an internal reset, then issue a restart condition from the master device, or issue a stop condition and resume communication from the start condition. If communication is restarted by initiating a reset solely in the slave device without issuing a start or restart condition from the master device, synchronization is lost because the master and slave devices operate asynchronously.

**Table 28.3 IIC resets**

| IICRST | ICE | State | Specifications |
|---|---|---|---|
| 1 | 0 | IIC reset | Resets all registers except ICCR1.IICRST and ICCR1.ICE bits, and the internal states of the IIC |
| | 1 | Internal reset | Reset the following:<br>• ICMR1.BC[2:0] bits<br>• ICSR1, ICSR2, ICDRS registers<br>• SDAO and SCLO Output Control/Monitor (ICCR1.SDAO and ICCR1.SCLO bits)<br>• I²C Bus Control Register 2 (except ICCR2.BBSY bit)<br>• Internal states of the IIC |

### ICE bit (I²C Bus Interface Enable)

The ICE bit selects the active or inactive state of the SCLn and SDAn pins. It can also be combined with the IICRST bit to initiate two types of resets. See Table 28.3 for the reset descriptions.

Set the ICE bit to 1 when using the IIC. The SCLn and SDAn pins are placed in the active state when the ICE bit is set to 1. Set the ICE bit to 0 when the IIC is not used. The SCLn and SDAn pins are placed in the inactive state when the ICE bit is set to 0. Do not assign the SCLn or SDAn pin to the IIC when setting up the pin function control. Slave address comparison is performed if the pins are assigned to the IIC.

## 28.2.2 ICCR2 : I²C Bus Control Register 2

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x01

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | BBSY | MST | TRS | — | SP | RS | ST | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | This bit is read as 0. The write value should be 0. | R/W |
| 1 | ST | Start Condition Issuance Request<br>0: Do not issue a start condition request<br>1: Issue a start condition request | R/W |
| 2 | RS | Restart Condition Issuance Request<br>0: Do not issue a restart condition request<br>1: Issue a restart condition request | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | SP | Stop Condition Issuance Request<br>0: Do not issue a stop condition request<br>1: Issue a stop condition request | R/W |
| 4 | — | This bit is read as 0. The write value should be 0. | R/W |
| 5 | TRS | Transmit/Receive Mode<br>0: Receive mode<br>1: Transmit mode | R/W*1 |
| 6 | MST | Master/Slave Mode<br>0: Slave mode<br>1: Master mode | R/W*1 |
| 7 | BBSY | Bus Busy Detection Flag<br>0: I2C bus released (bus free state)<br>1: I2C bus occupied (bus busy state) | R |

Note 1. The MST and TRS bits can be written to when the ICMR1.MTWP bit is set to 1.

### ST bit (Start Condition Issuance Request)

The ST bit requests transition to master mode and triggers a start condition. When this bit is set to 1, a start condition is issued when the BBSY flag is set to 0 (bus free state). For details on this function, see section 28.11. Start, Restart, and Stop Condition Issuing Function.

[Setting condition]

- When 1 is written to the ST bit.

[Clearing conditions]

- When 0 is written to the ST bit
- When a start condition is issued (a start condition is detected)
- When the AL (arbitration-lost) flag in ICSR2 is set to 1
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

Note: Only set the ST bit to 1 (start condition request) when the BBSY flag is set to 0 (bus free state). Arbitration might be lost if the ST bit is set to 1 (start condition request) when the BBSY flag is 1 (bus busy state).

### RS bit (Restart Condition Issuance Request)

The RS bit requests that a restart condition be issued in master mode. When this bit is set to 1 to request a restart condition, a restart condition is issued when the BBSY flag is set to 1 (bus busy state) and the MST bit is set to 1 (master mode). For details on this function, see section 28.11. Start, Restart, and Stop Condition Issuing Function.

[Setting condition]

- When 1 is written to the RS bit with the BBSY flag in ICCR2 set to 1.

[Clearing conditions]

- When 0 is written to the RS bit
- When a restart condition is issued (a start condition is detected)
- When the AL (arbitration-lost) flag in ICSR2 is set to 1
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

Note: Do not set the RS bit to 1 while issuing a stop condition.
Note: If 1 (restart condition request) is written to the RS bit in slave mode, the restart condition is not issued, but the RS bit remains set to 1. If the operating mode changes to master mode without the bit being cleared, a restart condition might be issued.

### SP bit (Stop Condition Issuance Request)

The SP bit requests that a stop condition be issued in master mode. When this bit is set to 1, a stop condition is issued when the BBSY flag is set to 1 (bus busy state) and the MST bit is set to 1 (master mode). For details on this function, see section 28.11. Start, Restart, and Stop Condition Issuing Function.

[Setting condition]

- When 1 is written to the SP bit with both the BBSY flag and the MST bit in ICCR2 set to 1.

[Clearing conditions]

- When 0 is written to the SP bit
- When a stop condition is issued (a stop condition is detected)
- When the AL (arbitration-lost) flag in ICSR2 is set to 1
- When a start condition and a restart condition are detected
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

Note: Writing to the SP bit is not possible while the BBSY flag is 0 (bus free state).
Note: Do not set the SP bit to 1 while a restart condition is being issued.

### TRS bit (Transmit/Receive Mode)

The TRS bit indicates transmit or receive mode. The IIC is in receive mode when the TRS bit is 0 and in transmit mode when the bit is 1. The combination of this bit and the MST bit indicates the IIC operating mode.

The value of the TRS bit automatically changes to 1 for transmit mode or 0 for receive mode when a start condition is issued or detected and the R/W# bit is set. Although writing to the TRS bit is possible when the MTWP bit in ICMR1 is set to 1, writing to this bit is not required during normal usage.

[Setting conditions]

- When a start condition is issued normally because of a start condition request (when a start condition is detected with the ST bit set to 1)
- When a restart condition is issued normally because of a restart condition request (when a restart condition is detected with the RS bit set to 1)
- When the R/W# bit appended to the slave address is set to 0 in master mode
- When the address received in slave mode matches the address enabled in ICSER, with the R/W# bit set to 1
- When 1 is written to the TRS bit with the MTWP bit in ICMR1 set to 1.

[Clearing conditions]

- When a stop condition is detected
- When the AL (arbitration-lost) flag in ICSR2 is set to 1
- When the R/W# bit appended to the slave address is set to 1 in master mode
- In slave mode, on a match between the received address and the address enabled in ICSER when the value of the received R/W# bit is 0, including when the received address is the general call address
- In slave mode, when a restart condition is detected (a restart condition is detected with ICCR2.BBSY = 1 and ICCR2.MST = 0)
- When 0 is written to the TRS bit with the MTWP bit in ICMR1 set to 1
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

### MST bit (Master/Slave Mode)

The MST bit indicates master or slave mode. The IIC is in slave mode when the MST bit is 0 and is in master mode when the bit is 1. The combination of this bit and the TRS bit indicates the IIC operating mode.

The value of the MST bit automatically changes to 1 for master mode or 0 for slave mode when a start condition is issued or a stop condition is issued or detected. Although writing to the MST bit is possible when the MTWP bit in ICMR1 is set to 1, writing to this bit is not required during normal usage.

[Setting conditions]

- When a start condition is issued normally because of a start condition request (when a start condition is detected with the ST bit set to 1)
- When 1 is written to the MST bit with the MTWP bit in ICMR1 set to 1.

[Clearing conditions]

- When a stop condition is detected
- When the AL (arbitration-lost) flag in ICSR2 is set to 1
- When 0 is written to the MST bit with the MTWP bit in ICMR1 set to 1
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

### BBSY flag (Bus Busy Detection Flag)

The BBSY flag indicates whether the $I^2C$ bus is occupied (bus busy state) or released (bus free state). The flag is set to 1 when the SDAn line changes from high to low when the SCLn line is high, assuming that a start condition was issued. The flag is set to 0 if a start condition is not detected for the bus free time (ICBRL setting), assuming that a stop condition was issued.

[Setting condition]

- When a start condition is detected.

[Clearing conditions]

- When a start condition is not detected for the bus free time (ICBRL setting) after detecting a stop condition
- When 1 is written to the IICRST bit in ICCR1 with the ICE bit in ICCR1 set to 0 (IIC reset).

## 28.2.3 ICMR1 : $I^2C$ Bus Mode Register 1

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x02

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | MTWP | CKS[2:0] | | | BCWP | BC[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | BC[2:0] | Bit Counter<br>0 0 0: 9 bits<br>0 0 1: 2 bits<br>0 1 0: 3 bits<br>0 1 1: 4 bits<br>1 0 0: 5 bits<br>1 0 1: 6 bits<br>1 1 0: 7 bits<br>1 1 1: 8 bits | R/W*1 |
| 3 | BCWP | BC Write Protect<br>This bit is read as 1.<br>0: Write enable BC[2:0] bits<br>1: Write protect BC[2:0] bits | W*1 |
| 6:4 | CKS[2:0] | Internal Reference Clock Select<br>Select the internal reference clock source (IICφ) for the IIC.<br>IICφ = (PCLKB / $2^{CKS[2:0]}$) clock | R/W |
| 7 | MTWP | MST/TRS Write Protect<br>0: Write protect MST and TRS bits in ICCR2<br>1: Write enable MST and TRS bits in ICCR2 | R/W |

Note 1. Rewrite the BC[2:0] bits and set the BCWP bit to 0 at the same time.

### BC[2:0] bits (Bit Counter)

The BC[2:0] bits function as a counter indicating the number of bits remaining to be transferred on detection of a rising edge on the SCLn line. Although BC[2:0] are read/write bits, it is not required to access these bits under normal conditions.

To write to these bits, specify the number of bits to be transferred plus one, for an additional acknowledge bit, between transferred frames when the SCLn line is at a low level. The value in the BC[2:0] bits returns to 000b at the end of a data transfer, including the acknowledge bit, or when a start or restart condition is detected.

## 28.2.4 ICMR2 : I2C Bus Mode Register 2

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x03

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DLCS | SDDL[2:0] | | | — | TMOH | TMOL | TMOS |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TMOS | Timeout Detection Time Select<br>0: Select long mode<br>1: Select short mode | R/W |
| 1 | TMOL | Timeout L Count Control<br>0: Disable count while SCLn line is low<br>1: Enable count while SCLn line is low | R/W |
| 2 | TMOH | Timeout H Count Control<br>0: Disable count while SCLn line is high<br>1: Enable count while SCLn line is high | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6:4 | SDDL[2:0] | SDA Output Delay Counter<br>0 0 0: No output delay<br>0 0 1: 1 IICφ cycle (When ICMR2.DLCS = 0 (IICφ))<br>1 or 2 IICφ cycles (When ICMR2.DLCS = 1 (IICφ/2))<br>0 1 0: 2 IICφ cycles (When ICMR2.DLCS = 0 (IICφ))<br>3 or 4 IICφ cycles (When ICMR2.DLCS = 1 (IICφ/2))<br>0 1 1: 3 IICφ cycles (When ICMR2.DLCS = 0 (IICφ))<br>5 or 6 IICφ cycles (When ICMR2.DLCS = 1 (IICφ/2))<br>1 0 0: 4 IICφ cycles (When ICMR2.DLCS = 0 (IICφ))<br>7 or 8 IICφ cycles (When ICMR2.DLCS = 1 (IICφ/2))<br>1 0 1: 5 IICφ cycles (When ICMR2.DLCS = 0 (IICφ))<br>9 or 10 IICφ cycles (When ICMR2.DLCS = 1 (IICφ/2))<br>1 1 0: 6 IICφ cycles (When ICMR2.DLCS = 0 (IICφ))<br>11 or 12 IICφ cycles (When ICMR2.DLCS = 1 (IICφ/2))<br>1 1 1: 7 IICφ cycles (When ICMR2.DLCS = 0 (IICφ))<br>13 or 14 IICφ cycles (When ICMR2.DLCS = 1 (IICφ/2)) | R/W |
| 7 | DLCS | SDA Output Delay Clock Source Select<br>0: Select internal reference clock (IICφ) as the clock source for SDA output delay counter<br>1: Select internal reference clock divided by 2 (IICφ/2) as the clock source for SDA output delay counter*1 | R/W |

Note 1. The setting DLCS = 1 (IICφ/2) is only valid when SCL is low. When SCL is high, the DLCS = 1 setting becomes invalid and the clock source becomes the internal reference clock (IICφ).

### TMOS bit (Timeout Detection Time Select)

The TMOS bit selects long or short mode for the timeout detection time when the timeout function is enabled (ICFER.TMOE = 1). When this bit is set to 0, long mode is selected. When it is set to 1, short mode is selected. In long mode, the timeout detection internal counter functions as a 16 bit-counter. In short mode, the counter functions as a 14- bit counter. While the SCLn line is in the state that enables this counter as specified in the TMOH and TMOL bits, the counter counts up in synchronization with the internal reference clock (IICφ) as a count source. For details on this function, see section 28.12.1. Timeout Function.

### TMOL bit (Timeout L Count Control)

The TMOL bit enables or disables up-counting on the internal counter of the timeout function while the SCLn line is held low and the timeout function is enabled (ICFER.TMOE = 1).

### TMOH bit (Timeout H Count Control)

The TMOH bit enables or disables up-counting on the internal counter of the timeout function while the SCLn line is held high and the timeout function is enabled (ICFER.TMOE = 1).

### SDDL[2:0] bits (SDA Output Delay Counter)

The SDDL[2:0] bits can be used to delay the SDA output. This counter works with the clock source selected in the DLCS bit. This setting can be used for all types of SDA output, including transmission of the acknowledge bit.

Set the SDA output delay to meet the I2C bus standard for the data enable time/acknowledge enable time,*1 or the SMBus standard, within [data hold time (300 ns or more + the SCL clock low-level period) - the data setup time (250 ns)]. If a value outside the standard is set, communication between devices might malfunction or falsely indicate a start or stop condition, depending on the bus state.

For details on this function, see section 28.5. SDA Output Delay Function.

Note 1. Data enable time/acknowledge enable time.
3,450 ns for up to 100 kbps: Standard mode (Sm)
900 ns for up to 400 kbps: Fast mode (Fm)
450 ns for up to 1 Mbps: Fast-mode Plus (Fm+)

## 28.2.5 ICMR3 : I2C Bus Mode Register 3

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SMBS | WAIT | RDRFS | ACKWP | ACKBT | ACKBR | NF[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | NF[1:0] | Noise Filter Stage Select<br>0 0: Filter out noise of up to 1 IICφ cycle (single-stage filter)<br>0 1: Filter out noise of up to 2 IICφ cycles (2-stage filter)<br>1 0: Filter out noise of up to 3 IICφ cycles (3-stage filter)<br>1 1: Filter out noise of up to 4 IICφ cycles (4-stage filter) | R/W |
| 2 | ACKBR | Receive Acknowledge<br>0: 0 received as the acknowledge bit (ACK reception)<br>1: 1 received as the acknowledge bit (NACK reception) | R |
| 3 | ACKBT | Transmit Acknowledge<br>0: Send 0 as the acknowledge bit (ACK transmission)<br>1: Send 1 as the acknowledge bit (NACK transmission) | R/W*1 |
| 4 | ACKWP | ACKBT Write Protect<br>0: Write protect ACKBT bit<br>1: Write enable ACKBT bit | R/W |
| 5 | RDRFS | RDRF Flag Set Timing Select<br>Low-hold is released by writing to ACKBT.<br>0: Set the RDRF flag on the rising edge of the 9th SCL clock cycle. The SCLn line is not held low on the falling edge of the 8th clock cycle.<br>1: Set the RDRF flag on the rising edge of the 8th SCL clock cycle. The SCLn line is held low on the falling edge of the 8th clock cycle. | R/W*2 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6 | WAIT | Low-hold is released by reading ICDRR.<br>0: No wait (The SCLn line is not held low during the period between the 9th clock cycle and the 1st clock cycle.)<br>1: Wait (The SCLn line is held low during the period between the 9th clock cycle and the 1st clock cycle.) | R/W*2 |
| 7 | SMBS | SMBus/I2C Bus Select<br>0: Select I2C Bus<br>1: Select SMBus | R/W |

Note 1. Write to the ACKBT bit only while the ACKWP bit is already 1. If the application writes 1 to the ACKWP and ACKBT bits at the same time, the ACKBT bit is not set to 1.
Note 2. The WAIT and RDRFS bits are only valid in receive mode (invalid in transmit mode).

### NF[1:0] bits (Noise Filter Stage Select)

The NF[1:0] bits select the number of stages in the digital noise filter. For details on this function, see section 28.6. Digital Noise Filter Circuits

Note: Set the noise range to be filtered within a range less than the SCLn line high- or low-level period. If the noise range is set to a value of [SCL clock width: high- or low-level period, whichever is shorter] - [1.5 internal reference clock (IICφ) cycles + analog noise filter: 120 ns (reference values)] or more, the SCL clock is regarded as noise, which might prevent the IIC from operating normally.

### ACKBR bit (Receive Acknowledge)

The ACKBR bit stores the acknowledge bit information received from the receive device in transmit mode.

[Setting condition]

- When 1 is received as the acknowledge bit with the TRS bit in ICCR2 set to 1.

[Clearing conditions]

- When 0 is received as the acknowledge bit with the TRS bit in ICCR2 set to 1
- When 1 is written to the IICRST bit in ICCR1 while the ICE bit in ICCR1 is 0 (IIC reset).

### ACKBT bit (Transmit Acknowledge)

The ACKBT bit sets the acknowledge bit to be sent in receive mode

[Setting condition]

- When 1 is written to this bit with the ACKWP bit set to 1.

[Clearing conditions]

- When 0 is written to this bit with the ACKWP bit set to 1
- When stop condition request is detected with the SP bit in ICCR2 set to 1
- When 1 is written to the IICRST bit in ICCR1 while the ICE bit in ICCR1 is 0 (IIC reset).

### ACKWP bit (ACKBT Write Protect)

The ACKWP bit controls write enabling of the ACKBT bit.

### RDRFS bit (RDRF Flag Set Timing Select)

The RDRFS bit selects the RDRF flag set timing in receive mode and also selects whether to hold the SCLn line low on the falling edge of the 8th SCL clock cycle.

When the RDRFS bit is 0, the SCLn line is not held low on the falling edge of the 8th SCL clock cycle, and the RDRF flag is set to 1 on the rising edge of the 9th SCL clock cycle.

When the RDRFS bit is 1, the RDRF flag is set to 1 on the rising edge of the 8th SCL clock cycle, and the SCLn line is held low on the falling edge of the 8th SCL clock cycle. The low-hold of the SCLn line is released by a write to the ACKBT bit.

After data is received with this setting, the SCLn line is automatically held low before the acknowledge bit is sent. This enables processing to send ACK (ACKBT = 0) or NACK (ACKBT = 1), based on the receive data.

### WAIT bit (WAIT)

The WAIT bit controls whether to force a low-hold between the ninth SCL clock cycle and the first SCL clock cycle, until the receive data buffer (ICDRR) is completely read each time single-byte data is received in receive mode.

When the WAIT bit is 0, the receive operation continues without a low-hold between the ninth and the first SCL clock cycle. When both the RDRFS and WAIT bits are 0, continuous receive operation is enabled with the double buffer.

When the WAIT bit is 1, the SCLn line is held low from the falling edge of the ninth clock cycle until the ICDRR value is read each time single-byte data is received. This enables receive operation in byte units.

Note: When the value of the WAIT bit is to be read, always read ICDRR first.

### SMBS bit (SMBus/I2C Bus Select)

Setting the SMBS bit to 1 selects the SMBus and enables the HOAE bit in ICSER.

## 28.2.6 ICFER : I2C Bus Function Enable Register

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x05

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | FMPE | SCLE | NFE | NACKE | SALE | NALE | MALE | TMOE |
| Value after reset: | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TMOE | Timeout Function Enable<br>0: Disable<br>1: Enable | R/W |
| 1 | MALE | Master Arbitration-Lost Detection Enable<br>0: Disable the arbitration-lost detection function and disable automatic clearing of the MST and TRS bits in ICCR2 when arbitration is lost<br>1: Enable the arbitration-lost detection function and enable automatic clearing of the MST and TRS bits in ICCR2 when arbitration is lost | R/W |
| 2 | NALE | NACK Transmission Arbitration-Lost Detection Enable<br>0: Disable<br>1: Enable | R/W |
| 3 | SALE | Slave Arbitration-Lost Detection Enable<br>0: Disable<br>1: Enable | R/W |
| 4 | NACKE | NACK Reception Transfer Suspension Enable<br>0: Do not suspend transfer operation during NACK reception (disable transfer suspension)<br>1: Suspend transfer operation during NACK reception (enable transfer suspension) | R/W |
| 5 | NFE | Digital Noise Filter Circuit Enable<br>0: Do not use the digital noise filter circuit<br>1: Use the digital noise filter circuit | R/W |
| 6 | SCLE | SCL Synchronous Circuit Enable<br>0: Do not use the SCL synchronous circuit<br>1: Use the SCL synchronous circuit | R/W |
| 7 | FMPE*1 | Fast-Mode Plus Enable<br>0: Do not use the Fm+ slope control circuit for the SCLn and SDAn pins<br>1: Use the Fm+ slope control circuit for the SCLn and SDAn pins. | R/W |

Note 1. The Fast-mode Plus enable bit (FMPE) is supported by IIC0 (SCL0_A, SDA0_A). Bit [7] is the reserved bit in the not supported channel.

### TMOE bit (Timeout Function Enable)

The TMOE bit enables or disables the timeout function. For details on this function, see section 28.12.1. Timeout Function.

### MALE bit (Master Arbitration-Lost Detection Enable)

The MALE bit specifies whether to use the arbitration-lost detection function in master mode. Normally, set this bit to 1.

### NALE bit (NACK Transmission Arbitration-Lost Detection Enable)

The NALE bit specifies whether to cause arbitration to be lost when ACK is detected during transmission of NACK in receive mode, for example when slaves with the same address exist on the bus or when two or more masters select the same slave device simultaneously with a different number of receive bytes.

### SALE bit (Slave Arbitration-Lost Detection Enable)

The SALE bit specifies whether to cause arbitration to be lost when a value different from the value being transmitted is detected on the bus in slave transmit mode, for example when slaves with the same address exist on the bus or when a mismatch with the transmit data occurs because of noise.

### NACKE bit (NACK Reception Transfer Suspension Enable)

The NACKE bit specifies whether to continue or discontinue the transfer operation when NACK is received in transmit mode. Normally, set this bit to 1.

When NACK is received with the NACKE bit set to 1, the next transfer operation is suspended. When the NACKE bit is 0, the next transfer operation continues regardless of the received acknowledge content.

For details, see section 28.9.2. NACK Reception Transfer Suspension Function.

### SCLE bit (SCL Synchronous Circuit Enable)

The SCLE bit specifies whether to synchronize the SCL clock with the SCL input clock. Normally, set this bit to 1.

When the SCLE bit is set to 0 (no SCL synchronous circuit used), the IIC does not synchronize the SCL clock with the SCL input clock. With this setting, the IIC outputs the SCL clock at the transfer rate set in ICBRH and ICBRL, regardless of the SCLn line state. For this reason, if the bus load of the $I^2C$ bus line is much larger than the specification value, or if the SCL clock output overlaps in multiple masters, a short-cycle SCL clock that does not meet the specification might be output. When no SCL synchronous circuit is used, it also affects the issuing of the start, restart, and stop conditions, and the continuous output of extra SCL clock cycles.

Do not set this bit to 0 except when checking the output of the set transfer rate.

### FMPE bit (Fast-Mode Plus Enable)

The FMPE bit specifies whether to use a slope control circuit for Fast-mode Plus (Fm+).

When this bit is set to 1, a slope control circuit conforming to the $I^2C$ bus Fast-mode Plus (Fm+) standard (tof) is selected. When this bit is set to 0, a slope control circuit conforming to the $I^2C$ bus Standard-mode (Sm) and Fast-mode (Fm) standards (tof) is selected.

Set this bit to 1 when using transmission rates up to 1 Mbps (Fast-mode Plus (Fm+) standard). Set it to 0 when using other transmission rates (up to 100 kbps (Sm) or up to 400 kbps (Fm)) or for SMBus (10 to 100 kbps).

## 28.2.7 ICSER : $I^2C$ Bus Status Enable Register

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x06

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | HOAE | — | DIDE | — | GCAE | SAR2E | SAR1E | SAR0E |
| Value after reset: | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SAR0E | Slave Address Register 0 Enable<br>0: Disable slave address in SARL0 and SARU0<br>1: Enable slave address in SARL0 and SARU0 | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | SAR1E | Slave Address Register 1 Enable<br>0: Disable slave address in SARL1 and SARU1<br>1: Enable slave address in SARL1 and SARU1 | R/W |
| 2 | SAR2E | Slave Address Register 2 Enable<br>0: Disable slave address in SARL2 and SARU2<br>1: Enable slave address in SARL2 and SARU2 | R/W |
| 3 | GCAE | General Call Address Enable<br>0: Disable general call address detection<br>1: Enable general call address detection | R/W |
| 4 | — | This bit is read as 0. The write value should be 0. | R/W |
| 5 | DIDE | Device-ID Address Detection Enable<br>0: Disable device-ID address detection<br>1: Enable device-ID address detection | R/W |
| 6 | — | This bit is read as 0. The write value should be 0. | R/W |
| 7 | HOAE | Host Address Enable<br>0: Disable host address detection<br>1: Enable host address detection | R/W |

### SARyE bit (Slave Address Register y Enable) (y = 0 to 2)

The SARyE bit enables or disables the received slave address and the slave address set in SARLy and SARUy.

When this bit is set to 1, the slave address set in SARLy and SARUy is enabled and is compared with the received slave address. When this bit is set to 0, the slave address set in SARLy and SARUy is disabled and is ignored even if it matches the received slave address.

### GCAE bit (General Call Address Enable)

The GCAE bit specifies whether to ignore the general call address (0000 000b + 0 [W]: All 0) when it is received.

When this bit is set to 1, if the received slave address matches the general call address, the IIC recognizes the received slave address as the general call address independently of the slave addresses set in SARLy and SARUy (y = 0 to 2) and performs the data receive operation. When this bit is set to 0, the received slave address is ignored even if it matches the general call address.

### DIDE bit (Device-ID Address Detection Enable)

The DIDE bit specifies whether to recognize and execute the device-ID address when a device ID (1111 100b) is received in the first frame after a start or restart condition is detected.

When this bit is set to 1, if the received first frame matches the device ID, the IIC recognizes that the device-ID address was received. When the next R/W# bit is 0 (W), the IIC recognizes the second and the subsequent frames as slave addresses and continues the receive operation. When this bit is set to 0, the IIC ignores the received first frame even if it matches the device-ID address, and it recognizes the first frame as a normal slave address.

For details on this function, see section 28.7.3. Device-ID Address Detection.

### HOAE bit (Host Address Enable)

The HOAE bit specifies whether to ignore the received host address (0001 000b) when the SMBS bit in ICMR3 is 1.

When this bit is set to 1 while the SMBS bit in ICMR3 is 1, if the received slave address matches the host address, the IIC recognizes the received slave address as the host address independently of the slave addresses set in SARLy and SARUy (y = 0 to 2) and performs the receive operation.

When the SMBS bit in ICMR3 or the HOAE bit is set to 0, the received slave address is ignored even if it matches the host address.

## 28.2.8 ICIER : I²C Bus Interrupt Enable Register

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x07

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TIE | TEIE | RIE | NAKIE | SPIE | STIE | ALIE | TMOIE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TMOIE | Timeout Interrupt Request Enable<br>0: Disable timeout interrupt (TMOI) request<br>1: Enable timeout interrupt (TMOI) request | R/W |
| 1 | ALIE | Arbitration-Lost Interrupt Request Enable<br>0: Disable arbitration-lost interrupt (ALI) request<br>1: Enable arbitration-lost interrupt (ALI) request | R/W |
| 2 | STIE | Start Condition Detection Interrupt Request Enable<br>0: Disable start condition detection interrupt (STI) request<br>1: Enable start condition detection interrupt (STI) request | R/W |
| 3 | SPIE | Stop Condition Detection Interrupt Request Enable<br>0: Disable stop condition detection interrupt (SPI) request<br>1: Enable stop condition detection interrupt (SPI) request | R/W |
| 4 | NAKIE | NACK Reception Interrupt Request Enable<br>0: Disable NACK reception interrupt (NAKI) request<br>1: Enable NACK reception interrupt (NAKI) request | R/W |
| 5 | RIE | Receive Data Full Interrupt Request Enable<br>0: Disable receive data full interrupt (IICn_RXI) request<br>1: Enable receive data full interrupt (IICn_RXI) request | R/W |
| 6 | TEIE | Transmit End Interrupt Request Enable<br>0: Disable transmit end interrupt (IICn_TEI) request<br>1: Enable transmit end interrupt (IICn_TEI) request | R/W |
| 7 | TIE | Transmit Data Empty Interrupt Request Enable<br>0: Disable transmit data empty interrupt (IICn_TXI) request<br>1: Enable transmit data empty interrupt (IICn_TXI) request | R/W |

### TMOIE bit (Timeout Interrupt Request Enable)

The TMOIE bit enables or disables timeout interrupt (TMOI) requests when the TMOF flag in ICSR2 is 1. To cancel a TMOI interrupt request, set the TMOF flag or the TMOIE bit to 0.

### ALIE bit (Arbitration-Lost Interrupt Request Enable)

The ALIE bit enables or disables arbitration-lost interrupt (ALI) requests when the AL flag in ICSR2 is 1. To cancel an ALI interrupt request, set the AL flag or the ALIE bit to 0.

### STIE bit (Start Condition Detection Interrupt Request Enable)

The STIE bit enables or disables start condition detection interrupt (STI) requests when the START flag in ICSR2 is 1. To cancel an STI interrupt request, set the START flag or the STIE bit to 0.

### SPIE bit (Stop Condition Detection Interrupt Request Enable)

The SPIE bit enables or disables stop condition detection interrupt (SPI) requests when the STOP flag in ICSR2 is 1. To cancel an SPI interrupt request, set the STOP flag or the SPIE bit to 0.

### NAKIE bit (NACK Reception Interrupt Request Enable)

The NAKIE bit enables or disables NACK reception interrupt (NAKI) requests when the NACKF flag in ICSR2 is 1. To cancel an NAKI interrupt request, set the NACKF flag or the NAKIE bit to 0.

### RIE bit (Receive Data Full Interrupt Request Enable)

The RIE bit enables or disables receive data full interrupt (IICn_RXI) requests when the RDRF flag in ICSR2 is 1.

### TEIE bit (Transmit End Interrupt Request Enable)

The TEIE bit enables or disables transmit end interrupt (IICn_TEI) requests when the TEND flag in ICSR2 is 1. To cancel an IICn_TEI interrupt request, set the TEND flag or the TEIE bit to 0.

### TIE bit (Transmit Data Empty Interrupt Request Enable)

The TIE bit enables or disables transmit data empty interrupt (IICn_TXI) requests when the TDRE flag in ICSR2 is 1.

## 28.2.9 ICSR1 : I2C Bus Status Register 1

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x08

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | HOA | — | DID | — | GCA | AAS2 | AAS1 | AAS0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | AAS0 | Slave Address 0 Detection Flag<br>0: Slave address 0 not detected<br>1: Slave address 0 detected | R/(W)*1 |
| 1 | AAS1 | Slave Address 1 Detection Flag<br>0: Slave address 1 not detected<br>1: Slave address 1 detected | R/(W)*1 |
| 2 | AAS2 | Slave Address 2 Detection Flag<br>0: Slave address 2 not detected<br>1: Slave address 2 detected | R/(W)*1 |
| 3 | GCA | General Call Address Detection Flag<br>0: General call address not detected<br>1: General call address detected | R/(W)*1 |
| 4 | — | This bit is read as 0. The write value should be 0. | R/W |
| 5 | DID | Device-ID Address Detection Flag<br>This bit is set to 1 when the first frame received immediately after a start condition is detected matches a value of (device ID (1111 100b) + 0[W]).<br>0: Device-ID command not detected<br>1: Device-ID command detected | R/(W)*1 |
| 6 | — | This bit is read as 0. The write value should be 0. | R/W |
| 7 | HOA | Host Address Detection Flag<br>This bit is set to 1 when the received slave address matches the host address (0001 000b).<br>0: Host address not detected<br>1: Host address detected | R/(W)*1 |

Note 1. Only 0 can be written to clear the flag.

### AASy flag (Slave Address y Detection flag) (y = 0 to 2)

The AASy flag indicates whether slave address y was detected.

[Setting conditions]

For 7-bit address format (SARUy.FS = 0):

- When the received slave address matches the SVA[6:0] value in SARLy, with the SAReE bit in ICSER set to 1 (slave address y detection enabled).
  The AASy flag is set to 1 on the rising edge of the ninth SCL clock cycle in the frame.

For 10-bit address format: (SARUy.FS = 1):

- When the received slave address matches a value of (11110b + SVA[1:0] in SARUy), and the subsequent address matches the SARLy value, with the SARyE bit in ICSER set to 1 (slave address y detection enabled).
  The AASy flag is set to 1 on the rising edge of the ninth SCL clock cycle in the frame.

[Clearing conditions]

- When 0 is written to the AASy flag after reading AASy = 1
- When a stop condition is detected
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

For 7-bit address format (SARUy.FS = 0):

- When the received slave address does not match the SVA[6:0] value in SARLy, with the SARyE bit in ICSER set to 1 (slave address y detection enabled).
  The AASy flag is set to 0 on the rising edge of the ninth SCL clock cycle in the frame.

For 10-bit address format (SARUy.FS = 1):

- When the received slave address does not match a value of (11110b + SVA[1:0] in SARUy), with the SARyE bit in ICSER set to 1 (slave address y detection enabled)
  The AASy flag is set to 0 on the rising edge of the ninth SCL clock cycle in the frame.
- When the received slave address matches a value of (11110b + SVA[1:0] in SARUy), and the subsequent address does not match the SARLy value, with the SARyE bit in ICSER set to 1 (slave address y detection enabled).
  The AASy flag is set to 0 on the rising edge of the ninth SCL clock cycle in the frame.

#### GCA flag (General Call Address Detection Flag)

The GCA flag indicates whether the general call address was detected.

[Setting condition]

- When the received slave address matches the general call address (0000 000b + 0 [W]), with the GCAE bit in ICSER set to 1 (general call address detection enabled).
  The GCA flag is set to 1 on the rising edge of the ninth SCL clock cycle in the frame.

[Clearing conditions]

- When 0 is written to the GCA flag after reading GCA = 1
- When a stop condition is detected
- When the received slave address does not match the general call address (0000 000b + 0 [W]), with the GCAE bit in ICSER set to 1 (general call address detection enabled)
  The GCA flag is set to 0 on the rising edge of the ninth SCL clock cycle in the frame.
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

#### DID flag (Device-ID Address Detection Flag)

The DID flag indicates whether the device-ID address was detected.

[Setting condition]

- When the first frame received immediately after a start or restart condition is detected matches a value of (device ID (1111 100b) + 0 [W]), with the DIDE bit in ICSER set to 1 (device-ID address detection enabled).
  The DID flag is set to 1 on the rising edge of the ninth SCL clock cycle in the frame.

[Clearing conditions]

- When 0 is written to the DID flag after reading DID = 1
- When a stop condition is detected
- When the first frame received immediately after a start or restart condition is detected does not match a value of (device ID (1111 100b)), with the DIDE bit in ICSER set to 1 (device-ID address detection enabled)
  The DID flag is set to 0 on the rising edge of the ninth SCL clock cycle in the frame.

- When the first frame received immediately after a start or restart condition is detected matches a value of (device ID (1111 100b) + 0 [W]), and the second frame does not match any slave address from 0 to 2, with the DIDE bit in ICSER set to 1 (device-ID address detection enabled)
  The DID flag is set to 0 on the rising edge of the ninth SCL clock cycle in the frame.
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

#### HOA flag (Host Address Detection Flag)

The HOA flag indicates whether the host address was detected.

[Setting condition]

- When the received slave address matches the host address (0001 000b), with the HOAE bit in ICSER set to 1 (host address detection enabled).
  The HOA flag is set to 1 on the rising edge of the ninth SCL clock cycle in the frame.

[Clearing conditions]

- When 0 is written to the HOA flag after reading HOA = 1
- When a stop condition is detected
- When the received slave address does not match the host address (0001 000b), with the HOAE bit in ICSER set to 1 (host address detection enabled)
  The HOA flag is set to 0 on the rising edge of the ninth SCL clock cycle in the frame.
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

### 28.2.10 ICSR2 : I²C Bus Status Register 2

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x09

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TDRE | TEND | RDRF | NACKF | STOP | START | AL | TMOF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TMOF | Timeout Detection Flag<br>0: Timeout not detected<br>1: Timeout detected | R/(W)*1 |
| 1 | AL | Arbitration-Lost Flag<br>0: Arbitration not lost<br>1: Arbitration lost | R/(W)*1 |
| 2 | START | Start Condition Detection Flag<br>0: Start condition not detected<br>1: Start condition detected | R/(W)*1 |
| 3 | STOP | Stop Condition Detection Flag<br>0: Stop condition not detected<br>1: Stop condition detected | R/(W)*1 |
| 4 | NACKF | NACK Detection Flag<br>0: NACK not detected<br>1: NACK detected | R/(W)*1 |
| 5 | RDRF | Receive Data Full Flag<br>0: ICDRR contains no receive data<br>1: ICDRR contains receive data | R/(W)*1 |
| 6 | TEND | Transmit End Flag<br>0: Data being transmitted<br>1: Data transmit complete | R/(W)*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | TDRE | Transmit Data Empty Flag<br>0: ICDRT contains transmit data<br>1: ICDRT contains no transmit data | R |

Note 1. Only 0 can be written, to clear the flag.

### TMOF flag (Timeout Detection Flag)

The TMOF flag is set to 1 when the IIC detects a timeout because the SCLn line state remains unchanged for the set period.

[Setting condition]

- When the SCLn line state remains unchanged for the period specified in the ICMR2.TMOH, TMOL, and TMOS bits while the ICFER.TMOE bit is 1 (timeout function enabled) in master or in slave mode and the received slave address matches.

[Clearing conditions]

- When 0 is written to the TMOF flag after reading TMOF = 1
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

### AL flag (Arbitration-Lost Flag)

The AL flag indicates that bus mastership was lost in arbitration because of a bus conflict or some other reason when a start condition was issued or an address and data was transmitted. The IIC monitors the level on the SDAn line during transmission and, if the level on the line does not match the value of the bit being output, is set the value of the AL flag to 1 to indicate that the bus is occupied by another device.

The IIC can also set the AL flag to indicate the detection of arbitration loss during NACK transmission or during data transmission.

[Setting conditions]

When master arbitration-lost detection is enabled (ICFER.MALE = 1):

- When the internal SDA output state does not match the SDAn line level on the rising edge of the SCL clock except for the ACK period during data transmission in master transmit mode
- When a start condition is detected while the ST bit in ICCR2 is 1 (start condition requested) or the internal SDA output state does not match the SDAn line level
- When the ST bit in ICCR2 is 1 (start condition requested), with the BBSY flag in ICCR2 set to 1.

When NACK arbitration-lost detection is enabled (ICFER.NALE = 1):

- When the internal SDA output state does not match the SDAn line level on the rising edge of the SCL clock in the ACK period during NACK transmission in receive mode.

When slave arbitration-lost detection is enabled (ICFER.SALE = 1):

- When the internal SDA output state does not match the SDAn line level on the rising edge of the SCL clock, except for the ACK period during data transmission in slave transmit mode.

[Clearing conditions]

- When 0 is written to the AL flag after reading AL = 1
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

**Table 28.4 Relationship between arbitration-lost generation sources and arbitration-lost enable functions**

| ICFER | | | ICSR2 | | |
|---|---|---|---|---|---|
| MALE | NALE | SALE | AL | Error | Arbitration-lost generation source |
| 1 | x | x | 1 | Start condition issuance error | When internal SDA output state does not match SDAn line level when a start condition is detected, while the ST bit in ICCR2 is 1 |
| | | | | | When ST in ICCR2 is set to 1 while BBSY in ICCR2 is 1 |
| | | | 1 | Transmit data mismatch | When transmit data (including slave address) does not match the bus state in master transmit mode |
| x | 1 | x | 1 | NACK transmission mismatch | When ACK is detected during transmission of NACK in master or slave receive mode |
| x | x | 1 | 1 | Transmit data mismatch | When transmit data does not match the bus state in slave transmit mode |

x: Don’t care

### START flag (Start Condition Detection Flag)

The START flag indicates whether a start or restart condition was detected.

[Setting condition]

- When a start or restart condition is detected.

[Clearing conditions]

- When 0 is written to the START flag after reading START = 1
- When a stop condition is detected
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

### STOP flag (Stop Condition Detection Flag)

The STOP flag indicates whether a stop condition was detected.

[Setting condition]

- When a stop condition is detected.

[Clearing conditions]

- When 0 is written to the STOP flag after reading STOP = 1
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

### NACKF flag (NACK Detection Flag)

The NACKF flag indicates whether a NACK was detected.

[Setting condition]

- When acknowledge is not received (NACK received) from the receive device in transmit mode, with the NACKE bit in ICFER set to 1 (transfer suspension enabled).

[Clearing conditions]

- When 0 is written to the NACKF flag after reading NACKF = 1
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

Note: When the NACKF flag is set to 1, the IIC suspends data transmission and reception. Writing to ICDRT in transmit mode or reading from ICDRR in receive mode with the NACKF flag set to 1 does not enable data transmit or receive operation. To restart data transmission or reception, set the NACKF flag to 0.

### RDRF flag (Receive Data Full Flag)

The RDRF flag indicates whether the ICDRR contains receive data.

[Setting conditions]

- When receive data is transferred from ICDRS to ICDRR
  The RDRF flag is set to 1 on the rising edge of the eighth or ninth SCL clock cycle (selected in the RDRFS bit in ICMR3).
- When the received slave address matches after a start or restart condition is detected with the TRS bit in ICCR2 set to 0.

[Clearing conditions]

- When 0 is written to the RDRF flag after reading RDRF = 1
- When data is read from ICDRR
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

#### TEND flag (Transmit End Flag)

The TEND flag indicates completion of transmission.

[Setting condition]

- On the rising edge of the ninth SCL clock cycle while the TDRE flag is 1.

[Clearing conditions]

- When 0 is written to the TEND flag after reading TEND = 1
- When data is written to ICDRT
- When a stop condition is detected
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

#### TDRE flag (Transmit Data Empty Flag)

The TDRE flag indicates whether the ICDRT contains transmit data.

[Setting conditions]

- When data is transferred from ICDRT to ICDRS and ICDRT becomes empty
- When the TRS bit in ICCR2 is set to 1
- When the received slave address matches while the TRS bit is 1.

[Clearing conditions]

- When data is written to ICDRT
- When the TRS bit in ICCR2 is set to 0
- When 1 is written to the IICRST bit in ICCR1 to apply an IIC reset or an internal reset.

Note: When the NACKF flag is set to 1 while the NACKE bit in ICFER is 1, the IIC suspends data transmission and reception. In this case, if the TDRE flag is 0 (next transmit data written), data is transferred to the ICDRS register and the ICDRT register becomes empty on the rising edge of the 9th clock cycle, but the TDRE flag does not set to 1.

### 28.2.11 ICWUR : I2C Bus Wakeup Unit Register

Base address: IIC0WU = 0x4009_F014

Offset address: 0x02

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | WUE | WUIE | WUF | WUACK | — | — | — | WUAFA |
| Value after reset: | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | WUAFA | Wakeup Analog Filter Additional Selection<br>0: Do not add the wakeup analog filter<br>1: Add the wakeup analog filter | R/W |
| 3:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | WUACK | ACK Bit for Wakeup Mode<br>Choice of four response modes in combination with ICCR1.IICRST and WUACK. See Table 28.5. | R/W |
| 5 | WUF | Wakeup Event Occurrence Flag<br>0: Slave address not matching during wakeup<br>1: Slave address matching during wakeup | R/W |
| 6 | WUIE | Wakeup Interrupt Request Enable<br>0: Disable wakeup interrupt request (IIC0_WUI)<br>1: Enable wakeup interrupt request (IIC0_WUI) | R/W |
| 7 | WUE | Wakeup Function Enable<br>0: Disable wakeup function<br>1: Enable wakeup function | R/W |

**Table 28.5 Wakeup mode**

| IICRST | WUACK | Operation mode | Description |
|---|---|---|---|
| 0 | 0 | Normal wakeup mode 1 | ACK response on 9th SCL, and SCL low-hold after 9th SCL. |
| 0 | 1 | Normal wakeup mode 2 | No ACK response immediately and SCL low-hold between 8th and 9th SCL. SCL low-hold release and ACK response on 9th SCL. |
| 1 | 0 | Command recovery mode | ACK response on 9th SCL and no SCL low-hold. |
| 1 | 1 | EEP response mode | NACK response on 9th SCL and no SCL low-hold. |

### WUF flag (Wakeup Event Occurrence Flag)

The WUF flag indicates whether the slave address is matching during wakeup.

[Setting condition]

- When PCLKB is supplied after a slave-address match in the first 8th SCL low during wakeup mode.

[Clearing conditions]

- When 0 is written to the WUF flag after reading WUF = 1
- When ICE = 0 and IICRST = 1.

## 28.2.12 ICWUR2 : I2C Bus Wakeup Unit Register 2

Base address: IIC0WU = 0x4009_F014

Offset address: 0x03

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | WUSYF | WUASYF | WUSEN |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | WUSEN | Wakeup Function Synchronous Enable<br>0: IIC asynchronous circuit enable<br>1: IIC synchronous circuit enable | R/W |
| 1 | WUASYF | Wakeup Function Asynchronous Operation Status Flag<br>0: IIC synchronous circuit enable condition<br>1: IIC asynchronous circuit enable condition | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2 | WUSYF | Wakeup Function Synchronous Operation Status Flag<br>0: IIC asynchronous circuit enable condition<br>1: IIC synchronous circuit enable condition | R |
| 7:3 | — | These bits are read as 1. The write value should be 1. | R/W |

### WUSEN bit (Wakeup Function Synchronous Enable)

The WUSEN bit is used in combination with the WUASYF flag (or WUSYF flag) to switch between the PCLKB synchronous and asynchronous operation, when the wakeup effective function is enabled (ICWUR.WUE = 1).

The PCLKB operation switches from synchronous to asynchronous operation:

When the ICCR2.BBSY flag is 0, if 0 is written to the WUSEN bit while the WUASYF flag is 0, the reception occurs independently of the operation of PCLKB (with PCLKB stopped) after it switches to the PCLKB asynchronous operation, on wakeup event detection.

The PCLKB operation switches from asynchronous to synchronous operation:

- When 1 is written to the WUSEN bit, with the WUASYF flag at 1, when a wakeup event is detected. After writing 1, the WUASYF flag immediately becomes 0.
- When the stop condition is detected with a wakeup event undetected.

### WUASYF flag (Wakeup Function Asynchronous Operation Status Flag)

The WUASYF flag can place the IIC in PCLKB asynchronous operation when the wakeup effective function is enabled (ICWUR.WUE = 1).

[Setting condition]

- When the ICCR2.BBSY flag is 0, and the WUSEN bit is set to 0 with the ICWUR.WUE bit set to 1.

[Clearing conditions]

- When 1 is written to the WUSEN bit after detecting the wake-up event with ICWUR.WUE bit set to 1.
- When a stop condition is detected with WUSEN bit set to 1 before detecting the wake-up event with WUASY flag set to 1 with ICWUR.WUE bit set to 1.
- When you write 1 in the WUSEN bit with the WUASYF flag detected 1 and the wake-up event in the state of ICWUR.WUE = 1.
- ICCR1.ICE = 0 and ICCRST = 1 (ICC reset)
- ICWUR.WUE = 0.

### WUSYF flag (Wakeup Function Synchronous Operation Status Flag)

It is shown that IIC is in the PCLKB synchronous operation at wake-up effective function (ICWUR.WUE = 1). This flag is a value in which the WUASYF flag is always reserved.

[Setting conditions]

- When 1 is written to the WUSEN bit after detecting the wake-up event with ICWUR.WUE bit set to 1 with WUSYF flag set to 0 with ICWUR.WUE bit set to 1.
- When a stop condition is detected with WUSEN bit set to 1 before detecting the wake-up event with WUSYF flag set to 0 with ICWUR.WUE bit set to 1.
- ICCR1.ICE = 0 and ICCRST = 1 (ICC reset)
- ICWUR.WUE = 0.

[Clearing condition]

- When the ICCR2.BBSY flag is 0 with the ICWUR.WUE bit set to 1 after writing 0 to the WUSEN bit.

## 28.2.13 SARLy : Slave Address Register Ly (y = 0 to 2)

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x0A + 0x02 × y

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SVA[6:0] | | | | | | | SVA0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SVA0 | 10-bit Address LSB<br>Slave address setting. | R/W |
| 7:1 | SVA[6:0] | 7-bit Address/10-bit Address Lower Bits<br>Slave address setting. | R/W |

### SVA0 bit (10-bit Address LSB)

When the 10-bit address format is selected (SARUy.FS = 1), the SVA0 bit functions as the LSB of a 10-bit address and is combined with the SVA[6:0] bits to form the lower 8 bits of a 10-bit address.

This bit is valid when the SARyE bit in ICSER is set to 1 (SARLy and SARUy enabled) and the SARUy.FS bit is 1. When the SARUy.FS or SARyE bit is 0, the setting in this bit is ignored.

### SVA[6:0] bits (7-bit Address/10-bit Address Lower Bits)

When the 7-bit address format is selected (SARUy.FS = 0), the SVA[6:0] bits function as a 7-bit address. When the 10- bit address format is selected (SARUy.FS = 1), these bits combine with the SVA0 bit to form the lower 8 bits of a 10-bit address.

When the SARyE bit in ICSER is 0, the setting in these bits is ignored.

## 28.2.14 SARUy : Slave Address Register Uy (y = 0 to 2)

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x0B + 0x02 × y

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | SVA[1:0] | | FS |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FS | 7-bit/10-bit Address Format Select<br>0: Select 7-bit address format<br>1: Select 10-bit address format | R/W |
| 2:1 | SVA[1:0] | 10-bit Address Upper Bits<br>Slave address setting. | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

### FS bit (7-bit/10-bit Address Format Select)

The FS bit selects 7- or 10-bit format for slave address y (in SARLy and SARUy).

When the SARyE bit in ICSER is set to 1 (SARLy and SARUy enabled) and the SARUy.FS bit is 0, the 7-bit address format is selected for slave address y, the SVA[6:0] setting in SARLy is valid, and the SVA[1:0] and SVA0 settings in SARLy are ignored.

When the SARyE bit in ICSER is set to 1 (SARLy and SARUy enabled) and the SARUy.FS bit is 1, the 10-bit address format is selected for slave address y and the SVA[1:0] and SARLy settings are valid.

When the SARyE bit in ICSER is 0 (SARLy and SARUy disabled), the SARUy.FS setting is invalid.

### SVA[1:0] bits (10-bit Address Upper Bits)

When the 10-bit address format is selected (FS = 1), the SVA[1:0] bits function as the upper 2 bits of a 10-bit address.

These bits are valid when the SARyE bit in ICSER is set to 1 (SARLy and SARUy enabled) and the SARUy.FS bit is 1. When the SARUy.FS or SARyE bit is 0, the setting in these bits is ignored.

## 28.2.15 ICBRL : I2C Bus Bit Rate Low-Level Register

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x10

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | BRL[4:0] | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | BRL[4:0] | Bit Rate Low-Level Period<br>Low-level period of SCL clock. | R/W |
| 7:5 | — | These bits are read as 1. The write value should be 1. | R/W |

### BRL[4:0] bits (Bit Rate Low-Level Period)

The BRL[4:0] bits set the low-level period of the SCL clock. ICBRL counts the low-level period with the internal reference clock source (IICφ) specified by the CKS[2:0] bits in ICMR1. ICBRL also generates the data setup time for automatic SCL low-hold operation, see section 28.9. Automatic Low-Hold Function for SCL. When the IIC is used in slave mode, the BRL[4:0] bits must be set to a value longer than the data setup time*1.

If the digital noise filter is enabled (NFE bit in ICFER is 1), set the BRL[4:0] bits to a value at least one greater than the number of stages in the noise filter. For details on the number of stages, see the description of the NF[1:0] bits in section 28.2.5. ICMR3 : I2C Bus Mode Register 3.

Note 1. Data setup time ($t_{SU}$: DAT)

250 ns for up to 100 kbps: Standard-mode (Sm)

100 ns for up to 400 kbps: Fast-mode (Fm)

50 ns for up to 1 Mbps: Fast-mode Plus (Fm+)

## 28.2.16 ICBRH : I2C Bus Bit Rate High-Level Register

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x11

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | BRH[4:0] | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | BRH[4:0] | Bit Rate High-Level Period<br>High-level period of SCL clock. | R/W |
| 7:5 | — | These bits are read as 1. The write value should be 1. | R/W |

### BRH[4:0] bits (Bit Rate High-Level Period)

The BRH[4:0] bits set the high-level period of SCL clock. BRH[4:0] bits are valid in master mode. If the IIC is used only in slave mode, do not set the BRH[4:0] bits.

ICBRH counts the high-level period with the internal reference clock source (IICφ) specified in the CKS[2:0] bits in ICMR1.

If the digital noise filter is enabled (the NFE bit in ICFER is 1), set these bits to a value at least one greater than the number of stages in the noise filter. For the number of stages in the noise filter, see the description of the NF[1:0] bits in section 28.2.5. ICMR3 : I2C Bus Mode Register 3.

The IIC transfer rate and the SCL clock duty are calculated using the following expressions (1) to (5):

1. ICFER.SCLE = 0
   Transfer rate = 1/[{(BRH + 1) + (BRL + 1)}/IICφ*1 + tr*2 + tf*2]
   Duty cycle = [tr + {(BRH + 1)/IICφ}]/[tr + tf + {(BRH + 1) + (BRL + 1)}/IICφ]
2. ICFER.SCLE = 1 and ICFER.NFE = 0 and CKS[2:0] = 000b (IICφ = PCLKB)
   Transfer rate = 1/[{(BRH + 3) + (BRL+ 3)}/IICφ + tr + tf]
   Duty cycle = [tr + {(BRH + 3)/IICφ}]/[tr + tf + {(BRH + 3) + (BRL + 3)}/IICφ]
3. ICFER.SCLE = 1 and ICFER.NFE = 1 and CKS[2:0] = 000b (IICφ = PCLKB)
   Transfer rate = 1/[{(BRH + 3 + nf*3) + (BRL + 3 + nf)}/IICφ + tr + tf]
   Duty cycle = [tr + {(BRH + 3 + nf)/IICφ}]/[tr + tf + {(BRH + 3 + nf) + (BRL + 3 + nf)}/IICφ]
4. ICFER.SCLE = 1 and ICFER.NFE = 0 and CKS[2:0] ≠ 000b
   Transfer rate = 1/[{(BRH + 2) + (BRL + 2)}/IICφ + tr + tf]
   Duty cycle = [tr + {(BRH + 2)/IICφ}]/[tr + tf + {(BRH + 2) + (BRL + 2)}/IICφ]
5. ICFER.SCLE = 1 and ICFER.NFE = 1 and CKS[2:0] ≠ 000b
   Transfer rate = 1/[{(BRH + 2 + nf) + (BRL + 2 + nf)}/IICφ + tr + tf]
   Duty cycle = [tr + {(BRH + 2 + nf)/IICφ}]/[tr + tf + {(BRH + 2 + nf) + (BRL + 2 + nf)}/IICφ]

Note 1. IICφ = PCLKB × division ratio
Note 2. The SCLn line rise time (tr) and SCLn line fall time (tf) depend on the total bus line capacitance (Cb) and the pull-up resistor (Rp). For details, see the I2C bus standard from NXP Semiconductors.
Note 3. nf = Number of digital noise filters selected in the ICMR3.NF bit.

**Table 28.6 Example of ICBRH/ICBRL Settings for Transfer Rate IIC when SCLE = 0**

| Transfer rate (kbps) | CKS[2:0] (ICMR1) | BRH[4:0] (ICBRH) | BRL[4:0] (ICBRL) | PCLKB (MHz) | NF[1:0] | Computation expression |
|---|---|---|---|---|---|---|
| 100 | 011b | 24 (0xF8) | 30 (0xFE) | 50 | — | (1) |
| 400 | 010b | 7 (0xE7) | 15 (0xEF) | 50 | — | (1) |
| 1000 | 000b | 12 (0xEC) | 24 (0xF8) | 50 | — | (1) |

**Table 28.7 Example of ICBRH/ICBRL Settings for Transfer Rate when SCLE = 1 and NFE = 0**

| Transfer rate (kbps) | CKS[2:0] (ICMR1) | BRH[4:0] (ICBRH) | BRL[4:0] (ICBRL) | PCLKB (MHz) | NF[1:0] | Computation expression |
|---|---|---|---|---|---|---|
| 100 | 100b | 11 (0xEB) | 13 (0xED) | 50 | — | (4) |
| 400 | 001b | 13 (0xED) | 31 (0xFF) | 50 | — | (4) |
| 1000 | 000b | 10 (0xEA) | 22 (0xF6) | 50 | — | (2) |

**Table 28.8 Example of ICBRH/ICBRL Settings for Transfer Rate when SCLE = 1 and NFE = 1**

| Transfer rate (kbps) | CKS[2:0] (ICMR1) | BRH[4:0] (ICBRH) | BRL[4:0] (ICBRL) | PCLKB (MHz) | NF[1:0] | Computation expression |
|---|---|---|---|---|---|---|
| 100 | 011b | 21 (0xF5) | 26 (0xFA) | 50 | 01b | (5) |
| 400 | 001b | 11 (0xEB) | 29 (0xFD) | 50 | 01b | (5) |
| 1000 | 000b | 8 (0xE8) | 20 (0xF4) | 50 | 01b | (3) |

### 28.2.17 ICDRT : I2C Bus Transmit Data Register

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x12

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

When ICDRT detects a space in the I2C Bus Shift Register (ICDRS), it transfers the transmit data that was written to ICDRT to ICDRS and starts transmitting data in transmit mode. The double-buffer structure of ICDRT and ICDRS allows continuous transmit operation if the next transmit data is written to ICDRT while the ICDRS data is being transmitted.

ICDRT can always be read and written to. Write transmit data to ICDRT once when a transmit data empty interrupt (IICn_TXI) request is generated.

### 28.2.18 ICDRR : I2C Bus Receive Data Register

Base address: IICn = 0x4009_F000 + 0x0100 × n (n = 0, 1)

Offset address: 0x13

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

When 1 byte of data is received, the received data is transferred from the I2C Bus Shift Register (ICDRS) to ICDRR to enable the next data to be received. The double-buffer structure of ICDRS and ICDRR allows continuous receive operation if the received data is read from ICDRR while ICDRS is receiving data. ICDRR cannot be written to. Read data from ICDRR once when a receive data full interrupt (IICn_RXI) request is generated.

If ICDRR receives the next receive data before the current data is read from ICDRR (while the RDRF flag in ICSR2 is 1), the IIC automatically holds the SCL clock low 1 cycle before the RDRF flag is set to 1 next.

### 28.2.19 ICDRS : I2C Bus Shift Register

Base address: n/a

Offset address: n/a

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | — | — | — | — | — | — | — | — |

ICDRS is an 8-bit shift register for data transmit and receive. During transmission, transmit data is transferred from ICDRT to ICDRS and is transmitted from the SDAn pin. During reception, data is transferred from ICDRS to ICDRR after 1 byte of data is received. ICDRS cannot be accessed directly.

## 28.3 Operation

### 28.3.1 Communication Data Format

The I2C bus format consists of 8-bit data and 1-bit acknowledge. The frame following a start or restart condition is an address frame that specifies a slave device with which the master device communicates. The specified slave is valid until a new slave is specified or a stop condition is issued.

Figure 28.3 shows the I2C bus format, and Figure 28.4 shows the I2C bus timing.

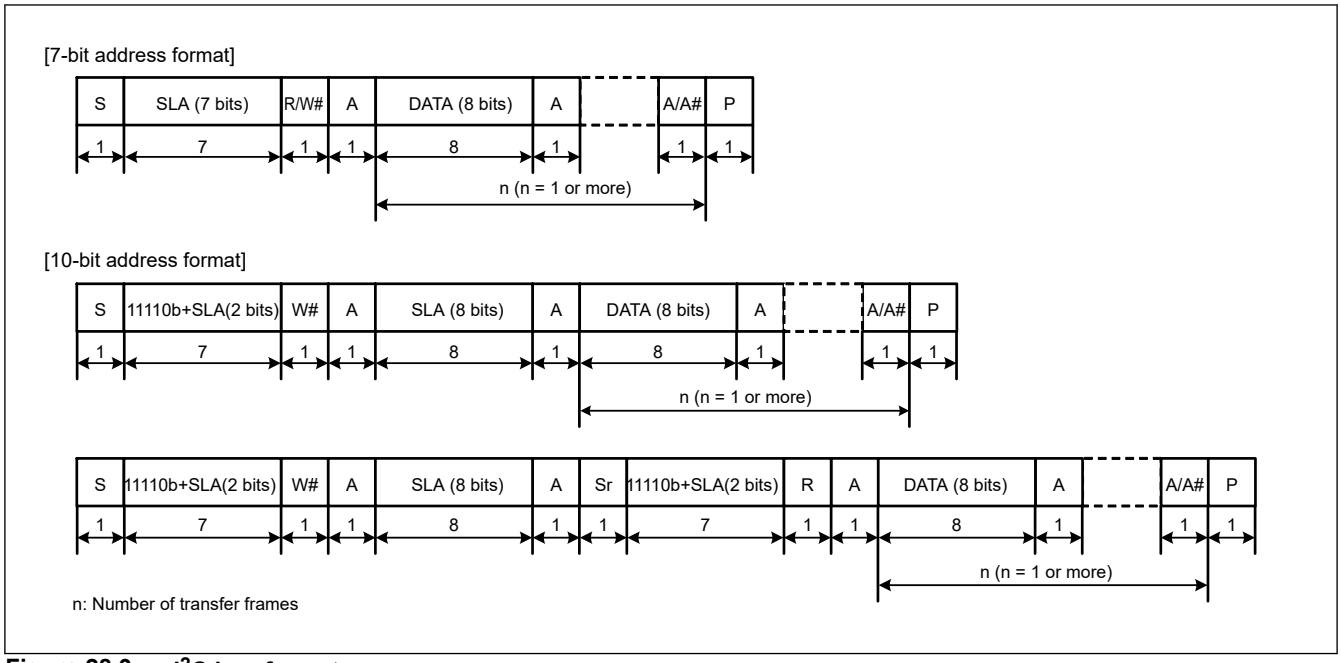



**Figure 28.3 I²C bus format**

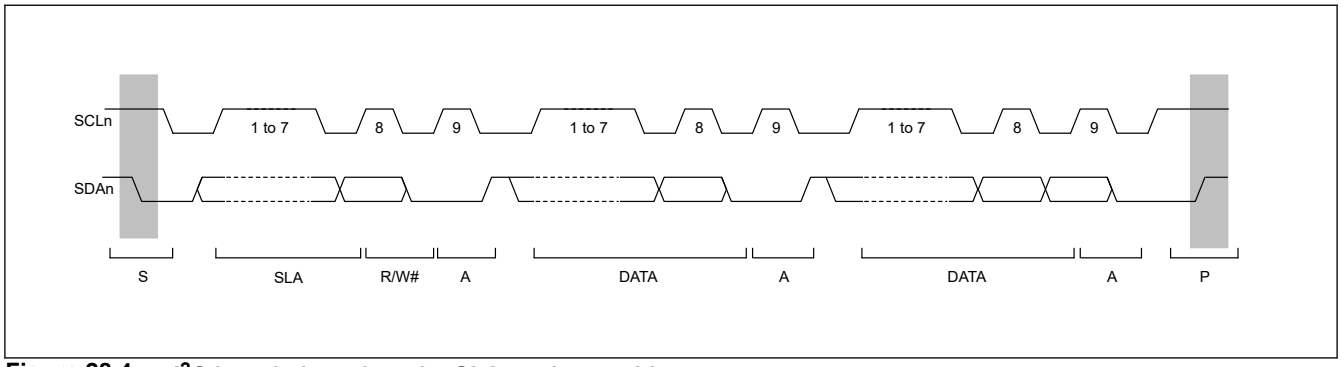



**Figure 28.4 I²C bus timing when the SLA setting = 7 bits**

| | |
|---|---|
| S: | Start condition. The master device drives the SDAn line low from high while the SCLn line is high. |
| SLA: | Slave address, by which the master device selects a slave device. |
| R/W#: | Indicates the direction of data transfer: from the slave device to the master device when R/W# is 1, or from the master device to the slave device when R/W# is 0. |
| A: | Acknowledge. The receive device drives the SDAn line low. In master transmit mode, the slave device returns acknowledge. In master receive mode, the master device returns acknowledge. |
| A#: | Not Acknowledge. The receive device drives the SDAn line high. |
| Sr: | Restart condition. The master device drives the SDAn line low from the high level after the setup time has elapsed with the SCLn line high. |
| DATA: | Transmitted or received data. |
| P: | Stop condition. The master device drives the SDAn line high from low while the SCLn line is high. |

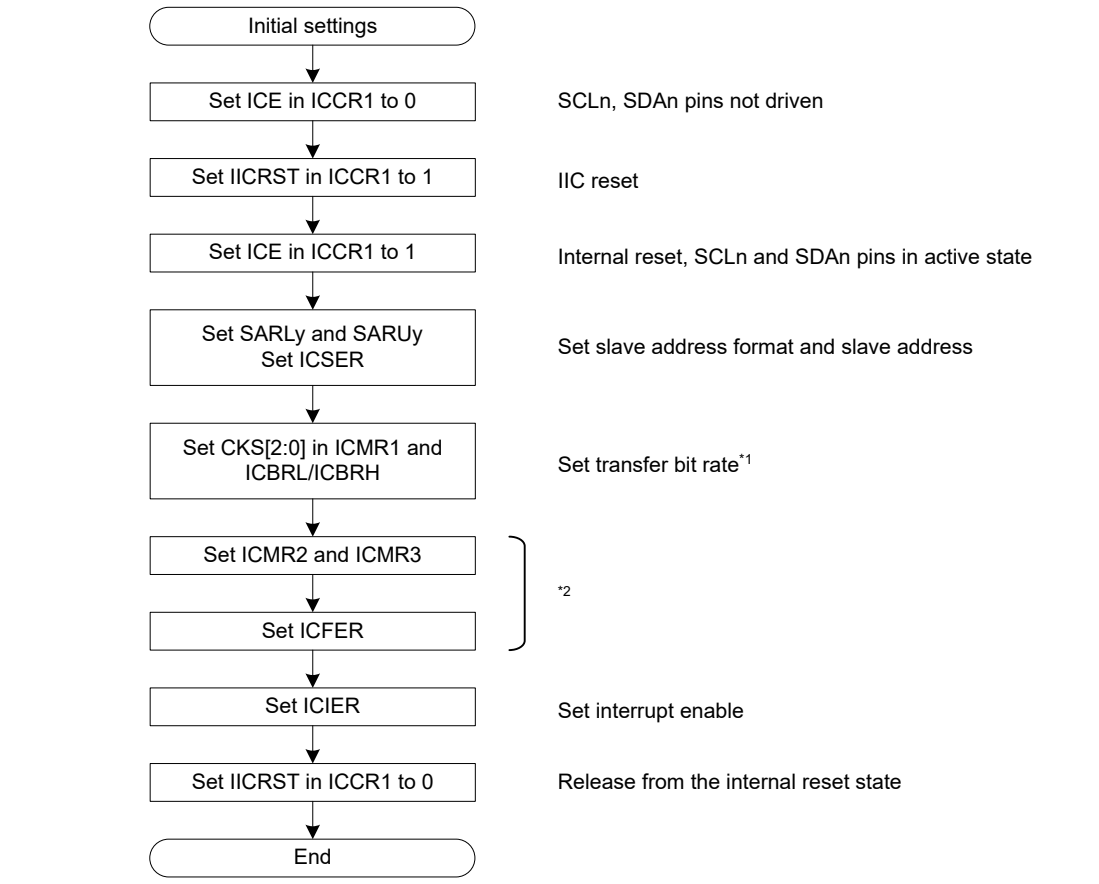
## 28.3.2 Initial Settings

Before starting data transmission or reception, initialize the IIC using the procedure shown in Figure 28.5.

1. Set the ICCR1.ICE bit to 0 to set the SCLn and SDAn pins to the inactive state.
2. Set the ICCR1.IICRST bit to 1 to initiate IIC reset.
3. Set the ICCR1.ICE bit to 1 to initiate internal reset.
4. Set the SARLy, SARUy, ICSER, ICMR1, ICBRH, and ICBRL registers (y = 0 to 2), and set the other registers as required. For initial settings of the IIC, see Figure 28.5.

5. When the required register settings are complete, set the ICCR1.IICRST bit to 0 to release the IIC reset.

This procedure is not required if the IIC initialization is already complete.




Note: n = 0
Note 1. When the IIC is used only in slave mode, set the ICBRL register to a value longer than the data setup time.
Note 2. Set these registers as required.

**Figure 28.5 Example IIC initialization flow**

## 28.3.3 Master Transmit Operation

In master transmit operation, the IIC outputs the SCL clock and transmitted data signals as the master device, and the slave device returns acknowledgments. Figure 28.6 shows an example of master transmission, and Figure 28.7 to Figure 28.9 show the operation timing in master transmission.

To set up and perform master transmission:

1. Process initial settings. For details, see section 28.3.2. Initial Settings.
2. Read the BBSY flag in ICCR2 to check that the bus is free, and then set the ST bit in ICCR2 to 1 (start condition request). On receiving the request, the IIC issues a start condition. At the same time, the BBSY and START flags in ICSR2 automatically set to 1 and the ST bit automatically is set to 0. At this time, if the start condition is detected and the internal levels for the SDA output state and the levels on the SDAn line match while the ST bit is 1, the IIC recognizes that issuance of the start condition as requested by the ST bit is successfully complete, and the MST and TRS bits in ICCR2 automatically set to 1, placing the IIC in master transmit mode. The TDRE flag in ICSR2 also automatically is set to 1 in response to the setting of the TRS bit to 1.

3. Check that the TDRE flag in ICSR2 is 1, and then write the value for transmission (the slave address and the R/W# bit) to ICDRT. When the transmit data is written to ICDRT, the TDRE flag automatically is set to 0, the data is transferred from ICDRT to ICDRS, and the TDRE flag again is set to 1. After the byte containing the slave address and R/W# bit is transmitted, the value of the TRS bit automatically updates to select master transmit or master receive mode based on the value of the transmitted R/W# bit. If the value of the R/W# bit was 0, the IIC continues in master transmit mode. Because the ICSR2.NACKF flag being 1 at this time indicates that no slave device recognized the address or there was an error in communications, write 1 to the ICCR2.SP bit to issue a stop condition.
   For data transmission with an address in the 10-bit format, start by writing 11110b, the 2 higher-order bits of the slave address, and W to ICDRT as the first address transmission. For the second address transmission, write the 8 lower-order bits of the slave address to ICDRT.
4. Check that the TDRE flag in ICSR2 is 1, and then write the transmit data to the ICDRT register. The IIC automatically holds the SCLn line low until the transmit data is ready or a stop condition is issued.
5. After all bytes of transmit data are written to the ICDRT register, wait until the value in the TEND flag in ICSR2 returns to 1, and then set the SP bit in ICCR2 to 1 (stop condition requested). On receiving a stop condition request, the IIC issues the stop condition. Regarding issuing a stop condition, see section 28.11.3. Issuing a Stop Condition.
6. On detecting the stop condition, the IIC automatically sets the MST and TRS bits in ICCR2 to 00b and enters slave receive mode. Additionally, it automatically sets the TDRE and TEND flags to 0, and sets the STOP flag in ICSR2 to 1.
7. Check that the ICSR2.STOP flag is 1, and then set the ICSR2.NACKF and STOP flags to 0 for the next transfer operation.

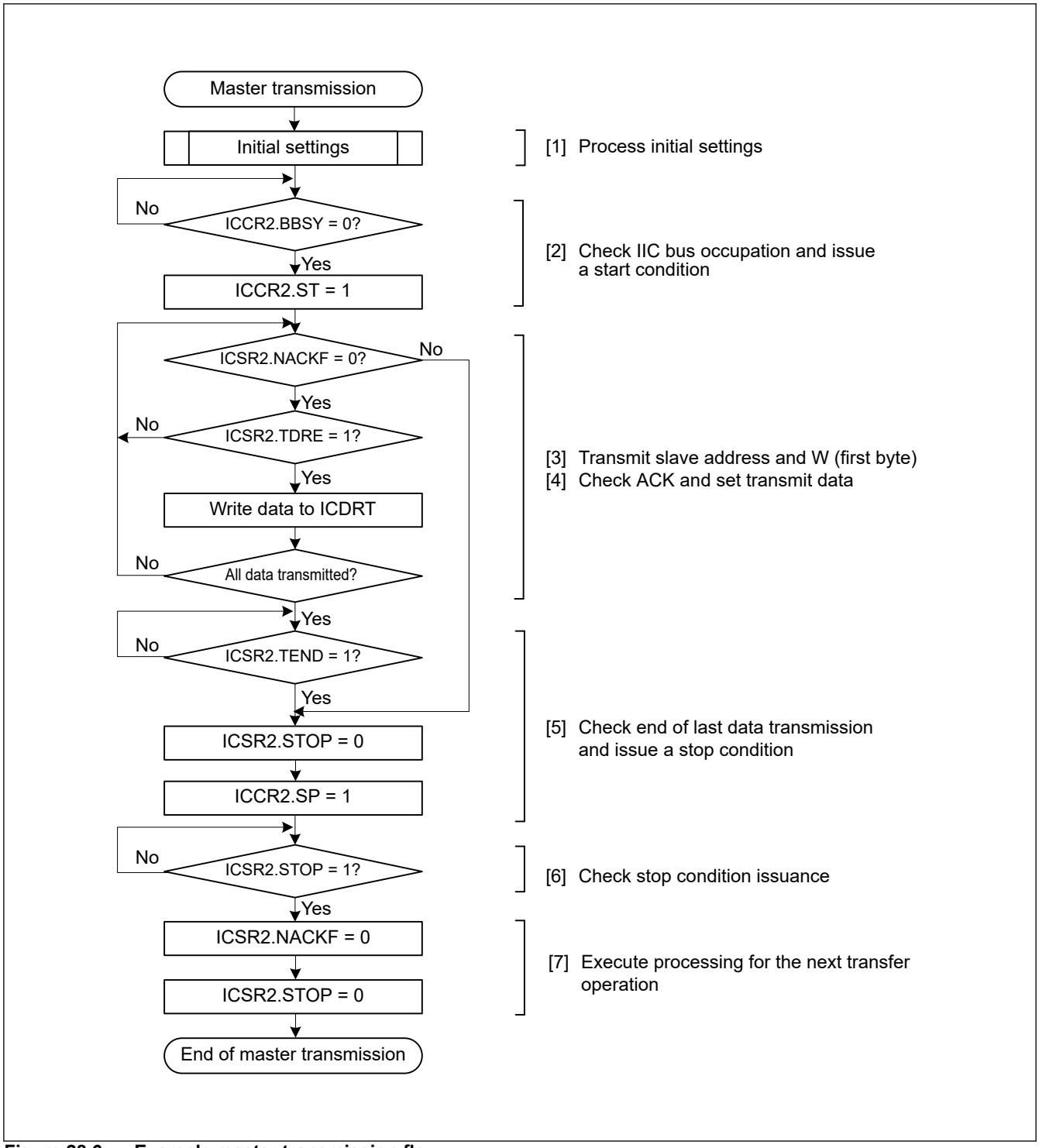



**Figure 28.6 Example master transmission flow**

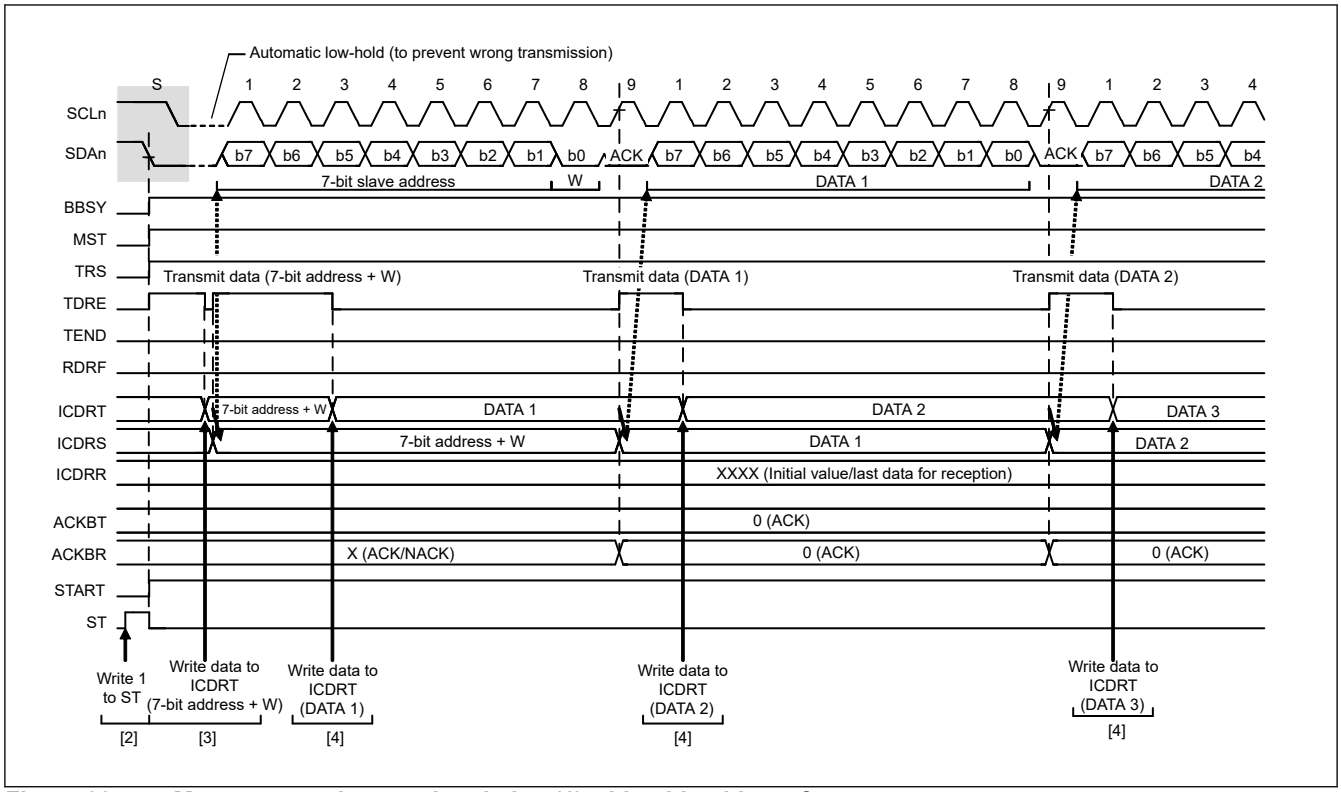



**Figure 28.7 Master transmit operation timing (1) with 7-bit address format**

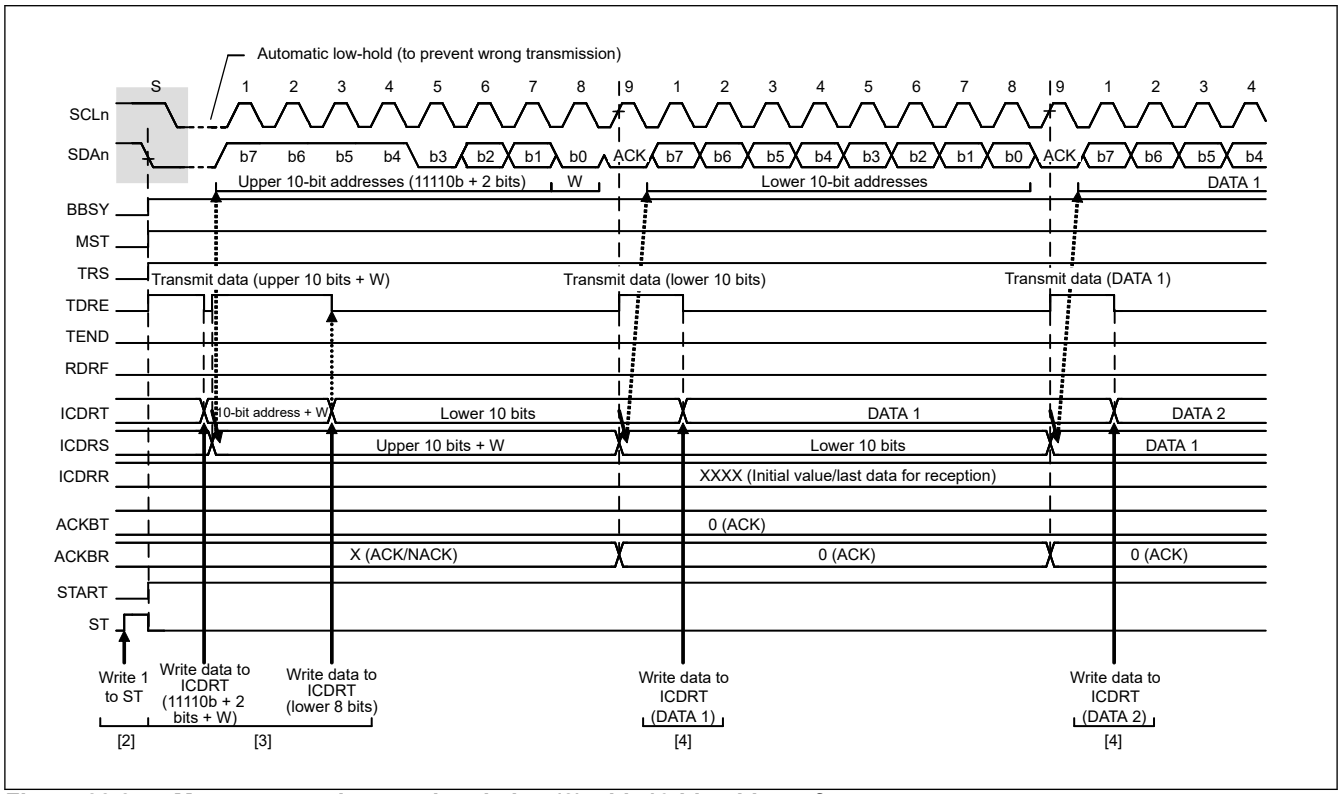



**Figure 28.8 Master transmit operation timing (2) with 10-bit address format**

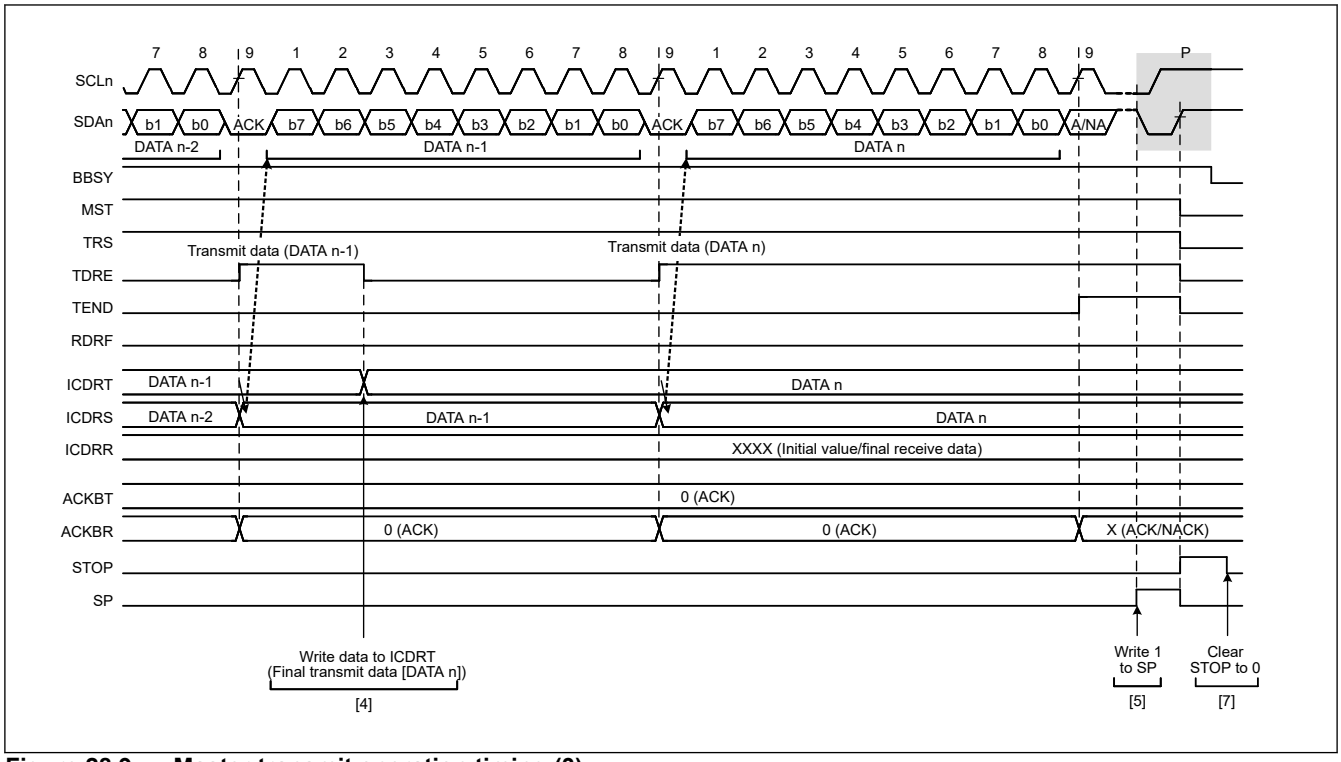



**Figure 28.9 Master transmit operation timing (3)**

## 28.3.4 Master Receive Operation

In master receive operation, the IIC as a master device outputs the SCL clock, receives data from the slave device, and returns acknowledgments. Because the IIC must start by sending a slave address to the associated slave device, this part of the procedure is performed in master transmit mode, but the subsequent steps are in master receive mode.

Figure 28.10 and Figure 28.11 show examples of master reception (7-bit address format), and Figure 28.12 to Figure 28.14 show the operation timing in master reception.

To set up and perform master reception:

1. Process initial settings. For details, see section 28.3.2. Initial Settings.
2. Read the BBSY flag in ICCR2 to check that the bus is free, and then set the ST bit in ICCR2 to 1 (start condition request). On receiving the request, the IIC issues a start condition. When the IIC detects the start condition, the BBSY and START flags in ICSR2 automatically set to 1, and the ST bit automatically is set to 0. At this time, if the start condition is detected and the levels for the SDA output and the levels on the SDAn line match while the ST bit is 1, the IIC recognizes that issuance of the start condition as requested by the ST bit is successfully complete, and the MST and TRS bits in ICCR2 automatically set to 1, placing the IIC in master transmit mode. The TDRE flag in ICSR2 also automatically is set to 1 in response to the setting of the TRS bit to 1.
3. Check that the TDRE flag in ICSR2 is 1, and then write the value for transmission (the first byte indicates the slave address and value of the R/W# bit) to ICDRT. When the transmit data is written to ICDRT, the TDRE flag automatically is set to 0, the data is transferred from ICDRT to ICDRS, and the TDRE flag again is set to 1. When the byte containing the slave address and R/W# bit is transmitted, the value of the ICCR2.TRS bit automatically updates to select transmit or receive mode based on the value of the transmitted R/W# bit. If the value of the R/W# bit is 1, the TRS bit is set to 0 on the rising edge of the ninth cycle of the SCL clock, placing the IIC in master receive mode. At this time, the TDRE flag is set to 0 and the ICSR2.RDRF flag automatically is set to 1.
   Because the ICSR2.NACKF flag being 1 at this time indicates that no slave device recognized the address or there was an error in communications, write 1 to the ICCR2.SP bit to issue a stop condition.
   For master reception from a device with a 10-bit address, start by using master transmission to issue the 10-bit address, and then issue a restart condition. After that, transmitting 11110b, the two higher-order bits of the slave address, and the R bit places the IIC in master receive mode.
4. Dummy read ICDRR after confirming that the RDRF flag in ICSR2 is 1. This makes the IIC start output of the SCL clock and start data reception.

5. After 1 byte of data is received, the RDRF flag in ICSR2 is set to 1 on the rising edge of the 8th or 9th cycle of the SCL clock, as selected in the RDRFS bit in ICMR3. Reading ICDRR at this time produces the received data, and the RDRF flag is automatically set to 0 at the same time. Additionally, the value of the acknowledgment field received during the ninth cycle of the SCL clock is returned as the value set in the ICMR3.ACKBT bit. If the next byte to be received is the second-to-last byte, set the ICMR3.WAIT bit to 1 for wait insertion before reading ICDRR, containing the second-to-last byte. In addition to enabling NACK output, even when interrupts or other operations result in delays in setting the ICMR3.ACKBT bit to 1 (NACK) in step (6), this fixes the SCLn line to the low level on the rising edge of the ninth clock cycle in reception of the last byte, which enables the issuing of a stop condition.
6. When the ICMR3.RDRFS bit is 0, and the slave device must be notified that it is to end transfer for data reception after transfer of the next and final byte, set the ICMR3.ACKBT bit to 1 (NACK).
7. After reading the second-to-last byte from the ICDRR register, if the value of the ICSR2.RDRF flag is 1, write 1 to the SP bit in ICCR2 (stop condition requested), and then read the last byte from ICDRR. When ICDRR is read, the IIC is released from the wait state and issues the stop condition after low-level output in the ninth clock cycle is complete or the SCLn line is released from the low-hold state.
8. On detecting the stop condition, the IIC automatically sets the MST and TRS bits in ICCR2 to 00b and enters slave receive mode. Additionally, detection of the stop condition sets the ICSR2.STOP flag to 1.
9. Check that the ICSR2.STOP flag is 1, then set the ICSR2.NACKF and STOP flags to 0 for the next transfer operation.

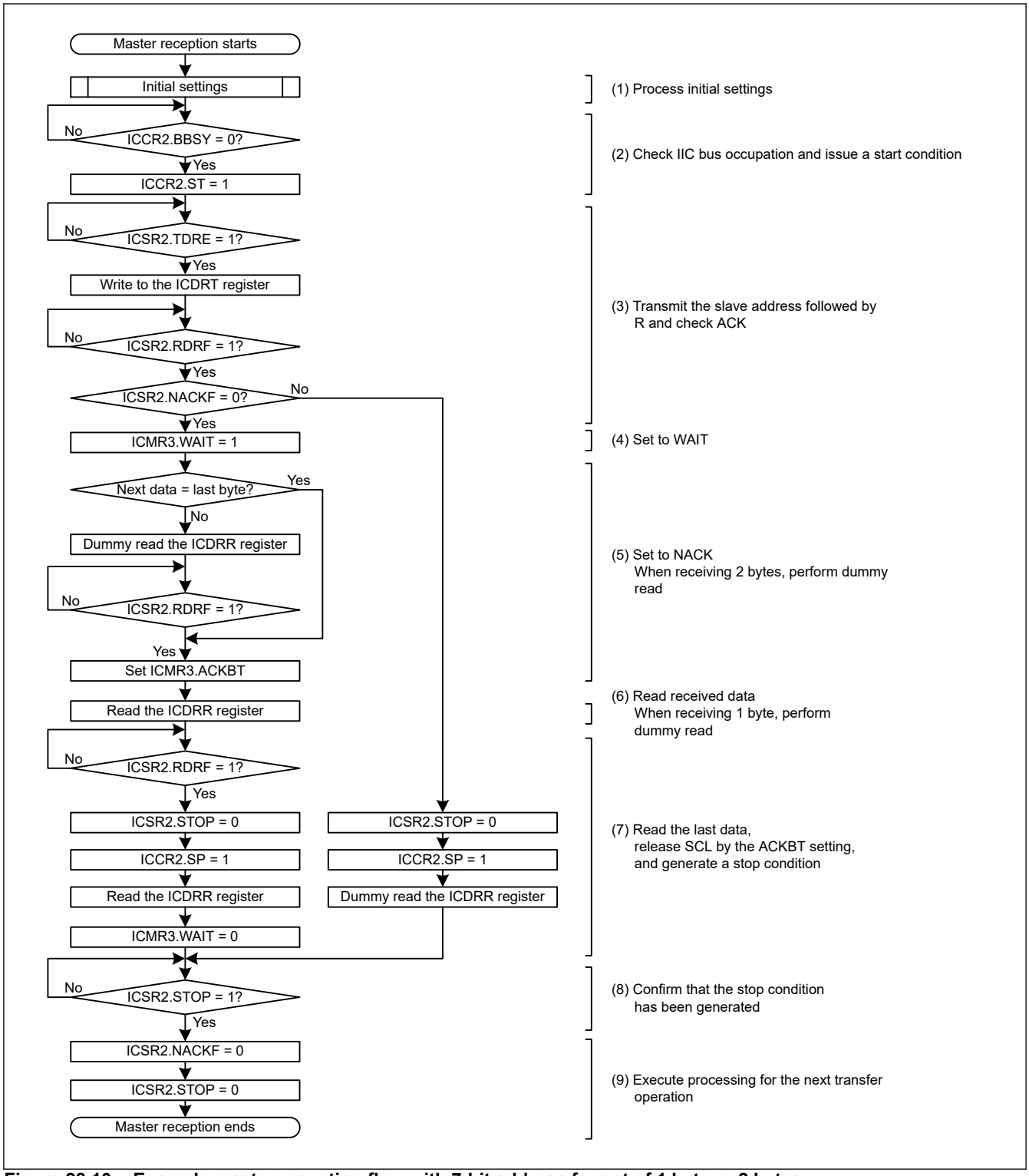



**Figure 28.10 Example master reception flow with 7-bit address format of 1 byte or 2 bytes**

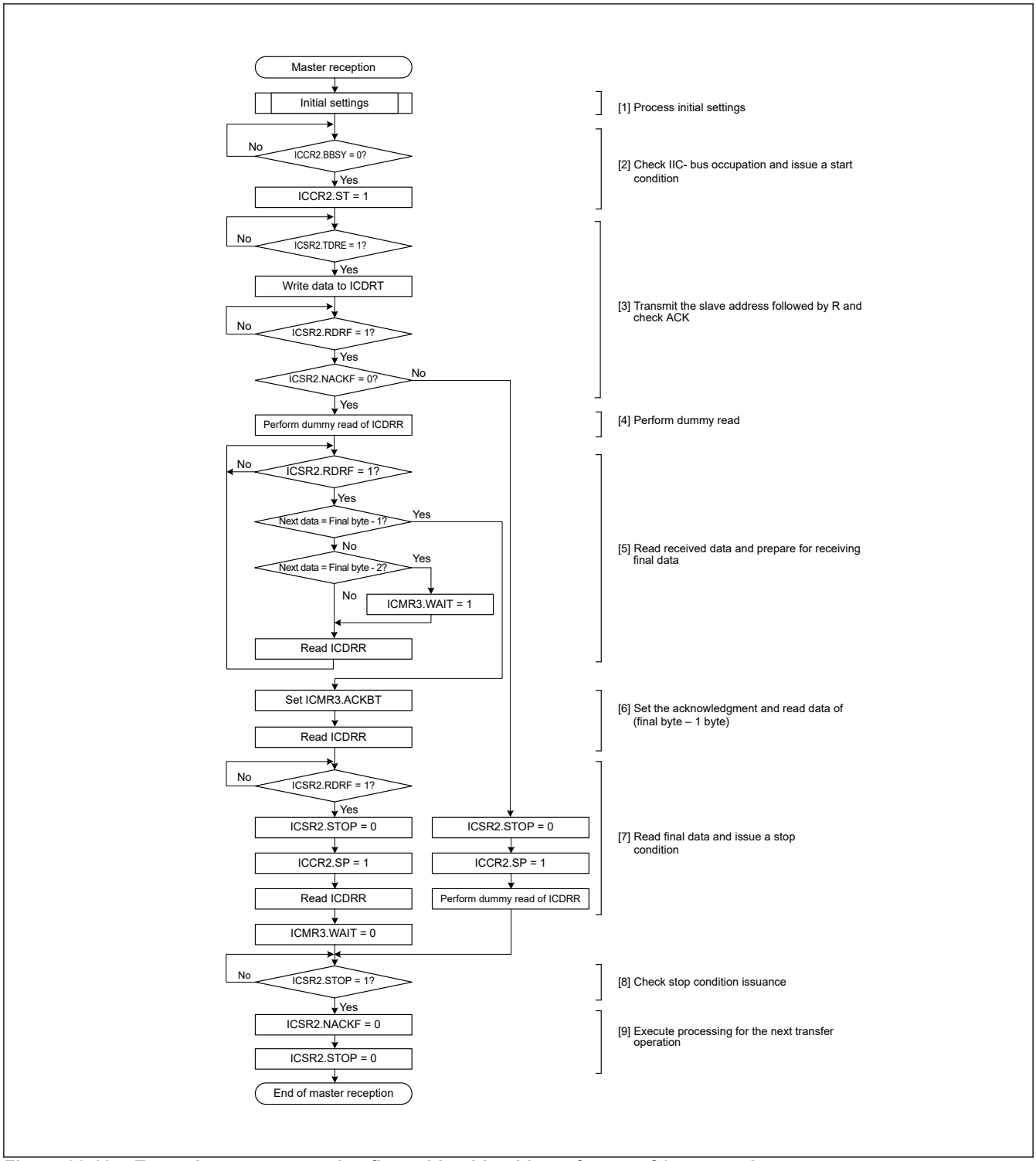



**Figure 28.11 Example master reception flow with 7-bit address format of 3 or more bytes**

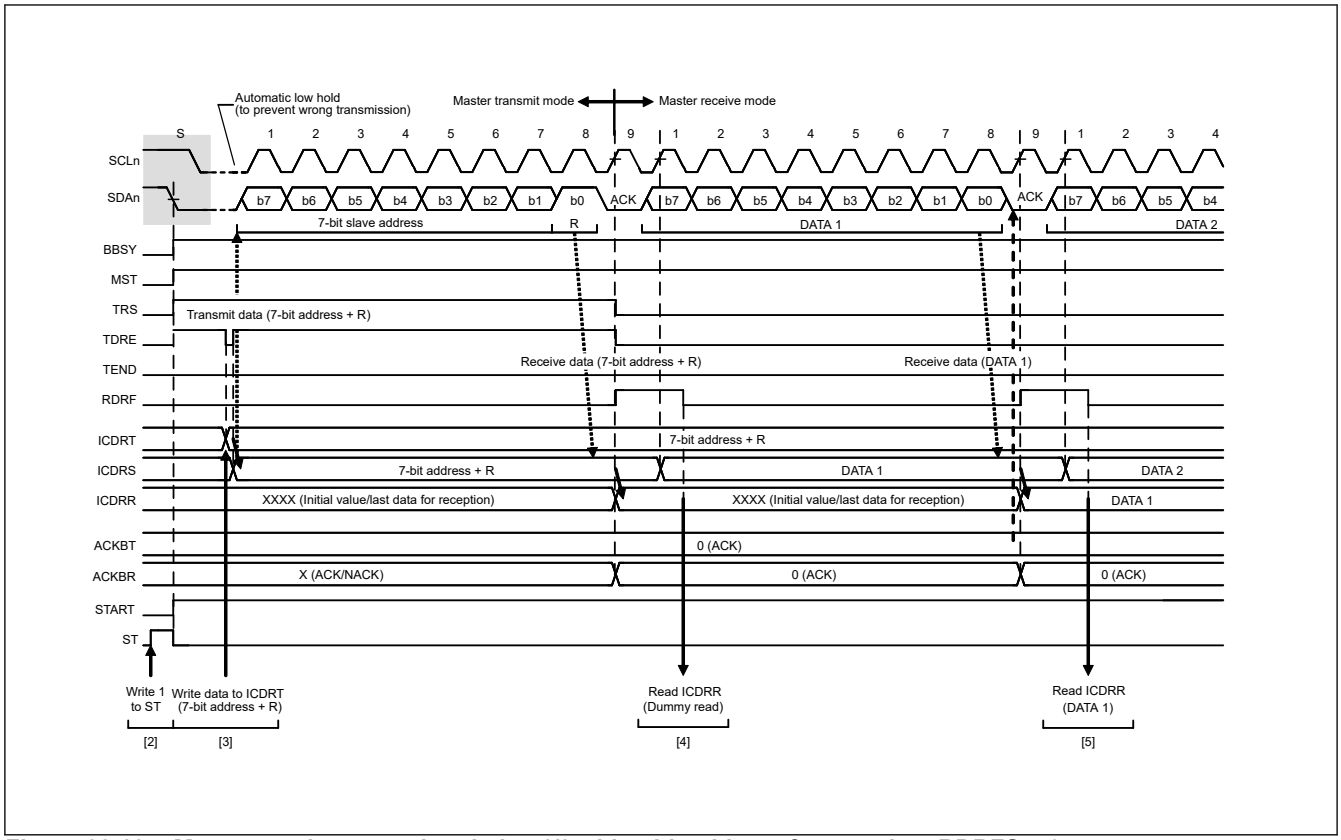



**Figure 28.12 Master receive operation timing (1) with 7-bit address format when RDRFS = 0**

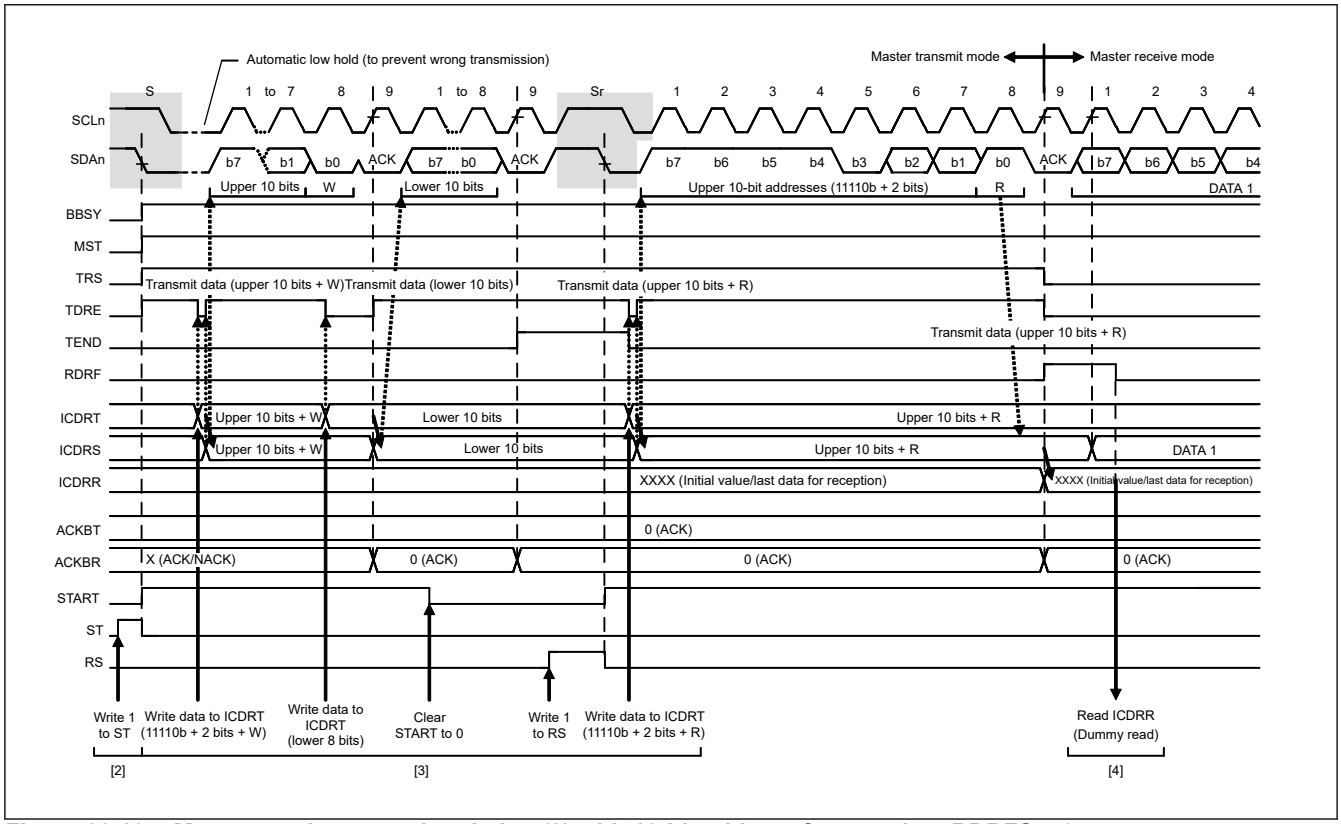



**Figure 28.13 Master receive operation timing (2) with 10-bit address format when RDRFS = 0**

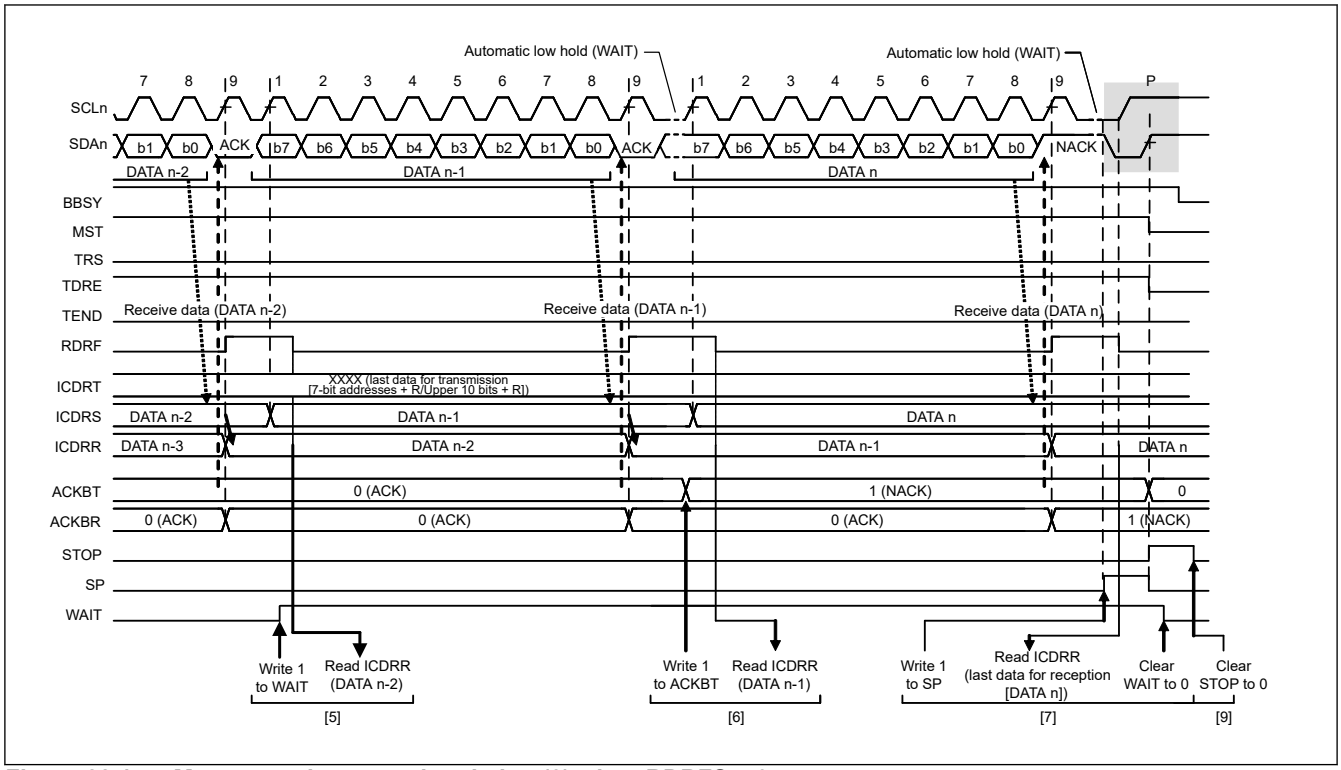



**Figure 28.14 Master receive operation timing (3) when RDRFS = 0**

## 28.3.5 Slave Transmit Operation

In slave transmit operation, the master device outputs the SCL clock, the IIC transmits data as a slave device, and the master device returns acknowledgments.

Figure 28.15 shows an example of slave transmission, and Figure 28.16 and Figure 28.17 show the operation timing in slave transmission.

To set up and perform slave transmission:

1. Initialize the IIC using the procedure in section 28.3.2. Initial Settings.
   After initialization, the IIC stays in the standby state until it receives a slave address that matches.
2. After receiving a matching slave address, the IIC sets one of the associated ICSR1.HOA, GCA, and AASy flags (y = 0 to 2) to 1 on the rising edge of the ninth cycle of the SCL clock and outputs the value set in the ICMR3.ACKBT bit to the acknowledge bit on the ninth cycle of the SCL clock. If the value of the R/W# bit is 1, the IIC automatically places itself in slave transmit mode by setting both the ICCR2.TRS bit and the ICSR2.TDRE flag to 1.
3. Check that the ICSR2.TDRE flag is 1, then write the transmit data to the ICDRT register. If the IIC receives no acknowledge from the master device (receives an NACK signal) while the ICFER.NACKE bit is 1, the IIC suspends transfer of the next data.
4. Wait until the ICSR2.TEND flag is set to 1 while the ICSR2.TDRE flag is 1, after the ICSR2.NACKF flag is set to 1 or the last byte for transmission is written to the ICDRT register. When the ICSR2.NACKF flag or the TEND flag is 1, the IIC drives the SCLn line low on the ninth falling edge of the SCL clock.
5. When the ICSR2.NACKF or ICSR2.TEND flag is 1, dummy read ICDRR to complete the processing. This releases the SCLn line.
6. On detecting the stop condition, the IIC automatically sets the ICSR1.HOA, GCA, and AASy flags (y = 0 to 2), the ICSR2.TDRE and TEND flags, and the ICCR2.TRS bit to 0, and enters slave receive mode.
7. Check that the ICSR2.STOP flag is 1, then set the ICSR2.NACKF and STOP flags to 0 for the next transfer operation.

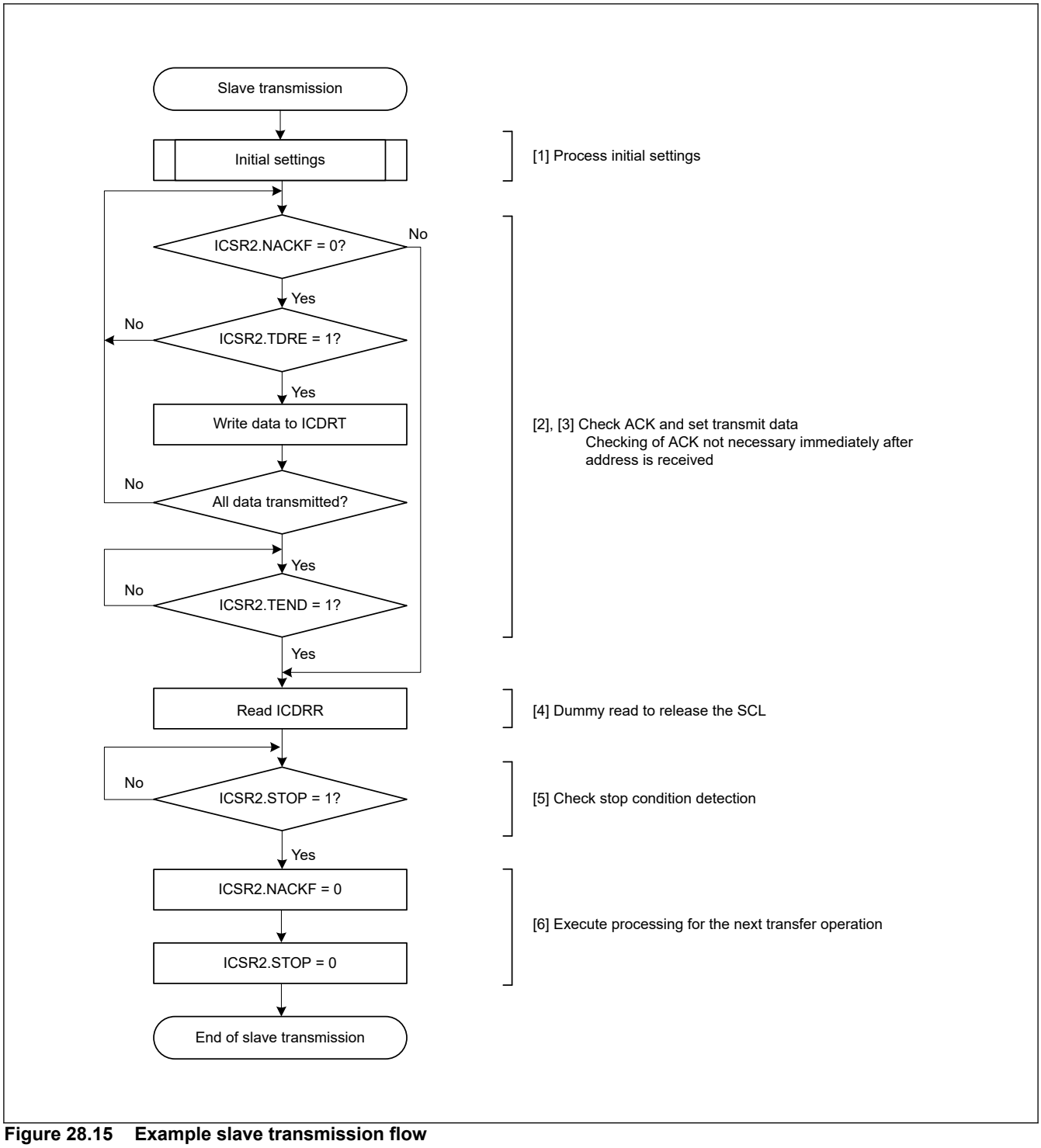



**Figure 28.15 Example slave transmission flow**

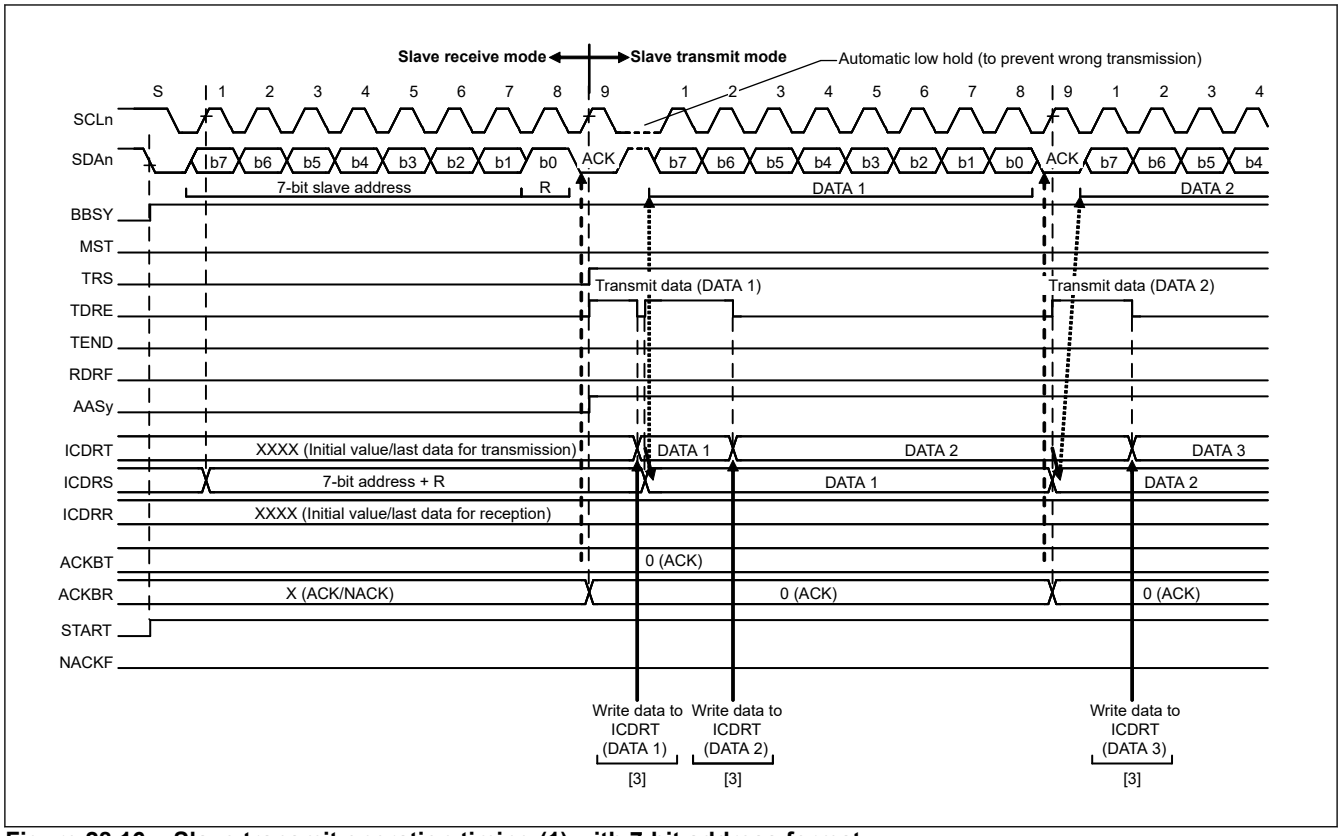



**Figure 28.16 Slave transmit operation timing (1) with 7-bit address format**

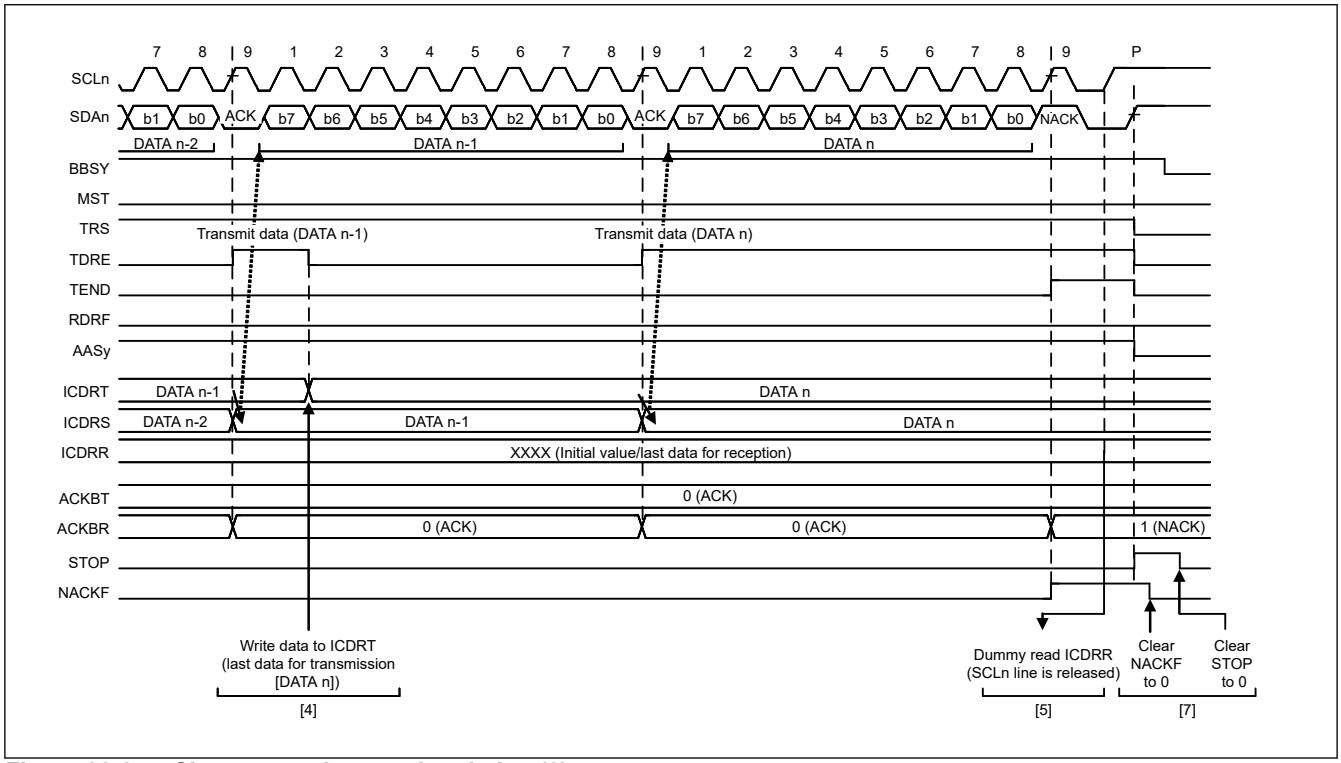



**Figure 28.17 Slave transmit operation timing (2)**

## 28.3.6 Slave Receive Operation

In slave receive operation, the master device outputs the SCL clock and transmit data, and the IIC returns acknowledgments as a slave device.

Figure 28.18 shows an example of slave reception, and Figure 28.19 and Figure 28.20 show the operation timing in slave reception.

To set up and perform slave reception:

1. Initialize the IIC using the procedure in section 28.3.2. Initial Settings.
   After initialization, the IIC stays in the standby state until it receives a slave address that matches.
2. After receiving a matching slave address, the IIC sets one of the associated ICSR1.HOA, GCA, and AASy flags (y = 0 to 2) to 1 on the rising edge of the ninth cycle of the SCL clock and outputs the value set in the ICMR3.ACKBT bit to the acknowledge bit on the ninth cycle of the SCL clock. If the value of the R/W# bit is 0, the IIC continues to place itself in slave receive mode and sets the RDRF flag in ICSR2 to 1.
3. Check that the ICSR2.STOP flag is 0 and the ICSR2.RDRF flag is 1, then dummy read ICDRR. The dummy value consists of the slave address and R/W# bit when the 7-bit address format is selected, or the lower 8 bits when the 10-bit address format is selected.
4. When ICDRR is read, the IIC automatically sets the ICSR2.RDRF flag to 0. If reading of ICDRR is delayed and a next byte is received while the RDRF flag is still set to 1, the IIC holds the SCLn line low until 1 SCL cycle before the point where RDRF must be set. In this case, reading ICDRR releases the SCLn line from being held low.
   When the ICSR2.STOP flag is 1 and the ICSR2.RDRF flag is also 1, read ICDRR until all the data is completely received.
5. On detecting the stop condition, the IIC automatically clears the ICSR1.HOA, GCA, and AASy flags (y = 0 to 2) to 0.
6. Check that the ICSR2.STOP flag is 1, then set the ICSR2.STOP flag to 0 for the next transfer operation.

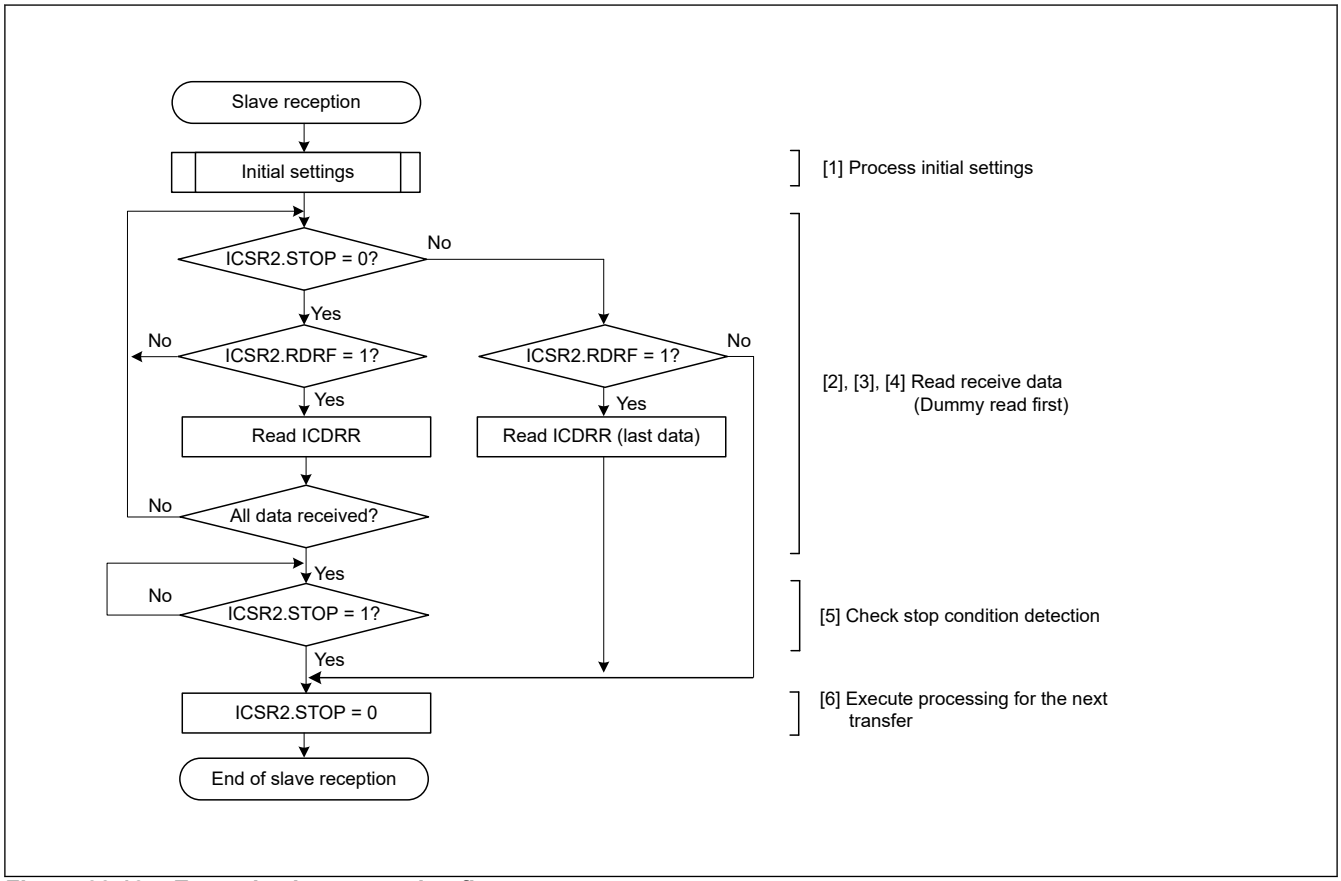



**Figure 28.18 Example slave reception flow**

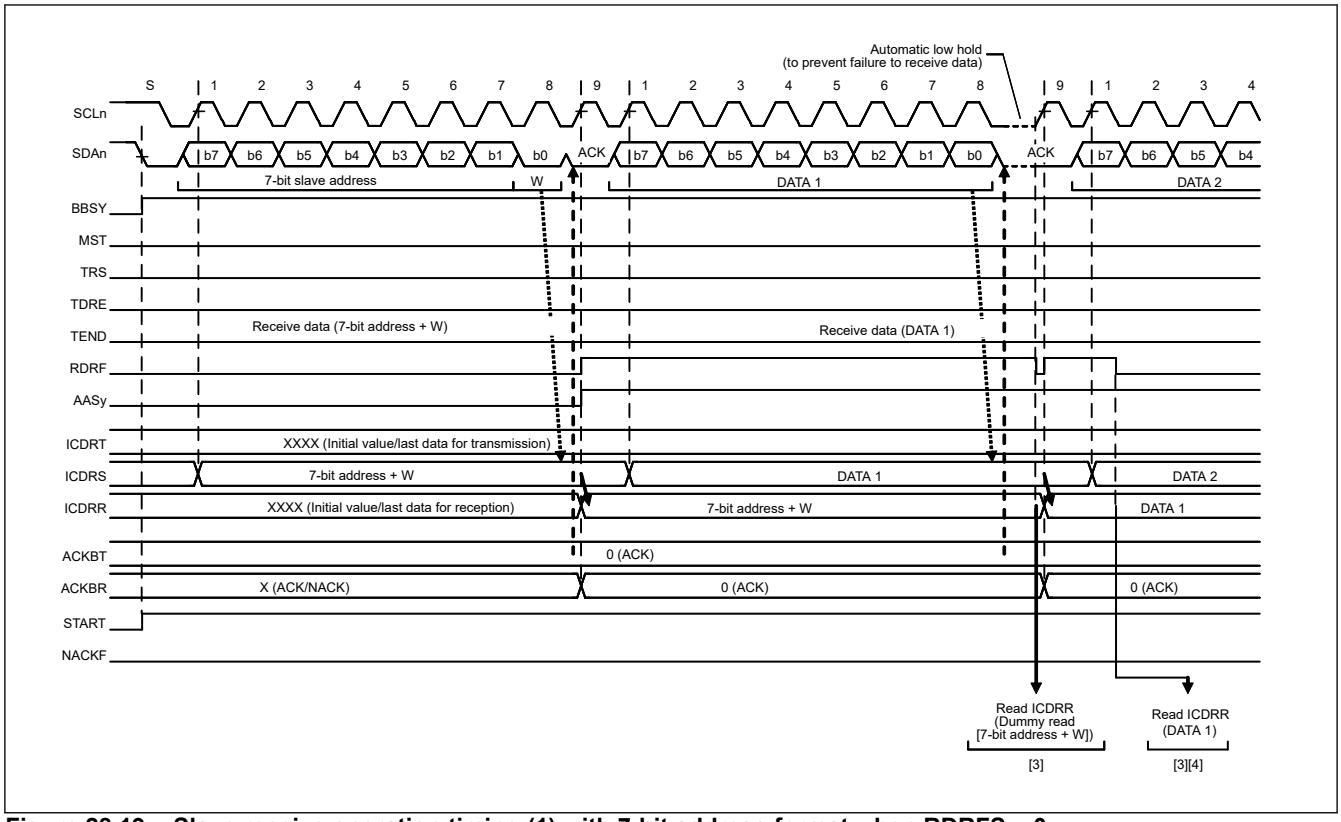



**Figure 28.19 Slave receive operation timing (1) with 7-bit address format when RDRFS = 0**

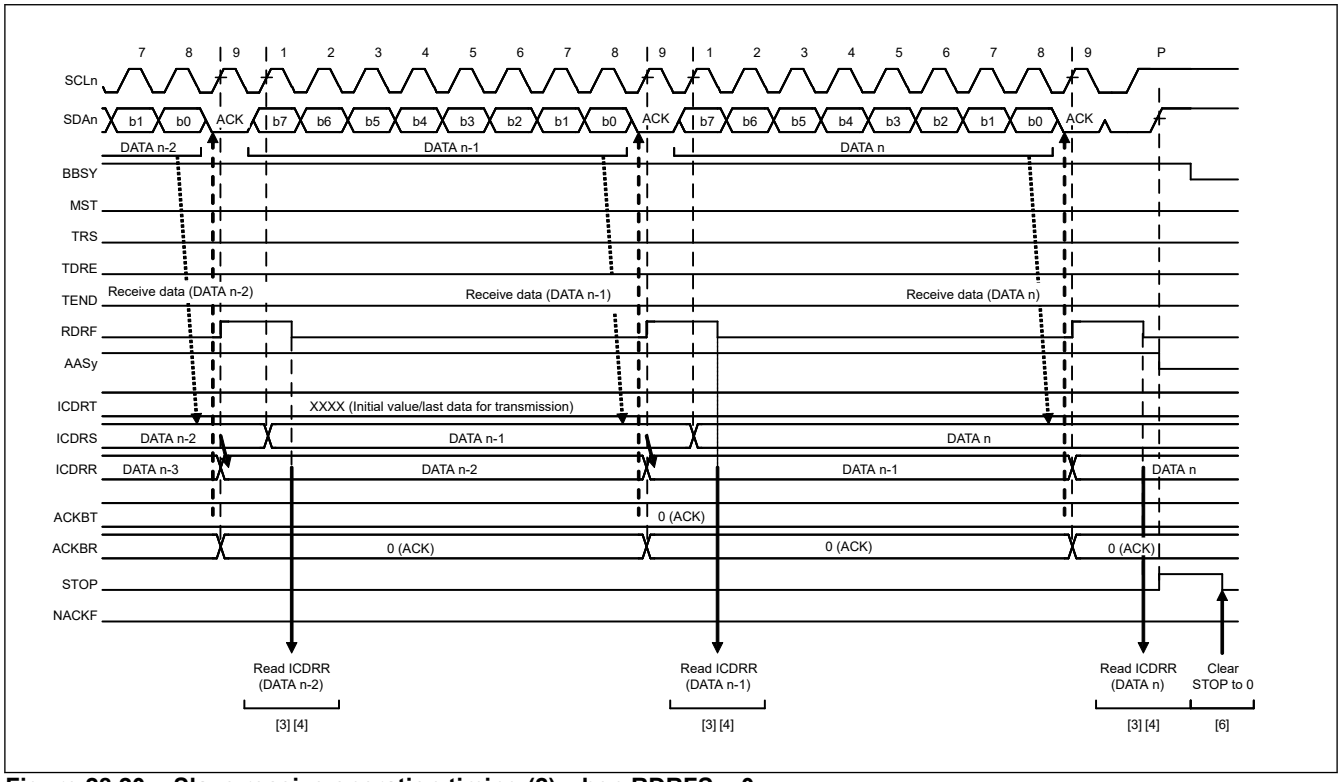



**Figure 28.20 Slave receive operation timing (2) when RDRFS = 0**

## 28.4 SCL Synchronization Circuit

For generation of the SCL clock, the IIC starts counting the value for the high-level period specified in ICBRH when it detects a rising edge on the SCLn line, and it drives the SCLn line low when it completes counting. When the IIC detects the falling edge of the SCLn line, it starts counting the value for the low-level period specified in ICBRL, and then it stops

driving the SCLn line, releasing the line, when it completes counting. The IIC repeats this process to generate the SCL clock.

If multiple master devices are connected to the $I^2C$ bus, a collision of SCL signals might arise because of contention with another master device. In such cases, the master devices must synchronize their SCL signals. Because this synchronization of SCL signals must be bit-by-bit, the IIC is equipped with an SCL synchronization circuit to obtain bit-by-bit synchronization of the SCL clock signals by monitoring the SCLn line while in master mode.

When the IIC detects a rising edge on the SCLn line and starts counting the high-level period specified in ICBRH.BRH[4:0], and the level on the SCLn line falls because an SCL signal is being generated by another master device, the IIC performs the following:

1. Stops counting when it detects the falling edge.
2. Drives the level on the SCLn line low.
3. Starts counting the low-level period specified in ICBRL.BRL[4:0].

When the IIC finishes counting the low-level period, it stops driving the SCLn line low to release the line. If the low-level period of the SCL clock signal from the other master device is longer than the low-level period set in the IIC, the low-level period of the SCL signal is extended. When the low-level period for the other master device ends, the SCL signal rises because the SCLn line is released.

When the IIC finishes outputting the low-level period of the SCL clock, the SCLn line is released and the SCL clock rises. That is, when SCL signals from more than one master are contending, the high-level period of the SCL signal is synchronized with that of the clock with the narrower period, and the low-level period of the SCL signal is synchronized with that of the clock with the broader period. However, such synchronization of the SCL signal is only enabled when the SCLE bit in ICFER is set to 1.

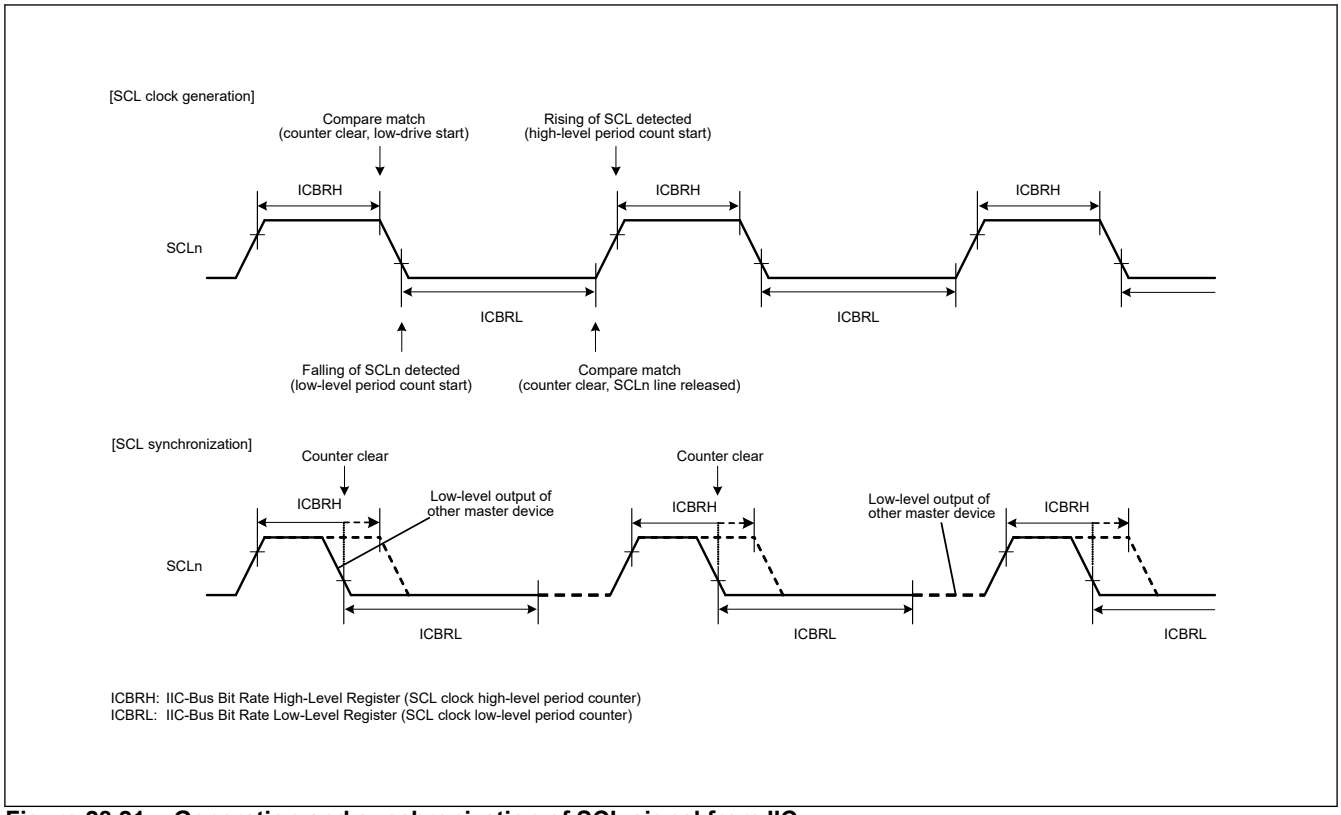


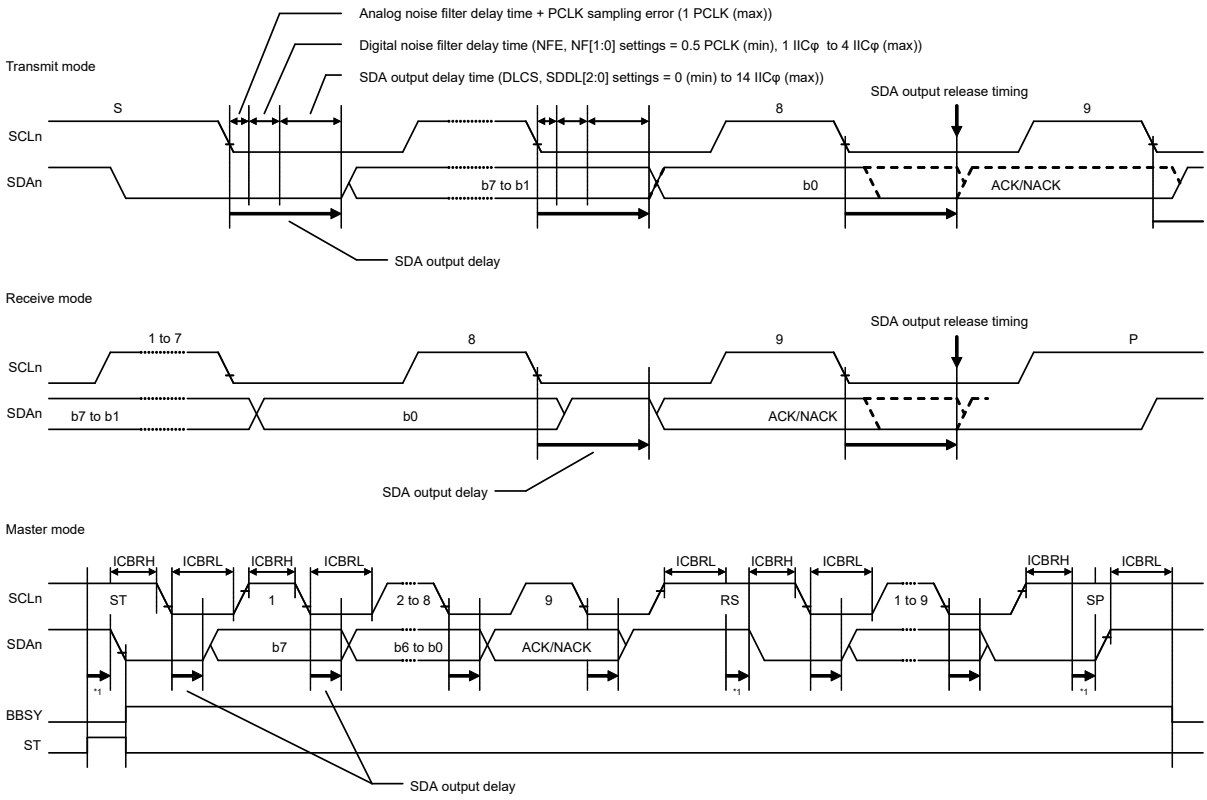

**Figure 28.21 Generation and synchronization of SCL signal from IIC**

## 28.5 SDA Output Delay Function

The IIC module incorporates a function for delaying output on the SDA line. The delay can be applied to all output on the SDA line, including issuing of the start, restart, and stop conditions, data, and the ACK and NACK signals.

With this function, SDA output is delayed from the detection of a falling edge of the SCL signal to ensure that the SDA signal is output within the interval during which the SCL clock is low. This approach helps prevent erroneous operation of communications devices, with the aim of satisfying the 300-ns minimum data-hold time requirement of the SMBus specification. The output delay function is enabled by setting the SDDL[2:0] bits in ICMR2 to a value other than 000b, and disabled by setting the same bits to 000b.

When the SDA output delay function is enabled, for example, the DLCS bit in ICMR2 selects the clock source for the SDA output delay counter, either as the internal base clock (IICφ) for the IIC module or as the internal base clock divided by 2 (IICφ/2). The counter counts the number of cycles set in the SDDL[2:0] bits in ICMR2. When the delay cycles count is reached, the IIC module places the required output (start, restart, or stop condition, data, or an ACK or NACK signal) on the SDA line.




Note: PCLK = PCLKB
Note 1. The output delay function is set by the DLCS and SDDL[2:0] bits when a start (ST), restart (RS), or stop (SP) condition is issued.

**Figure 28.22 SDA output delay function**

## 28.6 Digital Noise Filter Circuits

The internal circuitry sees the states of the SCLn and SDAn pins through analog and digital noise-filter circuits. Figure 28.23 shows a block diagram of the digital noise-filter circuit.

The on-chip digital noise-filter circuit of the IIC consists of four flip-flop circuit stages connected in series and a match-detection circuit. The number of valid stages in the digital noise filter is selected in the NF[1:0] bits in ICMR3. The selected number of valid stages determines the noise-filtering capability as a period from 1 to 4 IICφ cycles.

The input signal to the SCLn pin (or SDAn pin) is sampled on falling edges of the IICφ signal. When the input signal level matches the output level of the number of valid flip-flop circuit stages as selected in the NF[1:0] bits in ICMR3, the signal level is seen in the subsequent stage. If the signal levels do not match, the previous value is saved.

If the ratio between the frequency of the internal operating clock (PCLKB) and the transfer rate is small, for example, if data transfer is 400 kbps with PCLKB = 4 MHz, the characteristics of the digital noise filter might lead to the elimination of

required signals as noise. In such cases, it is possible to disable the digital noise-filter circuit by setting the ICFER.NFE bit to 0, and use only the analog noise filter circuit.

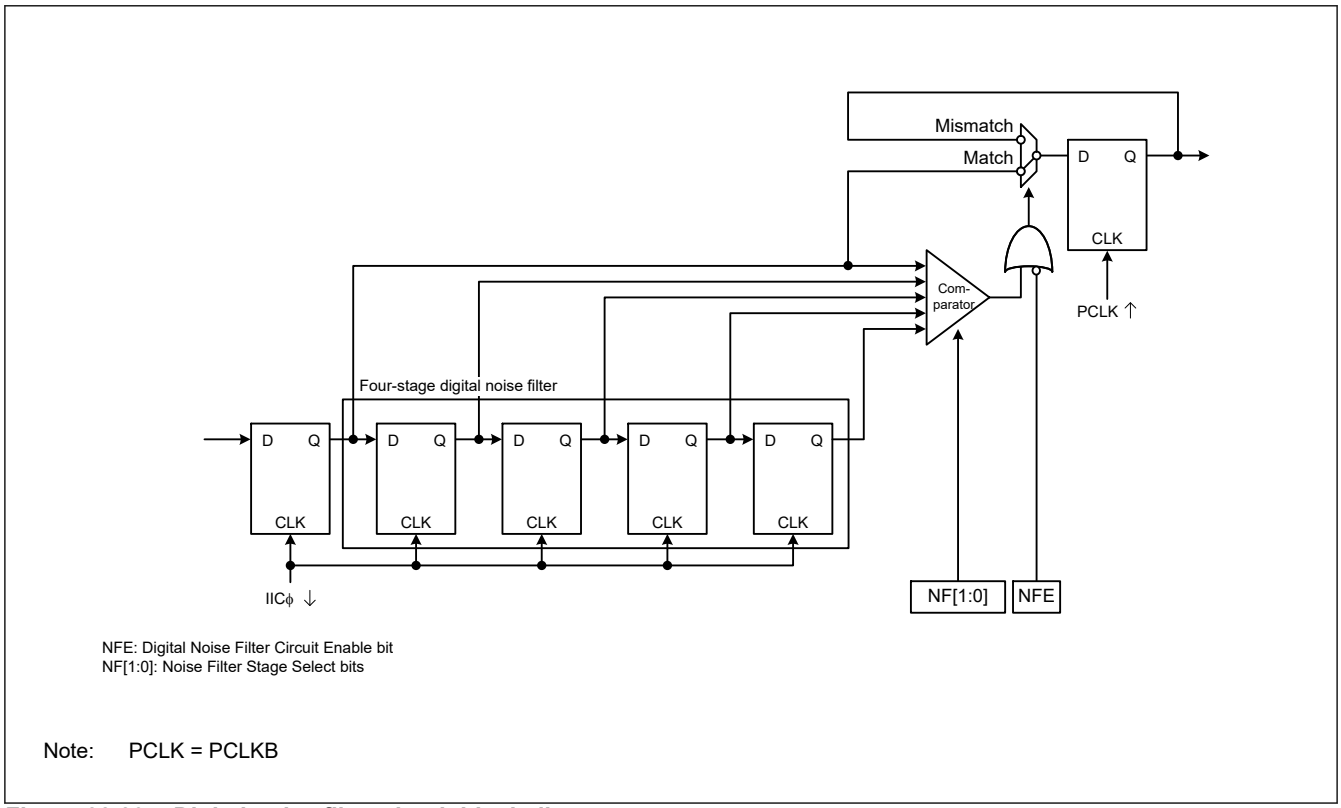



**Figure 28.23 Digital noise filter circuit block diagram**

## 28.7 Address Match Detection

The IIC can set three unique slave addresses in addition to the general call address and host address. The slave addresses can be 7-bit or 10-bit slave addresses.

### 28.7.1 Slave-Address Match Detection

The IIC can set three unique slave addresses and has a slave address detection function for each unique slave address. When the SARyE bit (y = 0 to 2) in ICSER is set to 1, the slave addresses set in SARUy and SARLy (y = 0 to 2) can be detected.

When the IIC detects a match of the set slave address, the associated AASy flag (y = 0 to 2) in ICSR1 is set to 1 on the rising edge of the ninth SCL clock cycle, and the RDRF flag in ICSR2 or the TDRE flag in ICSR2 is set to 1 by the subsequent R/W# bit. This causes a receive data full interrupt (IICn_RXI) or transmit data empty interrupt (IICn_TXI) to be generated. The AASy flag identifies which slave address is specified.

Figure 28.24 to Figure 28.26 show the AASy flag set timing in three cases.

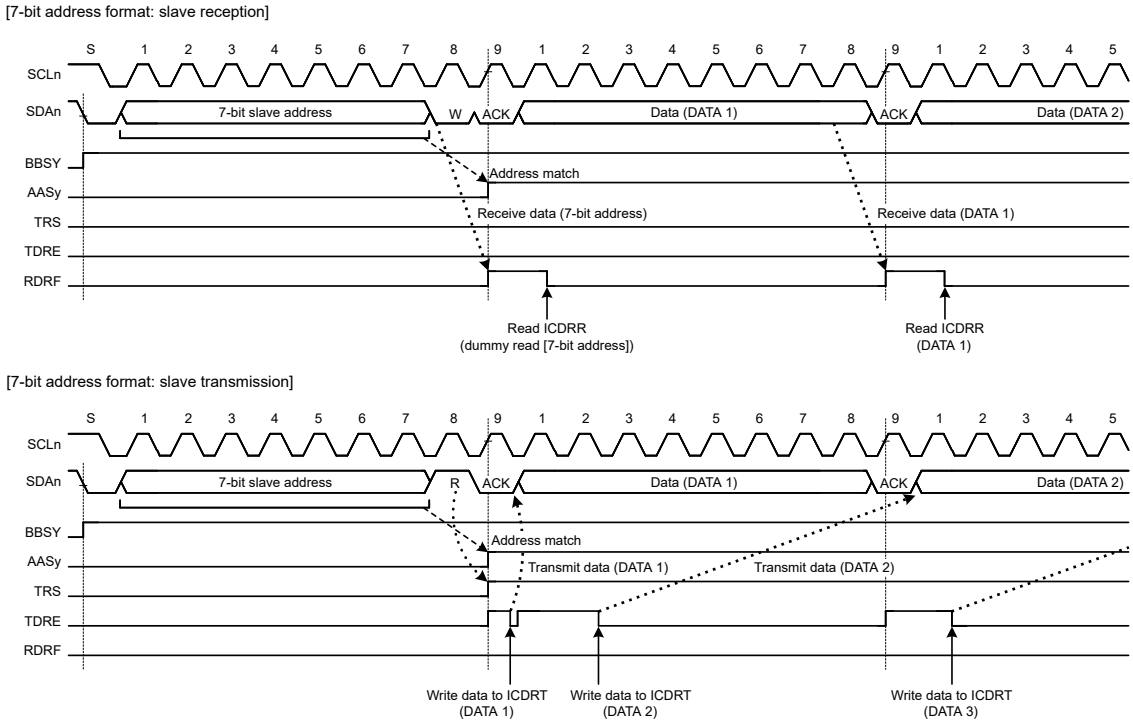



**Figure 28.24 AASy flag set timing with 7-bit address format**

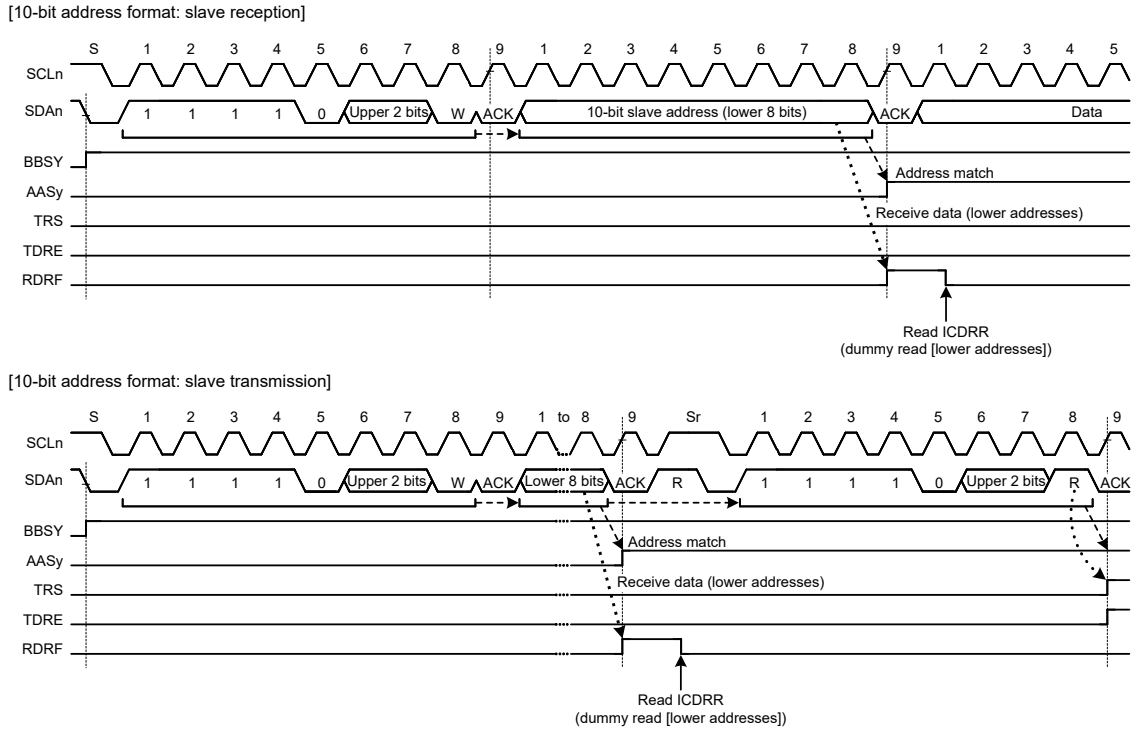



**Figure 28.25 AASy flag set timing with 10-bit address format**

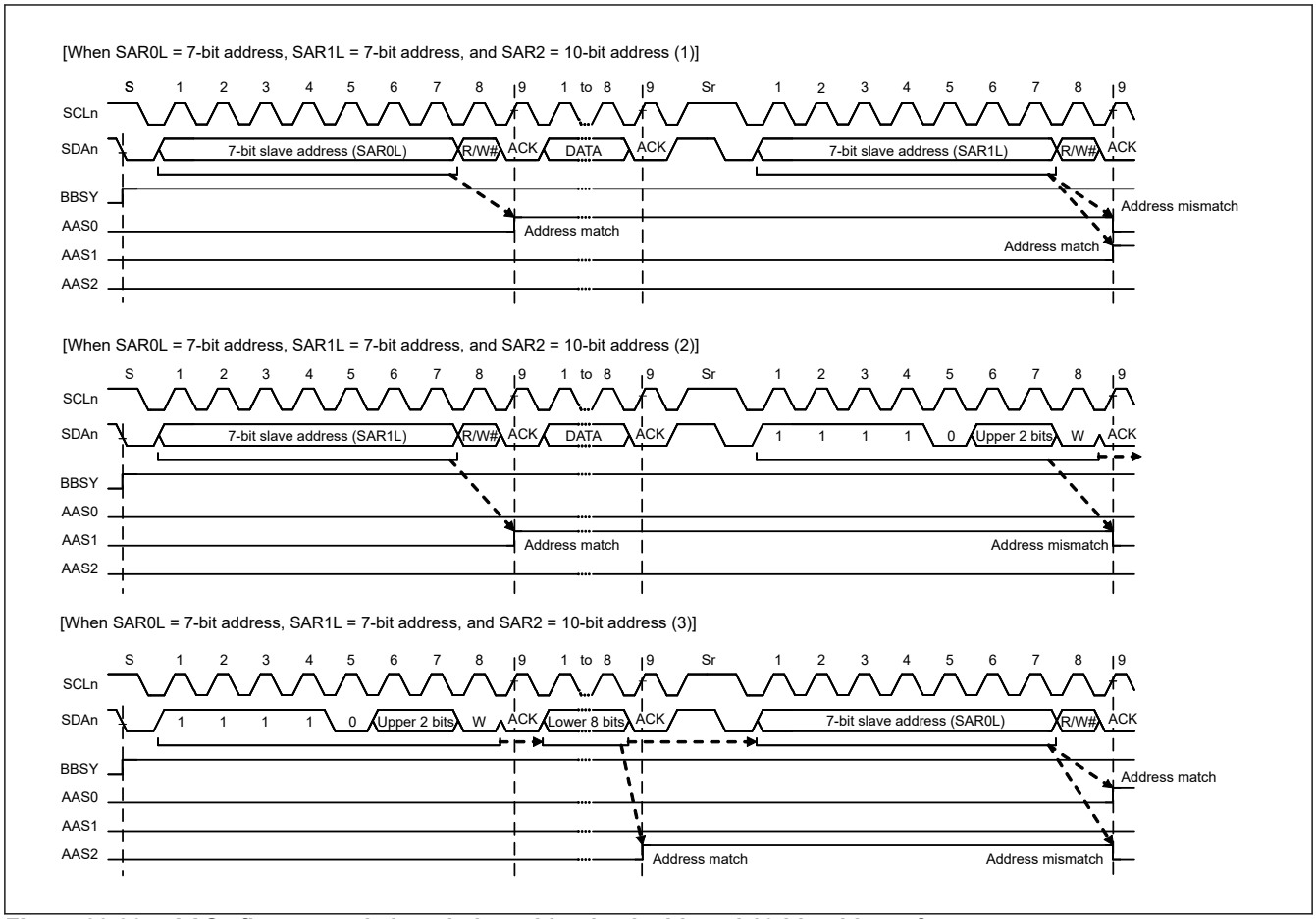



**Figure 28.26 AASy flag set and clear timing with mixed 7-bit and 10-bit address formats**

## 28.7.2 Detection of General Call Address

The IIC provides detection of the general call address (0000 000b + 0 [W]). This is enabled by setting the GCAE bit in ICSER to 1.

If the address received after a start or restart condition is issued is 0000 000b + 1[R] (start byte), the IIC recognizes this as the address of a slave device with an all-zero address, but not as the general call address.

When the IIC detects the general call address, both the GCA flag in ICSR1 and the RDRF flag in ICSR2 set to 1 on the rising edge of the ninth cycle of the SCL clock. This leads to the generation of a receive data full interrupt (IICn_RXI). The value of the GCA flag can be checked to confirm that the general call address was transmitted.

Operation after detection of the general call address is the same as normal slave receive operation.

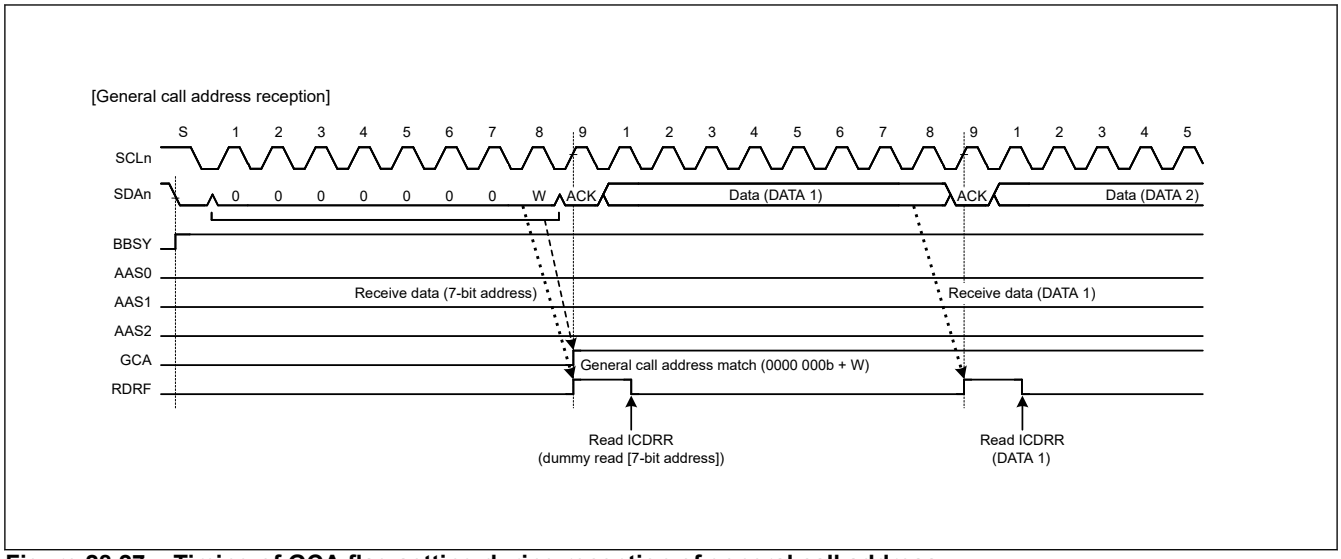



**Figure 28.27 Timing of GCA flag setting during reception of general call address**

## 28.7.3 Device-ID Address Detection

The IIC module provides detection of device-ID address compliant with the $I^2C$ bus specification (revision 03). When the IIC receives 1111 100b as the first byte after a start or restart condition is issued with the DIDE bit in ICSER set to 1, it recognizes the address as a device ID, sets the DID flag in ICSR1 to 1 on the rising edge of the ninth SCL clock cycle when the subsequent R/W# bit is 0, then compares the second and subsequent bytes with its own slave address. If the address matches the value in the slave address register, the IIC sets the associated AASy flag (y = 0 to 2) in ICSR1 to 1.

When the first byte received after the issue of a start or restart condition matches the device ID address (1111 100b) again and the subsequent R/W# bit is 1, the IIC does not compare the second and subsequent bytes and sets the ICSR2.TDRE flag to 1.

In the device ID address detection function, the IIC sets the DID flag to 0 if a match with the IIC slave address is not obtained or a match with the device ID address is not obtained after a match with the IIC slave address and a restart condition is not detected. If the first byte after detection of a start or restart condition matches the device ID address (1111 100b), and the R/W# bit is 0, the IIC sets the DID flag to 1 and compares the second and subsequent bytes with the slave address of the IIC. If the R/W# bit is 1, the DID flag holds the previous value and the IIC does not compare the second and subsequent bytes. Therefore, the reception of a device ID address can be checked by reading the DID flag after confirming that TDRE = 1.

Additionally, prepare the device ID fields (3 bytes: 12 bits indicating the manufacturer + 9 bits identifying the part + 3 bits indicating the revision) that must be sent to the host after reception of a continuous device-ID field as normal transmit data. For details on the information that must be included in device ID fields, contact NXP Semiconductors.

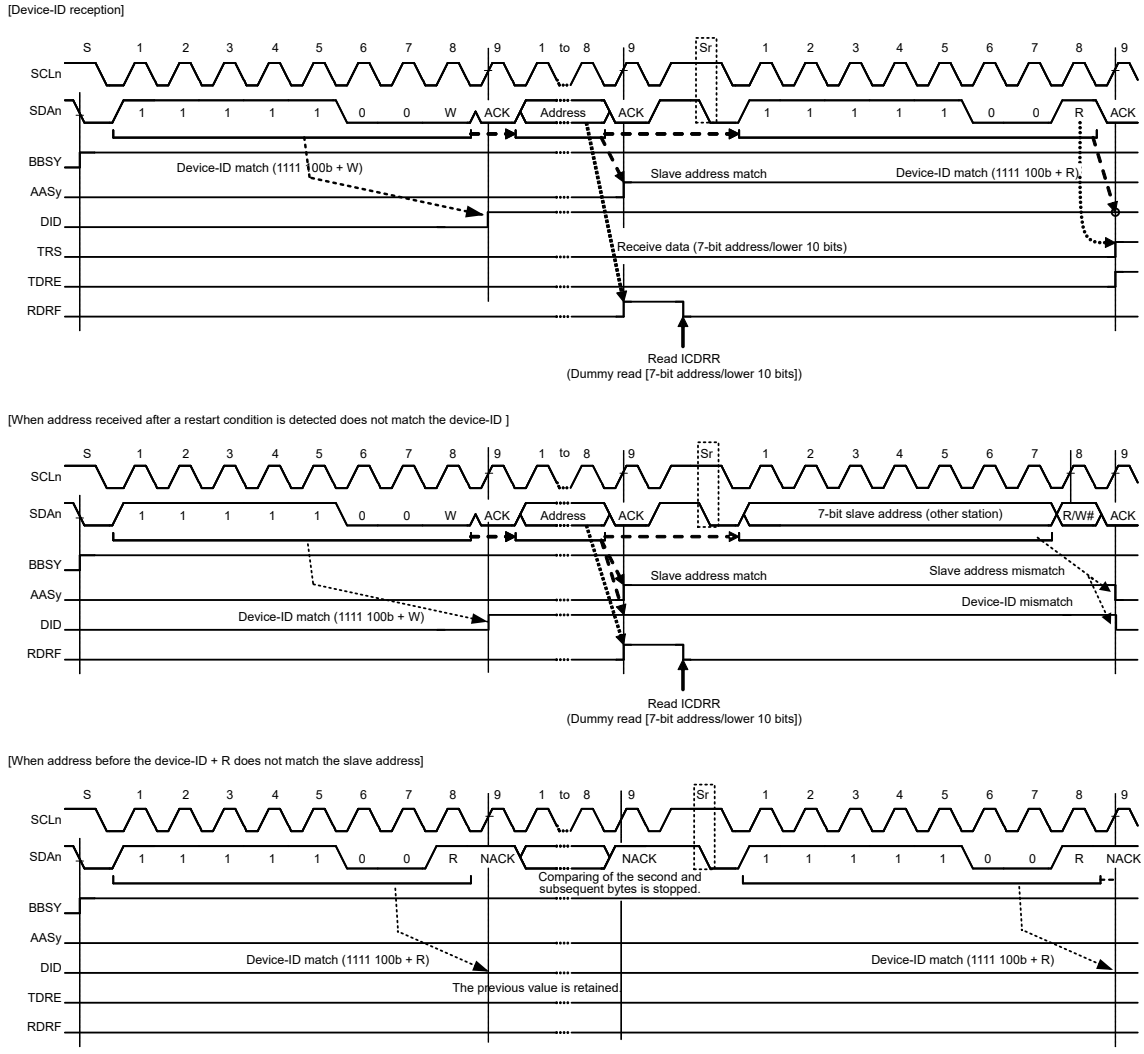



**Figure 28.28 AASy and DID flag set and clear timing during reception of device ID**

## 28.7.4 Host Address Detection

The IIC provides host address detection when operating in SMBus. When the HOAE bit in ICSER is set to 1 while the SMBS bit in ICMR3 is 1, the IIC can detect the host address (0001 000b) in slave receive mode (MST and TRS bits = 00b in ICCR2).

When the IIC detects the host address, the HOA flag in ICSR1 is set to 1 on the rising edge of the 9th SCL clock cycle, and at the same time, the RDRF flag in ICSR2 is set to 1 when the R/W# bit is 0 (Wr bit). This causes a receive data full interrupt (IICn_RXI) to be generated. The HOA flag indicates that the host address was sent from another device.

If the bit following the host address (0001 000b) is an Rd bit (R/W# bit = 1), the IIC can also detect the host address. After the host address is detected, the IIC operates in the same manner as in normal slave operation.

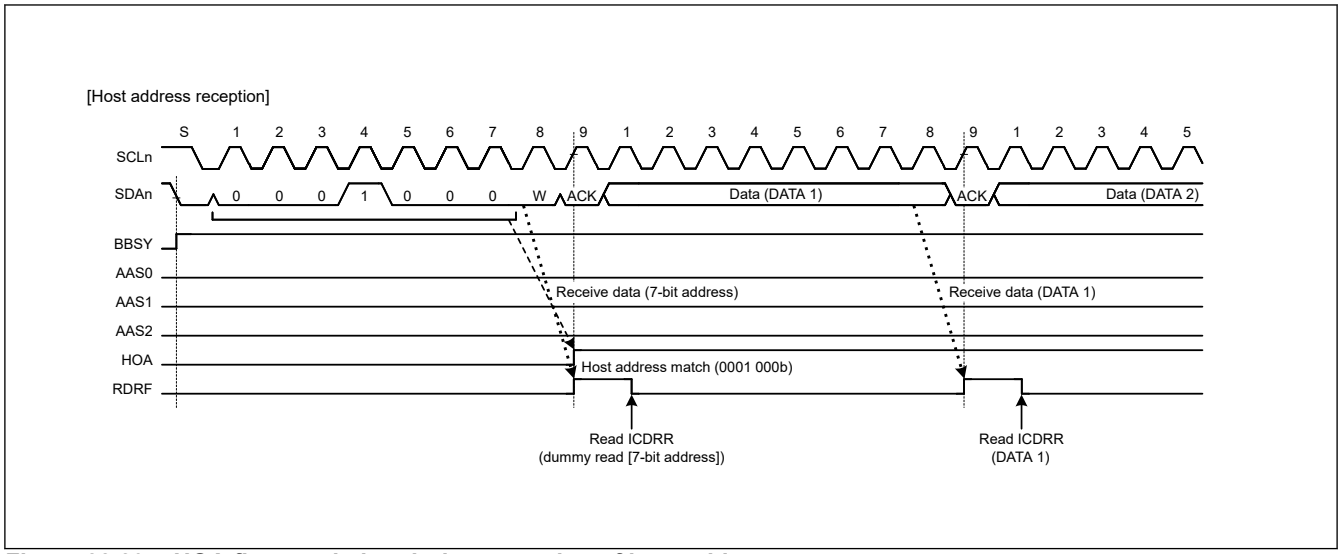



**Figure 28.29 HOA flag set timing during reception of host address**

## 28.8 Wakeup Function

The IIC provides a wakeup function that causes the MCU to transition from Software Standby mode or Snooze mode to normal operation. The wakeup function enables the reception of data when the system clock (PCLKB) is stopped, and generates a wakeup interrupt signal on a match of the slave address of the received data. This wakeup interrupt signal triggers the return to normal operation. After the wakeup interrupt occurs, switch the IIC to PCLKB synchronous operation so that communication can continue.

The wakeup function has four operation modes:

- Normal wakeup mode 1
- Normal wakeup mode 2
- Command recovery mode
- EEP response mode

Table 28.9 describes the behavior in these modes.

**Table 28.9 Wakeup operation modes**

| Operation mode | ACK response timing | ACK response before wakeup to PCLKB synchronous operation | SCL state during wakeup to PCLKB synchronous operation |
|---|---|---|---|
| Normal wakeup mode 1 | Before wakeup to PCLKB synchronous operation*1 | ACK | Fixed low |
| Normal wakeup mode 2 | After wakeup to PCLKB synchronous operation*2 | Before wakeup: no response (NACK level retained)<br>After wakeup: ACK response | Fixed low |
| Command recovery mode | Before wakeup to PCLKB synchronous operation*1 | ACK | Open |
| EEP response mode | Before recovery to PCLKB synchronous operation*1 | NACK | Open |

Note 1. Switching timing from PCLKB asynchronous operation to PCLKB synchronous operation is the falling edge of the 9th clock of the SCL.
Note 2. Switching timing from PCLKB asynchronous operation to PCLKB synchronous operation is the falling edge of the 8th clock of the SCL.

The following can be selected as wakeup interrupt sources:

- Host address detection (valid when ICSER.HOAE = 1)
- General call address detection (valid when ICSER.GCAE = 1)
- Slave address 0*1 detection (valid when ICSER.SAR0E = 1)

- Slave address 1[*1] detection (valid when ICSER.SAR1E = 1)
- Slave address 2[*1] detection (valid when ICSER.SAR2E = 1)

Note 1. Only 7-bit address can be set. Set the FS bit in SARUy (y = 0 to 2) to 0.

### Precautions on the use of the wakeup function

- Do not change the content of the IIC registers except the WUSEN bit in ICWUR2 while the WUASYF flag in ICWUR2 is 1 (during PCLKB asynchronous operation).
- Set ICWUR.WUE and ICWUR.WUIE to 1, and ICCR2.MST and ICCR2.TRS to 0 (slave reception mode) before switching to PCLKB asynchronous mode.
- The device ID and the 10-bit slave address cannot be selected for the wakeup interrupt source. Set the DIDE bit in ICSER and FS bit in SARUy (y = 0 to 2) to 0.
- Set bits TIE, TEIE, RIE, NAKIE, SPIE, STIE, ALIE, and TMOIE in the ICIER register to 0 (interrupt disabled) before switching to the asynchronous operation.
- When the wakeup function is enabled, do not use the timeout function (ICWUR.WUE = 1)
- Even when a wakeup interrupt is generated during PCLKB asynchronous operation (when ICWUR2.WUASYF = 1), if the slave addresses match in PCLKB synchronous mode (ICWUR2.WUASYF = 0), the wakeup interrupt does not occur and the WUF flag is not set.
- If the timing of writing 0 to the ICWUR2.WUSEN bit and the timing of detecting a start condition conflict, the IIC might start the next reception in PCLKB synchronous operation mode. In this case, ICWUR2.WUASYF flag becomes 1 (switch to PCLKB asynchronous mode) when data communication is complete, a stop condition is detected, and detection of a wakeup event starts.
- After writing 0 to the WUSEN bit in ICWUR2, do not change registers relate to the IIC operation mode setting (ICMR3, ICSER, and SARLy) until the mode is switched to PCLKB asynchronous operation from PCLKB synchronous operation (while the ICWUR2.WUASYF flag is 1). If the register value changes during this period by an interrupt handling or another factor, the IIC might malfunction before switching to the asynchronous operation.

## 28.8.1 Normal Wakeup Mode 1

This section describes the behavior, the timing, and an example operation of normal wakeup mode 1.

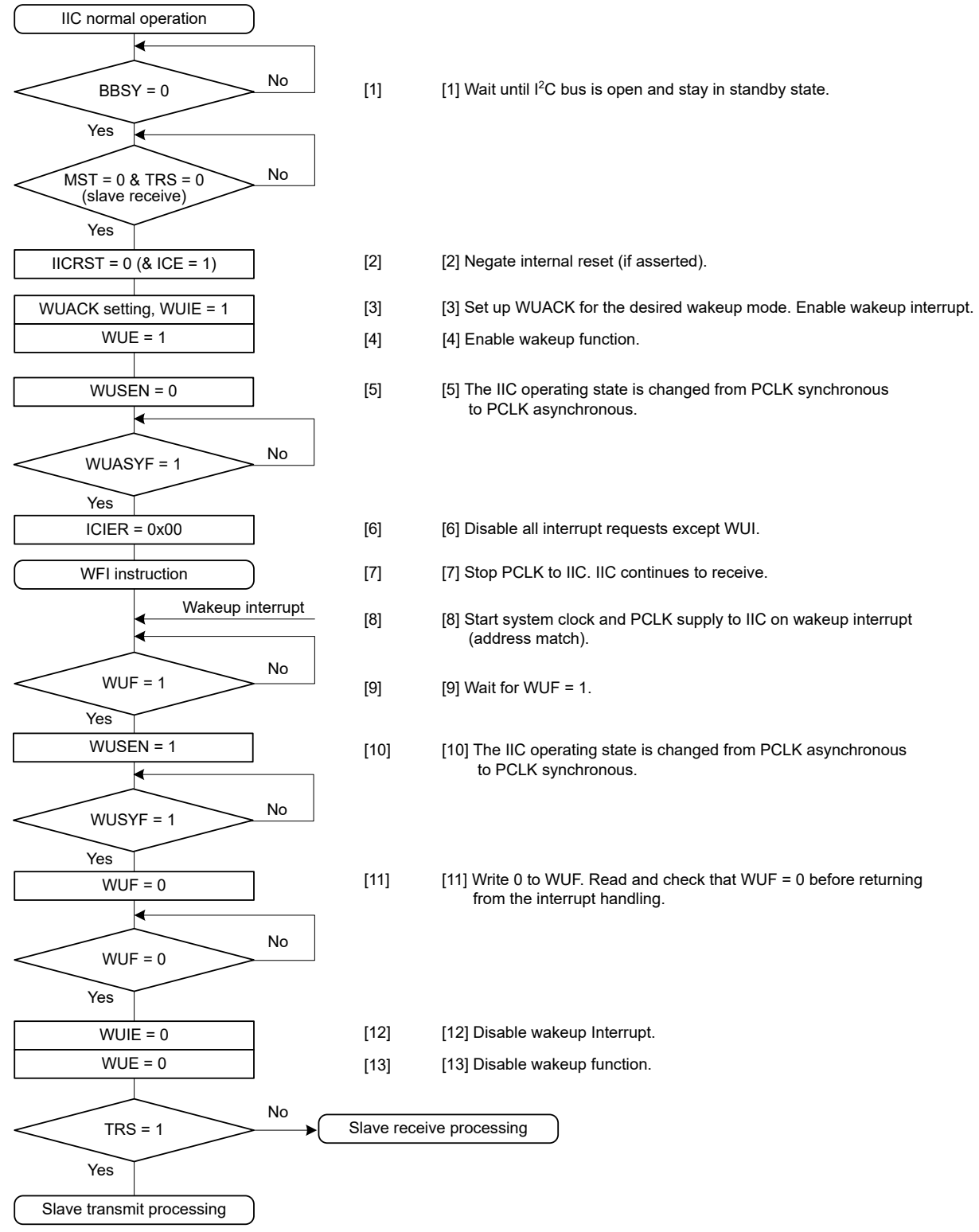
In normal wakeup mode 1, a wakeup interrupt triggered by the match of the slave address initiates the transition to normal operation as follows:

| | |
|---|---|
| Before wakeup: | ACK is sent in response to the data received with its own slave address of the IIC. |
| During wakeup: | ACK response is made on the 9th clock cycle of SCL, after which SCL is held low[*1]. |
| After wakeup: | Normal operation continues. |

Note 1. Between the 9th clock cycle and 1st clock cycle during wakeup, WAIT = 1 is invalid.

If the slave address does not match, the SCL line is not held low after the 9th clock cycle of SCL, and the slave operation continues. Figure 28.30 shows an operation example, and Figure 28.32 shows the detailed timing.

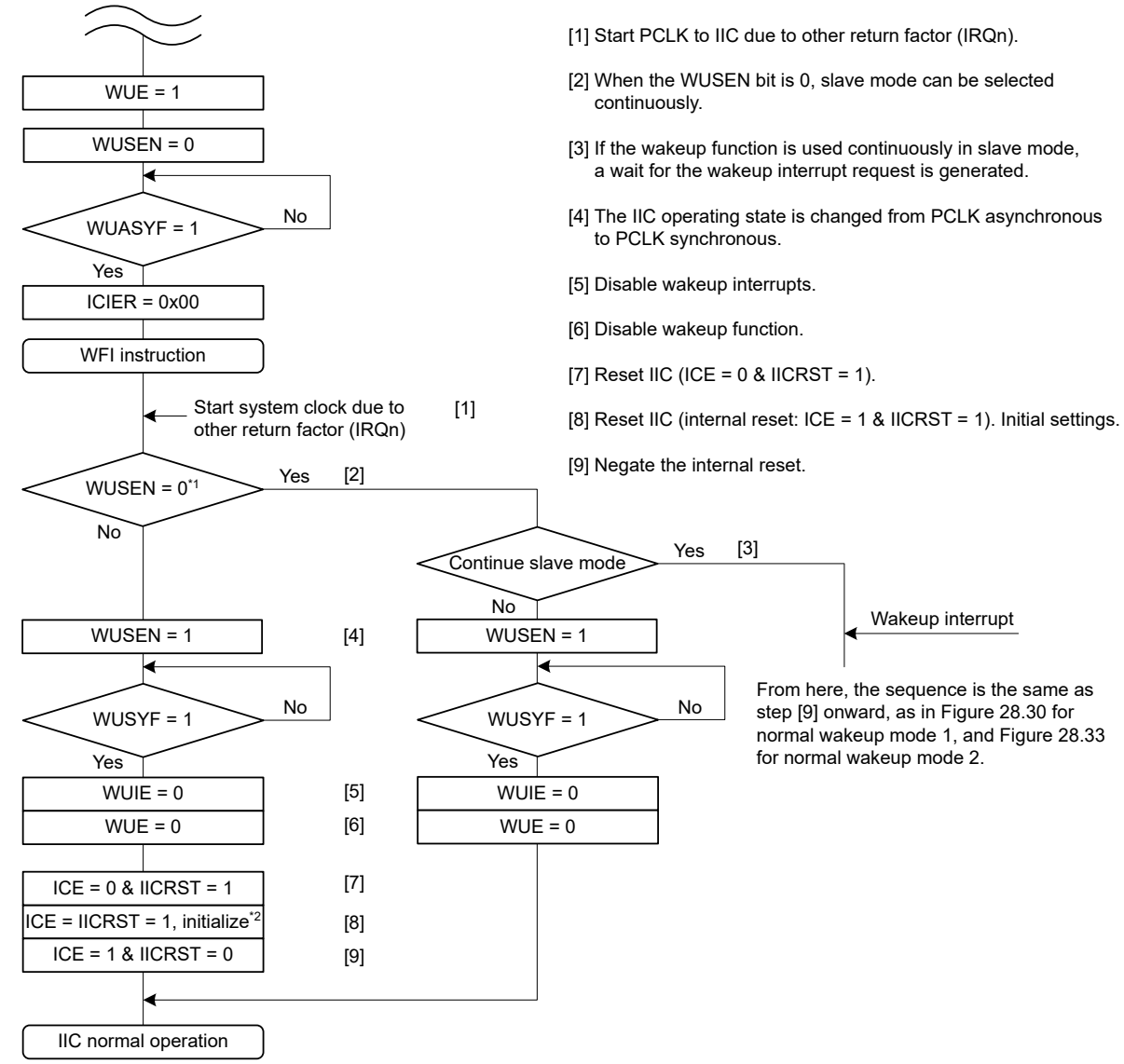
If the transition from Software Standby mode or Snooze mode is triggered by an interrupt other than a wakeup interrupt, for example the IRQn, the WUF flag is not set to 1. Figure 28.31 shows an operation example.




Note: PCLK = PCLKB

**Figure 28.30 Example operation of normal wakeup mode 1 when wakeup is triggered by a wakeup interrupt on match of the slave address**

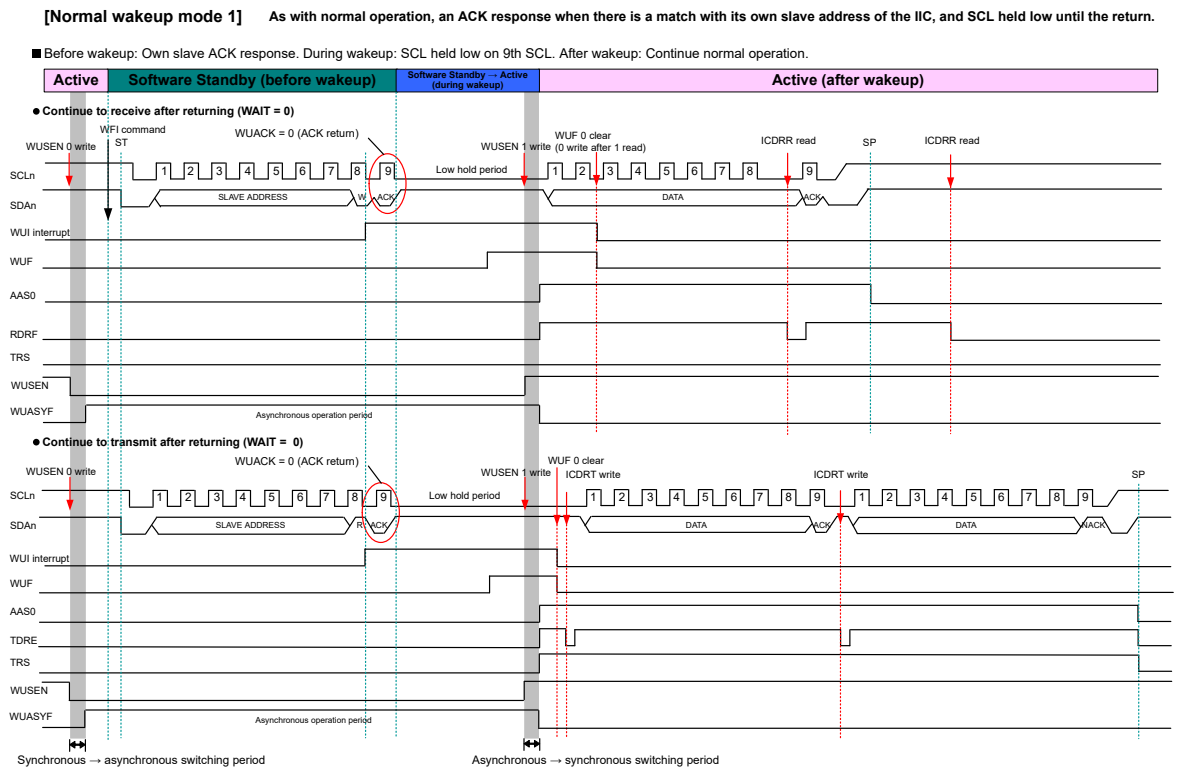
Note: See Precautions on the use of the wakeup function.




Note: PCLK = PCLKB
Note 1. In the MCU, WUSEN bit is always 0 in the IIC.
Note 2. For details on the IIC initial settings, see section 28.3.2. Initial Settings.

**Figure 28.31 Example operation of normal wakeup modes 1 and 2 when wakeup is triggered by an interrupt other than IIC wakeup interrupt, for example, the IRQn**

Note: For details on the IIC initial settings, see section 28.3.2. Initial Settings.




Note: n = 0

**Figure 28.32 Timing of normal wakeup mode 1**

## 28.8.2 Normal Wakeup Mode 2

This section describes the behavior, the timing, and an example operation of normal wakeup mode 2.

In normal wakeup mode 2, a wakeup interrupt triggered by a match of the slave address initiates the transition to normal operation as follows:

| | |
|---|---|
| Before wakeup: | No response to data received with its own slave address until the end of the 8th SCL cycle. |
| During wakeup: | SCL line held low during the 8th and 9th clock cycles. |
| After wakeup: | ACK returns on the 9th clock cycle of SCL, and normal operation continues. |

If the slave address does not match, the SCL line is not held low after the 8 th SCL clock cycle, and the slave operation continues. Figure 28.33 shows an example operation, and Figure 28.34 shows the detailed timing.

If the transition from Software Standby mode or Snooze mode is triggered by an interrupt other than a wakeup interrupt, such as the IRQn, for example, the WUF flag is not set to 1. Figure 28.31 shows an operation example.

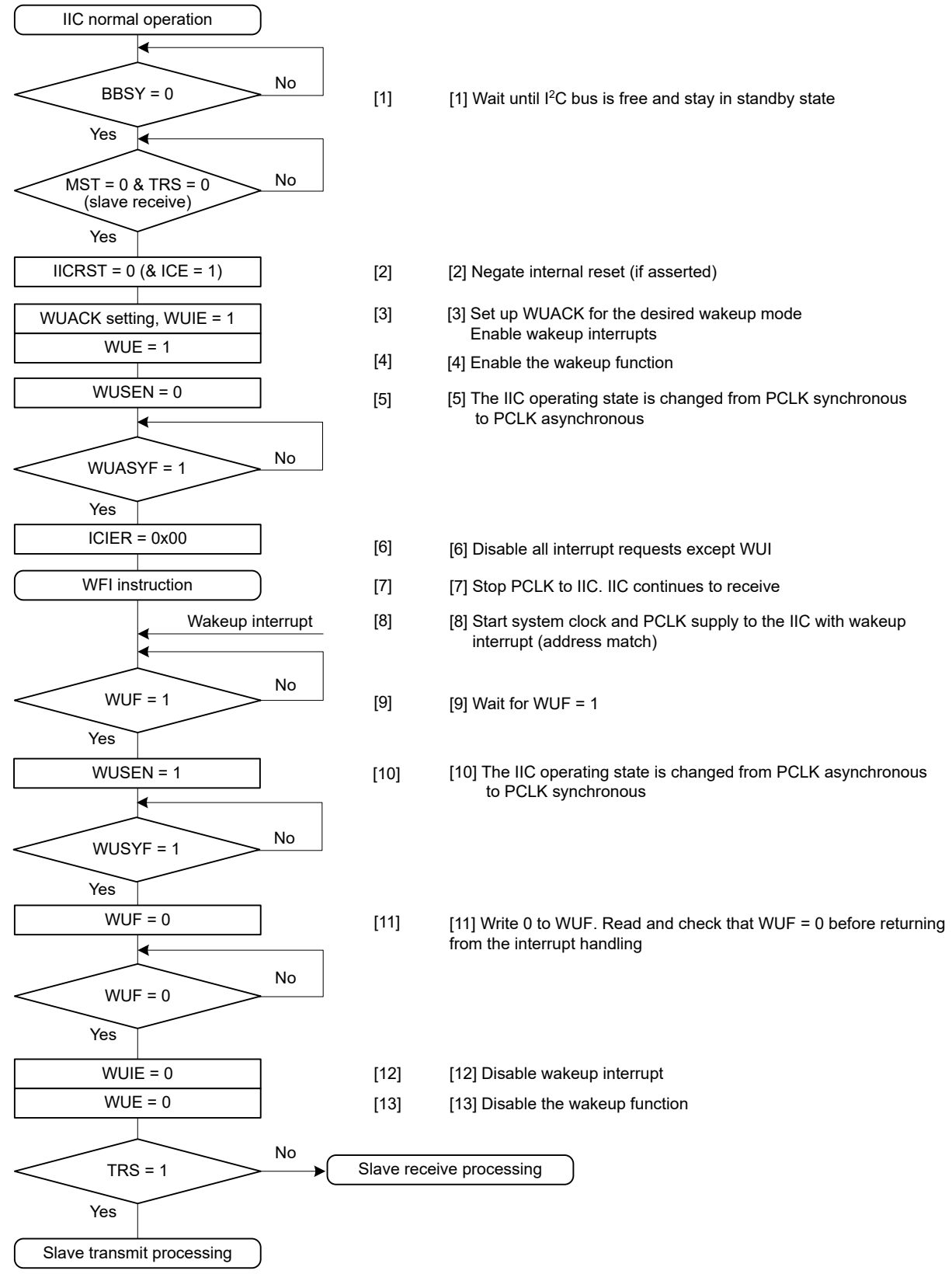
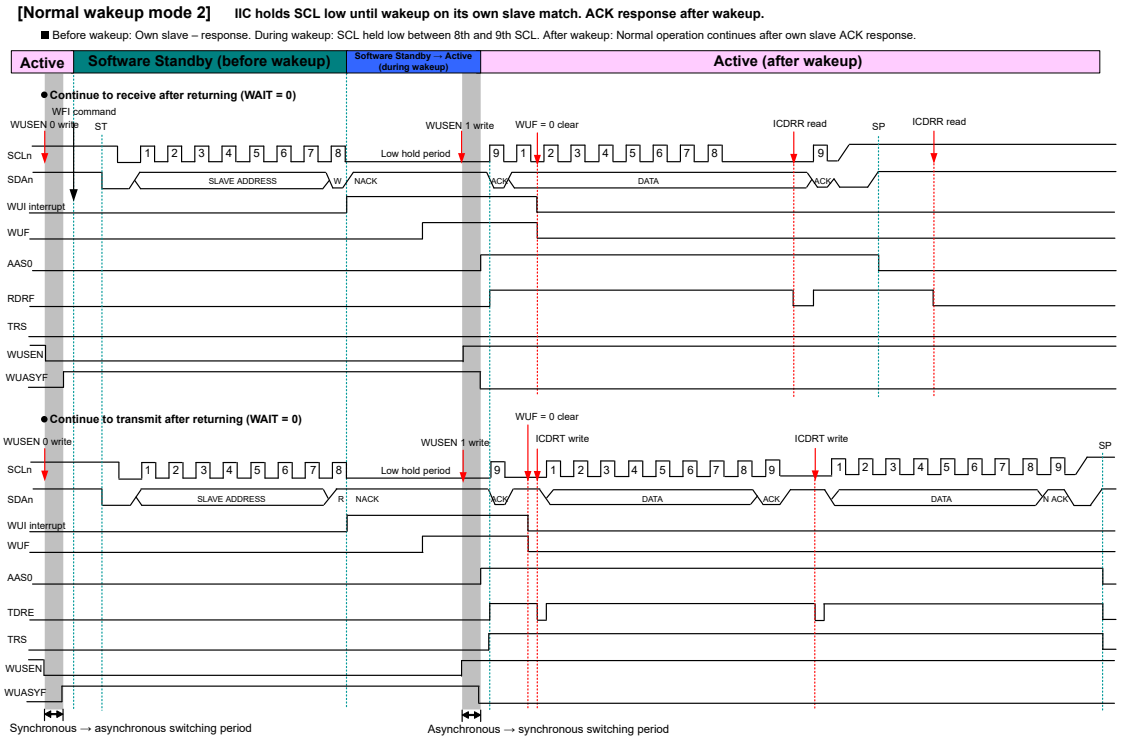



**Figure 28.33 Example operation of normal wakeup mode 2 when wakeup is triggered by a wakeup interrupt on match of the slave address**

Note: See Precautions on the use of the wakeup function.




Note: n = 0

**Figure 28.34 Timing of normal wakeup mode 2**

## 28.8.3 Command Recovery Mode and EEP Response Mode (Special Wakeup Modes)

This section describes the behavior, the timing, and example operations of the command recovery and EEP response modes.

In the command recovery and EEP response modes, the SCL line is not held low during the wakeup period (after the rise of the 9th clock cycle of SCL). Therefore, other IIC devices can use the $I^2C$ bus during this period.

A wakeup interrupt triggered by a match of the slave address initiates the transition to normal operation as follows:

| | |
|---|---|
| Before wakeup: | In response to the data received with its own slave address, the IIC returns ACK (command recovery mode) or NACK (EEP response mode). |
| During wakeup: | The SCL line is not held low. |
| After wakeup: | Normal operation continues after IIC initialization. |

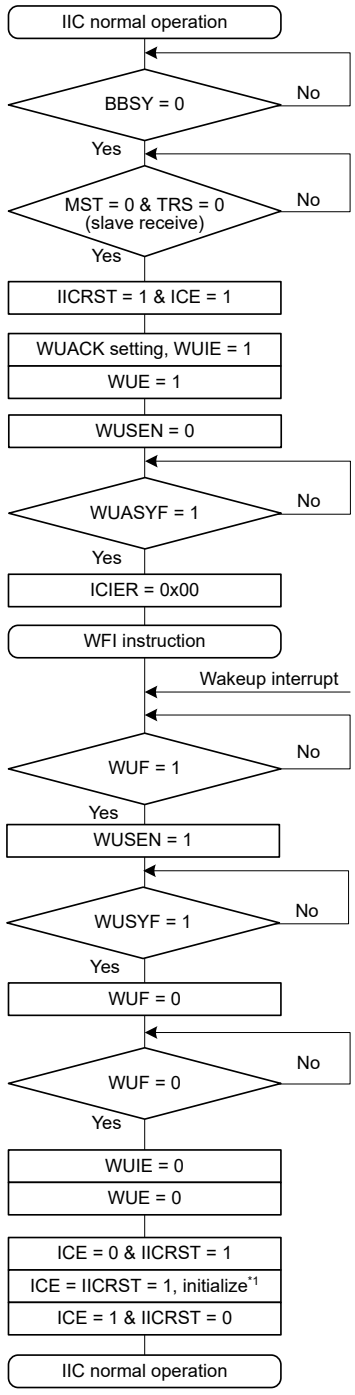
If the slave address does not match, the slave operation continues.

Note: Because the SCL line is not held low during wakeup, transmission or reception of the data that follows the slave address is not possible.

Note: The command recovery and EEP response modes are internal reset states (ICE = IICRST = 1). Therefore, the match of the slave address does not set the flags, HOA, GCA, ASS0, ASS1, and ASS2 in the ICSR1 register.

Figure 28.35 shows an example operation in recovery and EEP response modes. Figure 28.37 shows the detailed timing.

If the transition from Software Standby mode or Snooze mode is triggered by an interrupt other than a wakeup interrupt, such as IRQn for example, the WUF flag is not set to 1. Follow the processing shown in Figure 28.36.




[1] [1] Wait until $I^2C$ bus is open and stay in standby state.

[2] [2] Internal reset is asserted.

[3] [3] Set up WUACK for the desired wakeup mode. Enable wakeup interrupt.

[4] [4] Enable wakeup function.

[5] [5] The IIC operating state is changed from PCLK synchronous to PCLK asynchronous.

[6] [6] Disable all interrupt requests except WUI.

[7] [7] Stop PCLK to IIC. IIC continues to receive.

[8] [8] Start system clock and PCLK supply to the IIC with wakeup interrupt (address match).

[9] [9] Wait for WUF = 1.

[10] [10] The IIC operating state is changed from PCLK asynchronous to PCLK synchronous.

[11] [11] Write 0 to WUF. Read and check that WUF = 0 before returning from the interrupt handling.

[12] [12] Disable wakeup interrupt.

[13] [13] Disable wakeup function.

[14] [14] Reset IIC (ICE = 0 & IICRST = 1).

[15] [15] Reset IIC (internal reset: ICE = 1 & IICRST = 1). Initial settings.

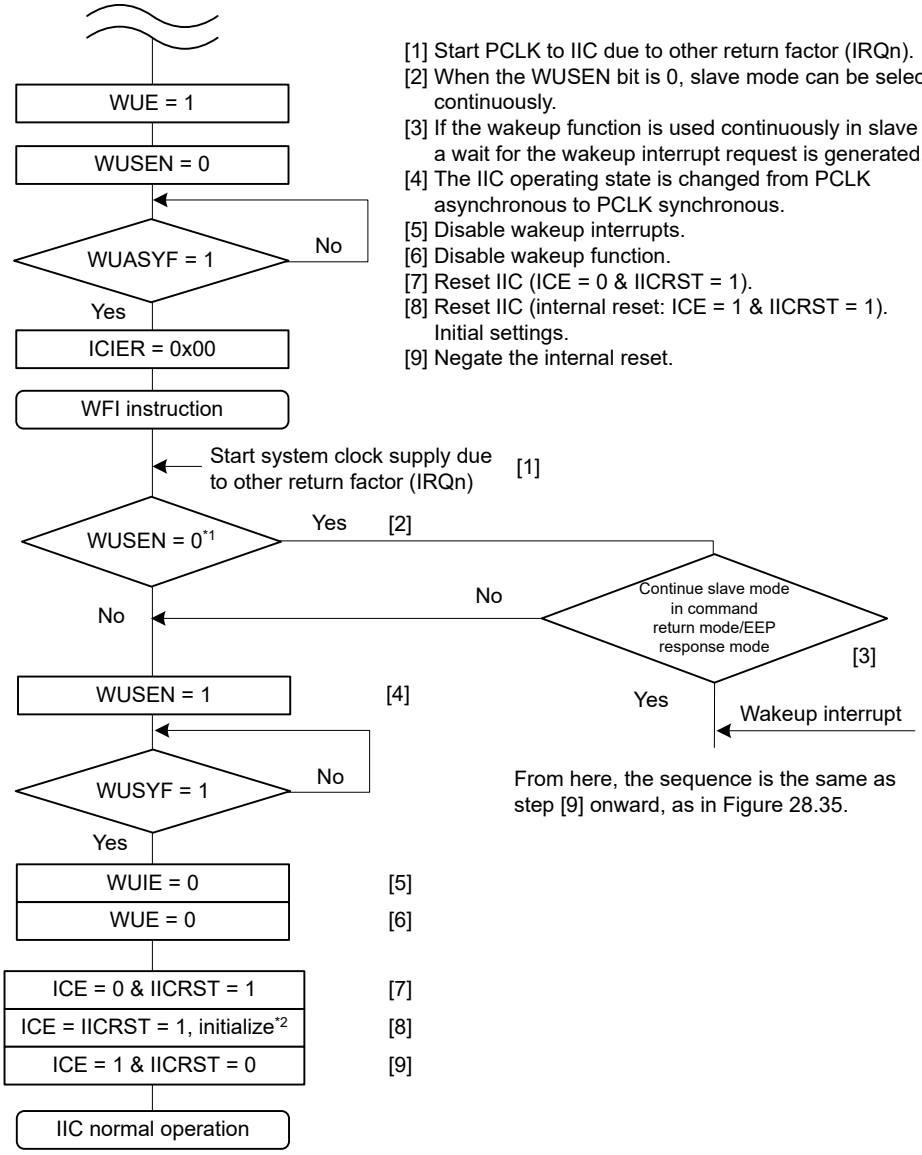
[16] [16] Negate the internal reset.

Note: PCLK = PCLKB

Note 1. For details on IIC initialization, see section 28.3.2. Initial Settings.

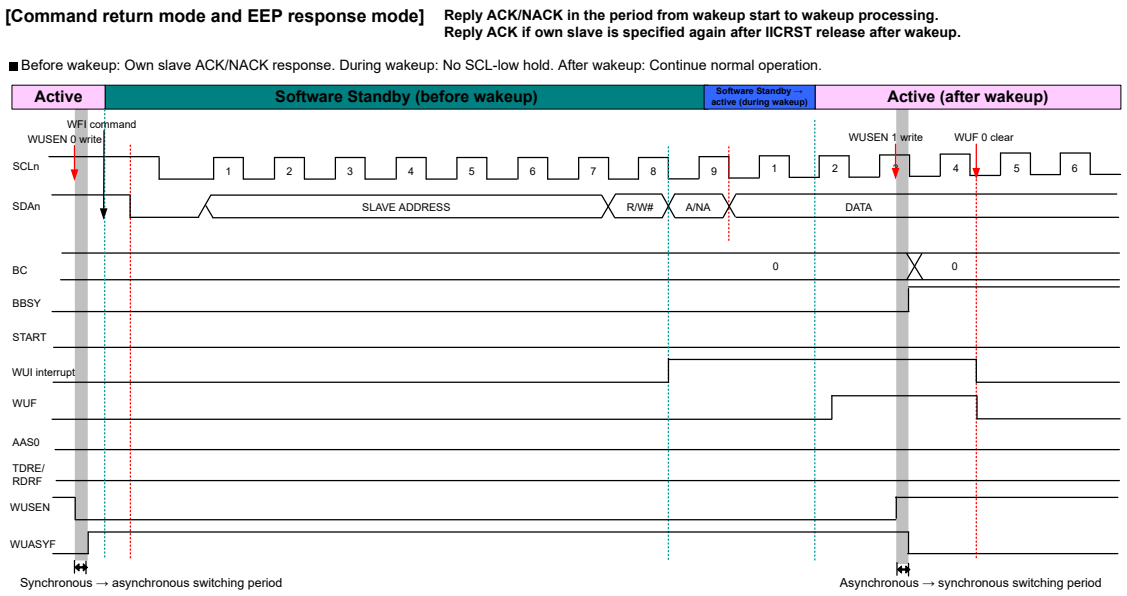
**Figure 28.35 Example operation of command recovery mode and EEP response mode when wakeup is triggered by a wakeup interrupt on match of the slave address**

Note: See Precautions on the use of the wakeup function.




Note: PCLK = PCLKB
Note 1. In the MCU, WUSEN bit is always 0 in the IIC.
Note 2. For details on IIC initialization, see section 28.3.2. Initial Settings.

**Figure 28.36 Example operation of command recovery and EEP response modes when wakeup is triggered by an interrupt other than IIC wakeup interrupt, for example, the IRQn**




Note: n = 0

**Figure 28.37 Timing of command recovery and EEP response modes**

## 28.9 Automatic Low-Hold Function for SCL

### 28.9.1 Function to Prevent Wrong Transmission of Transmit Data

If the $I^2C$ Bus Shift Register (ICDRS) is empty and data has not been written to the $I^2C$ Bus Transmit Data Register (ICDRT) with the IIC in transmission mode (TRS bit = 1 in ICCR2), the SCLn line is automatically held low over the subsequent intervals. This low-hold period is extended until the transmit data is written, which prevents the unintended transmission of erroneous data.

Master transmit mode:

- Low-level interval after a start or restart condition is issued
- Low-level interval between the 9th clock cycle of one transfer and the 1st clock cycle of the next.

Slave transmit mode:

- Low-level interval between the 9th clock cycle of one transfer and the 1st clock cycle of the next.

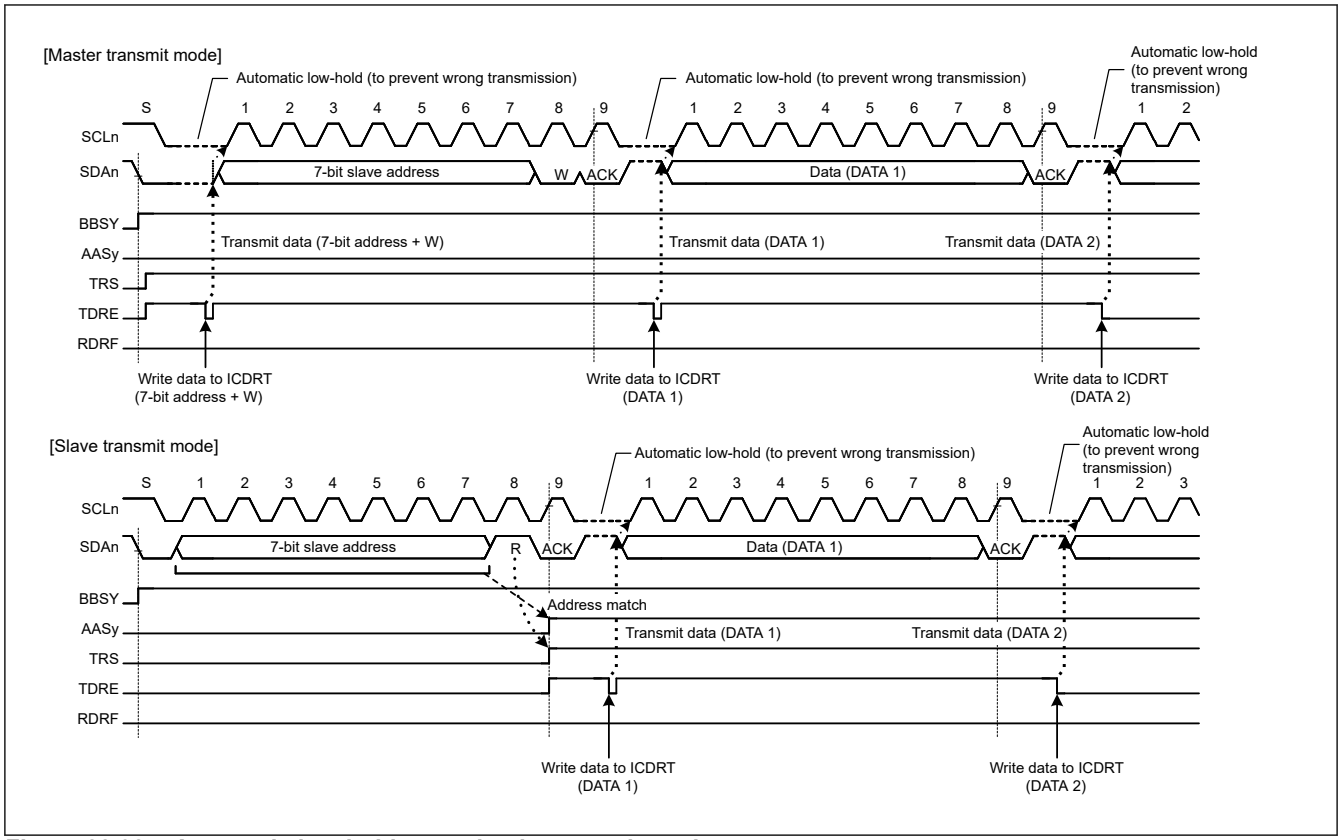



**Figure 28.38 Automatic low-hold operation in transmit mode**

## 28.9.2 NACK Reception Transfer Suspension Function

This function suspends transfer operation when NACK is received in transmit mode (TRS bit = 1 in ICCR2). This function is enabled when the NACKE bit in ICFER is set to 1. If the next transmit data is already written (TDRE flag = 0 in ICSR2) when NACK is received, the next data transmission on the falling edge of the 9th SCL clock cycle is automatically suspended. This prevents the SDAn line output level from being held low when the MSB of the next transmit data is 0.

If the transfer operation is suspended by this function (NACKF flag = 1 in ICSR2), transmit and receive operations are discontinued. To restore transmit or receive operation, after issuing the restart condition, you need to set the NACKF flag to 0 and try again, or set the NACKF flag to 0 after issuing the stop condition and then start again from issuing the start condition.

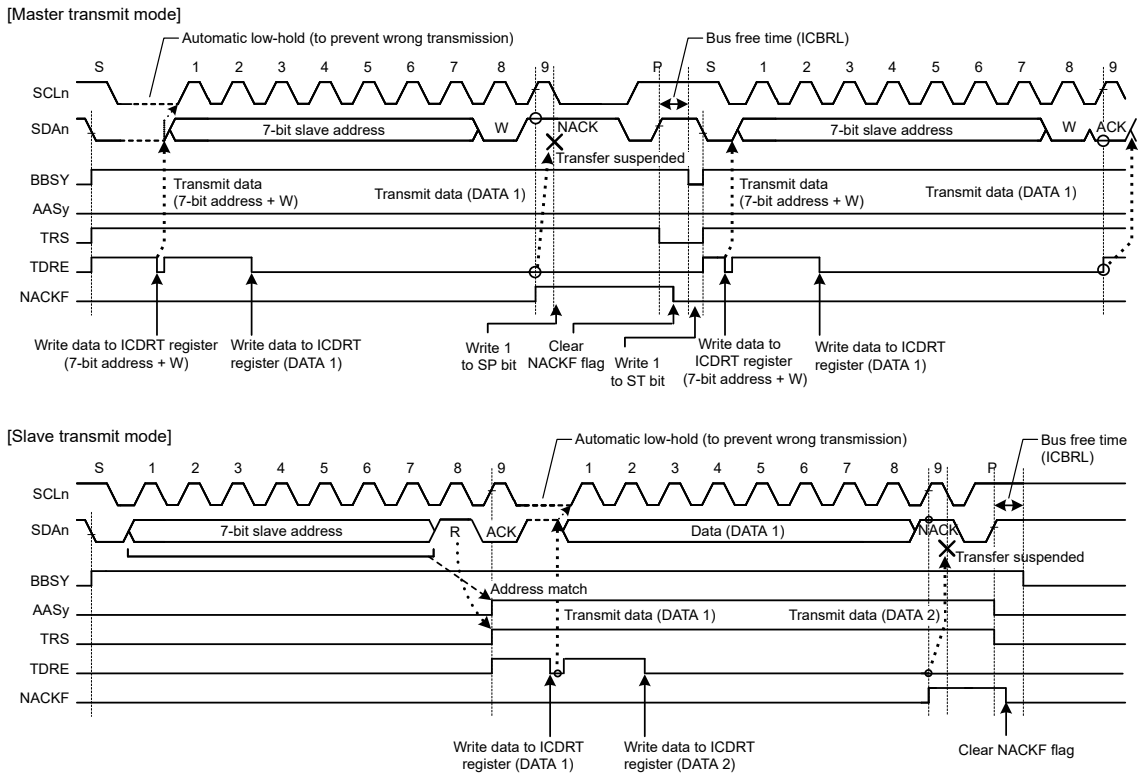



**Figure 28.39 Suspension of data transfer when NACK is received, when NACKE = 1**

## 28.9.3 Function to Prevent Failure to Receive Data

If response processing is delayed when receive data (ICDRR) read is delayed for a period of one transfer frame or more with receive data full (RDRF flag = 1 in ICSR2) in receive mode (TRS = 0 in ICCR2), the IIC holds the SCLn line low automatically immediately before the next data is received to prevent a failure to receive data.

This function is enabled even if the read processing of the final receive data is delayed and, in the meantime, the IIC slave address is designated after a stop condition is issued. This function does not interfere with other communication because the IIC does not hold the SCLn line low when a mismatch with its own slave address occurs after a stop condition is issued.

Periods in which the SCLn line is held low can be selected with a combination of the WAIT and RDRFS bits in ICMR3.

### (1) 1-byte receive operation and automatic low-hold function using the WAIT bit

When the WAIT bit in ICMR3 is set to 1, the IIC performs a 1-byte receive operation using the WAIT bit function. Additionally, when the ICMR3.RDRFS bit is 0, the IIC automatically sends the ACKBT bit value in ICMR3 for the acknowledge bit in the period from the falling edge of the 8th SCL clock cycle to the falling edge of the 9th SCL clock cycle, and automatically holds the SCLn line low on the falling edge of the 9th SCL clock cycle using the WAIT bit function. This low-hold is released by reading data from ICDRR, which enables byte-wise receive operation.

The WAIT bit function is enabled for receive frames after a match with the IIC slave address, including the general call address and host address, is obtained in master or slave receive mode.

### (2) 1-byte receive operation (ACK/NACK transmission control) and automatic low-hold function using the RDRFS bit

When the RDRFS bit in ICMR3 is set to 1, the IIC performs a 1-byte receive operation using the RDRFS bit function. When the RDRFS bit is set to 1, the RDRF flag in ICSR2 is set to 1 (receive data full) on the rising edge of the eighth SCL clock cycle, and the SCLn line is automatically held low on the falling edge of the eighth SCL clock cycle. This low-hold is released by writing a value to the ACKBT bit in ICMR3, but cannot be released by reading data from ICDRR, which enables receive operation through the ACK or NACK transmission control based on the data received in byte units.

The RDRFS bit function is enabled for receive frames after a match with the IIC slave address, including the general call address and host address, is obtained in master or slave receive mode.

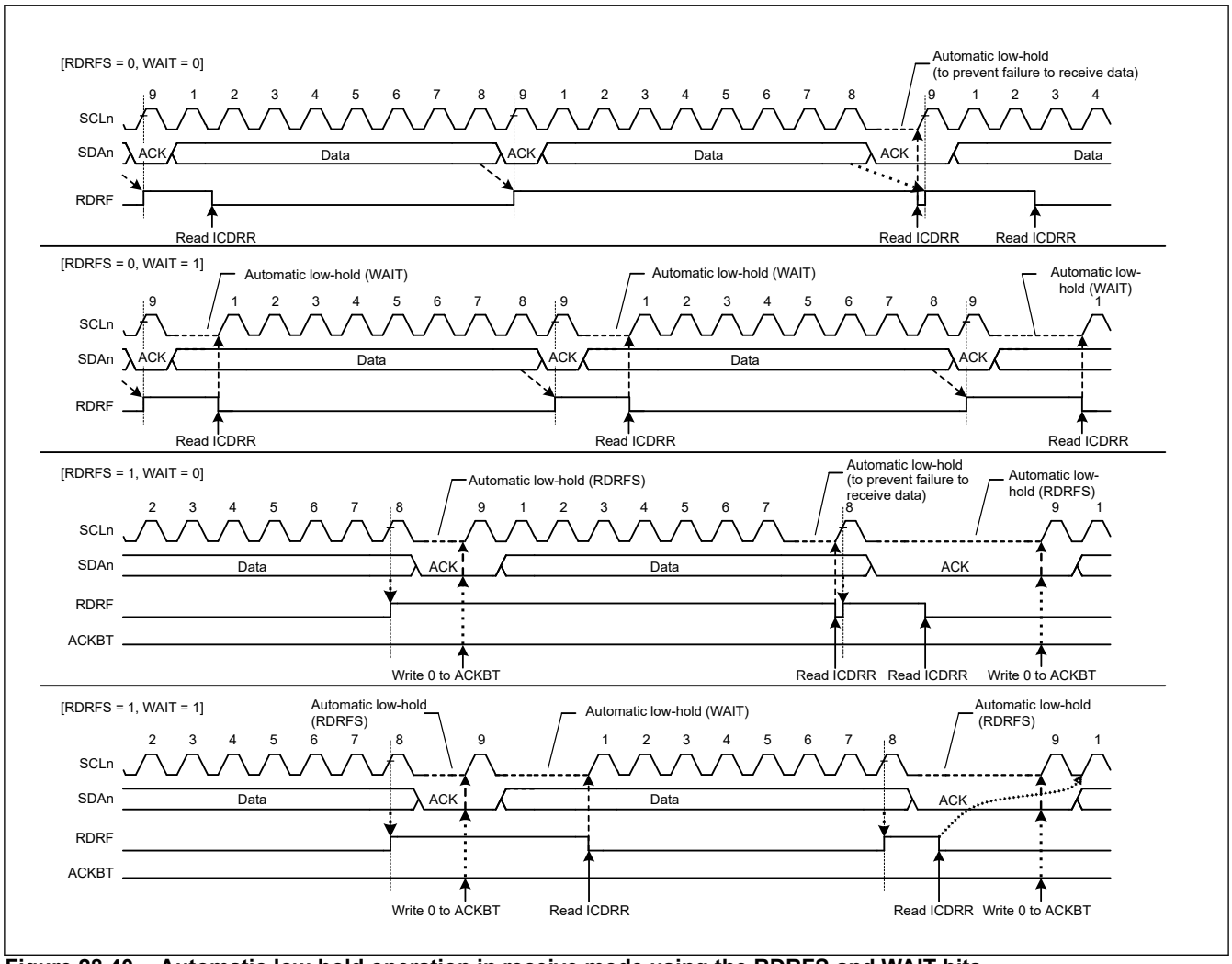



**Figure 28.40 Automatic low-hold operation in receive mode using the RDRFS and WAIT bits**

## 28.10 Arbitration-Lost Detection Functions

In addition to the normal arbitration-lost detection function defined by the $I^2C$ bus standard, the IIC provides functions to prevent double-issue of a start condition, detect arbitration-lost during transmission of NACK, and detect arbitration-lost in slave transmit mode.

### 28.10.1 Master Arbitration-Lost Detection (MALE Bit)

The IIC drives the SDAn line low to issue a start condition. However, if the SDAn line was already driven low by another master device issuing a start condition, the IIC regards its own start condition as an error and considers this a loss in arbitration. Priority is given to transfer by the other master device. Similarly, if a request to issue a start condition is made by setting the ST bit in ICCR2 to 1 while the bus is busy (BBSY flag = 1 in ICCR2), the IIC regards this as a double-issuing-of-start-condition error and considers itself to have lost in arbitration. This prevents a failure of transfer resulting from a start condition being issued while transfer is in progress.

When a start condition is issued successfully, if the transmit data including the address bits (internal SDA output level) and the level on the SDAn line do not match (high output as the internal SDA output, meaning the SDAn pin is in the high-impedance state and a low level is detected on the SDAn line), the IIC loses in arbitration.

After a loss in arbitration of mastership, the IIC immediately enters slave receive mode. If a slave address, including the general call address, matches its own address at this time, the IIC continues in slave operation.

A loss in arbitration of mastership is detected when the following conditions are met while the MALE bit in ICFER is 1 (master arbitration-lost detection enabled).

[Master arbitration-lost conditions]

- Mismatching of the internal level for output on SDA and the level on the SDAn line after a start condition was issued by setting the ST bit in ICCR2 to 1 while the BBSY flag in ICCR2 is set to 0 (erroneous issuing of a start condition)
- Setting of the ST bit in ICCR2 to 1 (start condition double-issue error) while the BBSY flag is 1
- When the transmit data excluding acknowledge (internal SDA output level) does not match the level on the SDAn line in master transmit mode (MST and TRS bits = 11b in ICCR2).

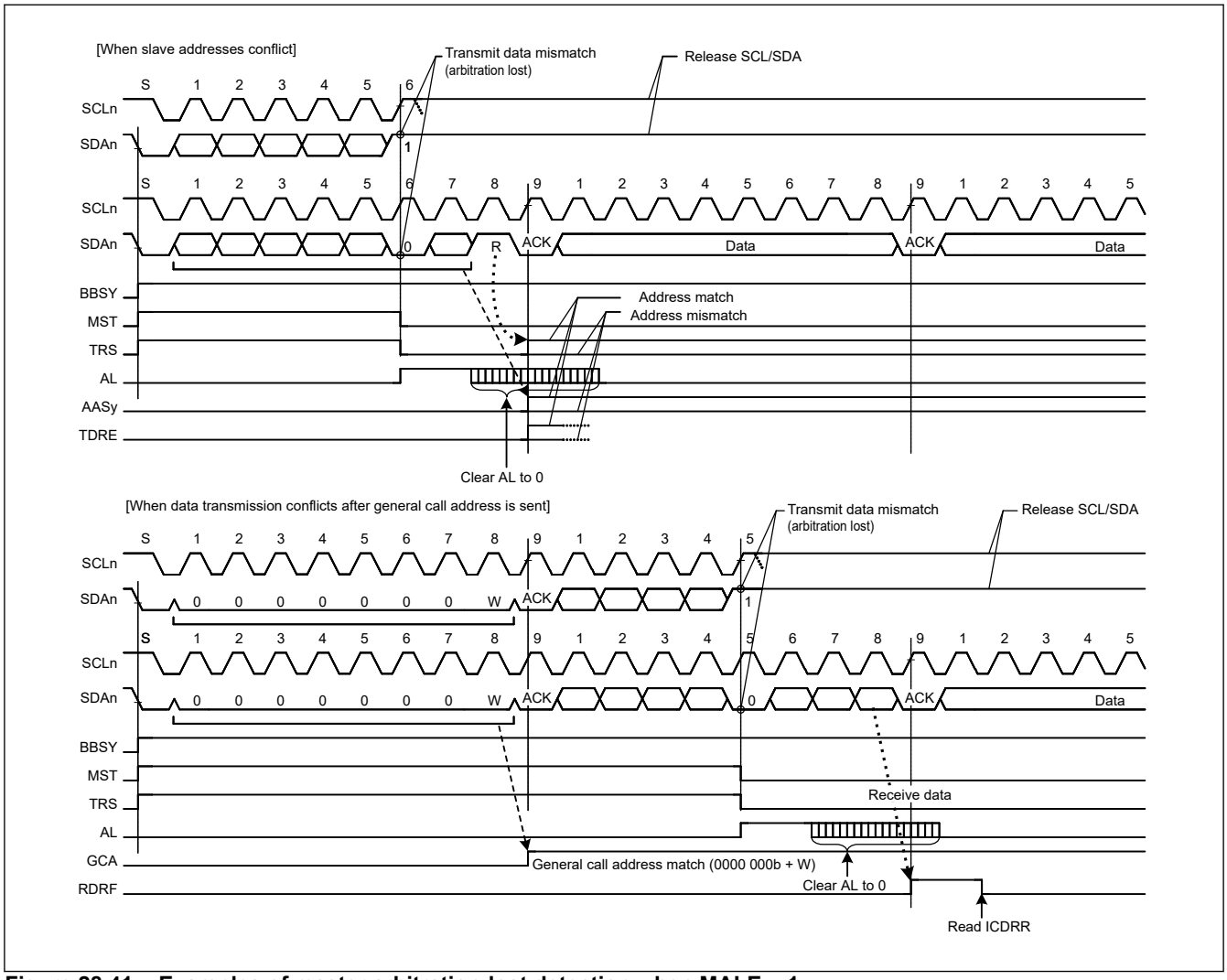



**Figure 28.41 Examples of master arbitration-lost detection when MALE = 1**

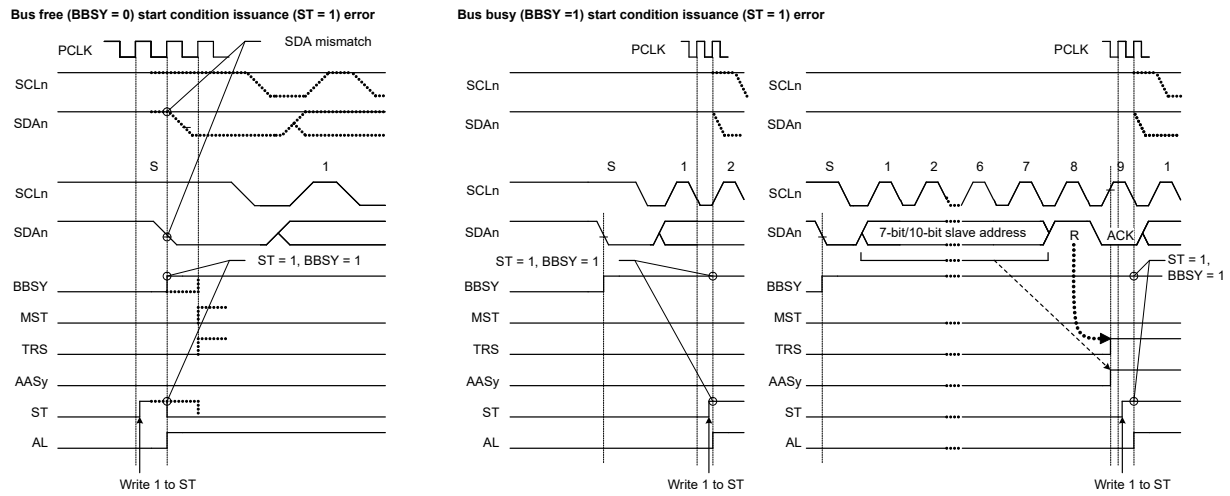



**Figure 28.42 Arbitration-lost when start condition is issued when MALE = 1**

## 28.10.2 Function to Detect Loss of Arbitration during NACK Transmission (NALE Bit)

This function causes arbitration to be lost if the internal SDA output level does not match the level on the SDAn line (high output as the internal SDA output, meaning the SDAn pin is in the high-impedance state) and the low level is detected on the SDAn line during transmission of NACK in receive mode. Arbitration is lost because of a conflict between NACK and ACK transmissions when two or more master devices receive data from the same slave device simultaneously in a multi-master system. Such a conflict occurs when multiple master devices send or receive the same information through a single slave device. Figure 28.43 shows an example of arbitration-lost detection during transmission of NACK.

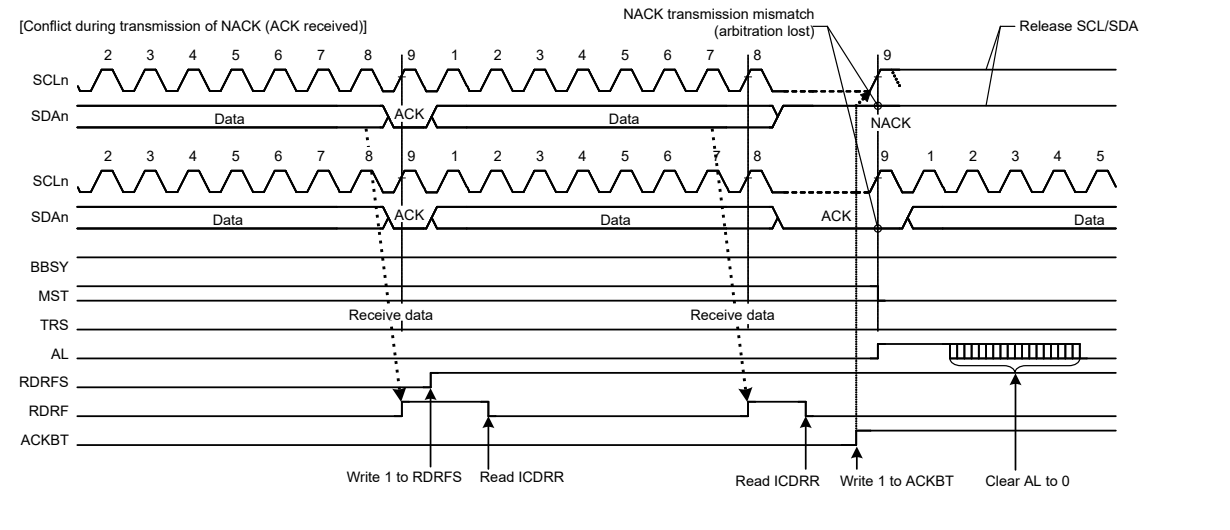



**Figure 28.43 Example of arbitration-lost detection during transmission of NACK when NALE = 1**

The following explains arbitration-lost detection using an example in which two master devices (master A and master B) and a single slave device are connected through the bus. In this example, master A receives 2 bytes of data from the slave device, and master B receives 4 bytes of data from the slave device.

If master A and master B access the slave device simultaneously, because the slave address is identical, arbitration is not lost in either master A or master B during access to the slave device. Therefore, both master A and master B recognize that they have obtained the bus mastership and operate as such. Master A sends NACK when it has received 2 final bytes of data from the slave device. Meanwhile, master B sends ACK because it has not received the required 4 bytes of data. At this time, the NACK transmission from master A and the ACK transmission from master B conflict. In general, if a conflict like

this occurs, master A cannot detect the ACK transmitted by master B and issues a stop condition. The stop condition issue conflicts with the SCL clock output of master B, which disrupts communication.

When the IIC receives ACK during transmission of NACK, it detects a defeat in conflict with other master devices and causes arbitration to be lost. If arbitration is lost during transmission of NACK, the IIC immediately cancels the slave match condition and enters slave receive mode. This prevents a stop condition from being issued, preventing a communication failure on the bus.

Similarly, in the ARP command processing of SMBus, the function to detect loss of arbitration during transmission of NACK is also available for eliminating the extra clock cycle processing, such as 0xFF transmission processing, which is required if the UDID (Unique Device Identifier) of the assigned address does not match in the Get UDID general processing after the Assign Address command.

The IIC detects arbitration-lost during transmission of NACK when the following condition is met with the NALE bit in ICFER set to 1 (arbitration-lost detection during NACK transmission enabled).

[Condition for arbitration-lost during NACK transmission]

- When the internal SDA output level does not match the SDAn line (ACK is received) during transmission of NACK (ACKBT bit = 1 in ICMR3).

## 28.10.3 Slave Arbitration-Lost Detection (SALE Bit)

This function causes arbitration to be lost if the transmit data (internal SDA output level) and the level on the SDAn line do not match (high output as the internal SDA output, meaning the SDAn pin is in the high-impedance state), and the low level is detected on the SDAn line in slave transmit mode. This arbitration-lost detection function is mainly used when transmitting a UDID (Unique Device Identifier) over an SMBus.

When the IIC loses slave arbitration, the IIC is immediately released from the slave-matched state and enters slave receive mode. This function can detect conflicts of data during transmission of UDIDs over an SMBus and eliminates subsequent redundant processing for the transmission of 0xFF.

The IIC detects slave arbitration-lost when the following condition is met with the SALE bit in ICFER set to 1 (slave arbitration-lost detection enabled).

[Condition for slave arbitration-lost]

- When transmit data excluding acknowledge (internal SDA output level) does not match the SDAn line in slave transmit mode (MST and TRS bits = 01b in ICCR2).

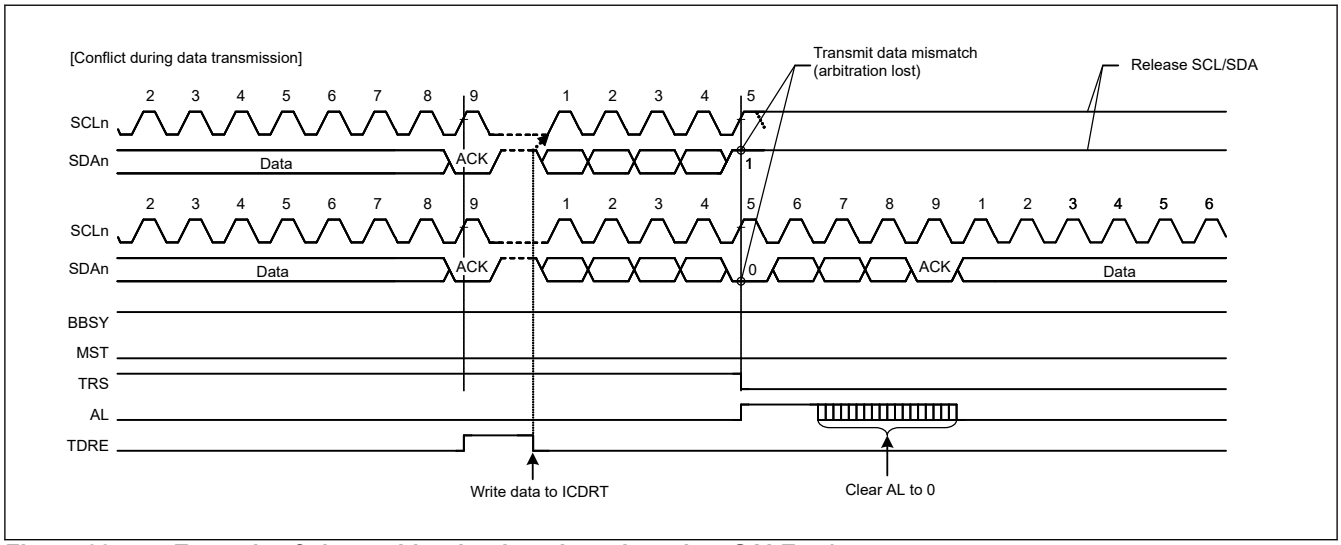



**Figure 28.44 Example of slave arbitration-lost detection when SALE = 1**

## 28.11 Start, Restart, and Stop Condition Issuing Function

### 28.11.1 Issuing a Start Condition

The IIC issues a start condition when the ST bit in ICCR2 is set to 1. When the ST bit is set to 1, a start condition request is made, and the IIC issues a start condition when the BBSY flag in ICCR2 is 0 (bus free state). When a start condition is issued normally, the IIC automatically shifts to the master transmit mode.

To issue a start condition:

1. Drive the SDAn line low (high level to low level).
2. Ensure that the time set in ICBRH and the start condition hold time elapse.
3. Drive the SCLn line low (high level to low level).
4. Detect low level on the SCLn line and ensure the low-level period of the SCLn line set in ICBRL elapses.

### 28.11.2 Issuing a Restart Condition

The IIC issues a restart condition when the RS bit in ICCR2 is set to 1. When the RS bit is set to 1, a restart condition request is made, and the IIC issues a restart condition when the BBSY flag in ICCR2 is 1 (bus busy state) and the MST bit in ICCR2 is 1 (master mode).

To issue a restart condition:

1. Release the SDAn line.
2. Ensure the low-level period of the SCLn line set in ICBRL elapses.
3. Release the SCLn line (low level to high level).
4. Detect a high level on the SCLn line and ensure the time set in ICBRL and the restart condition setup time elapse.
5. Drive the SDAn line low (high level to low level).
6. Ensure the time set in ICBRH and the restart condition hold time elapse.
7. Drive the SCLn line low (high level to low level).
8. Detect a low level on the SCLn line and ensure the low-level period of the SCLn line set in ICBRL elapses.

Note: When issuing restart condition requests, write the slave address to ICDRT after confirming that ICCR2.RS = 0. Data written while ICCR2.RS = 1 is not forwarded because of the retransmission condition before the occurrence.

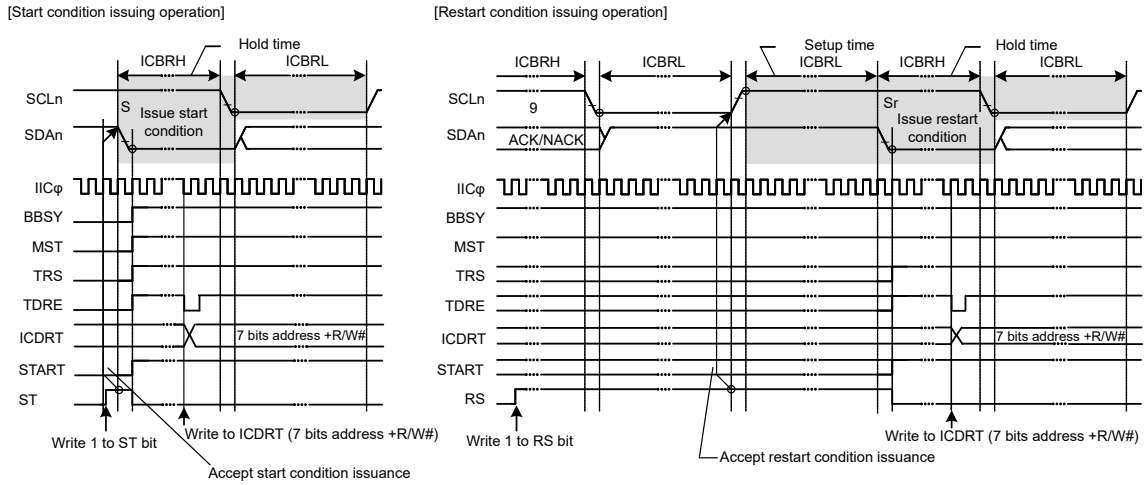



**Figure 28.45 Start and restart condition issue timing using the ST and RS bits**

Figure 28.46 shows the operation timing when a restart condition is issued after the master transmission.

[To issue a restart condition after the master transmission:]

1. Initialize the IIC using the procedure in section 28.3.2. Initial Settings.
2. Read the BBSY flag in ICCR2 to check that the bus is open, then set the ST bit in ICCR2 to 1 (start condition issuance request). On receiving the request, the IIC issues a start condition. At the same time, the BBSY and the START flags in ICSR2 are automatically set to 1 and the ST bit is automatically set to 0. If the start condition is detected and the internal levels for the SDA output state and the levels on the SDAn line match while the ST bit is 1, the IIC recognizes that a start condition is successfully issued as requested by the ST bit has been successfully completed. The MST and TRS bits in ICCR2 are automatically set to 1, placing the IIC in master transmit mode. The TDRE flag in ICSR2 is also automatically set to 1 when the TRS bit is set to 1.
3. Check that the TDRE flag in ICSR2 is 1, and then write the value for transmission (the slave address and the R/W# bit) to ICDRT. After the data for transmission is written to ICDRT, the TDRE flag is automatically set to 0, data is transferred from ICDRT to ICDRS, and the TDRE flag is again set to 1. After the byte containing the slave address and R/W# bit has been transmitted, the value of the TRS bit is automatically updated to select master transmit or master receive mode according to the value of the transmitted R/W# bit. If the value of the R/W# bit is 0, the IIC continues in master transmit mode. If the ICSR2.NACKF flag is 1 at this time, indicating that no slave device recognized the address or that there was an error in communications, write 1 to ICCR2.SP bit to issue a stop condition. To transmit data with an address in the 10-bit format, start by writing 1111 0b, the 2 upper bits of the slave address, and W to ICDRT as the first address transmission. Then, as the second address transmission, write the 8 lower bits of the slave address to ICDRT.
4. After confirming that the TDRE flag in ICSR2 is 1, write data for transmission to the ICDRT register. The IIC automatically holds the SCLn line low until data for transmission is ready, a restart condition is issued or a stop condition is issued.
5. After all bytes of data for transmission are written to the ICDRT register, wait until the value of the TEND flag in ICSR2 returns to 1. Then after checking that the START flag in ICSR2 is 1, set the START flag in ICSR2 to 0.
6. Set the RS bit in ICCR2 to 1 (restart condition issue request). On receiving the request, the IIC issues a restart condition.
7. After checking that the START flag in ICSR2 is 1, write the value for transmission (the slave address and the R/W# bit) to ICDRT.

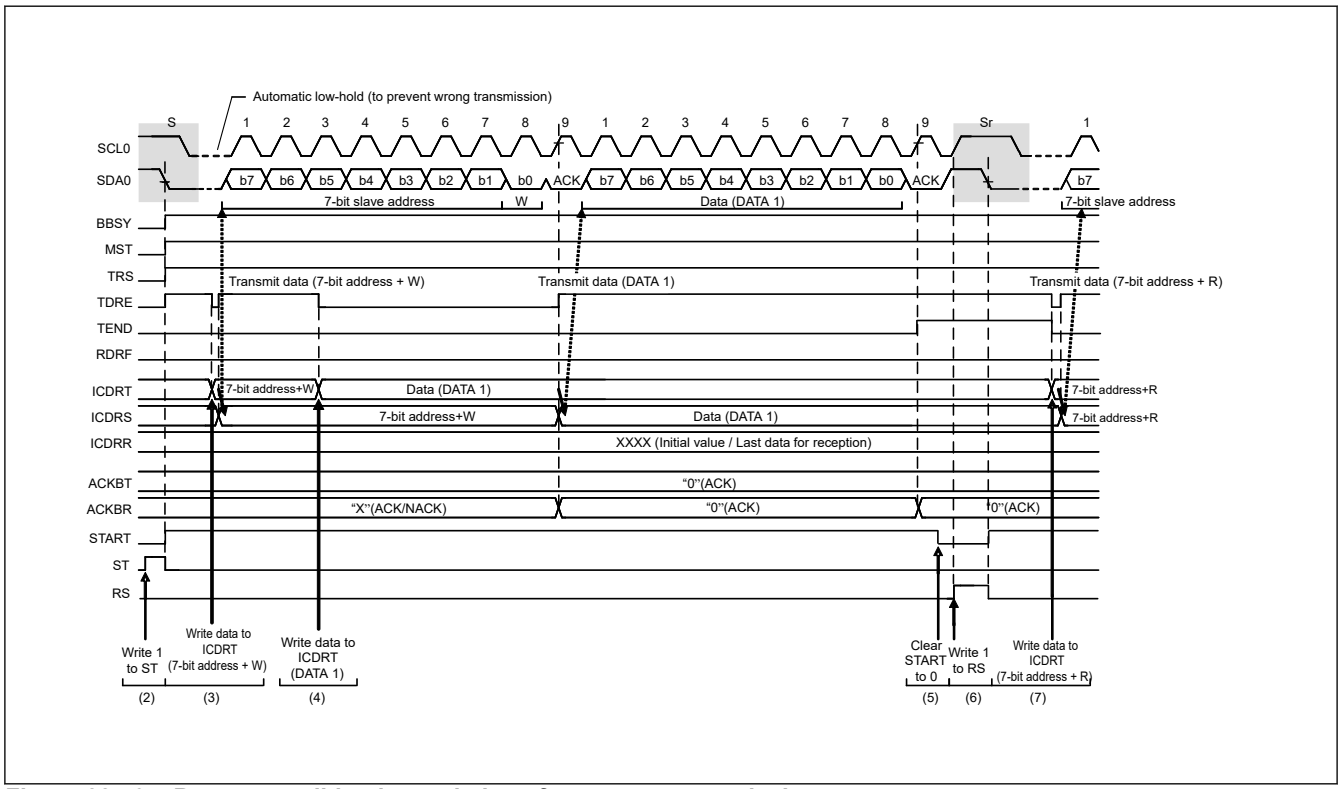



**Figure 28.46 Restart condition issue timing after master transmission.**

### 28.11.3 Issuing a Stop Condition

The IIC issues a stop condition when the SP bit in ICCR2 is set to 1. When the SP bit is set to 1, a stop condition request is made, and the IIC issues a stop condition when the BBSY flag in ICCR2 is 1 (bus busy state) and the MST bit in ICCR2 is 1 (master mode).

To issue a stop condition:

1. Drive the SDAn line low (high level to low level).
2. Ensure the low-level period of the SCLn line set in ICBRL elapses.
3. Release the SCLn line (low level to high level).
4. Detect a high level on the SCLn line and ensure the time set in ICBRH and the stop condition setup time elapse.
5. Release the SDAn line (low level to high level).
6. Ensure the time set in ICBRL and the bus free time elapse.
7. Clear the BBSY flag to 0 to release the bus mastership.

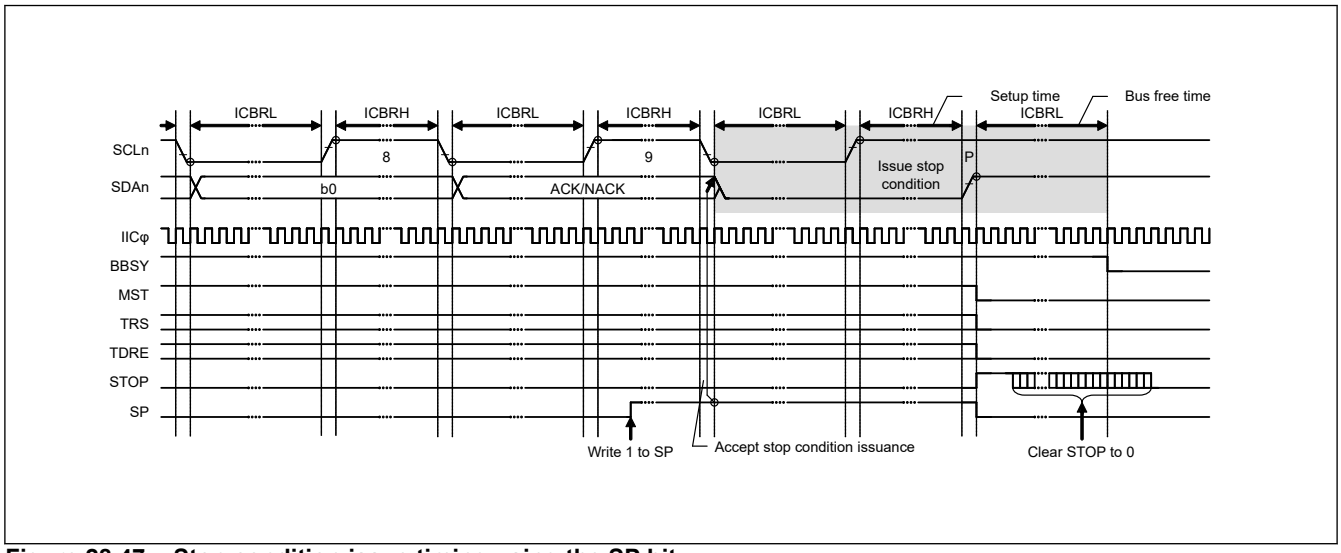



**Figure 28.47 Stop condition issue timing using the SP bit**

## 28.12 Bus Hanging

If the clock signals from the master and slave devices are out of synchronization because of noise or other factors, the $I^2C$ bus might hang with a fixed level on the SCLn line or SDAn line.

To manage bus hanging, the IIC has a timeout function to detect hanging by monitoring the SCLn line, and a function for outputting an extra SCL clock cycle to release the bus from:

- A timeout function to detect hanging by monitoring the SCLn line
- The IIC reset function
- An internal reset function.

By checking the SCLO, SDAO, SCLI, and SDAI bits in ICCR1, it is possible to see whether the IIC or its communicating partner is placing the low level on the SCLn or SDAn line.

### 28.12.1 Timeout Function

The timeout function can detect when the SCLn line is stuck longer than the predetermined time. The IIC can detect an abnormal bus state by monitoring that the SCLn line is stuck low or high for a predetermined time.

The timeout function monitors the SCLn line state and counts the low- or high-level period using the internal counter. The timeout function resets the internal counter each time the SCLn line changes (rises or falls), but continues to count unless the SCLn line changes. If the internal counter overflows because no SCLn line changes, the IIC can detect the timeout and report the bus hung state.

This timeout function is enabled when the ICFER.TMOE bit is 1. It detects a hung state when the SCLn line is stuck low or high during the following conditions:

- The bus is busy (ICCR2.BBSY flag is 1) in master mode (ICCR2.MST bit is 1)
- The IIC slave address is detected (ICSR1 register is not 0x00) and the bus is busy (ICCR2.BBSY flag is 1) in slave mode (ICCR2.MST bit is 0)
- The bus is open (ICCR2.BBSY flag is 0) while a start condition is requested (ICCR2.ST bit is 1).

The internal counter of the timeout function uses the internal reference clock (IICφ) set in the CKS[2:0] bits in ICMR1 as a count source. It functions as a 16-bit counter when long mode is selected (TMOS bit = 0 in ICMR2) or a 14-bit counter when short mode is selected (TMOS bit = 1).

The SCLn line level (low, high, or both levels) during which this counter is activated can be selected in the TMOH and TMOL bits in ICMR2. If both TMOL and TMOH bits are set to 0, the internal counter is disabled.

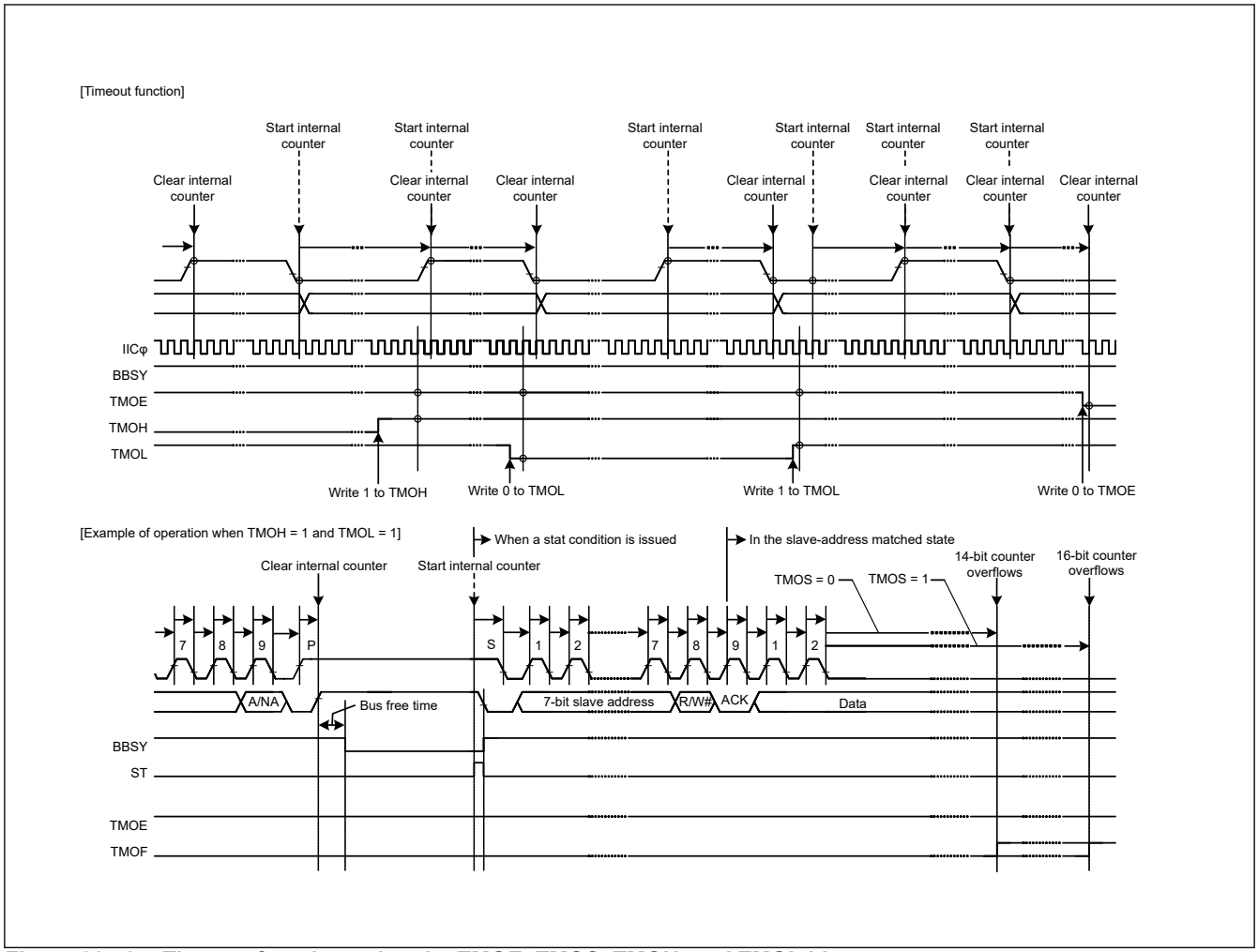



**Figure 28.48 Timeout function using the TMOE, TMOS, TMOH, and TMOL bits**

## 28.12.2 Extra SCL Clock Cycle Output Function

In master mode, this function outputs extra SCL clock cycles to release the SDAn line of the slave device from being held low because the master is out of synchronization with the slave device. This function is mainly used in master mode to release the SDAn line of the slave device from being fixed low by including extra cycles of SCL output from the IIC. It uses single cycles of the SCL clock for a bus error where the IIC cannot issue a stop condition because the slave device is holding the SDAn line at the low level. Do not use this function in normal situations. Using it when communications are proceeding correctly leads to malfunctions.

When the CLO bit in ICCR1 is set to 1 in master mode, a single cycle of the SCL clock at the frequency specified in the CKS[2:0] bits in ICMR1, and in the ICBRH and ICBRL registers, is output as an extra clock cycle. After output of this

single cycle of the SCL clock, the CLO bit is automatically set to 0. At this time, if ICCR2.BBSY = 1, the SCL pin goes low, and when ICCR2.BBSY = 0, the SCL pin goes high. After confirming that the CLO bit is 0 by software, write 1 to the CLO bit to output the additional clock continuously.

When the IIC module is in master mode and the slave device is holding the SDAn line low because synchronization with the slave device is lost because of effects like noise, the output of a stop condition is not possible. This function can be used to output extra cycles of SCL one by one to make the slave device release the SDAn line from being held low, and so recover the bus from an unusable state. Release of the SDAn line by the slave device can be monitored by reading the SDAI bit in ICCR1. After confirming the release of the SDAn line by the slave device, complete communications by reissuing the stop condition.

[Output conditions for using the CLO bit in ICCR1]

- When the bus is open (BBSY flag in ICCR2 = 0) or in master mode (MST bit = 1 and BBSY flag = 1 in ICCR2)
- When the communication device does not hold the SCLn line low.

Figure 28.49 shows the operation timing of the extra SCL clock cycle output function (CLO bit).

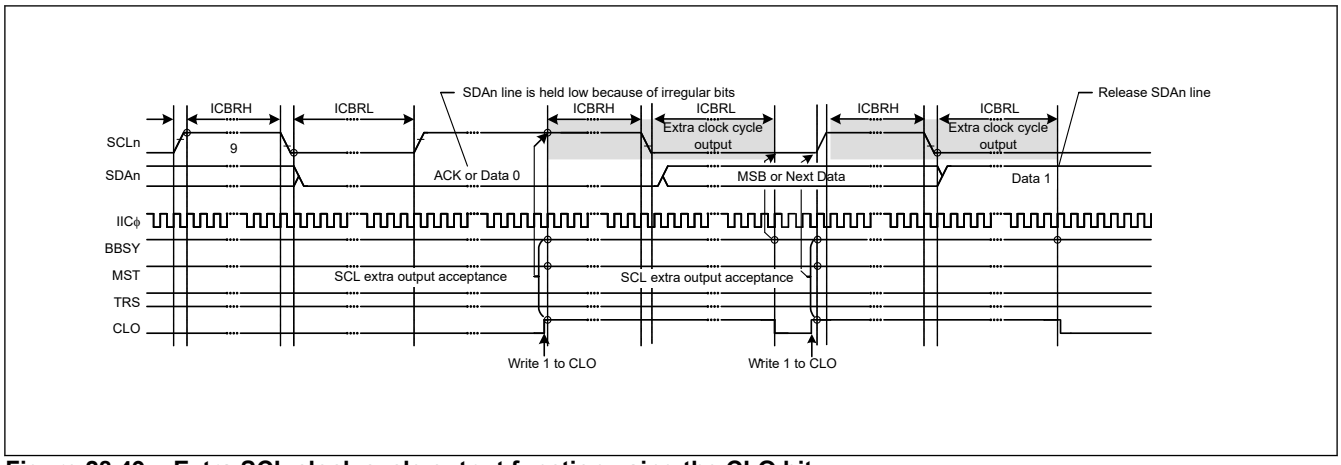



**Figure 28.49 Extra SCL clock cycle output function using the CLO bit**

## 28.12.3 IIC Reset and Internal Reset

The IIC module incorporates a function for resetting itself. It uses two types of resets:

- An IIC reset, which initializes all registers, including the BBSY flag in ICCR2.
- An internal reset, which releases the IIC from the slave-address matched state and initializes the internal counter while saving other settings.

After issuing a reset, always set the IICRST bit in ICCR1 to 0. Both types of resets are valid for release from bus-hung states, because both restore the output state of the SCLn and SDAn pins to the high-impedance state.

Issuing a reset during slave operation might lead to a loss of synchronization between the master device clock and the slave device clock, so avoid this when possible. In addition, monitoring of the bus state, such as for the presence of a start condition, is not possible during an IIC reset (ICE and IICRST bits = 01b in ICCR1).

For a detailed description of the IIC and internal resets, see section 28.15. State of Registers When Issuing Each Condition.

## 28.13 SMBus Operation

The IIC supports data communication conforming to the SMBus Specification (version 2.0). To perform SMBus communication, set the SMBS bit in ICMR3 to 1. To use the transfer rate within a range of 10 to 100 kbps of the SMBus standard, set the CKS[2:0] bits in ICMR1, the ICBRH, and ICBRL registers. In addition, specify the values in the DLCS bit in ICMR2 and the SDDL[2:0] bits in ICMR2 to meet the data hold time specification of 300 ns or more. When the IIC is used only as a slave device, the transfer rate setting is not required, but ICBRL must be set to a value longer than the data setup time (250 ns).

For the SMBus device default address (1100 001b), use one of the slave address registers L0 to L2 (SARL0, SARL1, and SARL2), and set the associated FS bit (7- or 10-bit address format select) in SARUy (y = 0 to 2) to 0 (7-bit address format).

When transmitting the UDID (Unique Device Identifier), set the SALE bit in ICFER to 1 to enable the slave arbitration-lost detection function.

### 28.13.1 SMBus Timeout Measurement

#### (1) Measuring slave device timeout

The following period (timeout interval: $T_{LOW:\ SEXT}$) must be measured for slave devices in SMBus communication:

- From start condition to stop condition.

To measure timeout for slave devices, measure the period from start condition detection to stop condition detection with the GPT using the IIC start condition detection interrupt (STIn) and stop condition detection interrupt (SPIn). The measured timeout period must be within the total clock low-level period [slave device] $T_{LOW:\ SEXT}$: 25 ms (maximum) of the SMBus standard.

If the time measured with the GPT exceeds the clock low-level detection timeout $T_{TIMEOUT}$: 25 ms (minimum) of the SMBus standard, the slave device must release the bus by writing 1 to the IICRST bit in ICCR1 to issue an internal reset of the IIC. When an internal reset is issued, the IIC stops driving the bus for the SCLn and SDAn pins, making them output high-impedance, which releases the bus.

#### (2) Measuring master device timeout

The following periods (timeout interval: $T_{LOW:\ MEXT}$) must be measured for master devices in SMBus communication:

- From start condition to acknowledge bit
- Between acknowledge bits
- From acknowledge bit to stop condition.

To measure timeout for master devices, measure these periods with the GPT using the IIC start condition detection interrupt (STIn), stop condition detection interrupt (SPIn), transmit end interrupt (IICn_TEI), or receive data full interrupt (IICn_RXI). The measured timeout period must be within the total clock low-level extended period (master device) $T_{LOW:\ MEXT}$: 10 ms (maximum) of the SMBus standard, and the total of all $T_{LOW:\ MEXT}$ values from start condition to stop condition must be within $T_{LOW:\ SEXT}$: 25 ms (maximum).

For the ACK receive timing (rising edge of the 9th SCL clock cycle), monitor the TEND flag in ICSR2 in master transmit mode (master transmitter) and the RDRF flag in ICSR2 in master receive mode (master receiver). Perform byte-wise transmit operations in master transmit mode, and hold the RDRFS bit in ICMR3 at 0 until the byte immediately before reception of the final byte in master receive mode. While the RDRFS bit is 0, the RDRF flag is set to 1 on the rising edge of the 9th SCL clock cycle.

If the period measured with the GPT exceeds the total clock low-level extended period (master device) $T_{LOW:\ MEXT}$: 10 ms (maximum) of the SMBus standard or the total of measured periods exceeds the clock low-level detection timeout $T_{TIMEOUT}$: 25 ms (minimum) of the SMBus standard, the master device must stop the transaction by issuing a stop condition. In master transmit mode, immediately stop the transmit operation (stop writing data to ICDRT).

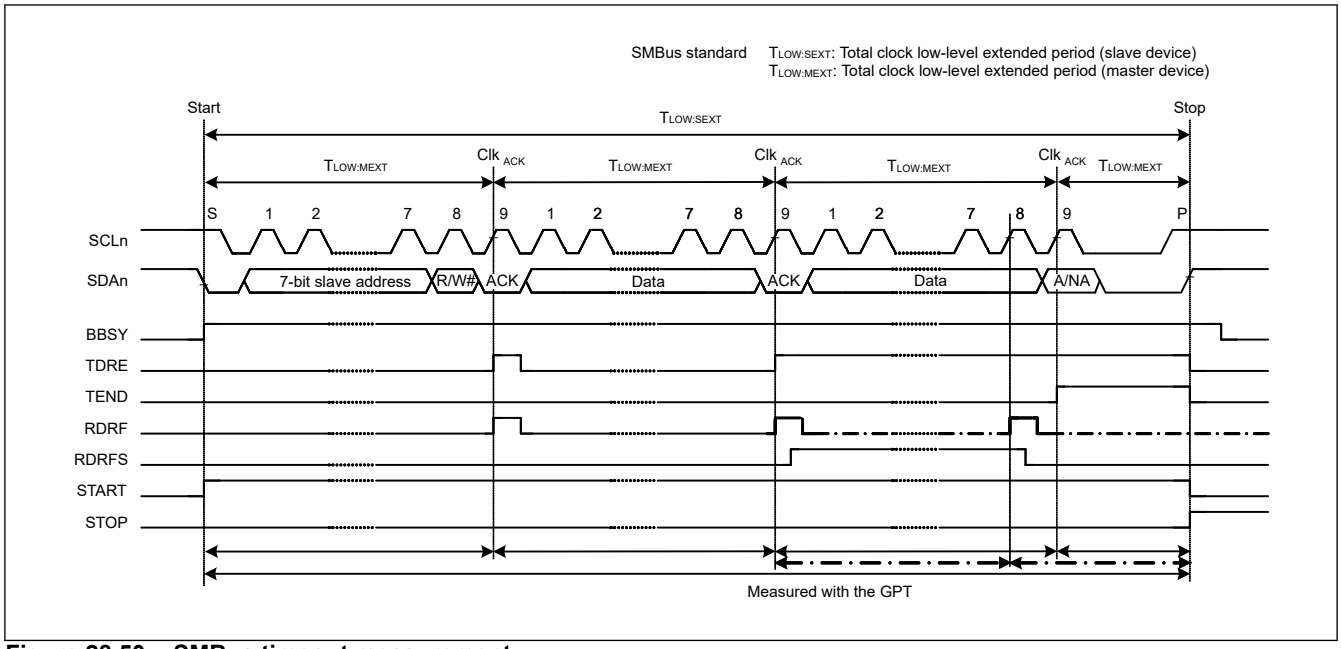



**Figure 28.50 SMBus timeout measurement**

## 28.13.2 Packet Error Code (PEC)

The MCU provides a CRC calculator that enables transmission of a Packet Error Code (PEC) or allows checking of the received data in SMBus data communication for the IIC. For the CRC-generating polynomials of the CRC calculator, see section 34, Cyclic Redundancy Check (CRC).

In master transmit mode, the PEC data can be generated by writing all transmit data to the CRC Data Input Register (CRCDIR) in the CRC calculator.

In master receive mode, the PEC data can be checked by writing all receive data to CRCDIR in the CRC calculator and comparing the obtained value in the CRC Data Output Register (CRCDOR) with the received PEC data.

To send ACK or NACK based on the match or mismatch result when the final byte is received as a result of the PEC code check, set the RDRFS bit in ICMR3 to 1 before the rising edge of the 8th SCL clock cycle during reception of the final byte, and hold the SCLn line low on the falling edge of the 8th clock cycle.

## 28.13.3 SMBus Host Notification Protocol (Notify ARP Master Command)

In communicating over an SMBus, a slave device can temporarily act as a master device to notify the SMBus host (or ARP master) of its own slave address, or to request its own slave address from the SMBus host.

For a product using the MCU to operate as an SMBus host or ARP master, the host address (0001 000b) sent from the slave device must be detected as a slave address, and so the IIC provides a function for detecting the host address. To detect the host address as a slave address, set the SMBS bit in ICMR3 and the HOAE bit in ICSER to 1. Operation after the host address is detected is the same as normal slave operation.

## 28.14 Interrupt Sources

The IIC issues five types of interrupt requests:

- Transfer error or event generation (arbitration-lost, NACK detection, timeout detection, start condition detection, and stop condition detection)
- Receive data full
- Transmit data empty
- Transmit end
- Address match during wakeup function

Table 28.10 lists details about the interrupt requests. The receive data full and transmit data empty interrupts can activate data transfer by the DTC or DMAC.

**Table 28.10 Interrupt sources**

| Symbol | Interrupt source | Interrupt flag | DTC or DMAC activation | Interrupt condition |
|---|---|---|---|---|
| IICn_EEI*5 | Transfer error or event occurrence | AL | Not possible | AL = 1, ALIE = 1 |
| | | NACKF | | NACKF = 1, NAKIE = 1 |
| | | TMOF | | TMOF = 1, TMOIE = 1 |
| | | START | | START = 1, STIE = 1 |
| | | STOP | | STOP = 1, SPIE = 1 |
| IICn_RXI*2 *5 | Receive data full | RDRF | Possible | RDRF = 1, RIE = 1 |
| IICn_TXI*1 *5 | Transmit data empty | RDRF | Possible | TDRE = 1, TIE = 1 |
| IICn_TEI*3 *5 | Transmit end | TEND | Not possible | TEND = 1, TEIE = 1 |
| IIC0_WUI*4 | Slave address match during wakeup function | WUF | Not possible | Slave address match<br>Slave receive complete<br>RWAK operation ASY0 = 1<br>WUIE = 1 |

Note: There is a delay between the execution of a write instruction for a peripheral module by the CPU and the actual writing to the module. When an interrupt flag is cleared or masked, read the relevant flag again to check whether clearing or masking is complete, then return from interrupt handling. Not doing so creates the possibility of repeated processing of the same interrupt.
Note 1. Because IICn_TXI is edge-detected, it does not require clearing. Additionally, the TDRE flag in ICSR2 (condition for IICn_TXI) is automatically set to 0 when transmit data is written to the ICDRT register or a stop condition is detected (STOP flag = 1 in ICSR2).
Note 2. Because IICn_RXI is edge-detected, it does not require clearing. Additionally, the RDRF flag in ICSR2 (condition for IICn_RXI) is automatically set to 0 when data is read from ICDRR.
Note 3. When using the IICn_TEI interrupt, clear the TEND flag in ICSR2 in the IICn_TEI interrupt handling. The TEND flag in ICSR2 automatically is set to 0 when transmit data is written to the ICDRT register or a stop condition is detected (STOP flag = 1 in ICSR2).
Note 4. Only channel 0 has a wakeup function, so IIC0_WUI is for channel 0 only.
Note 5. Channel number (n = 0, 1).

Clear or mask each flag during interrupt handling.

### 28.14.1 Buffer Operation for IICn_TXI and IICn_RXI Interrupts

If the conditions for generating an IICn_TXI or IICn_RXI interrupt are satisfied while the associated IR flag is 1, the interrupt request is not output for the ICU but is saved internally. One request per source can be saved internally.

An interrupt request that is saved in the ICU is output when the ICU.IELSRn.IR flag becomes 0. Internally saved interrupt requests are automatically cleared under normal conditions. They can also be cleared by writing 0 to the interrupt enable bit within the associated peripheral module.

## 28.15 State of Registers When Issuing Each Condition

The IIC has two dedicated resets, IIC reset and Internal reset. Table 28.11 lists the registers states when issuing each condition.

**Table 28.11 Register states when issuing each condition (1 of 2)**

| Registers | | Reset | IIC reset (ICE = 0, IICRST = 1) | Internal reset (ICE = 1, IICRST = 1) | Start or restart condition detection | Stop condition detection |
|---|---|---|---|---|---|---|
| ICCR1 | ICE, IICRST | In reset | Saved | Saved | Saved | Saved |
| | SCLO, SDAO | | In reset | In reset | | |
| | Others | | | Saved | | |
| ICCR2 | BBSY | In reset | In reset | Saved | Set | In reset |
| | ST, RS | | | In reset | In reset | Saved |
| | SP | | | | | In reset |
| | TRS | | | | Set or saved | |
| | MST | | | | | |

**Table 28.11 Register states when issuing each condition (2 of 2)**

| Registers | | Reset | IIC reset (ICE = 0, IICRST = 1) | Internal reset (ICE = 1, IICRST = 1) | Start or restart condition detection | Stop condition detection |
|---|---|---|---|---|---|---|
| ICMR1 | BC[2:0] | In reset | In reset | In reset | In reset | Saved |
| | Others | | | Saved | Saved | |
| ICMR2 | | In reset | In reset | Saved | Saved | Saved |
| ICMR3 | ACKBT | In reset | In reset | Saved | Saved | In reset |
| | Others | | | | | Saved |
| ICFER | | In reset | In reset | Saved | Saved | Saved |
| ICSER | | In reset | In reset | Saved | Saved | Saved |
| ICIER | | In reset | In reset | Saved | Saved | Saved |
| ICSR1 | | In reset | In reset | In reset | Saved | In reset |
| ICSR2 | TEND | In reset | In reset | In reset | Saved | In reset |
| | TDRE | | | | Set or saved | |
| | START | | | | Set | |
| | STOP | | | | Saved | Set |
| | Others | | | | Saved | Saved |
| ICWUR | | In reset | In reset | Saved | Saved | Saved |
| SARL0, SARL1, SARL2 SARU0, SARU1, SARU2 | | In reset | In reset | Saved | Saved | Saved |
| ICBRH, ICBRL | | In reset | In reset | Saved | Saved | Saved |
| ICDRT | | In reset | In reset | Saved | Saved | Saved |
| ICDRR | | In reset | In reset | Saved | Saved | Saved |
| ICDRS | | In reset | In reset | In reset | Saved | Saved |
| Timeout function | | In reset | In reset | In reset | Operating | Operating |
| Bus free time measurement | | In reset | In reset | Operating | Operating | Operating |
| ICWUR2 | WUSEN | In reset | In reset | Saved | Saved | Saved |
| | Others | | | | | Saved or set or reset |

## 28.16 Event Link Output

The IIC0 module handles the event output for the Event Link Controller (ELC) for the following sources:

### (1) Transfer error event

When a transfer error event occurs, the associated event signal can be output to another module by the ELC.

### (2) Receive data full

When a receive data register becomes full, the associated event signal can be output to another module by the ELC.

### (3) Transmit data empty

When a transmit data register becomes empty, the associated event signal can be output to another module by the ELC.

### (4) Transmit end

On completion of the transfer, the associated event signal can be output to another module by the ELC.

## 28.16.1 Interrupt Handling and Event Linking

Each of the IIC interrupt types (see Table 28.10) has an enable bit to control enabling and disabling of the associated interrupt signal. An interrupt request signal is output to the CPU when an interrupt source condition is satisfied while the associated enable bit is set.

The associated event link output signals are sent to other modules as event signals by the ELC when the interrupt source conditions are satisfied, regardless of the interrupt enable bit settings. For details on interrupt sources, see Table 28.10.

## 28.17 Usage Notes

### 28.17.1 Settings for the Module-Stop Function

The Module Stop Control Register B (MSTPCRB) can enable or disable IIC operation. The IIC is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

### 28.17.2 Notes on Starting Transfer

If the IR flag associated with the IIC interrupt is 1 when transfer is started (ICCR1.ICE bit = 1), follow the procedure in this section to clear the interrupts before enabling operations. Starting transfer with the IR flag set to 1 while the ICCR1.ICE bit is 1 leads to an interrupt request being internally saved after transfer starts, and this can lead to unexpected behavior of the IR flag.

To clear interrupts before starting transfer operation:

1. Confirm that the ICCR1.ICE bit is 0.
2. Set the relevant interrupt enable bits, such as ICIER.TIE to 0.
3. Read the relevant interrupt enable bits, such as ICIER.TIE, and confirm that the value is 0.
4. Set the IR flag to 0.

# 29. Controller Area Network (CAN) Module

## 29.1 Overview

The Controller Area Network (CAN) module uses a message-based protocol to receive and transmit data between multiple slaves and masters in electromagnetically noisy applications. The module complies with the ISO 11898-1 (CAN 2.0A/CAN 2.0B) standard and supports up to 32 mailboxes, which can be configured for transmission or reception in normal mailbox and FIFO modes. Both standard (11-bit) and extended (29-bit) messaging formats are supported. The CAN module requires an additional external CAN transceiver.

Table 29.1 lists the CAN specifications and Figure 29.1 shows a block diagram.

**Table 29.1 CAN specifications (1 of 2)**

| Parameter | Specifications |
|---|---|
| Data transfer rate | ISO11898-1-compliant for standard and extended frames |
| Bit rate | Data transfer rate programmable up to 1 Mbps (fCAN ≥ 8 MHz)<br>fCAN: CAN clock source |
| Message box | 32 mailboxes, with two selectable mailbox modes<br>● Normal mode: 32 mailboxes independently configurable for either transmission or reception<br>● FIFO mode: 24 mailboxes independently configurable for either transmission or reception, with remaining mailboxes used for receive (RX) and transmit (TX) 4-stage FIFOs |
| Reception | ● Support for data frame and remote frame reception<br>● Reception ID format selectable to only standard ID, only extended ID, or mixed IDs<br>● Programmable one-shot reception function<br>● Selectable between overwrite mode (unread message overwritten) and overrun mode (unread message saved)<br>● Reception complete interrupt independently enabled or disabled for each mailbox |
| Acceptance filter | ● Eight acceptance masks (one for every four mailboxes)<br>● Masks independently enabled or disabled for each mailbox |
| Transmission | ● Support for data frame and remote frame transmission<br>● Transmission ID format selectable to only standard ID, only extended ID, or mixed IDs<br>● Programmable one-shot transmission function<br>● Broadcast messaging function<br>● Priority mode selectable based on message ID or mailbox number<br>● Support for transmission request abort, with abort completion confirmed in status flag<br>● Transmission complete interrupt independently enabled or disabled for each mailbox |
| Mode transition for bus-off recovery | Mode transition for the recovery from the bus-off state selectable to:<br>● ISO11898-1 specification-compliant<br>● Automatic invoking of CAN halt mode on bus-off entry<br>● Automatic invoking of CAN halt mode on bus-off end<br>● Invoking of CAN halt mode through the software<br>● Transition to error-active state through the software |
| Error status monitoring | ● Monitoring of CAN bus errors, including stuff error, form error, ACK error, 15-bit CRC error, bit error, and ACK delimiter error<br>● Detection of transition to error states, including error-warning, error-passive, bus-off entry, and bus-off recovery<br>● Supports reading of error counters |
| Time stamping | ● Time stamp function using a 16-bit counter<br>● Reference clock selectable to 1-bit, 2-bit, 4-bit, and 8-bit time periods |
| Interrupt function | Support for five interrupt sources:<br>● Reception complete<br>● Transmission complete<br>● Receive FIFO<br>● Transmit FIFO<br>● Error interrupts |
| CAN sleep mode | CAN clock stopped to reduce power consumption |

**Table 29.1 CAN specifications (2 of 2)**

| Parameter | Specifications |
|---|---|
| Software support unit | Three software support units:<br>• Acceptance filter support<br>• Mailbox search support, including receive mailbox search, transmit mailbox search, and message lost search<br>• Channel search support |
| CAN clock source | CANMCLK or PCLKB |
| Test mode | Three test modes available for evaluation purposes:<br>• Listen-only mode<br>• Self-test mode 0 (external loopback)<br>• Self-test mode 1 (internal loopback) |
| TrustZone Filte | Security attribution can be set |

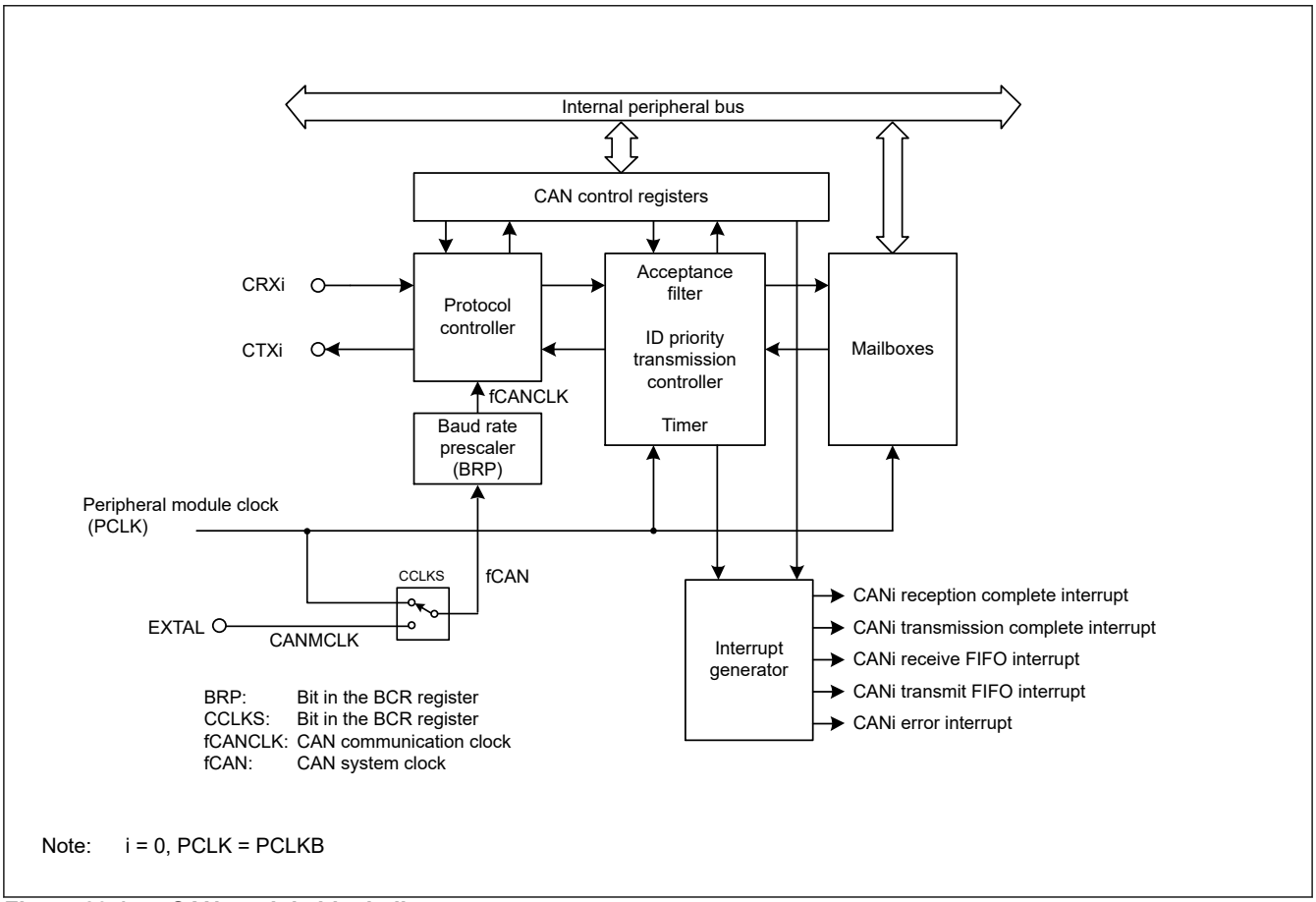



**Figure 29.1 CAN module block diagram**

The CAN module includes the following blocks:

- CAN input and output pins

  CRXi and CTXi (i = 0)

- Protocol controller

  Handles CAN protocol processing such as bus arbitration, bit timing during transmission and reception, stuffing, and error handling.

- Mailboxes

  Consists of 32 mailboxes, which can be configured as either transmit or receive. Each mailbox has an individual ID, data length code (DLC), data field (8 bytes), and a time stamp.

- Acceptance filter

  Performs filtering of received messages. MKR0 to MKR7 are used for the filtering process.

- Timer

  Used for the time stamp function. The timer value when a message is stored in the mailbox is written as the time stamp value.

- Interrupt generator for five types of interrupts:
  - CANi reception complete interrupt
  - CANi transmission complete interrupt
  - CANi receive FIFO interrupt
  - CANi transmit FIFO interrupt
  - CANi error interrupt

Table 29.2 lists the CAN module pins. These pins are multiplexed with other signals on the MCU. For details, see section 19, I/O Ports.

**Table 29.2 CAN module I/O pins (i = 0)**

| Pin name | I/O | Function |
|---|---|---|
| CRXi | Input | Data receive |
| CTXi | Output | Data transmit |

## 29.2 Register Descriptions

### 29.2.1 CTLR : Control Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x840

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | RBOC | BOM[1:0] | | SLPM | CANM[1:0] | | TSPS[1:0] | | TSRC | TPM | MLM | IDFM[1:0] | | MBM |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MBM | CAN Mailbox Mode Select*1<br>0: Normal mailbox mode<br>1: FIFO mailbox mode | R/W |
| 2:1 | IDFM[1:0] | ID Format Mode Select*1<br>0 0: Standard ID mode<br>All mailboxes, including FIFO mailboxes, handle only standard IDs<br>0 1: Extended ID mode<br>All mailboxes, including FIFO mailboxes, handle only extended IDs<br>1 0: Mixed ID mode<br>All mailboxes, including FIFO mailboxes, handle both standard and extended IDs. In normal mailbox mode, use the associated IDE bit to differentiate standard and extended IDs. In FIFO mailbox mode, the associated IDE bits are used for mailboxes 0 to 23, the IDE bits in FIDCR0 and FIDCR1 are used for the receive FIFO, and the IDE bit associated with mailbox 24 is used for the transmit FIFO.<br>1 1: Setting prohibited | R/W |
| 3 | MLM | Message Lost Mode Select*1<br>0: Overwrite mode<br>1: Overrun mode | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4 | TPM | Transmission Priority Mode Select[*1]<br>0: ID priority transmit mode<br>1: Mailbox number priority transmit mode | R/W |
| 5 | TSRC | Time Stamp Counter Reset Command[*4]<br>0: Do not reset time stamp counter<br>1: Reset time stamp counter[*3] | R/W |
| 7:6 | TSPS[1:0] | Time Stamp Prescaler Select[*1]<br>0 0: Every 1-bit time<br>0 1: Every 2-bit time<br>1 0: Every 4-bit time<br>1 1: Every 8-bit time | R/W |
| 9:8 | CANM[1:0] | CAN Operating Mode Select[*5]<br>0 0: CAN operation mode<br>0 1: CAN reset mode<br>1 0: CAN halt mode<br>1 1: CAN reset mode (forced transition) | R/W |
| 10 | SLPM | CAN Sleep Mode[*5 *6]<br>0: All other modes<br>1: CAN sleep mode | R/W |
| 12:11 | BOM[1:0] | Bus-Off Recovery Mode[*1]<br>0 0: Normal mode (ISO11898-1-compliant)<br>0 1: Enter CAN halt mode automatically on entering bus-off state<br>1 0: Enter CAN halt mode automatically at the end of bus-off state<br>1 1: Enter CAN halt mode during bus-off recovery period through a software request | R/W |
| 13 | RBOC | Forcible Return from Bus-Off[*2]<br>0: No return occurred<br>1: Forced return from bus-off state[*3] | R/W |
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Write to the BOM[1:0], TSPS[1:0], TPM, MLM, IDFM[1:0], and MBM bits in CAN reset mode.
Note 2. Set the RBOC bit to 1 in the bus-off state.
Note 3. This bit is automatically set to 0 after being set to 1. It should read as 0.
Note 4. Set the TSRC bit to 1 in CAN operation mode.
Note 5. When the CANM[1:0] and SLPM bits are changed, check STR to ensure that the mode is switched. Do not change the CANM[1:0] bits or SLPM bit until the mode is switched.
Note 6. Write to the SLPM bit in CAN reset mode or CAN halt mode. When changing the SLPM bit, write 0 or 1 only to the SLPM bit.

### MBM bit (CAN Mailbox Mode Select)

When the MBM bit is 0 (normal mailbox mode), mailboxes 0 to 31 are configured as transmit or receive mailboxes. When the MBM bit is 1 (FIFO mailbox mode):

- Mailboxes 0 to 23 are configured as transmit or receive mailboxes
- Mailboxes 24 to 27 are configured as transmit FIFO
- Mailboxes 28 to 31 are configured as receive FIFO

Transmit data is written into mailbox 24, a window mailbox for the transmit FIFO. Receive data is read from mailbox 28, a window mailbox for the receive FIFO.

Table 29.3 lists the mailbox configuration.

### IDFM[1:0] bits (ID Format Mode Select)

The IDFM[1:0] bits specify the ID format.

### MLM bit (Message Lost Mode Select)

The MLM bit specifies the operation when a new message is captured in an unread mailbox. Overwrite mode or overrun mode can be selected. In both cases, the mode applies to all mailboxes, including the receive FIFO.

When the MLM bit is 0, all mailboxes are set to overwrite mode. Any new message received overwrites the pre-existing message.

When the MLM bit is 1, all mailboxes are set to overrun mode. Any new message received does not overwrite the preexisting message, and it is discarded.

#### TPM bit (Transmission Priority Mode Select)

The TPM bit specifies the priority when transmitting messages. ID priority transmit mode or mailbox number transmit mode can be selected. All mailboxes are set for either ID priority transmission or mailbox number priority transmission.

When TPM is 0, ID priority transmit mode is selected and transmission priority is arbitrated as defined in the ISO11898- 1 CAN specification. In ID priority transmit mode, mailboxes 0 to 31 (in normal mailbox mode), and mailboxes 0 to 23 (in FIFO mailbox mode), and the transmit FIFO are compared for the IDs of mailboxes configured for transmission. If two or more mailbox IDs are the same, the mailbox with the smaller number has higher priority.

Only the next message to be transmitted from the transmit FIFO is included in the transmission arbitration. If a FIFO message is being transmitted, the next pending message within the transmit FIFO is included in the transmission arbitration.

When TPM is 1, mailbox number transmit mode is selected and the transmit mailbox with the smallest mailbox number has the highest priority. In FIFO mailbox mode, the transmit FIFO has lower priority than normal mailboxes (0 to 23).

#### TSRC bit (Time Stamp Counter Reset Command)

The TSRC bit resets the time stamp counter. When it is set to 1, TSR is set to 0x0000. TSRC is set to 0 automatically.

#### TSPS[1:0] bits (Time Stamp Prescaler Select)

The TSPS[1:0] bits select the prescaler for the time stamp. The reference clock for the time stamp can be selected to 1- bit, 2-bit, 4-bit, or 8-bit time periods.

#### CANM[1:0] bits (CAN Operating Mode Select)

The CANM[1:0] bits select one of the following modes for the CAN module:

- CAN operation mode
- CAN reset mode
- CAN halt mode

The CAN sleep mode is set in the SLPM bit. For details, see section 29.3. Operation Modes.

When the CAN module enters CAN halt mode based on the BOM[1:0] setting, the CANM[1:0] bits are automatically set to 10b.

#### SLPM bit (CAN Sleep Mode)

When the SLPM bit is set to 1, the CAN module enters CAN sleep mode. When the SLPM bit is set to 0, the CAN module exits CAN sleep mode. For details, see section 29.3. Operation Modes.

#### BOM[1:0] bits (Bus-Off Recovery Mode)

The BOM[1:0] bits select bus-off recovery mode for the CAN module.

When the BOM[1:0] bits are 00b, the recovery from bus-off is compliant with the ISO11898-1 CAN specification. The CAN module recovers CAN communication (error-active state) after detecting 11 consecutive recessive bits 128 times. A bus-off recovery interrupt request occurs when recovering from bus-off.

When the BOM[1:0] bits are 01b and the CAN module reaches the bus-off state, the CANM[1:0] bits in CTLR are set to 10b to enter CAN halt mode. No bus-off recovery interrupt request occurs when recovering from bus-off, and TECR and RECR are set to 0x00.

When the BOM[1:0] bits are 10b, the CANM[1:0] bits are set to 10b as soon as the CAN module reaches the bus-off state. The CAN module enters CAN halt mode after the recovery from the bus-off state, and after detecting 11 consecutive recessive bits 128 times. A bus-off recovery interrupt request occurs when recovering from bus-off, and TECR and RECR are set to 0x00.

When the BOM[1:0] bits are 11b, the CAN module enters CAN halt mode by setting the CANM[1:0] bits to 10b while the CAN module is still in the bus-off state. No bus-off recovery interrupt request occurs when recovering from bus-off, and TECR and RECR are set to 0x00. However, if the CAN module recovers from bus-off after detecting 11 consecutive recessive bits 128 times before the CANM[1:0] bits are set to 10b, a bus-off recovery interrupt request is generated.

If the CPU requests an entry to CAN reset mode at the same time as the CAN module attempts to enter CAN halt mode (at bus-off entry when the BOM[1:0] bits are 01b, or at bus-off end when the BOM[1:0] bits are 10b), then the CPU request has higher priority.

### RBOC bit (Forcible Return from Bus-Off)

When the RBOC bit is set to 1 in the bus-off state, the CAN module forcibly exits the bus-off state. It is set to 0 automatically, and the error state changes from bus-off to error-active. When the RBOC bit is set to 1, RECR and TECR are set to 0x00 and the BOST bit in STR is set to 0, indicating that the CAN module is not in bus-off state. The other registers remain unchanged when RBOC is set to 1. No bus-off recovery interrupt request occurs by this recovery from the bus-off state. Use the RBOC bit only when the BOM[1:0] bits are 00b (normal mode).

**Table 29.3 Mailbox configuration**

| Mailbox | MBM bit = 0 (normal mailbox mode) | MBM bit = 1 (FIFO mailbox mode)*1 *2 *3 *4 *5 |
|---|---|---|
| Mailboxes 0 to 23 | Normal mailbox | Normal mailbox |
| Mailboxes 24 to 27 | | Transmit FIFO |
| Mailboxes 28 to 31 | | Receive FIFO |

Note 1. The transmit FIFO is controlled by TFCR. The MCTL_TX[j] registers associated with mailboxes 24 to 27 are disabled. MCTL_TX[24] to MCTL_TX[27] cannot be used by the transmit FIFO.
Note 2. The receive FIFO is controlled by RFCR. The MCTL_RX[j] registers associated with mailboxes 28 to 31 are disabled. MCTL_RX[28] to MCTL_RX[31] cannot be used by the receive FIFO.
Note 3. See the MIER_FIFO description for information on the FIFO interrupts.
Note 4. The bits in MKIVLR associated with mailboxes 24 to 31 are disabled. Set these bits to 0.
Note 5. The transmit and receive FIFOs can be used for both data frames and remote frames.

## 29.2.2 BCR : Bit Configuration Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x844

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | TSEG1[3:0] | | | | — | — | BRP[9:0] | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | SJW[1:0] | | — | TSEG2[2:0] | | | — | — | — | — | — | — | — | CCLKS |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CCLKS | CAN Clock Source Selection<br>0: PCLKB (generated by the PLL clock)<br>1: CANMCLK (generated by the main clock oscillator) | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 10:8 | TSEG2[2:0] | Time Segment 2 Control<br>0 0 0: Setting prohibited<br>0 0 1: 2 Tq<br>0 1 0: 3 Tq<br>0 1 1: 4 Tq<br>1 0 0: 5 Tq<br>1 0 1: 6 Tq<br>1 1 0: 7 Tq<br>1 1 1: 8 Tq | R/W |
| 11 | — | This bit is read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 13:12 | SJW[1:0] | Synchronization Jump Width Control<br>0 0: 1 Tq<br>0 1: 2 Tq<br>1 0: 3 Tq<br>1 1: 4 Tq | R/W |
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |
| 25:16 | BRP[9:0] | Baud Rate Prescaler Select*1<br>These bits set the frequency of the CAN communication clock (fCANCLK). | R/W |
| 27:26 | — | These bits are read as 0. The write value should be 0. | R/W |
| 31:28 | TSEG1[3:0] | Time Segment 1 Control<br>0x3: 4 Tq<br>0x4: 5 Tq<br>0x5: 6 Tq<br>0x6: 7 Tq<br>0x7: 8 Tq<br>0x8: 9 Tq<br>0x9: 10 Tq<br>0xA: 11 Tq<br>0xB: 12 Tq<br>0xC: 13 Tq<br>0xD: 14 Tq<br>0xE: 15 Tq<br>0xF: 16 Tq<br>Others: Setting prohibited | R/W |

Note: Tq: Time Quantum
Note 1. Do not select a value less than or equal to 1 while the SCKSCR.CKSEL[2:0] bits are 011b (selecting the main clock oscillator).

For details about setting the bit timing, see section 29.4. Data Transfer Rate Configuration. Set BCR before entering CAN halt or CAN operation mode from CAN reset mode. After the setting is made once, this register can be written to in CAN reset or CAN halt mode. A 32-bit read/write accesses must be performed carefully so as not to change bits [7:0].

### CCLKS bit (CAN Clock Source Selection)

The CCLKS bit selects CAN clock source. When the CCLKS bit is 0, the peripheral module clock (PCLKB) produced by the PLL frequency synthesizer is used as the CAN clock source (fCAN). When the CCLKS bit is 1, CANMCLK produced externally by the EXTAL pins is used as the CAN clock source (fCAN).

### TSEG2[2:0] bits (Time Segment 2 Control)

The TSEG2[2:0] bits specify the length of the phase buffer segment 2 (PHASE_SEG2) with a Tq value. A value from 2 to 8 Tq can be set. Set a value smaller than that in the TSEG1[3:0] bits.

### SJW[1:0] bits (Synchronization Jump Width Control)

The SJW[1:0] bits specify the synchronization jump width with a Tq value. A value from 1 to 4 Tq can be set. Set a value smaller than or equal to that in the TSEG2[2:0] bits.

### BRP[9:0] bits (Baud Rate Prescaler Select)

The BRP[9:0] bits set the frequency of the CAN communication clock (fCANCLK). The fCANCLK cycle is 1 Tq. If the setting is P (0 to 1023), the baud rate prescaler divides fCAN by P + 1.

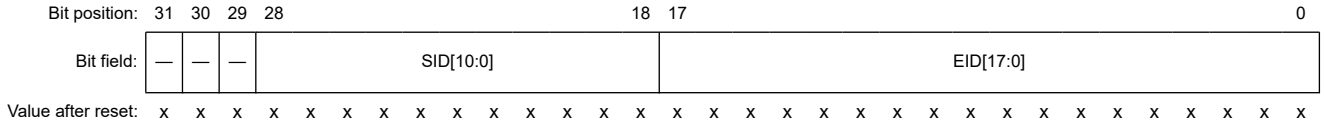
### TSEG1[3:0] bits (Time Segment 1 Control)

The TSEG1[3:0] bits specify the total length of the propagation time segment (PROP_SEG) and phase buffer segment 1 (PHASE_SEG1) with a time quantum (Tq) value. A value from 4 to 16 Tq can be set.

## 29.2.3 MKR[k] : Mask Register k (k = 0 to 7)

Base address: CAN0 = 0x400A_8000

Offset address: 0x400 + 0x04 × k




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 17:0 | EID[17:0] | Extended ID<br>0: Do not compare associated EID[17:0] bits<br>1: Compare associated EID[17:0] bits | R/W |
| 28:18 | SID[10:0] | Standard ID<br>0: Do not compare associated SID[10:0] bits<br>1: Compare associated SID[10:0] bits | R/W |
| 31:29 | — | The read values are undefined. The write value should be 0. | R/W |

For the mask function in FIFO mailbox mode, see section 29.6. Acceptance Filtering and Masking Functions.

Write to MKR0 to MKR7 in CAN reset mode or CAN halt mode.

### EID[17:0] bits (Extended ID)

The EID[17:0] bits are the filter mask bits associated with the CAN extended ID bits. They are used to receive extended ID messages. When an EID[17:0] bit is set to 0, the received ID is not compared with the associated mailbox ID. When an EID[17:0] bit is set to 1, the received ID is compared with the associated mailbox ID.

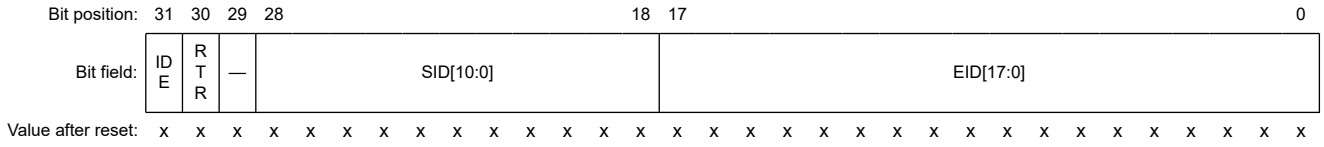
### SID[10:0] bits (Standard ID)

The SID[10:0] bits are the filter mask bits associated with the CAN standard ID bits. They are used to receive both standard ID and extended ID messages. When an SID[10:0] bit is set to 0, the received ID is not compared with the associated mailbox ID. When an SID[10:0] bit is set to 1, the received ID is compared with the associated mailbox ID.

## 29.2.4 FIDCRk : FIFO Received ID Compare Register k (k = 0, 1)

Base address: CAN0 = 0x400A_8000

Offset address: 0x420 + 0x04 × k




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 17:0 | EID[17:0] | Extended ID of data and remote frames | R/W |
| 28:18 | SID[10:0] | Standard ID of data and remote frames | R/W |
| 29 | — | The read value is undefined. The write value should be 0. | R/W |
| 30 | RTR | Remote Transmission Request<br>0: Data frame<br>1: Remote frame | R/W |
| 31 | IDE | ID Extension*1<br>0: Standard ID<br>1: Extended ID | R/W |

Note 1. The IDE bit is enabled when the CTLR.IDFM[1:0] bits are 10b (mixed ID mode). When the IDFM[1:0] bits are not 10b, only write 0 to IDE. It reads as 0.

FIDCR0 and FIDCR1 are enabled when the MBM bit in CTLR is set to 1 (FIFO mailbox mode). In this mode, the EID[17:0], SID[10:0], RTR, and IDE bits in mailboxes 28 to 31 are disabled. Write to FIDCR0 and FIDCR1 in CAN reset mode or CAN halt mode. For information on using FIDCR0 and FIDCR1, see section 29.6. Acceptance Filtering and Masking Functions.

#### EID[17:0] bits (Extended ID of data and remote frames)

The EID[17:0] bits set the extended ID of data frames and remote frames. These bits are used to receive extended ID messages.

#### SID[10:0] bits (Standard ID of data and remote frames)

The SID[10:0] bits set the standard ID of data frames and remote frames. These bits are used to receive both standard ID and extended ID messages.

#### RTR bit (Remote Transmission Request)

The RTR bit sets the frame format to data frames or remote frames:

- When the RTR bits in both FIDCR0 and FIDCR1 registers are set to 0, only data frames are received
- When the RTR bits in both FIDCR0 and FIDCR1 registers are set to 1, only remote frames are received
- When the RTR bits in both FIDCR0 and FIDCR1 registers are set to different values, both data frames and remote frames are received

#### IDE bit (ID Extension)

The IDE bit sets the ID format to standard ID or extended ID. The IDE bit is enabled when the IDFM[1:0] bits in CTLR are 10b (mixed ID mode):

- When the IDE bits in both FIDCR0 and FIDCR1 registers are set to 0, only standard ID frames are received
- When the IDE bits in both FIDCR0 and FIDCR1 registers are set to 1, only extended ID frames are received
- When the IDE bits in both FIDCR0 and FIDCR1 registers are set to different values, both standard ID and extended ID frames are received

### 29.2.5 MKIVLR : Mask Invalid Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x428

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MB31 | MB30 | MB29 | MB28 | MB27 | MB26 | MB25 | MB24 | MB23 | MB22 | MB21 | MB20 | MB19 | MB18 | MB17 | MB16 |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MB15 | MB14 | MB13 | MB12 | MB11 | MB10 | MB9 | MB8 | MB7 | MB6 | MB5 | MB4 | MB3 | MB2 | MB1 | MB0 |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | MB31 to MB0 | Mask Invalid<br>0: Mask valid<br>1: Mask invalid | R/W |

Each bit in MKIVLR is associated with a mailbox of the same number. Bit [0] in MKIVLR corresponds to mailbox 0 (MB0), and bit [31] corresponds to mailbox 31 (MB31).*1

When a bit is set to 1, the associated acceptance mask register becomes invalid for the associated mailbox. When a mask invalid bit is set to 1, a message is received by the associated mailbox only if the receive message ID matches the mailbox ID exactly.

Write to MKIVLR in CAN reset or halt mode.

Note 1. Set bits [31:24] to 0 in FIFO mailbox mode.

## 29.2.6 Mailbox Registers

Table 29.4 lists the CANi mailbox memory mapping, and Table 29.5 lists the CAN data frame configuration.

The value of the CANi mailbox after reset is undefined.

Write to MBj_ID, MBj_DL, MBj_Dm, and MBj_TS only when the related MCTL_TX[j] or MCTL_RX[j] register (j = 0 to 31) is 0x00 and the associated mailbox is not processing an abort request.

See Table 29.4 for detailed on register addresses.

**Table 29.4 CANi mailbox memory mapping (i = 0)**

| Address | Message content |
|---|---|
| CANn (n = 0) | Memory mapping |
| 0x4005_0200 + 1000 × n + 16 × j + 0 | IDE, RTR, SID10 to SID6 |
| 0x4005_0200 + 1000 × n + 16 × j + 1 | SID5 to SID0, EID17, EID16 |
| 0x4005_0200 + 1000 × n + 16 × j + 2 | EID15 to EID8 |
| 0x4005_0200 + 1000 × n + 16 × j + 3 | EID7 to EID0 |
| 0x4005_0200 + 1000 × n + 16 × j + 4 | — |
| 0x4005_0200 + 1000 × n + 16 × j + 5 | Data length code (DLC[3:0]) |
| 0x4005_0200 + 1000 × n + 16 × j + 6 | Data byte 0 |
| 0x4005_0200 + 1000 × n + 16 × j + 7 | Data byte 1 |
| 0x4005_0200 + 1000 × n + 16 × j + 8 | Data byte 2 |
| 0x4005_0200 + 1000 × n + 16 × j + 9 | Data byte 3 |
| 0x4005_0200 + 1000 × n + 16 × j + 10 | Data byte 4 |
| 0x4005_0200 + 1000 × n + 16 × j + 11 | Data byte 5 |
| 0x4005_0200 + 1000 × n + 16 × j + 12 | Data byte 6 |
| 0x4005_0200 + 1000 × n + 16 × j + 13 | Data byte 7 |
| 0x4005_0200 + 1000 × n + 16 × j + 14 | Time stamp upper byte |
| 0x4005_0200 + 1000 × n + 16 × j + 15 | Time stamp lower byte |

**Table 29.5 CAN data frame configuration**

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| SID10 to SID6 | SID5 to SID0 | EID17 to EID16 | EID15 to EID8 | EID7 to EID0 | DLC3 to DLC1 | DATA0 | DATA1 | … | DATA7 |

The previous value of each mailbox is retained unless a new message is received.

### 29.2.6.1 MBj_ID : Mailbox ID Register j (j = 0 to 31)

Base address: CAN0 = 0x400A_8000

Offset address: 0x200 + 0x10 × j

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | IDE | RTR | — | SID[10:0] | | | | | | | | | | | EID[17:0] | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | EID[17:0] | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 17:0 | EID[17:0] | Extended ID of data and remote frames*1 | R/W |
| 28:18 | SID[10:0] | Standard ID of data and remote frames | R/W |
| 29 | — | The read value is undefined. The write value should be 0. | R/W |
| 30 | RTR | Remote Transmission Request<br>0: Data frame<br>1: Remote frame | R/W |
| 31 | IDE | ID Extension*2<br>0: Standard ID<br>1: Extended ID | R/W |

Note 1. If the mailbox receives a standard ID message, the EID bits in the mailbox are undefined.
Note 2. The IDE bit is enabled when the CTLR.IDFM[1:0] bits are 10b (mixed ID mode). When the IDFM[1:0] bits are not 10b, only write 0 to IDE. It reads as 0.

#### EID[17:0] bits (Extended ID of data and remote frames)

The EID[17:0] bits set the extended ID of data frames and remote frames. They are used to transmit or receive extended ID messages.

#### SID[10:0] bits (Standard ID of data and remote frames)

The SID[10:0] bits set the standard ID of data frames and remote frames. They are used to transmit or receive both standard ID and extended ID messages.

#### RTR bit (Remote Transmission Request)

The RTR bit sets the frame format of data frames or remote frames.

- The receive mailbox only receives frames with the format specified in the RTR bit
- The transmit mailbox transmits with the frame format specified in the RTR bit
- The receive FIFO mailbox receives the data frame, remote frame, or both frames specified in the RTR bit in FIDCR0 and FIDCR1
- The transmit FIFO mailbox transmits the data frame or remote frame specified in the RTR bit in the transmit message

#### IDE bit (ID Extension)

The IDE bit sets the ID format to standard IDs or extended IDs. The IDE bit is enabled when the IDFM[1:0] bits in CTLR are 10b (mixed ID mode).

- The receive mailbox only receives the ID format specified in the IDE bit
- The transmit mailbox transmits with the ID format specified in the IDE bit
- The receive FIFO mailbox receives messages with the standard ID and extended ID settings specified in the IDE bit in FIDCR0 and FIDCR1
- The transmit FIFO mailbox transmits messages with the standard ID or extended ID settings specified in the IDE bit in the transmit message

### 29.2.6.2 MBj_DL : Mailbox Data Length Register j (j = 0 to 31)

Base address: CAN0 = 0x400A_8000

Offset address: 0x204 + 0x10 × j

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | DLC[3:0] | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | DLC[3:0] | Data Length Code*1<br>0x0: Data length = 0 byte<br>0x1: Data length = 1 byte<br>0x2: Data length = 2 bytes<br>0x3: Data length = 3 bytes<br>0x4: Data length = 4 bytes<br>0x5: Data length = 5 bytes<br>0x6: Data length = 6 bytes<br>0x7: Data length = 7 bytes<br>Others: Data length = 8 bytes | R/W |
| 15:4 | — | The read values are undefined. The write value should be 0. | R/W |

Note 1. If the mailbox receives a message with data length (set in DLC[3:0]) of n bytes, where n is less than 8, the data in DATAn to DATA7 registers in the mailbox is undefined. DATA0 to DATA7 are data registers for this mailbox. For example, if data length is 6 bytes (DLC[3:0] = 0x6), the data in DATA6 and DATA7 registers is undefined.

**DLC[3:0] bits (Data Length Code)**

The DLC[3:0] bits specify the data length to be transmitted in data frames. When a remote frame is used to request data, this field specifies the requested data length.

When a data frame is received, the received data length is stored in this field. When a remote frame is received, this field stores the requested data length.

### 29.2.6.3 MBj_Dm : Mailbox Data Register j (j = 0 to 31, m = 0 to 7)

Base address: CAN0 = 0x400A_8000

Offset address: (0x206 + m) + 0x10 × j

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DATAm | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | DATA0 to DATA7 | Data Bytes 0 to 7*1 *2<br>DATA0 to DATA7 store the transmitted or received CAN message data. Transmission or reception starts from DATA0. The bit order on the CAN bus is MSB-first, and transmission or reception starts from bit [7]. | R/W |

Note 1. If the mailbox receives a message with n bytes, where n is less than 8 bytes, the values of DATAn to DATA7 in the mailbox are undefined.

Note 2. If the mailbox receives a remote frame, the previous values of DATA0 to DATA7 in the mailbox are saved.

### 29.2.6.4 MBj_TS : Mailbox Time Stamp Register j (j = 0 to 31)

Base address: CAN0 = 0x400A_8000

Offset address: 0x20E + 0x10 × j

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | TSH[7:0] | | | | | | | | TSL[7:0] | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | TSL[7:0] | Time Stamp Lower Byte<br>The TSH[7:0] and TSL[7:0] bits store the counter value of the time stamp when received messages are stored in the mailbox. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:8 | TSH[7:0] | Time Stamp Higher Byte<br>The TSH[7:0] and TSL[7:0] bits store the counter value of the time stamp when received messages are stored in the mailbox. | R/W |

## 29.2.7 MIER : Mailbox Interrupt Enable Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x42C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MB31 | MB30 | MB29 | MB28 | MB27 | MB26 | MB25 | MB24 | MB23 | MB22 | MB21 | MB20 | MB19 | MB18 | MB17 | MB16 |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MB15 | MB14 | MB13 | MB12 | MB11 | MB10 | MB9 | MB8 | MB7 | MB6 | MB5 | MB4 | MB3 | MB2 | MB1 | MB0 |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | MB31 to MB0 | Interrupt Enable<br>Bit [31] is associated with mailbox 31 (MB31), and bit [0] with mailbox 0 (MB0).<br>0: Disable interrupt<br>1: Enable interrupt | R/W |

MIER can enable interrupts for each mailbox independently. This register is available in normal mailbox mode. Do not access this register in FIFO mailbox mode.

Each bit is associated with the mailbox having the same number. These bits enable or disable transmission and reception complete interrupts for the associated mailboxes as follows:

- Bit [0] in MIER corresponds to mailbox 0 (MB0)
- Bit [31] in MIER corresponds to mailbox 31 (MB31)

Write to MIER only when the associated MCTL_TX[j] or MCTL_RX[j] register (j = 0 to 31) is 0x00 and the associated mailbox is not processing a transmission or reception abort request.

## 29.2.8 MIER_FIFO : Mailbox Interrupt Enable Register for FIFO Mailbox Mode

Base address: CAN0 = 0x400A_8000

Offset address: 0x42C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | MB29 | MB28 | — | — | MB25 | MB24 | MB23 | MB22 | MB21 | MB20 | MB19 | MB18 | MB17 | MB16 |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | MB15 | MB14 | MB13 | MB12 | MB11 | MB10 | MB9 | MB8 | MB7 | MB6 | MB5 | MB4 | MB3 | MB2 | MB1 | MB0 |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 23:0 | MB23 to MB0 | Interrupt Enable<br>Bit [23] is associated with mailbox 23 (MB23), and bit [0] with mailbox 0 (MB0).<br>0: Disable interrupt<br>1: Enable interrupt | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 24 | MB24 | Transmit FIFO Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 25 | MB25 | Transmit FIFO Interrupt Generation Timing Control<br>0: Generate every time transmission completes<br>1: Generate when the transmit FIFO empties on transmission completion | R/W |
| 27:26 | — | The read values are undefined. The write value should be 0. | R/W |
| 28 | MB28 | Receive FIFO Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 29 | MB29 | Receive FIFO Interrupt Generation Timing Control*1<br>0: Generate every time reception completes<br>1: Generate when the receive FIFO becomes a buffer warning*2 on reception completion | R/W |
| 31:30 | — | The read values are undefined. The write value should be 0. | R/W |

Note 1. No interrupt request occurs when the receive FIFO becomes buffer warning because it is full.
Note 2. Buffer warning indicates a state in which the third message is stored in the receive FIFO.

MIER_FIFO can independently enable interrupts for each mailbox and FIFO. This register is available in FIFO mailbox mode. Do not access it in normal mailbox mode.

The MB0 to MB23 bits are associated with the mailbox of the same number. These bits enable or disable transmission and reception complete interrupts for the associated mailboxes:

- Bit [0] in MIER_FIFO is associated with mailbox 0 (MB0)
- Bit [23] in MIER_FIFO is associated with mailbox 23 (MB23)

MB24, MB25, MB28, and MB29 specify whether transmit and receive FIFO interrupts are enabled or disabled, and the timing of interrupt requests.

Write to MIER_FIFO only when the associated MCTL_TX[j] or MCTL_RX[j] register (j = 0 to 31) is 0x00 and the associated mailbox is not processing a transmission or reception abort request. In addition, change the bits in MIER_FIFO for the associated FIFO only when all of the following conditions are true:

- The TFE bit in TFCR is 0 and the TFEST bit is 1.
- The RFE bit in RFCR is 0 and the RFEST bit in RFCR is 1.

## 29.2.9 MCTL_TX[j] : Message Control Register for Transmit (j = 0 to 31)

Base address: CAN0 = 0x400A_8000

Offset address: 0x820 + 0x01 × j

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TRMREQ | RECREQ | — | ONESHOT | — | TRMABT | TRMACTIVE | SENTDATA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SENTDATA | Transmission Complete Flag*1 *2<br>0: Transmission not complete<br>1: Transmission complete | R/W |
| 1 | TRMACTIVE | Transmission-in-Progress Status Flag<br>0: Transmission pending or not requested<br>1: Transmission in progress | R |
| 2 | TRMABT | Transmission Abort Complete Flag*1 *2<br>0: Transmission started, transmission abort failed because transmission completed, or transmission abort not requested<br>1: Transmission abort complete | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | ONESHOT | One-Shot Enable*2 *3<br>0: Disable one-shot transmission<br>1: Enable one-shot transmission | R/W |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | RECREQ | Receive Mailbox Request*2 *3 *4 *5<br>0: Do not configure for reception<br>1: Configure for reception | R/W |
| 7 | TRMREQ | Transmit Mailbox Request*2 *4<br>0: Do not configure for transmission<br>1: Configure for transmission | R/W |

Note 1. Write 0 only. Writing 1 has no effect.
Note 2. When writing to bits of this register, write 1 to SENTDATA and TRMABT flags if these bits are not the write target.
Note 3. To enter one-shot transmit mode, write 1 to the ONESHOT bit at the same time as setting the TRMREQ bit to 1. To exit one-shot transmit mode, write 0 to the ONESHOT bit after the message is transmitted or aborted.
Note 4. Do not set both the RECREQ and TRMREQ bits to 1.
Note 5. When setting the RECREQ bit to 0, set the SENTDATA, TRMACTIVE, and TRMABT flags to 0 simultaneously.

MCTL_TX[j] sets mailbox j to transmit mode or receive mode. In transmit mode, MCTL_TX[j] also controls and indicates the transmission status. Do not access MCTL_TX[j] if mailbox j is in receive mode. Only write to MCTL_TX[j] in CAN operation or halt mode. Do not use MCTL_TX24 to MCTL_TX31 in FIFO mailbox mode.

#### SENTDATA flag (Transmission Complete Flag)

The SENTDATA flag is set to 1 when data transmission from the associated mailbox is complete. This flag is set to 0 through a software write. To set this flag to 0, first set the TRMREQ bit to 0. The SENTDATA flag and the TRMREQ bit cannot be set to 0 simultaneously. To transmit a new message from the associated mailbox, set the SENTDATA flag to 0.

#### TRMACTIVE flag (Transmission-in-Progress Status Flag)

The TRMACTIVE flag is set to 1 when the associated mailbox of the CAN module begins to transmit a message. It is set to 0 when the CAN module loses the CAN bus arbitration, when a CAN bus error occurs, or when data transmission completes.

#### TRMABT flag (Transmission Abort Complete Flag)

The TRMABT flag is set to 1 in the following cases:

- Following a transmission abort request, when the transmission abort is complete before starting transmission
- Following a transmission abort request, when the CAN module detects CAN bus arbitration-lost or a CAN bus error
- In one-shot transmission mode (RECREQ = 0, TRMREQ = 1, and ONESHOT = 1), when the CAN module detects a CAN bus arbitration-lost or a CAN bus error.

The TRMABT flag is not set to 1 when data transmission is complete. The SENTDATA flag is set to 1, and the TRMABT flag is set to 0 through a software write.

#### ONESHOT bit (One-Shot Enable)

When the ONESHOT bit is set to 1 in transmit mode (RECREQ = 0 and TRMREQ = 1), the CAN module transmits a message only one time. The CAN module does not transmit the message again if a CAN bus error or CAN bus arbitration-lost occurs. When transmission is complete, the SENTDATA flag is set to 1. If transmission does not complete because of a CAN bus error or CAN bus arbitration-lost error, the TRMABT flag is set to 1. Set the ONESHOT bit to 0 after the SENTDATA or TRMABT flag is set to 1.

#### RECREQ bit (Receive Mailbox Request)

The RECREQ bit selects the receive modes listed in Table 29.10.

When the RECREQ bit is set to 1, the associated mailbox is configured for reception of a data or remote frame.

When the RECREQ bit is set to 0, the associated mailbox is not configured for reception of a data or remote frame.

Due to hardware protection, the RECREQ bit cannot be set to 0 through a software write during the following period:

- Hardware protection is started from the acceptance filter processing (the beginning of the CRC field)
- Hardware protection is released:
  - For the mailbox that is specified to receive the incoming message, after the received data is stored in the mailbox or a CAN bus error occurs. This means that the maximum period of hardware protection is from the beginning of CRC field to the end of the 7th bit of EOF.
  - For the other mailboxes, after acceptance filter processing
  - If no mailbox is specified to receive the message, after acceptance filter processing.

When setting the RECREQ bit to 1, do not set the TRMREQ bit to 1. To change the configuration of a mailbox from transmission to reception, first abort the transmission, then set the SENTDATA and TRMABT flags to 0 before changing to reception.

Note: MCTL_TX[j].RECREQ is the mirror bit of MCTL_RX[j].RECREQ.

**TRMREQ bit (Transmit Mailbox Request)**

The TRMREQ bit selects the transmit modes listed in Table 29.10.

When the TRMREQ bit is set to 1, the associated mailbox is configured for transmission of a data or remote frame.

When the TRMREQ bit is set to 0, the associated mailbox is not configured for transmission of a data or remote frame.

If the TRMREQ bit is changed from 1 to 0 to cancel the associated transmission request, either the TRMABT or SENTDATA flag is set to 1. When setting the TRMREQ bit to 1, do not set the RECREQ bit to 1. To change the configuration of a mailbox from reception to transmission, first abort the reception, then set the NEWDATA and MSGLOST flags to 0 before changing to transmission.

Note: MCTL_TX[j].TRMREQ is the mirror bit of MCTL_RX[j].TRMREQ.

## 29.2.10 MCTL_RX[j] : Message Control Register for Receive (j = 0 to 31)

Base address: CAN0 = 0x400A_8000

Offset address: 0x820 + 0x01 × j

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TRMREQ | RECREQ | — | ONESHOT | — | MSGLOST | INVALDATA | NEWDATA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | NEWDATA | Reception Complete Flag*1 *2<br>0: No data received, or 0 was written to the flag<br>1: New message being stored or was stored in the mailbox | R/W |
| 1 | INVALDATA | Reception-in-Progress Status Flag<br>0: Message valid<br>1: Message being updated | R |
| 2 | MSGLOST | Message Lost Flag*1 *2<br>0: Message not overwritten or overrun<br>1: Message overwritten or overrun | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | ONESHOT | One-Shot Enable*2 *3<br>0: Disable one-shot reception<br>1: Enable one-shot reception | R/W |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | RECREQ | Receive Mailbox Request*2 *3 *4 *5<br>0: Do not configure for reception<br>1: Configure for reception | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | TRMREQ | Transmit Mailbox Request*2 *4<br>0: Do not configure for transmission<br>1: Configure for transmission | R/W |

Note 1. Write 0 only. Writing 1 has no effect.
Note 2. When writing to bits of this register, write 1 to the NEWDATA and MSGLOST flags if they are not the write target.
Note 3. To enter one-shot receive mode, write 1 to the ONESHOT bit at the same time as setting the RECREQ bit to 1. To exit one-shot receive mode, write 0 to the ONESHOT bit after writing 0 to the RECREQ bit and confirming that it is 0.
Note 4. Do not set both the RECREQ and TRMREQ bits to 1.
Note 5. When setting the RECREQ bit to 0, set MSGLOST, NEWDATA, and RECREQ to 0 simultaneously.

MCTL_RX[j] sets mailbox j to transmit or receive mode. In receive mode, MCTL_RX[j] also controls and indicates the reception status. Do not access MCTL_RX[j] if mailbox j is in transmit mode. Only write to the MCTL_RX[j] in CAN operation or halt mode. Do not use MCTL_RX24 to MCTL_RX31 in FIFO mailbox mode.

### NEWDATA flag (Reception Complete Flag)

The NEWDATA flag is set to 1 when a new message is being stored or was stored in the mailbox. Always set this bit to 1 simultaneously with the INVALDATA flag. The NEWDATA flag is set to 0 through a software write. The NEWDATA flag cannot be set to 0 through a software write while the associated INVALDATA flag is 1.

### INVALDATA flag (Reception-in-Progress Status Flag)

After the completion of a message reception, the INVALDATA flag is set to 1 while the received message is updated in the associated mailbox. The INVALDATA flag is set to 0 immediately after the message is stored. If the mailbox is read while the INVALDATA flag is 1, the data is undefined.

### MSGLOST flag (Message Lost Flag)

The MSGLOST flag is set to 1 when the mailbox is overwritten or overrun by a new received message while the NEWDATA flag is 1. The MSGLOST flag is set to 1 at the end of the 6th bit of EOF. The MSGLOST flag is set to 0 through a software write.

In both overwrite and overrun modes, the MSGLOST flag cannot be set to 0 through a software write during the 5 PCLKB cycles following the 6th bit of EOF.

### ONESHOT bit (One-Shot Enable)

When the ONESHOT bit is set to 1 in receive mode (RECREQ = 1 and TRMREQ = 0), the mailbox receives a message only one time. The mailbox does not behave as a receive mailbox after having received a message one time. The behavior of the NEWDATA and INVALDATA flags is the same as in normal receive mode. In one-shot receive mode, the MSGLOST flag is not set to 1. To set the ONESHOT bit to 0, first write 0 to the RECREQ bit and ensure that it is 0.

### RECREQ bit (Receive Mailbox Request)

The RECREQ bit selects receive modes listed in Table 29.10.

When the RECREQ bit is set to 1, the associated mailbox is configured for reception of a data or remote frame.

When the RECREQ bit is set to 0, the associated mailbox is not configured for reception of a data or remote frame.

Due to hardware protection, the RECREQ bit cannot be set to 0 through a software write during the following period:

- Hardware protection is started from the acceptance filter processing (the beginning of the CRC field)
- Hardware protection is released:
  - For the mailbox that is specified to receive the incoming message, after the received data is stored in the mailbox or a CAN bus error occurs. This means that the maximum period of hardware protection is from the beginning of the CRC field to the end of the 7th bit of EOF.
  - For the other mailboxes, after acceptance filter processing
  - If no mailbox is specified to receive the message, after acceptance filter processing.

When setting the RECREQ bit to 1, do not set the TRMREQ bit to 1. To change the configuration of a mailbox from transmission to reception, first abort the transmission, then set the SENTDATA and TRMABT flags to 0 before changing to reception.

Note: MCTL_RX[j].RECREQ is the mirror bit of MCTL_TX[j].RECREQ.

### TRMREQ bit (Transmit Mailbox Request)

The TRMREQ bit selects the transmit modes listed in Table 29.10.

When the TRMREQ bit is set to 1, the associated mailbox is configured for transmission of a data or remote frame.

When the TRMREQ bit is set to 0, the associated mailbox is not configured for transmission of a data or remote frame.

If the TRMREQ bit is changed from 1 to 0 to cancel the associated transmission request, either the TRMABT or SENTDATA flag is set to 1. When setting the TRMREQ bit to 1, do not set the RECREQ bit to 1. To change the configuration of a mailbox from reception to transmission, first abort the reception, then set the NEWDATA and MSGLOST flags to 0 before changing to transmission.

Note: MCTL_RX[j].TRMREQ is the mirror bit of MCTL_TX[j].TRMREQ.

## 29.2.11 RFCR : Receive FIFO Control Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x848

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | RFEST | RFWST | RFFST | RFMLF | RFUST[2:0] | | | RFE |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RFE | Receive FIFO Enable<br>0: Disable receive FIFO<br>1: Enable receive FIFO | R/W |
| 3:1 | RFUST[2:0] | Receive FIFO Unread Message Number Status<br>0 0 0: No unread message<br>0 0 1: 1 unread message<br>0 1 0: 2 unread messages<br>0 1 1: 3 unread messages<br>1 0 0: 4 unread messages<br>1 0 1: Reserved<br>1 1 0: Reserved<br>1 1 1: Reserved | R |
| 4 | RFMLF | Receive FIFO Message Lost Flag<br>0: Receive FIFO message not lost<br>1: Receive FIFO message lost | R/W |
| 5 | RFFST | Receive FIFO Full Status Flag<br>0: Receive FIFO not full<br>1: Receive FIFO full (4 unread messages) | R |
| 6 | RFWST | Receive FIFO Buffer Warning Status Flag<br>0: Receive FIFO is not buffer warning<br>1: Receive FIFO is buffer warning (3 unread messages) | R |
| 7 | RFEST | Receive FIFO Empty Status Flag<br>0: Unread message in receive FIFO<br>1: No unread message in receive FIFO | R |

Write to RFCR in CAN operation or halt mode.

### RFE bit (Receive FIFO Enable)

When the RFE bit is set to 1, the receive FIFO is enabled.

When the RFE bit is set to 0, the receive FIFO is disabled for reception and becomes empty (RFEST bit = 1). Write 0 to the RFE bit simultaneously with the RFMLF flag setting.

Do not set this bit to 1 in normal mailbox mode (MBM bit in CTLR = 0). Due to hardware protection, the RFE bit cannot be set to 0 through a software write during the following period:

- Hardware protection is started from acceptance filter processing (the beginning of the CRC field)
- When hardware protection is released:
  - If the receive FIFO is specified to receive the incoming message, after the received data is stored in the receive FIFO or a CAN bus error occurs. This means that the maximum period of hardware protection is from the beginning of the CRC field to the end of the 7th bit of EOF.
  - If the receive FIFO is not specified to receive the message, after acceptance filter processing.

#### RFUST[2:0] bits (Receive FIFO Unread Message Number Status)

The RFUST[2:0] bits indicate the number of unread messages in the receive FIFO. The value of the RFUST[2:0] bits are initialized to 000b when the RFE bit is set to 0.

#### RFMLF flag (Receive FIFO Message Lost Flag)

The RFMLF flag is set to 1 (receive FIFO message lost) when the receive FIFO receives a new message and is full. It is set to 1 at the end of the 6th bit of EOF.

The RFMLF flag is set to 0 through a software write. Writing 1 has no effect. In both overwrite and overrun modes, if the receive FIFO is full and determined to have received a message, the RFMLF flag cannot be set to 0 (no receive FIFO message lost) through a software write during the 5 PCLKB cycles following the 6th bit of EOF.

#### RFFST flag (Receive FIFO Full Status Flag)

The RFFST flag is set to 1 (receive FIFO is full) when the number of unread messages in the receive FIFO is 4. The flag is 0 (receive FIFO is not full) when the number of unread messages in the receive FIFO is less than 4. The flag is set to 0 when the RFE bit is 0.

#### RFWST flag (Receive FIFO Buffer Warning Status Flag)

The RFWST flag is set to 1 (receive FIFO is a buffer warning) when the number of unread messages in the receive FIFO is 3. The flag is 0 (receive FIFO is not a buffer warning) when the number of unread messages in the receive FIFO is less than 3 or equal to 4. The RFWST flag is set to 0 when the RFE bit is 0.

#### RFEST flag (Receive FIFO Empty Status Flag)

The RFEST flag is set to 1 (no unread message in receive FIFO) when the number of unread messages in the receive FIFO is 0. The flag is set to 1 when the RFE bit is set to 0. The flag is set to 0 (unread message in receive FIFO) when the number of unread messages in the receive FIFO is one or more.

Figure 29.2 shows the receive FIFO mailbox operation.

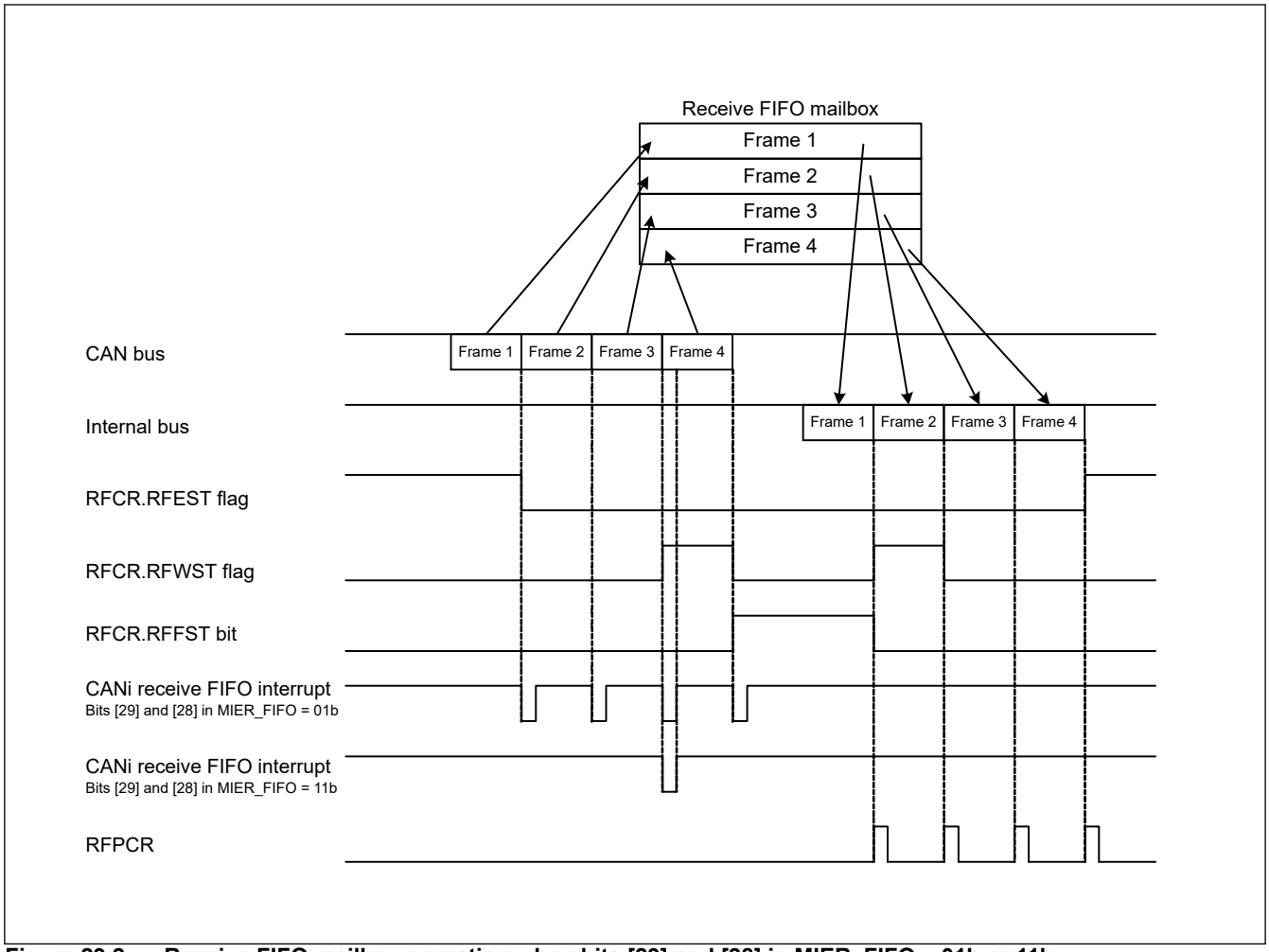



**Figure 29.2 Receive FIFO mailbox operation when bits [29] and [28] in MIER_FIFO = 01b or 11b**

## 29.2.12 RFPCR : Receive FIFO Pointer Control Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x849

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | The CPU pointer for the receive FIFO is incremented by writing 0xFF to RFPCR. | W |

When the receive FIFO is not empty, write 0xFF to RFPCR through software to increment the CPU pointer to the next mailbox location. Do not write to RFPCR when the RFE bit in RFCR is 0 (receive FIFO disabled).

Both the CAN and CPU pointers are incremented when a new message is received and the RFFST flag is 1 (receive FIFO is full) in overwrite mode. When the RFMLF flag is 1 in this condition, the CPU pointer does not increment on a software write to RFPCR.

## 29.2.13 TFCR : Transmit FIFO Control Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x84A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TFEST | TFFST | — | — | TFUST[2:0] | | | TFE |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TFE | Transmit FIFO Enable<br>0: Disable transmit FIFO<br>1: Enable transmit FIFO | R/W |
| 3:1 | TFUST[2:0] | Transmit FIFO Unsent Message Number Status<br>0 0 0: 0 unsent messages<br>0 0 1: 1 unsent message<br>0 1 0: 2 unsent messages<br>0 1 1: 3 unread messages<br>1 0 0: 4 unread messages<br>1 0 1: Reserved<br>1 1 0: Reserved<br>1 1 1: Reserved | R |
| 5:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | TFFST | Transmit FIFO Full Status<br>0: Transmit FIFO not full<br>1: Transmit FIFO full (4 unsent messages) | R |
| 7 | TFEST | Transmit FIFO Empty Status<br>0: Unsent message in transmit FIFO<br>1: No unsent message in transmit FIFO | R |

Write to TFCR in CAN operation or halt mode.

### TFE bit (Transmit FIFO Enable)

Setting the TFE bit set to 1 enables the transmit FIFO. Setting the TFE bit to 0 empties the transmit FIFO (TFEST bit = 1), and unsent messages in the transmit FIFO are lost in the following ways:

- Immediately if a message from the transmit FIFO is not scheduled for the next transmission or is already in transmission
- On completion of transmission, on a CAN bus error, CAN bus arbitration-lost, or entry to CAN halt mode, if a message from the transmit FIFO is scheduled for the next transmission or is already in transmission.

Before setting the TFE bit to 1 again, ensure that the TFEST bit is set to 1. After setting the TFE bit to 1, write transmit data to mailbox 24.

Do not set the TFE bit to 1 in normal mailbox mode (MBM bit in CTLR = 0).

### TFUST[2:0] bits (Transmit FIFO Unsent Message Number Status)

The TFUST[2:0] bits indicate the number of unsent messages in the transmit FIFO. These bits are set to 000b after the TFE bit is set to 0 and transmission is aborted or completes.

### TFFST bit (Transmit FIFO Full Status)

The TFFST bit is set to 1 (transmit FIFO is full) when the number of unsent messages in the transmit FIFO is 4. The TFFST bit is set to 0 (transmit FIFO is not full) when the number of unsent messages in the transmit FIFO is less than 4. The TFFST bit is set to 0 when transmission from the transmit FIFO is aborted.

### TFEST bit (Transmit FIFO Empty Status)

The TFEST bit is set to 1 (no message in transmit FIFO) when the number of unsent messages in the transmit FIFO is 0. The TFEST bit is set to 1 when transmission from the transmit FIFO is aborted. The TFEST bit is set to 0 (message in transmit FIFO) when the number of unsent messages in the transmit FIFO is not 0.

Figure 29.3 shows the transmit FIFO mailbox operation.

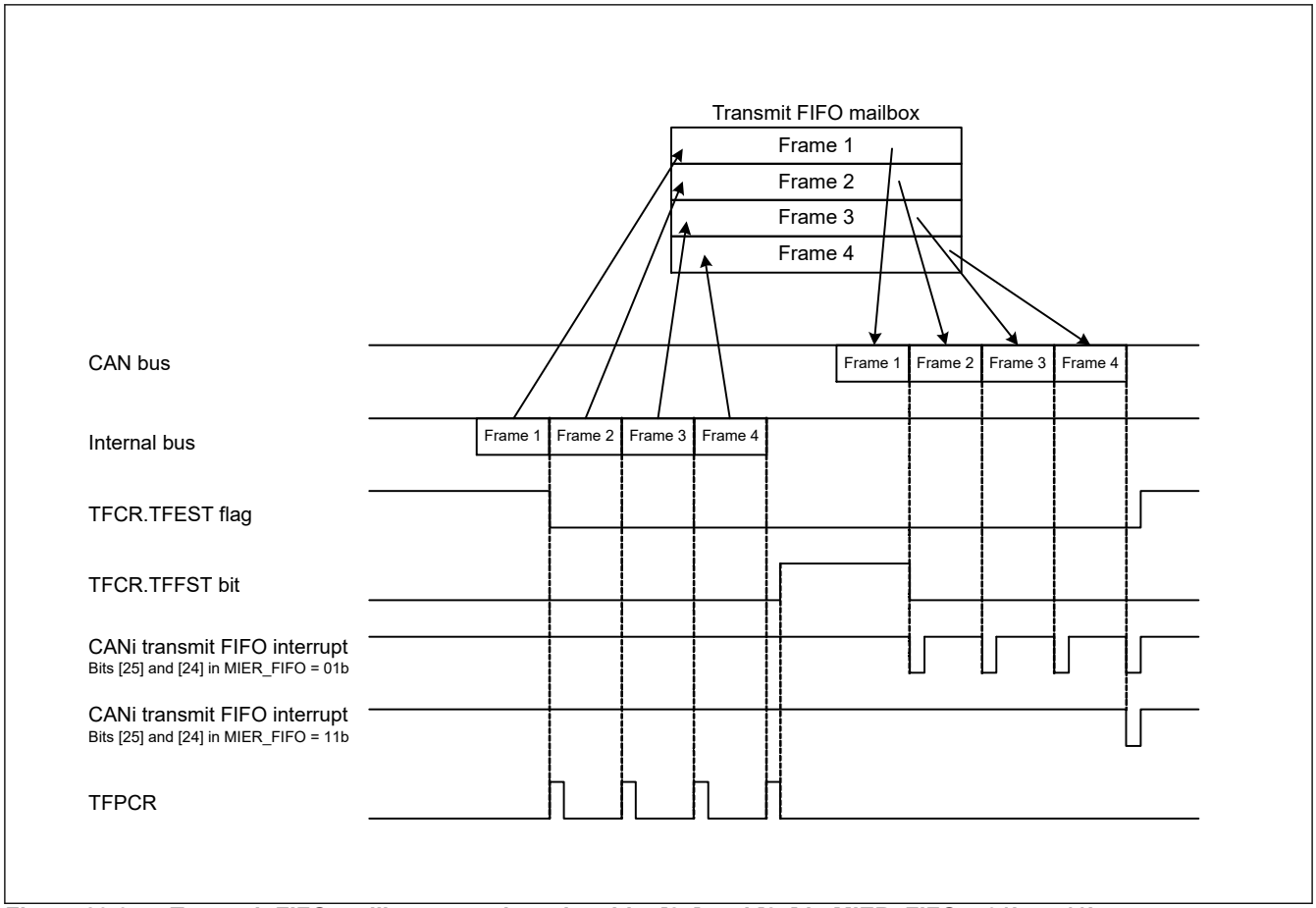



**Figure 29.3 Transmit FIFO mailbox operation when bits [25] and [24] in MIER_FIFO = 01b or 11b**

## 29.2.14 TFPCR : Transmit FIFO Pointer Control Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x84B

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | The CPU pointer for the transmit FIFO is incremented by writing 0xFF to TFPCR. | W |

When the transmit FIFO is not full, write 0xFF to TFPCR through software to increment the CPU pointer for the transmit FIFO to the next mailbox location.

Do not write to TFPCR when the TFE bit in TFCR is 0 (transmit FIFO disabled).

## 29.2.15 STR : Status Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x842

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | RECST | TRMST | BOST | EPST | SLPST | HLTST | RSTST | EST | TABST | FMLST | NMLST | TFST | RFST | SDST | NDST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | NDST | NEWDATA Status Flag<br>0: No mailbox with NEWDATA = 1<br>1: One or more mailboxes with NEWDATA = 1 | R |
| 1 | SDST | SENTDATA Status Flag<br>0: No mailbox with SENTDATA = 1<br>1: One or more mailboxes with SENTDATA = 1 | R |
| 2 | RFST | Receive FIFO Status Flag<br>0: Receive FIFO empty<br>1: Message in receive FIFO | R |
| 3 | TFST | Transmit FIFO Status Flag<br>0: Transmit FIFO is full<br>1: Transmit FIFO is not full | R |
| 4 | NMLST | Normal Mailbox Message Lost Status Flag<br>0: No mailbox with MSGLOST = 1<br>1: One or more mailboxes with MSGLOST = 1 | R |
| 5 | FMLST | FIFO Mailbox Message Lost Status Flag<br>0: RFMLF = 0<br>1: RFMLF = 1 | R |
| 6 | TABST | Transmission Abort Status Flag<br>0: No mailbox with TRMABT = 1<br>1: One or more mailboxes with TRMABT = 1 | R |
| 7 | EST | Error Status Flag<br>0: No error occurred<br>1: Error occurred | R |
| 8 | RSTST | CAN Reset Status Flag<br>0: Not in CAN reset mode<br>1: In CAN reset mode | R |
| 9 | HLTST | CAN Halt Status Flag<br>0: Not in CAN halt mode<br>1: In CAN halt mode | R |
| 10 | SLPST | CAN Sleep Status Flag<br>0: Not in CAN sleep mode<br>1: In CAN sleep mode | R |
| 11 | EPST | Error-Passive Status Flag<br>0: Not in error-passive state<br>1: In error-passive state | R |
| 12 | BOST | Bus-Off Status Flag<br>0: Not in bus-off state<br>1: In bus-off state | R |
| 13 | TRMST | Transmit Status Flag<br>0: Bus idle or reception in progress<br>1: Transmission in progress or module in bus-off state | R |
| 14 | RECST | Receive Status Flag<br>0: Bus idle or transmission in progress<br>1: Reception in progress | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15 | — | This bit is read as 0. | R |

### NDST flag (NEWDATA Status Flag)

The NDST flag is set to 1 when at least one NEWDATA flag in MCTL_RX[j] (j = 0 to 31) is 1, regardless of the value of MIER or MIER_FIFO. The NDST flag is set to 0 when all NEWDATA flags are 0.

### SDST flag (SENTDATA Status Flag)

The SDST flag is set to 1 when at least one SENTDATA flag in MCTL_TX[j] (j = 0 to 31) is 1, regardless of the value of MIER or MIER_FIFO. The SDST flag is set to 0 when all SENTDATA flags are 0.

### RFST flag (Receive FIFO Status Flag)

The RFST flag is set to 1 when the receive FIFO is not empty. The RFST flag is set to 0 when the receive FIFO is empty or normal mailbox mode is selected.

### TFST flag (Transmit FIFO Status Flag)

The TFST flag is set to 1 when the transmit FIFO is not full. The TFST flag is set to 0 when the transmit FIFO is full or normal mailbox mode is selected.

### NMLST flag (Normal Mailbox Message Lost Status Flag)

The NMLST flag is set to 1 when at least one MSGLOST flag in MCTL_RX[j] (j = 0 to 31) is 1, regardless of the value of MIER or MIER_FIFO. The NMLST flag is set to 0 when all MSGLOST flags are 0.

### FMLST flag (FIFO Mailbox Message Lost Status Flag)

The FMLST flag is set to 1 when the RFMLF flag in RFCR is 1, regardless of the value of MIER_FIFO. The FMLST flag is set to 0 when the RFMLF flag is 0.

### TABST flag (Transmission Abort Status Flag)

The TABST flag is set to 1 when at least one TRMABT flag in MCTL_TX[j] (j = 0 to 31) is 1, regardless of the value of MIER or MIER_FIFO. The TABST flag is set to 0 when all TRMABT flags are 0.

### EST flag (Error Status Flag)

The EST flag is set to 1 when at least one error is detected by EIFR, regardless of the value of EIER. The EST flag is set to 0 when no error is detected by EIFR.

### RSTST flag (CAN Reset Status Flag)

The RSTST flag is set to 1 when the CAN module is in CAN reset mode. The RSTST flag is 0 when the CAN module is not in CAN reset mode. Even when the state changes from CAN reset mode to CAN sleep mode, the flag remains 1.

### HLTST flag (CAN Halt Status Flag)

The HLTST flag is set to 1 when the CAN module is in CAN halt mode. The HLTST is set to 0 when the CAN module is not in CAN halt mode. Even when the state changes from CAN halt mode to CAN sleep mode, the flag remains 1.

### SLPST flag (CAN Sleep Status Flag)

The SLPST flag is set to 1 when the CAN module is in CAN sleep mode. The SLPST flag is set to 0 when the CAN module is not in CAN sleep mode.

### EPST flag (Error-Passive Status Flag)

The EPST flag is set to 1 when the value of TECR or RECR exceeds 127 and the CAN module is in the error-passive state ($128 \leq TEC < 256$ or $128 \leq REC < 256$). The EPST flag is set to 0 when the CAN module is not in the error-passive state.

### BOST flag (Bus-Off Status Flag)

The BOST flag is set to 1 when the value of TECR exceeds 255 and the CAN module is in the bus-off state ($TEC \geq 256$). The BOST flag is set to 0 when the CAN module is not in the bus-off state.

### TRMST flag (Transmit Status Flag)

The TRMST flag is set to 1 when the CAN module performs as a transmitter node or is in the bus-off state. The TRMST flag is set to 0 when the CAN module performs as a receiver node or is in the bus-idle state.

### RECST flag (Receive Status Flag)

The RECST flag is set to 1 when the CAN module performs as a receiver node. The RECST flag is set to 0 when the CAN module performs as a transmitter node or is in the bus-idle state.

## 29.2.16 MSMR : Mailbox Search Mode Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x853

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | MBSM[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | MBSM[1:0] | Mailbox Search Mode Select<br>0 0: Receive mailbox search mode<br>0 1: Transmit mailbox search mode<br>1 0: Message lost search mode<br>1 1: Channel search mode | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

Write to MSMR in CAN operation or halt mode.

### MBSM[1:0] bits (Mailbox Search Mode Select)

The MBSM[1:0] bits select the search mode for the mailbox search function.

When the MBSM[1:0] bits are 00b, receive mailbox search mode is selected. In this mode, the search targets are the NEWDATA flag in MCTL_RX[j] (j = 0 to 31) for the normal mailbox and the RFEST flag in RFCR.

When the MBSM[1:0] bits are 01b, transmit mailbox search mode is selected. In this mode, the search target is the SENTDATA flag in MCTL_TX[j].

When the MBSM[1:0] bits are 10b, message lost search mode is selected. In this mode, the search targets are the MSGLOST flag in MCTL_RX[j] for the normal mailbox and the RFMLF flag in RFCR.

When the MBSM[1:0] bits are 11b, channel search mode is selected. In this mode, the search target is CSSR. See section 29.2.18. CSSR : Channel Search Support Register.

## 29.2.17 MSSR : Mailbox Search Status Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x852

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SEST | — | — | MBNST[4:0] | | | | |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | MBNST[4:0] | Search Result Mailbox Number Status<br>These bits output the smallest mailbox number that is found in each search mode selected in the MSMR. | R |
| 6:5 | — | These bits are read as 0. | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7 | SEST | Search Result Status<br>0: Search result found<br>1: No search result | R |

### MBNST[4:0] bits (Search Result Mailbox Number Status)

In all mailbox search modes, the MBNST[4:0] bits output the smallest found mailbox number. In receive mailbox search mode, transmit mailbox search mode, and message lost search mode, the value of the mailbox (the search result to be output) is updated under the following conditions:

- When the associated NEWDATA, SENTDATA, or MSGLOST flag is set to 0 for a mailbox output by MBNST[4:0]
- When the associated NEWDATA, SENTDATA, or MSGLOST flag is set to 1 for a mailbox with a smaller number than that in MBNST[4:0]

If the MBSM[1:0] bits are set to 00b (receive mailbox search mode) or 10b (message lost search mode), the receive FIFO (mailbox 28) is output when it is not empty and there are no unread received messages and no lost messages in any of the normal mailboxes 0 to 23. If the MBSM[1:0] bits are set to 01b (transmit mailbox search mode), the transmit FIFO (mailbox 24) is not output. Table 29.6 lists the behavior of the MBNST[4:0] bits in FIFO mailbox mode.

In channel search mode, the MBNST[4:0] bits output the associated channel number. After MSSR is read by software, the next target channel number is output.

### SEST bit (Search Result Status)

The SEST bit is set to 1 (no search result) when no associated mailbox is found after searching all mailboxes. For example, in transmit mailbox search mode, the SEST bit is set to 1 when no SENTDATA flag is 1 for any mailbox. The SEST bit is set to 0 when at least one SENTDATA flag is 1. When the SEST bit is 1, the value of the MBNST[4:0] bits is undefined.

**Table 29.6 Behavior of MBNST[4:0] bits in FIFO mailbox mode**

| MBSM[1:0] bits | Mailbox 24 (transmit FIFO) | Mailbox 28 (receive FIFO) |
|---|---|---|
| 00b | Mailbox 24 is not output. | Mailbox 28 is output when no MCTL_RX[j].NEWDATA flag for the normal mailboxes is set to 1 (new message is being stored or was stored in the mailbox) and the receive FIFO is not empty. |
| 01b | | Mailbox 28 is not output. |
| 10b | | Mailbox 28 is output when no MCTL_RX[j].MSGLOST flag for the normal mailboxes is set to 1 (message is overwritten or overrun) and the RFCR.RFMLF flag is set to 1 (receive FIFO message lost) in the receive FIFO. |
| 11b | | Mailbox 28 is not output. |

## 29.2.18 CSSR : Channel Search Support Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x851

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | When the value for the channel search is input, the channel number is output to MSSR. | R/W |

The bits that are set to 1 in CSSR, are encoded by an 8/3 encoder (the LSB position has the higher priority) and output to the MBNST[4:0] bits in MSSR. MSSR outputs the updated value whenever it is read by software.

Write to CSSR only when the MSMR.MBSM[1:0] bits are 11b (channel search mode). Write to CSSR in CAN operation mode or CAN halt mode.

Figure 29.4 shows the write and read operations of the CSSR and MSSR registers.

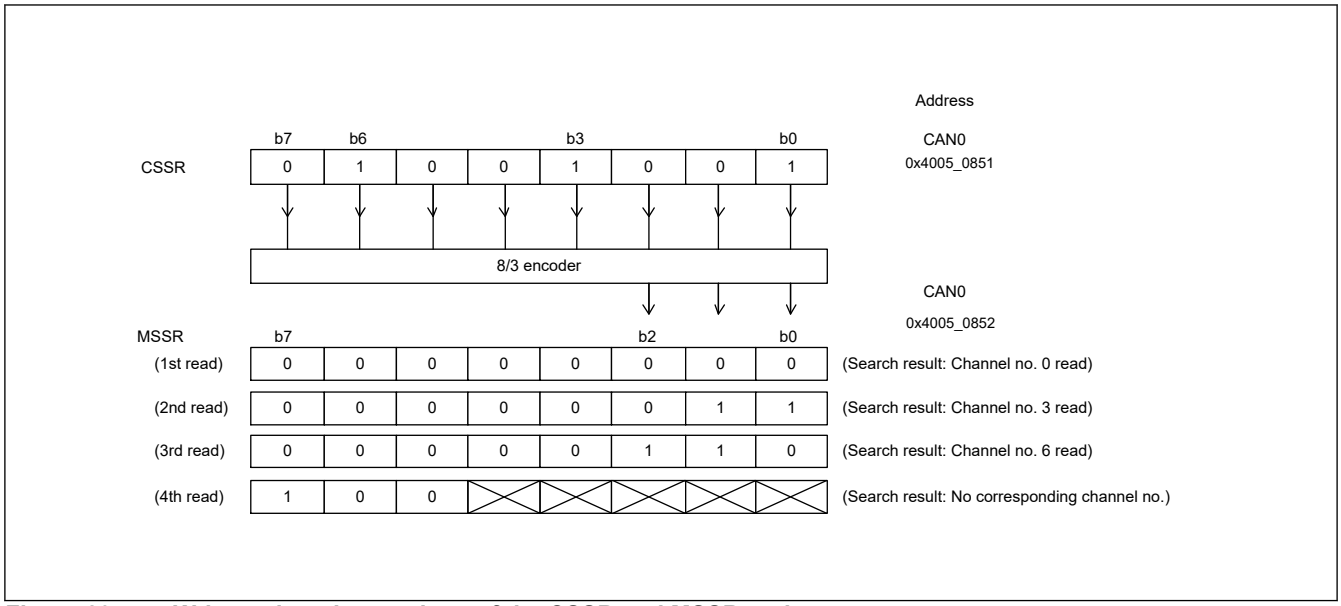



**Figure 29.4 Write and read operations of the CSSR and MSSR registers**

The value of CSSR is also updated whenever MSSR is read. On this read, the value prior to conversion by the 8/3 encoder can be read.

## 29.2.19 AFSR : Acceptance Filter Support Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x856

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | After the standard ID of a received message is written, the value converted for data table search can be read. | R/W |

Note: Write to AFSR in CAN operation mode or CAN halt mode.

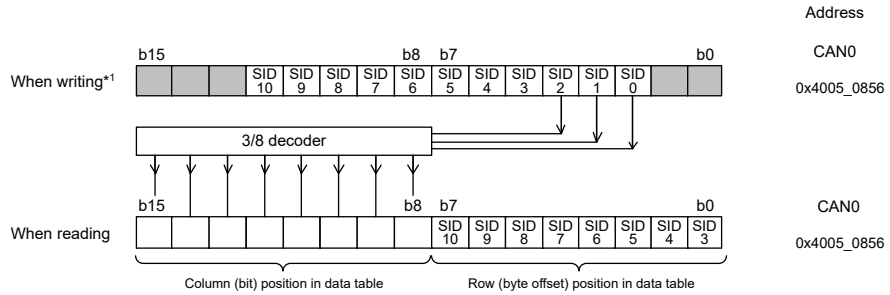
The acceptance filter support unit (ASU) can be used for data table (8 bits × 256) searches. In the data table, all standard IDs that are created are set as valid or invalid in bit units. When AFSR is written with data in 16-bit units including the SID[10:0] bits in MBj_ID (j = 0 to 31), in which a received standard ID is stored, a decoded row (byte offset) position and column (bit) position for data table search can be read. The ASU can be used for standard (11-bit) IDs only.

The ASU is enabled in the following cases:

- When the ID to be received cannot be masked by the acceptance filter. For example, if IDs to be received are 0x078, 0x087, and 0x111.
- When there are too many IDs to receive, and the software filtering time is expected to be shortened.

Note: AFSR cannot be set in CAN reset mode.

Figure 29.5 shows the write and read operations in the AFSR register.




Note 1. Write the same value as the 16-bit unit data, including the SID[10:0] bits in MBj_ID (j = 0 to 31).

**Figure 29.5 Write and read operations in the AFSR register**

## 29.2.20 EIER : Error Interrupt Enable Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x84C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | BLIE | OLIE | ORIE | BORIE | BOEIE | EPIE | EWIE | BEIE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BEIE | Bus Error Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 1 | EWIE | Error-Warning Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 2 | EPIE | Error-Passive Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 3 | BOEIE | Bus-Off Entry Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 4 | BORIE | Bus-Off Recovery Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 5 | ORIE | Overrun Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 6 | OLIE | Overload Frame Transmit Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |
| 7 | BLIE | Bus Lock Interrupt Enable<br>0: Disable interrupt<br>1: Enable interrupt | R/W |

The EIER register enables or disables each error interrupt source independently in EIFR. Write to EIER in CAN reset mode.

### BEIE bit (Bus Error Interrupt Enable)

When the BEIE bit is 0, no error interrupt request occurs even if the BEIF flag in EIFR is 1. When the BEIE bit is 1, an error interrupt request occurs if the BEIF flag is set to 1.

### EWIE bit (Error-Warning Interrupt Enable)

When the EWIE bit is 0, no error interrupt request occurs even if the EWIF flag in EIFR is 1. When the EWIE bit is 1, an error interrupt request is generated if the EWIF flag is set to 1.

### EPIE bit (Error-Passive Interrupt Enable)

When the EPIE bit is 0, no error interrupt request occurs even if the EPIF flag in EIFR is 1. When the EPIE bit is 1, an error interrupt request occurs if the EPIF flag is set to 1.

### BOEIE bit (Bus-Off Entry Interrupt Enable)

When the BOEIE bit is 0, no error interrupt request occurs even if the BOEIF flag in EIFR is 1. When the BOEIE bit is 1, an error interrupt request occurs if the BOEIF flag is set to 1.

### BORIE bit (Bus-Off Recovery Interrupt Enable)

When the BORIE bit is 0, no error interrupt request occurs even if the BORIF flag in EIFR is 1. When the BORIE bit is 1, an error interrupt request occurs if the BORIF flag is set to 1.

### ORIE bit (Overrun Interrupt Enable)

When the ORIE bit is 0, no error interrupt request occurs even if the ORIF flag in EIFR is 1. When the ORIE bit is 1, an error interrupt request occurs if the ORIF flag is set to 1.

### OLIE bit (Overload Frame Transmit Interrupt Enable)

When the OLIE bit is 0, no error interrupt request occurs even if the OLIF flag in EIFR is 1. When the OLIE bit is 1, an error interrupt request occurs if the OLIF flag is set to 1.

### BLIE bit (Bus Lock Interrupt Enable)

When the BLIE bit is 0, no error interrupt request occurs even if the BLIF flag in EIFR is 1. When the BLIE bit is 1, an error interrupt request occurs if the BLIF flag is set to 1.

## 29.2.21 EIFR : Error Interrupt Factor Judge Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x84D

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | BLIF | OLIF | ORIF | BORIF | BOEIF | EPIF | EWIF | BEIF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BEIF | Bus Error Detect Flag<br>0: No bus error detected<br>1: Bus error detected | R/W |
| 1 | EWIF | Error-Warning Detect Flag<br>0: No error-warning detected<br>1: Error-warning detected | R/W |
| 2 | EPIF | Error-Passive Detect Flag<br>0: No error-passive detected<br>1: Error-passive detected | R/W |
| 3 | BOEIF | Bus-Off Entry Detect Flag<br>0: No bus-off entry detected<br>1: Bus-off entry detected | R/W |
| 4 | BORIF | Bus-Off Recovery Detect Flag<br>0: No bus-off recovery detected<br>1: Bus-off recovery detected | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5 | ORIF | Receive Overrun Detect Flag<br>0: No receive overrun detected<br>1: Receive overrun detected | R/W |
| 6 | OLIF | Overload Frame Transmission Detect Flag<br>0: No overload frame transmission detected<br>1: Overload frame transmission detected | R/W |
| 7 | BLIF | Bus Lock Detect Flag<br>0: No bus lock detected<br>1: Bus lock detected | R/W |

If an event associated with an EIFR flag occurs, the associated bit in EIFR is set to 1, regardless of the setting of EIER.

Clear the bits to 0 through a software write. If a bit is set to 1 at the same time that the software clears it, it becomes 1. When setting a single bit to 0 in software, use the transfer instruction (MOV) to ensure that only the specified bit is set to 0 and the other bits are set to 1. Writing 1 has no effect to these bit values.

### BEIF flag (Bus Error Detect Flag)

The BEIF flag is set to 1 when a bus error is detected.

### EWIF flag (Error-Warning Detect Flag)

The EWIF flag is set to 1 when the value of the receive error counter (REC) or transmit error counter (TEC) exceeds 95. The EWIF flag is set to 1 only when REC or TEC initially exceeds 95. If software writes 0 to this flag while the REC or TEC value remains greater than 95, the EWIF flag does not set to 1 until the REC or TEC value goes below 95, and then exceeds 95 again.

### EPIF flag (Error-Passive Detect Flag)

The EPIF flag is set to 1 when the CAN error state becomes error-passive, when the REC or TEC value exceeds 127. The EPIF flag is set to 1 only when the REC or TEC value initially exceeds 127. If software writes 0 to this flag while REC or TEC remains greater than 127, the EPIF flag is not set to 1 until REC or TEC goes below 127, and then exceeds 127 again.

### BOEIF flag (Bus-Off Entry Detect Flag)

The BOEIF flag is set to 1 when the CAN error state becomes bus-off, when the TEC value exceeds 255. The BOEIF flag is also set to 1 when the BOM[1:0] bits in CTLR are 01b (automatic entry to CAN halt mode on bus-off entry) and the CAN module becomes the bus-off state.

### BORIF flag (Bus-Off Recovery Detect Flag)

The BORIF flag is set to 1 when the CAN module recovers from the bus-off state normally by detecting 11 consecutive bits 128 times in the following conditions:

- When the BOM[1:0] bits in CTLR are 00b
- When the BOM[1:0] bits in CTLR are 10b
- When the BOM[1:0] bits in CTLR are 11b

The BORIF flag is not set to 1 if the CAN module recovers from the bus-off state in the following conditions:

- When the CANM[1:0] bits in CTLR are set to 01b or 11b (CAN reset mode)
- When the RBOC bit in CTLR is set to 1 (forced return from bus-off)
- When the BOM[1:0] bits in CTLR are set to 01b
- When the BOM[1:0] bits in CTLR are set to 11b and the CANM[1:0] bits in CTLR are set to 10b (CAN halt mode) before normal recovery occurs

Table 29.7 lists the behavior of the BOEIF and BORIF flags for each CTLR.BOM[1:0] bits setting.

**Table 29.7 Behavior of BOEIF and BORIF flags for each CTLR.BOM[1:0] setting**

| BOM[1:0] bits | BOEIF flag | BORIF flag |
|---|---|---|
| 00b | Set to 1 on entry to the bus-off state | Sets to 1 on exit from the bus-off state |
| 01b | | Do not set to 1 |
| 10b | | Set to 1 on exit from the bus-off state |
| 11b | | Set to 1 if normal bus-off recovery occurs before the CANM[1:0] bits are set to 10b (CAN halt mode) |

### ORIF flag (Receive Overrun Detect Flag)

The ORIF flag is set to 1 when a receive overrun occurs. This flag is not set to 1 in overwrite mode.

In overwrite mode, a reception complete interrupt request occurs if an overwrite condition occurs and the ORIF flag is not set to 1.

In overrun mode with normal mailbox mode, if an overrun occurs in any of mailboxes 0 to 31, the ORIF flag is set to 1. In overrun mode with FIFO mailbox mode, if an overrun occurs in any of mailboxes 0 to 23 or the receive FIFO, this flag is set to 1.

### OLIF flag (Overload Frame Transmission Detect Flag)

The OLIF flag is set to 1 if the transmitting condition of an overload frame is detected when the CAN module is transmitting or receiving.

### BLIF flag (Bus Lock Detect Flag)

The BLIF flag is set to 1 if 32 consecutive dominant bits are detected on the CAN bus while the CAN module is in CAN operation mode. After the BLIF flag is set to 1, 32 consecutive dominant bits are detected again in either of the following conditions:

- Recessive bits are detected after the BLIF flag changes to 0 from 1
- The CAN module enters CAN reset or halt mode and then enters CAN operation mode again after the BLIF flag changes to 0 from 1

## 29.2.22 RECR : Receive Error Count Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x84E

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | Receive error count function. RECR increments or decrements the counter value based on the error status of the CAN module during reception. | R |

RECR indicates the value of the receive error counter. See the CAN specification (ISO11898-1) for information on the increment and decrement conditions of the receive error counter.

The value of RECR in the bus-off state is undefined.

## 29.2.23 TECR : Transmit Error Count Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x84F

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | Transmit error count function. TECR increments or decrements the counter value based on the error status of the CAN module during transmission. | R |

TECR indicates the value of the transmit error counter. See the CAN specification (ISO11898-1) for information on the increment and decrement conditions of the transmit error counter.

The value of TECR in the bus-off state is undefined.

## 29.2.24 ECSR : Error Code Store Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x850

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | EDPM | ADEF | BE0F | BE1F | CEF | AEF | FEF | SEF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SEF | Stuff Error Flag*1 *2<br>0: No stuff error detected<br>1: Stuff error detected | R/W |
| 1 | FEF | Form Error Flag*1 *2<br>0: No form error detected<br>1: Form error detected | R/W |
| 2 | AEF | ACK Error Flag*1 *2<br>0: No ACK error detected<br>1: ACK error detected | R/W |
| 3 | CEF | CRC Error Flag*1 *2<br>0: No CRC error detected<br>1: CRC error detected | R/W |
| 4 | BE1F | Bit Error (recessive) Flag*1 *2<br>0: No bit error (recessive) detected<br>1: Bit error (recessive) detected | R/W |
| 5 | BE0F | Bit Error (dominant) Flag*1 *2<br>0: No bit error (dominant) detected<br>1: Bit error (dominant) detected | R/W |
| 6 | ADEF | ACK Delimiter Error Flag*1 *2<br>0: No ACK delimiter error detected<br>1: ACK delimiter error detected | R/W |
| 7 | EDPM | Error Display Mode Select*3 *4<br>0: Output first detected error code<br>1: Output accumulated error code | R/W |

Note 1. Writing 1 has no effect on these bit values.

Note 2. To write 0 to the SEF, FEF, AEF, CEF, BE1F, BE0F, and ADEF bits, use the transfer (MOV) instruction to ensure that only the specified bit is set to 0 and the other bits are set to 1.

Note 3. Write to the EDPM bit in CAN reset or halt mode.
Note 4. If more than one error condition is detected simultaneously, all related bits are set to 1.

ECSR indicates whether an error occurs on the CAN bus. See the CAN specification (ISO11898-1) for the conditions when each error occurs.

Clear all of the bits, except for the EDPM bit, to 0 through a software write. If the ECSR bit is set to 1 at the same time that software writes 0 to it, the bit becomes 1.

#### SEF flag (Stuff Error Flag)

The SEF flag is set to 1 when a stuff error is detected.

#### FEF flag (Form Error Flag)

The FEF flag is set to 1 when a form error is detected.

#### AEF flag (ACK Error Flag)

The AEF flag is set to 1 when an ACK error is detected.

#### CEF flag (CRC Error Flag)

The CEF flag is set to 1 when a CRC error is detected.

#### BE1F flag (Bit Error (recessive) Flag)

The BE1F flag is set to 1 when a recessive bit error is detected.

#### BE0F flag (Bit Error (dominant) Flag)

The BE0F flag is set to 1 when a dominant bit error is detected.

#### ADEF flag (ACK Delimiter Error Flag)

The ADEF flag is set to 1 when a form error is detected with the ACK delimiter during transmission.

#### EDPM bit (Error Display Mode Select)

The EDPM bit selects the output mode of ECSR. When the EDPM bit is set to 0, ECSR outputs the first error code. When the EDPM bit is set to 1, ECSR outputs the accumulated error code.

### 29.2.25 TSR : Time Stamp Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x854

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | Free-running counter value for the time stamp function | R |

Note: Read TSR in 16-bit units.

Reading the TSR register returns the current value of the 16-bit free-running time stamp counter. The time stamp counter reference clock is configured in the TSPS[1:0] bits in CTLR. The counter stops in CAN sleep mode and CAN halt mode, and is initialized in CAN reset mode. The time stamp counter value is stored in the TSL[7:0] and TSH[7:0] bits in MBj_TS when a received message is stored in a receive mailbox.

## 29.2.26 TCR : Test Control Register

Base address: CAN0 = 0x400A_8000

Offset address: 0x858

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | TSTM[1:0] | | TSTE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TSTE | CAN Test Mode Enable<br>0: Disable CAN test mode<br>1: Enable CAN test mode | R/W |
| 2:1 | TSTM[1:0] | CAN Test Mode Select<br>0 0: Not CAN test mode<br>0 1: Listen-only mode<br>1 0: Self-test mode 0 (external loopback)<br>1 1: Self-test mode 1 (internal loopback) | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

TCR controls the CAN test mode. Write to TCR in CAN halt mode only.

### (1) Listen-only mode

The CAN specification (ISO11898-1) recommends an optional bus monitoring mode. In listen-only mode, valid data frames and valid remote frames can be received. However, only recessive bits can be sent on the CAN bus. The ACK bit, overload flag, and active error flag cannot be sent.

Listen-only mode can be used for baud rate detection. Do not request transmission from any mailboxes in listen-only mode.

Figure 29.6 shows the connection when listen-only mode is selected.

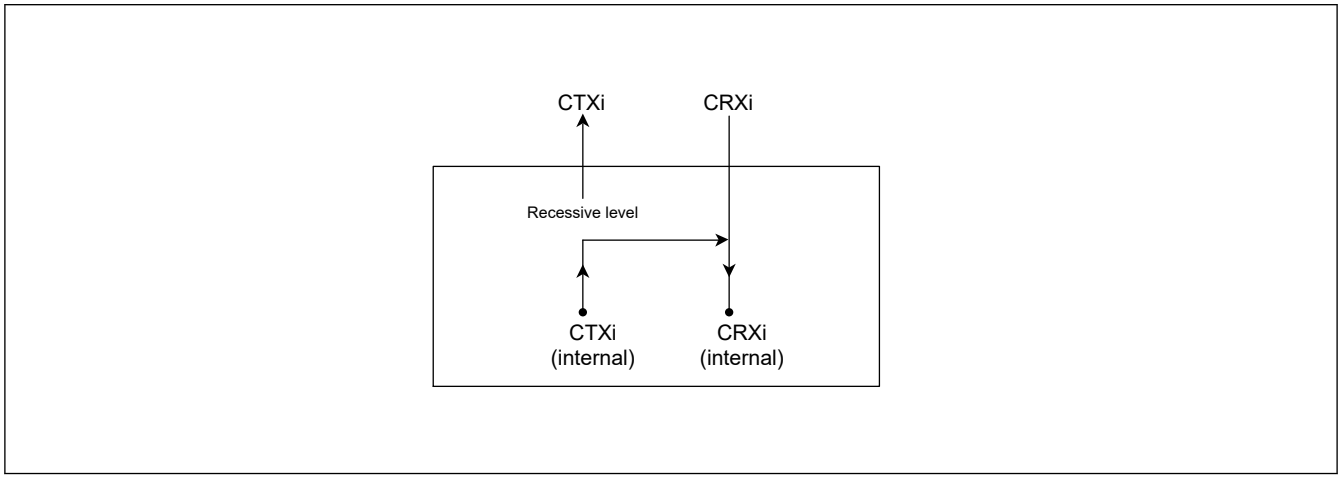



**Figure 29.6 Connection when listen-only mode is selected (i = 0)**

### (2) Self-test mode 0 (external loopback)

Self-test mode 0 is provided for CAN transceiver tests. In this mode, the protocol module treats its own transmitted messages as those received by the CAN transceiver and stores them into the receive mailbox. To be independent from external stimulation, the protocol module generates the ACK bit. Connect the CTXi and CRXi pins to the transceiver.

Figure 29.7 shows the connection when self-test mode 0 is selected.

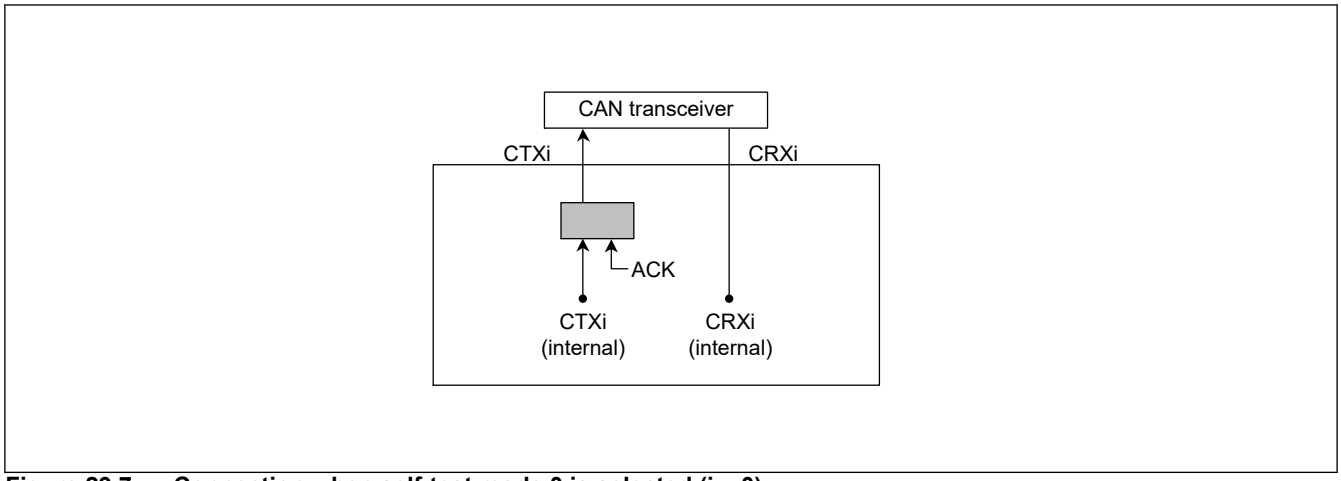



**Figure 29.7 Connection when self-test mode 0 is selected (i = 0)**

### (3) Self-test mode 1 (internal loopback)

Self-test mode 1 is provided for self-test functions. In this mode, the protocol controller treats its transmitted messages as received messages and stores them into the receive mailbox. To be independent from external stimulation, the protocol controller generates the ACK bit.

In self-test mode 1, the protocol controller performs an internal feedback from the internal CTXi pin to the internal CRXi pin. The input value of the external CRXi pin is ignored. The external CTXi pin outputs only recessive bits. The CTXi and CRXi pins are not required to be connected to the CAN bus or any external device.

Figure 29.8 shows the connection when self-test mode 1 is selected.

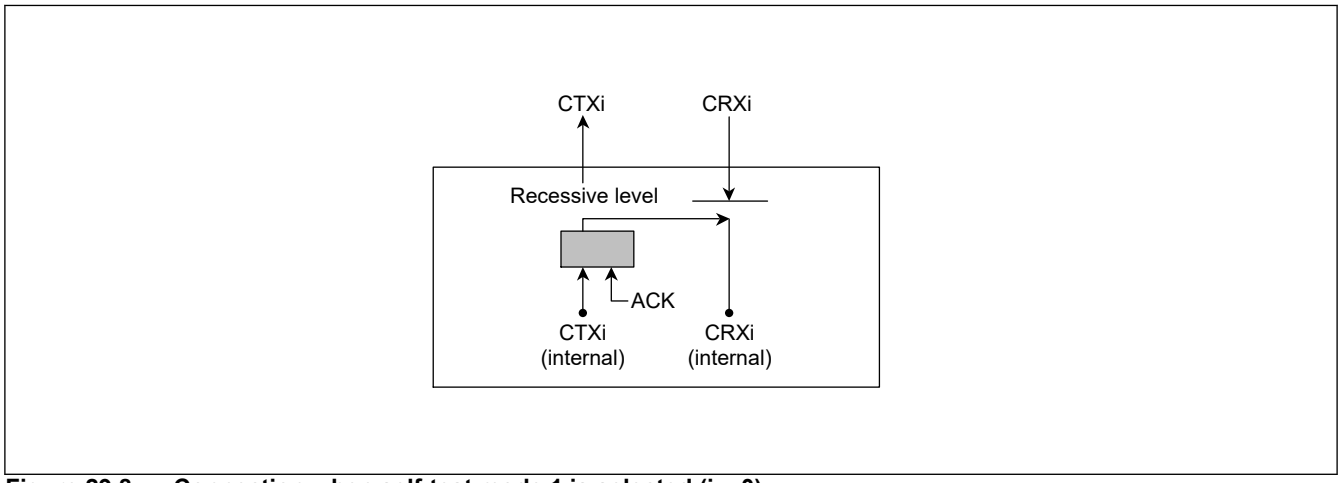


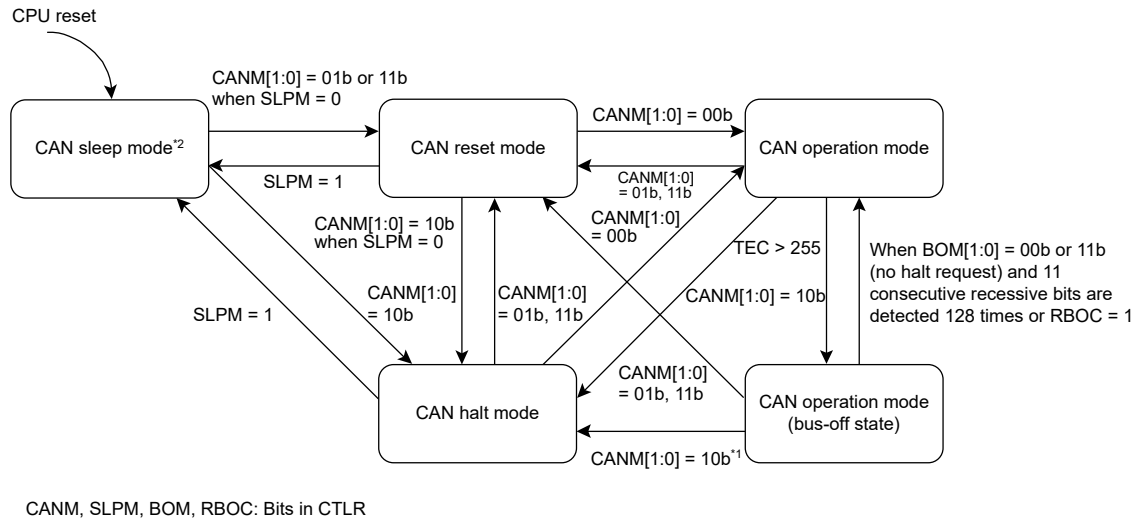

**Figure 29.8 Connection when self-test mode 1 is selected (i = 0)**

## 29.3 Operation Modes

The CAN module operation includes the following modes:

- CAN reset mode
- CAN halt mode
- CAN operation mode
- CAN sleep mode

Figure 29.9 shows the transitions between different operation modes.




Note 1. The transition timing from the bus-off state to CAN halt mode depends on the setting of the CTLR.BOM[1:0] bits.
When the CTLR.BOM[1:0] bits are 01b, the state transition occurs immediately after entering the bus-off state.
When the CTLR.BOM[1:0] bits are 10b, the state transition occurs at the end of the bus-off state.
When the CTLR.BOM[1:0] bits are 11b, the state transition occurs at the setting of the CTLR.CANM[1:0] bits to 10b (CAN halt mode).
Note 2. Change the CTLR.SLPM bit to set or cancel CAN sleep mode.

**Figure 29.9 Transition between different operation modes**

## 29.3.1 CAN Reset Mode

CAN reset mode is provided for CAN communication configuration. When the CTLR.CANM[1:0] bits are set to 01b or 11b, the CAN module enters CAN reset mode. The STR.RSTST bit is then set to 1. Do not change the CTLR.CANM[1:0] bits until the RSTST flag is 1. Set BCR before exiting CAN reset mode to enter any other modes.

The following registers are initialized to their reset values after entering CAN reset mode, and their initial values are saved during CAN reset mode:

- MCTL_TX[j] and MCTL_RX[j]
- STR (except for the SLPST and TFST bits)
- EIFR
- RECR
- TECR
- TSR
- MSSR
- MSMR
- RFCR
- TFCR
- TCR
- ECSR (except for the EDPM bit)

The following registers retain their previous values even after entering CAN reset mode:

- CTLR
- STR (only the SLPST and TFST bits)

- MIER and MIER_FIFO
- EIER
- BCR
- CSSR
- ECSR (only the EDPM bit)
- MBj_ID, MBj_DL, MBj_Dm and MBj_TS
- MKRk
- FIDCR0 and FIDCR1
- MKIVLR
- AFSR
- RFPCR
- TFPCR

### 29.3.2 CAN Halt Mode

CAN halt mode is used for mailbox configuration and test mode setting.

When the CTLR.CANM[1:0] bits are set to 10b, CAN halt mode is selected, and the STR.HLTST bit is set to 1. Do not change the CTLR.CANM[1:0] bits until the HLTST bit is 1.

See Table 29.8 for the state transition conditions when transmitting or receiving.

All registers except for the RSTST, HLTST, and SLPST bits in STR remain unchanged when the CAN enters CAN halt mode. Do not change CTLR (except for the CANM[1:0] and SLPM bits) and EIER in CAN halt mode. BCR can be changed in CAN halt mode only when listen-only mode is selected for automatic baud rate detection.

**Table 29.8 Operation in CAN reset mode and CAN halt mode**

| Operation mode | Receiver | Transmitter | Bus-off |
|---|---|---|---|
| CAN reset mode (forced transition) CANM[1:0] = 11b | CAN module enters CAN reset mode without waiting for the end of message reception. | CAN module enters CAN reset mode without waiting for the end of message transmission. | CAN module enters CAN reset mode without waiting for the end of bus-off recovery. |
| CAN reset mode CANM[1:0] = 01b | CAN module enters CAN reset mode without waiting for the end of message reception. | CAN module enters CAN reset mode after waiting for the end of message transmission.*1 *4 | CAN module enters CAN reset mode without waiting for the end of bus-off recovery. |
| CAN halt mode | CAN module enters CAN halt mode after waiting for the end of message reception.*2 *3 | CAN module enters CAN halt mode after waiting for the end of message transmission.*1 *4 | • When the BOM[1:0] bits are 00b: A halt request from software is accepted only after bus-off recovery.<br>• When the BOM[1:0] bits are 01b: CAN module automatically enters CAN halt mode without waiting for the end of bus-off recovery, regardless of a halt request from the software.<br>• When the BOM[1:0] bits are 10b: CAN module automatically enters CAN halt mode after waiting for the end of bus-off recovery, regardless of a halt request from software.<br>• When the BOM[1:0] bits are 11b: CAN module enters CAN halt mode without waiting for the end of bus-off recovery, if a halt is requested by software during bus-off. |

Note: BOM[1:0] bits: Bits in CTLR.
Note 1. If transmission of multiple messages is requested, a mode transition occurs on completion of the first transmission. If the CAN reset mode is being requested during suspend transmission, mode transition occurs when the bus is idle, the next transmission ends, or the CAN module becomes a receiver.
Note 2. If the CAN bus is locked at the dominant level, the program can detect this state by monitoring the BLIF flag in EIFR.
Note 3. If a CAN bus error occurs during reception after CAN halt mode is requested, the CAN module transitions to CAN halt mode.

Note 4. If a CAN bus error or arbitration-lost occurs during transmission after CAN reset mode or CAN halt mode is requested, the CAN module transitions to the requested CAN mode.

### 29.3.3 CAN Sleep Mode

CAN sleep mode reduces power consumption by stopping the clock supply to the CAN module. After a reset from an MCU pin or a software reset, the CAN module starts from CAN sleep mode.

When the SLPM bit in CTLR is set to 1, the CAN module enters CAN sleep mode, and the SLPST bit in STR is set to 1. Do not change the value of the SLPM bit until the SLPST bit is 1. The other registers remain unchanged when the CAN module enters CAN sleep mode.

Write to the SLPM bit in CAN reset mode and CAN halt mode. Do not change any registers (except for the SLPM bit) during CAN sleep mode. Read operation is still allowed.

When the SLPM bit is set to 0, the CAN module is released from CAN sleep mode. When the CAN module exits CAN sleep mode, the other registers remain unchanged.

### 29.3.4 CAN Operation Mode (Excluding Bus-Off State)

CAN operation mode is used for CAN communication.

When the CANM[1:0] bits in CTLR are set to 00b, the CAN module enters CAN operation mode. The RSTST and HLTST bits in STR are set to 0. Do not change the value of the CANM[1:0] bits until the RSTST and HLTST bits are 0.

If 11 consecutive recessive bits are detected after entering CAN operation mode:

- The CAN module becomes an active node on the network, which enables transmission and reception of CAN messages.
- Error monitoring of the CAN bus, such as receive and transmit error counters, is performed.

During CAN operation mode, the CAN module might be in one of the following three sub-modes, depending on the status of the CAN bus:

- Idle mode: No transmission or reception occurs.
- Receive mode: A CAN message sent by another node is being received.
- Transmit mode: A CAN message is being transmitted. The CAN module receives a message transmitted by the local node simultaneously when self-test mode 0 (TSTM[1:0] bits in TCR = 10b) or self-test mode 1 (TSTM[1:0] bits = 11b) is selected.

Figure 29.10 shows the sub-modes of CAN operation mode.

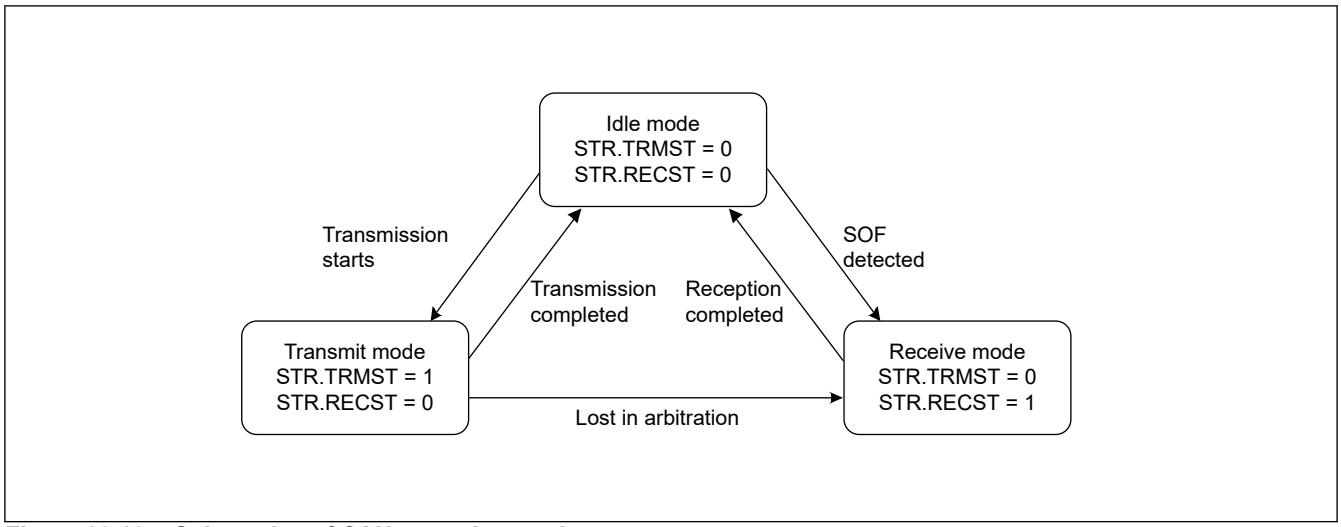



**Figure 29.10 Sub-modes of CAN operation mode**

### 29.3.5 CAN Operation Mode (Bus-Off State)

The CAN module enters the bus-off state based on the incrementing or decrementing rules for the transmit and error counters defined in the CAN specifications.

The following cases apply when the CAN module is recovering from the bus-off state. When the CAN module is in the bus-off state, the values of the CAN module registers remain unchanged, except for those in STR, EIFR, RECR, TECR, and TSR.

#### (1) When CTLR.BOM[1:0] = 00b (normal mode)

The CAN module enters the error-active state after it completes recovery from the bus-off state and CAN communication is enabled. The BORIF flag in EIFR is set to 1 (bus-off recovery detected).

#### (2) When CTLR.RBOC = 1 (forced return from bus-off)

The CAN module enters the error-active state when it is in the bus-off state and the RBOC bit is 1. CAN communication is enabled again after 11b consecutive recessive bits are detected. The BORIF flag is not set to 1.

#### (3) When CTLR.BOM[1:0] = 01b (automatic transition to CAN halt mode on bus-off entry)

The CAN module enters CAN halt mode when it reaches the bus-off state. The BORIF flag is not set to 1.

#### (4) When CTLR.BOM[1:0] = 10b (automatic transition to CAN halt mode on bus-off end)

The CAN module enters CAN halt mode when it completes recovery from bus-off. The BORIF flag is set to 1.

#### (5) When CTLR.BOM[1:0] = 11b (automatic transition to CAN halt mode through software) and CTLR.CANM[1:0] = 10b (CAN halt mode) during bus-off state

The CAN module enters CAN halt mode when it is in the bus-off state and the CANM[1:0] bits are set to 10b (CAN halt mode). The BORIF flag is not set to 1.

If the CANM[1:0] bits are not set to 10b during bus-off, the same behavior as (1) applies.

## 29.4 Data Transfer Rate Configuration

This section describes how to configure the data transfer rate.

### 29.4.1 Clock Setting

The CAN module provides a CAN clock generator. The CAN clock can be set in the CCLKS and BRP[9:0] bits in BCR. Figure 29.11 shows a block diagram of the CAN clock generator.

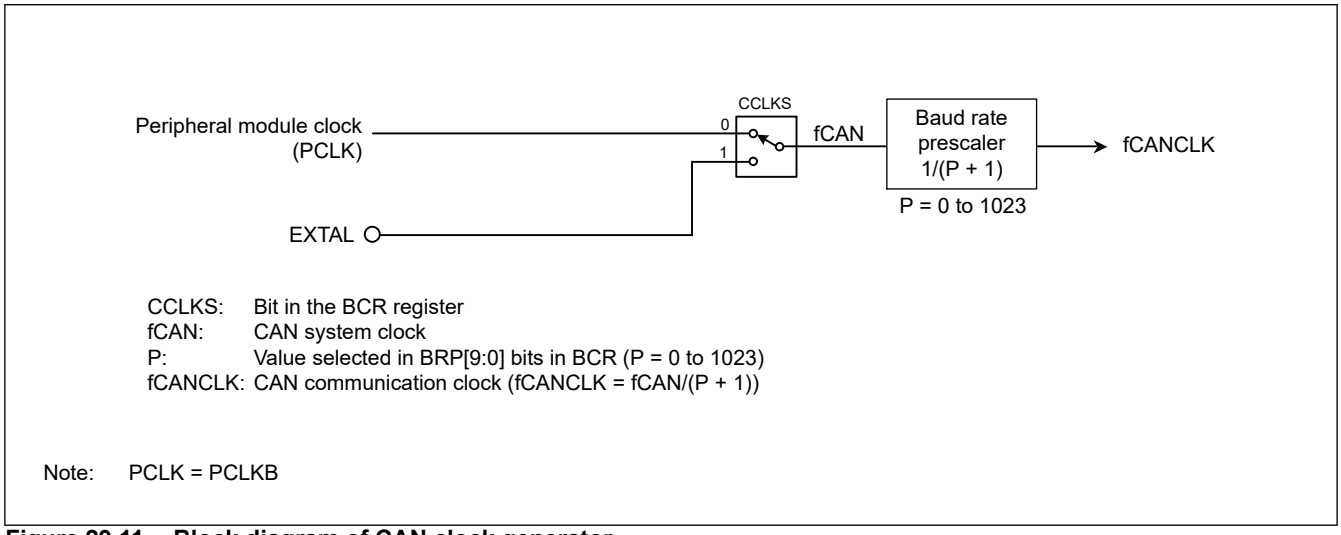
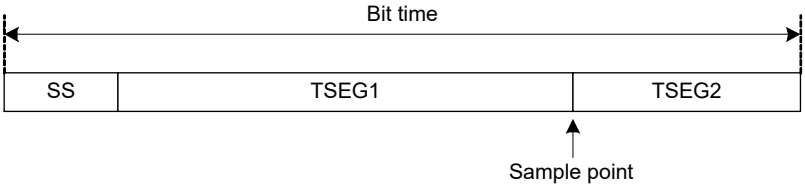



**Figure 29.11 Block diagram of CAN clock generator**

### 29.4.2 Bit Timing Setting

The bit timing consists of three segments as shown in Figure 29.12.




The range of each segment: Bit time = 8 Tq to 25 Tq
SS = 1 Tq
TSEG1 = 4 Tq to 16 Tq
TSEG2 = 2 Tq to 8 Tq
SJW = 1 Tq to 4 Tq
Setting of TSEG1 and TSEG2: TSEG1 > TSEG2 ≥ SJW

**Figure 29.12 Bit timing**

### 29.4.3 Data Transfer Rate

The data transfer rate depends on the division value of fCAN (CAN system clock), the division value of the baud rate prescaler, and the Tq count for 1 bit time.

The division value of baud rate prescaler = P + 1 (P: 0 to 1023), where P is the BRP[9:0] setting in BCR.

$$\text{Data transfer rate (bps)} = \frac{\text{fCAN}}{\text{Baud rate prescaler division value} \times \text{Tq count for 1 bit time}} = \frac{\text{fCANCLK}}{\text{Tq count for 1 bit time}}$$

Table 29.9 lists data transfer rate examples.

**Table 29.9 Data transfer rate examples**

| fCAN | 20 MHz | |
|---|---|---|
| **Data transfer rate** | **Tq count** | **P + 1** |
| 1 Mbps | 5 Tq<br>10 Tq | 4<br>2 |
| 500 kbps | 5 Tq<br>10 Tq | 8<br>4 |
| 250 kbps | 5 Tq<br>10 Tq | 16<br>8 |
| 125 kbps | 5 Tq<br>10 Tq | 32<br>16 |
| 83.3 kbps | 5 Tq<br>10 Tq | 48<br>24 |
| 33.3 kbps | 5 Tq<br>8 Tq<br>10 Tq | 120<br>75<br>60 |

## 29.5 Mailbox and Mask Register Structure

Figure 29.13 shows the structure of the 32 mailbox registers: MBj_ID, MBj_DL, MBj_Dm, and MBj_TS.

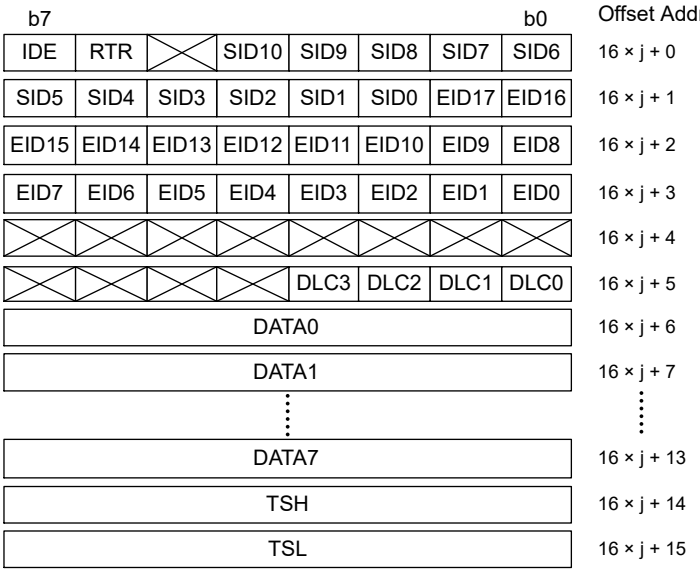



**Figure 29.13 Structure of the mailbox registers (j = 0 to 31)**

Figure 29.14 shows the structure of the eight mask registers MKRk.

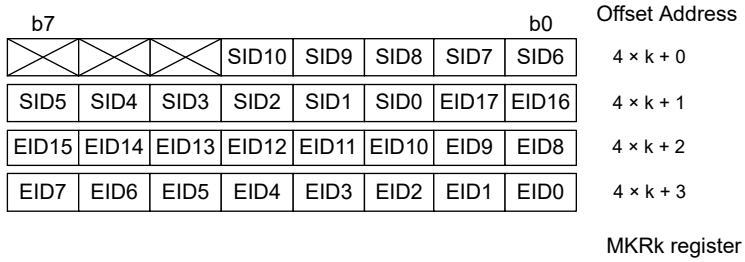



**Figure 29.14 Structure of the MKRk registers (k = 0 to 7)**

Figure 29.15 shows the structure of the two FIFO receive ID compare registers FIDCR0 and FIDCR1.

**Figure 29.15 Structure of the FIDCRn registers (n = 0)**

## 29.6 Acceptance Filtering and Masking Functions

The acceptance filtering and masking functions allow you to select and receive messages with multiple IDs for mailboxes within a specified range.

The MKRk registers can mask the standard ID and the extended ID of 29 bits.

- MKR0 controls mailboxes 0 to 3
- MKR1 controls mailboxes 4 to 7
- MKR2 controls mailboxes 8 to 11
- MKR3 controls mailboxes 12 to 15
- MKR4 controls mailboxes 16 to 19
- MKR5 controls mailboxes 20 to 23
- MKR6 controls mailboxes 24 to 27 in normal mailbox mode and the receive FIFO mailboxes 28 to 31 in FIFO mailbox mode
- MKR7 controls mailboxes 28 to 31 in normal mailbox mode and the receive FIFO mailboxes 28 to 31 in FIFO mailbox mode

MKIVLR disables acceptance filtering independently for each mailbox.

The IDE bit in MBj_ID is valid when the IDFM[1:0] bits in CTLR are 10b (mixed ID mode).

The RTR bit in MBj_ID selects a data or remote frame.

In FIFO mailbox mode, normal mailboxes 0 to 23 use one associated register from MKR0 to MKR5 for acceptance filtering. The receive FIFO mailboxes 28 to 31 use two registers, MKR6 and MKR7, for acceptance filtering.

The receive FIFO also uses two registers, FIDCR0 and FIDCR1, for ID comparison. The EID[17:0], SID[10:0], RTR, and IDE bits in MB28 to MB31 for the receive FIFO are disabled. As acceptance filtering depends on the result of two logic AND operations, two ranges of IDs can be received into the receive FIFO. MKIVLR is disabled for the receive FIFO.

If different standard ID and extended ID values are set in the IDE bits in FIDCR0 and FIDCR1, both ID formats are received.

If different data frame and remote frame values are set in the RTR bits in FIDCR0 and FIDCR1, both data and remote frames are received.

When a combination of two ranges of IDs is not required, set the same mask value and the same ID into both the FIFO ID and mask registers.

Figure 29.16 shows the associations between the mask registers and mailboxes. Figure 29.17 shows the acceptance filtering.

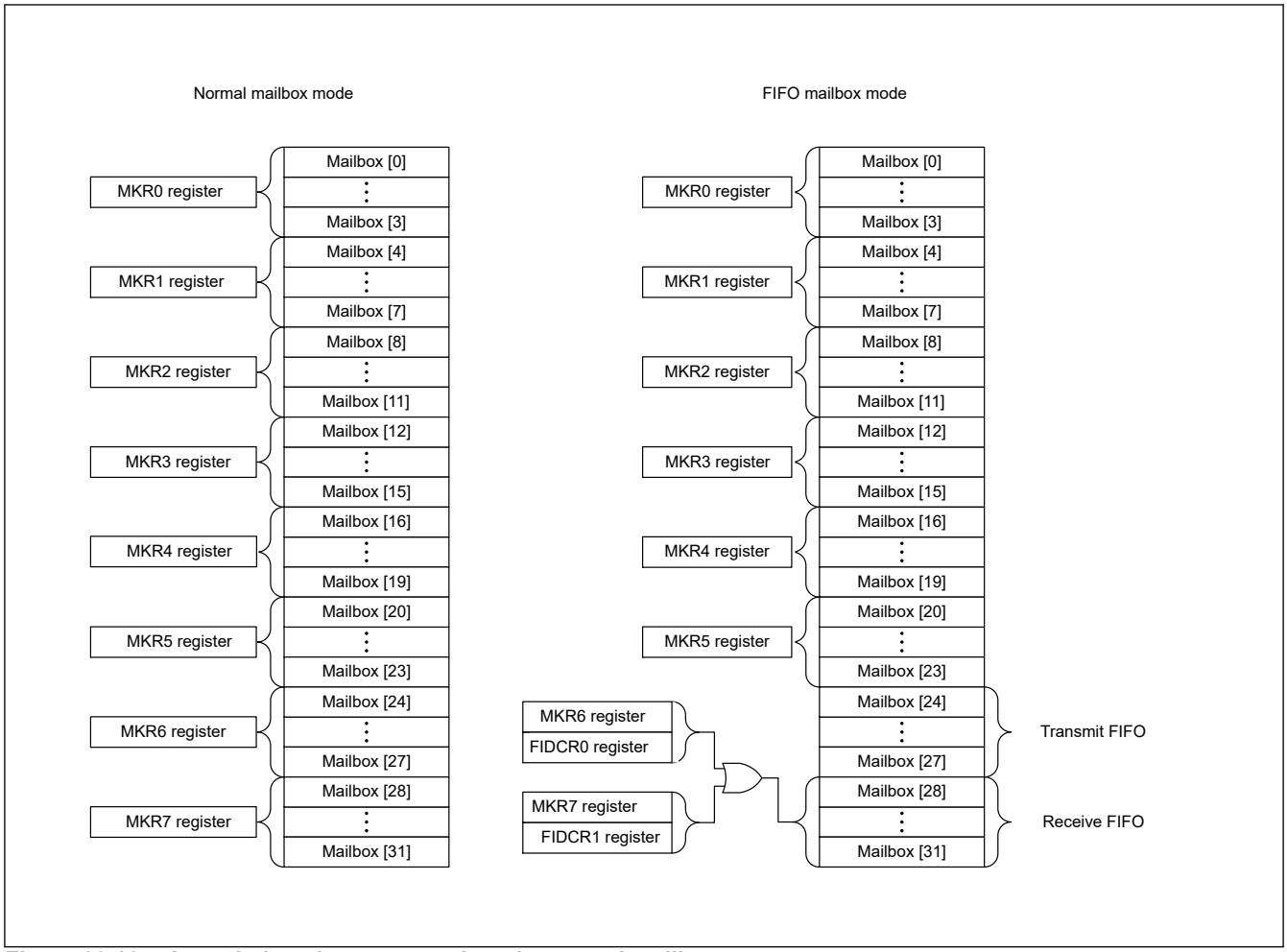
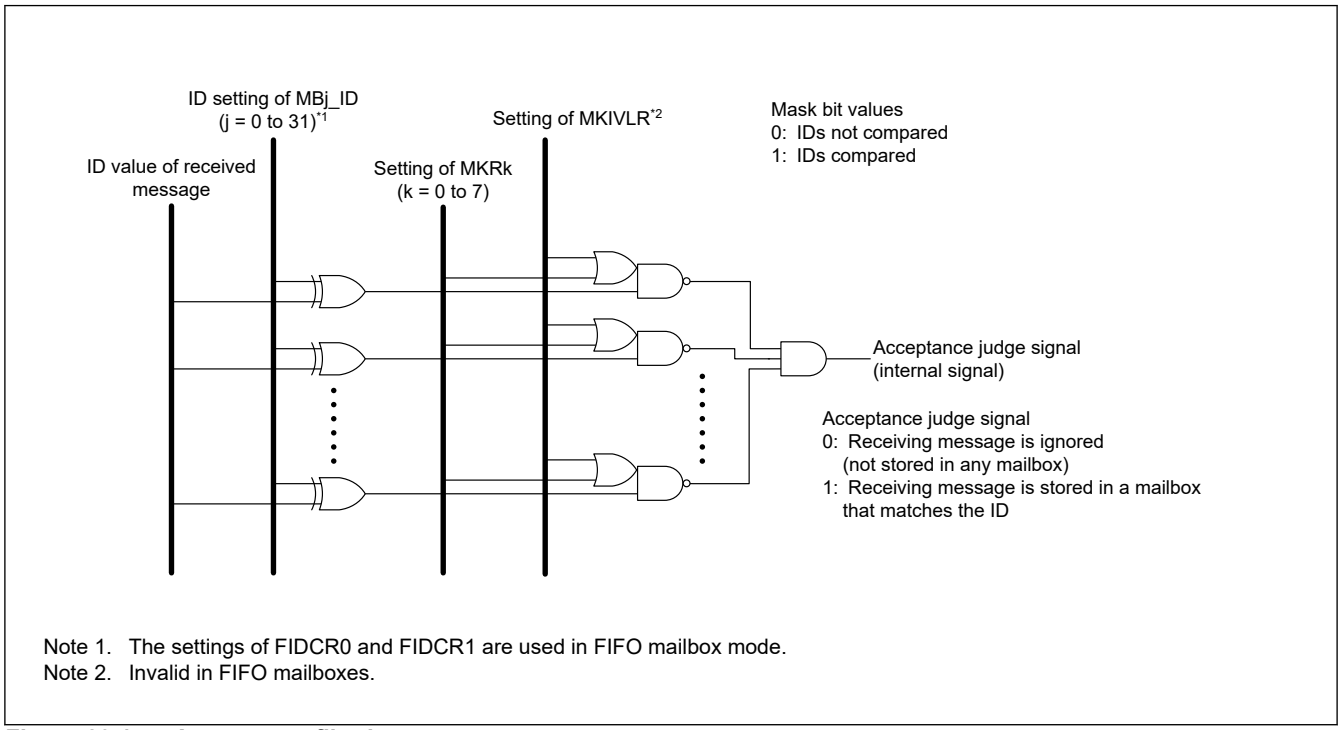



**Figure 29.16 Associations between mask registers and mailboxes**




Note 1. The settings of FIDCR0 and FIDCR1 are used in FIFO mailbox mode.
Note 2. Invalid in FIFO mailboxes.

**Figure 29.17 Acceptance filtering**

## 29.7 Reception and Transmission

Table 29.10 lists the CAN communication mode settings.

**Table 29.10 Settings for CAN receive and transmit modes**

| MCTL_TX[j].TRMREQ and MCTL_RX[j].TRMREQ | MCTL_TX[j].RECREQ and MCTL_RX[j].RECREQ | MCTL_TX[j].ONESHOT and MCTL_RX[j].ONESHOT | Mailbox communication mode |
|---|---|---|---|
| 0 | 0 | 0 | Mailbox disabled or transmission being aborted |
| 0 | 0 | 1 | Can be configured only when transmission or reception from a mailbox programmed in one-shot mode is aborted |
| 0 | 1 | 0 | Configured as a receive mailbox for a data or remote frame |
| 0 | 1 | 1 | Configured as a one-shot receive mailbox for a data or remote frame. |
| 1 | 0 | 0 | Configured as a transmit mailbox for a data or remote frame. |
| 1 | 0 | 1 | Configured as a one-shot transmit mailbox for a data or remote frame. |
| 1 | 1 | 0 | Do not set. |
| 1 | 1 | 1 | Do not set. |

Note: j = 0 to 31

When a mailbox is configured as a receive mailbox or a one-shot receive mailbox, the following restrictions apply:

- Before configuring the mailbox, set MCTL_RX[j] to 0x00.
- A received message is stored in the first mailbox that matches the condition resulting from the receive mode settings and acceptance filtering. The matching mailbox with the smallest number takes priority for storing the received message.
- In CAN operation mode, the CAN module does not receive its own transmitted data even when the ID is a match. In self-test mode, however, the CAN module receives its own transmitted data and returns ACK.

When configuring a mailbox as a transmit mailbox or a one-shot transmit mailbox, the following constraint applies:

- Before configuring a mailbox, ensure that MCTL_TX[j] is 0x00 and that there is no pending abort process.

### 29.7.1 Reception

Figure 29.18 shows an operation example of data frame reception in overwrite mode. The example shows the overwriting of the first message when the CAN module receives two consecutive CAN messages that match the receiving conditions in MCTL_RX[j] (j = 0 to 31).

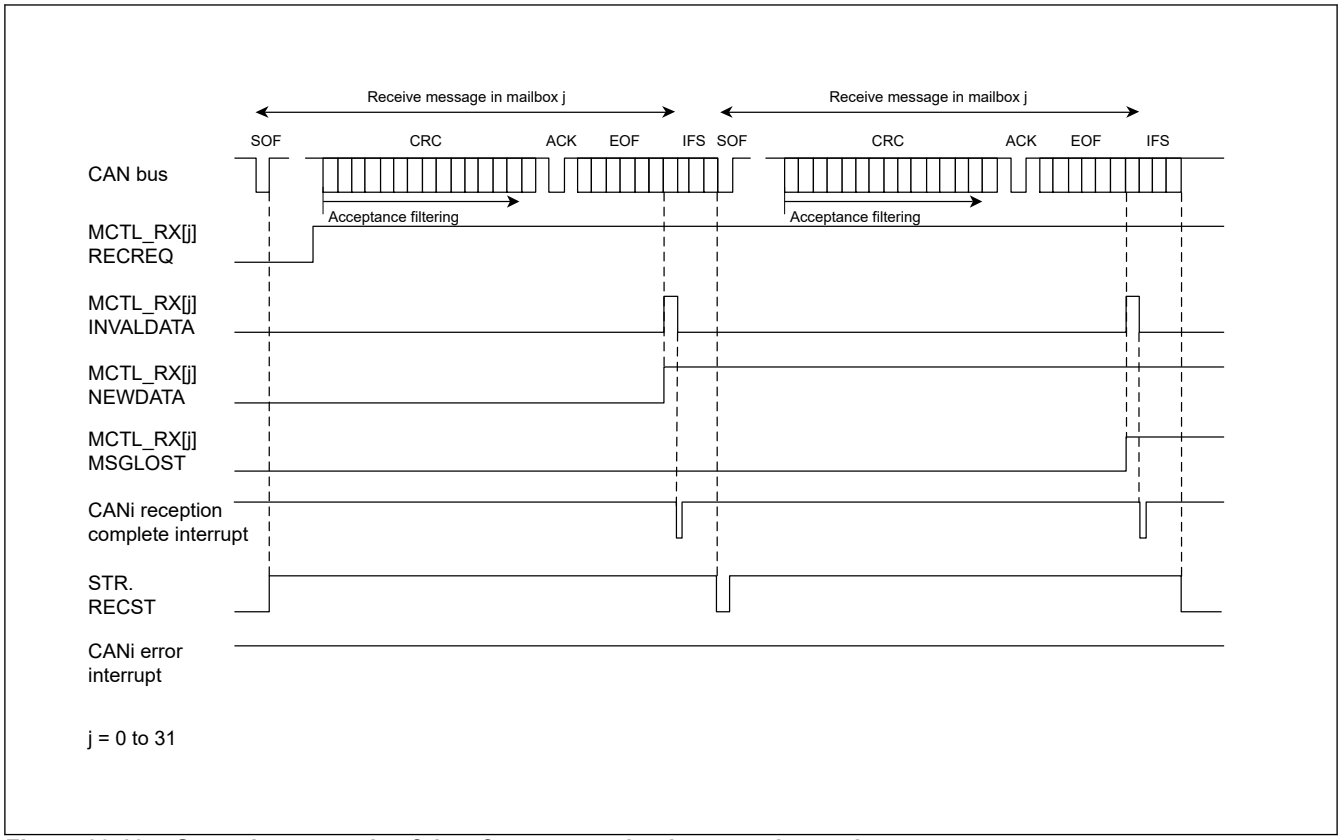



**Figure 29.18 Operation example of data frame reception in overwrite mode**

1. When an SOF is detected on the CAN bus, the RECST bit in STR is set to 1 (reception in progress) if the CAN module has no message ready to start transmission.
2. Acceptance filtering starts at the beginning of the CRC field to select the receive mailbox.
3. After a message is received, the NEWDATA flag in MCTL_RX[j] for the receive mailbox is set to 1 (new message is being stored or was stored in the mailbox). The INVALDATA flag in MCTL_RX[j] is set to 1 (message is being updated) at the same time, and then the INVALDATA flag is set to 0 (message valid) again after the complete message is transferred to the mailbox.
4. When the interrupt enable bit in MIER for the receive mailbox is 1 (interrupt enabled), the INVALDATA flag is set to 0, which triggers a CAN0 reception complete interrupt request.
5. After reading the message from the mailbox, the NEWDATA flag must be set to 0 by software.
6. In overwrite mode, if the next CAN message is received before the NEWDATA flag in MCTL_RX[j] is set to 1, the MSGLOST flag in MCTL_RX[j] is set to 1 (message was overwritten). The new received message is transferred to the mailbox. The CAN0 reception complete interrupt request occurs the same as in step 4.

Figure 29.19 shows an operation example of data frame reception in overrun mode. The example shows the overrunning of the second message when the CAN module receives two consecutive CAN messages that match the receiving conditions in MCTL_RX[j] (j = 0 to 31).

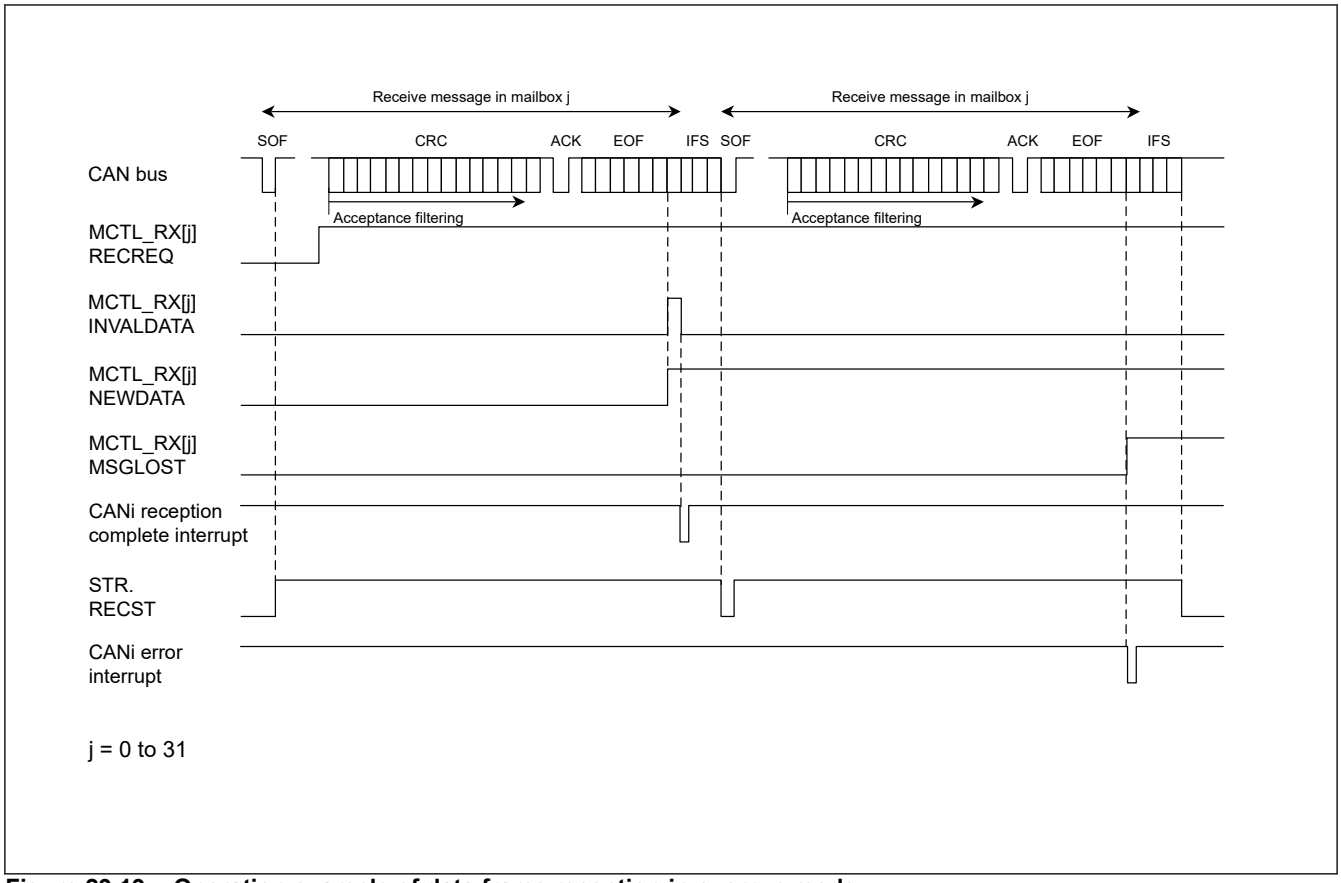



**Figure 29.19 Operation example of data frame reception in overrun mode**

1. When an SOF is detected on the CAN bus, the RECST bit in STR is set to 1 (reception in progress) if the CAN module has no message ready to start transmission.
2. Acceptance filtering starts at the beginning of the CRC field to select the receive mailbox.
3. After a message is received, the NEWDATA flag in MCTL_RX[j] for the receive mailbox is set to 1 (new message is being stored or was stored in the mailbox). The INVALDATA flag in MCTL_RX[j] is set to 1 (message is being updated) at the same time, and then the INVALDATA flag is set to 0 (message valid) again after the complete message is transferred to the mailbox.
4. When the interrupt enable bit in MIER for the receive mailbox is 1 (interrupt enabled), the INVALDATA flag is set to 0, which triggers a CAN0 reception complete interrupt request.
5. After reading the message from the mailbox, the NEWDATA flag must be set to 0 by software.
6. In overrun mode, if the next CAN message is received before the NEWDATA flag in MCTL_RX[j] is set to 0, the MSGLOST flag in MCTL_RX[j] is set to 1 (message was saved). The new received message is discarded and a CANi error interrupt request occurs if the associated interrupt enable bit in EIER is set to 1 (interrupt enabled).

## 29.7.2 Transmission

Figure 29.20 shows an operation example of data frame transmission.

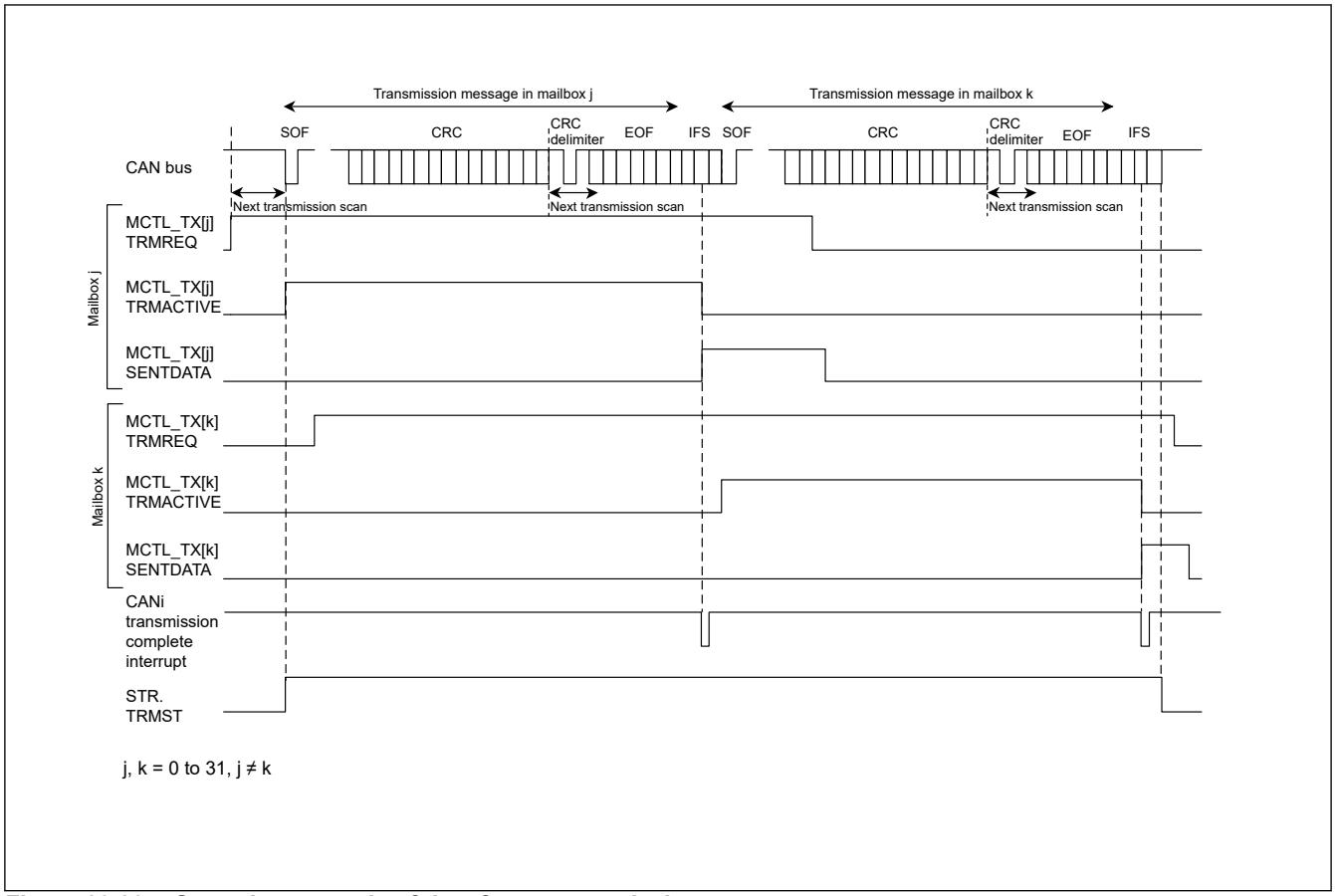



**Figure 29.20 Operation example of data frame transmission**

1. When a TRMREQ bit in MCTL_TX[j] (j = 0 to 31) is set to 1 (transmit mailbox) in the bus-idle state, mailbox scanning starts to decide the highest-priority mailbox for transmission. When the transmit mailbox is decided, the TRMACTIVE flag in MCTL_TX[j] is set to 1 (from acceptance of transmission request to completion of transmission, or error or arbitration-lost), the TRMST bit in STR is set to 1 (transmission in progress), and the CAN module starts transmission.*1
2. If other TRMREQ bits are set, the transmission scanning starts with the CRC delimiter for the next transmission.
3. If transmission is complete without losing arbitration, the SENTDATA flag in MCTL_TX[j] is set to 1 (transmission complete) and the TRMACTIVE flag is set to 0 (transmission is pending or transmission is not requested). If the interrupt enable bit in MIER is 1 (interrupt enabled), the CANi transmission complete interrupt request is generated.
4. When requesting the next transmission from the same mailbox, set the SENTDATA flag and TRMREQ bit to 0, and then set the TRMREQ bit to 1 after checking that the SENTDATA flag and TRMREQ bit are set to 0.

Note 1. If arbitration is lost after the CAN module starts transmission, the TRMACTIVE flag is set to 0. Transmission scanning is performed again to search for the highest-priority transmit mailbox from the beginning of the CRC delimiter. If an error occurs either during transmission or following arbitration-lost, transmission scanning is performed again to search for the highest-priority transmit mailbox from the start of the error delimiter.

## 29.8 Interrupts

The CAN module provides the following interrupts for each channel. Table 29.11 lists the CAN interrupts.

- CANi reception complete interrupt for mailboxes 0 to 31 (CANi_RXM)
- CANi transmission complete interrupt for mailboxes 0 to 31 (CANi_TXM)
- CANi receive FIFO interrupt (CANi_RXF)
- CANi transmit FIFO interrupt (CANi_TXF)
- CANi error interrupt (CANi_ERS)

Eight interrupt sources are available for CANi error interrupts. Check EIFR to determine the interrupt sources:

- Bus error
- Error-warning
- Error-passive
- Bus-off entry
- Bus-off recovery
- Receive overrun
- Overload frame transmission
- Bus lock

**Table 29.11 CAN interrupts**

| Module | Interrupt name | Interrupt source | Source flag |
|---|---|---|---|
| CANi (i = 0) | CANi_ERS | Bus lock detected | EIFR.BLIF |
| | | Overload frame transmission detected | EIFR.OLIF |
| | | Overrun detected | EIFR.ORIF |
| | | Bus-off recovery detected | EIFR.BORIF |
| | | Bus-off entry detected | EIFR.BOEIF |
| | | Error-passive detected | EIFR.EPIF |
| | | Error-warning detected | EIFR.EWIF |
| | | Bus error detected | EIFR.BEIF |
| | CANi_RXF | Receive FIFO message received (MIER_FIFO.MB29 = 0) | RFCR.RFUST[2:0] |
| | | Receive FIFO warning (MIER_FIFO.MB29 = 1) | |
| | CANi_TXF | Transmit FIFO message transmission completed (MIER_FIFO.MB25 = 0) | TFCR.TFUST[2:0] |
| | | FIFO last message transmission completed (MIER_FIFO.MB25 = 1) | |
| | CANi_RXM | Mailbox 0 to 31 message received | MCTL_RX0.NEWDATA to MCTL_RX31.NEWDATA |
| | CANi_TXM | Mailbox 0 to 31 message transmission completed | MCTL_TX0.SENTDATA to MCTL_TX31.SENTDATA |

## 29.9 Usage Notes

### 29.9.1 Settings for the Module-Stop State

The MSTPCRB can enable or disable CAN operation. The CAN module is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

### 29.9.2 Settings for the Operating Clock

The settings for the operating clock can be made as follows:

- The following clock constraint must be satisfied for the CAN module:

  $PCLKB \geq CANMCLK$

- The peripheral module clock source must be PLL for the CAN module when the CCLKS bit is 0.

# 30. Serial Peripheral Interface (SPI)

## 30.1 Overview

The Serial Peripheral Interface (SPI) has 1 channel. The SPI provides high-speed full-duplex synchronous serial communications with multiple processors and peripheral devices. Table 30.1 lists the SPI specifications, Figure 30.1 shows a block diagram, and Table 30.2 lists the I/O pins.

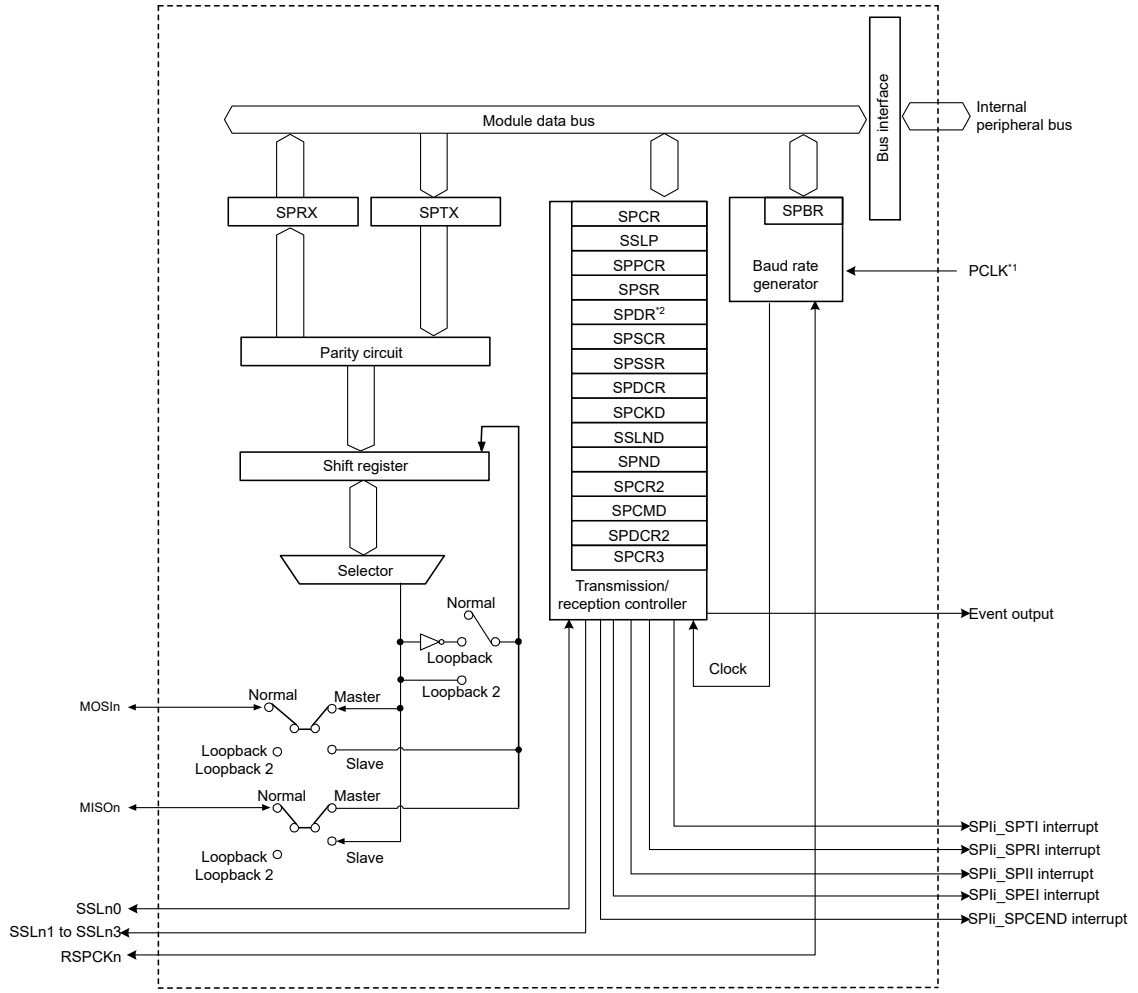
**Table 30.1 SPI specifications (1 of 2)**

| Parameter | Specifications |
|---|---|
| Number of channels | One channel |
| SPI transfer functions | • Use of MOSI (master out/slave in), MISO (master in/slave out), SSL (slave select), and RSPCK (SPI clock) signals allows serial communications through SPI operation (4-wire method) or clock synchronous operation (3-wire method)<br>• Transmit-only operation available<br>• Receive-only operation is available (Slave mode only)<br>• Communication mode selectable to full-duplex or transmit-only<br>• RSPCK polarity switching<br>• RSPCK phase switching |
| Data format | • MSB-first or LSB-first selectable<br>• Transfer bit length selectable to 8, 9, 10, 11, 12, 13, 14, 15, 16, 20, 24, or 32 bits<br>• 128-bit transmit and receive buffers<br>• Up to four frames transferable in one round of transmission or reception (each frame consisting of up to 32 bits)<br>• Byte swap operating function<br>• Transmit/receive data can be inverted. |
| Bit rate | • In master mode, the on-chip baud rate generator generates RSPCK by frequency-dividing PCLKA (the division ratio ranges from divided by 2 to divided by 4096)<br>• In slave mode, the minimum PCLKA clock divided by 4 can be input as RSPCK (PCLKA divided by 4 is the maximum RSPCK frequency)<br>Width at high level: 2 PCLKA cycles; width at low level: 2 PCLKA cycles |
| Buffer configuration | • Double buffer configuration for the transmit and receive buffers<br>• 128 bits for the transmit and receive buffers |
| Error detection | • Mode fault error detection<br>• Underrun error detection<br>• Overrun error detection*1<br>• Parity error detection |
| SSL control function | • Four SSL pins (SSLni: SSLn0 to SSLn3) (n = A) for each channel<br>• In single-master mode, SSLn0 to SSLn3 pins are output<br>• In multi-master mode, SSLn0 pin for input, and SSLn1 to SSLn3 pins either for output or unused<br>• In slave mode, SSLn0 pin for input and SSLn1 to SSLn3 pins unused<br>• Controllable delay from SSL output assertion to RSPCK operation (RSPCK delay)<br>Range: 1 to 8 RSPCK cycles (set in RSPCK-cycle units)<br>• Controllable delay from RSPCK stop to SSL output negation (SSL negation delay)<br>Range: 1 to 8 RSPCK cycles (set in RSPCK-cycle units)<br>• Controllable wait for next-access SSL output assertion (next-access delay)<br>Range: 1 to 8 RSPCK cycles (set in RSPCK-cycle units)<br>• Function for changing SSL polarity<br>• Delay between frames in burst transfer is settable |
| Control in master transfer | • Transfers of up to eight commands each can be executed sequentially in looped execution<br>• For each command, the following can be set:<br>SSL signal value, bit rate, RSPCK polarity and phase, transfer data length, MSB- or LSB-first, burst, RSPCK delay, SSL negation delay, and next-access delay<br>• Transfers can be initiated by writing to the transmit buffer<br>• MOSI signal value specifiable in SSL negation<br>• RSPCK auto-stop function |
| Interrupt sources | Interrupt sources:<br>• Receive buffer full interrupt<br>• Transmit buffer empty interrupt<br>• SPI error interrupt (mode fault error, overrun error, parity error)<br>• SPI idle interrupt (SPI idle)<br>• Transmission-complete interrupt |

**Table 30.1 SPI specifications (2 of 2)**

| Parameter | Specifications |
|---|---|
| Event link function | The following events can be output to the Event Link Controller (ELC):<br>● Receive buffer full signal<br>● Transmit buffer empty signal<br>● Mode fault, underrun, overrun, or parity error signal<br>● SPI idle signal<br>● Communication end signal |
| Other functions | ● Switching between CMOS output and open-drain output<br>● SPI initialization function<br>● Loopback mode |
| Module-stop function | Module-stop state can be set to reduce power consumption. |
| TrustZone Filter | Security attribution can be set |

Note 1. In master reception and when the RSPCK auto-stop function is enabled, an overrun error does not occur because the transfer clock is stopped on overrun error detection.




Note 1. PCLK = PCLKA
Note 2. SPDR = SPDR/SPDR_HA/SPDR_BY
Note 3. m = 0 to 7

**Figure 30.1 SPI block diagram**

The SPI automatically switches the I/O direction of the SSLn0 pin. SSLn0 is set as an output when the SPI is a single master, and as an input when the SPI is a multi-master or a slave. The RSPCKn, MOSIn, and MISOn pins are automatically set as inputs or outputs based on the master or slave setting and the level input on the SSLn0 pin. For details, see section 30.3.2. Controlling the SPI Pins.

**Table 30.2 SPI I/O pins**

| Channel | Pin name | I/O | Description |
|---|---|---|---|
| SPI0 | RSPCKA | I/O | Clock input/output pin |
| | SSLA0 | I/O | Slave selection input/output |
| | SSLA1 to SSLA3 | Output | Slave selection output |
| | MOSIA | I/O | Master transmit data input/output |
| | MISOA | I/O | Slave transmit data input/output |

## 30.2 Register Descriptions

### 30.2.1 SPCR : SPI Control Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SPRIE | SPE | SPTIE | SPEIE | MSTR | MODFEN | TXMD | SPMS |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SPMS | SPI Mode Select<br>0: Select SPI operation (4-wire method)<br>1: Select clock synchronous operation (3-wire method) | R/W |
| 1 | TXMD | Communications Operating Mode Select<br>0: Select full-duplex synchronous serial communications<br>1: Select serial communications with transmit-only | R/W |
| 2 | MODFEN | Mode Fault Error Detection Enable<br>0: Disable detection of mode fault errors<br>1: Enable detection of mode fault errors | R/W |
| 3 | MSTR | SPI Master/Slave Mode Select<br>0: Select slave mode<br>1: Select master mode | R/W |
| 4 | SPEIE | SPI Error Interrupt Enable<br>0: Disable SPI error interrupt requests<br>1: Enable SPI error interrupt requests | R/W |
| 5 | SPTIE | Transmit Buffer Empty Interrupt Enable<br>0: Disable transmit buffer empty interrupt requests<br>1: Enable transmit buffer empty interrupt requests | R/W |
| 6 | SPE | SPI Function Enable<br>0: Disable SPI function<br>1: Enable SPI function | R/W |
| 7 | SPRIE | SPI Receive Buffer Full Interrupt Enable<br>0: Disable SPI receive buffer full interrupt requests<br>1: Enable SPI receive buffer full interrupt requests | R/W |

**SPMS bit (SPI Mode Select)**

The SPMS bit selects SPI operation (4-wire method) or clock synchronous operation (3-wire method).

The SSLn0 to SSLn3 pins are not used in clock synchronous operation. The RSPCKn, MOSIn, and MISOn pins handle communications. For clock synchronous operation in master mode (MSTR = 1), the SPCMDm.CPHA bit can be set to either 0 or 1. For clock synchronous operation in slave mode (MSTR = 0), always set the CPHA bit to 1. Do not perform operations if the CPHA bit is set to 0 for clock synchronous operation in slave mode (MSTR = 0).

#### TXMD bit (Communications Operating Mode Select)

The TXMD bit selects full-duplex synchronous serial communications or transmit-only operations. When this bit is set to 1, the SPI only performs transmit operations and not receive operations (see section 30.3.6. Data Transfer Modes), and receive buffer full interrupt requests cannot be used.

TXMD setting is invalid in receive only slave mode.

#### MODFEN bit (Mode Fault Error Detection Enable)

The MODFEN bit enables or disables the detection of mode fault errors (see section 30.3.9. Error Detection). In addition, the SPI determines the I/O direction of the SSLni pins based on combination of the MODFEN and MSTR bits (see section 30.3.2. Controlling the SPI Pins).

#### MSTR bit (SPI Master/Slave Mode Select)

The MSTR bit selects master or slave mode for the SPI. Based on the MSTR bit settings, the SPI determines the direction of the RSPCKn, MOSIn, MISOn, and SSLni pins.

#### SPEIE bit (SPI Error Interrupt Enable)

The SPEIE bit enables or disables the generation of SPI error interrupt requests when one of the following occurs:

- The SPI detects a mode fault error or underrun error and sets the SPSR.MODF flag to 1
- The SPI detects an overrun error and sets the SPSR.OVRF flag to 1
- The SPI detects a parity error and sets the SPSR.PERF flag to 1

For details, see section 30.3.9. Error Detection.

#### SPTIE bit (Transmit Buffer Empty Interrupt Enable)

The SPTIE bit enables or disables the generation of transmit buffer empty interrupt requests when the SPI detects that the transmit buffer is empty. To generate a transmit buffer empty interrupt request when transmission starts, set the SPE and SPTIE bits to 1 at the same time or set the SPE bit to 1 after setting the SPTIE bit to 1.

When the SPTIE bit is 1, transmit buffer interrupts are generated even when the SPI function is disabled (when the SPE bit is changed to 0).

#### SPE bit (SPI Function Enable)

The SPE bit enables or disables the SPI function. The SPE bit cannot be set to 1 when the SPSR.MODF flag is 1. For details, see section 30.3.9. Error Detection.

Setting the SPE bit to 0 disables the SPI function and initializes a part of the module function. For details, see section 30.3.10. Initializing the SPI. In addition, a transmit buffer empty interrupt request is generated when the SPE bit is changed from 0 to 1 or from 1 to 0.

#### SPRIE bit (SPI Receive Buffer Full Interrupt Enable)

The SPRIE bit enables or disables the generation of an SPI receive buffer full interrupt request when the SPI detects a receive buffer full write after completion of a serial transfer.

## 30.2.2 SSLP : SPI Slave Select Polarity Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x01

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | SSL3P | SSL2P | SSL1P | SSL0P |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SSL0P | SSLn0 Signal Polarity Setting<br>0: Set SSLn0 signal to active-low<br>1: Set SSLn0 signal to active-high | R/W |
| 1 | SSL1P | SSLn1 Signal Polarity Setting<br>0: Set SSLn1 signal to active-low<br>1: Set SSLn1 signal to active-high | R/W |
| 2 | SSL2P | SSLn2 Signal Polarity Setting<br>0: Set SSLn2 signal to active-low<br>1: Set SSLn2 signal to active-high | R/W |
| 3 | SSL3P | SSLn3 Signal Polarity Setting<br>0: Set SSLn3 signal to active-low<br>1: Set SSLn3 signal to active-high | R/W |
| 7:4 | — | These bits are read as 0. The write value should be 0. | R/W |

## 30.2.3 SPPCR : SPI Pin Control Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x02

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | MOIFE | MOIFV | — | — | SPLP2 | SPLP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SPLP | SPI Loopback<br>0: Normal mode<br>1: Loopback mode (receive data = inverted transmit data) | R/W |
| 1 | SPLP2 | SPI Loopback 2<br>0: Normal mode<br>1: Loopback mode (receive data = transmit data) | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | MOIFV | MOSI Idle Fixed Value<br>0: Set level output on MOSIn pin during MOSI idling to low<br>1: Set level output on MOSIn pin during MOSI idling to high | R/W |
| 5 | MOIFE | MOSI Idle Value Fixing Enable<br>0: Set MOSI output value to equal final data from previous transfer<br>1: Set MOSI output value to equal value set in the MOIFV bit | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |

### SPLP bit (SPI Loopback)

The SPLP bit selects the mode of the SPI pins. When this bit is set to 1, the SPI shuts off the path between the MISOn pin and the shift register if the SPCR.MSTR bit is 1, and between the MOSIn pin and the shift register if the SPCR.MSTR bit is 0. The SPI then inverts the value of the input path for the shift register and connects it to the output path (loopback mode). For more information, see section 30.3.13. Loopback Mode.

### SPLP2 bit (SPI Loopback 2)

The SPLP2 bit selects the mode of the SPI pins. When this bit is set to 1, the SPI shuts off the path between the MISOn pin and the shift register if the SPCR.MSTR bit is 1, and between the MOSIn pin and the shift register if the SPCR.MSTR bit is 0. The SPI then connects the value of the input path for the shift register to the output path (loopback mode) without inverting the value. For more information, see section 30.3.13. Loopback Mode.

### MOIFV bit (MOSI Idle Fixed Value)

The MOIFV bit determines the MOSIn pin output value during the SSL negation period (including the SSL retention period during a burst transfer) when the MOIFE bit is 1 in master mode.

### MOIFE bit (MOSI Idle Value Fixing Enable)

The MOIFE bit fixes the MOSIn output value when the SPI is in master mode and in an SSL negation period (including the SSL retention period during a burst transfer).When the MOIFE bit is 0, the SPI outputs the last data from the previous serial transfer during the SSL negation period to the MOSIn pin. When the MOIFE bit is 1, the SPI outputs the fixed value set in the MOIFV bit to the MOSIn pin.

## 30.2.4 SPSR : SPI Status Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x03

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SPRF | CENDF | SPTEF | UDRF | PERF | MODF | IDLNF | OVRF |
| Value after reset: | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | OVRF | Overrun Error Flag<br>0: No overrun error occurred<br>1: Overrun error occurred | R/W*1 |
| 1 | IDLNF | SPI Idle Flag<br>0: SPI is in the idle state<br>1: SPI is in the transfer state | R |
| 2 | MODF | Mode Fault Error Flag<br>0: No mode fault or underrun error occurred<br>1: Mode fault error or underrun error occurred | R/W*1 |
| 3 | PERF | Parity Error Flag<br>0: No parity error occurred<br>1: Parity error occurred | R/W*1 |
| 4 | UDRF | Underrun Error Flag<br>The UDRF bit is valid when MODF flag is 1.<br>0: Mode fault error occurred (MODF = 1)<br>1: Underrun error occurred (MODF = 1) | R/W*1 *2 |
| 5 | SPTEF | SPI Transmit Buffer Empty Flag<br>0: Data is in the transmit buffer<br>1: No data is in the transmit buffer | R/W*3 |
| 6 | CENDF | Communication End Flag<br>0: Not communicating or communicating<br>1: Communication completed | R/W*1 |
| 7 | SPRF | SPI Receive Buffer Full Flag<br>0: No valid data is in SPDR/SPDR_HA<br>1: Valid data is in SPDR/SPDR_HA | R/W*3 |

Note 1. Only 0 can be written to clear the flag after reading 1.
Note 2. Clear the UDRF flag at the same time as the MODF flag.
Note 3. The write value should be 1.

### OVRF flag (Overrun Error Flag)

The OVRF flag indicates the occurrence of an overrun error. In master mode (SPCR.MSTR bit = 1) and when the RSPCK clock auto-stop function is enabled (SPCR1.SCKASE bit = 1), overrun errors do not occur. This flag does not set to 1. For details, see section 30.3.9.1. Overrun errors.

[Setting condition]

When the next serial transfer ends and the receive buffer is full, and satisfy one of following.

- The SPCR.TXMD bit = 0. (transmit-receive master mode or transmit-receive slave mode or receive only slave mode)
- The SPCR.MSTR bit = 0, and the SPCR3.ETXMD bit = 1. (receive only slave mode)

[Clearing condition]

- When 0 is written to the OVRF flag after the OVRF flag is confirmed to be 1 by a read of SPSR.

### IDLNF flag (SPI Idle Flag)

The IDLNF flag indicates the transfer status of the SPI.

[Setting conditions]

Master mode

- When none of the conditions in the master mode [Clearing condition] is met.

Slave mode

- When the SPE bit in SPCR is 1, enabling the SPI function.

[Clearing conditions]

Master mode

When condition 1 or all other conditions are satisfied.

Condition 1: The SPE bit in SPCR is 0, indicating that the SPI is initialized.

Condition 2: The transmit buffer (SPTX) is empty, indicating that data for the next transfer is not set.

Condition 3: The SPI internal sequencer is in the idle state, indicating that operation up to next-access delay is complete.

Condition 4: The SPCP[2:0] bits in SPSSR are 000 (at the beginning of sequence control)

Slave mode

- When condition 1 is satisfied.

### MODF flag (Mode Fault Error Flag)

The MODF flag indicates the occurrence of a mode fault error or an underrun error. The UDRF flag indicates which error occurred.

[Setting conditions]

Multi-master mode

- When the input level of the SSLni pin changes to the active level while the SPCR.MSTR bit is 1 (master mode) and the SPCR.MODFEN bit is 1 (mode fault error detection is enabled), triggering a mode fault error.

Slave mode

- When condition 1 or 2 is satisfied.

Condition 1: The SSLni pin is negated before the RSPCK cycle required for data transfer ends while the SPCR.MSTR bit is 0 (slave mode) and the SPCR.MODFEN bit is 1 (mode fault error detection is enabled), triggering a mode fault error.

Condition 2: The serial transfer begins with the SPCR.MSTR bit is set to 0 (slave mode), the SPCR.SPE bit is set to 1, and the transmission data not prepared, triggering an underrun error.

The active level of the SSLni signal is determined by the SSLP.SSLiP bit (SSLi signal polarity setting).

[Clearing condition]

- When SPSR is read while this flag is 1, and then 0 is written to this flag.

### PERF flag (Parity Error Flag)

The PERF flag indicates the occurrence of a parity error.

[Setting condition]

When a serial transfer ends while the SPCR2.SPPE bit is 1, triggering a parity error, and satisfy one of following.

- The SPCR.TXMD bit = 0. (transmit-receive master mode or transmit-receive slave mode or receive only slave mode)
- The SPCR.MSTR bit = 0, and the SPCR3.ETXMD bit = 1. (receive only slave mode)

[Clearing condition]

- When SPSR is read while this flag is 1, and then 0 is written to this flag.

### UDRF flag (Underrun Error Flag)

The UDRF flag indicates the occurrence of an underrun error.

[Setting condition]

- When the serial transfer begins with the SPCR.MSTR bit is set to 0 (slave mode), the SPCR3.ETXMD bit =0 (transmit-receive slave mode or transmit slave mode) the SPCR.SPE bit is set to 1, and the transmission data not prepared, triggering an underrun error.

[Clearing condition]

- When SPSR is read while this flag is 1, and then 0 is written to this flag.

### SPTEF flag (SPI Transmit Buffer Empty Flag)

The SPTEF flag indicates the status of the transmit buffer for the SPI Data Register (SPDR/SPDR_HA).

[Setting conditions]

- When condition 1. or 2. is satisfied.
  1. The SPCR.SPE bit is 0, indicating that the SPI is initialized.
  2. Transmit data (the frame size specified by the SPDCR.SPFC[1:0]) is transferred from the transmit buffer to the shift register.

[Clearing condition]

- When data written to SPDR/SPDR_HA/SPDR_BY equals the number of frames set in the SPFC[1:0] bits in the SPI Data Control Register (SPDCR).

Data can only be written to SPDR/SPDR_HA/SPDR_BY when the SPTEF flag is 1. If data is written to the transmit buffer of SPDR/SPDR_HA when the SPTEF flag is 0, data in the transmit buffer is not updated.

### CENDF flag (Communication End Flag)

This flag indicates communication end status of SPI. It turns 1 at communication end, and turns 0 at starting next communication.

[Setting condition]

Mastar mode

The following 3 conditions are met.

- The transmit buffer(SPTX) is empty. (There is no next transmission data.)
- The SPSSR.SPCP[2:0] are 000b. (It means the head of the sequential control.)
- The state of SPI internal sequencer transferred to the idle state. (It means the next access delay has been completed.)

Transmit-receive / transmit only slave mode in SPI serial communication (4-wire: the SPCR.SPMS bit is 0)

The following 3 conditions are met.

- The transmit buffer(SPTX) is empty. (There is no next transmission data.)

- The transmission shift register is empty. (It means SPI does not do serial transfer.)
- SSL0 was negated.

Transmit-receive / transmit only slave mode in clock synchronous (3-wire: the SPCR.SPMS bit is 1)

The following 3 conditions are met.

- The transmit buffer(SPTX) is empty. (There is no next transmission data.)
- The transmission shift register is empty. (It means SPI does not do serial transfer.)
- The last even edge of RSPCK of the last data was detected. (When the SPCMD.CPHA bit is “1”.)

Receive only slave mode in SPI serial communication (4-wire: the SPCR.SPMS bit is 0)

The following condition is met.

- SSL0 was negated after the last data was written in the received buffer.

Receive only slave mode in clock synchronous (3-wire: the SPCR.SPMS bit is 1)

The following condition is met.

- The last even edge of RSPCK of the last data was detected. (When the SPCMD.CPHA bit is 1.)

[Clearing condition]

Master mode

Satisfy one of following.

- The next transmit data was written to the transmit buffer (SPTX).
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1

Transmit-receive / transmit only slave mode

Satisfy one of following.

- The next transmit data was written to the transmit buffer(SPTX).
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1.

Receive only slave mode in SPI serial communication (4-wire: the SPCR.SPMS bit is 0)

Satisfy one of following.

- SSL0 assertion of next data was detected.
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1.

Receive only slave mode in clock synchronous (3-wire: the SPCR.SPMS bit is 1)

Satisfy one of following.

- The first edge of RSPCK of the next data was detected.
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1.

### SPRF flag (SPI Receive Buffer Full Flag)

The SPRF flag indicates the status of the receive buffer for the SPI Data Register (SPDR/SPDR_HA).

[Setting condition]

- Received data with the frame size specified by the SPDCR.SPFC[1:0] bits have been transferred to the SPDR from the shift register while the SPRF flag is 0. And satisfy one of following. However, the SPRF flag does not change from 0 to 1 while the OVRF flag = 1.
  - The SPCR.TXMD bit is 0 (transmit-receive master mode, transmit-receive slave mode, or receive only slave mode)
  - The SPCR.MSTR bit is 0 and the SPCR3.ETXMD bit is 1 (receive only slave mode)

[Clearing condition]

- When received data is read from the SPDR/SPDR_HA.

## 30.2.5 SPDR/SPDR_HA/SPDR_BY : SPI Data Register

Base address: SPI0 = 0x4011_A000

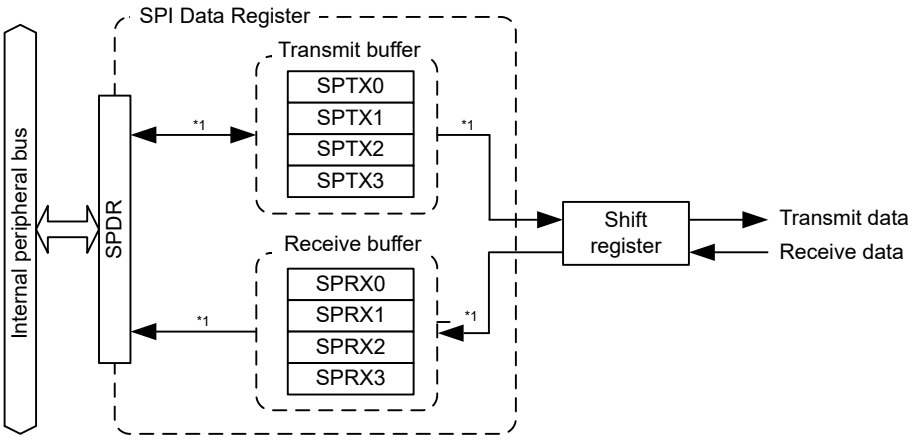
Offset address: 0x04

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | SPI Data | R/W |

SPDR/SPDR_HA/SPDR_BY is the interface with the buffers that hold data for transmission and reception by the SPI. When accessing this register in words (the SPDCR.SPLW bit is 1), access SPDR. When accessing it in halfwords (the SPLW bit is 0), access SPDR_HA. When accessing it in byte (the SPDCR.SPBYT bit is 1), access SPDR_BY.

The transmit buffer (SPTX) and receive buffer (SPRX) are independent but are both mapped to SPDR/SPDR_HA. Figure 30.2 shows the configuration of the SPDR/SPDR_HA register.




Note: SPDR = SPDR/SPDR_HA/SPDR_BY
Note 1. The destination buffer and stage for access is automatically switched by the hardware.

**Figure 30.2 Configuration of SPDR/SPDR_HA/SPDR_BY**

The transmit and receive buffers each have one stages. The four stages of the buffer are all mapped to the single address of SPDR/SPDR_HA/SPDR_BY.

Data written to SPDR/SPDR_HA/SPDR_BY is written to a transmit-buffer stage (SPTXn) (n = 0 to 3), and then transmitted from the buffer. The receive buffer holds received data on completion of reception. The receive buffer is not updated if an overrun is generated.

Additionally, if the data length is not 32 bits, bits not referred to in SPTXn (n = 0 to 3) are stored in the associated bits in SPRXn (n = 0 to 3). For example, if the data length is 9 bits, the received data is stored in the SPRXn[8:0] bits, and the SPTXn[31:9] bits are stored in the SPRXn[31:9] bits.

### (1) Bus interface

SPDR/SPDR_HA/SPDR_BY is an interface with 32-bit wide transmit and receive buffers, each of which has one stages, for a total of 32 bytes. The 32 bytes are mapped to the 4-byte address space for SPDR/SPDR_HA/SPDR_BY. Additionally, the unit of access for SPDR/SPDR_HA/SPDR_BY is selected by the SPI Word Access/Halfword Access Specification bit in the

SPI Data Control Register (SPDCR.SPLW). SPDR can also be accessed with the access size specified by the SPI Byte Access bit in the SPI Data Control Register (SPDCR.SPBYT).

Flush the transmission data at the LSB end of the register, and store the received data at the LSB end.

The following sections describe the operations involved in writing to and reading from SPDR/SPDR_HA/SPDR_BY.

### Writing

Data written to SPDR/SPDR_HA/SPDR_BY is written to a transmit buffer (SPTXn). This is not affected by the value of the SPDCR.SPRDTD bit, unlike when reading from SPDR/SPDR_HA/SPDR_BY. The transmit buffer includes a transmit buffer write pointer that is automatically updated to reference the next stage each time data is written to SPDR/SPDR_HA/SPDR_BY.

Figure 30.3 shows the configuration of the bus interface with the transmit buffer when writing to SPDR/SPDR_HA/SPDR_BY.

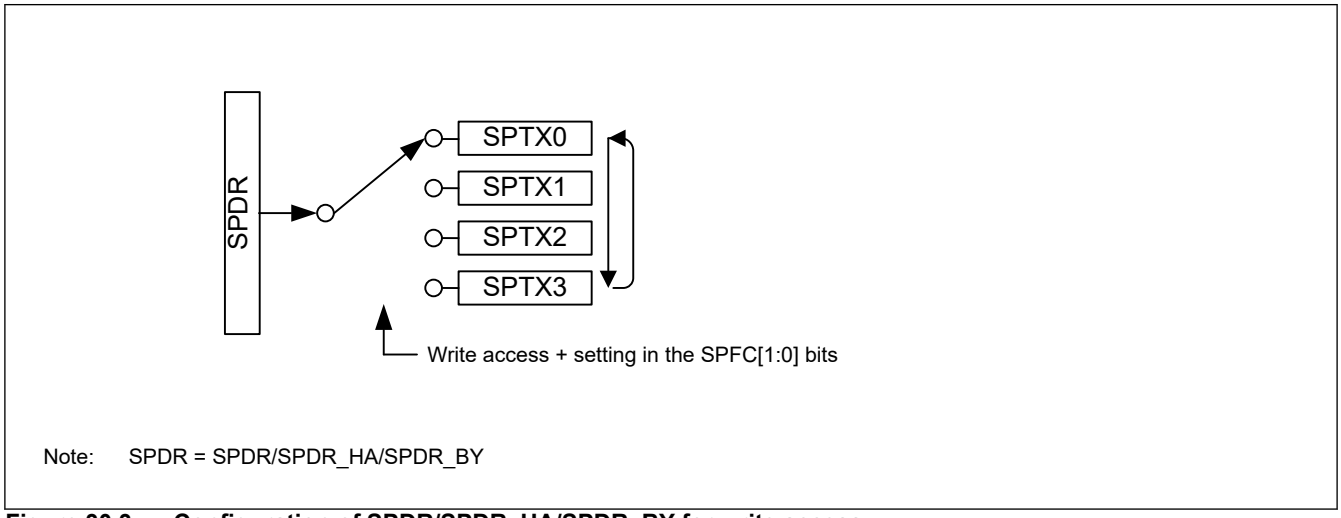



**Figure 30.3 Configuration of SPDR/SPDR_HA/SPDR_BY for write access**

The sequence for switching the transmit buffer write pointer differs with the setting of the number of frames specification bits in the SPI Data Control Register (SPDCR.SPFC[1:0]). The relationship of the SPFC[1:0] setting and the sequence of pointer switching from SPTX0 to SPTX3 is as follows:

- When SPFC[1:0] = 00b: SPTX0 → SPTX0 → SPTX0 → …
- When SPFC[1:0] = 01b: SPTX0 → SPTX1 → SPTX0 → SPTX1 → …
- When SPFC[1:0] = 10b: SPTX0 → SPTX1 → SPTX2 → SPTX0 → SPTX1 → …
- When SPFC[1:0] = 11b: SPTX0 → SPTX1 → SPTX2 → SPTX3 → SPTX0 → SPTX1 → …

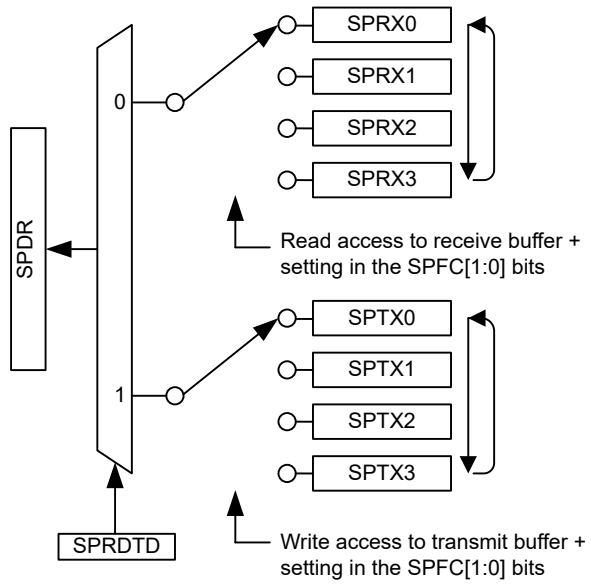
When 1 is written to the SPI Function Enable bit in the SPI Control Register (SPCR.SPE) while the bit is 0, SPTX0 is the destination for the next write.

When writing to the transmit buffer (SPTXn) after generating the transmit buffer empty interrupt (when SPSR.SPTEF is 1), write the number of frames set in SPFC[1:0] in the SPI Data Control Register (SPDCR). Even when the specified number of frames is written to the transmit buffer (SPTXn), the value of the buffer is not updated after completion of the writing and before generation of the next transmit buffer empty interrupt (when SPTEF is 0).

### Reading

SPDR/SPDR_HA/SPDR_BY can be accessed to read the value of a receive buffer (SPRXn) or a transmit buffer (SPTXn). The setting in the SPI Receive/Transmit Data Select bit in the SPI Data Control Register (SPDCR.SPRDTD) selects whether reading is of the receive or transmit buffer. The sequence of reading the SPDR/SPDR_HA/SPDR_BY register is controlled by the independent receive buffer and transmit buffer read pointers.

Figure 30.4 shows the configuration of the bus interface with the receive and transmit buffers for reading from SPDR/SPDR_HA/SPDR_BY.




Note: SPDR = SPDR/SPDR_HA/SPDR_BY

**Figure 30.4 Configuration of SPDR/SPDR_HA/SPDR_BY for read access**

Reading the receive buffer switches the receive buffer read pointer to the next buffer automatically. The switching sequence for the receive buffer read pointer is the same as that for the transmit buffer write pointer. However, when 1 is written to the SPI Function Enable bit in the SPI Control Register (SPCR.SPE) while the value of the bit is 1, SPRX0 is referenced by the buffer read pointer for the next read.

The transmit buffer read pointer is updated when writing to SPDR/SPDR_HA/SPDR_BY, but not updated when reading from the transmit buffer. When reading from the transmit buffer, the value most recently written to SPDR/SPDR_HA/SPDR_BY is read.

After a transmit buffer empty interrupt is generated, reading from the transmit buffer returns all 0s after the completion of writing the number of frames of data specified in the SPDCR.SPFC[1:0] bits, until the next buffer empty interrupt is generated (when SPTEF is 0).

## 30.2.6 SPSCR : SPI Sequence Control Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x08

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | SPSLN[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | SPSLN[2:0] | SPI Sequence Length Specification<br>The sequence length that is set in these bits determines the order in which the SPCMD0 to SPCMD07 registers are referenced. The setting defines the relationship between the sequence length and the SPCMD0 to SPCMD7 registers referenced by the SPI. In slave mode, the SPI references SPCMD0.<br>0 0 0: Sequence Length is 1 (Referenced SPCMDn, n = 0→0→…)<br>0 0 1: Sequence Length is 2 (Referenced SPCMDn, n = 0→1→0→…)<br>0 1 0: Sequence Length is 3 (Referenced SPCMDn, n = 0→1→2→0→…)<br>0 1 1: Sequence Length is 4 (Referenced SPCMDn, n = 0→1→2→3→0→…)<br>1 0 0: Sequence Length is 5 (Referenced SPCMDn, n = 0→1→2→3→4→0→…)<br>1 0 1: Sequence Length is 6 (Referenced SPCMDn, n = 0→1→2→3→4→5→0→…)<br>1 1 0: Sequence Length is 7 (Referenced SPCMDn, n = 0→1→2→3→4→5→6→0→…)<br>1 1 1: Sequence Length is 8 (Referenced SPCMDn, n = 0→1→2→3→4→5→6→7→0→…) | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

SPSCR specifies the sequence length when the SPI operates in master mode. Before changing the SPSLN[2:0] bits while both the SPCR.MSTR and SPCR.SPE bits are 1, check that the SPSR.IDLNF flag is 0.

### SPSLN[2:0] bits (SPI Sequence Length Specification)

The SPSLN[2:0] bits specify the sequence length when the SPI in master mode performs sequential operations. The SPI in master mode changes the SPCMD0 to SPCMD7 registers to be referenced, and the order in which they are referenced is based on this sequence length setting. In slave mode, SPCMD0 is referenced.

## 30.2.7 SPSSR : SPI Sequence Status Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x09

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | SPECM[2:0] | | | — | SPCP[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | SPCP[2:0] | SPI Command Pointer<br>0 0 0: SPCMD0<br>0 0 1: SPCMD1<br>0 1 0: SPCMD2<br>0 1 1: SPCMD3<br>1 0 0: SPCMD4<br>1 0 1: SPCMD5<br>1 1 0: SPCMD6<br>1 1 1: SPCMD7 | R |
| 3 | — | This bit is read as 0. | R |
| 6:4 | SPECM[2:0] | SPI Error Command<br>0 0 0: SPCMD0<br>0 0 1: SPCMD1<br>0 1 0: SPCMD2<br>0 1 1: SPCMD3<br>1 0 0: SPCMD4<br>1 0 1: SPCMD5<br>1 1 0: SPCMD6<br>1 1 1: SPCMD7 | R |
| 7 | — | This bit is read as 0. | R |

SPSSR indicates the sequence control status when the SPI operates in master mode. Any writes to SPSSR are ignored.

### SPCP[2:0] bits (SPI Command Pointer)

The SPCP[2:0] bits indicate the SPCMDm register that is referenced to by the pointer during sequence control by the SPI. For the SPI sequence control, see section 30.3.11.1. Master mode operation.

### SPECM[2:0] bits (SPI Error Command)

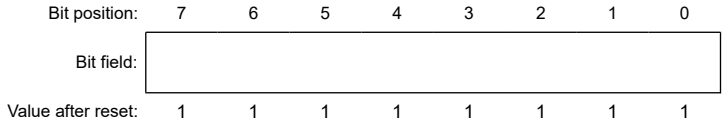
The SPECM[2:0] bits indicate the SPCMDm register that is specified in the SPCP[2:0] bits when an error is detected during sequence control by the SPI. The SPI updates the SPECM[2:0] bits only when an error is detected. If both the SPSR.OVRF and SPSR.MODF flags are 0 and there is no error, the values of the SPECM[2:0] bits have no meaning.

For the SPI error detection function, see section 30.3.9. Error Detection. For the SPI sequence control, see section 30.3.11.1. Master mode operation.

## 30.2.8 SPBR : SPI Bit Rate Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x0A




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | n/a | Bit rate | R/W |

SPBR sets the bit rate in master mode.

When the SPI is in slave mode, the bit rate depends on the bit rate of the input clock, regardless of the settings in SPBR and the SPCMDm.BRDV[1:0] bits (bit rate division setting). Use bit rates that satisfy the electrical characteristics of the device.

The bit rate is determined by combinations of the SPBR and SPCMDm.BRDV[1:0] settings in the SPI Command Register. The equation for calculating the bit rate is given as follows:

$$\text{Bit rate} = \frac{f(\text{PCLK})}{2 \times (n+1) \times 2^{N}}$$

( PCLK = PCLKA)

In the equation, n denotes an SPBR setting (0, 1, 2, …, 255), and N denotes a BRDV[1:0] setting (0, 1, 2, 3).

Table 30.3 lists examples of the relationship between the SPBR settings, the BRDV[1:0] settings, and bit rates.

**Table 30.3 Relationship between SPBR settings, BRDV[1:0] settings, and bit rates**

| SPBR(n) | BRDV[1:0] bits (N) | Division ratio | Bit rate |
|---|---|---|---|
| | | | PCLKA = 32 MHz |
| 0 | 0 | 2 | 16.0 Mbps |
| 1 | 0 | 4 | 8.00 Mbps |
| 2 | 0 | 6 | 5.33 Mbps |
| 3 | 0 | 8 | 4.00 Mbps |
| 4 | 0 | 10 | 3.20 Mbps |
| 5 | 0 | 12 | 2.67 Mbps |
| 5 | 1 | 24 | 1.33 Mbps |
| 5 | 2 | 48 | 667 kbps |
| 5 | 3 | 96 | 333 kbps |
| 255 | 3 | 4096 | 7.81 kbps |

## 30.2.9 SPDCR : SPI Data Control Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x0B

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | SPBYT | SPLW | SPRDTD | — | — | SPFC[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | SPFC[1:0] | Number of Frames Specification<br>0 0: 1 frame<br>0 1: 2 frames<br>1 0: 3 frames<br>1 1: 4 frames | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | SPRDTD | SPI Receive/Transmit Data Select<br>0: Read SPDR/SPDR_HA values from receive buffer<br>1: Read SPDR/SPDR_HA values from transmit buffer, but only if the transmit buffer is empty | R/W |
| 5 | SPLW | SPI Word Access/Halfword Access Specification<br>0: Set SPDR_HA to valid for halfword access<br>1: Set SPDR to valid for word access | R/W |
| 6 | SPBYT | SPI Byte Access Specification<br>0: SPDR/SPDR_HA is accessed in halfword or word (SPLW is valid)<br>1: SPDR_BY is accessed in byte (SPLW is invalid) | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |

The SPI Data Control Register (SPDCR) is used to read the number of frames that can be stored in the SPDR register,read the SPDR register, and to set the access width for the SPDR register to word access, halfword access, or byte access. Up to four frames can be transmitted or received in one round of transmission or reception. The amount of data in each transfer is controlled by the combination of the SPCMDm.SPB[3:0] bits, the SPSCR.SPSLN[2:0] bits, and the SPFC[1:0] bits.

When changing the SPFC[1:0] bits while the SPCR.SPE bit is 1, check that the SPSR.IDLNF flag is 0.

### SPFC[1:0] bits (Number of Frames Specification)

The SPFC[1:0] bits specify the number of frames that can be stored in SPDR/SPDR_HA per transfer activation. Up to four frames can be transmitted or received in one round of transmission or reception.

When the number of transmission data frames specified in the SPFC[1:0] bits is written to the SPDR/SPDR_HA register, SPI clears the SPSR.SPTEF flag to 0 and begins transmitting. After that, when the number of transmission data frames specified in the SPFC[1:0] bits is transmitted to the shift register, the SPI generates the transmit buffer empty interrupt (SPSR.SPTEF sets to 1).

When the number of data frames specified in the SPFC[1:0] bits is received, the SPI generates the receive buffer full interrupt (SPSR.SPRF sets to 1).

**Table 30.4 Settable combinations of the SPSLN[2:0] and SPFC[1:0] bits (1 of 2)**

| Setting | SPSLN[2:0] | SPFC[1:0] | Number of frames in a single sequence | Number of frames at which transmission or receive buffer is filled |
|---|---|---|---|---|
| 1-1 | 000b | 00b | 1 | 1 |
| 1-2 | 000b | 01b | 2 | 2 |
| 1-3 | 000b | 10b | 3 | 3 |
| 1-4 | 000b | 11b | 4 | 4 |
| 2-1 | 001b | 01b | 2 | 2 |

**Table 30.4 Settable combinations of the SPSLN[2:0] and SPFC[1:0] bits (2 of 2)**

| Setting | SPSLN[2:0] | SPFC[1:0] | Number of frames in a single sequence | Number of frames at which transmission or receive buffer is filled |
|---|---|---|---|---|
| 2-2 | 001b | 11b | 4 | 4 |
| 3 | 010b | 10b | 3 | 3 |
| 4 | 011b | 11b | 4 | 4 |
| 5 | 100b | 00b | 5 | 1 |
| 6 | 101b | 00b | 6 | 1 |
| 7 | 110b | 00b | 7 | 1 |
| 8 | 111b | 00b | 8 | 1 |

### SPRDTD bit (SPI Receive/Transmit Data Select)

The SPRDTD bit selects whether the SPDR/SPDR_HA reads values from the receive buffer or from the transmit buffer. If reading is from the transmit buffer, the last value written to SPDR/SPDR_HA register is read. Read the transmit buffer after an SPI transmit buffer empty interrupt is generated until data of frames specified by SPFC[1:0] has been written (while the SPSR.SPTEF flag is 1).

For details, see section 30.2.5. SPDR/SPDR_HA/SPDR_BY : SPI Data Register.

### SPLW bit (SPI Word Access/Halfword Access Specification)

The SPLW bit specifies the access width for SPDR. Access to SPDR_HA in halfwords is valid when the SPLW bit is 0 and access to SPDR in words is valid when the SPLW bit is 1. Also, when this bit is 0, set the SPI data length setting bits, SPCMDm.SPB[3:0], from 8 to 16 bits. Do not perform any operations when a data length of 20, 24, or 32 bits is specified.

### SPBYT bit (SPI Byte Access Specification)

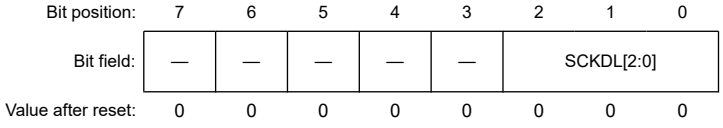
The SPBYT bit is used to set the data width of access to the SPI Data Register (SPDR). When SPBYT = 0, use word or half word access to SPDR/SPDR_HA. When SPBYT = 1 (in that case, SPLW is invalid), use byte access to SPDR_BY.

When SPBYT = 1, set the SPI data length bits (SPB[3:0]) in the SPI Command Register m (SPCMDm) to 8 bits. If SPB[3:0] are set to 9 to 16, 20, 24, or 32 bit, subsequent operation is not guaranteed.

## 30.2.10 SPCKD : SPI Clock Delay Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x0C




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | SCKDL[2:0] | RSPCK Delay Setting<br>0 0 0: 1 RSPCK<br>0 0 1: 2 RSPCK<br>0 1 0: 3 RSPCK<br>0 1 1: 4 RSPCK<br>1 0 0: 5 RSPCK<br>1 0 1: 6 RSPCK<br>1 1 0: 7 RSPCK<br>1 1 1: 8 RSPCK | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

SPCKD specifies the RSPCK delay, the period from the beginning of SSLni signal assertion to RSPCK oscillation, when the SPCMDm.SCKDEN bit is 1.

### SCKDL[2:0] bits (RSPCK Delay Setting)

The SCKDL[2:0] bits specify an RSPCK delay value when the SPCMDm.SCKDEN bit is 1. When using the SPI in slave mode, set the SCKDL[2:0] bits to 000b.

## 30.2.11 SSLND : SPI Slave Select Negation Delay Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x0D

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | SLNDL[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | SLNDL[2:0] | SSL Negation Delay Setting<br>0 0 0: 1 RSPCK<br>0 0 1: 2 RSPCK<br>0 1 0: 3 RSPCK<br>0 1 1: 4 RSPCK<br>1 0 0: 5 RSPCK<br>1 0 1: 6 RSPCK<br>1 1 0: 7 RSPCK<br>1 1 1: 8 RSPCK | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

SSLND specifies the SSL negation delay, the period from the transmission of a final RSPCK edge to the negation of the SSLni signal during a serial transfer by the SPI in master mode. If the contents of SSLND are changed while both the SPCR.MSTR and SPCR.SPE bits are 1, do not perform subsequent operations.

### SLNDL[2:0] bits (SSL Negation Delay Setting)

The SLNDL[2:0] bits specify an SSL negation delay value when the SLNDEN bit in SPCMDn is 1 and the SPI is in master mode. When using the SPI in slave mode, set the SLNDL[2:0] bits to 000b.

## 30.2.12 SPND : SPI Next-Access Delay Register

Base address: SPI0 = 0x4011_A000

Offset address: 0x0E

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | SPNDL[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | SPNDL[2:0] | SPI Next-Access Delay Setting<br>0 0 0: 1 RSPCK + 2 PCLKA<br>0 0 1: 2 RSPCK + 2 PCLKA<br>0 1 0: 3 RSPCK + 2 PCLKA<br>0 1 1: 4 RSPCK + 2 PCLKA<br>1 0 0: 5 RSPCK + 2 PCLKA<br>1 0 1: 6 RSPCK + 2 PCLKA<br>1 1 0: 7 RSPCK + 2 PCLKA<br>1 1 1: 8 RSPCK + 2 PCLKA | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

SPND specifies the next-access delay, the non-active period of the SSLni signal after termination of a serial transfer, when the SPCMDm.SPNDEN bit is 1.

### SPNDL[2:0] bits (SPI Next-Access Delay Setting)

The SPNDL[2:0] bits specify a next-access delay when the SPCMDm.SPNDEN bit is 1. When using the SPI in slave mode, set the SPNDL[2:0] bits to 000b.

## 30.2.13 SPCR2 : SPI Control Register 2

Base address: SPI0 = 0x4011_A000

Offset address: 0x0F

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | SCKASE | PTE | SPIIE | SPOE | SPPE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SPPE | Parity Enable<br>0: Do not add parity bit to transmit data and do not check parity bit of receive data<br>1: When SPCR.TXMD = 0: Add parity bit to transmit data and check parity bit of receive data<br>When SPCR.TXMD = 1: Add parity bit to transmit data but do not check parity bit of receive data | R/W |
| 1 | SPOE | Parity Mode<br>0: Select even parity for transmission and reception<br>1: Select odd parity for transmission and reception | R/W |
| 2 | SPIIE | SPI Idle Interrupt Enable<br>0: Disable idle interrupt requests<br>1: Enable idle interrupt requests | R/W |
| 3 | PTE | Parity Self-Testing<br>0: Disable self-diagnosis function of the parity circuit<br>1: Enable self-diagnosis function of the parity circuit | R/W |
| 4 | SCKASE | RSPCK Auto-Stop Function Enable<br>0: Disable RSPCK auto-stop function<br>1: Enable RSPCK auto-stop function | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

### SPPE bit (Parity Enable)

The SPPE bit enables or disables the parity function.

When the SPCR.TXMD bit is 0 and this bit is 1, the parity bit is added to transmit data and parity checking is performed for receive data.

When the SPCR.TXMD bit is 1 and this bit is 1, the parity bit is added to transmit data but parity checking is not performed for receive data.

### SPOE bit (Parity Mode)

The SPOE bit specifies odd or even parity.

When even parity is set, parity bit addition is performed so that the total number of bits whose value is 1 in the transmit or receive character plus the parity bit is even. Similarly, when odd parity is set, parity bit addition is performed so that the total number of bits whose value is 1 in the transmit or receive character plus the parity bit is odd.

The SPOE bit is only valid when the SPPE bit is 1.

### SPIIE bit (SPI Idle Interrupt Enable)

The SPIIE bit enables or disables the generation of SPI idle interrupt requests when an idle state is detected in the SPI and the SPSR.IDLNF flag clears is set to 0.

### PTE bit (Parity Self-Testing)

The PTE bit enables self-diagnosis of the parity circuit to check whether the parity function is operating correctly.

### SCKASE bit (RSPCK Auto-Stop Function Enable)

The SCKASE bit enables or disables the RSPCK auto-stop function. When this function is enabled, the RSPCK clock is stopped before an overrun error occurs, when data is received in master mode. For details, see section 30.3.9.1. Overrun errors.

## 30.2.14 SPCMDm : SPI Command Register m (m = 0 to 7)

Base address: SPI0 = 0x4011_A000

Offset address: 0x10 + 0x02 × m

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SCKDEN | SLNDEN | SPNDEN | LSBF | SPB[3:0] | | | | SSLKP | SSLA[2:0] | | | BRDV[1:0] | | CPOL | CPHA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CPHA | RSPCK Phase Setting<br>0: Select data sampling on leading edge, data change on trailing edge<br>1: Select data change on leading edge, data sampling on trailing edge | R/W |
| 1 | CPOL | RSPCK Polarity Setting<br>0: Set RSPCK low during idle<br>1: Set RSPCK high during idle | R/W |
| 3:2 | BRDV[1:0] | Bit Rate Division Setting<br>0 0: Base bit rate<br>0 1: Base bit rate divided by 2<br>1 0: Base bit rate divided by 4<br>1 1: Base bit rate divided by 8 | R/W |
| 6:4 | SSLA[2:0] | SSL Signal Assertion Setting<br>0 0 0: SSL0<br>0 0 1: SSL1<br>0 1 0: SSL2<br>0 1 1: SSL3<br>Others: Setting prohibited | R/W |
| 7 | SSLKP | SSL Signal Level Keeping<br>0: Negate all SSL signals on completion of transfer<br>1: Keep SSL signal level from the end of transfer until the beginning of the next access | R/W |
| 11:8 | SPB[3:0] | SPI Data Length Setting<br>0x0: 20 bits<br>0x1: 24 bits<br>0x2: 32 bits<br>0x3: 32 bits<br>0x8: 9 bits<br>0x9: 10 bits<br>0xA: 11 bits<br>0xB: 12 bits<br>0xC: 13 bits<br>0xD: 14 bits<br>0xE: 15 bits<br>0xF: 16 bits<br>Others: 8 bits | R/W |
| 12 | LSBF | SPI LSB First<br>0: MSB-first<br>1: LSB-first | R/W |
| 13 | SPNDEN | SPI Next-Access Delay Enable<br>0: Select next-access delay of 1 RSPCK + 2 PCLKA<br>1: Select next-access delay equal to the setting in the SPI Next-Access Delay Register (SPND) | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14 | SLNDEN | SSL Negation Delay Setting Enable<br>0: Select SSL negation delay of 1 RSPCK<br>1: Select SSL negation delay equal to the setting in the SPI Slave Select Negation Delay Register (SSLND) | R/W |
| 15 | SCKDEN | RSPCK Delay Setting Enable<br>0: Select RSPCK delay of 1 RSPCK<br>1: Select RSPCK delay equal to the setting in the SPI Clock Delay Register (SPCKD) | R/W |

The SPCMDm registers specify the transfer format for the SPI in master mode. Each channel has eight SPCMDm (m = 0 to 7). Some of the bits in the SPCMD0 registers are used to set the transfer mode for the SPI in slave mode. The SPI in master mode sequentially references the SPCMDm registers based on the settings in the SPSCR.SPSLN[2:0] bits and executes the serial transfer that is set in the referenced SPCMDm registers.

Set the SPCMDm registers while the transmit buffer is empty (SPSR.SPTEF is 1 and data for the next transfer is not set) and before the setting of the data to be transmitted when that SPCMDm registers is referenced.

The SPCMDm registers referenced by the SPI in master mode can be checked by means of the SPSSR.SPCP[2:0] bits.

#### CPHA bit (RSPCK Phase Setting)

The CPHA bit selects the RSPCK phase of the SPI in master or slave mode. Data communications between SPI modules require the same RSPCK phase setting between the modules.

#### CPOL bit (RSPCK Polarity Setting)

The CPOL bit selects the RSPCK polarity of the SPI in master or slave mode. Data communications between SPI modules require the same RSPCK polarity setting between the modules.

#### BRDV[1:0] bits (Bit Rate Division Setting)

The BRDV[1:0] bits determine the bit rate in combination with the settings in the SPBR register. (see section 30.2.8. SPBR : SPI Bit Rate Register). The SPBR settings determine the base bit rate. The BRDV[1:0] setting selects the bit rate obtained by dividing the base bit rate by 1, 2, 4, or 8. Different BRDV[1:0] bit settings can be specified in the SPCMD0 register. This enables execution of serial transfers at different bit rates for each command.

#### SSLA[2:0] bits (SSL Signal Assertion Setting)

The SSLA[2:0] bits control the SSLni signal assertion when the SPI performs serial transfers in master mode. When an SSLni signal is asserted, its polarity is determined by the value set in the associated SSLP. When the SSLA[2:0] bits are set to 000b in multi-master mode, serial transfers are performed with all the SSL signals in the negated state (as the SSLn0 pin acts as input).

When using the SPI in slave mode, set the SSLA[2:0] bits to 000b.

#### SSLKP bit (SSL Signal Level Keeping)

When the SPI in master mode performs a serial transfer, the SSLKP bit specifies whether the SSLni signal level for the current command is to be kept or negated between the SSL negation associated with the current command and the SSL assertion associated with the next command. Setting the SSLKP bit to 1 enables a burst transfer. For details, see section 30.3.11.1. Master mode operation. When using the SPI in slave mode, set the SSLKP bit to 0.

#### SPB[3:0] bits (SPI Data Length Setting)

The SPB[3:0] bits specify the transfer data length for the SPI in master or slave mode.

#### LSBF bit (SPI LSB First)

The LSBF bit specifies the data format of the SPI in master or slave mode to MSB-first or LSB-first.

#### SPNDEN bit (SPI Next-Access Delay Enable)

The SPNDEN bit specifies the next-access delay, the period from the time the SPI in master mode terminates a serial transfer and sets the SSLni signal inactive until the SPI enables the SSLni signal assertion for the next access. If the SPNDEN bit is 0, the SPI sets the next-access delay to 1 RSPCK + 2 PCLKA. If the SPNDEN bit is 1, the SPI inserts a next-access delay according to the SPND setting.

When using the SPI in slave mode, set the SPNDEN bit to 0.

### SLNDEN bit (SSL Negation Delay Setting Enable)

The SLNDEN bit specifies the SSL negation delay, the period from the time the SPI in master mode stops RSPCK oscillation until the SPI sets the SSLni signal to inactive. If the SLNDEN bit is 0, the SPI sets the SSL negation delay to 1 RSPCK. If the SLNDEN bit is 1, the SPI negates the SSL signal at the SSL negation delay according to the SSLND setting.

When using the SPI in slave mode, set the SLNDEN bit to 0.

### SCKDEN bit (RSPCK Delay Setting Enable)

The SCKDEN bit specifies the SPI clock delay, the period from the point when the SPI in master mode asserts the SSLni signal until the RSPCK starts oscillation. If the SCKDEN bit is 0, the SPI sets the RSPCK delay to 1 RSPCK. If the SCKDEN bit is 1, the SPI starts the oscillation of RSPCK at an RSPCK delay according to the SPCKD setting.

When using the SPI in slave mode, set the SCKDEN bit to 0.

## 30.2.15 SPDCR2 : SPI Data Control Register 2

Base address: SPI0 = 0x4011_A000

Offset address: 0x20

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | SINV | BYSW |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BYSW | Byte Swap Operating Mode Select<br>0: Byte Swap OFF<br>1: Byte Swap ON | R/W |
| 1 | SINV | Serial Data Invert Bit<br>0: Not invert serial data<br>1: Invert serial data | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

SPI Data Control Register 2 (SPDCR2) is the setting register, that is to swap a transmit/receive data in byte units and to invert serial data. If these bits are modified while the SPI in slave mode is enabled (SPCR.SPE = 1), subsequent operation is not guaranteed.

### BYSW bit (Byte Swap Operating Mode Select)

It is a setting bit, that is to swap a transmit/receive data in byte units. When byte access is valid (SPDCR.SPBYT = 1), byte swap is invalid. When byte swap is valid, parity function must be invalid (SPCR2.SPPE bit = 0). Setting change of BYSW bit must be SPCR.SPE bit = 0.

A data after byte swap is different by a data length (setting of SPCMD.SPB[3:0]).

When byte swap, A data length (setting of SPB[3:0]) must be set to 32 bit or 16bit. Other case of data length (that is 8 to 15, 20, 24 bit length), byte swap is not guaranteed. Before swap and after swap are shown below (length data (32 bit/16 bit)).

- Length data 32 bits (SPB[3:0] = 0010b or 0011b)

  Before swap: [31:24] [23:16] [15:8] [7:0]

  After swap: [7:0] [15:8] [23:16] [31:24]
- Length data 16 bit (SPB[3:0] = 1111b)

  Before swap: [31:24] [23:16]

  After swap: [23:16] [31:24]

When byte access mode (SPDCR.SPBT = 1), byte swap setting is invalid.

When byte swap is valid, set parity function to invalid (SPCR2.SPPE = 0). When the parity function set to valid, the behavior is not guaranteed.

### SINV bit (Serial Data Invert Bit)

This bit is used to invert transmit data and receive data.

When the SINV bit is set to 1, transmit buffer (SPTX) data is inverted to invert transmit data and receive data, and then the inverted data is stored in the receive buffer (SPRX). The parity bit is the value corresponding to the inverted transmission/ reception data.

## 30.2.16 SPCR3 : SPI Control Register 3

Base address: SPI0 = 0x4011_A000

Offset address: 0x21

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | CENDIE | — | — | BFDS | ETXMD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ETXMD | Extended Communication Mode Select<br>0: Full-duplex synchronous or transmit-only serial communications. [the SPCR.TXMD bit is enabled]<br>1: Receive-only serial communications in slave mode (SPCR.MSTR bit = 0). [the SPCR.TXMD bit is disabled]<br>Setting is prohibited in master mode (SPCR.MSTR bit = 1). | R/W |
| 1 | BFDS | Between Burst Transfer Frames Delay Select<br>0: Delay (RSPCK delay, SSL negation delay and next-access delay) between frames is inserted in burst transfer.<br>1: Delay between frames is not inserted in burst transfer. | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | CENDIE | SPI Communication End Interrupt Enable<br>0: Communication end interrupt request is disabled.<br>1: Communication end interrupt request is enabled. | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

SPI control register 3 (SPCR3) is control register for operation mode. If you change the value of ETXMD and BFDS when the SPCR.SPE bit is 1, the SPI operation does not guarantee.

### ETXMD bit (Extended Communication Mode Select)

This bit is valid on slave mode only (the SPCR.MSTR bit is 0). This bit select receive only operation. When the ETXMD bit is 1 on slave mode, the communication is only received not transmit (see section 30.3.6. Data Transfer Modes). When the ETXMD is 1, transmit data empty interrupt can not be used.

The communication state by each mode (master mode, slave mode) is shown as below. It is controlled by the ETXMD bit, the SPCR.MSTR bit and the TXMD bit.

**Table 30.5 SPI communication state (master/slave mode)**

| SPCR.MSTR bit | SPCR3.ETXMD bit | SPCR.TXMD bit | Communication state |
|---|---|---|---|
| 1 | 0 | 0 | Transmit-receive master mode |
| 1 | 0 | 1 | Transmit master mode |
| 0 | 0 | 0 | Transmit-receive slave mode (default) |
| 0 | 0 | 1 | Transmit slave mode |
| 0 | 1 | — | Receive slave mode |

### BFDS bit (Between Burst Transfer Frames Delay Select)

This bit controls whether insert the delay time between the burst transfer frames.

This bit is valid when the SPCMD.SSLKP bit is 1 in master mode (the SPCR.MSTR bit is 1).

This bit should be set to 0 in slave mode. The usage of SSL delay control between transfer frames is shown as below. For details, see (4)Burst transfers.

**Table 30.6 Usage of SSL delay control between transfer frames (Master mode)**

| Transmit | | SPCMD.SSLKP bit | SPCR3.BFDS bit | SSL delay control register*1 (RSPCK clock delay, SSL negation delay, next access delay) |
|---|---|---|---|---|
| Non-burst transmit | | 0 | 0 | Any given value. You can control each delay value according to setting for RSPCK clock delay, SSL negation delay and next access delay. |
| Burst transmit with delay between frames | From the 1st frame to the last previous frame | 1 | 0 | |
| | The last frame | 0 | 0 | |
| Burst transmit with no delay between frames | From the 1st frame to the last previous frame | 1 | 1 | Any given value. But delay is inserted only below.<br>• RSPCK clock delay of the 1st frame<br>• SSL negation delay and next access delay of the last frame |
| | The last frame | 0 | 1 | |

Note 1. Whether the setting value of following bits are valid or not depends on the setting value of the SPCMD.SPNDEN bit (see section 30.2.14. SPCMDm : SPI Command Register m (m = 0 to 7)).
The SPCKD.SCKDL[2:0] bits: RSPCK delay
The SSLND.SLNDL[2:0] bits: SSL negate delay
The SPND.SPNDL[2:0] bits: Next access delay

### CENDIE bit (SPI Communication End Interrupt Enable)

This bit controls generation of a communication end interrupt request.

## 30.3 Operation

In this section, the serial transfer period refers to the period from the beginning of driving valid data to the fetching of the final valid data.

### 30.3.1 Overview of SPI Operation

The SPI is capable of synchronous serial transfers in the following modes:

- Slave mode (SPI operation)
- Single master mode (SPI operation)
- Multi-master mode (SPI operation)
- Slave mode (clock synchronous operation)
- Master mode (clock synchronous operation)

The SPI mode can be selected by using the MSTR, MODFEN, and SPMS bits in SPCR. Table 30.7 lists the relationship between SPI modes and SPCR settings, and a description of each mode.

**Table 30.7 Relationship between SPCR settings and SPI modes (1 of 2)**

| Mode | Slave (SPI operation) | Single-master (SPI operation) | Multi-master (SPI operation) | Slave (clock synchronous operation) | Master (clock synchronous operation) |
|---|---|---|---|---|---|
| MSTR bit setting | 0 | 1 | 1 | 0 | 1 |
| MODFEN bit setting | 0 or 1 | 0 | 1 | 0 | 0 |
| SPMS bit setting | 0 | 0 | 0 | 1 | 1 |
| RSPCKn pins | Input | Output | Output/Hi-Z | Input | Output |
| MOSIn pin | Input | Output | Output/Hi-Z | Input | Output |
| MISOn pin | Output/Hi-Z | Input | Input | Output | Input |

**Table 30.7 Relationship between SPCR settings and SPI modes (2 of 2)**

| Mode | Slave (SPI operation) | Single-master (SPI operation) | Multi-master (SPI operation) | Slave (clock synchronous operation) | Master (clock synchronous operation) |
|---|---|---|---|---|---|
| SSLn0 pins | Input | Output | Input | Hi-Z*1 | Hi-Z*1 |
| SSLn1 to SSLn3 pins | Hi-Z*1 | Output | Output/Hi-Z | Hi-Z*1 | Hi-Z*1 |
| SSL polarity change function | Supported | Supported | Supported | — | — |
| Max transfer rate | PCLKA/4 | PCLKA/2 | PCLKA/2 | PCLKA/4 | PCLKA/2 |
| Clock source | RSPCK input | On-chip baud rate generator | On-chip baud rate generator | RSPCK input | On-chip baud rate generator |
| Clock polarity | Two | | | | |
| Clock phase | Two | Two | Two | One (CPHA = 1) | Two |
| First transfer bit | MSB/LSB | | | | |
| Transfer data length | 8 to 16, 20, 24, 32 bits | | | | |
| Burst transfer | Possible (CPHA = 1) | Possible (CPHA = 0, 1) | Possible (CPHA = 0, 1) | — | — |
| RSPCK delay control | Not supported | Supported | Supported | Not supported | Supported |
| SSL negation delay control | Not supported | Supported | Supported | Not supported | Supported |
| Next-access delay control | Not supported | Supported | Supported | Not supported | Supported |
| Transfer trigger | SSL input active or RSPCK oscillation | Write to transmit buffer on generation of transmit buffer empty interrupt request (SPTEF = 1) | Write to transmit buffer on generation of transmit buffer empty interrupt request (SPTEF = 1) | RSPCK oscillation | Write to transmit buffer on generation of transmit buffer empty interrupt request (SPTEF = 1) |
| Sequence control | Not supported | Supported | Supported | Not supported | Supported |
| Transmit buffer empty detection | Supported*5 | | | | |
| Receive buffer full detection | Supported*2 | | | | |
| Overrun error detection | Supported*2 | Supported*2*4 | Supported*2*4 | Supported*2 | Supported*2 |
| Parity error detection | Supported*3*2 | | | | |
| Mode fault error detection | Supported (MODFEN = 1) | Not supported | Supported | Not supported | Not supported |
| Underrun error detection | Supported*5 | Not supported | Not supported | Supported*5 | Not supported |

Note 1. This function is not supported in this mode.
Note 2. When the SPCR.TXMD bit is 1, detection of receiver buffer full, overrun error, and parity error are not performed.
Note 3. When the SPCR2.SPPE bit is 0, parity error detection is not performed.
Note 4. When the SPCR2.SCKASE bit is 1, overrun error detection does not proceed.
Note 5. When SPI is receive only slave mode, none of transmit buffer empty and underrun error is detected.

## 30.3.2 Controlling the SPI Pins

Based on the settings of the MSTR, MODFEN, and SPMS bits in SPCR and the PmnPFS.NCODR bit for I/O Ports, the SPI can switch pin states. Table 30.8 lists the relationship between pin states and bit settings. Setting the PmnPFS.NCODR bit for an I/O port to 0 selects the CMOS output. Setting it to 1 selects the open-drain output. The I/O port settings must follow this relationship.

**Table 30.8 Relationship between pin states and bit settings**

| Mode | Pin | Pin state*2 | |
|---|---|---|---|
| | | PmnPFS.NCODR bit for I/O ports = 0 | PmnPFS.NCODR bit for I/O ports = 1 |
| Single-master mode (SPI operation) (MSTR = 1, MODFEN = 0, SPMS = 0) | RSPCKn | CMOS output | Open-drain output |
| | SSLn0 to SSLn3 | CMOS output | Open-drain output |
| | MOSIn | CMOS output | Open-drain output |
| | MISOn | Input | Input |
| Multi-master mode (SPI operation) (MSTR = 1, MODFEN = 1, SPMS = 0) | RSPCKn*3 | CMOS output/Hi-Z | Open-drain output/Hi-Z |
| | SSLn0 | Input | Input |
| | SSLn1 to SSLn3*3 | CMOS output/Hi-Z | Open-drain output/Hi-Z |
| | MOSIn*3 | CMOS output/Hi-Z | Open-drain output/Hi-Z |
| | MISOn | Input | Input |
| Slave mode (SPI operation) (MSTR = 0, SPMS = 0) | RSPCKn | Input | Input |
| | SSLn0 | Input | Input |
| | SSLn1 to SSLn3*5 | Hi-Z*1 | Hi-Z*1 |
| | MOSIn | Input | Input |
| | MISOn*4 | CMOS output/Hi-Z | Open-drain output/Hi-Z |
| Master mode (clock synchronous operation) (MSTR = 1, MODFEN = 0, SPMS = 1) | RSPCKn | CMOS output | Open-drain output |
| | SSLn0 to SSLn3*5 | Hi-Z*1 | Hi-Z*1 |
| | MOSIn | CMOS output | Open-drain output |
| | MISOn | Input | Input |
| Slave mode (clock synchronous operation) (MSTR = 0, SPMS = 1) | RSPCKn | Input | Input |
| | SSLn0 to SSLn3*5 | Hi-Z*1 | Hi-Z*1 |
| | MOSIn | Input | Input |
| | MISOn | CMOS output | Open-drain output |

Note 1. This function is not supported in this mode.
Note 2. SPI settings are not reflected in multiplexed pins for which the SPI function is not selected.
Note 3. When SSLn0 is at the active level, the pin state is Hi-Z. Whether or not the input signal is at the active level determines the setting of the SSLP.SSL0P bit.
Note 4. When SSLn0 is at the non-active level or the SPCR.SPE bit is 0, the pin state is Hi-Z. Whether or not the input signal is at the active level determines the setting of the SSLP.SSL0P bit.
Note 5. These pins are available for use as I/O port pins.

The SPI in single-master mode (SPI operation) or multi-master mode (SPI operation) determines the MOSI signal values during the SSL negation period (including the SSL retention period during a burst transfer) based on the MOIFE and MOIFV bit settings in SPPCR, as listed in Table 30.9.

**Table 30.9 MOSI signal value determination during SSL negation**

| MOIFE bit | MOIFV bit | MOSIn signal value during SSL negation |
|---|---|---|
| 0 | 0, 1 | Final data from previous transfer |
| 1 | 0 | Low |
| 1 | 1 | High |

## 30.3.3 SPI System Configuration Examples

### 30.3.3.1 Single-master/single-slave with the MCU as a master

Figure 30.5 shows a single-master/single-slave SPI system configuration example where the MCU is used as a master. In the single-master/single-slave configuration, the SSLni outputs of the MCU (master) are not used. The SSL input of the SPI slave is fixed to the low level, and the SPI slave is maintained in the selected state.*1

Note 1. In the transfer format configured when the SPCMDm.CPHA bit is 0, the SSL signal for some slave devices cannot be fixed to an active level. In this case, always connect the SSLni output of the MCU to the SSL input of the slave device.

The MCU (master) drives the RSPCKn and MOSIn signals. The SPI slave drives the MISO signal.

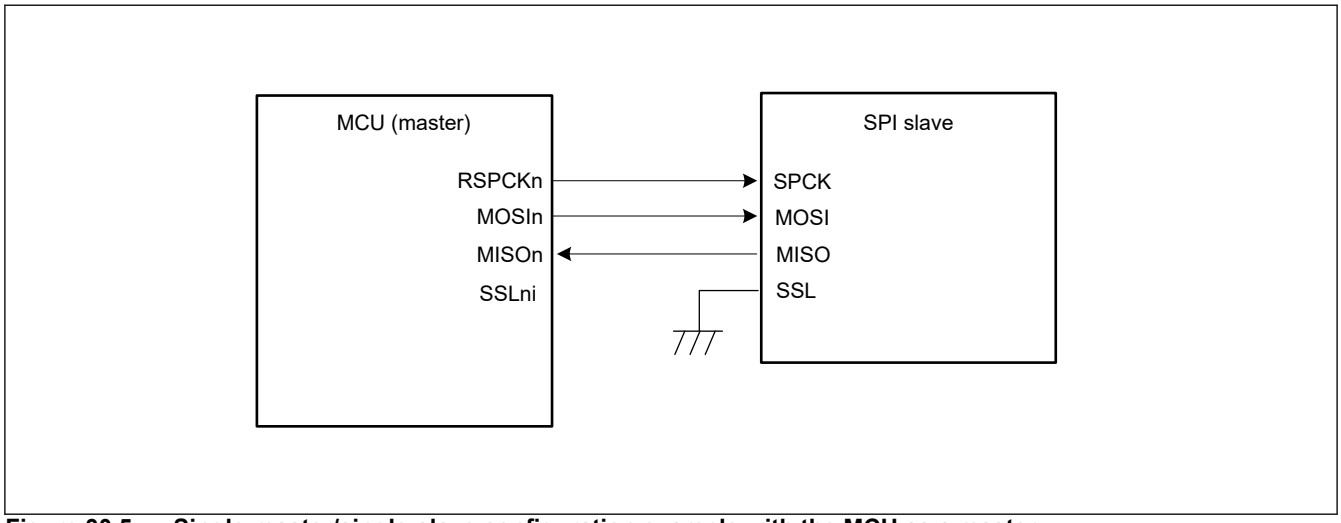



**Figure 30.5 Single-master/single-slave configuration example with the MCU as a master**

### 30.3.3.2 Single-master/single-slave with the MCU as a slave

Figure 30.6 shows a single-master/single-slave SPI system configuration example where the MCU is used as a slave. When the MCU operates as a slave, the SSLn0 pin is used as SSL input. The SPI master drives the RSPCK and MOSI signals. The MCU (slave) drives the MISOn signal.*1

Note 1. When SSLn0 is at a non-active level, the pin state is Hi-Z.

In the single-slave configuration when the SPCMDm.CPHA bit is set to 1, the SSLn0 input of the MCU (slave) is fixed to the low level and the MCU (slave) is maintained in the selected state. This enables serial transfer execution (Figure 30.7). However, the communication end interrupt does not output when SSL0 input is fixed as Figure 30.7.

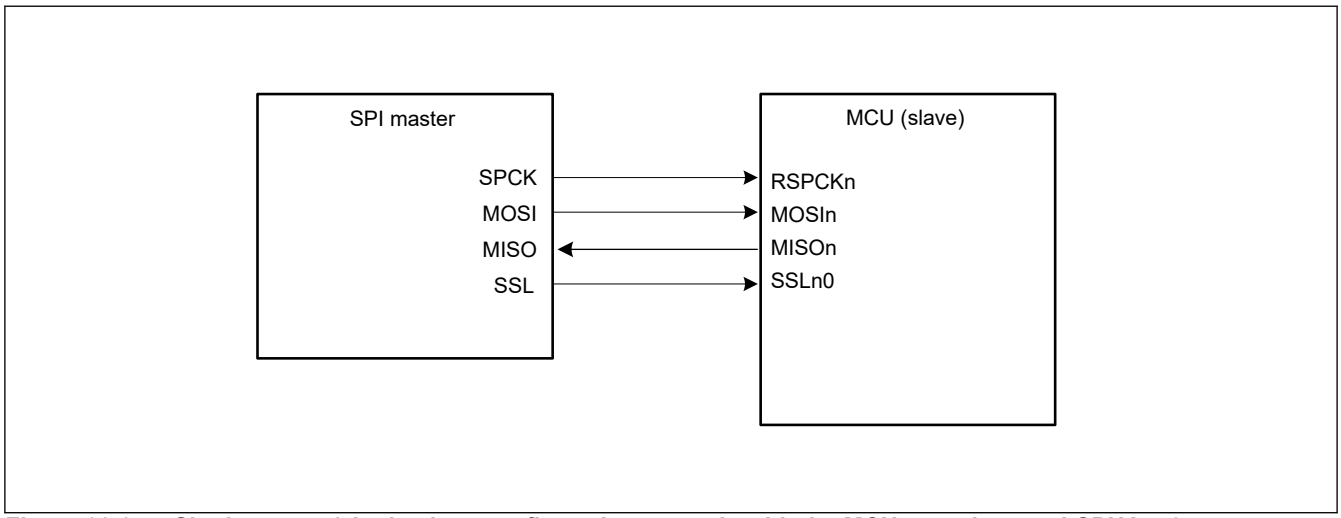



**Figure 30.6 Single-master/single-slave configuration example with the MCU as a slave and CPHA = 0**

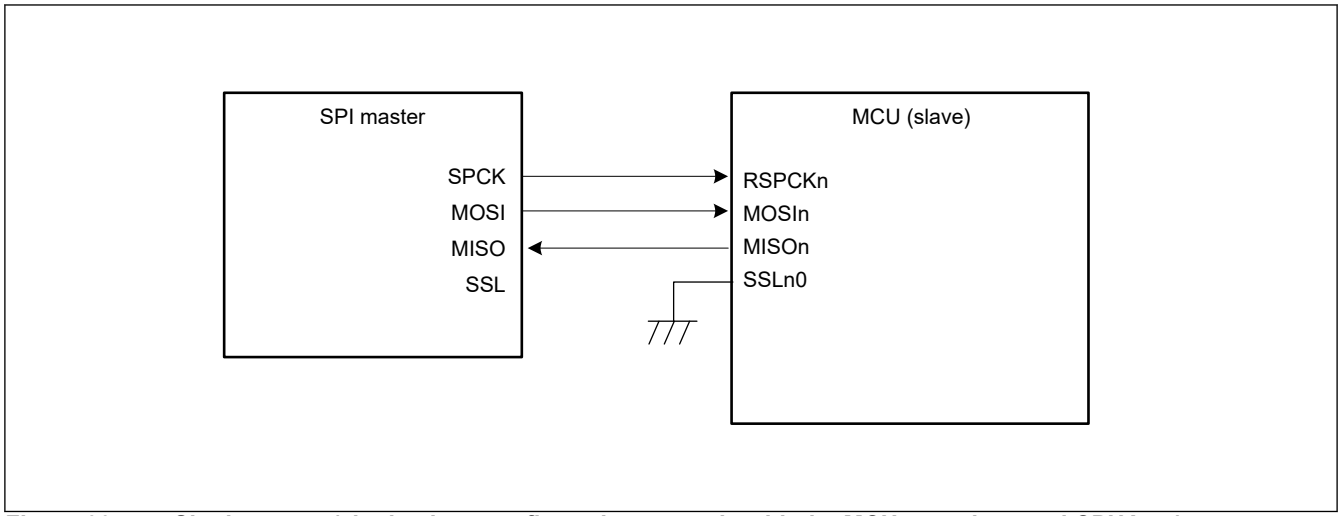



**Figure 30.7 Single-master/single-slave configuration example with the MCU as a slave and CPHA = 1**

### 30.3.3.3 Single-master/multi-slave with the MCU as a master

Figure 30.8 shows a single-master/multi-slave SPI system configuration example where the MCU is used as a master. In this example, the SPI system includes the MCU (master) and four slaves (SPI slave 0 to SPI slave 3).

The RSPCKn and MOSIn outputs of the MCU (master) are connected to the RSPCK and MOSI inputs of SPI slaves 0 to 3. The MISO outputs of SPI slaves 0 to 3 are all connected to the MISOn input of the MCU (master). The SSLn0 to SSLn3 outputs of the MCU (master) are connected to the SSL inputs of SPI slaves 0 to 3, respectively.

The MCU (master) drives the RSPCKn, MOSIn, and SSLn0 to SSLn3 signals. Out of the SPI slaves 0 to 3, the slave that receives low-level input into the SSL input drives the MISO signal.

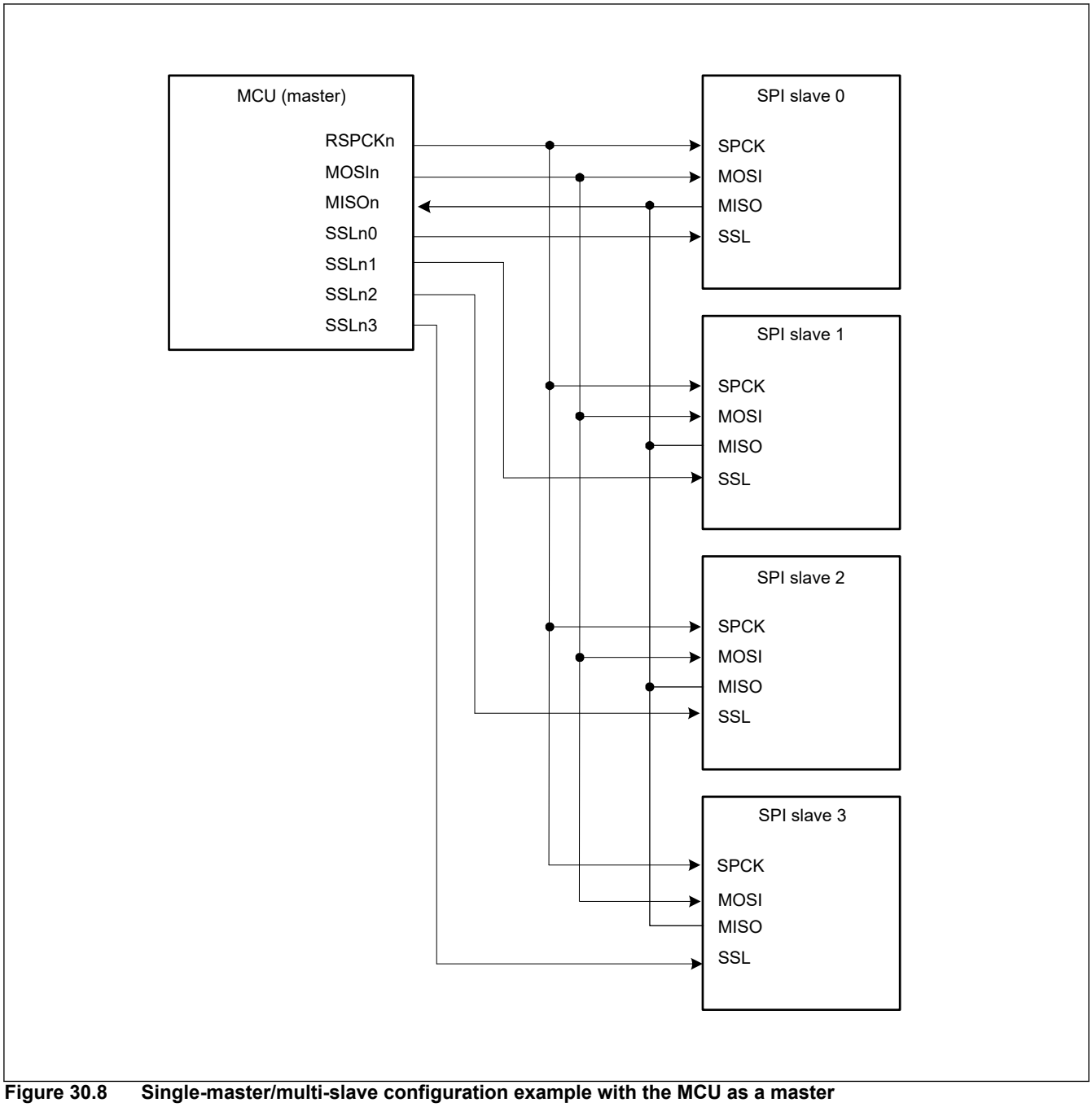



**Figure 30.8 Single-master/multi-slave configuration example with the MCU as a master**

### 30.3.3.4 Single-master/multi-slave with the MCU as a slave

Figure 30.9 shows a single-master/multi-slave SPI system configuration example where the MCU is used as a slave. In this example, the SPI system includes an SPI master and two MCUs (slaves X and Y).

The SPCK and MOSI outputs of the SPI master are connected to the RSPCKn and MOSIn inputs of the MCUs (slaves X and Y). The MISOn outputs of the MCUs (slaves X and Y) are all connected to the MISO input of the SPI master. The SSLX and SSLY outputs of the SPI master are connected to the SSLn0 inputs of the MCUs (slaves X and Y, respectively).

The SPI master drives the SPCK, MOSI, SSLX, and SSLY signals. Of the MCUs (slaves X and Y), the slave that receives low-level input into the SSLn0 input drives the MISOn signal.

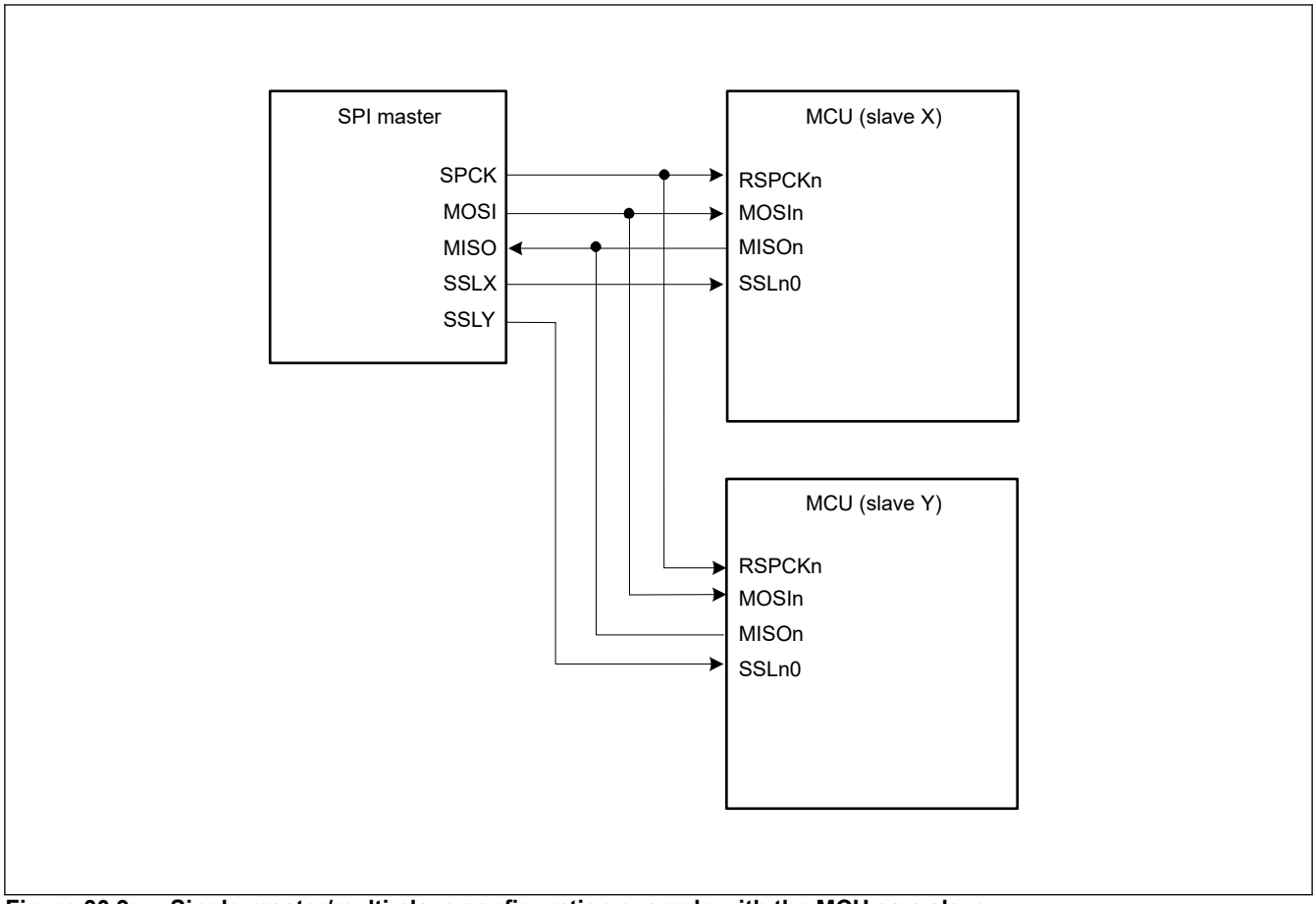



**Figure 30.9 Single-master/multi-slave configuration example with the MCU as a slave**

### 30.3.3.5 Multi-master/multi-slave with the MCU as a master

Figure 30.10 shows a multi-master/multi-slave SPI system configuration example where the MCU is used as a master. In this example, the SPI system includes two MCUs (masters X and Y) and two SPI slaves (SPI slaves 1 and 2).

The RSPCKn and MOSIn outputs of the MCUs (masters X and Y) are connected to the RSPCK and MOSI inputs of SPI slaves 1 and 2. The MISO outputs of SPI slaves 1 and 2 are connected to the MISOn inputs of the MCUs (masters X and Y). Any generic port Y output from the MCU (master X) is connected to the SSLn0 input of the MCU (master Y). Any generic port X output of the MCU (master Y) is connected to the SSLn0 input of the MCU (master X). The SSLn1 and SSLn2 outputs of the MCUs (masters X and Y) are connected to the SSL inputs of the SPI slaves 1 and 2. In this configuration example, because the system can be comprised solely of SSLn0 input, and SSLn1 and SSLn2 outputs for slave connections, the SSLn3 output of the MCU is not required.

The MCU drives the RSPCKn, MOSIn, SSLn1, and SSLn2 signals when the SSLn0 input level is high. When the SSLn0 input level is low, the MCU detects a mode fault error, sets RSPCKn, MOSIn, SSLn1, and SSLn2 to Hi-Z, and releases the SPI bus directly to the other master. Of the SPI slaves 1 and 2, the slave that receives low-level input into the SSL input drives the MISO signal.

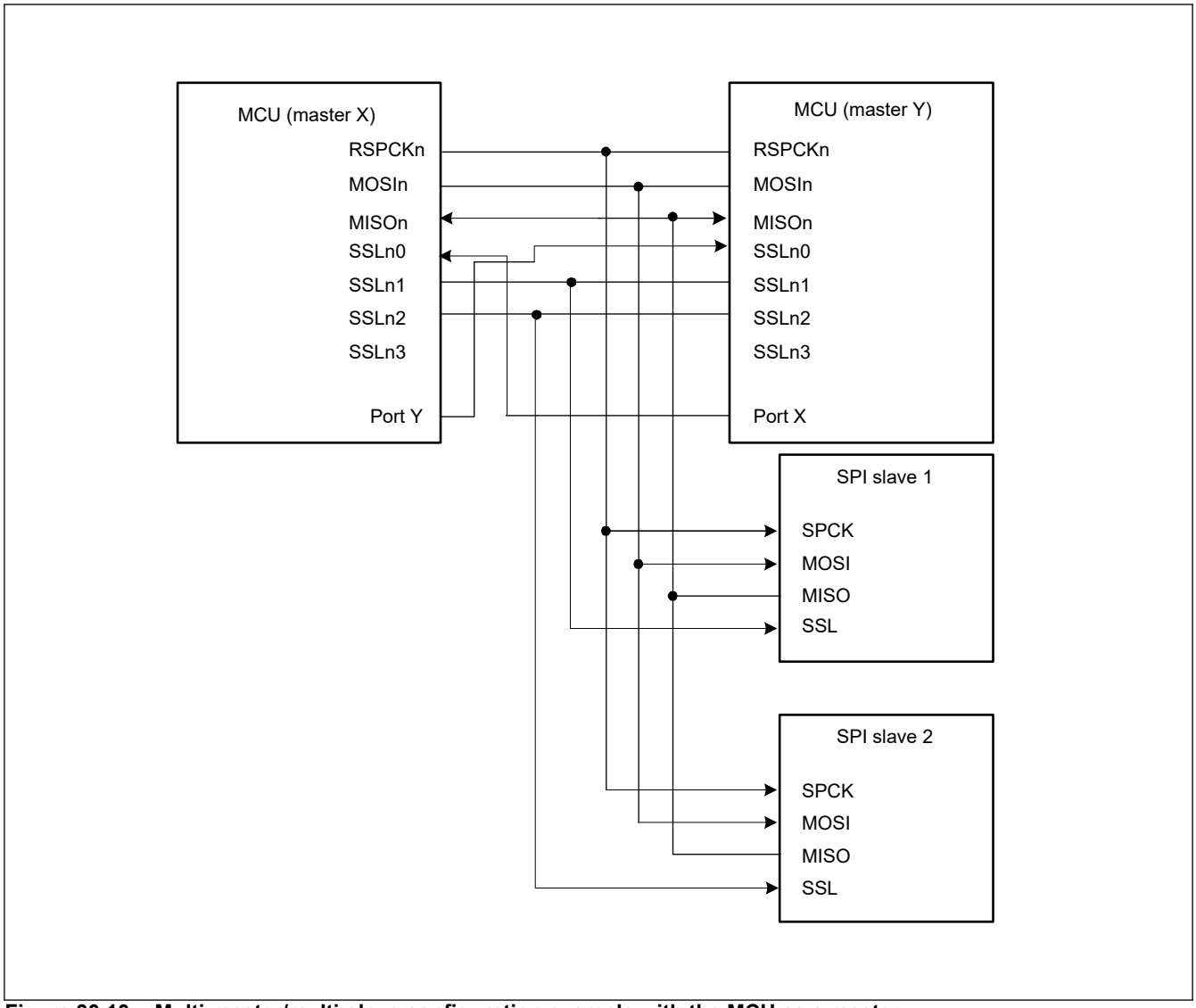



**Figure 30.10 Multi-master/multi-slave configuration example with the MCU as a master**

### 30.3.3.6 Master and slave in clock synchronous mode with the MCU configured as a master

Figure 30.11 shows a master and slave in clock synchronous mode configuration example where the MCU is used as a master. In this configuration, SSLni of the MCU (master) are not used.

The MCU (master) drives the RSPCKn and MOSIn signals. The SPI slave drives the MISO signal.

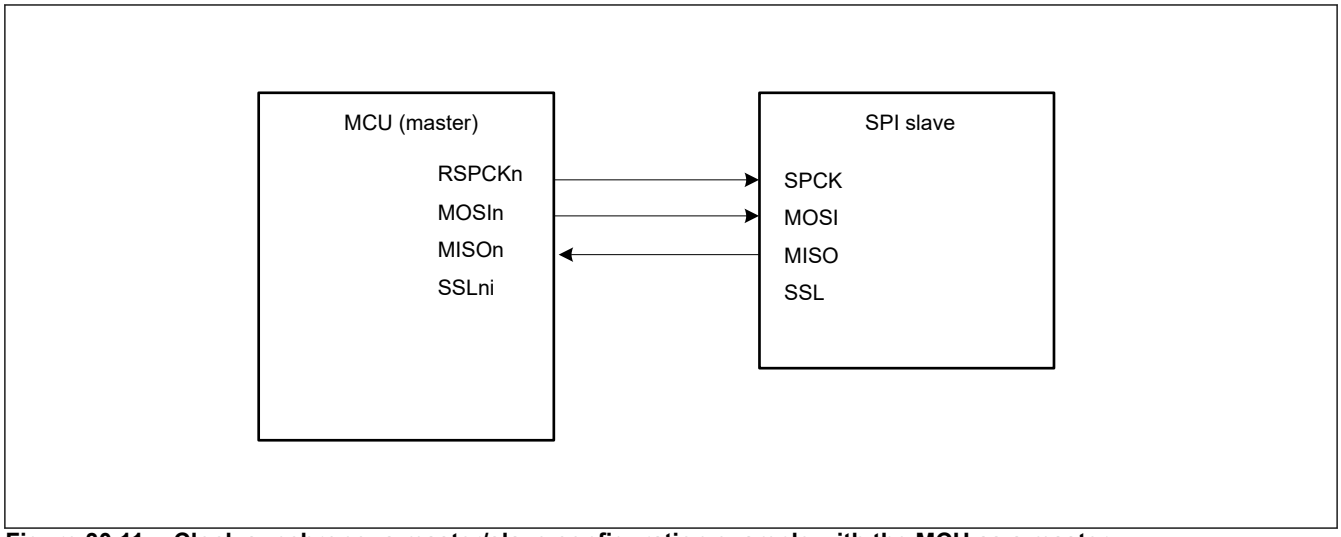



**Figure 30.11 Clock synchronous master/slave configuration example with the MCU as a master**

### 30.3.3.7 Master and slave in clock synchronous mode with the MCU as a slave

Figure 30.12 shows a master and slave in clock synchronous mode configuration example where the MCU is used as a slave. When the MCU operates as a slave (clock synchronous operation), the MCU (slave) drives the MISOn signal and the SPI master drives the SPCK and MOSI signals. In addition, SSLn0 to SSLn3 of the MCU (slave) are not used.

The MCU (slave) can only execute serial transfers in the single-slave configuration when the SPCMDm.CPHA bit is set to 1.

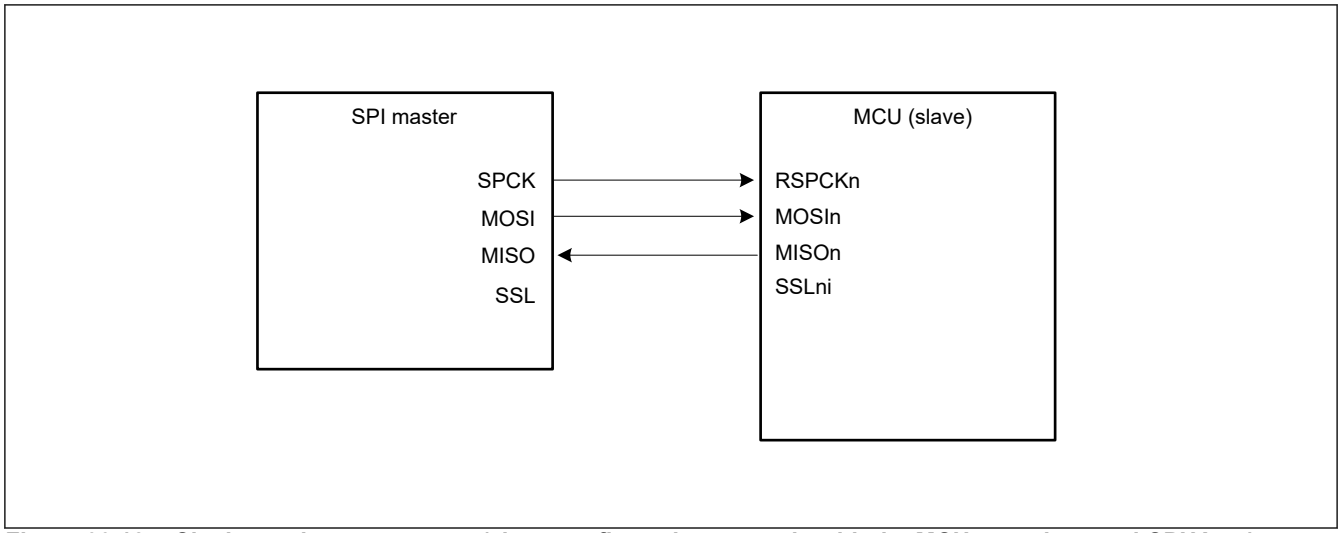



**Figure 30.12 Clock synchronous master/slave configuration example with the MCU as a slave and CPHA = 1**

## 30.3.4 Data Formats

The data format of the SPI depends on the settings in SPI Command Register m (SPCMDm) and the parity enable bit in SPI Control Register 2 (SPCR2.SPPE). Regardless of whether the MSB or LSB is first, the SPI treats the range from the LSB bit in the SPI Data Register (SPDR/SPDR_HA) to the bit associated with the selected data length, as transfer data.

This section shows the format of one frame of data before or after transfer.

#### Data format with parity disabled

When parity is disabled, transmission or reception of data proceeds with the length in bits selected in the SPI data length setting in SPI Command Register m (SPCMDm.SPB[3:0]).

**Data format with parity enabled**

When parity is enabled, transmission or reception of data proceeds with the length in bits selected in the SPI data length setting in SPI Command Register m (SPCMDm.SPB[3:0]). In this case, however, the last bit is a parity bit.

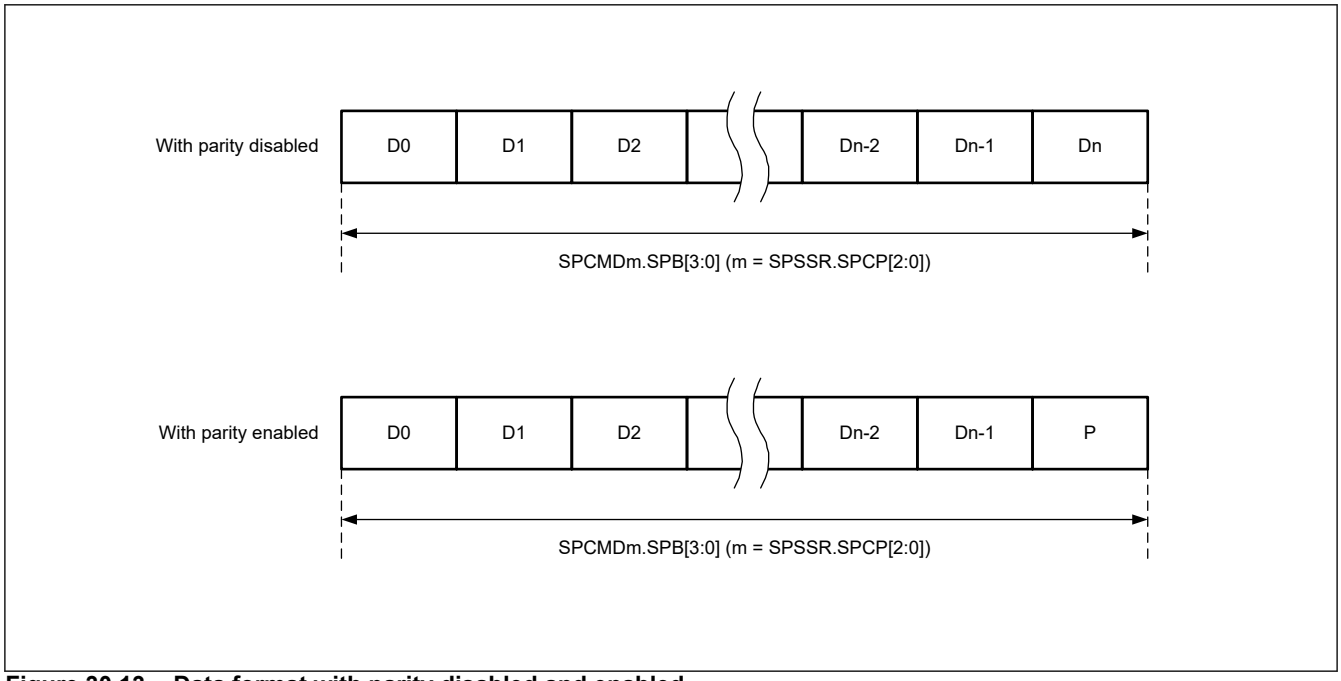



**Figure 30.13 Data format with parity disabled and enabled**

### 30.3.4.1 Operation when parity is disabled (SPCR2.SPPE = 0)

When parity is disabled, data for transmission is copied to the shift register with no pre-processing. This section describes the connection between the SPI Data Register (SPDR/SPDR_HA) and the shift register in terms of the combination of MSB- or LSB-first order and data length.

#### (1) MSB-first transfer with 32-bit data

Figure 30.14 shows the operation of the SPI Data Register (SPDR) and the shift register in a transfer with parity disabled, a SPI data length of 32 bits, and MSB-first selected.

In transmission, bits T31 to T00 from the current stage of the transmit buffer are copied to the shift register. Data for transmission is shifted out from the shift register from T31 to T30, and continuing to T00.

In reception, received data is shifted in bit-by-bit through bit[0] of the shift register. When the R31 to R00 bits are collected after input of the required number of RSPCK cycles, the value in the shift register is copied to the receive buffer.

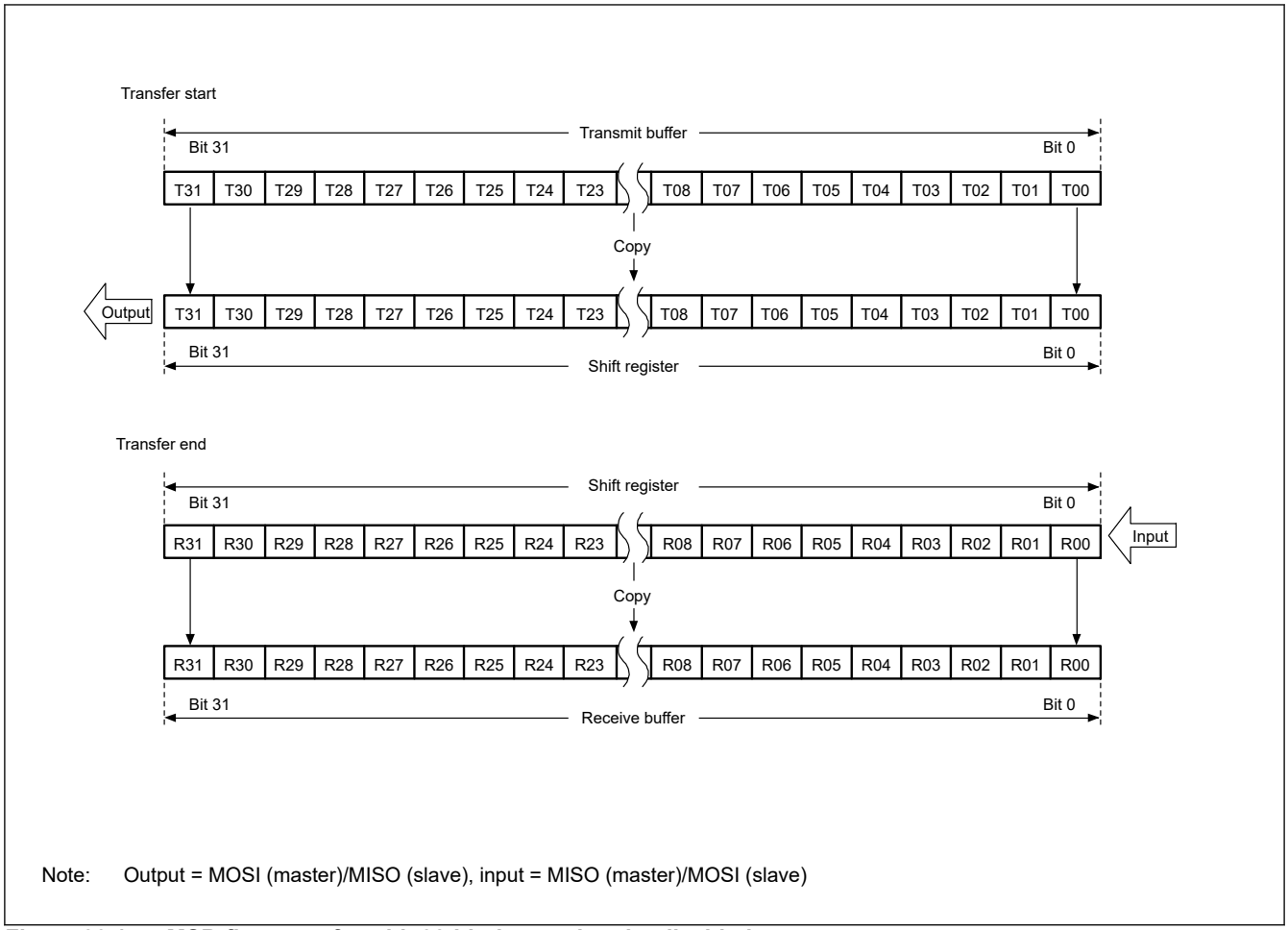


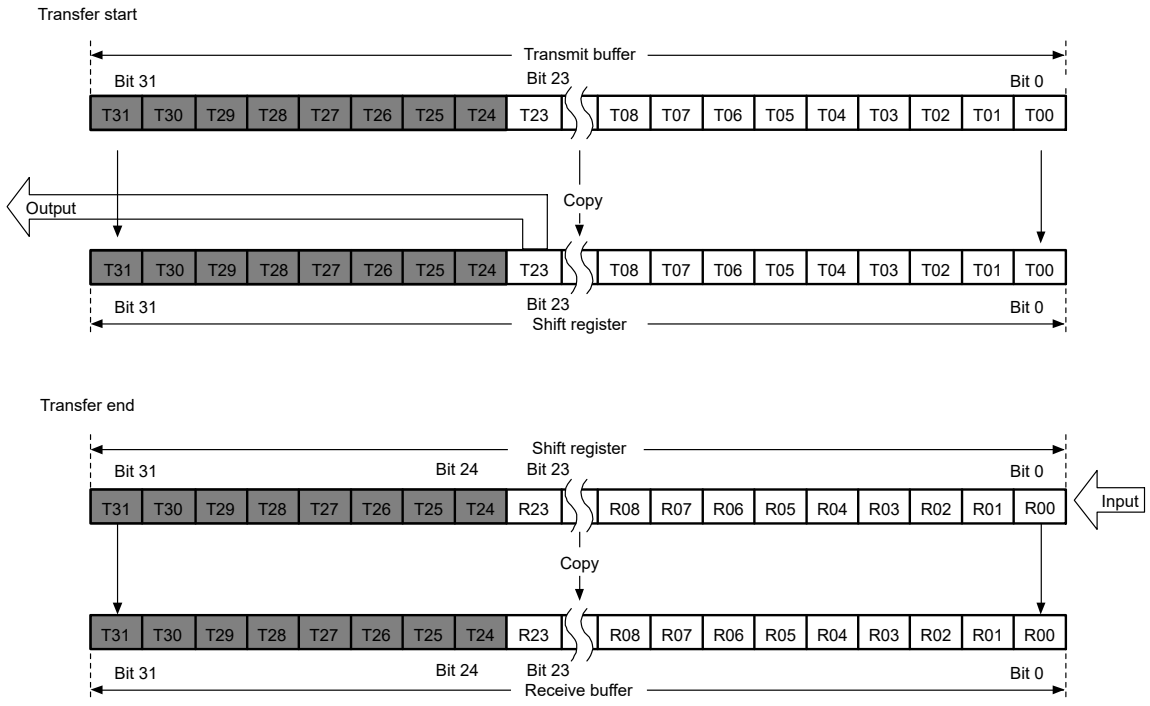

**Figure 30.14 MSB-first transfer with 32-bit data and parity disabled**

## (2) MSB-first transfer with 24-bit data

Figure 30.15 shows the operation of the SPI Data Register (SPDR) and the shift register in a transfer with parity disabled, an SPI data length of 24 bits for an example that is not 32 bits, and MSB-first selected.

In transmission, the lower 24 bits (T23 to T00) from the current stage of the transmit buffer are copied to the shift register. Data for transmission is shifted out from the shift register from T23 to T22, and continuing to T00.

In reception, received data is shifted in bit-by-bit through bit[0] of the shift register. When the R23 to R00 bits are collected after input of the required number of RSPCK cycles, the value in the shift register is copied to the receive buffer. The upper 8 bits of the transmit buffer are stored in the upper 8 bits of the receive buffer in case of transmit-receive operation. Writing 0 to bits T31 to T24 during transmission leads to 0 being inserted in the upper 8 bits of the receive buffer. On the other hand, the upper 8-bits of the receive buffer is written 0 in case of receive only operation.




Note: Output = MOSI (master)/MISO (slave), input = MISO (master)/MOSI (slave)

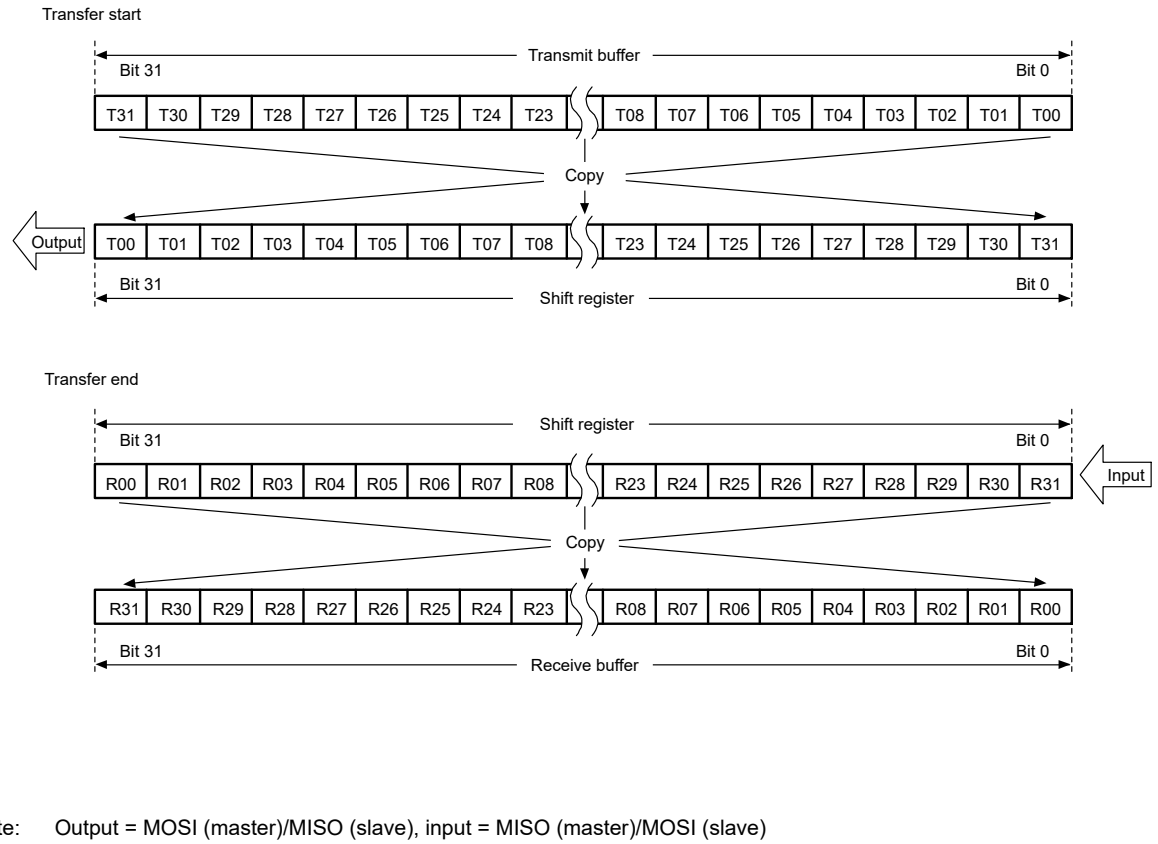
**Figure 30.15 MSB-first transfer with 24-bit data and parity disabled**

### (3) LSB-first transfer with 32-bit data

Figure 30.16 shows the operation of the SPI Data Register (SPDR) and the shift register in a transfer with parity disabled, an SPI data length of 32 bits, and LSB-first selected.

In transmission, bits T31 to T00 from the current stage of the transmit buffer are reordered bit-by-bit to obtain the order T00 to T31 for copying to the shift register. Data for transmission is shifted out from the shift register in order from T00 to T01, and continuing to T31.

In reception, received data is shifted in bit-by-bit through bit[0] of the shift register. When the R00 to R31 bits are collected after input of the required number of RSPCK cycles, the value in the shift register is copied to the receive buffer.




Note: Output = MOSI (master)/MISO (slave), input = MISO (master)/MOSI (slave)

**Figure 30.16 LSB-first transfer with 32-bit data and parity disabled**

## (4) LSB-first transfer with 24-bit data

Figure 30.17 shows the operation of the SPI Data Register (SPDR) and the shift register in transfers with parity disabled, an SPI data length of 24 bits for an example that is not 32, and LSB-first selected.

In transmission, the lower 24 bits (T23 to T00) from the current stage of the transmit buffer are reordered bit-by-bit to obtain the order T00 to T23 for copying to the shift register. Data for transmission is shifted out from the shift register from T00 to T01, and continuing to T23.

In reception, received data is shifted in bit-by-bit through bit[8] of the shift register. When the R00 to R23 bits are collected after input of the required number of RSPCK cycles, the value in the shift register is copied to the receive buffer.

The upper 8 bits of the transmit buffer are stored in the upper 8 bits of the receive buffer in case of transmit-receive operation. Writing 0 to T31 to T24 during transmission leads to 0 being inserted in the upper 8 bits of the receive buffer. On the other hand, the upper 8-bits of the receive buffer is written 0 in case of receive only operation.

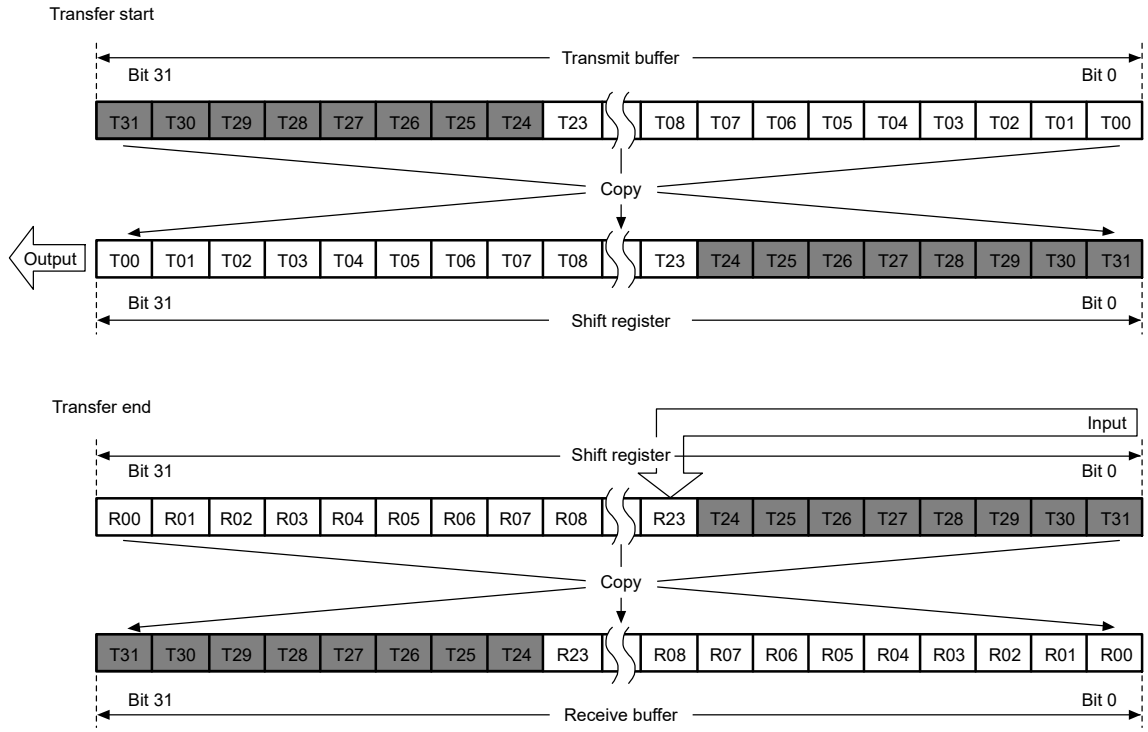



**Figure 30.17 LSB-first transfer with 24-bit data and parity disabled**

### 30.3.4.2 Operation when parity is enabled (SPCR2.SPPE = 1)

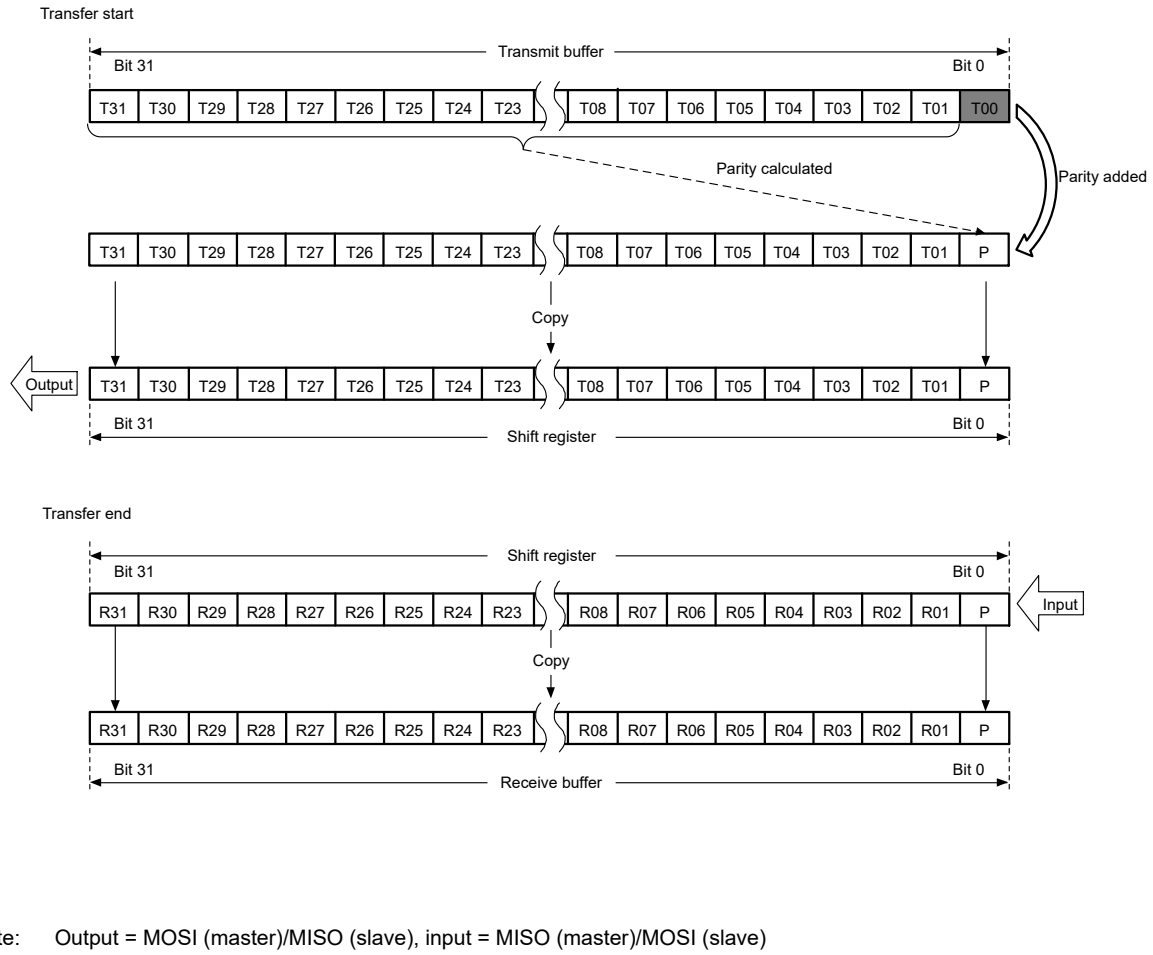
When parity is enabled, the lowest-order bit of the data for transmission becomes a parity bit. Hardware calculates the value of the parity bit.

#### (1) MSB-first transfer with 32-bit data

Figure 30.18 shows the operation of the SPI Data Register (SPDR) and the shift register in a transfer with parity enabled, an SPI data length of 32 bits, and MSB-first selected.

In transmission, the value of the parity bit (P) is calculated from bits T31 to T01. This replaces the final bit, T00, and the whole value is copied to the shift register. Data is transmitted in the order T31, T30, …, T01, and P.

In reception, received data is shifted in bit-by-bit through bit[0] of the shift register. When the R31 to P bits are collected after input of the required number of RSPCK cycles, the value in the shift register is copied to the receive buffer. On copying of data to the shift register, the data from R31 to P is checked for parity.




Note: Output = MOSI (master)/MISO (slave), input = MISO (master)/MOSI (slave)

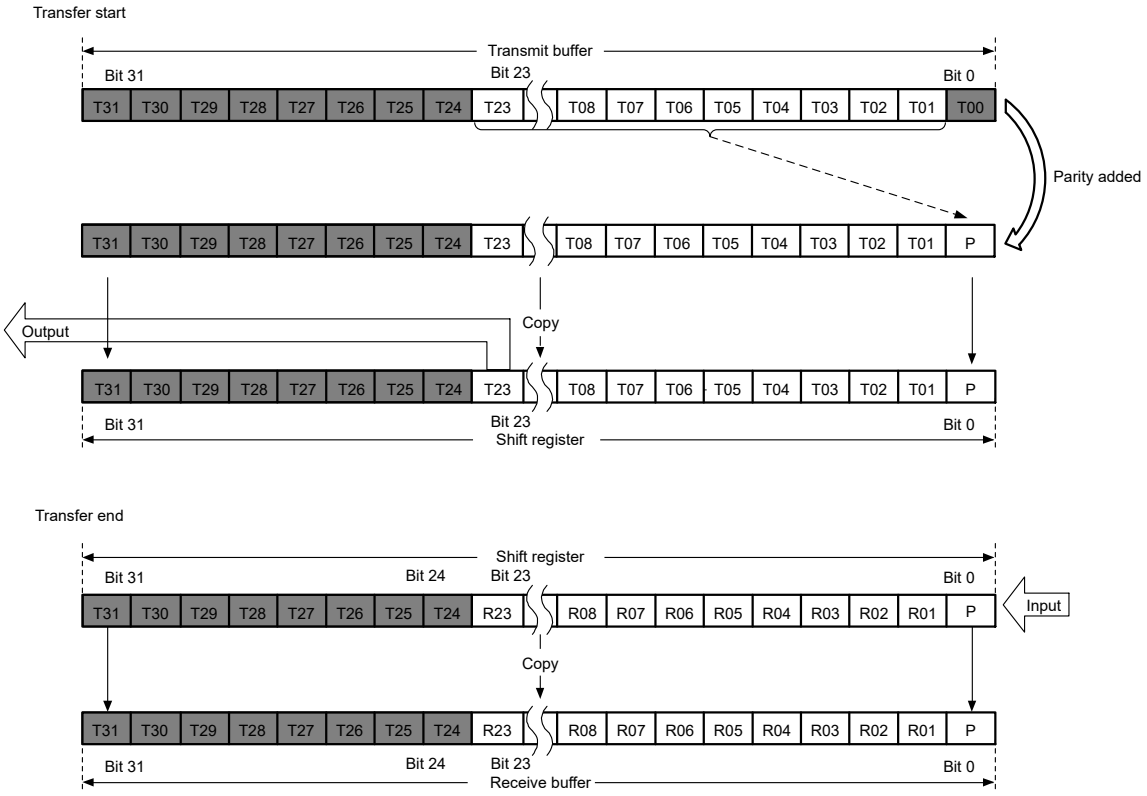
**Figure 30.18 MSB-first transfer with 32-bit data and parity enabled**

## (2) MSB-first transfer with 24-bit data

Figure 30.19 shows the operation of the SPI Data Register (SPDR) and the shift register in a transfer with parity enabled, a SPI data length of 24 bits, and MSB-first selected.

In transmission, the value of the parity bit (P) is calculated from bits T23 to T01. This replaces the final bit, T00, and the whole value is copied to the shift register. Data is transmitted in the order T23, T22, …, T01, and P.

In reception, received data is shifted in bit-by-bit through bit[0] of the shift register. When the R23 to P bits are collected after input of the required number of RSPCK cycles, the value in the shift register is copied to the receive buffer. On copying of data to the shift register, the data from R23 to P is checked for parity. The upper 8 bits of the transmit buffer is stored in the upper 8 bits of the receive buffer in case of transmit-receive operation. Writing 0 to T31 to T24 during transmission leads to 0 being inserted in the upper 8 bits of the receive buffer. On the other hand, the upper 8-bits of the receive buffer is written 0 in case of receive only operation.




Note: Output = MOSI (master)/MISO (slave), input = MISO (master)/MOSI (slave)

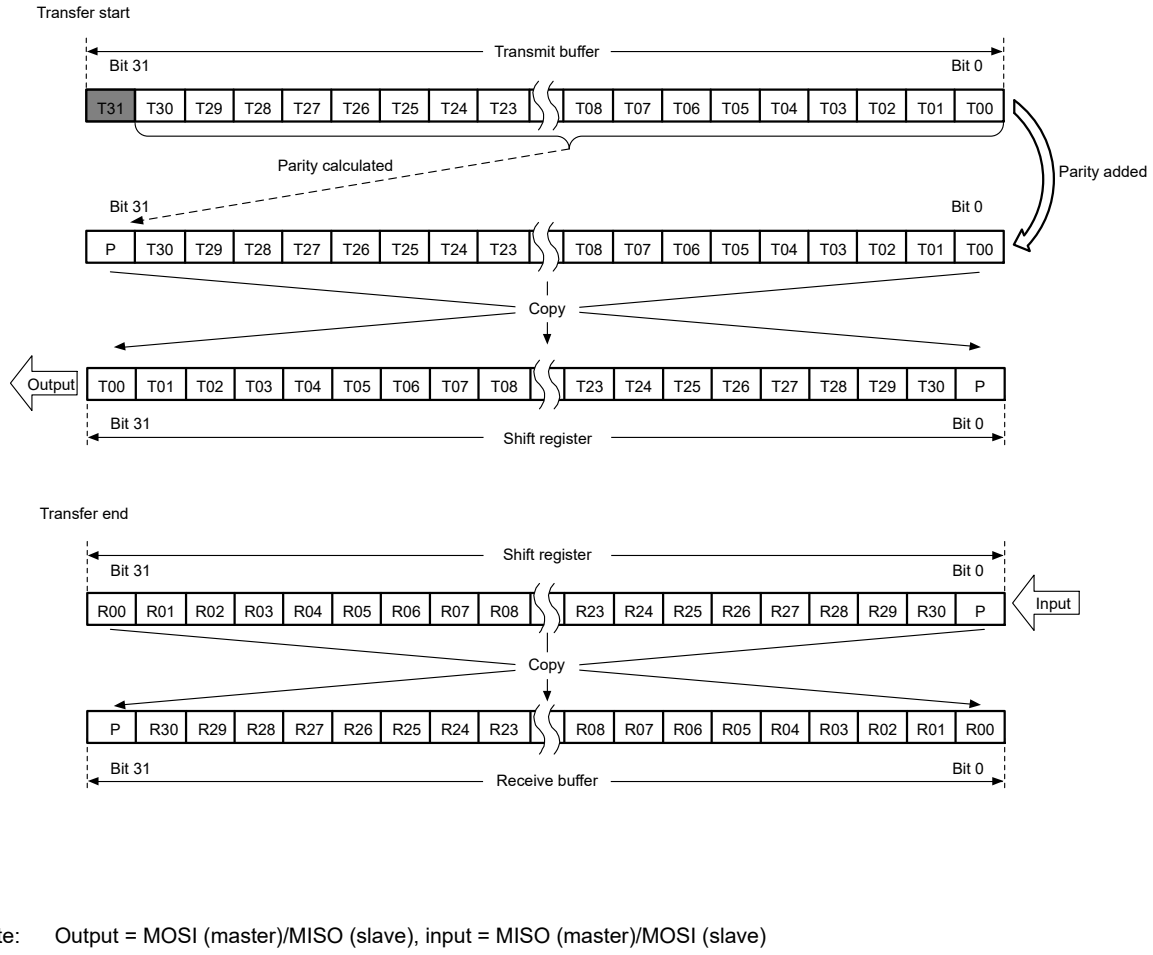
**Figure 30.19 MSB-first transfer with 24-bit data and parity enabled**

## (3) LSB-first transfer with 32-bit data

Figure 30.20 shows the operation of the SPI Data Register (SPDR) and the shift register in a transfer with parity enabled, an SPI data length of 32 bits, and LSB-first selected.

In transmission, the value of the parity bit (P) is calculated from bits T30 to T00. This replaces the final bit, T31, and the whole value is copied to the shift register. Data is transmitted in the order T00, T01, …, T30, and P.

In reception, received data is shifted in bit-by-bit through bit[0] of the shift register. When the R00 to P bits are collected after input of the required number of RSPCK cycles, the value in the shift register is copied to the receive buffer. On copying of data to the shift register, the data from R00 to P is checked for parity.




Note: Output = MOSI (master)/MISO (slave), input = MISO (master)/MOSI (slave)

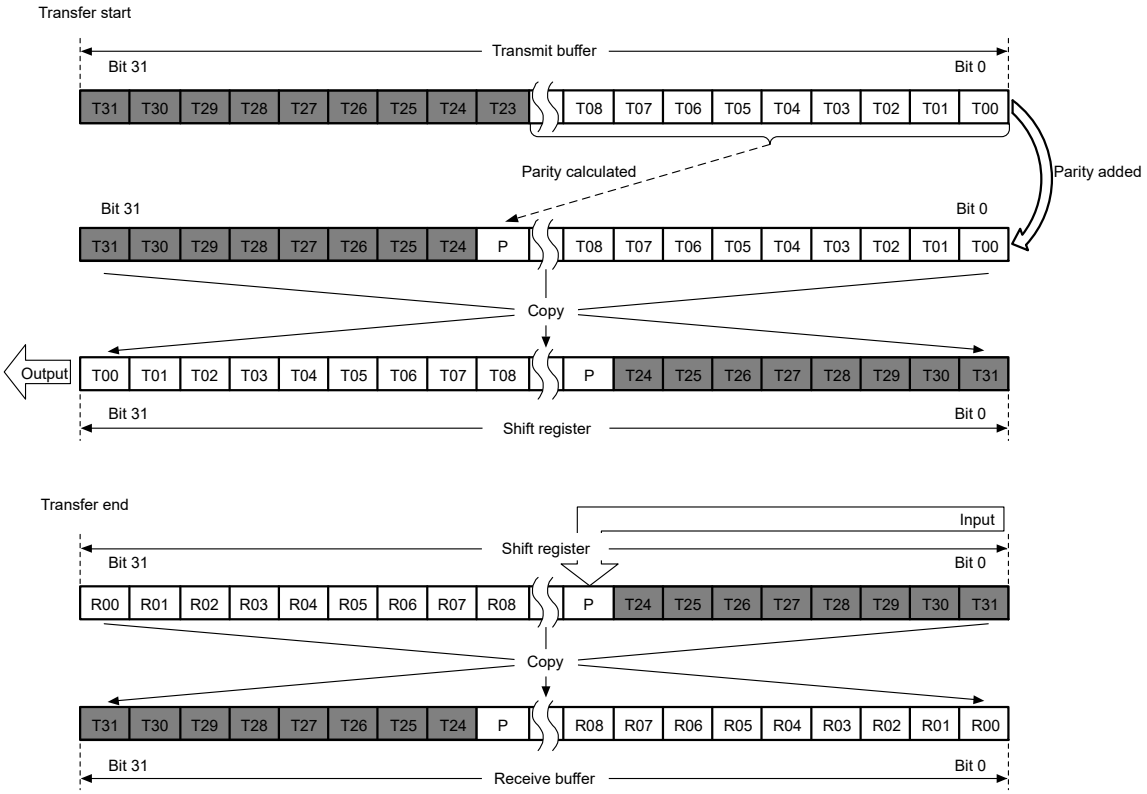
**Figure 30.20 LSB-first transfer with 32-bit data and parity enabled**

## (4) LSB-first transfer with 24-bit data

Figure 30.21 shows the operation of the SPI Data Register (SPDR) and the shift register in a transfer with parity enabled, a SPI data length of 24 bits, and LSB-first selected.

In transmission, the value of the parity bit (P) is calculated from bits T22 to T00. This replaces the final bit, T23, and the whole value is copied to the shift register. Data is transmitted in the order T00, T01, …, T22, and P.

In reception, received data is shifted in bit-by-bit through bit[8] of the shift register. When the R00 to P bits are collected after input of the required number of RSPCK cycles, the value in the shift register is copied to the receive buffer. On copying of data to the shift register, the data from R00 to P is checked for parity. The upper 8 bits of the transmit buffer is stored in the upper 8 bits of the receive buffer in case of transmit-receive operation. Writing 0 to T31 to T24 during transmission leads to 0 being inserted in the upper 8 bits of the receive buffer. On the other hand, the upper 8-bits of the receive buffer is written 0 in case of receive only operation.




Note: Output = MOSI (master)/MISO (slave), input = MISO (master)/MOSI (slave)

**Figure 30.21 LSB-first transfer with 24-bit data and parity enabled**

### 30.3.4.3 Byte Swap Transmission

#### (1) MSB-first transfer. (When the byte swap is disabled.)

Data (Byte0 [T31 to T24] to Byte3 [T07 to T00]) in the transmit buffer are copied to the shift register.

Bit values in the shift register are shifted and transmitted in the order of T31 → T30 → … → T00 as transmit data.

#### (2) MSB-first transfer. (When the byte swap is enabled.)

Byte values of the transmit buffer (Byte0 [T31 to T24] to Byte3 [T07 to T00]) are reversed in byte units and are copied to the shift register in the order of Byte3 [T07 to T00] to Byte0 [T31 to T24].

Bit values in the shift register are shifted and transmitted in the order of T07 → T06 → … → T00 → T15 → T14 → … → T08 → T23 → T22 → … → T16 → T31 → T30 → … → T24 as transmit data.

#### (3) LSB-first transfer. (When the byte swap is disabled.)

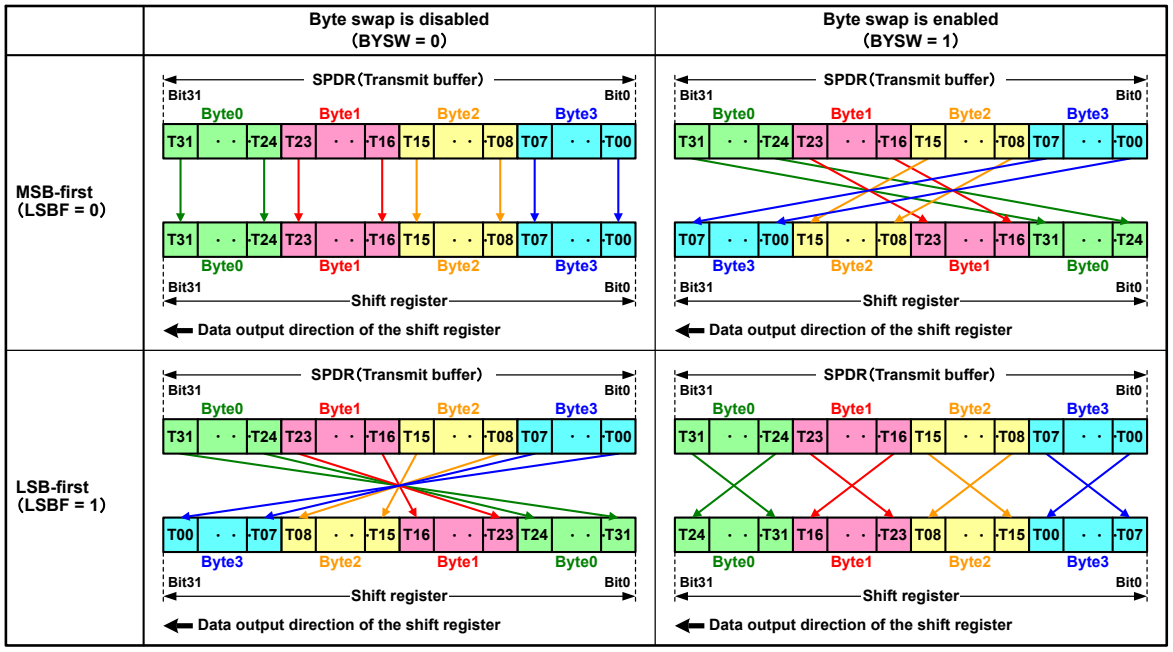
Bit values of the transmit buffer (Byte0 [T31 to T24] to Byte3 [T07 to T00]) are reversed in bit units and are copied to the shift register in the order of Byte3 [T00 to T07] to Byte0 [T24 to T31].

Bit values in the shift register are shifted and transmitted in the order of T00 → T01 → … → T31 as transmit data.

#### (4) LSB-first transfer. (When the byte swap is enabled.)

Bit values of each byte of the transmit buffer (Byte0 [T31 to T24] to Byte3 [T07 to T00]) are reversed in bit units and are copied to the shift register in the order of Byte0 [T24 to T31] to Byte3 [T00 to T07].

Bit values in the shift register are shifted and transmitted in the order of T24 → T25 → … → T31 → T16 → T17 → … → T23 → T08 → T09 → … → T15 → T00 → T01 → … → T07 as transmit data.




Note: 1. When using the byte swap, set 16 bits or 32 bits to the data length (SPCMD.SPB[3:0] setting). If setting the other length (8 to 15, 20, 24 bits), the behavior is not guaranteed.
2. When the byte swap is valid, set the parity function as invalid (SPCR2.SPPE bit = 0). If setting the parity function as valid (SPPE bit = 1), the behavior is not guaranteed.
3. Set SPDCR2.BYSW bit, when SPCR.SPE bit is 0. If rewriting BYSW bit, when SPE bit is 1, the behavior after it is not guaranteed.

**Figure 30.22 Byte swap with MSB/LSB transfer**

### 30.3.4.4 Byte Swap Reception

#### (1) MSB-first transfer. (When the byte swap is disabled.)

The first received data (R31) is stored in bit 0 of the shift register, and received data is shifted in the order of R31 → R30 → … → R00.

When necessary RSPCK cycles are input and data is stored from Byte0 [R31 to R24] to Byte3 [R07 to R00], the shift register value is copied to the receive buffer.

#### (2) MSB-first transfer. (When the byte swap is enabled.)

The first received data (R07) is stored in bit 0 of the shift register, and received data is shifted in the order of R07 → R06 → … → R00 → R15 → R14 → … → R08 → R23 → R22 → … → R16 → R31 → R30 → … → R24.

When necessary RSPCK cycles are input and data is stored from Byte3 [R07 to R00] to Byte0 [R31 to R24], byte values in the shift register are reversed in byte units and are copied to the receive buffer in the order of Byte0 [R31 to R24] to Byte3 [R07 to R00].

#### (3) LSB-first transfer. (When the byte swap is disabled.)

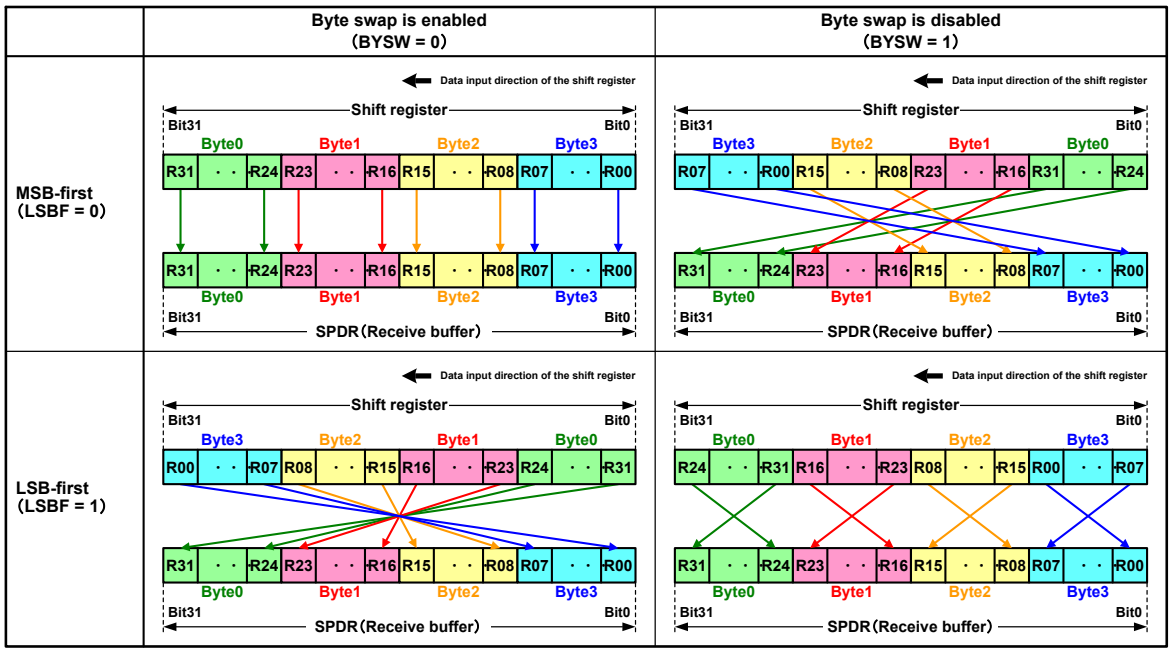
The first received data (R00) is stored in bit 0 of the shift register, and received data is shifted in the order of R00 → R01 → … → R31.

When necessary RSPCK cycles are input and data is stored from Byte3 [R00 to R07] to Byte0 [R24 to R31], bit values in the shift register are reversed in bit units and are copied to the receive buffer in the order of Byte0 [R31 to R24] to Byte3 [R07 to R00].

### (4) LSB-first transfer. (When the byte swap is enabled.)

The first received data (R24) is stored in bit 0 of the shift register, and received data is shifted in the order of R24 → R25 → … → R31 → R16 → R17 → … → R23 → R08 → R09 → … → R15 → R00 → R01 → … → R07.

When necessary RSPCK cycles are input and data is stored from Byte0 [R24 to R31] to Byte3 [R00 to R07], bit values of each byte in the shift register are reversed in bit units and are copied to the receive buffer in the order of Byte0 [R31 to R24] to Byte3 [R07 to R00].




Note: 1. When using the byte swap, set 16 bits or 32 bits to the data length (SPCMD.SPB[3:0] setting). If setting the other length (8 to 15, 20, 24 bits), the behavior is not guaranteed.
2. When the byte swap is valid, set the parity function as invalid (SPCR2.SPPE bit = 0). If setting the parity function as valid (SPPE bit = 1), the behavior is not guaranteed.
3. Set SPDCR2.BYSW bit, when SPCR.SPE bit is 0. If rewriting BYSW bit, when SPE bit is 1, the behavior after it is not guaranteed.

**Figure 30.23 Byte swap with MSB/LSB transfer**

## 30.3.5 Transfer Formats

### 30.3.5.1 When CPHA = 0

Figure 30.24 shows an example transfer format for the serial transfer of 8-bit data when the SPCMDm.CPHA bit is 0. Do not perform clock synchronous operation (SPCR.SPMS = 1) when the SPI operates in slave mode (SPCR.MSTR = 0) and the CPHA bit is 0. In Figure 30.24, RSPCKn (CPOL = 0) indicates the RSPCKn signal waveform when the SPCMDm.CPOL bit is 0, and RSPCKn (CPOL = 1) indicates the RSPCKn signal waveform when the CPOL bit is 1. The sampling timing represents the timing at which the SPI fetches serial transfer data into the shift register. The I/O directions of the signals depend on the SPI settings. For details, see section 30.3.2. Controlling the SPI Pins.

When the SPCMDm.CPHA bit is 0, the driving of valid data to the MOSIn and MISOn signals begins at an SSLni signal assertion. The first RSPCKn signal change that occurs after the SSLni signal assertion becomes the first transfer data fetch. After this, data is sampled every 1 RSPCKn cycle. The change timing for MOSIn and MISOn signals is 1/2 RSPCK cycles after the transfer data fetch timing. The CPOL bit setting does not affect the RSPCK signal operation timing as it only affects the signal polarity.

t1 denotes the RSPCK delay, the period from an SSLni signal assertion to RSPCKn oscillation. t2 denotes the SSL negation delay, the period from the termination of RSPCKn oscillation to an SSLni signal negation. t3 denotes the next-access delay,

the period in which SSLni signal assertion is suppressed for the next transfer after the end of serial transfer. t1, t2, and t3 are controlled by a master device running on the SPI system. For a description of t1, t2, and t3 when the SPI is in master mode, see section 30.3.11.1. Master mode operation.

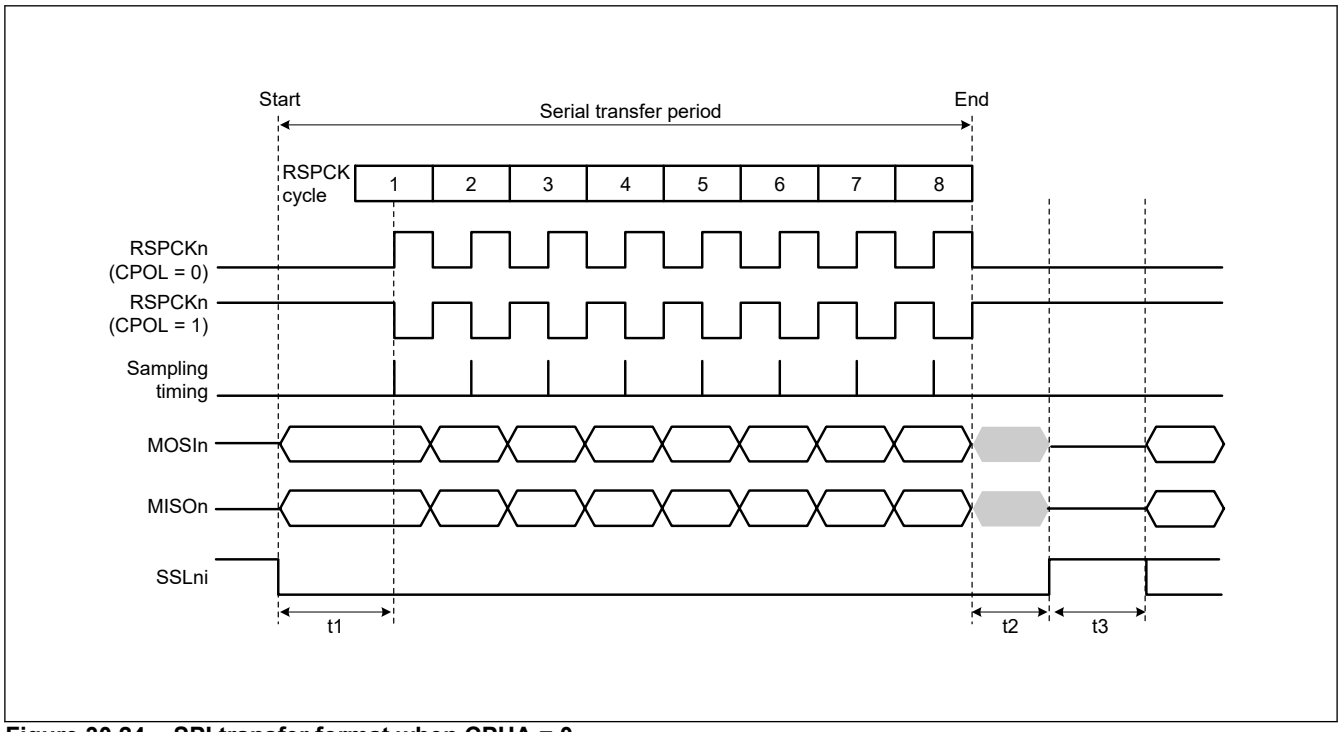



**Figure 30.24 SPI transfer format when CPHA = 0**

### 30.3.5.2 When CPHA = 1

Figure 30.25 shows an example transfer format for the serial transfer of 8-bit data when the SPCMDm.CPHA bit is 1. However, when the SPCR.SPMS bit is 1, the SSLni signals are not used, and only the three signals RSPCKn, MOSIn, and MISOn handle communications. In Figure 30.25, RSPCK (CPOL = 0) indicates the RSPCKn signal waveform when the SPCMDm.CPOL bit is 0 and RSPCK (CPOL = 1) indicates the RSPCKn signal waveform when the CPOL bit is 1. The sampling timing represents the timing at which the SPI fetches serial transfer data into the shift register. The I/O directions of the signals depend on the SPI mode (master or slave mode). For details, see section 30.3.2. Controlling the SPI Pins.

When the SPCMDm.CPHA bit is 1, the driving of invalid data to the MISOn signal begins at an SSLni signal assertion. The output of valid data to the MOSIn and MISOn signals begins at the first RSPCKn signal change that occurs after the SSLni signal assertion. After this, data is updated every 1 RSPCK cycle. The transfer data fetch timing is 1/2 RSPCK cycles after the data update timing. The SPCMDm.CPOL bit setting does not affect the RSPCKn signal operation timing. It only affects the signal polarity.

t1, t2, and t3 are the same as those when CPHA = 0. For a description of t1, t2, and t3 when the SPI of the MCU is in master mode, see section 30.3.11.1. Master mode operation.

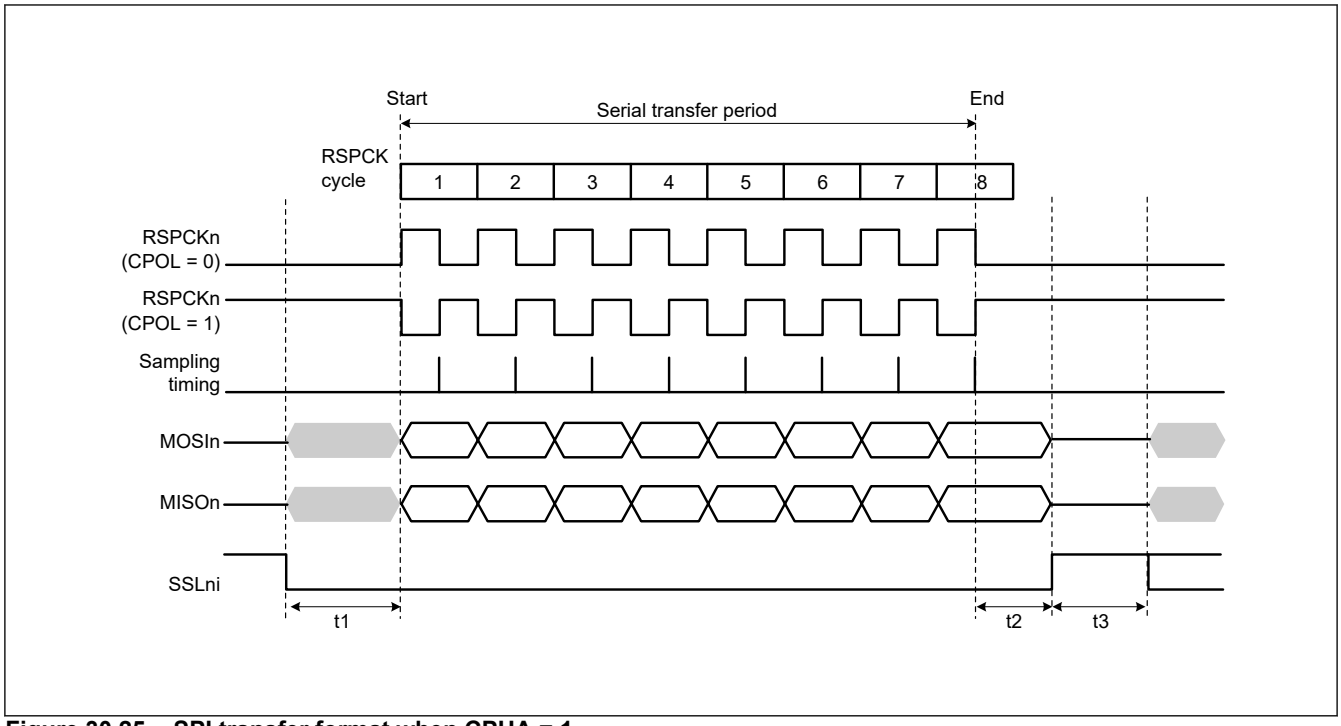



**Figure 30.25 SPI transfer format when CPHA = 1**

## 30.3.6 Data Transfer Modes

Full-duplex synchronous serial communications or transmit operations can only be selected in the Communications Operating Mode Select bit (SPCR.TXMD) when the extended communication mode select bit (ETXMD) in the SPI control register 3 (SPCR3) is 0. The SPI operation is receive only in slave mode (SPCR.MSTR = 0) when the SPCR3.ETXMD bit is 1, because SPCR.TXMD bit does not affect the SPI operation. The register accesses shown in Figure 30.26, Figure 30.27, and Figure 30.28 indicate the condition of access to the SPDR/SPDR_HA register, where W denotes a write cycle.

### 30.3.6.1 Full-duplex synchronous serial communications (SPCR3.ETXMD = 0, SPCR.TXMD = 0)

Figure 30.26 shows an example of operation when the extended communication mode select bit (ETXMD) in the SPI control register 3 (SPCR3) is set to 0 and the Communications Operating Mode Select bit (SPCR.TXMD) is set to 0. In this example, the SPI performs an 8-bit serial transfer when SPDCR.SPFC[1:0] bits are 00b, the SPCMDm.CPHA bit is 1, and the SPCMDm.CPOL bit is 0. The numbers given for RSPCKn in the waveform represent the number of RSPCK cycles, such as the number of transferred bits.

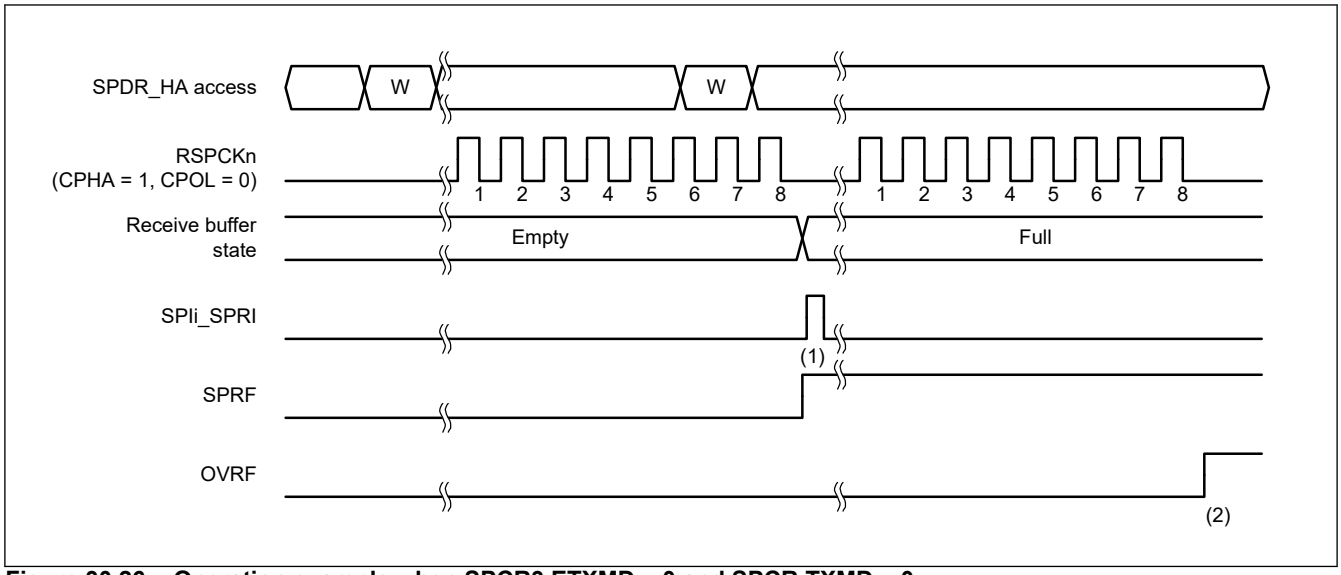



**Figure 30.26 Operation example when SPCR3.ETXMD = 0 and SPCR.TXMD = 0**

The operation of the flags at timings (1) and (2) in Figure 30.26 is as follows:

1. When a serial transfer ends with the receive buffer of SPDR_HA empty, the SPI generates a receive buffer full interrupt request (SPIi_SPRI), the SPI sets the SPSR.SPRF flag to 1, and copies the received data in the shift register to the receive buffer.
2. When a serial transfer ends with the receive buffer of SPDR_HA holding data that was received in the previous serial transfer, the SPI sets the SPSR.OVRF flag to 1, and discards the received data in the shift register. For details about the operation of the SPSR.OVRF flag, see section 30.3.9.1. Overrun errors.

### 30.3.6.2 Transmit-Only Serial Communications (SPCR3.ETXMD = 0, SPCR.TXMD = 1)

Figure 30.27 shows an example of operation when the extended communication mode select bit (ETXMD) in the SPI control register 3 (SPCR3) is set to 0 and the Communications Operating Mode Select bit (SPCR.TXMD) is set to 1. In this example, the SPI performs an 8-bit serial transfer when SPDCR.SPFC[1:0] bits are 00b, the SPCMDm.CPHA bit is 1, and the SPCMDm.CPOL bit is 0. The numbers given for RSPCKn in the waveform represent the number of RSPCK cycles, such as the number of transferred bits.

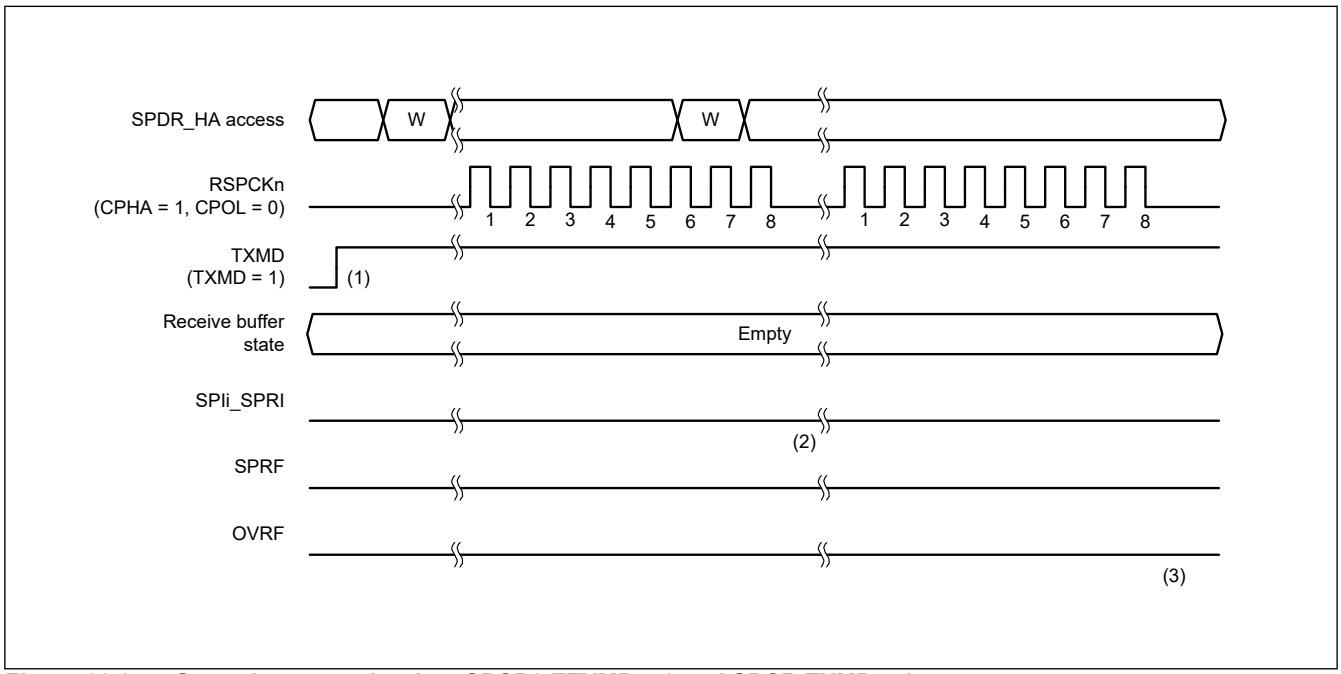



**Figure 30.27 Operation example when SPCR3.ETXMD = 0 and SPCR.TXMD = 1**

The operation of the flags at timings (1) to (3) in Figure 30.27 is as follows:

1. Make sure there is no data left in the receive buffer (the SPSR.SPRF flag is 0) and the SPSR.OVRF flag is 0 before entering transmit-only mode (SPCR3.ETXMD = 0, SPCR.TXMD = 1).
2. When a serial transfer ends with the receive buffer of SPDR_HA empty, if the transmit-only mode is selected (SPCR3.ETXMD = 0, SPCR.TXMD = 1), the SPSR.SPRF flag retains the value of 0, and the SPI does not copy the data in the shift register to the receive buffer.
3. Because the receive buffer of SPDR_HA does not hold data that was received in the previous serial transfer, even when a serial transfer ends, the SPSR.OVRF flag retains the value of 0, and the data in the shift register is not copied to the receive buffer.

In transmit-only mode (SPCR3.ETXMD = 0, SPCR.TXMD = 1), the SPI transmits data but does not receive data. Therefore, the SPSR.SPRF and SPSR.OVRF flags remain 0 at timings (1) to (3).

### 30.3.6.3 Receive-Only Serial Communication (MSTR = 0, ETXMD = 1)

Figure 30.28 shows an example of operation when the SPI master/slave mode select bit (MSTR) in the SPI control register (SPCR) is set to 0 and the extended communication mode select bit (ETXMD) in the SPI control register 3 (SPCR3) is set to 1. In the example in Figure 30.28, the SPI performs 8-bit data serial transfer with the settings of SPFC[1:0] in the SPI data control register (SPDCR) = 00b, CPHA in the SPI command register (SPCMD) = 1, and CPOL in SPCMD = 0. Numbers under the RSPCK waveform show the number of RSPCK cycles (number of transfer bits).

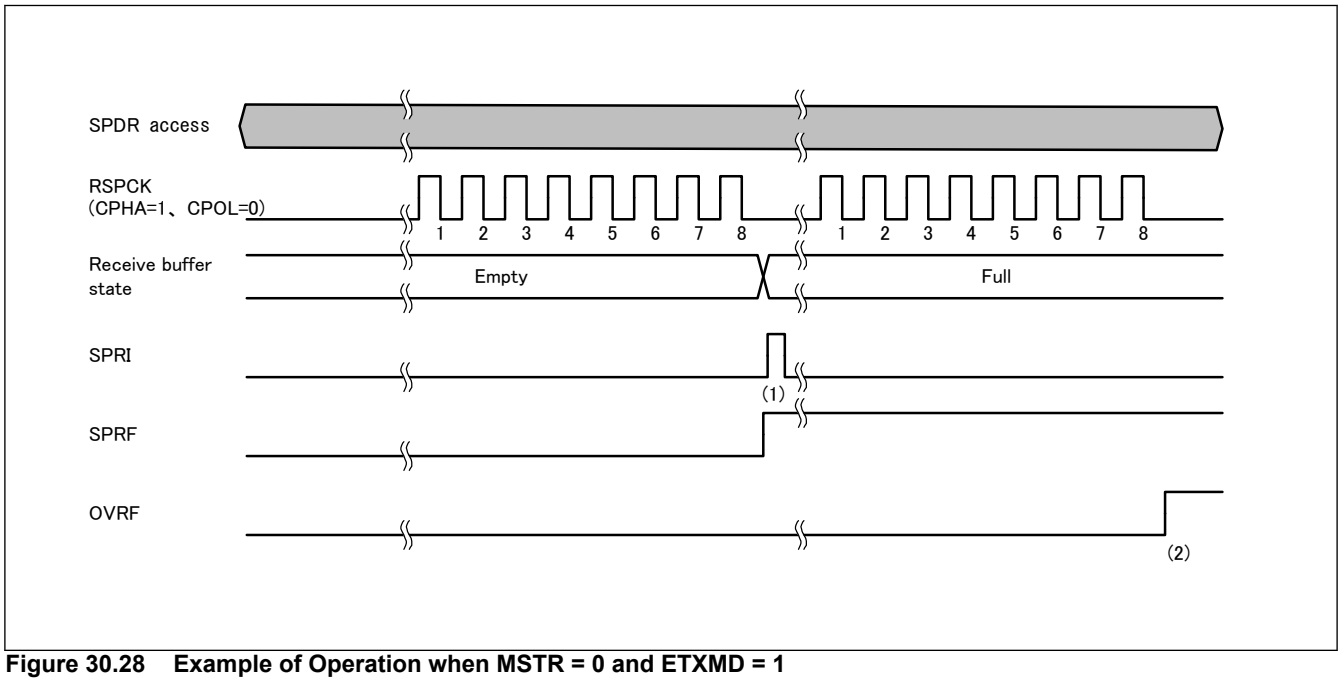



**Figure 30.28 Example of Operation when MSTR = 0 and ETXMD = 1**

The following describes operation of flags at timings (1) and (2) in the figure above.

(1) When serial transfer ends while the SPDR's receive buffer is empty, the SPI generates a receive buffer full interrupt request SPRI (setting the SPSR.SPRF flag to 1) and copies the received data in the shift register to the receive buffer

(2) When serial transfer ends while previously received data is remaining in the SPDR's receive buffer, the SPI sets the OVRF flag in the SPI status register (SPSR) to 1 and discards the received data in the shift register.

## 30.3.7 Transmit Buffer Empty and Receive Buffer Full Interrupts

Figure 30.29 and Figure 30.30 show examples of operation of the transmit buffer empty interrupt (SPIi_SPTI) and the receive buffer full interrupt (SPIi_SPRI). The SPDR_HA register accesses shown in these figures indicate the conditions of access to the register, where W denotes a write cycle and R a read cycle. In Figure 30.29, the SPI performs an 8-bit serial transfer when SPCR.TXMD bit is 0, the SPDCR.SPFC[1:0] bits are 00b, the SPCMDm.CPHA bit is 0, and the SPCMDm.CPOL bit is 0. In Figure 30.30, the SPI performs an 8-bit serial transfer when SPCR.TXMD bit is 0, the

SPDCR.SPFC[1:0] bits are 00b, the SPCMDm.CPHA bit is 1, and the SPCMDm.CPOL bit is 0. The numbers given for RSPCKn in the waveform represent the number of RSPCK cycles, such as the number of transferred bits.

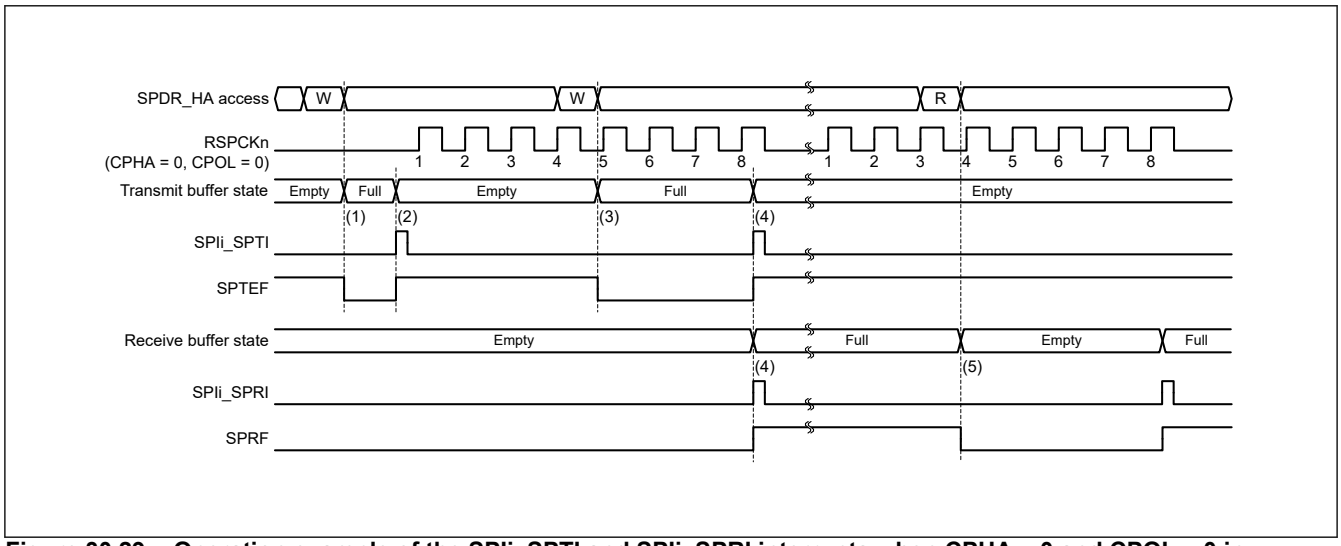



**Figure 30.29 Operation example of the SPIi_SPTI and SPIi_SPRI interrupts when CPHA = 0 and CPOL = 0 in master mode**

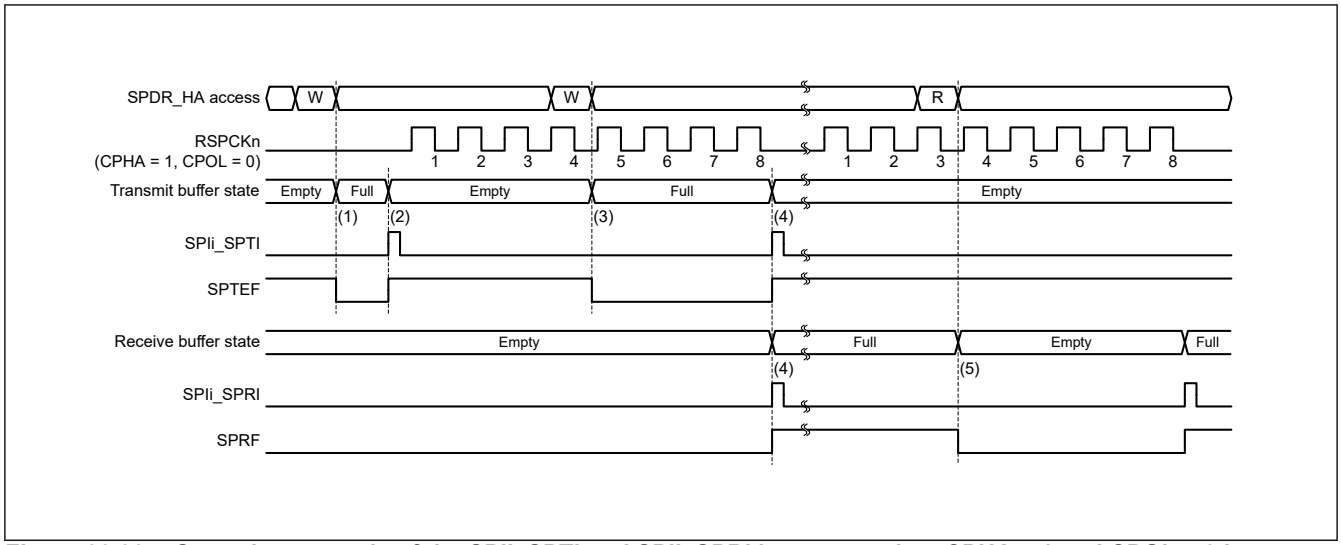



**Figure 30.30 Operation example of the SPIi_SPTI and SPIi_SPRI interrupts when CPHA = 1 and CPOL = 0 in master mode**

The operation of the SPI at timings (1) to (5) in Figure 30.29 and Figure 30.30 is as follows:

1. When transmit data is written to SPDR_HA with the transmit buffer of SPDR_HA is empty and data for the next transfer is not set, the SPI writes data to the transmit buffer and clears the SPSR.SPTEF flag to 0.
2. If the shift register is empty, the SPI copies the data in the transmit buffer to the shift register, generates a transmit buffer empty interrupt request (SPIi_SPTI), and sets the SPSR.SPTEF flag to 1. How a serial transfer is started depends on the SPI mode. For details, see section 30.3. Operation, and section 30.3.12. Clock Synchronous Operation.
3. When transmit data is written to SPDR_HA either by the transmit buffer empty interrupt routine, or by the processing of the transmit buffer empty using the SPTEF flag, the SPI writes data to the transmit buffer and clears the SPTEF flag to 0. Because the data being transferred serially is stored in the shift register, the SPI does not copy the data in the transmit buffer to the shift register.
4. When the serial transfer ends with the receive buffer of SPDR_HA empty, the SPI copies the receive data in the shift register to the receive buffer, generates a receive buffer full interrupt request (SPIi_SPRI), and sets the SPRF flag to 1. Because the shift register becomes empty on completion of the serial transfer, if the transmit buffer is full before the serial transfer ended, the SPI sets the SPTEF flag to 1 and copies data in the transmit buffer to the shift register. Even

when received data is not copied from the shift register to the receive buffer in an overrun error status, on completion of the serial transfer, the SPI determines that the shift register is empty, so data transfer from the transmit buffer to the shift register is enabled.

5. When SPDR_HA is read either by the receive buffer full interrupt routine or processing of the receive buffer full interrupt using the SPRF flag, the receive data can be read.

If SPDR_HA is written to when the transmit buffer holds data that is not yet transmitted (the SPTEF flag is 0), the SPI does not update data in the transmit buffer. When writing to SPDR_HA, always use either a transmit buffer empty interrupt request or processing of the transmit buffer empty interrupt using the SPTEF flag. To use a transmit buffer empty interrupt, set the SPTIE bit in SPCR to 1. If the SPI function is disabled (the SPCR.SPE bit is 0), set the SPTIE bit to 0.

When serial transfer ends and the receive buffer is full (the SPRF flag is 1), the SPI does not copy data from the shift register to the receive buffer, and it detects an overrun error (see section 30.3.9. Error Detection). To prevent a receive data overrun error, read the received data using a receive buffer full interrupt request before the next serial transfer ends. To use an SPI receive buffer full interrupt, set the SPCR.SPRIE bit to 1.

Transmission and reception interrupts or the associated IELSRn.IR flags (where n is the interrupt vector number) in the ICU can be used to confirm the states of the transmit and receive buffers.

Similarly, the SPTEF and SPRF flags can be used to confirm the states of the transmit and receive buffers. See section 13, Interrupt Controller Unit (ICU) for the interrupt vector numbers.

## 30.3.8 Communication End Interrupt

### 30.3.8.1 Transmit-Receive/Transmit in Master Mode

The communication end interrupt (SPCI) is made and the CENDF flag is set to 1, when following conditions are satisfied in transmit-receive master mode and transmit master mode. The set timing of the CENDF flag is same as IDLNF flag. The communication end interrupt (SPCI) is one PCLKA width and low active.

- When the value of the SPSSR.SPCP[2:0] bits are same as the SPSCR.SPSLN[2:0] bits.
- When there is no next transmission data.

The CENDF flag will not be cleared even if the SPCR.SPE bit is cleared after CENDF = 1. The CENDF flag will be cleared in one of following tow conditions.

- When the next transmission data is written to the transmit buffer (SPTX).
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1.

Figure 30.31 shows an example of communication end interrupt operation during transmit-receive/transmit master mode.

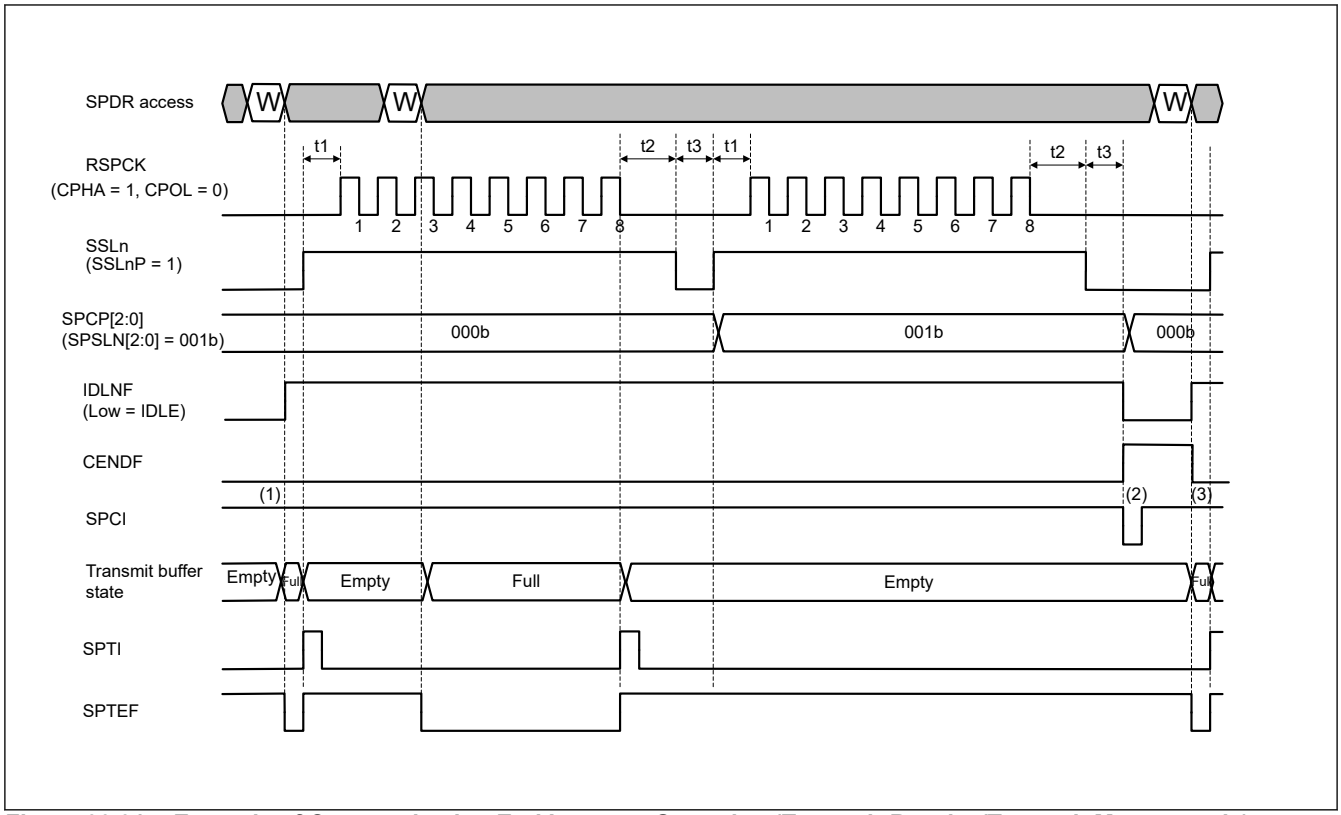



**Figure 30.31 Example of Communication End Interrupt Operation (Transmit-Receive/Transmit Master mode)**

1. The CNEDF flag is 0 and the level of SPCI is 1 before communication start, and these have kept during communication.
2. The CENDF flag will be 1 (Communication End) at the end of t3 cycle, because the next command is 000b and there is no next transmit data, and then the SPCI interrupt output when the CENDIE bit is 1.
3. The CENDF flag is cleared when the next transmission data is written to the transmit buffer (SPTX). Or the CENDF flag is cleared if the CENDF flag was written 0 after reading the SPCR when the CENDF flag was 1.

In slave mode operation, the output timing of the communication end interrupt is deferent due to the value of the SPCR.SPMS bit (SPI mode select bit), and the clear timing of the communication end interrupt is deferent due to the communication mode (transmit-receive or transmit-only or receive-only).

### 30.3.8.2 Transmit-Receive/Transmit in Slave Mode on SPI Operation (4-wire)

The communication end interrupt (SPCI) is made and the CENDF flag is set to 1, when both SPTX buffer and transmit shift buffer are empty in transmit-receive/transmit slave mode on SPI Operation (4-wire). The set timing of the CENDF flag is same as SSL0 negate timing. The communication end interrupt (SPCI) is one PCLKA width and low active.

The CENDF flag will not be cleared even if the SPCR.SPE bit is cleared after CENDF = 1. The CENDF flag will be cleared in one of following tow conditions.

- When the next transmission data is written to the transmit buffer (SPTX).
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1.

Figure 30.32 shows an example of communication end interrupt operation during transmit-receive/transmit slave mode on SPI operation.

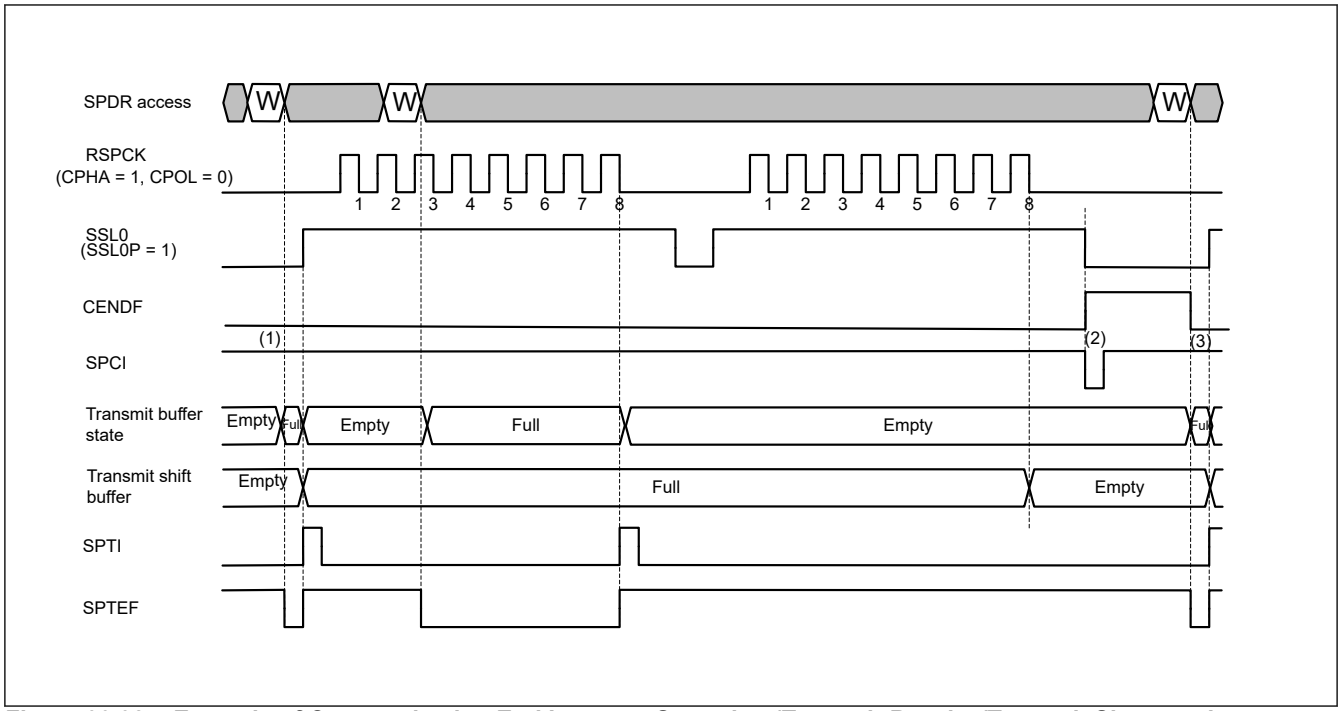



**Figure 30.32 Example of Communication End Interrupt Operation (Transmit-Receive/Transmit Slave mode on SPI Operation)**

1. The CNEDF flag is 0 and the level of SPCI is 1 before communication start, and these have kept during communication.
2. The CENDF flag will be 1 (Communication End) at the timing of SSL0 negate, when both SPTX buffer and transmit shift buffer are empty, and then the SPCI interrupt output when the CENDIE bit is 1.
3. The CENDF flag is cleared when the next transmission data is written to the transmit buffer (SPTX). Or the CENDF flag is cleared if the CENDF flag was written 0 after reading the SPCR when the CENDF flag was 1.

### 30.3.8.3 Receive Only in Slave Mode on SPI Operation (4-wire)

The communication end interrupt (SPCI) is made and the CENDF flag is set to 1 at the SSL0 negate timing in receive only slave mode on SPI operation (4-wire). The number of transmission frame is set by the SPDCR.SPFC[1:0]. Then the SSL0 is negated at the last frame transmission end. The communication end interrupt (SPCI) is one PCLKA width and low active.

The CENDF flag will not be cleared even if the SPCR.SPE bit is cleared after CENDF = 1. The CENDF flag will be cleared in one of following tow conditions.

- The SSL0 assert timing of next transmission.
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1.

Figure 30.33 shows an example of communication end interrupt operation during receive only slave mode on SPI operation (4-wire).

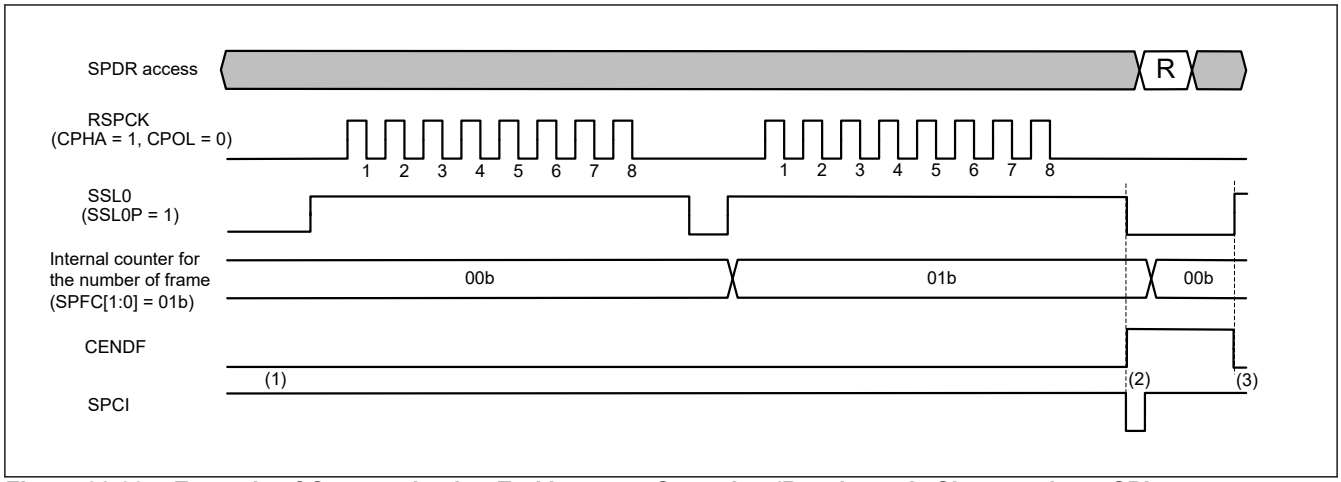



**Figure 30.33 Example of Communication End Interrupt Operation (Receive only Slave mode on SPI Operation)**

1. The CNEDF flag is 0 and the level of SPCI is 1 before communication start, and these have kept during communication.
2. The CENDF flag will be 1 (Communication End) at the timing of SSL0 negate, when the last frame transmission ends, and then the SPCI interrupt output when the CENDIE bit is 1.
3. The CENDF flag is cleared at the SSL0 assert when the next transmission start. Or the CENDF flag is cleared if the CENDF flag was written 0 after reading the SPCR when the CENDF flag was 1.

### 30.3.8.4 Transmit-Receive/Transmit in Slave Mode on Clock Synchronous Operation (3-wire)

The communication end interrupt (SPCI) is made and the CENDF flag is set to 1 when both SPTX buffer and transmit shift register are empty in transmit-receive/transmit slave mode on clock synchronous operation (3-wire). The set timing of CENDF flag is same as the last data sampling of the RSPCK (the last odd edge of RSPCK when the SPCMD0.CPHA bit is 0, the last even edge of RSPCK when the SPCMD0.CPHA bit is 1). The communication end interrupt (SPCI) is one PCLKA width and low active.

The CENDF flag will not be cleared even if the SPCR.SPE bit is cleared after CENDF = 1. The CENDF flag will be cleared in one of following tow conditions.

- The SSL0 assert timing of next transmission.
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1.

Figure 30.34 shows an example of communication end interrupt operation during transmig-receive/transmit slave mode on clock synchronous operation (3-wire).

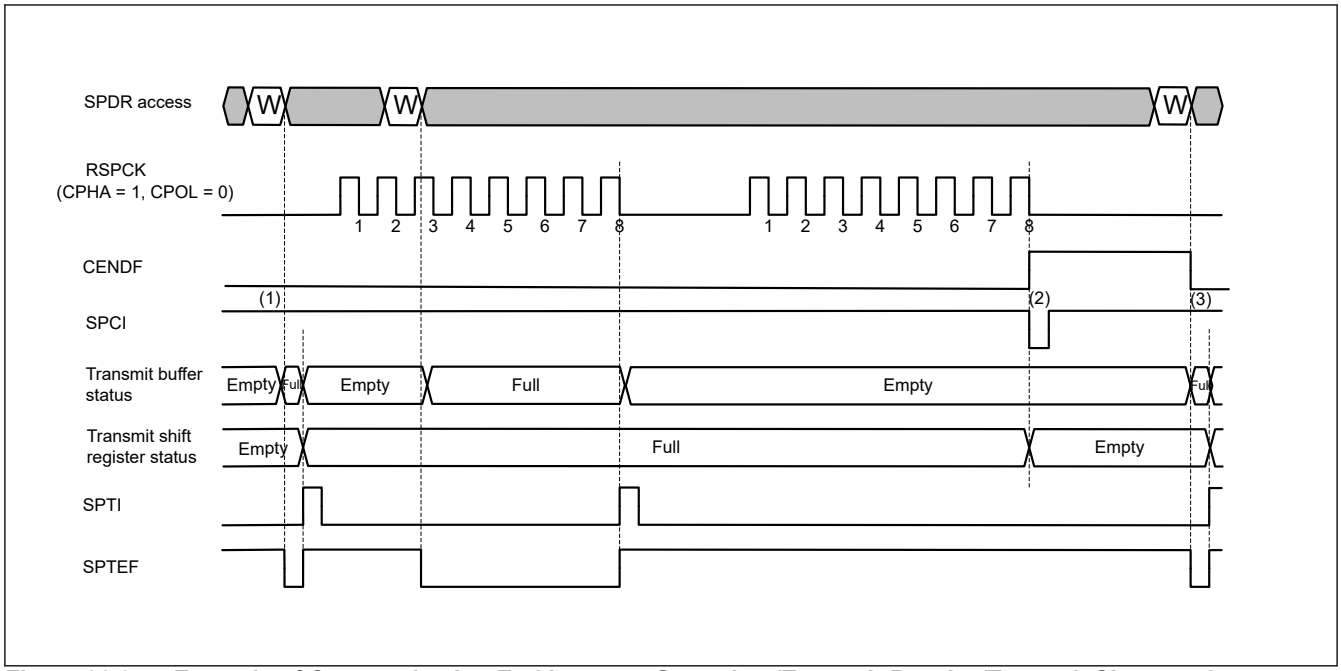



**Figure 30.34 Example of Communication End Interrupt Operation (Transmit-Receive/Transmit Slave mode on Clock Synchronous Operation)**

1. The CNEDF flag is 0 and the level of SPCI is 1 before communication start, and these have kept during communication.
2. The CENDF flag will be 1 (Communication End) at the last data sampling timing of RSPCK, when both SPTX buffer and transmit shift buffer are empty, and then the SPCI interrupt output when the CENDIE bit is 1.
3. The CENDF flag is cleared when the next transmission data is written to the transmit buffer (SPTX). Or the CENDF flag is cleared if the CENDF flag was written 0 after reading the SPCR when the CENDF flag was 1.

### 30.3.8.5 Receive Only in Slave Mode on Clock Synchronous Operation (3-wire)

The communication end interrupt (SPCI) is made and the CENDF flag is set to 1 at the last data sampling of the last transmission frame in receive only slave mode on clock synchronous operation (3-wire). The sampling timing is the last odd edge of RSPCK when the SPCMD0.CPHA bit is 0, the last even edge of RSPCK when the SPCMD0.CPHA bit is 1. The number of transmission frame is set by the SPDCR.SPFC[1:0]. The communication end interrupt (SPCI) is one PCLKA width and low active.

The CENDF flag will not be cleared even if the SPCR.SPE bit is cleared after CENDF = 1. The CENDF flag will be cleared in one of following tow conditions.

- The first edge of RSPCK for next transmission.
- The CENDF flag was written 0 after reading the SPSR when the CENDF flag was 1.

Figure 30.35 shows an example of communication end interrupt operation during receive only slave mode on clock synchronous operation.

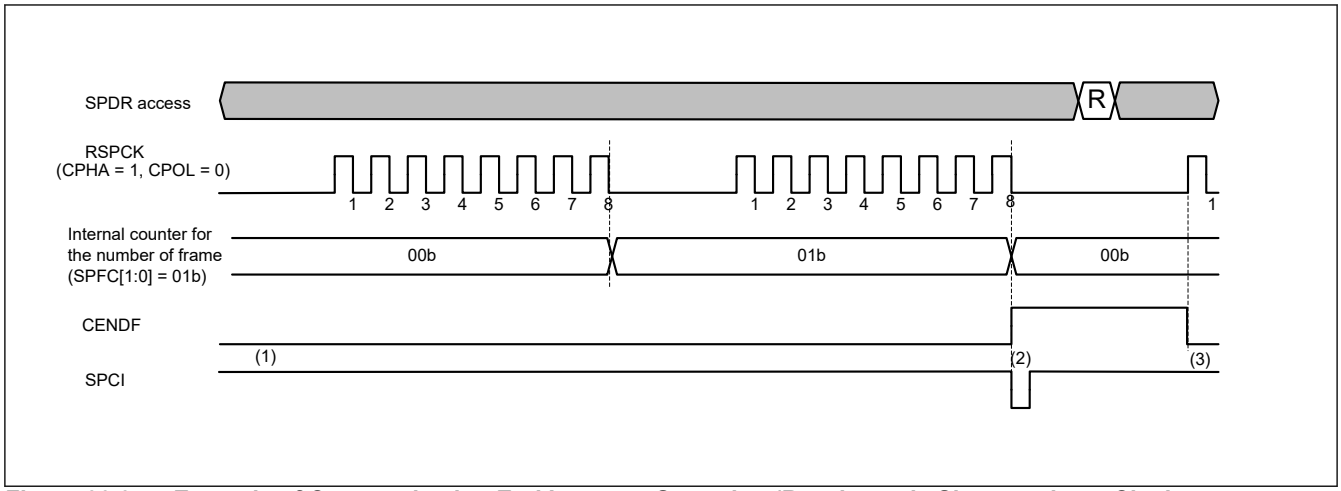



**Figure 30.35 Example of Communication End Interrupt Operation (Receive-only Slave mode on Clock Synchronous Operation)**

1. The CNEDF flag is 0 and the level of SPCI is 1 before communication start, and these have kept during communication.
2. The CENDF flag will be 1 (Communication End) at the last data sampling timing of RSPCK, when the last frame transmission end. The number of transmission frame is set by the SPDCR.SPFC[1:0]. And then the SPCI interrupt output when the CENDIE bit is 1.
3. The CENDF flag is cleared at the first edge of RSPCK for the next transmission. Or the CENDF flag is cleared if the CENDF flag was written 0 after reading the SPCR when the CENDF flag was 1.

### 30.3.8.6 Common Operation

In this chapter, the operation common to each mode / area option communication in section 30.3.8.1. Transmit-Receive/ Transmit in Master Mode to section 30.3.8.5. Receive Only in Slave Mode on Clock Synchronous Operation (3-wire) is explained. When the enable of SPI communication end interrupt (CENDIE) is 0, at the time of communication completion, a flag of communication end (CENDF) is set and an event of communication end (sp_elccend) is output, but no interrupt is output. However, if the enable of communication end interrupt (CENDIE) is set to 1 before clearing the flag of communication end (CENDF) while the enable of SPI function (SPE) is 1, the communication end interrupt is output.

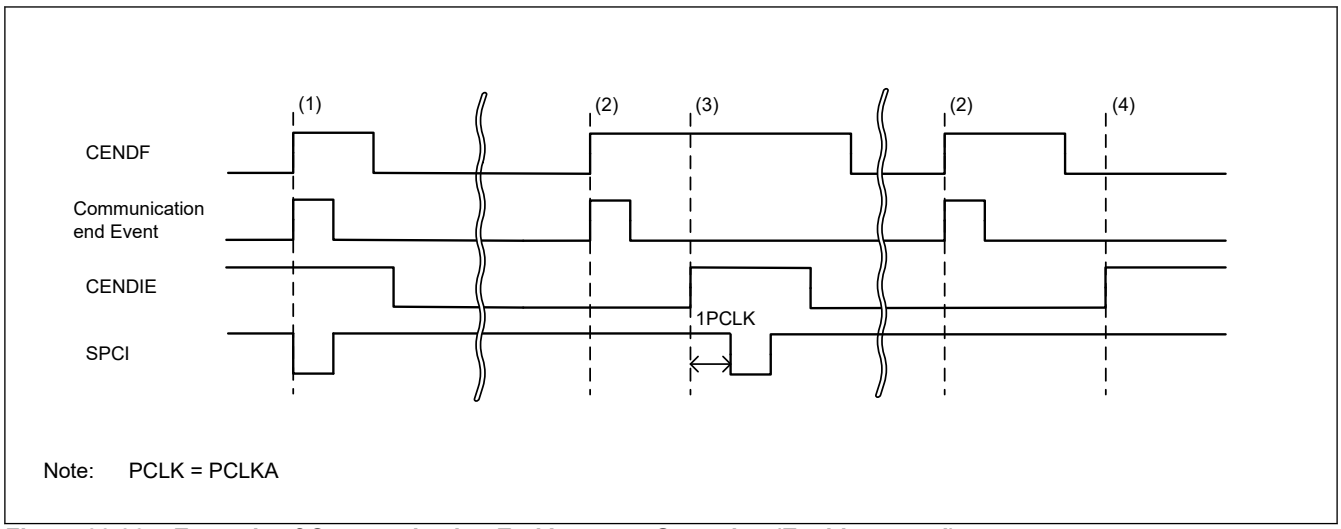



**Figure 30.36 Example of Communication End Interrupt Operation (Enable control)**

1. When the enable of SPI communication end interrupt (CENDIE) is 1, at the time of communication completion, the following three are the same timing.
   - A flag of communication end (CENDF)
   - An event of communication end (sp_elccend)
   - The communication end interrupt

2. When the enable of SPI communication end interrupt (CENDIE) is 0, at the time of communication completion, the following two are the same timing, but no interrupt.
   - A flag of communication end (CENDF)
   - An event of communication end (sp_elccend)
3. After (2), if the enable of communication end interrupt (CENDIE) is set when the enable of SPI function (SPE) and the flag of communication end (CENDF) are 1, the communication end interrupt is output after 1 PCLKA.
4. After (2), even if the enable of communication end interrupt (CENDIE) is set when the enable of SPI function (SPE) or the flag of communication end (CENDF) is 0, the communication end interrupt is not output.

### 30.3.9 Error Detection

In normal SPI serial transfers, data written to the transmit buffer of SPDR/SPDR_HA is transmitted, and received data can be read from the receive buffer of SPDR/SPDR_HA. If access is made to SPDR/SPDR_HA, an abnormal transfer might occur, depending on the status of the transmit or receive buffer or the status of the SPI at the beginning or end of serial transfer.

If an abnormal transfer occurs, the SPI detects the event as an underrun error, overrun error, parity error, or mode fault error. Table 30.10 lists the relationship between non-normal transfer operations and the SPI error detection function.

**Table 30.10 Relationship between non-normal transfer operations and SPI error detection**

| Operation | Occurrence condition | SPI operation | Error detection |
|---|---|---|---|
| 1 | SPDR/SPDR_HA is written when the transmit buffer is full. | • The contents of the transmit buffer are kept<br>• Write data is missing | None |
| 2 | SPDR/SPDR_HA is read when the receive buffer is empty. | The contents of the receive buffer and previously received data are output. | None |
| 3 | Serial transfer is started in slave mode when the SPI is not able to transmit data. | • Serial transfer is suspended<br>• Transmit or receive data is missing<br>• Driving of the MISOn output signal is stopped<br>• SPI function is disabled | Underrun error |
| 4 | Serial transfer terminates when the receive buffer is full. | • Keeps the contents of the receive buffer<br>• Missing receive data | Overrun error |
| 5 | An incorrect parity bit is received during full-duplex synchronous serial communication with the parity function enabled in following mode:<br>• Transmit-receive master mode<br>• Transmit-receive slave mode<br>• Receive-only slave mode | The parity error flag is asserted | Parity error |
| 6 | The SSLn0 input signal is asserted when the serial transfer is idle in multi-master mode. | • Driving of the RSPCKn, MOSIn, SSLn1 to SSLn3 output signals is stopped<br>• SPI function is disabled | Mode fault error |
| 7 | The SSLn0 input signal is asserted during serial transfer in multi-master mode. | • Serial transfer is suspended<br>• Transmit or receive data is missing<br>• Driving of the RSPCKn, MOSIn, SSLn1 to SSLn3 output signals is stopped<br>• SPI function is disabled | Mode fault error |
| 8 | The SSLn0 input signal is negated during serial transfer in slave mode. | • Serial transfer is suspended<br>• Transmit or receive data is missing<br>• Driving of the MISOn output signal is stopped<br>• SPI function is disabled | Mode fault error |

In operation 1 described in Table 30.10, the SPI does not detect an error. To prevent data omission during writes to SPDR/SPDR_HA, the writes to SPDR/SPDR_HA must be executed using a transmit buffer empty interrupt request (when the SPSR.SPTEF flag is 1).

Similarly, the SPI does not detect an error in operation 2. To prevent extraneous data from being read, SPDR/SPDR_HA read must be executed with an SPI receive buffer full interrupt request (when the SPSR.SPRF flag is 1).

For information on the other errors, see the following sections:

- Underrun error, indicated in operation 3, see section 30.3.9.4. Underrun errors
- Overrun error, indicated in operation 4, see section 30.3.9.1. Overrun errors
- Parity error, indicated in operation 5, see section 30.3.9.2. Parity errors
- Mode fault error, indicated in operations 6 to 8, see section 30.3.9.3. Mode fault errors
- For the transmit and receive interrupts, see section 30.3.7. Transmit Buffer Empty and Receive Buffer Full Interrupts.

### 30.3.9.1 Overrun errors

If a serial transfer ends when the receive buffer of SPDR/SPDR_HA is full, the SPI detects an overrun error and sets the SPSR.OVRF flag to 1. When the OVRF flag is 1, the SPI does not copy data from the shift register to the receive buffer, so the data prior to the error occurrence is retained in the receive buffer. To set the OVRF flag to 0, write 0 to the OVRF flag after the CPU reads SPSR with the OVRF flag set to 1.

Figure 30.37 shows an example of operation of the OVRF and SPRF flags. The SPSR and SPDR_HA accesses shown in Figure 30.37 indicate the condition of accesses to the SPSR and SPDR_HA register, where W denotes a write cycle, and R a read cycle. In this example, the SPI performs an 8-bit serial transfer when SPCMDm.CPHA bit is 1 and the SPCMDm.CPOL bit is 0. The numbers given for RSPCKn in the waveform represent the number of RSPCK cycles, such as the number of transferred bits.

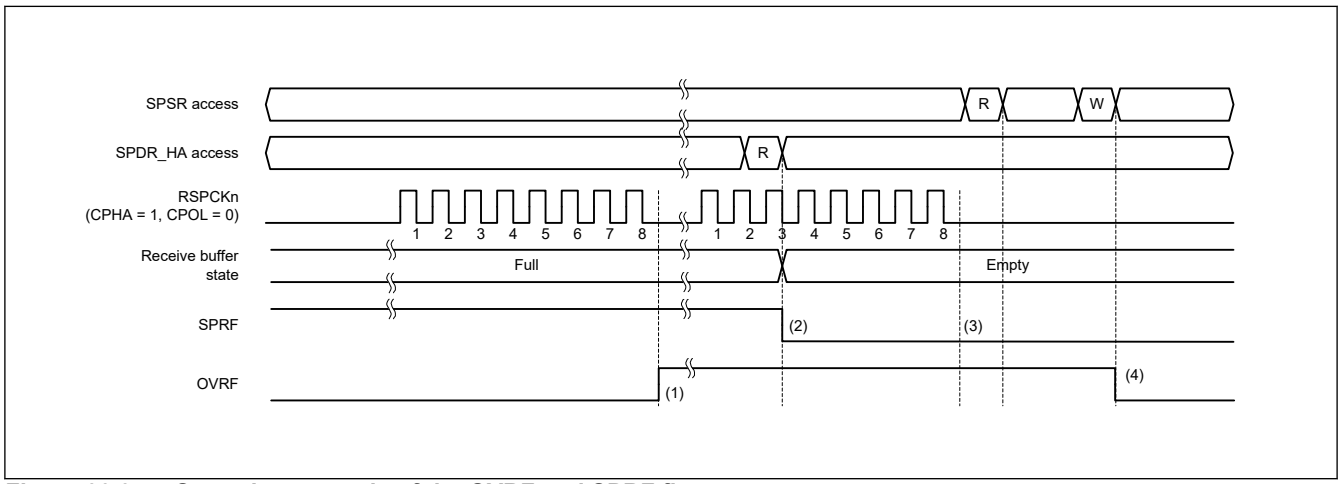



**Figure 30.37 Operation example of the OVRF and SPRF flags**

The operation of the flags at timings (1) to (4) in Figure 30.37 is as follows:

1. If a serial transfer terminates with the SPRF flag set to 1 (receive buffer full), the SPI detects an overrun error, and sets the OVRF flag to 1. The SPI does not copy the data in the shift register to the receive buffer. Even when the SPPE bit is 1, parity errors are not detected.
2. When SPDR/SPDR_HA is read, the SPI outputs the data in the receive buffer. The SPRF flag is then set to 0. The receive buffer becoming empty does not set the OVRF flag to 0.
3. If the serial transfer ends with the OVRF flag set to 1 (overrun error occurred), the SPI does not copy data in the shift register to the receive buffer (the SPRF flag does not set to 1). A receive buffer full interrupt is not generated. Even when the SPPE bit is 1, parity errors are not detected. In an overrun error state when the SPI does not copy the received data from the shift register to the receive buffer, on termination of the serial transfer, the SPI determines that the shift register is empty. This enables data transfer from the transmit buffer to the shift register.
4. If 0 is written to the OVRF flag after SPSR is read when the OVRF flag is 1, the OVRF flag clears is set to 0.

The occurrence of an overrun can be checked either by reading SPSR or by using an SPI error interrupt and reading SPSR. When executing a serial transfer, you must ensure that overrun errors are detected early, for example by reading SPSR immediately after SPDR/SPDR_HA/SPDR_BY is read.

If an overrun error occurs and the OVRF flag sets to 1, normal reception operations cannot be performed until the OVRF flag is set to 0.

When the RSPCK auto-stop function is enabled (SPCR2.SCKASE = 1) in master mode, an overrun error does not occur. Figure 30.38 and Figure 30.39 show the clock stop waveform when a serial transfer continues while the receive buffer is full in master mode.

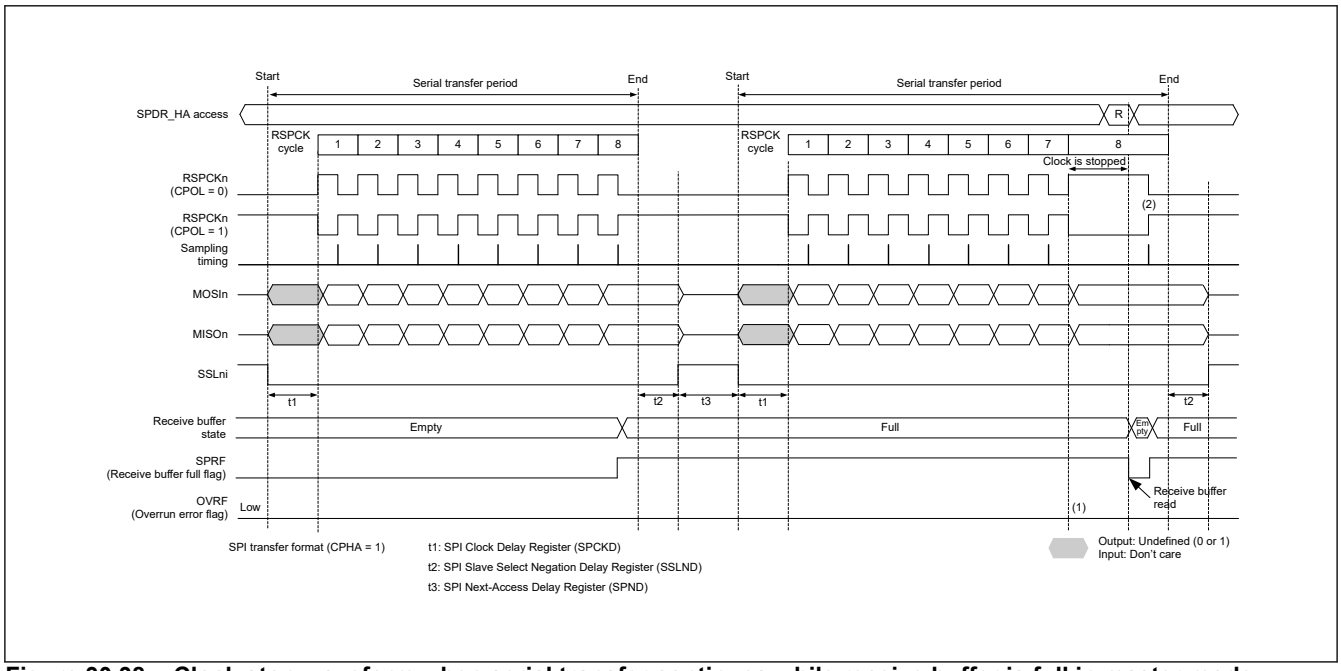



**Figure 30.38 Clock stop waveform when serial transfer continues while receive buffer is full in master mode (CPHA = 1)**

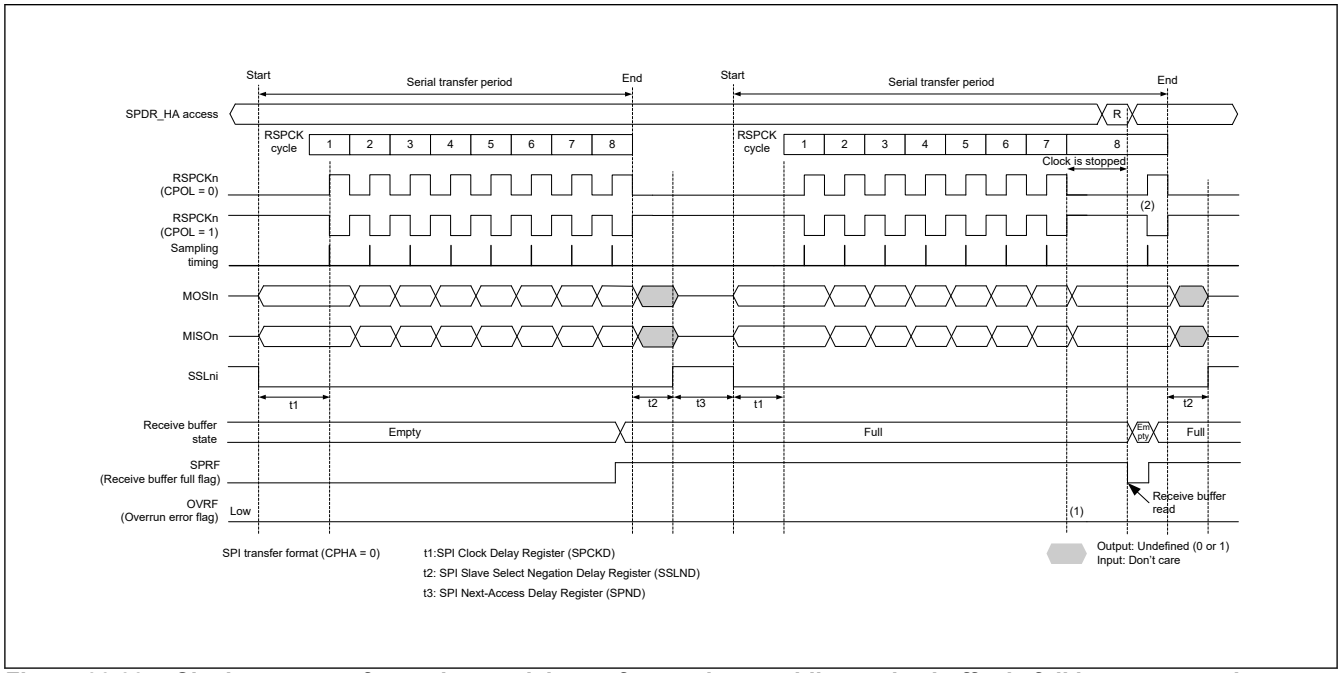



**Figure 30.39 Clock stop waveform when serial transfer continues while receive buffer is full in master mode (CPHA = 0)**

The operation of the flags at timings (1) and (2) in Figure 30.38 and Figure 30.39 is as follows:

1. When the receive buffer is full, an overrun error does not occur because the RSPCK clock is stopped.
2. If SPDR/SPDR_HA is read while the clock is stopped, data in the receive buffer can be read. The RSPCK clock restarts after reading the receive buffer (after the SPSR.SPRF flag is set to 0).

Overrun error does not occur when RSPCK automatic stop function is enabled for transfer with no delay of between frames during burst transfer in master mode. Figure 30.40 and Figure 30.41 show the clock stop waveform, when there is no delay between frames at burst transfer and the serial transfer continues in the reception buffer full state.

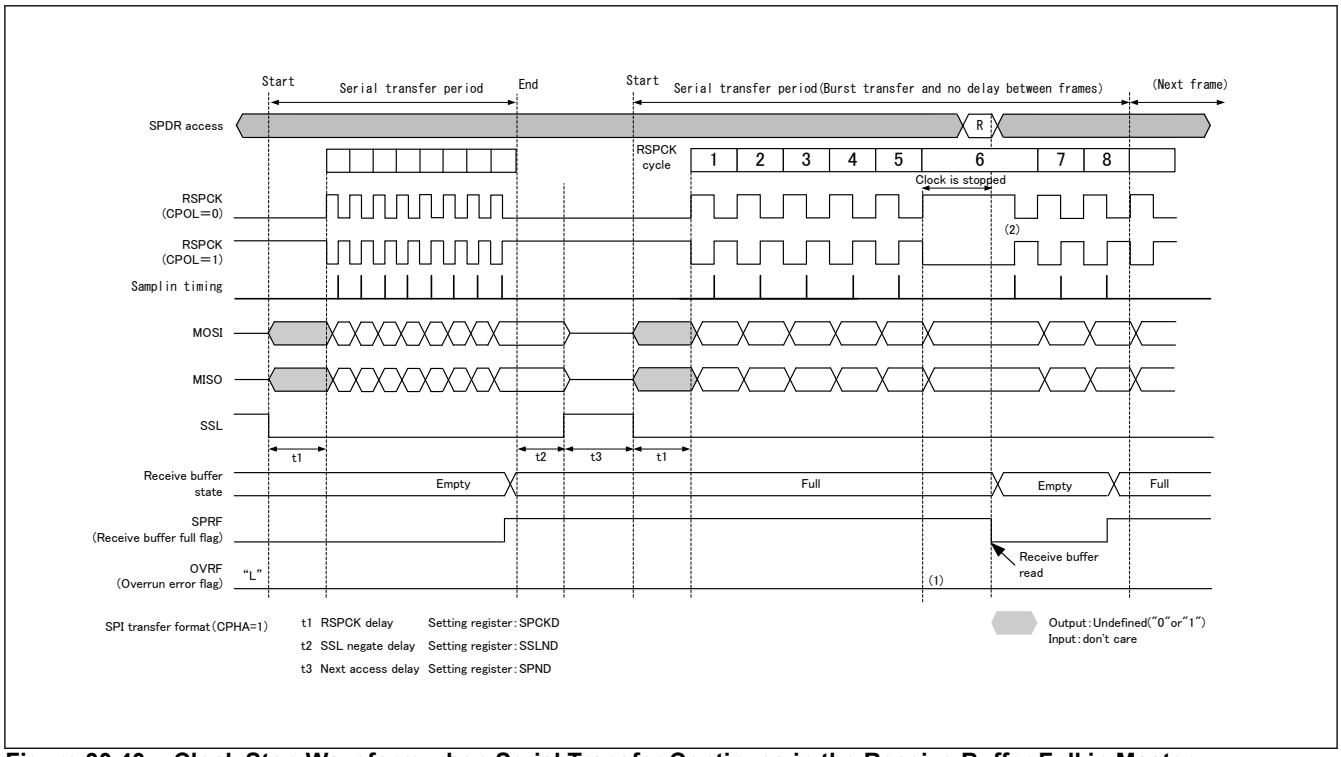



**Figure 30.40 Clock Stop Waveform when Serial Transfer Continues in the Receive Buffer Full in Master Mode (at burst transfer and no delay between frames CPHA = 1)**

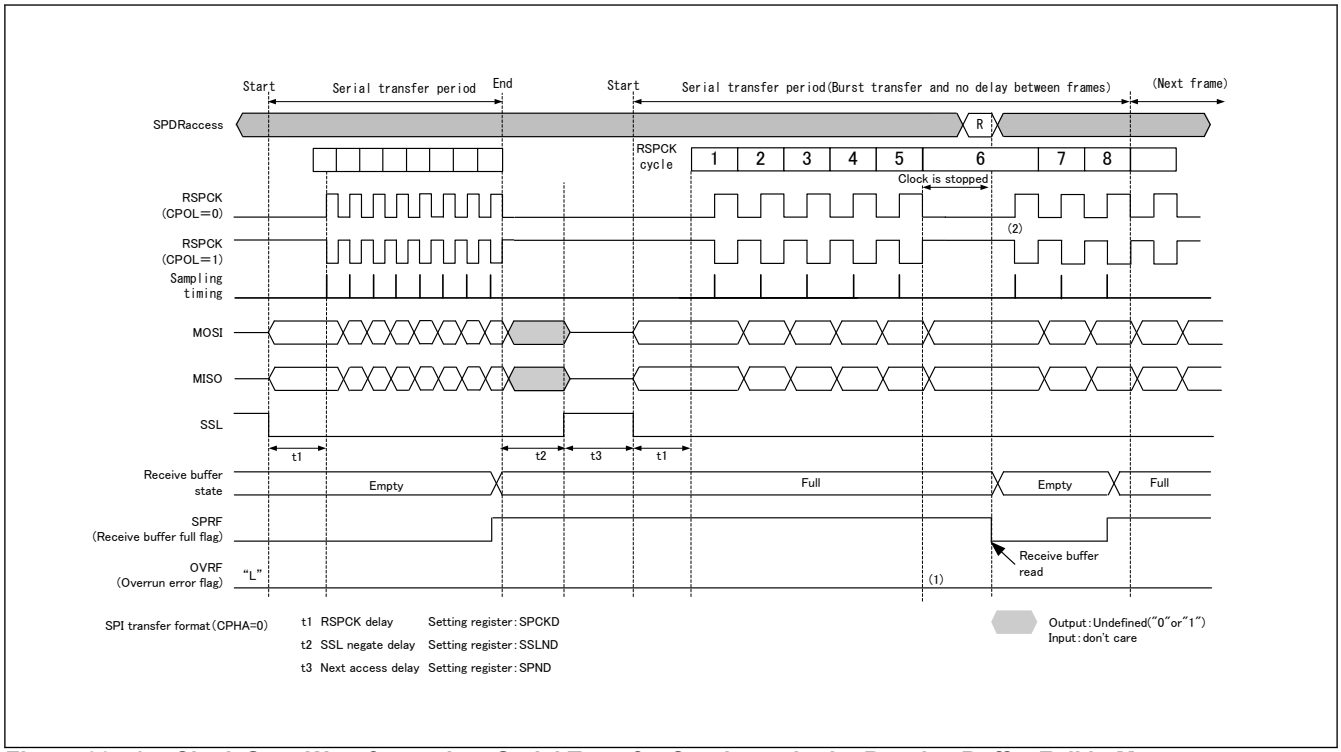



**Figure 30.41 Clock Stop Waveform when Serial Transfer Continues in the Receive Buffer Full in Master Mode (at burst transfer and no delay between frames CPHA = 0)**

The following describes operation of flags at timings (1) and (2) in the figure above.

1. While the receive buffer is full, the RSPCK clock is deactivated and no overrun error occurs.
2. Receive buffer data can be read by reading SPDR during clock stop. After the receive buffer data has been read (after the SPSR.SPRF flag has been cleared to 0), the RSPCK clock restarts.

### 30.3.9.2 Parity errors

When full-duplex synchronous serial communications is performed with the SPCR.TXMD bit set to 0 and the SPCR2.SPPE bit set to 1, when serial transfer ends, the SPI checks whether there are parity errors. On detecting a parity error in the received data, the SPI sets the SPSR.PERF flag to 1. Because the SPI does not copy data in the shift register to the receive buffer when the SPSR.OVRF flag is set to 1, parity error detection is not performed for the received data. To set the PERF flag to 0, write 0 to the PERF flag after the SPSR register is read with the PERF flag set to 1.

Figure 30.42 shows an example of operation of the OVRF and PERF flags. The SPSR access shown in Figure 30.42 indicates the condition of access to the register, where W denotes a write cycle, and R a read cycle. In this example, full-duplex serial communication is performed while the SPCR2.SPPE bit is 1. The SPI performs an 8-bit serial transfer when SPCMDm.CPHA bit is 1 and the SPCMDm.CPOL bit is 0. The numbers given for RSPCKn in the waveform represent the number of RSPCK cycles, such as the number of transferred bits.

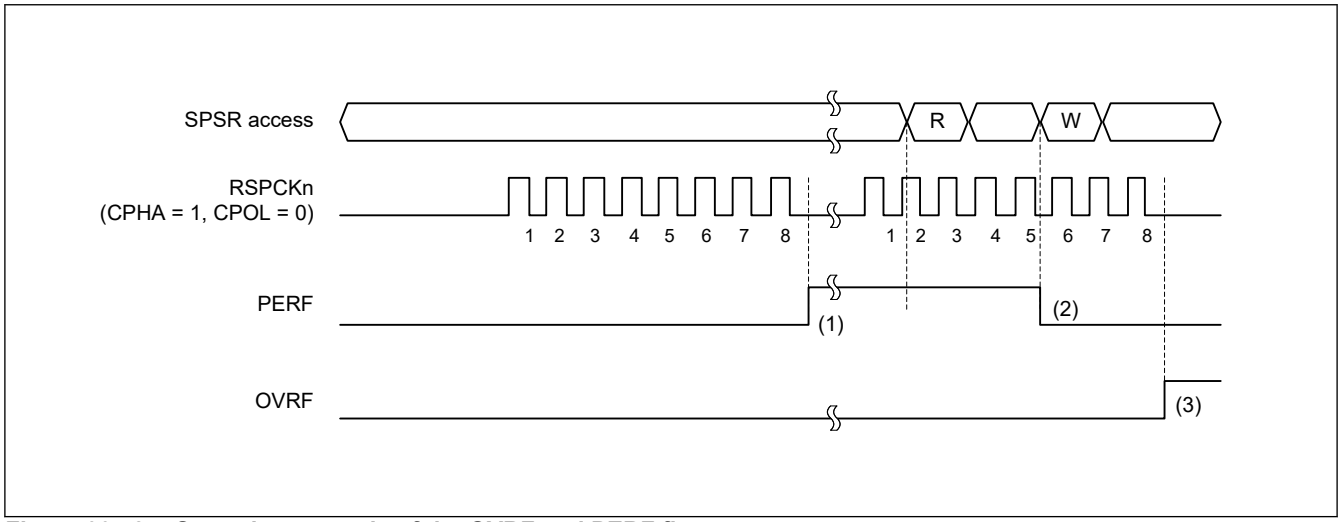



**Figure 30.42 Operation example of the OVRF and PERF flags**

The operation of the flags at timings (1) to (3) in Figure 30.42 is as follows:

1. If a serial transfer terminates with the SPI not detecting an overrun error, the SPI copies the data in the shift register to the receive buffer. The SPI checks the received data at this time and sets the PERF flag to 1 if a parity error is detected.
2. If 0 is written to the PERF flag after the SPSR register is read when the PERF flag is 1, the PERF flag is set to 0.
3. When the SPI detects an overrun error and serial transfer is terminated, the data in the shift register is not copied to the receive buffer. The SPI does not perform parity error detection at this time.

Parity errors can be checked for by either reading the SPSR register or using an SPI error interrupt and reading the SPSR register. When executing a serial transfer, such checks are required to ensure early detection of parity errors. When the SPI is used in master mode, the pointer value to the SPCMDm register at the occurrence of the error can be checked by reading the SPSSR.SPECM[2:0] bits (Only SPI0).

### 30.3.9.3 Mode fault errors

The SPI operates in multi-master mode when the SPCR.MSTR bit is 1, the SPCR.SPMS bit is 0, and the SPCR.MODFEN bit is 1. If the active level is input for the SSLn0 input signal of the SPI in multi-master mode, the SPI detects a mode fault error regardless of the status of the serial transfer, and sets the SPSR.MODF flag to 1. On detecting the mode fault error, the SPI copies the value of the pointer to SPCMDm to the SPDCR.SPFC[1:0] bits. The active level of the SSLn0 signal is determined by the SSLP.SSL0P bit.

When the MSTR bit is 0, the SPI operates in slave mode. The SPI detects a mode fault error if the MODFEN bit of the SPI in slave mode is 1, and the SPMS bit is 0, and if the SSLn0 input signal is negated during the serial transfer period (from the time the driving of valid data is started to the time the final valid data is fetched).

On detecting a mode fault error, the SPI stops the driving of the output signals and clears the SPCR.SPE bit to 0 (see section 30.3.10. Initializing the SPI). For multi-master configuration, detection of a mode fault error is used to stop the driving of output signals and the SPI function, which allows the master to be released.

The occurrence of a mode fault error can be checked either by reading SPSR or by using an SPI error interrupt and reading SPSR. Detecting mode-fault errors without using the SPI error interrupt requires polling of SPSR. When using the SPI in master mode, the value of the pointer to the SPCMDm register at the occurrence of the error can be checked by reading the SPSSR.SPECM[2:0] bits.

When the MODF flag is 1, writing 1 to the SPE bit is ignored by the SPI. To enable the SPI function after the detection of a mode fault error, the MODF flag must be set to 0.

#### 30.3.9.4 Underrun errors

While the SPI is operating in slave mode (SPCR.MSTR bit = 0) and the Extended Communication Mode Select bit (ETXMD) in the SPI Control Register 3 (SPCR3) is set to 0, if serial transfer is started before transmit data output is ready with the SPCR.SPE bit set to 1 (SPI function enabled), the SPI detects an underrun error and sets the SPSR.MODF and SPSR.UDRF flags to 1.

On detecting an underrun error, the SPI stops the driving of output signals and clears the SPCR.SPE bit to 0 (see section 30.3.10. Initializing the SPI).

The occurrence of underrun errors can be checked either by reading SPSR or by using an SPI error interrupt and reading SPSR. Detecting underrun errors without using the SPI error interrupt requires polling of SPSR.

When the MODF flag is 1, writing 1 to the SPE bit is ignored by the SPI. To enable the SPI function after the detection of an underrun error, the MODF flag must be set to 0.

### 30.3.10 Initializing the SPI

If 0 is written to the SPCR.SPE bit or if the SPI sets the SPE bit to 0 because it detected a mode fault error or an underrun error, the SPI disables the SPI function and initializes some of the module functions. When a system reset is generated, the SPI initializes all of the module functions. This section describes initialization by clearing of the SPCR.SPE bit and by a system reset.

#### 30.3.10.1 Initialization by clearing of the SPCR.SPE bit

When the SPCR.SPE bit is set to 0, the SPI initializes by:

- Suspending any serial transfer that is being executed
- Stopping the driving of output signals (Hi-Z) in slave mode
- Initializing the internal state of the SPI
- Initializing the transmit buffer of the SPI (the SPSR.SPTEF flag sets to 1)

Initialization by clearing of the SPE bit does not initialize the control bits of the SPI. For this reason, the SPI can be started in the same transfer mode in use prior to initialization when the SPE bit is set to 1 again.

The SPSR.CENDF, SPSR.SPRF, SPSR.OVRF, SPSR.MODF, SPSR.PERF, and SPSR.UDRF flags are not initialized, and the value of the SPI Sequence Status Register (SPSSR) is not initialized. Therefore, even after the SPI is initialized, data from the receive buffer can be read to check the communication completion status and the error status during an SPI transfer.

The transmit buffer is initialized to an empty state (the SPSR.SPTEF flag sets to 1). Therefore, if the SPCR.SPTIE bit is set to 1 after SPI initialization, a transmit buffer empty interrupt is generated. To disable any transmit buffer empty interrupts when the SPI is initialized, write 0 to the SPTIE bit simultaneously while writing 0 to the SPE bit.

#### 30.3.10.2 Initialization by system reset

A system reset completely initializes the SPI by initializing all SPI control bits, status bits, and data registers, in addition to meeting the requirements described in section 30.3.10.1. Initialization by clearing of the SPCR.SPE bit.

## 30.3.11 SPI Operation

### 30.3.11.1 Master mode operation

The only difference between single- and multi-master mode operation is the use of mode fault error detection (see section 30.3.9. Error Detection). In single-master mode, the SPI does not detect mode fault errors whereas in multi-master mode, it does. This section explains operations that are common to both modes.

#### (1) Starting a serial transfer

The SPI updates the data in the transmit buffer (SPTX) when data is written to the SPI Data Register (SPDR/SPDR_HA) with the SPI transmit buffer empty, data for the next transfer is not set, and the SPSR.SPTEF flag is 0. When the shift register is empty after the number of frames set in the SPDCR.SPFC[1:0] bits are written to the SPDR/SPDR_HA/ SPDR_BY, the SPI copies data from the transmit buffer to the shift register and starts serial transfer. On copying transmit data to the shift register, the SPI changes the status of the shift register to full. On termination of the serial transfer, it changes the status of the shift register to empty. The status of the shift register cannot be referenced.

The polarity of the SSLni output pins depends on the SSLP register settings. For details on the SPI transfer format, see section 30.3.5. Transfer Formats.

#### (2) Terminating a serial transfer

Regardless of the SPCMDm.CPHA bit setting, the SPI terminates the serial transfer after transmitting an RSPCKn edge associated with the final sampling timing. If free space is available in the receive buffer (SPRX) (the SPSR.SPRF flag is 0), on termination of the serial transfer, the SPI copies data from the shift register to the receive buffer of the SPDR/SPDR_HA register.

The final sampling timing varies depending on the bit length of transfer data. In master mode, the SPI data length depends on the SPCMDm.SPB[3:0] bit settings. The polarity of the SSLni output pin depends on the SSLP register settings. For details on the SPI transfer format, see section 30.3.5. Transfer Formats.

#### (3) Sequence control

The transfer format used in master mode is determined by the SPSCR, SPCMDm, SPBR, SPCKD, SSLND, and SPND registers.

The SPSCR register determines the sequence configuration for serial transfers that are executed by the SPI in master mode. The following items are set in the SPCMDm register:

- SSLni pin output signal value
- MSB- or LSB-first
- Data length
- Some of the bit rate settings
- RSPCK polarity and phase
- Whether SPCKD is to be referenced
- Whether SSLND is to be referenced
- Whether SPND is to be referenced

SPBR holds some of the bit rate settings, including SPCKD (SPI clock delay), SSLND (SSL negation delay), and SPND (next-access delay).

Based on the sequence length assigned in SPSCR, the SPI makes up a sequence comprised of a part or all of the SPCMDm register. The SPI contains a pointer to the SPCMDm register that makes up the sequence. The value of this pointer can be checked by reading the SPSSR.SPCP[2:0] bits. When the SPCR.SPE bit is set to 1 and the SPI function is enabled, the SPI loads the pointer to the commands in SPCMD0, and incorporates the SPCMD0 settings into the transfer format at the beginning of serial transfer. The SPI increments the pointer each time the next-access delay period for a data transfer ends. On completion of the serial transfer that corresponds to the final command in the sequence, the SPI sets the pointer to SPCMD0, and in this way the sequence is executed repeatedly.

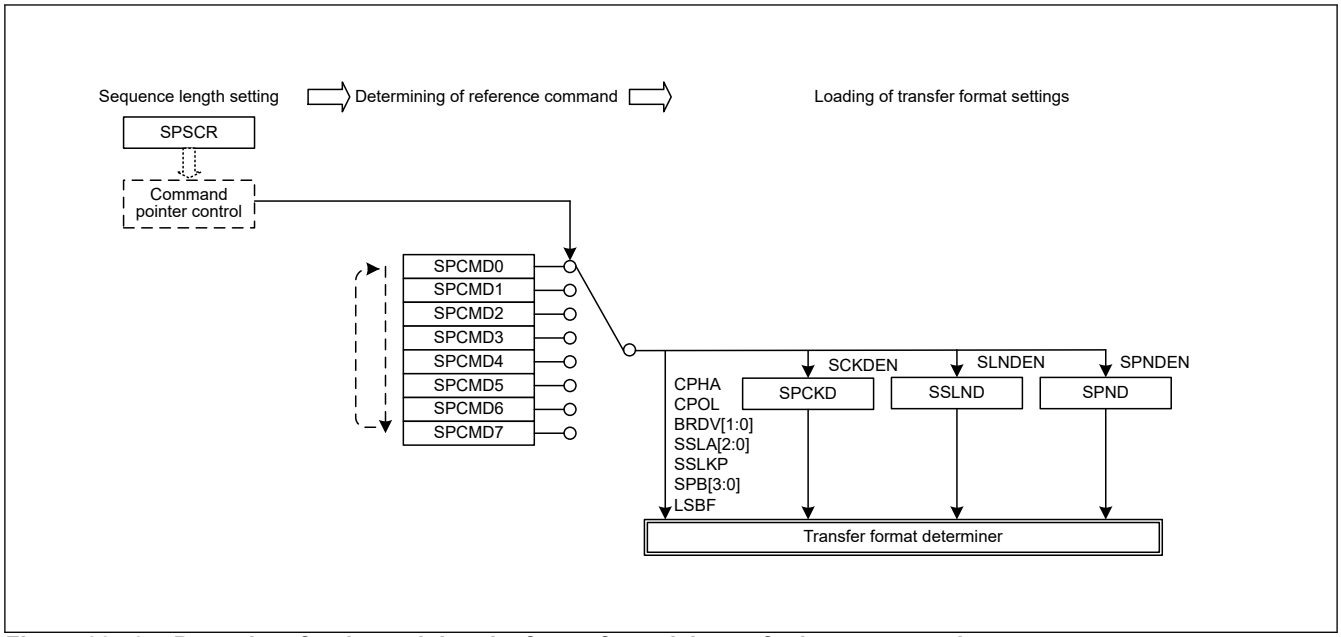



**Figure 30.43 Procedure for determining the form of a serial transfer in master mode**

In this section, a frame is the combination of the data in SPDR/SPDR_HA and the settings in SPCMDm.

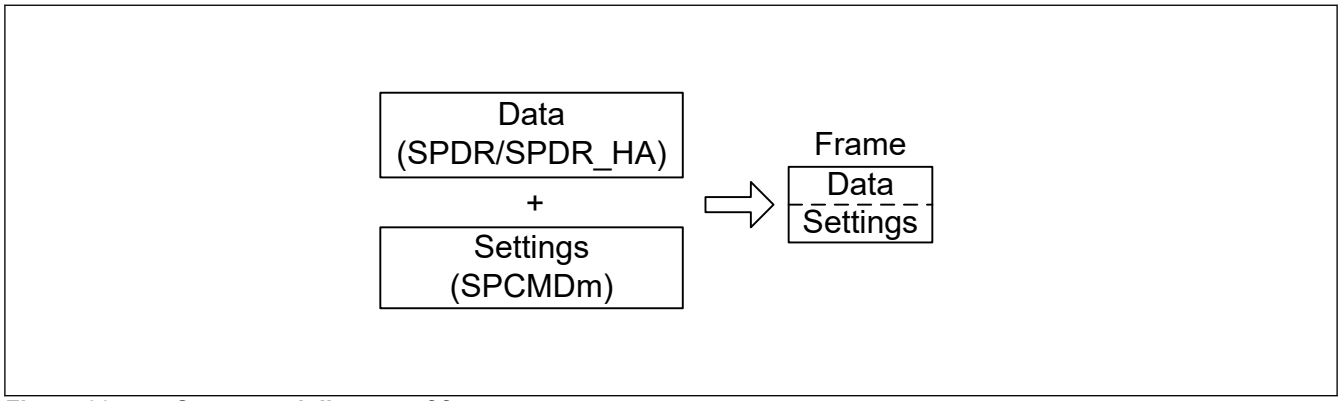



**Figure 30.44 Conceptual diagram of frames**

Figure 30.45 shows the correspondence between the commands and the transmit and receive buffers in the sequence of operations specified by the settings in Table 30.4.

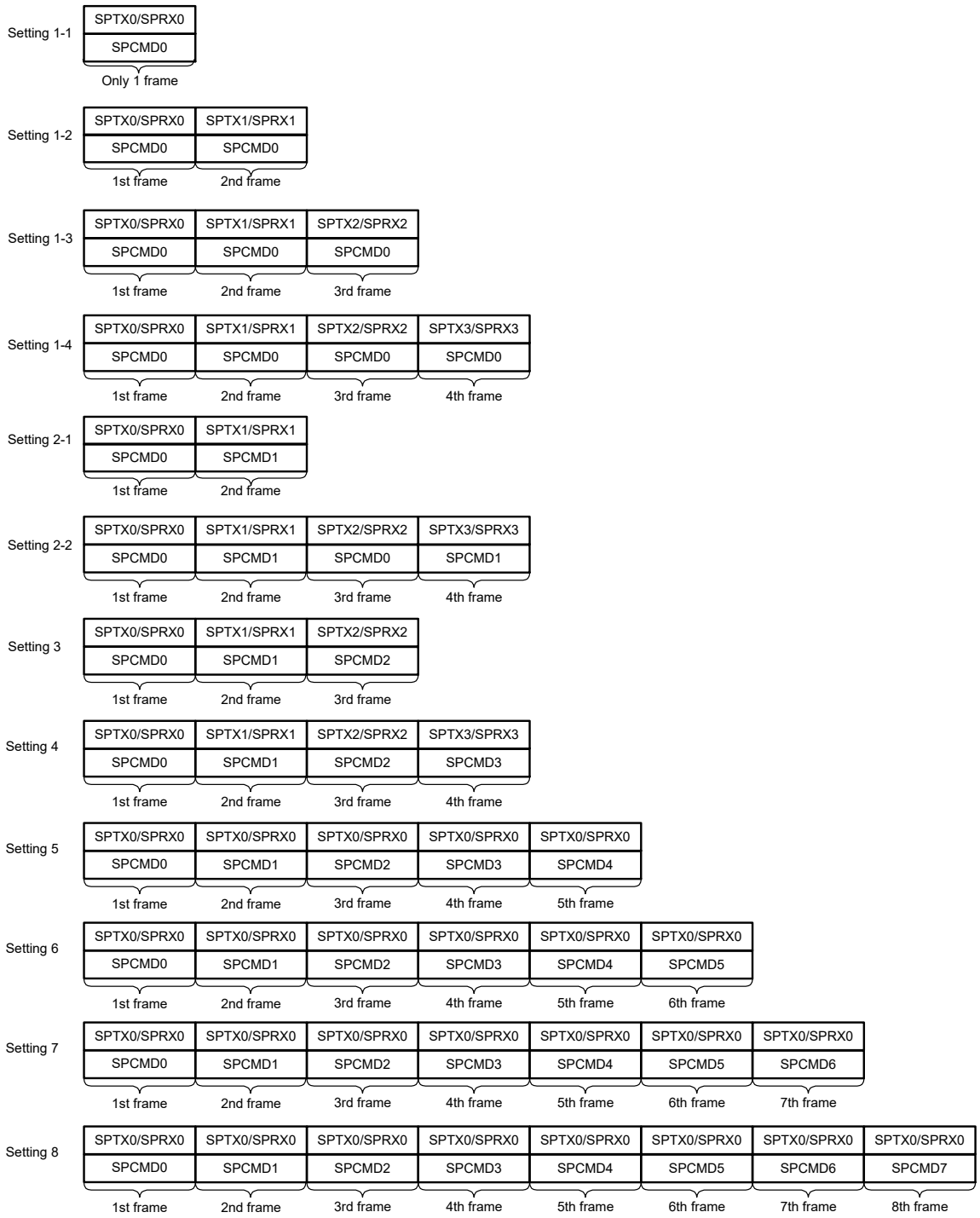



**Figure 30.45 Correspondence between SPI Command Register and transmit and receive buffers in sequence operations**

## (4) Burst transfers

If the SPCMDm.SSLKP bit that the SPI references during the current serial transfer is 1, the SPI maintains the SSLni signal level during the serial transfer until the beginning of the SSLni signal assertion for the next serial transfer. If the SSLni

signal level for the next serial transfer is the same as the SSLni signal level for the current serial transfer, the SPI can execute continuous serial transfers while keeping the SSLni signal assertion status (burst transfer).

- When Between Burst Transfer Frames Delay Select bit (BFDS) of SPI control register 3 (SPCR3) is 0.

Figure 30.46 shows an example of an SSLni signal operation for a burst transfer that is implemented using the SPCMD0 and SPCMD1 register settings. This section describes SPI operations (1) to (8) shown in Figure 30.46.

Note: The polarity of the SSLni output signal depends on the SSLP register settings.

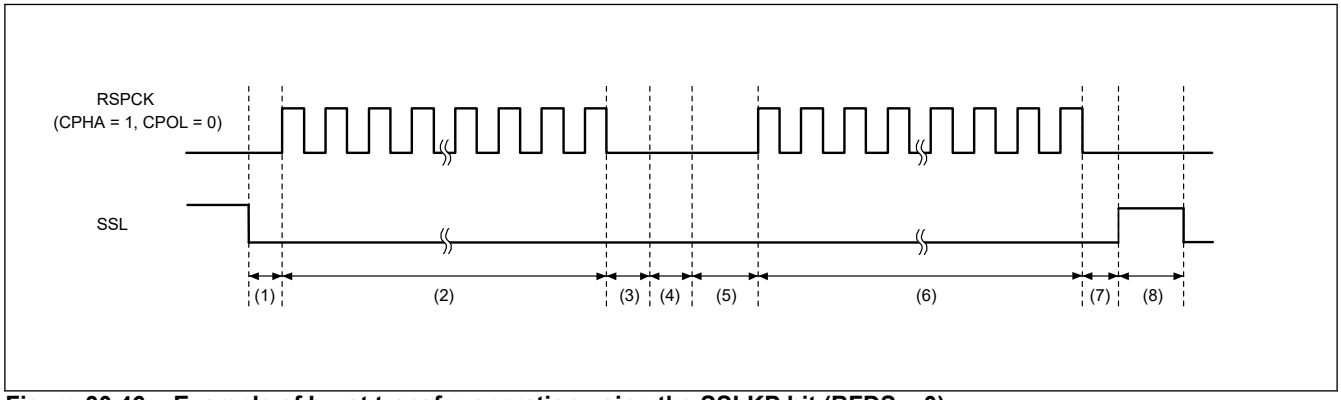



**Figure 30.46 Example of burst transfer operation using the SSLKP bit (BFDS = 0)**

The SPI operation at times (1) to (8) in the figure is as follows:

1. Based on the SPCMD0 settings, the SPI asserts the SSLni signal and inserts RSPCK delays.
2. The SPI executes serial transfers in accordance with the SPCMD0 settings.
3. The SPI inserts an SSL negation delay.
4. Because the SPCMD0.SSLKP bit is 1, the SPI keeps the SSLni signal value specified in SPCMD0. This period is sustained at a minimum for a period equal to the next-access delay in SPCMD0. If the shift register is empty after the passage of the minimum period, this period is sustained until the transmit data is stored in the shift register for the next transfer.
5. Based on the SPCMD1 settings, the SPI asserts the SSLni signal and inserts RSPCK delays.
6. The SPI executes serial transfers in accordance with the SPCMD1 settings.
7. Insert SSL negate delay.
8. Because the SPCMD1.SSLKP bit is 0, the SPI negates the SSLni signal. In addition, a next-access delay is inserted in accordance with SPCMD1.

If the SSLni signal output settings in the SPCMDm register where 1 is assigned to the SSLKP bit are different from the SSLni signal output settings in the SPCMDm register to be used in the next transfer, the SPI switches the SSLni signal status to SSLni signal assertion as shown in (5) in Figure 30.46. This corresponds to the command for the next transfer.

Note: If such an SSLni signal switching occurs, the slaves that drive the MISOn signal compete, and collision of signal levels might occur.

The SPI in master mode references the SSLni signal operation within the module when the SSLKP bit is not used. When the SPCMDm.CPHA bit is 0, the SPI can accurately start serial transfers by using the SSLni signal assertion for the next transfer that is detected internally.

- When Between Burst Transfer Frames Delay Select bit (BFDS) of SPI control register 3 (SPCR3) is 1.

Figure 30.47 shows an example of SSL signal operation when burst transfer is achieved by using the settings of SPCMD0 and SPCMD1. The following describes SPI operations of (1) to (6) shown in Figure 30.47. The SSL output signal polarity depends on the set SPI slave select polarity register (SSLP) value.

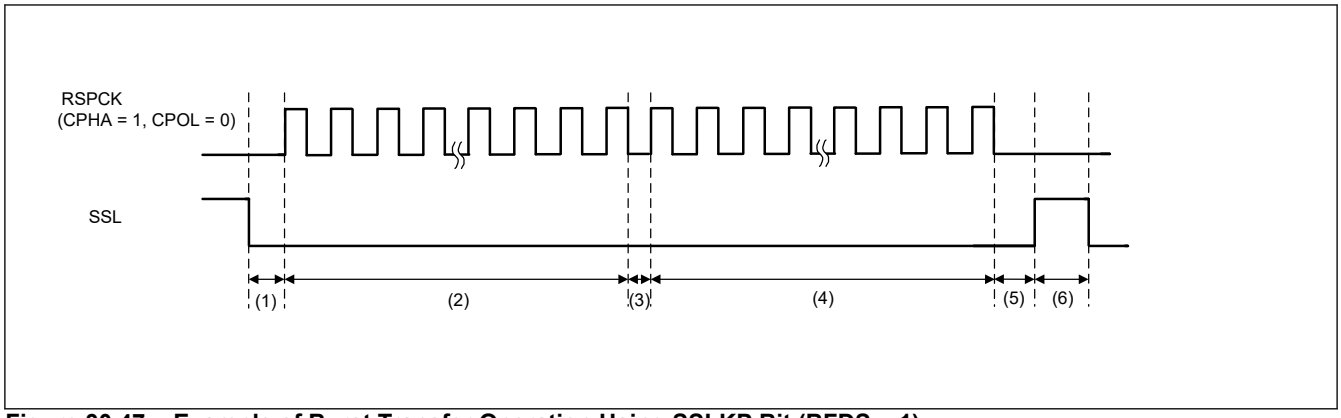



**Figure 30.47 Example of Burst Transfer Operation Using SSLKP Bit (BFDS = 1)**

1. Assert the SSL signal and insert an RSPCK delay according to SPCMD0. The RSPCK delay is inserted only the first frame of burst transmission.
2. Perform serial transfer according to SPCMD0. Wait last clock until the next transmit data is stored in the shift register, if the shift register is empty during RSPCK negate period between frames.
3. The value of SSL signal according to SPCMD0 was hold, because the SPCMD0.SSLKP bit is 1. RSPCK negate period between frames is 0.5RSPCK, if the shift register is not empty.
4. Perform serial transfer according to SPCMD1.
5. Insert SSL negate delay for the last frame.
6. The SSL signal is negated because the SSLKP bit in SPCMD1 is 0. Furthermore, the next-access delay is inserted according to SPCMD1.

### (5) RSPCK delay (t1)

The RSPCK delay value of the SPI in master mode depends on the SPCMDm.SCKDEN bit setting and the SPCKD.SCKDL[2:0] bits setting. The SPI determines the SPCMDm register to be referenced during a serial transfer by pointer control, and determines an RSPCK delay using the SPCMDm.SCKDEN bit and SPCKD.SCKDL[2:0] bits, as listed in Table 30.11. For a definition of RSPCK delay, see section 30.3.5. Transfer Formats.

RSPCK delay insert to only the first frame of burst transmission, when transmit without “Between Burst Transfer Frames Delay”. (The SPCMD.SSLKP bit is 1 and the SPCR3.BFDS bit is 1.)

**Table 30.11 Relationship between the SPCMDm.SCKDEN bit, SPCKD.SCKDL[2:0] bits, and RSPCK delay**

| SPCMDm.SCKDEN bit | SPCKD.SCKDL[2:0] bits | RSPCK delay |
|---|---|---|
| 0 | 000b to 111b | 1 RSPCK |
| 1 | 000b | 1 RSPCK |
| | 001b | 2 RSPCK |
| | 010b | 3 RSPCK |
| | 011b | 4 RSPCK |
| | 100b | 5 RSPCK |
| | 101b | 6 RSPCK |
| | 110b | 7 RSPCK |
| | 111b | 8 RSPCK |

### (6) SSL negation delay (t2)

The SSL negation delay value of the SPI in master mode depends on the SPCMDm.SLNDEN bit setting and the SSLND.SLNDL[2:0] bits setting. The SPI determines the SPCMDm register to be referenced by pointer control during a serial transfer, and determines an SSL negation delay using the SPCMDm.SLNDEN bit and SSLND.SLNDL[2:0] bits, as listed in Table 30.12. For a definition of SSL negation delay, see section 30.3.5. Transfer Formats.

An SSL negation delay is inserted to only the last frame of the burst transmission, that is, transmit without “between burst transfer frames delay”. (SPCMD.SSLKP bit is 1 and SPCR3.BFDS bit is 1).

**Table 30.12 Relationship between the SPCMDm.SLNDEN bit, SSLND.SLNDL[2:0] bits, and SSL negation delay**

| SPCMDm.SLNDEN bit | SSLND.SLNDL[2:0] bits | SSL negation delay |
|---|---|---|
| 0 | 000b to 111b | 1 RSPCK |
| 1 | 000b | 1 RSPCK |
| | 001b | 2 RSPCK |
| | 010b | 3 RSPCK |
| | 011b | 4 RSPCK |
| | 100b | 5 RSPCK |
| | 101b | 6 RSPCK |
| | 110b | 7 RSPCK |
| | 111b | 8 RSPCK |

### (7) Next-access delay (t3)

The next-access delay value of the SPI in master mode depends on the SPCMDm.SPNDEN bit setting and the SPND.SPNDL[2:0] bits setting. The SPI determines the SPCMDm register to be referenced during serial transfer by pointer control, and then determines a next-access delay during serial transfer using the SPCMDm.SPNDEN bit and SPND.SPNDL[2:0] bits, as listed in Table 30.13. For a definition of next-access delay, see section 30.3.5. Transfer Formats.

A next-Access delay is inserted to only the last frame of the burst transmission, that is, transmit without “between burst transfer frames delay”. (SPCMD.SSLKP bit is 1 and SPCR3.BFDS bit is 1).

**Table 30.13 Relationship between the SPCMDm.SPNDEN bit, SPND.SPNDL[2:0] bits, and next-access delay**

| SPCMDm.SPNDEN bit | SPND.SPNDL[2:0] bits | Next-access delay |
|---|---|---|
| 0 | 000b to 111b | 1 RSPCK + 2 PCLKA |
| 1 | 000b | 1 RSPCK + 2 PCLKA |
| | 001b | 2 RSPCK + 2 PCLKA |
| | 010b | 3 RSPCK + 2 PCLKA |
| | 011b | 4 RSPCK + 2 PCLKA |
| | 100b | 5 RSPCK + 2 PCLKA |
| | 101b | 6 RSPCK + 2 PCLKA |
| | 110b | 7 RSPCK + 2 PCLKA |
| | 111b | 8 RSPCK + 2 PCLKA |

### (8) Initialization flow

Figure 30.48 shows an example of SPI initialization flow when the SPI is in master mode. For information on how to set up the Interrupt Controller Unit (ICU), DMAC and I/O ports, see the descriptions given in the individual blocks.

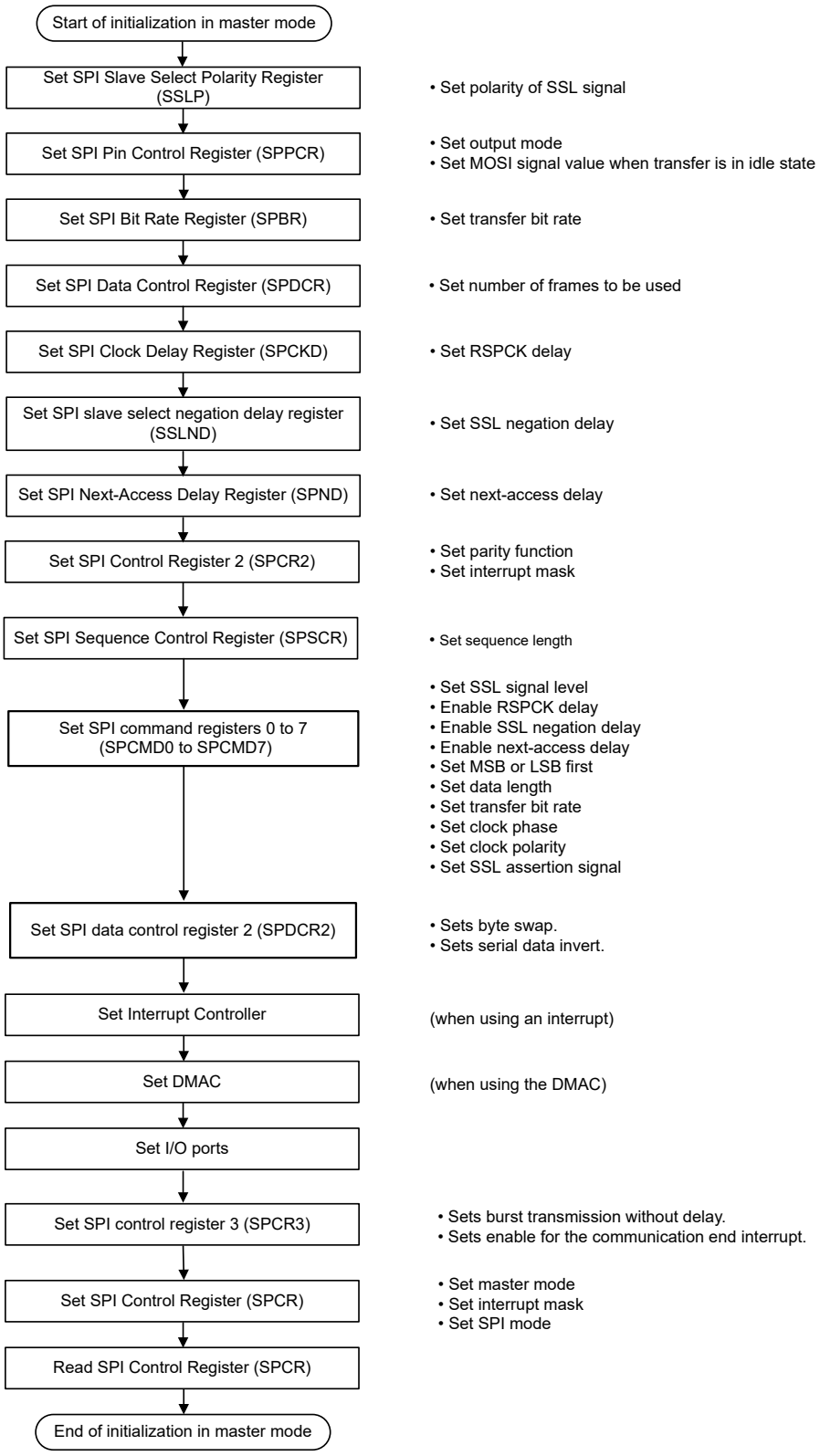


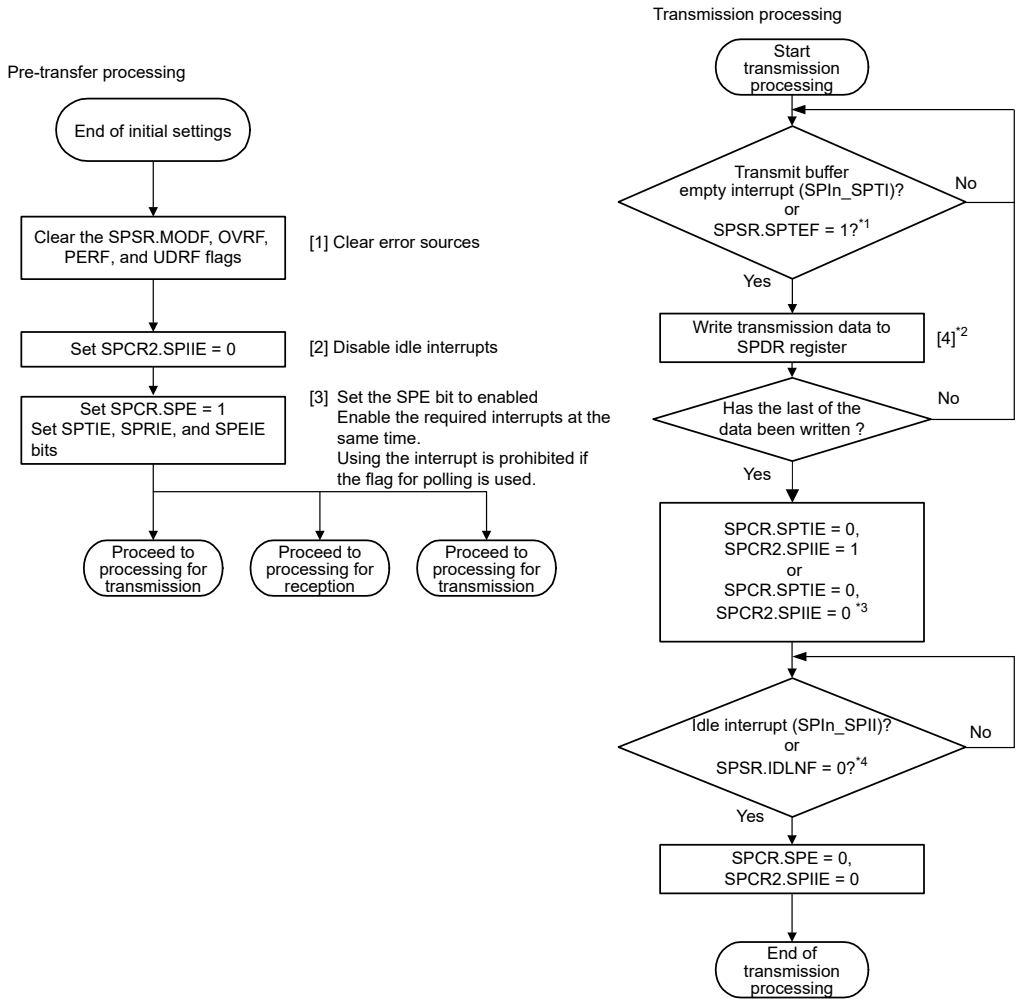

**Figure 30.48 Example of initialization flow in master mode for SPI operation**

## (9) Software processing flow

Figure 30.49 to Figure 30.51 show examples of the software processing flow.

### Transmit processing flow

When transmitting data, with the SPIi_SPII interrupt enabled, the CPU is notified of the completion of data transmission after the last data write for transmission.

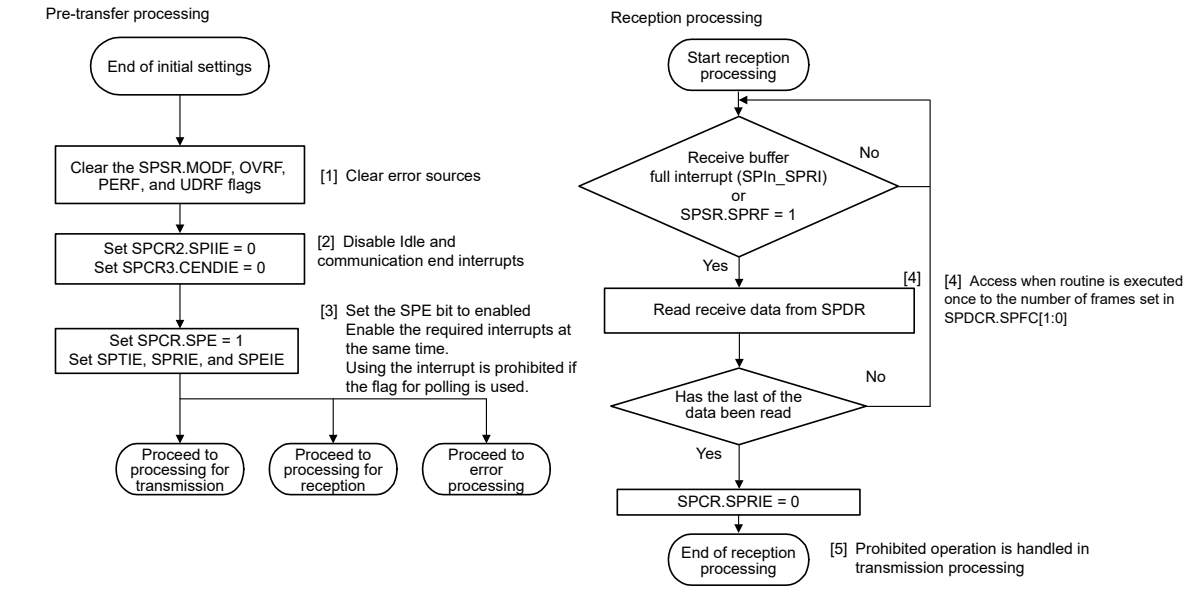



Note 1. Before writing data for transmission to SPDR/SPDR_HA/SPDR_BY, check that the transmit buffer is empty by reading the SPSR.SPTEF flag, if the flag for polling is used.
Note 2. Setting SPCR2.SPIIE and SPCR3.CENDIE to 0 is prohibited if the flag for polling is used.
Note 3. Wait more than 1 PCLKA, after writing data for transmission to SPDR register, and before starting to poll SPSR.IDLNF flag or SPSR.CENDF, if the flag for polling is used.
Note 4. SPDR = SPDR/SPDR_HA/SPDR_BY
Access when interrupt handling routine is executed once to the number of frames set in SPDCR.SPFC[1:0]

**Figure 30.49 Transmission flow in master mode**

### Receive processing flow

The SPI has receive only operation in slave mode.




Note 1. SPDR = SPDR/SPDR_HA/SPDR_BY

**Figure 30.50 Reception flow in master mode**

### Error processing flow

The SPI detects the following errors:

- Mode fault error
- Underrun error
- Overrun error
- Parity error

When a mode fault error is generated, the SPCR.SPE bit is automatically cleared, stopping operations for transmission and reception. For errors from other sources, the SPCR.SPE bit is not cleared and operations for transmission and reception continue. Therefore, Renesas recommends clearing the SPCR.SPE bit to stop operations for errors other than mode fault errors. Not doing so leads to updating of the SPSSR.SPECM[2:0] bits.

When an error is detected using an interrupt, clear the ICU.IELSRn.IR flag in the error processing routine. If this is not done, the ICU.IELSRn.IR flag might continue to indicate the SPIi_SPTI or SPIi_SPRI interrupt request. If the SPIi_SPRI interrupt request is indicated, read the receive buffer and initialize the sequencer in the SPI.

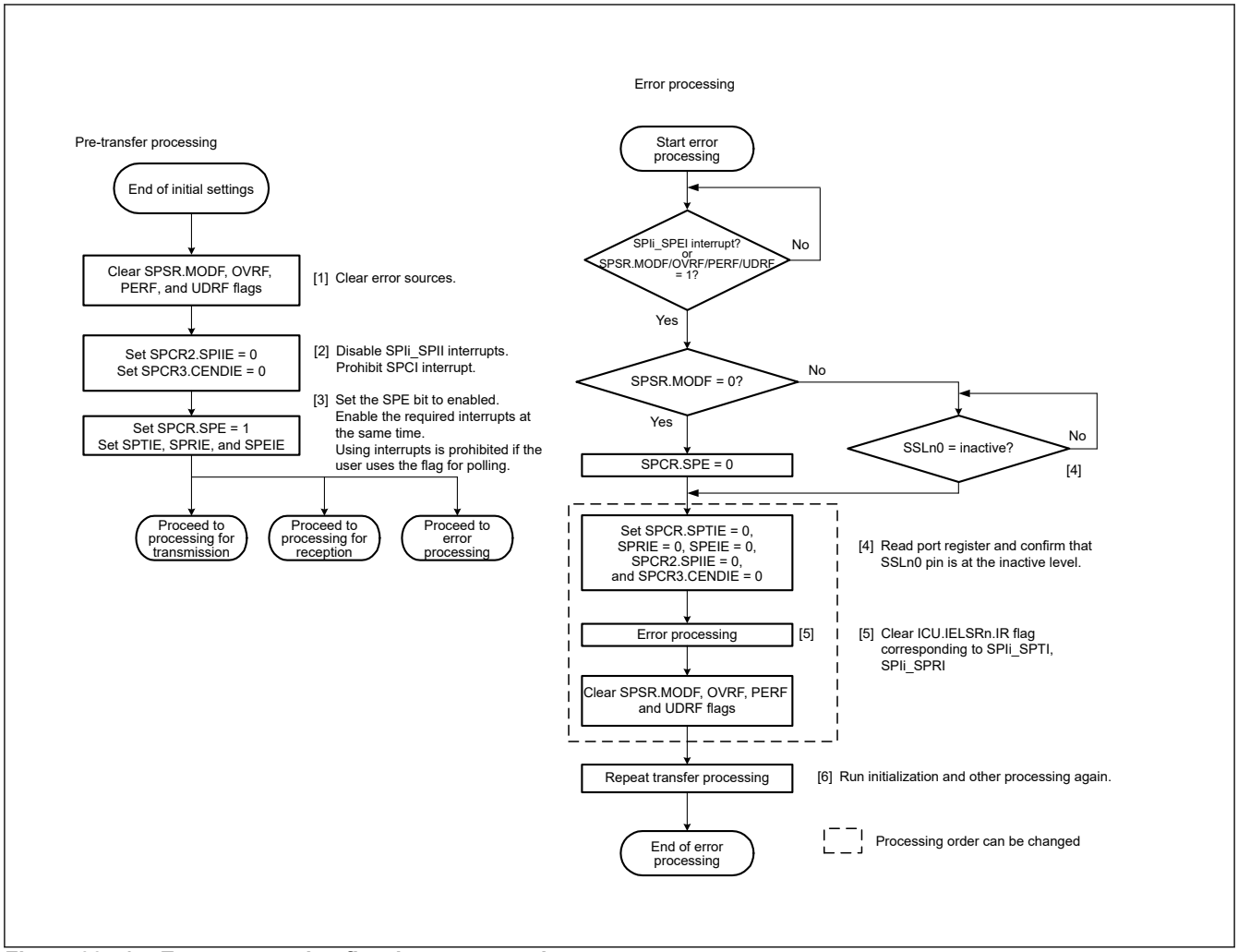



**Figure 30.51 Error processing flow in master mode**

### 30.3.11.2 Slave mode operation

#### (1) Starting a serial transfer

When the SPCMD0.CPHA bit is 0, if the SPI detects an SSLn0 input signal assertion, it must drive valid data to the MISOn output signal. For this reason, when the CPHA bit is 0, the assertion of the SSLn0 input signal triggers the start of a serial transfer.

When the CPHA bit is 1, if the SPI detects the first RSPCKn edge in an SSLn0 signal asserted condition, it must drive valid data to the MISOn output signal. For this reason, when the CPHA bit is 1, the first RSPCKn edge in an SSLn0 signal asserted condition triggers the start of a serial transfer.

Regardless of the CPHA bit setting, the SPI drives the MISOn output signal on SSLn0 signal assertion. The data that is output by the SPI is either valid or invalid, depending on the CPHA bit setting.

For details on the SPI transfer format, see section 30.3.5. Transfer Formats. The polarity of the SSLn0 input signal depends on the SSLP.SSL0P setting.

#### (2) Terminating a serial transfer

Regardless of the SPCMD0.CPHA bit setting, the SPI terminates the serial transfer after detecting an RSPCKn edge corresponding to the final sampling timing. When free space is available in the receive buffer (the SPSR.SPRF flag is 0), on termination of serial transfer, the SPI copies received data from the shift register to the receive buffer of the SPDR/ SPDR_HA register. On termination of a serial transfer, the SPI changes the status of the shift register to empty, regardless of the receive buffer state. A mode fault error occurs if the SPI detects an SSLn0 input signal negation from the beginning of serial transfer to the end of serial transfer (see section 30.3.9. Error Detection).

The final sampling timing changes depending on the bit length of transfer data. In slave mode, the SPI data length is determined by the SPCMD0.SPB[3:0] bits setting. The polarity of the SSLn0 input signal is determined by the SSLP.SSL0P bit setting. For details on the SPI transfer format, see section 30.3.5. Transfer Formats.

### (3) Notes on single-slave operations

If the SPCMD0.CPHA bit is 0, the SPI starts serial transfers when it detects the assertion edge for an SSLn0 input signal. In the configuration shown in Figure 30.7, if the SPI is used in single-slave mode, the SSLn0 signal is fixed at an active state. Therefore, when the CPHA bit is set to 0, the SPI cannot correctly start a serial transfer. For the SPI to correctly execute transmit and receive operations in slave mode when the SSLn0 input signal is fixed at an active state, the CPHA bit must be set to 1. Do not fix the SSLn0 input signal if there is a requirement for setting the CPHA bit to 0.

### (4) Burst transfer

If the SPCMD0.CPHA bit is 1, continuous serial transfer (burst transfer) can be executed while retaining the assertion state for the SSLn0 input signal. When the CPHA bit is 1, the serial transfer period is the period from the first RSPCKn edge to the sampling timing for the reception of the final bit in an SSLn0 signal active state. Even when the SSLn0 input signal remains at the active level, the SPI can accommodate burst transfers, because it can detect the start of an access.

When the CPHA bit is 0, the second and subsequent serial transfers during burst transfer cannot be executed correctly.

### (5) Initialization flow

Figure 30.52 shows an example of initialization flow for SPI operation when the SPI is in slave mode. For a description of how to set up the ICU, DTC, and I/O ports, see the descriptions given in the individual blocks.

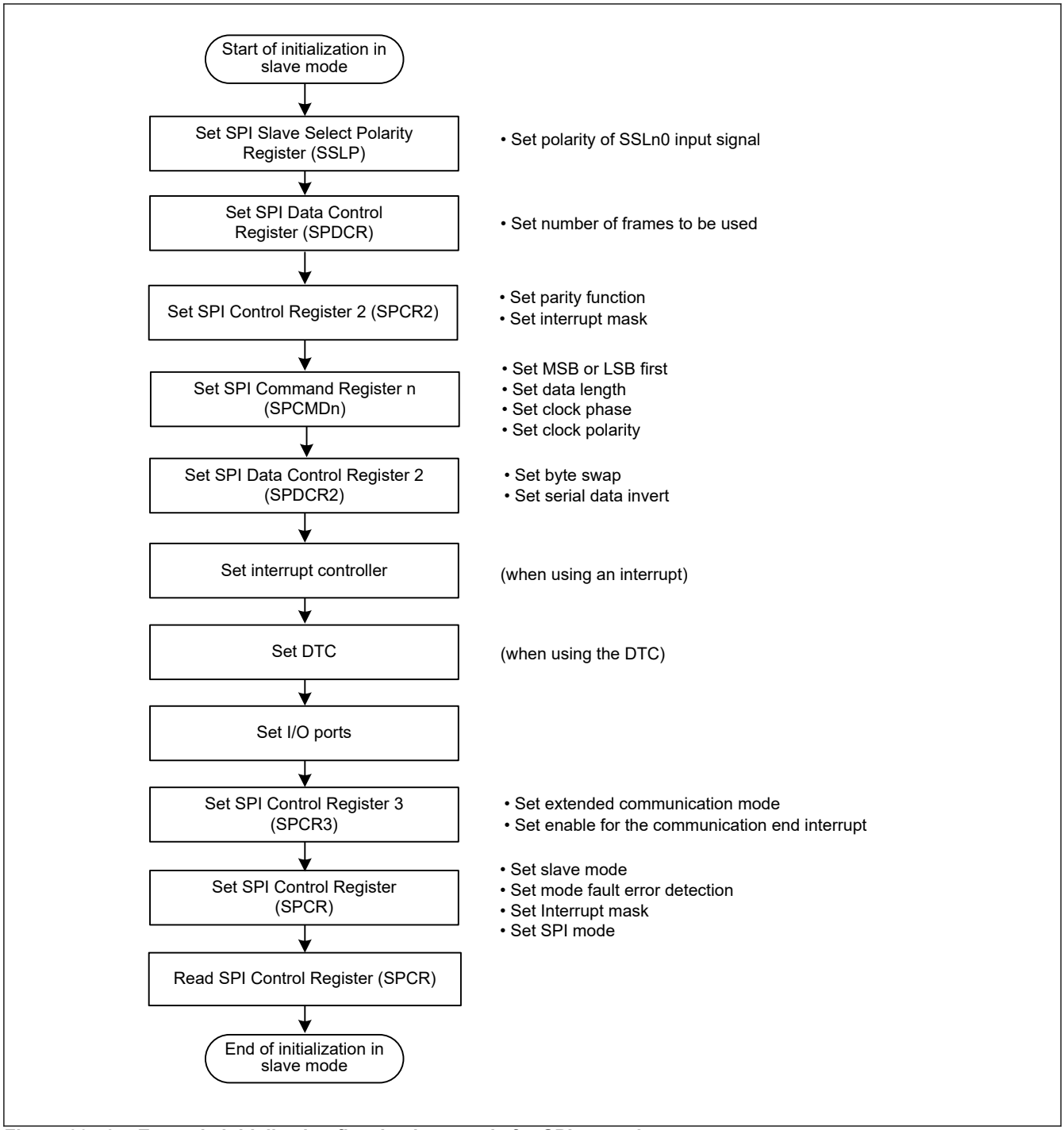
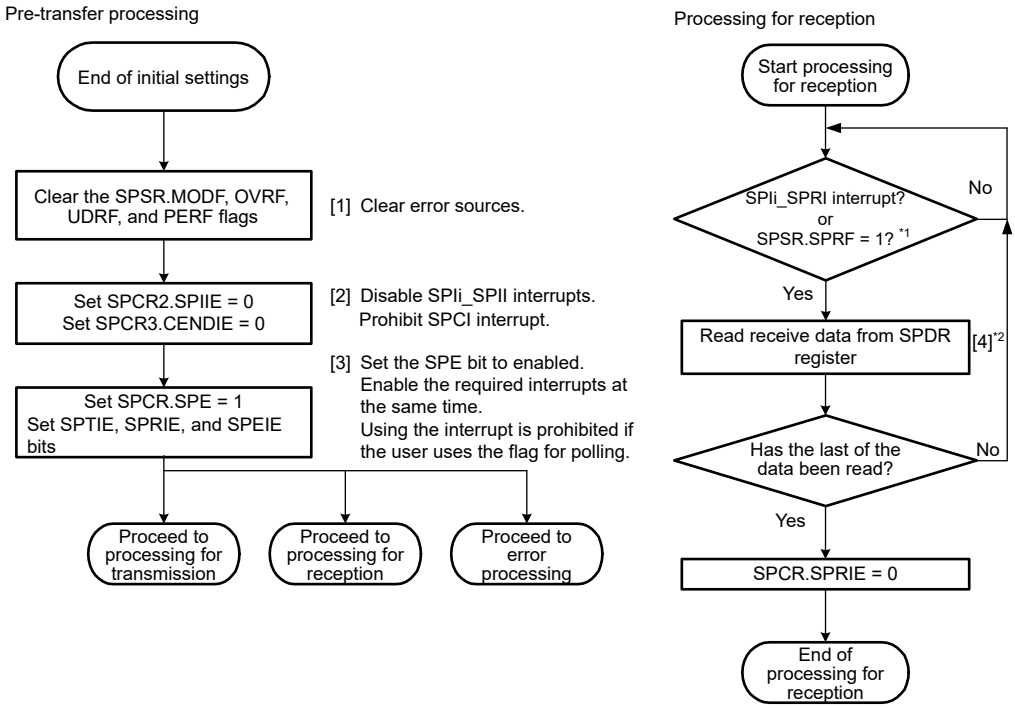



**Figure 30.52 Example initialization flow in slave mode for SPI operation**

## (6) Software processing flow

Figure 30.53 to Figure 30.55 show examples of the flow of software processing.

### Transmit processing flow

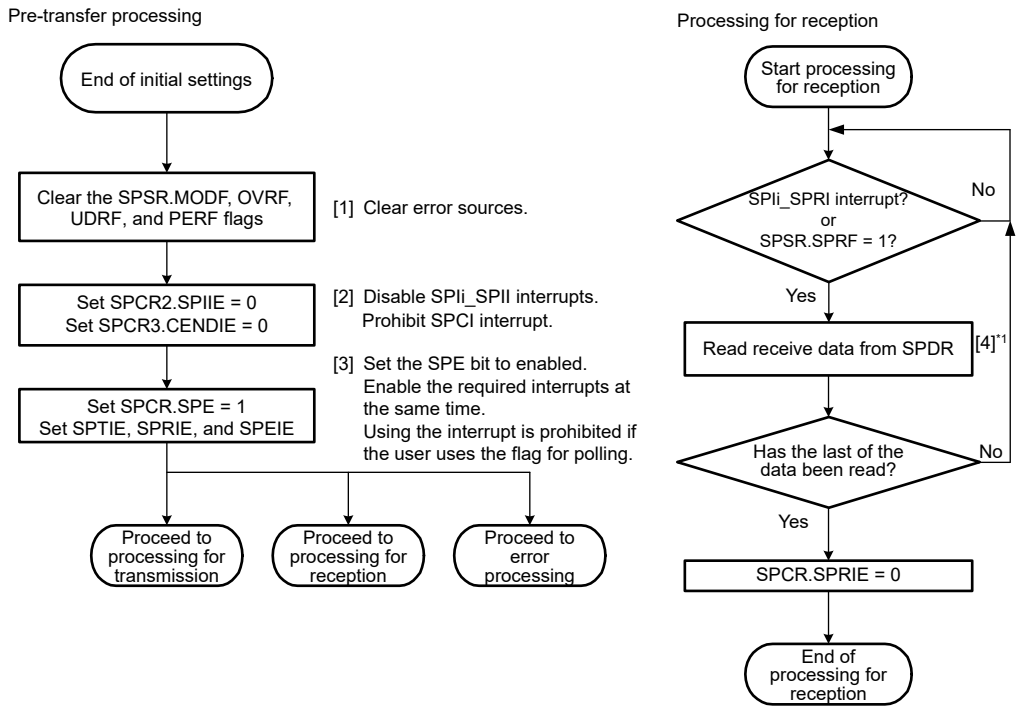



Note 1. Before writing data for transmission to SPDR register, check the transmit buffer empty interrupt by reading SPSR.SPTEF flag, if the flag for polling is used.
Note 2. SPDR = SPDR/SPDR_HA/SPDR_BY
Access when interrupt handling routine is executed once to the number of frames set in SPDCR.SPFC[1:0] bits

**Figure 30.53 Transmission flow in slave mode**

### Receive processing flow

The SPI does not handle receive-only operation, so processing for transmission is required.




Note 1. Before writing data for transmission to the SPDR register, check that the transmit buffer is empty by reading the SPSR.SPTEF flag, if the flag for polling is used.
Note 2. SPDR = SPDR/SPDR_HA/SPDR_BY
Access when interrupt handling routine is executed once to the number of frames set in SPDCR.SPFC[1:0] bits

**Figure 30.54 Reception flow in slave mode**

### Error processing flow

In slave mode operation, even when a mode fault error is generated, the SPSR.MODF flag can be cleared regardless of the state of the SSLn0 pin.

When an error is detected by using an interrupt, clear the ICU.IELSRn.IR flag in the error processing routine. If this is not done, the ICU.IELSRn.IR flag might continue to indicate the SPIi_SPTI or SPIi_SPRI interrupt request. If the SPIi_SPRI interrupt request is indicated, read the receive buffer and initialize the sequencer in the SPI.

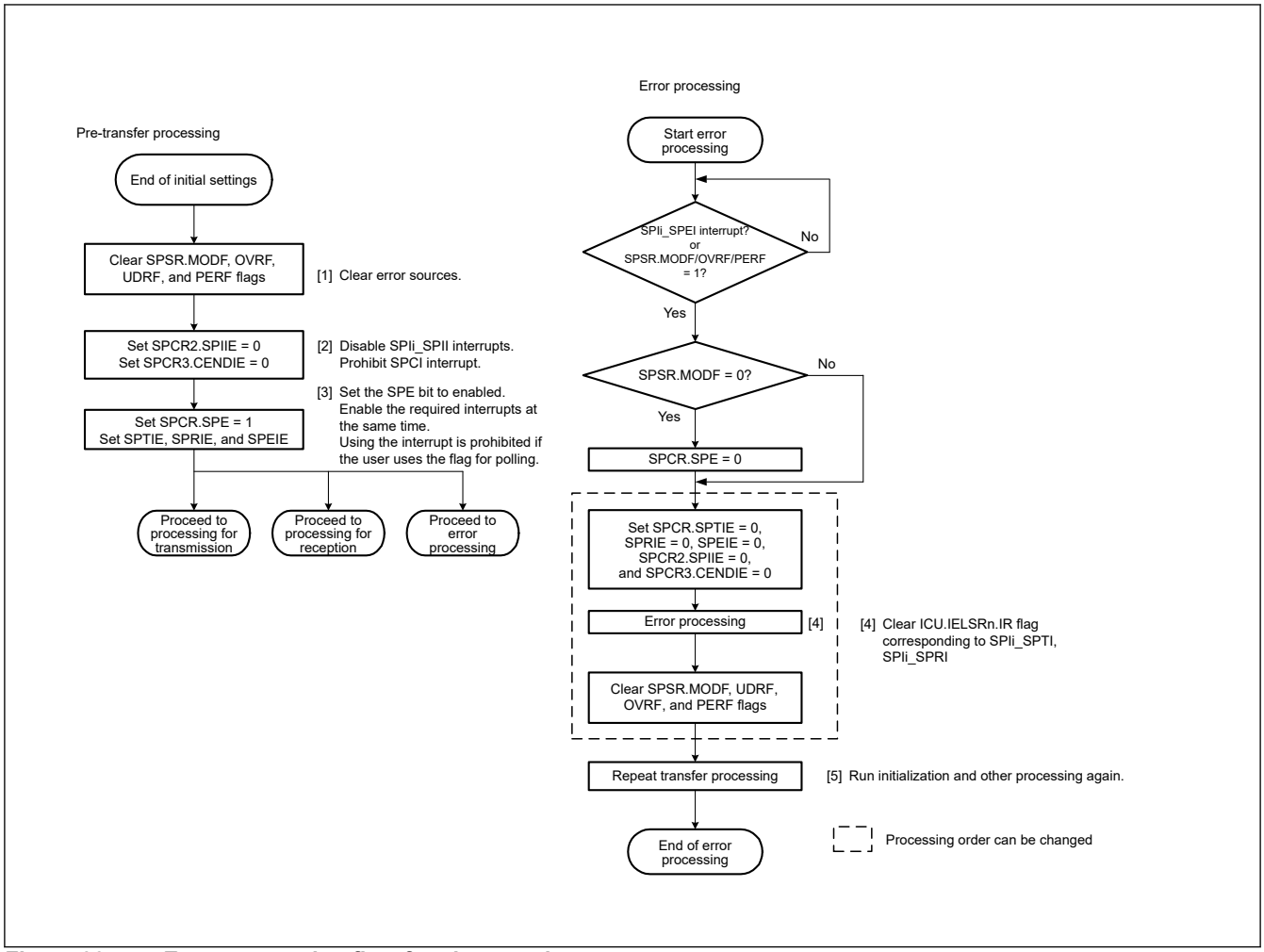



**Figure 30.55 Error processing flow for slave mode**

## 30.3.12 Clock Synchronous Operation

Setting the SPCR.SPMS bit to 1 selects clock synchronous operation of the SPI. In clock synchronous operation, the SSLni pin is not used, and the RSPCKn, MOSIn, and MISOn pins handle communications. All SSLni pins are available as I/O port pins.

Although clock synchronous operation does not require the use of the SSLni pin, operation of the module is the same as in SPI operation. In both master mode and slave mode operations, communications can be performed with the same flow as in SPI operation. However, mode fault errors are not detected, because the SSLni pin is not used.

Additionally, do not perform operation if clock synchronous operation is enabled when the SPCMDm.CPHA bit is set to 0 in slave mode (SPCR.MSTR = 0).

### 30.3.12.1 Master mode operation

#### (1) Starting serial transfer

The SPI updates the data in the transmit buffer (SPTX) of SPDR/SPDR_HA when data is written to the SPDR/SPDR_HA register with the transmit buffer empty, the data for the next transfer not set and the SPSR.SPTEF flag is 1. When the shift register is empty after the number of frames set in the SPDCR.SPFC[1:0] bits are written to the SPDR/SPDR_HA, the SPI copies data from the transmit buffer to the shift register and starts serial transmission. On copying transmit data to the shift register, the SPI changes the status of the shift register to full, and on termination of serial transfer, it changes the status of the shift register to empty. The status of the shift register cannot be referenced.

Transfer in clock synchronous operation is conducted without the SSLn0 output signal. For details on the SPI transfer format, see section 30.3.5. Transfer Formats.

### (2) Terminating serial transfer

The SPI terminates the serial transfer after transmitting an RSPCKn edge corresponding to the sampling timing. If free space is available in the receive buffer (the SPSR.SPRF flag is 0), on termination of serial transfer, the SPI copies data from the shift register to the receive buffer of the SPI Data Register (SPDR/SPDR_HA).

The final sampling timing varies depending on the bit length of transfer data. In master mode, the SPI data length depends on the SPCMDm.SPB[3:0] bits setting. Transfer in clock synchronous operation is conducted without the SSLn0 output signal. For details on the SPI transfer format, see section 30.3.5. Transfer Formats.

### (3) Sequence control

The transfer format used in master mode is determined by the SPSCR, SPCMDm, SPBR, SPCKD, SSLND, and SPND registers. Although the SSLni signals are not output in clock synchronous operation, these settings are valid.

The SPSCR register determines the sequence configuration for serial transfers that are executed by the SPI in master mode. The following parameters are specified in the SPCMDm register:

- SSLni output signal value
- MSB or LSB first
- Data length
- Some of the bit rate settings
- RSPCKn polarity and phase
- Whether SPCKD is to be referenced
- Whether SSLND is to be referenced
- Whether SPND is to be referenced

SPBR holds some of the bit rate settings such as SPCKD, an SPI clock delay value, SSLND, an SSL negation delay, and SPND, a next-access delay value.

Based on the sequence length that is assigned to SPSCR, the SPI makes up a sequence comprised of a part or all of SPCMDm register. The SPI contains a pointer to the SPCMDm register that makes up the sequence. The value of this pointer can be checked by reading the SPSSR.SPCP[2:0] bits. When the SPCR.SPE bit is set to 1 and the SPI function is enabled, the SPI loads the pointer to the commands in SPCMD0 register, and incorporates the SPCMD0 register setting into the transfer format at the beginning of serial transfer. The SPI increments the pointer each time the next-access delay period for a data transfer ends. On completion of the serial transfer that corresponds to the final command comprising the sequence, the SPI sets the pointer to the SPCMD0 register, and in this manner the sequence is executed repeatedly.

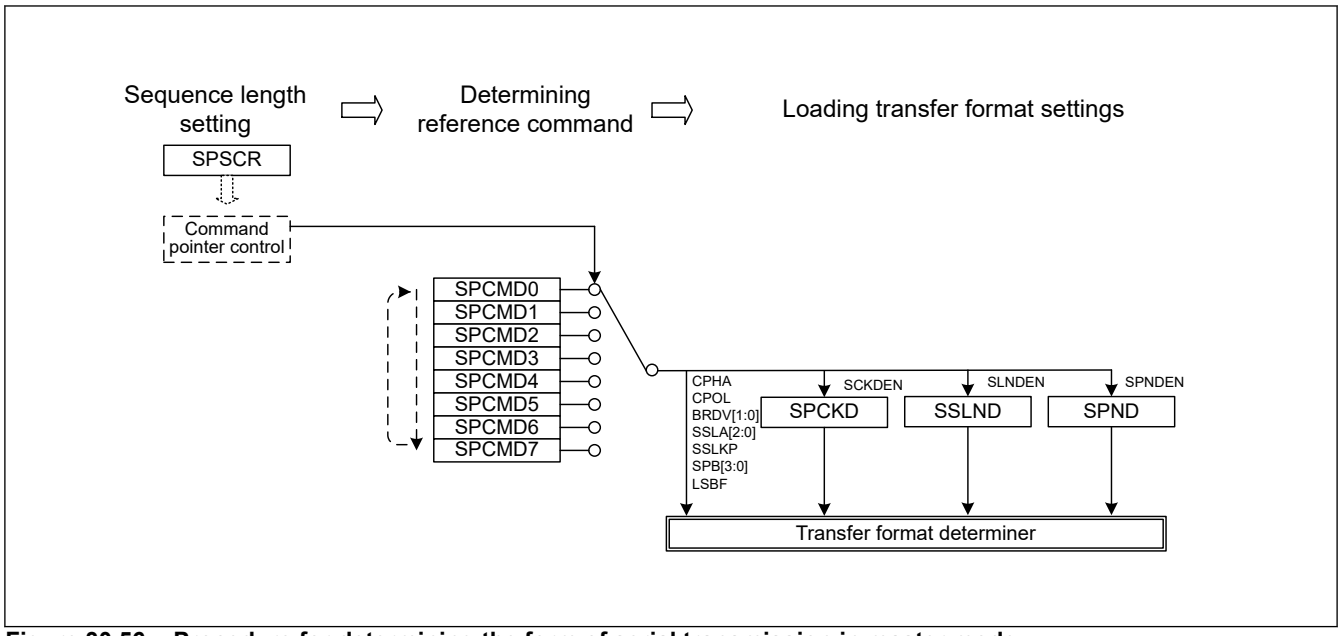



**Figure 30.56 Procedure for determining the form of serial transmission in master mode**

In this section, a frame is the combination of the data (SPDR/SPDR_HA) and the settings (SPCMDm).

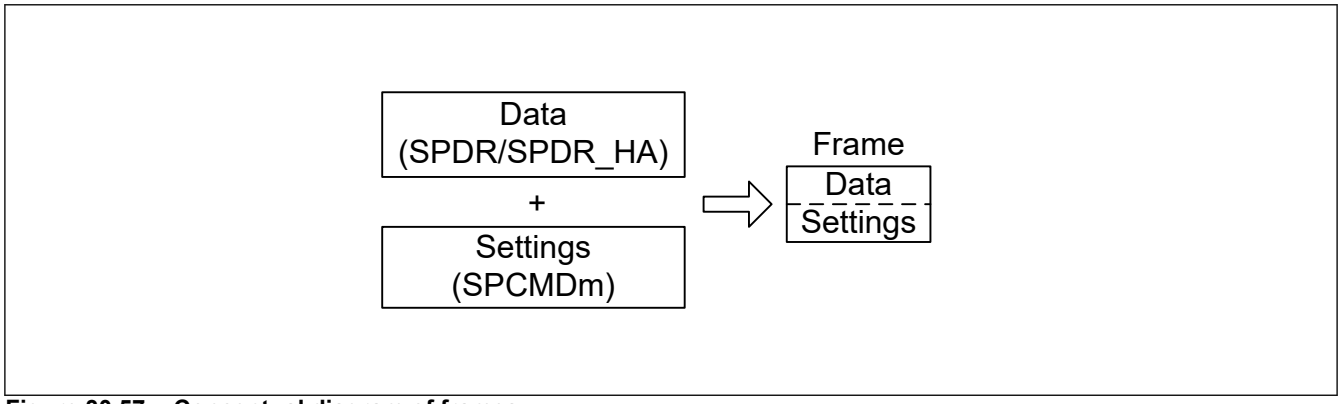



**Figure 30.57 Conceptual diagram of frames**

Figure 30.58 shows the relationship between the command and the transmit and receive buffers in the sequence of operations specified by the settings in Table 30.4.

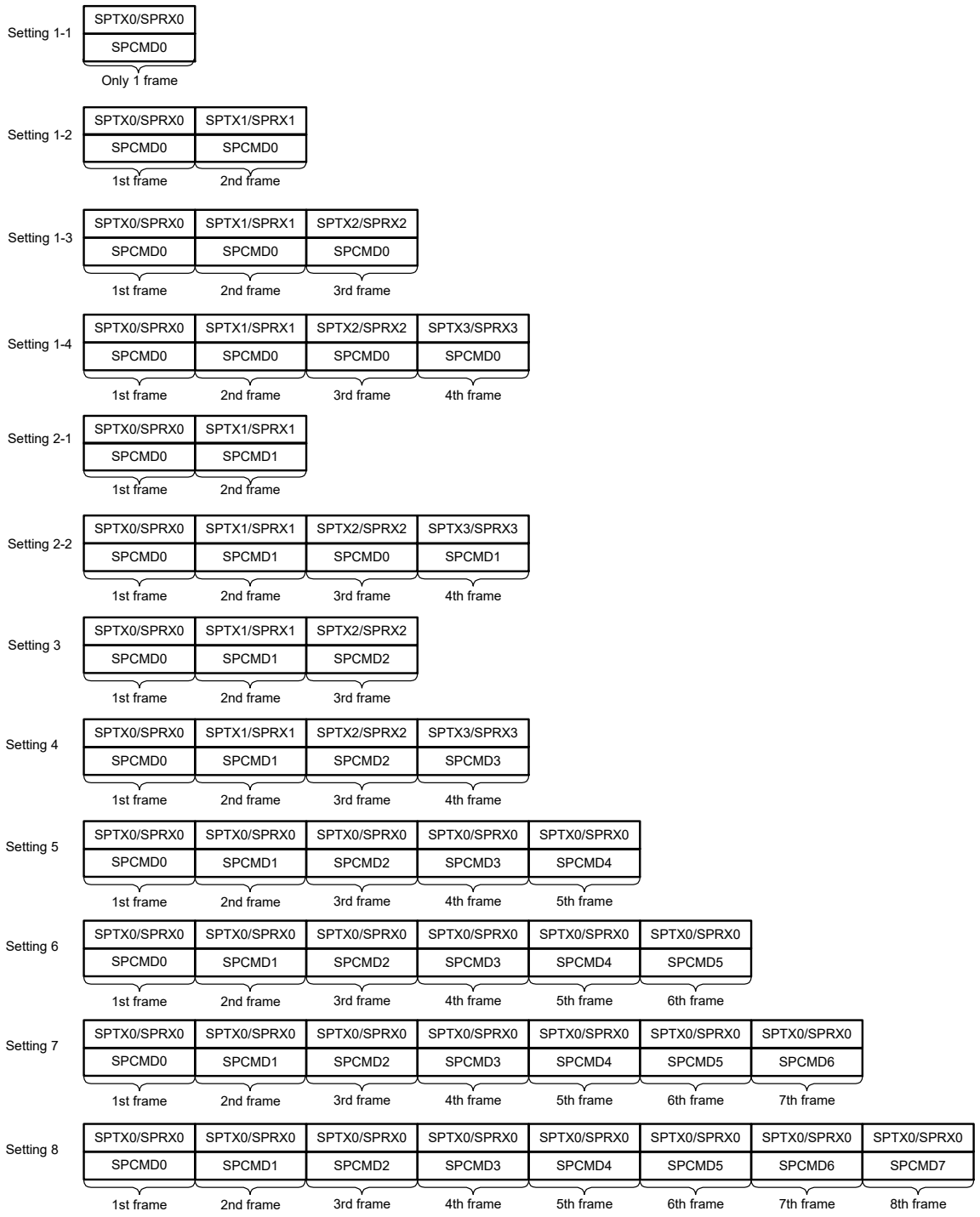



**Figure 30.58 Correspondence between SPI Command Register and transmit and receive buffers in sequence operations**

## (4) Initialization flow

Figure 30.59 shows an example of initialization flow for clock synchronous operation when the SPI is used in master mode. For information on how to set up the ICU, DMAC or DTC, and I/O ports, see the descriptions given in the individual blocks.

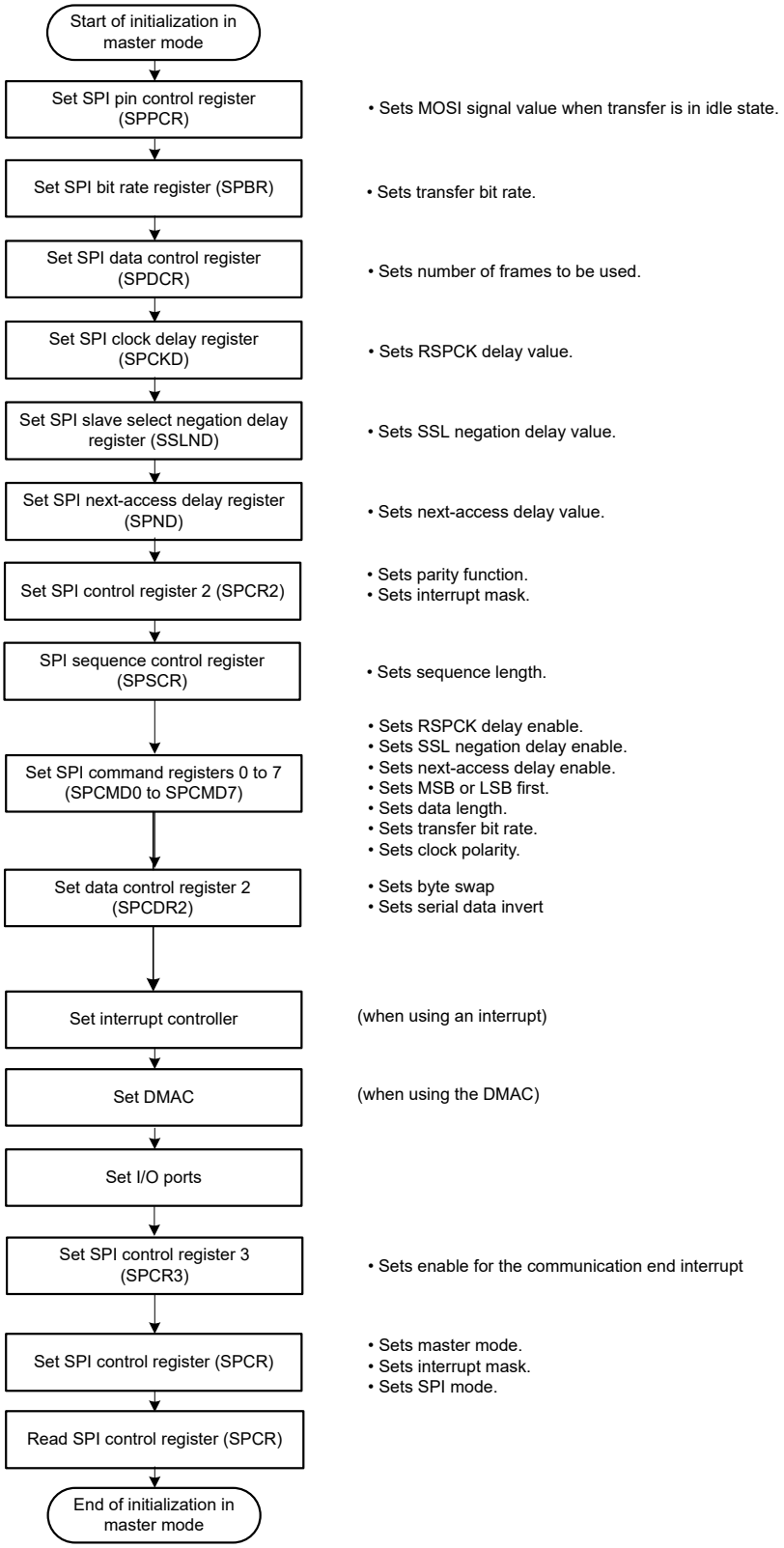



**Figure 30.59 Example of initialization flow in master mode for clock synchronous operation**

#### (5) Software processing flow

Software processing during clock synchronous master operation is the same as that for SPI master operation. For details, see (9) Software processing flow in section 30.3.11.1. Master mode operation. Mode fault errors do not occur in clock synchronous operation.

### 30.3.12.2 Slave mode operation

#### (1) Starting serial transfer

When the SPCR.SPMS bit is 1, the first RSPCKn edge triggers the start of a serial transfer in the SPI, and the SPI drives the MISOn output signal. The SSL0 input signal is not used in clock synchronous operation. For details on the SPI transfer format, see section 30.3.5. Transfer Formats.

#### (2) Terminating serial transfer

The SPI terminates the serial transfer after detecting an RSPCKn edge corresponding to the final sampling timing. When free space is available in the receive buffer (the SPSR.SPRF flag is 0), on termination of serial transfer, the SPI copies received data from the shift register to the receive buffer of the SPDR/SPDR_HA register. On termination of a serial transfer, the SPI changes the status of the shift register to empty regardless of the receive buffer.

The final sampling timing changes depending on the bit length of transfer data. In slave mode, the SPI data length depends on the SPCMD0.SPB[3:0] bits setting. For details on the SPI transfer format, see section 30.3.5. Transfer Formats.

#### (3) Initialization flow

Figure 30.60 shows an example of initialization flow for clock synchronous operation when the SPI is used in slave mode. For a description of how to set up the ICU, DTC, and I/O ports, see the descriptions given in the individual blocks.

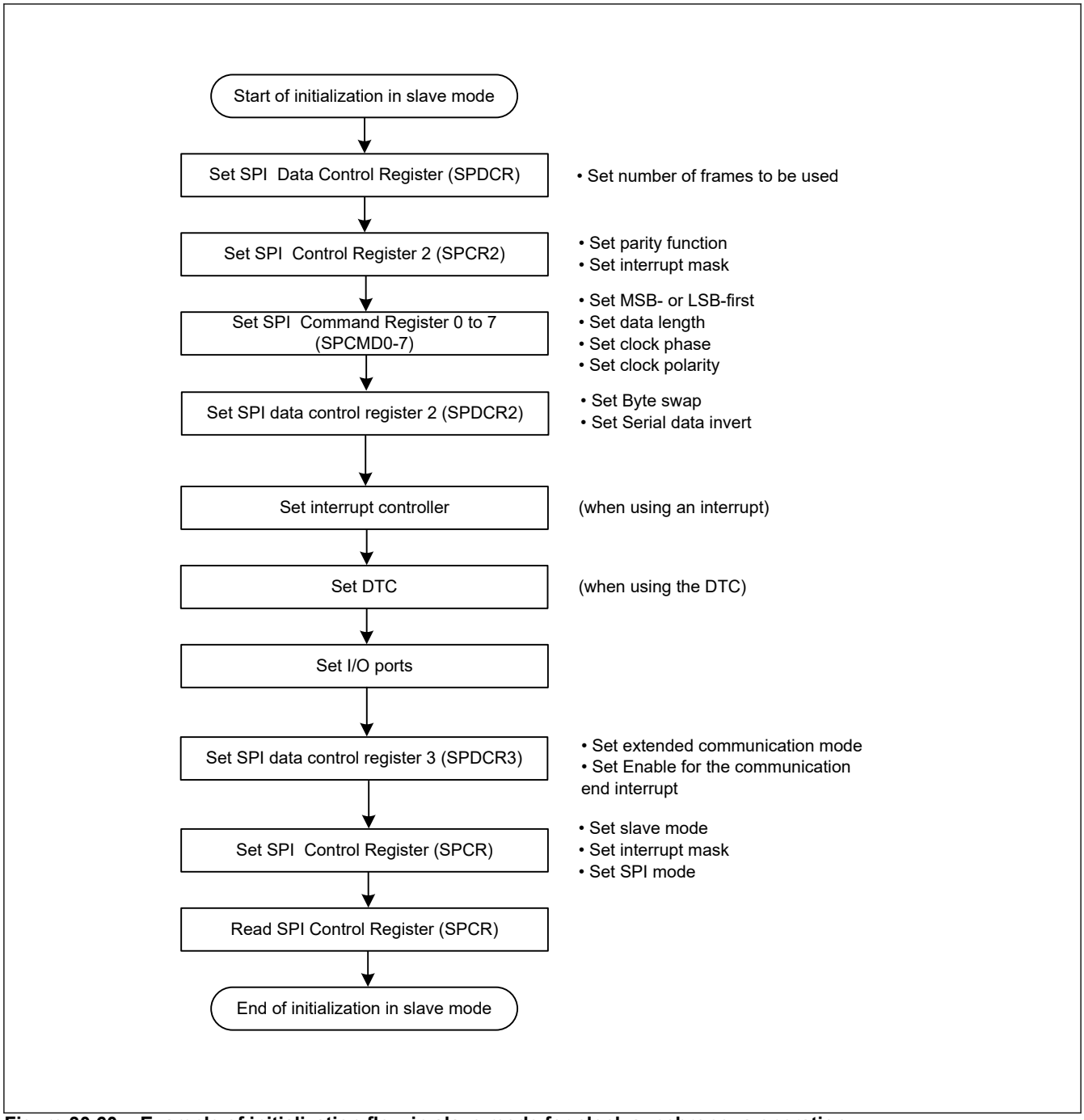



**Figure 30.60 Example of initialization flow in slave mode for clock synchronous operation**

### (4) Software processing flow

Software processing during clock synchronous slave operation is the same as that for SPI slave operation. For details, see (6)Software processing flow. Mode fault errors do not occur in clock synchronous mode.

## 30.3.13 Loopback Mode

When 1 is written to the SPPCR.SPLP2 bit or SPPCR.SPLP bit, the SPI shuts off the path between the MISOn pin and the shift register if the SPCR.MSTR bit is 1, or between the MOSIn pin and the shift register if the SPCR.MSTR bit is 0, and connects the input and output paths of the shift register, establishing a loopback mode. The SPI does not shut off the path between the MOSIn pin and the shift register if the SPCR.MSTR bit is 1, or between the MISOn pin and the shift register if the SPCR.MSTR bit is 0. This is called loopback mode. When a serial transfer is executed in loopback mode, the transmit data for the SPI or the reversed transmit data becomes the received data for the SPI.

Table 30.14 lists the relationship between the SPLP2 and SPLP bits and the received data. Figure 30.61 shows the configuration of the shift register I/O paths when the SPI in master mode is set to loopback mode (SPPCR.SPLP2 = 0, SPPCR.SPLP = 1).

**Table 30.14 SPLP2 and SPLP bit settings and received data**

| SPPCR.SPLP2 bit | SPPCR.SPLP bit | Received data |
|---|---|---|
| 0 | 0 | Input data from the MOSIn pin or MISOn pin |
| 0 | 1 | Inverted transmit data |
| 1 | 0 | Transmit data |
| 1 | 1 | Transmit data |

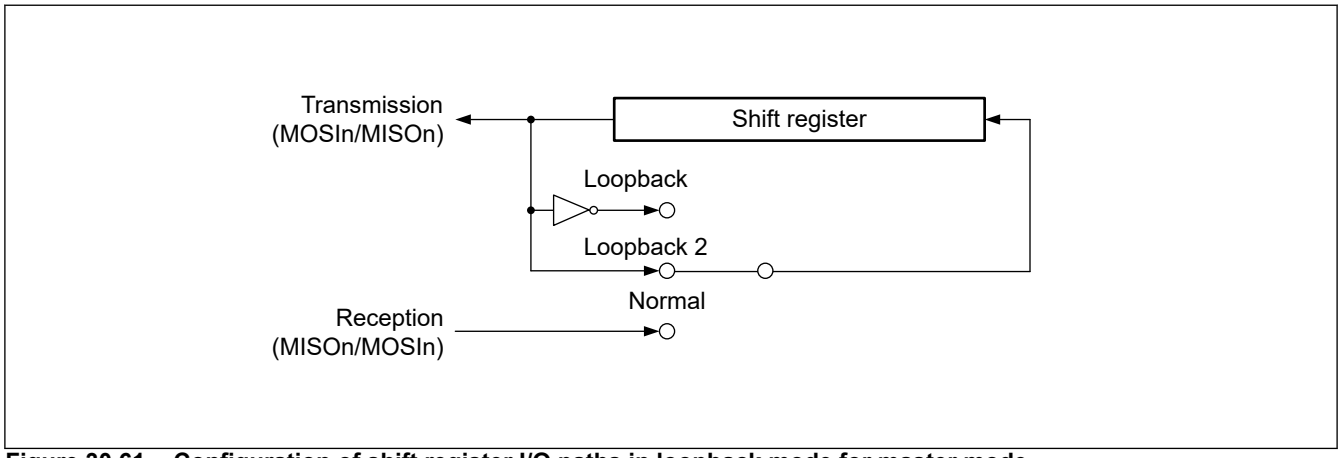



**Figure 30.61 Configuration of shift register I/O paths in loopback mode for master mode**

## 30.3.14 Self-Diagnosis of Parity Bit Function

The parity circuit consists of a parity bit adding unit used for transmit data and an error detecting unit used for received data. To detect defects in the parity bit adding unit and error detecting unit, the parity circuit performs self-diagnosis as shown in Figure 30.62.

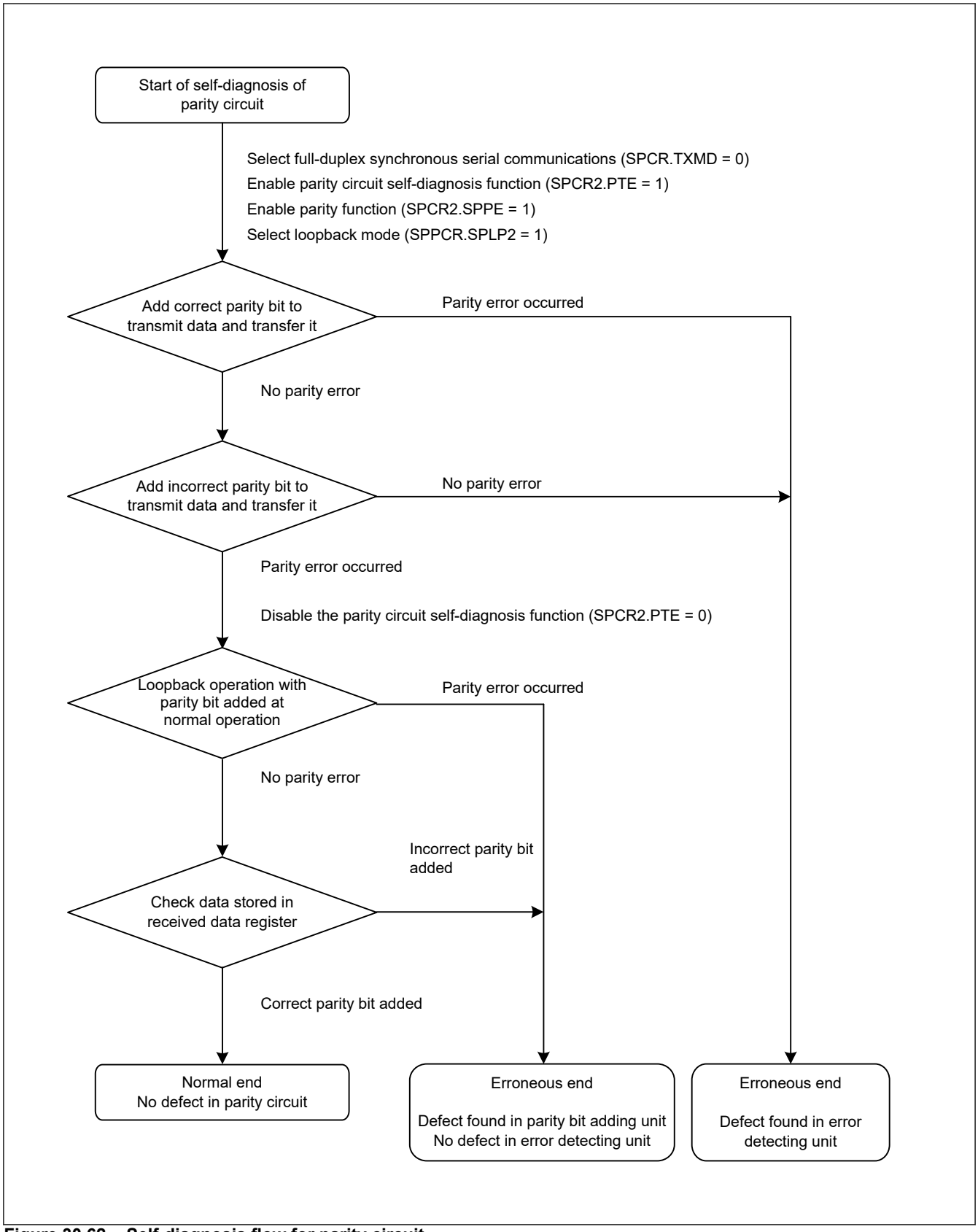



**Figure 30.62 Self-diagnosis flow for parity circuit**

## 30.3.15 Interrupt Sources

The SPI has the following interrupt sources:

- Receive buffer full
- Transmit buffer empty
- SPI error (mode-fault, underrun, overrun, or parity error)
- SPI idle
- Communication-end

The DMAC or DTC can be activated by the receive buffer full or transmit buffer empty interrupt to perform data transfer.

Because the vector address for the SPIi_SPEI (SPI error interrupt) is allocated to interrupt requests on mode-fault, underrun, overrun, and parity errors, the actual interrupt source must be determined from the flags. Interrupt sources for the SPI are listed in Table 30.15. An interrupt is generated on satisfaction of one of the interrupt conditions in Table 30.15. Clear the receive buffer full and transmit buffer empty sources through a data transfer.

When using the DMAC or DTC to perform data transmission and reception, you must first set up the DMAC or DTC to be in a transfer-enabled status before setting the SPI. For information on setting up the DMAC or DTC, see section 16, DMA Controller (DMAC) and section 17, Data Transfer Controller (DTC).

If the conditions for generating a transmit buffer empty or receive buffer full interrupt occur while the ICU.IELSRn.IR flag is 1, the interrupt is not output as a request for the ICU but is retained internally (the capacity for retention is one request per source). A retained interrupt request is output when the ICU.IELSRn.IR flag becomes 0. A retained interrupt request is automatically discarded when it is output as an actual interrupt request. The interrupt enable bit (the SPCR.SPTIE or SPCR.SPRIE bit) for an internally retained interrupt request can also be set to 0.

**Table 30.15 SPI interrupt sources**

| Interrupt source | Symbol | Interrupt condition | DTC/DMAC activation |
|---|---|---|---|
| Receive buffer full | SPIi_SPRI | The receive buffer becomes full (SPSR.SPRF flag is 1) while the SPCR.SPRIE bit is 1 | Possible |
| Transmit buffer empty | SPIi_SPTI | The transmit buffer becomes empty (SPSR.SPTEF flag is 1) while the SPCR.SPTIE bit is 1 | Possible |
| SPI error (mode-fault, underrun, overrun, or parity error) | SPIi_SPEI | The SPSR.MODF, OVRF, UDRF or PERF flag sets to 1 while the SPCR.SPEIE bit is 1 | Impossible |
| SPI idle | SPIi_SPII | The SPSR.IDLNF flag sets to 0 while the SPCR2.SPIIE bit is 1 | Impossible |
| Communication-end | SPIi_SPCI | CENDIE = 1 and CENDF = 1 | Impossible |

## 30.4 Event Link Controller Event Output

The Event Link Controller (ELC) can produce the following event output signals:

- Receive buffer full event output
- Transmit buffer empty event output
- Mode-fault, underrun, overrun, or parity error event output
- SPI idle event output
- Transmission-completed event output

The event link output signal is output regardless of the interrupt enable bit setting.

### 30.4.1 Receive Buffer Full Event Output

This event signal is output when received data is transferred from the shift register to the SPDR/SPDR_HA on completion of serial transfer.

### 30.4.2 Transmit Buffer Empty Event Output

This event signal is output when data for transmission is transferred from the transmit buffer to the shift register and when the value of the SPE bit changes from 0 to 1.

### 30.4.3 Mode-Fault, Underrun, Overrun, or Parity Error Event Output

This event signal is output when mode-fault, underrun, overrun, or parity error is detected. See section 30.5.4. Constraints on Mode-Fault, Underrun, Overrun, or Parity Error Event Output if using this event signal.

#### (1) Mode-fault

Table 30.16 lists the conditions for occurrence of a mode-fault event.

**Table 30.16 Conditions for mode-fault occurrence**

| SPI mode | SPCR.MODFEN bit | SSLn0 pin | Remarks |
|---|---|---|---|
| SPI operation (SPMS = 0)<br>Slave (SPCR.MSTR = 0) | 1 | Not active | Event is output only when the SSLn0 pin is deactivated during transmission |

#### (2) Underrun

This event signal is output in response to an underrun when a serial transfer starts while the transmission data is not ready, and the value of the SPCR.MSTR bit is 0 and the SPCR.SPE bit is 1. Under these conditions, the MODF and UDRF flags are set to 1.

#### (3) Overrun

This event signal is output in response to an overrun when a serial transfer completes while the receive buffer contains unread data and the value of the SPCR.TXMD bit is 0. Under these conditions, the OVRF flag is set to 1.

#### (4) Parity error

This event signal is output in response to a parity error detected on completion of a serial transfer while the value of the TXMD bit in SPCR is 0 and the value of the SPPE bit in SPCR2 is 1.

### 30.4.4 SPI Idle Event Output

#### (1) In master mode

In master mode, an event is output when the condition for setting the IDLNF flag (SPI idle flag) to 0 is satisfied.

#### (2) In slave mode

In slave mode, an event is output when the SPCR.SPE bit is set to 0 (SPI is initialized).

### 30.4.5 Communication End Event Output

In master mode, an event is output when the IDLNF flag (SPI idle flag) changes from 1 to 0. In slave mode, an event occurs with conditions shown in Table 30.17 and Table 30.18

**Table 30.17 Communication End Event Generating Conditions (transmit-receive/transmit slave mode)**

| | Transmit Buffer Status | Shift Register Status | Others |
|---|---|---|---|
| SPI operation (SPMS = 0) | Empty | Empty | SSL0 input is negated |
| Clock synchronous operation (SPMS = 1) | Empty | Empty | The last even edge of RSPCK of last data was detected (CPHA = 1) |

**Table 30.18 Communication End Event Generating Conditions (receive only slave mode)**

| | Others |
|---|---|
| SPI operation (SPMS = 0) | SSL0 input is negated |
| Clock synchronous operation (SPMS = 1) | The last even edge of RSPCK of last data was detected (CPHA = 1) |

Regardless of master mode or slave mode, no event is output when 0 is written to the SPCR.SPE bit during transmission or when the SPCR.SPE bit is cleared due to a mode fault error or an underrun error.

A communication end event is output at the following timing. The communication end event output timing in master operation is omitted because it is output at the same timing as an idle event.

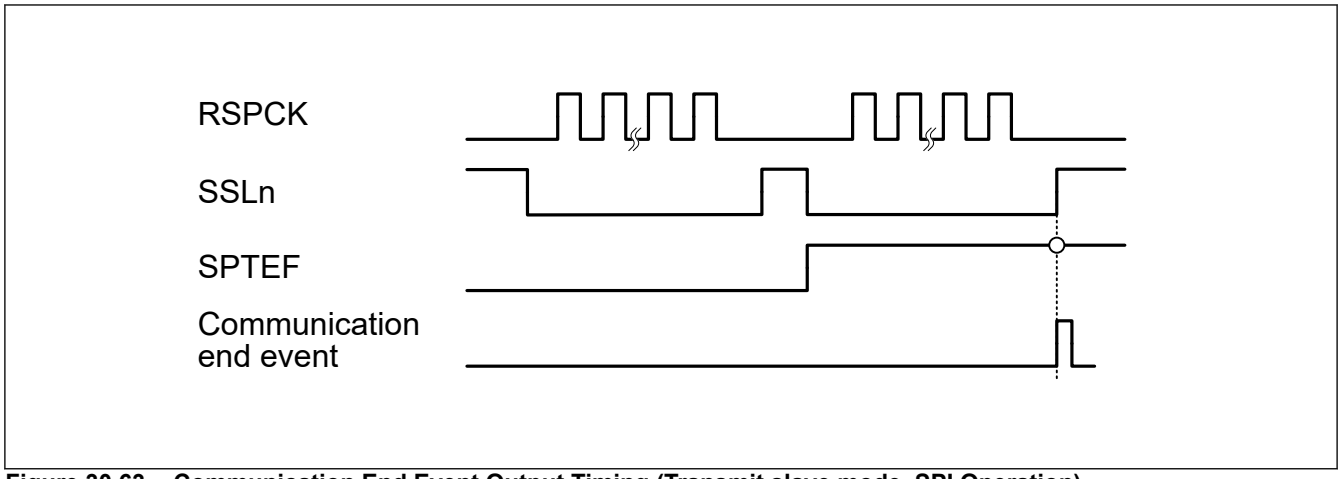



**Figure 30.63 Communication End Event Output Timing (Transmit slave mode, SPI Operation)**

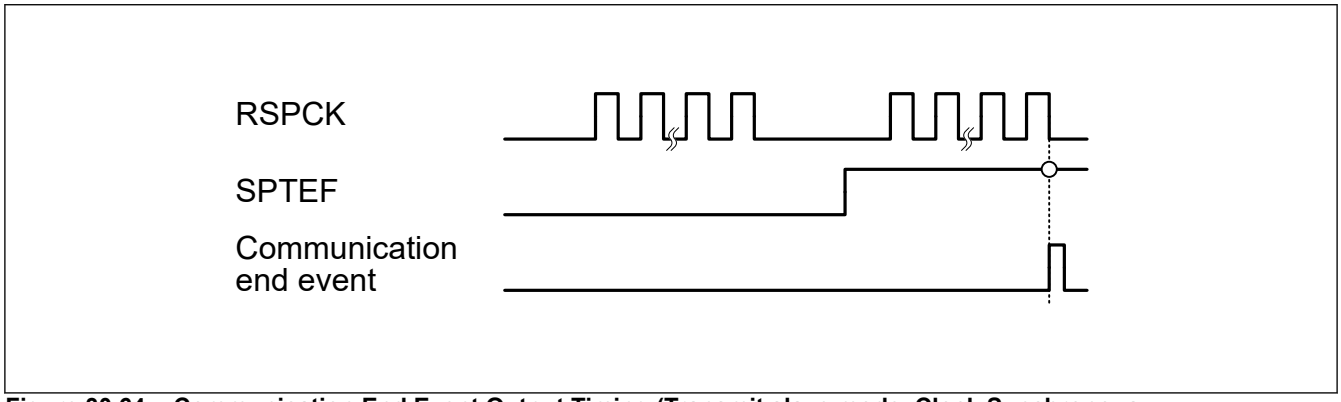



**Figure 30.64 Communication End Event Output Timing (Transmit slave mode, Clock Synchronous Operation)**

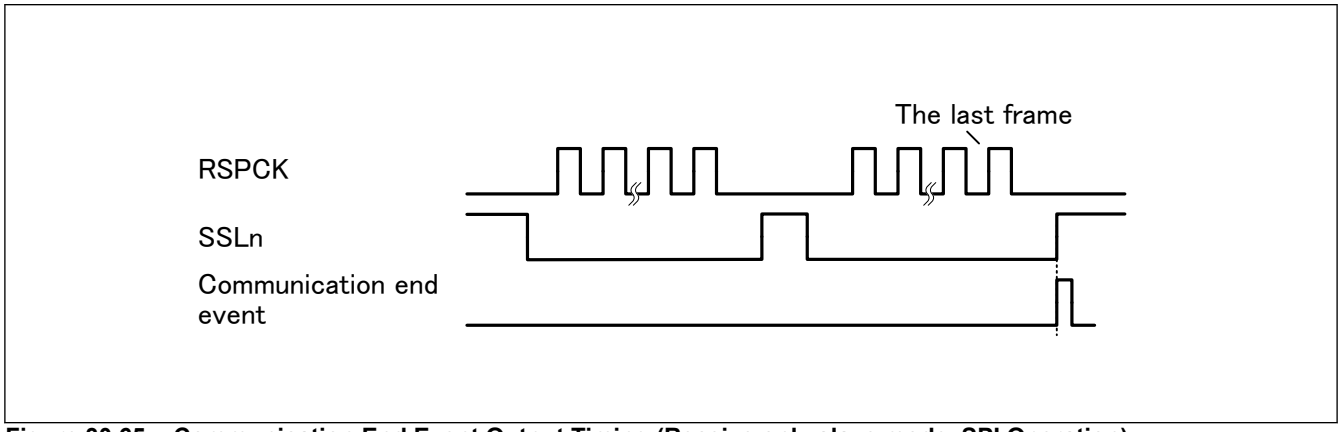



**Figure 30.65 Communication End Event Output Timing (Receive only slave mode, SPI Operation)**

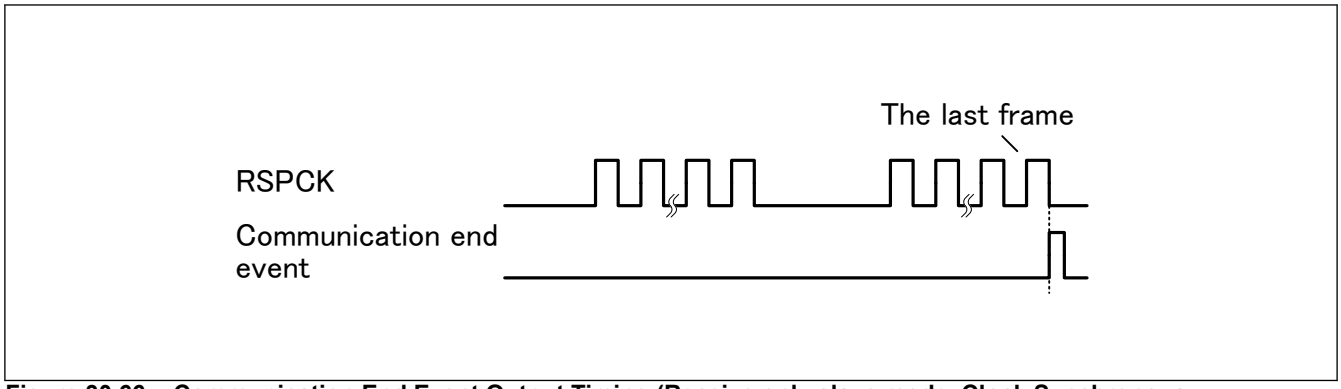



**Figure 30.66 Communication End Event Output Timing (Receive only slave mode, Clock Synchronous Operation)**

## 30.5 Usage Notes

### 30.5.1 Settings for the Module-Stop State

The Module Stop Control Register B (MSTPCRB) can enable or disable the SPI operation. The SPI is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details on the Module Stop Control Register B, see section 10, Low Power Modes.

### 30.5.2 Constraint on Low-Power Functions

When using the module-stop function and entering a low-power mode other than Sleep mode, set the SPCR.SPE bit to 0 before completing communication.

### 30.5.3 Constraints on Starting Transfer

If the ICU.IELSRn.IR flag is 1 when transfer starts, the interrupt request is internally retained, which can lead to unanticipated behavior of the ICU.IELSRn.IR flag.

To prevent this, use the following procedure to clear interrupt requests before enabling operations (by setting the SPCR.SPE bit to 1):

1. Confirm that transfer stopped (the SPCR.SPE bit is 0).
2. Set the associated interrupt enable bit (SPCR.SPTIE bit or SPCR.SPRIE bit) to 0.
3. Read the associated interrupt enable bit (SPCR.SPTIE bit or SPCR.SPRIE bit) and confirm that its value is 0.
4. Set the ICU.IELSRn.IR flag to 0.

### 30.5.4 Constraints on Mode-Fault, Underrun, Overrun, or Parity Error Event Output

Using the mode-fault, underrun, overrun, or parity error event is prohibited if the SPI is in multi-master mode (when the SPCR.SPMS bit is 0, the SPCR.MSTR bit is 1, and the SPCR.MODFEN bit is 1).

### 30.5.5 Constraints on the SPSR.SPRF and SPSR.SPTEF Flags

If the polling flags, SPRF and SPTEF, are used, using the interrupts is prohibited, and you must set the SPCR.SPRIE and SPCR.SPTIE bits to 0. Either the interrupts or the flags can be used, but not both.

# 31. Quad Serial Peripheral Interface (QSPI)

## 31.1 Overview

The QSPI is a memory controller for connecting a serial ROM (nonvolatile memory such as a serial flash memory, serial EEPROM, or serial FeRAM) that has an SPI-compatible interface.

Table 31.1 lists the QSPI specifications, Figure 31.1 shows a block diagram, and Table 31.2 lists the I/O pins.

**Table 31.1 QSPI specifications**

| Parameter | Specifications |
|---|---|
| Number of channels | 1 channel |
| SPI protocols | • Single SPI protocol, extended SPI protocol to achieve full-duplex communications<br>Note: Standard or fast reading can only be used in single SPI operation.<br>Four-wire communications with the serial flash memory by using the QSSL, QSPCLK, QIO0, and QIO1 pins (QIO0, QSSL, and QSPCLK for output, and QIO1 for input)<br>• Dual SPI protocol to achieve half-duplex communications<br>Four-wire communications with the serial flash memory by using the QSSL, QSPCLK, QIO0, and QIO1 pins (QSSL and QSPCLK for output, and QIO0 and QIO1 for input and output)<br>• Quad SPI protocol to achieve half-duplex communications<br>Six-wire communications with the serial flash memory by using the QSSL, QSPCLK, and QIO0 to QIO3 pins (QSSL and QSPCLK for output, and QIO0 to QIO3 for input and output) |
| SPI mode | • SPI mode 0: The QSPCLK signal is driven low when the SPI bus is not active.<br>• SPI mode 3: The QSPCLK signal is driven high when the SPI bus is not active. |
| SPI timing adjustment function | The following settings are possible to suit various types of serial flash memory device:<br>• SPI bus reference cycle (SFMSKC.SFMDV[4:0])<br>• Duty cycle correction (SFMSKC.SFMDTY)<br>• Adjustment of the number of dummy cycles (SFMSDC.SFMDN[3:0])<br>• Minimum width at high level for the QSSL signal (SFMSSC.SFMSW[3:0])<br>• QSSL signal setup time (SFMSSC.SFMSLD)<br>• QSSL signal hold time (SFMSSC.SFMSHD)<br>• Serial data output enable hold time (SFMSMD.SFMOEX) |
| ROM access mode | • Support for Standard Read, Fast Read, Fast Read Dual Output, Fast Read Dual I/O, Fast Read Quad Output, and Fast Read Quad I/O instructions<br>• Substitutable instruction code<br>• Prefetch function (data are sequentially stored in a buffer after one request without waiting for further requests to read the serial flash memory)<br>• Polling processing<br>• SPI bus cycle extension function<br>• XIP mode (allowing skipping of the reception of an instruction code to read the serial flash memory)<br>Note: ROM access mode is only possible with reading . |
| Direct communication mode | Flexible support for a wide variety of serial flash memory instructions and functions through software control, including erase, ID read, and power-down control |
| Interrupt source | Error interrupts |
| Module-stop function | Module-stop state can be set to reduce power consumption. |
| TrustZone Filter | Security attribution is always non-secure |

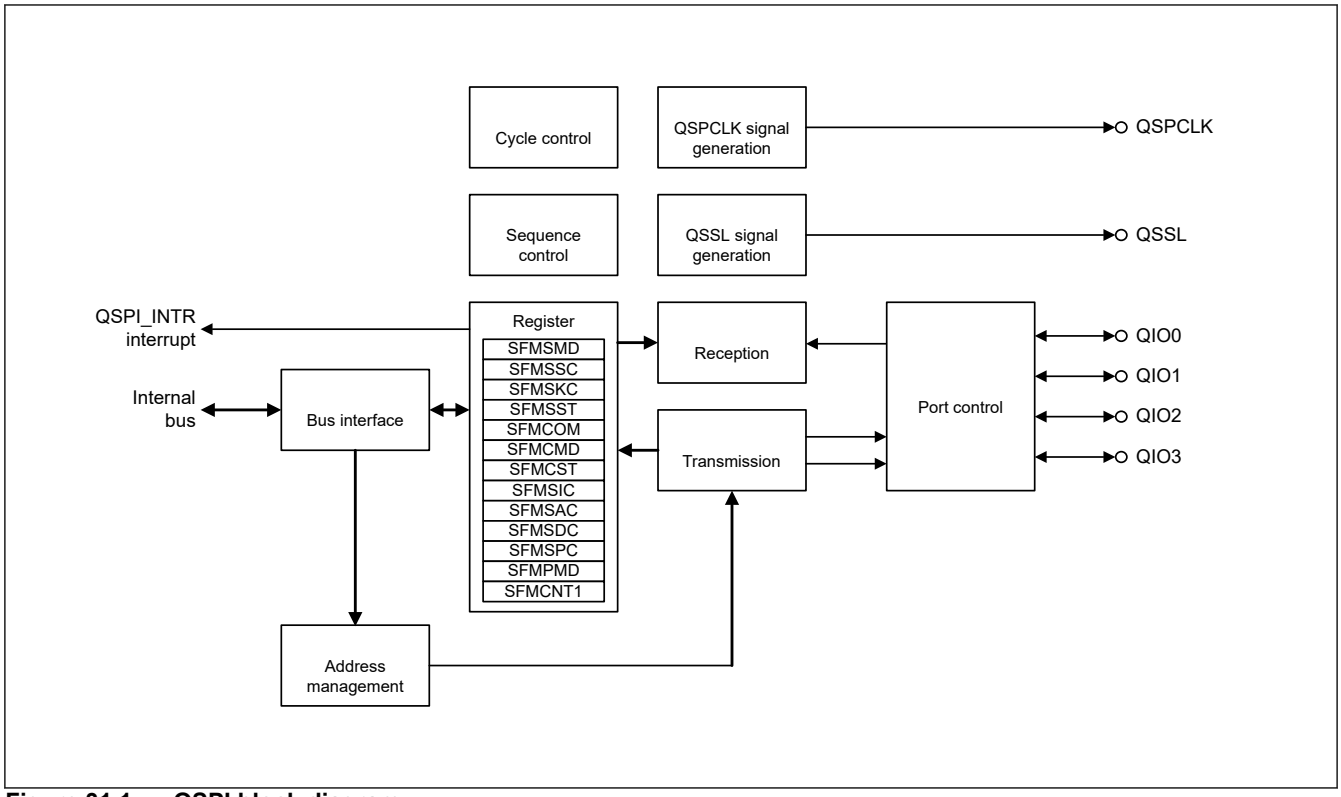



**Figure 31.1 QSPI block diagram**

**Table 31.2 QSPI I/O pins**

| Function | Pin name | I/O | Description |
|---|---|---|---|
| QSPI | QSPCLK | Output | QSPI clock output pin. |
| | QSSL | Output | QSPI slave output pin. |
| | QIO0 to QIO3 | I/O | Data0 to Data3 |

## 31.2 Register Descriptions

### 31.2.1 SFMSMD : Transfer Mode Control Register

Base address: QSPI = 0x6400_0000

Offset address: 0x000

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SFMCCE | — | — | — | SFMOSW | SFMOHW | SFMOEX | SFMMD3 | SFMPAE | SFMPFE | SFMSE[1:0] | | — | SFMRM[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | SFMRM[2:0] | Serial interface read mode select<br>0 0 0: Standard Read<br>0 0 1: Fast Read<br>0 1 0: Fast Read Dual Output<br>0 1 1: Fast Read Dual I/O<br>1 0 0: Fast Read Quad Output<br>1 0 1: Fast Read Quad I/O<br>Others: Setting prohibited | R/W |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 5:4 | SFMSE[1:0] | QSSL extension function select after SPI bus access<br>0 0: Do not extend QSSL<br>0 1: Extend QSSL by 33 QSPCLK<br>1 0: Extend QSSL by 129 QSPCLK<br>1 1: Extend QSSL infinitely | R/W |
| 6 | SFMPFE | Prefetch function select<br>0: Disable function<br>1: Enable function | R/W |
| 7 | SFMPAE | Function select for stopping prefetch at locations other than on byte boundaries*1<br>0: Disable function<br>1: Enable function | R/W |
| 8 | SFMMD3 | SPI mode select.<br>0: SPI mode 0<br>1: SPI mode 3 | R/W |
| 9 | SFMOEX | Extension select for the I/O buffer output enable signal for the serial interface<br>0: Do not extend<br>1: Extend by 1 QSPCLK | R/W |
| 10 | SFMOHW | Hold time adjustment for serial transmission<br>0: Do not extend high-level width of QSPCLK during transmission<br>1: Extend high-level width of QSPCLK by 1 PCLKA during transmission | R/W |
| 11 | SFMOSW | Setup time adjustment for serial transmission<br>0: Do not extend low-level width of QSPCLK during transmission<br>1: Extend low-level width of QSPCLK by 1 PCLKA during transmission | R/W |
| 14:12 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15 | SFMCCE | Read instruction code select<br>0: Uses automatically generated SPI instruction code*2<br>1: Use instruction code in the SFMSIC register | R/W |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The QSPI outputs additional one clock without accompanying data reception. For details, see section 31.5.9. Serial Data Receiving Latency.

Note 2. When QSPI accesses serial flash memory, the instruction code is based on the SFMSAC register and SFMSMD register settings. See section 31.6.1. SPI Instructions That Are Automatically Generated.

## 31.2.2 SFMSSC : Chip Selection Control Register

Base address: QSPI = 0x6400_0000

Offset address: 0x004

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | SFMSLD | SFMSHD | SFMSW[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | SFMSW[3:0] | Minimum High-level Width Select for QSSL Signal<br>0x0: 1 QSPCLK<br>0x1: 2 QSPCLK<br>0x2: 3 QSPCLK<br>0x3: 4 QSPCLK<br>0x4: 5 QSPCLK<br>0x5: 6 QSPCLK<br>0x6: 7 QSPCLK<br>0x7: 8 QSPCLK<br>0x8: 9 QSPCLK<br>0x9: 10 QSPCLK<br>0xA: 11 QSPCLK<br>0xB: 12 QSPCLK<br>0xC: 13 QSPCLK<br>0xD: 14 QSPCLK<br>0xE: 15 QSPCLK<br>0xF: 16 QSPCLK | R/W |
| 4 | SFMSHD | QSSL Signal Hold Time<br>0: QSSL outputs high after 0.5 QSPCLK cycles from the last rising edge of QSPCLK.<br>1: QSSL outputs high after 1.5 QSPCLK cycles from the last rising edge of QSPCLK. | R/W |
| 5 | SFMSLD | QSSL Signal Setup Time<br>0: QSSL outputs low before 0.5 QSPCLK cycles from the first rising edge of QSPCLK.<br>1: QSSL outputs low before 1.5 QSPCLK cycles from the first rising edge of QSPCLK. | R/W |
| 31:6 | — | These bits are read as 0. The write value should be 0. | R/W |

## 31.2.3 SFMSKC : Clock Control Register

Base address: QSPI = 0x6400_0000

Offset address: 0x008

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | SFMDTY | SFMDV[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | SFMDV[4:0] | Serial interface reference cycle select. (Pay attention to irregularities.)<br>0x00: 2 PCLKA<br>0x01: 3 PCLKA (divided by an odd number)*1<br>0x02: 4 PCLKA<br>0x03: 5 PCLKA (divided by an odd number)*1<br>0x04: 6 PCLKA<br>0x05: 7 PCLKA (divided by an odd number)*1<br>0x06: 8 PCLKA<br>0x07: 9 PCLKA (divided by an odd number)*1<br>0x08: 10 PCLKA<br>0x09: 11 PCLKA (divided by an odd number)*1<br>0x0A: 12 PCLKA<br>0x0B: 13 PCLKA (divided by an odd number)*1<br>0x0C: 14 PCLKA<br>0x0D: 15 PCLKA (divided by an odd number)*1<br>0x0E: 16 PCLKA<br>0x0F: 17 PCLKA (divided by an odd number)*1<br>0x10: 18 PCLKA<br>0x11: 20 PCLKA<br>0x12: 22 PCLKA<br>0x13: 24 PCLKA<br>0x14: 26 PCLKA<br>0x15: 28 PCLKA<br>0x16: 30 PCLKA<br>0x17: 32 PCLKA<br>0x18: 34 PCLKA<br>0x19: 36 PCLKA<br>0x1A: 38 PCLKA<br>0x1B: 40 PCLKA<br>0x1C: 42 PCLKA<br>0x1D: 44 PCLKA<br>0x1E: 46 PCLKA<br>0x1F: 48 PCLKA | R/W |
| 5 | SFMDTY | Duty ratio correction function select for the QSPCLK signal when devided by an odd number<br>0: Make no correction<br>1: Make correction | R/W |
| 31:6 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Set the SFMDTY bit to 1 when PCLKA is to be divided by an odd number.

## 31.2.4 SFMSST : Status Register

Base address: QSPI = 0x6400_0000

Offset address: 0x00C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | PFOFF | PFFUL | — | PFCNT[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | PFCNT[4:0] | Number of bytes of prefetched data<br>0x00: 0 byte<br>0x01: 1 byte<br>0x02: 2 bytes<br>0x03: 3 bytes<br>0x04: 4 bytes<br>0x05: 5 bytes<br>0x06: 6 bytes<br>0x07: 7 bytes<br>0x08: 8 bytes<br>0x09: 9 bytes<br>0x0A: 10 bytes<br>0x0B: 11 bytes<br>0x0C: 12 bytes<br>0x0D: 13 bytes<br>0x0E: 14 bytes<br>0x0F: 15 bytes<br>0x10: 16 bytes<br>0x11: 17 bytes<br>0x12: 18 bytes<br>Others: Reserved | R |
| 5 | — | This bit is read as 0. | R |
| 6 | PFFUL | Prefetch buffer state<br>0: Prefetch buffer has free space<br>1: Prefetch buffer is full | R |
| 7 | PFOFF | Prefetch function operating state<br>0: Prefetch function operating<br>1: Prefetch function not enabled or not operating | R |
| 31:8 | — | These bits are read as 0. | R |

## 31.2.5 SFMCOM : Communication Port Register

Base address: QSPI = 0x6400_0000

Offset address: 0x010

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | SFMD[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x | x | x | x | x | x |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | SFMD[7:0] | Port for direct communication with the SPI bus<br>Input and output from this port are converted to an SPI bus cycle in direct communications mode (DCOM = 1). Access to this port is ignored in ROM access mode. | R/W |
| 31:8 | — | These bits are read as 0. The write value should be 0. | R/W |

## 31.2.6 SFMCMD : Communication Mode Control Register

Base address: QSPI = 0x6400_0000

Offset address: 0x014

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | DCOM |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | DCOM | Mode select for communication with the SPI bus<br>0: ROM access mode<br>1: Direct communication mode*1 | R/W |
| 31:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. SFMCMD.DCOM = 1 must be written when the transaction ends. For details, see section 31.10. Direct Communication Mode.

## 31.2.7 SFMCST : Communication Status Register

Base address: QSPI = 0x6400_0000

Offset address: 0x018

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | EROMR | — | — | — | — | — | — | COMBSY |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | COMBSY | SPI bus cycle completion state in direct communication<br>0: No serial transfer being processed<br>1: Serial transfer being processed | R |
| 6:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | EROMR | ROM access detection status in direct communication mode<br>0: ROM access not detected<br>1: ROM access detected | R/(W)*1 |
| 31:8 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Only 0 can be written to this bit.

## 31.2.8 SFMSIC : Instruction Code Register

Base address: QSPI = 0x6400_0000

Offset address: 0x020

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | SFMCIC[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | SFMCIC[7:0] | Serial flash instruction code to substitute | R/W |
| 31:8 | — | These bits are read as 0. The write value should be 0. | R/W |

## 31.2.9 SFMSAC : Address Mode Control Register

Base address: QSPI = 0x6400_0000

Offset address: 0x024

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | SFM4BC | — | — | SFMAS[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | SFMAS[1:0] | Number of address bytes select for the serial interface<br>0 0: 1 byte<br>0 1: 2 bytes<br>1 0: 3 bytes<br>1 1: 4 bytes | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | SFM4BC | Selection of instruction code automatically generated when the serial interface address width is 4 bytes<br>0: Do not use 4-byte address read instruction code<br>1: Use 4-byte address read instruction code | R/W |
| 31:5 | — | These bits are read as 0. The write value should be 0. | R/W |

## 31.2.10 SFMSDC : Dummy Cycle Control Register

Base address: QSPI = 0x6400_0000

Offset address: 0x028

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SFMXD[7:0] | | | | | | | | SFMX EN | SFMX ST | — | — | SFMDN[3:0] | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | SFMDN[3:0] | Number of dummy cycles select for Fast Read instructions<br>0x0: Default dummy cycles for each instruction:<br>- Fast Read Quad I/O: 6 QSPCLK<br>- Fast Read Quad Output: 8 QSPCLK<br>- Fast Read Dual I/O: 4 QSPCLK<br>- Fast Read Dual Output: 8 QSPCLK<br>- Fast Read: 8 QSPCLK<br>0x1: 3 QSPCLK*1<br>0x2: 4 QSPCLK<br>0x3: 5 QSPCLK<br>0x4: 6 QSPCLK<br>0x5: 7 QSPCLK<br>0x6: 8 QSPCLK<br>0x7: 9 QSPCLK<br>0x8: 10 QSPCLK<br>0x9: 11 QSPCLK<br>0xA: 12 QSPCLK<br>0xB: 13 QSPCLK<br>0xC: 14 QSPCLK<br>0xD: 15 QSPCLK<br>0xE: 16 QSPCLK<br>0xF: 17 QSPCLK | R/W |
| 5:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | SFMXST | XIP mode status<br>0: Normal (non-XIP) mode<br>1: XIP mode | R |
| 7 | SFMXEN | XIP mode permission<br>0: Prohibit XIP mode<br>1: Permit XIP mode | R/W |
| 15:8 | SFMXD[7:0] | Mode data for serial flash (Controls XIP mode.)*2 | R/W |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. To avoid a conflict with the input/output switch of the serial flash memory pin connected to QIO0 pin, select more than four cycles of QSPCLK as the number of dummy cycles for the fast read instruction when the output enable signal is extended by setting the SFMOEX bit in the SFMSMD register to 1.

Note 2. As the mode data for serial flash memory, specify the XIP mode setting data set in actual serial flash memory.

## 31.2.11 SFMSPC : SPI Protocol Control Register

Base address: QSPI = 0x6400_0000

Offset address: 0x030

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | SFMSDE | — | — | SFMSPI[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | SFMSPI[1:0] | SPI protocol select*1<br>0 0: Single SPI Protocol, Extended SPI protocol<br>0 1: Dual SPI protocol<br>1 0: Quad SPI protocol<br>1 1: Setting prohibited | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | SFMSDE | QSPCLK extended selection bit when switching I/O of QIOn pin<br>0: No QSPCLK extension<br>1: QSPCLK expansion when switching I/O direction of QIOn pin | R/W |
| 31:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The states of the QIO2 and QIO3 pins change depending on the settings of the SFMSMD.SFMRM[2:0] and SFMSPI[1:0] bits. For details, see section 31.9. QIO2 and QIO3 Pin States.

## 31.2.12 SFMPMD : Port Control Register

Base address: QSPI = 0x6400_0000

Offset address: 0x034

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | SFMWPL | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 2 | SFMWPL | WP pin level specification<br>0: Low level<br>1: High level | R/W |
| 31:3 | — | These bits are read as 0. The write value should be 0. | R/W |

### 31.2.13 SFMCNT1 : External QSPI Address Register

Base address: QSPI = 0x6400_0000

Offset address: 0x804

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | QSPI_EXT[5:0] | | | | | | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 25:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 31:26 | QSPI_EXT[5:0] | Bank switching address<br>When accessing from 0x60000000 to 0x63FFFFFF, the address bus is set from QSPI_EXT[5:0] to the upper 6 bits of the internal bus address for the address bus.<br>0x00: QSPI bank 00<br>0x01: QSPI bank 01<br>0x02: QSPI bank 02<br>⋮ ⋮<br>0x3C: QSPI bank 60<br>0x3D: QSPI bank 61<br>0x3E: QSPI bank 62<br>0x3F: Setting prohibited | R/W |

## 31.3 Memory Map

### 31.3.1 External Bus Space

The locations of a serial flash memory and control register on the address space are determined by the address range of the area set in the configuration.

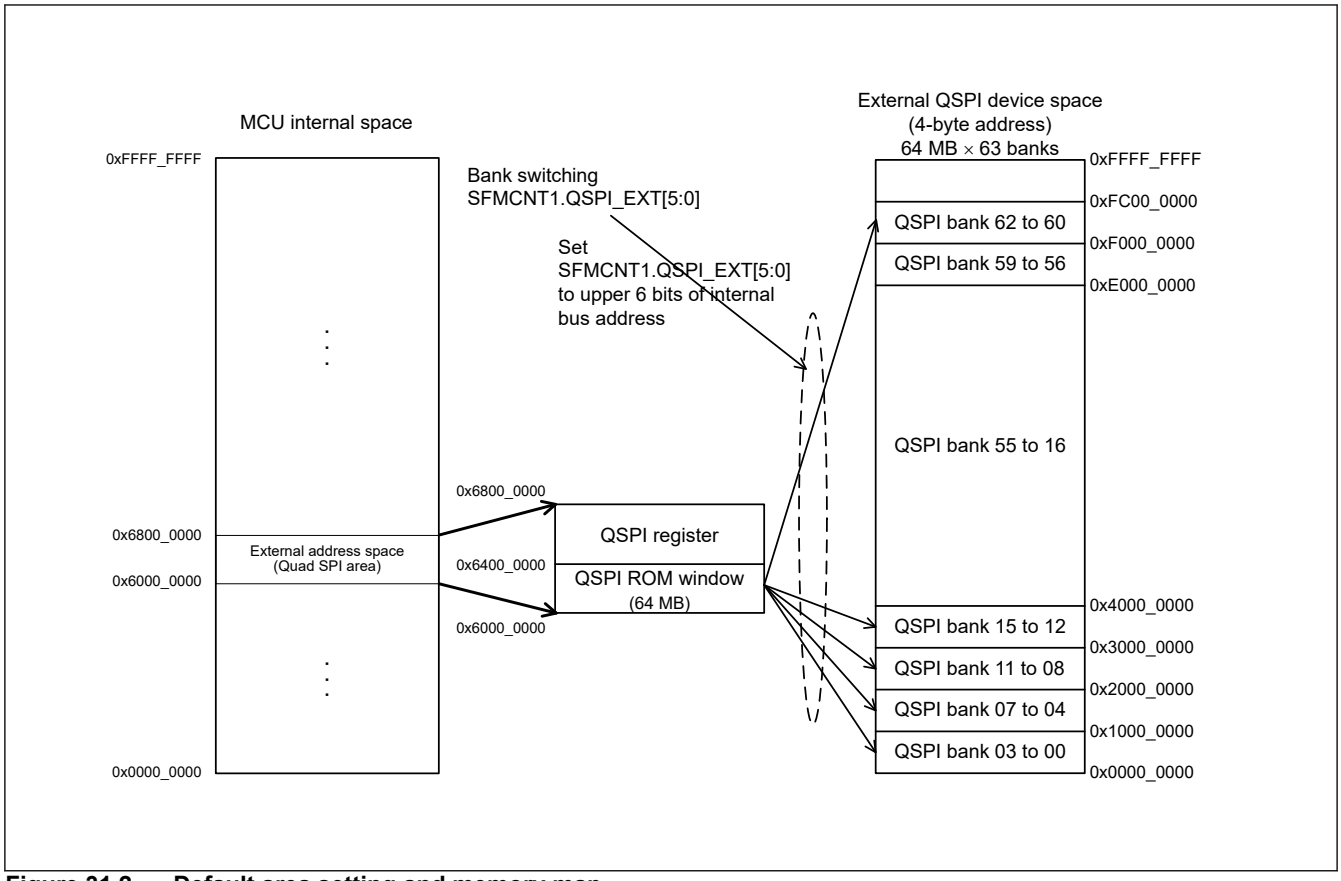



**Figure 31.2 Default area setting and memory map**

## 31.3.2 Address Width of the SPI Space and SPI Bus

The SPI space has a 32-bit address width for referencing the serial flash memory. When the SPI space is accessed for a read, an SPI bus cycle starts automatically, and data read from the serial flash is returned.

The address width of the SPI space is fixed at 32 bits. However, the address width of the SPI bus is selectable to 8, 16, 24, or 32 bits in the SFMAS[1:0] bits in the Address Mode Control Register (SFMSAC) register. If 8, 16, or 24 bits is selected as the address width of the SPI bus, only the lower part of the address used to access the SPI space is posted to the serial flash memory through the SPI bus. As a result, the mirror image of the serial flash corresponding to the address width of the SPI bus repeatedly appears in the SPI space.

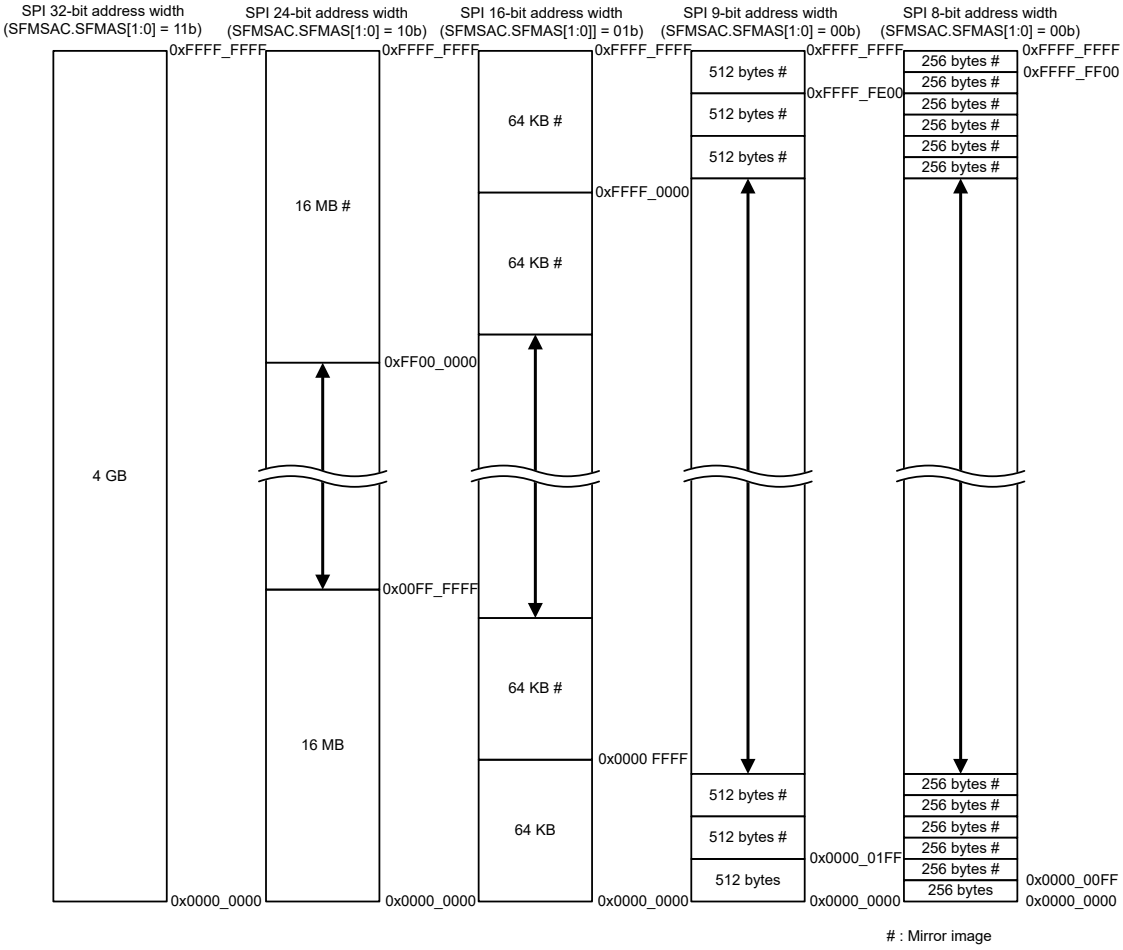



**Figure 31.3 Memory map of SPI space**

Note: The SPI bus address width is selectable to 8, 16, 24, or 32 bits in the SFMAS[1:0] bits in the SFMSAC register (cases 1 to 3 and 5 in the figure correspond to the respective address widths). When an 8-bit address width is selected, the address information of the ninth bit can be embedded in the Read instruction code. The memory map in case 4 in the figure is for the 9-bit address width. For details on the Read instruction, see section 31.6.2. Standard Read Instruction.

## 31.4 SPI Bus

### 31.4.1 SPI Protocol

Single SPI, extended SPI, dual SPI, and quad SPI are supported in addition to the SPI protocol used for serial flash memory connection.

The initial state of the SPI protocol is Single SPI, extended SPI and can be changed with the SFMSPI[1:0] bits in the SPI Protocol Control Register (SFMSPC) register.

The address and data pins used in the Single SPI, extended SPI protocol change depending on the setting of the serial interface read mode select bits SFMRM[2:0] in Transfer Mode Control Register (SFMSMD). Table 31.3 and Table 31.4 list the pins used for instruction code, addresses, and data in each of the SPI protocols.

Note: In read operation, the QSPI outputs additional one clock without accompanying data reception per one SPI bus cycle. For details, see section 31.5.9. Serial Data Receiving Latency.

**Table 31.3 List of SPI Protocols (1)**

| SPI Protocol (SFMSPC.SFMSPI[1:0]) | Single SPI Protocol, Extended SPI Protocol | | | | | |
|---|---|---|---|---|---|---|
| Serial interface read mode select (SFMSMD.SFMRM[2:0]) | Standard read | Fast read | Fast read dual output | Fast read dual I/O | Fast read quad output | Fast read quad I/O |
| All pins used | QSPCLK, QSSL, QIO0, QIO1 | QSPCLK, QSSL, QIO0, QIO1 | QSPCLK, QSSL, QIO0, QIO1 | QSPCLK, QSSL, QIO0, QIO1 | QSPCLK, QSSL, QIO0, QIO1, QIO2, QIO3 | QSPCLK, QSSL, QIO0, QIO1, QIO2, QIO3 |
| Pins used for instruction code | QIO0 | QIO0 | QIO0 | QIO0 | QIO0 | QIO0 |
| Pins used for addresses | QIO0 | QIO0 | QIO0 | QIO0, QIO1 | QIO0 | QIO0, QIO1, QIO2, QIO3 |
| Pins used for data | QIO0/QIO1 | QIO0/QIO1 | QIO0, QIO1 | QIO0, QIO1 | QIO0, QIO1, QIO2, QIO3 | QIO0, QIO1, QIO2, QIO3 |

Note: Single SPI protocol operation is for standard read and fast read. Extended SPI protocol operation is fast read dual output, fast read dual I/O, fast read quad output, and fast read quad I/O.

**Table 31.4 List of SPI Protocols (2)**

| SPI Protocol (SFMSPC.SFMSPI[1:0]) | Dual-SPI Protocol | | Quad-SPI Protocol | |
|---|---|---|---|---|
| Serial interface read mode select (SFMSMD.SFMRM[2:0]) | Fast read dual output | Fast read dual I/O | Fast read quad output | Fast read quad I/O |
| All pins used | QSPCLK, QSSL, QIO0, QIO1 | QSPCLK, QSSL, QIO0, QIO1 | QSPCLK, QSSL, QIO0, QIO1, QIO2, QIO3 | QSPCLK, QSSL, QIO0, QIO1, QIO2, QIO3 |
| Pins used for instruction code | QIO0, QIO1 | QIO0, QIO1 | QIO0, QIO1, QIO2, QIO3 | QIO0, QIO1, QIO2, QIO3 |
| Pins used for addresses | QIO0, QIO1 | QIO0, QIO1 | QIO0, QIO1, QIO2, QIO3 | QIO0, QIO1, QIO2, QIO3 |
| Pins used for data | QIO0, QIO1 | QIO0, QIO1 | QIO0, QIO1, QIO2, QIO3 | QIO0, QIO1, QIO2, QIO3 |

In single SPI protocol and extended SPI protocol, the instruction code is always output from the QIO0 pin. Address and data input/output operations are performed according to the settings in SFMSMD.SFMRM[2:0].

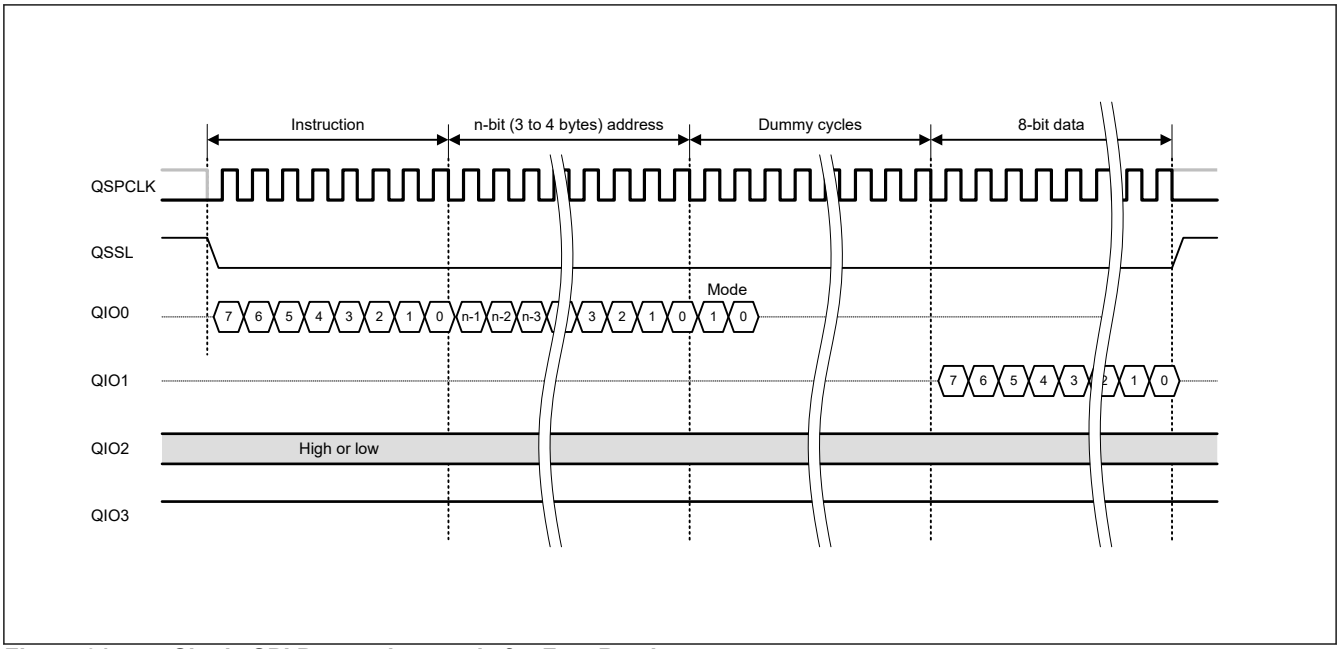



**Figure 31.4 Single SPI Protocol example for Fast Read**

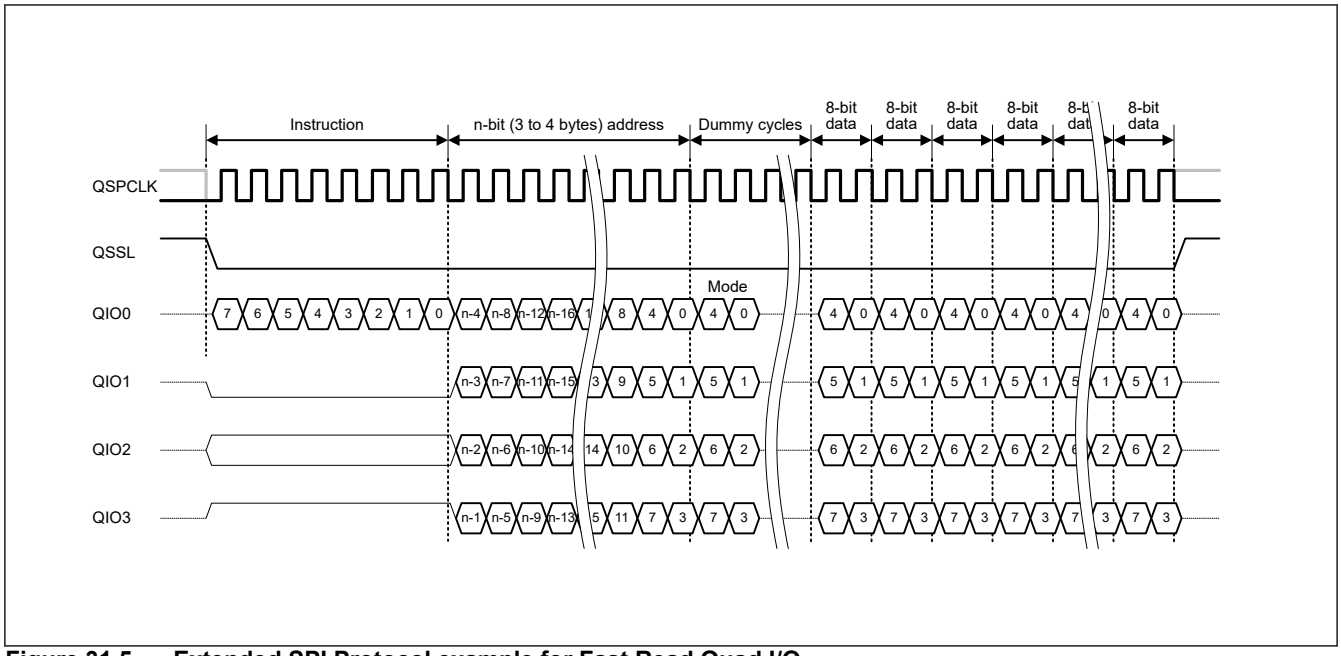



**Figure 31.5 Extended SPI Protocol example for Fast Read Quad I/O**

The Dual SPI protocol performs I/O operation of all signals such as instruction codes, addresses, and data using two pins, QIO0 and QIO1.

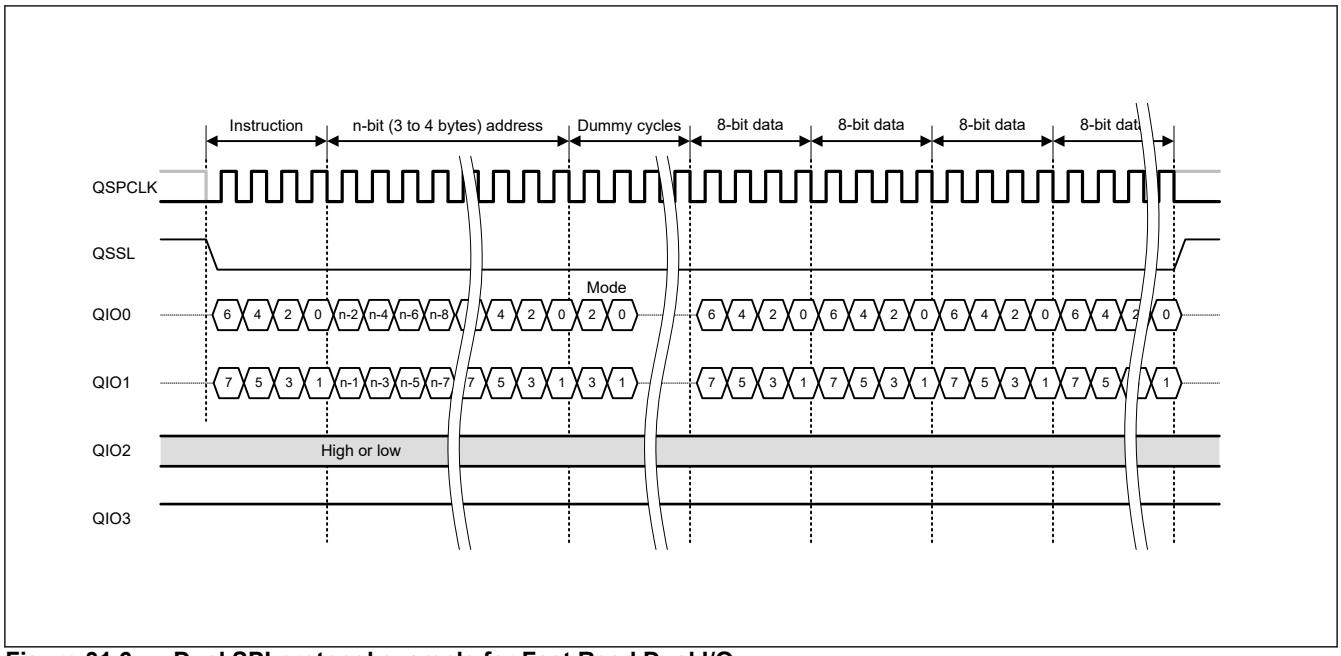



**Figure 31.6 Dual SPI protocol example for Fast Read Dual I/O**

The Quad SPI protocol performs I/O operation of all signals such as instruction codes, addresses, and data using four pins, QIO0, QIO1, QIO2, and QIO3.

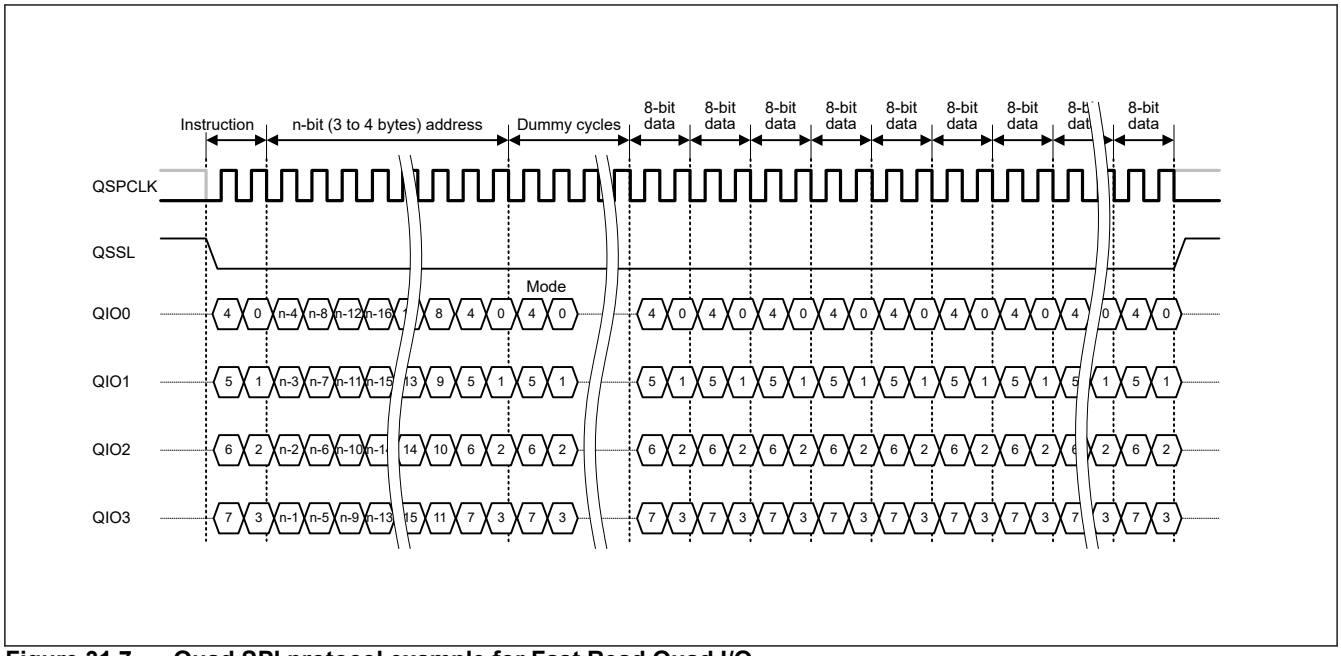



**Figure 31.7 Quad SPI protocol example for Fast Read Quad I/O**

## 31.4.2 SPI Mode

Either SPI mode 0 or SPI mode 3 can be selected as the SPI mode by the SFMSMD.SFMMD3 bit. This can be switched by changing the register setting during operation. The difference between SPI modes 0 and 3 is the state of the QSPCLK signal when it is inactive. The standby level of the QSPCLK signal in SPI mode 0 is low, and high in SPI mode 3.

Serial data is output from the QSPI on a falling edge of the serial clock and is read into the serial flash memory on a rising edge of the serial clock. Serial data is output from the serial flash memory on a falling edge of the serial clock and is read into the QSPI on the next falling edge of the serial clock.

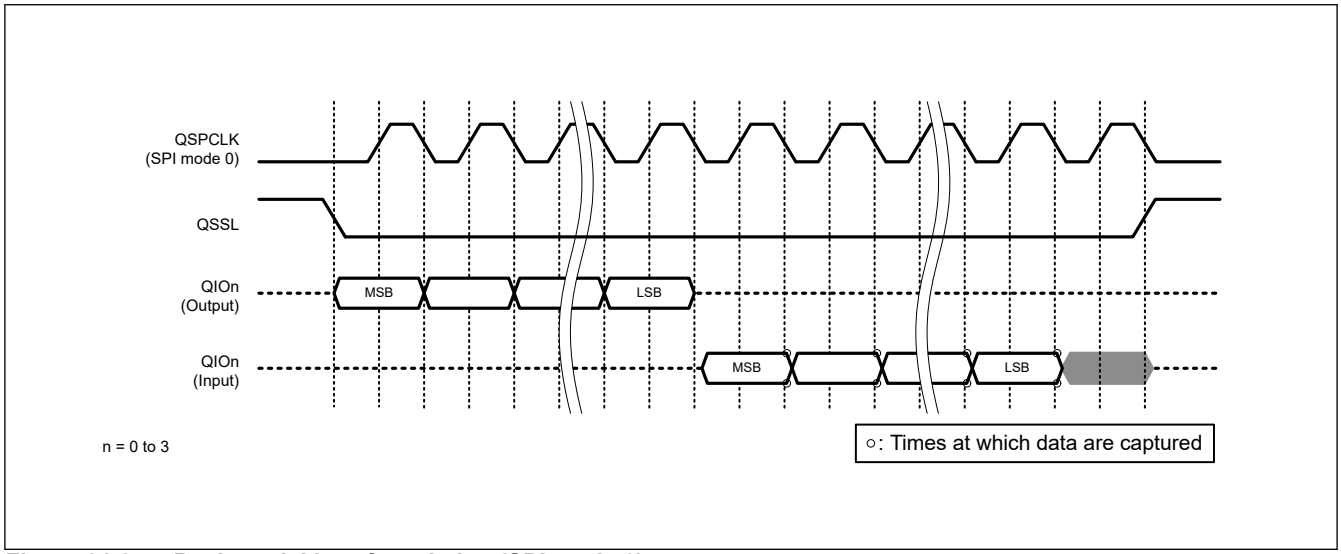



**Figure 31.8 Basic serial interface timing (SPI mode 0)**

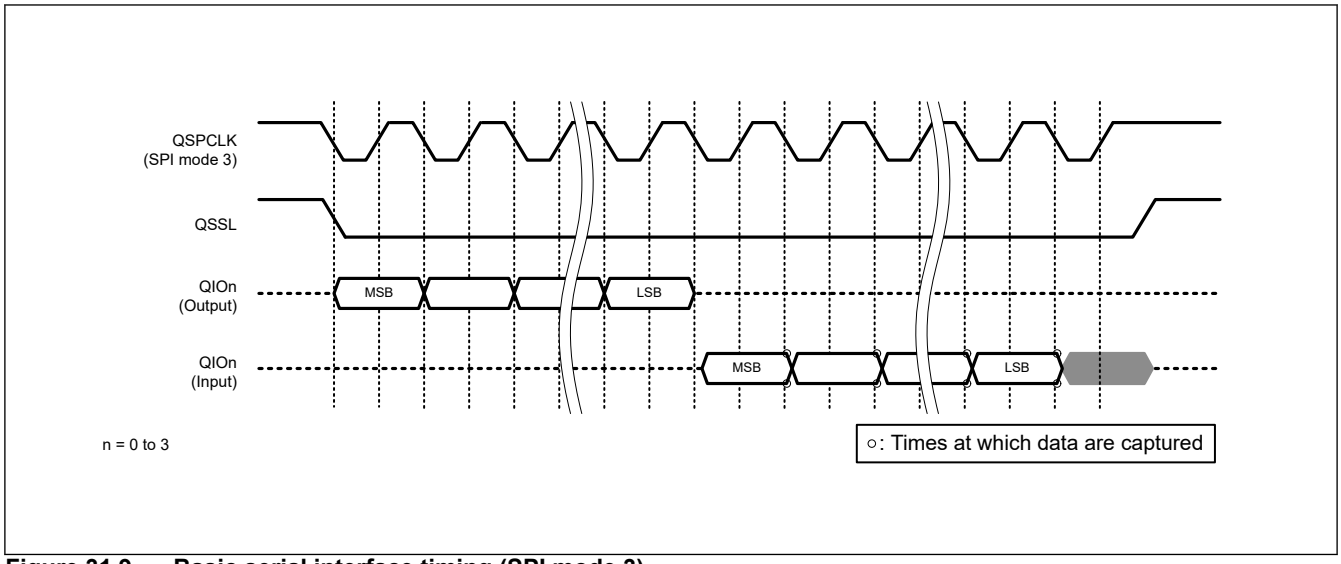



**Figure 31.9 Basic serial interface timing (SPI mode 3)**

## 31.5 SPI Bus Timing Adjustment

The timing of the SPI bus signal can be adjusted in the registers. The configured timing is applied to all SPI bus accesses, for both ROM access and direct communication.

### 31.5.1 SPI Bus Reference Cycles

The SPI bus operates on reference cycles obtained by multiplying PCLKA by an integer. The reference cycles are selectable within the range of PCLKA multiplied by 2 to 48 in the SFMDV[4:0] bits in the Transfer Mode Control Register (SFMSKC) register.

**Table 31.5 Relationship among SFMDV[4:0] bits, cycle multiplier, and serial clock frequencies (1 of 2)**

| SFMDV[4:0] | Cycle multiplier | PCLKA = 100 [MHz] |
|---|---|---|
| 11111b | 48 | 2.08 |
| 11110b | 46 | 2.17 |
| 11101b | 44 | 2.27 |
| 11100b | 42 | 2.38 |
| 11011b | 40 | 2.50 |
| 11010b | 38 | 2.63 |
| 11001b | 36 | 2.78 |
| 11000b | 34 | 2.94 |
| 10111b | 32 | 3.13 |
| 10110b | 30 | 3.33 |
| 10101b | 28 | 3.57 |
| 10100b | 26 | 3.85 |
| 10011b | 24 | 4.17 |
| 10010b | 22 | 4.55 |
| 10001b | 20 | 5.00 |
| 10000b | 18 | 5.56 |
| 01111b | 17 | 5.88 |
| 01110b | 16 | 6.25 |
| 01101b | 15 | 6.67 |

**Table 31.5 Relationship among SFMDV[4:0] bits, cycle multiplier, and serial clock frequencies (2 of 2)**

| SFMDV[4:0] | Cycle multiplier | PCLKA = 100 [MHz] |
|---|---|---|
| 01100b | 14 | 7.14 |
| 01011b | 13 | 7.69 |
| 01010b | 12 | 8.33 |
| 01001b | 11 | 9.09 |
| 01000b | 10 | 10.00 |
| 00111b | 9 | 11.11 |
| 00110b | 8 | 12.50 |
| 00101b | 7 | 14.29 |
| 00100b | 6 | 16.67 |
| 00011b | 5 | 20.00 |
| 00010b | 4 | 25.00 |
| 00001b | 3 | 33.33 |
| 00000b | 2 | 50.00 |

## 31.5.2 QSPCLK Signal Duty Ratio

When the reference clock is configured as PCLKA divided by an odd number without duty ratio correction, the duty ratio of the QSPCLK signal will not be 50%. When the reference clock is PCLKA divided by an odd number, be sure to enable the duty ratio correction function (SFMSKC.SFMDTY = 1).

When the reference clock is PCLKA divided by an even number, the SFMDTY setting in the SFMSKC register is ignored.

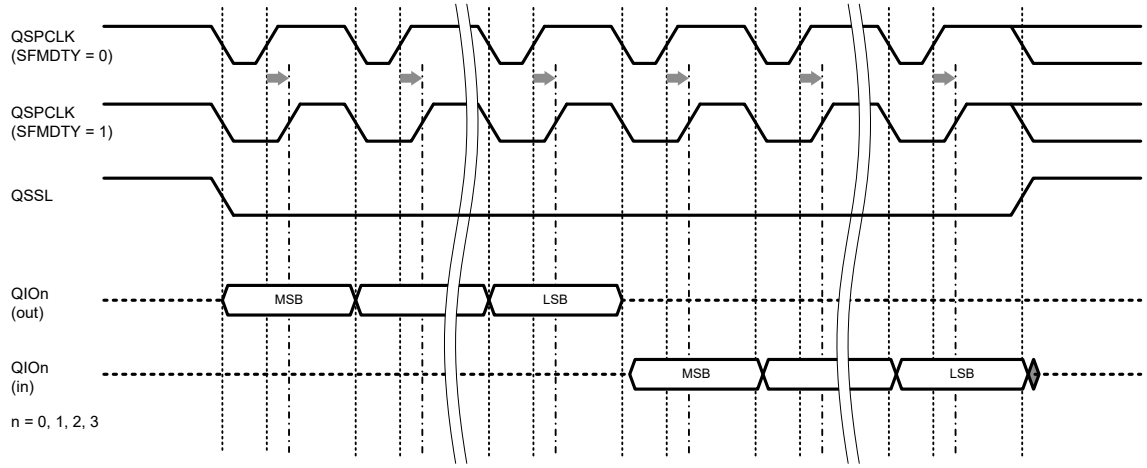



**Figure 31.10 Example correction of the QSPCLK signal duty ratio using the SFMDTY bit, when PCLKA is multiplied by 3**

## 31.5.3 Minimum High-Level Width for the QSSL Signal

Between adjacent SPI bus cycles, the QSSL signal must be held high (inactive) for a sufficient time to satisfy the deselect time required by the serial flash memory. The minimum high-level width of the QSSL output signal is selectable as the reference cycle multiplied by an integer from 1 to 16 in the SFMSW[3:0] bits in the Instruction Code Register (SFMSSC) register.

### 31.5.4 QSSL Signal Setup Time

The QSSL signal setup time that the serial flash memory requires after the QSSL signal is driven active low until the first rising edge of the QSPCLK signal can be configured. The setup time can be selected as 0.5 or 1.5 cycles of QSPCLK in the SFMSLD bit of the SFMSSC register.

Set a value that meets the most constrained timing condition for your application.

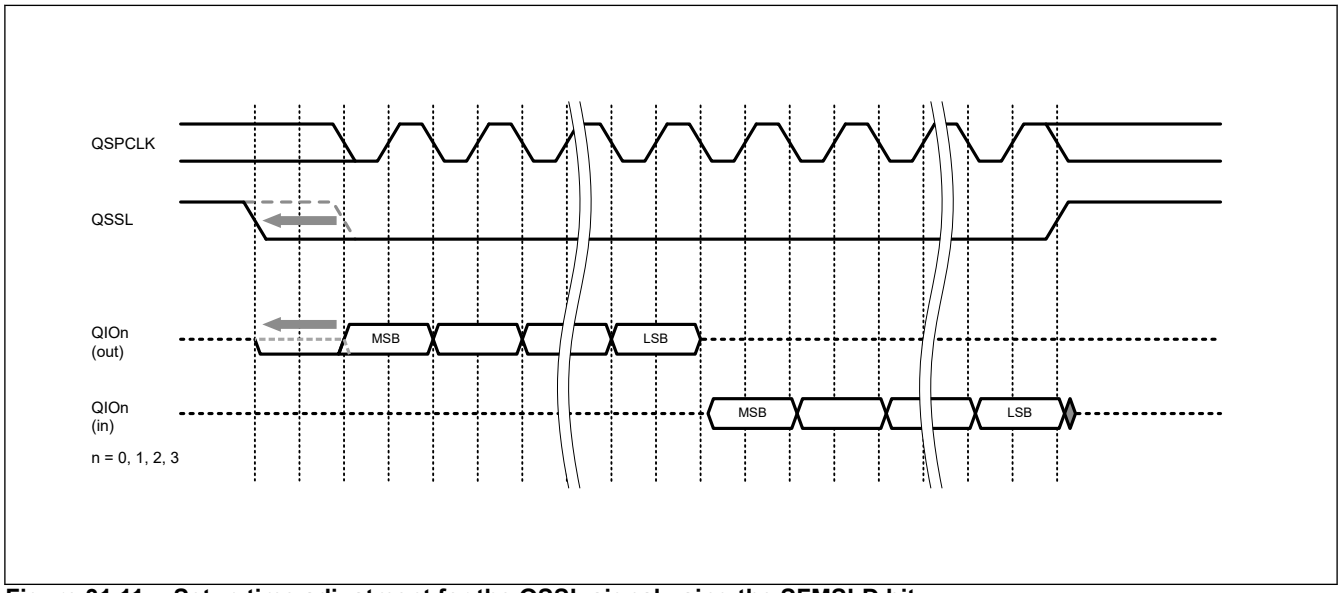



**Figure 31.11 Setup time adjustment for the QSSL signal using the SFMSLD bit**

### 31.5.5 QSSL Signal Hold Time

The QSSL signal hold time that the serial flash memory requires until the QSSL signal is driven high after the last rising edge of the QSPCLK signal can be configured. The hold time can be selected as 0.5 or 1.5 cycles of QSPCLK in the SFMSHD bit of the SFMSSC register.

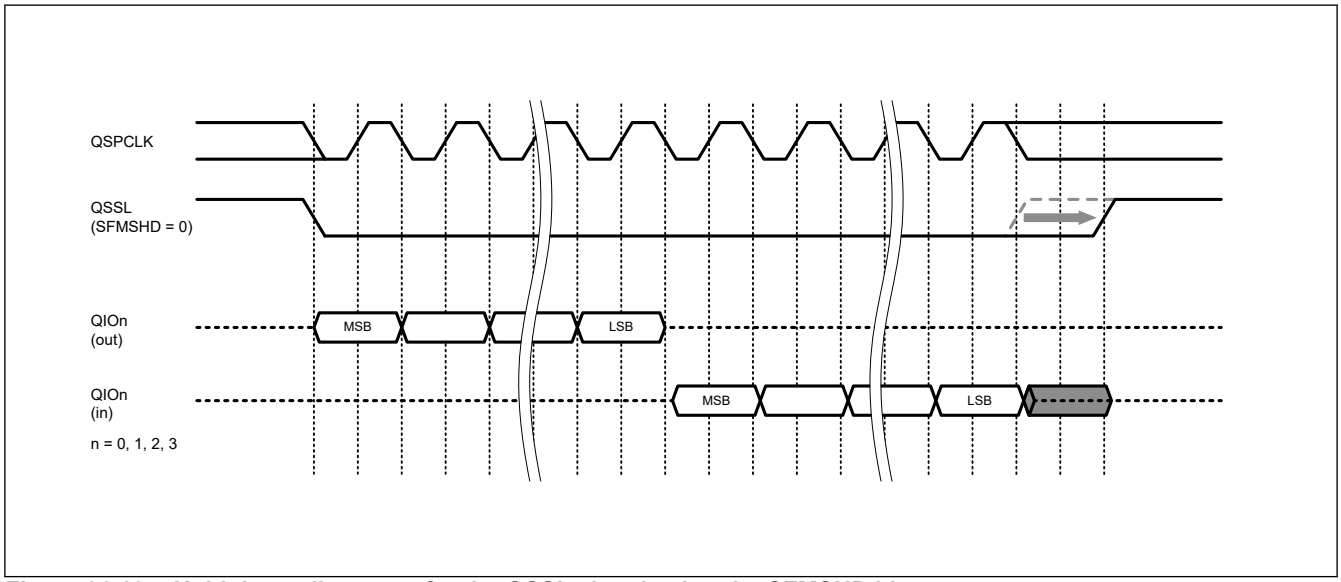



**Figure 31.12 Hold time adjustment for the QSSL signal using the SFMSHD bit**

### 31.5.6 Hold Time of the Serial Data Output Enable

The buffer output enable of the QIO0, QIO1, QIO2, or QIO3 pin can be extended by 1 QSPCLK using the SFMOEX bit in the SFMSMD register.

For a standard read instruction, it is extended immediately after an address code. For other read instructions, it is extended after two cycles of mode data (XIP mode control) of the serial flash memory in dummy cycles.

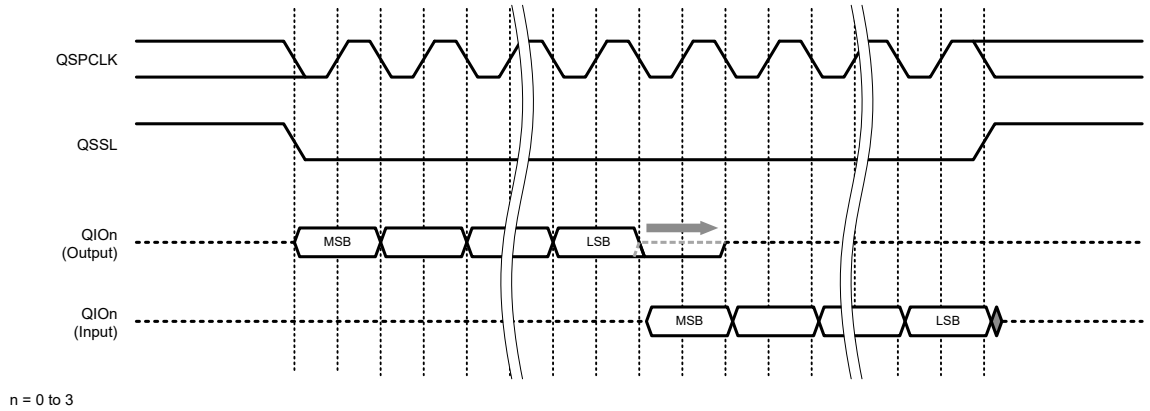



**Figure 31.13 Hold time adjustment for output enable using the SFMOEX bit (Standard Read)**

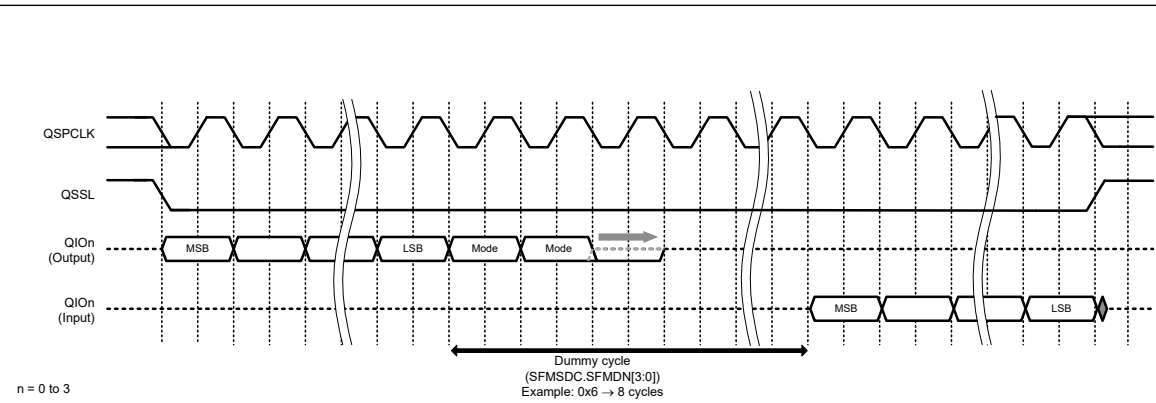



**Figure 31.14 Hold Time Adjustment of Output Enabling Using the SFMOEX Bit (Fast Read)**

## 31.5.7 Setup Time for Serial Data Output

When a command or address is transmitted to the serial flash memory, the setup time begins on serial data output and ends when the QSPCLK signal rises. If this setup time is insufficient, it can be extended by 1 PCLKA using the SFMOSW bit in the SFMSMD register. When SFMOSW is 1, the low-level width of QSPCLK during serial data transmission is extended by 1 PCLKA while data is being output from the QSPI. This function has no effect on serial data reception.

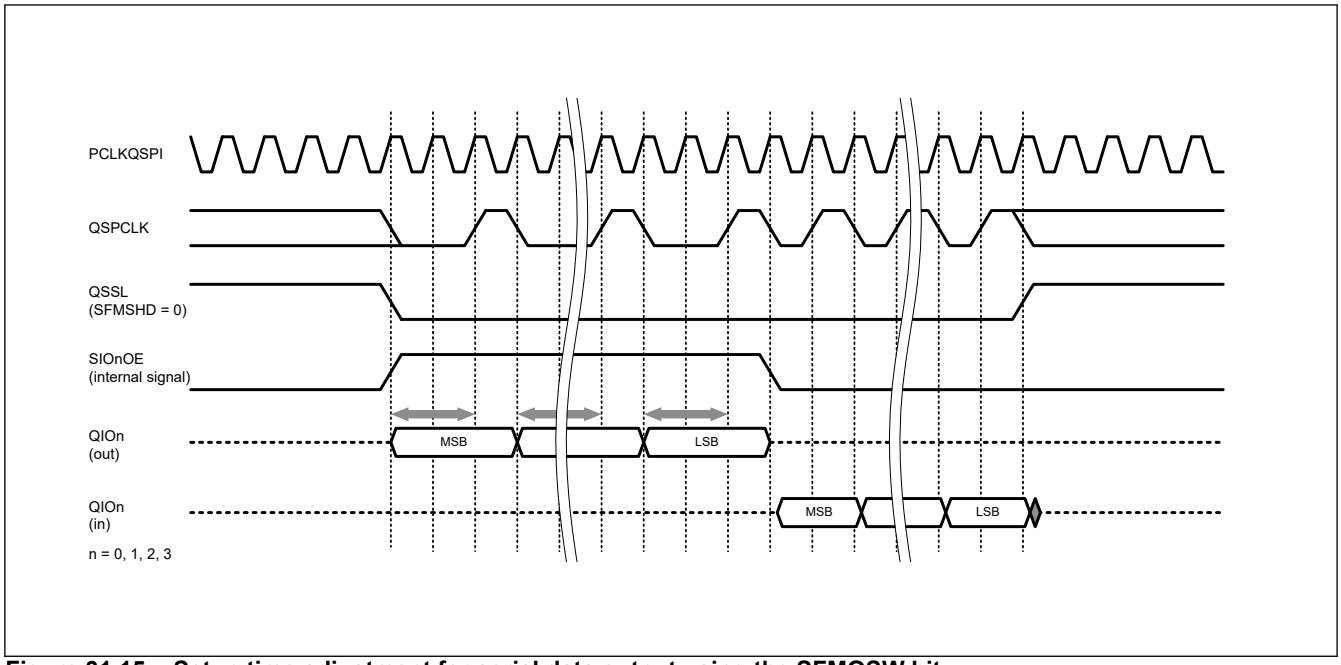



**Figure 31.15 Setup time adjustment for serial data output using the SFMOSW bit**

## 31.5.8 Hold Time for Serial Data Output

When a command or address is transmitted to the serial flash memory, the hold time begins on the rising edge of QSPCLK and ends when the serial data makes another transmission. If this hold time is insufficient, it can be extended by 1 PCLKA using the SFMOHW bit in the SFMSMD register. When SFMOHW is 1, the high-level width of QSPCLK during serial data transmission is extended by 1 PCLKA while data is being output from the QSPI. This function has no effect on serial data reception.

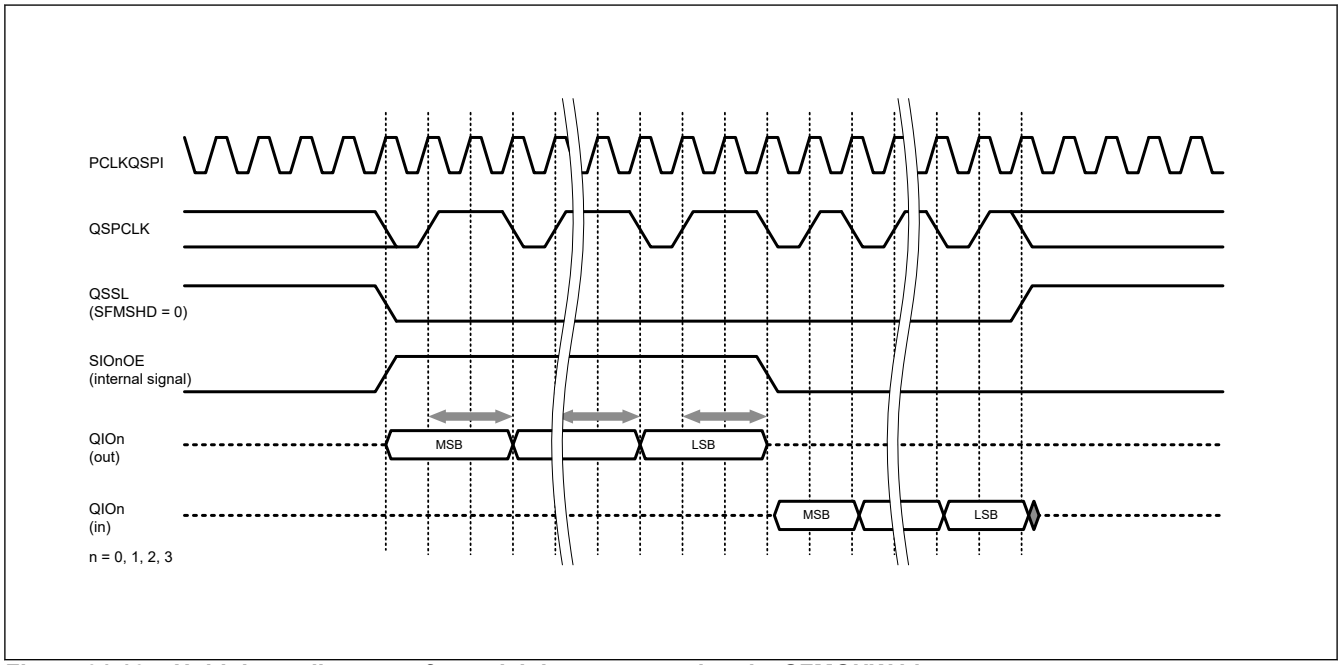



**Figure 31.16 Hold time adjustment for serial data output using the SFMOHW bit**

## 31.5.9 Serial Data Receiving Latency

The serial flash outputs data in synchronization with the falling edge of the QSPCLK signal. The QSPI receives that data in synchronization with the falling edge of the subsequent QSPCLK signal. The delay from when the serial flash starts outputting data until the QSPI receives that data is called the receiving latency. The QSPI adds a latency adjustment cycle immediately before the first data reception cycle in the SPI bus cycle. From the serial flash side, this is seen as an increase

in the number of data reception cycles. This added latency adjustment cycle is not generated in the SPI bus cycle without accompanying data reception.

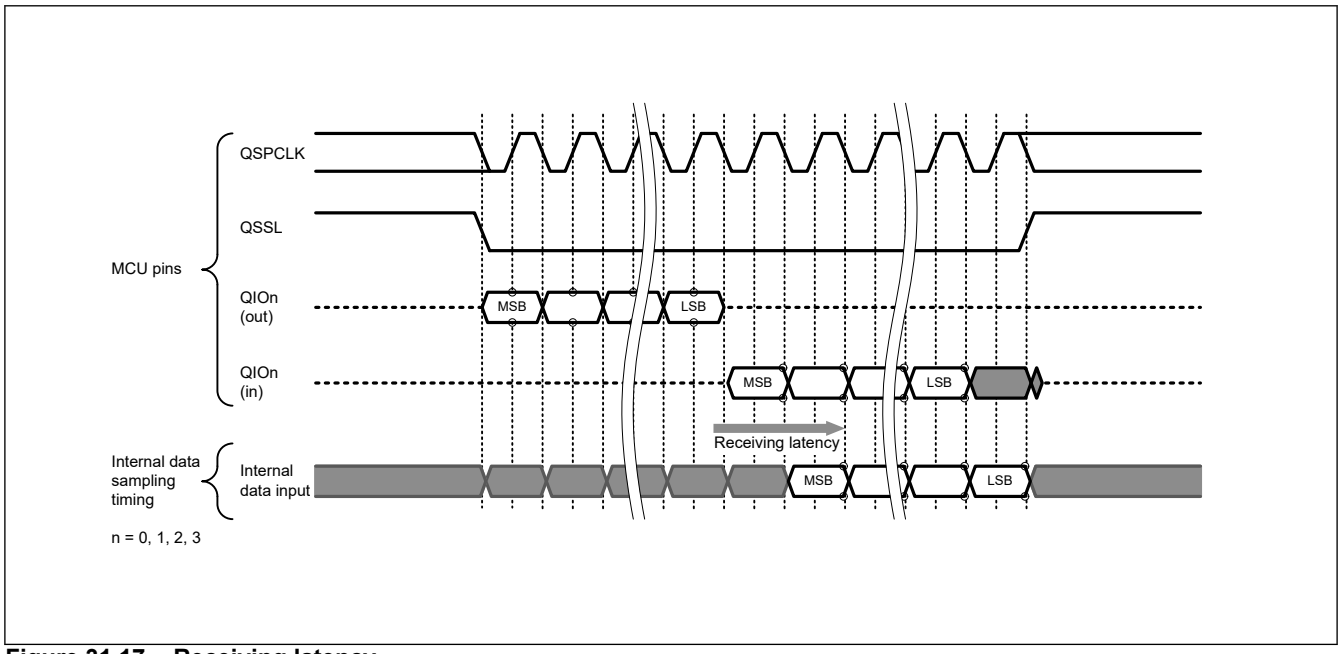



**Figure 31.17 Receiving latency**

## 31.6 SPI Instruction Set Used for Serial Flash Memory Access

### 31.6.1 SPI Instructions That Are Automatically Generated

When the serial flash memory is accessed, an SPI bus cycle using the instructions described in Table 31.6 to Table 31.10 is automatically generated based on the settings in the SFMSAC register and in the SFMSMD register.

**Table 31.6 SPI instructions automatically generated when SFMAS[1:0] = 00b**

| SPI instruction | Instruction code | Address bytes | Dummy cycles | Data bytes | Remarks |
|---|---|---|---|---|---|
| Standard Read | 0x03*1 | 1 | — | 1 to ∞ | SFMRM[2:0] = 000b, A8 = 0 |
| | 0x0B*1 | 1 | — | 1 to ∞ | SFMRM[2:0] = 000b, A8 = 1 |

Note 1. If the SFMSMD.SFMCCE bit is set to 1, the SFMCIC[7:0] bits in the Instruction Code Register (SFMSIC) setting is used as an instruction code.

**Table 31.7 SPI instructions automatically generated when SFMAS[1:0] = 01b**

| SPI instruction | Instruction code | Address bytes | Dummy cycles | Data bytes | Remarks |
|---|---|---|---|---|---|
| Standard Read | 0x03*1 | 2 | — | 1 to ∞ | SFMRM[2:0] = 000b |

Note 1. If the SFMSMD.SFMCCE bit is set to 1, the SFMSIC.SFMCIC[7:0] setting is used as an instruction code.

**Table 31.8 SPI instructions automatically generated when SFMAS[1:0] = 10b (1 of 2)**

| SPI instruction | Instruction code | Address bytes | Dummy cycles | Data bytes | Remarks |
|---|---|---|---|---|---|
| Standard Read | 0x03*1 | 3 | — | 1 to ∞ | SFMRM[2:0] = 000b |
| Fast Read | 0x0B*1 | 3 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 001b |
| Fast Read Dual Output | 0x3B*1 | 3 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 010b |
| Fast Read Dual I/O | 0xBB*1 | 3 | 4*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 011b |
| Fast Read Quad Output | 0x6B*1 | 3 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 100b |

**Table 31.8 SPI instructions automatically generated when SFMAS[1:0] = 10b (2 of 2)**

| SPI instruction | Instruction code | Address bytes | Dummy cycles | Data bytes | Remarks |
|---|---|---|---|---|---|
| Fast Read Quad I/O | 0xEB*1 | 3 | 6*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 101b |

Note 1. If the SFMSMD.SFMCCE bit is set to 1, the SFMSIC.SFMCIC[7:0] setting is used as an instruction code.
Note 2. The number of dummy cycles is configurable by using SFMDN[3:0] bits in the Dummy Cycle Control Register (SFMSDC).

**Table 31.9 SPI instructions automatically generated when SFMAS[1:0] = 11b and SFM4BC = 0**

| SPI instruction | Instruction code | Address bytes | Dummy cycles | Data bytes | Remarks |
|---|---|---|---|---|---|
| Standard Read | 0x03*1 | 4 | — | 1 to ∞ | SFMRM[2:0] = 000b |
| Fast Read | 0x0B*1 | 4 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 001b |
| Fast Read Dual Output | 0x3B*1 | 4 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 010b |
| Fast Read Dual I/O | 0xBB*1 | 4 | 4*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 011b |
| Fast Read Quad Output | 0x6B*1 | 4 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 100b |
| Fast Read Quad I/O | 0xEB*1 | 4 | 6*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 101b |

Note 1. If the SFMSMD.SFMCCE bit is set to 1, the SFMSIC.SFMCIC[7:0] setting is used as an instruction code.
Note 2. The number of dummy cycles is configurable by using the SFMSDC.SFMDN[3:0] bits.

**Table 31.10 SPI instructions automatically generated when SFMAS[1:0] = 11b and SFM4BC = 1**

| SPI instruction | Instruction code | Address bytes | Dummy cycles | Data bytes | Remarks |
|---|---|---|---|---|---|
| Standard Read | 0x13*1 | 4 | — | 1 to ∞ | SFMRM[2:0] = 000b |
| Fast Read | 0x0C*1 | 4 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 001b |
| Fast Read Dual Output | 0x3C*1 | 4 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 010b |
| Fast Read Dual I/O | 0xBC*1 | 4 | 4*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 011b |
| Fast Read Quad Output | 0x6C*1 | 4 | 8*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 100b |
| Fast Read Quad I/O | 0xEC*1 | 4 | 6*2 | 1 to ∞ | Selectable: SFMRM[2:0] = 101b |

Note 1. If the SFMSMD.SFMCCE bit is set to 1, the SFMSIC.SFMCIC[7:0] setting is used as an instruction code.
Note 2. The number of dummy cycles is configurable by using the SFMSDC.SFMDN[3:0] bits..

## 31.6.2 Standard Read Instruction

The standard Read instruction is a common read instruction supported by most serial flash memory. When an SPI bus cycle starts, the QSSL signal (serial flash memory select) is asserted, and the instruction code (0x03 or 0x13)*1 is output. Next, an address with a width of 1 to 4 bytes, specified in the SFMAS[1:0] bits in the SFMSAC register, is transmitted. Data is then received.

This standard Read instruction is selected in the initial QSPI settings.

Note 1. Many 4-KB serial flash memory devices have an address field not larger than 1 byte (A7-A0) to minimize the overhead and to receive A8 information from bit 3 of the Read instruction code. To support these devices, the QSPI only outputs A8 (address bit 8) to bit [3] of the standard Read instruction code when an address width of 1 byte is specified (SFMAS[1:0] = 00). This means that 0x0B might be output instead of 0x03 as the standard Read instruction code. This code duplicates the Fast Read instruction code. However, for most of the 2-KB or smaller serial flash memory, with an address width of 1 byte, bit 3 of a command is designed to be excluded from decoding as a don’t-care bit, so such a Read instruction code is recognized correctly as the standard Read instruction code. In rare cases, some serial flash memory allow bit 3 to be decoded. When such a serial flash memory is connected, configure your application to avoid access resulting in A8 = 1.

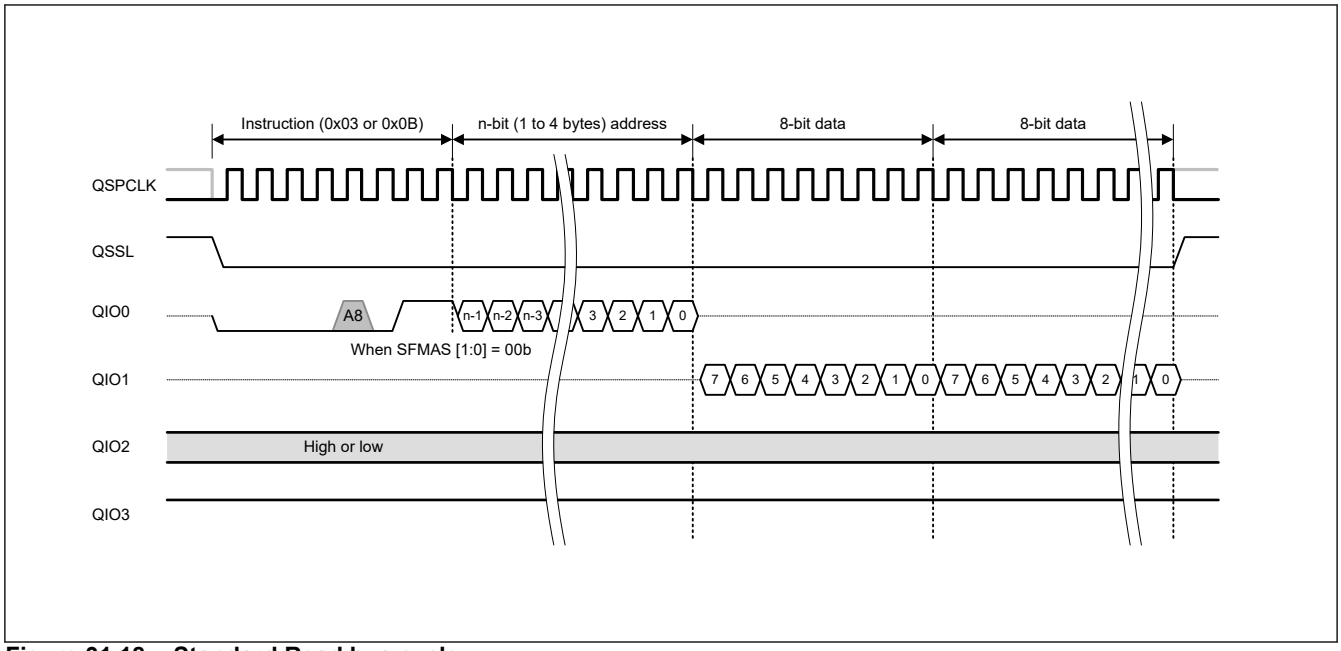



**Figure 31.18 Standard Read bus cycle**

## 31.6.3 Fast Read Instruction

The Fast Read instruction is a read instruction that supports a higher communication clock speed than the standard Read instruction. When an SPI bus cycle starts, the QSSL signal is asserted, and the instruction code (0x0B or 0x0C) is output. Next, an address with a width of 1 to 4 bytes specified by the SFMSAC.SFMAS [1: 0] bits is transmitted, a dummy cycle specified by the SFMSDC register is generated, then the data is received.

The first two dummy cycles are used to select the XIP mode. When the XIP mode is selected, the same instruction used this time is applied to the next SPI bus cycle, and the instruction code is not output the next SPI bus cycle. For details on the XIP mode, see section 31.8. XIP Control.

Switching to the Fast Read instruction is controlled in the SFMSMD register.

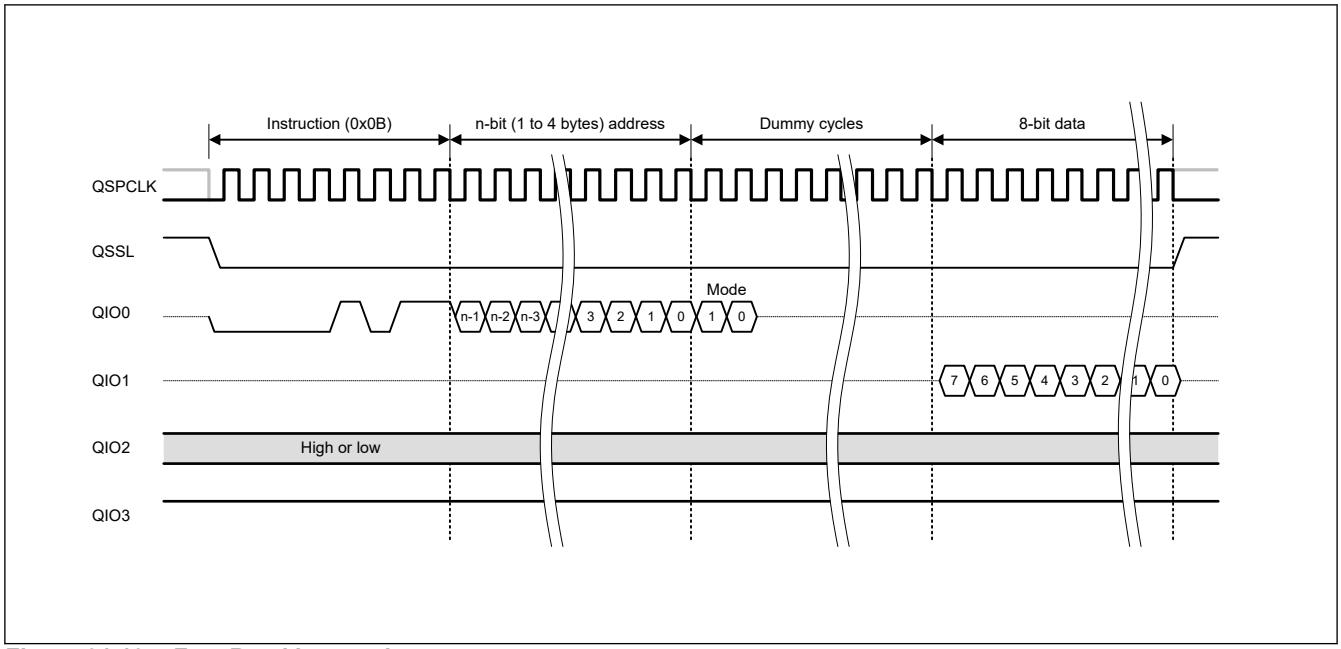



**Figure 31.19 Fast Read bus cycle**

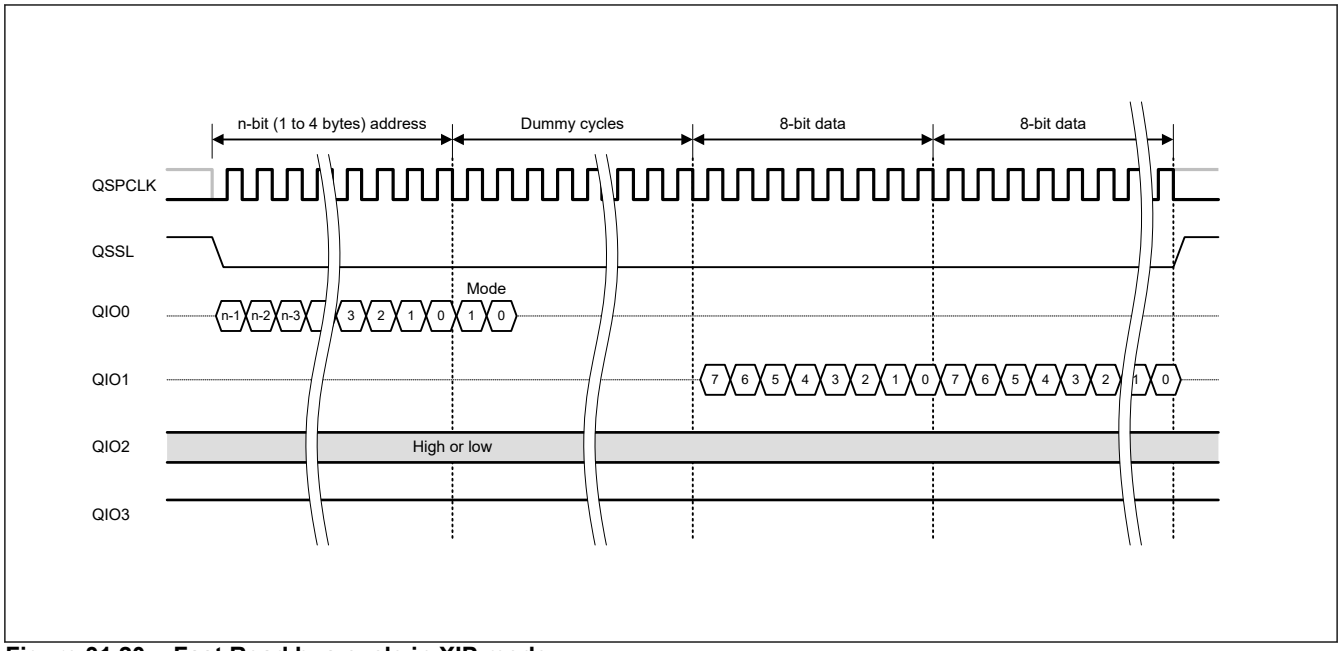



**Figure 31.20 Fast Read bus cycle in XIP mode**

Note: To use the Fast Read instruction, a serial flash memory that supports Fast Read transfers is required.

## 31.6.4 Fast Read Dual Output Instruction

The Fast Read Dual Output instruction is a read instruction that uses two signal lines to receive data. When the SPI bus cycle starts, the QSSL signal is asserted. The instruction code (0x3B or 0x3C) and an address with a width of 1 to 4 bytes, specified by the SFMSAC.SFMAS [1: 0] bits are transmitted from the QIO0 pin in the extended SPI protocol, and transmitted from the QIO0 and QIO1 pins in the Dual-SPI protocol. Next, a certain number of dummy cycles, specified in the SFMSDC register, is generated. Data is then received through the QIO0 and QIO1 pins. Even bit data is received from the QIO0 pin and odd bit data is received from the QIO1 pin.

The first two dummy cycles are used to select the XIP mode. When the XIP mode is selected, the same instruction used this time is applied to the next SPI bus cycle, and the instruction code is not output the next SPI bus cycle. For details on the XIP mode, see section 31.8. XIP Control.

Switching to Fast Read Dual Output is controlled in the SFMSMD register.

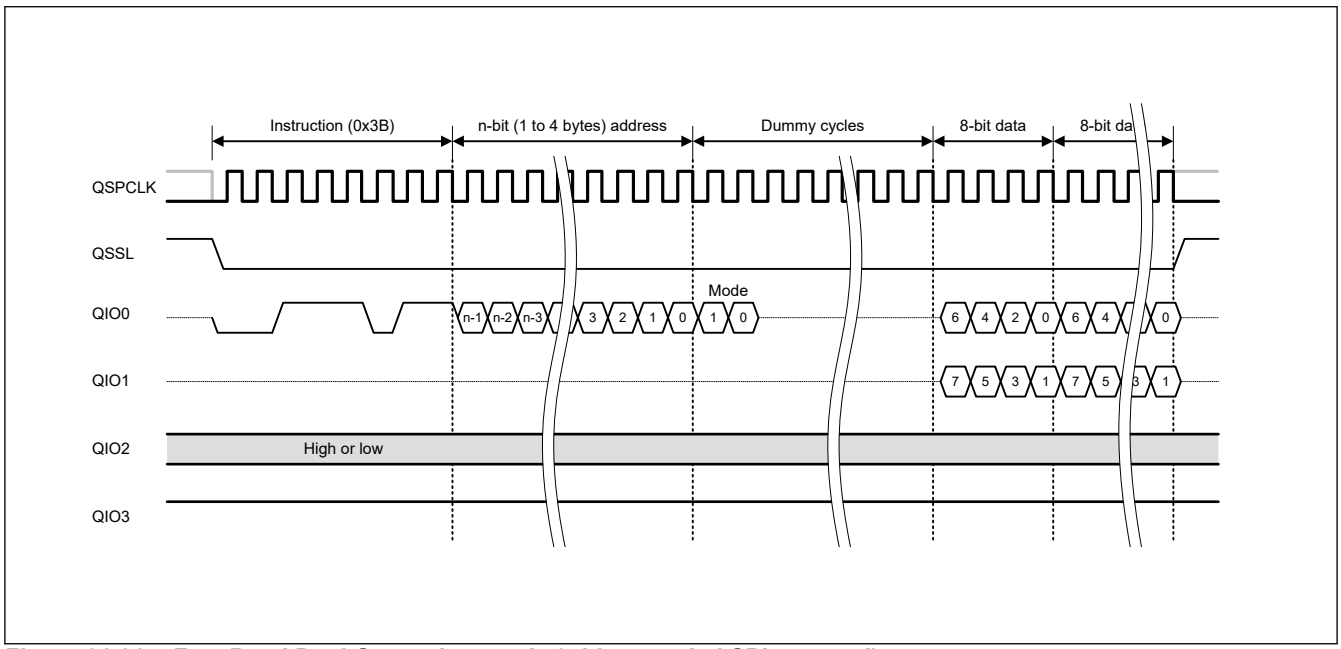



**Figure 31.21 Fast Read Dual Output bus cycle (with extended SPI protocol)**

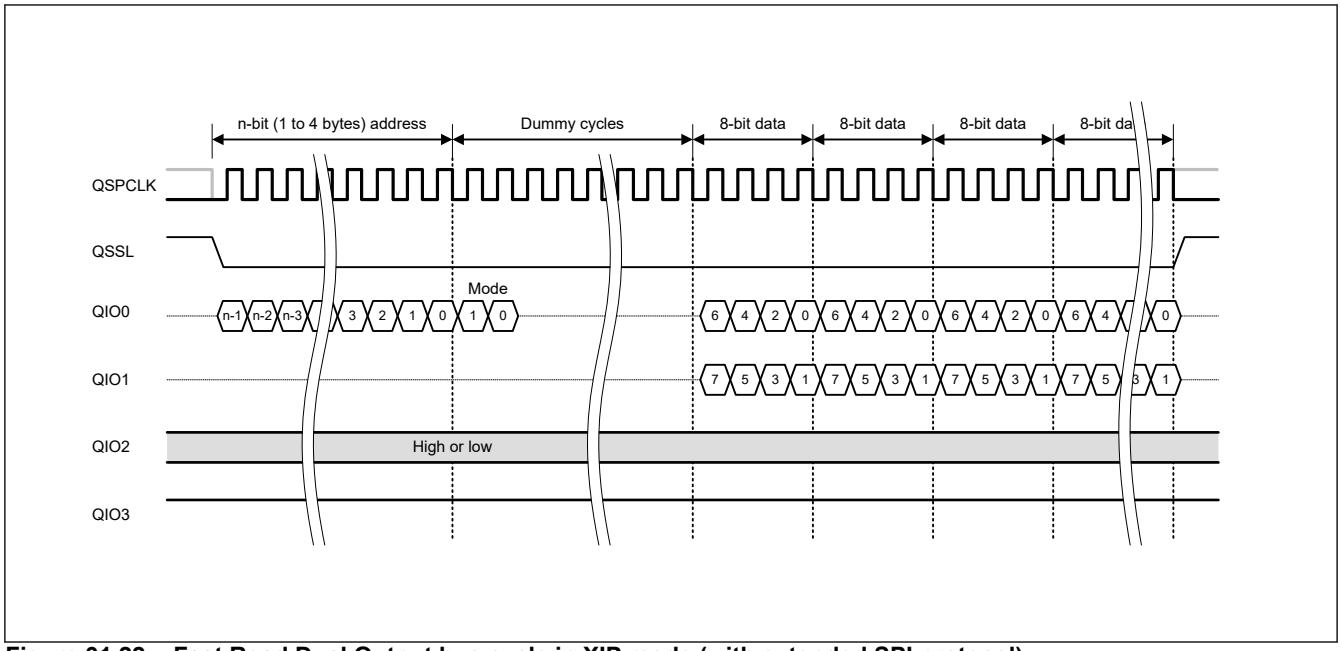



**Figure 31.22 Fast Read Dual Output bus cycle in XIP mode (with extended SPI protocol)**

Note: To use the Fast Read Dual Output instruction, a serial flash memory that supports Fast Read Dual Output transfers is required.

### 31.6.5 Fast Read Dual I/O Instruction

The Fast Read Dual I/O instruction is a read instruction that uses two signal lines to transmit an address and receive data. When the SPI bus cycle starts, the QSSL signal is asserted, and the instruction code (0xBB / 0xBC) is transmitted from QIO0 pin in the extended SPI protocol and from QIO0, and QIO1 pins in the Dual SPI protocol. Next, an address with a width of 1 to 4 bytes, specified in the SFMAS[1:0] bits in the SFMSAC register, is transmitted through the QIO0 and QIO1 pins, and a certain number of dummy cycles, specified in the SFMSDC register, is generated. Data is then is received through the QIO0 and QIO1 pins. Address and dummy cycle transmission and data reception are performed through the QIO0 pin for even bits and through the QIO1 pin for odd bits.

The first two dummy cycles are used to select the XIP mode. When the XIP mode is selected, the same instruction used this time is applied to the next SPI bus cycle, and the instruction code is not output the next SPI bus cycle. For details on the XIP mode, see section 31.8. XIP Control.

Switching to Fast Read Dual I/O is controlled in the SFMSMD register.

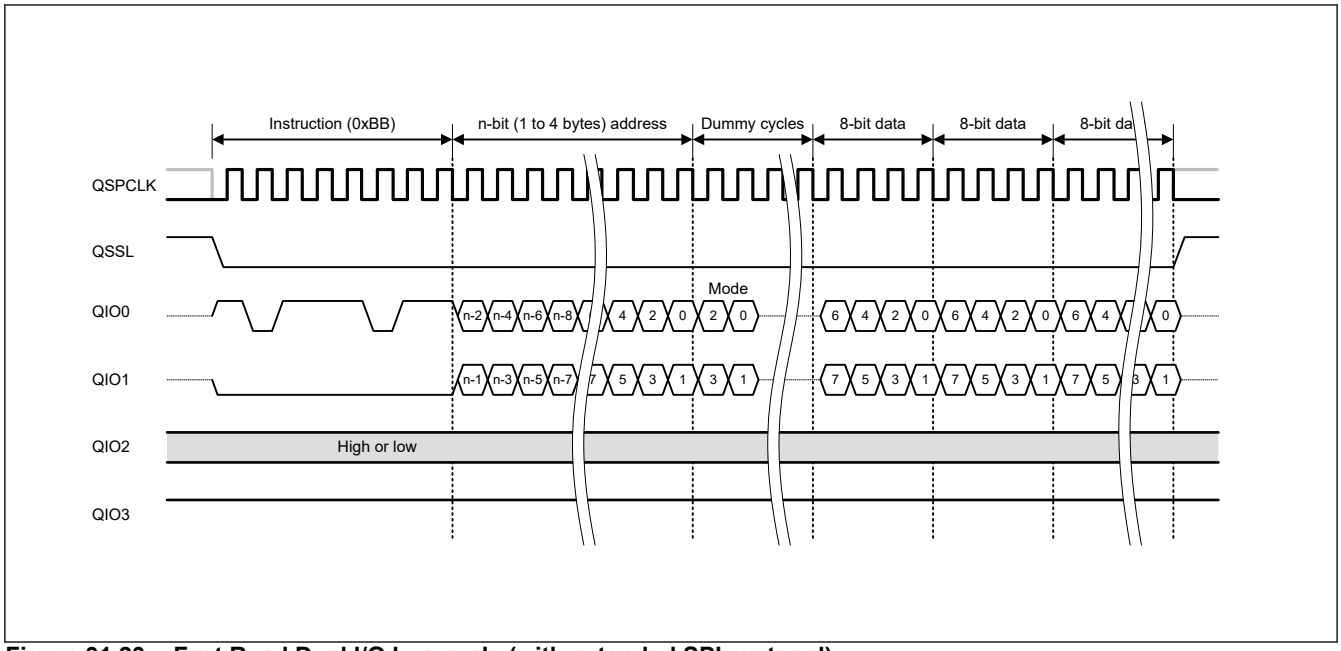



**Figure 31.23 Fast Read Dual I/O bus cycle (with extended SPI protocol)**

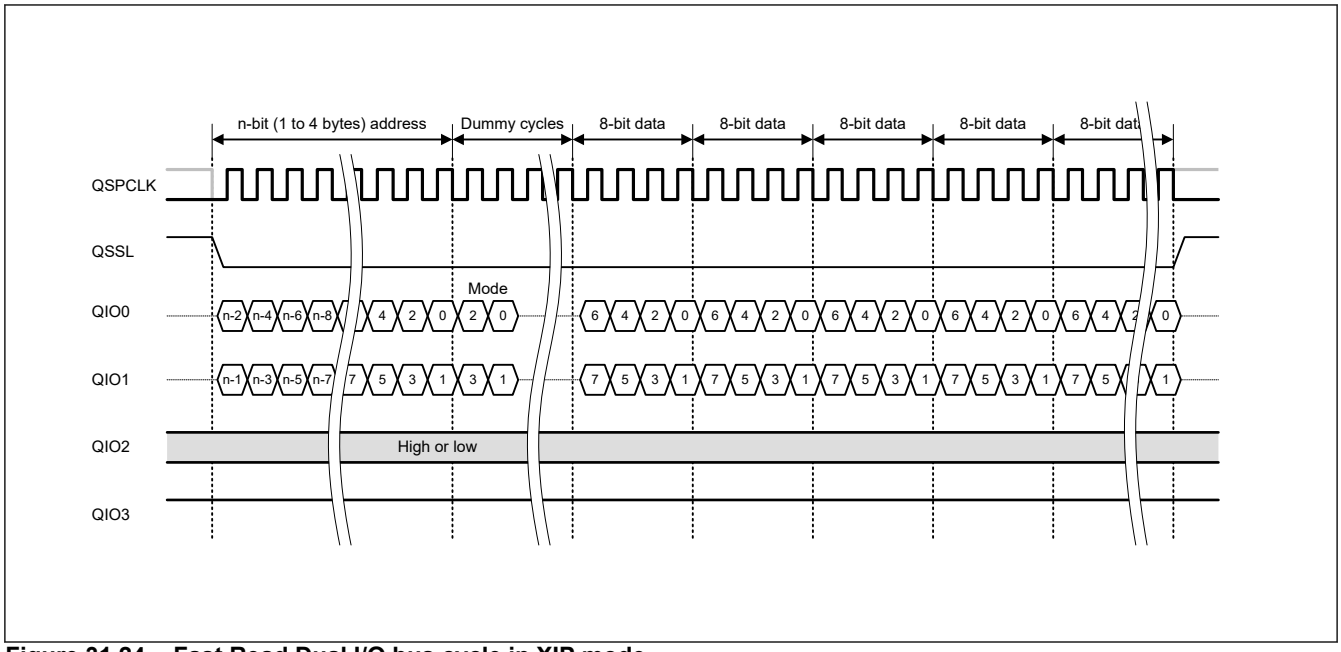



**Figure 31.24 Fast Read Dual I/O bus cycle in XIP mode**

Note: To use the Fast Read Dual I/O instruction, a serial flash memory that supports Fast Read Dual I/O transfers is required.

## 31.6.6 Fast Read Quad Output Instruction

The Fast Read Quad Output instruction is a read instruction that uses four signal lines to receive data. When the SPI bus cycle starts, the QSSL signal is asserted. The instruction code (0x6B or 0x6C) and an address with a width of 1 to 4 bytes, specified in the SFMAS[1:0] bits in the SFMSAC register, are output from the QIO0 pin. Next, a certain number of dummy cycles, specified in the SFMDN[3:0] bits in the SFMSDC register, are generated. Data is then received through the QIO0, QIO1, QIO2, and QIO3 pins.

The first two dummy cycles are used to select the XIP mode. When the XIP mode is selected, the same instruction used this time is applied to the next SPI bus cycle, and the instruction code is not output the next SPI bus cycle. For details on the XIP mode, see section 31.8. XIP Control.

Switching to Fast Read Quad Output is controlled in the SFMSMD register.

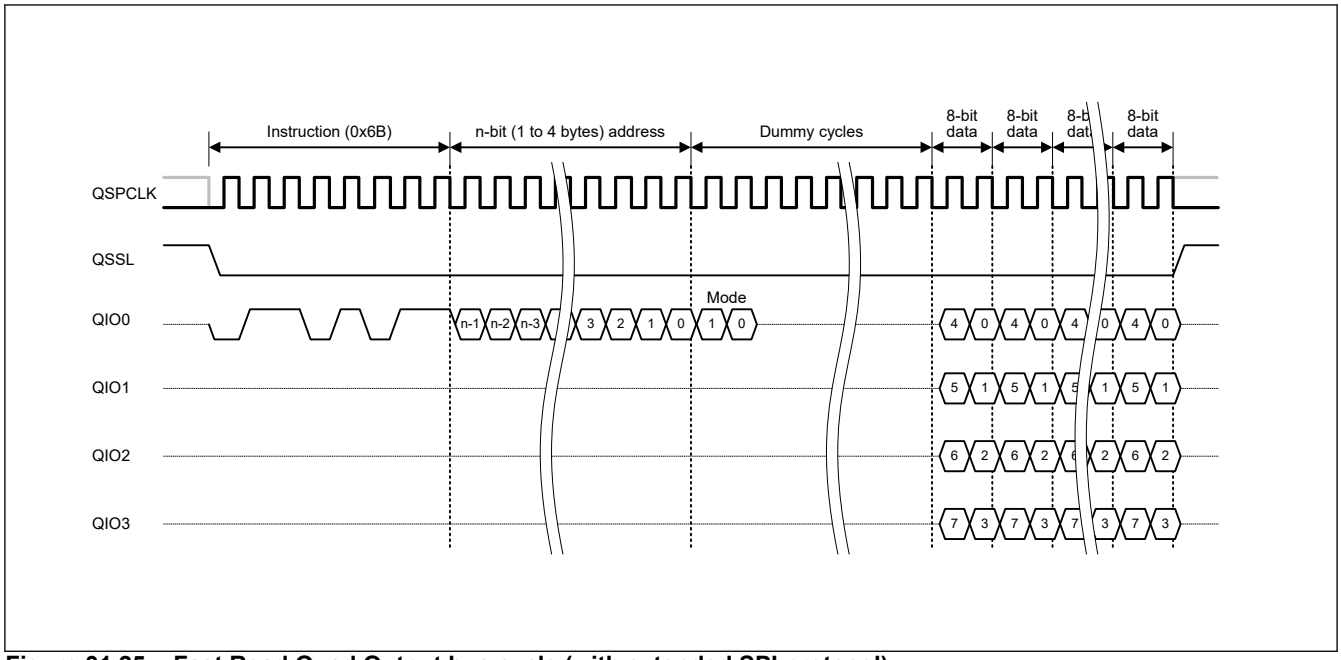



**Figure 31.25 Fast Read Quad Output bus cycle (with extended SPI protocol)**

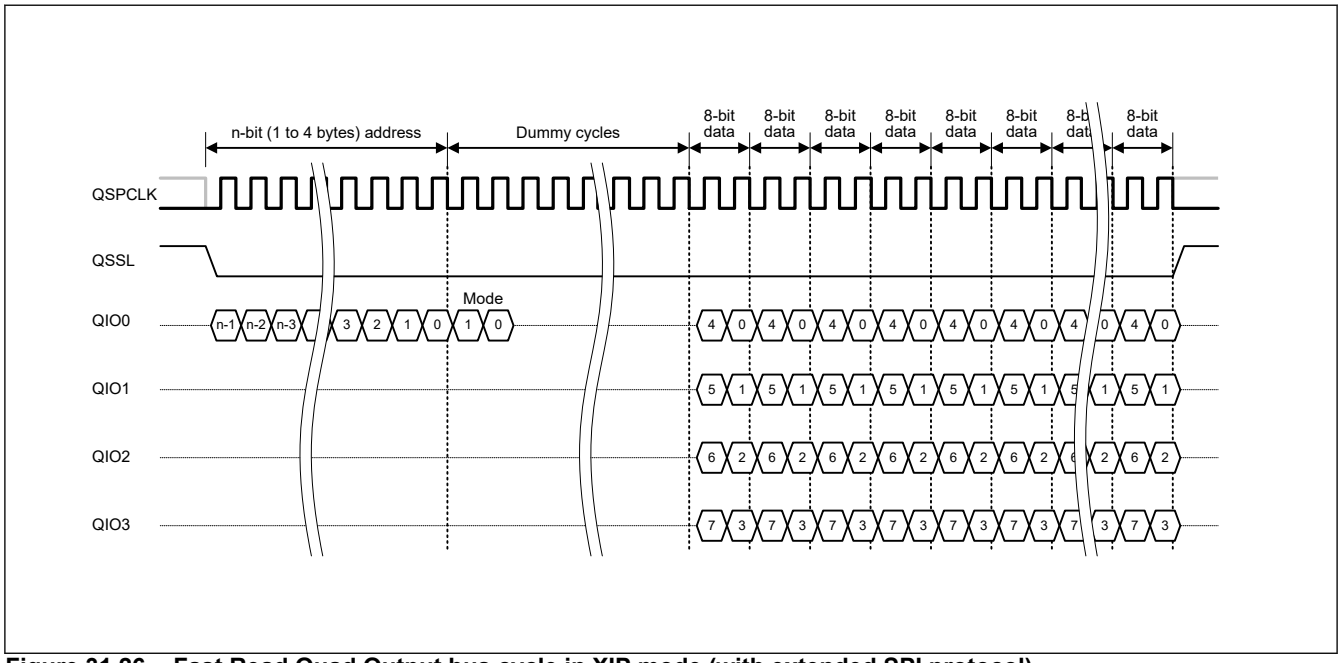



**Figure 31.26 Fast Read Quad Output bus cycle in XIP mode (with extended SPI protocol)**

Note: To use Fast Read Quad Output, a serial flash memory that supports Fast Read Quad Output transfer is required.

## 31.6.7 Fast Read Quad I/O Instruction

The Fast Read Quad I/O instruction is a read instruction that uses four signal lines to transmit an address and receive data. When the SPI bus cycle starts, the QSSL signal is asserted, and the instruction code (0xEB / 0xEC) is transmitted from QIO0 pin in the extended SPI protocol and from QIO0, QIO1, QIO2, and QIO3 pins in the Quad-SPI protocol. Next, an address with a width of 1 to 4 bytes, specified in the SFMAS[1:0] bits in the SFMSAC register, is transmitted through the QIO0, QIO1, QIO2, and QIO3 pins, and a certain number of dummy cycles, specified in the SFMDN[3:0] bits in the SFMSDC register, is generated. Data is then received through the QIO0, QIO1, QIO2, and QIO3 pins.

The first two dummy cycles are used to select the XIP mode. When the XIP mode is selected, the same instruction used this time is applied to the next SPI bus cycle, and the instruction code is not output the next SPI bus cycle. For details on the XIP mode, see section 31.8. XIP Control.

Switching to Fast Read Quad I/O is controlled in the SFMSMD register.

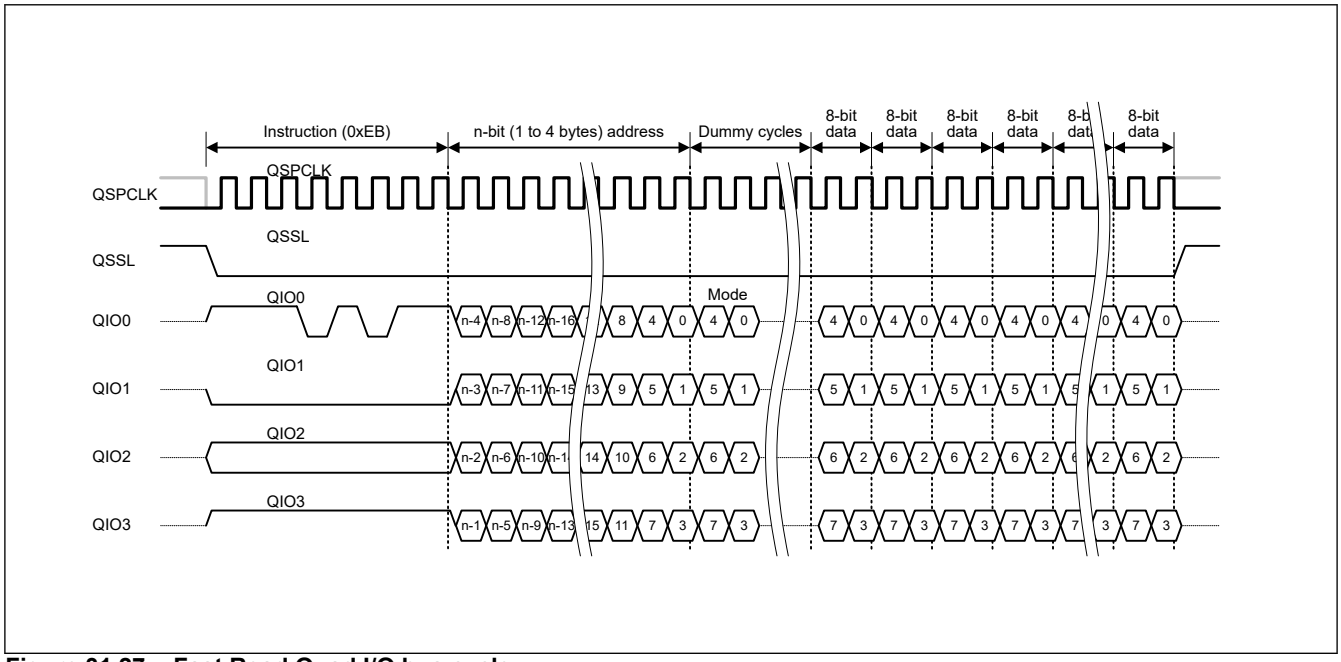



**Figure 31.27 Fast Read Quad I/O bus cycle**

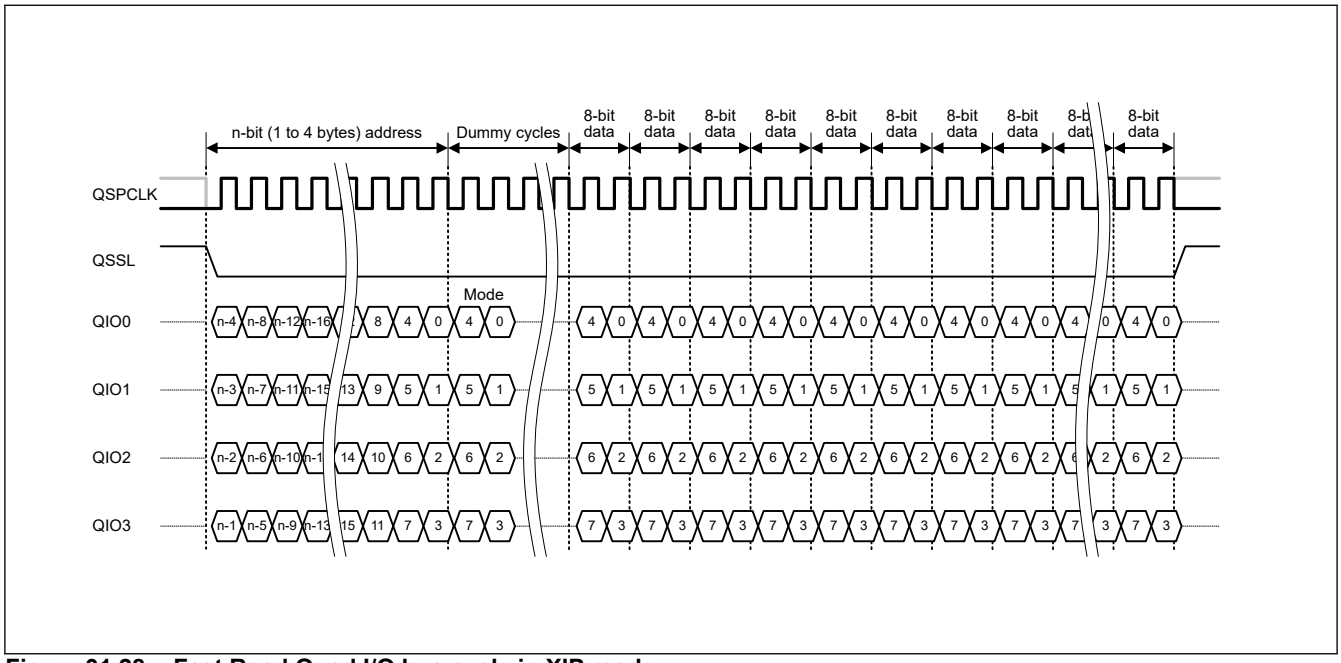



**Figure 31.28 Fast Read Quad I/O bus cycle in XIP mode**

Note: To use the Fast Read Quad I/O instruction, a serial flash memory that supports Fast Read Quad I/O transfers is required.

## 31.6.8 Enter 4-Byte Mode Instruction

The Enter 4-Byte Mode instruction sets the serial flash address width to 4 bytes. When the SPI bus cycle starts, the serial flash select signal is asserted, and the instruction code (0xB7) is output.

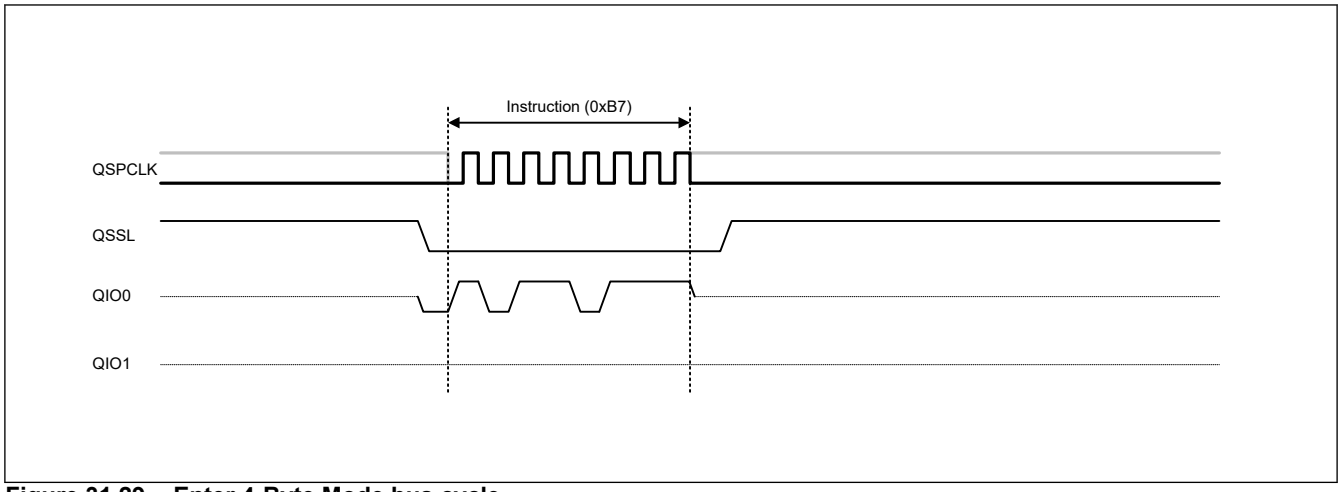



**Figure 31.29 Enter 4-Byte Mode bus cycle**

Note: The Enter 4-Byte Mode instruction is issued regardless of whether the serial flash is in 3- or 4-byte mode.

### 31.6.9 Exit 4-Byte Mode Instruction

The Exit 4-Byte Mode instruction sets the serial flash address width to 3 bytes. When the SPI bus cycle starts, the serial flash select signal is asserted, and the instruction code (0xE9) is output.

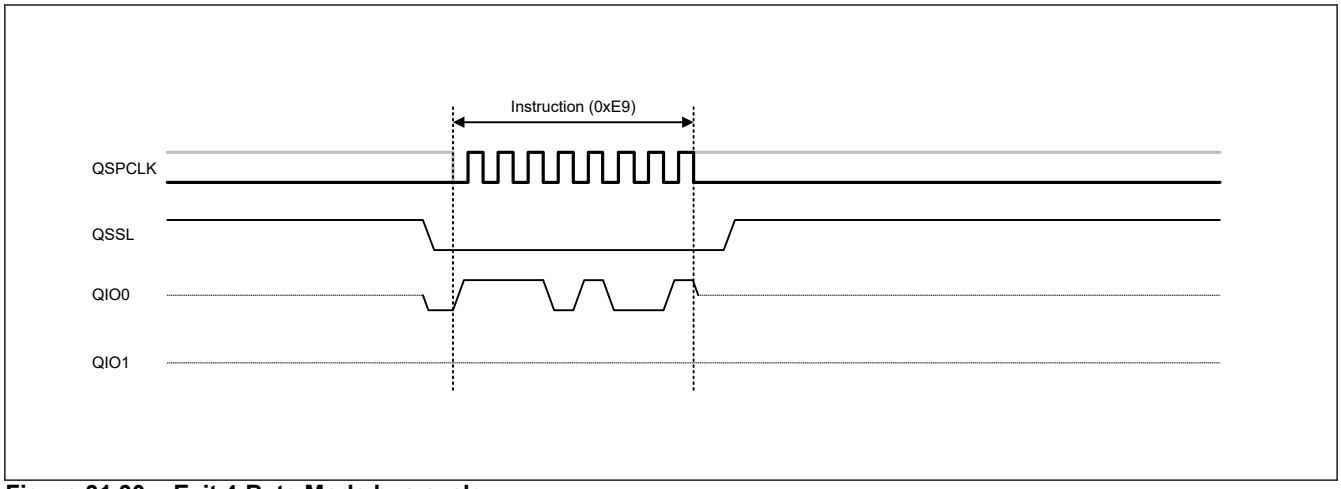



**Figure 31.30 Exit 4-Byte Mode bus cycle**

Note: The Exit 4-Byte Mode instruction is issued regardless of whether the serial flash is in 3- or 4-byte mode.

### 31.6.10 Write Enable Instruction

The Write Enable instruction enables changing of the serial flash address width. When the SPI bus cycle starts, the serial flash select signal is asserted, and the instruction code (0x06) is output.

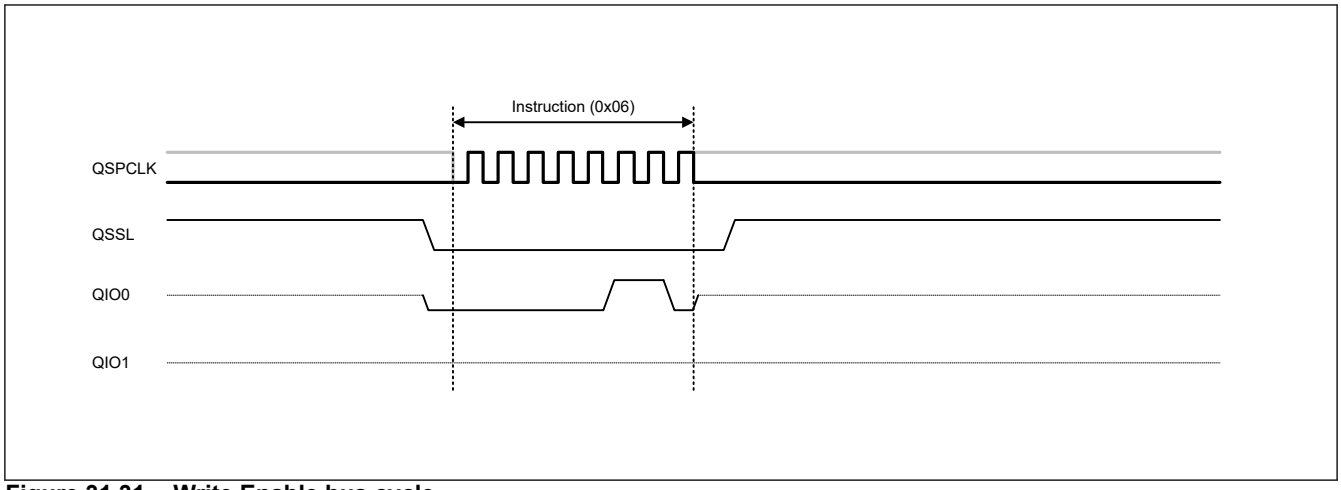



**Figure 31.31 Write Enable bus cycle**

## 31.7 SPI Bus Cycle Arrangement

### 31.7.1 Serial Flash Memory Read Based on Individual Conversion

ROM read bus cycles are individually converted to SPI bus cycles on a one-to-one basis. When a ROM read bus cycle is detected, the QSSL signal is asserted, and an SPI bus cycle starts. When the data receiving is finished from the serial flash memory, the QSSL signal is negated and the SPI bus cycle is complete.

When the next ROM read bus cycle is detected, the QSSL signal set by the SFMSSC.SFMSW[3: 0] bits is asserted again, then the next SPI bus cycle starts.

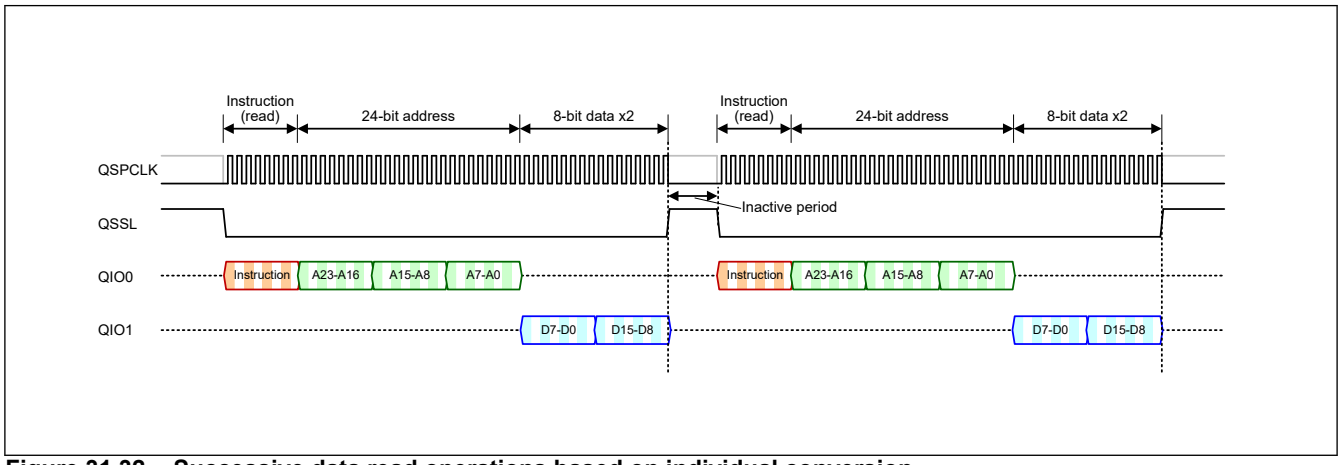



**Figure 31.32 Successive data read operations based on individual conversion**

### 31.7.2 Serial Flash Memory Read Using the Prefetch Function

In operations such as CPU instruction execution and block data transfer, data is often read in ascending order from contiguous addresses. Serial flash memory provides the ability to repeat data reception without reissuing an instruction code and address. To work with this function, the QSPI has a prefetch function for continuous data reception. However, if the CPU issues a flash read request for discontinuous flash addresses, SPI bus cycles are separated from each other, disabling the prefetch function.

To enable the prefetch function of the QSPI, set the SFMPFE bit in the SFMSMD register to 1. When the prefetch function is enabled, data is received continuously and stored in the prefetch buffer of the QSPI, without waiting for another flash read request. When the CPU issues a flash read request, an address check is made. If an address match is confirmed, the data in the buffer is passed to the CPU. If an address mismatch is detected, the data in the buffer is discarded and a new SPI bus cycle is issued.

The buffer for prefetching is 18 bytes long. When this buffer is full, the SPI bus cycle is ended. When the buffer data is read to create free space, a new SPI bus cycle is automatically started to resume prefetching.

The prefetch function allows for efficient transfer operations when data is read in ascending order from contiguous addresses, as in instruction fetch and block data transfer.

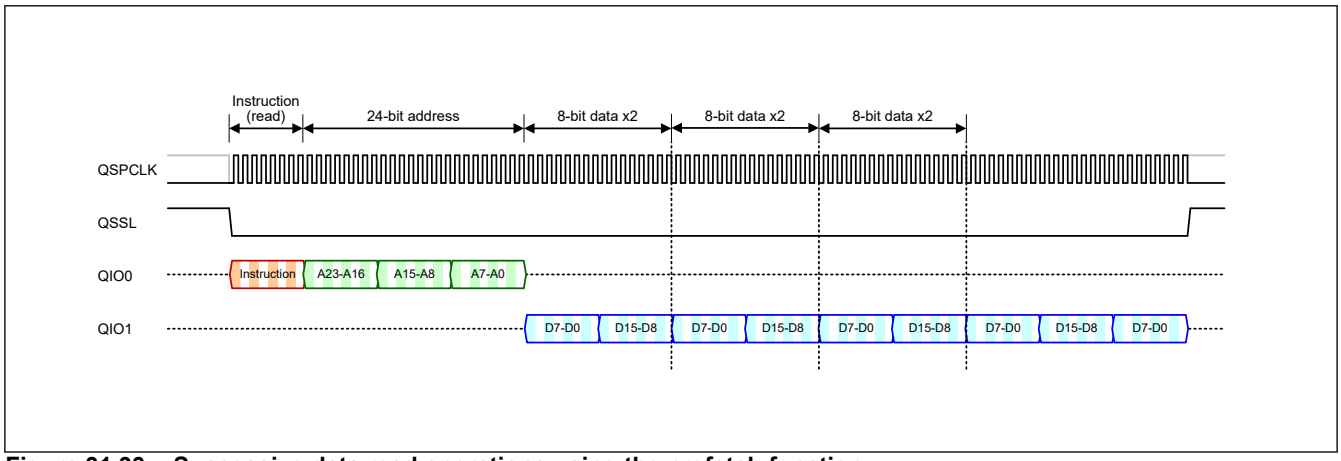



**Figure 31.33 Successive data read operations using the prefetch function**

## 31.7.3 Halt of Prefetching

If a serial flash memory read request for discontinuous addresses is issued when continuous data is being received by the prefetch function, the transfer of continuous data being made is halted and a new SPI bus cycle is started. Usually, such a halt of serial transfer occurs on data reception byte boundaries. However, if the SFMPAE bit in the SFMSMD register is set to 1, the halt can occur on locations other than byte boundaries.

## 31.7.4 Direct Specification of Prefetch Destination

When the prefetch function is enabled (SFMSMD.SFMPFE = 1), when writing to the QSPI window area occurs, after the writing is completed, prefetching starts from the write start address. Writes to serial flash memory cannot be performed.

Combining this function with described in section 31.7.5. Prefetch State Polling, can reduce the load on the internal bus when data is read from a low-speed serial flash memory.

Note: Writing to the QSPI window area with a data size of 2 bytes or more causes a hardfault.

## 31.7.5 Prefetch State Polling

A read by CPU from a low-speed serial flash memory causes the CPU system bus to be occupied until completion of the SPI reception bus cycle. The prefetch state polling function is provided to reduce this load.

The PFOFF bit in the Status Register (SFMSST) register indicates the state of the prefetch function, and the PFCNT[4:0] bits in the SFMSST register indicate the number of data bytes already prefetched. Place the polling program in the SRAM of this device.

```
//
// copy 1K byte (32bit x 256 word) data from serial flash to internal SRAM
//
unsigned long *sptr; // pointer for the serial flash
unsigned long *dptr; // pointer for the destination
int i;

SFMSMD |= 0x0040; // set SFMPFE bit to enable prefetch
*( (volatile unsigned char *) sptr ) = 0; // make the TAG valid to start prefetch

for ( i = 0 ; i < 256 ; i++ ){
while ( ( SFMSST & 0x00FF ) < 0x04 ){}; // waiting for 4-byte data to be received
*(dptr++) = *(sptr++);
}
```

Note: When executing a polling program, place the program outside of the serial flash memory. If the polling program is executed when the program is placed on the serial flash memory, the prefetch target frequently switches to an instruction code. This eliminates the effect of polling, and an infinite loop can result because the prefetch buffer is not filled.

### 31.7.6 SPI Bus Cycle Extension Function

If the SFMSE[1:0] bits in the SFMSMD register are set to a value other than 00b, the QSPI waits for the next flash read. At this time, the QSPCLK signal stops, the QSSL signal is kept active low even after data is obtained from the serial flash memory, and the SPI bus cycle is suspended.

If the address of the next flash read is contiguous in ascending order, the toggling of the QSPCLK signal is restarted to continue reception of subsequent data. If the address of the next flash read is not contiguous in ascending order, the QSSL signal is driven high once to end the SPI bus cycle being suspended. A new SPI bus cycle is then started.

When data is read intermittently from ascending order contiguous addresses, this function enables an efficient transfer operation to be performed by reducing the overhead for instruction code and address transmission.

The SPI bus cycle extension time is selectable in the SFMSE[1:0] bits in the SFMSMD register. When the specified extension time elapses, the QSSL signal returns to the high level to automatically end the SPI bus cycle being suspended. If the SFMSE[1:0] bits are set to 11b, QSSL is extended infinitely. This increases the power consumption of the serial flash memory.

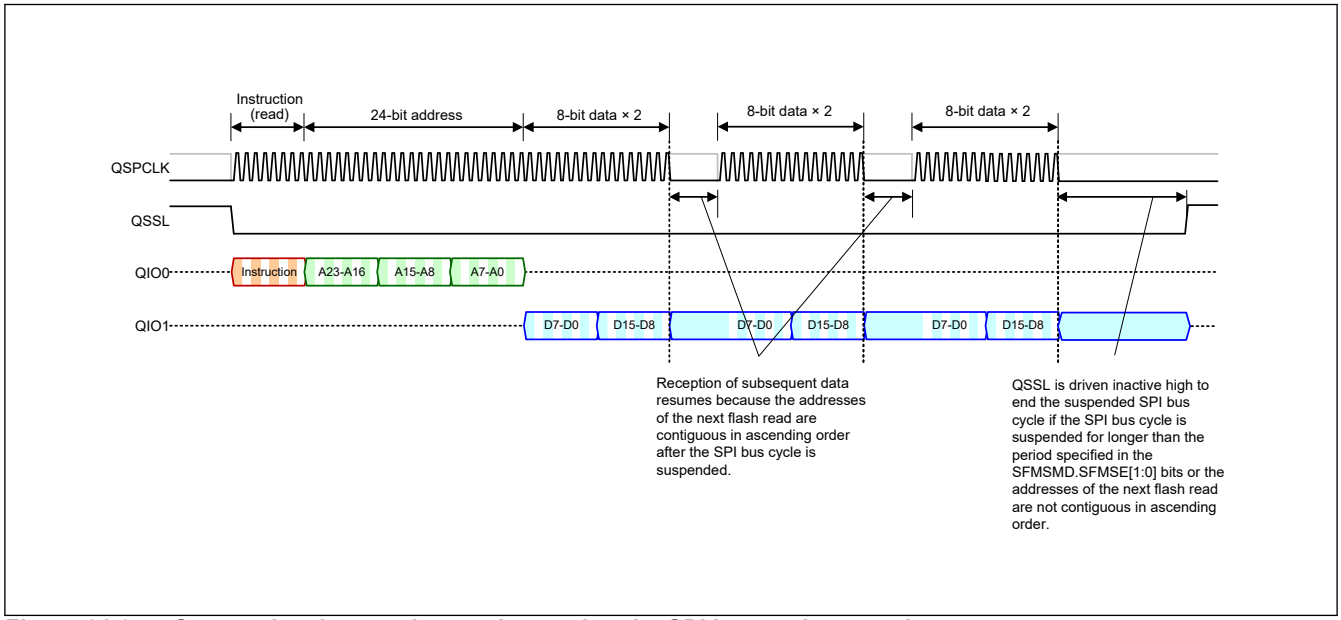



**Figure 31.34 Successive data read operations using the SPI bus cycle extension**

## 31.8 XIP Control

Some serial flash memory devices allow latencies to be reduced by skipping instruction code reception for flash reads. This instruction code skip function is selected in mode data received during the dummy cycle period of the previous serial bus cycle.

In the dummy cycle of the Fast Read instructions, the QSPI controls the XIP mode of the serial flash memory by using the serial data signal to send the mode data set in the SFMXD[7:0] bits in the SFMSDC register during the first 2 cycles, as shown in Figure 31.35.

The mode data to enable the XIP mode differs for each serial flash memory. Accordingly, set the appropriate mode data in the SFMXD[7:0] bits.

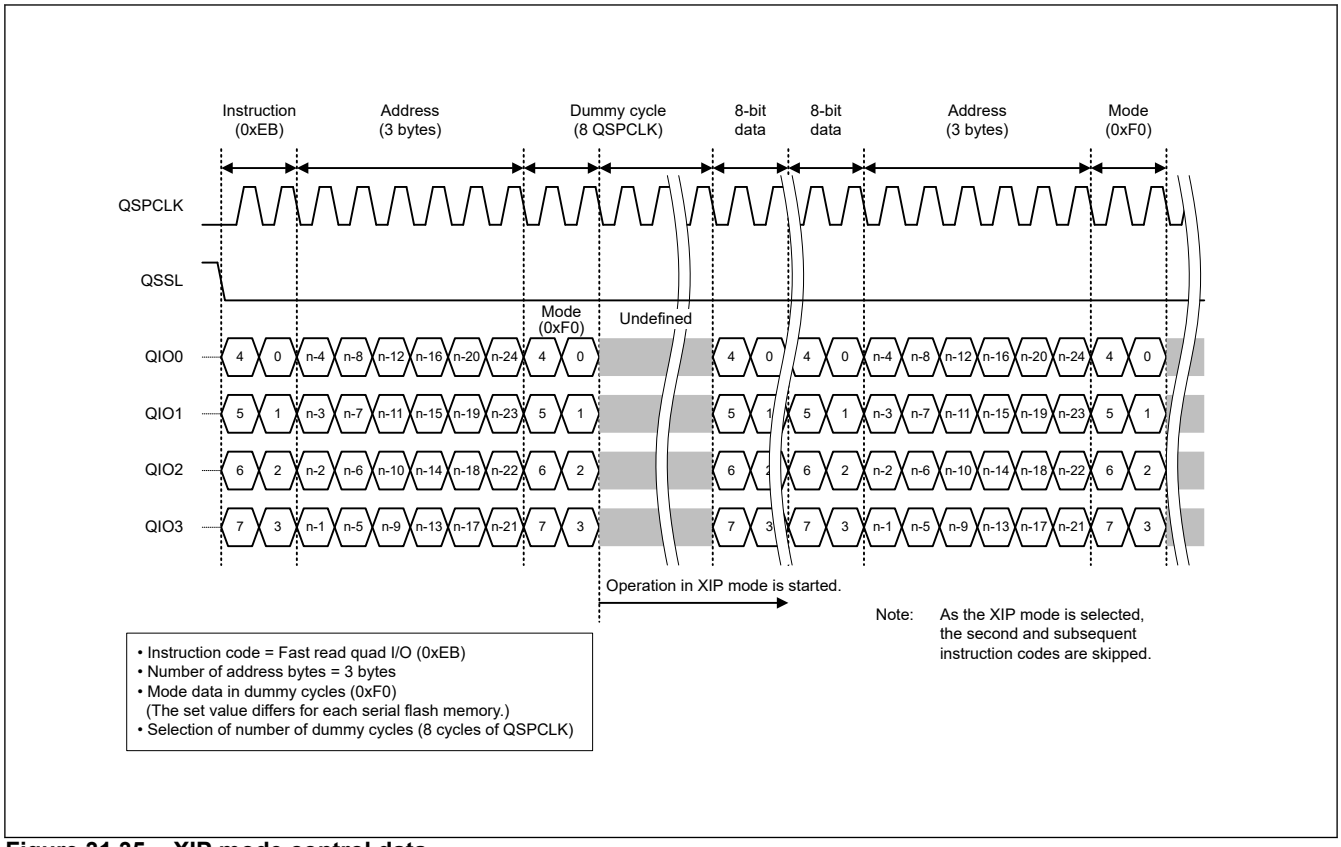



**Figure 31.35 XIP mode control data**

## 31.8.1 Setting XIP Mode

To start XIP mode in serial flash memory, perform the following register settings:

- Set a mode data value in the SFMXD[7:0] bits in the SFMSDC register.*1
- Set the SFMXEN bit in the SFMSDC register to 1.

In the dummy cycle of the first fast read cycle after these registers are set, the mode data value set in the register is transferred. From that point, XIP mode is enabled in the serial flash memory. To confirm the current XIP mode status, read the SFMXST flag in the SFMSDC register.

Note 1. In the SFMXD[7:0] bits in the SFMSDC register, set the mode data that follows the specifications for the actual serial flash memory.

The following figure shows an example of the XIP mode setting procedure.

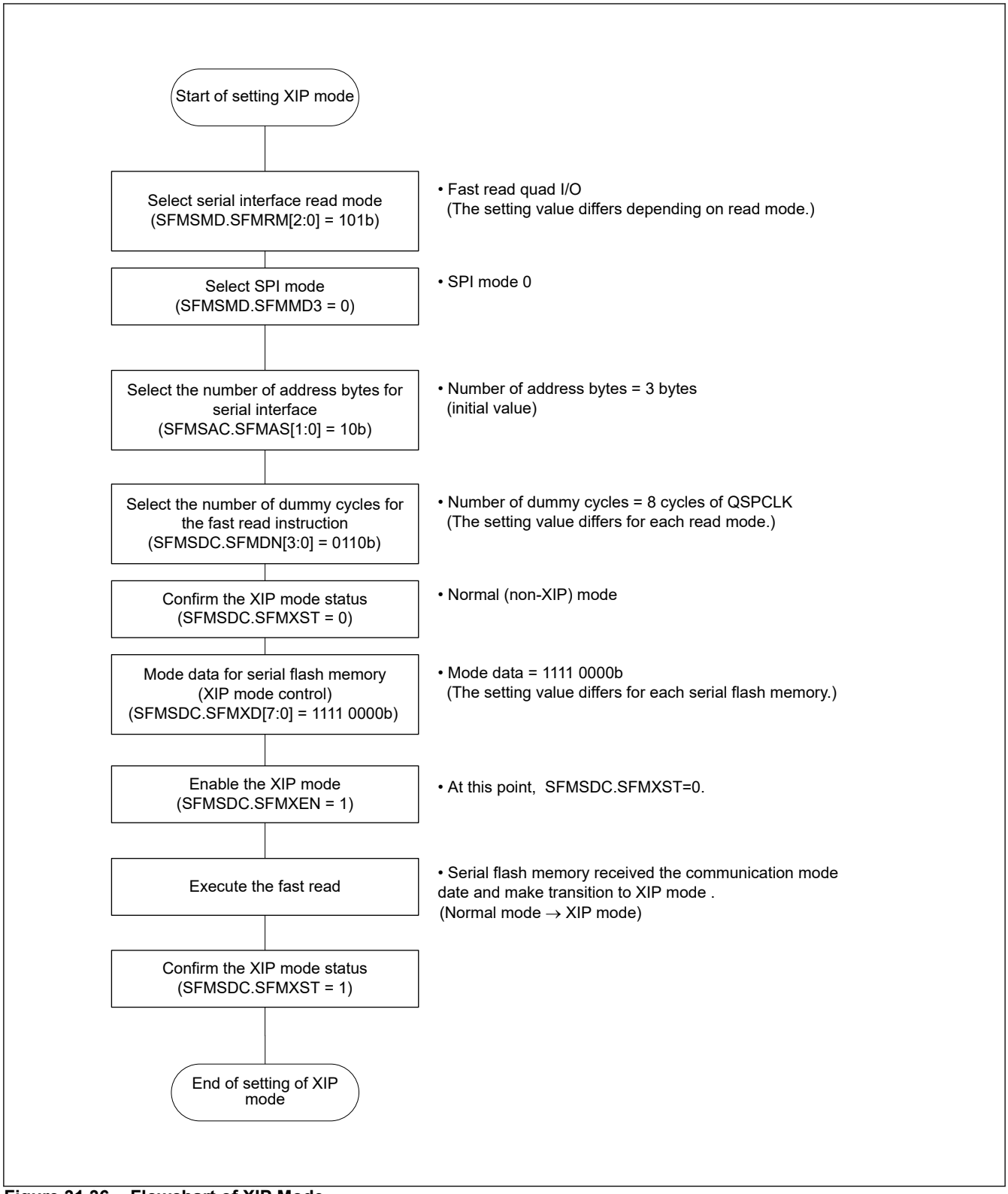



**Figure 31.36 Flowchart of XIP Mode**

## 31.8.2 Releasing the XIP Mode

To release XIP mode in serial flash memory, perform the following register setting:

- Set the mode data value to disable XIP mode in SFMSDC.SFMXD [7: 0] bits*1
- Set the SFMXEN bit in the SFMSDC register to 0.

In the dummy cycle of the first fast read cycle after this register is set, The mode data value that disables the XIP mode set in the register is transferred. From that point, XIP mode is disabled in the serial flash memory. To confirm the current XIP mode status, read the SFMXST flag in the SFMSDC register.

Note 1. Set the mode data in the SFMSDC.SFMXD [7: 0] bits according to the specifications of the serial flash memory.

Figure 31.37 shows an example of the procedure for releasing XIP mode.

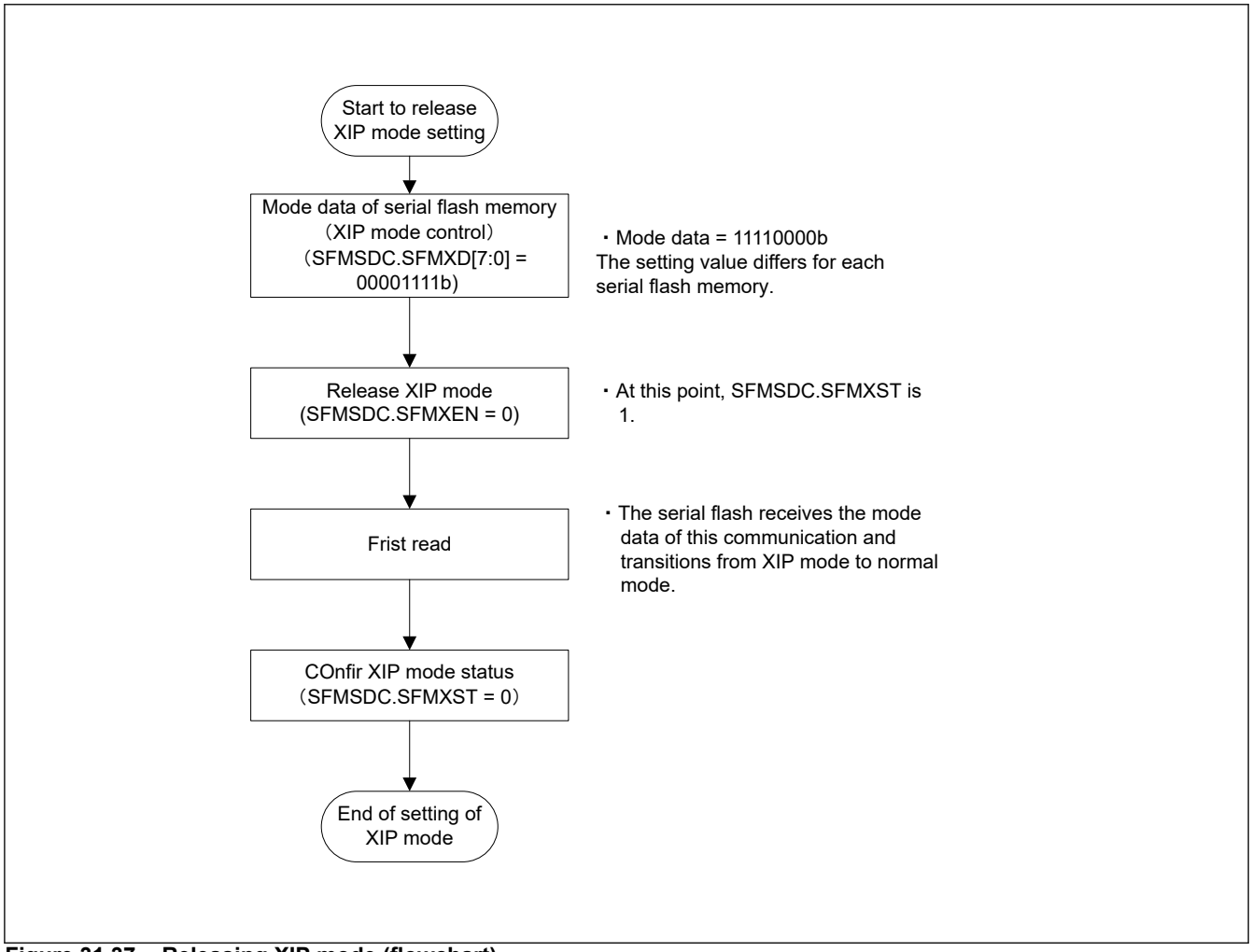



**Figure 31.37 Releasing XIP mode (flowchart)**

## 31.9 QIO2 and QIO3 Pin States

The QIO2 and QIO3 pin states depend on the serial interface read mode specified in the SFMRM[2:0] bits in the SFMSMD register.

**Table 31.11 QIO2 and QIO3 pin states (1 of 2)**

| SFMSMD.SFMRM[2:0] bits | QIO2 pin state*1 | QIO3 pin state*2 | Remarks |
|---|---|---|---|
| 111 | Setting prohibited | | |
| 110 | | | |
| 101 | Input or output as serial data signal (The pin is in the Hi-Z state when it is inactive.) | Input or output as serial data signal (The pin is in the Hi-Z state when it is inactive.) | Fast Read Quad I/O |
| 100 | | | Fast Read Quad Output |

**Table 31.11 QIO2 and QIO3 pin states (2 of 2)**

| SFMSMD.SFMRM[2:0] bits | QIO2 pin state*1 | QIO3 pin state*2 | Remarks |
|---|---|---|---|
| 011 | Output SFMWPL bit variable of the Port Control Register (SFMPMD) (initial value is low level) | Output high level | Fast Read Dual I/O |
| 010 | | | Fast Read Dual Output |
| 001 | | | Fast Read |
| 000 | | | Standard Read (Initial State) |

Note 1. The serial flash memory can also use the QIO2 pin for the write protect (WP) function. The WP function prohibits writes to the status registers. (The function is available in mode other than Quad-SPI mode.)

Note 2. The serial flash memory can also use the QIO3 pin for the HOLD or RESET function.
The hold function places the I/O pin in an inactive state without deselecting the chip. (The function is available in mode other than Quad-SPI mode.)
The reset function resets the serial flash memory. (The function is available when the QSSL pin function is disabled or in a mode in which the QIO3 pin is not used.)

## 31.10 Direct Communication Mode

### 31.10.1 About Direct Communication

The QSPI can read the serial flash memory contents by automatically converting from reading the QSPI window area to SPI bus cycles. However, serial flash memory have many different functions in addition to memory data read, including ID information read, erase, programming, and status information read. There is no standardized instruction set for using these functions, and more functions are being added rapidly by different vendors to different devices. Therefore, to support these functions, the software can create any required SPI bus cycle by communicating directly with serial flash memory.

### 31.10.2 Using Direct Communication Mode

To communicate directly with serial flash memory, transition to direct communication mode by setting the DCOM bit in the Communication Mode Control Register (SFMCMD) register to 1. While direct communications mode is selected, the read operation to the serial flash memory by the QSPI window is invalid.

Note: If the QSPI is set to the XIP mode, you must terminate the XIP mode before starting direct communication mode.

### 31.10.3 Generating the SPI Bus Cycle during Direct Communication

The SPI bus cycle in direct communications starts on the first access to the SFMCOM register, and after a series of input / output operations are executed via the SFMCOM register, the bus cycle ends when 1 is written to the SFMCMD register. At that point, a write to the SFMCOM port is converted to a one-byte transmission to the SPI bus, and a read from the SFMCOM register is converted to a one-byte reception from the SPI bus.

During the period from the first access to the SFMCOM register to the last write operation to the SFMCMD register, the QSSL signal is held active to notify the serial flash memory that a series of SPI bus cycles is in progress.

Note: In direct communication mode, all writes to registers other than SFMCMD and SFMCOM (including SFMSMD, SFMSSC, SFMSKC, SFMSST, SFMCST, SFMSIC, SFMSAC, SFMSDC, SFMSPC, and SFMPMD) are disabled.

Figure 31.38 to Figure 31.40 show direct communication program examples, and Figure 31.41 shows ID read direct communication timing examples.

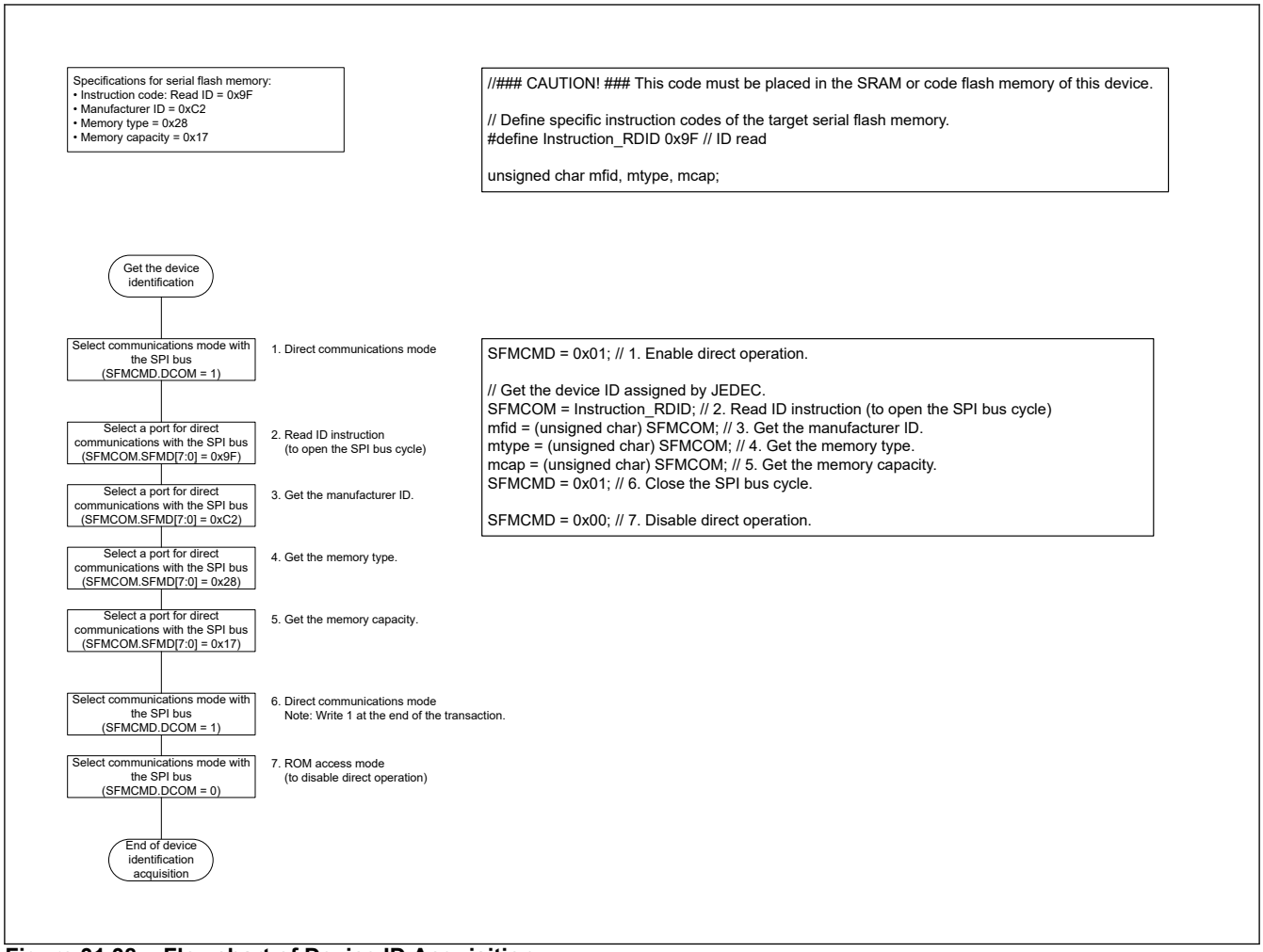



**Figure 31.38 Flowchart of Device ID Acquisition**

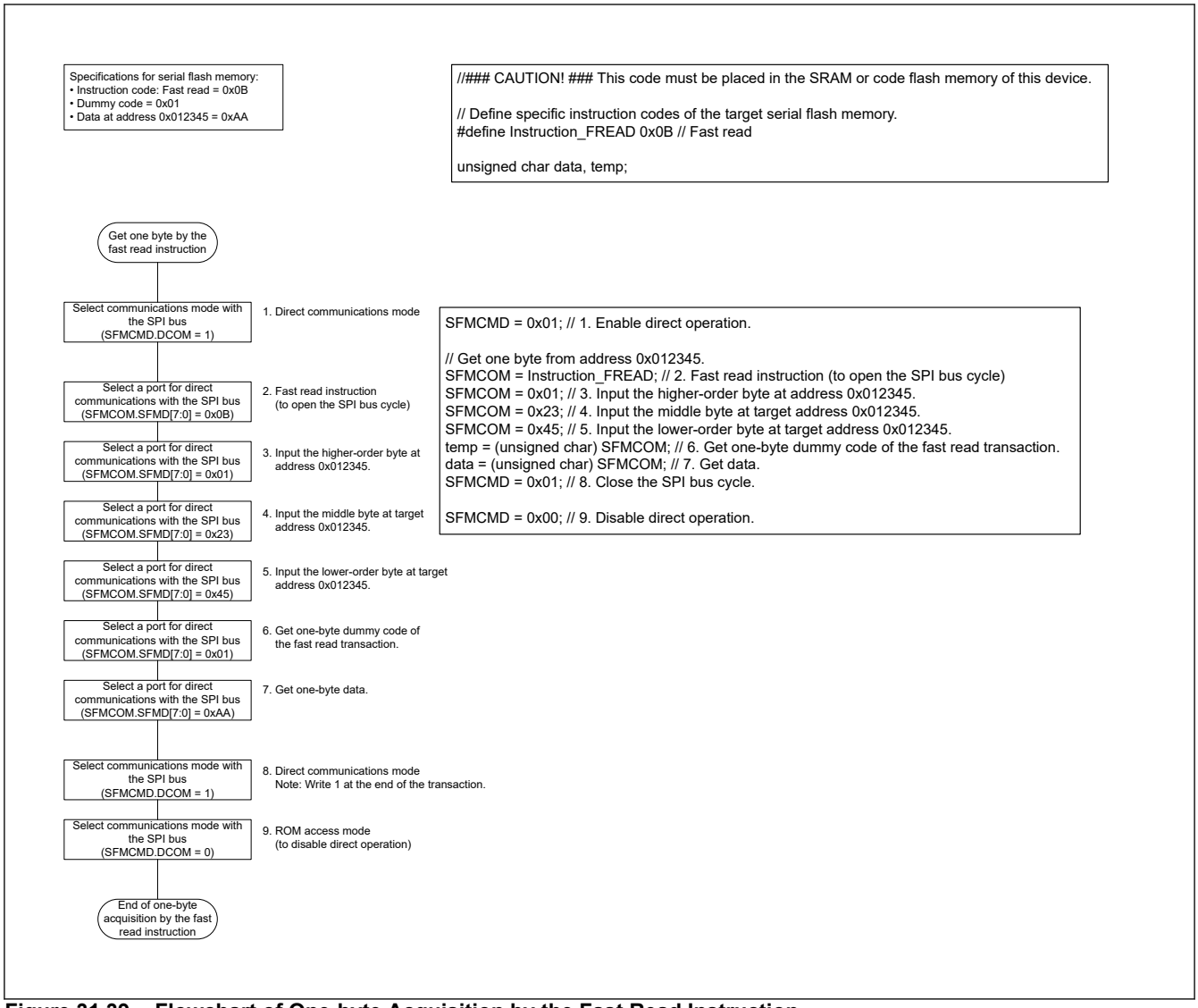



**Figure 31.39 Flowchart of One-byte Acquisition by the Fast Read Instruction**

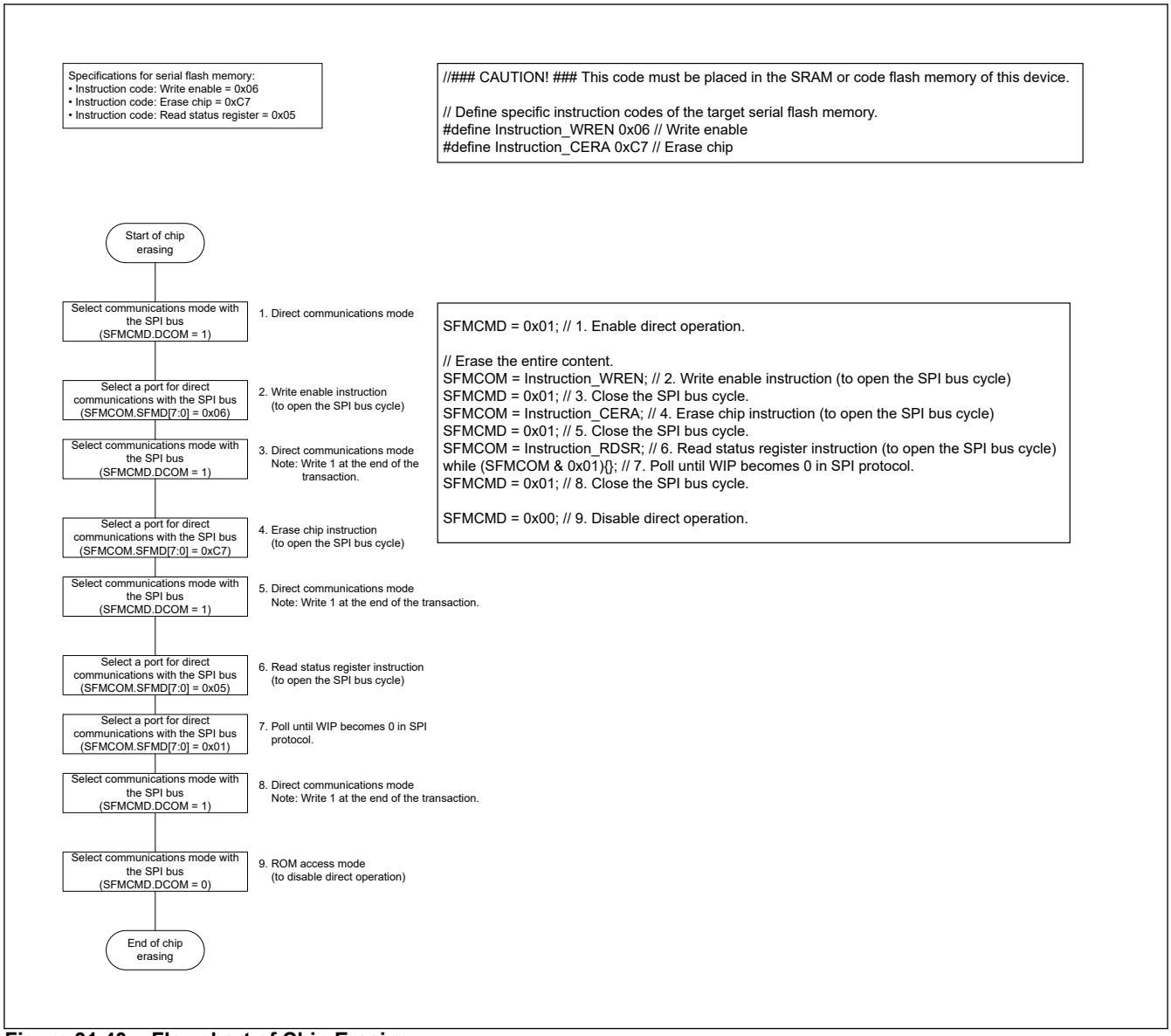
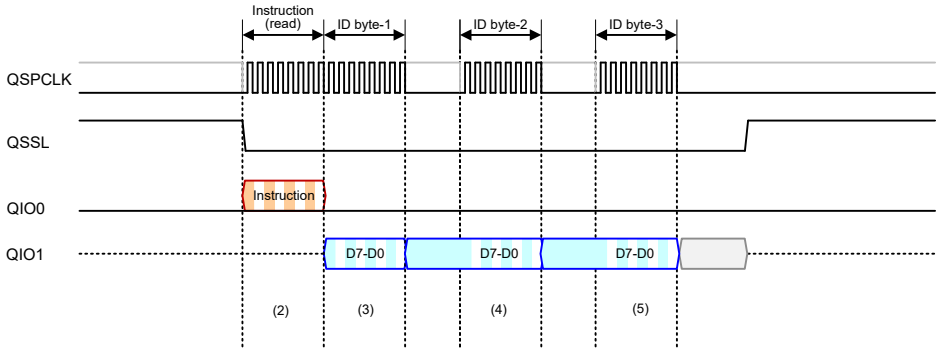



**Figure 31.40 Flowchart of Chip Erasing**




Note: (1) SFMCMD = 0x01; // Enable direct operation. Get the device ID assigned by JEDEC.
(2) SFMCOM = Instruction_RDID; // Read ID instruction (to open the transaction)
(3) mfid = (unsigned char) SFMCOM; // Get the manufacturer ID from ID byte 1.
(4) mtype = (unsigned char) SFMCOM; // Get the memory type from ID byte 2.
(5) mcap = (unsigned char) SFMCOM; // Get the memory capacity from ID byte 3.
(6) SFMCMD = 0x01; // Close the transaction.
(7) SFMCMD = 0x00; // Disable direct operation.

The timings (1), (6) and (7) for enabling or disabling direct communications mode are omitted in the figure.

**Figure 31.41 Example of direct communication timing for ID read**

Note: When the Single SPI Protocol, Extended SPI Protocol is used in direct communication mode, the standard Read or Fast Read instruction must be used to reference the contents of the serial flash memory. The QSPI does not support Fast Read Dual Output, Fast Read Dual I/O, Fast Read Quad Output, or Fast Read Quad I/O transfers in this configuration. When these Fast Read operations are required, use ROM access memory.

## 31.11 Interrupts

When ROM read access is detected in direct communication mode, the SFMCST.EROMR flag is set to 1 and QSPI generates an interrupt request. Interrupt requests are retained until the EROMR flag is cleared by a 0 write. For details, see section 13, Interrupt Controller Unit (ICU).

## 31.12 Usage Note

### 31.12.1 Settings for the Module-Stop Function

QSPI operation can be disabled or enabled using Module Stop Control Register B (MSTPCRB). The QSPI is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

### 31.12.2 Procedure for Changing Settings in Multiple Control Registers

The settings of the QSPI control registers can be modified dynamically during system operation. However, when the settings of multiple control registers are changed sequentially, an SPI bus cycle might occur before all of the registers are updated. The register setting sequence must be designed so that the SPI bus timing specifications are satisfied at all stages of register setting modification.

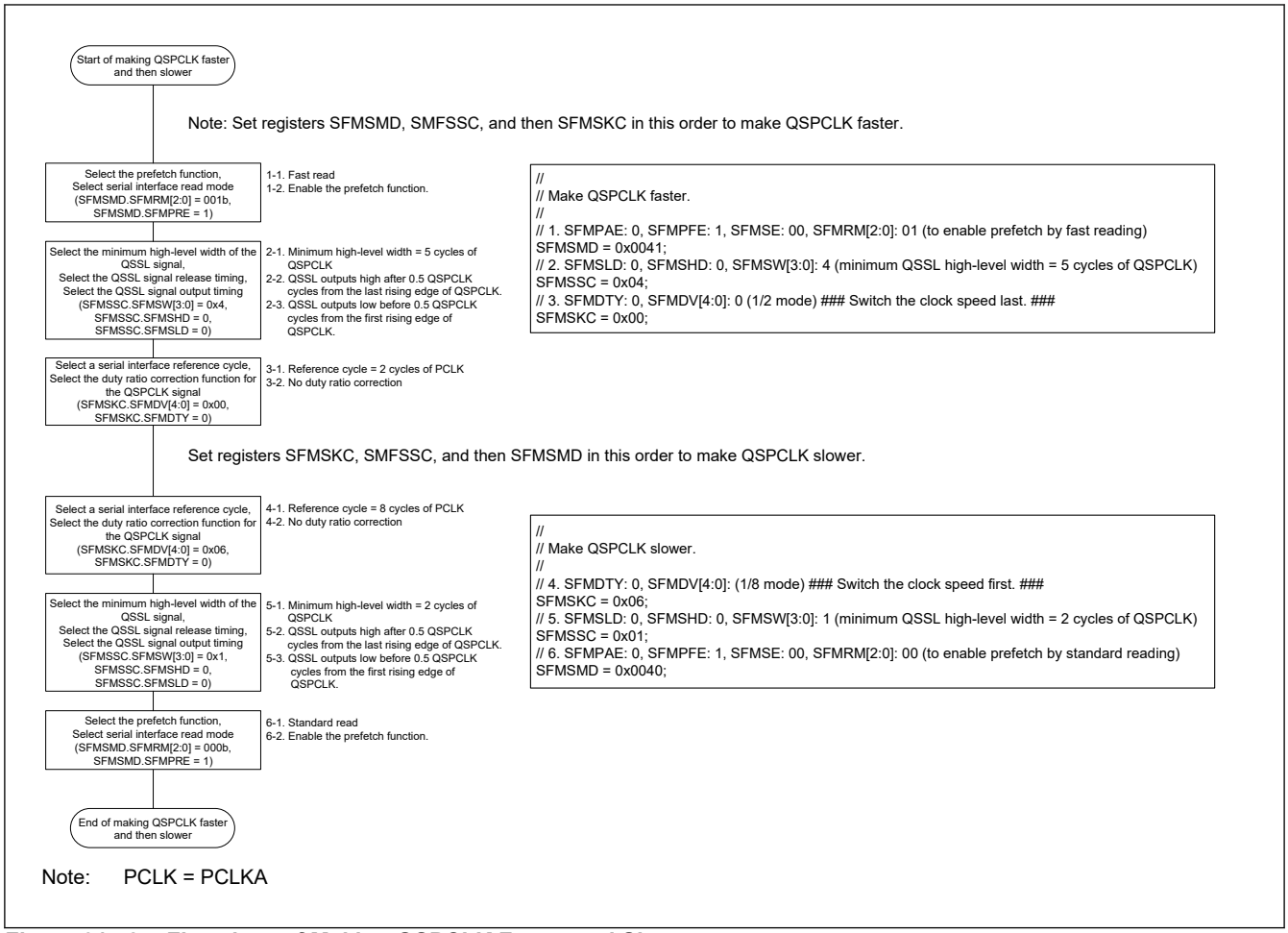



**Figure 31.42 Flowchart of Making QSPCLK Faster and Slower**

# 32. Serial Sound Interface Enhanced (SSIE)

## 32.1 Overview

The Serial Sound Interface Enhanced (SSIE) can transmit and receive audio data to and from various devices that support any of audio data formats, such as $I^2S$, monaural, and TDM.

## 32.2 Features

**Table 32.1 Features of SSIE**

| Item | | Description |
|---|---|---|
| Number of channels | | One channel, SSIE0 |
| Communication mode | | • Master/slave<br>• Transmission/reception (SSIE0 full duplex communication) |
| Communication format | | • $I^2S$ format<br>• Monaural format<br>• TDM format |
| Serial data | | • MSB first<br>• Data can be left-justified or right-justified.<br>• Data delay (1 clock cycle) or no delay selectable for the period from SSILRCK/SSIFS to SSITXD0/SSIRXD0<br>• System word length: 8, 16, 24, 32, 48, 64, 128, or 256 bits<br>• Data word length: 8, 16, 18, 20, 22, 24, or 32 bits<br>• Padding polarity: Low or high |
| Bit clock (SSIBCK) | In master mode | • Two clock sources available (AUDIO_CLK/GPT output (GTIOC2A))<br>• Clock source division ratio: 1/1, 1/2, 1/4, 1/6, 1/8, 1/12, 1/16, 1/24, 1/32, 1/48, 1/64, 1/96, and 1/128.<br>• Supply/stop is selectable while communication is halted. |
| | In master/slave mode | • Polarity (rising edge or falling edge) selectable |
| LR clock/frame synchronization (SSILRCK/ SSIFS) | In master mode | • Polarity (low level or high level) selectable<br>• Supply/stop is selectable while communication is halted. |
| Transmit data (SSITXD0) and receive data (SSIRXD0) | Transmission | • Muting method (transmission of transmit FIFO data or transmission of data fixed to 0) selectable |
| FIFO | Capacity | • Transmit FIFO/receive FIFO: 4 bytes × 32 stages |
| | Data alignment | • Data alignment method (left-justification or right-justification) selectable for the data transfer between FIFO and shift register |
| Interrupt | Interrupt output | • Communication error/idle mode<br>• Receive data full<br>• Transmit data empty |
| Low power consumption function | | • Whether to supply the audio clock selectable in master mode |
| Module stop function | • Module stop state can be set to reduce power consumption. | |
| TrustZone Filter | • Security attribution can be set. | |

The following table lists and defines the terms used for the communication formats SSIE can use:

**Table 32.2 Definition of terms (1 of 2)**

| Term | Definition |
|---|---|
| Start trigger | First edge of the signal on the SSILRCK/SSIFS pin when the signal is set to the value specified in LRCKP to enable communication |
| Frame boundary | Point where SSIE starts transferring the first data of a frame or the point where SSIE ends transferring the last data of the frame |

**Table 32.2 Definition of terms (2 of 2)**

| Term | Definition |
|---|---|
| Frame word number | Number of sound channels per frame |
| System word length | Number of bits per channel |
| Data word length | Number of significant bits per channel |
| Control bits for communication formats | • SSICR register: FRM, DWL, SWL, LRCKP, SPDP, SDTA, PDTA, and DEL bits<br>• SSIFCR register: BSW bit<br>• SSIOFR register: OMOD bit<br>• SSISCR register: TDES[4:0] and RDFS[4:0] bits |

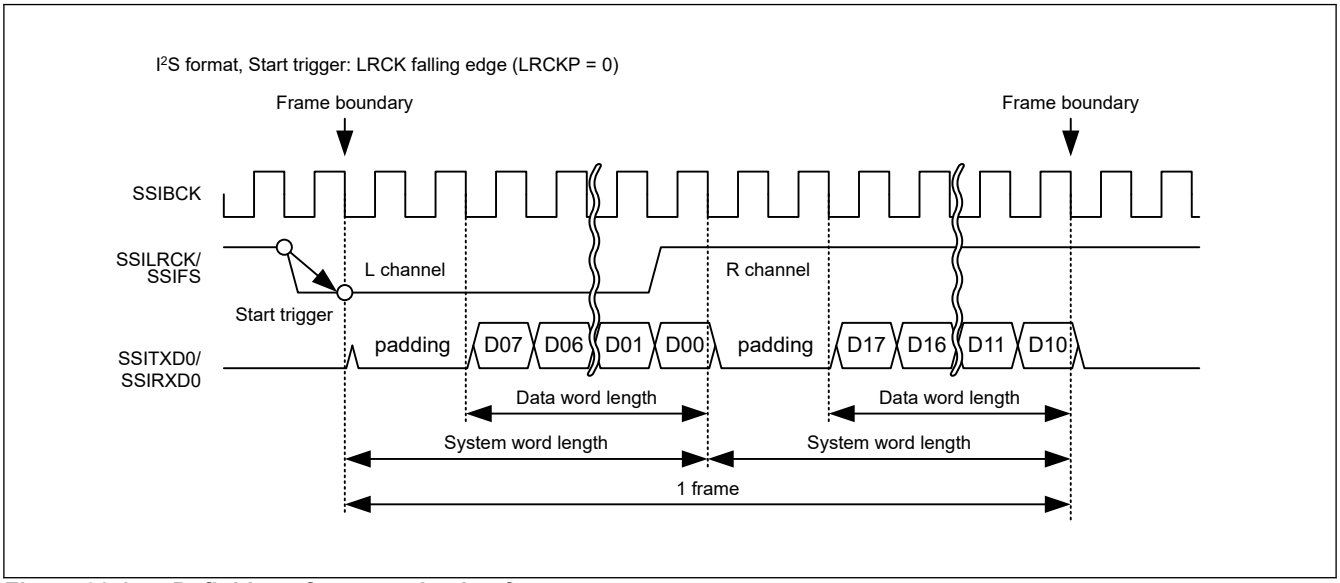



**Figure 32.1 Definition of communication format**

## 32.3 Block Diagram

Figure 32.2 shows a block diagram of SSIE.

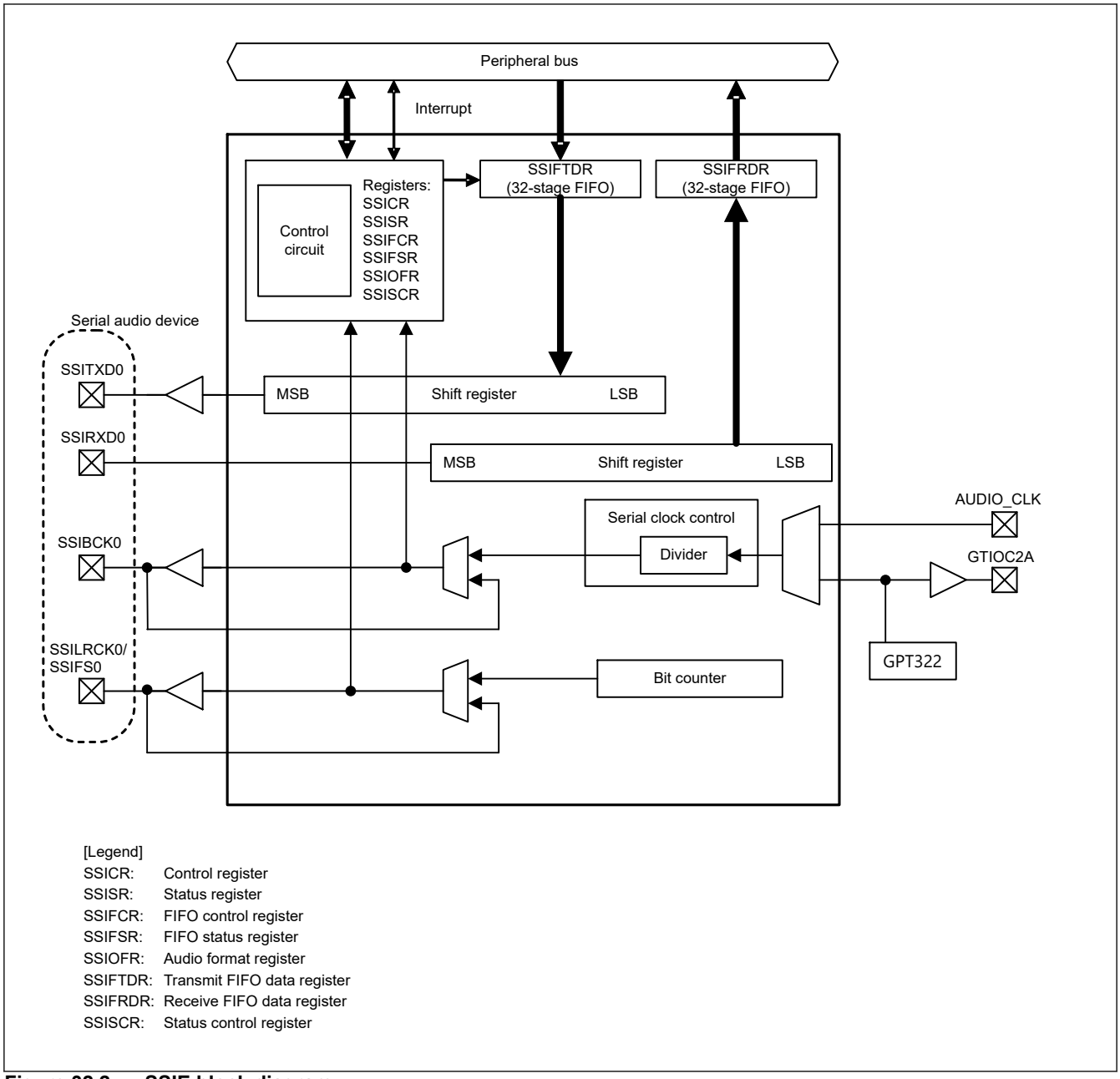



**Figure 32.2 SSIE block diagram**

Figure 32.3 shows the clock configuration of SSIE.

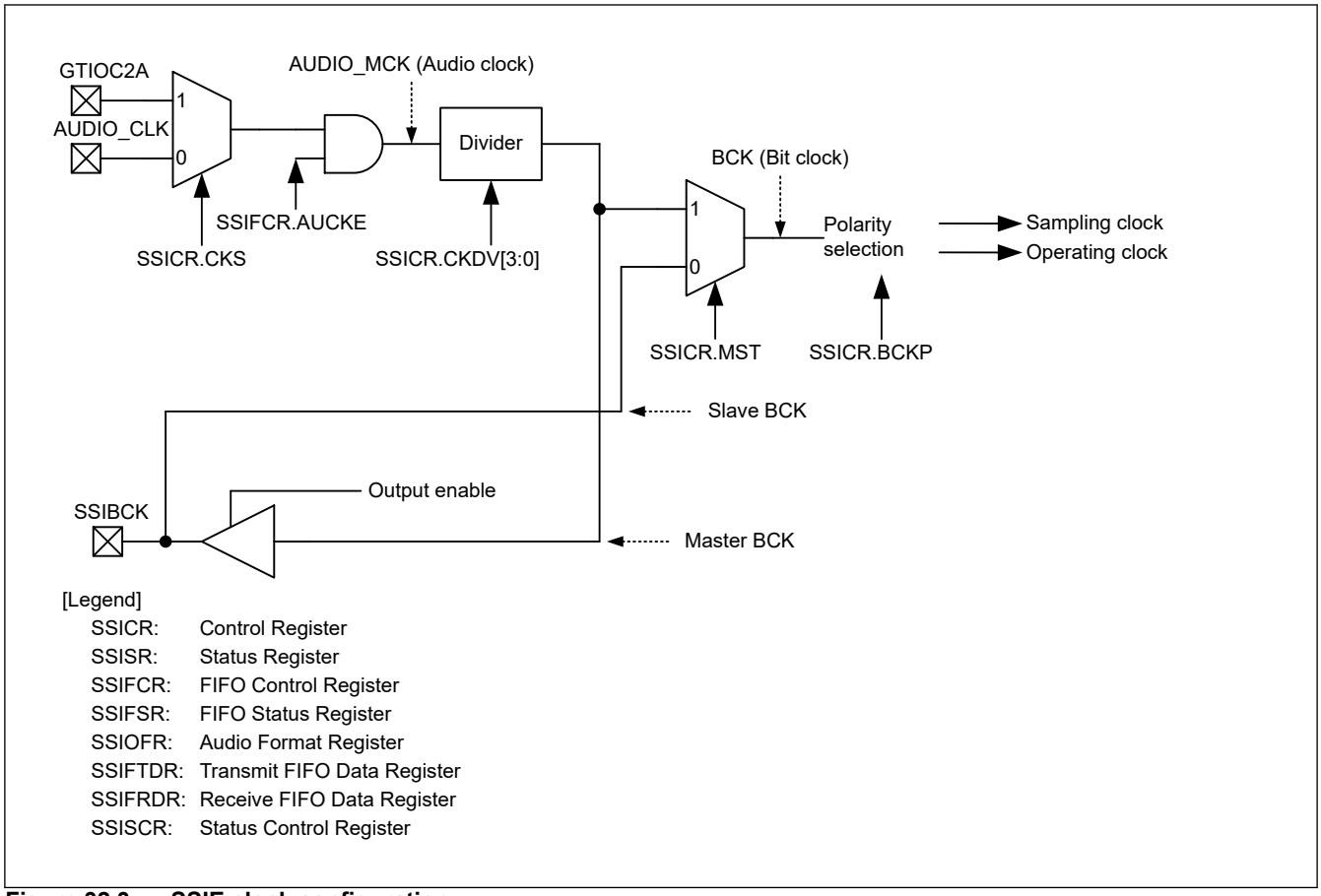



**Figure 32.3 SSIE clock configuration**

## 32.4 Register Descriptions

### 32.4.1 SSICR : Control Register

Base address: SSIE0 = 0x4009_D000

Offset address: 0x00

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | CKS | TUIEN | TOIEN | RUIEN | ROIEN | IIEN | — | FRM[1:0] | | DWL[2:0] | | | SWL[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | MST | BCKP | LRCKP | SPDP | SDTA | PDTA | DEL | CKDV[3:0] | | | | MUEN | — | TEN | REN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | REN | Reception Enable*2<br>0: Disables reception<br>1: Enables reception (starts reception) | R/W |
| 1 | TEN | Transmission Enable*2<br>0: Disables transmission<br>1: Enables transmission (starts transmission) | R/W |
| 2 | — | This bit is read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | MUEN | Mute Enable<br>0: Disables muting on the next frame boundary<br>1: Enables muting on the next frame boundary | R/W |
| 7:4 | CKDV[3:0] | Selects Bit Clock Division Ratio*1<br>0x0: AUDIO_MCK<br>0x1: AUDIO_MCK/2<br>0x2: AUDIO_MCK/4<br>0x3: AUDIO_MCK/8<br>0x4: AUDIO_MCK/16<br>0x5: AUDIO_MCK/32<br>0x6: AUDIO_MCK/64<br>0x7: AUDIO_MCK/128<br>0x8: AUDIO_MCK/6<br>0x9: AUDIO_MCK/12<br>0xA: AUDIO_MCK/24<br>0xB: AUDIO_MCK/48<br>0xC: AUDIO_MCK/96<br>Others: Setting prohibited | R/W |
| 8 | DEL | Selects Serial Data Delay*1<br>In the monaural format, this bit controls the waveform of SSILRCK/SSIFS.<br>For details, see section 32.5.2. Monaural Format.<br>0: Delay of 1 cycle of SSIBCK between SSILRCK/SSIFS and SSITXD0/SSIRXD0<br>1: No delay between SSILRCK/SSIFS and SSITXD0/SSIRXD0 | R/W |
| 9 | PDTA | Selects Placement Data Alignment*1<br>0: Left-justifies placement data (SSIFTDR, SSIFRDR)<br>1: Right-justifies placement data (SSIFTDR, SSIFRDR) | R/W |
| 10 | SDTA | Selects Serial Data Alignment*1<br>0: Transmits and receives serial data first and then padding bits<br>1: Transmit and receives padding bits first and then serial data | R/W |
| 11 | SPDP | Selects Serial Padding Polarity*1<br>0: Padding data is at a low level<br>1: Padding data is at a high level | R/W |
| 12 | LRCKP | Selects the Initial Value and Polarity of LR Clock/Frame Synchronization Signal*1<br>0: The initial value is at a high level.<br>The start trigger for a frame is synchronized with a falling edge of SSILRCK/SSIFS.<br>1: The initial value is at a low level.<br>The start trigger for a frame is synchronized with a rising edge of SSILRCK/SSIFS. | R/W |
| 13 | BCKP | Selects Bit Clock Polarity*1<br>0: SSILRCK/SSIFS and SSITXD0/SSIRXD0 change at a falling edge (SSILRCK/SSIFS and SSIRXD0 are sampled at a rising edge of SSIBCK).<br>1: SSILRCK/SSIFS and SSITXD0/SSIRXD0 change at a rising edge (SSILRCK/SSIFS and SSIRXD0 are sampled at a falling edge of SSIBCK). | R/W |
| 14 | MST | Master Enable*1<br>0: Slave-mode communication<br>1: Master-mode communication | R/W |
| 15 | — | This bit is read as 0. The write value should be 0. | R/W |
| 18:16 | SWL[2:0] | Selects System Word Length*1<br>0 0 0: 8 bits<br>0 0 1: 16 bits<br>0 1 0: 24 bits<br>0 1 1: 32 bits<br>1 0 0: 48 bits<br>1 0 1: 64 bits<br>1 1 0: 128 bits<br>1 1 1: 256 bits | R/W |

<table>
<tr><th>Bit</th><th>Symbol</th><th>Function</th><th>R/W</th></tr>
<tr><td>21:19</td><td>DWL[2:0]</td><td>Selects Data Word Length*1<br>0 0 0: 8 bits<br>0 0 1: 16 bits<br>0 1 0: 18 bits<br>0 1 1: 20 bits<br>1 0 0: 22 bits<br>1 0 1: 24 bits<br>1 1 0: 32 bits<br>1 1 1: Setting prohibited</td><td>R/W</td></tr>
<tr><td>23:22</td><td>FRM[1:0]</td><td>Selects Frame Word Number*1
<table>
<tr><th colspan="4">Communication format (SSIOFR.OMOD[1:0])</th></tr>
<tr><th>FRM[1:0]</th><th>I²S (00b)</th><th>Monaural (10b)</th><th>TDM (01b)</th></tr>
<tr><td>00b</td><td>2</td><td>1</td><td>Setting prohibited</td></tr>
<tr><td>01b</td><td rowspan="3">Setting prohibited</td><td rowspan="3">Setting prohibited</td><td>4</td></tr>
<tr><td>10b</td><td>5</td></tr>
<tr><td>11b</td><td>6</td></tr>
</table></td><td>R/W</td></tr>
<tr><td>24</td><td>—</td><td>This bit is read as 0. The write value should be 0.</td><td>R/W</td></tr>
<tr><td>25</td><td>IIEN</td><td>Idle Mode Interrupt Output Enable<br>0: Disables idle mode interrupt output<br>1: Enables idle mode interrupt output</td><td>R/W</td></tr>
<tr><td>26</td><td>ROIEN</td><td>Receive Overflow Interrupt Output Enable<br>0: Disables receive overflow interrupt output<br>1: Enables receive overflow interrupt output</td><td>R/W</td></tr>
<tr><td>27</td><td>RUIEN</td><td>Receive Underflow Interrupt Output Enable<br>0: Disables receive underflow interrupt output<br>1: Enables receive underflow interrupt output</td><td>R/W</td></tr>
<tr><td>28</td><td>TOIEN</td><td>Transmit Overflow Interrupt Output Enable<br>0: Disables transmit overflow interrupt output<br>1: Enables transmit overflow interrupt output</td><td>R/W</td></tr>
<tr><td>29</td><td>TUIEN</td><td>Transmit Underflow Interrupt Output Enable<br>0: Disables transmit underflow interrupt output<br>1: Enables transmit underflow interrupt output</td><td>R/W</td></tr>
<tr><td>30</td><td>CKS</td><td>Selects an Audio Clock for Master-mode Communication*1<br>0: Selects the AUDIO_CLK input<br>1: Selects the GTIOC2A (GPT output)</td><td>R/W</td></tr>
<tr><td>31</td><td>—</td><td>This bit is read as 0. The write value should be 0.</td><td>R/W</td></tr>
</table>

Note 1. Writing to these bits while SSIE is in a communication state (SSISR.IIRQ = 0) is prohibited. If the value of these bits is changed by rewriting, subsequent operation is unpredictable.

Note 2. If the TEN bit or REN bit is rewritten, make sure that the SSISR.IIRQ bit is in the desired status. If the value of the TEN or REN bit is changed by rewriting, subsequent operation is unpredictable. For example, when transmission or reception is enabled, check that SSISR.IIRQ is 0; when transmission or reception is disabled, check that SSISR.IIRQ is 1.

With this register, select an audio clock, control interrupt requests, select data formats, and set an operation mode.

### TEN and REN bits (Transmission and Reception Enable)

The TEN and REN bits enable/disable transmission and reception. When 1 is written to one of these bits, the corresponding communication operation starts in synchronization with a start trigger by the SSILRCK/SSIFS signal. For details, see section 32.8.2. Transmission to section 32.8.4. Transmission and Reception. When 0 is written to this bit, the current communication operation stops at the next frame boundary. To use SSIE for both transmission and reception, always write 1 to these bits together. When stopping the communication using SSIE, always disable both transmission and reception (write 0 to the TEN and REN bits).

If you want to stop SSIE before a frame boundary is reached, perform a software reset procedure.

### MUEN bit (Mute Enable)

The MUEN bit sets/clears the mute function for the data output from the SSITXD0 pin. When this bit is set to 1 in the middle of a frame, the SSITXD0 output changes to 0 at the next frame boundary. When this bit is set to 0 in the middle of a frame, the SSITXD0 output changes to the data of transmit FIFO data register at the next frame boundary. Note that this bit controls data only. Status flags and interrupt signals are normally generated.

Changing the value of this bit must be performed only after setting the communication format to be used.

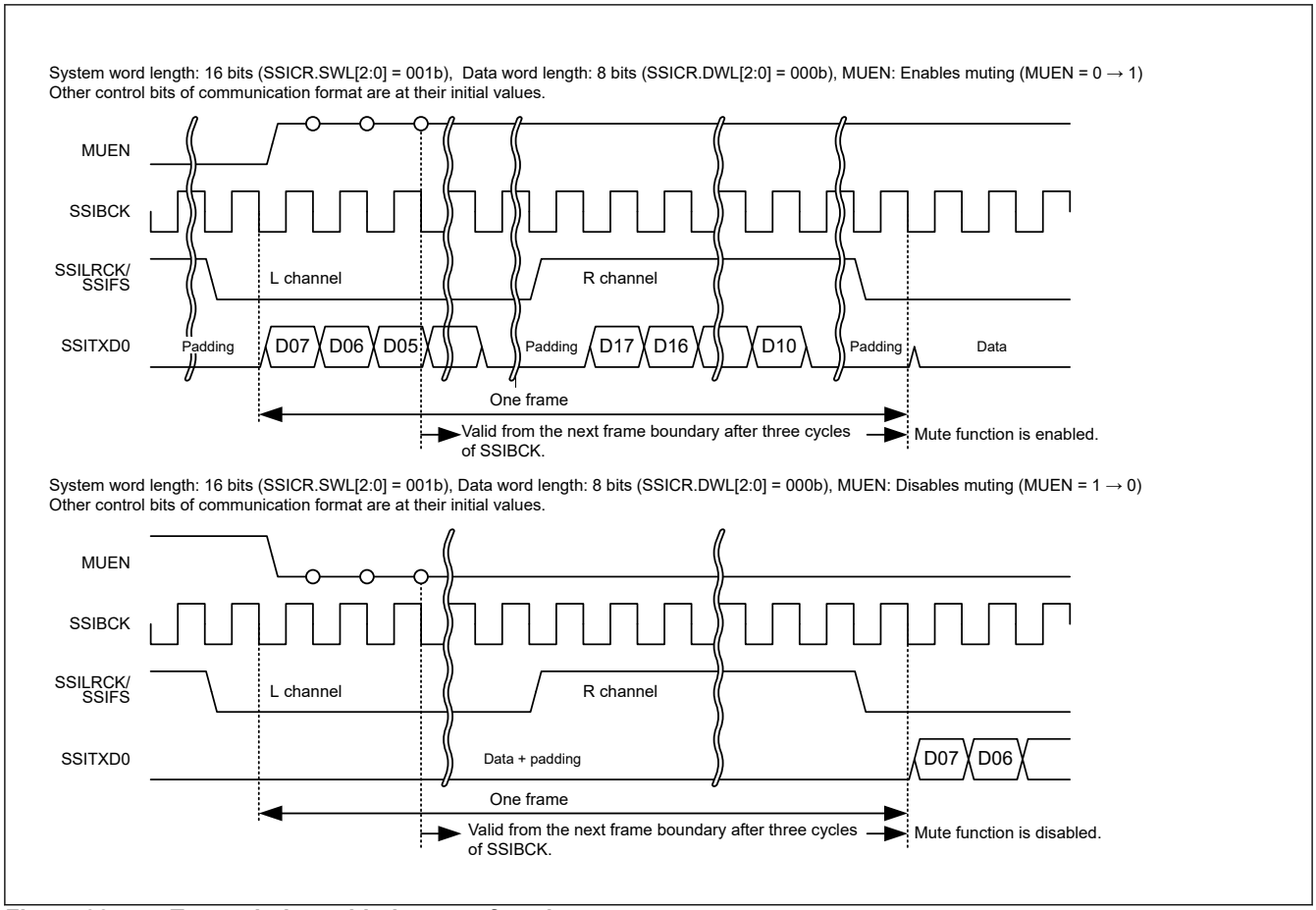



**Figure 32.4 Transmit data with the mute function set**

### CKDV[3:0] bits (Selects Bit Clock Division Ratio)

The CKDV[3:0] bits set the division ratio of the bit clock based on AUDIO_MCK in master-mode communication (MST= 1). In slave-mode communication (MST = 0), setting of these bits are invalid.

Writing to this bit must be performed when the supply of AUDIO_MCK is stopped. For details about the timing, see the detailed description of the AUCKE bit in section 32.4.3. SSIFCR : FIFO Control Register.

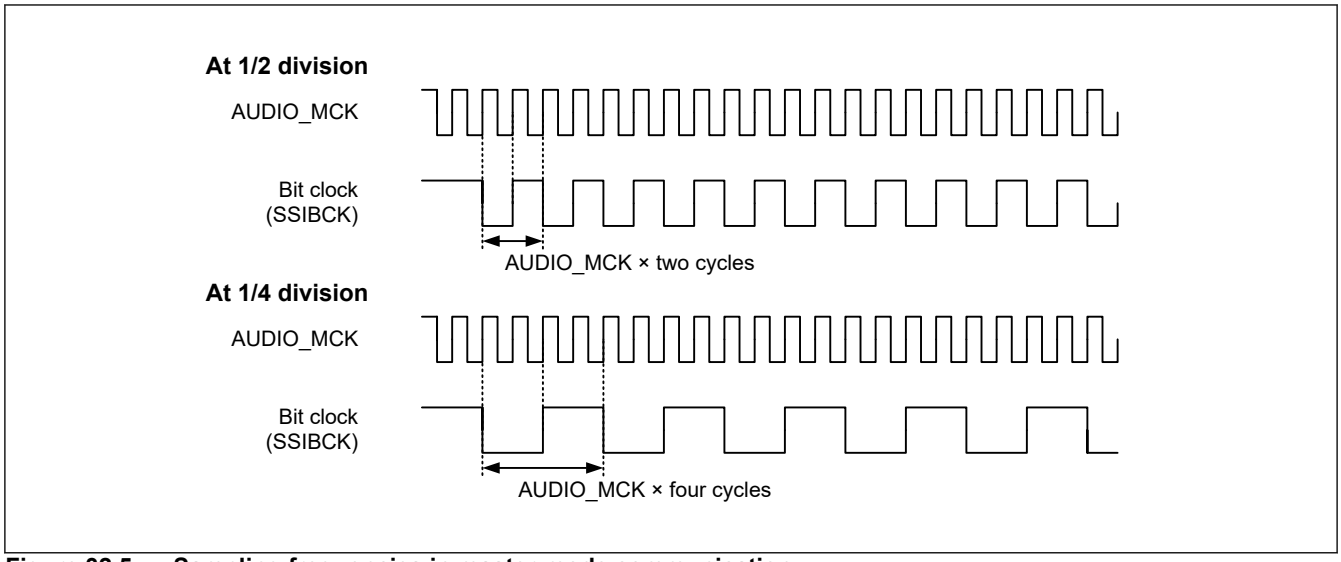



**Figure 32.5 Sampling frequencies in master-mode communication**

### DEL bit (Selects Serial Data Delay)

The DEL bit sets whether or not there will be a delay between SSILRCK/SSIFS and SSITXD0/SSIRXD0.

For the $I^2S$ or TDM format, set the DEL bit to 0. When the monaural format is used, setting of this bit changes the high period width of SSILRCK/SSIFS. For details, see section 32.5.2. Monaural Format. When using a compatible communication format, specify a setting of this bit that enables communication.

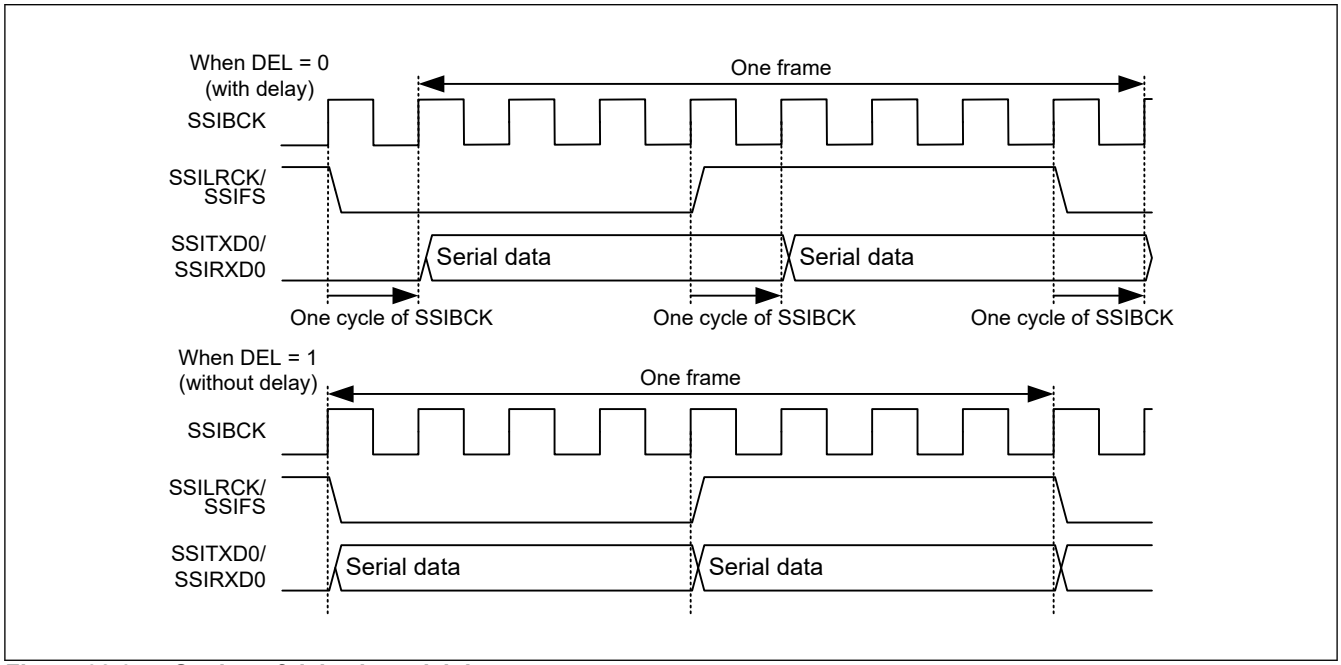



**Figure 32.6 Setting of delay in serial data**

### PDTA bit (Selects Placement Data Alignment)

The PDTA bit sets how to align placement data. With the setting of data word length as 32 bits (SSICR.DWL[2:0] = 110b), this bit is invalid.

At transmission, see Figure 32.7.

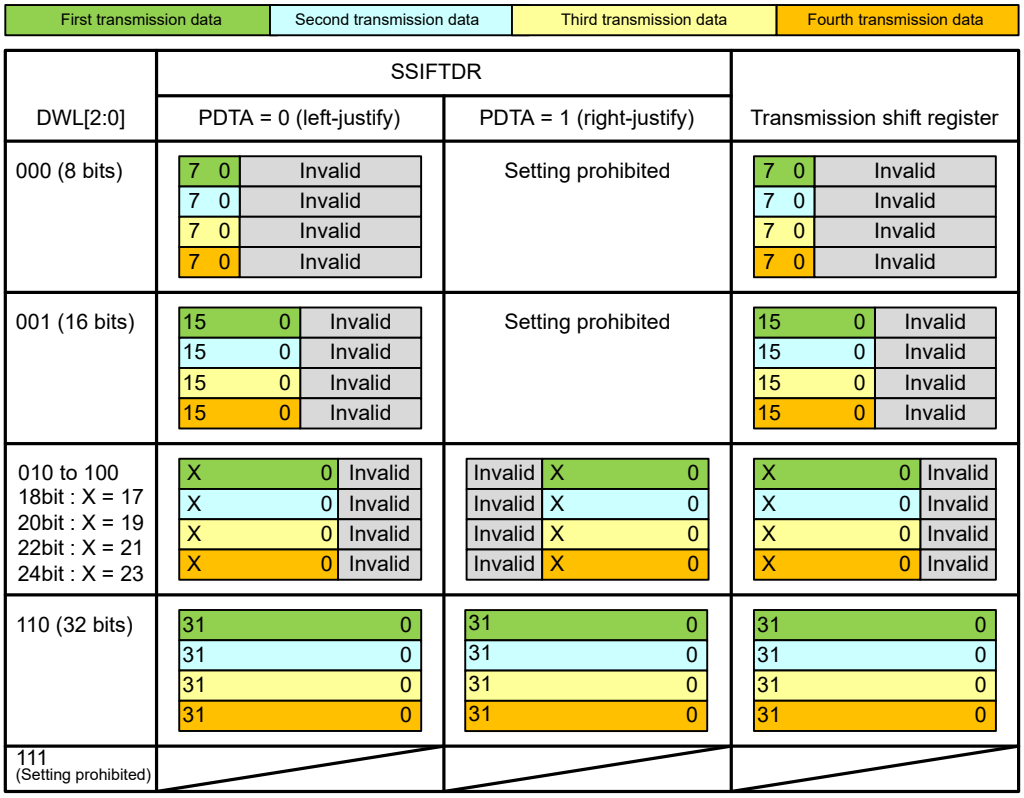



**Figure 32.7 Alignment of placement data at transmission**

At reception, see Figure 32.8.

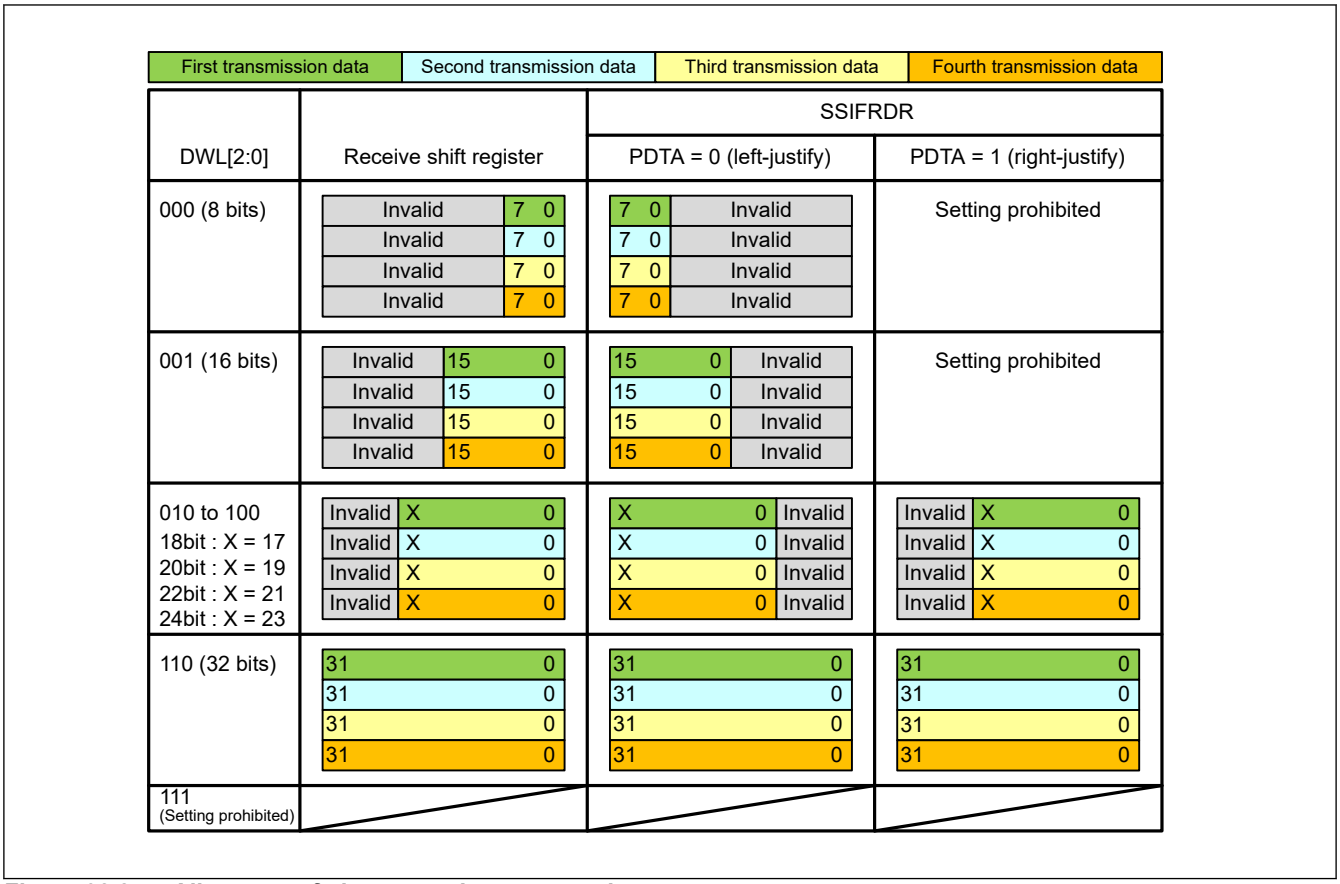



**Figure 32.8 Alignment of placement data at reception**

### SDTA bit (Selects Serial Data Delay)

The SDTA bit sets how to align serial data and padding bits. For communication without padding bits, this bit is invalid.

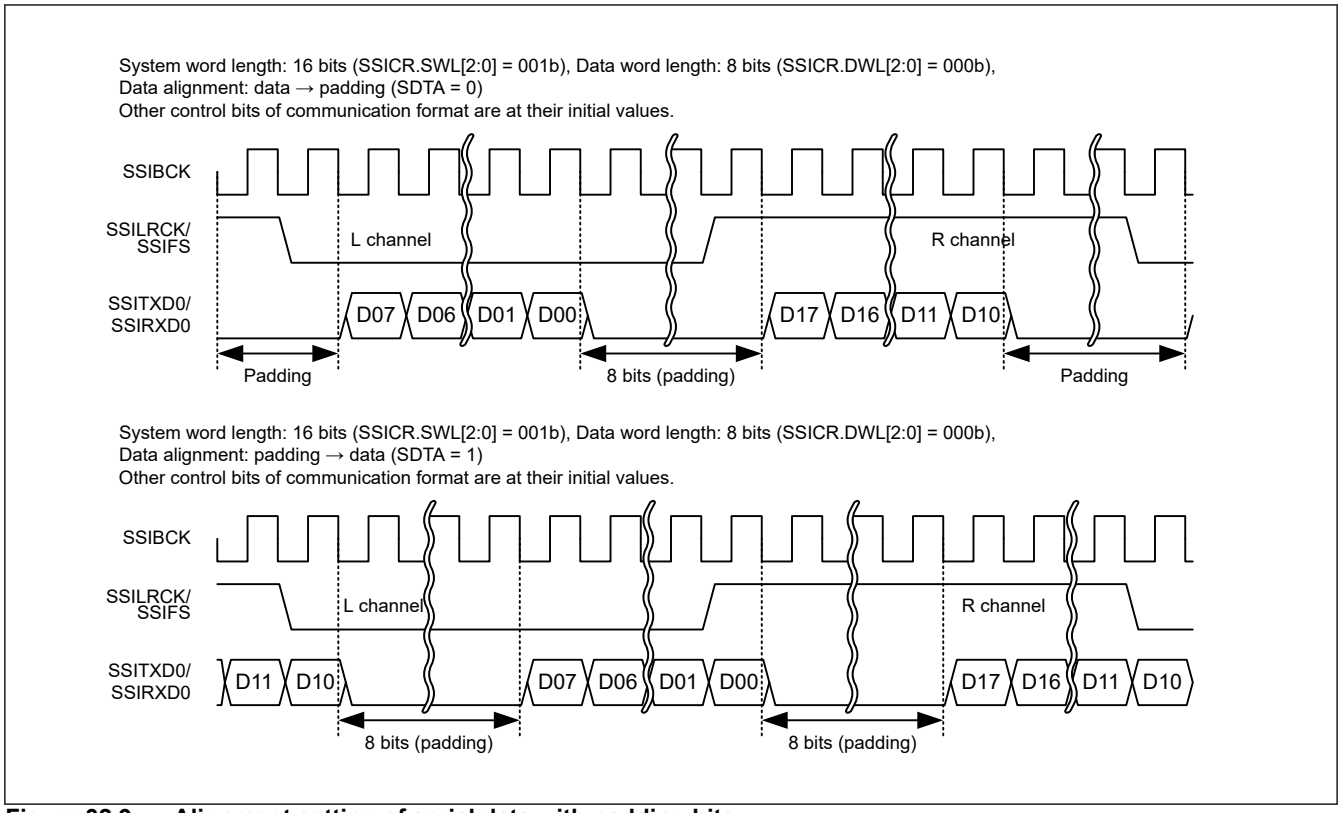



**Figure 32.9 Alignment setting of serial data with padding bits**

### SPDP bit (Selects Serial Padding Polarity)

The SPDP bit sets polarity of padding bits.

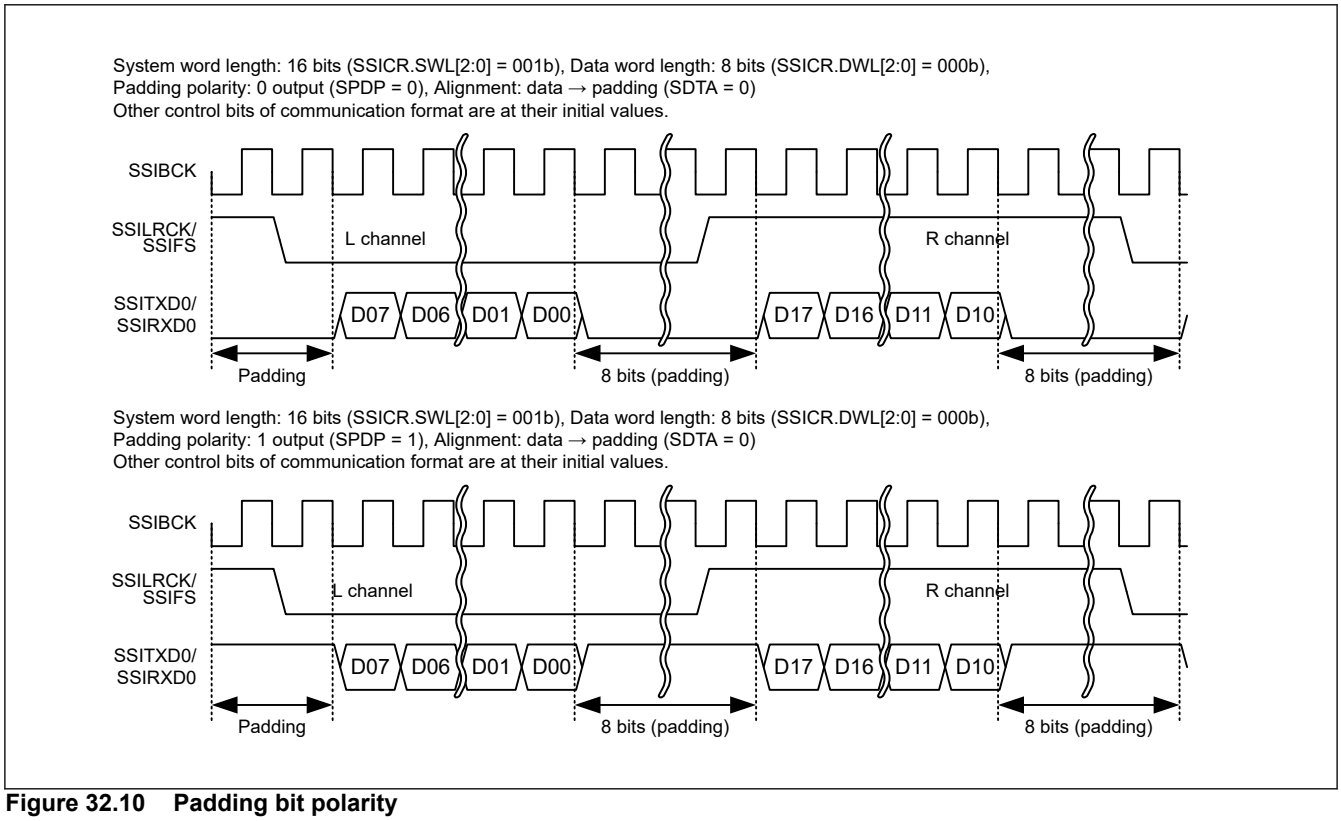



**Figure 32.10 Padding bit polarity**

### LRCKP bit (Selects the Initial Value and Polarity of LR Clock/Frame Synchronization Signal)

The LRCKP bit sets the initial value and polarity of SSILRCK/SSIFS. Set this bit according to the communication format to be used in SSIE. See Table 32.3 Initial output value and polarity of SSILRCK/SSIFS pin. For the slave-mode communication (MST = 0), only the start trigger is used.

Writing to these bits must be performed when the LR clock supply to the SSILRCK/SSIFS pin is stopped. For details about the output of LR clock, see the detailed description of the LRCONT bit in section 32.4.7. SSIOFR : Audio Format Register.

**Table 32.3 Initial output value and polarity of SSILRCK/SSIFS pin**

| Communication Format | Expected Initial State | Setting Value of LRCKP |
|---|---|---|
| $I^2S$ | High | 0 |
| Monaural | Low | 1 |
| TDM | Low | 1 |

Note: When the format to be used is compatible with the $I^2S$, monaural, or TDM format, specify settings to enable communication with the respective formats.

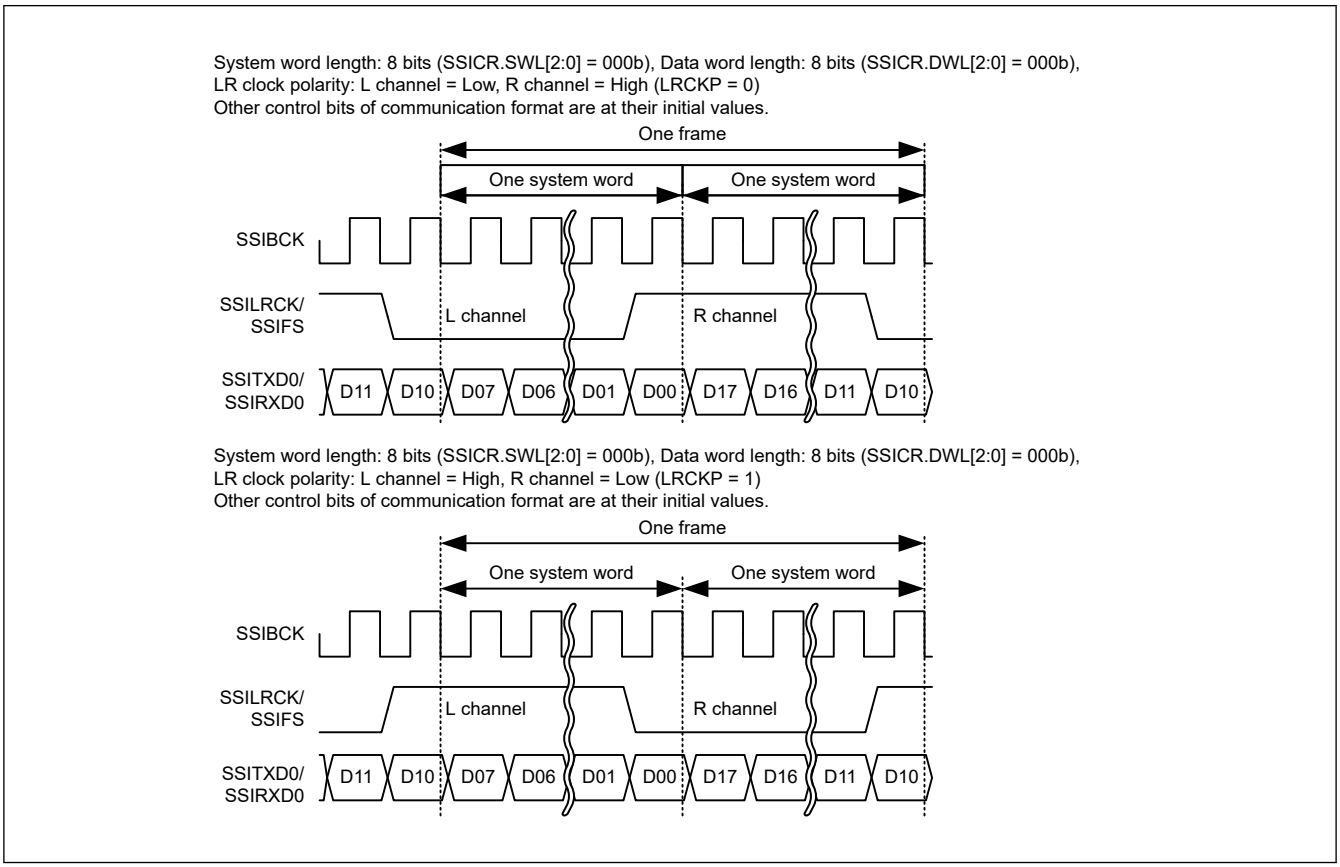



**Figure 32.11 LR clock/frame synchronization polarity setting**

### BCKP bit (Selects Bit Clock Polarity)

The BCKP bit sets the bit clock polarity.

Writing to this bit must be performed when the supply of AUDIO_MCK is stopped. For details about the timing, see the detailed description of the AUCKE bit in section 32.4.3. SSIFCR : FIFO Control Register.

**Table 32.4 Bit clock polarity (1 of 2)**

| Communication | Master/Slave | Timing | BCKP = 0 | BCKP = 1 |
|---|---|---|---|---|
| Reception | Slave | At SSILRCK/SSIFS sampling | SSIBCK rising edge | SSIBCK falling edge |
| | Master/slave | At SSIRXD0 sampling | SSIBCK rising edge | SSIBCK falling edge |

**Table 32.4 Bit clock polarity (2 of 2)**

| Communication | Master/Slave | Timing | BCKP = 0 | BCKP = 1 |
|---|---|---|---|---|
| Transmission | Master | At change of SSILRCK/SSIFS output | SSIBCK falling edge | SSIBCK rising edge |
| | Master/slave | At change of SSITXD0 output | SSIBCK falling edge | SSIBCK rising edge |

### MST bit (Master Enable)

The MST bit sets master-/slave-mode communication.

Writing to this bit must be performed when the supply of AUDIO_MCK is stopped. For details about the timing, see the detailed description of the AUCKE bit in section 32.4.3. SSIFCR : FIFO Control Register.

### SWL[2:0] bits (Selects System Word Length)

The SWL[2:0] bits set the number of bits in one system word. Padding bits are sent and received in relation with one data word set with DWL[2:0]. See Table 32.11 for details.

Writing to these bits must be performed when the LR clock supply to the SSILRCK/SSIFS pin is stopped. For details about the output of LR clock, see the detailed description of the LRCONT bit in section 32.4.7. SSIOFR : Audio Format Register.

### DWL[2:0] bits (Selects Data Word Length)

The DWL[2:0] bits set the number of bits in one data word. The data word length (number of bits per data word) must not exceed the system word length (number of bits per system word). For details, see Table 32.11.

### FRM[1:0] bits (Selects Frame Word Number)

The FRM[1:0] bits set the frame word number in individual communication formats.

Writing to these bits must be performed when the LR clock supply to the SSILRCK/SSIFS pin is stopped. For details about the output of LR clock, see the detailed description of the LRCONT bit in section 32.4.7. SSIOFR : Audio Format Register.

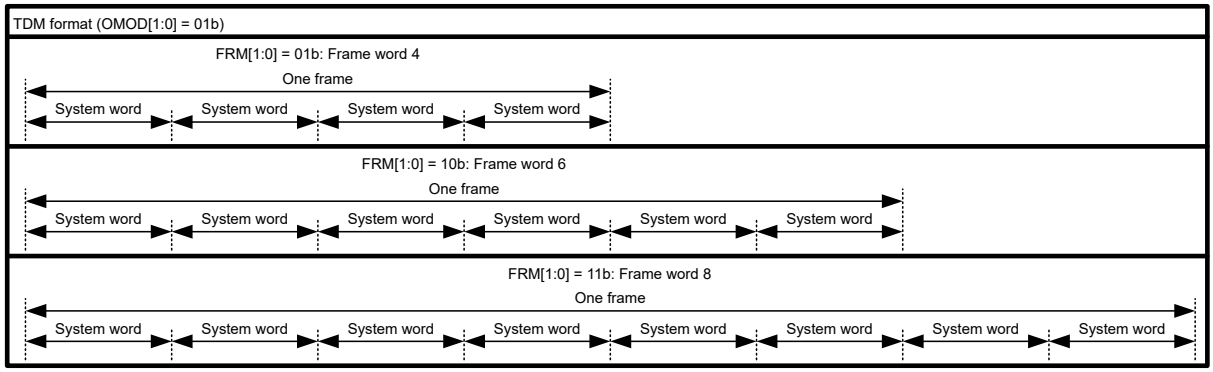



**Figure 32.12 Frame word number**

### IIEN bit (Idle Mode Interrupt Output Enable)

The IIEN bit enables/disables output of idle mode interrupts. By enabling this bit (set it to 1), an interrupt is output at a rising edge of SSISR.IIRQ = 1. An interrupt is also output when this bit is changed from 0 to 1 while SSISR.IIRQ = 1.

### ROIEN bit (Receive Overflow Interrupt Output Enable)

The ROIEN bit enables/disables output of receive overflow interrupts. By enabling this bit (set it to 1), an interrupt is output at a rising edge of SSISR.ROIRQ = 1. An interrupt is also output when this bit is changed from 0 to 1 while SSISR.ROIRQ = 1.

### RUIEN bit (Receive Underflow Interrupt Output Enable)

The RUIEN bit enables/disables output of receive underflow interrupts. By enabling this bit (set it to 1), an interrupt is output at a rising edge of SSISR.RUIRQ = 1. An interrupt is also output when this bit is changed from 0 to 1 while SSISR.RUIRQ = 1.

### TOIEN bit (Transmit Overflow Interrupt Output Enable)

The TOIEN bit enables/disables output of transmit overflow interrupts. By enabling this bit (set it to 1), an interrupt is output at a rising edge of SSISR.TOIRQ = 1. An interrupt is also output when this bit is changed from 0 to 1 while SSISR.TOIRQ = 1.

### TUIEN bit (Transmit Underflow Interrupt Output Enable)

The TUIEN bit enables/disables output of transmit underflow interrupts. By enabling this bit (set it to 1), an interrupt is output at a rising edge of SSISR.TUIRQ = 1. An interrupt is also output when this bit is changed from 0 to 1 while SSISR.TUIRQ = 1.

### CKS bit (Selects an Audio Clock for Master-mode Communication)

The CKS bit sets the audio clock in master-mode communication (MST = 1). In slave-mode communication (MST = 0), setting of this bit is invalid.

Writing to this bit must be performed when the supply of AUDIO_MCK is stopped. For details about the timing, see the detailed description of the AUCKE bit in section 32.4.3. SSIFCR : FIFO Control Register.

## 32.4.2 SSISR : Status Register

Base address: SSIE0 = 0x4009_D000

Offset address: 0x04

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | TUIRQ | TOIRQ | RUIRQ | ROIRQ | IIRQ | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 24:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 25 | IIRQ | Idle Mode Status Flag<br>0: In the communication state<br>1: In the idle state | R |
| 26 | ROIRQ | Receive Overflow Error Status Flag<br>0: No receive overflow error is generated.<br>1: A receive overflow error is generated. | R/W |
| 27 | RUIRQ | Receive Underflow Error Status Flag<br>0: No receive underflow error is generated.<br>1: A receive underflow error is generated. | R/W |
| 28 | TOIRQ | Transmit Overflow Error Status Flag<br>0: No transmit overflow error is generated.<br>1: A transmit overflow error is generated. | R/W |
| 29 | TUIRQ | Transmit Underflow Error Status flag<br>0: No transmit underflow error is generated.<br>1: A transmit underflow error is generated. | R/W |
| 31:30 | — | These bits are read as 0. The write value should be 0. | R/W |

This register is configured with status flags that indicate SSIE operational state.

### IIRQ flag (Idle Mode Status Flag)

The IIRQ flag is a status flag that indicates the idle state. It indicates whether SSIE is in the idle state or communication state.

For details, see Figure 32.13 and Figure 32.14.

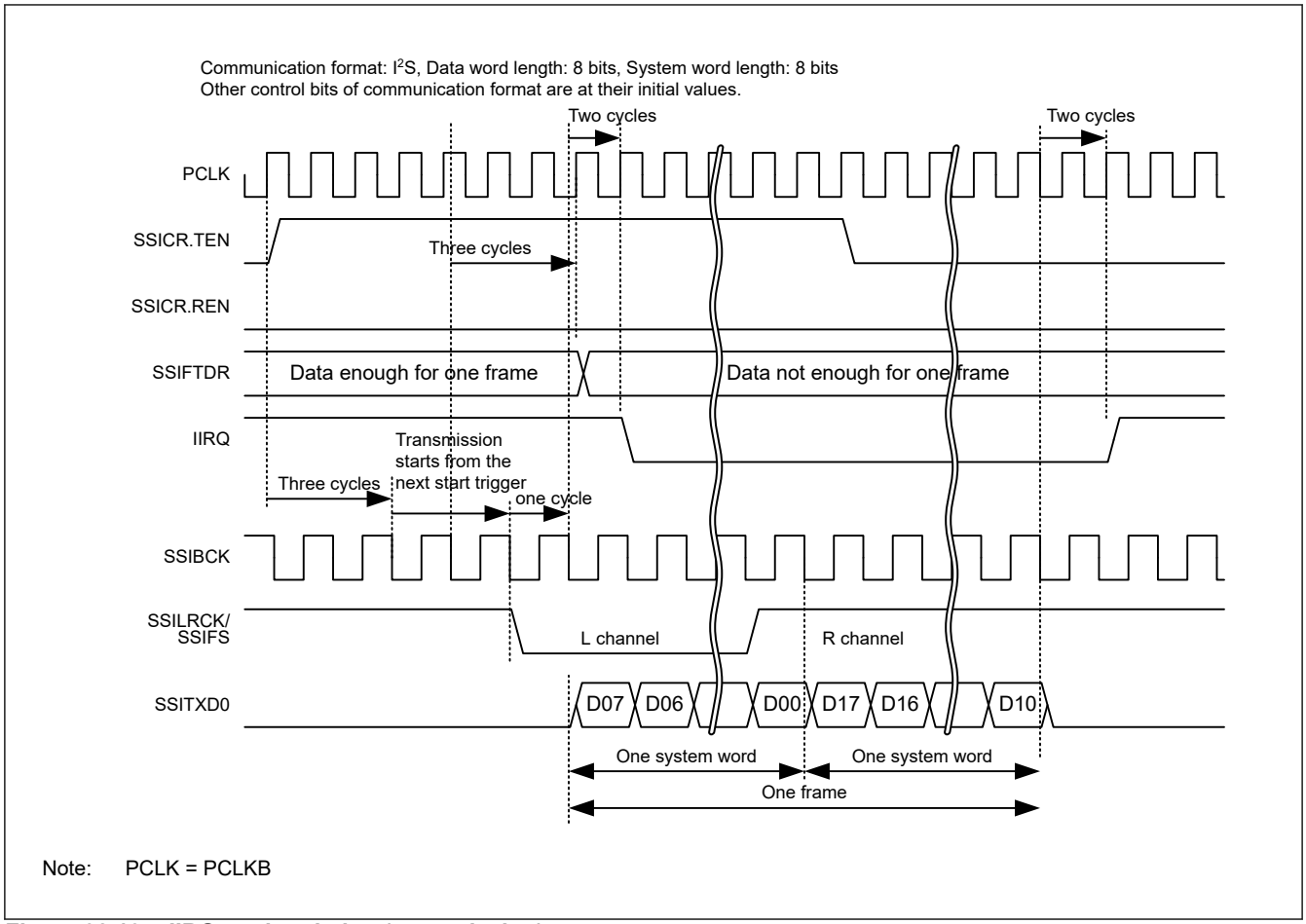



**Figure 32.13 IIRQ setting timing (transmission)**

- Transmitter (dedicated to transmission)

[Clearing condition]

While transmission was enabled (SSICR.TEN = 1 and SSICR.REN = 0), the transmit data for a transmission frame was written to the SSIFTDR register, and a start trigger was generated by the SSILRCK/SSIFS signal.

[Clearing timing]

1 SSIBCK cycle + 2 PCLKB cycles after generation of the start trigger that is the clearing condition.

[Setting condition]

While transmission and reception were disabled (SSICR.TEN = 0 and SSICR.REN = 0), transmission of one frame was complete.

[Setting timing]

2 PCLKB cycles after the end of transmission (at a frame boundary) that is the setting condition.

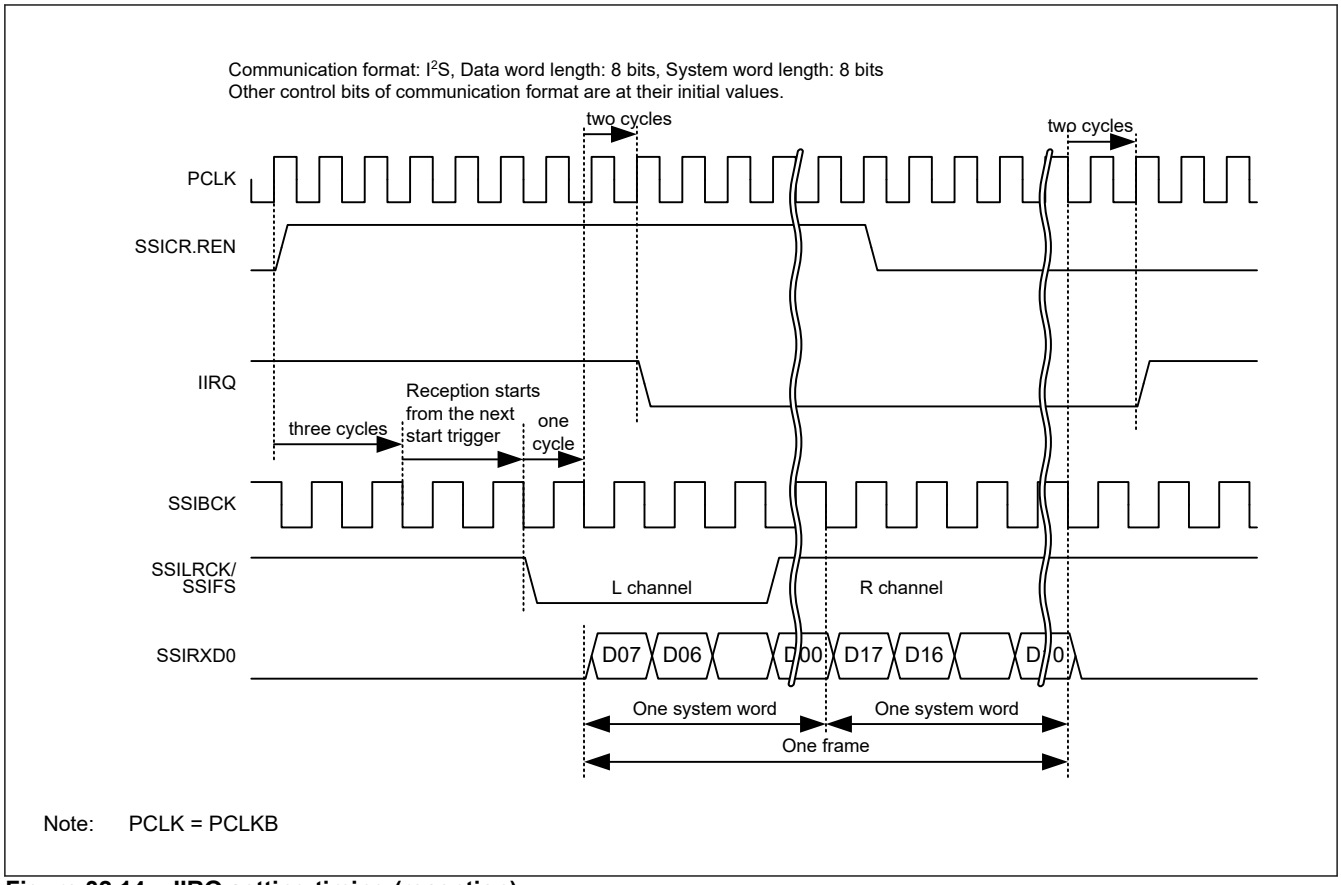



**Figure 32.14 IIRQ setting timing (reception)**

- Receiver (dedicated to reception)

[Clearing condition]

While reception was enabled (SSICR.TEN = 0 and SSICR.REN = 01, a start trigger was generated by the SSILRCK/SSIFS signal.

[Clearing timing]

1 SSIBCK cycle + 2 PCLKB cycles after generation of the start trigger that is the clearing condition.

[Setting condition]

While transmission and reception were disabled (SSICR.TEN = 0 and SSICR.REN = 0), reception of one frame was complete.

[Setting timing]

2 PCLKB cycles after the end of reception (at a frame boundary) that is the setting condition.

- Transceiver (transmission and reception)

[Clearing condition]

While transmission and reception were enabled (SSICR.TEN = 1 and SSICR.REN = 1), the transmit data for a transmission frame was written to the SSIFTDR register, and a start trigger is generated by the SSILRCK/SSIFS signal.

[Clearing timing]

1 SSIBCK cycle + 2 PCLKB cycles after generation of the start trigger that is the clearing condition.

[Setting condition]

While transmission and reception were disabled (SSICR.TEN = 0 and SSICR.REN = 0), transmission of one frame was complete.

[Setting timing]

2 PCLKB cycles after the end of transmission (at a frame boundary) that is the setting condition.

### ROIRQ flag (Receive Overflow Error Status Flag)

The ROIRQ flag is a status flag that indicates a receive overflow error. This flags is set by automatic determination but it must be cleared by register access. This flag indicates that received data is supplied at a higher rate than requested. Data is not transferred from the receive shift register to SSIFRDR where a receive overflow error is generated. For the procedure to recover from the overflow error, see section 32.8.6. Error Handling. This flag is not cleared by a receive FIFO data register reset (SSIFCR.RFRST).

[Priority order for setting and clearing]

Setting is prioritized.*1

[Clearing condition]

When either of the following operations is done:

1. Writing 0 to this bit after reading 1 from this bit*2
2. Enabling communication (changing SSICR.REN from 0 to 1).

[Clearing timing]

Clearing timing corresponding to the above clearing condition:

1. When 0 is written to this bit after reading 1 from this bit (same as the timing in Figure 32.18)
2. 1 PCLKB cycle after writing 1 to SSICR.REN.*3

[Setting condition]
At completion of receiving new data while SSIFRDR is full.

[Setting timing]
3 cycles of PCLKB after reception is completed.

Note 1. This bit is cleared by a software reset (SSIFCR.SSIRST = 1). The software reset has priority over all the clearing conditions described above.
Note 2. After reading 1 from this bit, this bit is cleared when one of the following three conditions is met:
- A software reset (SSIFCR.SSIRST = 1) is done.
- After 1 has been read, writing of 0 is complete.
- 1 PCLKB cycle passes after 1 has been written to SSICR.REN.

Note 3. After communication is enabled (by changing the value of SSICR.REN bit from 0 to 1), the reception error flags (RUIRQ and ROIRQ in the SSISR register) are cleared. If, however, the SSISR register is read continuously, the cleared status of the reception error flags might be unable to be read.

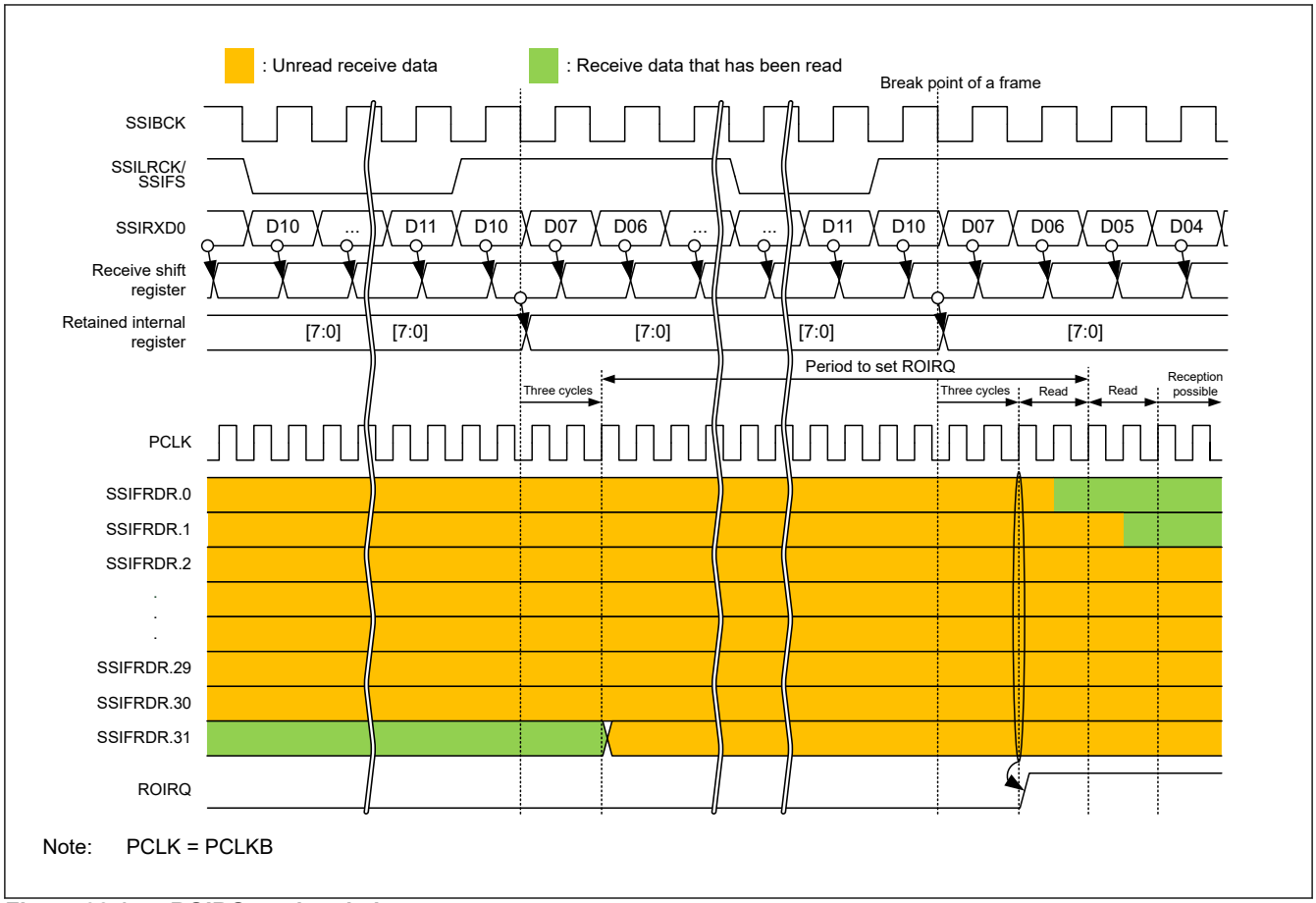



**Figure 32.15 ROIRQ setting timing**

### RUIRQ flag (Receive Underflow Error Status Flag)

The RUIRQ flag is a status flag that indicates a receive underflow error. This flags is set by automatic determination but it must be cleared by register access. This flag indicates that SSIFRDR is read while it is empty. Data read from SSIFRDR where a receive underflow error is generated is invalid. See section 32.8.6. Error Handling for the error recovery procedure. This flag is not cleared by a receive FIFO data register reset (SSIFCR.RFRST). Note, however, that this flag is not set even if the SSIFRDR register is read while the receive FIFO data register is reset (by setting SSIFCR.RFRST to 1).

[Priority order for setting and clearing]

Setting is prioritized.*1

[Clearing condition]

When either of the following operations is done:

1. Writing 0 to this bit after reading 1 from this bit*2
2. Enabling communication (changing SSICR.REN from 0 to 1).

[Clearing timing]
Clearing timing corresponding to the above clearing condition

1. When 0 is written to this bit after reading 1 from this bit (same as the timing in Figure 32.18)
2. 1 PCLKB cycle after writing 1 to SSICR.REN.*3

[Setting condition]

Reading from SSIFRDR while it is empty.

[Setting timing]

At completion of reading from SSIFRDR. See Figure 32.16.

Note 1. This bit is cleared by a software reset (SSIFCR.SSIRST = 1). The software reset has priority over all the clearing conditions described above.

Note 2. After reading 1 from this bit, this bit is cleared when one of the following three conditions is met:

- A software reset (SSIFCR.SSIRST = 1) is done.
- After 1 has been read, writing of 0 is complete.
- 1 PCLKB cycle passes after 1 has been written to SSICR.REN.

Note 3. After communication is enabled (by changing the value of SSICR.REN bit from 0 to 1), the reception error flags (RUIRQ and ROIRQ in the SSISR register) are cleared. If, however, the SSISR register is read continuously, the cleared status of the reception error flags might be unable to be read.

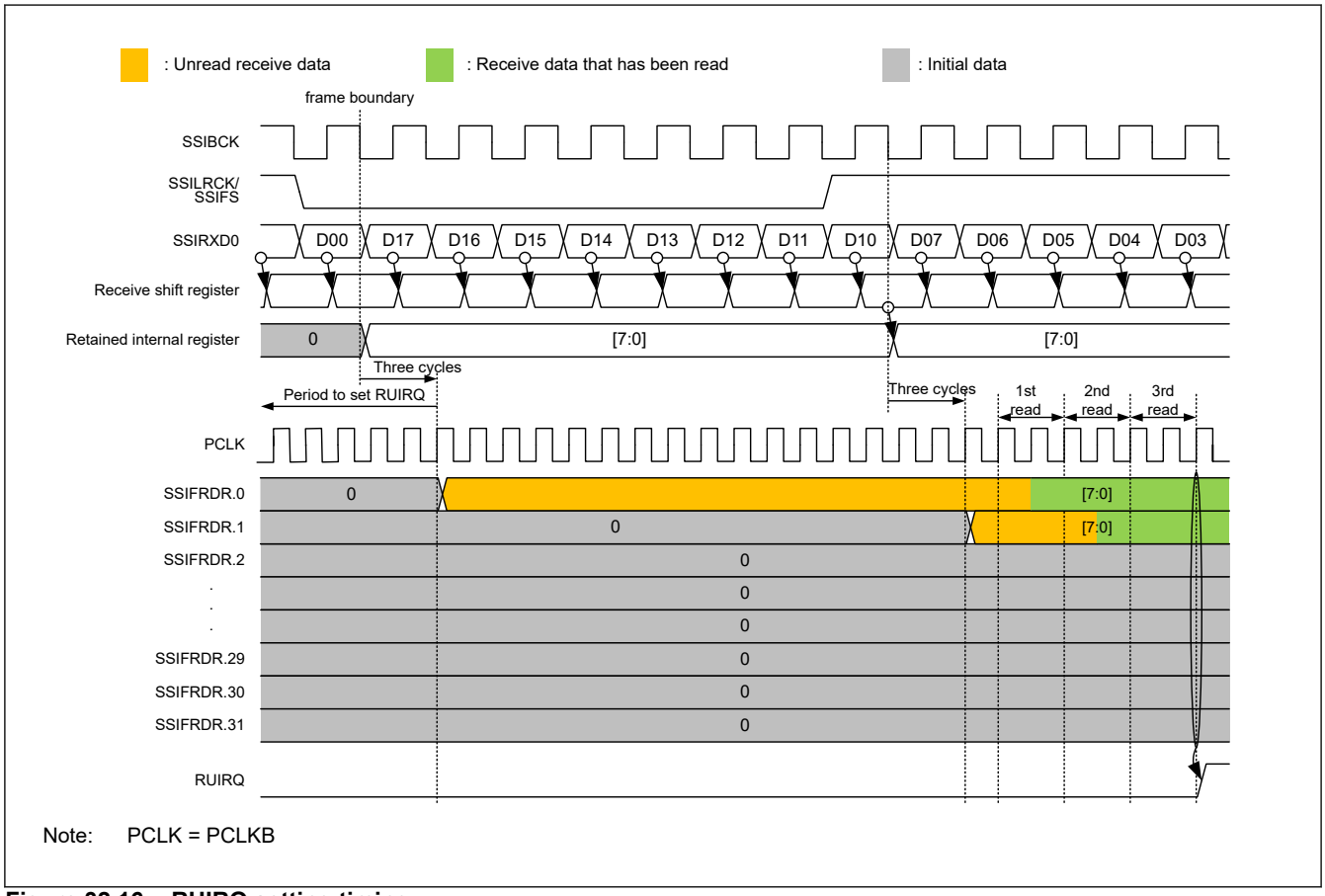



**Figure 32.16 RUIRQ setting timing**

### TOIRQ flag (Transmit Overflow Error Status Flag)

The TOIRQ flag is a status flag that indicates a transmit overflow error. This flag is set by automatic determination but it must be cleared by register access. This flag indicates that an attempt has been made to write data to the SSIFTDR register when the register is full of data. The data writing that causes a transmit overflow is ignored. For the procedure to recover from the overflow error, see section 32.8.6. Error Handling. This flag is not cleared by a transmit FIFO data register reset (SSIFCR.TFRST).

[Priority order for setting and clearing]

Setting is prioritized.*1

[Clearing condition]

When either of the following operations is done:

1. Writing 0 to this bit after reading 1 from this bit*2
2. Enabling communication (changing SSICR.TEN from 0 to 1).

[Clearing timing]

Clearing timing corresponding to the above clearing condition

1. When 0 is written to this bit after reading 1 from this bit (same as the timing in Figure 32.18)
2. 1 PCLKB cycle after writing 1 to SSICR.TEN.*3

[Setting condition]

An attempt was made to write data to the SSIFTDR register when the register is full of data.

[Setting timing]

At completion of writing to SSIFTDR. For details, see Figure 32.17.

Note 1. This bit is cleared by a software reset (SSIFCR.SSIRST = 1). The software reset has priority over all the clearing conditions described above.

Note 2. After reading 1 from this bit, this bit is cleared when one of the following three conditions is met:
- A software reset (SSIFCR.SSIRST = 1) is done.
- After 1 has been read, writing of 0 is complete.
- 1 PCLKB cycle passes after 1 has been written to SSICR.TEN.

Note 3. After communication is enabled (by changing the value of SSICR.TEN bit from 0 to 1), the transmission error flags (TOIRQ and TUIRQ in the SSISR register) are cleared. If, however, the SSISR register is read continuously, the cleared status of the transmission error flags might be unable to be read.

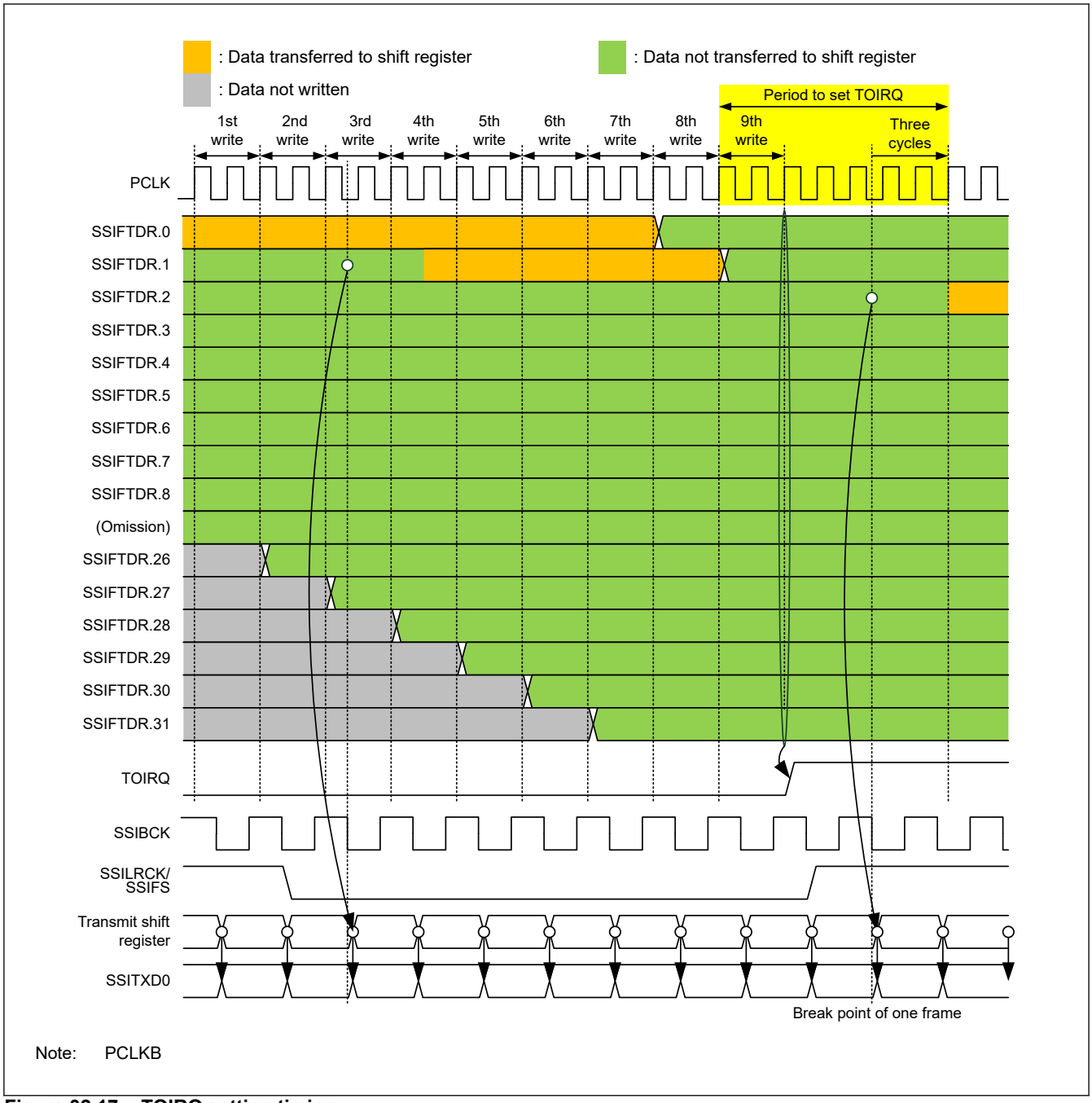



**Figure 32.17 TOIRQ setting timing**

### TUIRQ flag (Transmit Underflow Error Status flag)

The TUIRQ flag is a status flag that indicates a transmit underflow error. This flag is set by automatic determination but it must be cleared by register access. This flag indicates that writing the serial data required for a frame to SSIFTDR did not catch up with transmission of the frame. Even if this flag is cleared after it has been set, the SSITXD0 output remains to be 0. To output the data written to the transmit FIFO data register (SSIFTDR) to the SSITXD0 pin, follow the communication stop procedure in Figure 32.55 and error-handling procedure in Figure 32.56. For the procedure to recover from an error, see section 32.8.6. Error Handling. This flag is not cleared by a reset of transmit FIFO data register (by the SSIFCR.TFRST signal).

[Priority order for setting and clearing]

Setting is prioritized.*1

[Clearing condition]

When either of the following operations is done:

1. Writing 0 to this bit after reading 1 from this bit*2
2. Enabling communication (changing SSICR.TEN from 0 to 1).

[Clearing timing]

Clearing timing corresponding to the above clearing condition

1. When 0 is written to this bit after reading 1 from this bit
2. 1 PCLKB cycle after writing 1 to SSICR.TEN.*3

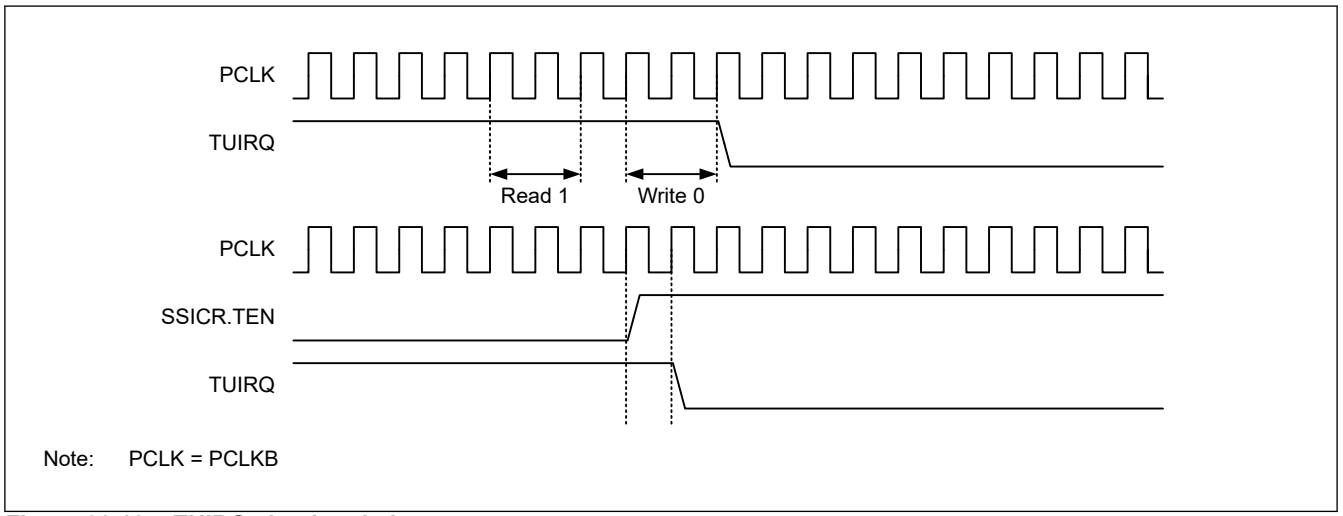



**Figure 32.18 TUIRQ clearing timing**

Note 1. This bit is cleared by a software reset (SSIFCR.SSIRST = 1). The software reset has priority over all the clearing conditions described above.

Note 2. After reading 1 from this bit, this bit is cleared when one of the following three conditions is met:
- A software reset (SSIFCR.SSIRST = 1) is done.
- After 1 has been read, writing of 0 is complete.
- 1 PCLKB cycle passes after 1 has been written to SSICR.TEN.

Note 3. After communication is enabled (by changing the value of SSICR.TEN bit from 0 to 1), the transmission error flags (TOIRQ and TUIRQ in the SSISR register) are cleared. If, however, the SSISR register is read continuously, the cleared status of the transmission error flags might be unable to be read.

[Setting condition]

When communication continues over a frame boundary, the transmit data required for the next frame has not been written to SSIFTDR. For details, see Figure 32.19 and Figure 32.20.

[Setting timing]

3 PCLKB cycles after the frame boundary. For details, see Figure 32.19.

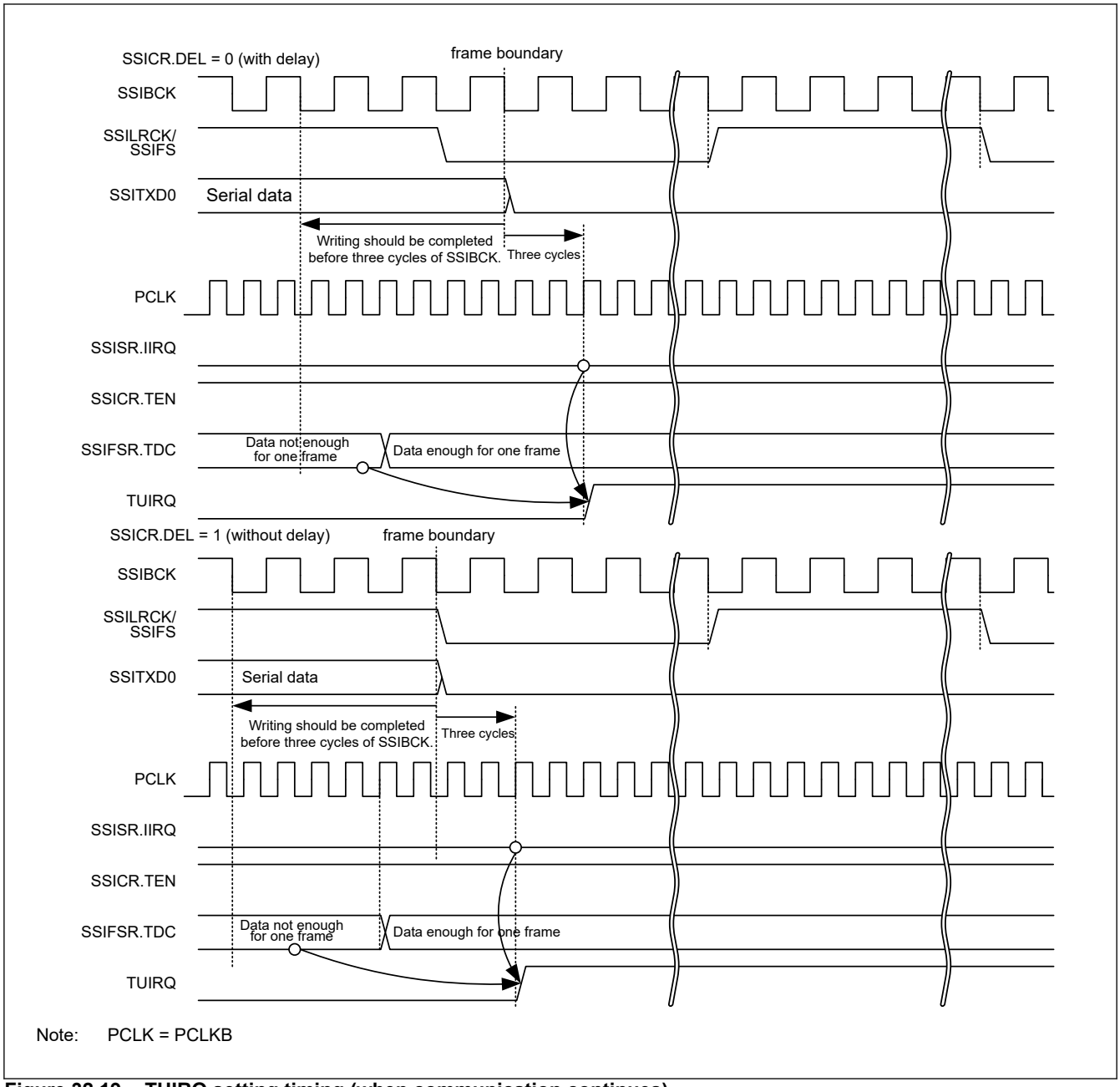



**Figure 32.19 TUIRQ setting timing (when communication continues)**

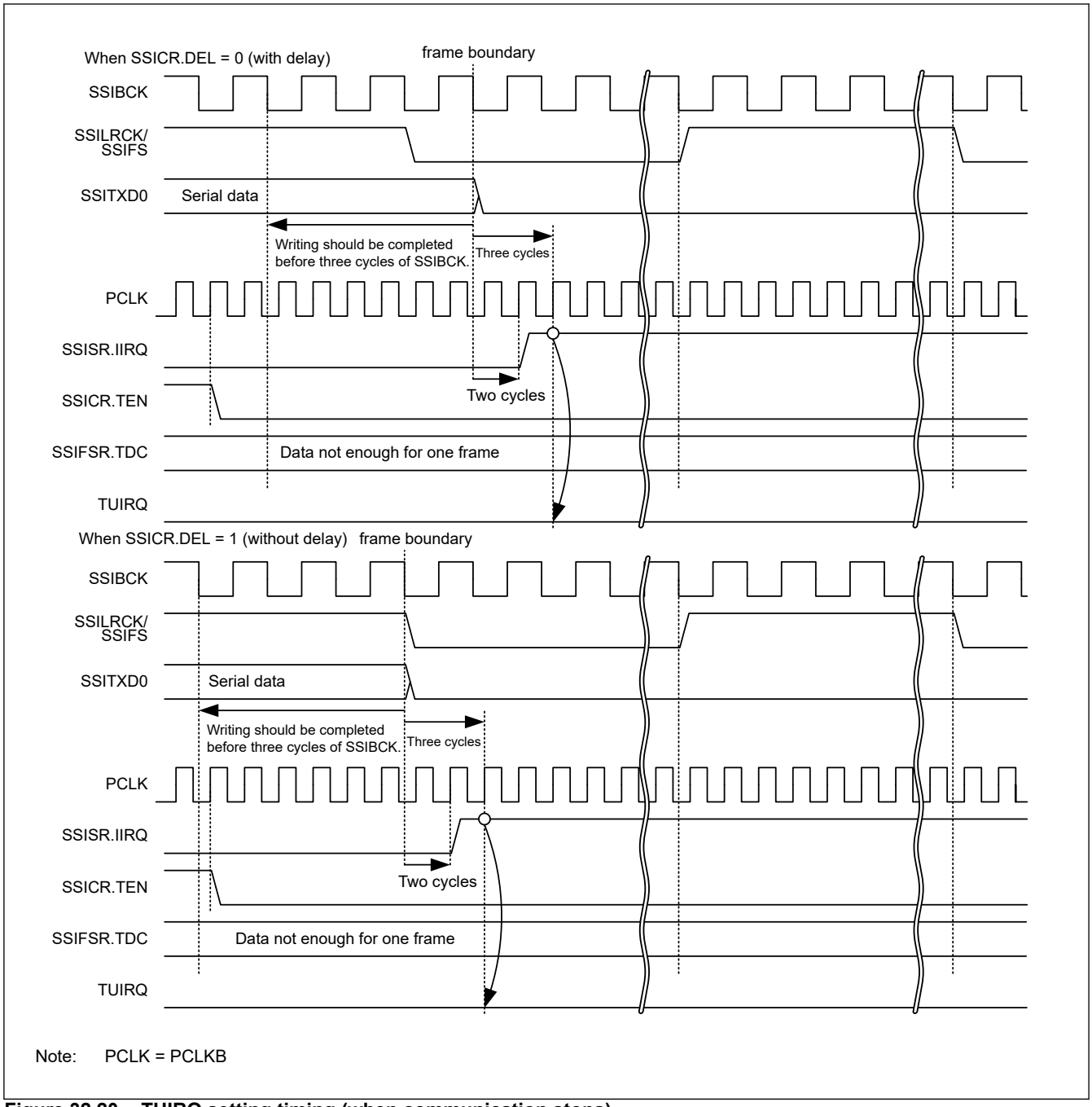



**Figure 32.20 TUIRQ setting timing (when communication stops)**

## 32.4.3 SSIFCR : FIFO Control Register

Base address: SSIE0 = 0x4009_D000

Offset address: 0x10

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | AUCKE | — | — | — | — | — | — | — | — | — | — | — | — | — | — | SSIRST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | BSW | — | — | — | — | — | — | — | TIE | RIE | TFRST | RFRST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RFRST | Receive FIFO Data Register Reset*1<br>0: Clears a receive data FIFO reset condition<br>1: Sets a receive data FIFO reset condition | R/W |
| 1 | TFRST | Transmit FIFO Data Register Reset*1<br>0: Clears a transmit data FIFO reset condition<br>1: Sets a transmit data FIFO reset condition | R/W |
| 2 | RIE | Receive Data Full Interrupt Output Enable<br>0: Disables receive data full interrupts<br>1: Enables receive data full interrupts | R/W |
| 3 | TIE | Transmit Data Empty Interrupt Output Enable<br>0: Disables transmit data empty interrupts<br>1: Enables transmit data empty interrupts | R/W |
| 10:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 11 | BSW | Byte Swap Enable*1<br>0: Disables byte swap<br>1: Enables byte swap | R/W |
| 15:12 | — | These bits are read as 0. The write value should be 0. | R/W |
| 16 | SSIRST | Software Reset<br>0: Clears a software reset condition<br>1: Sets a software reset condition | R/W |
| 30:17 | — | These bits are read as 0. The write value should be 0. | R/W |
| 31 | AUCKE | AUDIO_MCK Enable in Mastermode Communication*1<br>0: Disables supply of AUDIO_MCK<br>1: Enables supply of AUDIO_MCK | R/W |

Note 1. Writing to these bits while SSIE is in a communication state (SSISR.IIRQ = 0) is prohibited. If the value of these bits is changed by rewriting, subsequent operation is unpredictable.

This register sets a software reset, byte swap, and enable/disable of interrupt requests.

### RFRST bit (Receive FIFO Data Register Reset)

The RFRST bit sets a software reset of the receive FIFO data register (SSIFRDR). Writing 1 to this bit initializes the internal state related to SSIFRDR. The register bits subject to the software reset triggered by this bit are indicated by shading in Table 32.5. Because this bit is not automatically cleared after it has been set, write 0 to this bit to release the register bits from the software reset. After writing 0 to this bit, be sure to check that this bit is 0 before starting the next procedural step.

This bit is subject to the software reset by the SSIRST bit. Because the software reset by the SSIRST bit has priority over the reset by this bit, setting this bit is ignored when the SSIRST bit is set.

**Table 32.5 Bits subject to software reset by the RFRST bit**

| Symbol | Address (BASE+) | | +0 | | | | | | | | +1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SSICR | 0x00 | +0 | — | CKS | TUIEN | TOIEN | RUIEN | ROIEN | IIEN | — | FRM[1:0] | | DWL[2:0] | | | SWL[2:0] | | |
| | | +2 | — | MST | BCKP | LRCKP | SPDP | SDTA | PDTA | DEL | CKDV[3:0] | | | | MUEN | — | TEN | REN |
| SSISR | 0x04 | +0 | — | — | TUIRQ | TOIRQ | RUIRQ | ROIRQ | IIRQ | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| SSIFCR | 0x10 | +0 | AUCKE | — | — | — | — | — | — | — | — | — | — | — | — | — | — | SSIRST |
| | | +2 | — | — | — | — | BSW | — | — | — | — | — | — | — | TIE | RIE | TFRST | RFRST |
| SSIFSR | 0x14 | +0 | — | — | TDC[5:0] | | | | | | — | — | — | — | — | — | — | TDE |
| | | +2 | — | — | RDC[5:0] | | | | | | — | — | — | — | — | — | — | RDF |
| SSIFTDR | 0x18 | +0 | FTDR[31:16] | | | | | | | | | | | | | | | |
| | | +2 | FTDR[15:0] | | | | | | | | | | | | | | | |
| SSIFRDR | 0x1C | +0 | FRDR[31:16] | | | | | | | | | | | | | | | |
| | | +2 | FRDR[15:0] | | | | | | | | | | | | | | | |
| SSIOFR | 0x20 | +0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | — | — | — | BCKASTP | LRCONT | — | — | — | — | — | — | OMOD[1:0] | |
| SSISCR | 0x24 | +0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | TDES[4:0] | | | | | — | — | — | RDFS[4:0] | | | | |

### TFRST bit (Transmit FIFO Data Register Reset)

The TFRST bit sets a software reset of the transmit FIFO data register (SSIFTDR). Writing 1 to this bit initializes the internal state related to SSIFTDR. The register bits subject to the software reset triggered by this bit are indicated by shading in Table 32.6. Because this bit is not automatically cleared after it has been set, write 0 to this bit to release the register bits from the software reset. After writing 0 to this bit, be sure to check that this bit is 0 before starting the next procedural step.

This bit is subject to the software reset by the SSIRST bit. Because the software reset by the SSIRST bit has priority over the reset by this bit, setting this bit is ignored when the SSIRST bit is set.

**Table 32.6 Bits subject to software reset by the TFRST bit (1 of 2)**

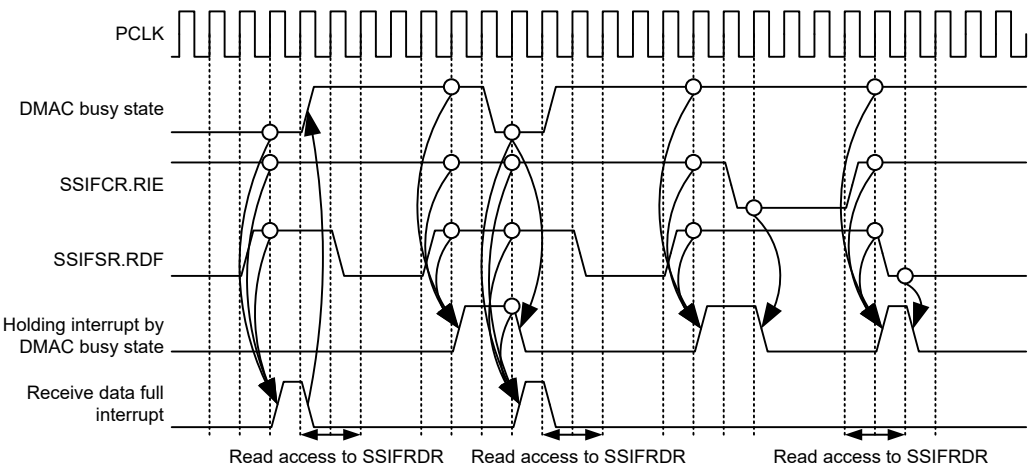
| Symbol | Address (BASE+) | | +0 | | | | | | | | +1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SSICR | 0x00 | +0 | — | CKS | TUIEN | TOIEN | RUIEN | ROIEN | IIEN | — | FRM[1:0] | | DWL[2:0] | | | SWL[2:0] | | |
| | | +2 | — | MST | BCKP | LRCKP | SPDP | SDTA | PDTA | DEL | CKDV[3:0] | | | | MUEN | — | TEN | REN |
| SSISR | 0x04 | +0 | — | — | TUIRQ | TOIRQ | RUIRQ | ROIRQ | IIRQ | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |

**Table 32.6 Bits subject to software reset by the TFRST bit (2 of 2)**

| Symbol | Address (BASE+) | | +0 | | | | | | | | +1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SSIFCR | 0x10 | +0 | AUCKE | — | — | — | — | — | — | — | — | — | — | — | — | — | — | SSIRST |
| | | +2 | — | — | — | — | BSW | — | — | — | — | — | — | — | TIE | RIE | TFRST | RFRST |
| SSIFSR | 0x14 | +0 | — | — | TDC[5:0] | | | | | | — | — | — | — | — | — | — | TDE |
| | | +2 | — | — | RDC[5:0] | | | | | | — | — | — | — | — | — | — | RDF |
| SSIFTDR | 0x18 | +0 | FTDR[31:16] | | | | | | | | | | | | | | | |
| | | +2 | FTDR[15:0] | | | | | | | | | | | | | | | |
| SSIFRDR | 0x1C | +0 | FRDR[31:16] | | | | | | | | | | | | | | | |
| | | +2 | FRDR[15:0] | | | | | | | | | | | | | | | |
| SSIOFR | 0x20 | +0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | — | — | — | BCKASTP | LRCONT | — | — | — | — | — | — | OMOD[1:0] | |
| SSISCR | 0x24 | +0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | TDES[4:0] | | | | | — | — | — | RDFS[4:0] | | | | |

### RIE bit (Receive Data Full Interrupt Output Enable)

The RIE bit enables/disables output of receive data full interrupts. Use a receive data full interrupt as an interrupt to trigger data reading from the receive FIFO data register. Write 1 to this bit after specifying the setting condition for receive data full interrupt (by using the SSISCR.RDFS bit). Figure 32.21 shows the timing of generating the receive data full interrupt.

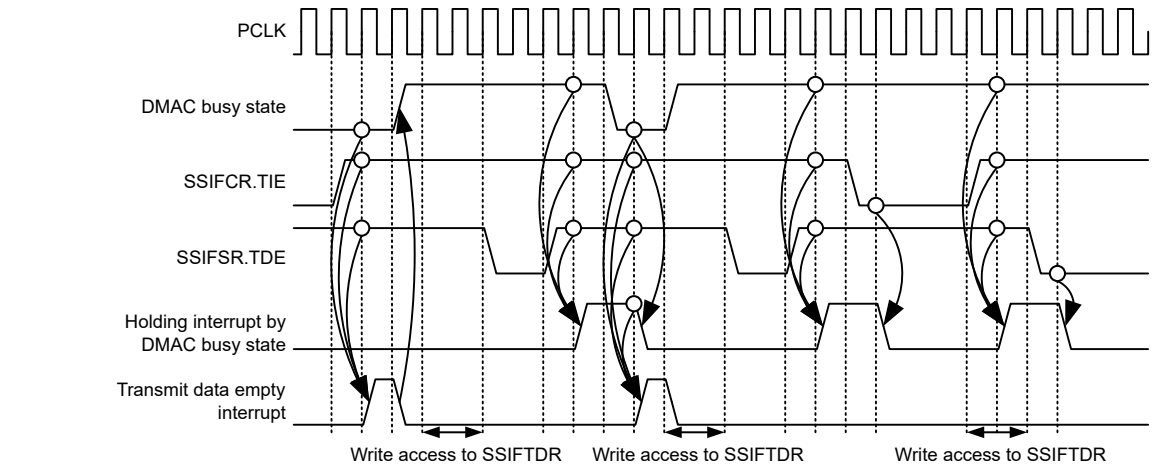



Note: PCLK = PCLKB

**Figure 32.21 Timing of receive data full interrupt**

### TIE bit (Transmit Data Empty Interrupt Output Enable)

The TIE bit enables/disables output of transmit data empty interrupts. Use a transmit data empty interrupt as an interrupt to trigger data writing to the transmit FIFO data register. Write 1 to this bit after specifying the setting condition for transmit data empty interrupt (by using the SSISCR.TDES bit). Figure 32.22 shows the timing of generating the transmit data empty interrupt.




Note: PCLK = PCLKB

**Figure 32.22 Timing of transmit data empty interrupt**

### BSW bit (Byte Swap Enable)

The BSW bit enables/disables byte swap of register access for the transmit FIFO data register (SSIFTDR) and the receive FIFO data register (SSIFRDR). This bit is valid only with 16-bit access or 32-bit access to SSIFTDR and SSIFRDR. For details, see Figure 32.23.

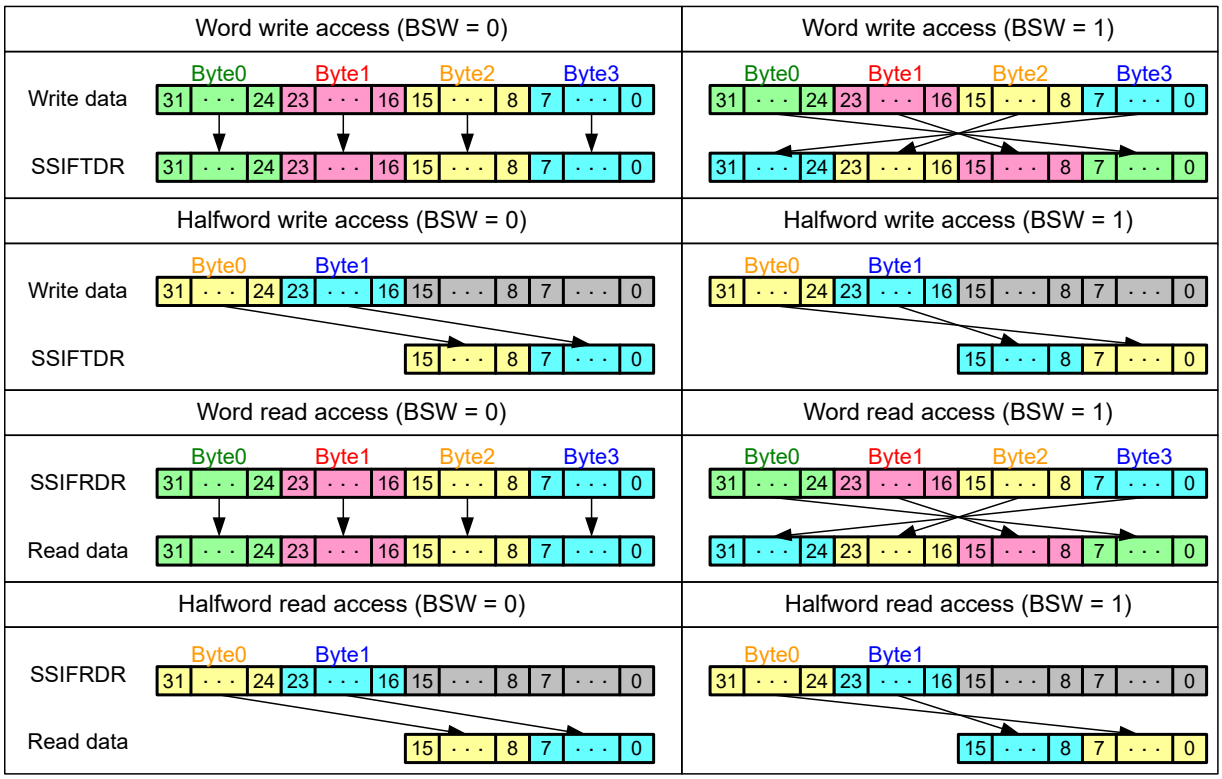



**Figure 32.23 Operation example of byte swap**

### SSIRST bit (Software Reset)

The SSIRST bit sets a software reset of SSIE. Writing 1 to this bit initializes the internal state of SSIE. The register bits subject to the software reset triggered by this bit are indicated by shading in Table 32.7. Because this bit is not automatically

cleared after it has been set, write 0 to this bit to release the register bits from the software reset. After writing 0 to this bit, be sure to check that this bit is 0 before starting the next procedural step.

To stop communication of SSIE immediately, after turning off the peripheral functions, write 1 to this bit. Initialization by a software reset is performed without any relation with the bit clock.

**Table 32.7 Bits subject to software reset by the SSIRST bit**

| Symbol | Address (BASE+) | | +0 | | | | | | | | +1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SSICR | 0x00 | +0 | — | CKS | TUIEN | TOIEN | RUIEN | ROIEN | IIEN | — | FRM[1:0] | | DWL[2:0] | | | SWL[2:0] | | |
| | | +2 | — | MST | BCKP | LRCKP | SPDP | SDTA | PDTA | DEL | CKDV[3:0] | | | | MUEN | — | TEN | REN |
| SSISR | 0x04 | +0 | — | — | TUIRQ | TOIRQ | RUIRQ | ROIRQ | IIRQ | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| SSIFCR | 0x10 | +0 | AUCKE | — | — | — | — | — | — | — | — | — | — | — | — | — | — | SSIRST |
| | | +2 | — | — | — | — | BSW | — | — | — | — | — | — | — | TIE | RIE | TFRST | RFRST |
| SSIFSR | 0x14 | +0 | — | — | TDC[5:0] | | | | | | — | — | — | — | — | — | — | TDE |
| | | +2 | — | — | RDC[5:0] | | | | | | — | — | — | — | — | — | — | RDF |
| SSIFTDR | 0x18 | +0 | FTDR[31:16] | | | | | | | | | | | | | | | |
| | | +2 | FTDR[15:0] | | | | | | | | | | | | | | | |
| SSIFRDR | 0x1C | +0 | FRDR[31:16] | | | | | | | | | | | | | | | |
| | | +2 | FRDR[15:0] | | | | | | | | | | | | | | | |
| SSIOFR | 0x20 | +0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | — | — | — | BCKASTP | LRCONT | — | — | — | — | — | — | OMOD[1:0] | |
| SSISCR | 0x24 | +0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | TDES[4:0] | | | | | — | — | — | RDFS[4:0] | | | | |

### AUCKE bit (AUDIO_MCK Enable in Mastermode Communication)

The AUCKE bit enables/disables supply to AUDIO_MCK while in master-mode communication (MST = 1).

Changing the value of this bit must be performed only after specifying the settings related to AUDIO_MCK (by using the CKS, MST, BCKP, and CKDV bits in the SSICR register).

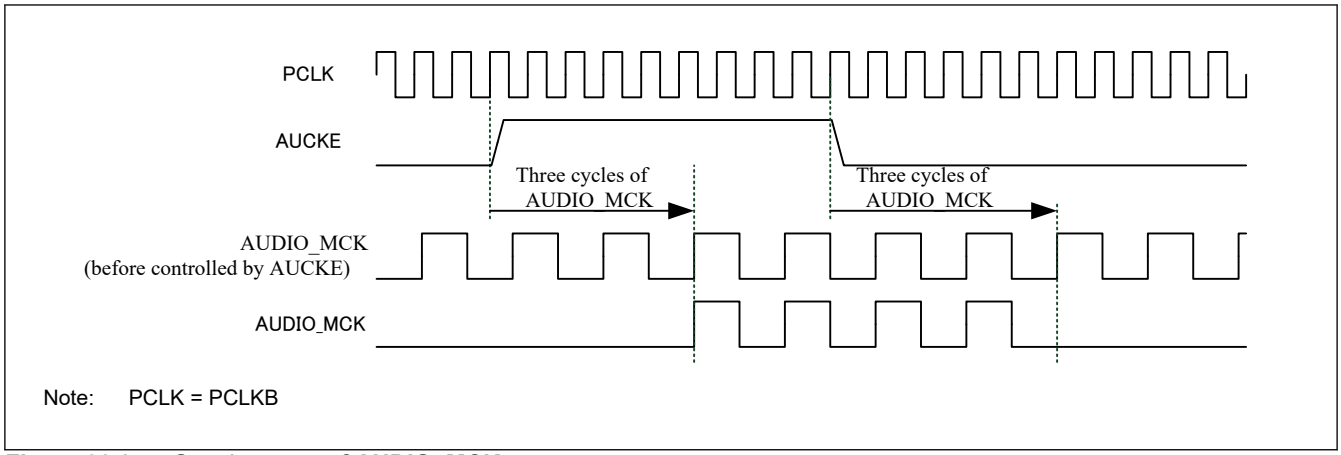



**Figure 32.24 Stop/resume of AUDIO_MCK**

Note: In slave-mode communication (SSICR.MST = 0), SSIE needs supply of SSIBCK. To stop BCK on the master side, make sure that SSIE is in the idle state (SSISR.IIRQ = 1). If BCK is stopped before SSIE becomes idle, take the procedure to start communication in Figure 32.51 or wait for an idle state by taking the procedure to resume communication in Figure 32.57.

In master-mode communication (SSICR.MST = 1), SSIE operates with the audio clock (AUDIO_MCK). To stop SSIE completely, make sure that SSIE is in the idle state (SSISR.IIRQ = 1) and then write 0 to SSIFCR.AUCKE. If 0 is written to SSIFCR.AUCKE before SSIE becomes idle, take the procedure to start communication in Figure 32.51.

Figure 32.25 and Figure 32.26 show the timings of signal operation in the period from setting this bit to 1 to the output to the SSIBCK pin.

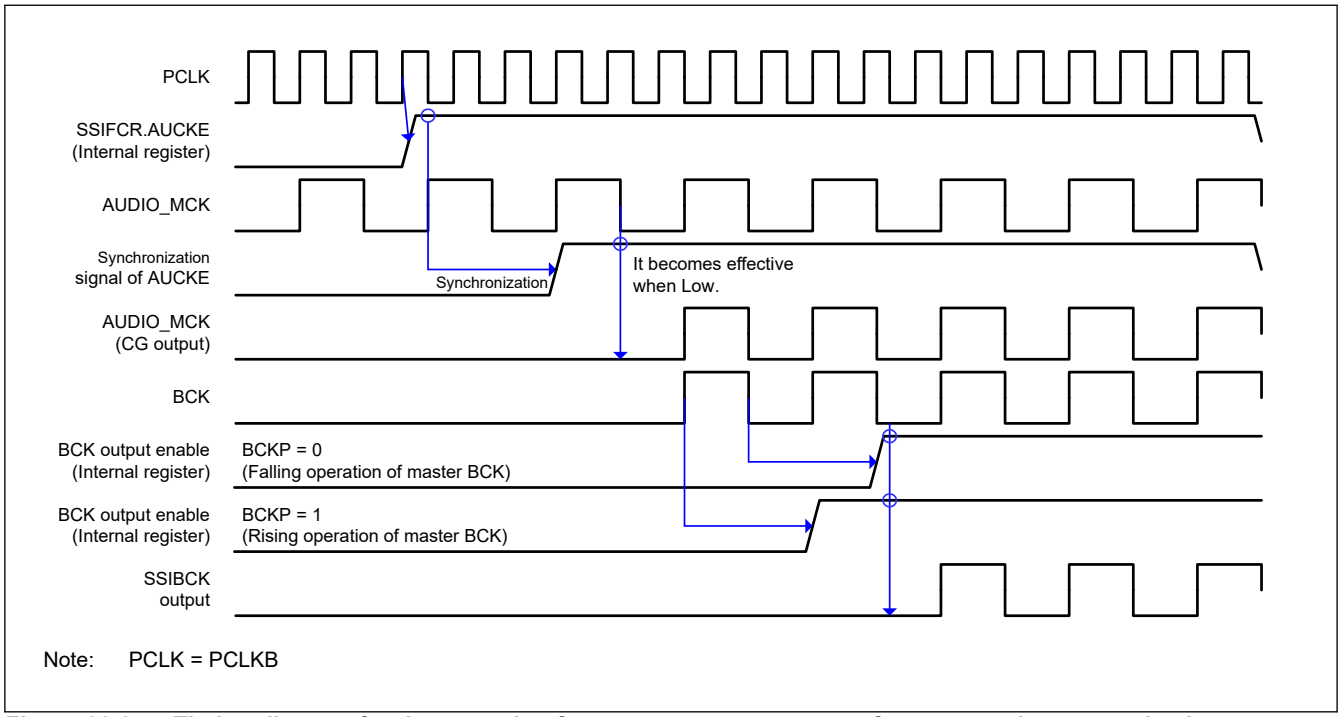



**Figure 32.25 Timing diagram for the operation from system reset to start of master-mode communication**

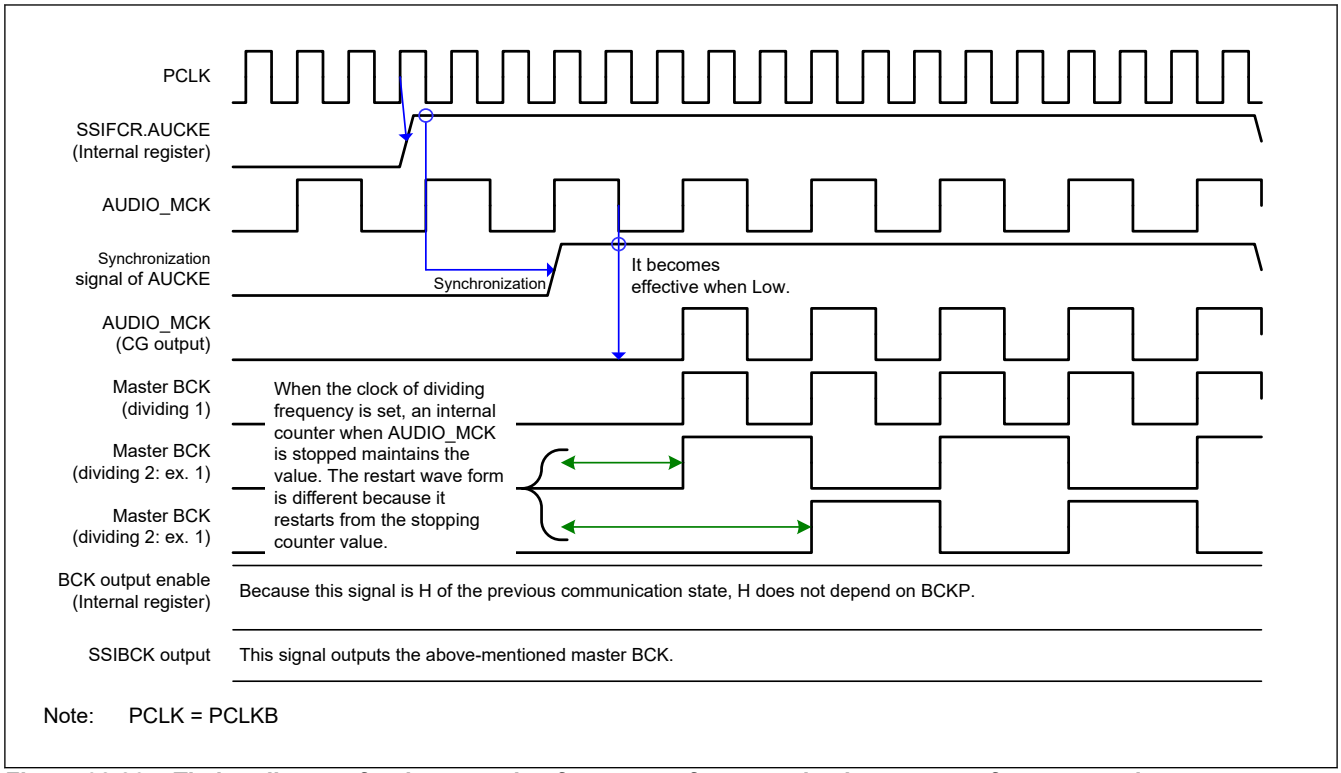



**Figure 32.26 Timing diagram for the operation from stop of communication to start of master-mode communication**

Note: If the supply of AUDIO_MCK stops, the value of the SSIBCK pin is held. Therefore, the SSIBCK signal might stop in the H (high level) state.

## 32.4.4 SSIFSR : FIFO Status Register

Base address: SSIE0 = 0x4009_D000

Offset address: 0x14

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | TDC[5:0] | | | | | | — | — | — | — | — | — | — | TDE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | RDC[5:0] | | | | | | — | — | — | — | — | — | — | RDF |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RDF | Receive Data Full Flag<br>0: The size of received data in SSIFRDR is not more than the value of SSISCR.RDFS.<br>1: The size of received data in SSIFRDR is not less than the value of SSISCR.RDFS plus one. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13:8 | RDC[5:0] | Receive Data Count<br>Number of valid data stored in the receive FIFO data register | R |
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 16 | TDE | Transmit Data Empty Flag<br>0: The free space of SSIFTDR is not more than the value of SSISCR.TDES.<br>1: The free space of SSIFTDR is not less than the value of SSISCR.TDES plus one. | R/W |
| 23:17 | — | These bits are read as 0. The write value should be 0. | R/W |
| 29:24 | TDC[5:0] | Transmit Data Count<br>Number of valid data stored in the transmit FIFO data register | R |
| 31:30 | — | These bits are read as 0. The write value should be 0. | R/W |

This register is configured with status flags that indicate the status of the transmit FIFO data register and the receive FIFO data register.

### RDF flag (Receive Data Full Flag)

The RDF flag indicates that the receive FIFO data register (SSIFRDR) has unread received data not less than the amount set with the SSISCR.RDFS bit plus one. This flag is set by automatic determination but it must be cleared by register access.

[Priority order for setting and clearing]

Clearing is prioritized.

[Clearing condition]

Either of the following two:*1

1. Writing 0 to this bit after reading 1 from this bit (CPU operation)*2
2. Last access (DTC/DMAC operation) to read data from SSIFRDR by an interrupt routine using the DTC and DMAC.

[Clearing timing]

Clearing timing corresponding to the above clearing condition

1. When 0 is written to this bit after reading 1 from this bit (same as the timing in Figure 32.18)
2. After the PCLKB cycle in which the last access instruction is issued to read data from SSIFRDR by an interrupt routine using the DTC and DMAC.

[Setting condition]

SSIFTDR has free space not less than the amount set with the SSIFCR.TTRG bit plus one.

[Setting timing]

At completion of transfer from the shift register that results in SSIFRDR having data not less than the amount set with the SSISCR.RDFS bit plus one.

Note 1. These bits are cleared by a software reset (SSIFCR.SSIRST = 1) and receive FIFO data register reset (SSIFCR.RFRST = 1). Reset conditions available for these bits are the software reset and receive FIFO data register reset as well as the clearing conditions described above.

Note 2. After reading 1 from this bit, this bit is cleared when one of the following four conditions is met:
- A software reset is done (SSIFCR.SSIRST = 1).
- A receive FIFO data register reset is done (SSIFCR.RFRST = 1).
- After 1 has been read, writing of 0 is complete.
- Last access is performed to read data from SSIFRDR by an interrupt routine using the DTC and DMAC.

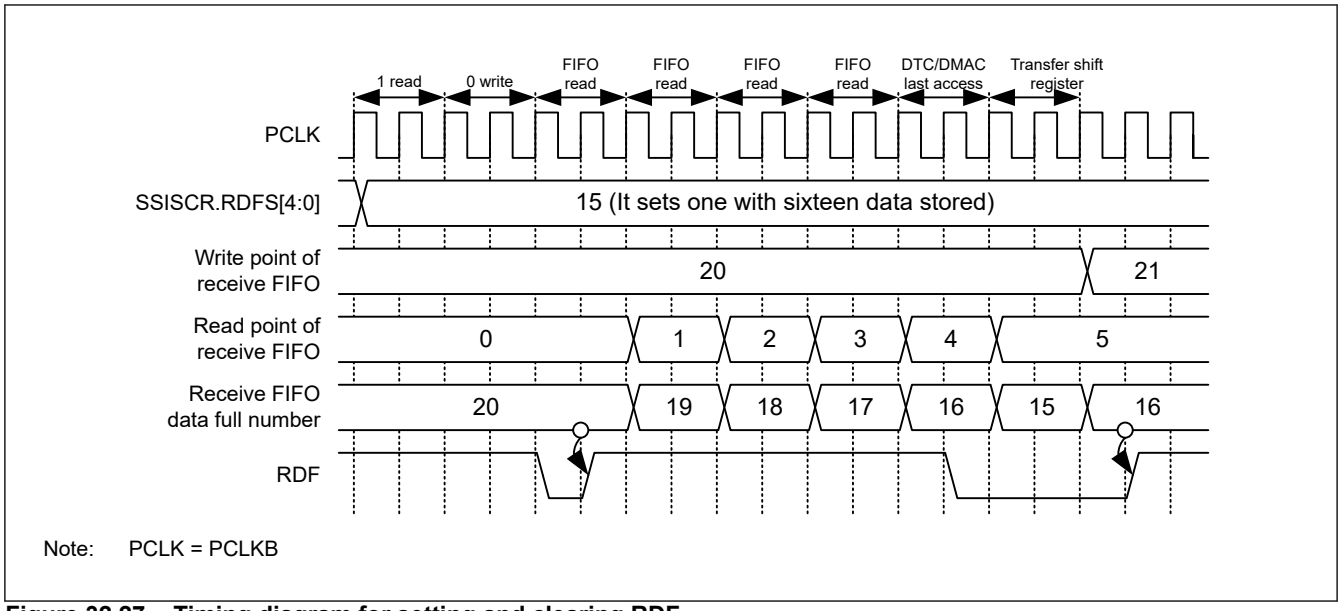



**Figure 32.27 Timing diagram for setting and clearing RDF**

### RDC[5:0] bits (Receive Data Count)

The RDC[5:0] bits indicate the number of valid data that are stored in the receive FIFO data register (SSIFRDR). With these bits as 0x00, there is no received data. With 0x20, the register is filled with received data and there is no free space.

### TDE flag (Transmit Data Empty Flag)

The TDE flag indicates that the transmit FIFO data register (SSIFTDR) has free space not less than the amount set with the SSIFCR.TTRG bit plus one. This flag is set by automatic determination but it must be cleared by register access.

[Priority order for setting and clearing]

Clearing is prioritized.*1

[Clearing condition]

Either of the following two:

1. Writing 0 to this bit after reading 1 from this bit (CPU operation)*2
2. Last access (DTC/DMAC operation) to write data to SSIFTDR by an interrupt routine using the DTC and DMAC.

[Clearing timing]

Clearing timing corresponding to the above clearing condition

1. When 0 is written to this bit after reading 1 from this bit (same as the timing in Figure 32.18)
2. Last access (DTC/DMAC operation) to write data to SSIFTDR by an interrupt routine using the DTC and DMAC.

[Setting condition]

SSIFTDR has free space not less than the amount set with the SSIFCR.TTRG bit plus one.

[Setting timing]

While operating on PCLKB, SSIFTDR is found to have free space not less than “size set in the SSISCR.TDES bits + 1.”

Note 1. This bit is cleared by a software reset (SSIFCR.SSIRST = 1) and transmit FIFO data register reset (SSIFCR.TFRST = 1). The software reset and transmit FIFO data register reset have priority over all the clearing conditions described above.

Note 2. After reading 1 from this bit, this bit is cleared when one of the following four conditions is met:
- A software reset is done (SSIFCR.SSIRST = 1).
- A transmit FIFO data register reset is done (SSIFCR.TFRST = 1).
- After 1 has been read, writing of 0 is complete.
- Last access is performed to write data to SSIFTDR by an interrupt routine using the DTC and DMAC.

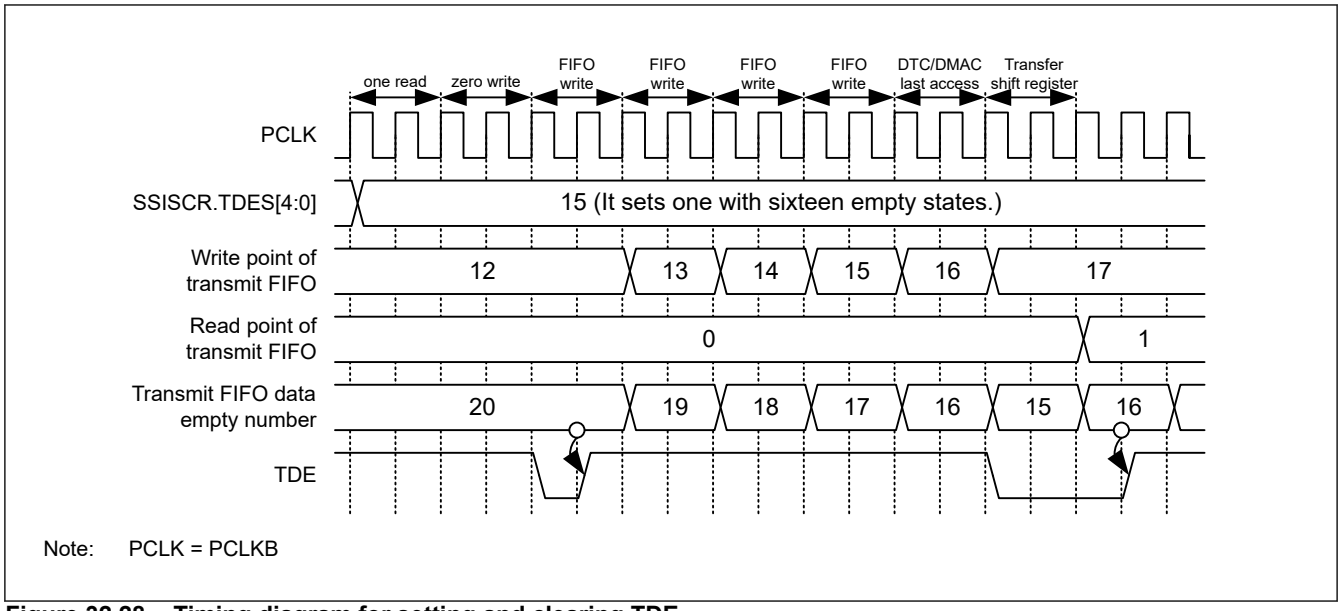



**Figure 32.28 Timing diagram for setting and clearing TDE**

### TDC[5:0] bits (Transmit Data Count)

The TDC[5:0] bits indicate the number of valid data that are stored in the transmit FIFO data register (SSIFTDR). With these bits as 0x00, there is no data to be transmitted. With 0x20, there is no space to write data.

## 32.4.5 SSIFTDR : Transmit FIFO Data Register

Base address: SSIE0 = 0x4009_D000

Offset address: 0x18

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SSIFTDR[31:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | SSIFTDR[31:0] | Transmit FIFO Data | W |

This register stores data to be serially transmitted. 0 is returned when this register is read.

When you use this register for transmission, specify data writing to this register as the DTC/DMAC operation that is triggered by a transmit data empty interrupt. Determine the access size to this register according to the data word length to be communicated in Table 32.8.

**Table 32.8 Register access restriction to FIFOs (1 of 2)**

| Access Size | | | | |
|---|---|---|---|---|
| SSICR.DWL[2:0] | Data Word Length | Byte | Halfword | Word |
| 000b | 8 | ✓ | — | — |
| 001b | 16 | — | ✓ | — |
| 010b | 18 | — | — | ✓ |
| 011b | 20 | — | — | ✓ |
| 100b | 22 | — | — | ✓ |
| 101b | 24 | — | — | ✓ |
| 110b | 32 | — | — | ✓ |

**Table 32.8 Register access restriction to FIFOs (2 of 2)**

| Access Size | | | | |
|---|---|---|---|---|
| SSICR.DWL[2:0] | Data Word Length | Byte | Halfword | Word |
| 111b | Setting prohibited | — | — | — |

Figure 32.29 shows register access to the transmit FIFO data register.

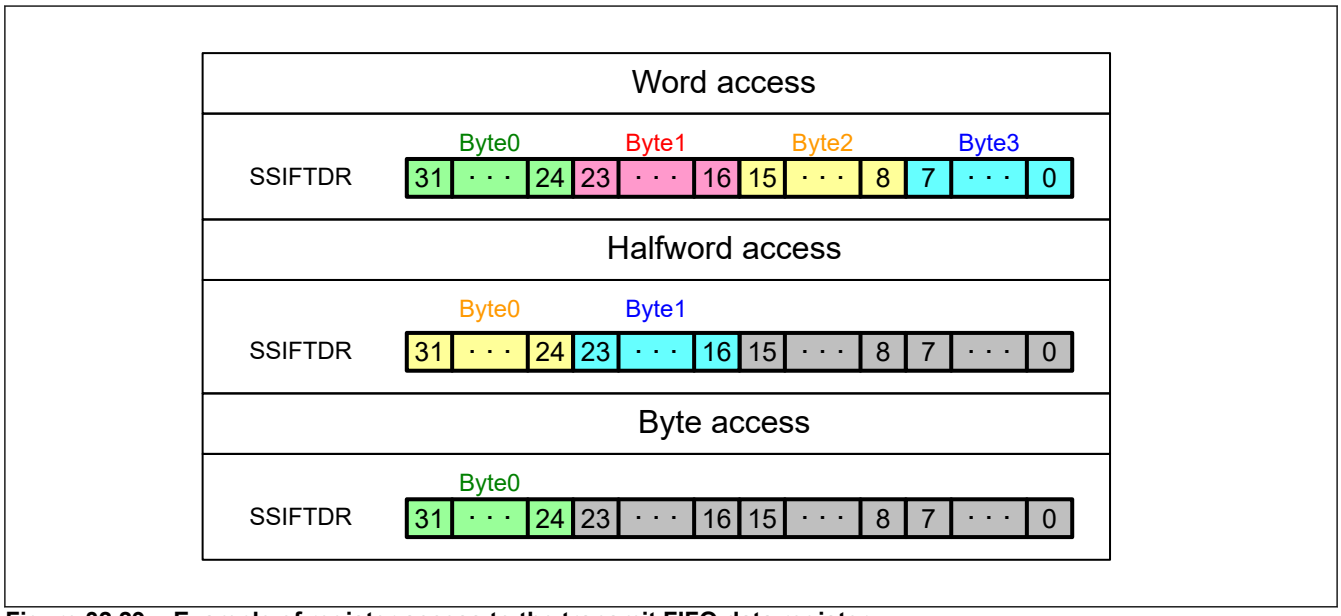
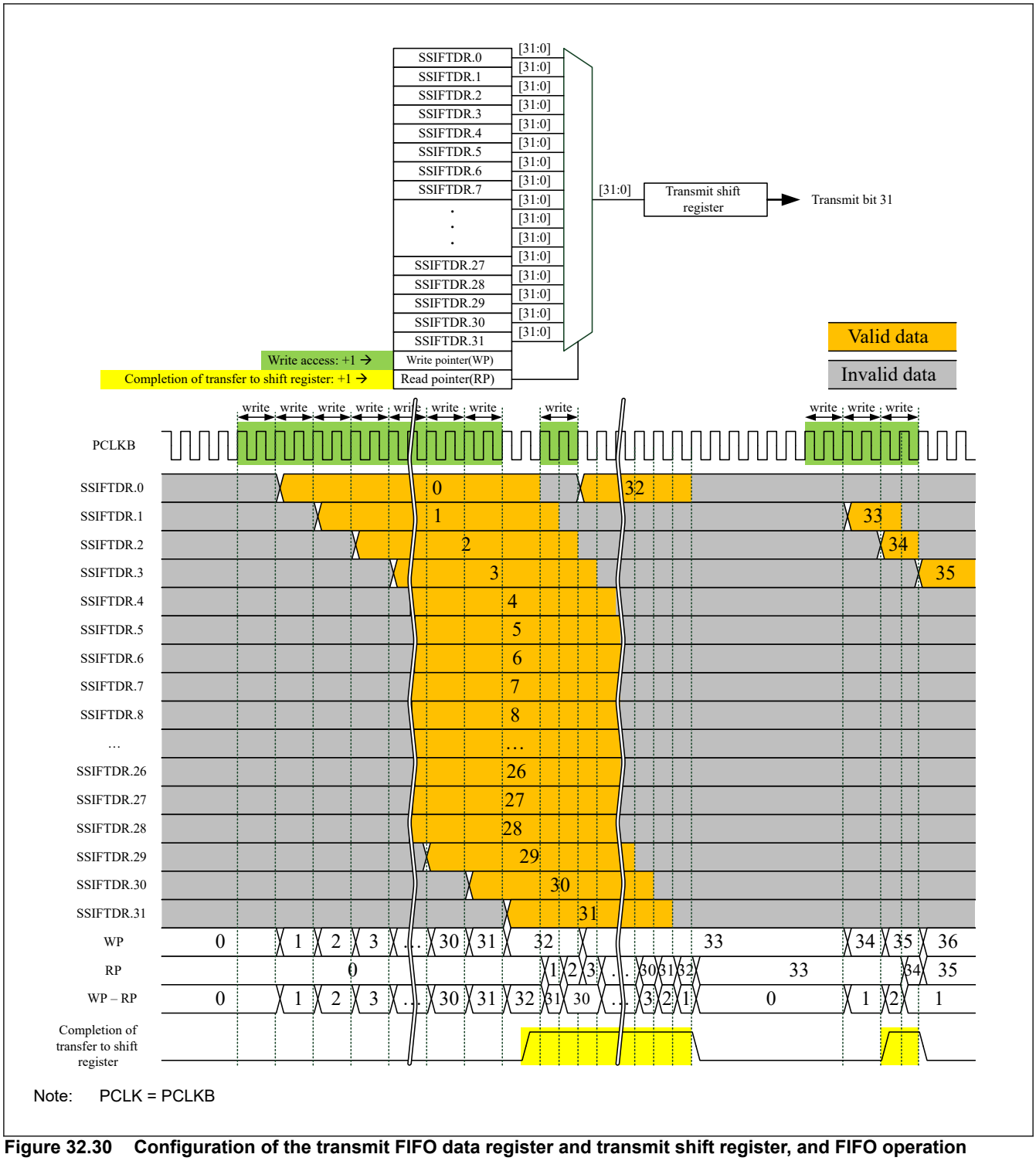



**Figure 32.29 Example of register access to the transmit FIFO data register**

Figure 32.30 shows the configurations and operation examples of the transmit FIFO data register and transmit shift register. The configurations are for storing data to FIFO and not related with communication.




Note: PCLK = PCLKB

**Figure 32.30 Configuration of the transmit FIFO data register and transmit shift register, and FIFO operation example**

### 32.4.6 SSIFRDR : Receive FIFO Data Register

Base address: SSIE0 = 0x4009_D000

Offset address: 0x1C

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SSIFRDR[31:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | SSIFRDR[31:0] | Receive FIFO Data | R |

When you use this register for reception, specify data reading from this register as the DTC/DMAC operation that is triggered by a transmit data empty interrupt. Determine the access size to this register according to the data word length to be communicated in Table 32.8.

Register access to the receive FIFO data register is same as for the transmit FIFO data register.

Figure 32.31 shows the configurations and operation examples of the receive FIFO data register and receive shift register.

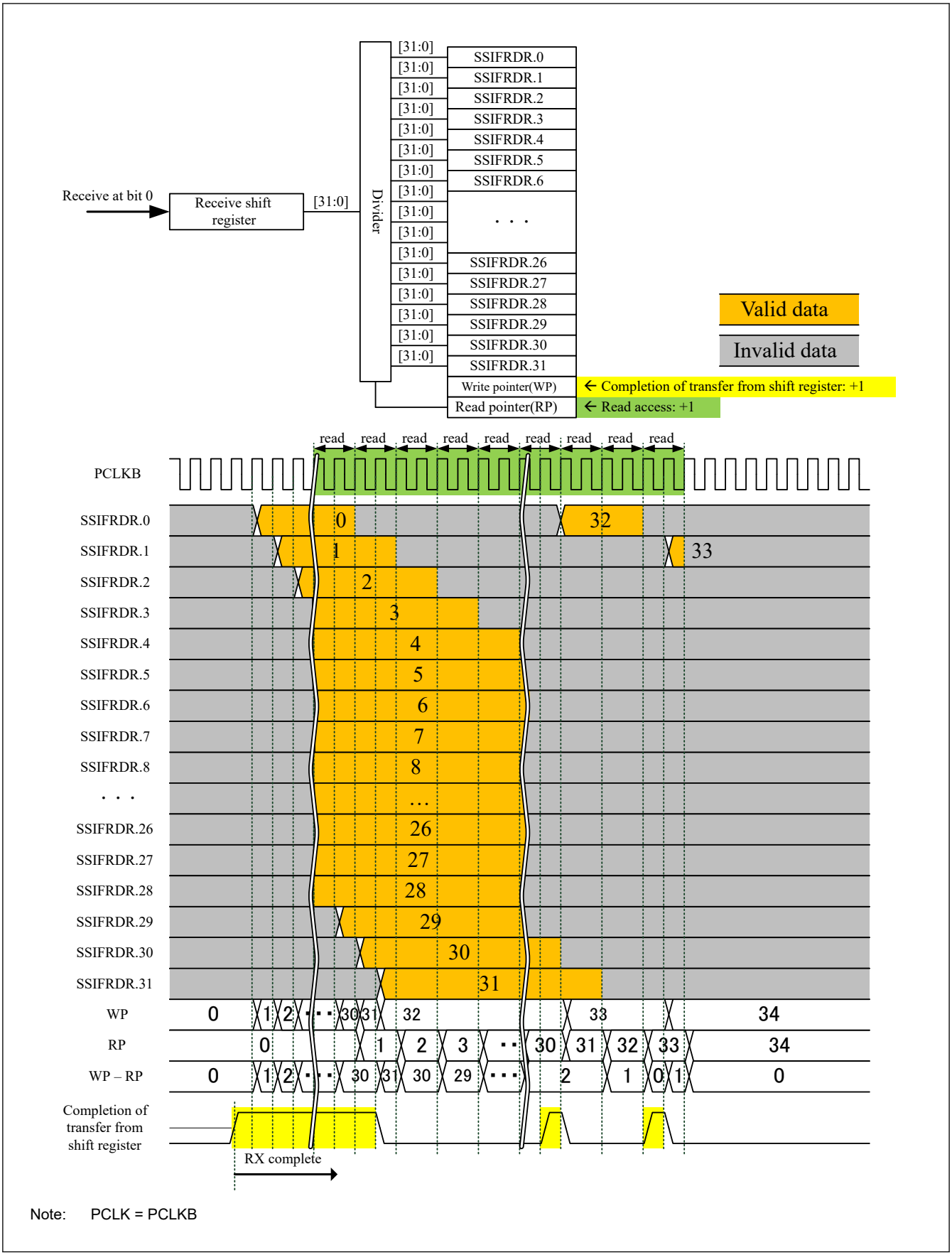



**Figure 32.31 Configuration of the receive FIFO data register and receive shift register, and FIFO operation example**

## 32.4.7 SSIOFR : Audio Format Register

Base address: SSIE0 = 0x4009_D000

Offset address: 0x20

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | BCKASTP | LRCONT | — | — | — | — | — | — | OMOD[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | OMOD[1:0] | Audio Format Select*3 *4<br>0 0: $I^2S$ format<br>0 1: TDM format<br>1 0: Monaural format<br>1 1: Setting prohibited | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | LRCONT | Whether to Enable LRCK/FS Continuation*1 *2<br>0: Disables LRCK/FS continuation<br>1: Enables LRCK/FS continuation | R/W |
| 9 | BCKASTP | Whether to Enable Stopping BCK Output When SSIE is in Idle Status*1 *2<br>0: Always outputs BCK to the SSIBCK pin<br>1: Automatically controls output of BCK to the SSIBCK pin | R/W |
| 31:10 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. This bit is valid only in master-mode communication (SSICR.MST = 1). The setting is invalid in slave-mode communication (SSICR.MST = 0).

Note 2. The BCKASTP and LRCONT bits must not be set to 1 together.

Note 3. While SSIE is communicating (SSISR.IIRQ = 0), writing to these bits is prohibited. If the value of these bits is changed by writing, subsequent operation is unpredictable.

Note 4. If the communication format of other-party device is compatible with a communication format of SSIE, specify and use the communication format that enables communication with the other-party device.

This register is used to set an audio format (which involves the settings of communication format, LR clock/frame synchronization continuation mode, and BCK output stop).

### OMOD[1:0] bits (Audio Format Select)

The OMOD[1:0] bits set an audio format. Writing to these bits must be performed when the LR clock supply to the SSILRCK/SSIFS pin is stopped. For details about the output of LR clock, see the detailed description of the LRCONT bit in section 32.4.7. SSIOFR : Audio Format Register.

### LRCONT bit (Whether to Enable LRCK/FS Continuation)

The LRCONT bit enables or disables the output from SSILRCK/SSIFS pin when the communication mode is master-mode communication (SSICR.MST = 1) and SSIE is in the idle state (SSISR.IIRQ = 1).

Even in the idle state, a signal can output from the SSILRCK/SSIFS pin when this bit is set to 1 (to enable LR clock/frame synchronization continuation) in master mode (SSICR.MST = 1).

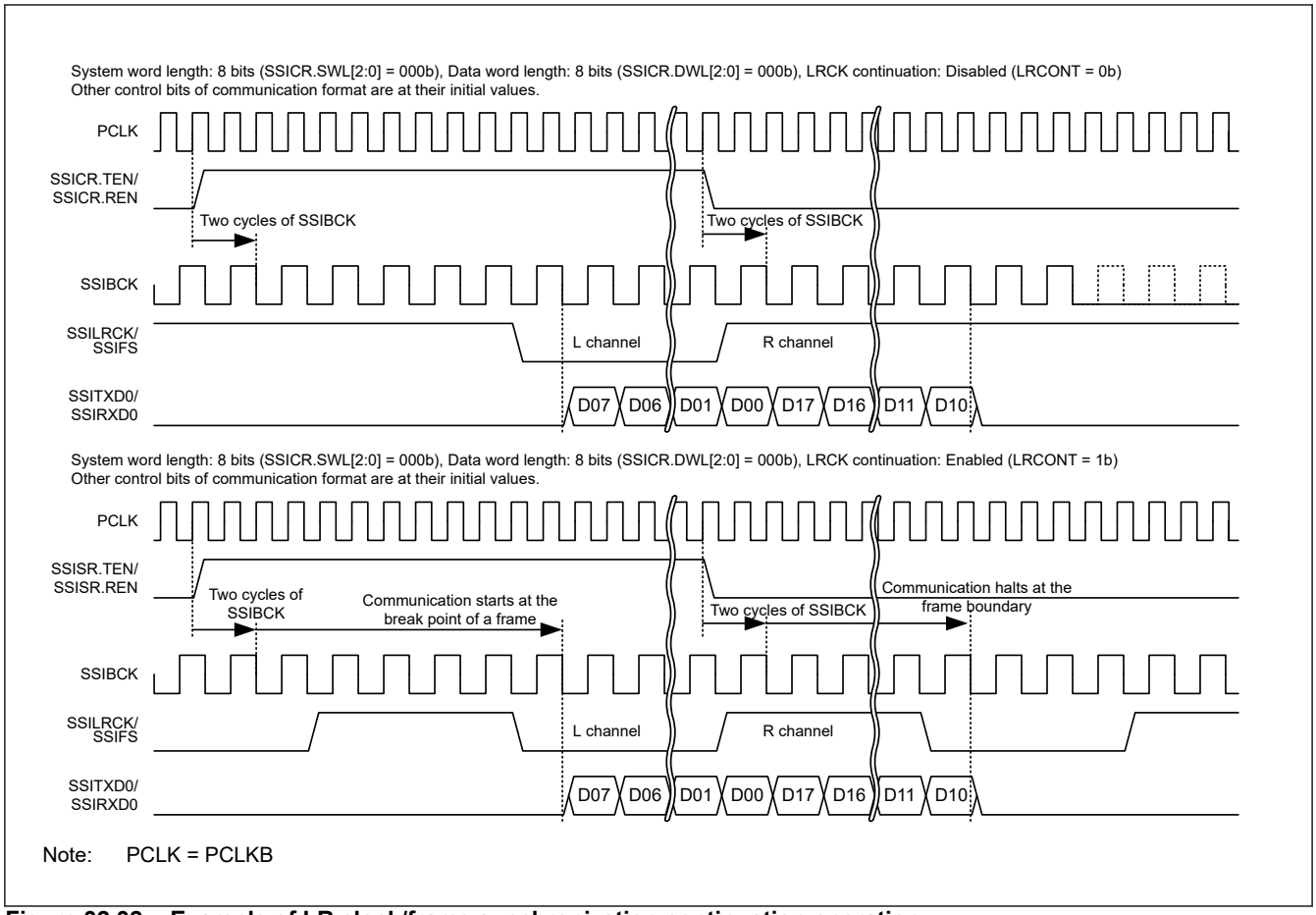



**Figure 32.32 Example of LR clock/frame synchronization continuation operation**

### BCKASTP bit (Whether to Enable Stopping BCK Output When SSIE is in Idle Status)

The BCKASTP bit turns on or off the function to output BCK to the SSIBCK pin according to the communication shown in Figure 32.33 and Figure 32.34 in master-mode communication (SSICR.MST = 1).

Changing the value of this bit must be performed only after setting the communication format to be used.

This bit must be used in the following way:

Write 0 to the BCKASTP bit, and then start communication. During the communication, write 1 to the BCKASTP bit. By this operation, the bit clock output to the SSIBCK pin stops automatically when the communication stops. To resume the communication, set SSIE to the idle state (SSICR.IIRQ = 1), enable the supply of AUDIO_MCK (SSIFCR.AUCKE = 1), and then write 0 to the BCKASTP bit.

When the communication mode is master-mode communication (SSICR.MST = 1) and SSIE is in the idle state (SSICR.IIRQ = 1):

**Table 32.9 BCKASTP bit status and SSIBCK pin output**

| BCKASTP Bit | SSIBCK Pin Output Status |
|---|---|
| 0 | Output |
| 1 | Stopped |

Note: The BCKASTP bit cannot be used when the other-party device (which is a slave) requires the clock output from the SSIBCK pin before and during communication. In such a case, use the BCKASTP bit to stop the clock only after communication. For the timing of enabling the clock stop function, see Figure 32.33.

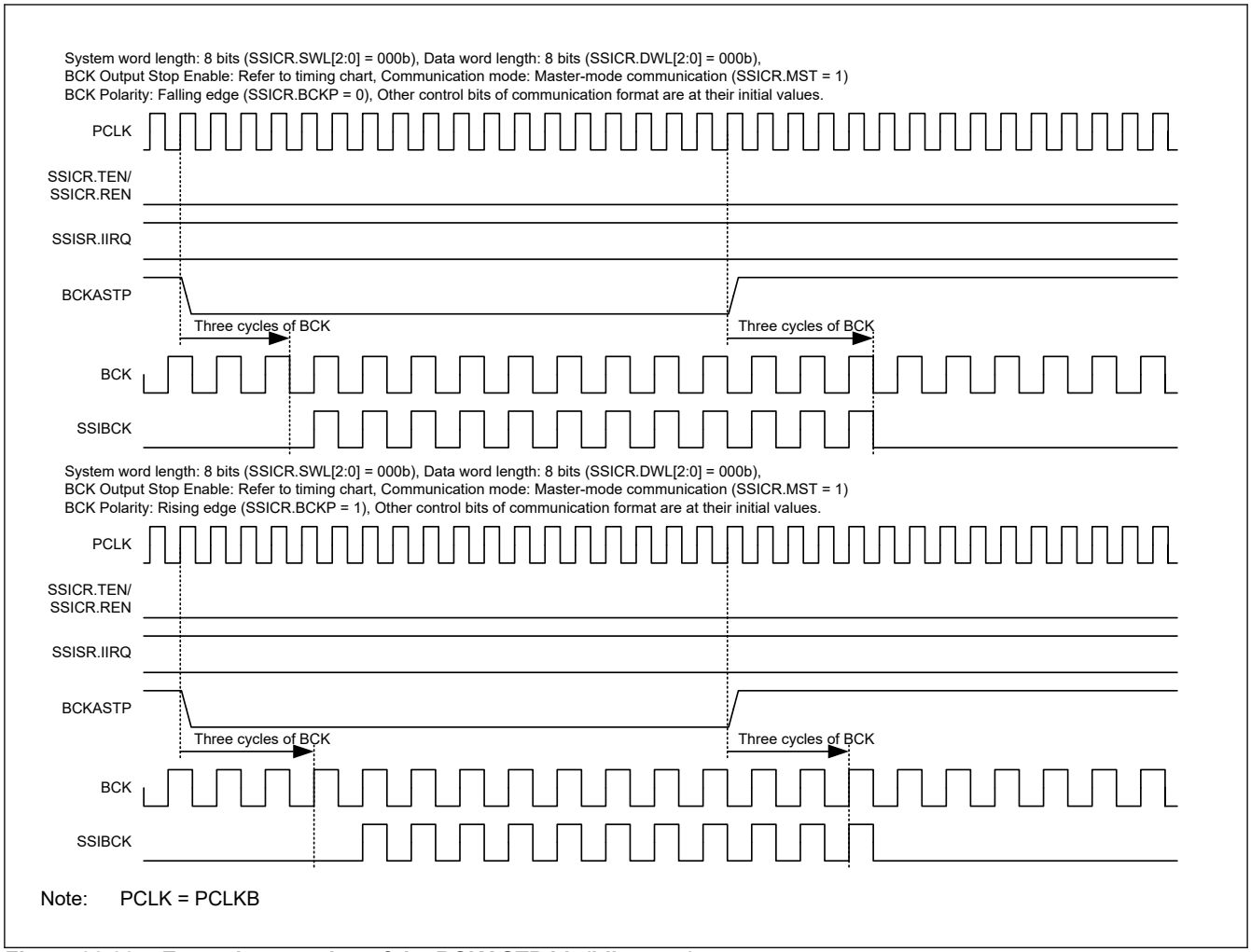



**Figure 32.33 Example operation of the BCKASTP bit (idle state)**

When the communication mode is master-mode communication (SSICR.MST = 1) and the BCK output stop function is enabled (BCKASTP = 1):

Details of the BCK output to the SSIBCK pin are as follows:

Output start timing: BCK is output in appropriate timing so that a valid edge is generated when the LR clock/frame synchronization signal shifts to a valid value.

Output stop timing: 1 to 1.5 clock cycles after a frame boundary.

For details about the timings, see the timing diagram in Figure 32.34.

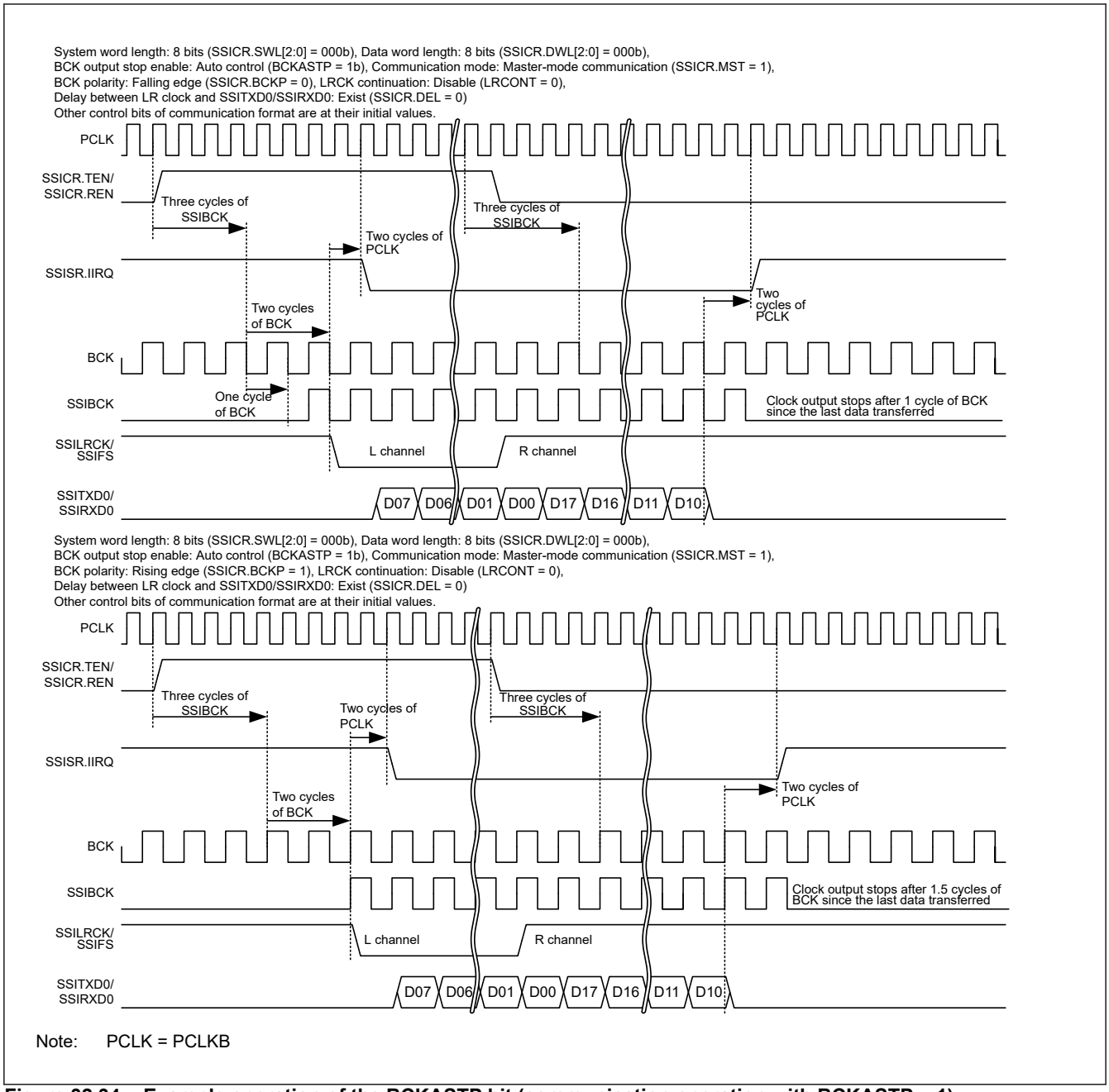



**Figure 32.34 Example operation of the BCKASTP bit (communication operation with BCKASTP = 1)**

## 32.4.8 SSISCR : Status Control Register

Base address: SSIE0 = 0x4009_D000

Offset address: 0x24

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | TDES[4:0] | | | | | — | — | — | RDFS[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | RDFS[4:0] | RDF Setting Condition Select*1<br>0x00: SSIFRDR has one stage or more data size.<br>0x01: SSIFRDR has two stages or more data size.<br>⋮<br>0x1E: SSIFRDR has thirty-one stages or more data size.<br>0x1F: SSIFRDR has thirty-two stages or more data size. | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 12:8 | TDES[4:0] | TDE Setting Condition Select*1<br>0x00: SSIFTDR has one stage or more free space.<br>0x01: SSIFTDR has two stages or more free space.<br>⋮<br>0x1E: SSIFTDR has thirty-one stages or more free space.<br>0x1F: SSIFTDR has thirty-two stages or more free space. | R/W |
| 31:13 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Writing to these bits while SSIE is in a communication state (SSISR.IIRQ = 0) is prohibited. If written, the operation performed immediately after writing is not guaranteed.

### RDFS[4:0] bits (RDF Setting Condition Select)

The RDFS[4:0] bits set the setting condition of the receive data full flag (RDF).

### TDES[4:0] bits (TDE Setting Condition Select)

The TDES[4:0] bits set the setting condition of the transmit data empty flag (TDE).

## 32.5 Communication Formats

SSIE supports three communication formats. Table 32.10 shows supported communication formats.

**Table 32.10 Supported communication formats**

| Communication Format | SSIOFR.OMOD[1:0] |
|---|---|
| I²S format | 00 |
| TDM format | 01 |
| Monaural format | 10 |

The following describes the serial data structure shared by communication formats. A serial data structure is defined by the system word length (set in SSICR.SWL[2:0]) and the data word length (set in SSICR.DWL[2:0]). If the data word length is shorter than the system word length, padding bits are transferred in the serial data. For details, see Figure 32.35.

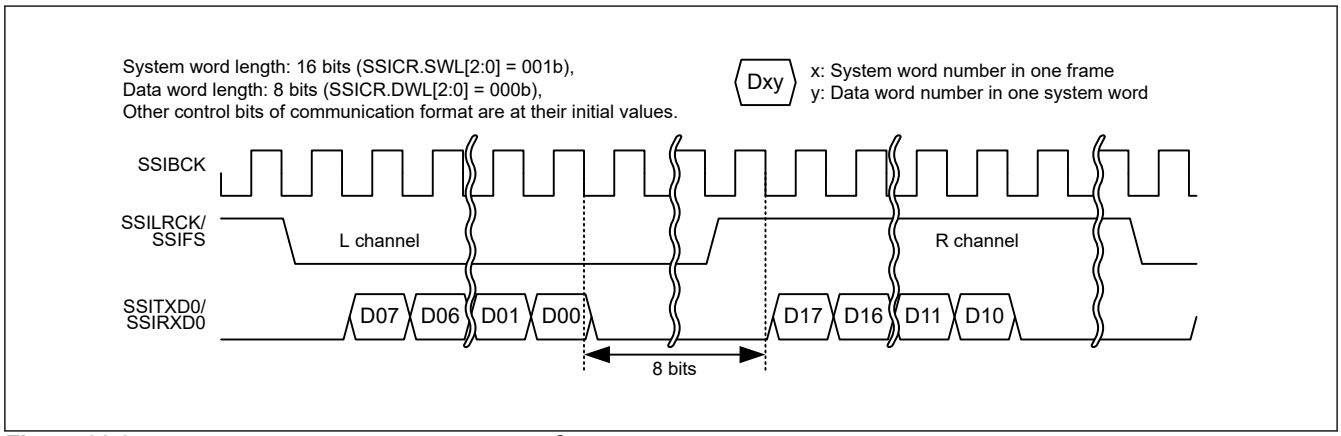



**Figure 32.35 Example of padding bit transfer (I²S format: system word length > data word length)**

Table 32.11 lists the number of padding bits to be transferred with each combination of system word length (SSICR.SWL[2:0]) and data word length (SSICR.DWL[2:0]). "-" indicates that the setting is prohibited.

**Table 32.11 Number of padding bits**

| | SSICR.DWL[2:0] | 000b | 001b | 010b | 011b | 100b | 101b | 110b | 111b |
|---|---|---|---|---|---|---|---|---|---|
| SSICR.SWL[2:0] | System Word Length | 8 | 16 | 18 | 20 | 22 | 24 | 32 | Setting prohibited |
| 000b | 8 | 0 | — | — | — | — | — | — | — |
| 001b | 16 | 8 | 0 | — | — | — | — | — | — |
| 010b | 24 | 16 | 8 | 6 | 4 | 2 | 0 | — | — |
| 011b | 32 | 24 | 16 | 14 | 12 | 10 | 8 | 0 | — |
| 100b | 48 | 40 | 32 | 30 | 28 | 26 | 24 | 16 | — |
| 101b | 64 | 56 | 48 | 46 | 44 | 42 | 40 | 32 | — |
| 110b | 128 | 120 | 112 | 110 | 108 | 106 | 104 | 96 | — |
| 111b | 256 | 248 | 240 | 238 | 236 | 234 | 232 | 224 | — |

## 32.5.1 I²S Format

The I²S format is a communication format used for connection with I²S-compatible serial devices. With this format setting (SSIOFR.OMOD[1:0] = 00b), one frame is configured with two system words, one for the channel L and the other for channel R. The SSILRCK/SSIFS signals are at a low level for the channel L and at a high level for the channel R. Set the polarity of the signals with the SSICR.LRCKP bit. Figure 32.36 shows the I²S format without padding. See Figure 32.35 for the format with padding.

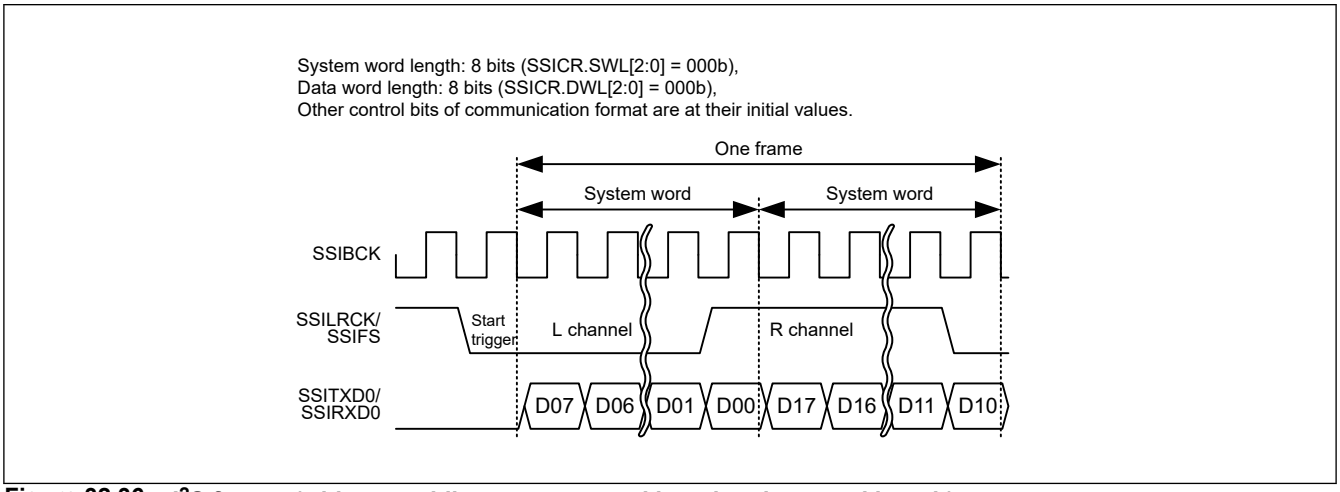



**Figure 32.36 I²S format (without padding: system word length = data word length)**

For the state of external pins when SSIE is in the idle state, see section 32.7.1. Idle State.

Note: SSIE has the SSILRCK/SSIFS pin, which indicates the synchronization of communication. When SSIE is in slave mode (SSICR.MST = 0), the communication format SSIE uses must match that of the other-party device to communicate. SSIE uses the signal input by the SSILRCK/SSIFS pin only as a trigger to start communication.

## 32.5.2 Monaural Format

The monaural format is a communication format used for connection with monaural-compatible serial devices. When the monaural format is specified (SSIOFR.OMOD[1:0] = 10b) for use, one frame consists of one system word. Also, a rising edge of the SSILRCK/SSIFS signal indicates a communication start trigger. Figure 32.37 and Figure 32.38 respectively show the monaural formats without and with padding.

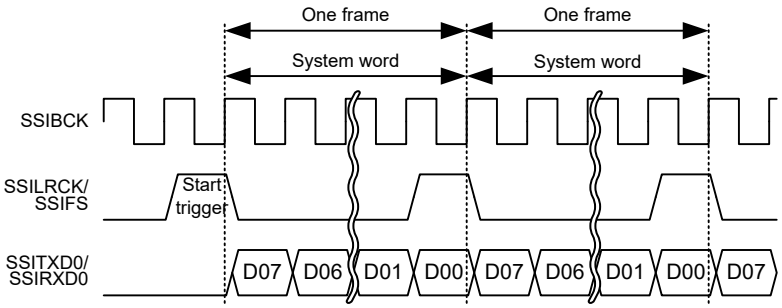



**Figure 32.37 Short frame in monaural format (without padding: system word length = data word length)**

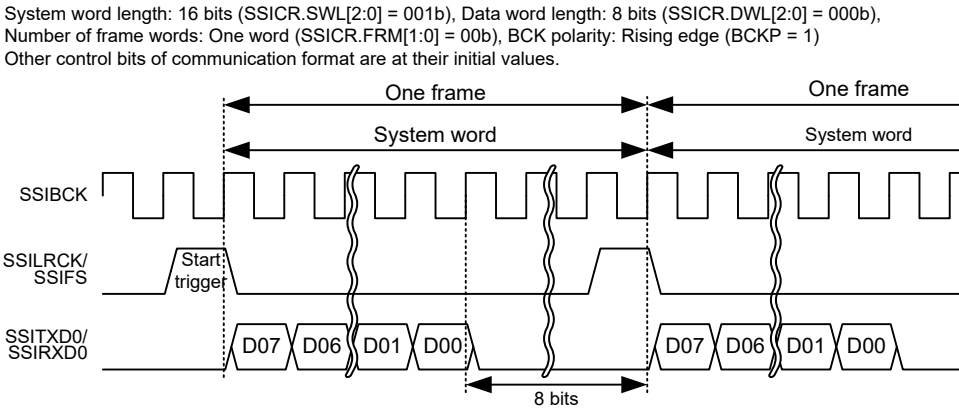



**Figure 32.38 Short frame in monaural format (with padding: system word length > data word length)**

The monaural formats supported by SSIE consist of short frames and long frames. See section 32.5.2.1. Short frame and section 32.5.2.2. Long frame for the difference between these two frames.

For the state of external pins state when SSIE is in the idle state, see section 32.7.1. Idle State.

Note: SSIE has the SSILRCK/SSIFS pin, which indicates the synchronization of communication. When SSIE is in slave mode (SSICR.MST = 0), the communication format SSIE uses must match that of the other-party device to communicate. SSIE uses the signal input by the SSILRCK/SSIFS pin only as a trigger to start communication.

### 32.5.2.1 Short frame

When a short frame is used (SSICR.DEL = 0), the SSILRCK/SSIFS signal indicating the start of serial data is set to high level only for 1 cycle of SSIBCK. Data transfer starts at the falling edge of the signal.

### 32.5.2.2 Long frame

When a long frame is used (SSICR.DEL = 1), the SSILRCK/SSIFS signal indicating the start of serial data is set to high level only for 2 cycles of SSIBCK. See Figure 32.39. Data transfer starts at the rising edge of the signal.

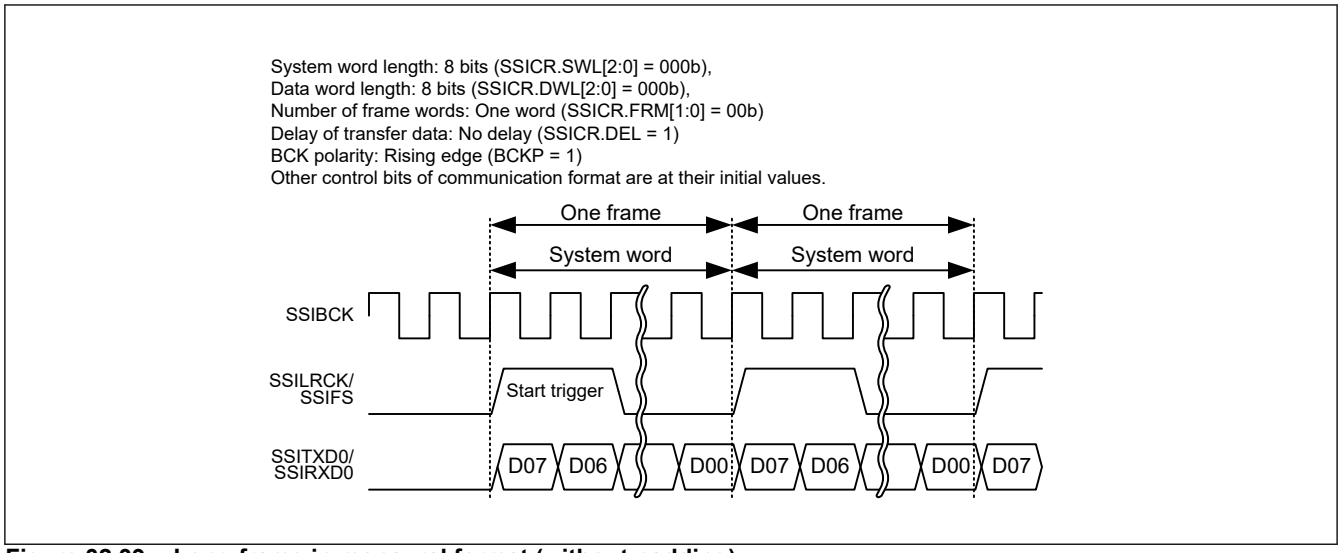



**Figure 32.39 Long frame in monaural format (without padding)**

## 32.5.3 TDM Format

The TDM format is a communication format used for connection with TDM-compatible multi-channel devices. With this format setting (SSIOFR.OMOD[1:0] = 01b), one frame is configured with four to eight system words set with the SSICR.FRM[1:0] bits. With this format, the SSILRCK/SSIFS signal is at a high level for the first one system word and at a low level for the rest. The pulse generated on the SSILRCK/SSIFS signal is defined as the SYNC pulse and its rising edge means a start of one frame. Figure 32.40 and Figure 32.41 respectively show the TDM formats without and with padding.

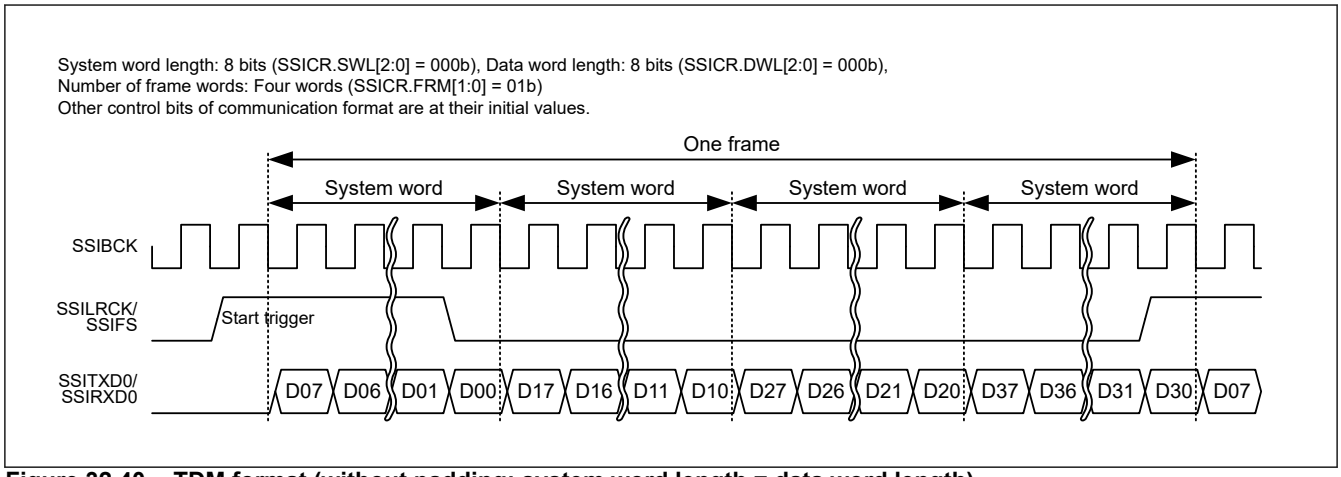



**Figure 32.40 TDM format (without padding: system word length = data word length)**

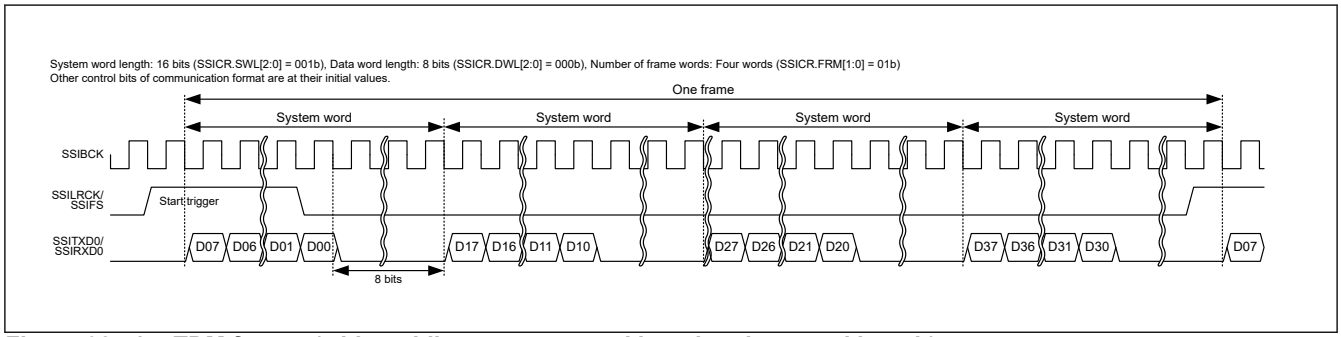



**Figure 32.41 TDM format (with padding: system word length > data word length)**

For the state of external pins when SSIE is in the idle state, see section 32.7.1. Idle State.

Note: SSIE has the SSILRCK/SSIFS pin, which indicates the synchronization of communication. When SSIE is in slave mode (SSICR.MST = 0), the communication format SSIE uses must match that of the other-party device to communicate. SSIE uses the signal input by the SSILRCK/SSIFS pin only as a trigger to start communication.

## 32.6 Communication Modes

SSIE supports the following communication modes. Table 32.13 lists the control bits that are not available with each communication mode. See section 32.6.1. Slave-mode Communication to section 32.6.5. Transmission and Reception for details of these communication modes.

**Table 32.12 Communication modes**

| Communication Mode | SSICR.MST Bit | SSICR.REN Bit | SSICR.TEN Bit |
|---|---|---|---|
| Slave-mode transmission | 0 | 0 | 1 |
| Slave-mode reception | 0 | 1 | 0 |
| Slave-mode transmission and reception | 0 | 1 | 1 |
| Master-mode transmission | 1 | 0 | 1 |
| Master-mode reception | 1 | 1 | 0 |
| Master-mode transmission and reception | 1 | 1 | 1 |

**Table 32.13 Control bits that cannot be used in each communication mode**

| Control Bit | Slave-mode Reception | Slave-mode Transmission | Slave-mode Transmission and Reception | Master-mode Reception | Master-mode Transmission | Master-mode Transmission and Reception |
|---|---|---|---|---|---|---|
| SSICR.CKS | Invalid | Invalid | Invalid | Available | Available | Available |
| SSICR.CKDV | Invalid | Invalid | Invalid | Available | Available | Available |
| SSICR.MUEN | Invalid | Available | Available | Invalid | Available | Available |
| SSICR.TEN | Invalid | Available | Available | Invalid | Available | Available |
| SSICR.REN | Available | Invalid | Available | Available | Invalid | Available |
| SSIFCR.AUCKEN | Invalid | Invalid | Invalid | Available | Available | Available |
| SSIFCR.TIE | Invalid | Available | Available | Invalid | Available | Available |
| SSIFCR.RIE | Available | Invalid | Available | Available | Invalid | Available |
| SSIFCR.TFRST | Invalid | Available | Available | Invalid | Available | Available |
| SSIFCR.RFRST | Available | Invalid | Available | Available | Invalid | Available |
| SSIOFR.BCKASTP | Invalid | Invalid | Invalid | Available | Available | Available |
| SSIOFR.LRCONT | Invalid | Invalid | Invalid | Available | Available | Available |
| SSIOFR.OMOD | Available | Available | Available | Available | Available | Available |
| SSISCR.TDES | Invalid | Available | Available | Invalid | Available | Available |
| SSISCR.RDFS | Available | Invalid | Available | Available | Invalid | Available |

“Invalid” means it has no effect on operation. Writing is possible.

### 32.6.1 Slave-mode Communication

SSIE operates in slave mode with SSICR.MST = 0. The SSIBCK and SSILRCK/SSIFS signals to be used for serial-data communication must be supplied from an external device. If these signals do not match the communication format set for SSIE, operation is not guaranteed.

### 32.6.2 Master-mode Communication

SSIE operates in master mode with SSICR.MST = 1. The SSIBCK and SSILRCK/SSIFS signals to be used for serialdata communication must be internally generated from the audio clock. These signals use the format according to the setting of

SSIE. If the communication format the slave device uses does not match the communication format set for SSIE, the operation is unpredictable.

### 32.6.3 Transmission

SSIE transmits serial data to the other-party device when the SSICR.TEN bit is 1 and the SSICR.REN bit is 0. If the communication format the other-party device uses does not match the communication format set for SSIE, the operation is unpredictable.

### 32.6.4 Reception

SSIE receives serial data from the other-party device when the SSICR.TEN bit is 0 and the SSICR.REN bit is 1. If the communication format the other-party device uses does not match the communication format set for SSIE, the operation is unpredictable.

### 32.6.5 Transmission and Reception

SSIE transmits and receives serial data to and from the other-party device when the SSICR.TEN bit is 1 and the SSICR.REN bit is 1. If the communication format the other-party device uses does not match the communication format set for SSIE, the operation is unpredictable.

## 32.7 Operation

SSIE has the following two main operation states Figure 32.42 shows SSIE state transition.

- Idle state (SSISR.IIRQ = 1)
- Communication state (SSISR.IIRQ = 0).

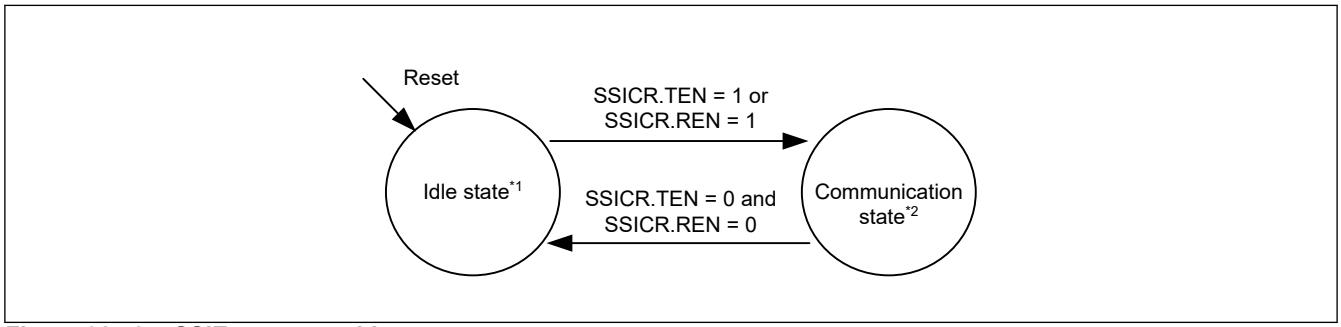



**Figure 32.42 SSIE state transition**

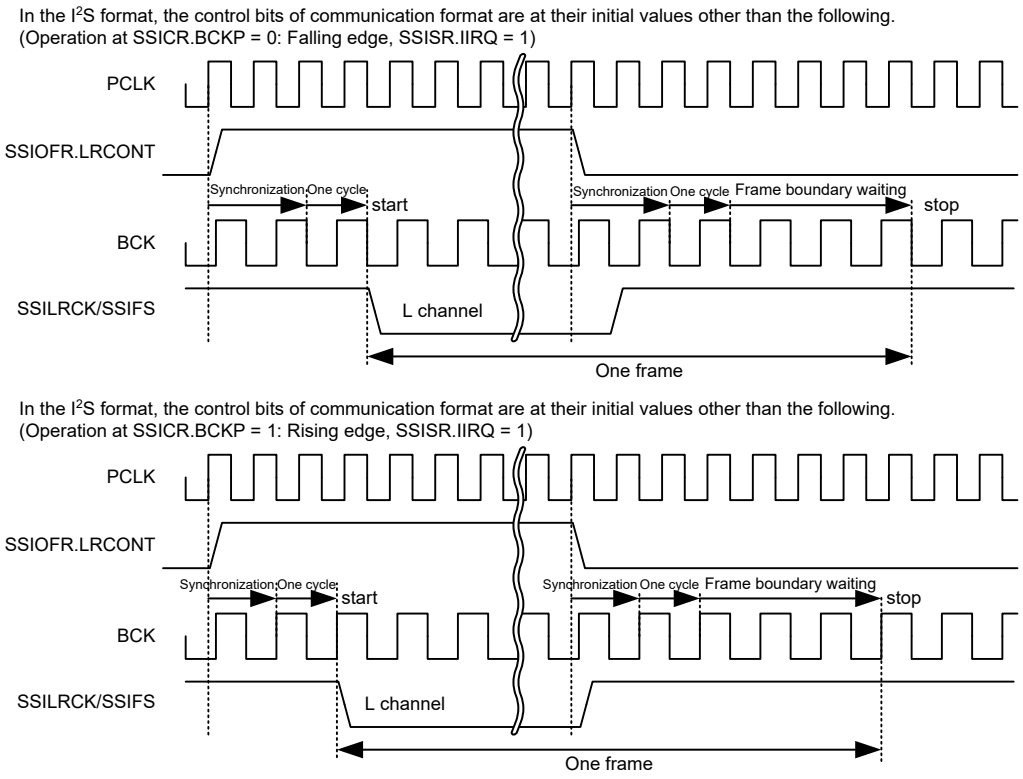
Note: See section 32.8.1. Start Communication for details of the idle state.
See section 32.8.2. Transmission for details of the communication state.

### 32.7.1 Idle State

In this state, communication of SSIE is halted. If, however, the SSICR.MST bit is 1, output of the BCK and LR clock/frame synchronization signals to external pins can be controlled according to the settings of SSIOFR.BCKASTP and SSIOFR.LRCONT bits. This function is common to all formats. For details, see Table 32.14.

**Table 32.14 Output from external pins in the idle state**

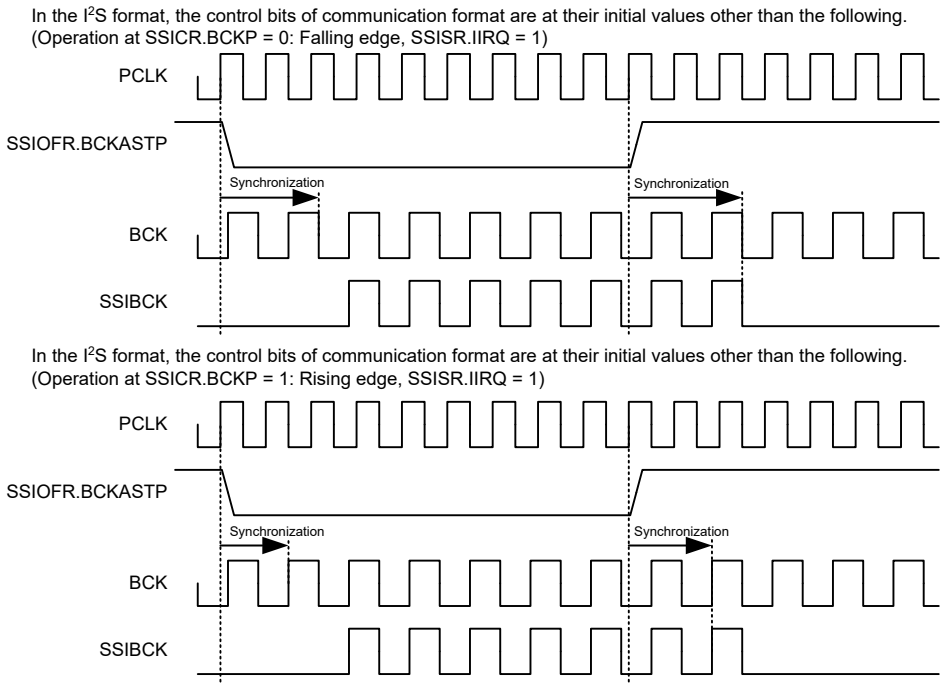
| SSICR.MST | SSIOFR.BCKASTP | SSIOFR.LRCONT | Output from Pins | | |
|---|---|---|---|---|---|
| | | | SSIBCK | SSILRCK/SSIFS | SSITXD0 |
| 0 | — | — | Stop | Stop | Stop |
| 1 | 0 | 0 | Supply | Stop | Stop |
| 1 | 0 | 1 | Supply | Supply | Stop |
| 1 | 1 | 0 | Stop | Stop | Stop |
| 1 | 1 | 1 | Stop | Supply | Stop |




Note: PCLK = PCLKB

**Figure 32.43 Example of disabling LR clock/frame synchronization continuation by SSIOFR.LRCONT**

Note: To stop the output to the SSILRCK/SSIFS pin with SSIOFR.LRCONT when SSIE is in the idle state in mastermode communication (SSICR.MST = 1), note the following: The output stops when the value of the SSIOFR.LRCONT bit is changed from 1 to 0. Make sure that the other-party device is not affected.




Note: PCLK = PCLKB

**Figure 32.44 Example of stopping SSIBCK with SSIOFR.BCKASTP**

Note: To stop the output to the SSIBCK pin with SSIOFR.BCKASTP in master-mode communication (SSICR.MST = 1) and while SSIE is in the idle state, note the following: The output stops when the value of the SSIOFR.BCKASTP bit is changed from 0 to 1. So, make sure that the other-party device is not affected.

## 32.7.2 Communication States

In this state, SSIE is during communication. Figure 32.45 shows transitions of communication states and Table 32.15 lists the conditions for transition. If the transition condition is not satisfied, the state does not transit.

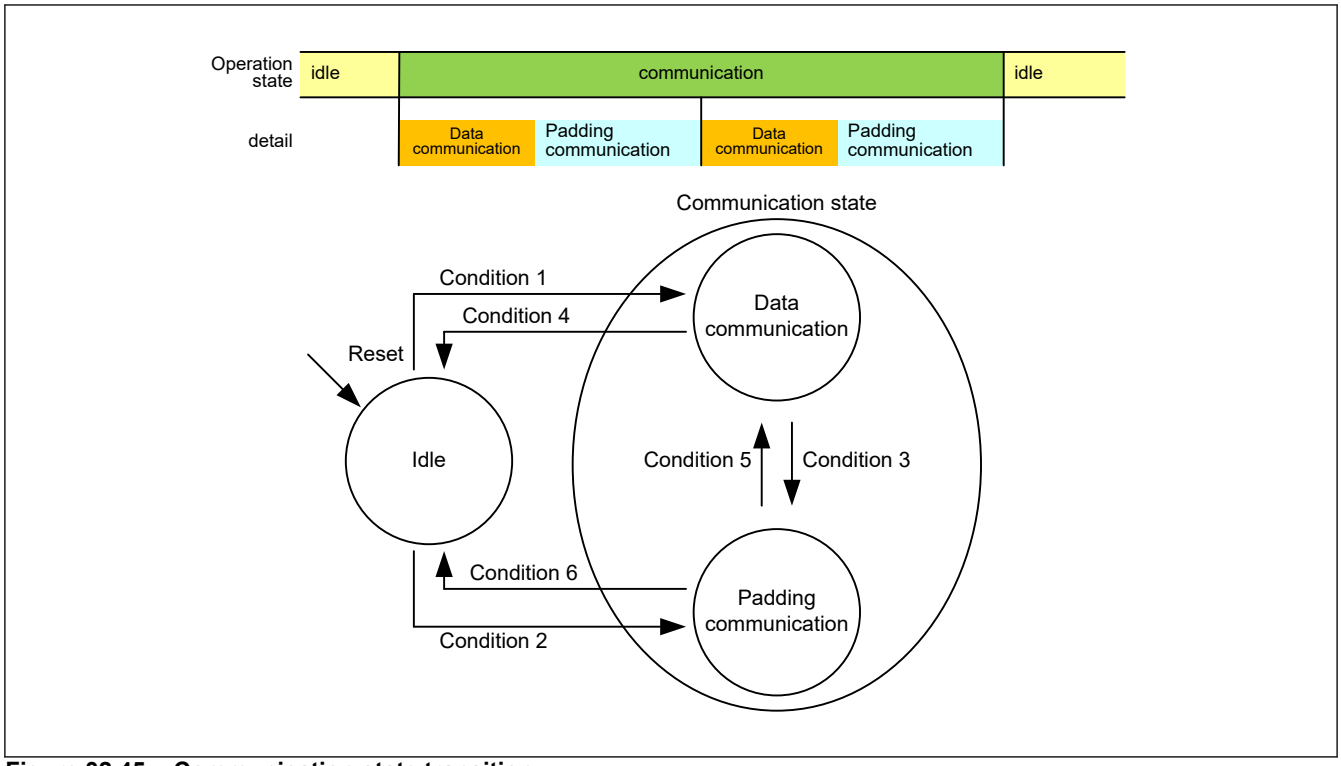



**Figure 32.45 Communication state transition**

**Table 32.15 Condition for communication state transition**

| Condition Number | Condition for Transition |
|---|---|
| 1 | Writing SSICR.TEN = 1 or SSICR.REN = 1 while SSICR.SDTA = 0 or in the setting without padding bits. |
| 2 | Writing SSICR.TEN = 1 or SSICR.REN = 1 while SSICR.SDTA = 1 and in the setting with padding bits. |
| 3 | The following three conditions are all met:<br>● SSICR.TEN = 1 or SSICR.REN = 1<br>● In the setting with padding bits<br>● The last bit of the data words has been transferred. |
| 4 | Both the following two conditions are met:<br>● SSICR.SDTA = 1 or without padding bits<br>● While SSICR.TEN = 0 and SSICR.REN = 0, the last bit of the data words in a frame has been transferred. |
| 5 | Transfer of the last padding bit is completed while SSICR.TEN = 1 or SSICR.REN = 1 |
| 6 | Both the following two conditions are met:<br>● SSICR.SDTA = 0 and with padding bits<br>● While SSICR.TEN = 0 and SSICR.REN = 0, the last padding bit has been transferred. |

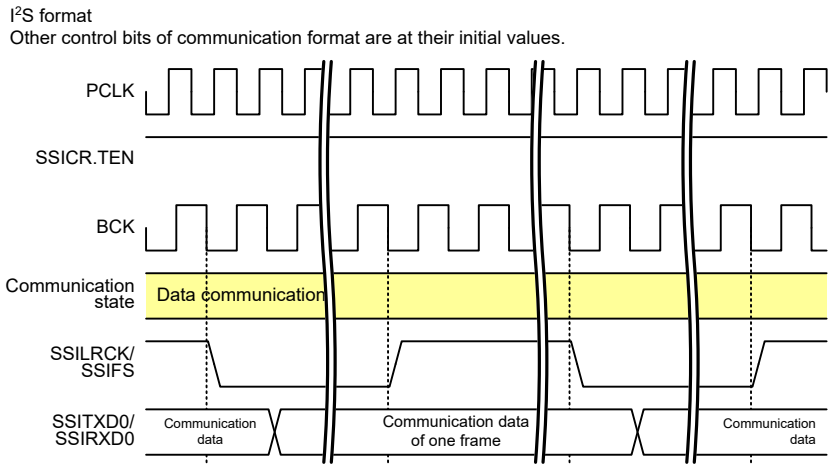
See Table 32.11 for the setting with/without padding bits.

### 32.7.2.1 Data communication state

In this state, SSIE is during communication. Data of the data word length set with the SSICR.DWL[2:0] bits is transmitted, received, or transmitted and received.

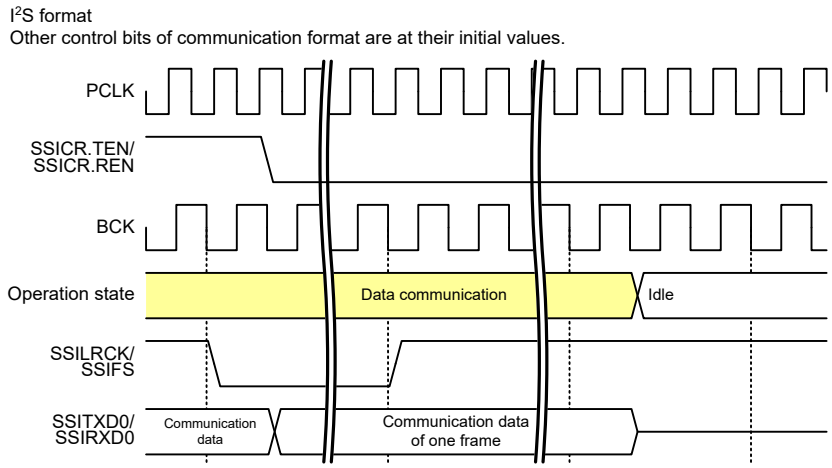
- State Transition in the Setting without Padding Bits

During communication (SSISR.IIRQ = 0), SSIE is during data communication for all the time. By disabling transmission and reception (SSICR.TEN = 0, SSICR.REN = 0), SSIE transits to the idle state. For details, see Figure 32.46 and Figure 32.47.




Note: PCLK = PCLKB

**Figure 32.46 Continuation of the data communication**




Note: PCLK = PCLKB

**Figure 32.47 Halt from the data communication (without padding bits)**

- State Transition in the Setting with Padding Bits

When SSIE ends transfer of the last bit of a data word during communication (SSISR.IIRQ = 0), SSIE transitions from the data communication state to the padding communication state in Figure 32.48. Except in the status with SSICR.SDTA = 1 and transmission and reception disabled (SSICR.TEN = 0 and SSICR.REN = 0), SSIE transitions from the data communication state to the idle state when it stops communication in Figure 32.49.

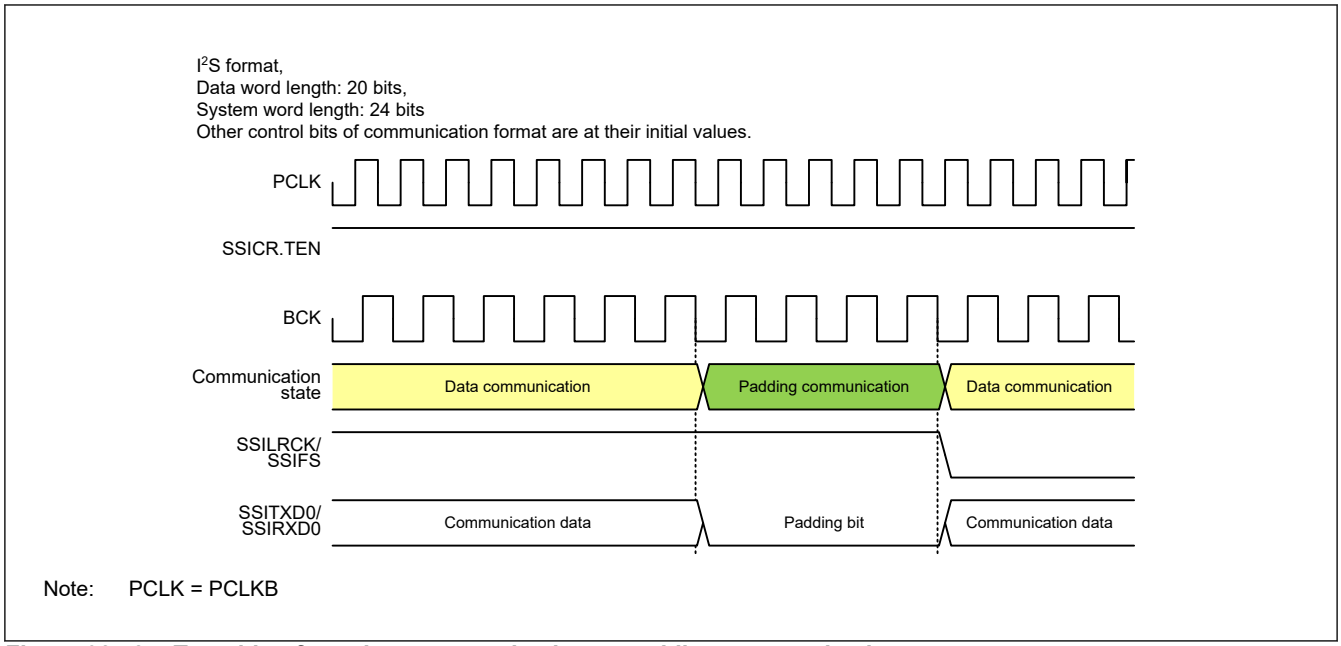



**Figure 32.48 Transition from data communication to padding communication**

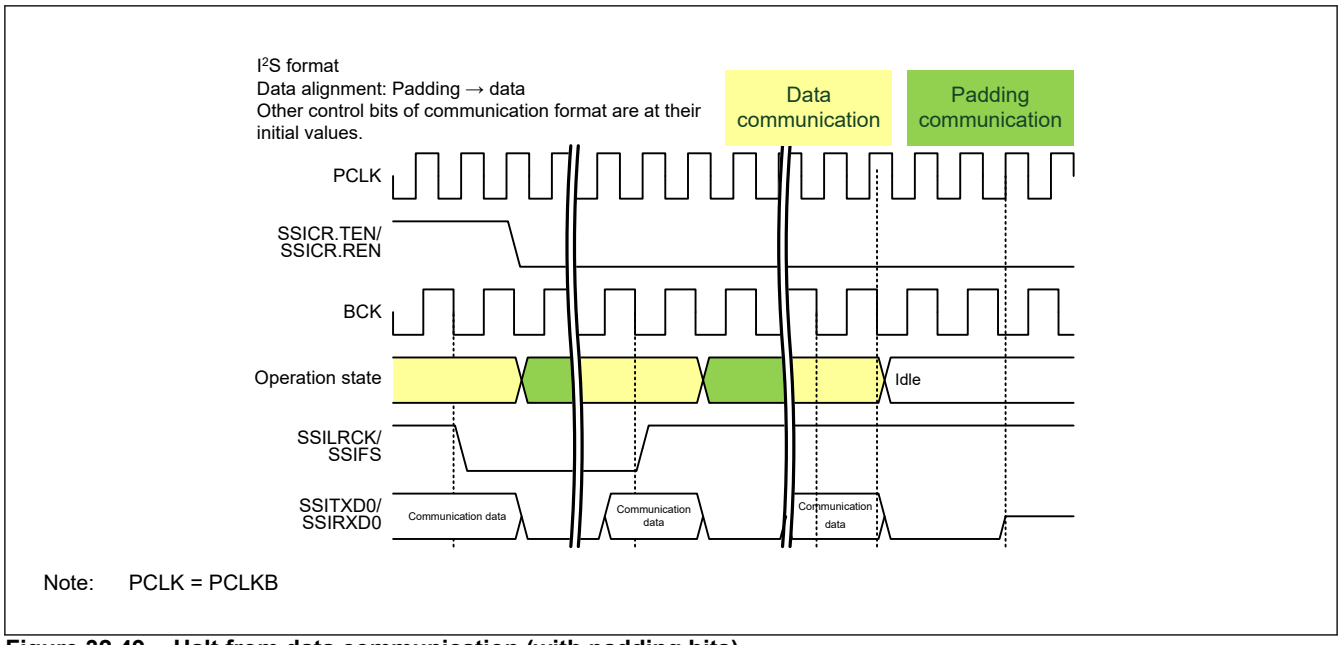



**Figure 32.49 Halt from data communication (with padding bits)**

### 32.7.2.2 Padding communication

In this state, SSIE is during communication. The padding bits set with the SSICR.SWL[2:0] bits and SSICR.DWL[2:0] bits are transmitted, received, or transmitted and received.

- State Transition in the Setting with Padding Bits

When SSIE ends transfer of the last padding bit during communication (SSISR.IIRQ = 0), SSIE transitions to the data communication state in Figure 32.48. If SSIE is in the status with SSICR.SDTA = 0 and transmission and reception disabled (SSICR.TEN = 0 and SSICR.REN = 0), SSIE transitions from the padding communication state to the idle state when it stops communication in Figure 32.50.

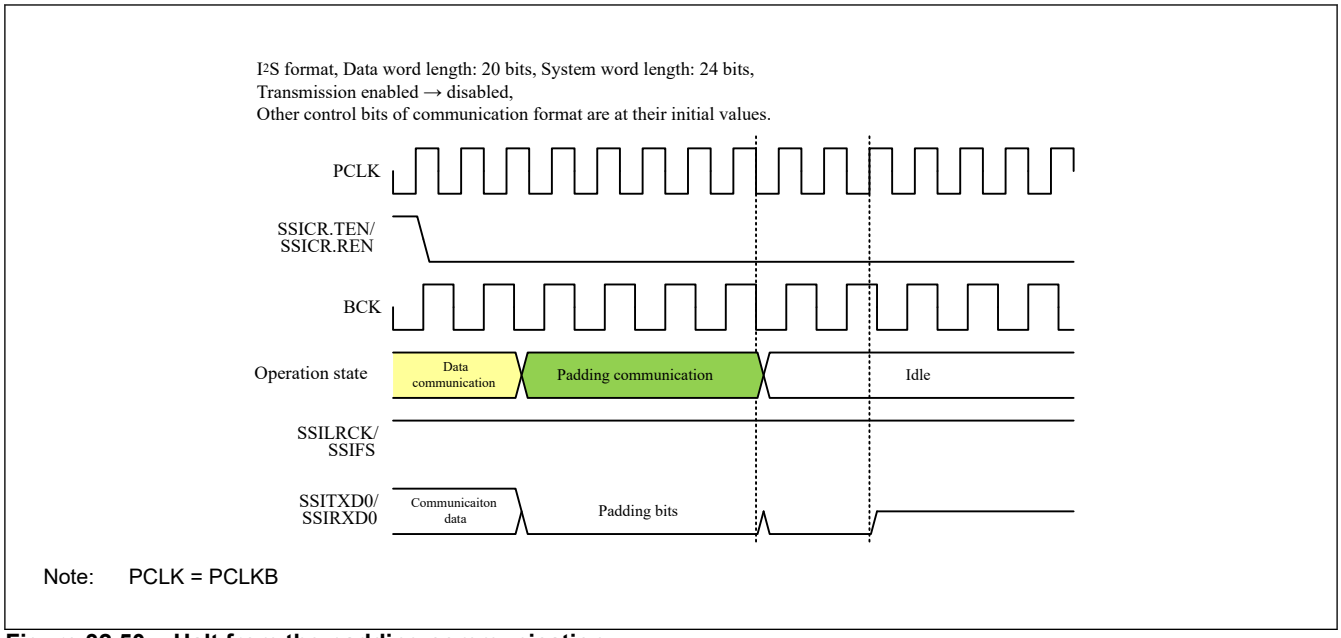



**Figure 32.50 Halt from the padding communication**

## 32.8 Communication Operation

Figure 32.51 shows the communication flow of SSIE.

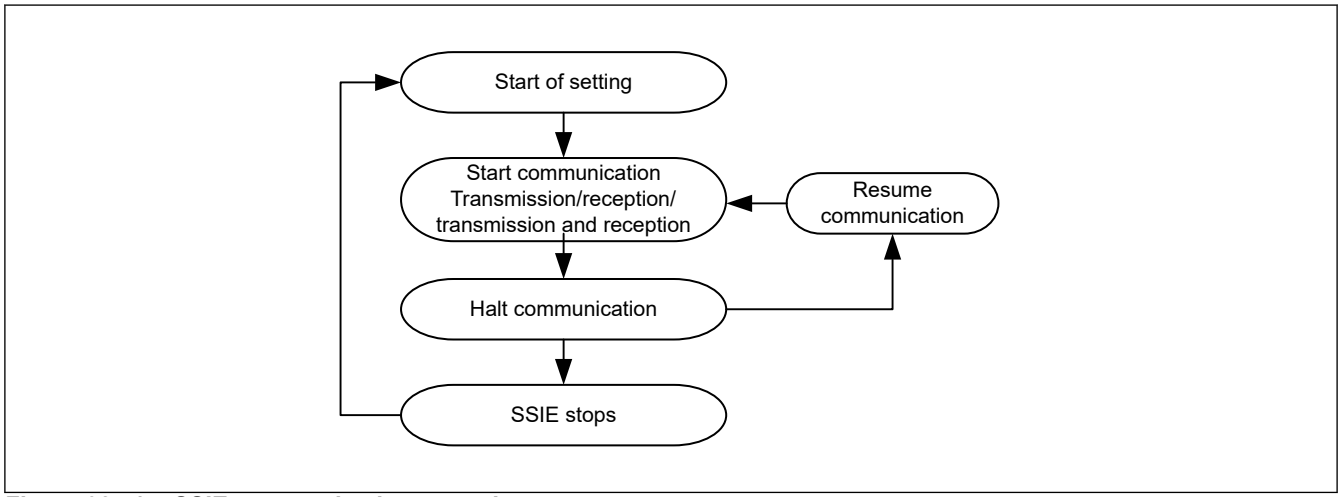
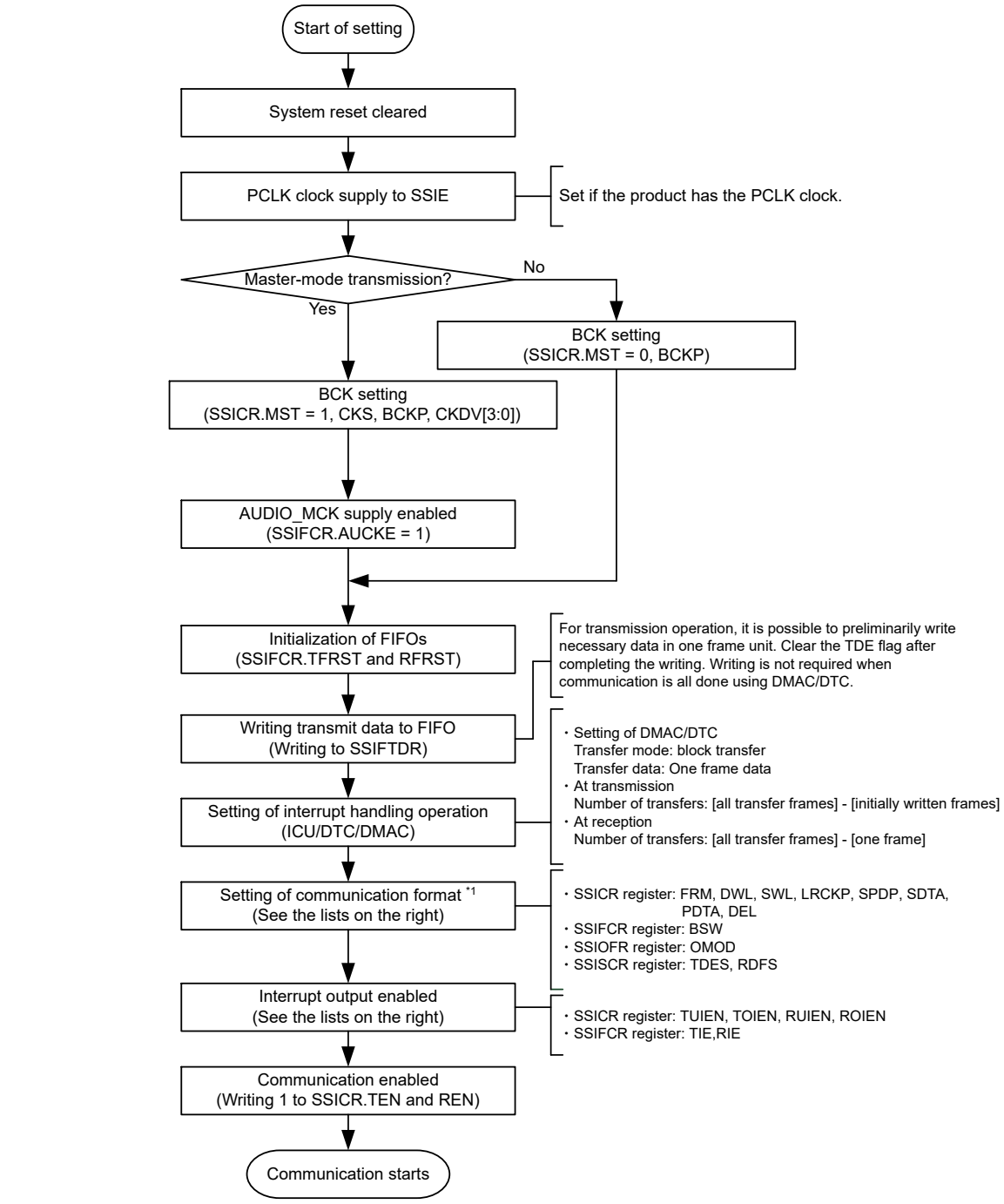



**Figure 32.51 SSIE communication operation**

The procedure of each operation is described from section 32.8.1. Start Communication to section 32.8.7. Resume Communication.

### 32.8.1 Start Communication

This section describes how to start communication of SSIE. Figure 32.52 shows the procedure to start communication. Be sure to follow the procedure. See section 32.8.2. Transmission for transmission operation and section 32.8.3. Reception for reception operation.




Note: PCLK = PCLKB
Note 1. Set SSICR.MUEN, SSIOFR.BCKASTP, and SSIOFR.LRCONT after completing the setting of the communication format.

**Figure 32.52 Procedure to start communication (CPU operation procedure)**

SSIE can perform continuous communication based on interrupts by the DTC/DMAC. For transmission, write 1 to SSIFCR.TIE, SSICR.TUIEN, and SSICR.TOIEN. For reception, write 1 to SSIFCR.RIE, SSICR.RUIEN, and SSICR.ROIEN.

## 32.8.2 Transmission

The transmission procedure in Figure 32.53 must be followed throughout a transmission operation.

After transmission is enabled (SSICR.TEN = 1 and SSICR.REN = 0), SSIE starts transmission when a start trigger is generated by SSILRCK/SSIFS with the serial data for at least a frame contained in the transmit FIFO data register

(SSIFTDR). SSIE outputs a transmit data empty interrupt to the DTC/DMAC according to the TDE setting condition (SSISCR.TDES) and the status of transmit data empty interrupt enable (SSIFCR.TIE) bit specified in the communication start procedure. This interrupt requests writing to the transmit FIFO data register (SSIFTDR). In the communication start procedure, specify writing to the transmit FIFO data register (SSIFTDR) as the DTC/DMAC operation in response to the transmit data empty interrupt. With this setting, SSIE can continuously transmit data not through the CPU. The transmit data empty interrupt is generated when the free space size of transmit FIFO data register reaches the value set in SSISCR.TDES. The number of times of writing must be specified in accordance with the free space size of the transmit FIFO data register indicated by the transmit data empty interrupt. If an error occurs, perform the error-handling procedure as instructed in the communication stop procedure.

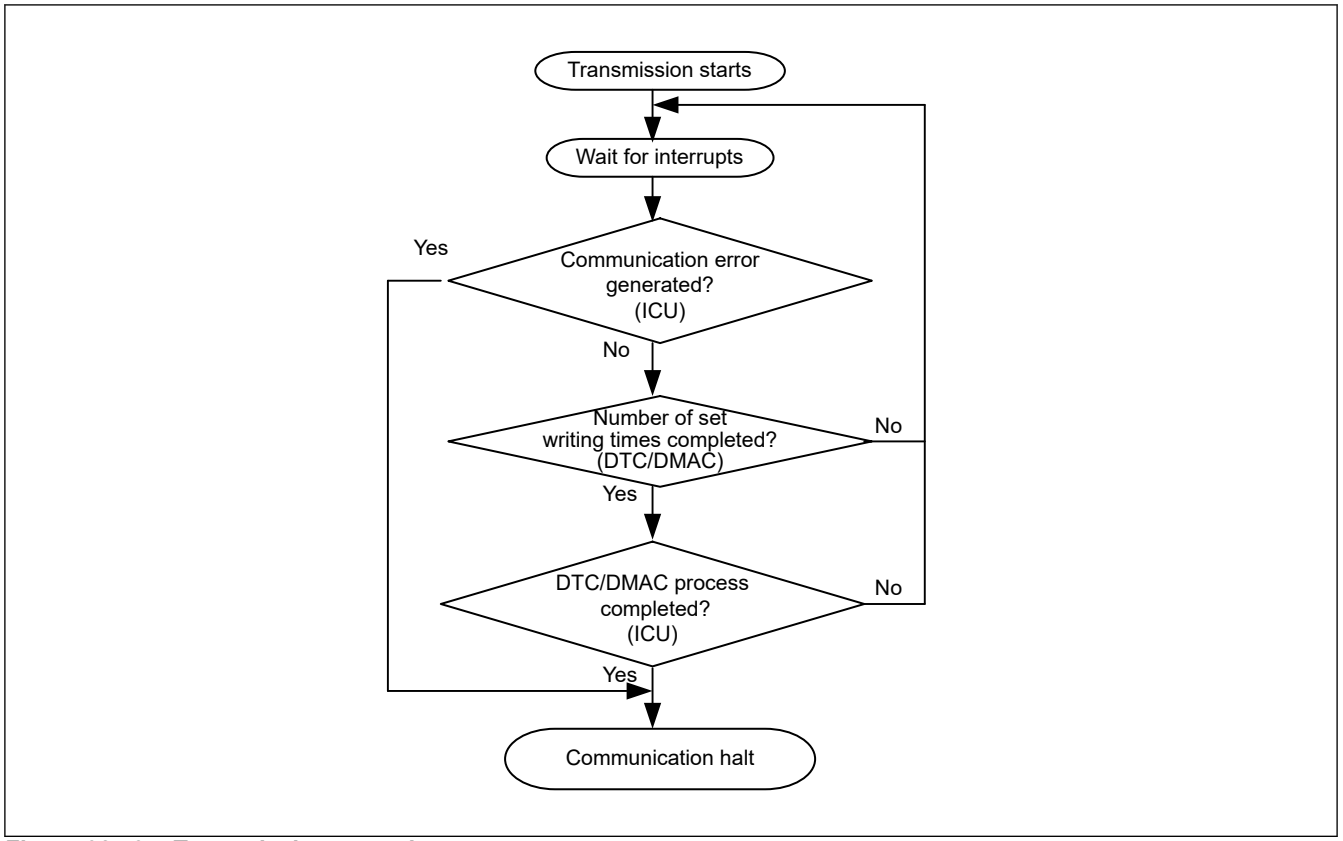



**Figure 32.53 Transmission procedure**

Note: The communication flow defined in SSIE uses the DTC/DMAC. If you do not use the DTC/DMAC, perform polling of the value 1 of SSIFSR.TDE to write data to SSIFTDR. The number of times of writing data to SSIFTDR by detecting the value 1 of SSIFSR.TDE must be in accordance with the free space size of the transmit FIFO data register specified by SSISCR.TDES. After as much transmit data as the free space size is written to SSIFTDR, the SSIFSR.TDE flag must be cleared. Continuous transmission is enabled by repeating data writing. If the SSIFSR.TDE flag is not cleared, the flag is not cleared automatically.

### 32.8.3 Reception

The reception procedure in Figure 32.54 must be followed throughout a reception operation.

After reception is enabled (SSICR.TEN = 0 and SSICR.REN = 1), SSIE starts reception when a start trigger is generated by SSILRCK/SSIFS. SSIE outputs a receive data full interrupt to the DTC/DMAC according to the RDF setting condition (SSISCR.RDFS) and the status of receive data full interrupt enable (SSIFCR.RIE) bit specified in the communication start procedure. This interrupt requests data reading from the receive FIFO data register (SSIFRDR). In the communication start procedure, specify reading from the receive FIFO data register (SSIFRDR) as the DTC/DMAC operation in response to the receive data full interrupt. With this setting, SSIE can continuously read data not through the CPU. The receive data full interrupt is generated when data as much as the capacity of receive FIFO data register has been stored. The number of times of reading must be specified in accordance with the data size of the receive FIFO data register indicated by the receive data full interrupt. If an error occurs, perform the error-handling procedure as instructed in the communication stop procedure.

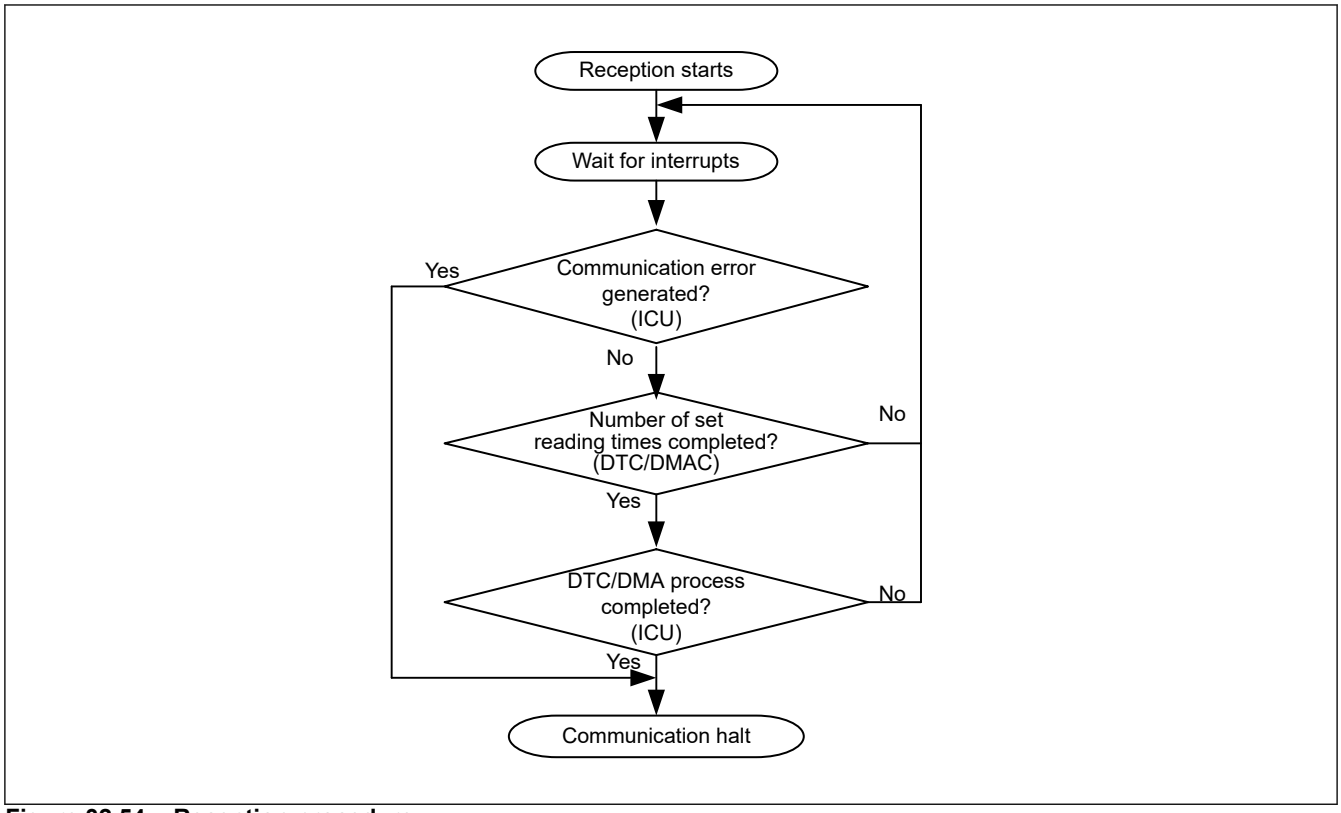



**Figure 32.54 Reception procedure**

Note: The communication flow defined in SSIE uses the DTC/DMAC. If you do not use the DTC/DMAC, perform polling of the value 1 of SSIFSR.RDF to read data from SSIFRDR. The number of times of reading data from SSIFRDR by detecting the value 1 of SSIFSR.RDF must be in accordance with the receive data storage capacity of the receive FIFO data register specified by SSISCR.RDFS. After received data is read from SSIFRDR, the SSIFSR.RDF flag must be cleared. Continuous reception is enabled by repeating data reading. If the SSIFSR.RDF flag is not cleared, the flag is not cleared automatically.

### 32.8.4 Transmission and Reception

After transmission and reception are enabled (SSICR.TEN = 1 and SSICR.REN = 1), SSIE starts transmission and reception when a start trigger is generated by SSILRCK/SSIFS with the serial data for at least a frame contained in the transmit FIFO data register (SSIFTDR). SSIE can continuously transmit and receive data by performing the procedures described in section 32.8.2. Transmission and section 32.8.3. Reception, respectively. For how to stop transmission and reception, see section 32.8.5. Halt Communication.

### 32.8.5 Halt Communication

This section describes how to halt communication of SSIE. Figure 32.55 shows the procedure to halt communication. Be sure to follow the procedure.

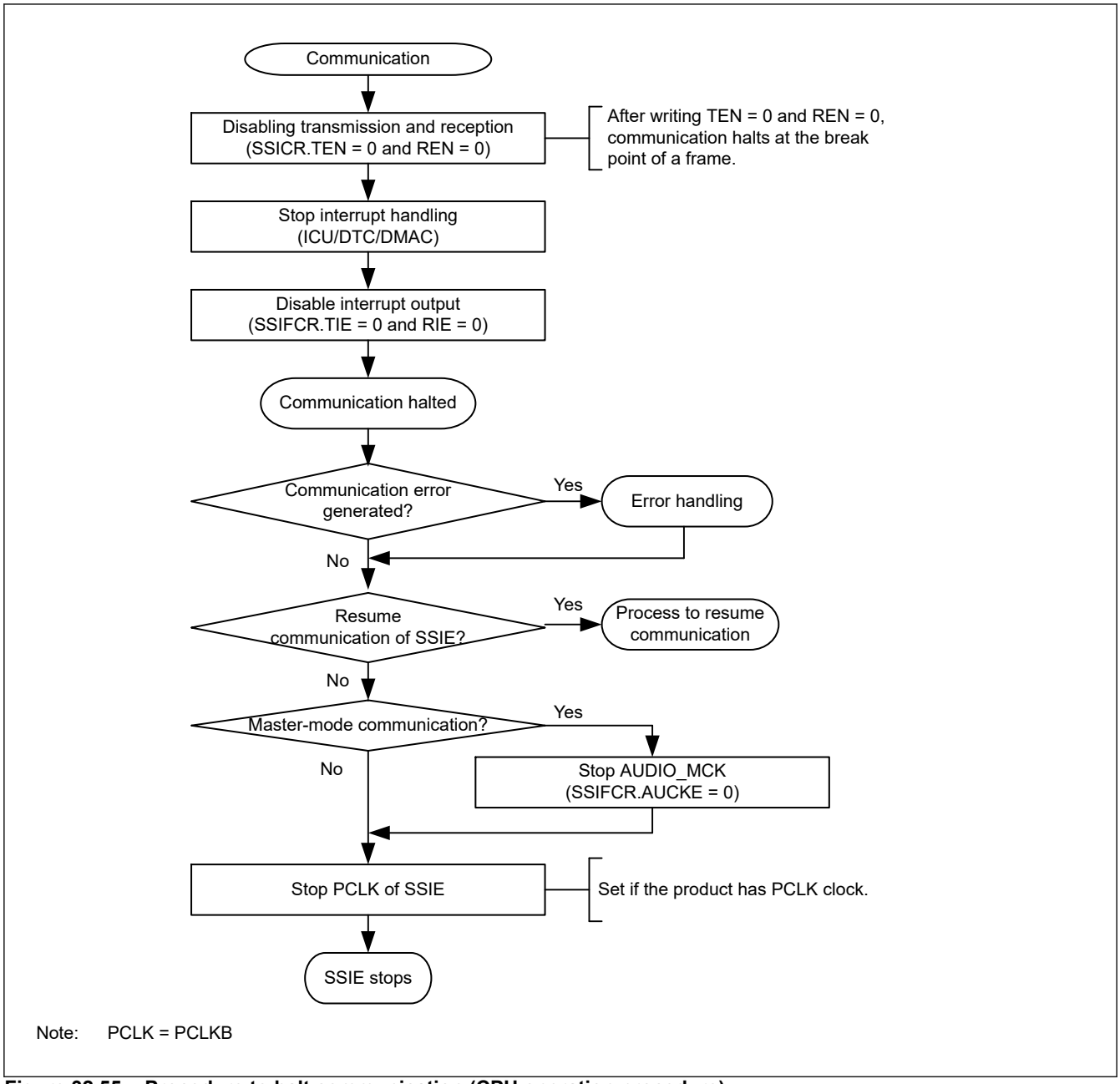



**Figure 32.55 Procedure to halt communication (CPU operation procedure)**

To halt the communication of SSIE, supply of the following clocks are required until the SSISR.IIRQ bit indicates an idle state.

- Input clock from the SSIBCK pin when SSICR.MST = 0
- AUDIO_MCK when SSICR.MST = 1
  To resume communication of SSIE in the previous setting, see section 32.8.7. Resume Communication.

Note: When communication of SSIE is halted according to the procedure to halt communication in Figure 32.55, resume communication according to the procedure to resume communication in Figure 32.57.

## 32.8.6 Error Handling

SSIE has the following four errors.

- Transmit underflow error
- Transmit overflow error
- Receive underflow error

- Receive overflow error.

When an underflow error or overflow error is generated, SSIE need to be restarted. Follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

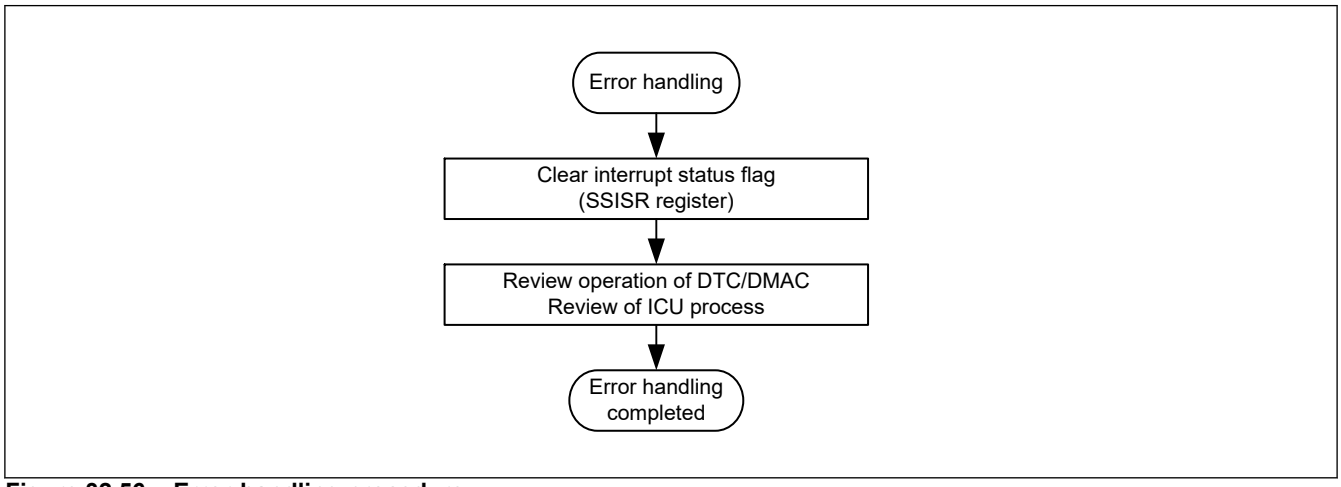



**Figure 32.56 Error-handling procedure**

Four error operations are described as follows. When the interrupt output enable bit of the SSICR register is enabled and error flags are set, an error interrupt is generated. See section 32.4.2. SSISR : Status Register for the setting conditions of error flags.

### (1) Transmit Underflow Error

If a transmit underflow error occurs, review the number of times of writing data to the transmit FIFO data register (SSIFTDR) in response to a transmit data empty interrupt. After a transmit underflow error occurs, SSIE outputs 0s as data. To normally output the serial data written to the transmit FIFO data register (SSIFTDR) to the SSITXD0 pin, follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56. After this error occurs, serial data is consumed as usual. If you resume communication, write the serial data from the beginning.

### (2) Transmit Overflow Error

If a transmit overflow error occurs, review the number of times of writing data to the transmit FIFO data register (SSIFTDR) in response to transmit data empty interrupts. The serial data written to the transmit FIFO data register (SSIFTDR) that caused the transmit overflow error becomes invalid. This error can occur regardless of whether a transmission operation is being done. To recover from the error, follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56. When you resume communication, deal with the invalid serial data appropriately.

### (3) Receive Underflow Error

If a receive underflow error occurs, review the number of times of reading data from the receive FIFO data register (SSIFRDR) in response to receive data full interrupts. The values read from the receive FIFO data register (SSIFRDR) that caused the receive underflow error are undefined. This error can occur regardless of whether a reception operation is being done. To recover from the error, follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

### (4) Receive Overflow Error

If a receive overflow error occurs, review the number of times of reading data from the receive FIFO data register (SSIFRDR) in response to receive data full interrupts. The receive data that caused the receive overflow error cannot be stored in the receive FIFO data register (SSIFRDR). To recover from the error, follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

## 32.8.7 Resume Communication

When you resume the communication using SSIE, follow the communication resume procedure in Figure 32.57. The communication resume procedure is designed on the assumption that you resume the communication stopped by the

communication stop procedure without changing any settings. If you want to change clock and slave/master settings, use and follow the communication start procedure in Figure 32.52. For details about the transmission operation and reception operation after starting communication, see section 32.8.2. Transmission and section 32.8.3. Reception, respectively.

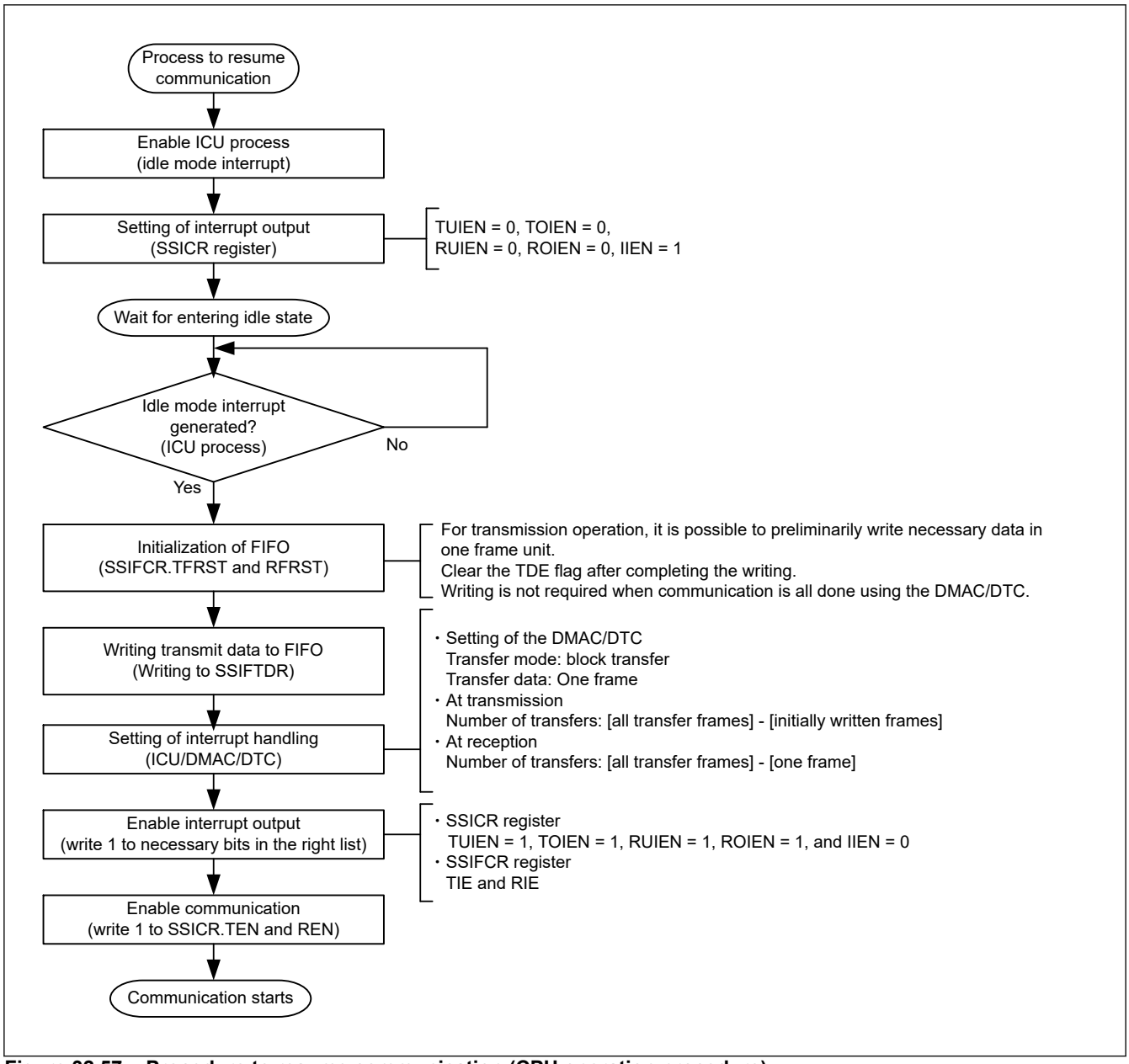



**Figure 32.57 Procedure to resume communication (CPU operation procedure)**

## 32.9 Interrupts

Table 32.16 lists the interrupt sources. Set enable/disable of interrupt output of each source with the TUIEN, TOIEN, RUIEN, ROIEN, and IIEN bits in the SSICR register and the TIE and RIE bits in the SSIFCR register.

**Table 32.16 SSIE interrupt sources**

| Channel | Interrupt source | Description | Interrupt flag | DMAC/DTC activation |
|---|---|---|---|---|
| SSIE0 | SSIE0_SSIF | • Transmit underflow interrupt<br>• Transmit overflow interrupt<br>• Receive underflow interrupt<br>• Receive overflow interrupt<br>• Idle interrupt | SSISR.TUIRQ<br>SSISR.TOIRQ<br>SSISR.RUIRQ<br>SSISR.ROIRQ<br>SSISR.IIRQ | Not possible |
| | SSIE0_SSIRXI | Receive data full interrupt | SSIFSR.RDF | Possible |
| | SSIE0_SSITXI | Transmit data empty interrupt | SSIFSR.TDE | Possible |

## 32.9.1 SSIE0_SSIF Interrupt

This interrupt source combines five interrupts. Enable output of necessary interrupts before using SSIE. The five interrupts are operated by using the flags assigned to individual interrupts and interrupt output enable bits. To clear an interrupt, set the interrupt enable to 0 or clear the interrupt flag to 0.

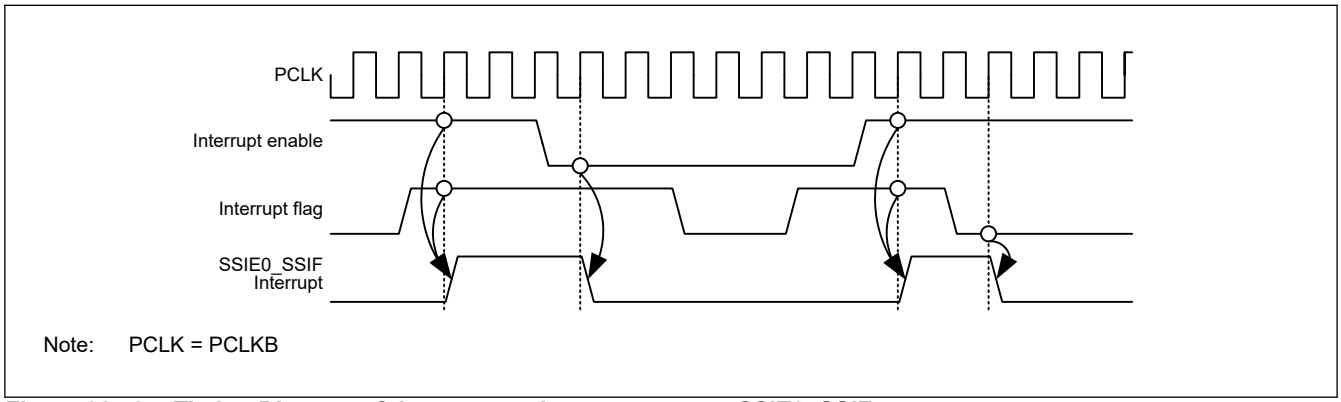



**Figure 32.58 Timing Diagram of the common interrupt source, SSIE0_SSIF**

- Transmit underflow interrupt

As the transmit underflow interrupt, SSISR.TUIRQ is output while SSICR.TUIEN = 1. When you use SSIE for transmission, enable the output of this interrupt (SSICR.TUIRQ = 1). If this interrupt occurs, follow instructions in the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

- Transmit overflow interrupt

As the transmit overflow interrupt, SSISR.TOIRQ is output while SSICR.TOIRQ = 1. When you use SSIE for transmission, enable the output of this interrupt (SSICR.TOIRQ = 1). If this interrupt occurs, follow instructions in the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

- Receive underflow interrupt

As the receive underflow interrupt, SSISR.RUIRQ is output while SSICR.RUIRQ = 1. When you use SSIE for reception, enable the output of this interrupt (SSICR.RUIRQ = 1). If this interrupt occurs, follow instructions in the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

- Receive overflow interrupt

As the receive overflow interrupt, SSISR.ROIRQ is output while SSICR.ROIRQ = 1. When you use SSIE for reception, enable the output of this interrupt (SSICR.ROIRQ = 1). If this interrupt occurs, follow instructions in the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

- Idle mode interrupt

As the idle mode interrupt, SSISR.IIRQ is output while SSICR.IIEN = 1. This interrupt is used to make sure that communication has stopped fully.

### 32.9.2 SSIE0_SSITXI Interrupt

The transmit data empty interrupt is a pulse interrupt that is output when the following condition is met:

- SSIFCR.TIE = 1 and SSIFSR.TDE = 1
  SSIE operation: When the value of SSIFSR.TDE changes from 0 to 1 while the value of SSIFCR.TIE is 1
  CPU instruction: When the value of SSIFCR.TIE changes from 0 to 1 while the value of SSIFSR.TDE is 1

This interrupt is subject to the interrupt suppression function. If an interrupt condition for this interrupt occurs when the DTC/DMAC is busy (when the DTC/DMAC cannot accept interrupts), the interrupt suppression function holds the output of this interrupt. The held interrupt will be output after the DTC/DMAC is enabled to accept interrupts. For details, see Figure 32.59.

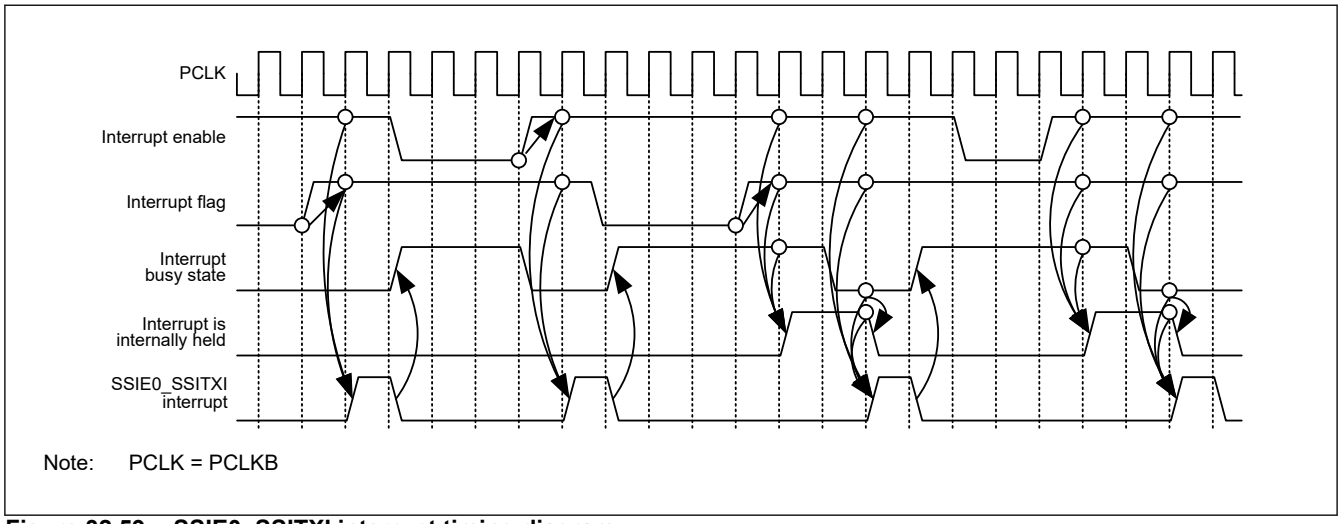



**Figure 32.59 SSIE0_SSITXI interrupt timing diagram**

### 32.9.3 SSIE0_SSIRXI Interrupt

The receive data full interrupt is a pulse interrupt that is output when the following condition is met:

- SSIFCR.RIE = 1 and SSIFSR.RDF = 1.
  SSIE operation: When the value of SSIFSR.RDF changes from 0 to 1 while the value of SSIFCR.RIE is 1
  CPU instruction: When the value of SSIFCR.RIE changes from 0 to 1 while the value of SSIFSR.RDE is 1

This interrupt is subject to the interrupt suppression function. If an interrupt condition for this interrupt occurs when the DTC/DMAC is busy (when the DTC/DMAC cannot accept interrupts), the interrupt suppression function holds the output of this interrupt. The held interrupt will be output after the DTC/DMAC is enabled to accept interrupts. The behavior of this interrupt is the same as the behavior shown in Figure 32.59.

## 32.10 Software Resets

SSIE has three software reset bits to reset its states.

- SSIE software reset (SSIFCR.SSIRST)
- Transmit FIFO data register reset (SSIFCR.TFRST)
- Receive FIFO data register reset (SSIFCR.RFRST).

This section describes the procedures for the three types of software resets.

### 32.10.1 Software Reset Procedure

#### (1) SSIE Software Reset

For the SSIE software reset bit (SSIFCR.SSIRST), follow the procedure shown in Figure 32.60. After a reset, the same setting is applied when it is resumed. To change the settings of clocks and slave/master mode, follow the procedure to start

communication in Figure 32.52. See section 32.8.2. Transmission and section 32.8.3. Reception respectively for transmission and reception after communication is resumed.

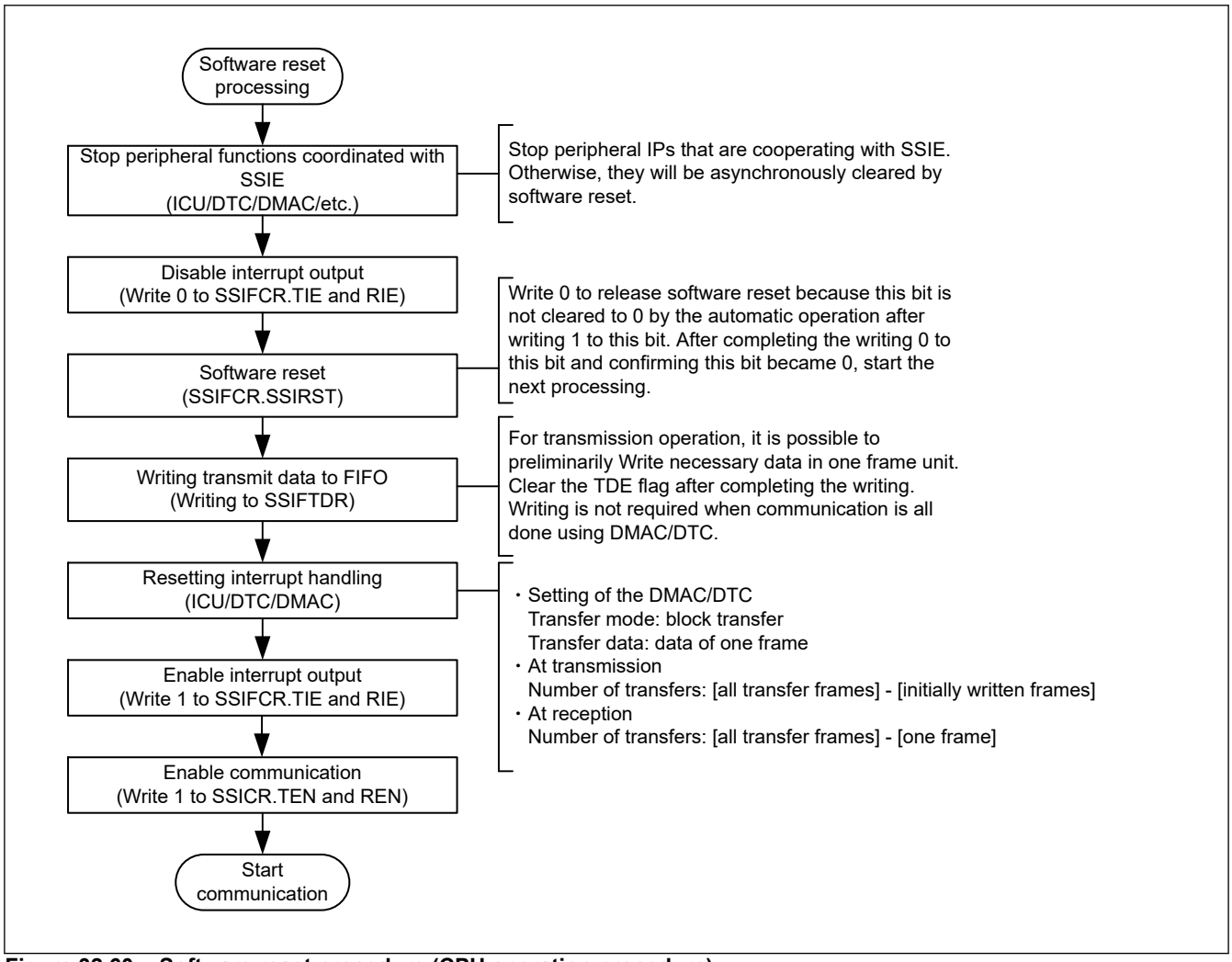



**Figure 32.60 Software reset procedure (CPU operation procedure)**

### (2) Transmit FIFO data register reset

To perform a transmit FIFO data register reset, follow instructions in the procedure to start communication in Figure 32.52 and procedure to resume communication in Figure 32.57.

### (3) Receive FIFO data register reset

To perform a receive FIFO data register reset, follow instructions in the procedure to start communication in Figure 32.52 and procedure to resume communication in Figure 32.57.

## 32.11 Notes

## 32.11.1 Notes for Slave-mode Communication

### 32.11.1.1 SSIBCK control

In slave-mode communication (SSICR.MST = 0), SSIE needs supply of SSIBCK. To stop BCK on the master side, make sure that SSIE is in the idle state (SSISR.IIRQ = 1). If BCK is stopped before SSIE becomes idle, take the procedure to start communication in Figure 32.52 or wait for an idle state by taking the procedure to resume communication in Figure 32.57.

#### 32.11.1.2 SSILRCK/SSIFS pin

SSIE has the SSILRCK/SSIFS pin, which indicates the synchronization of communication. When SSIE is in slave mode (SSICR.MST = 0), the communication format SSIE uses must match that of the other-party device to communicate. SSIE uses the signal input by the SSILRCK/SSIFS pin only as a trigger to start communication.

### 32.11.2 Notes for Master-mode Communication

#### 32.11.2.1 AUCKE control

In master-mode communication (SSICR.MST = 1), SSIE operates with the audio clock (AUDIO_MCK). To stop SSIE completely, make sure that SSIE is in the idle state (SSISR.IIRQ = 1) and then write 0 to SSIFCR.AUCKE.

#### 32.11.2.2 LRCONT control

To stop the output to the SSILRCK/SSIFS pin with SSIOFR.LRCONT when SSIE is in the idle state in master-mode communication (SSICR.MST = 1), note the following: The output stops when the value of the SSIOFR.LRCONT bit is changed from 1 to 0. Make sure that the other-party device is not affected. For details, see Figure 32.43.

#### 32.11.2.3 BCKASTP control

To stop the output to the SSIBCK pin with SSIOFR.BCKASTP in master-mode communication (SSICR.MST = 1) and while SSIE is in the idle state, note the following: The output stops when the value of the SSIOFR.BCKASTP bit is changed from 0 to 1. So, make sure that the other-party device is not affected. For details, see Figure 32.44.

The BCKASTP bit cannot be used when the other-party device (which is a slave) requires the clock output from the SSIBCK pin before and during communication.

### 32.11.3 Notes for Communication Flow

#### 32.11.3.1 When an error interrupt is generated

SSIE has the following four errors.

- Transmit underflow error
- Transmit overflow error
- Receive underflow error
- Receive overflow error

When an underflow error or overflow error is generated, SSIE need to be restarted. Follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

##### (1) Transmit Underflow Error

If a transmit underflow error occurs, review the number of times of writing data to the transmit FIFO data register (SSIFTDR) in response to a transmit data empty interrupt. After a transmit underflow error occurs, SSIE outputs 0s as data. To normally output the serial data written to the transmit FIFO data register (SSIFTDR) to the SSITXD0 pin, follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56. After this error occurs, serial data is consumed as usual. If you resume communication, write the serial data from the beginning.

##### (2) Transmit Overflow Error

If a transmit overflow error occurs, review the number of times of writing data to the Transmit FIFO Data Register (SSIFTDR) in response to transmit data empty interrupts. The serial data written to the Transmit FIFO Data Register (SSIFTDR) that caused the transmit overflow error becomes invalid. This error can occur regardless of whether a transmission operation is being done. To recover from the error, follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56. When you resume communication, deal with the invalid serial data appropriately.

#### (3) Receive Underflow Error

If a receive underflow error occurs, review the number of times of reading data from the receive FIFO data register (SSIFRDR) in response to receive data full interrupts. The values read from the receive FIFO data register (SSIFRDR) that caused the receive underflow error are undefined. This error can occur regardless of whether a reception operation is being done. To recover from the error, follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

#### (4) Receive Overflow Error

If a receive overflow error occurs, review the number of times of reading data from the receive FIFO data register (SSIFRDR) in response to receive data full interrupts. The receive data that caused the receive overflow error cannot be stored in the receive FIFO data register (SSIFRDR). To recover from the error, follow the procedure to halt communication in Figure 32.55 and error-handling procedure in Figure 32.56.

### 32.11.3.2 Transmit data empty interrupt

The communication flow defined in SSIE uses the DTC/DMAC. If you do not use the DTC/DMAC, perform polling of the value 1 of SSIFSR.TDE to write data to SSIFTDR. The number of times of writing data to SSIFTDR by detecting the value 1 of SSIFSR.TDE must be in accordance with the free space size of the transmit FIFO data register specified by SSISCR.TDES. After as much transmit data as the free space size is written to SSIFTDR, the SSIFSR.TDE flag must be cleared. Continuous transmission is enabled by repeating data writing. If the SSIFSR.TDE flag is not cleared, the flag is not cleared automatically.

### 32.11.3.3 Receive data full interrupt

The communication flow defined in SSIE uses the DTC/DMAC. If you do not use the DTC/DMAC, perform polling of the value 1 of SSIFSR.RDF to read data from SSIFRDR. The number of times of reading data from SSIFRDR by detecting the value 1 of SSIFSR.RDF must be in accordance with the receive data storage capacity of the receive FIFO data register specified by SSISCR.RDFS. After received data is read from SSIFRDR, the SSIFSR.RDF flag must be cleared. Continuous reception is enabled by repeating data reading. If the SSIFSR.RDF flag is not cleared, the flag is not cleared automatically.

### 32.11.3.4 Switching transfer modes

1. For state transition from transmission, reception, and transmission and reception, disable transmission and reception (SSICR.TEN = 0, SSICR.REN = 0).
2. Confirm it is in the idle state (SSISR.IIRQ = 1).
3. In the idle state, set the SSICR.TEN bit or the SSICR.REN bit again and resume transfer.

### 32.11.3.5 Resume communication after halting SSIE

When communication of SSIE is halted according to the procedure to halt communication in Figure 32.55, resume communication according to the procedure to resume communication in Figure 32.57.

## 32.11.4 Write Access Restriction

### 32.11.4.1 SSICR register

If the TEN bit or REN bit is rewritten, make sure that the SSISR.IIRQ bit is in the desired status. If the value of the TEN or REN bit is changed by rewriting, subsequent operation is unpredictable. For example, when transmission or reception is enabled, check that SSISR.IIRQ is 0; when transmission or reception is disabled, check that SSISR.IIRQ is 1.

#### (1) TEN Bit and REN Bit

These bits enable/disable transmission and reception. When 1 is written to one of these bits, the corresponding communication operation starts in synchronization with a start trigger by the SSILRCK/SSIFS signal. For details, see section 32.8.2. Transmission, section 32.8.3. Reception, and section 32.8.4. Transmission and Reception. When 0 is written to this bit, the current communication operation stops at the next frame boundary. To use SSIE for both transmission and reception, always write 1 to these bits together. When stopping the communication using SSIE, always disable both transmission and reception (write 0 to the TEN and REN bits).

### 32.11.4.2 SSISR register

#### (1) Clearing TUIRQ and TOIRQ

After communication is enabled (by changing the value of SSICR.TEN bit from 0 to 1), the transmission error flags (TOIRQ and TUIRQ in the SSISR register) are cleared. If, however, the SSISR register is read continuously, the cleared status of the transmission error flags might be unable to be read.

#### (2) Clearing RUIRQ and ROIRQ

After communication is enabled (by changing the value of SSICR.REN bit from 0 to 1), the reception error flags (RUIRQ and ROIRQ in the SSISR register) are cleared. If, however, the SSISR register is read continuously, the cleared status of the reception error flags might be unable to be read.

### 32.11.4.3 Communication state

Writing to the bits with orange-shaded area in Table 32.17 is prohibited. If written, the operation performed immediately after writing is not guaranteed.

**Table 32.17 Bits protected from writing during communication**

| Symbol | Address (BASE+) | | +0 | | | | | | | | +1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SSICR | 0x00 | +0 | — | CKS | TUIEN | TOIEN | RUIEN | ROIEN | IIEN | — | FRM[1:0] | | DWL[2:0] | | | SWL[2:0] | | |
| | | +2 | — | MST | BCKP | LRCKP | SPDP | SDTA | PDTA | DEL | CKDV[3:0] | | | | MUEN | — | TEN | REN |
| SSISR | 0x04 | +0 | — | — | TUIRQ | TOIRQ | RUIRQ | ROIRQ | IIRQ | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| SSIFCR | 0x10 | +0 | AUCKE | — | — | — | — | — | — | — | — | — | — | — | — | — | — | SSIRST |
| | | +2 | — | — | — | — | BSW | — | — | — | — | — | — | — | TIE | RIE | TFRST | RFRST |
| SSIFSR | 0x14 | +0 | — | — | TDC[5:0] | | | | | | — | — | — | — | — | — | — | TDE |
| | | +2 | — | — | RDC[5:0] | | | | | | — | — | — | — | — | — | — | RDF |
| SSIFTDR | 0x18 | +0 | FTDR[31:16] | | | | | | | | | | | | | | | |
| | | +2 | FTDR[15:0] | | | | | | | | | | | | | | | |
| SSIFRDR | 0x1C | +0 | FRDR[31:16] | | | | | | | | | | | | | | | |
| | | +2 | FRDR[15:0] | | | | | | | | | | | | | | | |
| SSIOFR | 0x20 | +0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | — | — | — | BCKASTP | LRCONT | — | — | — | — | — | — | OMOD[1:0] | |
| SSISCR | 0x24 | +0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | +2 | — | — | — | TDES[4:0] | | | | | — | — | — | RDFS[4:0] | | | | |

# 33. SD/MMC Host Interface (SDHI)

## 33.1 Overview

The Secure Digital Host Interface (SDHI) and MultiMediaCard (MMC) Interface provide the functionality required to connect a variety of external memory cards with the MCU. The SDHI supports both 1- and 4-bit buses for connecting different memory cards that support SD, SDHC, and SDXC formats. When developing host devices that are compliant with the SD Specifications, you must comply with the SD Host/Ancillary Product License Agreement (SD HALA).

The MMC interface supports 1-bit, and 4-bit MMC buses that provide eMMC 4.51 (JEDEC Standard JESD 84- B451) device access. This interface also provides backward compatibility and supports for high-speed SDR transfer modes.

Table 33.1 lists the SD/MMC Host Interface specifications and Figure 33.1 shows a block diagram.

**Table 33.1 SD/MMC Host Interface specifications**

| Interface | Parameter | Specifications |
|---|---|---|
| SD | SD bus interface | • Compatible with SD memory card and SDIO card<br>• Transfer bus mode selectable from 4-bit wide bus mode or 1-bit default bus mode<br>• Compatible with SD, SDHC, and SDXC formats |
| SD and MMC shared | SDHI clock frequency | The SDHI clock is generated by dividing PCLKB by $2^n$ (n = 0 to 9) |
| | Error check functions | CRC7 (command/response), CRC16 (transfer data) |
| | Interrupt sources | Card access interrupt (SDHI_MMCn_ACCS), SDIO access interrupt (SDHI_MMCn_SDIO), Card detection interrupt (SDHI_MMCn_CARD) n = (0) |
| | DMA transfer sources | DMAC and DTC triggerable by the (SDHI_MMCn_ODMSDBREQ (n = 0, 1)) interrupt SD buffer is read and write accessible using the DMAC |
| | Other functions | • Card detect function<br>• Write protect support |
| MMC | MMC bus interface | Transfer bus mode selectable from 1-bit, and 4-bit |
| | Transfer modes | Backward compatible mode or high-speed SDR mode selectable |
| | Other functions | e.MMC device access supported |
| TrustZone Filter | Security attribution can be set | |

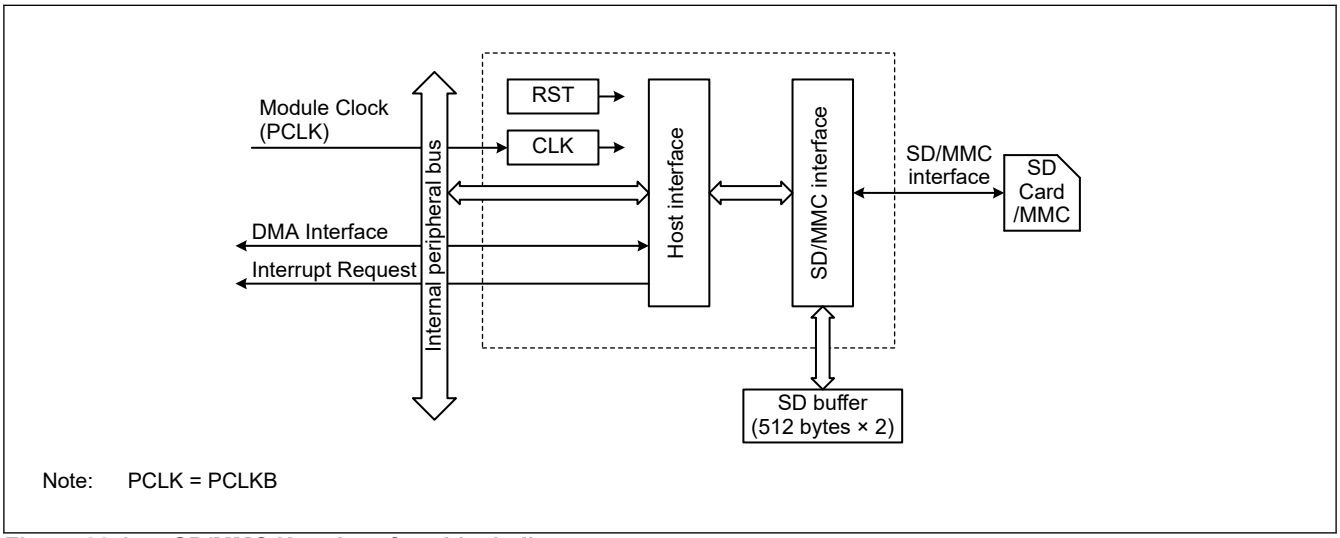



**Figure 33.1 SD/MMC Host Interface block diagram**

**Table 33.2 SDHI I/O pins (n = 0)**

| Channel | Pin name | I/O | Description |
|---|---|---|---|
| Ch n | SDnCLK | Output | SDHI clock |
| | SDnCMD | I/O | Command output, response input |
| | SDnDAT0 | I/O | Data 0 (DAT0) |
| | SDnDAT1 | I/O | Data 1 (DAT1), SDIO interrupt |
| | SDnDAT2 | I/O | Data 2 (DAT2), SDIO Read wait |
| | SDnDAT3 | I/O | Data 3 (DAT3), SD Card detect |
| | SDnCD | Input | SD card detection |
| | SDnWP | Input | SD card write protection |

## 33.2 Register Descriptions

### 33.2.1 SD_CMD : Command Type Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x000

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CMD12AT[1:0] | | TRSTP | CMDRW | CMDTP | RSPTP[2:0] | | | ACMD[1:0] | | CMDIDX[5:0] | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | CMDIDX[5:0] | Command Index Field Value Select<br>These bits configure the command index field value. The examples shown include the bit values for the ACMD[1:0] bits.<br>0x06: CMD6<br>0x12: CMD18<br>0x4D: ACMD13 | R/W |
| 7:6 | ACMD[1:0] | Command Type Select<br>0 0: CMD<br>0 1: ACMD<br>Others: Setting prohibited | R/W |
| 10:8 | RSPTP[2:0] | Response Type Select*1<br>0 0 0: Normal mode<br>Depending on the command, the response type and transfer method are selected in the ACMD[1:0] and CMDIDX[5:0] bits. At this time, the values for bits 15 to 11 in this register are invalid.<br>0 1 1: Extended mode and no response<br>1 0 0: Extended mode and R1, R5, R6, or R7 response<br>1 0 1: Extended mode and R1b response<br>1 1 0: Extended mode and R2 response<br>1 1 1: Extended mode and R3 or R4 response<br>Others: Setting prohibited | R/W |
| 11 | CMDTP | Data Transfer Select*2<br>0: Do not include data transfer (bc, bcr, or ac) in command<br>1: Include data transfer (adtc) in command | R/W |
| 12 | CMDRW | Data Transfer Direction Select*3<br>0: Write (SD/MMC Host Interface → SD card/MMC)<br>1: Read (SD/MMC Host Interface ← SD card/MMC) | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 13 | TRSTP | Block Transfer Select*3<br>0: Single block transfer<br>1: Multiple blocks transfer | R/W |
| 15:14 | CMD12AT[1:0] | CMD12 Automatic Issue Select*4<br>0 0: Automatically issue CMD12 during multiblock transfer<br>0 1: Do not automatically issue CMD12 during multiblock transfer<br>Others: Setting prohibited | R/W |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Some commands cannot be used in normal mode.
Note 2. The CMDTP bit is only valid when the RSPTP[2:0] bits are 011b, 100b, 101b, 110b, or 111b.
Note 3. Bits CMDRW and TRSTP are only valid when the RSPTP[2:0] bits are 011b, 100b, 101b, 110b, or 111b, and the CMDTP bit is 1.
Note 4. The CMD12AT[1:0] bits are only valid when the RSPTP[2:0] bits are 011b, 100b, 101b, 110b, or 111b, and the TRSTP bit is 1.

The command type and response type are set in the SD_CMD register. The command type and transfer mode must be set when the RSPTP[2:0] bits are 011b, 100b, 101b, 110b, or 111b. The sequence starts when a value is written to this register. See Table 33.8 and Table 33.9 for setting examples. Do not write to the SD_CMD register when the SD_INFO2.CBSY flag is 1.

## 33.2.2 SD_ARG : SD Command Argument Register

Base address: SDHI0 = 0x4009_2000
Offset address: 0x008

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | — | These bits specify command format[39:8] (argument). | R/W |

The SD_ARG register is used for setting the argument field value. Set the SD_ARG register before setting the SD_CMD register. The argument field value of the automatically issued CMD12 is 0x0000_0000 regardless of the SD_ARG register value.

## 33.2.3 SD_ARG1 : SD Command Argument Register 1

Base address: SDHI0 = 0x4009_2000
Offset address: 0x00C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | — | These bits specify command format[39:24] (argument). | R/W |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

The SD_ARG1 register is used for setting the argument field value. Set the SD_ARG1 register before setting the SD_CMD register. The argument field value of the automatically issued CMD12 is 0x0000_0000 regardless of the SD_ARG1 register value.

## 33.2.4 SD_STOP : Data Stop Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x010

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | SEC | — | — | — | — | — | — | — | STP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0*1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | STP | Transfer Stop<br>Data transfer stops when this bit is set to 1. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | SEC | Block Count Register Value Select*2<br>0: Disable SD_SECCNT register value<br>1: Enable SD_SECCNT register value | R/W |
| 31:9 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.
Note 2. Do not rewrite this bit when the SD_INFO2.CBSY flag is 1.

The SD_STOP register stops data transfer. During a multiblock transfer sequence, the SD_SECCNT register value (number of blocks to be transferred) can be set to valid or invalid by setting the SD_STOP register.

### STP bit (Transfer Stop)

When the STP bit is set to 1 during multiple block transfer, CMD12 is issued to halt the transfer through the SDHI. However, if a command sequence is halted because of a communications error or timeout, CMD12 is not issued. Although continued buffer access is possible even after STP is set to 1, the buffer access error bit (ILR or ILW) in SD_INFO2 is set accordingly.

When STP is set to 1 during transfer for single block write, the access end flag sets when SD_BUF becomes empty, and CMD12 is not issued. If SD_BUF does contain data, the access end flag sets on completion of reception of the busy state without CMD12 being issued.

When STP is set to 1 during transfer for single block read, the access end flag sets immediately after the STP bit is set, and CMD12 is not issued.

When STP is set to 1 during reception of the busy state after an R1b response, the access end flag sets on completion of reception of the busy state without CMD12 being issued.

When STP is set to 1 after a command sequence is completed, CMD12 is not issued and the access end flag does not set.

Set STP to 1 after the response end flag sets.

Set STP to 0 after the access end flag sets.

**SEC bit (Block Count Register Value Select)**

When SD_CMD is set in the following section to start the command sequence while the SEC bit is set to 1, CMD12 is automatically issued to stop multiblock transfer with the number of blocks set in SD_SECCNT.

CMD18 or CMD25 in normal mode (SD_CMD[10:8] = 000b)

SD_CMD[15:13] = 001b in extended mode (CMD12 is automatically issued, multiple block transfer)

When the command sequence is halted because of a communications error or timeout, CMD12 is not automatically issued.

## 33.2.5 SD_SECCNT : Block Count Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x014

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

When performing a multiblock transfer, SD_SECCNT is a read/write register used to set the number of blocks to be transferred. For example, when the register value is 0x0000_0001, 1 block is transferred. When the register value is 0x0000_FFFF, 65,535 blocks are transferred and when the register value is 0xFFFF_FFFF, 4,294,967,295 blocks are transferred. Do not set this register to 0x0000_0000. Do not rewrite the SD_SECCNT register when the SD_INFO2.CBSY flag is 1.

## 33.2.6 SD_RSPi : SD Card Response Register i (i = 10, 32, 54)

Base address: SDHI0 = 0x4009_2000

Offset address: 0x018 (SD_RSP10)
0x020 (SD_RSP32)
0x028 (SD_RSP54)

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 33.2.7 SD_RSPj : SD Card Response Register j (j = 1, 3, 5)

Base address: SDHI0 = 0x4009_2000

Offset address: 0x01C (SD_RSP1)
0x024 (SD_RSP3)
0x02C (SD_RSP5)

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | — | These bits store the response from the SD card/MMC. | R |
| 31:16 | — | These bits are read as 0. | R |

## 33.2.8 SD_RSP76 : SD Card Response Register 76

Base address: SDHI0 = 0x4009_2000

Offset address: 0x030

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | SD_RSP76[23:16] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | SD_RSP76[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 23:0 | SD_RSP76[23:0] | These bits store the response from the SD card/MMC. | R |
| 31:24 | — | These bits are read as 0. | R |

## 33.2.9 SD_RSP7 : SD Card Response Register 7

Base address: SDHI0 = 0x4009_2000

Offset address: 0x034

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | SD_RSP7[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | SD_RSP7[7:0] | These bits store the response from the SD card/MMC. | R |
| 31:8 | — | These bits are read as 0. | R |

SD_RSP10, SD_RSP32, SD_RSP54, SD_RSP1, SD_RSP3, SD_RSP5, SD_RSP76, and SD_RSP7 are read-only registers that store the response from the SD card/MMC. Depending on the type of response from the SD card/MMC, the SD/MMC Host Interface divides and stores the response among the four registers.

Table 33.3 lists the correspondence between the response type and its storage destination.

**Table 33.3 Correspondence between response type and storage destination**

| Response type | SD_RSP10 register | SD_RSP32 register | SD_RSP54 register | SD_RSP1 register | SD_RSP3 register | SD_RSP5 register | SD_RSP76 register | SD_RSP7 register |
|---|---|---|---|---|---|---|---|---|
| R1 | [39:8] | — | [39:8]*1 | — | — | — | — | — |
| R1b | [39:8] | — | [39:8]*1 | — | — | — | — | — |
| R2 | [39:8] | [71:40] | [103:72] | — | — | — | [127:104] | — |
| R3 | [39:8] | — | — | — | — | — | — | — |
| R4 | [39:8] | — | — | — | — | — | — | — |
| R5 | [39:8] | — | — | — | — | — | — | — |
| R6 | [39:8] | — | — | — | — | — | — | — |
| R7 | [39:8] | — | — | — | — | — | — | — |

Note 1. The responses for CMD18 and CMD25 are stored in registers SD_RSP10 and SD_RSP54. Therefore, even if the SD_RSP10 register is overwritten with the response for the automatically issued CMD12, the response for CMD18 or CMD25 can be confirmed by reading the SD_RSP54 register.

## 33.2.10 SD_INFO1 : SD Card Interrupt Flag Register 1

Base address: SDHI0 = 0x4009_2000

Offset address: 0x038

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | SDD3 MON | SDD3I N | SDD3 RM | SDWP MON | — | SDCD MON | SDCDI N | SDCD RM | ACEN D | — | RSPE ND |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | x | 0 | 0 | x | 0 | x | 0 | 0 | 0*1 | 0 | 0*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RSPEND | Response End Detection Flag<br>0: Response end not detected<br>1: Response end detected | R/(W)*2 |
| 1 | — | This bit is read as 0. The write value should be 0. | R/W |
| 2 | ACEND | Access End Detection Flag<br>0: Access end not detected<br>1: Access end detected | R/(W)*2 |
| 3 | SDCDRM | SDnCD Removal Flag<br>0: SD card/MMC removal not detected by the SDnCD pin<br>1: SD card/MMC removal detected by the SDnCD pin | R/(W)*2 |
| 4 | SDCDIN | SDnCD Insertion Flag<br>0: SD card/MMC insertion not detected by the SDnCD pin<br>1: SD card/MMC insertion detected by the SDnCD pin | R/(W)*2 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5 | SDCDMON | SDnCD Pin Monitor Flag<br>0: SDnCD pin level is high*3<br>1: SDnCD pin level is low*3 | R |
| 6 | — | This bit is read as 0. The write value should be 0. | R/W |
| 7 | SDWPMON | SDnWP Pin Monitor Flag<br>0: SDnWP pin level is high<br>1: SDnWP pin level is low | R |
| 8 | SDD3RM | SDnDAT3 Removal Flag<br>0: SD card/MMC removal not detected by the SDnDAT3 pin<br>1: SD card/MMC removal detected by the SDnDAT3 pin | R/(W)*2 |
| 9 | SDD3IN | SDnDAT3 Insertion Flag<br>0: SD card/MMC insertion not detected by the SDnDAT3 pin<br>1: SD card/MMC insertion detected by the SDnDAT3 pin | R/(W)*2 |
| 10 | SDD3MON | SDnDAT3 Pin Monitor Flag<br>0: SDnDAT3 pin level is low<br>1: SDnDAT3 pin level is high | R |
| 31:11 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.
Note 2. The flag does not change even if set to 1. Writing 0 changes the flag value to 0.
Note 3. The flag changes when the pin level continues for the period set in the SD_OPTION.CTOP[3:0] bits or longer.

The SD_INFO1 register indicates the detection of a response end or access end for a command sequence. The SD_INFO1 register also indicates the detection SD card/MMC insertion/removal and the write protection status.

During a multiblock transfer sequence, if CMD12 or CMD52 (SDIO abort) is issued, the ACEND flag sets to 1, but the RSPEND flag remains set to 0.

If the command sequence is stopped because of a communication error or timeout, the ACEND flag or RSPEND flag sets to 1.

After a reset is canceled, the SDD3MON bit, SDD3IN flag, and SDD3RM flag values are changed in accordance with the status of the SDnDAT3 (n = 0) pin, and their values are changed when data is being transferred in wide bus mode. These 3 bits are used only for SD card. Set flags to be cleared to 0. Set flags that are not being cleared to 1.

### RSPEND flag (Response End Detection Flag)

The RSPEND flag indicates that a response end was detected.

[Setting conditions]

- When reception of the response is completed
- When transmission of a command without response is completed
- When reception of the busy state after R1b response is completed
- When reception of the response to CMD52 that was issued by setting the C52PUB bit to 1 is completed for transfer of multiple block read
- When reception of the response to CMD52 that was issued by setting the C52PUB bit to 1 is completed for transfer of multiple block write
- This bit is set when a command sequence is halted because of a communications error or timeout.

[Clearing conditions]

- When 0 is written to RSPEND
- When a command without data is issued.

Note: When a command is issued in absence of data transfer, the RSPEND flag becomes 1 after the command sequence ends.

### ACEND flag (Access End Detection Flag)

The ACEND flag indicates that an access end was detected.

[Setting conditions]

- When read access to the buffer is completed for transfer of single block read
- When read access to the buffer for the last block of data is completed for transfer of multiple block read
- When read access to the buffer and reception of the response to CMD12 are completed for transfer of multiple block read with automatic issuing of CMD12
- When reception of the busy state after reception of the CRC status is completed for transfer of single block write
- When reception of the busy state after reception of the CRC status of the last block of data is completed for transfer of multiple block write
- When reception of the response busy state for CMD12 is completed for transfer of multiple block write with automatic issuing of CMD12
- When reception of the response to CMD12 that was issued by setting the STP bit to1 is completed for transfer of multiple block read
- When reception of the response busy state for CMD12 that was issued by setting the STP bit to1 is completed for transfer of multiple block write
- When reception of the response to CMD52 that was issued by setting the IOABT bit to 1 is completed for transfer of multiple block read
- When reception of the response to CMD52 that was issued by setting the IOABT bit to1 is completed for transfer of multiple block write
- This bit is set when a command sequence is halted because of a communications error or timeout.

[Clearing conditions]

- When 0 is written to ACEND
- When the access end bit is set to 1.

Note: The ACEND flag becomes 1 after the command sequence ends.

### SDCDRM flag (SDnCD Removal Flag)

The SDCDRM flag indicates that SDnCD was removed.

[Setting condition]

- After a change in SDnCD from 0 to 1, Mcycle elapsed with SDnCD held at 1.

[Clearing conditions]

- When 0 is written to SDCDRM.

Note: Mcycle is set in bits [3:0] in SD_OPTION.

### SDCDIN flag (SDnCD Insertion Flag)

The SDCDIN flag indicates that SDnCD was inserted.

[Setting condition]

- After a change in SDnCD from 1 to 0, Mcycle elapsed with SDnCD held at 0.

[Clearing conditions]

- When 0 is written to SDCDIN.

Note: Mcycle is set in bits [3:0] in SD_OPTION.

### SDD3RM flag (SDnDAT3 Removal Flag)

The SDD3RM flag indicates that SDnDAT3 was removed.

[Setting condition]

- After a change in SDnDAT3 from 1 to 0, two cycles of PCLKB elapsed with SDnDAT3 held at 0.

[Clearing condition]

- When 0 is written to SDD3RM.

**SDD3IN flag (SDnDAT3 Insertion Flag)**

The SDD3IN flag indicates that SDnDAT3 was inserted.

[Setting condition]

- After a change in SDnDAT3 from 0 to 1, two cycles of PCLKB elapsed with SDnDAT3 held at 1.

[Clearing condition]

- When 0 is written to SDD3IN.

## 33.2.11 SD_INFO2 : SD Card Interrupt Flag Register 2

Base address: SDHI0 = 0x4009_2000

Offset address: 0x03C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ILA | CBSY | SD_CLK_CTRLEN | — | — | — | BWE | BRE | SDD0MON | RSPTO | ILR | ILW | DTO | ENDE | CRCE | CMDE |
| Value after reset: | 0*1 | 0*1 | 1*1 | 0 | x | 0 | 0*1 | 0*1 | x | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CMDE | Command Error Detection Flag<br>0: Command error not detected<br>1: Command error detected | R/W*2 |
| 1 | CRCE | CRC Error Detection Flag<br>0: CRC error not detected<br>1: CRC error detected | R/W*2 |
| 2 | ENDE | End Bit Error Detection Flag<br>0: End bit error not detected<br>1: End bit error detected | R/W*2 |
| 3 | DTO | Data Timeout Detection Flag<br>0: Data timeout not detected<br>1: Data timeout detected | R/W*2 |
| 4 | ILW | SD_BUF0 Illegal Write Access Detection Flag<br>0: Illegal write access to the SD_BUF0 register not detected<br>1: Illegal write access to the SD_BUF0 register detected | R/W*2 |
| 5 | ILR | SD_BUF0 Illegal Read Access Detection Flag<br>0: Illegal read access to the SD_BUF0 register not detected<br>1: Illegal read access to the SD_BUF0 register detected | R/W*2 |
| 6 | RSPTO | Response Timeout Detection Flag<br>0: Response timeout not detected<br>1: Response timeout detected | R/W*2 |
| 7 | SDD0MON | SDnDAT0 Pin Status Flag<br>0: SDnDAT0 pin is low<br>1: SDnDAT0 pin is high | R |
| 8 | BRE | SD_BUF0 Read Enable Flag<br>0: Disable read access to the SD_BUF0 register<br>1: Enable read access to the SD_BUF0 register | R/W*2 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9 | BWE | SD_BUF0 Write Enable Flag<br>0: Disable write access to the SD_BUF0 register<br>1: Enable write access to the SD_BUF0 register | R/W*2 |
| 10 | — | This bit is read as 0. The write value should be 0. | R/W |
| 11 | — | The read value is undefined. The write value should be 1. | R/W |
| 12 | — | This bit is read as 0. The write value should be 0. | R/W |
| 13 | SD_CLK_CTRLEN | SD_CLK_CTRL Write Enable Flag<br>0: SD/MMC bus (CMD and DAT lines) is busy, so write access to the SD_CLK_CTRL.CLKEN and CLKSEL[7:0] bits is disabled<br>1: SD/MMC bus (CMD and DAT lines) is not busy, so write access to the SD_CLK_CTRL.CLKEN and CLKSEL[7:0] bits is enabled | R |
| 14 | CBSY | Command Sequence Status Flag<br>0: Command sequence complete<br>1: Command sequence in progress (busy) | R |
| 15 | ILA | Illegal Access Error Detection Flag<br>0: Illegal access error not detected<br>1: Illegal access error detected | R/W*2 |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.
Note 2. The flag does not change even if set to 1. Writing 0 changes the flag value to 0.

The SD_INFO2 register indicates the status of the SD buffer and the status of the SD card/MMC. Set flags to be cleared to 0. Set flags that are not being cleared to 1.

### CMDE flag (Command Error Detection Flag)

The CMDE flag indicates that a command error was detected. The command sequence stops when a command error occurs. When the SDIO_MODE.C52PUB bit is set to 1 and CMD52 is automatically issued, if a communication error or response timeout occurs, the command sequence is not completed. Perform the error processing shown in section 33.3.12. IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Read) or section 33.3.13. IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Write), and complete the command sequence.

[Setting conditions]

- The command index of the transmitted command differs from the command index of the received response.
- The command index of a command issued within a command sequence differs from the command index of the received response.

[Clearing condition]

- When 0 is written to CMDE.

### CRCE flag (CRC Error Detection Flag)

The CRCE flag indicates that a CRC error was detected. The command sequence stops when a CRC error occurs. When the SDIO_MODE.C52PUB bit is set to 1 and CMD52 is automatically issued, if a communication error or response timeout occurs, the command sequence is not completed. Perform the error processing shown in section 33.3.12. IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Read) or section 33.3.13. IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Write), and complete the command sequence.

[Setting conditions]

- When an error occurs in the CRC status.
- When a CRC error occurs in the read data.
- When a CRC error occurs in the response.
- A CRC error in the response to a command issued within a command sequence.

[Clearing condition]

- When 0 is written to CRCE.

### ENDE flag (End Bit Error Detection Flag)

The ENDE flag indicates that an end bit error was detected. The command sequence is stopped when an end bit error occurs. When the SDIO_MODE.C52PUB bit is set to 1 and CMD52 is automatically issued, if a communication error or response timeout occurs, the command sequence is not completed. Perform the error processing shown in section 33.3.12. IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Read) or section 33.3.13. IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Write), and complete the command sequence.

[Setting conditions]

- When an error occurs in the response length (and the end bit is not detected).
- When an error occurs in the read data length (and the end bit is not detected among the valid bits).
- When an error occurs in the CRC status length (and the end bit is not detected).
- An error in the length of a response to a command issued within a command sequence, for example when the end bit is not detected.

[Clearing condition]

- When 0 is written to ENDE.

### DTO flag (Data Timeout Detection Flag)

The DTO flag indicates that a data timeout was detected. The command sequence stops when a data timeout occurs.

[Setting conditions]

- After R1b response, the busy state (SDnDAT0 = 0) continues for longer than Ncycle.
- After CRC status, the busy state (SDnDAT0 = 0) continues for longer than Ncycle.
- After write data, the CRC status is not received though Ncycle has elapsed.
- After read command, read data is not received though a time longer than Ncycle has elapsed.
- After CMD12 is issued within a command sequence, the busy state (SDnDAT0 = 0) for longer than Ncycle continues.
- After the reception of read data, read data for the next block are not received though a time longer than Ncycle has elapsed.
- After release of the read wait state, read data for the next block are not received though a time longer than Ncycle has elapsed.

Note: Ncycle is set in bits [7:4] in SD_OPTION.

[Clearing condition]

- When 0 is written to DTO.

### ILW flag (SD_BUF0 Illegal Write Access Detection Flag)

The ILW flag indicates that an SD_BUF0 illegal write access was detected.

[Setting conditions]

- When data is written to SD_BUF0 while it is not in the data read/write command state.
- When data is written to SD_BUF0 while SD_BUF is full.
- When data is written to SD_BUF0 while an error occurs in the CRC status or CRC status length.
- When data is written to SD_BUF0 while a busy state after the CRC status continues for longer than Ncycle.

Note: Ncycle is set in bits [7:4] in SD_OPTION.

[Clearing condition]

- When 0 is written to ILW.

### ILR flag (SD_BUF0 Illegal Read Access Detection Flag)

The ILR flag indicates that an SD_BUF0 illegal read access was detected.

[Setting conditions]

- When SD_BUF is empty while SD_BUF0 is read.
- When data with a CRC error or END error is read from SD_BUF0.

[Clearing condition]

- When 0 is written to ILR.

#### RSPTO flag (Response Timeout Detection Flag)

The RSPTO flag indicates that a response timeout was detected. The command sequence is stopped when a response timeout occurs. When the SDIOMD.C52PUB bit is set to 1 and CMD52 is automatically issued, if a communication error or response timeout occurs, the command sequence is not completed. Perform the error processing shown in section 33.3.12. IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Read) or section 33.3.13. IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Write), and complete the command sequence.

[Setting condition]

- When a response is not received though a time longer than 640 cycles of SD/MMC clock has elapsed (including a response to a command issued within a command sequence).

[Clearing condition]

- When 0 is written to RSPTO.

#### SDD0MON flag (SDnDAT0 Pin Status Flag)

The SDD0MON flag indicates the status of the SDnDAT0 pin. If the data timeout (DTO) is set but the response timeout (RSPTO) is not set after the Erase command is issued, the end of the Erase sequence (SDD0MON = 1) is confirmed by polling DAT0.

If a communication error or timeout occurs during a write sequence, the DAT0 bit might retain the value 0.

While the SD/MMC clock is stopped, the DAT0 bit retains the value before the clock is stopped.

#### BRE flag (SD_BUF0 Read Enable Flag)

The BRE flag indicates that SD_BUF0 is enabled for reading.

[Setting conditions]

- When data set in SD_SIZE is stored in SD_BUF0 at single block transfer.
- When data set in SD_SIZE is stored in either bank 1 or bank 2 of SD_BUF0 at multiple block transfer.

[Clearing conditions]

- When 0 is written to BRE
- Reading of a block of data from SD_BUF0 by DMA transfer

When data is read from SD_BUF0 by the CPU, clear BRE then read the amount of data specified in SD_SIZE.

Even if a CRC error or an END error occurs while block data is read, data is stored in SD_BUF0 and BRE is set.

#### BWE flag (SD_BUF0 Write Enable Flag)

The BWE flag indicates that SD_BUF0 is enabled for writing.

[Setting conditions]

- When SD_BUF0 is empty at single block transfer.
- When either bank 1 or bank 2 of SD_BUF0 is empty at multiple block transfer.

[Clearing conditions]

- When 0 is written to BWE.
- Writing of a block of data to SD_BUF0 by DMA transfer.

When data is written to SD_BUF0 by the CPU, clear BWE and then write the amount of data specified in SD_SIZE.

### SD_CLK_CTRLEN flag (SD_CLK_CTRL Write Enable Flag)

When a command sequence is started by writing to SD_CMD, the CBSY bit is set to 1 and, at the same time, the SD_CLK_CTRLEN bit is set to 0. The SD_CLK_CTRLEN bit is set to 1 after 8 cycles of SDCLK have elapsed after the CBSY bit clears to 0 on completion of the command sequence.

### ILA flag (Illegal Access Error Detection Flag)

The ILA flag indicates that an illegal access error was detected.

[Setting conditions]

- Writing of data to SD_CMD within a command sequence (CBSY = 1).
- When SD_CMD[11] = 1 (command with data transfer) and SD_CMD[7:0] = 0000 1100b (CMD12) are set in SD_CMD.

[Clearing condition]

- When 0 is written to ILA.

## 33.2.12 SD_INFO1_MASK : SD INFO1 Interrupt Mask Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x040

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | SDD3INM | SDD3RMM | — | — | — | SDCDINM | SDCDRMM | ACENDM | — | RSPENDM |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RSPENDM | Response End Interrupt Request Mask<br>0: Do not mask response end interrupt request<br>1: Mask response end interrupt request | R/W |
| 1 | — | This bit is read as 0. The write value should be 0. | R/W |
| 2 | ACENDM | Access End Interrupt Request Mask<br>0: Do not mask access end interrupt request<br>1: Mask access end interrupt request | R/W |
| 3 | SDCDRMM | SDnCD Removal Interrupt Request Mask<br>0: Do not mask SD card/MMC removal interrupt request by the SDnCD pin<br>1: Mask SD card/MMC removal interrupt request by the SDnCD pin | R/W |
| 4 | SDCDINM | SDnCD Insertion Interrupt Request Mask<br>0: Do not mask SD card/MMC insertion interrupt request by the SDnCD pin<br>1: Mask SD card/MMC insertion interrupt request by the SDnCD pin | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | SDD3RMM | SDnDAT3 Removal Interrupt Request Mask<br>0: Do not mask SD card/MMC removal interrupt request by the SDnDAT3 pin<br>1: Mask SD card/MMC removal interrupt request by the SDnDAT3 pin | R/W |
| 9 | SDD3INM | SDnDAT3 Insertion Interrupt Request Mask<br>0: Do not mask SD card/MMC insertion interrupt request by the SDnDAT3 pin<br>1: Mask SD card/MMC insertion interrupt request by the SDnDAT3 pin | R/W |
| 31:10 | — | These bits are read as 0. The write value should be 0. | R/W |

The SD_INFO1_MASK register enables or disables interrupt requests from the status flags in the SD_INFO1 register. See Table 33.5, for details on the relationship between the status flags and the requested interrupt source.

## 33.2.13 SD_INFO2_MASK : SD INFO2 Interrupt Mask Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x044

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ILAM | — | — | — | — | — | BWEM | BREM | — | RSPTOM | ILRM | ILWM | DTOM | ENDEM | CRCEM | CMDEM |
| Value after reset: | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CMDEM | Command Error Interrupt Request Mask<br>0: Do not mask command error interrupt request<br>1: Mask command error interrupt request | R/W |
| 1 | CRCEM | CRC Error Interrupt Request Mask<br>0: Do not mask CRC error interrupt request<br>1: Mask CRC error interrupt request | R/W |
| 2 | ENDEM | End Bit Error Interrupt Request Mask<br>0: Do not mask end bit detection error interrupt request<br>1: Mask end bit detection error interrupt request | R/W |
| 3 | DTOM | Data Timeout Interrupt Request Mask<br>0: Do not mask data timeout interrupt request<br>1: Mask data timeout interrupt request | R/W |
| 4 | ILWM | SD_BUF0 Register Illegal Write Interrupt Request Mask<br>0: Do not mask illegal write detection interrupt request for the SD_BUF0 register<br>1: Mask illegal write detection interrupt request for the SD_BUF0 register | R/W |
| 5 | ILRM | SD_BUF0 Register Illegal Read Interrupt Request Mask<br>0: Do not mask illegal read detection interrupt request for the SD_BUF0 register<br>1: Mask illegal read detection interrupt request for the SD_BUF0 register | R/W |
| 6 | RSPTOM | Response Timeout Interrupt Request Mask<br>0: Do not mask response timeout interrupt request<br>1: Mask response timeout interrupt request | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |
| 8 | BREM*1 | BRE Interrupt Request Mask<br>0: Do not mask read enable interrupt request for the SD buffer<br>1: Mask read enable interrupt request for the SD buffer | R/W |
| 9 | BWEM*1 | BWE Interrupt Request Mask<br>0: Do not mask write enable interrupt request for the SD_BUF0 register<br>1: Mask write enable interrupt request for the SD_BUF0 register | R/W |
| 10 | — | This bit is read as 0. The write value should be 0. | R/W |
| 11 | — | This bit is read as 1.The write value should be 1. | R/W |
| 14:12 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15 | ILAM | Illegal Access Error Interrupt Request Mask<br>0: Do not mask illegal access error interrupt request<br>1: Mask illegal access error interrupt request | R/W |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. When the SD_INFO2_MASK.BWEM bit is 0 or the SD_INFO2_MASK.BREM bit is 0, set the SD_DMAEN.DMAEN bit to 0. When the SD_DMAEN.DMAEN bit is 1, set the SD_INFO2_MASK.BWEM bit to 1 and the SD_INFO2_MASK.BREM bit to 1.

The SD_INFO2_MASK register enables or disables interrupt requests from the status flags in the SD_INFO2 register. See Table 33.5 for details on the relationship between the status flags and the requested interrupt source.

## 33.2.14 SD_CLK_CTRL : SD Clock Control Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x048

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | CLKCTRLEN | CLKEN | CLKSEL[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0*1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | CLKSEL[7:0] | SDHI Clock Frequency Select*2<br>0xFF PCLKB<br>0x00: PCLKB/2<br>0x01: PCLKB/4<br>0x02: PCLKB/8<br>0x04: PCLKB/16<br>0x08: PCLKB/32<br>0x10: PCLKB/64<br>0x20: PCLKB/128<br>0x40: PCLKB/256<br>0x80: PCLKB/512<br>Others: Setting prohibited | R/W |
| 8 | CLKEN | SD/MMC Clock Output Control*2<br>0: Disable SD/MMC clock output (fix SDnCLK signal low)<br>1: Enable SD/MMC clock output | R/W |
| 9 | CLKCTRLEN | SD/MMC Clock Output Automatic Control Select<br>0: Disable automatic control of SD/MMC clock output<br>1: Enable automatic control of SD/MMC clock output | R/W |
| 31:10 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.
Note 2. Bits CLKSEL[7:0] and CLKEN cannot be write accessed when the SD_INFO2.SD_CLK_CTRLEN flag is 0.

The SD_CLK_CTRL register controls the SD/MMC clock frequency settings and output. Set the CLKEN bit to 1 before writing to the SD_CMD register to start a command sequence. Do not write to the SD_CLK_CTRL register when the SD_INFO2.SD_CLK_CTRLEN flag is 0.

### CLKCTRLEN bit (SD/MMC Clock Output Automatic Control Select)

The CLKCTRLEN bit enables or disables the automatic control function for SD/MMC clock output, which causes the SD/MMC clock to output only within a command sequence.

The timing with which SD/MMC clock output starts and stops is as follows:

- SD/MMC clock output starts after writing to SD_CMD
- SD/MMC clock output stops when 8 cycles of SD/MMC clock have elapsed after the end of the command sequence.

In addition, SD/MMC clock is fixed to 0 while SCLKEN of SD_CLK_CTRL is 0, regardless of the value of this bit.

## 33.2.15 SD_SIZE : Transfer Data Length Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x04C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | LEN[9:0] | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9:0 | LEN[9:0] | Transfer Data Size Setting<br>These bits specify the transfer data size.*1 | R/W |
| 31:10 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Do not rewrite these bits when the SD_INFO2.CBSY flag is 1.

The SD_SIZE register sets the transfer data size.

### LEN[9:0] bits (Transfer Data Size Setting)

When using single block transfer, the transfer data size can be set in the LEN[9:0] bits from 1 byte to 512 bytes. When CMD12 is automatically issued during a multiblock transfer sequence (CMD18 and CMD25), the transfer data size can only be set to 512 bytes. When CMD12 is not automatically issued during a multiblock transfer sequence, the transfer data size can be set to 32, 64, 128, 256, or 512 bytes. However, a 32-, 64-, 128-, or 256-byte multiblock read transfer can only be performed during an SDIO multiblock transfer (CMD53). Do not set these bits to 0 when using a command that includes data transfer.

## 33.2.16 SD_OPTION : SD Card Access Control Option Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x050

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | WIDTH | — | WIDTH8 | — | — | — | — | TOUTMASK | TOP[3:0] | | | | CTOP[3:0] | | | |
| Value after reset: | 0*1 | 1 | 0*1 | 0 | 0 | 0 | 0 | 0*1 | 1*1 | 1*1 | 1*1 | 0*1 | 1*1 | 1*1 | 1*1 | 0*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | CTOP[3:0] | Card Detection Time Counter*2<br>0x0: PCLKB × $2^{10}$<br>0x1: PCLKB × $2^{11}$<br>0x2: PCLKB × $2^{12}$<br>0x3: PCLKB × $2^{13}$<br>0x4: PCLKB × $2^{14}$<br>0x5: PCLKB × $2^{15}$<br>0x6: PCLKB × $2^{16}$<br>0x7: PCLKB × $2^{17}$<br>0x8: PCLKB × $2^{18}$<br>0x9: PCLKB × $2^{19}$<br>0xA: PCLKB × $2^{20}$<br>0xB: PCLKB × $2^{21}$<br>0xC: PCLKB × $2^{22}$<br>0xD: PCLKB × $2^{23}$<br>0xE: PCLKB × $2^{24}$<br>0xF: Setting prohibited | R/W |
| 7:4 | TOP[3:0] | Timeout Counter*2<br>0x0: SDHI clock × $2^{13}$<br>0x1: SDHI clock × $2^{14}$<br>0x2: SDHI clock × $2^{15}$<br>0x3: SDHI clock × $2^{16}$<br>0x4: SDHI clock × $2^{17}$<br>0x5: SDHI clock × $2^{18}$<br>0x6: SDHI clock × $2^{19}$<br>0x7: SDHI clock × $2^{20}$<br>0x8: SDHI clock × $2^{21}$<br>0x9: SDHI clock × $2^{22}$<br>0xA: SDHI clock × $2^{23}$<br>0xB: SDHI clock × $2^{24}$<br>0xC: SDHI clock × $2^{25}$<br>0xD: SDHI clock × $2^{26}$<br>0xE: SDHI clock × $2^{27}$<br>0xF: Setting prohibited | R/W |
| 8 | TOUTMASK | Timeout Mask<br>0: Activate timeout<br>1: Inactivate timeout (do not set RSPTO and DTO bits of SD_INFO2 or CRCBSYTO, CRCTO, RDTO, BSYTO1, BSYTO0, RSPTO1 and RSPTO0 bits of SD_ERR_STS2)<br>When timeout occurs because of an inactivated timeout, execute a software reset to terminate the command sequence. | R/W |
| 12:9 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13 | WIDTH8 | Bus Width*2<br>See bit 15 WIDTH bit. | R/W |
| 14 | — | This bit is read as 1. The write value should be 1. | R/W |
| 15 | WIDTH | Bus Width*2<br>**WIDTH / WIDTH8 / Bus Width**<br>0 / 1 / Setting prohibited<br>0 / 0 / 4-bit width<br>1 / 0 / 1-bit width<br>1 / 1 / 1-bit width | R/W |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.
Note 2. Do not rewrite these bits when the SD_INFO2.CBSY flag is 1.

The SD bus width and timeout counter are set in the SD_OPTION register.

## 33.2.17 SD_ERR_STS1 : SD Error Status Register 1

Base address: SDHI0 = 0x4009_2000

Offset address: 0x058

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | CRCTK[2:0] | | | CRCTKE | RDCRCE | RSPCRCE1 | RSPCRCE0 | — | — | CRCLENE | RDLENE | RSPLENE1 | RSPLENE0 | CMDE1 | CMDE0 |
| Value after reset: | 0 | 0*1 | 1*1 | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 | 0 | 0 | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CMDE0 | Command Error Flag 0<br>0: No error exists in command index field value of a command*2 response<br>1: Error exists in command index field value of a command*2 response | R |
| 1 | CMDE1 | Command Error Flag 1<br>0: No error exists in command index field value of a command*3 response<br>1: Error exists in command index field value of a command*3 response (with SD_CMD.CMDIDX[5:0] setting, an error that occurs with CMD12 issue is indicated in the CMDE0 flag) | R |
| 2 | RSPLENE0 | Response Length Error Flag 0<br>0: No error exists in command*2 response length<br>1: Error exists in command*2 response length | R |
| 3 | RSPLENE1 | Response Length Error Flag 1<br>0: No error exists in command*3 response length<br>1: Error exists in command*3 response length (with SD_CMD.CMDIDX[5:0] setting, an error that occurs with CMD12 issue is indicated in the RSPLENE0 flag) | R |
| 4 | RDLENE | Read Data Length Error Flag<br>0: No read data length error occurred<br>1: Read data length error occurred | R |
| 5 | CRCLENE | CRC Status Token Length Error Flag<br>0: No CRC status token length error occurred<br>1: CRC status token length error occurred | R |
| 7:6 | — | These bits are read as 0. | R |
| 8 | RSPCRCE0 | Response CRC Error Flag 0<br>0: No CRC error detected in command*2 response<br>1: CRC error detected in command*2 response | R |
| 9 | RSPCRCE1 | Response CRC Error Flag 1<br>0: No CRC error detected in command*3 response (with SD_CMD.CMDIDX[5:0] setting, an error that occurs with CMD12 issue is indicated in the RSPCRCE0 flag)<br>1: CRC error detected in command*3 response | R |
| 10 | RDCRCE | Read Data CRC Error Flag<br>0: No CRC error detected in read data<br>1: CRC error detected in read data | R |
| 11 | CRCTKE | CRC Status Token Error Flag<br>0: No error detected in CRC status token<br>1: Error detected in CRC status token | R |
| 14:12 | CRCTK[2:0] | CRC Status Token<br>These bits store the CRC status token value (normal value is 010b). | R |
| 15 | — | This bit is read as 0. | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:16 | — | These bits are read as 0. | R |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.
Note 2. CMD other than CMD12 when automatic issuing is enabled for multiple block transfer by the setting in SD_CMD, CMD12 when the STP bit in SD_STOP is set to 1, or CMD52 when the C52PUB or IOABT bit in SDIO_MODE is set to 1.
Note 3. CMD12 when automatic issuing is enabled for multiple block transfer by the setting in SD_CMD, CMD12 when the STP bit in SD_STOP is set to 1, or CMD52 when the C52PUB or IOABT bit in SDIO_MODE is set to 1.

The SD_ERR_STS1 register indicates the CRC status token, CRC error, end bit error, and command error.

## 33.2.18 SD_ERR_STS2 : SD Error Status Register 2

Base address: SDHI0 = 0x4009_2000

Offset address: 0x05C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | CRCB SYTO | CRCT O | RDTO | BSYT O1 | BSYT O0 | RSPT O1 | RSPT O0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 | 0*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | RSPTO0 | Response Timeout Flag 0<br>0: After command*2 was issued, response was received in less than 640 cycles of the SD/MMC clock<br>1: After command*2 was issued, response was not received in 640 or more cycles of the SD/MMC clock | R |
| 1 | RSPTO1 | Response Timeout Flag 1<br>0: After command*3 was issued, response was received in less than 640 cycles of the SD/MMC clock<br>1: After command*3 was issued, response was not received after 640 or more cycles of the SD/MMC clock (with SD_CMD.CMDIDX[5:0] setting, an error that occurs with CMD12 issue is indicated in the RSPTO0 flag) | R |
| 2 | BSYTO0 | Busy Timeout Flag 0<br>0: After R1b response was received, SD/MMC was released from the busy state during the specified period*4<br>1: After R1b response was received, SD/MMC was in the busy state after the specified period*4 elapsed | R |
| 3 | BSYTO1 | Busy Timeout Flag 1<br>0: After CMD12 was automatically issued, SD/MMC was released from the busy state during the specified period*4<br>1: After CMD12 was automatically issued, SD/MMC was in the busy state after the specified period*4 elapsed (with SD_CMD.CMDIDX[5:0] setting, an error that occurs with CMD12 issue is indicated in the BSYTO0 flag) | R |
| 4 | RDTO | Read Data Timeout Flag<br>When a read command is issued, this flag sets to 1 when read data is not received after the specified period*4 elapses.<br>When read data is received, this flag sets to 1 when the next block of read data is not received after the specified period*4 elapses.<br>When the SD/MMC exits the read wait state, this flag sets to 1 when the next block of read data is not received after the specified period*4 elapses. | R |
| 5 | CRCTO | CRC Status Token Timeout Flag<br>0: After CRC data was written to the SD card/MMC, a CRC status token was received during the specified period*4<br>1: After CRC data was written to the SD card/MMC, a CRC status token was not received after the specified period*4 elapsed | R |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6 | CRCBSYTO | CRC Status Token Busy Timeout Flag<br>0: After a CRC status token was received, the SD/MMC was released from the busy state during the specified period*4<br>1: After a CRC status token was received, the SD/MMC was in the busy state after the specified period*4 elapsed | R |
| 31:7 | — | These bits are read as 0. | R |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.
Note 2. CMD other than CMD12 when automatic issuing is enabled for multiple block transfer by the setting in SD_CMD, CMD12 when the STP bit in SD_STOP is set to 1, or CMD52 when the C52PUB or IOABT bit in SDIO_MODE is set to 1.
Note 3. CMD12 when automatic issuing is enabled for multiple block transfer by the setting in SD_CMD, CMD12 when the STP bit in SD_STOP is set to 1, or CMD52 when the C52PUB or IOABT bit in SDIO_MODE is set to 1.
Note 4. Set the SD_OPTION.TOP[3:0] bits to select the number of *n* cycles.

The SD_ERR_STS2 register indicates the timeout status.

## 33.2.19 SD_BUF0 : SD Buffer Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x060

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

When writing to the SD card, the write data is written to this register. When reading from the SD card, the read data is read from this register. This register is internally connected to two 512-byte buffers.

If both buffers are not empty when executing multiple block read, the SD card/MMC clock is stopped to suspend receiving data. When one of the buffers is empty, the SD card/MMC clock is supplied to resume receiving data.

## 33.2.20 SDIO_MODE : SDIO Mode Control Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x068

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | C52PUB | IOABT | — | — | — | — | — | RWREQ | — | INTEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | INTEN | SDIO Interrupt Acceptance Enable*1<br>0: Disable SDIO interrupt acceptance<br>1: Enable SDIO interrupt acceptance | R/W |
| 1 | — | This bit is read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2 | RWREQ | Read Wait Request<br>0: Allow SD/MMC to exit read wait state<br>1: Request for SD/MMC to enter read wait state | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | IOABT | SDIO Abort<br>If this bit is set to 1 during multiblock transfer triggered by CMD53, CMD52 is immediately issued, and the command sequence is aborted. | R/W |
| 9 | C52PUB | SDIO None Abort<br>If this bit is set to 1 during multiblock transfer triggered by CMD53, CMD52 is issued after the transfer process is complete, and the command sequence is completed. | R/W |
| 31:10 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Do not rewrite this bit when the SD_INFO2.CBSY flag is 1.

The SDIO_MODE register controls reception of the SDIO interrupt, CMD52 issuance during multiblock transfer, and read wait request. Do not set bits C52PUB and IOABT to 1 at the same time.

### RWREQ bit (Read Wait Request)

When RWREQ is set to 1 in the CMD53 (multiple block) read sequence, the block transfer enters the read wait state between blocks.

[Read wait state releasing]

- The read wait state is released, when RWREQ is cleared to 0 in the read wait state.
- When IOABT is set to 1 in the read wait state, RWREQ is automatically cleared to 0 after CMD52 is issued, and then the read wait state is released.
- When C52PUB and RWREQ are set to 1 simultaneously in the CMD53 (multiple block) read sequence, the read wait state is not automatically released. Therefore, after the CMD52 response is received, clear RWREQ. You must set RWREQ and C52PUB simultaneously.

When RWREQ is set to 1 while the last block in the CMD53 (multiple block) read sequence is transferred, the read wait state is not entered and RWREQ is automatically cleared to 0 by setting access end. Set RWREQ to 1 after the response end flag sets.

### IOABT bit (SDIO Abort)

When the IOABT bit is set to 1 in a CMD53 (multiple block) sequence, the CMD53 sequence is halted and CMD52 is issued. However, if a command sequence is halted because of a communication error or timeout, CMD52 is not issued. Although continued buffer access is possible even after IOABT is set to 1, the buffer access error bit (ILR or ILW) in SD_INFO2 is set accordingly. Set SD_ARG before setting IOABT to 1.

When IOABT is set to 1 during transfer for a single block write, the access end flag sets when SD_BUF0 becomes empty, and CMD52 is not issued. If SD_BUF0 contains data, the access end flag sets on completion of reception of the busy state without CMD52 being issued.

When IOABT is set to 1 during transfer for single block read, the access end flag sets immediately after IOABT is set, and CMD52 is not issued.

When IOABT is set to 1 during reception of the busy state after an R1b response, the access end flag sets on completion of reception of the busy state without CMD52 being issued.

When IOABT is set to 1 after a command sequence is completed, CMD52 is not issued and the access end flag does not set.

Set IOABT to 1 after the response end flag sets.

Set IOABT to 0 after the access end flag sets.

### C52PUB bit (SDIO None Abort)

When the C52PUB bit is set to 1 in the CMD53 (multiple block) write sequence, CMD52 is automatically issued between blocks if SD_BUF0 becomes empty. C52PUB is automatically cleared to 0 after reception of the response to CMD52 is completed. Additionally, if C52PUB is set to 1 while the last block is being transferred, CMD52 is not issued. In this case, C52PUB is automatically cleared to 0 after the access end flag sets to 1.

When C52PUB and RWREQ are set to 1 in the CMD53 (multiple block) read sequence, the block transfer enters the read wait state between blocks and CMD52 is automatically issued. C52PUB is automatically cleared to 0 after reception of the response to CMD52 is completed. Additionally, if C52PUB is set to 1 while the last block is being transferred, CMD52 is not issued. In this case, C52PUB is automatically cleared to 0 after the access end flag sets to 1.

If C52PUB is set to 1 in the CMD53 (multiple block) read sequence, you must set RWREQ to 1 in addition to C52PUB.

Set SD_ARG before setting C52PUB to 1.

Set C52PUB to 1 after the response end flag sets.

## 33.2.21 SDIO_INFO1 : SDIO Interrupt Flag Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x06C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | EXWT | EXPUB52 | — | — | — | — | — | — | — | — | — | — | — | — | — | IOIRQ |
| Value after reset: | 0*1 | 0*1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | 0*1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IOIRQ | SDIO Interrupt Status Flag<br>0: No SDIO interrupt detected<br>1: SDIO interrupt detected | R/(W)*2 |
| 2:1 | — | The read values are undefined. The write value should be 1. | R/W |
| 13:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 14 | EXPUB52 | EXPUB52 Status Flag<br>Indicates the status of the EXPUB52. | R/(W)*2 |
| 15 | EXWT | EXWT Status Flag<br>Indicates the status of the EXWT. | R/(W)*2 |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.
Note 2. Only 0 can be written to clear the bit.

The SDIO_INFO1 register indicates the status of the SDIO card access. Set flags to be cleared to 0. Set flags that are not being cleared to 1.

### IOIRQ flag (SDIO Interrupt Status Flag)

The IOIRQ flag indicates that an SDIO interrupt occurred.

[Setting condition]

- When SDIO interrupt from an SDIO card is received while INTEN in SDIO_MODE is set to 1.

[Clearing condition]

- When 0 is written to IOIRQ.*1

Note 1. Before clearing this bit, access the SDIO card to negate the SDIO interrupt signal from the SDIO card. If the interrupt signal is not negated, this bit can be set again.

### EXPUB52 flag (EXPUB52 Status Flag)

The EXPUB52 flag indicates the EXPUB52 status.

[Setting conditions]

- While the last block in the CMD53 (multiple block) sequence is transferred, C52PUB in SDIO_MODE is set to 1.
- While C52PUB is set to 1 in the CMD53 (multiple block) write sequence, the last block is transferred.

[Clearing condition]

- When 0 is written to EXPUB52.

**EXWT flag (EXWT Status Flag)**

The EXWT flag indicates the EXWT status.

[Setting condition]

- While the last block in the CMD53 (multiple block) read sequence is transferred, RWREQ in SDIO_MODE is set to 1.

[Clearing condition]

- When 0 is written to EXWT.

## 33.2.22 SDIO_INFO1_MASK : SDIO INFO1 Interrupt Mask Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x070

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | EXWTM | EXPUB52M | — | — | — | — | — | — | — | — | — | — | — | — | — | IOIRQM |
| Value after reset: | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | IOIRQM | IOIRQ Interrupt Mask Control<br>0: Do not mask IOIRQ interrupts<br>1: Mask IOIRQ interrupts | R/W |
| 2:1 | — | These bits are read as 1.The write value should be 1. | R/W |
| 13:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 14 | EXPUB52M | EXPUB52 Interrupt Request Mask Control<br>0: Do not mask EXPUB52 interrupt requests<br>1: Mask EXPUB52 interrupt requests | R/W |
| 15 | EXWTM | EXWT Interrupt Request Mask Control<br>0: Do not mask EXWT interrupt requests<br>1: Mask EXWT interrupt requests | R/W |
| 31:16 | — | These bits are read as 0. The write value should be 0. | R/W |

The SDIO_INFO1_MASK register enables or disables interrupt requests from the status flags in the SDIO_INFO1 register. See Table 33.5 for details on the relationship between the status flags and the requested interrupt source.

## 33.2.23 SD_DMAEN : DMA Mode Enable Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x1B0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | DMAEN | — |
| Value after reset: | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | This bit is read as 0. The write value should be 0. | R/W |
| 1 | DMAEN | DMA Transfer Enable*1 *2<br>0: Disable use of DMA transfer to access SD_BUF0 register<br>1: Enable use of DMA transfer to access SD_BUF0 register | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | — | This bit is read as 1. The write value should be 1. | R/W |
| 11:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 12 | — | This bit is read as 1. The write value should be 1. | R/W |
| 31:13 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Do not rewrite this bit when the SD_INFO2.CBSY bit is 1.

Note 2. When the SD_INFO2_MASK.BWEM bit is 0 or the SD_INFO2_MASK.BREM bit is 0, set the SD_DMAEN.DMAEN bit to 0. When the SD_DMAEN.DMAEN bit is 1, set the SD_INFO2_MASK.BWEM bit to 1 and the SD_INFO2_MASK.BREM bit to 1.

The SD_DMAEN register enables or disables DMA transfers.

### DMAEN bit (DMA Transfer Enable)

When using DMA transfer to access the SD buffer, set the DMAEN bit to 1 before setting the SD_CMD register.

## 33.2.24 SOFT_RST : Software Reset Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x1C0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | SDRST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SDRST | Software Reset Control<br>0: Reset SD/MMC Host Interface software<br>1: Cancel reset of SD/MMC Host Interface software | R/W |
| 2:1 | — | These bits are read as 1. The write value should be 1. | R/W |
| 31:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Table 33.4 lists the bits and flags initialized by SD/MMC Host Interface software reset.

**Table 33.4 Bits and flags initialized by SD/MMC Host Interface software reset**

| Register | Bit/flag |
|---|---|
| SD_STOP | SEC, STP |
| SD_INFO1 | RSPEND, ACEND |
| SD_INFO2 | CMDE, CRCE, ENDE, DTO, ILW, ILR, RSPTO, BRE, BWE, SD_CLK_CTRLEN, CBSY, ILA |
| SD_CLK_CTRL | CLKEN |
| SD_OPTION | CTOP[3:0], TOP[3:0], TOUTMASK, WIDTH8, WIDTH |
| SD_ERR_STS1 | CMDE0, CMDE1, RSPLENE0, RSPLENE1, RDLENE, CRCLENE, RSPCRCE0, RSPCRCE1, RDCRCE, CRCTKE, CRCTK[2:0] |
| SD_ERR_STS2 | RSPTO0, RSPTO1, BSYTO0, BSYTO1, RDTO, CRCTO, CRCBSYTO |
| SDIO_INFO1 | IOIRQ, EXPUB52, EXWT |
| SDIF_MODE | NOCHKCR |

## 33.2.25 SDIF_MODE : SD Interface Mode Setting Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x1CC

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | NOCHKCR | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0*1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | NOCHKCR | CRC Check Mask<br>CRC check mask bit for MMC test commands. Set when CRC16 or CRC status value check is not executed.<br>0: Enable CRC check<br>1: Disable CRC Check (ignore CRC16 valued when reading and ignore CRC status value when writing) | R/W |
| 31:9 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. The value is initialized by a reset and also on reset triggered by the SOFT_RST.SDRST flag.

### NOCHKCR bit (CRC Check Mask)

The NOCHKCR bit is used for MMC test commands. This bit is set when CRC16 or CRC status value check is not executed.

### 33.2.26 EXT_SWAP : Swap Control Register

Base address: SDHI0 = 0x4009_2000

Offset address: 0x1E0

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | BRSWP | BWSWP | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | BWSWP | SD_BUF0 Swap Write*1<br>0: Normal write operation<br>1: Swap the byte endian order before writing to SD_BUF0 register | R/W |
| 7 | BRSWP | SD_BUF0 Swap Read*1<br>0: Normal read operation<br>1: Swap the byte endian order before reading SD_BUF0 register | R/W |
| 31:8 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Do not rewrite this bit when the SD_INFO2.CBSY flag is 1.

The EXT_SWAP register selects whether or not the byte endian order is swapped when accessing the SD_BUF0 register. See section 33.3.1. SD/MMC Interface for details on the differences in accessing the SD_BUF0 register based on the EXT_SWAP register value.

## 33.3 Operation

### 33.3.1 SD/MMC Interface

When data is read from the SD card/MMC, the process is as follows:

1. The SD/MMC Host Interface receives data from the SD card/MMC through the SDnDAT signal (see Figure 33.2 and Figure 33.3).
2. The received data is stored in SD_BUF of the MMC Host Interface.
3. The data stored in SD_BUF is read from SD_BUF0 (see Figure 33.4).

When data is written to the SD card/MMC, the specified procedure is reversed.

When accessing SD_BUF0, pay attention to the transfer order in SDnDAT and the store order in SD_BUF. If required, you can change the byte endian of the data read from or written to SD_BUF0 using the SDSWAP register. See Figure 33.5.

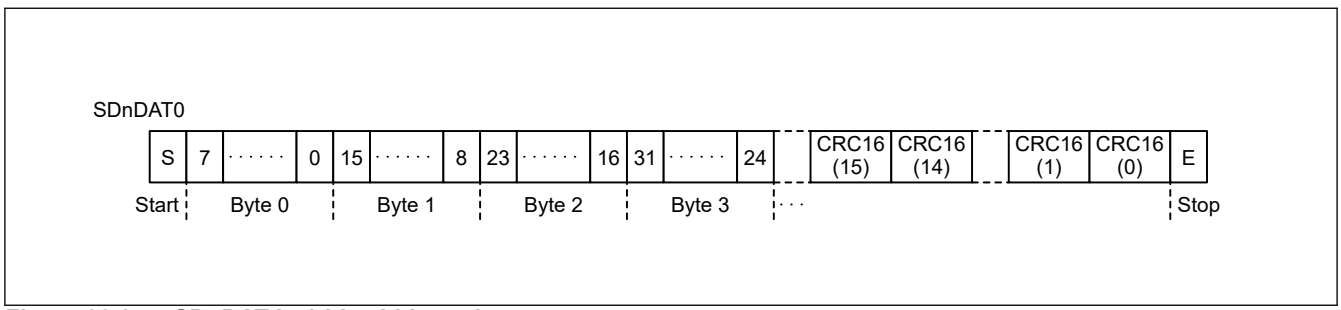



**Figure 33.2 SDnDAT in 1-bit width mode**

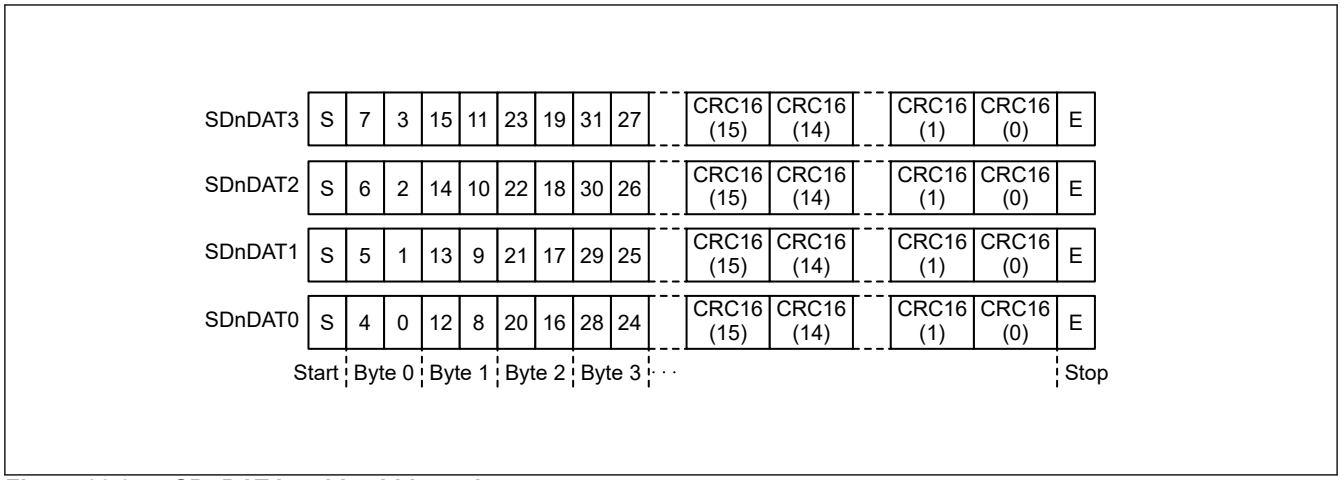



**Figure 33.3 SDnDAT in 4-bit width mode**

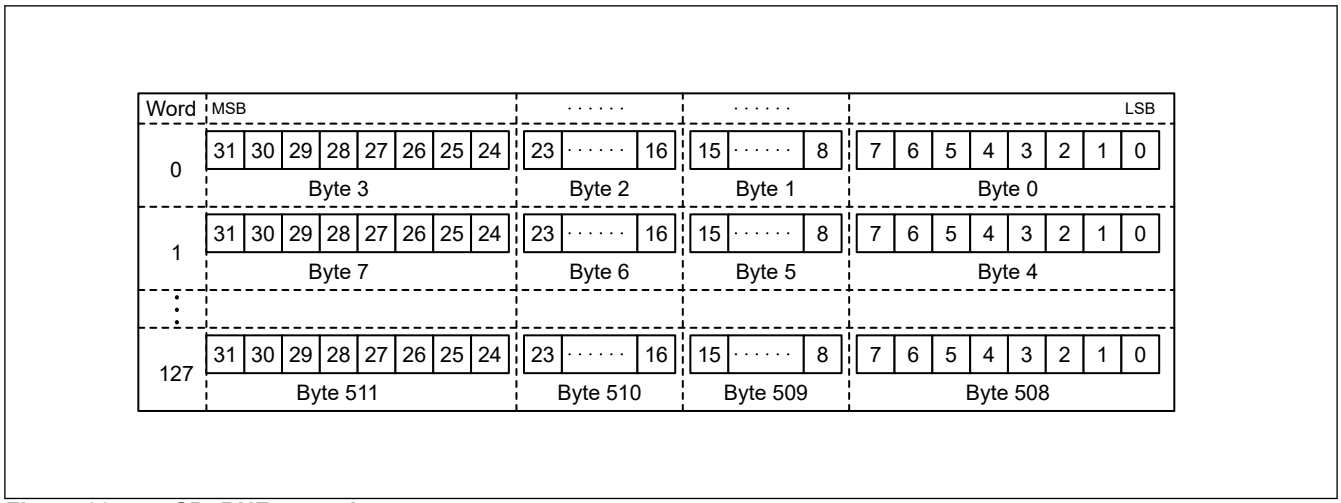



**Figure 33.4 SD_BUF store data**

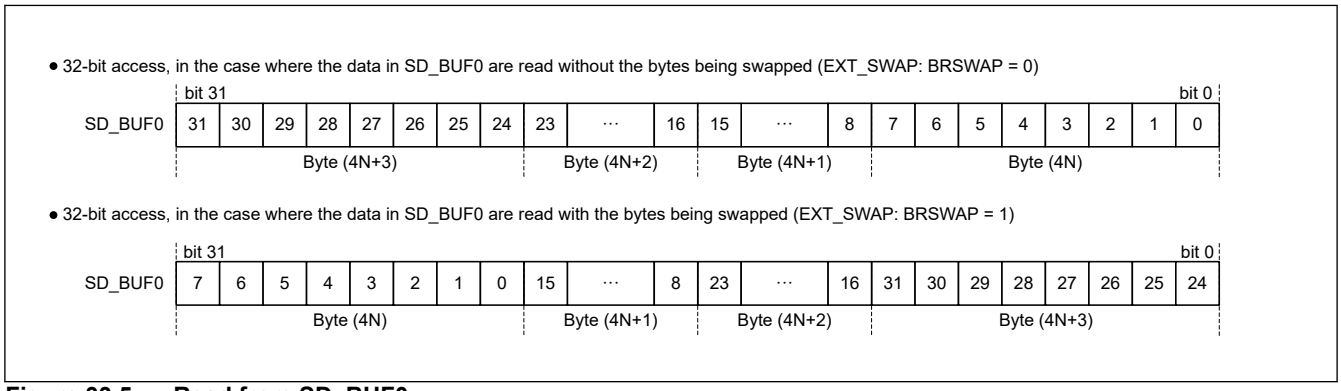



**Figure 33.5 Read from SD_BUF0**

## 33.3.2 Card Detect/Write Protect

### 33.3.2.1 Card detect

The SD/MMC Host Interface has two types of card detect functions.

#### (1) Card detect with SDnCD (n = 0)

Figure 33.6 shows the timing for card detect using SDnCD. SDnCD is connected to the card socket and pulled up on the host device. The resistance of the pull-up resistor is determined by the specification of the SD/MMC host device.

### (2) Card insertion

SDnCD is pulled down when a card is inserted. At this point, if SDnCD is pulled down for the Mcycle period (set in SD_OPTION), SDCDIN in SD_INFO1 is set to 1. It is cleared by writing 0.

### (3) Card removal

SDnCD is pulled up when a card is removed. At this point, if SDnCD is pulled up for the Mcycle period (set in SD_OPTION), SDCDRM in SD_INFO1 is set to 1. It is cleared by writing 0.

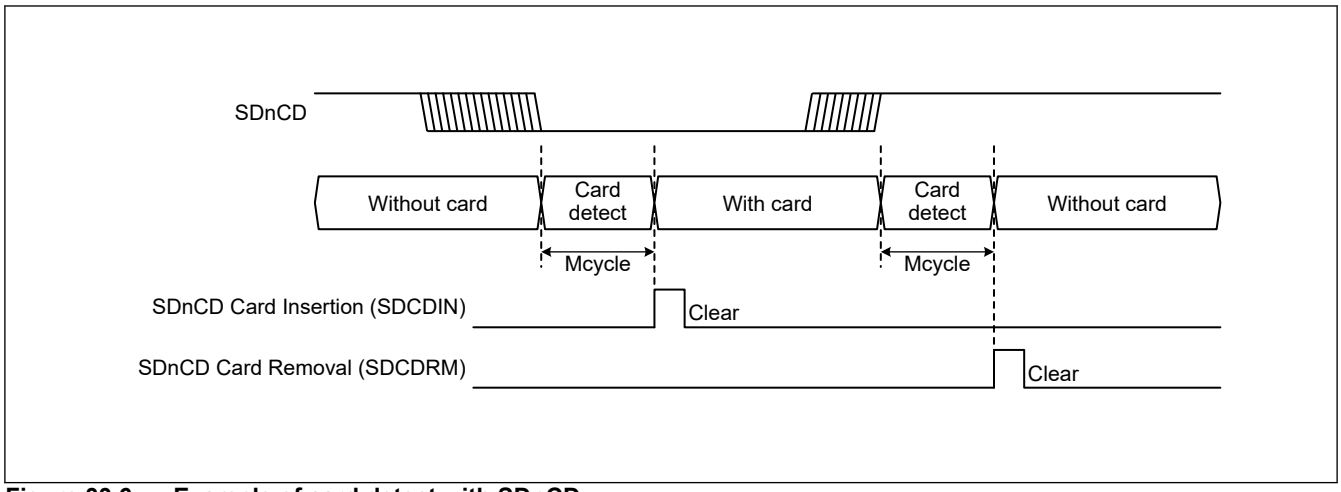



**Figure 33.6 Example of card detect with SDnCD**

### (4) SD card detect with SDnDAT3 (n = 0)

Figure 33.7 shows the timing when the SD card is detected with SDnDAT3. In addition, SDnDAT3 is pulled down by the host device, and the resistance value for pulling down is determined by the specification of the SD host device.

### (5) Card insertion

When an SD card is inserted, SDnDAT3 is pulled up and SDD3IN in SD_INFO1 is set to 1. It is cleared by writing 0.

### (6) Card removal

When an SD card is removed, SDnDAT3 is pulled down and SDD3RM in SD_INFO1 is set to 1. It is cleared by writing 0.

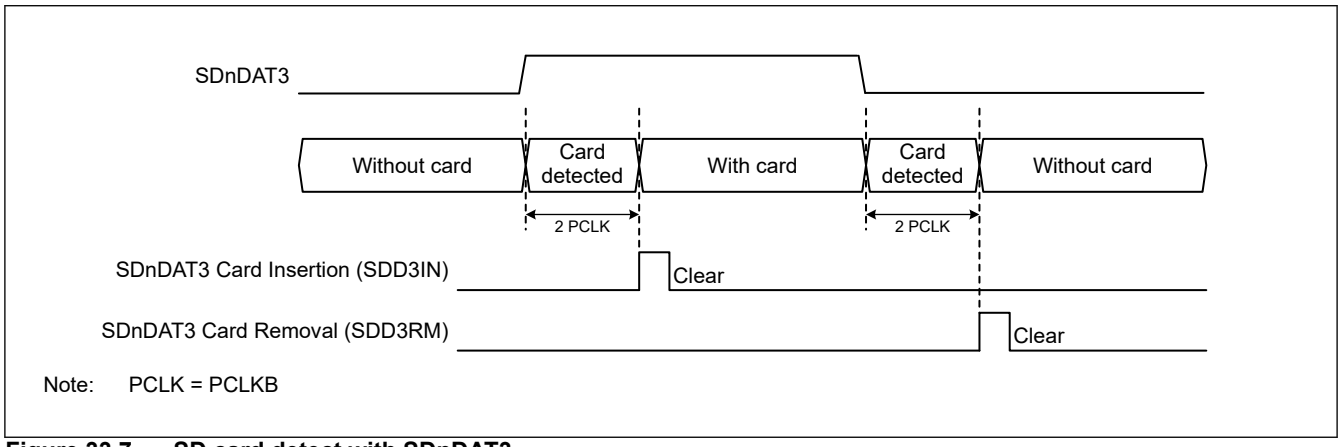



**Figure 33.7 SD card detect with SDnDAT3**

## 33.3.2.2 Write protect

The SD/MMC Host Interface has two types of write protect functions.

### (1) Write protect with SDnWP (n = 0)

SDnWP is connected to the card socket and pulled up or pulled down by the card insertion. The selection of pulling up or pulling down and the resistance value is determined by the specification of the SD host device. When the SDnWP state is reflected to SDWPMON in SD_INFO1, the write protect state is set after the SD card is inserted.

### (2) Write protect with command

The internal write protection of the card and the lock/unlock operation of the card are realized by the command.

## 33.3.3 Interrupt Request and DMA Transfer Request

### 33.3.3.1 Interrupts

Table 33.5 lists the SDHI interrupt sources. The SDHI requests an interrupt when:

- The status flags in registers SD_INFO1, SD_INFO2, and SDIO_INFO1 set to 1.
- The associated bits in the SD_INFO1_MASK, SD_INFO2_MASK, and SDIO_INFO1_MASK registers are 0.

When clearing the status flags in registers SD_INFO1, SD_INFO2, and SDIO_INFO1, write 0 to the status flags to be cleared and write 1 to the status flags that are not being cleared.

**Table 33.5 Interrupt sources**

| | Status flag register | | Interrupt mask register | | Interrupt name |
|---|---|---|---|---|---|
| **Interrupt sources** | **Register symbol** | **Bit symbol** | **Register symbol** | **Bit symbol** | **Ch n (n = 0)** |
| Card Access Interrupt (CACI) | SD_INFO1 | ACEND | SD_INFO1_MASK | ACENDM | SDHI_MMCn_ACCS |
| | | RSPEND | | RSPENDM | |
| | SD_INFO2 | ILA | SD_INFO2_MASK | ILAM | |
| | | BWE | | BWEM | |
| | | BRE | | BREM | |
| | | RSPTO | | RSPTOM | |
| | | ILR | | ILRM | |
| | | ILW | | ILWM | |
| | | DTO | | DTOM | |
| | | ENDE | | ENDEM | |
| | | CRCE | | CRCEM | |
| | | CMDE | | CMDEM | |
| SDIO Access Interrupt (SDACI) | SDIO_INFO1 | EXWT | SDIO_INFO1_MASK | EXWTM | SDHI_MMCn_SDIO |
| | | EXPUB52 | | EXPUB52M | |
| | | IOIRQ | | IOIRQM | |
| Card Detect Interrupt (CDETI) | SD_INFO1 | SDD3IN | SD_INFO1_MASK | SDD3INM | SDHI_MMCn_CARD |
| | | SDD3RM | | SDD3RMM | |
| | | SDCDIN | | SDCDINM | |
| | | SDCDRM | | SDCDRMM | |

### 33.3.3.2 DMA transfer requests (SDHI_MMCn_ODMSDBREQ, n = 0)

The SD/MMC Host Interface has two types of DMA transfer requests.

### (1) SD_BUF write DMA transfer request

- When the BWE bit in SD_INFO2 is set to 1 while the DMAEN bit in SD_DMAEN is set to 1, the SD_BUF write DMA transfer request is asserted.

- The SD_BUF write DMA transfer request is negated when the last data in one block (based on the transfer data size set in SD_SIZE) is transferred. The SD_BUF write DMA transfer request is also negated by clearing the SDRST bit in SOFT_RST to 0 or setting the STP bit in SD_STOP to 1. However, if a communications error or timeout occurs at the DMA transfer, the SD_BUF write DMA transfer request is not negated.
- The BWE bit in SD_INFO2 is cleared after transfer of the last data in one block following a request for writing to SD_BUF by DMA transfer.
- The number of DMA transfers must be n × one block. (n = integer, one block = the transfer data size set in SD_SIZE)
- When the IOABT bit in SDIO_MODE is set to 1, the SD_BUF write DMA transfer request is negated.
- The DMA transfer request is also negated by clearing the DMAEN bit to 0. However, the DMA transfer request is asserted again when the DMAEN bit is set to 1 before writing to SD_CMD.
- Because the BWE bit in SD_INFO2 is not cleared in response to setting the STP/IOABT bit, or to a communications error or timeout, clear the bit to 0 before issuing the next command. The next request to write to SD_BUF by DMA transfer is not issued while the BWE bit is set.

#### (2) SD_BUF read DMA transfer request

- When the BRE bit in SD_INFO2 is set to 1 while the DMAEN bit in the SD_DMAEN register is set to 1, the SD_BUF read DMA transfer request is asserted.
- The SD_BUF read DMA transfer request is negated when the last data in one block (based on the transfer data size set in SD_SIZE) is transferred. The SD_BUF read DMA transfer request is also negated by clearing the SDRST bit in SOFT_RST to 0 or setting the STP bit in SD_STOP to 1. However, if a communications error or timeout occurs at the DMA transfer, the SD_BUF read DMA transfer request is not negated.
- The BRE bit in SD_INFO2 is cleared after transfer of the last data in one block following a request to write to SD_BUF by DMA transfer.
- The number of DMA transfers must be n × one block. (n = integer, one block = the transfer data size set in SD_SIZE)
- When the IOABT bit in SDIO_MODE is set to 1, the SD_BUF read DMA transfer request is negated.
- The DMA transfer request is also negated by clearing the DMAEN bit to 0. However, the DMA transfer request is asserted again when the DMAEN bit is set to 1 before writing to SD_CMD.
- Because the BRE bit in SD_INFO2 is not cleared in response to setting the STP/IOABT bit or in response to a communications error or timeout, clear the bit to 0 before issuing the next command. The next request to write to SD_BUF by DMA transfer is not issued while the BRE bit is set.

### 33.3.4 Communication Errors and Timeouts

When a communication error or timeout error occurs, depending on the type of error, the associated status flag in the SD_INFO2 register sets to 1. Also, depending on the source of the error, the associated flag in the SD_ERR_STS1 or SD_ERR_STS2 register sets to 1.

The status flags in registers SD_ERR_STS1 and SD_ERR_STS2 clear to 0 by writing to the SD_CMD register, or by setting the SOFT_RST.SDRST bit to 0.

**Table 33.6 Communication errors**

| Communication error | Interrupt flag register | | Error status register | | This occurs when… |
|---|---|---|---|---|---|
| | Register symbol | Bit symbol | Register symbol | Bit symbol | |
| End bit error | SD_INFO2 | ENDE | SD_ERR_STS1 | CRCLENE | The CRC status token length is in error |
| | | | | RDLENE | The read data length is in error |
| | | | | RSPLENE1 | The response length is in error*1 |
| | | | | RSPLENE0 | The response length is in error*2 |
| CRC error | | CRCE | | CRCTKE | The CRC status token is in error |
| | | | | RDCRCE | There is a CRC error in the read data |
| | | | | RSPCRCE1 | There is a CRC error in the response*1 |
| | | | | RSPCRCE0 | There is a CRC error in the response*2 |
| Command error | | CMDE | | CMDE1 | The command index field value for the transmitted command and received response do not match*1 |
| | | | | CMDE0 | The command index field value for the transmitted command and received response do not match*2 |

Note 1. CMD12 when automatic issuing is enabled for multiple block transfer by the setting in SD_CMD, CMD12 when the STP bit in SD_STOP is set to 1, or CMD52 when the C52PUB or IOABT bit in SDIO_MODE is set to 1.
Note 2. CMD other than CMD12 when automatic issuing is enabled for multiple block transfer by the setting in SD_CMD, CMD12 when the STP bit in SD_STOP is set to 1, or CMD52 when the C52PUB or IOABT bit in SDIO_MODE is set to 1.

**Table 33.7 Timeouts**

| Timeout | Interrupt flag register | | Error status register | | This occurs when… |
|---|---|---|---|---|---|
| | Register symbol | Bit symbol | Register symbol | Bit symbol | |
| Response timeout | SD_INFO2 | RSPTO | SD_ERR_STS2 | RSPTO1 | A response is not received even after a minimum of 640 SDHI clock cycles elapse*1 |
| | | | | RSPTO0 | A response is not received even after a minimum of 640 SDHI clock cycles elapse*2 |
| Data timeout (excluding response timeout) | | DTO | | CRCBSYTO | After the CRC status token is received, the SDHI is busy for at least the period set*3 |
| | | | | CRCTO | After the write data is transmitted, the CRC status token is not received even after at least the period set*3 elapses |
| | | | | RDTO | After the read command is issued, the read data is not received even after at least the period set*3 elapses |
| | | | | | After the read data is received, the next block read data is not received even after at least the period set*3 elapses |
| | | | | | After the SDHI exits the read wait state, the next block read data is not received even after at least the period set*3 elapses |
| | | | | BSYTO1 | After CMD12 is issued during the command sequence, the SDHI is busy for at least the period set*3 |
| | | | | BSYTO0 | After the R1b response is received, the SDHI is busy for at least the period set*3 (a command other than CMD12 is issued during the command sequence) |

Note 1. CMD12 when automatic issuing is enabled for multiple block transfer by the setting in SD_CMD, CMD12 when the STP bit in SD_STOP is set to 1, or CMD52 when the C52PUB or IOABT bit in SDIO_MODE is set to 1.
Note 2. CMD other than CMD12 when automatic issuing is enabled for multiple block transfer by the setting in SD_CMD, CMD12 when the STP bit in SD_STOP bit is set to 1, or CMD52 when the C52PUB or IOABT bit in SDIO_MODE is set to 1.

Note 3. The period is set in the SD_OPTION.TOP[3:0] bits.

### 33.3.5 Command without Data Transfer (SD/MMC)

Figure 33.8 and Figure 33.9 show example flows.

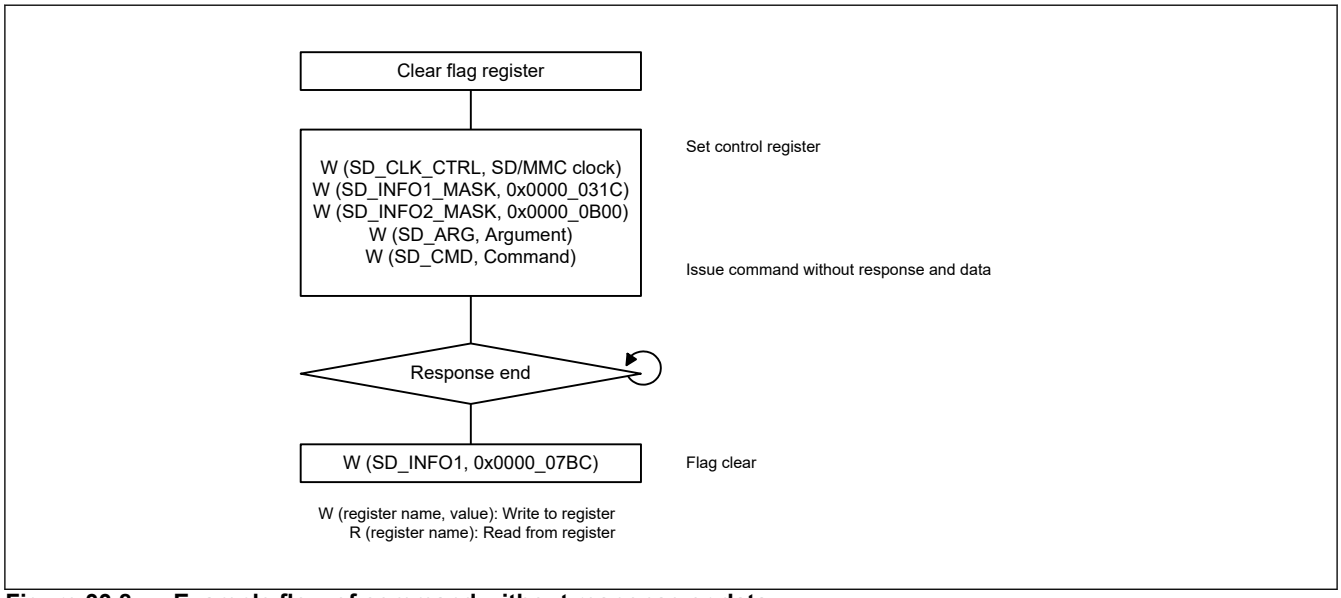



**Figure 33.8 Example flow of command without response or data**

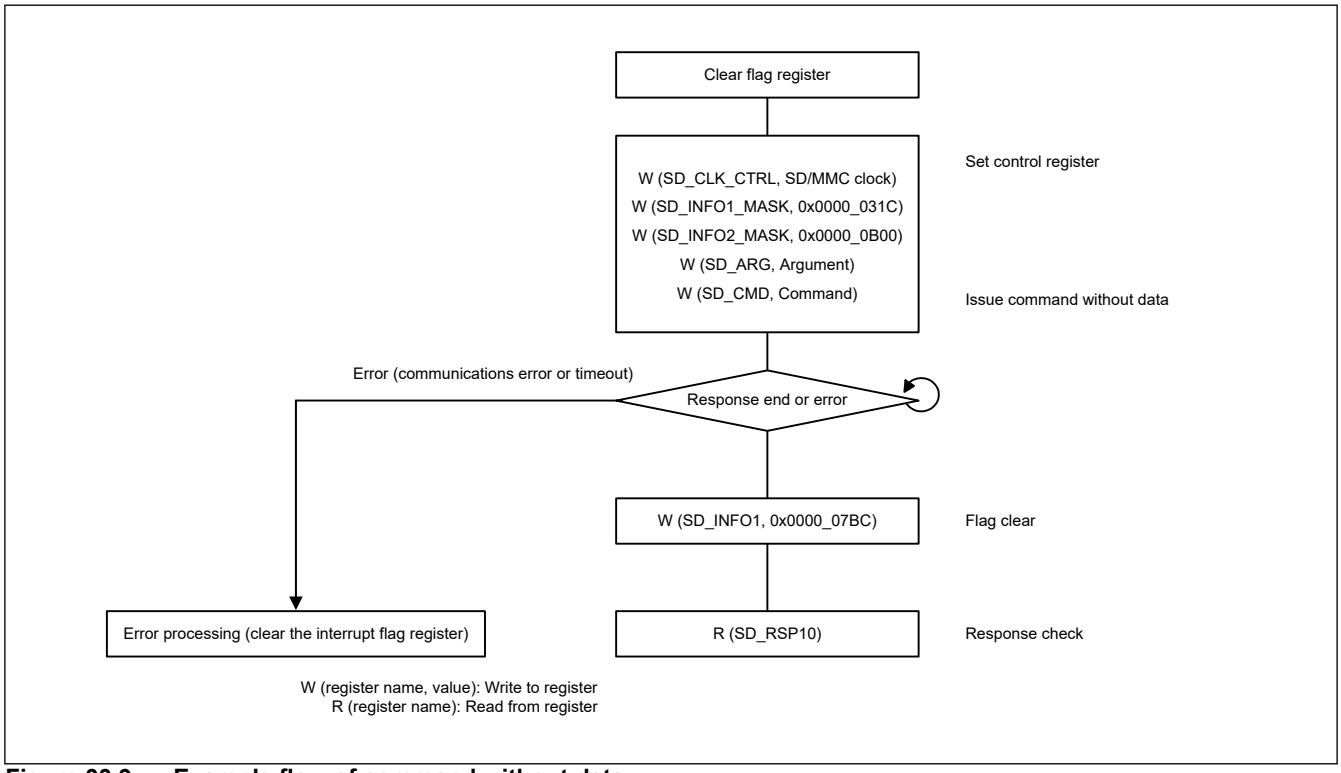



**Figure 33.9 Example flow of command without data**

#### 33.3.5.1 Operation for command without data transfer

The following legend is used for description of register read/write.

W (register name, value): Write to register

R (register name): Read from register

The operation is described in the following section.

#### (1) Command without response and data

a. Flag register clear
First, clear the bits in the flag register. (SD_INFO1 and SD_INFO2)

b. Control register set
Set the SD/MMC clock and interrupt masking. (SD_CLK_CTRL, SD_INFO1_MASK, and SD_INFO2_MASK)

c. Command issue
Set CMD argument in SD_ARG and write to SD_CMD.
Accordingly, CMD is issued, and the operation is started.

d. Flag clear
When transmission of a command is completed, RSPEND (response end) in SD_INFO1 is set to 1 to generate an interrupt. Clear RSPEND to 0.

#### (2) Command without data

a. Flag register clear
First, clear the bits in the flag register. (SD_INFO1 and SD_INFO2)

b. Control register set
Set the SD/MMC clock and interrupt masking. (SD_CLK_CTRL, SD_INFO1_MASK, and SD_INFO2_MASK)

c. Command issue
Set CMD argument in SD_ARG and write to the SD_CMD.
Accordingly, CMD is issued, and the operation is started.

d. Flag clear
When a response is received, RSPEND (response end) in SD_INFO1 is set to 1 to generate an interrupt. Clear RSPEND to 0.

e. Read a response from SD_RSP10. Additionally, perform error processing (clear the interrupt flag register) if a communication error or timeout occurs.

### 33.3.6 Single Block Read (SD/MMC)

Figure 33.10 shows an example flow of a single block read operation.

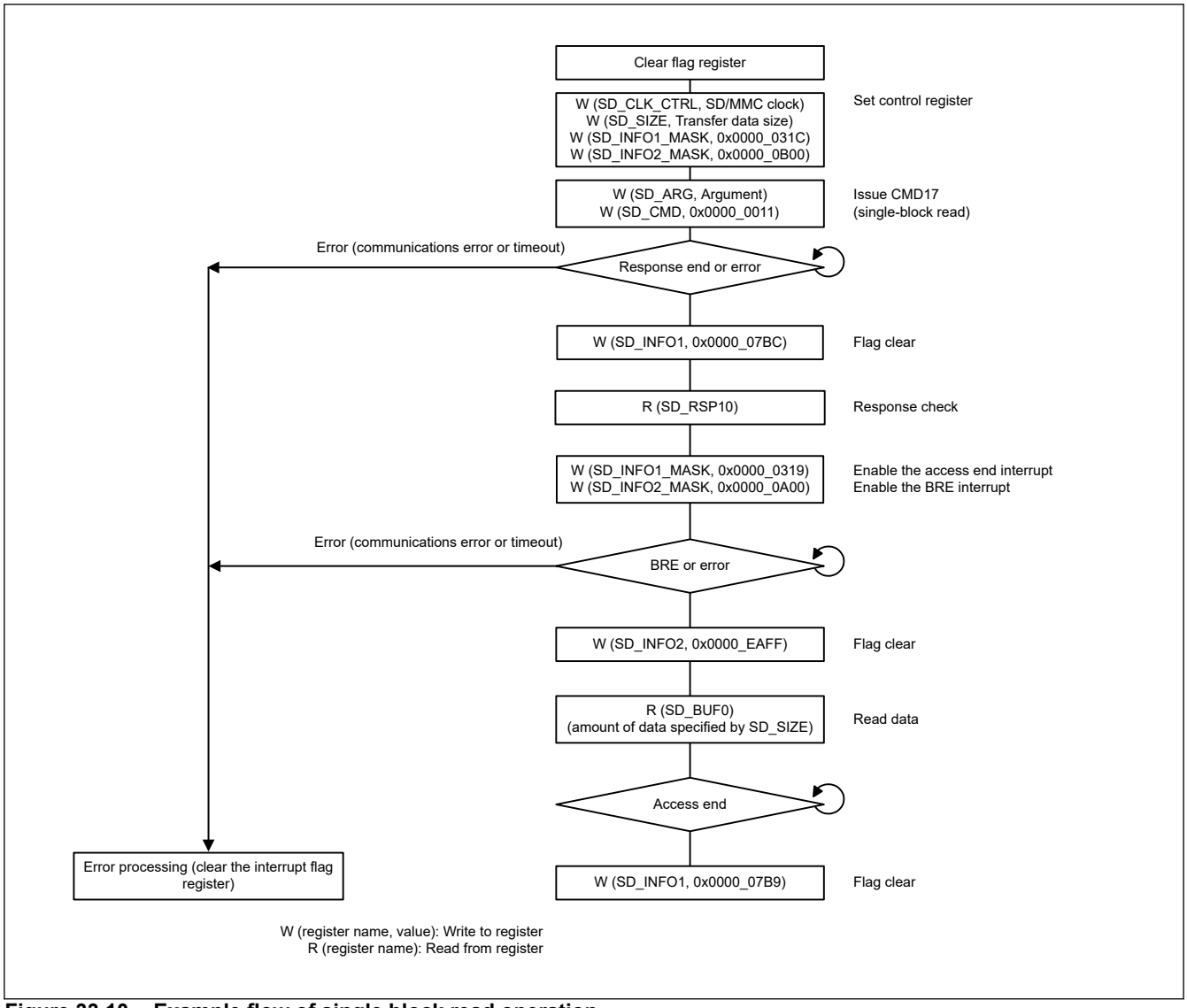



**Figure 33.10 Example flow of single block read operation**

### 33.3.6.1 Single block read operation

The operation of the single block read is described as follows:

a. Flag register clear
First, clear the bits in the flag register (SD_INFO1 and SD_INFO2).

b. Control register set
Set the SD/MMC clock, transfer data size, interrupt mask (SD_CLK_CTRL, SD_SIZE, SD_INFO1_MASK, and SD_INFO2_MASK).

c. Command issue (CMD17)
Set CMD17 argument in SD_ARG and write 0x0000_0011 to SD_CMD. CMD17 is issued and the single block read operation is started.

d. Response check
On receiving the response, RSPEND (response end) in SD_INFO1 is set to 1 to generate an interrupt. Clear RSPEND to 0 and read the response from SD_RSP10. If the result of response decoding is an error, the command sequence can be halted by setting the STP bit in SD_STP or the IOABT bit in SDIO_MODE to 1. In addition, this causes CMD12 and CMD52 to not be issued. If the ACEND bit (access end) in SD_INFO1 is set, halting the command sequence also leads to the generation of an interrupt.

e. Data receive from SD card/MMC and data read

Write 0x0000_0319 to SD_INFO1_MASK to enable the access end interrupt. In addition, write 0x0000_0A00 to SD_INFO2_MASK to enable the BRE interrupt. When the data received from the SD card/MMC is completed, the BRE bit in SD_INFO2 is set to 1 to generate an interrupt. Clear the BRE bit to 0 and read the amount of data specified in SD_SIZE from SD_BUF0.
A communication error or timeout might be generated if data is being received while reading of SD_BUF0 is in progress.

f. Operation complete
When the data read from SD_BUF0 is completed, ACEND (access end) in SD_INFO1 is set to 1 to generate an interrupt. Clear ACEND to 0 to end the single block read operation.
Additionally, perform error processing (clear the interrupt flag register) if a communication error or timeout occurs.

## 33.3.7 Single Block Write (SD/MMC)

Figure 33.11 shows an example flow of a single block write operation.

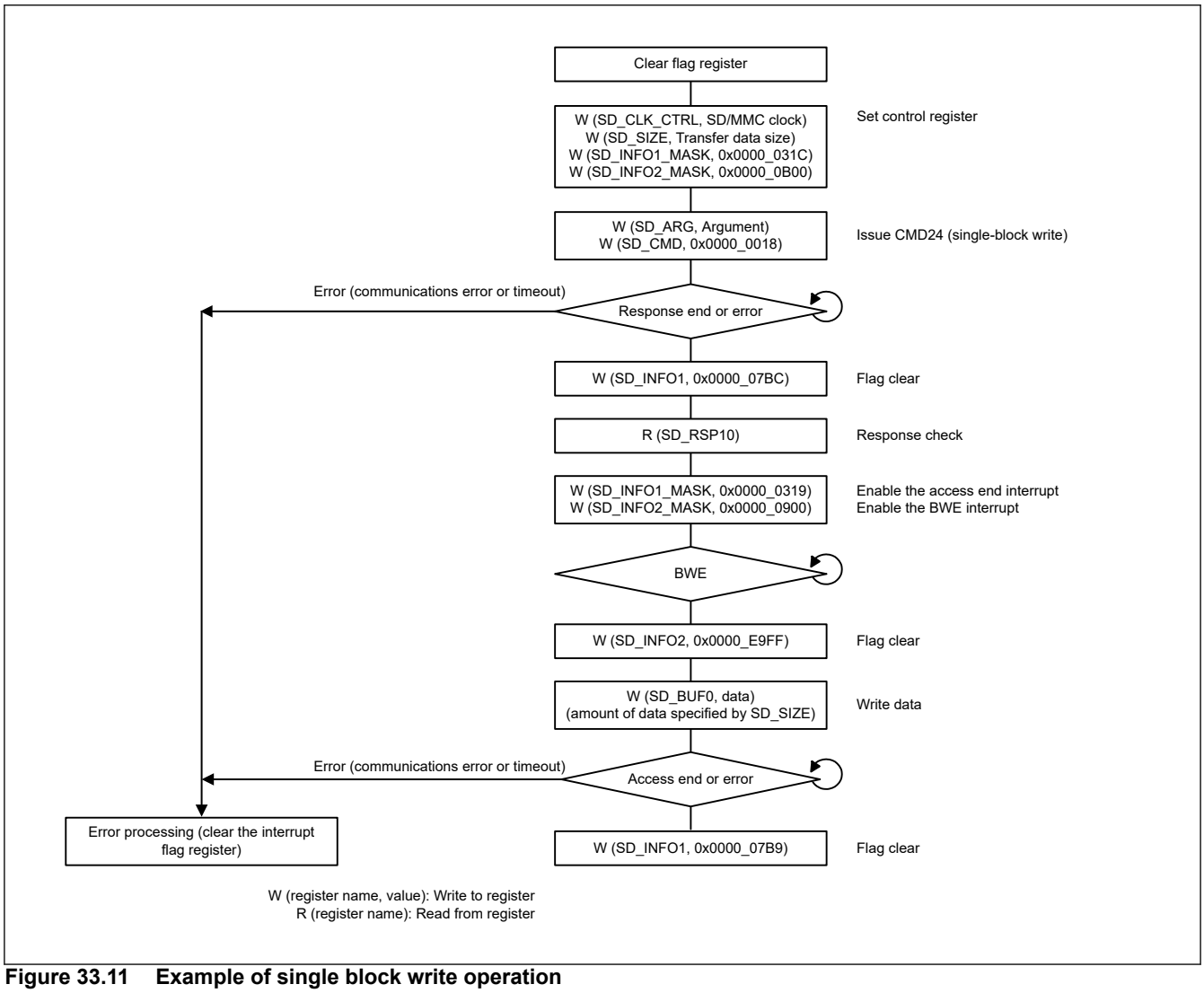



**Figure 33.11 Example of single block write operation**

### 33.3.7.1 Single block write operation

The operation of the single block write is described as follows:

a. Flag register clear
First, clear the bits in the flag register (SD_INFO1 and SD_INFO2).

b. Control register set

Set the SD/MMC clock, transfer data size, interrupt mask (SD_CLK_CTRL, SD_SIZE, SD_INFO1_MASK, and SD_INFO2_MASK).

c. Command issue (CMD24)
Set CMD24 argument in SD_ARG and write 0x0000_0018 to SD_CMD. CMD24 is issued and the single block write operation is started.

d. Response check
On receiving the response, RSPEND (response end) in SD_INFO1 is set to 1 to generate an interrupt. Clear RSPEND to 0 and read the response from SD_RSP10. If the result of response decoding is an error, the command sequence can be halted by setting the STP bit in SD_STP or the IOABT bit in SDIO_MODE to 1. In addition, this causes CMD12 and CMD52 to not be issued. If the ACEND bit (access end) in SD_INFO is set, halting the command sequence also leads to the generation of an interrupt.

e. Data write and data transmit to SD card/MMC
Write 0x0000_0319 to SD_INFO1_MASK to enable the access end interrupt. In addition, write 0x0000_0900 to SD_INFO2_MASK to enable the BWE interrupt. When SD_BUF0 is ready for the data to be written, the BWE bit in SD_INFO2 is set to 1 to generate an interrupt. Clear the BWE bit to 0 and write the amount of data specified in SD_SIZE to SD_BUF0. When the data write to SD_BUF0 is completed, data is transmitted to the SD card. Then, the CRC status and busy state are received from the SD card/MMC.
However, a communications error or timeout might be generated if data is being transmitted after writing to SD_BUF0.

f. Operation complete
When the CRC status and busy state are received from the SD card/MMC, ACEND (access end) in SD_INFO1 is set to 1 to generate an interrupt. Clear the ACEND bit to 0 to end the single block write operation.
In addition, perform error processing (clear the interrupt flag register) if a communication error or timeout occurs.

### 33.3.8 Multiple Block Read (SD/MMC)

Figure 33.12 shows an example flow of a multiple block read operation.

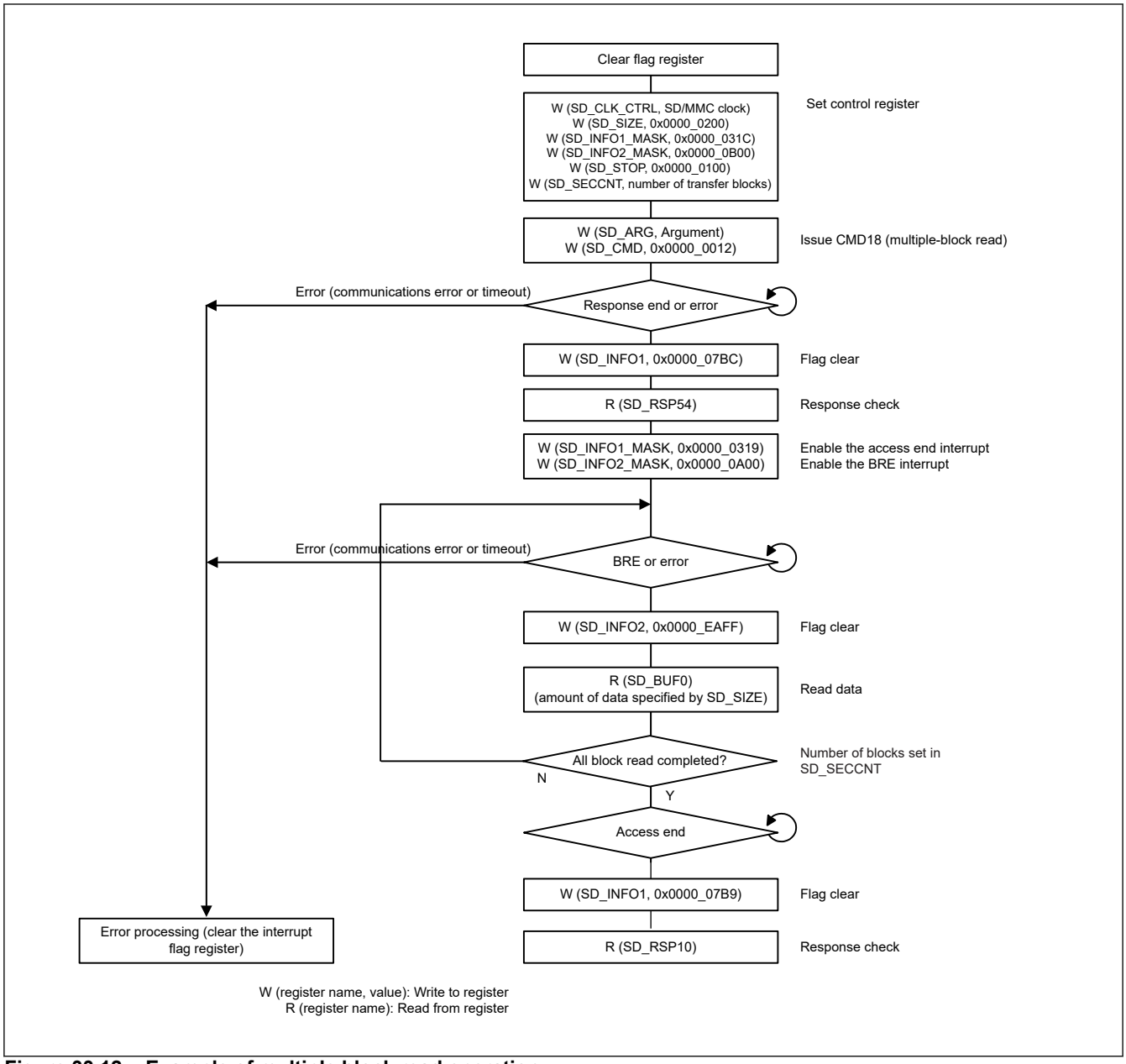



**Figure 33.12 Example of multiple block read operation**

### 33.3.8.1 Multiple block read operation

The operation of the multiple block read is described as follows:

a. Flag register clear
First, clear the bits in the flag register (SD_INFO1 and SD_INFO2).

b. Control register set
Set the SD/MMC clock, transfer data size, interrupt mask (SD_CLK_CTRL, SD_SIZE, SD_INFO1_MASK, and SD_INFO2_MASK).
Set SEC in SD_STOP to 1, and set the number of transfer blocks in SD_SECCNT.

c. Command issue (CMD18)
Set CMD18 argument in SD_ARG and write 0x0000_0012 to SD_CMD. CMD18 is issued and the multiple block read operation is started.

d. Response check
On receiving the response, RSPEND (response end) in SD_INFO1 is set to 1 to generate an interrupt. Clear RSPEND to 0 and read the response from SD_RSP54. If the result of response decoding is an error, the command sequence can be

halted by setting the STP bit in SD_STP to 1. Setting the STP bit to 1 also causes CMD12 to be issued and the response received. If the command sequence is halted because the access end interrupt is enabled, an interrupt is generated when the ACEND bit (access end) bit in SD_INFO1 sets to 1 on completion of response reception. Clear the ACEND bit to 0 and read the response.

e. Data receive from SD card/MMC and data read
Write 0x0000_0319 to SD_INFO1_MASK to enable the access end interrupt. In addition, write 0x0000_0A00 to SD_INFO2_MASK to enable the BRE interrupt. When one-block data received from the SD card/MMC is completed, the BRE bit in SD_INFO2 is set to 1 to generate an interrupt. Clear the BRE bit to 0 and read the amount of data specified in SD_SIZE from SD_BUF0. Doing this repeats transfer of the number of blocks set in SD_SECCNT. However, a communication error or timeout might be generated if data is being received while reading of SD_BUF0 is in progress. CMD12 is automatically issued to stop multiblock transfer with the number of blocks that is set to SD_SECCNT and the response is received. At this point, CMD12 argument is automatically set to 0x0000_0000.

f. Operation complete
When all-block data read and the CMD12 response received are completed, ACEND (access end) in SD_INFO1 is set to 1 to generate an interrupt. Clear ACEND to 0 to read the response. This is the end of multiple block read operation. In addition, perform error processing (clear the interrupt flag register) if a communication error or timeout occurs.

### 33.3.9 Multiple Block Write (SD/MMC Using Internal Timer)

Figure 33.13 shows an example flow of a multiple block write using internal timer.

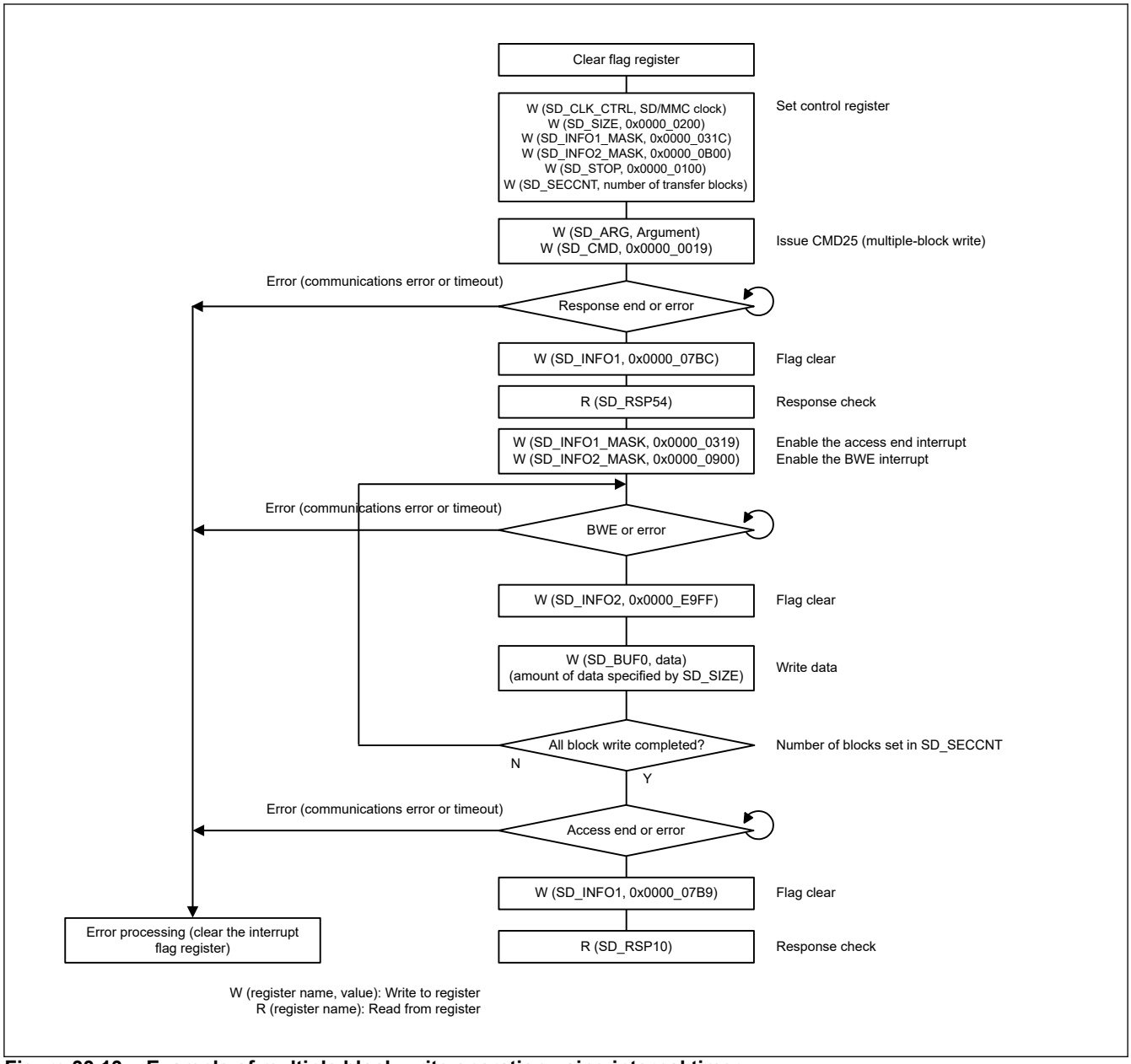



**Figure 33.13 Example of multiple block write operation using internal timer**

### 33.3.9.1 Multiple block write operation using internal timer

The operation of the multiple block write is described as follows:

a. Flag register clear
First, clear the bits in the flag register (SD_INFO1 and SD_INFO2).

b. Control register set
Set the SD/MMC clock, transfer data size, interrupt mask (SD_CLK_CTRL, SD_SIZE, SD_INFO1_MASK, and SD_INFO2_MASK).
Set the SEC bit in SD_STOP to 1, and set the number of transfer blocks in SD_SECCNT.

c. Command issue (CMD25)
Set CMD25 argument in SD_ARG and write 0x0000_0019 to SD_CMD. CMD25 is issued and the multiple block write operation is started.

d. Response check
On receiving the response, the RSPEND bit (response end) in SD_INFO1 is set to 1 to generate an interrupt.
Clear the RSPEND bit to 0 and read the response from SD_RSP54. If the result of response decoding is an error, the command sequence can be halted by setting the STP bit in SD_STP to 1. Setting the STP bit to 1 also causes CMD12 to

be issued and the response received. If the command sequence is halted because the access end interrupt is enabled, an interrupt is generated by when the ACEND bit (access end) bit in SD_INFO1 sets to 1 on completion of response reception. Clear the ACEND bit to 0 and read the response.

e. Data write and data transmit to SD card/MMC
Write 0x0000_0319 to SD_INFO1_MASK to enable the access end interrupt. In addition, write 0x0000_0900 to SD_INFO2_MASK to enable the BWE interrupt. When SD_BUF0 is ready for the data to be written, the BWE bit in the SD_INFO2 resister is set to 1 to generate an interrupt. Clear the BWE bit to 0 and write the amount of data specified in SD_SIZE to SD_BUF0. When the data write to SD_BUF0 is completed, data is transmitted to the SD card/MMC. The CRC status and busy state are received from the SD card/MMC. This repeats transfer of the number of blocks set in SD_SECCNT. However, a communication error or timeout might be generated if data is being received while writing to SD_BUF0 is in progress. CMD12 is automatically issued to stop multiblock transfer with the number of blocks which is set to SD_SECCNT and the response is received. At this point, CMD12 argument is automatically set to 0x0000_0000.

f. Operation complete
When all-block data transmit and the CRC status receive are completed, the ACEND bit (access end) in SD_INFO1 is set to 1 to generate an interrupt. Clear the ACEND bit to 0 to read the response. This is the end of multiple block write operation. Additionally, perform error processing (clear the interrupt flag register) if a communications error or timeout occurs.

### 33.3.10 Multiple Block Write (MMC using external timer)

Figure 33.14 shows an example flow of a multiple block write using an external timer.

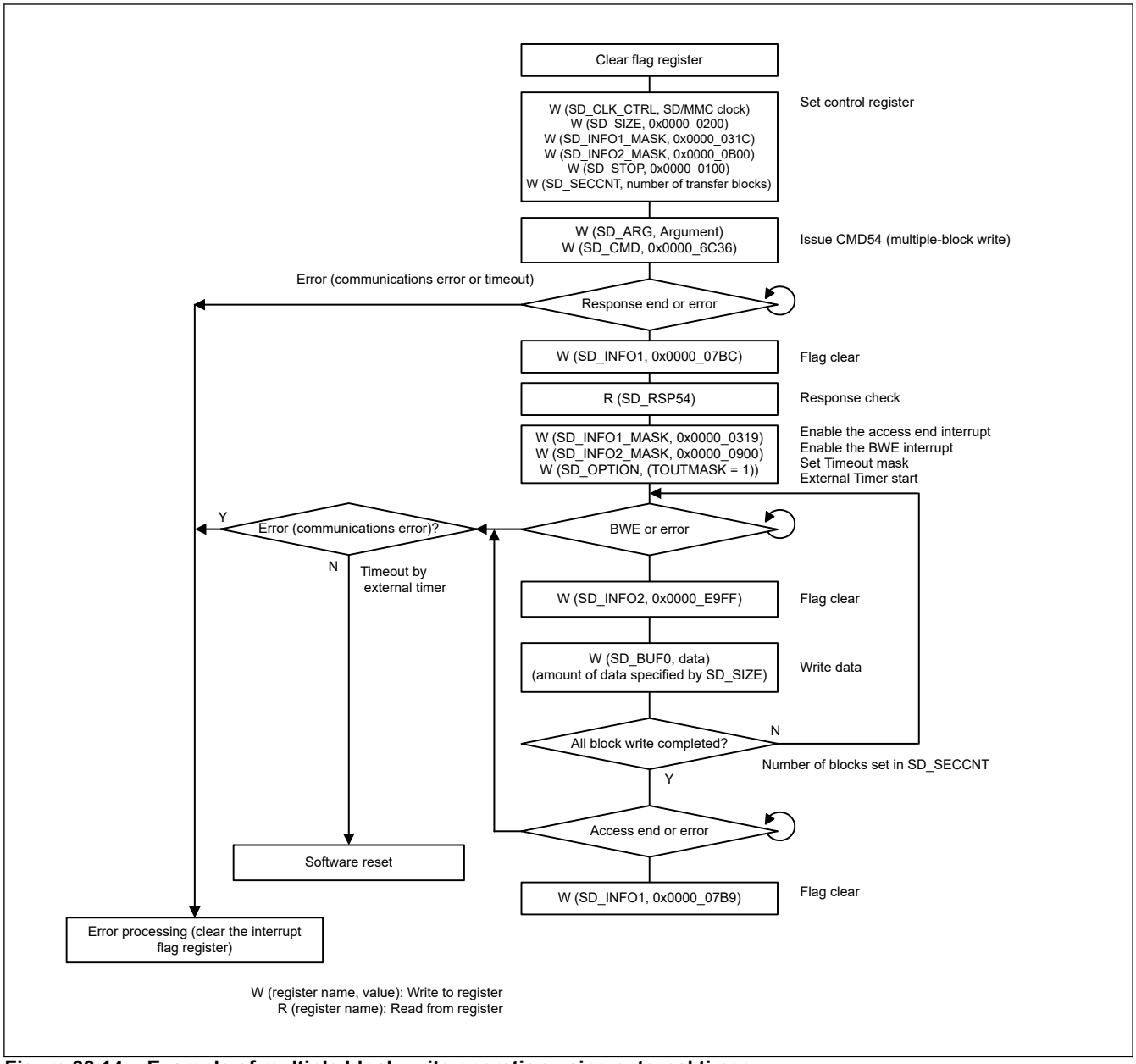



**Figure 33.14 Example of multiple block write operation using external timer**

### 33.3.10.1 Multiple block write operation using external timer

The operation of the multiple block write is described as follows:

a. Flag register clear
First, clear the bits in the flag register (SD_INFO1 and SD_INFO2).

b. Control register set
Set the MMC clock, transfer data size, interrupt mask (SD_CLK_CTRL, SD_SIZE, SD_INFO1_MASK, and SD_INFO2_MASK).
Set the SEC bit in SD_STOP to 1, and set the number of transfer blocks in SD_SECCNT.

c. Command issue (CMD54)
Set CMD54 Argument in SD_ARG and write 0x0000_6C36 to SD_CMD. CMD54 is issued and the multiple block write operation is started.

d. Response check
On receiving the response, the RSPEND bit (response end) in SD_INFO1 is set to 1 to generate an interrupt. Clear the RSPEND bit to 0 and read the response from SD_RSP54. If the result of response decoding is an error, the command sequence can be halted by setting the STP bit in SD_STP to 1. Setting the STP bit to 1 also causes CMD12 to be issued

and the response received. If the command sequence is halted because the access end interrupt is enabled, an interrupt is generated by when the ACEND bit (access end) bit in SD_INFO1 sets to 1 on completion of response reception. Clear the ACEND bit to 0 and read the response.

e. Data write and data transmit to MMC
Write 0x0000_0319 to SD_INFO1_MASK to enable the access end interrupt, write 0x0000_0900 to SD_INFO2_MASK to enable the BWE interrupt and set 1 to TOUTMASK of SD_OPTION to inactivate timeout. In addition, start external timer. When SD_BUF0 is ready for the data to be written, the BWE bit in the SD_INFO2 resister is set to 1 to generate an interrupt. Clear the BWE bit to 0 and write the amount of data specified in SD_SIZE to SD_BUF0. When the data write to SD_BUF0 is completed, data is transmitted to the MMC. The CRC status and busy state are received from the MMC. Doing this repeats transfer of the number of blocks set in SD_SECCNT. However, a communication error or timeout might be generated if data is being received while writing to SD_BUF0 is in progress.

f. Operation complete
When all-block data transmit and the CRC status receive are completed, the ACEND bit (access end) in SD_INFO1 is set to 1 to generate an interrupt. Clear the ACEND bit to 0 to read the response. This is the end of multiple block write operation. Additionally, perform error processing (clear the interrupt flag register) if a communications error or timeout occurs when receiving response. Execute software reset if a timeout by external timer occurs when transmitting data.

## 33.3.11 IO_RW_DIRECT Command (SD: CMD52)

Figure 33.15 shows an example flow of an IO_DIRECT command (CMD52) operation.

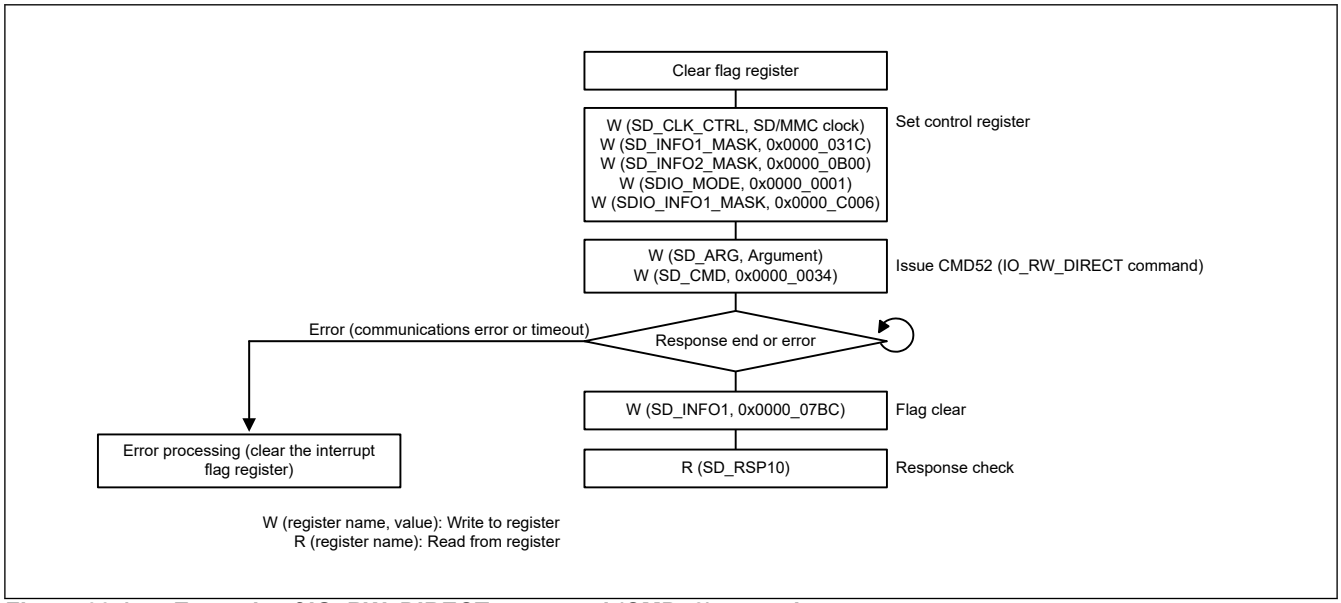



**Figure 33.15 Example of IO_RW_DIRECT command (CMD52) operation**

## 33.3.12 IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Read)

Figure 33.16 shows an example flow for a CMD53 multiple block read operation.

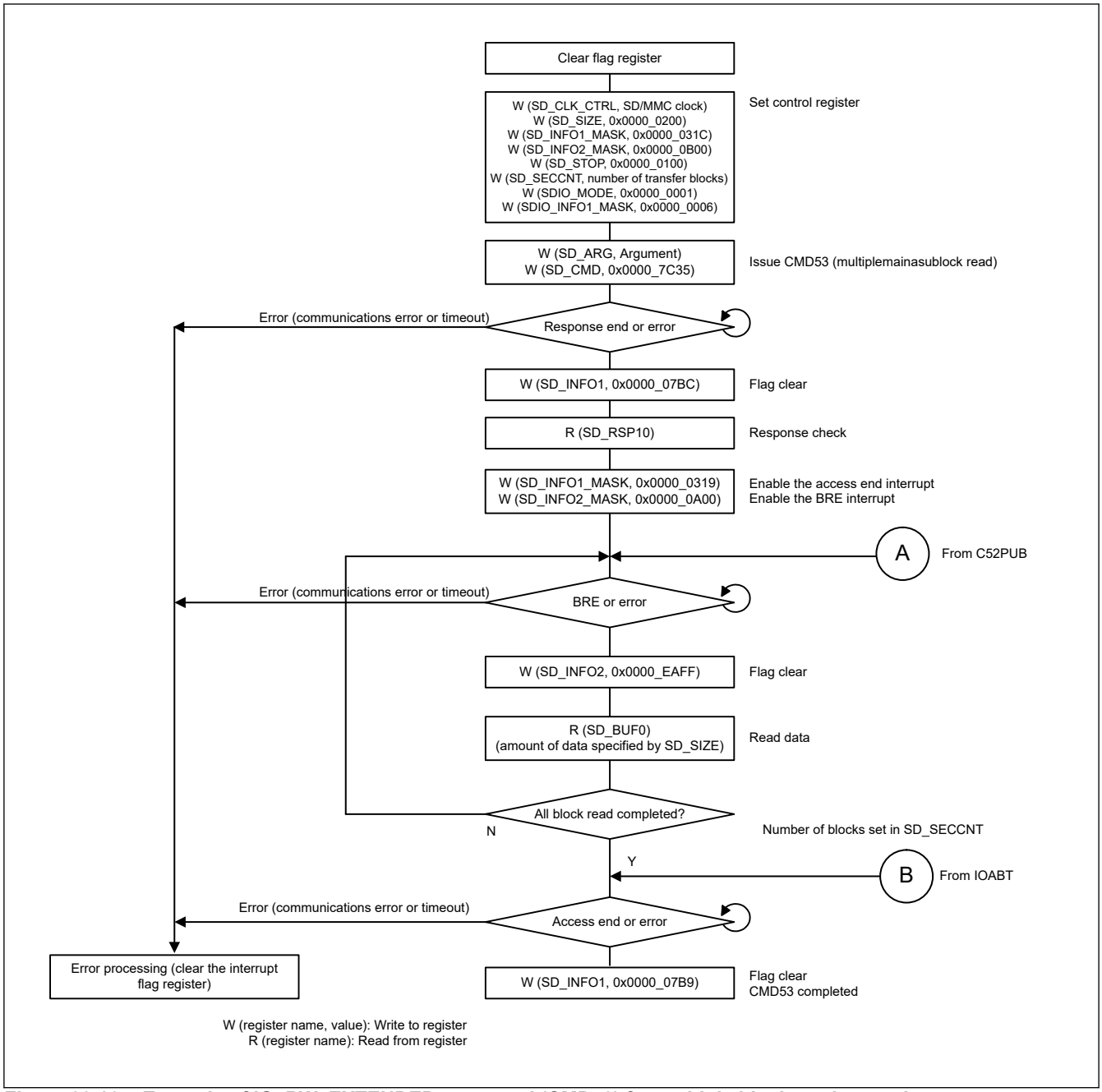



**Figure 33.16 Example of IO_RW_EXTENDED command (CMD53) for multiple block read operation**

Figure 33.17 shows an example flow when CMD52 (SDIO abort) is issued during a CMD53 multiple block read.

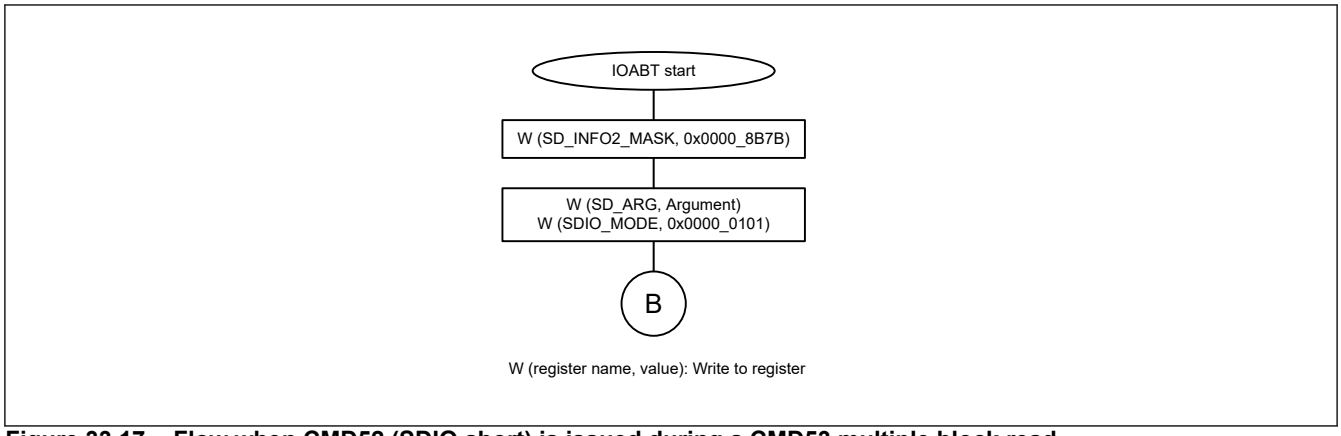



**Figure 33.17 Flow when CMD52 (SDIO abort) is issued during a CMD53 multiple block read**

Figure 33.18 shows an example flow when CMD52 (SDIO none abort) is issued at a CMD53 multiple block read while the SDHI is in the read wait state.

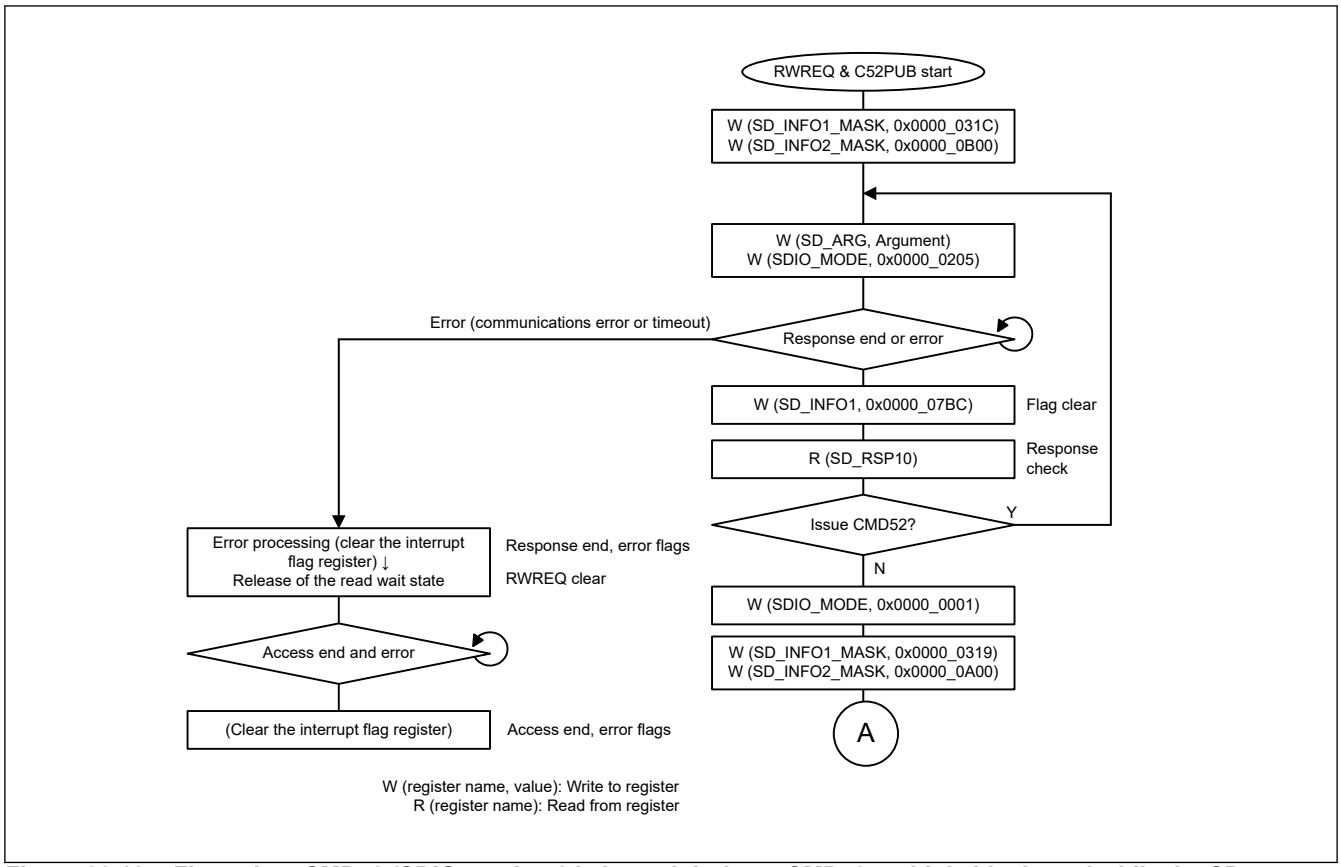



**Figure 33.18 Flow when CMD52 (SDIO no abort) is issued during a CMD53 multiple block read while the SD Host Interface is in read wait state**

## 33.3.13 IO_RW_EXTENDED Command (SD: CMD53/Multiple Block Write)

Figure 33.19 shows an example flow for a CMD53 multiple block write.

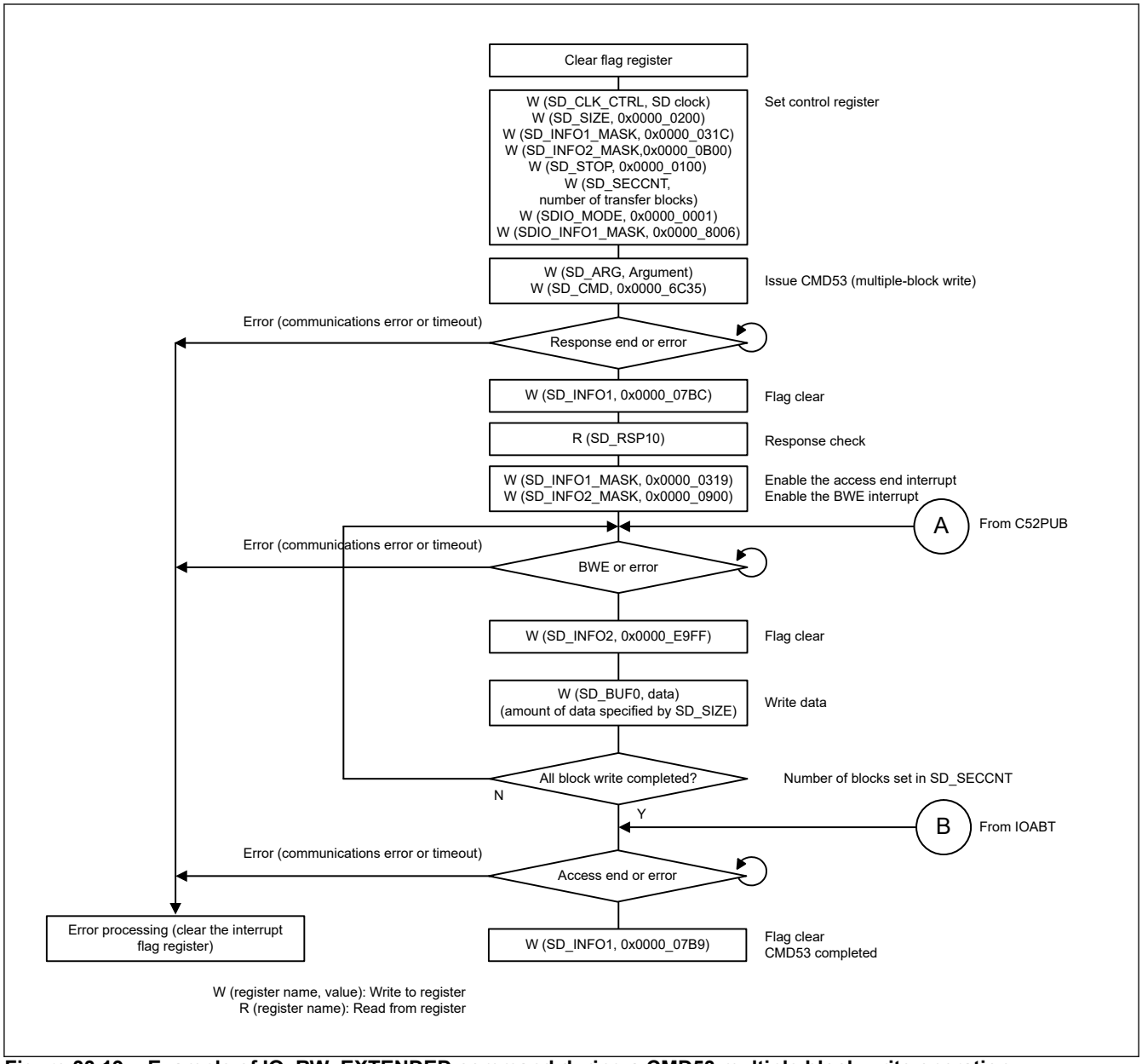



**Figure 33.19 Example of IO_RW_EXTENDED command during a CMD53 multiple block write operation**

Figure 33.20 shows an example flow when CMD52 (SDIO abort) is issued at CMD53 multiple block write.

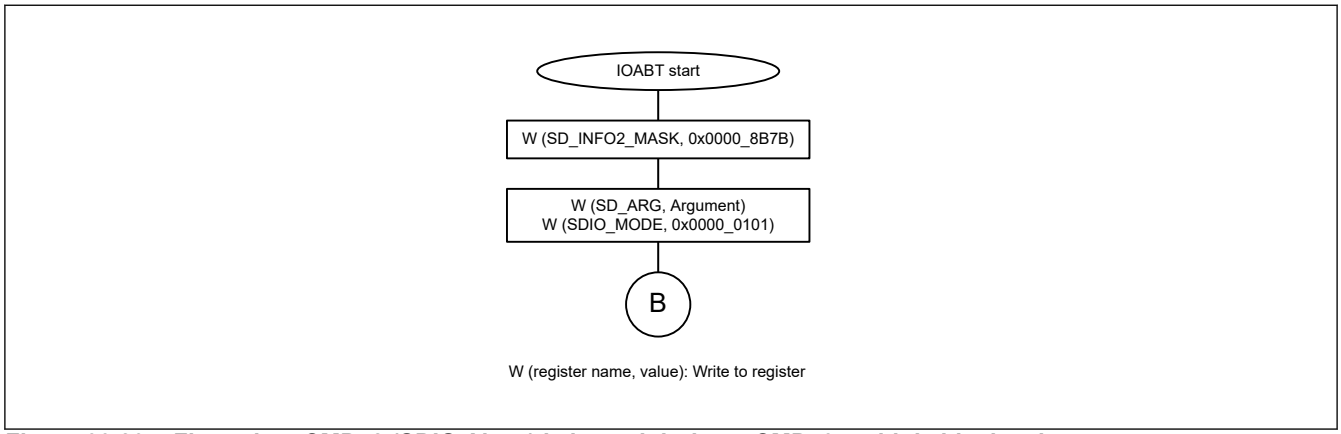



**Figure 33.20 Flow when CMD52 (SDIO Abort) is issued during a CMD53 multiple block write**

Figure 33.21 shows an example flow when CMD52 (SDIO none abort) is issued at CMD53 multiple block write.

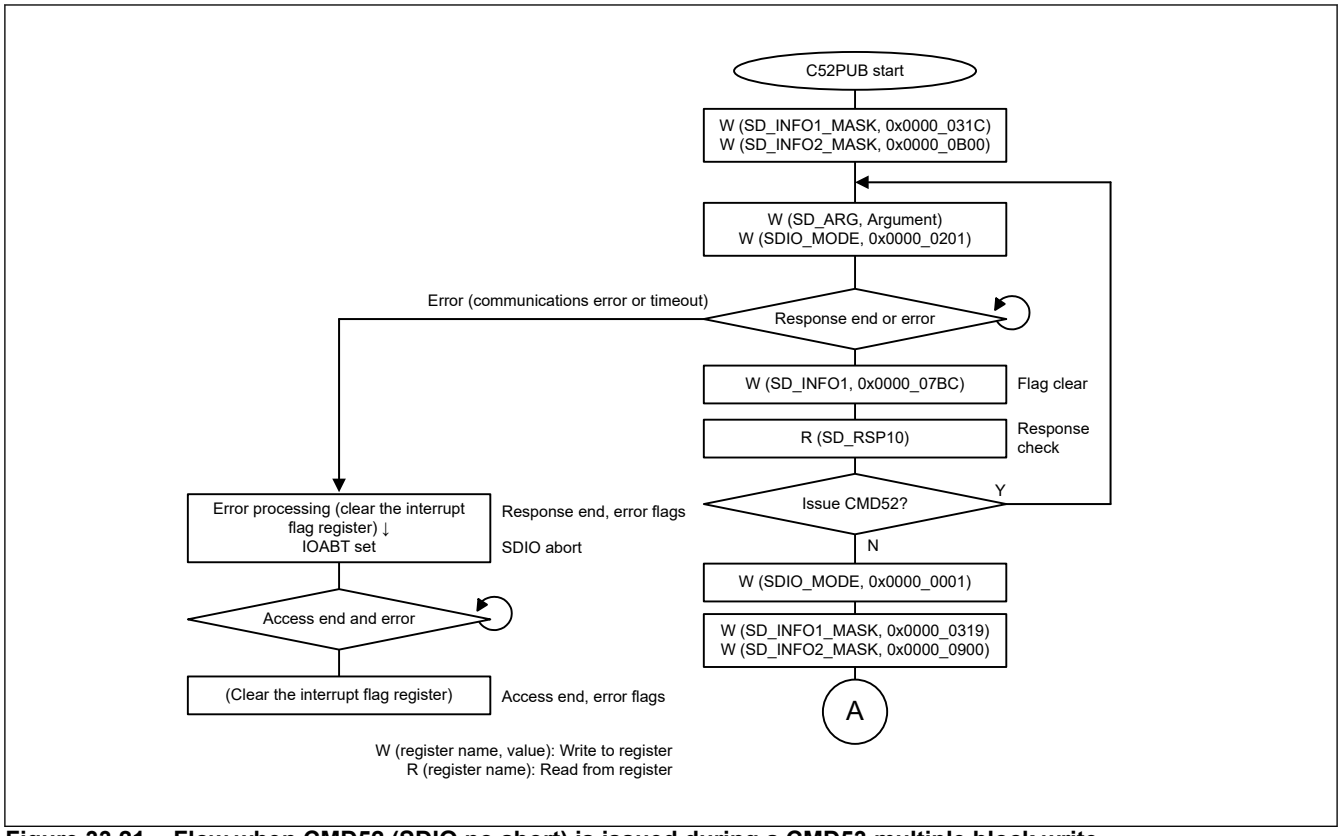



**Figure 33.21 Flow when CMD52 (SDIO no abort) is issued during a CMD53 multiple block write**

## 33.3.14 DMA Transfer (SD/MMC)

### 33.3.14.1 SD_BUF DMA transfer

Figure 33.22 shows an example flow for SD_BUF DMA read when CMD18 multiple block read is issued.

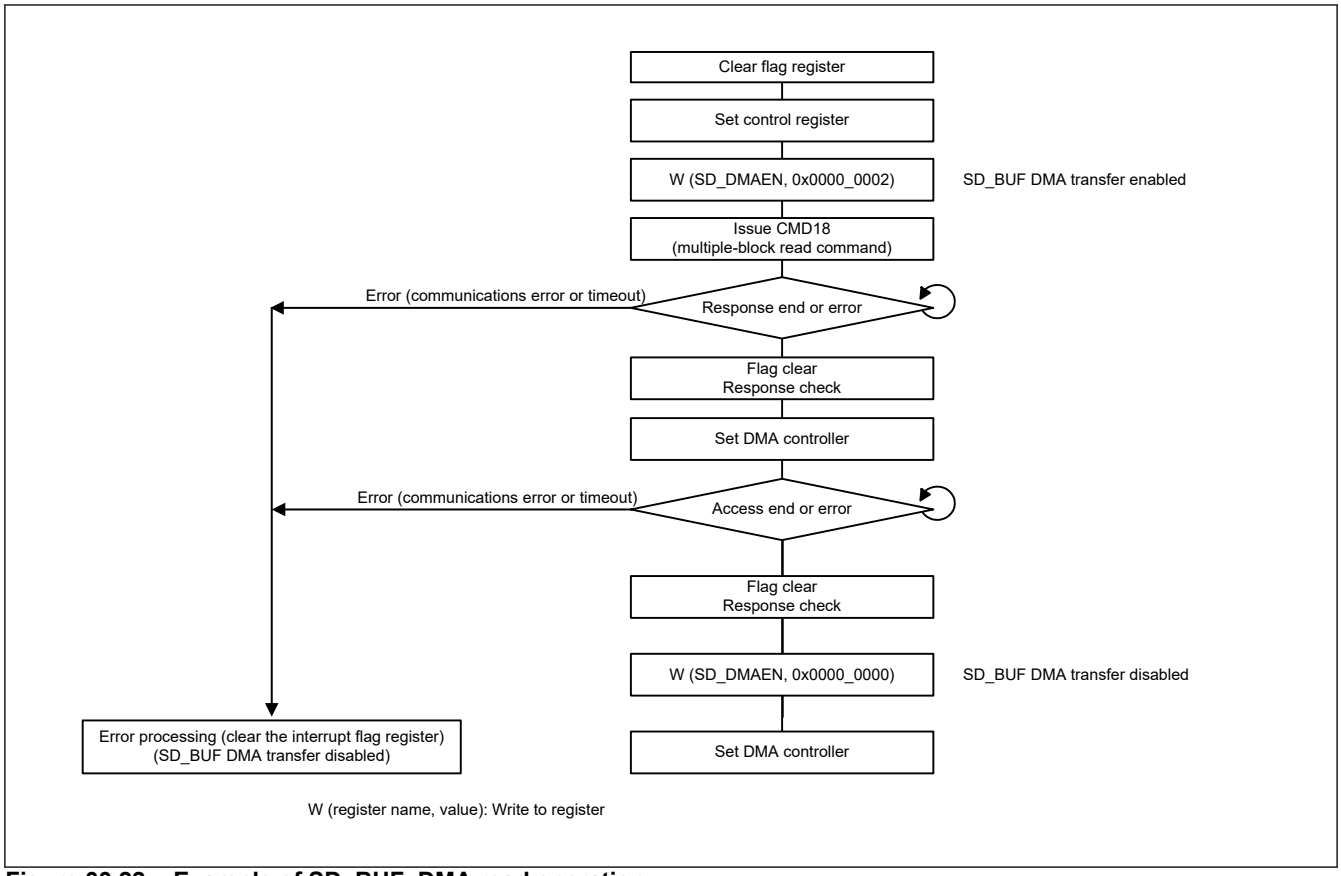



**Figure 33.22 Example of SD_BUF_DMA read operation**

Figure 33.23 shows an example flow for SD_BUF DMA write when CMD25 multiple block write is issued.

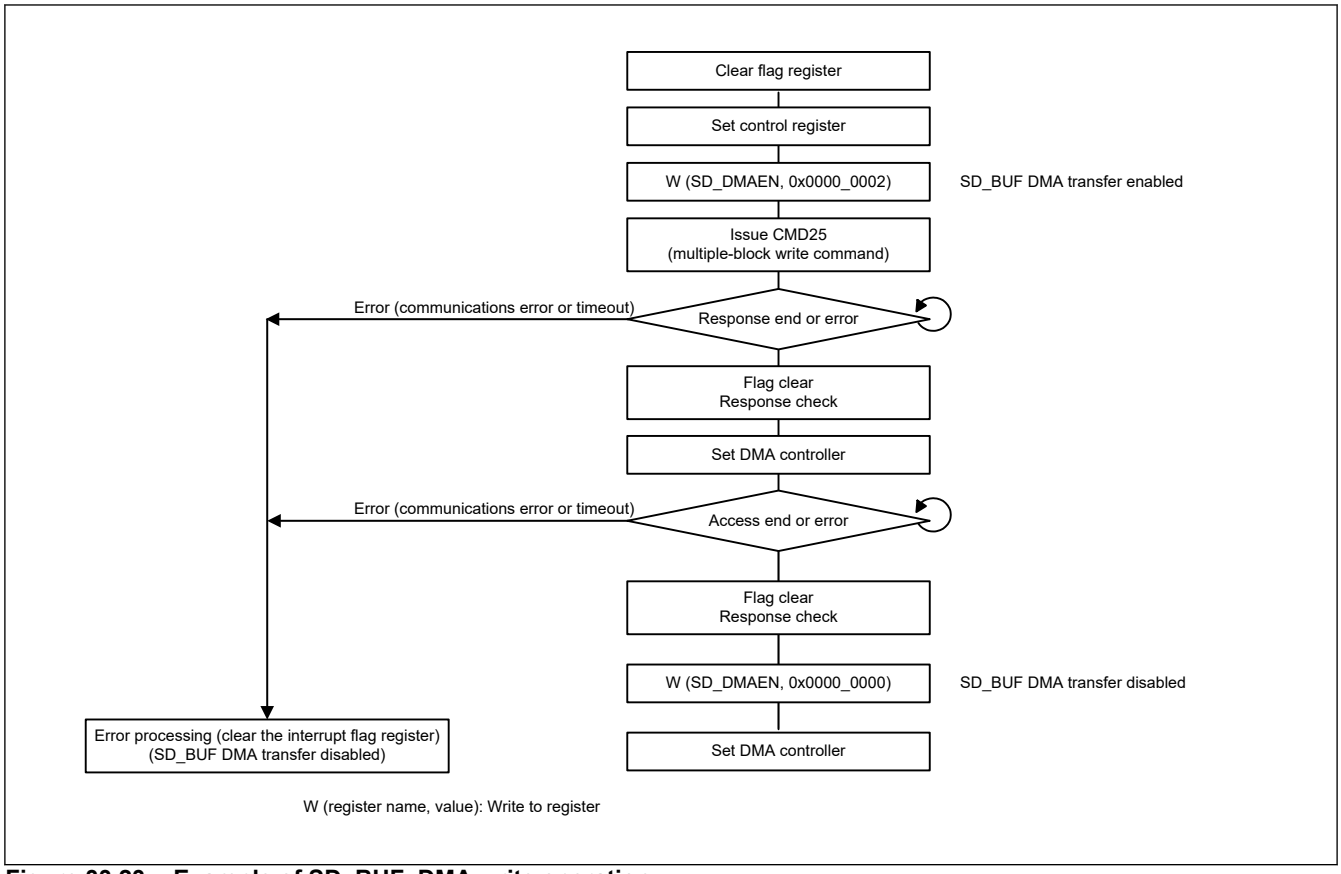



**Figure 33.23 Example of SD_BUF_DMA write operation**

## 33.3.15 Example of SD_CMD Register Setting

Table 33.8 and Table 33.9 list the SD_CMD register setting.

**Table 33.8 Example SD_CMD register settings for SD (1 of 2)**

| Type | Command | Example SD_CMD register setting | Remark |
|---|---|---|---|
| CMD | CMD0 | 0x0000_0000 | — |
| | CMD2 | 0x0000_0002 | — |
| | CMD3 | 0x0000_0003 | — |
| | CMD4 | 0x0000_0004 | — |
| | CMD5 | 0x0000_0705 or 0x0000_0005 | — |
| | CMD6 | 0x0000_1C06 or 0x0000_0006 | — |
| | CMD7 | 0x0000_0007 | When the card is placed in the deselected state, the response timeout flag sets because there is no response. |
| | CMD8 | 0x0000_0408 or 0x0000_0008 | — |
| | CMD9 | 0x0000_0009 | — |
| | CMD10 | 0x0000_000A | — |
| | CMD11 | 0x0000_040B or 0x0000_000B | — |
| | CMD12 | 0x0000_000C | — |
| | CMD13 | 0x0000_000D | — |
| | CMD15 | 0x0000_000F | — |
| | CMD16 | 0x0000_0010 | — |
| | CMD17 | 0x0000_0011 | — |
| | CMD18 | 0x0000_0012 | With automatic CMD12 |
| | CMD20 | 0x0000_0514 or 0x0000_0014 | — |
| | CMD24 | 0x0000_0018 | — |
| | CMD25 | 0x0000_0019 | With automatic CMD12 |
| | CMD27 | 0x0000_001B | — |
| | CMD28 | 0x0000_001C | — |
| | CMD29 | 0x0000_001D | — |
| | CMD30 | 0x0000_001E | — |
| | CMD32 | 0x0000_0020 | — |
| | CMD33 | 0x0000_0021 | — |
| | CMD38 | 0x0000_0026 | — |
| | CMD42 | 0x0000_002A | — |
| | CMD52 | 0x0000_0434 or 0x0000_0034 | — |
| | CMD53 | 0x0000_1C35 | Single read |
| | | 0x0000_0C35 | Single write |
| | | 0x0000_7C35 | Multiple read |
| | | 0x0000_6C35 | Multiple write |
| | | 0x0000_0035 | The value on the left can be set for both single and multiple operations. However, the CF39 bit in SD_ARG must be set as follows.<br>Read: 0 Write: 1 |
| | CMD55 | 0x0000_0037 | — |
| | CMD56 | 0x0000_0038 | — |

**Table 33.8 Example SD_CMD register settings for SD (2 of 2)**

| Type | Command | Example SD_CMD register setting | Remark |
|---|---|---|---|
| ACMD | ACMD6 | 0x0000_0046 | — |
| | ACMD13 | 0x0000_004D | — |
| | ACMD22 | 0x0000_0056 | — |
| | ACMD23 | 0x0000_0057 | — |
| | ACMD41 | 0x0000_0069 | — |
| | ACMD42 | 0x0000_006A | — |
| | ACMD51 | 0x0000_0073 | — |

**Table 33.9 Example SD_CMD register settings for MMC**

| Type | Command | Example SD_CMD register setting | Remark |
|---|---|---|---|
| CMD | CMD0 | 0x0000_0000 | — |
| | CMD1 | 0x0000_0701 | — |
| | CMD2 | 0x0000_0002 | — |
| | CMD3 | 0x0000_0003 | — |
| | CMD4 | 0x0000_0004 | — |
| | CMD5 | 0x0000_0505 | — |
| | CMD6 | 0x0000_0506 | (with response busy) |
| | | 0x0000_0406 | (without response busy) |
| | CMD7 | 0x0000_0007 | When the card is placed in the deselected state, the response timeout flag sets because there is no response. |
| | CMD8 | 0x0000_1C08 | — |
| | CMD9 | 0x0000_0009 | — |
| | CMD10 | 0x0000_000A | — |
| | CMD12 | 0x0000_000C | — |
| | CMD13 | 0x0000_000D | — |
| | CMD14 | 0x0000_1C0E | Required setting: SD_IFMODE = 0x0000_0100 (CRC check is invalid) |
| | CMD15 | 0x0000_000F | — |
| | CMD16 | 0x0000_0010 | — |
| | CMD17 | 0x0000_0011 | — |
| | CMD18 | 0x0000_7C12 | Pre-defined |
| | CMD19 | 0x0000_0C13 | Required setting: SD_IFMODE = 0x0000_0100 (CRC check is invalid) |
| | CMD21 | 0x0000_1C15 | DDR mode is inhibited |
| | CMD23 | 0x0000_0017 | — |
| | CMD24 | 0x0000_0018 | — |
| | CMD25 | 0x0000_6C19 | Pre-defined |
| | CMD26 | 0x0000_0C1A | — |
| | CMD27 | 0x0000_001B | — |
| | CMD28 | 0x0000_001C | — |
| | CMD29 | 0x0000_001D | — |
| | CMD30 | 0x0000_001E | — |
| | CMD31 | 0x0000_1C1F | — |
| | CMD35 | 0x0000_0423 | — |
| | CMD36 | 0x0000_0424 | — |
| | CMD38 | 0x0000_0026 | — |
| | CMD39 | 0x0000_0427 | — |
| | CMD40 | 0x0000_0428 | — |
| | CMD42 | 0x0000_002A | — |
| | CMD49 | 0x0000_0C31 | — |
| | CMD53 | 0x0000_7C35 | — |
| | CMD54 | 0x0000_6C36 | — |
| | CMD55 | 0x0000_0037 | — |
| | CMD56 | 0x0000_0038 | — |

## 33.4 Usage Notes

### 33.4.1 SD_BUF Illegal Write Access (SD/MMC)

When writing data to SD_BUF0 after the single block write or multi block write command is issued, the data of the size specified in SD_SIZE must be written.

If the data exceeds the size specified in SD_SIZE is written, the ERR4 bit in SD_INFO2 is set to 1. In addition, the data written to SD_BUF0 might not be transmitted and the SD_CLK_CTRLEN bit in SD_INFO2 is held at the value of 0. If this occurs, clearing the SDRST bit in SOFT_RST to 0 and then restoring its value to 1 clears the SD_CLK_CTRLEN bit to 1.

However, this does not apply to the single byte or three bytes when the SD_SIZE setting is odd, or to the fraction of bytes when the SD_SIZE setting is even (the 2 bytes that are not in a 4-byte unit), because the portion of dummy data writing is regarded as excess data and ignored.

### 33.4.2 Block Number Constraint for Multiple Block Read (SD)

When performing a multiple block read of one or two blocks, depending on the timing with which the SD card response register is read, the response value might not be read properly. To prevent this, do one of the following:

1. When receiving one or two blocks of data, use single block reading.
2. Read the response to CMD18 from SD_RSP54.

#### 33.4.2.1 Mechanism of incorrect reading

Figure 33.24 shows the processing flows of the SDHI (hardware) operation and software operation when a multiple block read is performed on two blocks. As shown in the incorrect operation in Figure 33.24, when an interrupt is generated on reception of the CMD18 response and the timing with which the SD card response register (SD_RSP10) is read by the interrupt is delayed, the data during the CMD12 response reception or the CMD12 response might be read. This problem does not occur for multiple block reads of three or more blocks, because CMD12 is not issued until the block of data is read. The problem also does not occur for multiple block writes, because the CMD25 response is read before the block of data is sent.

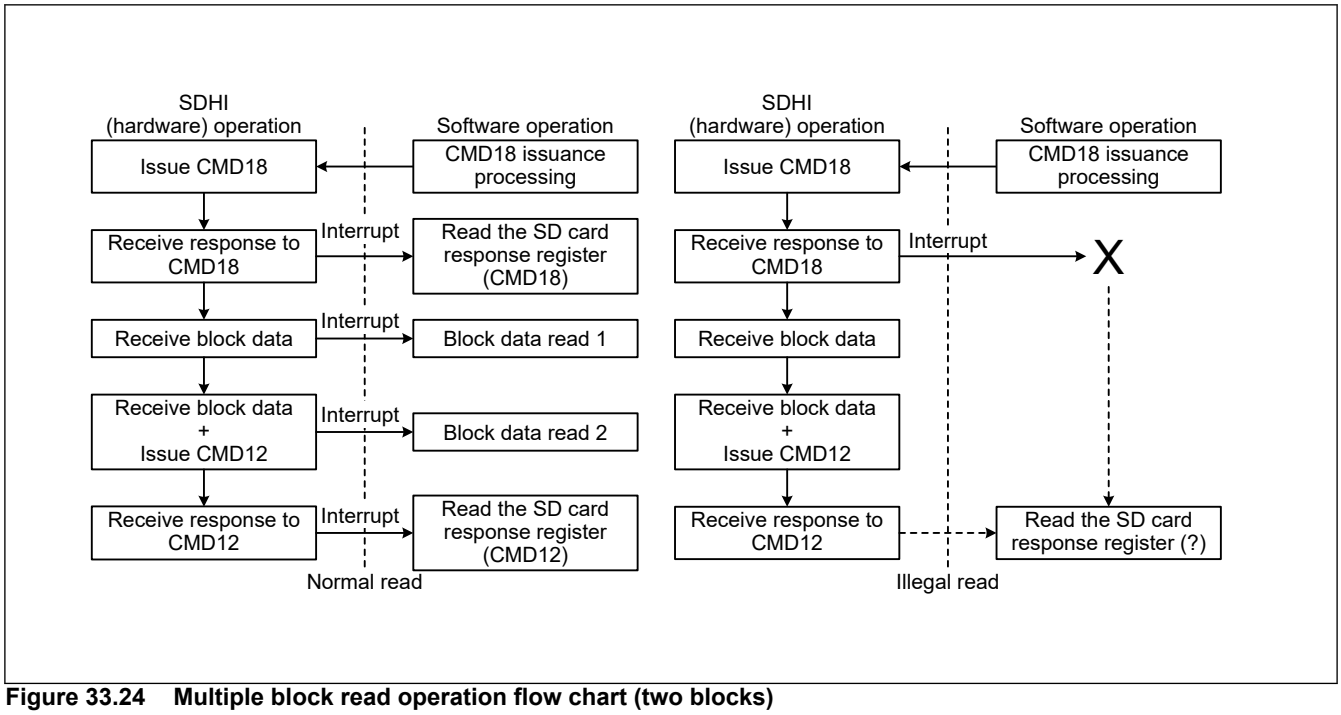



**Figure 33.24 Multiple block read operation flow chart (two blocks)**

### 33.4.3 Automatic Control of SD/MMC Clock Output (SD/MMC)

In the SD Card/MMC standard, 74 cycles of SD/MMC clock must be output before initialization of the card. For this reason, use automatic control of SD/MMC clock output after 74 cycles of SD/MMC clock are output. In addition, if

automatic control of SD/MMC clock output was in use, SD/MMC clock output is stopped on completion of the sequence for a communications error or timeout. When state transitions within the SD card/MMC are necessary after completion of the sequence, release automatic control of SD/MMC clock output and restart supply of the SD/MMC clock to the SD card/MMC.

### 33.4.4 Control of the C52PUB Setting for Multiple Block Write (SD)

If the C52PUB bit in SDIO_MODE is set to 1 during a sequence of multiple block write because of CMD53, CMD52 is not issued until SD_BUF becomes empty. For this reason, set the C52PUB bit after suspending writing to SD_BUF by using one of the following procedures, as appropriate:

**(a) When DMA transfer is not in use**

1. Before setting the C52PUB bit, suspend writing to SD_BUF by making the setting in SD_INFO2 to disable BWE interrupts.
2. Set the C52PUB bit in SDIO_MODE to 1 (so that CMD52 is issued when SD_BUF becomes empty).
3. After the RSPEND interrupt processing in SD_INFO1 because the issuing of CMD52 is completed, restart writing to SD_BUF by making the setting in SD_INFO2 to enable BWE interrupts.

**(b) When DMA transfer is in use**

1. Every time DMA transfer of the value set in SD_SIZE × n blocks (where n = 1, 2,…) proceeds, suspend writing to SD_BUF by DMA transfer before the C52PUB bit is set.
2. Set the C52PUB bit in SDIO_MODE to 1 (so that CMD52 is issued when SD_BUF becomes empty).
3. After the RSPEND interrupt processing in SD_INFO1 because the issuing of CMD52 is completed, restart writing to SD_BUF by DMA transfer.

### 33.4.5 Notes on SD_CLK_CTRL Register Settings (SD/MMC)

When the SD_CLK_CTRLEN bit in SD_INFO2 is 0, SD_CLK_CTRL cannot be written to. Before writing to SD_CLK_CTRL, you must check that the SD_CLK_CTRLEN bit in SD_INFO2 is 1.

### 33.4.6 Specification Limitations

1. The Suspend/Resume operation of the SDIO is not supported.
2. The SPI bus is not supported. (SD/MMC)
3. The shared bus and 8-bit SD bus of the embedded SDIO are not supported.
4. Stream transfer of MMC is not supported.
5. High Priority Interrupt (HPI) of MMC is not supported.
6. Boot Operation/Alternative Boot Operation of MMC is not supported.
7. Open-ended multiple block transfer of MMC is not supported.

### 33.4.7 STP Bit Setting during Multiple Block Read (SD/MMC)

During execution of multiple block read with automatic CMD12 execution by setting the SEC bit in SD_STOP to 1, even if the STP bit in SD_STOP is set to 1 to forcibly stop the execution, the command sequence might not stop depending on the timing of setting the STP bit.

To avoid this, when setting the STP bit in SD_STOP to 1 during multiple block transfer, clear the SEC bit in SD_STOP to 0 at the same time. (Even when the SD_CLK_CTRLEN bit in SD_INFO2 is 0, change the SEC bit from 1 to 0.)

When the command sequence does not stop because the SEC bit was not cleared to 0, the command sequence can be stopped by clearing the SDRST bit in SOFT_RST to 0.

When forcibly terminating the CMD53 multiple block transfer through the IOABT bit in SDIO_MODE, you must leave the SEC bit in SD_STOP as 1.

### 33.4.8 Register Setting Notes

1. All registers in section 33.2. Register Descriptions are accessed in 32-bit access-only.
2. When setting registers, set them after the I/O Port Register setting.

# 34. Cyclic Redundancy Check (CRC)

## 34.1 Overview

The Cyclic Redundancy Check (CRC) generates CRC codes to detect errors in the data. The bit order of CRC calculation results can be switched for LSB-first or MSB-first communication. Additionally, various CRC-generation polynomials are available.

Table 34.1 lists the CRC calculator specifications and Figure 34.1 shows a block diagram.

**Table 34.1 CRC calculator specifications**

| Item | Description | |
|---|---|---|
| Data size | 8-bit | 32-bit |
| Data for CRC calculation*1 | CRC code generated for data in 8n-bit units (where n is a natural number) | CRC code generated for data in 32n-bit units (where n is a natural number) |
| CRC processor unit | Operation executed on 8 bits in parallel | Operation executed on 32 bits in parallel |
| CRC generating polynomial | One of three generating polynomials that is selectable:<br>[8-bit CRC]<br>• $X^8 + X^2 + X + 1$ (CRC-8)<br>[16-bit CRC]<br>• $X^{16} + X^{15} + X^2 + 1$ (CRC-16)<br>• $X^{16} + X^{12} + X^5 + 1$ (CRC-CCITT). | One of two generating polynomials that is selectable:<br>[32-bit CRC]<br>• $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$ (CRC-32)<br>• $X^{32} + X^{28} + X^{27} + X^{26} + X^{25} + X^{23} + X^{22} + X^{20} + X^{19} + X^{18} + X^{14} + X^{13} + X^{11} + X^{10} + X^9 + X^8 + X^6 + 1$ (CRC-32C). |
| CRC calculation switching | The bit order of CRC calculation results can be switched for LSB-first or MSB-first communication | |
| Module-stop function | Module-stop state can be set to reduce power consumption | |
| TrustZone Filter | Security attribution can be set | |

Note 1. This function cannot divide data used in CRC calculations. Write data in 8-bit or 32-bit units.

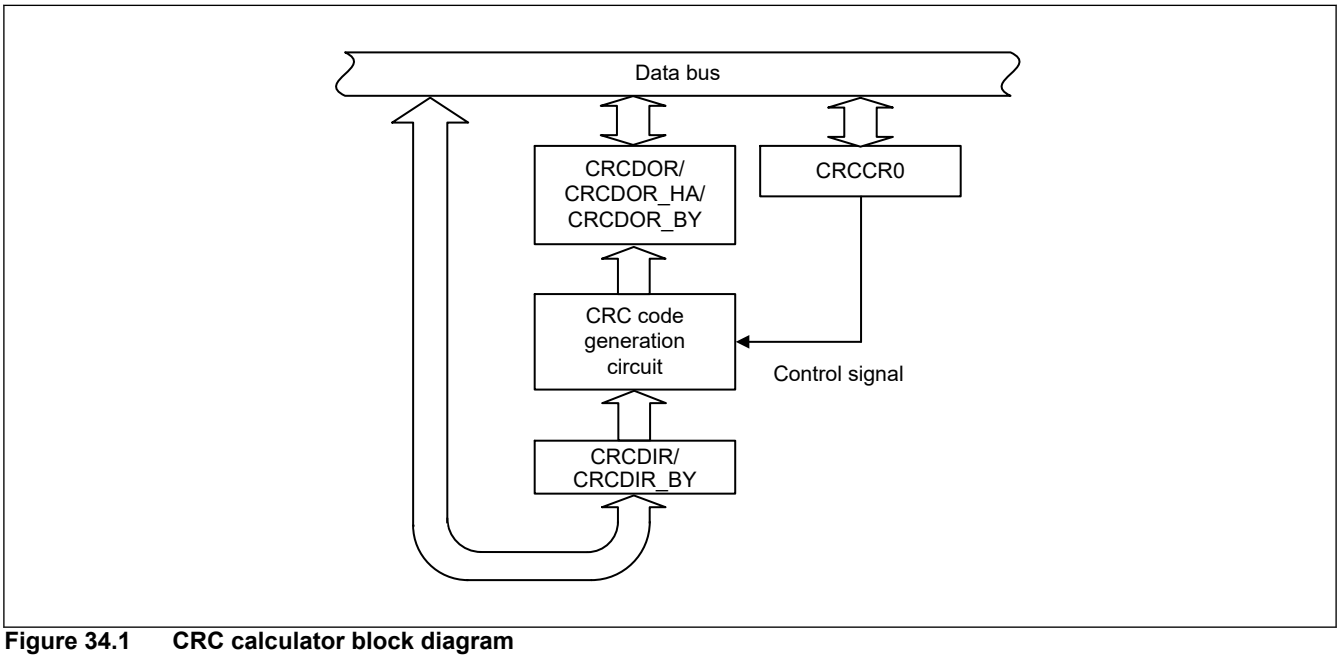



**Figure 34.1 CRC calculator block diagram**

## 34.2 Register Descriptions

### 34.2.1 CRCCR0 : CRC Control Register 0

Base address: CRC = 0x4010_8000

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DORCLR | LMS | — | — | — | GPS[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | GPS[2:0] | CRC Generating Polynomial Switching<br>0 0 1: 8-bit CRC-8 ($X^8 + X^2 + X + 1$)<br>0 1 0: 16-bit CRC-16 ($X^{16} + X^{15} + X^2 + 1$)<br>0 1 1: 16-bit CRC-CCITT ($X^{16} + X^{12} + X^5 + 1$)<br>1 0 0: 32-bit CRC-32 ($X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$)<br>1 0 1: 32-bit CRC-32C ($X^{32} + X^{28} + X^{27} + X^{26} + X^{25} + X^{23} + X^{22} + X^{20} + X^{19} + X^{18} + X^{14} + X^{13} + X^{11} + X^{10} + X^9 + X^8 + X^6 + 1$)<br>Others: No calculation is executed | R/W |
| 5:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | LMS | CRC Calculation Switching<br>0: Generate CRC code for LSB-first communication<br>1: Generate CRC code for MSB-first communication | R/W |
| 7 | DORCLR | CRCDOR/CRCDOR_HA/CRCDOR_BY Register Clear<br>0: No effect<br>1: Clear the CRCDOR/CRCDOR_HA/CRCDOR_BY register | W |

#### GPS[2:0] bits (CRC Generating Polynomial Switching)

The GPS[2:0] bits select the CRC generating polynomial.

#### LMS bit (CRC Calculation Switching)

The LMS bit selects the bit order of generated CRC code. Transmit the lower byte of the CRC code first for LSB-first communication and the upper byte first for MSB-first communication. For details on transmitting and receiving CRC code, see section 34.3. Operation.

#### DORCLR bit (CRCDOR/CRCDOR_HA/CRCDOR_BY Register Clear)

Write 1 to the DORCLR bit to set the CRCDOR/CRCDOR_HA/CRCDOR_BY register to 0x00000000. This bit is read as 0. Only 1 can be written to it.

### 34.2.2 CRCDIR/CRCDIR_BY : CRC Data Input Register

Base address: CRC = 0x4010_8000

Offset address: 0x04

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | CRC input data<br>The CRCDIR register is a 32-bit read/write register to write data for CRC-32 or CRC-32C calculation. The CRCDIR_BY (CRCDIR[31:24]) is an 8-bit read/write register to write data for CRC-8, CRC-16, or CRC-CCITT calculation. | R/W |

### 34.2.3 CRCDOR/CRCDOR_HA/CRCDOR_BY : CRC Data Output Register

Base address: CRC = 0x4010_8000

Offset address: 0x08

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | n/a | CRC output data<br>The CRCDOR register is a 32-bit read/write register for CRC-32 or CRC-32C calculation. The CRCDOR_HA (CRCDOR[31:16]) register is a 16-bit read/write register for CRC-16 or CRC-CCITT calculation.<br>The CRCDOR_BY (CRCDOR[31:24]) register is an 8-bit read/write register for CRC-8 calculation. Because its initial value is 0x00000000, rewrite the CRCDOR/CRCDOR_HA/CRCDOR_BY register to perform the calculations using a value other than the initial value. Data written to the CRCDIR/CRCDIR_BY register is CRC calculated and the result is stored in the CRCDOR/CRCDOR_HA/CRCDOR_BY register. If the CRC code is calculated following the transferred data and the result is 0x00000000, there is no CRC error. | R/W |

## 34.3 Operation

### 34.3.1 Basic Operation

The CRC calculator generates CRC codes for use in LSB-first or MSB-first transfer.

The following examples show CRC code generation for input data (0xF0) using the 16-bit CRC-CCITT generating polynomial ($X^{16} + X^{12} + X^5 + 1$). In these examples, the value of the CRC Data Output Register (CRCDOR_HA) is cleared before CRC calculation.

When an 8-bit CRC (with the polynomial $X^8 + X^2 + X + 1$) is in use, the valid bits of the CRC code are obtained in CRCDOR_BY. When a 32-bit CRC is in use, the valid bits of the CRC code are obtained in CRCDOR.

Figure 34.2 and Figure 34.3 show the LSB-first and MSB-first data transmission examples respectively. Figure 34.4 and Figure 34.5 show the LSB-first and MSB-first data reception examples.

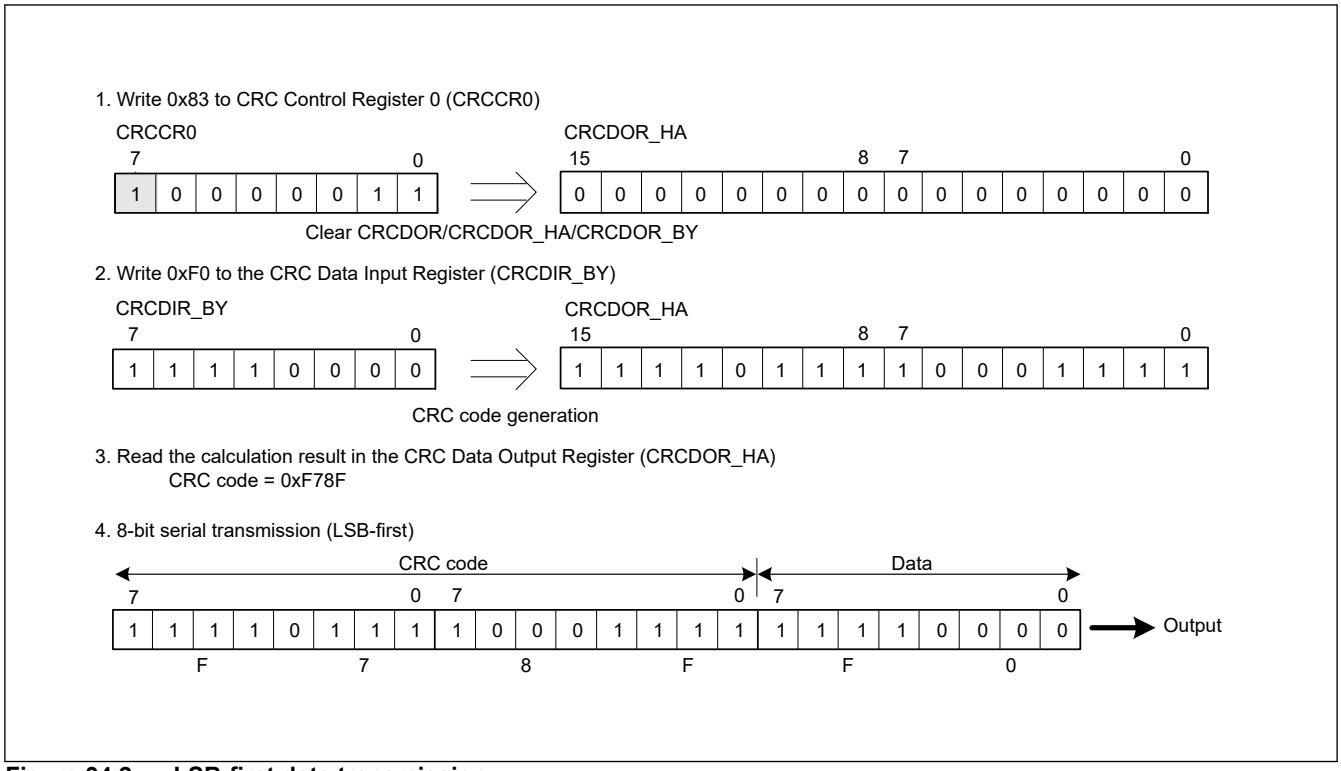



**Figure 34.2 LSB-first data transmission**

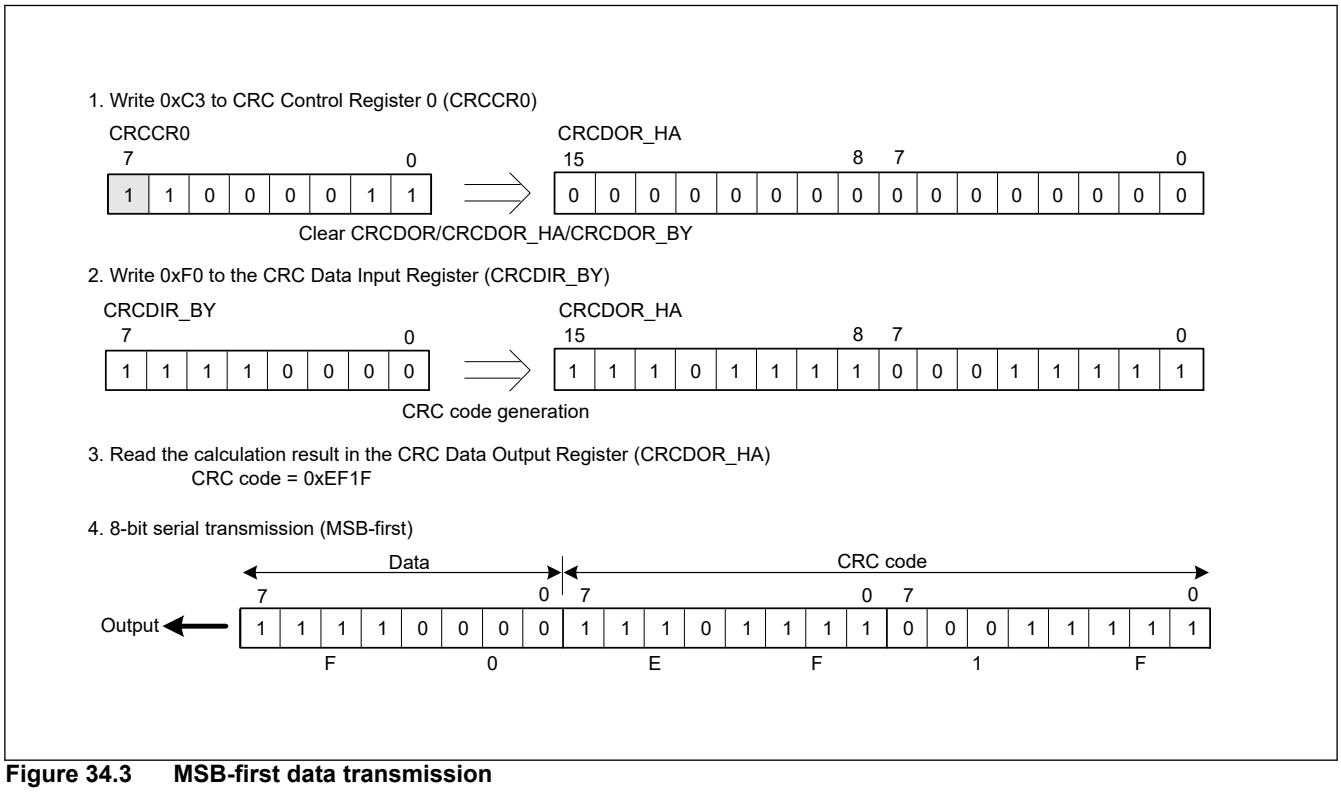



**Figure 34.3 MSB-first data transmission**

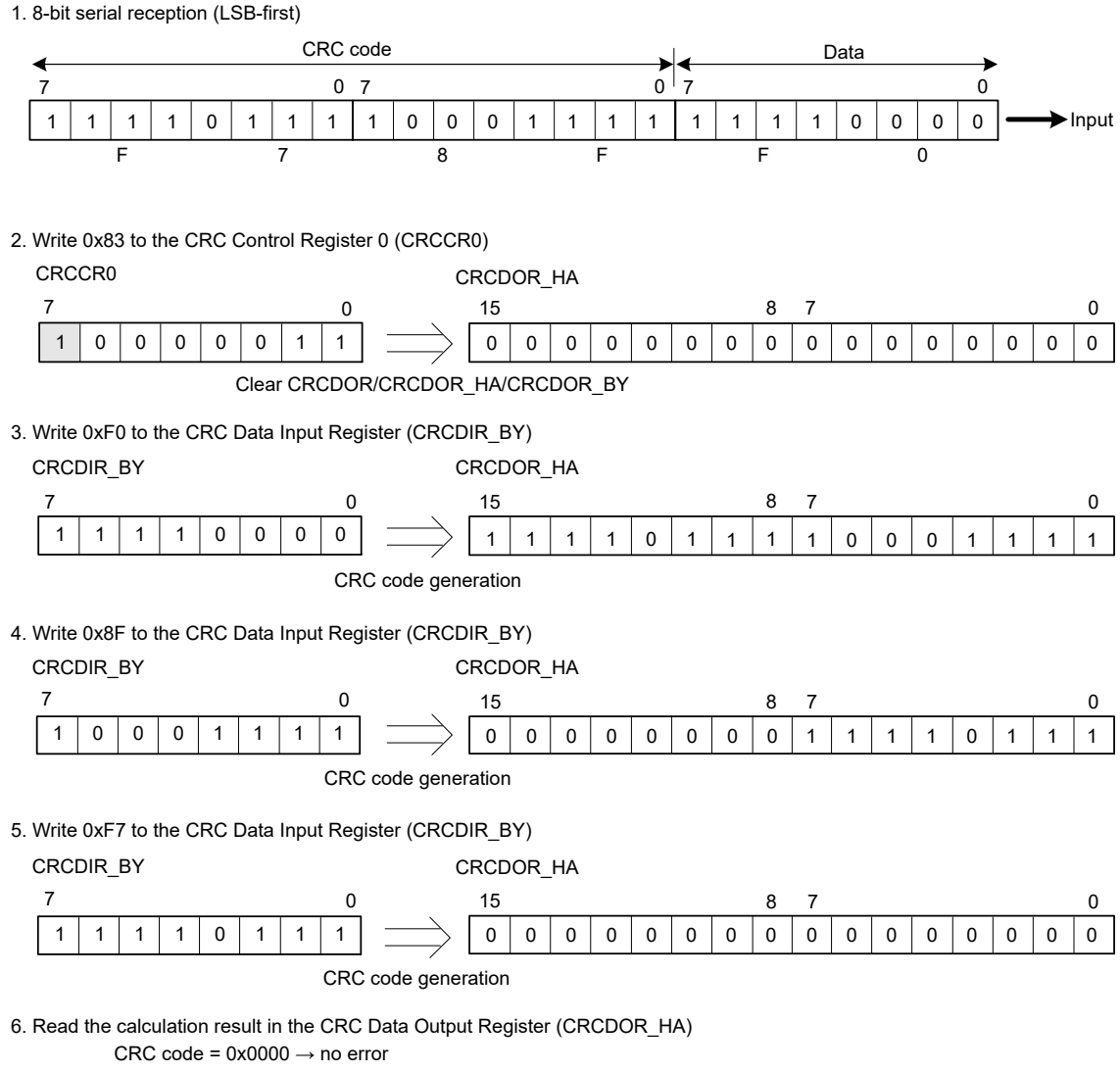



**Figure 34.4 LSB-first data reception**

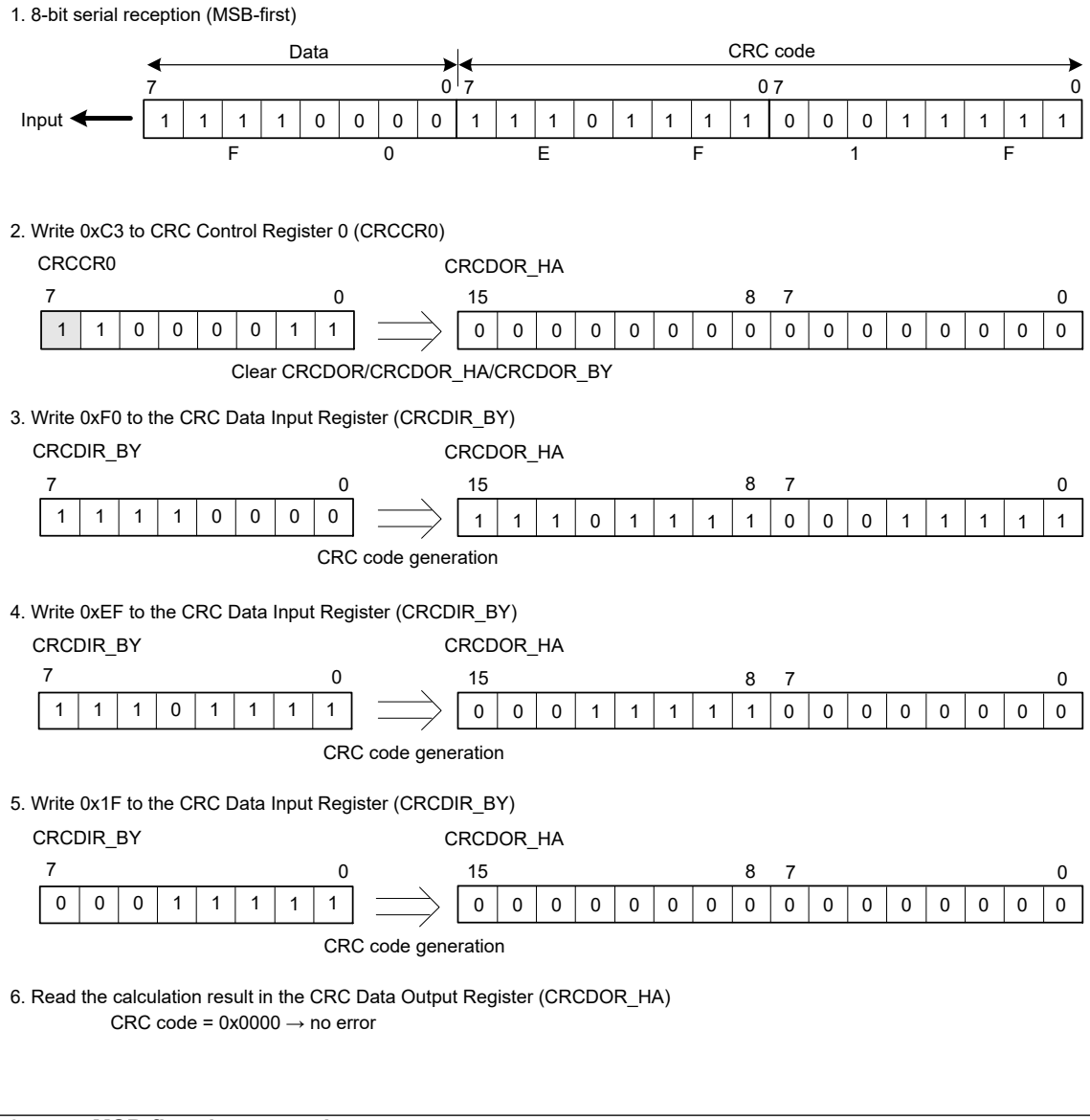



**Figure 34.5 MSB-first data reception**

## 34.4 Usage Notes

### 34.4.1 Settings for the Module-Stop State

The Module Stop Control Register C (MSTPCRC) can enable or disable CRC calculator operation. The CRC calculator is initially stopped after a reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

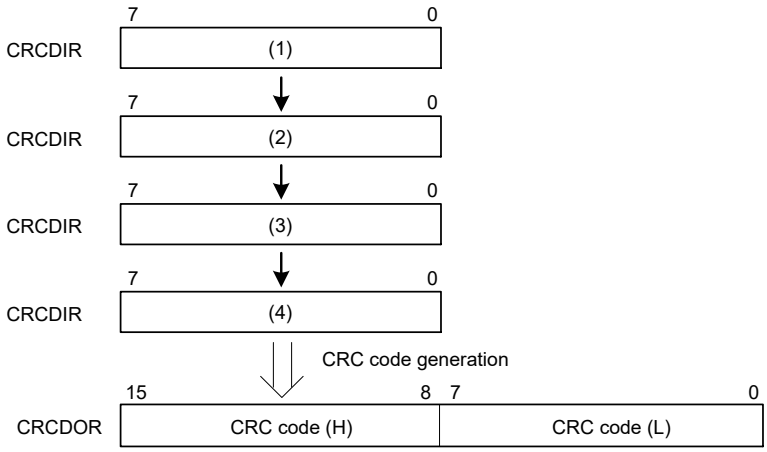
### 34.4.2 Note on Transmission

The transmission sequence for the CRC code differs based on whether the transmission is LSB-first or MSB-first. Figure 34.6 shows an LSB-first and MSB-first data transmission.

When transmitting 32-bit data (for operation executed on 8 bits in parallel)

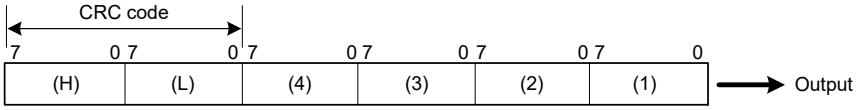
1. CRC code

After specifying the method for generation calculation, write data to CRCDIR in order of (1), (2), (3), and (4).




2. Transmit data

(i) When transmission is LSB-first




(ii) When transmission is MSB-first

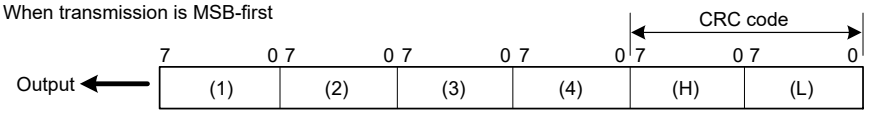



**Figure 34.6 LSB-first and MSB-first data transmission**

# 35. Boundary Scan

## 35.1 Overview

The boundary scan function provides a serial I/O interface based on the JTAG (Joint Test Action Group), IEEE Std.1149.1, and IEEE Standard Test Access Port and Boundary Scan Architecture. Table 35.1 lists the boundary scan specifications, Figure 35.1 shows a block diagram, and Table 35.2 lists the I/O pins.

**Table 35.1 Boundary scan specifications**

| Parameter | Specifications |
|---|---|
| Execution condition | Boundary scan must be executed when the RES pin is driven low. |
| Test modes | • BYPASS mode<br>• EXTEST mode<br>• SAMPLE/PRELOAD mode<br>• CLAMP mode<br>• HIGHZ mode<br>• IDCODE mode |

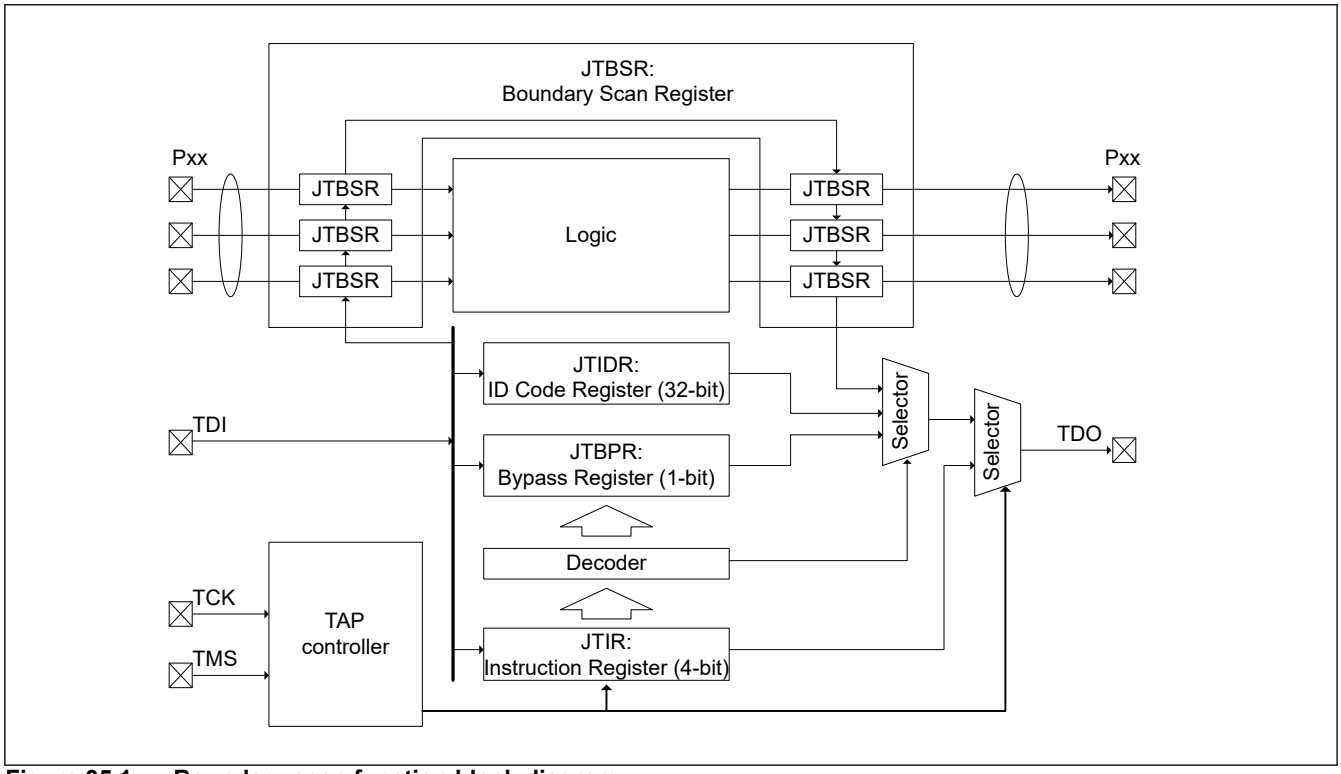



**Figure 35.1 Boundary scan function block diagram**

**Table 35.2 Boundary scan I/O pins**

| Pin name | I/O | Description |
|---|---|---|
| TCK | Input | Test clock input pin<br>Clock signal for boundary scan. The input clock duty cycle is 50% when the boundary scan function is used. |
| TMS | Input | Test mode select pin |
| TDI | Input | Test data input pin |
| TDO | Output | Test data output pin |

Note: This device does not support the TRST pin for the JTAG interface.

## 35.2 Register Descriptions

Table 35.3 lists the boundary scan registers.

**Table 35.3 Boundary scan registers**

| Register name | Symbol | Value after reset |
|---|---|---|
| Instruction Register | JTIR | 0xE |
| ID Code Register | JTIDR | 0x0840_E447 |
| Bypass Register | JTBPR | Undefined |
| Boundary Scan Register | JTBSR | Undefined |

Usage notes for the boundary scan registers:

- Instructions can be input to the Instruction Register (JTIR) through the TDI pin by serial transfer.
- The Bypass Register (JTBPR), which is a 1-bit register, is connected between the TDI and TDO pins in BYPASS mode.
- The Boundary Scan Register (JTBSR), which is configured according to the BSDL description, is connected between the TDI and TDO pins when test data is being shifted in.

Table 35.4 shows the availability of serial transfer for the registers.

**Table 35.4 Serial transfer for registers**

| Register name | Serial input | Serial output |
|---|---|---|
| Instruction Register (JTIR) | Available | Available |
| ID Code Register (JTIDR) | Available | Available |
| Bypass Register (JTBPR) | Available | Available |
| Boundary Scan Register (JTBSR) | Available | Available |

## 35.2.1 JTIR : Instruction Register

| Bit position: | 3 | 2 | 1 | 0 |
|---|---|---|---|---|
| Bit field: | TS[3:0] | | | |
| Value after reset: | 1 | 1 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | TS[3:0] | Test Bit Set<br>The command configuration for these bits | — |

| TS[3:0] | Instruction |
|---|---|
| 0x0 | EXTEST |
| 0x1 | SAMPLE/PRELOAD |
| 0x3 | IDCODE (Renesas code) |
| 0x5 | CLAMP |
| 0x6 | HIGHZ |
| 0xF | BYPASS |
| Others | Reserved |

JTAG instructions can be transferred to the JTIR register by serial input from the TDI pin. The JTIR register is initialized when a power-on reset occurs, or when the TAP controller is in the Test-Logic-Reset state.

## 35.2.2 JTIDR : ID Code Register

| Bit position: | 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DID[31:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | DID[31:0] | Device ID<br>These bits store the fixed value that indicates the device IDCODE (0x0840_E447). | — |

The JTIDR register data is output from the TDO pin when the IDCODE instruction is executed. After a reset release, the DID[31:0] of JTIDR changes into the Arm® debug code. See the *Arm® CoreSight™ SoC-400 Technical Reference Manual* (ARM DDI 0480F).

### 35.2.3 JTBPR : Bypass Register

The JTBPR register is a 1-bit register and is connected between the TDI and TDO pins when the JTIR register is set to BYPASS mode. The JTBPR register cannot be read from or written to by the CPU.

### 35.2.4 JTBSR : Boundary Scan Register

The JTBSR register is a shift register for controlling the external input and output pins of this device, and is distributed across the pads. To apply the JTBSR register in boundary-scan testing, issue the EXTEST, SAMPLE/PRELOAD, CLAMP, and HIGHZ instructions. The BSDL file describes the associations between the JTBSR register bits and the pins of this device. The value after reset is undefined.

## 35.3 Operation

During a reset, the JTAG ports, TCK, TMS, TDI, and TDO, are assigned as default pin functions. The TCK, TMS, and TDI pins are pulled up by the pull-up resistors. Boundary scan testing can be executed after the setup time elapses when POR is negated and RES is driven low.

### 35.3.1 TAP Controller

Figure 35.2 shows the state transition diagram of the TAP controller. All transitions are controlled by the TMS signal.

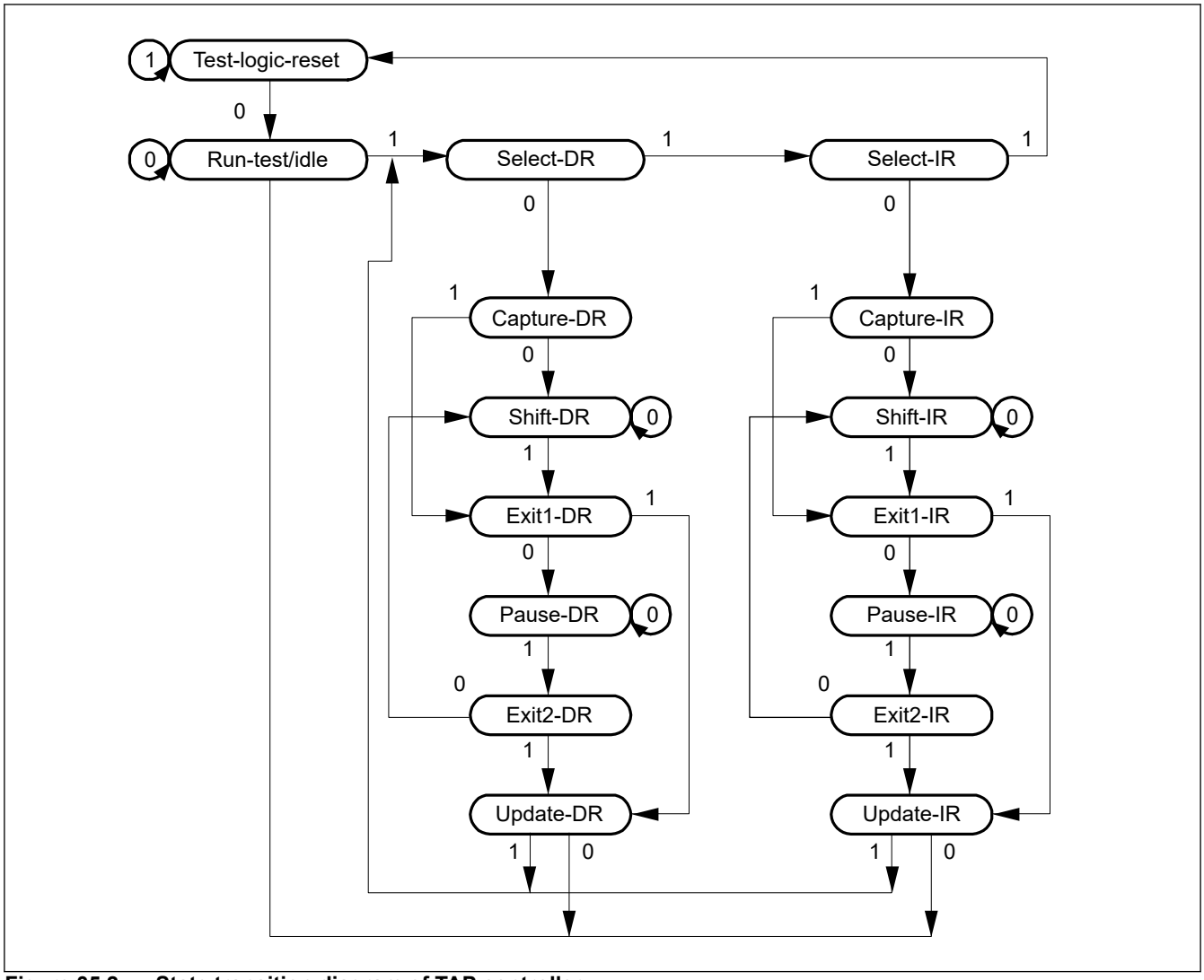



**Figure 35.2 State transition diagram of TAP controller**

## 35.3.2 Commands

### (1) BYPASS

The BYPASS instruction drives the Bypass Register (JTBPR). This instruction shortens the shift path, facilitating the transfer of serial data to other LSIs on a printed circuit board at higher speeds. While this instruction is being executed, the test circuit has no effect on the system circuits.

The JTBPR register is connected between the TDI and TDO pins. Bypass operation is initiated from the Shift-DR operation. The TDO is low in the first clock cycle in the Shift-DR state. In the subsequent clock cycles, values input to the TDI pin are output from the TDO pin.

### (2) EXTEST

The EXTEST instruction is used to test external circuits when this device is installed on the printed circuit board. If this instruction is executed, output pins are used to output test data (specified in the SAMPLE/PRELOAD instruction) from the Boundary Scan Register (JTBSR) to the other devices, and input pins are used to input the test result.

### (3) SAMPLE/PRELOAD

The SAMPLE/PRELOAD instruction is used to input data from the internal circuits of this device to the JTBSR register, output data from the scan path, and reload the data to the scan path. While this instruction is executed, input signals are directly input to this device and output signals are also directly output to the external circuits. This device system circuit is not affected by this instruction.

In SAMPLE operation, the JTBSR register latches a snapshot of the data transferred from the input pins to the internal circuit or data transferred from the internal circuit to the output pins. The latched data is read from the scan path. The JTBSR register latches the data snapshot on the rising edge of the TCK pin in the Capture-DR state. The data snapshot is only transferred from the internal circuit to the output pins during a reset.

In PRELOAD operation, the initial value is written from the scan path to the parallel output latch of the JTBSR register prior to the EXTEST instruction execution. If EXTEST is executed without executing this PRELOAD operation, undefined values are output from the beginning to the end (transfer to the output latch) of the EXTEST sequence. (In the EXTEST instruction, output parallel latches are always output to the output pins.)

#### (4) IDCODE

When the IDCODE instruction is selected, the ID Code Register (JTIDR) value is output to the TDO pin in the Shift-DR state of the TAP controller. In this case, the JTIDR register value is output LSB-first. During this instruction execution, the test circuit does not affect the system circuit.

#### (5) CLAMP

When the CLAMP instruction is selected, output pins output the JTBSR register value that was specified in the SAMPLE/PRELOAD instruction in advance. While the CLAMP instruction is selected, the status of the JTBSR register is maintained regardless of the TAP controller state.

The JTBPR register is connected between the TDI and TDO pins, leading to the same operation as when the BYPASS instruction is selected.

#### (6) HIGHZ

When the HIGHZ instruction is selected, all output pins enter high-impedance state. While the HIGHZ instruction is selected, the JTBSR register is maintained regardless of the state of the TAP controller.

The JTBPR register is connected between the TDI and TDO pins, leading to the same operation as when the BYPASS instruction is selected.

### 35.4 Usage Notes

The boundary scan function is subject to the following constraints:

- The boundary scan must be executed when the RES pin is driven low
- Serial data input/output is in LSB order, as shown in Figure 35.3

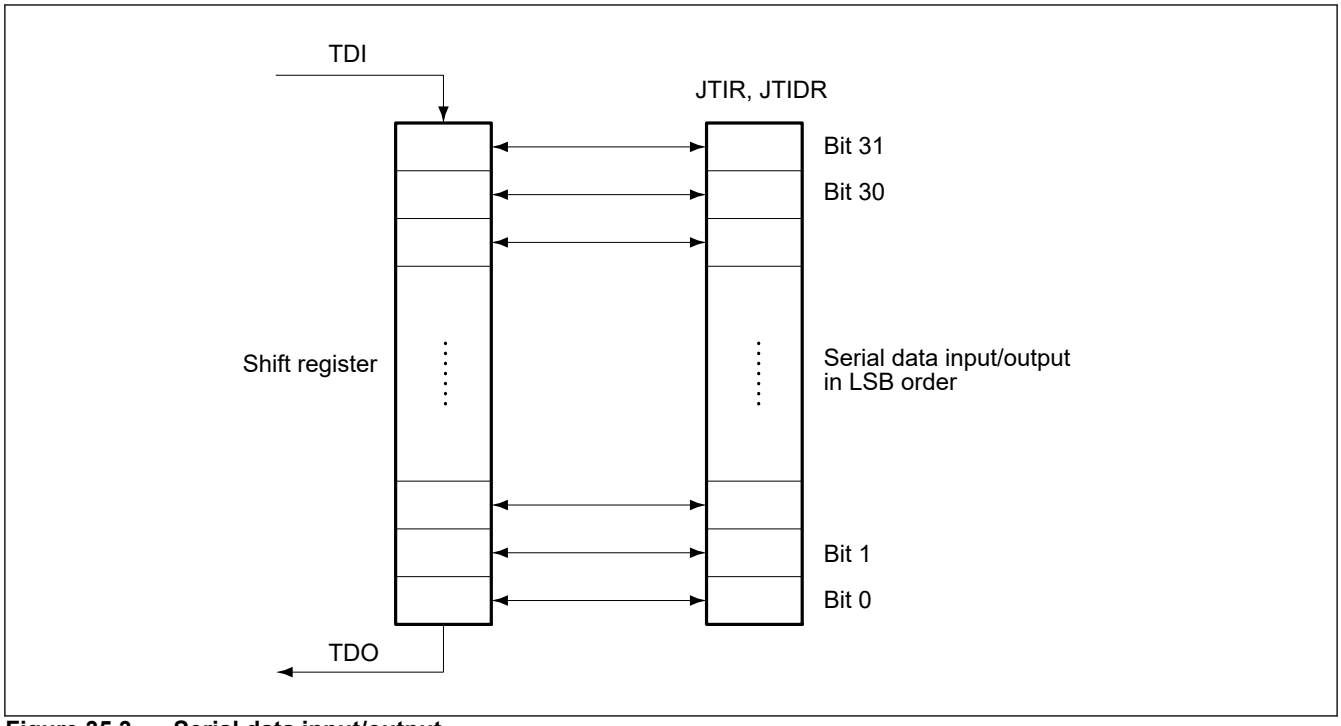



**Figure 35.3 Serial data input/output**

The following pins cannot be boundary-scanned:

- Power supply pins (VCC, VCL, VSS, VBATT, AVCC0, AVSS0, VCC_USB, and VSS_USB)
- Analog reference pins (VREFH0, VREFL0, VREFH, VREFL)
- Clock pins (EXTAL, XTAL, XCIN, and XCOUT)
- Reset pin (RES)
- USB-dedicated pins (USB_DP, USB_DM)
- The boundary-scan pins (TCK, TMS, TDI, and TDO).

# 36. Secure Cryptographic Engine (SCE9)

## 36.1 Overview

The Secure Cryptographic Engine (SCE9) consists of the access management circuit, encryption engine, and random number generation circuit. In combination with the SCE9 library, the SCE9 can prevent eavesdropping (to maintain confidentiality), falsification of information (to ensuret integrity), and impersonation (to verify authenticity).

Because key information required for encryption and decryption is stored only in the SCE9 and all accesses from the outside can be blocked, SCE9 enables building a more robust security system.

Only access control circuit, random number generation circuit, and unique ID are supported. The operation of other circuits is not guaranteed.

Table 36.1 lists the SCE9 specifications. Figure 36.1 shows the SCE9 block diagram.

**Table 36.1 SCE9 specifications**

| Parameter | Specifications |
|---|---|
| Access control | Access management circuit<br>● In case of irregular access to the SCE9 due to a tampered program or CPU runaway, this circuit blocks all subsequent accesses and stops data output from the SCE9 |
| Encryption engine | AES: Compliant with NIST FIPS PUB 197<br>● Key length: 128, 192, or 256 bits<br>● Data block size: 128 bits<br>● Encryption usage modes<br>ECB, CBC, CTR: Compliant with NIST SP 800-38A<br>CMAC: Compliant with NIST SP 800-38B<br>CCM: Compliant with NIST SP 800-38C<br>GCM: Compliant with NIST SP 800-38D<br>XTS: Compliant with NIST SP 800-38E<br>GCTR<br>● Throughput for 128-bit data<br>11 PCLKA cycles for 128-bit key<br>13 PCLKA cycles for 192-bit key<br>15 PCLKA cycles for 256-bit key*1<br>AES-GCM<br>● AES-GCM is realized by combining AES-GCTR and GHASH.<br>Key management<br>● Wrapped keys are only valid within the SCE9 |
| Random number generation | 128-bit true random number generation circuit |
| Signature generation and verification | RSA<br>● Support for 1024-bit, 2048-bit, 3072-bit, and 4096-bit key sizes<br>● Signature generation, signature verification, public-key encryption, private-key decryption<br>DSA<br>● Support for DSA key sizes:<br>– (1024-bit, 160-bit)<br>– (2048-bit, 224-bit)<br>– (2048-bit, 256-bit)<br>● Signature generation, signature verification<br>ECC<br>● Support for curve<br>– NIST P-192, P-224, P-256, and P-384<br>– Brainpool P256r1, P384r1, and P512r1<br>● Signature generation, signature verification |
| Message digest computation | HASH<br>● SHA224 and SHA256 |
| Unique ID | ● A read-only, 128-bit ID unique to an MCU (Unique ID) is accessible from the access management circuit. |
| Low power consumption | Setting of the module-stop state is possible |

Note 1. This does not include the overhead of calling SCE9 library functions.

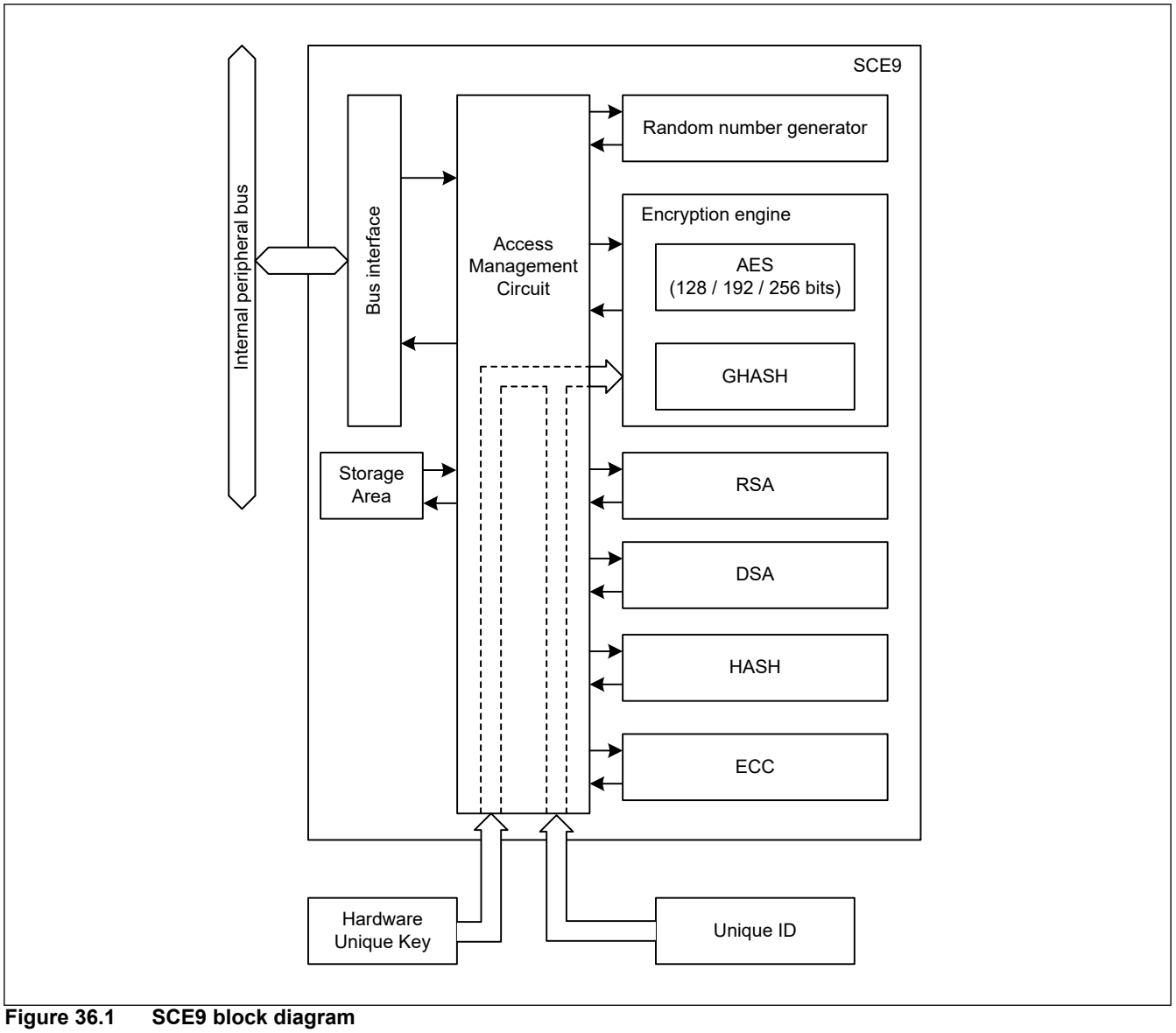



**Figure 36.1 SCE9 block diagram**

## 36.2 Operation

### 36.2.1 Encryption Engine

Figure 36.2 shows conceptual diagram of the encryption engine installed in the SCE9.

The encryption engine uses the key generation information, and converts the plaintext data to ciphertext or ciphertext data to plaintext through the hardware.

The encryption/decryption process can be completed without exposing the key data and the process's intermediate data to the outside of the SCE9.

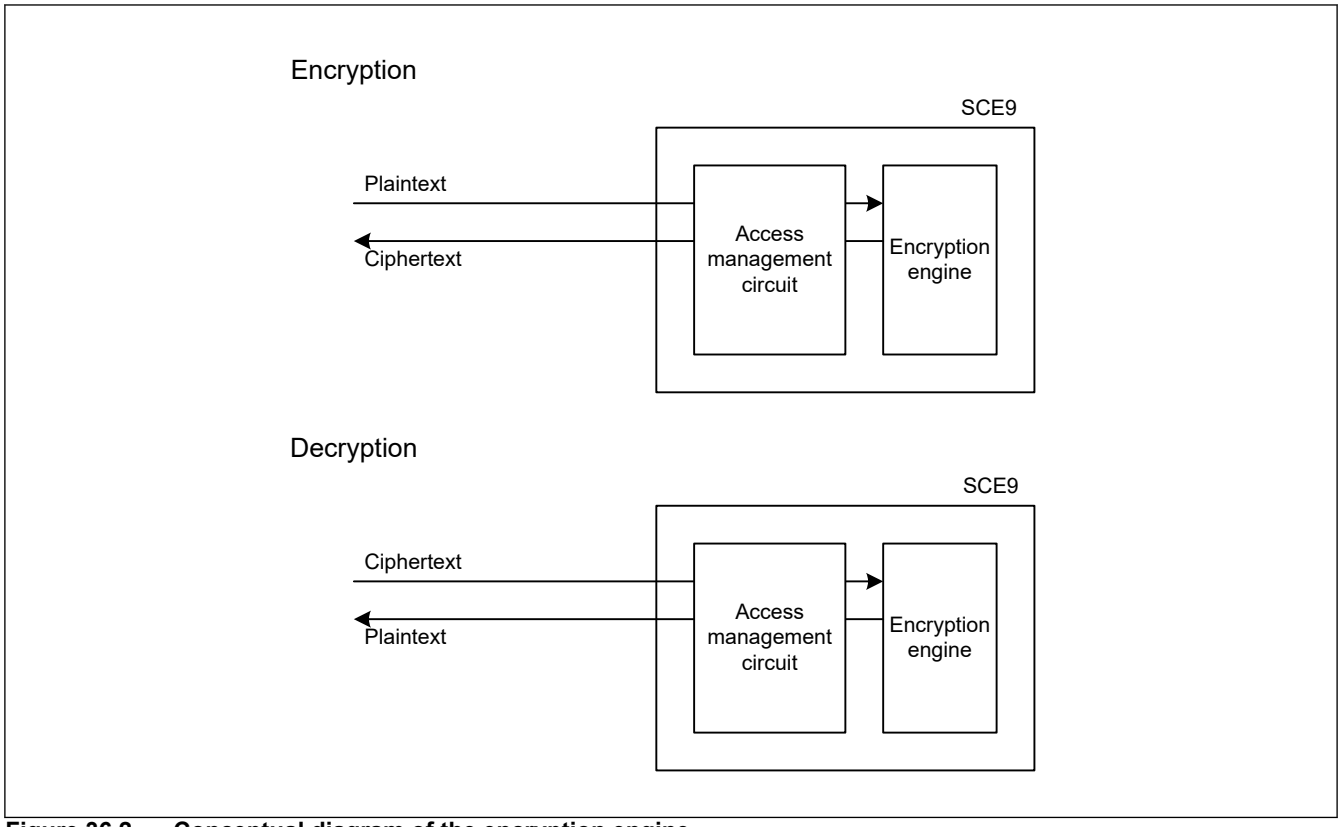



**Figure 36.2 Conceptual diagram of the encryption engine**

## 36.2.2 Encryption and Decryption

Follow the procedure below to encrypt and decrypt the data:

1. Enter the key generation information to the SCE9 and restore the key data.
2. Enter the target data to the SCE9. Plaintext data is converted to ciphertext and ciphertext data to plaintext.
3. Read the converted data.

The encryption engine has input and output buffers, and can perform encryption/decryption in parallel with data input/output. Figure 36.3 shows the encryption engine timing.

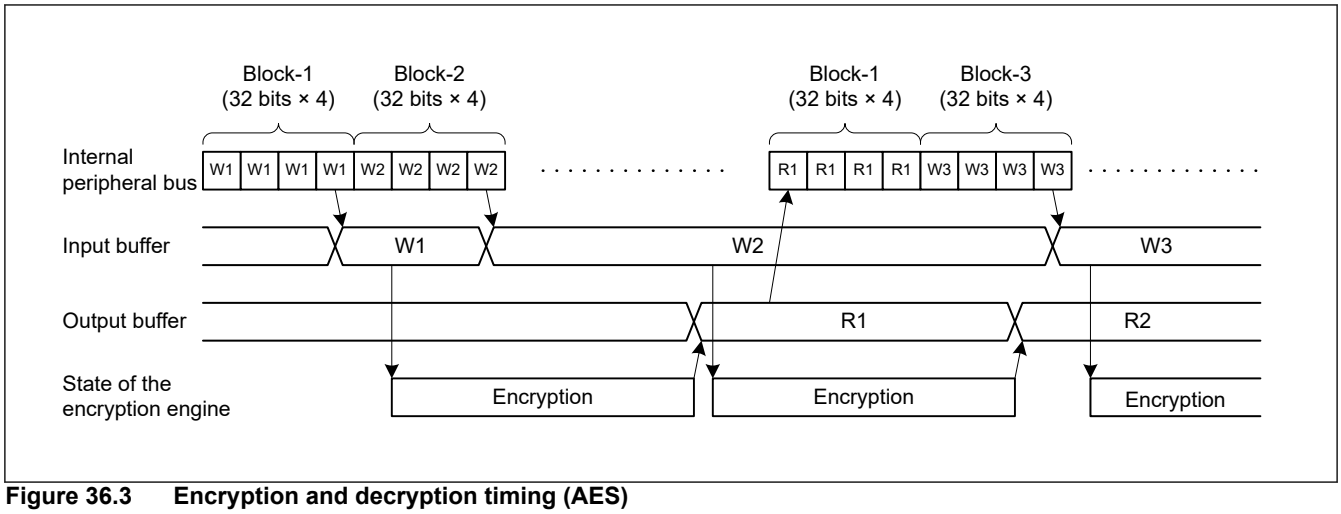



**Figure 36.3 Encryption and decryption timing (AES)**

## 36.3 Usage Notes

### 36.3.1 Software Standby Mode

When Software Standby mode is entered while the encryption engine is in process, proper processing cannot be resumed after cancelling Software Standby mode. Software Standby mode should therefore be entered while the encryption engine is not running.

### 36.3.2 Module-Stop Function Setting

SCE9 operation can be disabled or enabled using Module Stop Control Register C (MSTPCRC). The SCE9 module is initially stopped after reset. Releasing the module-stop state enables access to the registers.

# 37. 12-Bit A/D Converter (ADC12)

## 37.1 Overview

The MCU includes 12-bit successive approximation A/D converter (ADC12) unit. Analog input channels are selectable up to 13. The temperature sensor output and an internal reference voltage are selectable for conversion.

The A/D conversion accuracy is selectable from 12-bit, 10-bit, 8-bit conversion, making it possible to optimize the trade-off between speed and resolution in generating a digital value.

The ADC12 supports the following operating modes:

- Single scan mode to convert analog inputs of selected channels in ascending order of channel number
- Continuous scan mode to convert analog inputs of selected channels continuously in ascending order of channel number
- Group scan mode to divide analog inputs of channels into two groups (group A and B) and convert the analog inputs of selected channels for each group in ascending order of channel number.

In group scan mode, select two groups (group A and B). You can individually select the scan start conditions for each group (group A, B) and start scanning of each group at different times. In addition, when group A priority control operation is set, the ADC12 accepts group A scan start during group B A/D conversion, suspending group B conversion. This allows you to assign higher priority to A/D conversion start for group A.

In double trigger mode, the analog input of a selected channel is converted in single scan mode or group scan mode (group A), and data converted by the first and second A/D conversion start triggers are stored in different registers, providing duplexing of A/D converted data.

Self-diagnosis is performed once at the beginning of each scan, and one of the three reference voltage values generated in ADC12 is A/D converted.

The temperature sensor output and the internal reference voltage is selectable at the same time as the analog input of the channel. First A/D conversion is performed for the analog input of the channel, next the temperature sensor output and then for the internal reference voltage.

The ADC12 also provides a compare function (window A and window B). The compare function specifies the upper reference value for window A and lower reference value for window B, and outputs an interrupt when the A/D converted value of the selected channel meets the comparison conditions.

The A/D data storage buffer is a ring buffer consisting of 16 buffers to sequentially store A/D converted data.

Table 37.1lists the ADC12 specifications and Table 37.2 list the functions. Figure 37.1 shows a block diagram of ADC12 and Table 37.3 lists the I/O pins.

**Table 37.1 ADC12 specifications (1 of 3)**

| Parameter | Specifications |
|---|---|
| Number of units | one unit |
| Input channels | Up to 13 channels (AN000 to AN008, AN011 to AN013, AN016) Extended |
| Analog function | Temperature sensor output, , internal reference voltage |
| Conversion method | Successive approximation method |
| Resolution | 12-bit, 10-bit, 8-bit |
| Conversion time | 0.4 µs/channel (when 12-bit A/D conversion clock PCLKC (ADCLK) is operating at 50 MHz) |
| A/D conversion clock | Peripheral module clock PCLKA and A/D conversion clock PCLKC (ADCLK) can be set with the following division ratios: PCLKA to PCLKC (ADCLK) frequency ratio = 1:1, 2:1, 4:1, 8:1, 1:2, 1:4 |

**Table 37.1 ADC12 specifications (2 of 3)**

| Parameter | Specifications |
|---|---|
| Data registers[*1] | ● 13 registers for analog input<br>● One register for A/D-converted data duplication in double trigger mode<br>● Two registers for A/D-converted data duplication during extended operation in double trigger mode<br>● One register for temperature sensor output<br>● One register for internal reference voltage<br>● One register for self-diagnosis<br>● A/D conversion results are stored in A/D data registers<br>● 12-bit, 10-bit, 8-bit accuracy for A/D conversion results<br>● A/D-converted value addition mode, in which the sum of all A/D-converted results is stored in the A/D data registers as a value with the conversion accuracy bit count + extended bits<br>● Double-trigger mode (selectable in single scan and group scan modes):<br>– The first unit of A/D-converted analog input data on one selected channel is stored in the data register for the channel, and the second unit is stored in the duplication register.<br>● Extended operation in double trigger mode (available for specific triggers):<br>– A/D-converted analog input data on one selected channel is stored in the duplication register provided for the associated trigger. |
| Operating modes[*2] | ● Single scan mode:<br>– A/D conversion is performed only once on the analog inputs of arbitrarily selected channels , on the temperature sensor output, on the internal reference voltage.<br>● Continuous scan mode:<br>A/D conversion is performed repeatedly on the analog inputs of the selected channels on the temperature sensor output, and on the internal reference voltage.<br>● Group scan mode:<br>– Analog inputs of selected channels, the temperature sensor output, and the internal reference voltage are divided into groups A and B. Then A/D conversion of the analog inputs selected on a group basis is performed once.<br>– The scan start conditions can be independently selected for group A, B, allowing A/D conversion of group A, B to be started independently.<br>● Group scan mode (when group priority operation is selected):<br>– If a priority group trigger is input during scanning of a lower-priority group, the scanning of the lower-priority group is stopped and then scanning of the priority group is started. The order of priority is group A > group B.<br>– It is possible to select whether to restart scanning (rescan) of the lower-priority group upon completion of the priority group scan. It is also possible to specify rescanning to be started from the first channel of the selected channels or from the channel for which A/D conversion has not been completed. |
| Conditions for A/D conversion start | ● Software trigger<br>● Synchronous triggers from the Event Link Controller (ELC)<br>● Asynchronous triggering by the external trigger pins, ADTRG0 |
| Functions | ● Variable sampling state count<br>● Self-diagnosis of A/D converter<br>● Selectable A/D-converted value addition mode or average mode<br>● Analog input disconnection detection function (discharge and precharge functions)<br>● Double-trigger mode (duplication of A/D conversion data)<br>● Automatic clear function for A/D data registers<br>● Digital comparison of values in the comparison register and data register, and comparison between values in the data registers<br>● Ring buffer |

**Table 37.1 ADC12 specifications (3 of 3)**

| Parameter | Specifications |
|---|---|
| Interrupt sources | • In single scan mode (double trigger deselected), an A/D scan end interrupt request (ADC120_ADI) and ELC event signal (ADC120_ADI) can be generated on completion of single scan.<br>– A compare interrupt request (ADC120_CMPAI/ADC120_CMPBI) can be generated in response to a match with a digital comparison condition.<br>– A window compare ELC event signal (ADC120_WCMPM) can be generated in response to a match with a digital comparison condition.<br>– A window compare ELC event signal (ADC120_WCMPUM) can be generated in response to a mismatch with a digital comparison condition.<br>• In single scan mode (double trigger selected), an A/D scan end interrupt request (ADC120_ADI) and ELC event signal (ADC120_ADI) is generated on completion of two scans.<br>• In continuous scan mode, an A/D scan end interrupt request (ADC120_ADI) and ELC event signal (ADC120_ADI) is generated on completion of all the selected channel scans.<br>• In group scan mode (double trigger deselected), an A/D scan end interrupt request (ADC120_ADI) and ELC event signal (ADC120_ADI) is generated on completion of group A scan, and an A/D scan end interrupt request for group B (ADC120_GBADI) can be generated on completion of group B scan.<br>• In group scan mode (double trigger selected), an A/D scan end interrupt request (ADC120_ADI) and ELC event signal (ADC120_ADI) is generated on completion of two group A scans, and an A/D scan end interrupt request for group B (ADC120_GBADI) can be generated on completion of group B scan.<br>• ADC120_ADI, ADC120_GBADI, ADC120_WCMPM, and ADC120_WCMPUM can activate the Data Transfer Controller (DTC). |
| ELC interface | • An event is generated upon completion of group A scan in group-scan mode.<br>• An event is generated upon completion of group B scan in group-scan mode.<br>• An event is generated when all scans complete.<br>• Scan can be started by a trigger from the ELC.<br>• An event is generated according to conditions of the compare function window in single-scan mode. |
| Reference voltage | • VREFH0 is the analog reference voltage.<br>• VREFL0 is the analog reference ground. |
| Module-stop function | Module-stop state can be set to reduce power consumption.*3 |
| TrustZone Filter | Security attribution can be set |

Note 1. Changing the A/D conversion accuracy also changes the A/D conversion time. For details, see section 37.3.6. Analog Input Sampling and Scan Conversion Time.
Note 2. When selecting the temperature sensor output and the internal reference voltage, do not use continuous scan mode or group scan mode.
Note 3. For details, see section 10, Low Power Modes.

**Table 37.2 ADC12 functions**

| Parameter | | | function |
|---|---|---|---|
| Analog input channel | | | AN000 to AN008, AN011 to AN013, AN016<br>Internal reference voltage<br>Temperature sensor output |
| Conditions for A/D conversion start | Software | Software trigger | Enabled |
| | Asynchronous trigger (external trigger) | Trigger input pin | ADTRG0 |
| | Synchronous trigger (trigger from ELC) | ELC trigger | ELC_AD00, ELC_AD01 |
| Interrupt | | | ADC120_ADI<br>ADC120_GBADI<br>ADC120_CMPAI<br>ADC120_CMPBI |
| Output to ELC | | | ADC120_ADI<br>ADC120_WCMPM<br>ADC120_WCMPUM |
| Module-stop function settings*1 *2 | | | MSTPCRD.MSTPD16 bit |

Note 1. For details, see section 10, Low Power Modes.
Note 2. Wait 1 µs or longer to start A/D conversion after release from the module-stop state.

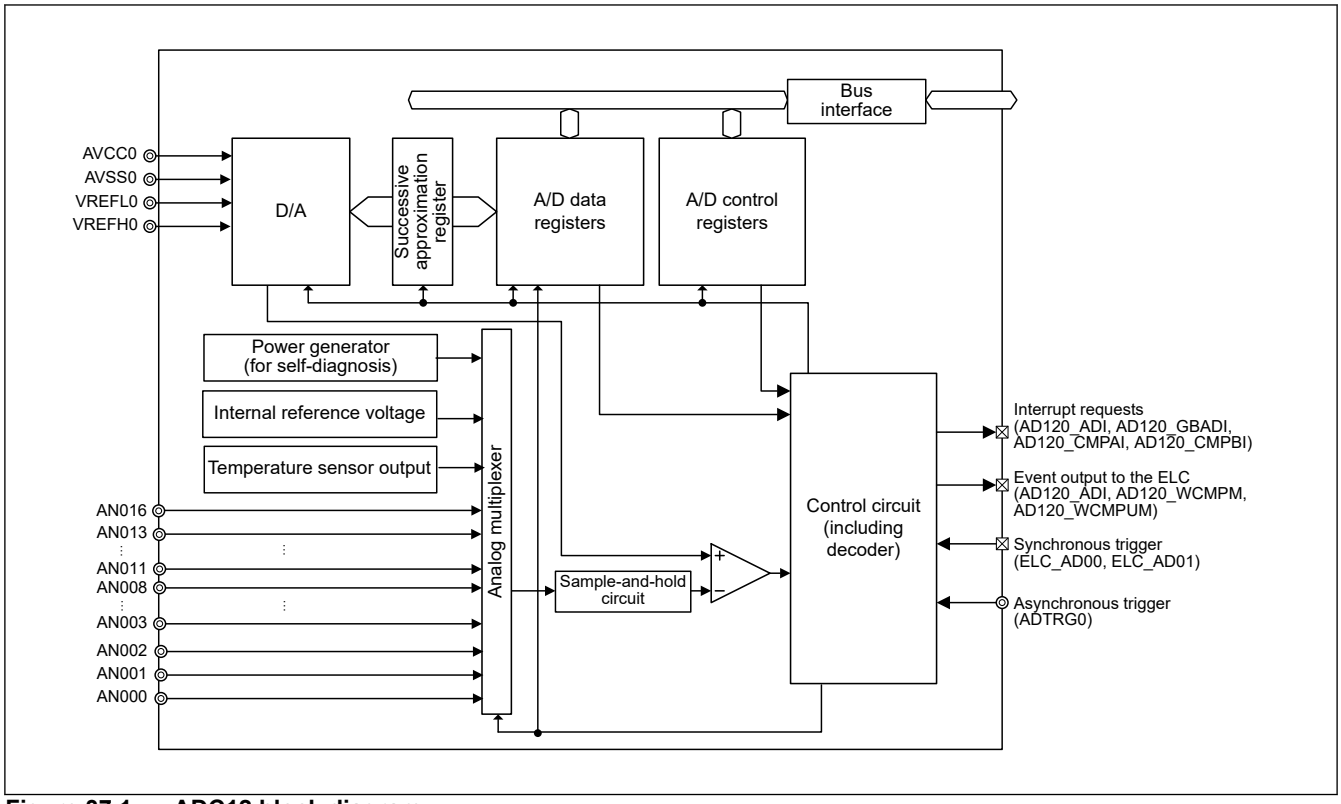



**Figure 37.1 ADC12 block diagram**

Table 37.3 lists the ADC12 I/O pins.

**Table 37.3 ADC12 I/O pins**

| Pin name | I/O | Function |
|---|---|---|
| AVCC0 | Input | Analog block power supply pin<br>(Connect to VCC when ADC12/DAC12 is not used.) |
| AVSS0 | Input | Analog block power supply ground pin<br>(Connect to VSS when ADC12/DAC12 is not used.) |
| VREFH0 | Input | Analog reference voltage supply pin |
| VREFL0 | Input | Analog reference ground pin |
| AN000 to AN008, AN011 to AN013, AN016 | Input | Analog input pins 0 to 8, 11 to 13, 16 |
| ADTRG0 | Input | External trigger input pin for starting A/D conversion |

## 37.2 Register Descriptions

### 37.2.1 ADDRn : A/D Data Registers n (n = 0 to 8, 11 to 13, 16)

Base address: ADC120 = 0x4017_0000

Offset address: 0x020 + 0x2 × n (n = 0 to 8, 11 to 13, 16)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADDR [15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ADDR [15:0] | Converted Value 15 to 0<br>Functions vary depending on the selected mode and accuracy. See Table 37.4 and Table 37.5. | R |

ADDRn registers are 16-bit read-only registers to store A/D conversion results.

The following conditions determine the formats for data in the A/D data registers:

- Setting of the A/D Data Register Format Select bit (ADCER.ADRFMT) (flush-left or flush-right)
- The setting in the A/D Conversion Accuracy Select bits (ADCER.ADPRC[1:0]) (12-bit, 10-bit, 8-bit is selectable.)
- Setting of the Addition/Average Count Select bits (ADADC.ADC[2:0]) (1, 2, 3, 4, or 16 times)
- Setting of the Average Mode Enable bit (ADADC.AVEE) (addition or average).

This section describes the data formats for these conditions in different modes.

### (1) When A/D-converted value addition/average mode is not selected

Table 37.4 shows the example of bit assignment for 12-bit accuracy.

**Table 37.4 Example of bit assignment for 12-bit accuracy**

| Accuracy | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | These bits are read as 0. | | | | Converted Value 11 to 0:<br>12-bit A/D-converted value | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | Converted Value 11 to 0:<br>12-bit A/D-converted value | | | | | | | | | | | | These bits are read as 0. | | | |

### (2) When A/D-converted value average mode is selected

A/D-converted value average mode can be selected when 2 or 4 times is specified in the A/D-converted value addition mode. When A/D converted value average mode is selected, these registers indicate the mean of A/D-converted values on a specific channel. The value is stored in the A/D data register based on the setting of the A/D Data Register Format Select bit in the same way as for normal A/D conversion.

### (3) When A/D-converted value addition mode is selected

For 12-bit, 10-bit, 8-bit accuracy, 1, 2, 3, or 4 times can be selected in the A/D-converted value addition mode. A/D conversion results are stored in the A/D data register as a 2-bit-extended value of the specified conversion accuracy.

For 12-bit accuracy, 16 times can also be selected in the A/D-converted value addition mode. In A/D-converted value addition mode, these registers indicate the value that is obtained by adding A/D-converted values on a specific channel. A/D conversion results are stored in the A/D data register as a 4-bit-extended value of the specified conversion accuracy.

When A/D-converted value addition mode is selected, the value is stored in the A/D data register based on the settings of the A/D Data Register Format Select bits.

Table 37.5 shows example of the bit assignment for 12-bit accuracy.

**Table 37.5 Example of bit assignment for 12-bit accuracy when A/D-converted value addition mode is selected (1 of 2)**

| Accuracy | | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | When 16 conversion times is specified | Added Value 15 to 0:<br>16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 1, 2, 3, or 4 conversion times is specified | These bits are read as 0. | Added Value 13 to 0:<br>14-bit sum of A/D conversion results | | | | | | | | | | | | | | |

**Table 37.5 Example of bit assignment for 12-bit accuracy when A/D-converted value addition mode is selected (2 of 2)**

| Accuracy | | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Left-justified data with 12-bit accuracy | When 1, 2, 3, or 4 conversion times is specified | Added Value 15 to 0: 16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 16 conversion times is specified | Added Value 13 to 0: 14-bit sum of A/D conversion results | | | | | | | | | | | | | | These bits are read as 0. | |

## 37.2.2 ADDBLDR : A/D Data Duplexing Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x018

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADDBLDR [15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ADDBLDR [15:0] | Converted Value 15 to 0<br>Functions vary depending on the selected mode and accuracy. See Table 37.6 and Table 37.7. | R |

ADDBLDR register is a 16-bit read-only register to store A/D conversion results in response to a second trigger in double-trigger mode.

The following conditions determine the formats for data in the A/D data registers:

- Setting of the A/D Data Register Format Select bit (ADCER.ADRFMT) (flush-left or flush-right)
- The setting in the A/D Conversion Accuracy Select bits (ADCER.ADPRC[1:0]) (12-bit, 10-bit, 8-bit is selectable.)
- Setting of the Addition/Average Count Select bits (ADADC.ADC[2:0]) (1, 2, 3, 4, or 16 times)
- Setting of the Average Mode Enable bit (ADADC.AVEE) (addition or average).

This section describes the data formats for these conditions in different modes.

### (1) When A/D-converted value addition/average mode is not selected

Table 37.6 shows the example of bit assignment for 12-bit accuracy.

**Table 37.6 Example of bit assignment for 12-bit accuracy**

| Accuracy | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | These bits are read as 0. | | | | Converted Value 11 to 0: 12-bit A/D-converted value | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | Converted Value 11 to 0: 12-bit A/D-converted value | | | | | | | | | | | | These bits are read as 0. | | | |

### (2) When A/D-converted value average mode is selected

A/D-converted value average mode can be selected when 2 or 4 times is specified in the A/D-converted value addition mode. When A/D converted value average mode is selected, this register indicates the mean of A/D-converted values on a specific channel. The value is stored in the A/D data register based on the setting of the A/D Data Register Format Select bit in the same way as for normal A/D conversion.

### (3) When A/D-converted value addition mode is selected

For 12-bit, 10-bit, 8-bit accuracy, 1, 2, 3, or 4 times can be selected in the A/D-converted value addition mode. A/D conversion results are stored in the A/D data register as a 2-bit-extended value of the specified conversion accuracy.

For 12-bit accuracy, 16 times can also be selected in the A/D-converted value addition mode. In A/D-converted value addition mode, this register indicates the value that is obtained by adding A/D-converted values on a specific channel. A/D conversion results are stored in the A/D data register as a 4-bit-extended value of the specified conversion accuracy.

When A/D-converted value addition mode is selected, the value is stored in the A/D data register based on the settings of the A/D Data Register Format Select bits.

Table 37.7 shows example of the bit assignment for 12-bit accuracy.

**Table 37.7 Example of bit assignment for 12-bit accuracy when A/D-converted value addition mode is selected**

| Accuracy | | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | When 16 conversion times is specified | Added Value 15 to 0:<br>16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 1, 2, 3, or 4 conversion times is specified | These bits are read as 0. | | Added Value 13 to 0:<br>14-bit sum of A/D conversion results | | | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | When 1, 2, 3, or 4 conversion times is specified | Added Value 15 to 0:<br>16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 16 conversion times is specified | Added Value 13 to 0:<br>14-bit sum of A/D conversion results | | | | | | | | | | | | | | These bits are read as 0. | |

## 37.2.3 ADDBLDRn : A/D Data Duplexing Register n (n = A, B)

Base address: ADC120 = 0x4017_0000

Offset address: 0x084 (n = A)
0x086 (n = B)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADDBLDR [15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ADDBLDR [15:0] | Converted Value 15 to 0<br>Functions vary depending on the selected mode and accuracy. See Table 37.8 and Table 37.9. | R |

ADDBLDRn registers are 16-bit read-only registers to store A/D conversion results in response to respective triggers during extended operation in double-trigger mode.

The following conditions determine the formats for data in the A/D data registers:

- Setting of the A/D Data Register Format Select bit (ADCER.ADRFMT) (flush-left or flush-right)
- The setting in the A/D Conversion Accuracy Select bits (ADCER.ADPRC[1:0]) (12-bit, 10-bit, 8-bit is selectable.)
- Setting of the Addition/Average Count Select bits (ADADC.ADC[2:0]) (1, 2, 3, 4, or 16 times)
- Setting of the Average Mode Enable bit (ADADC.AVEE) (addition or average).

This section describes the data formats for these conditions in different modes.

### (1) When A/D-converted value addition/average mode is not selected

Table 37.8 shows the example of bit assignment for 12-bit accuracy.

**Table 37.8 Example of bit assignment for 12-bit accuracy**

| Accuracy | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | These bits are read as 0. | | | | Converted Value 11 to 0: 12-bit A/D-converted value | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | Converted Value 11 to 0: 12-bit A/D-converted value | | | | | | | | | | | | These bits are read as 0. | | | |

### (2) When A/D-converted value average mode is selected

A/D-converted value average mode can be selected when 2 or 4 times is specified in the A/D-converted value addition mode. When A/D converted value average mode is selected, these registers indicate the mean of A/D-converted values on a specific channel. The value is stored in the A/D data register based on the setting of the A/D Data Register Format Select bit in the same way as for normal A/D conversion.

### (3) When A/D-converted value addition mode is selected

For 12-bit, 10-bit, 8-bit accuracy, 1, 2, 3, or 4 times can be selected in the A/D-converted value addition mode. A/D conversion results are stored in the A/D data register as a 2-bit-extended value of the specified conversion accuracy.

For 12-bit accuracy, 16 times can also be selected in the A/D-converted value addition mode. In A/D-converted value addition mode, these registers indicate the value that is obtained by adding A/D-converted values on a specific channel. A/D conversion results are stored in the A/D data register as a 4-bit-extended value of the specified conversion accuracy.

When A/D-converted value addition mode is selected, the value is stored in the A/D data register based on the settings of the A/D Data Register Format Select bits.

Table 37.9 shows example of the bit assignment for 12-bit accuracy.

**Table 37.9 Example of bit assignment for 12-bit accuracy when A/D-converted value addition mode is selected**

| Accuracy | | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | When 16 conversion times is specified | Added Value 15 to 0: 16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 1, 2, 3, or 4 conversion times is specified | These bits are read as 0. | Added Value 13 to 0: 14-bit sum of A/D conversion results | | | | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | When 1, 2, 3, or 4 conversion times is specified | Added Value 15 to 0: 16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 16 conversion times is specified | Added Value 13 to 0: 14-bit sum of A/D conversion results | | | | | | | | | | | | | | These bits are read as 0. | |

## 37.2.4 ADTSDR : A/D Temperature Sensor Data Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x01A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADTSDR [15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ADTSDR [15:0] | Converted Value 15 to 0<br>Functions vary depending on the selected mode and accuracy. See Table 37.10 and Table 37.11. | R |

ADTSDR register is a 16-bit read-only register to store A/D conversion result of the temperature sensor output.

The following conditions determine the formats for data in the A/D data registers:

- Setting of the A/D Data Register Format Select bit (ADCER.ADRFMT) (flush-left or flush-right)
- The setting in the A/D Conversion Accuracy Select bits (ADCER.ADPRC[1:0]) (12-bit, 10-bit, 8-bit is selectable.)
- Setting of the Addition/Average Count Select bits (ADADC.ADC[2:0]) (1, 2, 3, 4, or 16 times)
- Setting of the Average Mode Enable bit (ADADC.AVEE) (addition or average).

This section describes the data formats for these conditions in different modes.

### (1) When A/D-converted value addition/average mode is not selected

Table 37.10 shows the example of bit assignment for 12-bit accuracy.

**Table 37.10 Example of bit assignment for 12-bit accuracy**

| Accuracy | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | These bits are read as 0. | | | | Converted Value 11 to 0:<br>12-bit A/D-converted value | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | Converted Value 11 to 0:<br>12-bit A/D-converted value | | | | | | | | | | | | These bits are read as 0. | | | |

### (2) When A/D-converted value average mode is selected

A/D-converted value average mode can be selected when 2 or 4 times is specified in the A/D-converted value addition mode. When A/D converted value average mode is selected, this register indicates the mean of A/D-converted values on a specific channel. The value is stored in the A/D data register based on the setting of the A/D Data Register Format Select bit in the same way as for normal A/D conversion.

### (3) When A/D-converted value addition mode is selected

For 12-bit, 10-bit, 8-bit accuracy, 1, 2, 3, or 4 times can be selected in the A/D-converted value addition mode. A/D conversion results are stored in the A/D data register as a 2-bit-extended value of the specified conversion accuracy.

For 12-bit accuracy, 16 times can also be selected in the A/D-converted value addition mode. In A/D-converted value addition mode, this register indicates the value that is obtained by adding A/D-converted values on a specific channel. A/D conversion results are stored in the A/D data register as a 4-bit-extended value of the specified conversion accuracy.

When A/D-converted value addition mode is selected, the value is stored in the A/D data register based on the settings of the A/D Data Register Format Select bits.

Table 37.11 shows example of the bit assignment for 12-bit accuracy.

**Table 37.11 Example of bit assignment for 12-bit accuracy when A/D-converted value addition mode is selected**

| Accuracy | | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | When 16 conversion times is specified | Added Value 15 to 0:<br>16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 1, 2, 3, or 4 conversion times is specified | These bits are read as 0. | | Added Value 13 to 0:<br>14-bit sum of A/D conversion results | | | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | When 1, 2, 3, or 4 conversion times is specified | Added Value 15 to 0:<br>16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 16 conversion times is specified | Added Value 13 to 0:<br>14-bit sum of A/D conversion results | | | | | | | | | | | | | | These bits are read as 0. | |

## 37.2.5 ADOCDR : A/D Internal Reference Voltage Data Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x01C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADOCDR [15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ADOCDR [15:0] | Converted Value 15 to 0<br>Functions vary depending on the selected mode and accuracy. See Table 37.12 and Table 37.13. | R |

ADOCDR register is a 16-bit read-only register to store A/D conversion result of the internal reference voltage.

The following conditions determine the formats for data in the A/D data registers:

- Setting of the A/D Data Register Format Select bit (ADCER.ADRFMT) (flush-left or flush-right)
- The setting in the A/D Conversion Accuracy Select bits (ADCER.ADPRC[1:0]) (12-bit, 10-bit, 8-bit is selectable.)
- Setting of the Addition/Average Count Select bits (ADADC.ADC[2:0]) (1, 2, 3, 4, or 16 times)
- Setting of the Average Mode Enable bit (ADADC.AVEE) (addition or average).

This section describes the data formats for these conditions in different modes.

### (1) When A/D-converted value addition/average mode is not selected

Table 37.12 shows the example of bit assignment for 12-bit accuracy.

**Table 37.12 Example of bit assignment for 12-bit accuracy**

| Accuracy | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | These bits are read as 0. | | | | Converted Value 11 to 0:<br>12-bit A/D-converted value | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | Converted Value 11 to 0:<br>12-bit A/D-converted value | | | | | | | | | | | | These bits are read as 0. | | | |

### (2) When A/D-converted value average mode is selected

A/D-converted value average mode can be selected when 2 or 4 times is specified in the A/D-converted value addition mode. When A/D converted value average mode is selected, this register indicates the mean of A/D-converted values on a specific channel. The value is stored in the A/D data register based on the setting of the A/D Data Register Format Select bit in the same way as for normal A/D conversion.

### (3) When A/D-converted value addition mode is selected

For 12-bit, 10-bit, 8-bit accuracy, 1, 2, 3, or 4 times can be selected in the A/D-converted value addition mode. A/D conversion results are stored in the A/D data register as a 2-bit-extended value of the specified conversion accuracy.

For 12-bit accuracy, 16 times can also be selected in the A/D-converted value addition mode. In A/D-converted value addition mode, this register indicates the value that is obtained by adding A/D-converted values on a specific channel. A/D conversion results are stored in the A/D data register as a 4-bit-extended value of the specified conversion accuracy.

When A/D-converted value addition mode is selected, the value is stored in the A/D data register based on the settings of the A/D Data Register Format Select bits.

Table 37.13 shows example of the bit assignment for 12-bit accuracy.

**Table 37.13 Example of bit assignment for 12-bit accuracy when A/D-converted value addition mode is selected**

| Accuracy | | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | When 16 conversion times is specified | Added Value 15 to 0:<br>16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 1, 2, 3, or 4 conversion times is specified | These bits are read as 0. | | Added Value 13 to 0:<br>14-bit sum of A/D conversion results | | | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | When 1, 2, 3, or 4 conversion times is specified | Added Value 15 to 0:<br>16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 16 conversion times is specified | Added Value 13 to 0:<br>14-bit sum of A/D conversion results | | | | | | | | | | | | | | These bits are read as 0. | |

## 37.2.6 ADRD : A/D Self-Diagnosis Data Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x01E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | DIAGST[1:0] | | — | — | AD[11:0] | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 11:0 | AD[11:0] | Converted Value 11 to 0<br>12-bit A/D-converted value | R |
| 13:12 | — | These bits are read as 0. | R |
| 15:14 | DIAGST[1:0] | Self-Diagnosis Status<br>For details on self-diagnosis, see section 37.2.15. ADCER : A/D Control Extended Register.<br>0 0: Self-diagnosis not executed after power-on.<br>0 1: Self-diagnosis was executed using the 0 V voltage.<br>1 0: Self-diagnosis was executed using the reference voltage*1 × 1/2.<br>1 1: Self-diagnosis was executed using the reference voltage *1. | R |

Note: The example of the bit assignment for the right-justified data with 12-bit accuracy is indicated.
Note 1. The reference voltage refers to VREFH0 (Unit 0).

ADRD is a 16-bit read-only register that holds the A/D conversion results based on the self-diagnosis of the ADC12. In addition to the AD[11:0] bits indicating the A/D-converted value, it includes the Self-Diagnosis Status bit (DIAGST[1:0]).

The settings of the A/D data register format and the A/D conversion accuracy determines the formats for data in this register.

The A/D-converted value addition and average modes cannot be applied to the A/D self-diagnosis function. For details on self-diagnosis, see section 37.2.15. ADCER : A/D Control Extended Register.

This section describes the data formats for each condition. The register diagram and the register bit table shown in this section indicate example of the bit assignment for the left and right-justified data with 12-bit accuracy.

**Table 37.14 Bit assignment for each right-justified accuracy**

| Accuracy | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | DIAGST[1:0] | | — | | AD[11:0] | | | | | | | | | | | |

**Table 37.15 Bit assignment for each left-justified accuracy**

| Accuracy | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Left-justified data with 12-bit accuracy | AD[11:0] | | | | | | | | | | | | — | | DIAGST[1:0] | |

## 37.2.7 ADCSR : A/D Control Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x000

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADST | ADCS[1:0] | | — | — | — | TRGE | EXTRG | DBLE | GBADIE | — | DBLANS[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | DBLANS[4:0] | Double Trigger Channel Select<br>These bits select one analog input channel for double-trigger operation. The setting is only valid in double-trigger mode. | R/W |
| 5 | — | This bit is read as 0. The write value should be 0. | R/W |
| 6 | GBADIE | Group B Scan End Interrupt and ELC Event Enable<br>Group B scan only works in group scan mode.<br>0: Disable ADC120_GBADI interrupt generation on group B scan completion.<br>1: Enable ADC120_GBADI interrupt generation on group B scan completion. | R/W |
| 7 | DBLE | Double Trigger Mode Select<br>0: Deselect double-trigger mode.<br>1: Select double-trigger mode. | R/W |
| 8 | EXTRG | Trigger Select*1<br>0: Start A/D conversion by the synchronous trigger (ELC).<br>1: Start A/D conversion by the asynchronous trigger (ADTRG0). | R/W |
| 9 | TRGE | Trigger Start Enable<br>0: Disable A/D conversion to be started by the synchronous or asynchronous trigger<br>1: Enable A/D conversion to be started by the synchronous or asynchronous trigger | R/W |
| 10 | — | These bits are read as 0. The write value should be 0. | R/W |
| 11 | — | These bits are read as 0. The write value should be 0. | R/W |
| 12 | — | These bits are read as 0. The write value should be 0. | R/W |
| 14:13 | ADCS[1:0] | Scan Mode Select<br>0 0: Single scan mode<br>0 1: Group scan mode<br>1 0: Continuous scan mode<br>1 1: Setting prohibited | R/W |
| 15 | ADST | A/D Conversion Start<br>0: Stop A/D conversion process.<br>1: Start A/D conversion process. | R/W |

Note 1. To start A/D conversion using an external pin (asynchronous trigger):
After a high-level signal is input to the external pin (ADTRG0), write 1 to both the TRGE and EXTRG bits in the ADCSR register and drive the ADTRG0 pin low. With these settings, the scan conversion process starts on detection of the falling edge of ADTRG0. The pulse width of the low-level input must be at least PCLKA 1.5 clock cycles.

The ADCSR register sets double-trigger mode and A/D conversion start trigger, enables or disables scan end interrupt, selects the scan mode, and starts or stops A/D conversion.

### DBLANS[4:0] bits (Double Trigger Channel Select)

The DBLANS[4:0] bits select one channel for A/D conversion data duplication in double-trigger mode. This can be selected by setting the binary value of the channel number to be duplicated. The A/D conversion results of the analog input of the

channel selected in the DBLANS[4:0] bits are stored in A/D Data Register y when conversion is started by the first trigger, and in the A/D Data Duplexing Register when conversion is started by the second trigger.

In double-trigger mode, the channels selected in the ADANSA0 and ADANSA1 registers, are invalid, and the channel selected in the DBLANS[4:0] bits is A/D converted instead.

When double-trigger mode is used in group scan mode, double-trigger control is only applied to group A and not to group B. Therefore, multiple channel analog input can be selected for group B even in double-trigger mode.

Only set the DBLANS[4:0] bits when the ADST bit is 0. Do not set the DBLANS[4:0] bits at the same time that you write 1 to the ADST bit.

To enter A/D-converted value addition/average mode when in double-trigger mode, select the channel using the DBLANS[4:0] bits in the ADADS0 and ADADS1 registers.

A/D-converted data from the self-diagnosis function temperature sensor output and internal reference voltage cannot be used in double-trigger mode.

#### GBADIE bit (Group B Scan End Interrupt and ELC Event Enable)

The GBADIE bit enables or disables group B scan end interrupt (ADC120_GBADI) in group scan mode.

#### DBLE bit (Double Trigger Mode Select)

The DBLE bit selects or deselects double-trigger mode. Double-trigger mode can only be operated by the synchronous trigger (ELC) selected in the ADSTRGR.TRSA[5:0] bits.

Double-trigger operation is as follows:

1. The ADC120_ADI interrupt is not output on completion of the first conversion but on completion of the second conversion.
2. The A/D conversion results from the duplication channel (selected in DBLANS[4:0]) started by the first trigger are stored in A/D Data Register y and those started by the second trigger are stored in the A/D Data Duplexing Register.

When the DBLE bit is set (double-trigger mode is selected), the channels specified in the ADANSA0 and ADANSA1 registers are invalid. Double-trigger mode is deselected by setting DBLE to 0. Setting DBLE to 1 again enables the same double-trigger operation described in 1. and 2. for first time scanning with the first trigger.

Do not select double-trigger mode in continuous scan mode. Software triggering cannot be used in double-trigger mode. Always set the ADST bit to 0 before setting the DBLE bit. Do not set the DBLE bit at that same time as writing 1 to the ADST bit.

#### EXTRG bit (Trigger Select)

The EXTRG bit selects the synchronous or asynchronous trigger as the trigger for starting A/D conversion.

In group scan mode, the setting of this bit takes effect on the trigger selected for group A. For group B, A/D conversion is started by the selected synchronous trigger regardless of this bit setting.

#### TRGE bit (Trigger Start Enable)

The TRGE bit enables or disables A/D conversion by the synchronous and asynchronous triggers. In group scan mode, set this bit to 1.

#### ADCS[1:0] bits (Scan Mode Select)

The ADCS[1:0] bits select the scan mode.

In single scan mode, A/D conversion is performed for the analog inputs of the channels selected in the ADANSA0 and ADANSA1 registers, in ascending order of channel number. When 1 cycle of A/D conversion completes for all the selected channels, the scan conversion stops.

In continuous scan mode, when the ADCSR.ADST bit is 1, A/D conversion is performed for the analog inputs of the channels selected with the ADANSA0 and ADANSA1 registers , in ascending order of channel number. When 1 cycle of A/D conversion completes for all the selected channels, A/D conversion repeats from the first channel. If the ADCSR.ADST bit is set to 0 during continuous scan, A/D conversion stops even if scanning is in progress.

In group scan mode:

- Group A scanning is started by the synchronous trigger (ELC) selected in the TRSA[5:0] bits in the ADSTRGR register. A/D conversion is performed on group A analog inputs of the channels selected in the ADANSA0 and ADANSA1 registers, in ascending order of channel number. When 1 cycle of A/D conversion completes for all the selected channels, A/D conversion stops.
- Group B scanning is started by the synchronous trigger (ELC) selected in the ADSTRGR.TRSB[5:0] bits. A/D conversion is performed on group B analog inputs of the channels selected in the ADANSB0 and ADANSB1 registers, in ascending order of channel number. When 1 cycle of A/D conversion completes for all the selected channels, A/D conversion stops.

If the conversion processes in group A and group B occur at the same time, those conversions cannot be controlled separately. In this case, set group A Priority Control Setting bit (ADGSPCR.PGS) in the A/D Group Scan Priority Control Register (ADGSPCR) to 1 to assign a priority to group A conversion.

In group scan mode, select different channels and triggers for group A and group B.

Only set the ADCS[1:0] bits when the ADST bit is 0. Do not set the ADCS[1:0] bits at the same time that you write 1 to the ADST bit.

**Table 37.16 Selectable targets for A/D conversion depending on scan and double-trigger mode settings**

| Scan mode setting | Double-trigger mode setting | Targets for A/D conversion | | | | |
|---|---|---|---|---|---|---|
| | | Self-diagnosis | Analog input (group A) | Analog input (group B) | Temperature sensor output | Internal reference voltage |
| Single scan | DBLE = 0 | ✓ | ✓ | — | ✓ | ✓ |
| | DBLE = 1 | — | ✓ (1 ch only) | — | — | — |
| Continuous scan | DBLE = 0 | ✓ | ✓ | — | ✓ | ✓ |
| | DBLE = 1 | — | — | — | — | — |
| Group scan | DBLE = 0 | ✓ | ✓ | ✓ | ✓ | ✓ |
| | DBLE = 1 | — | ✓ (1 ch only) | ✓ | ✓ | ✓ |

Note: ✓: Selectable, —: Not selectable

### ADST bit (A/D Conversion Start)

The ADST bit starts or stops the A/D conversion process. Before the ADST bit is set to 1, set the A/D conversion clock, the conversion mode, and the conversion target analog input.

[Setting conditions]

- 1 is written.
- The synchronous trigger (ELC) selected in the ADSTRGR.TRSA[5:0] bits is detected when ADCSR.EXTRG is 0 and ADCSR.TRGE is 1.
- The synchronous trigger (ELC) selected in the ADSTRGR.TRSB[5:0] bits is detected when ADCSR.TRGE is set to 1 in group scan mode.
- The asynchronous trigger is detected when the ADCSR.TRGE and ADCSR.EXTRG bits are set to 1 and the ADSTRGR.TRSA[5:0] bits are set to 0x00.
- When group priority operation mode is enabled (ADCSR.ADCS[1:0] = 01b and ADGSPCR.PGS = 1), the ADGSPCR.GBRP bit is set to 1, and each time A/D conversion on the group with the lowest priority is started.

[Clearing conditions]

- 0 is written.
- The A/D conversion of all the selected channels, the temperature sensor output the internal reference voltage completes in single scan mode.
- Group A scan completes in group scan mode.
- Group B scan completes in group scan mode.

- When group priority operation mode is enabled (ADCSR.ADCS[1:0] = 01b and ADGSPCR.PGS = 1), the ADGSPCR.GBRSCN bit is set to 1, and A/D conversion on the group with the lowest priority started by trigger completes.

Note: When group priority operation mode is enabled (ADCSR.ADCS[1:0] = 01b and ADGSPCR.PGS = 1), do not set the ADST bit to 1.
Note: When group priority operation mode is enabled (ADCSR.ADCS[1:0] = 01b and ADGSPCR.PGS = 1), do not set the ADST bit to 0. When forcing A/D conversion to terminate, follow the procedure for clearing the ADST bit.

## 37.2.8 ADANSA0 : A/D Channel Select Register A0

Base address: ADC120 = 0x4017_0000

Offset address: 0x004

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ANSA15 | ANSA14 | ANSA13 | ANSA12 | ANSA11 | ANSA10 | ANSA9 | ANSA8 | ANSA7 | ANSA6 | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ANSA15 to ANSA0 | A/D Conversion Channels Select<br>Bit 15 (ANSA15) is associated with AN015 and bit 0 (ANSA0) is associated with AN000.<br>0: Do not select associated input channel.<br>1: Select associated input channel. | R/W |

Note: n = 00 to 08, 11 to 13
Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

ADANSA0 register selects analog input channels for A/D conversion. In group scan mode, this register selects group A channels.

Only set the ADANSA0 register when the ADCSR.ADST bit is 0.

### ANSAn bits (A/D Conversion Channels Select)

The ADANSA0 register selects any combination of analog input channels for A/D conversion. The channels and the number of channels can be arbitrarily set.

In double trigger mode, the channels selected in the ADANSA0 register are invalid, and the channel selected in the ADCSR.DBLANS[4:0] bits is selected in group A instead.

When group scan mode is selected, do not select the channels specified in A/D Channel Select Register B0 (ADANSB0) and A/D Channel Select Register B1 (ADANSB1).

## 37.2.9 ADANSA1 : A/D Channel Select Register A1

Base address: ADC120 = 0x4017_0000

Offset address: 0x006

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ANSA31 | ANSA30 | ANSA29 | ANSA28 | ANSA27 | ANSA26 | ANSA25 | ANSA24 | ANSA23 | ANSA22 | ANSA21 | ANSA20 | ANSA19 | ANSA18 | ANSA17 | ANSA16 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ANSA31 to ANSA16 | A/D Conversion Channels Select<br>Bit 15 (ANSA31) is associated with AN031 and bit 0 (ANSA16) is associated with AN016.<br>0: Do not select associated input channel.<br>1: Select associated input channel. | R/W |

Note: n = 16
Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

ADANSA1 register selects analog input channels for A/D conversion. In group scan mode, this register selects group A channels.

Only set the ADANSA1 register when the ADCSR.ADST bit is 0.

### ANSAn bits (A/D Conversion Channels Select)

The ADANSA1 register selects any combination of analog input channels for A/D conversion. The channels and the number of channels can be arbitrarily set.

In double trigger mode, the channels selected in the ADANSA1 register are invalid, and the channel selected in the ADCSR.DBLANS[4:0] bits is selected in group A instead.

When group scan mode is selected, do not select the channels specified in A/D Channel Select Register B0 (ADANSB0) and A/D Channel Select Register B1 (ADANSB1).

## 37.2.10 ADANSB0 : A/D Channel Select Register B0

Base address: ADC120 = 0x4017_0000

Offset address: 0x014

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ANSB15 | ANSB14 | ANSB13 | ANSB12 | ANSB11 | ANSB10 | ANSB9 | ANSB8 | ANSB7 | ANSB6 | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ANSB15 to ANSB0 | A/D Conversion Channels Select<br>Bit 15 (ANSB15) is associated with AN015 and bit 0 (ANSB0) is associated with AN000.<br>0: Do not select associated input channel.<br>1: Select associated input channel. | R/W |

Note: n = 00 to 08, 11 to 13

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

ADANSB0 selects analog input channels for A/D conversion in group B when group scan mode is selected. The ADANSB0 register is not used in any scan mode other than group scan mode.

Only set the ADANSB0 register when the ADCSR.ADST bit is 0.

### ANSBn bits (A/D Conversion Channels Select)

The ADANSB0 register selects any combination of analog input channels in group B for A/D conversion when group scan mode is selected. The ADANSB0 register is used for group scan mode only and not for any other modes.

Do not select channels specified in group A as selected in the ADANSA0 and ADANSA1 registers or the ADCSR.DBLANS[4:0] bits in double-trigger mode.

## 37.2.11 ADANSB1 : A/D Channel Select Register B1

Base address: ADC120 = 0x4017_0000

Offset address: 0x016

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ANSB31 | ANSB30 | ANSB29 | ANSB28 | ANSB27 | ANSB26 | ANSB25 | ANSB24 | ANSB23 | ANSB22 | ANSB21 | ANSB20 | ANSB19 | ANSB18 | ANSB17 | ANSB16 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ANSB31 to ANSB16 | A/D Conversion Channels Select<br>Bit 15 (ANSB31) is associated with AN031 and bit 0 (ANSB16) is associated with AN016.<br>0: Do not select associated input channel.<br>1: Select associated input channel. | R/W |

Note: n = 16

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

ADANSB1 selects analog input channels for A/D conversion in group B when group scan mode is selected. The ADANSB1 register is not used in any scan mode other than group scan mode.

Only set the ADANSB1 register when the ADCSR.ADST bit is 0.

### ANSBn bits (A/D Conversion Channels Select)

The ADANSB1 register selects any combination of analog input channels in group B for A/D conversion when group scan mode is selected. The ADANSB1 register is used for group scan mode only and not for any other modes.

Do not select channels specified in group A as selected in the ADANSA0 and ADANSA1 registers or the ADCSR.DBLANS[4:0] bits in double-trigger mode.

## 37.2.12 ADADS0 : A/D-Converted Value Addition/Average Channel Select Register 0

Base address: ADC120 = 0x4017_0000

Offset address: 0x008

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADS15 | ADS14 | ADS13 | ADS12 | ADS11 | ADS10 | ADS9 | ADS8 | ADS7 | ADS6 | ADS5 | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ADS15 to ADS0 | A/D-Converted Value Addition/Average Channel Select<br>Bit 15 (ADS15) is associated with AN015 and bit 0 (ADS0) is associated with AN000.<br>0: Do not select associated input channel.<br>1: Select associated input channel. | R/W |

Note: n = 00 to 08, 11 to 13

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

### ADSn bits (A/D-Converted Value Addition/Average Channel Select)

The ADSn bits determine which A/D-converted channels are subject to A/D-converted value addition/averaging. When an ADSn bit associated with a channel selected for A/D conversion is set to 1, A/D conversion of the analog input of the respective channel is performed successively 1, 2, 3, 4, or 16 times, as specified in the ADC[2:0] bits in the ADADC register.

When the ADADC.AVEE bit is 0, the value obtained by addition is stored in the A/D data register. When the ADADC.AVEE bit is 1, the mean value of the results obtained by addition is stored in the A/D data register.

The ADSn bits apply only to channels that are selected for A/D conversion in:

- The ANSAn bits in the ADANSA0 register or the DBLANS[4:0] bits in the ADCSR register
- The ANSBn bits in the ADANSB0 register

For channels on which the A/D conversion is performed and for which addition/average mode is not selected, a normal 1-time conversion is executed, and the conversion result is stored in the A/D data register.

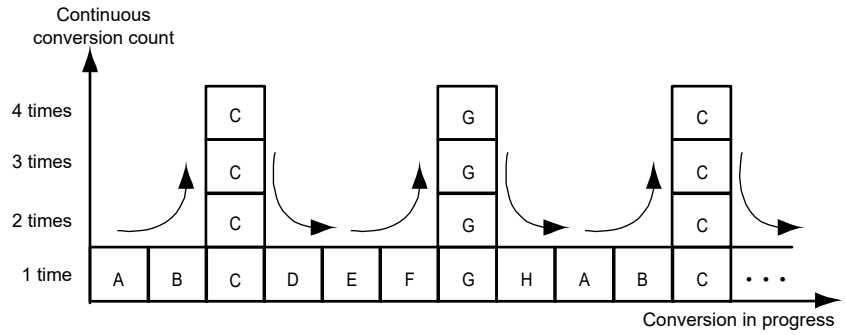
Only set ADADS0 register bits when the ADCSR.ADST bit is 0.

Figure 37.2 shows a scanning operation sequence in which the ADADS0 register bits (channel c and g) are set to 1. In this figure:

- Addition mode is selected (ADADC.AVEE = 0)
- The number of conversions is set to 4 (ADADC.ADC[1:0] = 11b)
- The analog input channels (a to h) are selected by ADANSA0 register in continuous scan mode (ADCSR.ADCS[1:0] = 10b).

The conversion process begins with analog input A (channel a). The analog input C (channel c) conversion is performed successively 4 times and the added value is returned to A/D Data Register c (ADDRc). Next, the analog input D (channel d) conversion process is started. The analog input G (channel g) is performed successively 4 times and the added value is

returned to A/D Data Register g (ADDRg). After conversion of analog input H (channel h), the conversion operation repeats in the same sequence starting with analog input A (channel a).




Note: A to H are analog inputs (channel a to h)

**Figure 37.2 Scan conversion sequence with ADADC.ADC[2:0] = 011b, set 1 for analog inputs C and G by ADADS0/1**

## 37.2.13 ADADS1 : A/D-Converted Value Addition/Average Channel Select Register 1

Base address: ADC120 = 0x4017_0000

Offset address: 0x00A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADS31 | ADS30 | ADS29 | ADS28 | ADS27 | ADS26 | ADS25 | ADS24 | ADS23 | ADS22 | ADS21 | ADS20 | ADS19 | ADS18 | ADS17 | ADS16 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ADS31 to ADS16 | A/D-Converted Value Addition/Average Channel Select<br>Bit 15 (ADS31) is associated with AN031 and bit 0 (ADS16) is associated with AN016.<br>0: Do not select associated input channel.<br>1: Select associated input channel. | R/W |

Note: n = 16

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

### ADSn bits (A/D-Converted Value Addition/Average Channel Select)

The ADSn bits determine which A/D-converted channels are subject to A/D-converted value addition/averaging. When an ADSn bit associated with a channel selected for A/D conversion is set to 1, A/D conversion of the analog input of the respective channel is performed successively 1, 2, 3, 4, or 16 times, as specified in the ADC[2:0] bits in the ADADC register.

When the ADADC.AVEE bit is 0, the value obtained by addition is stored in the A/D data register. When the ADADC.AVEE bit is 1, the mean value of the results obtained by addition is stored in the A/D data register.

The ADSn bits apply only to channels that are selected for A/D conversion in:

- The ANSAn bits in the ADANSA1 register or the DBLANS[4:0] bits in the ADCSR register
- The ANSBn bits in the ADANSB1 register.

For channels on which the A/D conversion is performed and for which addition/average mode is not selected, a normal 1-time conversion is executed, and the conversion result is stored in the A/D data register.

Only set ADADS1 register when the ADCSR.ADST bit is 0.

## 37.2.14 ADADC : A/D-Converted Value Addition/Average Count Select Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x00C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | AVEE | — | — | — | — | ADC[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | ADC[2:0] | Addition/Average Count Select<br>0 0 0: 1-time conversion (no addition, same as normal conversion)<br>0 0 1: 2-time conversion (1 addition)<br>0 1 0: 3-time conversion (2 additions)Setting prohibited<br>0 1 1: 4-time conversion (3 additions)<br>1 0 1: 16-time conversion (15 additions)<br>Others: Setting prohibited | R/W |
| 6:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | AVEE | Average Mode Select<br>0: Enable addition mode<br>1: Enable average mode | R/W |

ADADC sets the addition or average mode and addition count for A/D conversion. Table 37.17 lists the settable combinations of ADADC register.

**Table 37.17 Settable combinations of ADADC register**

| Mode select (AVEE) | Resolution | Conversion time | | | | |
|---|---|---|---|---|---|---|
| | | 1-time | 2-time | 3-time | 4-time | 16-time |
| Addition mode (AVEE = 0) | 8-bit | ✓ | ✓ | ✓ | ✓ | — |
| | 10-bit | ✓ | ✓ | ✓ | ✓ | — |
| | 12-bit | ✓ | ✓ | ✓ | ✓ | ✓ |
| Average mode (AVEE = 1) | 8, 10, 12 bits | — | ✓ | — | ✓ | — |

Note: ✓: Selectable, —: Not selectable

### ADC[2:0] bits (Addition/Average Count Select)

The ADC[2:0] bits set the addition count in all channels for which A/D conversion and addition/average mode are selected, including the channel selected in double trigger mode with the ADCSR.DBLANS[4:0] bits. The count also applies to A/D conversion of the temperature sensor output and the internal reference voltage.
When self-diagnosis is executed (ADCER.DIAGM = 1), do not set the ADC[2:0] bits to any value other than 000b.

### AVEE bit (Average Mode Select)

The AVEE bit selects addition or average mode in all channels for which A/D conversion and addition/average mode are selected, including the channels selected in double-trigger mode in the ADCSR.DBLANS[4:0] bits, temperature sensor output, internal reference voltage.

## 37.2.15 ADCER : A/D Control Extended Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x00E

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADRFMT | — | — | — | DIAGM | DIAGLD | DIAGVAL[1:0] | | — | — | ACE | — | — | ADPRC[1:0] | | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 2:1 | ADPRC[1:0] | A/D Conversion Accuracy Specify<br>0 0: 12-bit accuracy<br>0 1: 10-bit accuracy<br>1 0: 8-bit accuracy<br>1 1: Setting prohibited | R/W |
| 4:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5 | ACE | A/D Data Register Automatic Clearing Enable<br>0: Disable automatic clearing<br>1: Enable automatic clearing | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |
| 9:8 | DIAGVAL[1:0] | Self-Diagnosis Conversion Voltage Select<br>0 0: Setting prohibited when self-diagnosis is enabled<br>0 1: 0 volts<br>1 0: Reference voltage*1 × 1/2<br>1 1: Reference voltage*1 | R/W |
| 10 | DIAGLD | Self-Diagnosis Mode Select<br>0: Select rotation mode for self-diagnosis voltage<br>1: Select mixed mode for self-diagnosis voltage | R/W |
| 11 | DIAGM | Self-Diagnosis Enable<br>0: Disable ADC12 self-diagnosis<br>1: Enable ADC12 self-diagnosis | R/W |
| 14:12 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15 | ADRFMT | A/D Data Register Format Select<br>0: Select right-justified for the A/D data register format<br>1: Select left-justified for the A/D data register format | R/W |

Note 1. The reference voltage refers to VREFH0 (Unit 0).

### ADPRC[1:0] bit (A/D Conversion Accuracy Specify)

The ADPRC[1:0] bits set the A/D conversion accuracy. Changing the A/D conversion accuracy also changes the bit width of valid data stored in the result register and the A/D conversion time. For details, see section 37.3.6. Analog Input Sampling and Scan Conversion Timesection 45.3.6, Analog Input Sampling and Scan Conversion Time. Only set the ADPRC[1:0] bits while the ADCSR.ADST bit is 0.

### ACE bit (A/D Data Register Automatic Clearing Enable)

The ACE bit enables or disables automatic clearing (all 0) of the ADDRy, ADRD, ADDBLDR, ADDBLDRA, ADDBLDRB, ADTSDR, or ADOCDR register after any of these registers is read by the CPU or DTC. Automatic clearing of the A/D data registers enables detection of failures that are not updated in the A/D data registers. For details, see section 37.3.7. Usage Example of A/D Data Register Automatic Clearing Function.

### DIAGVAL[1:0] bits (Self-Diagnosis Conversion Voltage Select)

The DIAGVAL[1:0] bits select the voltage value used in self-diagnosis fixed voltage mode. For details, see the DIAGLD bit description.

Do not execute self-diagnosis by setting the DIAGLD bit to 1 when the DIAGVAL[1:0] bits are set to 00b.

### DIAGLD bit (Self-Diagnosis Mode Select)

The DIAGLD bit selects whether the three voltage values are rotated, or the fixed voltage is used in self-diagnosis.

Setting the DIAGLD bit to 0 selects conversion of the voltages in rotation mode, where 0 V, the reference voltage × 1/2, and the reference voltage are converted, in that order. After reset and when self-diagnosis voltage rotation mode is selected, self-diagnosis is executed from 0 V. The self-diagnosis voltage value does not return to 0 V when scan conversion completes. When scan conversion is restarted, rotation starts at the voltage value following the previous value.

Setting the DIAGLD bit to 1 selects fixed voltage, in which the fixed voltage specified in the ADCER.DIAGVAL[1:0] bits is converted. If fixed mode is switched to rotation mode, rotation starts at the fixed voltage value.

Only set the DIAGLD bit when the ADCSR.ADST bit is 0.

### DIAGM bit (Self-Diagnosis Enable)

The DIAGM bit enables or disables self-diagnosis.

Self-diagnosis is used to detect a failure of the ADC12. In self-diagnosis mode, one of the three voltage values (0 V, the reference voltage × 1/2, or the reference voltage) is converted. When conversion completes, information on the converted voltage and the conversion result is stored into the A/D Self-Diagnosis Data Register (ADRD). The ADRD register can be read to determine whether the conversion result falls within the normal or abnormal range.

Self-diagnosis is executed once at the beginning of each scan, and one of the three voltages is converted. In double trigger mode (ADCSR.DBLE = 1), self-diagnosis (DIAGM = 0) is deselected. When self-diagnosis is selected in group scan mode, self-diagnosis is executed separately for group A and group B.

Only set the DIAGM bit when the ADCSR.ADST bit is 0.

### ADRFMT bit (A/D Data Register Format Select)

The ADRFMT bit specifies flush-right or flush-left for data to be stored in the ADDRy, ADDBLDR, ADDBLDRA, ADDBLDRB, ADTSDR, ADOCDR, ADCMPDR0/1, ADWINLLB, ADWINULB, or ADRD register.

Only set the ADRFMT bit when the ADCSR.ADST bit is 0.

## 37.2.16 ADSTRGR : A/D Conversion Start Trigger Select Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x010

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | TRSA[5:0] | | | | | | — | — | TRSB[5:0] | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | TRSB[5:0] | A/D Conversion Start Trigger Select for Group B<br>Select the A/D conversion start trigger for group B in group scan mode. | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |
| 13:8 | TRSA[5:0] | A/D Conversion Start Trigger Select<br>Select the A/D conversion start trigger in single scan mode and continuous scan mode. In group scan mode, the A/D conversion start trigger for group A is selected. | R/W |
| 15:14 | — | These bits are read as 0. The write value should be 0. | R/W |

### TRSB[5:0] bits (A/D Conversion Start Trigger Select for Group B)

The TRSB[5:0] bits select the trigger to start scanning of the analog input selected in group B. The TRSB[5:0] bits must only be set in group scan mode and are not used in any other scan mode. For the scan conversion start trigger for group B, setting a software trigger or an asynchronous trigger is prohibited. In group scan mode, set the TRSB[5:0] bits to a value other than 0x00 and set the ADCSR.TRGE bit to 1.

When group A is given priority in group scan mode, setting the ADGSPCR.GBRP bit to 1 allows group B to continuously operate in single scan mode. When setting the ADGSPCR.GBRP bit to 1, set the TRSB[5:0] bits to 0x3F. The issuance

period for a conversion trigger must be more than or equal to the actual scan conversion time ($t_{SCAN}$). If the issuance period is less than $t_{SCAN}$, A/D conversion by the trigger might have no effect.

Table 37.18 lists the A/D conversion startup sources selected in the TRSB[5:0] bits.

**Table 37.18 Selection of A/D conversion start sources in the TRSB[5:0] bits**

| Source | Remarks | TRSB[5] | TRSB[4] | TRSB[3] | TRSB[2] | TRSB[1] | TRSB[0] |
|---|---|---|---|---|---|---|---|
| Trigger source deselected state | — | 1 | 1 | 1 | 1 | 1 | 1 |
| ELC_AD00 | ELC | 0 | 0 | 1 | 0 | 0 | 1 |
| ELC_AD01 | ELC | 0 | 0 | 1 | 0 | 1 | 0 |
| ELC_AD00, ELC_AD01 | ELC | 0 | 0 | 1 | 0 | 1 | 1 |

### TRSA[5:0] bits (A/D Conversion Start Trigger Select)

The TRSA[5:0] bits select the trigger to start A/D conversion in single scan mode and continuous scan mode, or the trigger to start scanning of group A analog inputs in group scan mode. When scanning is executed in group scan mode or double trigger mode, software trigger or asynchronous trigger is prohibited.

- When using a synchronous trigger (ELC), set the TRGE bit in the ADCSR register to 1 and set the EXTRG bit in the ADCSR register to 0.
- When using the asynchronous trigger (ADTRG0), set the TRGE bit in the ADCSR register to 1 and set the EXTRG bit in the ADCSR register to 1.
- Software trigger (ADCSR.ADST) is enabled regardless of the settings of the ADCSR.TRGE bit, the ADCSR.EXTRG bit, or the TRSA[5:0] bits.

The issuance period for a conversion trigger must be more than or equal to the actual scan conversion time (tSCAN). If the issuance period is less than tSCAN, A/D conversion by a trigger might have no effect.

Table 37.19 lists the A/D conversion start sources selected in the TRSA[5:0] bits.

**Table 37.19 Selection of A/D activation sources in the TRSA[5:0] bits**

| Source | Remarks | TRSA[5] | TRSA[4] | TRSA[3] | TRSA[2] | TRSA[1] | TRSA[0] |
|---|---|---|---|---|---|---|---|
| Trigger source deselected state | — | 1 | 1 | 1 | 1 | 1 | 1 |
| ADTRG0 | Input pin for the trigger | 0 | 0 | 0 | 0 | 0 | 0 |
| ELC_AD00 | ELC | 0 | 0 | 1 | 0 | 0 | 1 |
| ELC_AD01 | ELC | 0 | 0 | 1 | 0 | 1 | 0 |
| ELC_AD00, ELC_AD01 | ELC | 0 | 0 | 1 | 0 | 1 | 1 |

## 37.2.17 ADEXICR : A/D Conversion Extended Input Control Registers

Base address: ADC120 = 0x4017_0000

Offset address: 0x012

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | OCSB | TSSB | OCSA | TSSA | — | — | — | — | — | — | OCSAD | TSSAD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TSSAD | Temperature Sensor Output A/D-Converted Value Addition/Average Mode Select<br>0: Do not select addition/average mode for temperature sensor output.<br>1: Select addition/average mode for temperature sensor output. | R/W |
| 1 | OCSAD | Internal Reference Voltage A/D-Converted Value Addition/Average Mode Select<br>0: Do not select addition/average mode for internal reference voltage.<br>1: Select addition/average mode for internal reference voltage. | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | TSSA | Temperature Sensor Output A/D Conversion Select<br>0: Disable A/D conversion of temperature sensor output<br>1: Enable A/D conversion of temperature sensor output | R/W |
| 9 | OCSA | Internal Reference Voltage A/D Conversion Select<br>0: Disable A/D conversion of internal reference voltage<br>1: Enable A/D conversion of internal reference voltage | R/W |
| 10 | TSSB | Temperature Sensor Output A/D Conversion Select for Group B<br>0: Disable A/D conversion of temperature sensor output<br>1: Enable A/D conversion of temperature sensor output | R/W |
| 11 | OCSB | Internal Reference Voltage A/D Conversion Select for Group B<br>0: Disable A/D conversion of internal reference voltage<br>1: Enable A/D conversion of internal reference voltage | R/W |
| 15:12 | — | These bits are read as 0. The write value should be 0. | R/W |

### TSSAD bit (Temperature Sensor Output A/D-Converted Value Addition/Average Mode Select)

When the TSSAD bit is set to 1, A/D conversion of the temperature sensor output is selected and performed successively the number of times specified in the ADC[2:0] bits in ADADC. When the ADADC.AVEE bit is 0, the value obtained by addition (integration) is returned to the A/D Temperature Sensor Data Register (ADTSDR). When the ADADC.AVEE bit is 1, the mean value is returned to ADTSDR.

Only set the TSSAD bit while the ADCSR.ADST bit is 0.

### OCSAD bit (Internal Reference Voltage A/D-Converted Value Addition/Average Mode Select)

When the OCSAD bit is set to 1, A/D conversion of the internal reference voltage is selected and performed successively the number of times specified in the ADC[2:0] bits in ADADC. When the ADADC.AVEE bit is 0, the value obtained by addition (integration) is returned to the A/D Internal Reference Voltage Data Register (ADOCDR). When the ADADC.AVEE bit is 1, the mean value is returned to ADOCDR.

Only set the OCSAD bit while the ADCSR.ADST bit is 0.

### TSSA bit (Temperature Sensor Output A/D Conversion Select)

The TSSA bit selects A/D conversion of the temperature sensor output for group A in single scan mode, continuous scan mode, or group scan mode. When A/D conversion of the temperature sensor output is selected and performed, set the ADCSR.DBLE bit to 0.

Only set the TSSA bit while the ADCSR.ADST bit is 0.

### OCSA bit (Internal Reference Voltage A/D Conversion Select)

The OCSA bit selects A/D conversion of the internal reference voltage for group A in single scan mode, continuous scan mode, or group scan mode. When A/D conversion of the internal reference voltage is selected and performed, set the ADCSR.DBLE bit to 0.

Only set the OCSA bit while the ADCSR.ADST bit is 0. In addition, wait for 400 ns or more after the OCSA bit is set to 1 before starting A/D conversion.

### TSSB bit (Temperature Sensor Output A/D Conversion Select for Group B)

The TSSB bit selects A/D conversion of the temperature sensor output for group B in group scan mode. Only set the TSSB bit while the ADCSR.ADST bit is 0. Do not set the TSSB bit to 1 while the TSSA bit is 1.

### OCSB bit (Internal Reference Voltage A/D Conversion Select for Group B)

The OCSB bit selects A/D conversion of the internal reference voltage for group B in group scan mode. Only set the OCSB bit while the ADCSR.ADST bit is 0. Do not set the OCSB bit to 1 while the OCSA bit is 1. Moreover, start the A/D conversion after waiting for 400 ns or more after the OCSB bit is set to 1.

## 37.2.18 ADSSTRn/ADSSTRL/ADSSTRT/ADSSTRO : A/D Sampling State Register (n = 0 to 8, 11 to 13)

Base address: ADC120 = 0x4017_0000

Offset address: 0x0E0 + 0x1 × n (n = 0 to 8, 11 to 13)
0x0DD (ADSSTRL)
0x0DE (ADSSTRT)
0x0DF (ADSSTRO)

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | SST[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | SST[7:0] | Sampling Time Setting<br>These bits set the sampling time in the range from 5 to 255 states. | R/W |

The ADSSTRn register sets the sampling time for analog input.

The sampling time can be adjusted if the impedance of the analog input signal source is too high to secure sufficient sampling time, or if the ADCLK clock is slow. The set value indicates the time for one ADCLK cycle, and the required sampling time is specified by the voltage conditions. For details, see section 47.5. ADC12 Characteristics.

The lower limit of the sampling time setting depends on the frequency ratio:

- If the frequency ratio of PCLKA to PCLKC (ADCLK) = 1:1, 2:1, 4:1, or 8:1 the sampling time must be set to a value of more than 5 states
- If the frequency ratio of PCLKA to PCLKC (ADCLK) = 1:2 or 1:4, the sampling time must be set to a value of more than 6 states.

Table 37.20 shows the relationship between the A/D Sampling State Register and the associated channels. For details, see section 37.3.6. Analog Input Sampling and Scan Conversion Time.

Only set the SST[7:0] bits when the ADCSR.ADST bit is 0.

**Table 37.20 Relationship between A/D sampling state register and associated channels**

| Bit name | Associated channels |
|---|---|
| ADSSTRn.SST[7:0] bits (n = 0 to 8, 11 to 13)*1 | AN000 to AN008, AN011 to AN013 |
| ADSSTRL.SST[7:0] bits | AN016 |
| ADSSTRT.SST[7:0] bits | Temperature sensor output |
| ADSSTRO.SST[7:0] bits | Internal reference voltage |

Note 1. When the self-diagnosis function is selected, the sampling time set in the ADSSTR0.SST[7:0] bits is applied.

## 37.2.19 ADDISCR : A/D Disconnection Detection Control Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x07A

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | PCHG | ADNDIS[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | ADNDIS[3:0] | Disconnection Detection Assist Setting<br>0x0: The disconnection detection assist function is disabled<br>0x1: Setting prohibited<br>Others: The number of states for the discharge or precharge period. | R/W |
| 4 | PCHG | Precharge/discharge select<br>0: Discharge<br>1: Precharge | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

The ADDISCR register selects either precharge or discharge, and the period of precharge or discharge for the A/D disconnection detection assist function. Only set the ADDISCR register when the ADCSR.ADST bit is 0. When the temperature sensor output or internal reference voltage is converted, the A/D converter executes discharge automatically.

Disable the disconnection detection assist function if any of the following functions are used:

- The temperature sensor
- The internal reference voltage
- A/D self-diagnosis

### ADNDIS[3:0] bits (Disconnection Detection Assist Setting)

The ADNDIS[3:0] bits specify the period of precharge or discharge. When ADNDIS[3:0] = 0000b, the disconnection detection assist function is disabled. Setting the ADNDIS[3:0] bits to 0001b is prohibited. Except when ADNDIS[3:0] = 0000b or 0001b, the specified value indicates the number of states for the period of precharge or discharge. When the ADNDIS[3:0] bits are set to any values other than 0000b or 0001b, the disconnection detection assistance function is enabled.

### PCHG bit (Precharge/discharge select)

The PCHG bit selects either precharge or discharge.

## 37.2.20 ADGSPCR : A/D Group Scan Priority Control Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x080

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | GBRP | LGRRS | — | — | — | — | — | — | — | — | — | — | — | — | GBRSCN | PGS |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PGS | Group Priority Operation Setting*1<br>0: Operate without group priority control.<br>1: Operate with group priority control. | R/W |
| 1 | GBRSCN | Lower-Priority Group Restart Setting<br>(enabled only when PGS = 1 and reserved when PGS = 0.)<br>0: Disable rescanning of the group that was stopped in group priority operation<br>1: Enable rescanning of the group that was stopped in group priority operation. | R/W |
| 13:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 14 | LGRRS | Restart Channel Select<br>Enabled only when PGS = 1 and GBRSCN = 1.<br>0: Start rescanning from the first channel for scanning<br>1: Start rescanning from the channel for which A/D conversion is not completed. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15 | GBRP | Single Scan Continuous Start*2<br>(enabled only when PGS = 1 and reserved when PGS = 0.)<br>0: Single scan is not continuously activated.<br>1: Single scan for the group with the lower-priority is continuously activated. | R/W |

Note 1. The ADCSR.ADCS[1:0] bits must be set to 01b (group scan mode) before setting PGS to 1. Operation is not guaranteed if these bits are set to any other value.
Note 2. When the GBRP bit is set to 1, single scan is performed continuously for the group with the lower-priority regardless of the setting in the GBRSCN bit.

### PGS bit (Group Priority Operation Setting)

The PGS bit controls group priority operation in group scan mode. Set the PGS bit to 1 to enable group priority operation.

The ADCSR.ADCS[1:0] bits must be set to 01b (group scan mode) before setting the PGS bit to 1. Operation is not guaranteed if the bits are set to any other value.

When the PGS bit is set to 0, a clear operation must be performed by software as described in section 37.6.3. Constraints on Stopping A/D Conversion. When the PGS bit is set to 1, use the settings described in section 37.3.4.3. Group Priority Operation.

### GBRSCN bit (Lower-Priority Group Restart Setting)

The GBRSCN bit controls the restarting of scan operation in group priority operation.

When the GBRSCN bit is set to 1, if the scan operation of a lower-priority group is stopped by a trigger input of a priority group, the lower-priority group scanning is restarted on completion of the priority group scanning. If a trigger of a lower-priority group is input during scanning of the priority group, the lower-priority group scanning is started on completion of the priority group scanning.

When the GBRSCN bit is set to 0, triggers input during scanning are ignored. Set the GBRSCN bit while the ADCSR.ADST bit is 0.

### LGRRS bit (Restart Channel Select)

This bit sets the channel from which rescanning is to be started in group priority operation. The setting of the LGRRS bit is valid when the PGS and GBRSCN bits are 1.

If the LGRRS bit is 0, scanning of a lower-priority group that was stopped in group priority operation is restarted from the first channel after scanning of the priority group completes.

If the LGRRS bit is 1, scanning of a lower-priority group that was stopped in group priority operation is restarted (upon completion of scanning of the priority group) from the channel for which A/D conversion is not complete. If A/D conversion of the addition setting channel was not completed the specified number of times when scanning stopped, A/D conversion of the addition setting channel is performed again the specified number of times when scanning restarts.

Set the LGRRS bit while the ADCSR.ADST bit is 0.

### GBRP bit (Single Scan Continuous Start)

The GBRP bit is set when a single scan operation is to be performed continuously on the group with the lower-priority.

Setting the GBRP bit to 1 starts a single scan of the group with the lower-priority. On completion of the scan, another single scan of the group with the lower-priority is started automatically. If scanning has been stopped during group priority operation, single scan of the group with the lower-priority is automatically restarted on completion of the A/D conversion of the priority group.

Before setting the GBRP bit to 1, disable input of a trigger for the lower-priority group. If the GBRP bit is set to 1, rescanning is performed only on the group with the lower-priority even if the GBRSCN bit is set to 0.

## 37.2.21 ADCMPCR : A/D Compare Function Control Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x090

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CMPAIE | WCMPE | CMPBIE | — | CMPAE | — | CMPBE | — | — | — | — | — | — | — | CMPAB[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | CMPAB[1:0] | Window A/B Composite Conditions Setting<br>These bits are valid when both window A and window B are enabled (CMPAE = 1 and CMPBE = 1).<br>0 0: Output ADC120_WCMPM when window A OR window B comparison conditions are met. Otherwise, output ADC120_WCMPUM.<br>0 1: Output ADC120_WCMPM when window A EXOR window B comparison conditions are met. Otherwise, output ADC120_WCMPUM.<br>1 0: Output ADC120_WCMPM when window A AND window B comparison conditions are met. Otherwise, output ADC120_WCMPUM.<br>1 1: Setting prohibited. | R/W |
| 8:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 9 | CMPBE | Compare Window B Operation Enable<br>0: Disable compare window B operation.<br>Disable ADC120_WCMPM and ADC120_WCMPUM outputs.<br>1: Enable compare window B operation. | R/W |
| 10 | — | This bit is read as 0. The write value should be 0. | R/W |
| 11 | CMPAE | Compare Window A Operation Enable<br>0: Disable compare window A operation.<br>Disable ADC120_WCMPM and ADC120_WCMPUM outputs.<br>1: Enable compare window A operation. | R/W |
| 12 | — | This bit is read as 0. The write value should be 0. | R/W |
| 13 | CMPBIE | Compare B Interrupt Enable<br>0: Disable ADC120_CMPBI interrupt when comparison conditions (window B) are met.<br>1: Enable ADC120_CMPBI interrupt when comparison conditions (window B) are met. | R/W |
| 14 | WCMPE | Window Function Setting<br>0: Disable window function<br>Window A and window B operate as a comparator to compare the single value on the lower side with the A/D conversion result.<br>1: Enable window function<br>Window A and window B operate as a comparator to compare the two values on the upper and lower sides with the A/D conversion result. | R/W |
| 15 | CMPAIE | Compare A Interrupt Enable<br>0: Disable ADC120_CMPAI interrupt when comparison conditions (window A) are met.<br>1: Enable ADC120_CMPAI interrupt when comparison conditions (window A) are met. | R/W |

### CMPAB[1:0] bits (Window A/B Composite Conditions Setting)

The CMPAB[1:0] bits are valid when both window A and window B are enabled (CMPAE = 1 and CMPBE = 1) in single scan mode. These bits specify the compare function match/mismatch event output conditions and monitoring conditions of ADWINMON.MONCOMB. Only set the CMPAB[1:0] bits while the ADCSR.ADST bit is 0.

### CMPBE bit (Compare Window B Operation Enable)

The CMPBE bit enables or disables the compare window B operation. Set the CMPBE bit while the ADCSR.ADST bit is 0.

Set this bit to 0 before setting the following registers:

- A/D Channel Select Registers A0, A1, B0, B1 (ADANSA0, ADANSA1, ADANSB0, ADANSB1)
- OCSB, TSSB, OCSA, or TSSA bits in the A/D Conversion Extended Input Control Register (ADEXICR)
- CMPCHB[5:0] bits in the Window B Channel Select Register (ADCMPBNSR)

### CMPAE bit (Compare Window A Operation Enable)

The CMPAE bit enables or disables the compare window A operation. Set the CMPAE bit while the ADCSR.ADST bit is 0.

Set this bit to 0 before setting the following registers:

- A/D Channel Select Registers A0, A1, B0, B1 (ADANSA0, ADANSA1, ADANSB0, ADANSB1)
- OCSB, TSSB, OCSA, or TSSA bits in the A/D Conversion Extended Input Control Register (ADEXICR)
- Window A Channel Select Registers 0 and 1 (ADCMPANSR0 and ADCMPANSR1)
- Window A Extended Input Select Register (ADCMPANSER)

### CMPBIE bit (Compare B Interrupt Enable)

The CMPBIE bit enables or disables the ADC120_CMPBI interrupt output when the comparison conditions (window B) are met.

### WCMPE bit (Window Function Setting)

The WCMPE bit enables or disables the window function. Set the WCMPE bit while the ADCSR.ADST bit is 0.

### CMPAIE bit (Compare A Interrupt Enable)

The CMPAIE bit enables or disables the ADC120_CMPAI interrupt output when the comparison conditions (window A) are met.

## 37.2.22 ADCMPANSR0 : A/D Compare Function Window A Channel Select Register 0

Base address: ADC120 = 0x4017_0000

Offset address: 0x094

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CMPCHA15 | CMPCHA14 | CMPCHA13 | CMPCHA12 | CMPCHA11 | CMPCHA10 | CMPCHA9 | CMPCHA8 | CMPCHA7 | CMPCHA6 | CMPCHA5 | CMPCHA4 | CMPCHA3 | CMPCHA2 | CMPCHA1 | CMPCHA0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | CMPCHA15 to CMPCHA0 | Compare Window A Channel Select<br>Bit 15 (CMPCHA15) is associated with AN015 and bit 0 (CMPCHA0) is associated with AN000.<br>0: Disable compare function for associated input channel<br>1: Enable compare function for associated input channel | R/W |

Note: n = 00 to 08, 11 to 13

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

### CMPCHAn bits (Compare Window A Channel Select)

The compare function is enabled by writing 1 to the CMPCHAn bits with the same number as the A/D conversion channel selected in the ADANSA0.ANSAn bits and the ADANSB0.ANSBn bits.

Set the CMPCHAn bits while the ADCSR.ADST bit is 0.

## 37.2.23 ADCMPANSR1 : A/D Compare Function Window A Channel Select Register 1

Base address: ADC120 = 0x4017_0000

Offset address: 0x096

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CMPCHA31 | CMPCHA30 | CMPCHA29 | CMPCHA28 | CMPCHA27 | CMPCHA26 | CMPCHA25 | CMPCHA24 | CMPCHA23 | CMPCHA22 | CMPCHA21 | CMPCHA20 | CMPCHA19 | CMPCHA18 | CMPCHA17 | CMPCHA16 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | CMPCHA31 to CMPCHA16 | Compare Window A Channel Select<br><br>Bit 15 (CMPCHA31) is associated with AN031 and bit 0 (CMPCHA16) is associated with AN016.<br>0: Disable compare function for associated input channel<br>1: Enable compare function for associated input channel | R/W |

Note: n = 16

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

### CMPCHAn bits (Compare Window A Channel Select)

The compare function is enabled by writing 1 to the CMPCHAn bits with the same number as the A/D conversion channel selected in the ADANSA1.ANSA bits and the ADANSB1.ANSB bits.

Set the CMPCHAn bits while the ADCSR.ADST bit is 0.

## 37.2.24 ADCMPANSER : A/D Compare Function Window A Extended Input Select Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x092

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | CMPOCA | CMPTSA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CMPTSA | Temperature Sensor Output Compare Select<br>0: Exclude the temperature sensor output from the compare Window A target range.<br>1: Include the temperature sensor output in the compare Window A target range. | R/W |
| 1 | CMPOCA | Internal Reference Voltage Compare Select<br>0: Exclude the internal reference voltage from the compare Window A target range.<br>1: Include the internal reference voltage in the compare Window A target range. | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

### CMPTSA bit (Temperature Sensor Output Compare Select)

The compare Window A function is enabled by setting the CMPTSA bit to 1 while the ADEXICR.TSSA bit or the ADEXICR.TSSB bit is 1. Set the CMPTSA bit while the ADCSR.ADST bit is 0.

### CMPOCA bit (Internal Reference Voltage Compare Select)

The compare window A function is enabled by setting the CMPOCA bit to 1 when the ADEXICR.OCSA and ADEXICR.OCSB bit is 1. Set the CMPOCA bit when the ADCSR.ADST bit is 0.

## 37.2.25 ADCMPLR0 : A/D Compare Function Window A Comparison Condition Setting Register 0

Base address: ADC120 = 0x4017_0000

Offset address: 0x098

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CMPLCHA15 | CMPLCHA14 | CMPLCHA13 | CMPLCHA12 | CMPLCHA11 | CMPLCHA10 | CMPLCHA9 | CMPLCHA8 | CMPLCHA7 | CMPLCHA6 | CMPLCHA5 | CMPLCHA4 | CMPLCHA3 | CMPLCHA2 | CMPLCHA1 | CMPLCHA0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | CMPLCHA15 to CMPLCHA0 | Compare Window A Comparison Condition Select<br>These bits set comparison conditions for channels to which Window A comparison conditions are applied.<br>Bit 15 (CMPLCHA15) is associated with AN015 and bit 0 (CMPLCHA0) is associated with AN000.<br>Comparison conditions are shown in Figure 37.3.<br>0: When window function is disabled (ADCMPCR.WCMPE = 0):<br>ADCMPDR0 value > A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1):<br>A/D-converted value < ADCMPDR0 value,<br>or ADCMPDR1 value < A/D-converted value<br>1: When window function is disabled (ADCMPCR.WCMPE = 0):<br>ADCMPDR0 value < A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1):<br>ADCMPDR0 value < A/D-converted value < ADCMPDR1 value | R/W |

Note: n = 00 to 08, 11 to 13

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

### CMPLCHAn bits (Compare Window A Comparison Condition Select)

The CMPLCHAn bits specify the comparison conditions for channels to which Window A comparison conditions are applied. These bits can be set for each analog input to be compared. When the comparison result of each analog input meets the set condition, the ADCMPSR0.CMPSTCHAn flag sets to 1 and a compare interrupt (ADC120_CMPAI) is generated.

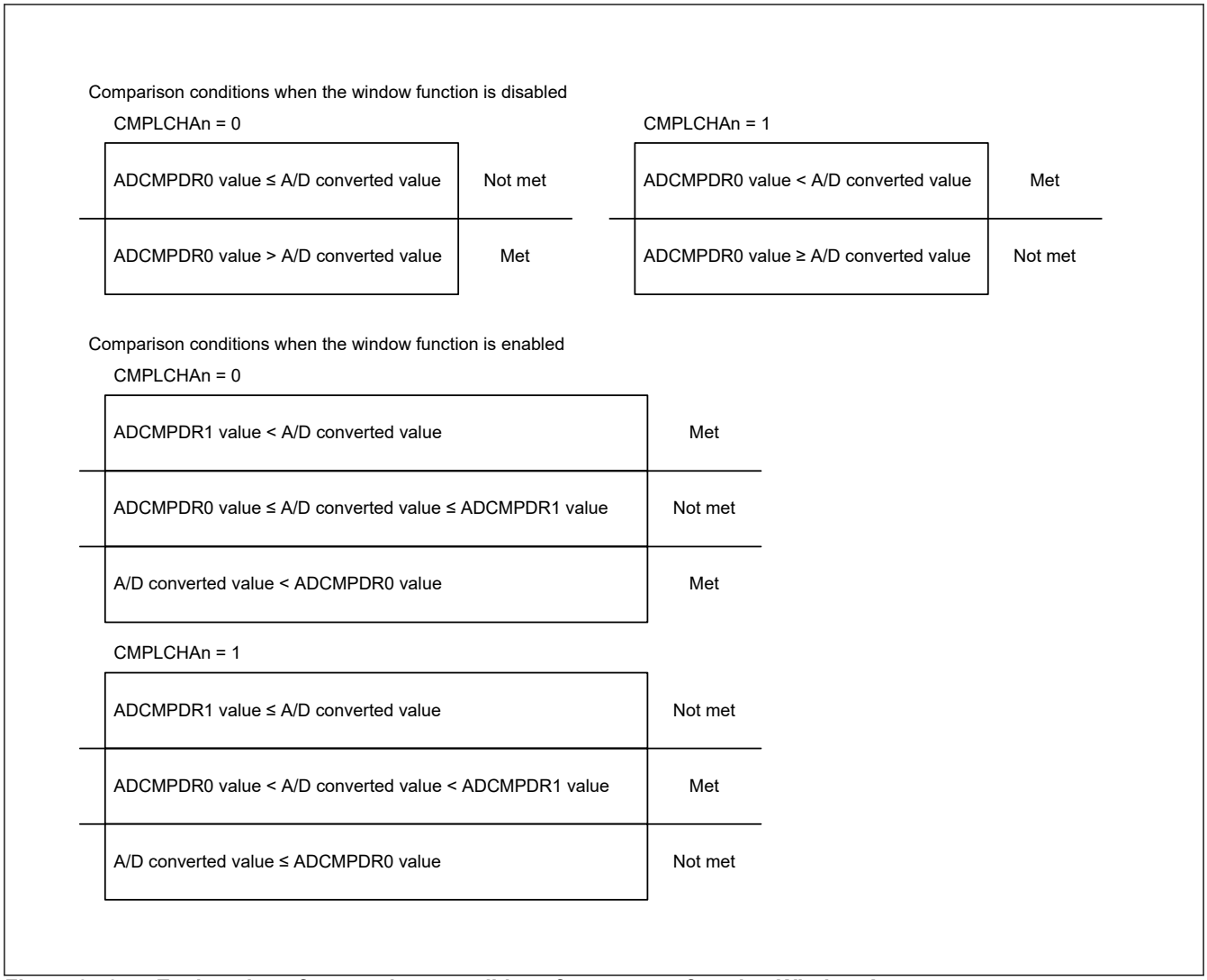



**Figure 37.3 Explanation of comparison conditions for compare function Window A**

## 37.2.26 ADCMPLR1 : A/D Compare Function Window A Comparison Condition Setting Register 1

Base address: ADC120 = 0x4017_0000

Offset address: 0x09A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CMPLCHA31 | CMPLCHA30 | CMPLCHA29 | CMPLCHA28 | CMPLCHA27 | CMPLCHA26 | CMPLCHA25 | CMPLCHA24 | CMPLCHA23 | CMPLCHA22 | CMPLCHA21 | CMPLCHA20 | CMPLCHA19 | CMPLCHA18 | CMPLCHA17 | CMPLCHA16 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | CMPLCHA31 to CMPLCHA16 | Compare Window A Comparison Condition Select<br>These bits set comparison conditions for channels to which Window A comparison conditions are applied.<br><br>Bit 15 (CMPLCHA31) is associated with AN031 and bit 0 (CMPLCHA16) is associated with AN016.<br>Comparison conditions are shown in Figure 37.3.<br>0: When window function is disabled (ADCMPCR.WCMPE = 0):<br>ADCMPDR0 value > A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1):<br>A/D-converted value < ADCMPDR0 value,<br>or ADCMPDR1 value < A/D-converted value<br>1: When window function is disabled (ADCMPCR.WCMPE = 0):<br>ADCMPDR0 value < A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1):<br>ADCMPDR0 value < A/D-converted value < ADCMPDR1 value | R/W |

Note: n = 16
Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

### CMPLCHAn bits (Compare Window A Comparison Condition Select)

The CMPLCHAn bits specify the comparison conditions for analog channels to which window A comparison conditions are applied. These bits can be set for each analog input to be compared. When the comparison result of each analog input meets the set condition, the ADCMPSR1.CMPSTCHAn bit is set to 1 and a compare interrupt (ADC120_CMPAI) is generated.

## 37.2.27 ADCMPLER : A/D Compare Function Window A Extended Input Comparison Condition Setting Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x093

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | CMPLOCA | CMPLTSA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CMPLTSA | Compare Window A Temperature Sensor Output Comparison Condition Select<br>Comparison conditions are shown in Figure 37.3.<br>0: When window function is disabled (ADCMPCR.WCMPE = 0) :<br>ADCMPDR0 value > A/D-converted value Compare Window A Temperature Sensor Output Comparison Condition Select<br>When window function is enabled (ADCMPCR.WCMPE = 1) :<br>Compare Window A Temperature Sensor Output Comparison Condition A/D-converted value < ADCMPDR0 value,<br>or A/D-converted value > ADCMPDR1 value<br>1: When window function is disabled (ADCMPCR.WCMPE = 0) :<br>ADCMPDR0 value < A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1) :<br>ADCMPDR0 value < A/D-converted value < ADCMPDR1 value | R/W |
| 1 | CMPLOCA | Compare Window A Internal Reference Voltage Comparison Condition Select<br>Comparison conditions are shown in Figure 37.3.<br>0: When window function is disabled (ADCMPCR.WCMPE = 0) :<br>ADCMPDR0 value > A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1):<br>A/D-converted value < ADCMPDR0 value,<br>or A/D-converted value > ADCMPDR1 value<br>1: When window function is disabled (ADCMPCR.WCMPE = 0):<br>ADCMPDR0 value < A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1):<br>ADCMPDR0 value < A/D-converted value < ADCMPDR1 value | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

### CMPLTSA bit (Compare Window A Temperature Sensor Output Comparison Condition Select)

The CMPLTSA bit specifies comparison conditions when the temperature sensor output is the target for the Window A comparison condition. When the temperature sensor output comparison result meets the set condition, the ADCMPSER.CMPSTTSA flag sets to 1 and a compare interrupt (ADC120_CMPAI) is generated.

### CMPLOCA bit (Compare Window A Internal Reference Voltage Comparison Condition Select)

The CMPLOCA bit specifies comparison conditions when the internal reference voltage is the target for the Window A comparison condition. When the internal reference voltage comparison result meets the set condition, the ADCMPSER.CMPSTOCA flag sets to 1 and a compare interrupt (ADC120_CMPAI) is generated.

## 37.2.28 ADCMPDRn : A/D Compare Function Window A Lower-Side/Upper-Side Level Setting Register (n = 0, 1)

Base address: ADC120 = 0x4017_0000

Offset address: 0x09C + (0x2 × n)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The ADCMPDRy (y = 0, 1) register specifies the reference data when the compare window A function is used. ADCMPDR0 sets the lower reference for window A, and ADCMPDR1 sets the upper reference for window A.

ADCMPDRy are read/write registers.

ADCMPDRy are writable even during A/D conversion. The reference data can be dynamically changed by rewriting register values during A/D conversion*1.

Set these registers so that the upper reference is not less than the lower reference (ADCMPDR1 ≥ ADCMPDR0). ADCMPDR1 are not used when the window function is disabled.

Note 1. The lower and the upper references are changed when each register is written. For example, when the upper reference value is changed and the lower reference value is being changed, the MCU compares the upper reference (after rewrite), and the lower reference (before rewrite) with the A/D conversion result. See Figure 37.4. If the comparison during the rewriting of these two references is erroneous, then rewrite these reference values when both ADCSR.ADST and the target Compare Window Operation Enable bit (ADCMPCR.CMPAE or ADCMPCR.CMPBE) are 0.

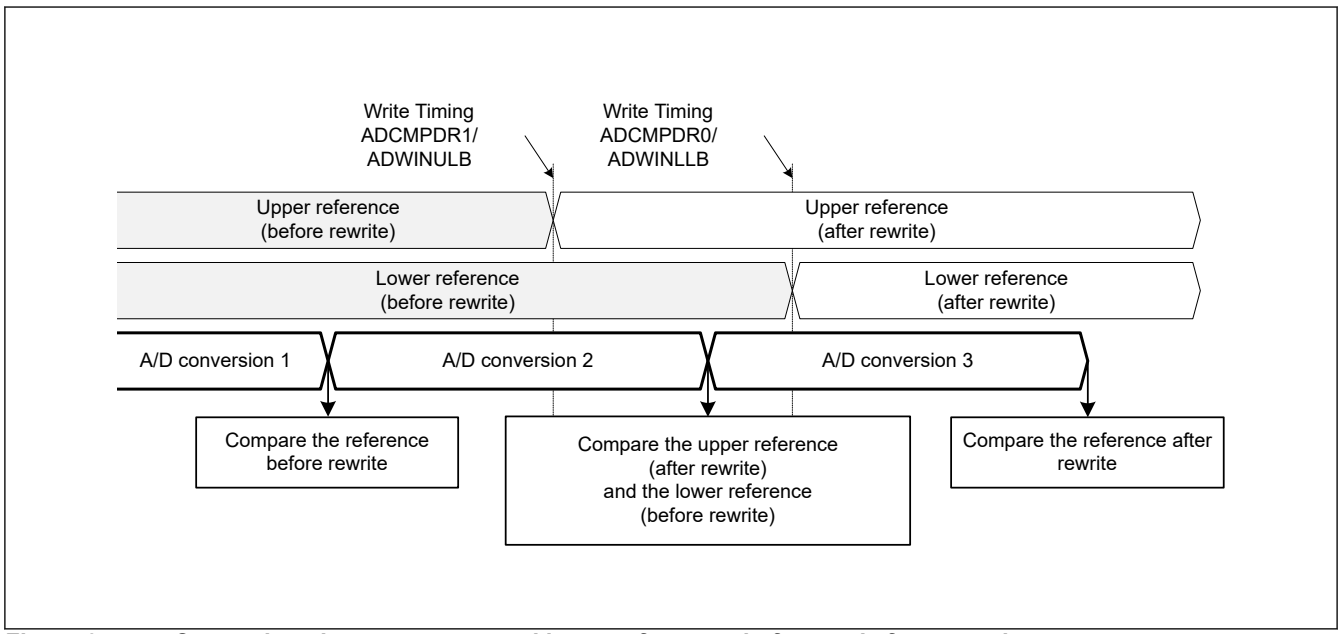



**Figure 37.4 Comparison between upper and lower references before and after a rewrite**

The ADCMPDRy registers use different formats depending on the following conditions:

- The value of A/D Data Register Format Select bit (flush-right or flush-left)
- The value of the A/D Conversion Accuracy Select bit (12-bit, 10-bit, 8-bit)
- The value of A/D-Converted Value Addition/Average Channel Select bits (A/D-converted value addition mode selected or not selected).

The data formats for each condition are shown as follows:

1. When A/D-converted value addition mode is not selected
   - Flush-right data with 12-bit accuracy — Lower 12 bits ([11:0]) are valid
   - Flush-right data with 10-bit accuracy — Lower 10 bits ([9:0]) are valid
   - Flush-right data with 8-bit accuracy — Lower 8 bits ([7:0]) are valid
   - Flush-left data with 12-bit accuracy — Upper 12 bits ([15:4]) are valid
   - Flush-left data with 10-bit accuracy — Upper 10 bits ([15:6]) are valid
   - Flush-left data with 8-bit accuracy — Upper 8 bits ([15:8]) are valid
2. When A/D-converted value addition mode is selected
   - Flush-right data with 12-bit accuracy — Lower 14 bits ([13:0]) or 16 bits ([15:0]) are valid
   - Flush-right data with 10-bit accuracy — Lower 12 bits ([11:0]) are valid
   - Flush-right data with 8-bit accuracy — Lower 10 bits ([9:0]) are valid
   - Flush-left data with 12-bit accuracy — Upper 14 bits ([15:2]) or 16 bits ([15:0]) are valid
   - Flush-left data with 10-bit accuracy — Upper 12 bits ([15:4]) are valid
   - Flush-left data with 8-bit accuracy — Upper 10 bits ([15:6]) are valid

Note: The number of extended bits for addition varies with the A/D conversion accuracy and the number of addition times. A 2-bit extension is up to 4 times conversion (3 times addition) when the A/D conversion accuracy is 8, 10, or 12 bits. A 4-bit extension is 16 times conversion (15 times addition) when the A/D conversion accuracy is 12 bits.

## 37.2.29 ADWINnLB : A/D Compare Function Window B Lower-Side/Upper-Side Level Setting Register (n = L, U)

Base address: ADC120 = 0x4017_0000

Offset address: 0x0A8 (n = L)
0x0AA (n = U)

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The ADWINULB and ADWINLLB registers specify the reference data when the compare window B function is used. ADWINLLB sets the lower reference for window B, and ADWINULB sets the upper reference for window B.

ADWINnLB are read/write registers.

ADWINnLB are writable even during A/D conversion. The reference data can be dynamically changed by rewriting register values during A/D conversion*1.

Set these registers so that the upper reference is not less than the lower reference (ADWINULB ≥ ADWINLLB). ADWINULB are not used when the window function is disabled.

Note 1. The lower and the upper references are changed when each register is written. For example, when the upper reference value is changed and the lower reference value is being changed, the MCU compares the upper reference (after rewrite), and the lower reference (before rewrite) with the A/D conversion result. See Figure 37.5. If the comparison during the rewriting of these two references is erroneous, then rewrite these reference values when

both ADCSR.ADST and the target Compare Window Operation Enable bit (ADCMPCR.CMPAE or ADCMPCR.CMPBE) are 0.

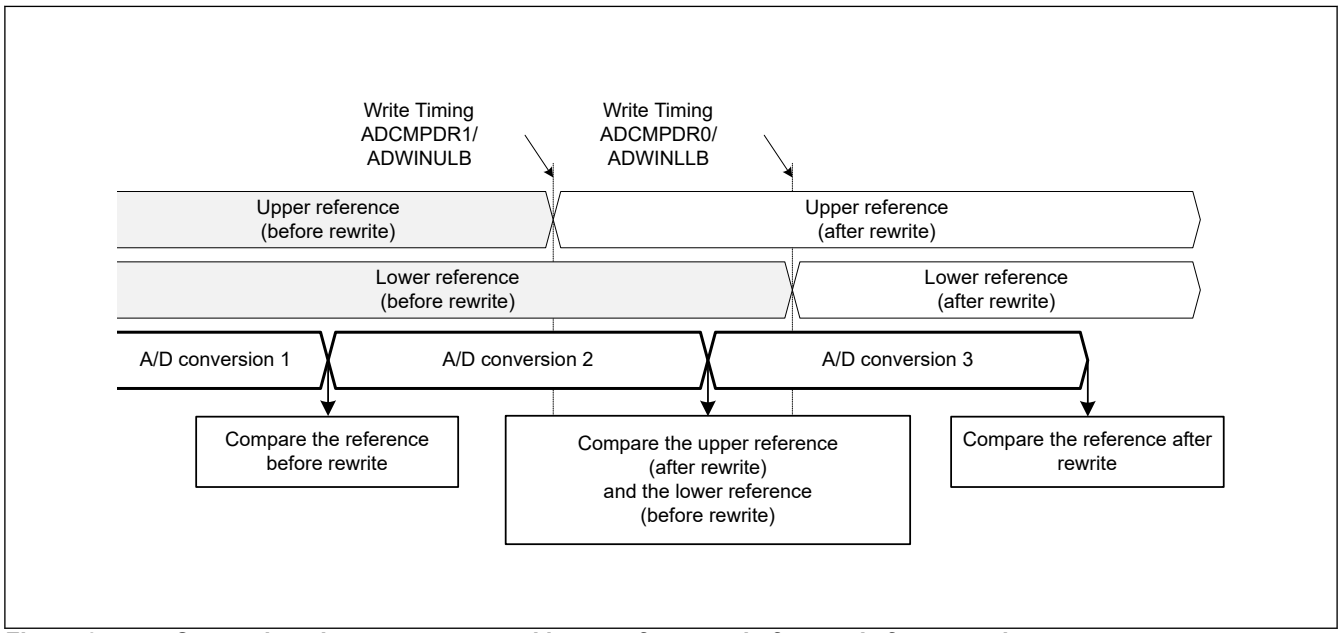



**Figure 37.5 Comparison between upper and lower references before and after a rewrite**

The ADWINnLB registers use different formats depending on the following conditions:

- The value of A/D Data Register Format Select bit (flush-right or flush-left)
- The value of the A/D Conversion Accuracy Select bit (12-bit, 10-bit, 8-bit)
- The value of A/D-Converted Value Addition/Average Channel Select bits (A/D-converted value addition mode selected or not selected).

The data formats for each condition are shown as follows:

1. When A/D-converted value addition mode is not selected
   - Flush-right data with 12-bit accuracy — Lower 12 bits ([11:0]) are valid
   - Flush-right data with 10-bit accuracy — Lower 10 bits ([9:0]) are valid
   - Flush-right data with 8-bit accuracy — Lower 8 bits ([7:0]) are valid
   - Flush-left data with 12-bit accuracy — Upper 12 bits ([15:4]) are valid
   - Flush-left data with 10-bit accuracy — Upper 10 bits ([15:6]) are valid
   - Flush-left data with 8-bit accuracy — Upper 8 bits ([15:8]) are valid
2. When A/D-converted value addition mode is selected
   - Flush-right data with 12-bit accuracy — Lower 14 bits ([13:0]) or 16 bits ([15:0]) are valid
   - Flush-right data with 10-bit accuracy — Lower 12 bits ([11:0]) are valid
   - Flush-right data with 8-bit accuracy — Lower 10 bits ([9:0]) are valid
   - Flush-left data with 12-bit accuracy — Upper 14 bits ([15:2]) or 16 bits ([15:0]) are valid
   - Flush-left data with 10-bit accuracy — Upper 12 bits ([15:4]) are valid
   - Flush-left data with 8-bit accuracy — Upper 10 bits ([15:6]) are valid

Note: The number of extended bits for addition varies with the A/D conversion accuracy and the number of addition times. A 2-bit extension is up to 4 times conversion (3 times addition) when the A/D conversion accuracy is 8, 10, or 12 bits. A 4-bit extension is 16 times conversion (15 times addition) when the A/D conversion accuracy is 12 bits.

## 37.2.30 ADCMPSR0 : A/D Compare Function Window A Channel Status Register 0

Base address: ADC120 = 0x4017_0000

Offset address: 0x0A0

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CMPSTCHA15 | CMPSTCHA14 | CMPSTCHA13 | CMPSTCHA12 | CMPSTCHA11 | CMPSTCHA10 | CMPSTCHA9 | CMPSTCHA8 | CMPSTCHA7 | CMPSTCHA6 | CMPSTCHA5 | CMPSTCHA4 | CMPSTCHA3 | CMPSTCHA2 | CMPSTCHA1 | CMPSTCHA0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | CMPSTCHA15 to CMPSTCHA0 | Compare Window A Flag<br>When Window A operation is enabled (ADCMPCR.CMPAE = 1b), these bits indicate the comparison result of channels to which Window A comparison conditions are applied.<br>Bit 15 (CMPSTCHA15) is associated with AN015 and bit 0 (CMPSTCHA0) is associated with AN000.<br>0: Comparison conditions are not met.<br>1: Comparison conditions are met. | R/W |

Note: n = 00 to 08, 11 to 13

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

### CMPSTCHAn flags (Compare Window A Flag)

The CMPSTCHAn flags indicate the comparison results for channels to which Window A comparison conditions are applied. When a comparison condition set in ADCMPLR0.CMPLCHA is met at the end of A/D conversion, the associated CMPSTCHAn flag sets to 1. When the ADCMPCR.CMPAIE bit is 1, a compare interrupt request (ADC120_CMPAI) is generated when this flag sets to 1.

Writing 1 to the CMPSTCHAn flags is invalid.

[Setting condition]

- The condition set in ADCMPLR0.CMPLCHA is met when ADCMPCR.CMPAE = 1.

[Clearing condition]

- Writing 0 after reading 1.

## 37.2.31 ADCMPSR1 : A/D Compare Function Window A Channel Status Register1

Base address: ADC120 = 0x4017_0000

Offset address: 0x0A2

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CMPSTCHA31 | CMPSTCHA30 | CMPSTCHA29 | CMPSTCHA28 | CMPSTCHA27 | CMPSTCHA26 | CMPSTCHA25 | CMPSTCHA24 | CMPSTCHA23 | CMPSTCHA22 | CMPSTCHA21 | CMPSTCHA20 | CMPSTCHA19 | CMPSTCHA18 | CMPSTCHA17 | CMPSTCHA16 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | CMPSTCHA31 to CMPSTCHA16 | Compare Window A Flag<br>When Window A operation is enabled (ADCMPCR.CMPAE = 1), these bits indicate the comparison result of channels to which Window A comparison conditions are applied.<br>Bit 15 (CMPSTCHA31) is associated with AN031 and bit 0 (CMPSTCHA16) is associated with AN016.<br>0: Comparison conditions are not met.<br>1: Comparison conditions are met. | R/W |

Note: n = 16

Note: Bits associated with non-existent pins are reserved. This bit is read as 0. The write value should be 0.

### CMPSTCHAn flags (Compare Window A Flag)

The CMPSTCHAn flags indicate the comparison results for channels to which Window A comparison conditions are applied. When the comparison condition set in ADCMPLR1.CMPLCHA is met at the end of A/D conversion, the associated CMPSTCHAn flag sets to 1. When the ADCMPCR.CMPAIE bit is 1, a compare interrupt request (ADC120_CMPAI) is generated when this flag sets to 1.

Writing 1 to the CMPSTCHAn flags is invalid.

[Setting condition]

- The condition set in ADCMPLR1.CMPLCHA is met when ADCMPCR.CMPAE = 1.

[Clearing condition]

- Writing 0 after reading 1.

## 37.2.32 ADCMPSER : A/D Compare Function Window A Extended Input Channel Status Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x0A4

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | CMPSTOCA | CMPSTTSA |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CMPSTTSA | Compare Window A Temperature Sensor Output Compare Flag<br>When Window A operation is enabled (ADCMPCR.CMPAE = 1), this bit indicates the temperature sensor output comparison result.<br>0: Comparison conditions are not met.<br>1: Comparison conditions are met. | R/W |
| 1 | CMPSTOCA | Compare Window A Internal Reference Voltage Compare Flag<br>When Window A operation is enabled (ADCMPCR.CMPAE = 1), this bit indicates the internal reference voltage comparison result.<br>0: Comparison conditions are not met.<br>1: Comparison conditions are met. | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

The ADCMPSER register stores compare results of compare function window A.

### CMPSTTSA flag (Compare Window A Temperature Sensor Output Compare Flag)

The CMPSTTSA flag indicates the temperature sensor output comparison result. When the comparison condition set in ADCMPLER.CMPLTSA is met at the end of A/D conversion, this flag sets to 1. When the ADCMPCR.CMPAIE bit is 1, a compare interrupt request (ADC120_CMPAI) is generated when this flag sets to 1.

Writing 1 to the CMPSTTSA flag is invalid.

[Setting condition]

- The condition set in ADCMPLER.CMPLTSA is met when ADCMPCR.CMPAE = 1.

[Clearing condition]

- Writing 0 after reading 1.

### CMPSTOCA flag (Compare Window A Internal Reference Voltage Compare Flag)

The CMPSTOCA flag indicates the internal reference voltage comparison result. When the comparison condition set in ADCMPLER.CMPLOCA is met at the end of A/D conversion, this flag sets to 1. When the ADCMPCR.CMPAIE bit is 1, a compare interrupt request (ADC120_CMPAI) is generated when this flag sets to 1.

Writing 1 to the CMPSTOCA flag is invalid.

[Setting condition]

- The condition set in ADCMPLER.CMPLOCA is met when ADCMPCR.CMPAE = 1.

[Clearing condition]

- Writing 0 after reading 1.

## 37.2.33 ADCMPBNSR : A/D Compare Function Window B Channel Select Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x0A6

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CMPLB | — | CMPCHB[5:0] | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | CMPCHB[5:0] | Compare Window B Channel Select<br>These bits select channels to be compared with the compare Window B conditions. | R/W |
| 6 | — | This bit is read as 0. The write value should be 0. | R/W |
| 7 | CMPLB | Compare Window B Comparison Condition Setting<br>This bit sets comparison conditions for channels for Window B. The comparison conditions are shown in Figure 37.6.<br>0: When window function is disabled (ADCMPCR.WCMPE = 0):<br>ADWINLLB value > A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1):<br>A/D-converted value < ADWINLLB value, or ADWINULB value < A/D-converted value<br>1: When window function is disabled (ADCMPCR.WCMPE = 0):<br>ADWINLLB value < A/D-converted value<br>When window function is enabled (ADCMPCR.WCMPE = 1):<br>ADWINLLB value < A/D-converted value < ADWINULB value | R/W |

| CMPCHB[5:0] | Unit 0 |
|---|---|
| 0x00 | AN000 |
| 0x01 | AN001 |
| 0x02 | AN002 |
| 0x03 | AN003 |
| ⋮ | ⋮ |
| 0x08 | AN008 |
| 0x0B | AN011 |
| 0x0C | AN012 |
| 0x0D | AN013 |
| 0x20 | Temparature sensor |
| 0x21 | Internal reference voltage |
| Others | Setting prohibited |

### CMPCHB[5:0] bits (Compare Window B Channel Select)

The CMPCHB[5:0] bits specify the channels to be compared with the compare Window B conditions from AN000 to AN008, AN011 to AN013, AN016, the temperature sensor, the internal reference voltage. The compare Window B function is enabled by specifying the hexadecimal number of the A/D conversion channel selected in the ADANSA0, ADANSA1, ADANSB0, ADANSB1 registers.

Set the CMPCHB[5:0] bits while the ADCSR.ADST bit is 0.

**CMPLB bit (Compare Window B Comparison Condition Setting)**

The CMPLB bit specifies the comparison conditions for channels for Window B. When the comparison result of an analog input meets the set condition, the associated ADCMPBSR.CMPSTB flag sets to 1 and a compare interrupt request (ADC120_CMPBI) is generated.

Compare conditions when the window function is disabled

| CMPLB = 0 | | CMPLB = 1 | |
|---|---|---|---|
| ADWINLLB value ≤ A/D converted value | Not met | ADWINLLB value < A/D converted value | Met |
| ADWINLLB value > A/D converted value | Met | ADWINLLB value ≥ A/D converted value | Not met |

Compare conditions when the window function is enabled

| CMPLB = 0 | |
|---|---|
| A/D converted value > ADWINULB value | Met |
| ADWINLLB value ≤ A/D converted value ≤ ADWINULB value | Not met |
| A/D converted value < ADWINLLB value | Met |

| CMPLB = 1 | |
|---|---|
| A/D converted value ≥ ADWINULB value | Not met |
| ADWINLLB value <A/D converted value < ADWINULB value | Met |
| A/D converted value ≤ ADWINLLB value | Not met |

**Figure 37.6 Explanation of compare conditions for compare function Window B**

## 37.2.34 ADCMPBSR : A/D Compare Function Window B Status Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x0AC

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | CMPSTB |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CMPSTB | Compare Window B Flag<br>When Window B operation is enabled (ADCMPCR.CMPBE = 1), this bit indicates the comparison result of channels to which Window B comparison conditions are applied,temperature sensor output, internal reference voltage.<br>0: Comparison conditions are not met.<br>1: Comparison conditions are met. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

### CMPSTB flag (Compare Window B Flag)

The CMPSTB flag indicates the comparison result of channels to which Window B comparison conditions are applied, the temperature sensor output, internal reference voltage. When the comparison condition set in ADCMPBNSR.CMPLB is met at the end of A/D conversion, this flag sets to 1. When the ADCMPCR.CMPBIE bit is 1, a compare interrupt request (ADC120_CMPBI) is generated when this flag sets to 1.

Writing 1 to the CMPSTB flag is invalid.

[Setting condition]

- The condition set in ADCMPBNSR.CMPLB is met when ADCMPCR.CMPBE = 1.

[Clearing condition]

- Writing 0 after reading 1.

## 37.2.35 ADWINMON : A/D Compare Function Window A/B Status Monitor Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x08C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | MONCMPB | MONCMPA | — | — | — | MONCOMB |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MONCOMB | Combination Result Monitor<br>This bit indicates the combination result. This bit is valid when both Window A and Window B operations are enabled.<br>0: Window A/B composite conditions are not met.<br>1: Window A/B composite conditions are met. | R |
| 3:1 | — | These bits are read as 0. | R |
| 4 | MONCMPA | Comparison Result Monitor A<br>0: Window A comparison conditions are not met.<br>1: Window A comparison conditions are met. | R |
| 5 | MONCMPB | Comparison Result Monitor B<br>0: Window B comparison conditions are not met.<br>1: Window B comparison conditions are met. | R |
| 7:6 | — | These bits are read as 0. | R |

### MONCOMB bit (Combination Result Monitor)

The read-only MONCOMB bit indicates the combined result of comparison condition results A and B based on the combination condition set in the ADCMPCR.CMPAB[1:0] bits.

[Setting condition]

- The combined result meets the combination condition set in the ADCMPCR.CMPAB[1:0] bits when ADCMPCR.CMPAE = 1 and ADCMPCR.CMPBE = 1.

[Clearing conditions]

- The combined result does not meet the combination condition set in the ADCMPCR.CMPAB[1:0] bits.
- ADCMPCR.CMPAE = 0 or ADCMPCR.CMPBE = 0.

#### MONCMPA bit (Comparison Result Monitor A)

The read-only MONCMPA bit is read as 1 when the A/D-converted value of the Window A target channel meets the condition set in ADCMPLR0/ADCMPLR1 and ADCMPLER. Otherwise, it is read as 0.

[Setting condition]

- The A/D-converted value meets the condition set in the ADCMPLR0/ADCMPLR1 and ADCMPLER registers when ADCMPCR.CMPAE = 1.

[Clearing conditions]

- The A/D-converted value does not meet the condition set in the ADCMPLR0/ADCMPLR1 and ADCMPLER registers when ADCMPCR.CMPAE = 1.
- ADCMPCR.CMPAE = 0 (automatically cleared when the ADCMPCR.CMPAE value changes from 1 to 0).

#### MONCMPB bit (Comparison Result Monitor B)

The read-only MONCMPB bit is read as 1 when the A/D-converted value of the Window B target channel meets the condition set in the ADCMPBNSR.CMPLB bit. Otherwise, it is read as 0.

[Setting condition]

- The A/D-converted value meets the condition set in ADCMPBNSR.CMPLB when ADCMPCR.CMPBE = 1.

[Clearing conditions]

- The A/D-converted value does not meet the condition set in ADCMPBNSR.CMPLB when ADCMPCR.CMPBE = 1.
- ADCMPCR.CMPBE = 0 (automatically cleared when the ADCMPCR.CMPBE value changes from 1 to 0).

### 37.2.36 ADBUFEN : A/D Data Buffer Enable Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x0D0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | BUFEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BUFEN | Data Buffer Enable<br>0: The data buffer is not used.<br>1: The data buffer is used. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

The ADBUFEN register sets whether to enable the data buffer.

#### BUFEN bit (Data Buffer Enable)

This bit enables the use of the data buffer.

When BUFEN = 1b, A/D conversion result (addition result) other than self-diagnosis result is stored in ADBUFn.

Disable the data storage operation (BUFEN = 0b) before reading ADBUFPTR.

Do not use the data buffer for data duplexing, or group scan.

## 37.2.37 ADBUFPTR : A/D Data Buffer Pointer Register

Base address: ADC120 = 0x4017_0000

Offset address: 0x0D2

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | PTROVF | BUFPTR[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | BUFPTR[3:0] | Data Buffer Pointer<br>These bits indicate the number of data buffer to which the next A/D converted data is transferred. | R/W |
| 4 | PTROVF | Pointer Overflow Flag<br>0: The data buffer pointer has not overflowed.<br>1: The data buffer pointer has overflowed. | R/W |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

ADBUFPTR is a register that indicates the data buffer pointer and overflow status.

### BUFPTR[3:0] bit (Data Buffer Pointer)

These bits indicate the number of data buffer to which the next A/D converted data is transferred.

When data has been transferred to data buffer 15, the pointer value becomes 0000b and the PTROVF bit is set to 1.

When the next data has been transferred, the data in data buffer 0 is overwritten.

Writing 0x00 to this register clears the value of these bits. Writing a value other than 0x00 is disabled.

### PTROVF bit (Pointer Overflow Flag)

This bit indicates whether the data buffer pointer has overflowed. This bit is set to 1 when the pointer value becomes 0000b (overflow).

Writing 0x00 to this register clears this bit value. Writing a value other than 0x00 is disabled.

## 37.2.38 ADBUFn : A/D Data Buffer Registers n (n = 0 to 15)

Base address: ADC120 = 0x4017_0000

Offset address: 0x0B0 + 0x2 × n (n = 0 to 15))

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | ADBUF[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | ADBUF[15:0] | Converted Value 15 to 0<br>Functions vary depending on the selected mode and accuracy. See Table 37.21 and Table 37.22. | R |

ADBUFn registers are 16-bit read-only registers that sequentially store all A/D conversion results. The automatic clear function is not applied to these registers.

ADBUFn settings are the same as the A/D data register format settings.

The following conditions determine the formats for data in the ADBUFn registers:

- Setting of the Register Format Select bit (ADCER.ADRFMT) (flush-left or flush-right)
- Setting of the Addition/Average Count Select bits (ADADC.ADC[2:0]) (1, 2, 3, 4, or 16 times)

- Setting of the Average Mode Enable bit (ADADC.AVEE) (addition or average).

This section describes the data formats for these conditions in different modes.

### (1) When A/D-converted value addition/average mode is not selected

Table 37.21 shows the bit assignment for each accuracy.

**Table 37.21 Bit assignment for each accuracy**

| Accuracy | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | These bits are read as 0. | | | | Converted Value 11 to 0: 12-bit A/D-converted value | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | Converted Value 11 to 0: 12-bit A/D-converted value | | | | | | | | | | | | These bits are read as 0. | | | |

### (2) When A/D-converted value average mode is selected

A/D-converted value average mode can be selected when 2 or 4 times is specified in the A/D-converted value addition mode. When A/D converted value average mode is selected, This register indicatesThese registers indicate the mean of A/D-converted values on a specific channel. The value is stored in the A/D data register based on the setting of the A/D Data Register Format Select bit in the same way as for normal A/D conversion.

### (3) When A/D-converted value addition mode is selected

For 12-bit, 10-bit, 8-bit accuracy, 1, 2, 3, or 4 times can be selected in the A/D-converted value addition mode. A/D conversion results are stored in the A/D data register as a 2-bit-extended value of the specified conversion accuracy.

For 12-bit accuracy, 16 times can also be selected in the A/D-converted value addition mode. In A/D-converted value addition mode, this register indicatesthese registers indicate the value that is obtained by adding A/D-converted values on a specific channel. A/D conversion results are stored in the A/D data register as a 4-bit-extended value of the specified conversion accuracy.

When A/D-converted value addition mode is selected, the value is stored in the A/D data register based on the settings of the A/D Data Register Format Select bits.

Table 37.22 shows the bit assignment for each accuracy.

**Table 37.22 Bit assignment for each accuracy when A/D-converted value addition mode is selected**

| Accuracy | | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right-justified data with 12-bit accuracy | When 16 conversion times is specified | Added Value 15 to 0: 16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 1, 2, 3, or 4 conversion times is specified | These bits are read as 0. | Added Value 13 to 0: 14-bit sum of A/D conversion results | | | | | | | | | | | | | | |
| Left-justified data with 12-bit accuracy | When 1, 2, 3, or 4 conversion times is specified | Added Value 15 to 0: 16-bit sum of A/D conversion results | | | | | | | | | | | | | | | |
| | When 16 conversion times is specified | Added Value 13 to 0: 14-bit sum of A/D conversion results | | | | | | | | | | | | | | These bits are read as 0. | |

## 37.3 Operation

### 37.3.1 Scanning Operation

In scanning, A/D conversion is performed sequentially on the analog inputs of the specified channels.

Scan conversion is performed in any of the three operating modes:

- Single scan mode
- Continuous scan mode

- Group scan mode

In single scan mode, one or more specified channels are scanned once. In continuous scan mode, one or more specified channels are scanned repeatedly until software sets the ADCSR.ADST bit to 0. In group scan mode, the selected channels in group A, B are scanned once after scan starts in response to the respective synchronous trigger.

In single scan mode and continuous scan mode, A/D conversion is performed for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n. In group scan mode, A/D conversion is performed for the ANn channels in group A selected in the ADANSA0 and ADANSA1 registers, and for the ANn channels in group B selected in the ADANSB0 and ADANSB1 registers, starting from the channel with the smallest number n.

When self-diagnosis is selected, it is executed once at the beginning of each scan and one of the three reference voltages is converted.

The temperature sensor output and internal reference voltage can be selected at the same time as the analog input of the channels, and A/D conversion is performed on the analog input of channels, temperature sensor output, and internal reference voltage, in that order.

Double trigger mode can be used with single scan mode or group scan mode. With double trigger mode enabled (ADCSR.DBLE = 1), A/D conversion data of a channel selected in the ADCSR.DBLANS[4:0] bits is duplicated only if the conversion is started by the synchronous trigger (ELC) selected in the ADSTRGR.TRSA[5:0] bits. In group scan mode, only group A can use double trigger mode.

In the extended operation of double trigger mode, the A/D conversion operation is generated from the synchronous trigger combination selected in the ADSTRGR.TRSA[5:0] bits. In addition to normal double trigger mode operation, A/ D conversion data with odd number trigger (ELC_AD00) is stored in A/D Data Duplexing Register A (ADDBLDRA), and A/D conversion data with even number trigger (ELC_AD01) is stored in A/D Data Duplexing Register B (ADDBLDRB). In the extended operation of double trigger mode, when one of the trigger combinations occurs at the same time, the data duplexing register settings for the specified triggers do not work, and A/D conversion data is stored in A/D Data Duplexing Register B (ADDBLDRB).

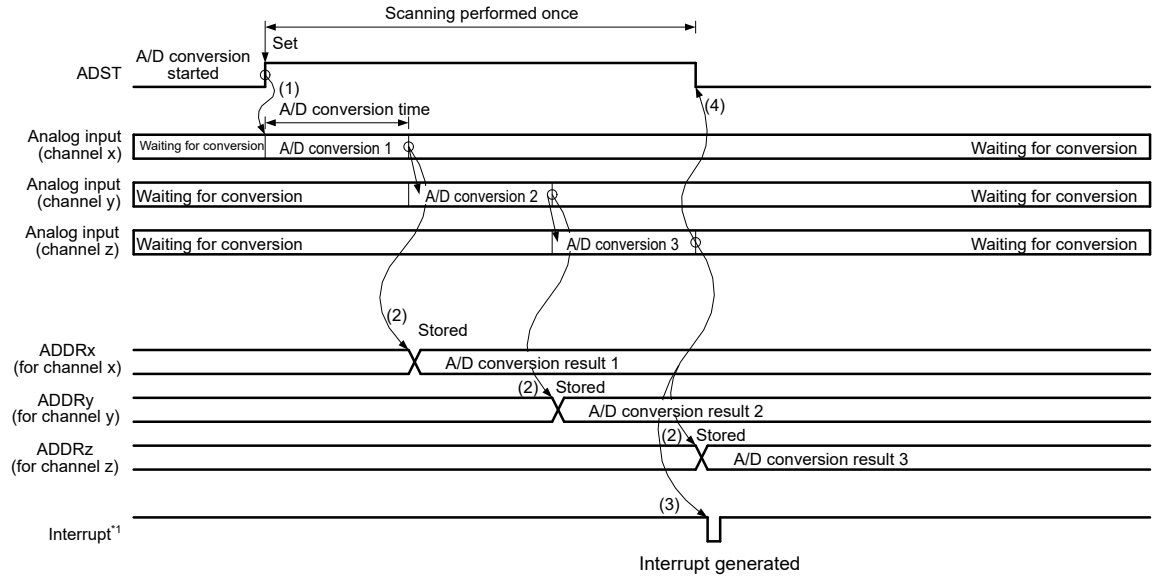
The ADC12 ignores a synchronous trigger that occurs during the A/D conversion started by another synchronous trigger.

## 37.3.2 Single Scan Mode

### 37.3.2.1 Basic Operation

In basic operation of single scan mode, A/D conversion is performed once on the analog input of the specified channels as follows:

1. When the ADCSR.ADST bit is set to 1 (A/D conversion start) by a software trigger, a synchronous trigger input (ELC), or an asynchronous trigger input, A/D conversion is performed for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.
2. Each time A/D conversion of a single channel is completed, the A/D conversion result is stored in the associated A/D data register (ADDRy).
3. When A/D conversion of all the selected channels is completed, an ADC120_ADI interrupt request is generated .
4. The ADST bit remains 1 (A/D conversion start) during A/D conversion and is automatically set to 0 when A/D conversion of all the selected channels is completed. The ADC12 then enters a wait state.



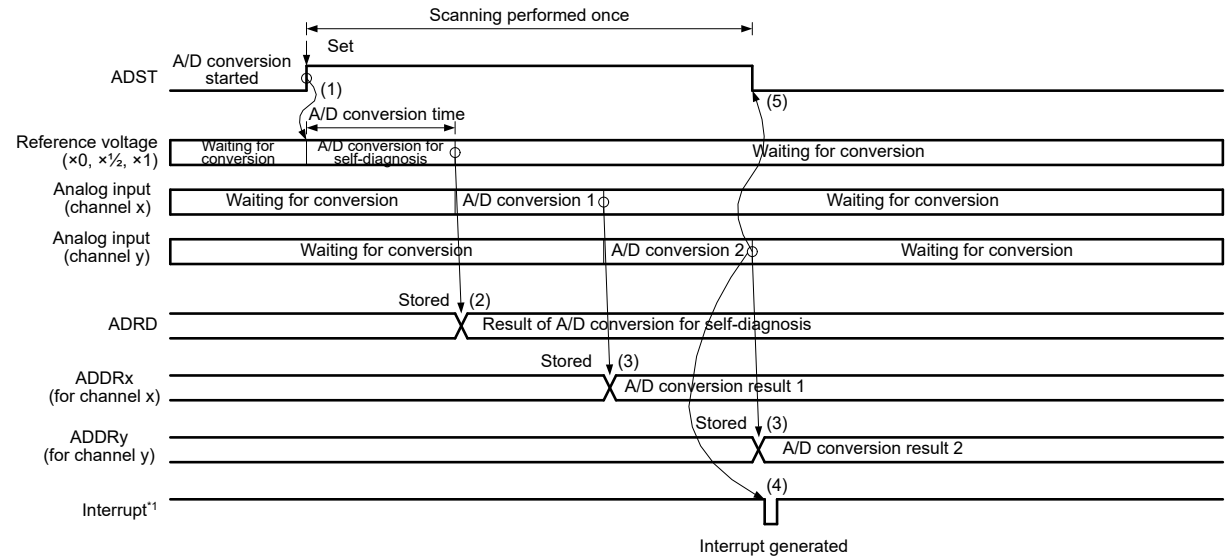

Note: x, y, and z (x < y < z) are the selected channel numbers by ADANSA0/1.
Note 1. ADC120_ADI

**Figure 37.7 Example basic operation in single scan mode when the analog inputs (channel x to z) are selected**

### 37.3.2.2 Channel Selection and Self-Diagnosis

When channels and self-diagnosis are selected, A/D conversion is first performed for the reference voltage (×0, ×1/2, or ×1) , then A/D conversion is performed once on the analog input of the selected channels as follows:

1. A/D conversion for self-diagnosis is first started when the ADCSR.ADST bit is set to 1 (A/D conversion start) by a software trigger input, a synchronous trigger input (ELC), or an asynchronous trigger input.
2. When A/D conversion for self-diagnosis is completed, the A/D conversion result is stored in the A/D Self-Diagnosis Data Register (ADRD). A/D conversion is then performed for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.
3. Each time A/D conversion of a single channel is completed, the A/D conversion result is stored in the associated A/D data register (ADDRy).
4. When A/D conversion of all the selected channels is completed, an ADC120_ADI interrupt request is generated .
5. The ADCSR.ADST bit remains 1 (A/D conversion start) during A/D conversion and is automatically set to 0 when A/D conversion of all the selected channels is completed. The ADC12 then enters a wait state.




Note: x and y (x < y) are the selected channel numbers by ADANSA0/1.
Note 1. ADC120_ADI

**Figure 37.8 Example basic operation in single scan mode when the analog inputs (channel x and y) are selected with self- diagnosis**

### 37.3.2.3 A/D Conversion of Temperature Sensor Output or Internal Reference Voltage

When the channels and temperature sensor output or internal reference voltage are selected at the same time, A/D conversion is performed first on the analog input of the selected channels, and once on the temperature sensor output or internal reference voltage. When both temperature sensor output and internal reference voltage are selected, A/D conversion of the temperature sensor output and internal reference voltage is performed, in that order. With the channels deselected, selecting only the temperature sensor output or internal reference voltage is also possible.

The operation is as follows:

1. When a software trigger, synchronous trigger (ELC), or asynchronous trigger sets the ADCSR.ADST bit to 1 (A/D conversion start), A/D conversion is performed for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.
2. On completion of A/D conversion on the channels, the result is stored in the associated A/D Data Register y (ADDRy), and then A/D conversion of the temperature sensor output starts.
3. On completion of A/D conversion of the temperature sensor output, the result is stored in the associated A/D Temperature Sensor Data Register (ADTSDR), and then A/D conversion of the internal reference voltage starts.
4. On completion of A/D conversion of the internal reference voltage, the result is stored in the associated A/D Internal Reference Voltage Data Register (ADOCDR), and an ADC120_ADI interrupt request is generated (no register setting).
5. The ADCSR.ADST bit remains 1 (A/D conversion start) during A/D conversion, and is automatically cleared to 0 on completion of A/D conversion. Then, the ADC12 enters a wait state.

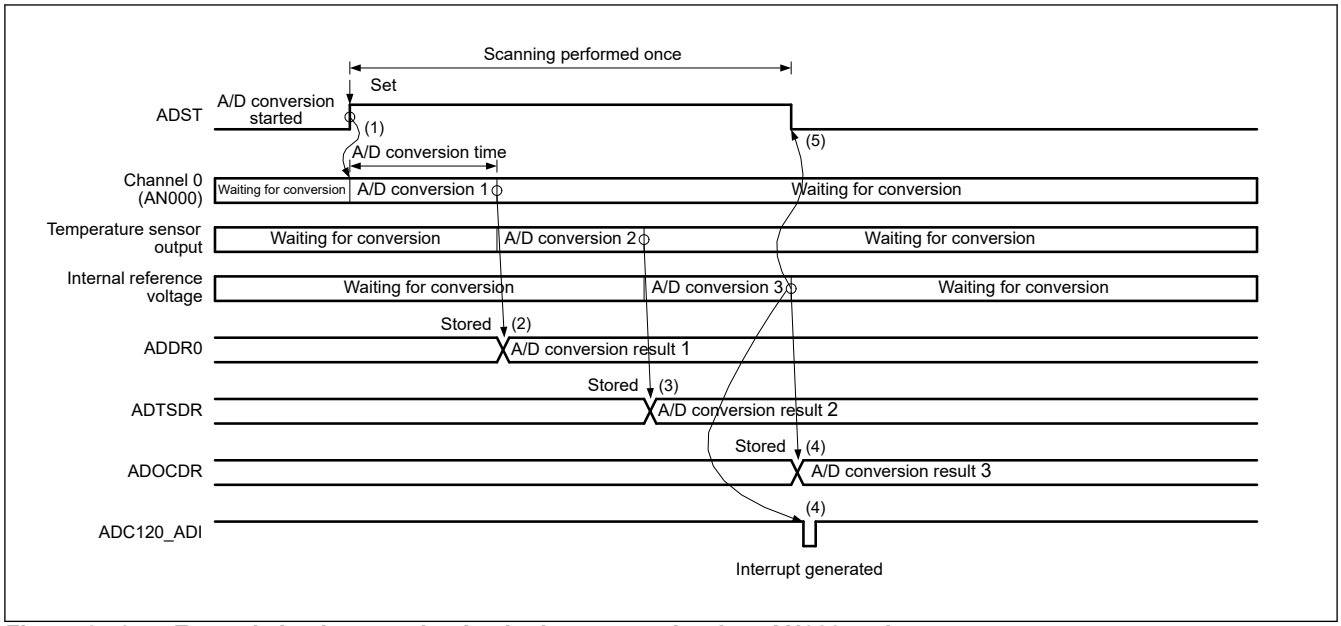



**Figure 37.9 Example basic operation in single scan mode when AN000 and temperature sensor output or internal reference voltage are selected**

### 37.3.2.4 A/D Conversion in Double-Trigger Mode

When double trigger mode is selected in single scan mode, two rounds of single scan operation started by a synchronous trigger (ELC) are performed in sequence.

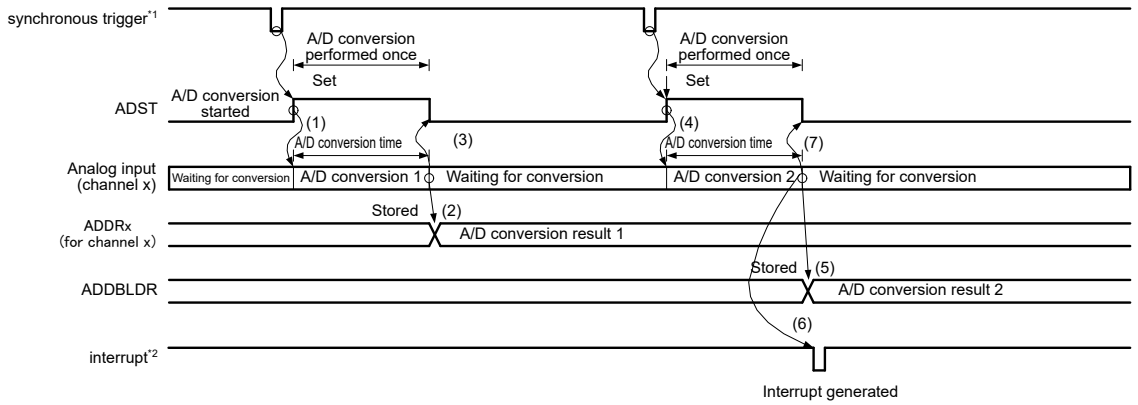
Deselect self-diagnosis and set the temperature sensor output A/D conversion select bit (ADEXICR.TSSA and ADEXICR.TSSB) and the internal reference voltage A/D conversion select bit (ADEXICR.OCSA and ADEXICR.OCSB) to 0.

Duplication of A/D conversion data is enabled by setting the channel numbers to be duplicated in the ADCSR.DBLANS[4:0] bits and setting the ADCSR.DBLE bit to 1. When the ADCSR.DBLE bit is set to 1, channel selection using the ADANSA0 and ADANSA1 registers is invalid.

In double trigger mode, select a synchronous trigger (ELC) with the ADSTRGR.TRSA[5:0] bits. Additionally, set the ADCSR.EXTRG bit to 0 and the ADCSR.TRGE bit to 1. Do not use a software trigger.

The operation is as follows:

1. When the ADCSR.ADST bit is set to 1 (A/D conversion start) by a synchronous trigger input (ELC), A/D conversion starts on the single channel selected in the ADCSR.DBLANS[4:0] bits.
2. Each time A/D conversion of a single channel is completed, the A/D conversion result is stored in the associated A/D Data Register y (ADDRy).
3. The ADCSR.ADST bit is automatically set to 0 and the ADC12 enters a wait state. An ADC120_ADI interrupt request is not generated .
4. When the ADCSR.ADST bit is set to 1 (A/D conversion start) by the second trigger input, A/D conversion starts on the single channel selected in the ADCSR.DBLANS[4:0] bits.
5. When A/D conversion is completed, the result is stored in the A/D Data Duplexing Register (ADDBLDR), which is exclusively used in double-trigger mode.
6. An ADC120_ADI interrupt request is generated.
7. The ADCSR.ADST bit remains 1 (A/D conversion start) during A/D conversion and is automatically set to 0 when A/D conversion is completed. Then the ADC12 enters a wait state.




Note: x is the selected channel number by ADANSA0/1.
Note 1. ELC_AD00/ELC_AD01
Note 2. ADC120_ADI

**Figure 37.10 Example operation in single scan mode when double-trigger mode is selected and the analog input (channel x) is duplicated**

### 37.3.2.5 Extended Operations When Double-Trigger Mode Is Selected

When double trigger mode is selected in single scan mode, and a synchronous trigger (ELC_AD00/ELC_AD01) is selected as the trigger for the start of A/D conversion, two rounds of single scan operation are performed.

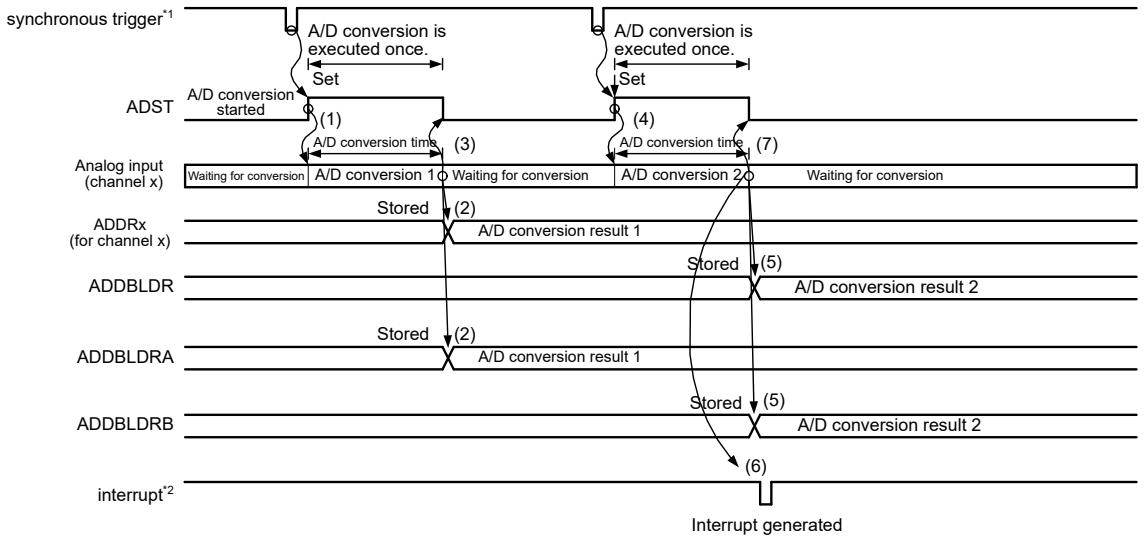
Deselect self-diagnosis and set the temperature sensor output A/D conversion select bit (ADEXICR.TSSA and ADEXICR.TSSB), and the internal reference voltage A/D conversion select bit (ADEXICR.OCSA and ADEXICR.OCSB) to 0.

Duplication of A/D conversion data is enabled by setting the channel number to be duplicated to the ADCSR.DBLANS[4:0] bits and setting the ADCSR.DBLE bit to 1. When the ADCSR.DBLE bit is set to 1, channel selection using the ADANSA0 and ADANSA1 registers is invalid.

In extended double trigger mode, select a synchronous trigger combination ELC_AD00/ELC_AD01 by setting the ADSTRGR.TRSA[5:0] bits to 0x0B, set the ADCSR.EXTRG bit to 0, and set the ADCSR.TRGE bit to 1. Do not use a software trigger.

The operation is as follows:

1. When the ADCSR.ADST bit is set to 1 (A/D conversion start) by a synchronous trigger input (ELC_AD00/ ELC_AD01), A/D conversion starts on the single channel selected in the ADCSR.DBLANS[4:0] bits.
2. When A/D conversion completes, the A/D conversion result is stored in the associated A/D Data Register (ADDRy) and in A/D Data Duplexing Register A (ADDBLDRA) or A/D Data Duplexing Register B (ADDBLDRB) when the ELC_ADi0 or ELC_ADi1 trigger is input respectively (i = 0).
3. The ADCSR.ADST bit is automatically set to 0 and the ADC12 enters a wait state. An ADC120_ADI interrupt request is not generated.
4. When the ADCSR.ADST bit is set to 1 (A/D conversion start) by the second trigger (ELC_AD00/ELC_AD01), A/ D conversion starts on the single channel selected in the ADCSR.DBLANS[4:0] bits.
5. When A/D conversion completes, the A/D conversion result is stored in the A/D Data Duplexing Register (ADDBLDR) and in A/D Data Duplexing Register A (ADDBLDRA) or A/D Data Duplexing Register B (ADDBLDRB) when the ELC_ADi0 or ELC_ADi1 trigger is input respectively (i = 0).
6. An ADC120_ADI interrupt request is generated.
7. The ADCSR.ADST bit remains 1 (A/D conversion start) during A/D conversion and is automatically set to 0 when A/D conversion completes. The ADC12 then enters a wait state.




Note: x is the selected channel number by ADANSA0/1.
Note 1. ELC_AD00/ELC_AD01
Note 2. ADC120_ADI

**Figure 37.11 Example extended operation in double trigger mode with duplication selected for the analog input (channel x) and ELC_AD00/ELC_AD01**

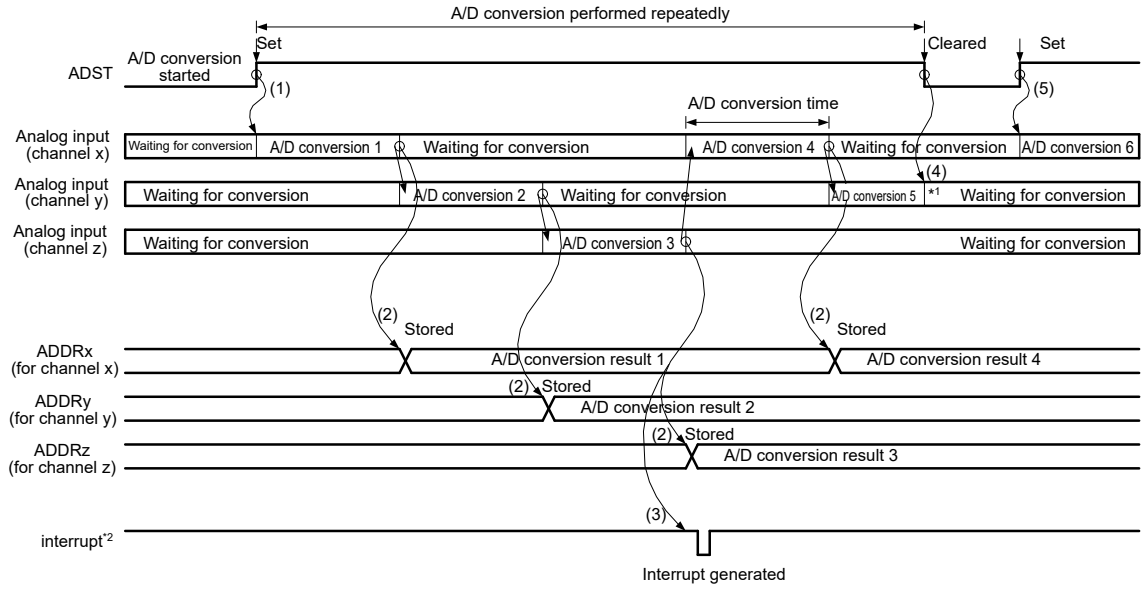
## 37.3.3 Continuous Scan Mode

### 37.3.3.1 Basic Operation

In continuous scan mode, A/D conversion is performed repeatedly on the analog input of the specified channels.

The operation is as follows:

1. When the ADCSR.ADST bit is set to 1 (A/D conversion start) by a software trigger, a synchronous trigger input (ELC), or an asynchronous trigger input, A/D conversion is performed for ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.
2. Each time A/D conversion of a single channel is completed, the A/D conversion result is stored in the associated A/D Data Register (ADDRy).
3. When A/D conversion of all the selected channels is completed, an ADC120_ADI interrupt request is generated. The ADC12 sequentially starts A/D conversion for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.
4. The ADCSR.ADST bit is not automatically cleared, and steps 2. and 3. are repeated as long as ADCSR.ADST remains 1 (A/ D conversion start). When the ADCSR.ADST bit is set to 0 (A/D conversion stop), A/D conversion stops and the ADC12 enters a wait state.
5. When the ADCSR.ADST bit is later set to 1 (A/D conversion start), A/D conversion starts again for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.




Note: x, y, and z (x < y < z) are the selected channel numbers by ADANSA0/1.
Note 1. Data for A/D conversion 5 is ignored.
Note 2. ADC120_ADI

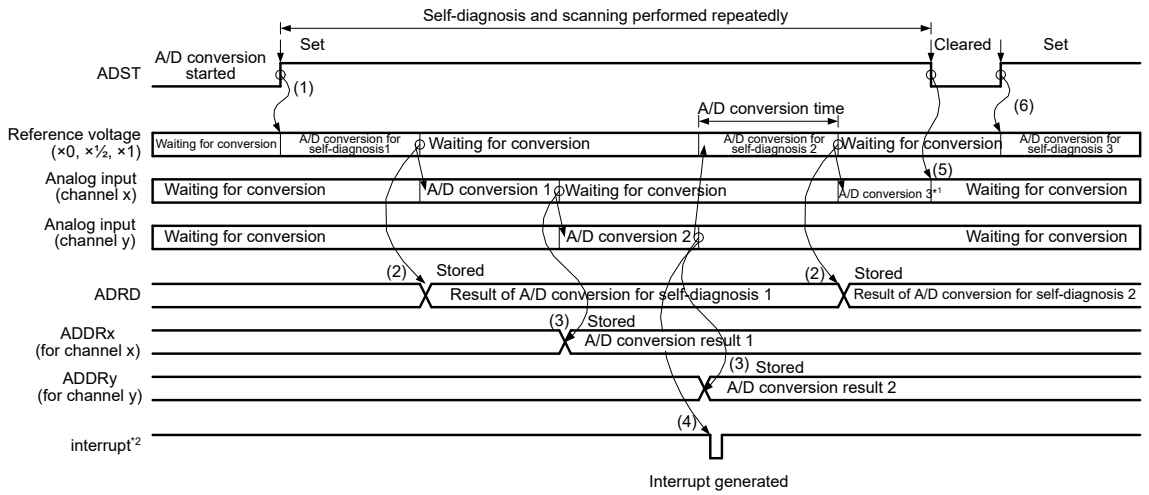
**Figure 37.12 Example basic operation in continuous scan mode when the analog inputs (channel x to z) are selected**

### 37.3.3.2 Channel Selection and Self-Diagnosis

When channels and self-diagnosis are selected at the same time, A/D conversion is first performed for the reference voltage (×0, ×1/2, or ×1) supplied to the ADC12, and A/D conversion is performed on the analog input of the selected channels. This sequence is repeated as described in the section that follows.

The operation is as follows:

1. A/D conversion for self-diagnosis is first started when the ADCSR.ADST bit is set to 1 (A/D conversion start) by a software trigger input, a synchronous trigger input (ELC), or an asynchronous trigger input.
2. When A/D conversion for self-diagnosis is completed, the A/D conversion result is stored in the A/D Self-Diagnosis Data Register (ADRD). A/D conversion is then performed for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.
3. Each time A/D conversion of a single channel is completed, the A/D conversion result is stored in the corresponding A/D Data Register (ADDRy).
4. When A/D conversion of all the selected channels is completed, an ADC120_ADI interrupt request is generated. At the same time, the ADC12 starts A/D conversion for self-diagnosis and then on the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.
5. The ADCSR.ADST bit is not automatically cleared, and steps 2. to 4. are repeated as long as the ADCSR.ADST bit remains 1. When the ADST bit is set to 0 (A/D conversion stop), A/D conversion stops and the ADC12 enters a wait state.
6. When the ADST bit is later set to 1 (A/D conversion start), the A/D conversion for self-diagnosis is started again.




Note: x and y (x < y) are the selected channel numbers by ADANSA0/1.
Note 1. Data for A/D conversion 3 is ignored.
Note 2. ADC120_ADI

**Figure 37.13 Example basic operation in continuous scan mode when the analog inputs (channel x and y) are selected with self-diagnosis**

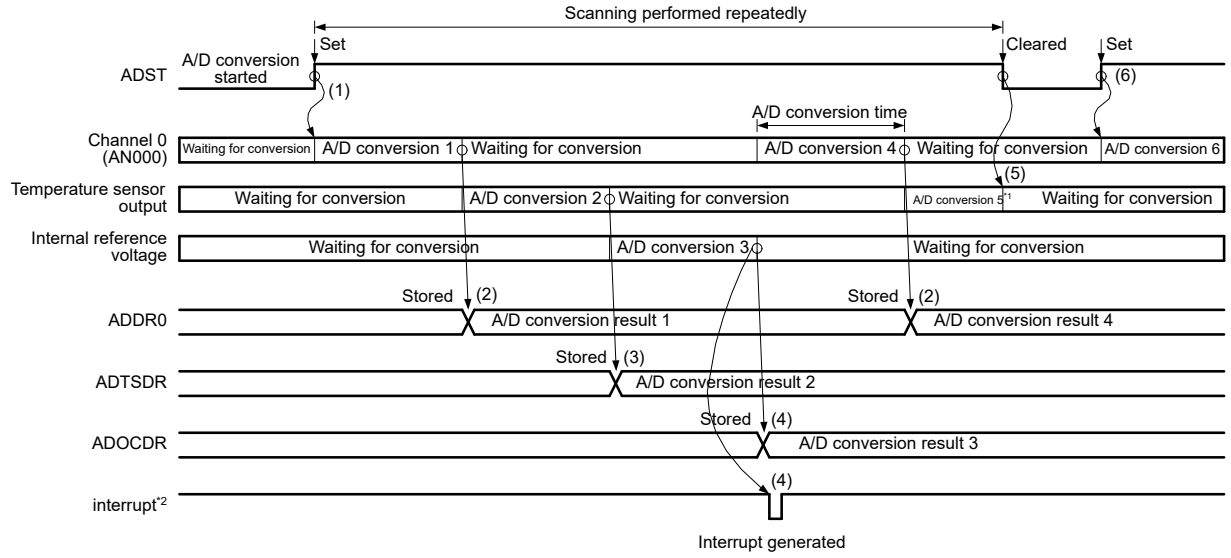
### 37.3.3.3 A/D Conversion of Temperature Sensor Output or Internal Reference Voltage

When the channels and temperature sensor output or internal reference voltage are selected at the same time, A/D conversion is first performed on the analog input of the selected channels, and then the A/D conversion of the temperature sensor output or internal reference voltage is repeated. When both the temperature sensor output and internal reference voltage are selected, A/D conversion of the temperature sensor output and internal reference voltage is performed, in that order.

With the channels deselected, selecting only the temperature sensor output or internal reference voltage is also possible.

The operation is as follows:

1. When a software trigger, synchronous trigger (ELC), or asynchronous trigger sets the ADCSR.ADST bit to 1 (A/D conversion start), A/D conversion is performed for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the smallest number n.
2. On completion of A/D conversion on the channels, the result is stored in the associated A/D Data Register y (ADDRy), and then A/D conversion of temperature sensor output starts.
3. On completion of A/D conversion of the temperature sensor output, the result is stored in the associated A/D Temperature Sensor Data Register (ADTSDR), and then A/D conversion of the internal reference voltage starts.
4. On completion of A/D conversion of the internal reference voltage, the result is stored in the associated A/D Internal Reference Voltage Data Register (ADOCDR), and an ADC120_ADI interrupt request is generated. In addition, the ADC12 continuously starts A/D conversion for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the lowest number n.
5. The ADCSR.ADST bit is not cleared automatically, and steps 2 to 4 are repeated as long as this bit remains set to 1. When the ADCSR.ADST bit is set to 0 (A/D conversion stop), A/D conversion stops and the ADC12 enters a wait state.
6. When the ADCSR.ADST bit is then set to 1 (A/D conversion start), A/D conversion starts again for the ANn channels selected in the ADANSA0 and ADANSA1 registers, starting from the channel with the lowest number n.




Note 1. Data for A/D conversion 5 is ignored.
Note 2. ADC120_ADI

**Figure 37.14 Example basic operation in continuous scan mode when AN000 and temperature sensor output or internal reference voltage are selected**

## 37.3.4 Group Scan Mode

### 37.3.4.1 Basic Operation

In group scan mode, A/D conversion is performed once on the analog input of all the specified channels in group A and B after scanning is started by a synchronous trigger (ELC). The scan operation of each group is similar to the scan operation in single scan mode.

The synchronous triggers can be selected in the ADSTRGR.TRSA[5:0] bits for group A and in the ADSTRGR.TRSB[5:0] bits for group B. Use different triggers for group A and B to prevent simultaneous A/D conversion of the two groups. Do not use a software trigger.

The group A channels to be A/D-converted are selected using the ADANSA0 and ADANSA1 registers and the ADEXICR.TSSA and OCSA bits. The group B channels to be A/D-converted are selected using the ADANSB0 and ADANSB1 registers and the ADEXICR.TSSB and OCSB bits. Group A and B cannot use the same channels.

When self-diagnosis is selected in group scan mode, self-diagnosis is separately executed for Group A and B.

The following sequence describes operation in group scan mode using a synchronous trigger from the ELC. In this example, the ELC_AD00 trigger from the ELC is used to start conversion of group A and the ELC_AD01 trigger from the ELC is used to start conversion of group B. In addition, ELC_AD00 and ELC_AD01 are selected for the GPT event in the associated ELC.ELSRn registers.

The operation is as follows:

1. Scanning of group A is started by ELC_AD00.
2. When group A scanning completes, an ADC120_ADI interrupt is generated (no register setting).
3. Scanning of group B is started by ELC_AD01.
4. When group B scanning completes, an ADC120_GBADI interrupt is generated if the ADCSR.GBADIE bit is 1 (ADC120_GBADI interrupt when scanning completion is enabled).

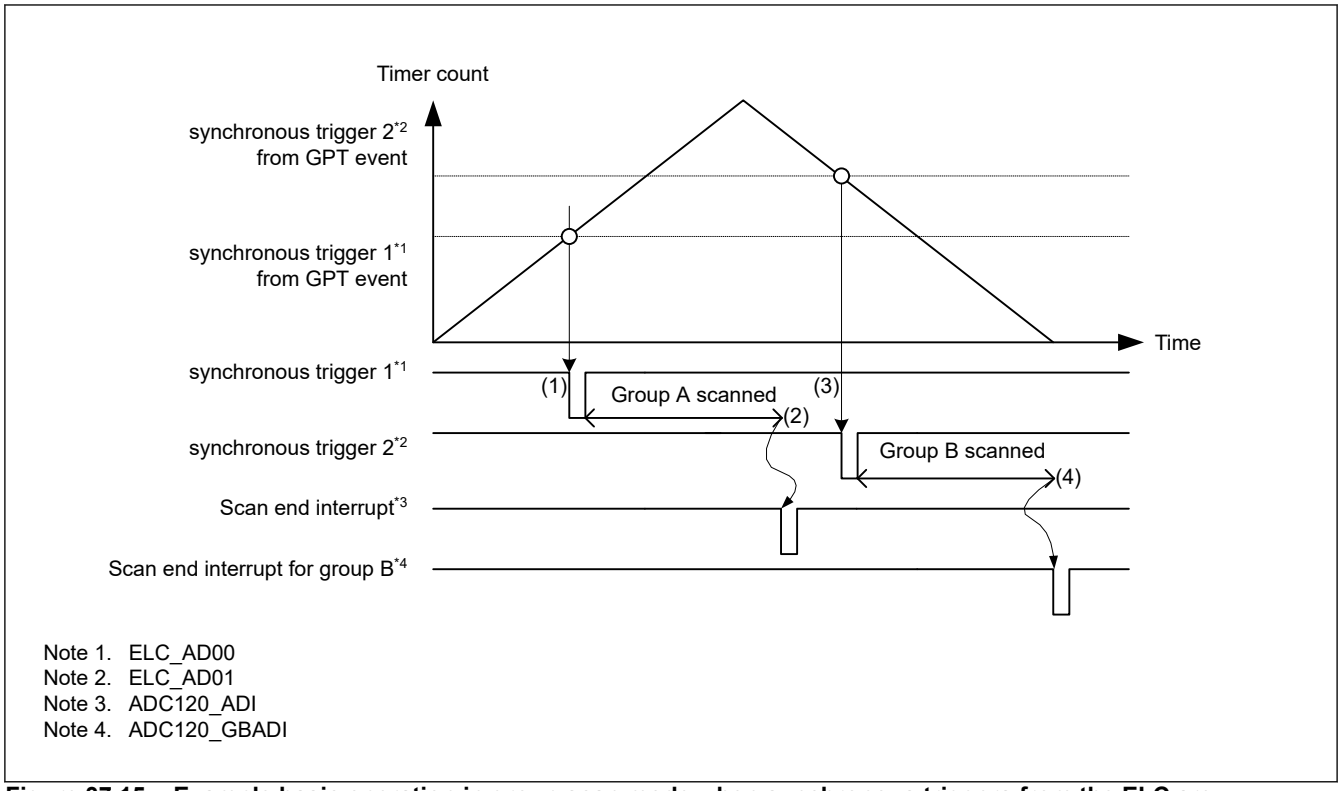



**Figure 37.15 Example basic operation in group scan mode when synchronous triggers from the ELC are used**

### 37.3.4.2 A/D Conversion in Double-Trigger Mode

When double trigger mode is selected in group scan mode, two rounds of single scan operation started by a synchronous trigger (ELC) are performed as a sequence for group A. For group B, single scan operation started by a synchronous trigger (ELC) is performed once.

In group scan mode, the synchronous trigger can be selected in the ADSTRGR.TRSA[5:0] bits for group A and in the ADSTRGR.TRSB[5:0] bits for group B. Use different triggers for groupA, B to prevent simultaneous A/D conversion of the two groups. Do not use a software trigger or an asynchronous trigger.

When an ELC_AD00/ELC_AD01 is selected as group A synchronous triggers by setting the ADSTRGR.TRSA[5:0] bits to 0x0B, operation proceeds in extended double trigger mode.

The group A channel to be A/D-converted is selected using the DBLANS[4:0] bits in the ADCSR register, while the group B channels to be A/D-converted are selected using the ADANSB0 and ADANSB1 registers. Group A, B cannot use the same channels.

When double-trigger mode is selected in group scan mode, set the A/D conversion select bits for both the temperature sensor output (ADEXICR.TSSA) and the internal reference voltage (ADEXICR.OCSA) to 0 (deselected).

Self-diagnosis cannot be selected when double trigger mode is selected in group scan mode.

Duplication of A/D conversion data is enabled by setting the channel numbers to be duplicated in the ADCSR.DBLANS[4:0] bits and setting the ADCSR.DBLE bit to 1.

The following sequence describes operation in group scan mode with double trigger mode selected and using a synchronous trigger from the ELC. In this example, the ELC_AD00 trigger is used to start conversion of group A and the ELC_AD01 trigger is used to start conversion of group B. In addition, ELC_AD00 and ELC_AD01 are selected for the GPT event in the associated ELC.ELSRn registers.

The operation is as follows:

1. Scanning of group B is started by the ELC_AD00 trigger from the ELC.
2. When group B scanning completes, an ADC120_GBADI interrupt is generated if the GBADIE bit in ADCSR is 1 (ADC120_GBADI interrupt when scanning completion is enabled).
3. The first scan of group A is started by the first ELC_AD01 trigger.

4. When the first scan of group A completes, the conversion result is stored in the associated A/D Data Register y (ADDRy); an ADC120_ADI interrupt request is not generated.
5. The second scan of group A is started by the second ELC_AD01 trigger.
6. When the second scan of group A completes, the conversion result is stored in ADDBLDR. An ADC120_ADI interrupt is generated.

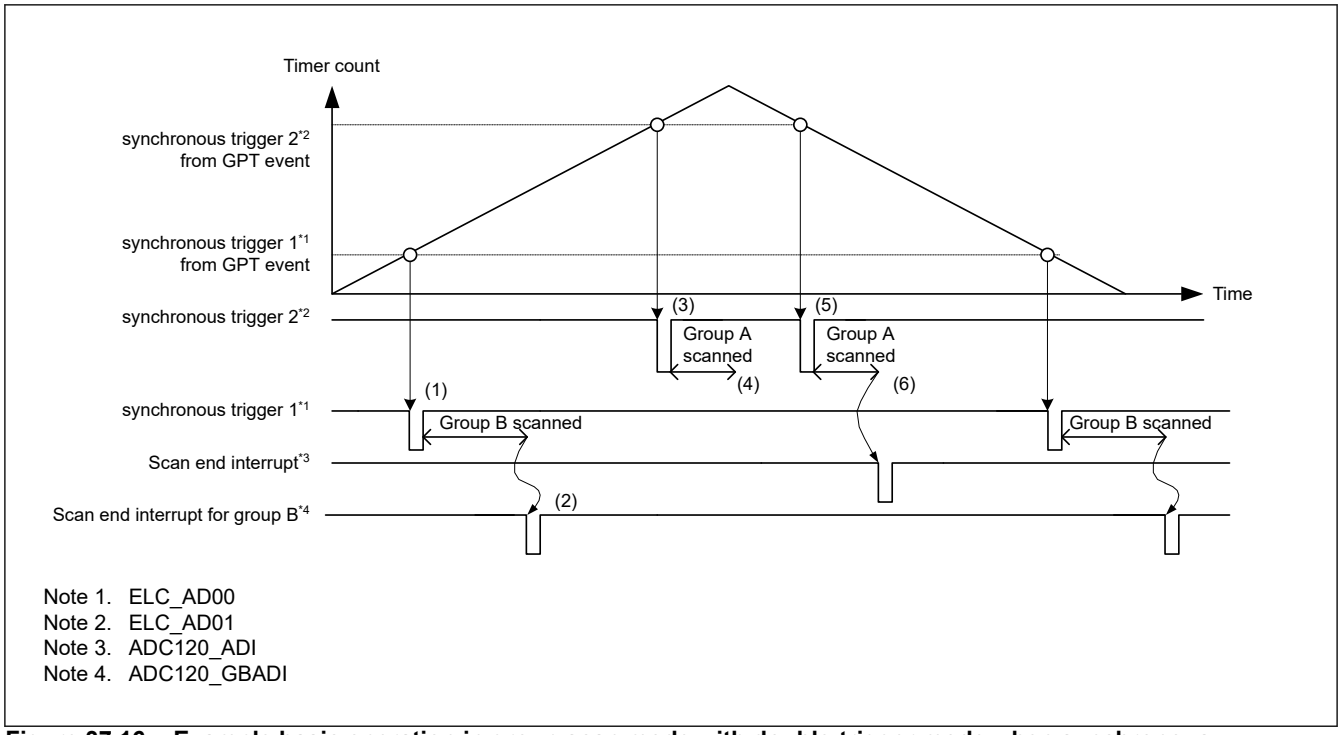



**Figure 37.16 Example basic operation in group scan mode with double-trigger mode when synchronous triggers from the ELC are used**

### 37.3.4.3 Group Priority Operation

Group priority operation is performed by setting the ADGSPCR.PGS bit to 1 in group-scan mode. The priority of groups is group A > group B.

When setting the PGS bit in the ADGSPCR register to 1, follow the procedure described in Figure 37.17. If the procedure is not followed, A/D conversion operation and stored data are not guaranteed.

As the basic operation in group-scan mode, a trigger input generated during A/D conversion of group A, B is ignored, and the A/D conversion operation of each group is similar to the operation in single-scan mode.

In group priority operation, if a trigger for a priority group is input during scanning of a lower-priority group, A/D conversion for the lower-priority group is stopped and A/D conversion for the priority group is performed.

If the setting of the ADGSPCR.GBRSCN bit is 0, the lower-priority group enters a wait state when A/D conversion for the priority group completes. A trigger input of the lower-priority group generated during A/D conversion is ignored.

If the setting of the ADGSPCR.GBRSCN bit is 1, A/D conversion for the lower-priority group automatically restarts upon completion of A/D conversion for the priority group. A trigger input of the lower-priority group generated during A/D conversion on the priority group takes effect, and A/D conversion for the lower-priority group is automatically performed upon completion of A/D conversion on the priority group.

If the ADGSPCR.GBRSCN bit is 1 and the ADGSPCR.LGRRS bit is 0, A/D conversion for the lower-priority group is restarted from the first channel. If the setting of the ADGSPCR.LGRRS bit is 1, A/D conversion for the lower-priority group is restarted from the channel for which the conversion stopped. However, if the self-diagnosis function is used, the A/D conversion is restarted from the channel for which the conversion stopped after self-diagnosis completed.

Table 37.23 summarizes operations in response to the input of a trigger during A/D conversion with the settings of the ADGSPCR.GBRSCN bit.

If the setting of the ADGSPCR.GBRP bit is 1, A/D conversion operation for the lowest-priority group is to continuously perform single scans.

For the trigger settings in group-scan mode, select a synchronous trigger for group A by using the ADSTRGR.TRSA[5:0] bits, a synchronous trigger for group B by using the ADSTRGR.TRSB[5:0] bits. Each trigger must be different from each other. Set the ADSTRGR.TRSB[5:0] bits to 0x3F when setting the ADGSPCR.GBRP bit to 1.

The channels to be scanned must be selected in the registers shown in section 37.3.4. Group Scan Mode.

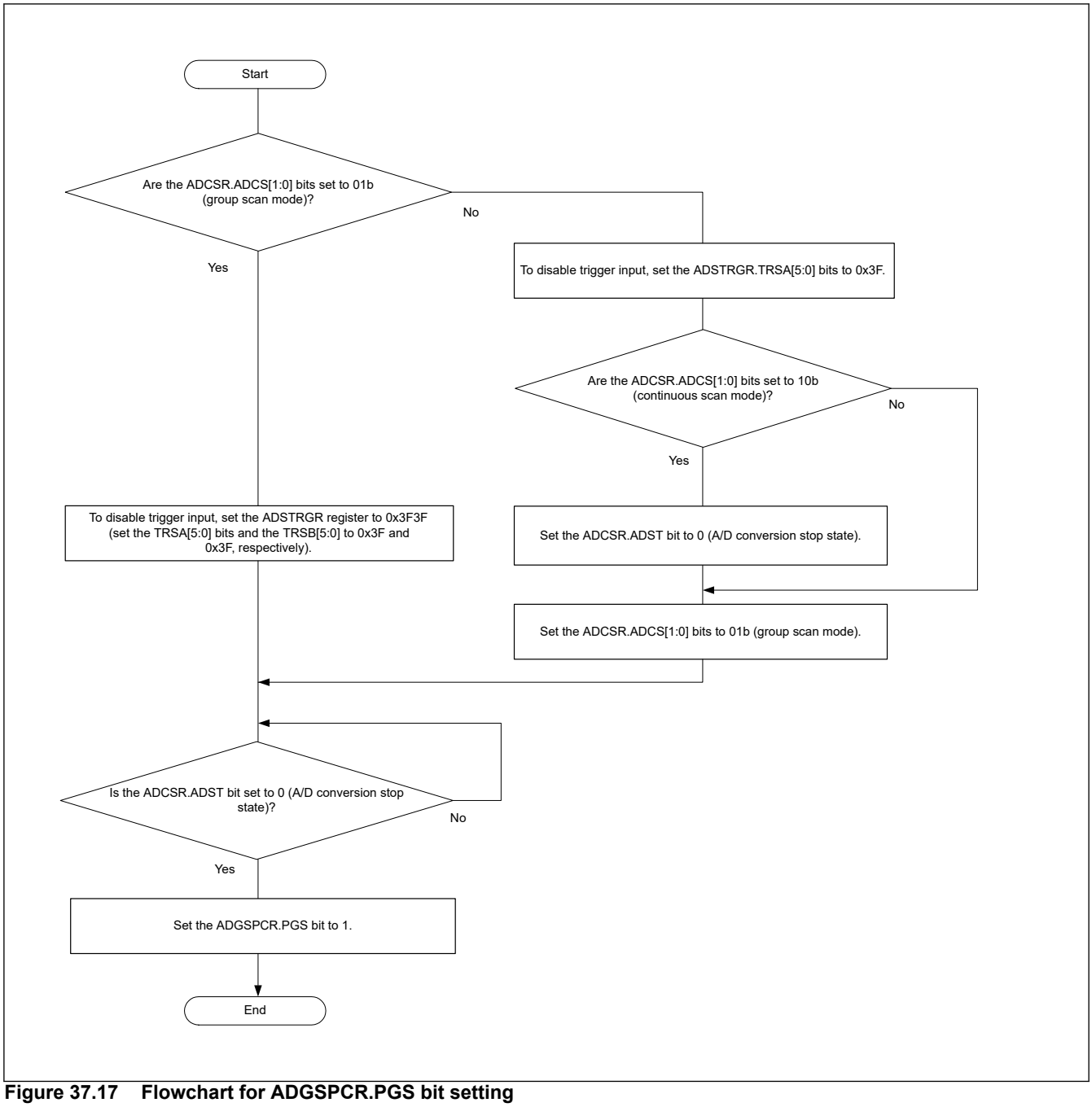



**Figure 37.17 Flowchart for ADGSPCR.PGS bit setting**

**Table 37.23 Control of A/D conversion operations according to ADGSPCR.GBRSCN bit setting**

| A/D conversion operation | Trigger input | ADGSPCR.GBRSCN = 0 | ADGSPCR.GBRSCN = 1 |
|---|---|---|---|
| When A/D conversion for group A is in progress | Input of trigger for group A | Trigger input is ineffective. | Trigger input is ineffective. |
| | Input of trigger for group B | Trigger input is ineffective. | A/D conversion for group B is performed after A/D conversion for group A completes. |
| When A/D conversion for group B is in progress | Input of trigger for group A | A/D conversion for group B is discontinued and A/D conversion for group A starts. | • A/D conversion for group B is discontinued and A/D conversion for group A starts.<br>• A/D conversion for group B starts after A/D conversion for group A completes. |
| | Input of trigger for group B | Trigger input is ineffective. | Trigger input is ineffective. |

To use group priority operation mode, select the operation mode to be implemented and set the registers according to the following table.

**Table 37.24 Group priority operation setting and operation mode for two groups (ADGSPCR.PGS = 1)**

| ADGSPCR | | | Operation category |
|---|---|---|---|
| GBRSCN | LGRRS | GBRP | |
| 0 | x | 0 | Group priority operation for two groups (groups A and B)<br>• When a trigger of group A is input, A/D conversion for group B is terminated (and will not be restarted). |
| 1 | 0 | 0 | Group priority operation for two groups (groups A and B)<br>• After A/D conversion for group B stopped, when A/D conversion for group A completes, A/D conversion for the group B channels selected in the ADANSB0 and ADANSB1 registers restarts according to the conversion order of smaller channel number. |
| 1 | 1 | 0 | Group priority operation for two groups (groups A and B)<br>• After A/D conversion for group B stopped, when A/D conversion for group A completes, A/D conversion for the group B channels selected in the ADANSB0/1 register restarts according to the conversion order of smaller channel number, beginning from the channel for which A/D conversion stopped.*1 |
| x | 0 | 1 | Group priority operation for two groups (groups A and B)<br>• Single scanning for group B is continuously performed without a start trigger input. After A/D conversion for group B stopped, when A/D conversion for group A completes, single scanning for the channels selected in the ADANSB0/1 register restarts according to the conversion order of smaller channel number. |
| 1 | 1 | 1 | Group priority operation for two groups (groups A and B)<br>• Single scanning for group B is continuously performed without a start trigger input. After A/D conversion for group B stopped, when A/D conversion for group A completes, single scanning for the channels selected in the ADANSB0/1 register restarts according to the conversion order of smaller channel number, beginning from the channel for which A/D conversion stopped.*1 |

Note: x: Don't care.
Note 1. When the self-diagnosis function is enabled (ADCER.DIAGM = 1), A/D conversion for the channel that has been stopped is started after self-diagnosis is performed.

### (1) Group priority operation for two groups (when ADGSPCR.PGS = 1)

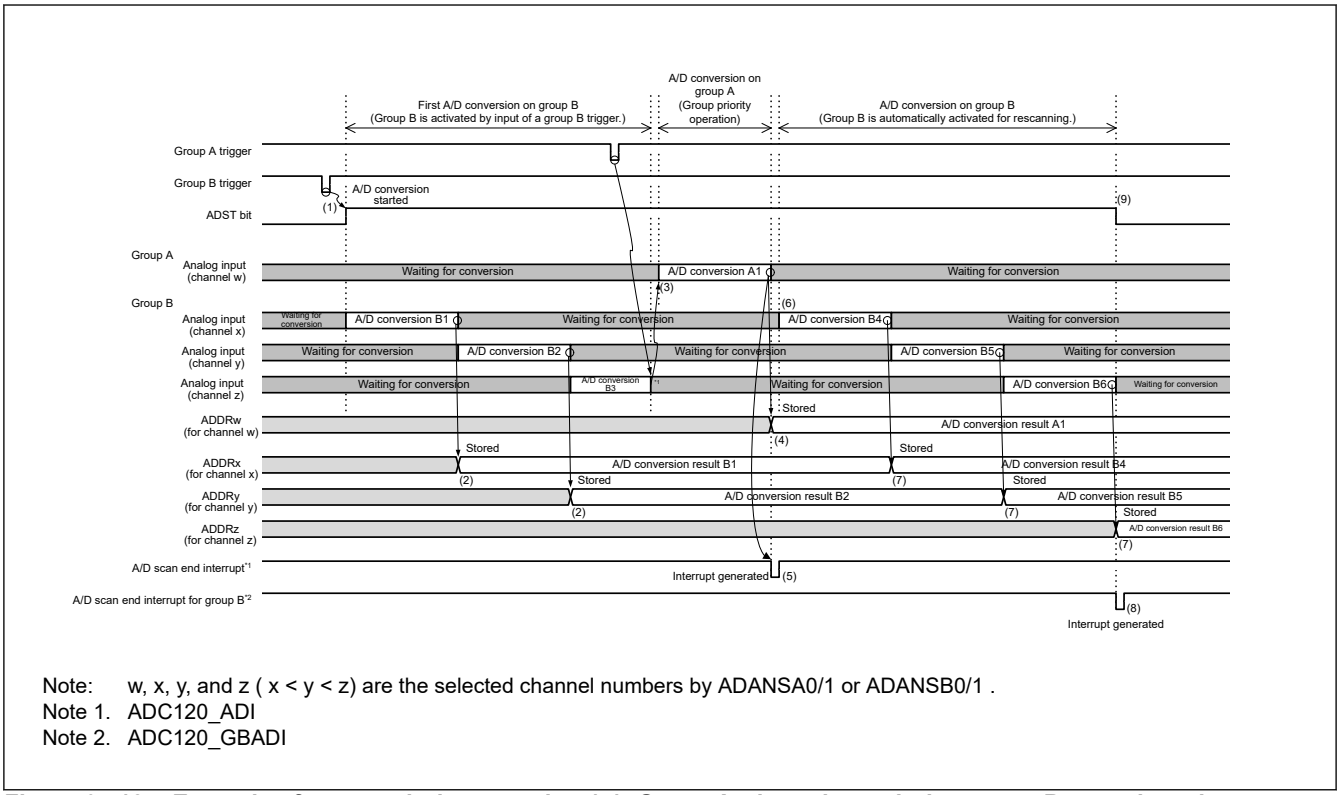
Operation examples 1-1 to 1-3 show group priority operations in group-scan mode (when ADGSPCR.GBRSCN = 1, ADGSPCR.GBRP = 0, and ADGSPCR.LGRRS = 0).

#### Operation example 1-1: "Group A trigger input during group B scan" when rescanning is enabled

1. When input of a trigger for group B sets the ADCSR.ADST bit to 1 (starting A/D conversion), A/D conversion for the analog input channels selected in the ADANSB0 and ADANSB1 registers starts according to the conversion order from the channel with the smallest number n.
2. On completion of A/D conversion for each channel in group B, the result is stored in the corresponding A/D Data Register y (ADDRy).
3. When a trigger for group A is input during A/D conversion for group B, A/D conversion for group B stops while the ADCSR.ADST bit remains 1. Then A/D conversion for the group A analog input channels selected in the ADANSA0

and ADANSA1 registers starts according to the conversion order from the channel with the smallest number n. If A/D conversion stops before it is completed, the conversion result is not stored in the A/D Data Register y (ADDRy).

4. On completion of A/D conversion on the channels, the result is stored in the corresponding A/D Data Register y (ADDRy).
5. An ADC120_ADI interrupt request is generated.
6. If the setting of the ADGSPCR.GBRSCN bit is 1 (enabling rescanning of the group that was stopped in group priority operation), A/D conversion for the group B analog input channels selected in the ADANSB0 and ADANSB1 registers restarts according to the conversion order from the channel with the smallest number n while the ADCSR.ADST remains 1.
7. On completion of A/D conversion on the channels, the result is stored in the corresponding A/D Data Register y (ADDRy).
8. If the setting of the ADCSR.GBADIE bit is 1 (enabling interrupt generation on completion of group B scan), a group B scan end interrupt request is generated.
9. When A/D conversion for all the channels completes, the ADCSR.ADST bit is automatically cleared and the A/D converter enters a wait state.




Note: w, x, y, and z ( x < y < z) are the selected channel numbers by ADANSA0/1 or ADANSB0/1 .
Note 1. ADC120_ADI
Note 2. ADC120_GBADI

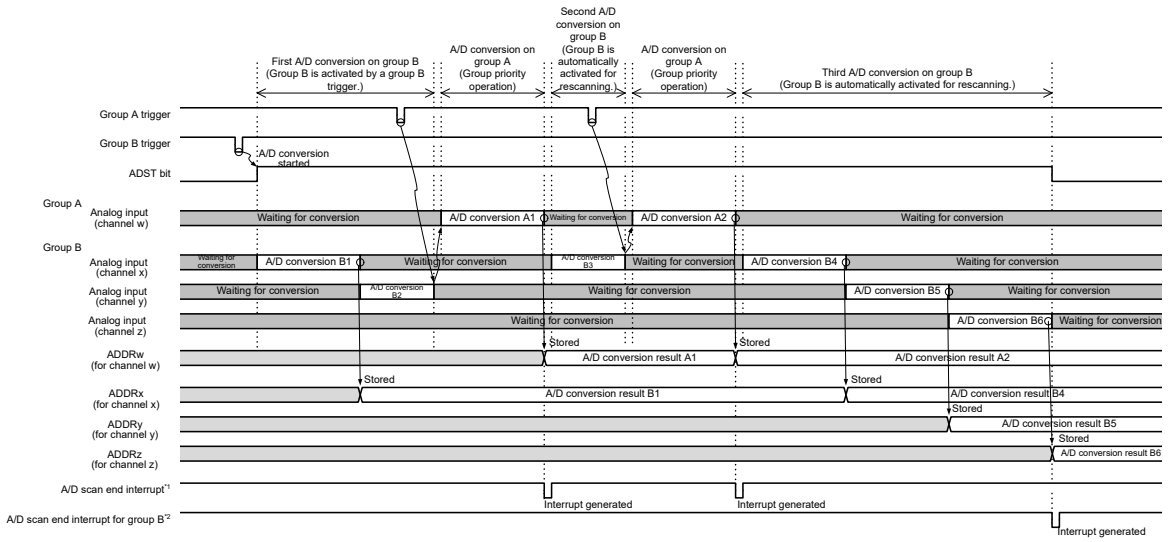
**Figure 37.18 Example of group priority operation 1-1: Group A trigger input during group B scanning when rescanning is enabled (when ADGSPCR.GBRSCN = 1, ADGSPCR.GBRP = 0, and ADGSPCR.LGRRS = 0)**

### Operation example 1-2: “Group A trigger input during rescanning of group B” when rescanning is enabled

Figure 37.19 shows the operation when a group A trigger is input during rescanning operation for group B.

Even during rescanning operation, when a trigger for group A is input, A/D conversion on group B stops and A/D conversion for group A starts. A/D conversion for group B starts after A/D conversion for group A completes.

Operations for setting the ADCSR.ADST bit, storing the A/D conversion result in the corresponding A/D Data Register y (ADDRy), and generating interrupt requests are the same as those in operation example 1-1.




Note: w, x, y, and z (x < y < z) are the selected channel numbers by ADANSA0/1 or ADANSB0/1.
Note 1. ADC120_ADI
Note 2. ADC120_GBADI

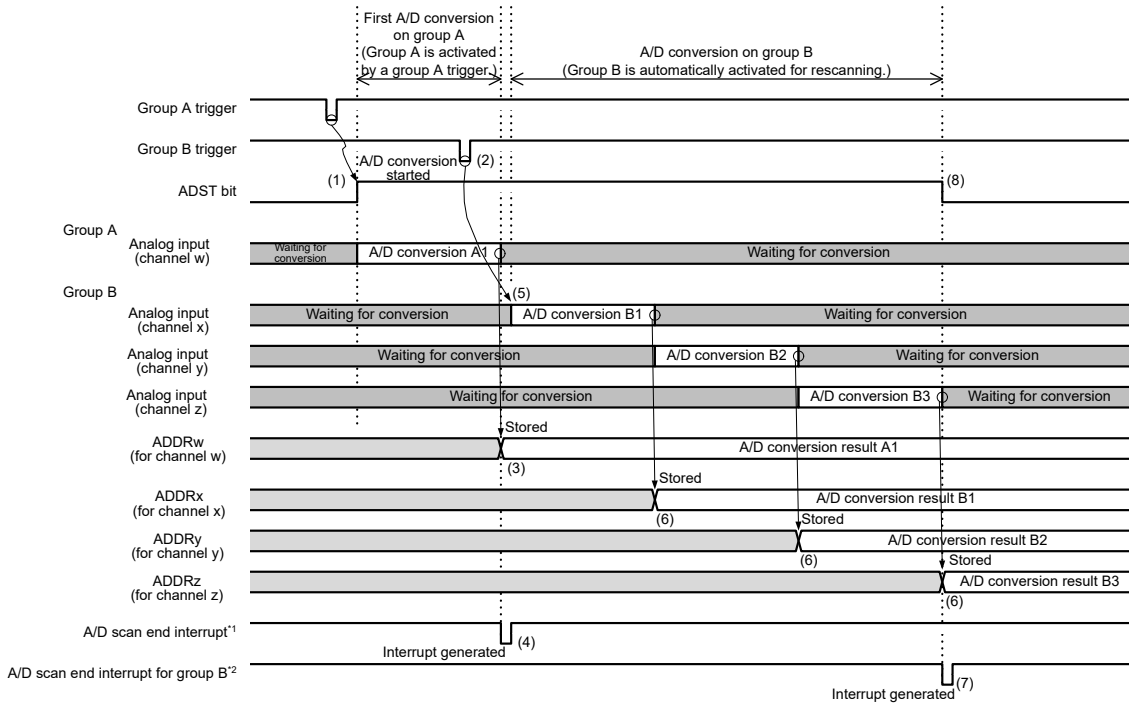
**Figure 37.19 Example of group priority operation 1-2: Group A trigger input during rescanning of group B when rescanning is enabled (when ADGSPCR.GBRSCN = 1, ADGSPCR.GBRP = 0, and ADGSPCR.LGRRS = 0)**

### Operation example 1-3: “Group B trigger input during group A scan” when rescanning is enabled

The following describes the operation when the setting of the ADGSPCR.GBRSCN bit is 1 (enabling rescanning of the group that was stopped in group priority operation) and a trigger for group B is input during scanning operation for group A.

If the setting of the ADGSPCR.GBRSCN bit is 0, any trigger for group B that is input during scanning operation for group A is invalid.

1. When input of a trigger for group A sets the ADCSR.ADST bit to 1 (starting A/D conversion), A/D conversion for the group A analog input channels selected in the ADANSA0 and ADANSA1 registers starts according to the conversion order from the channel with the smallest number n.
2. When a trigger for group B is input during A/D conversion for group A, group B is ready for A/D conversion.
3. On completion of A/D conversion for each channel in group A, the result is stored in the corresponding A/D Data Register y (ADDRy).
4. An ADC120_ADI interrupt request is generated.
5. When A/D conversion for group A completes, while the ADCSR.ADST bit remains 1, A/D conversion for the group B analog input channels selected in the ADANSB0 and ADANSB1 registers starts according to the conversion order from the channel with the smallest number n.
   (As with the case of operation example 1-1, if a trigger for group A is input during A/D conversion for group B, A/D conversion for group A starts. Then A/D conversion for group B starts upon completion of A/D conversion for group A.)
6. On completion of A/D conversion of a single channel, the result is stored in the corresponding A/D Data Register y (ADDRy).
7. Upon completion of A/D conversion for group B, a group B scan end interrupt request is generated if the setting of the ADCSR.GBADIE bit is 1 (enabling interrupt generation on completion of group B scan).
8. When A/D conversion for all the channels completes, the ADCSR.ADST bit is automatically cleared and the A/D converter enters a wait state.



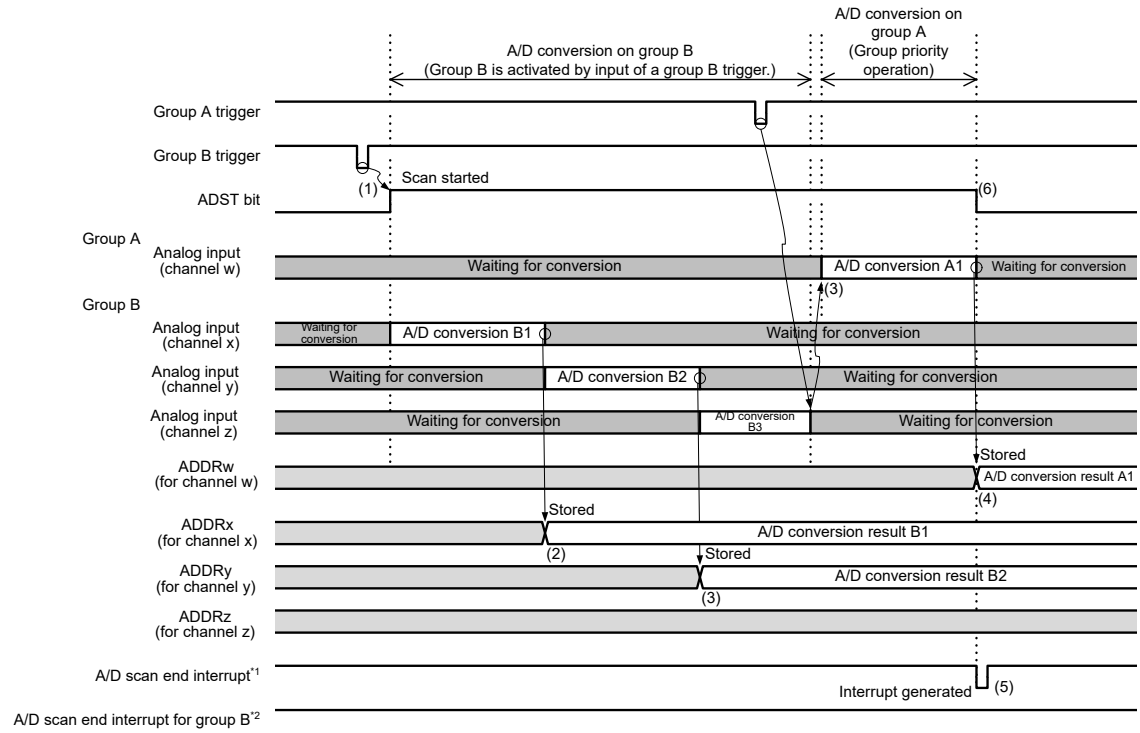

Note: w, x, y, and z (x < y < z) are the selected channel numbers by ADANSA0/1 or ADANSB0/1.
Note 1. ADC120_ADI
Note 2. ADC120_GBADI

**Figure 37.20 Example of group priority operation 1-3: Group B trigger input during group A scan when rescanning is enabled (when ADGSPCR.GBRSCN = 1, ADGSPCR.GBRP = 0, and ADGSPCR.LGRRS = 0)**

Operation example 1-4 shows the group priority operation in group-scan mode (when ADGSPCR.GBRSCN = 0, ADGSPCR.GBRP = 0, and ADGSPCR.LGRRS = 0).

### Operation example 1-4: “Group A trigger input during group B scan” when rescanning is disabled

1. When input of a trigger for group B sets the ADCSR.ADST bit to 1 (starting A/D conversion), A/D conversion for the analog input channels selected in the ADANSB0 and ADANSB1 registers starts according to the conversion order from the channel with the smallest number n.
2. On completion of A/D conversion for each channel in group B, the result is stored in the corresponding A/D Data Register y (ADDRy).
3. When a trigger for group A is input during A/D conversion for group B, A/D conversion for group B stops while the ADCSR.ADST bit remains 1, and then A/D conversion for the group A analog input channels selected in the ADANSA0 and ADANSA1 registers starts according to the conversion order from the channel with the smallest number n. If A/D conversion stops before it is completed, the conversion result is not stored in the A/D Data Register y (ADDRy).
4. On completion of A/D conversion of a single channel, the result is stored in the corresponding A/D Data Register y (ADDRy).
5. On completion of A/D conversion for group A, an ADC120_ADI interrupt request is generated.
6. When A/D conversion for group A completes, the ADCSR.ADST bit is automatically cleared and the A/D converter enters a wait state. A/D conversion for group B is not performed until a trigger for group B is input the next time.




Note: w, x, y, and z (x < y < z) are the selected channel numbers by ADANSA0/1 or ADANSB0/1.
Note 1. ADC120_ADI
Note 2. ADC120_GBADI

**Figure 37.21 Group priority operation example 1-4: “Group A trigger is input during group B scan” when rescanning is disabled (when ADGSPCR.GBRSCN = 0, ADGSPCR.GBRP = 0, and ADGSPCR.LGRRS = 0)**

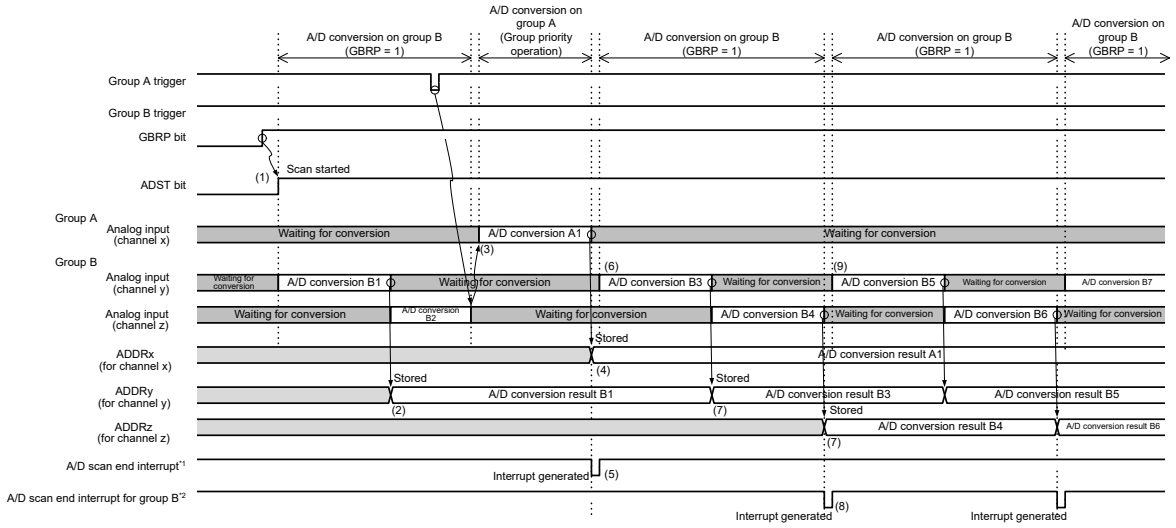
Operation example 1-5 shows the group priority operation in group-scan mode (when ADGSPCR.GBRP = 1, and ADGSPCR.LGRRS = 0).

### Operation example 1-5: Continuously activating single-scan operation for group B

1. When ADGSPCR.GBRP = 1 is set, the ADCSR.ADST bit is set to 1 (starting A/D conversion) and A/D conversion for the analog input channels selected in the ADANSB0 and ADANSB1 registers starts according to the conversion order from the channel with the smallest number n.
2. On completion of A/D conversion for each channel in group B, the result is stored in the corresponding A/D Data Register y (ADDRy).
3. When a trigger for group A is input during A/D conversion for group B, A/D conversion for group B stops while the ADCSR.ADST bit remains 1, and then A/D conversion for group A analog input channels selected in the ADANSA0 and ADANSA1 registers starts according to the conversion order zfrom the channel with the smallest number n. If A/D conversion stops before it is completed, the conversion result is not stored in the A/D Data Register y (ADDRy).
4. On completion of A/D conversion of a single channel, the result is stored in the corresponding A/D Data Register y (ADDRy).
5. On completion of A/D conversion for group A, an ADC120_ADI interrupt request is generated.
6. If ADGSPCR.GBRP = 1 is set (performing single scan continuously), A/D conversion for the group B analog input channels selected in the ADANSB0 and ADANSB1 registers restarts according to the conversion order from the channel with the smallest number n while the ADCSR.ADST remains 1 (starting A/D conversion).
7. On completion of A/D conversion of a single channel, the result is stored in the corresponding A/D Data Register y (ADDRy).

8. If the setting of the ADCSR.GBADIE bit is 1 (enabling interrupt generation on completion of group B scan), a group B scan end interrupt request is generated.
9. If ADGSPCR.GBRP = 1 is set (performing single scan continuously), A/D conversion for the group B analog input channels selected in the ADANSB0 and ADANSB1 registers restarts according to the conversion order from the channel with the smallest number n while the ADCSR.ADST remains 1 (starting A/D conversion).

Steps 6 to 9 are repeated as long as the ADGSPCR.GBRP bit remains 1. Do not clear the ADCSR.ADST bit as long as the ADGSPCR.GBRP bit is 1. To forcibly stop A/D conversion while ADGSPCR.GBRP = 1, follow the procedure shown in Figure 37.33.




Note: x, y, and z (y < z) are the selected channel numbers by ADANSA0/1 or ADANSB0/1.
Note 1. ADC120_ADI
Note 2. ADC120_GBADI

**Figure 37.22 Group priority operation example 1-5: Continuously activating single scan for group B (when ADGSPCR.GBRP = 1 , ADGSPCR.LGRRS = 0)**

Note: To continuously activate single-scan operation for group B, disable group B trigger input.

## 37.3.5 Compare Function for Windows A and B

### 37.3.5.1 Compare Function Windows A and B

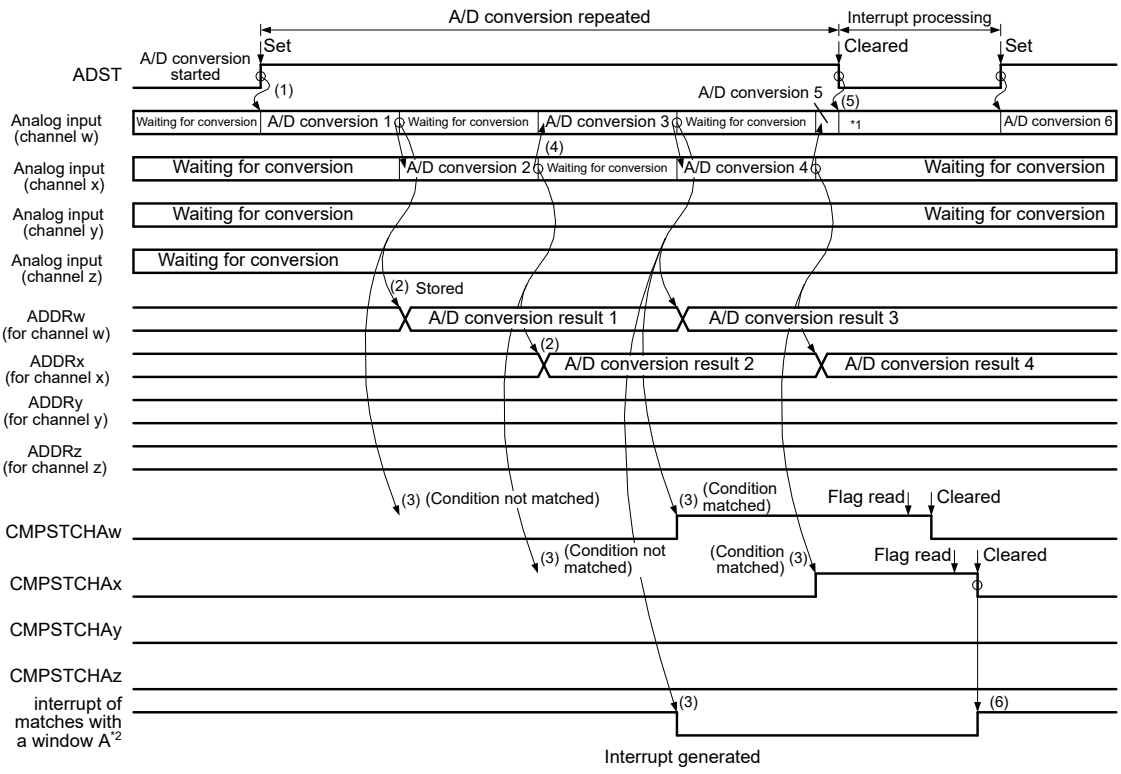
The compare function compares a reference value with the A/D conversion result. The reference value can be set for Window A and Window B independently. When the compare function is in use, the self-diagnosis function and double trigger mode cannot be used. The main differences between Window A and Window B are their different interrupt output signals and the constraint on Window B of only one selectable channel.

This section provides an example operation that combines continuous scan mode and the compare function.

The operation is as follows:

1. When the ADCSR.ADST bit is set to 1 (A/D conversion start) by software, a synchronous trigger (ELC) or an asynchronous trigger, A/D conversion starts in the order of the selected channels , temperature sensor, and internal reference voltage.
2. On completion of A/D conversion, the A/D conversion result is stored in the associated A/D Data Register y (ADDRy, ADTSDR, or ADOCDR). When ADCMPCR.CMPAE = 1, if bits in the ADCMPANSRy register or the ADCMPANSER register are set for Window A, the A/D conversion result is compared with the set ADCMPDR0/1 register value. When ADCMPCR.CMPBE = 1, if bits in the ADCMPBNSR register are set for Window B, the A/D conversion result is compared with the ADWINULB/ADWINLLB register setting.

3. As a result of the comparison, when Window A meets the condition set in ADCMPLR0/1 or ADCMPLER, the Compare Window A Flag (ADCMPSR0.CMPSTCHAn, ADCMPSR1.CMPSTCHAn, ADCMPSER.CMPSTTSA or ADCMPSER.CMPSTOCA) sets 1. At this time, if the ADCMPCR.CMPAIE bit is 1, an ADC120_CMPAI interrupt request is generated. In the same way, when Window B meets the condition set in ADCMPBNSR.CMPLB, the Compare Window B Flag (ADCMPBSR.CMPSTB) sets to 1. At this time, if the ADCMPCR.CMPBIE bit is 1, an ADC120_CMPBI interrupt request is generated.
4. On completion of all selected A/D conversions and comparisons, scan restarts.
5. After the ADC120_CMPAI and ADC120_CMPBI interrupts are accepted, the ADCSR.ADST bit is set to 0 (A/D conversion stop) and processing is performed for channels for which the compare flag is set to 1.
6. When all compare flags of Window A are cleared, the ADC120_CMPAI interrupt request is canceled. In the same way, when all compare flags of Window B are cleared, the ADC120_CMPBI interrupt request is reset. To perform comparison again, restart the A/D conversion.




Note: w, x, y, and z (w < x < y < z) are the selected channel numbers by ADANSA0/1.
Note 1. Data on conversion is ignored.
Note 2. ADC120_CMPAI

**Figure 37.23 Example of compare function operation, when the analog inputs (channel w to z) are compared**

### 37.3.5.2 Event Output of Compare Function

The event output of the compare function specifies the upper-side reference voltage value and the lower-side reference voltage value for window A and window B, respectively. The output compares the A/D converted value of the selected channel with the upper and lower side reference voltage value and outputs events (ADC120_WCMPM/ ADC120_WCMPUM) based on event conditions (A or B, A and B, A xor B) and comparison result of window A and window B.

If more than one channel is selected for window A, and even when one channel in window A meets the comparison condition, the comparison result of window A is met. When using this function, perform A/D conversion in single scan mode.

Any channels from analog input, internal reference voltage, and temperature sensor output are selectable for window A.

One channel from analog input, internal reference voltage, and temperature sensor output is selectable for window B.

The following sequence is an example of how to set up and use the event output of the compare function:

1. Confirm that the value in the ADCSR.ADCS bits is 00b (single scan mode).
2. Select the channel for window A in the ADCMPANSR0/1 and ADCMPANSER registers. Set the window comparison conditions in the ADCMPLR0/1 and ADCMPLER registers. Set the upper-side and lower-side reference values in the ADCMPDR0/1 registers.
3. Select the channel and comparison conditions for Window B in the ADCMPBNSR register, and set the upper and lower reference values in the ADWINULB and ADWINLLB registers.
4. Set the composite conditions for window A/B, window A/B operation enable, and interrupt output enable in the ADCMPCR register.

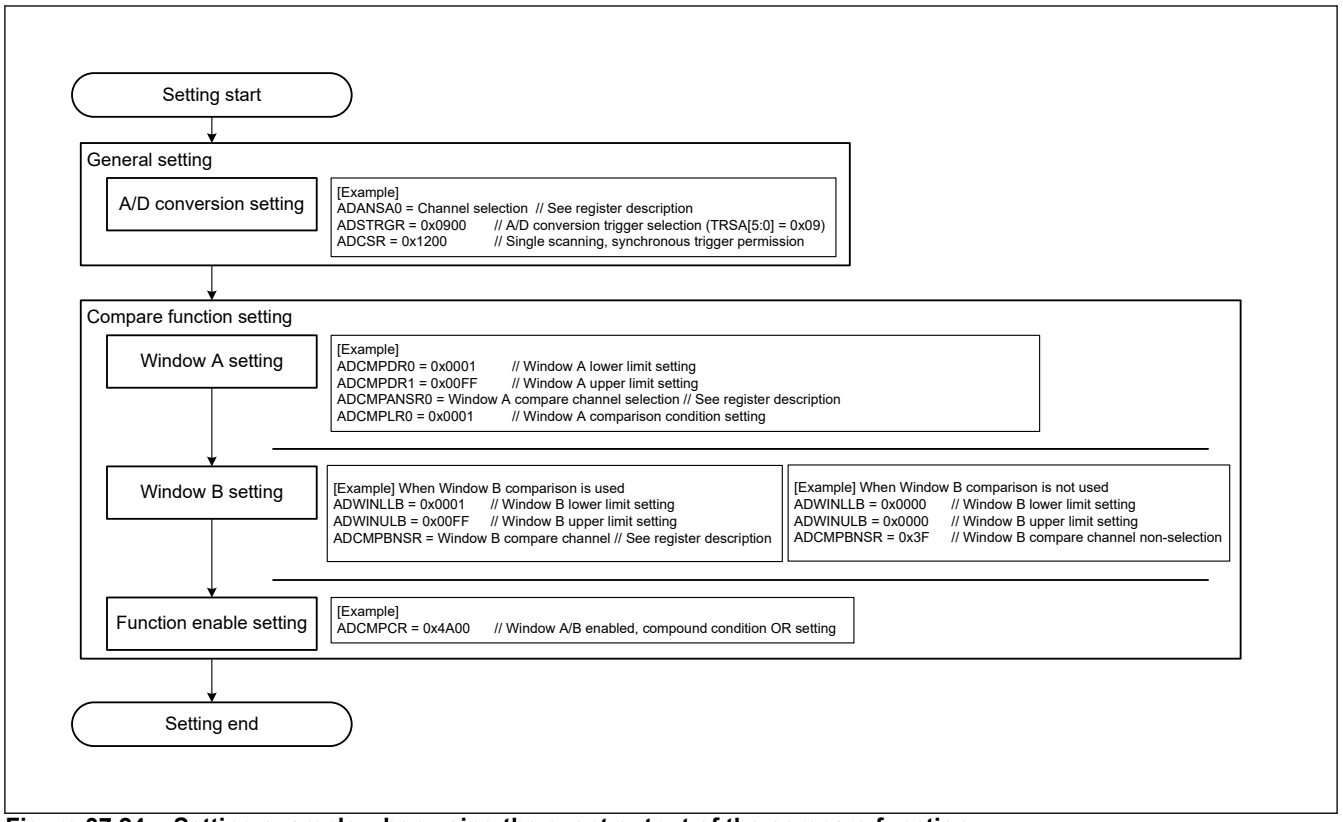



**Figure 37.24 Setting example when using the event output of the compare function**

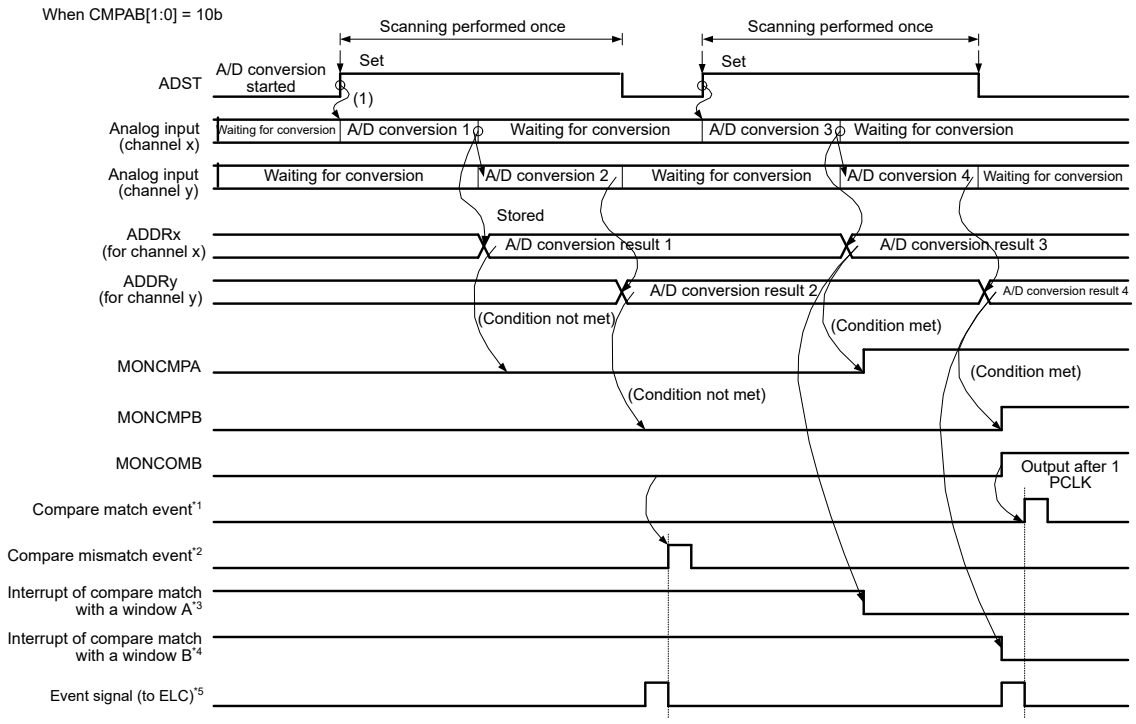
For event output usage when using only window A for the compare function, note the following:

- Enable both Window A and Window B (ADCMPCR.CMPAE = 1, ADCMPCR.CMPBE = 1)
- Set the compound condition of Window A and Window B to “OR condition” (ADCMPCR.CMPAB[1:0] = 00b)
- Set the compared channel of Window B to “No selection” (ADCMPBNSR.CMPCHB[5:0] = 0x3F)
- Set the compare condition of Window B to “0 < results < 0 always means mismatch”. (ADCMPCR.WCMPE = 1, ADWINLLB[15:0] = ADWINULB[15:0] = 0x0000, and ADCMPBNSR.CMPLB = 1)

Figure 37.25 shows the event output operation example of compare function.

A scan end event (ADC120_ADI) is output with the same timing as single scan completion. A match or mismatch event (ADC120_WCMPM/ADC120_WCMPUM) is output with 1 PCLKA cycle delay depending on the ADCMPCR.CMPAB[1:0] settings.

Note: The match and mismatch events are exclusive, so both events are never output simultaneously.




Note: x, and y (x < y) are the selected channel numbers by ADANSA0/1.
Note 1. ADC120_WCMPM
Note 2. ADC120_WCMPUM
Note 3. ADC120_CMPAI
Note 4. ADC120_CMPBI
Note 5. ADC120_ADI

**Figure 37.25 Example operation of the compare function event output, when the analog inputs (channel x and y) are compared**

Note: Event output of compare function outputs match/mismatch from the comparison results of Window A and Window B, based on the ADCMPCR.CMPAB[1:0] settings.

Note: The comparison result of Window A is the logical addition of the comparison results of the comparison target channels of Window A. The comparison results of Window A and Window B are updated by each A/D conversion, and are kept even when single scan ends. Set ADCMPCR.CMPAE and ADCMPCR.CMPBE to 0 to clear the comparison results to 0.

### 37.3.5.3 Restrictions on Compare Function

The following constraints apply for the compare function:

- The compare function cannot be used together with the self-diagnosis function or double-trigger mode. (The compare function is not available for ADRD, ADDBLDR, ADDBLDRA, and ADDBLDRB.)
- Specify single scan mode when using match/mismatch event outputs.
- When the temperature sensor output, internal reference voltage is selected for Window A, Window B operations are prohibited.
- When the temperature sensor output, internal reference voltage is selected for Window B, Window A operations are prohibited.
- Setting the same channel for Window A and Window B is prohibited.
- Set the reference voltage values so that the high-potential reference voltage value is equal to or larger than the low potential reference voltage value.

### 37.3.6 Analog Input Sampling and Scan Conversion Time

Scan conversion can be activated either by a software trigger, a synchronous trigger (ELC), or an asynchronous trigger (ADTRG0). After the start-of-scanning-delay time ($t_D$) has elapsed, processing for disconnection detection assistance, and processing of conversion for self-diagnosis all proceed, followed by processing for A/D conversion.

Figure 37.26 shows the scan conversion timing, in which scan conversion is activated by a software trigger or a synchronous trigger (ELC). Figure 37.27 shows the scan conversion timing, in which scan conversion is activated by an asynchronous trigger (ADTRG0). The scan conversion time ($t_{SCAN}$) includes the start-of-scanning-delay time ($t_D$), disconnection detection assistance processing time ($t_{DIS}$)*1, self-diagnosis A/D conversion processing time ($t_{DIAG}$ and $t_{DSD}$)*2, A/D conversion processing time ($t_{CONV}$), and end-of-scanning-delay time ($t_{ED}$).

The A/D conversion processing time ($t_{CONV}$) consists of input sampling time ($t_{SPL}$) and time for conversion by successive approximation ($t_{SAM}$). The sampling time ($t_{SPL}$) is used to charge sample-and-hold circuits in the A/D converter. If there is not sufficient sampling time due to the high impedance of an analog input signal source, or if the A/ D conversion clock (ADCLK) is slow, sampling time can be adjusted using the ADSSTRn register.

The time for conversion by successive approximation ($t_{SAM}$) is the following

- 13 ADCLK states with 12-bit accuracy selected.
- 11 ADCLK states with 10-bit accuracy selected.
- 9 ADCLK states with 8-bit accuracy selected.

Table 37.25 shows the time for conversion by successive approximation ($t_{SAM}$).

The scan conversion time ($t_{SCAN}$) in single scan mode for which the number of selected channels is n can be determined as follows:

$$t_{SCAN} = t_D + (t_{DIS} \times n) + t_{DIAG} + t_{ED} + (t_{CONV} \times n)$$ *3

The scan conversion time for the first cycle in continuous scan mode is $t_{SCAN}$ for single scan minus $t_{ED}$ . The scan conversion time for the second and subsequent cycles in continuous scan mode is fixed in the following:

$$(t_{DIS} \times n) + t_{DIAG} + t_{DSD} + (t_{CONV} \times n)$$ *3

Note 1. When disconnection detection assistance is not selected, $t_{DIS}$ = 0.

Only when the temperature sensor or internal reference voltage is A/D-converted, the auto-discharge period of 15 ADCLK states is inserted.

Note 2. When the self-diagnosis function is not used, $t_{DIAG}$ = 0, $t_{DSD}$ = 0.

Note 3. When input sampling times ($t_{SPL}$) of all selected channels are the same, this element equals $t_{CONV}$ × n. If each channel has a different sampling time, this element equals that of $t_{SPL}$ and $t_{SAM}$ set to each selected channel.

Table 37.25 shows the times for conversion during scanning.

**Table 37.25 Conversion times during scanning (in numbers of cycles of ADCLK and PCLKA)**

| Item | | | Symbol | | Type/Conditions: Synchronous trigger*4 | Type/Conditions: Asynchronous trigger | Type/Conditions: Software trigger | Unit |
|---|---|---|---|---|---|---|---|---|
| Scan start processing time*1 *2 | A/D conversion on group A under group A priority control. | Group B is to be stopped (Group A is activated after group B is stopped by of an A/D conversion source from group A). | $t_D$ | | 3 PCLKA + 6 ADCLK<br>5 PCLKA + 3 ADCLK*5 | — | — | Cycles |
| | | Group B is not to be stopped (activation by an A/D conversion source from group A). | | | 2 PCLKA + 4 ADCLK | — | — | |
| | A/D conversion when self-diagnosis is enabled. | A/D conversion for self-diagnosis is to be started. | | | 2 PCLKA + 6 ADCLK | 4 PCLKA + 6 ADCLK | 6 ADCLK | |
| | All other | | | | 2 PCLKA + 4 ADCLK | 2 PCLKA + 4 ADCLK | 4 ADCLK | |
| Disconnection detection assistance processing time | | | $t_{DIS}$ | | Setting in ADNDIS[3:0] (initial value = 0x0) × ADCLK | | | |
| Self-diagnosis conversion processing time*1 | Sampling time | | $t_{DIAG}$ | $t_{SPL}$ | Setting in ADSSTR00 (initial value = 0x0B) × ADCLK*3 | — | — | |
| | Time for conversion by successive approximation | 12-bit conversion accuracy | | $t_{SAM}$ | 15 ADCLK | — | — | |
| | | 10-bit conversion accuracy | | | 13 ADCLK | — | — | |
| | | 8-bit conversion accuracy | | | 11 ADCLK | — | — | |
| | Wait time between self-diagnosis conversion end and analog channel sampling start. | | | $t_{DED}$ | 2 ADCLK | | | |
| | Wait time between last channel conversion end and self-diagnosis sampling start in continuous scan mode. | | $t_{DSD}$ | | 2 ADCLK | | | |
| A/D conversion processing time*1 | Sampling time | | $t_{CONV}$ | $t_{SPL}$ | Setting in ADSSTRn (n = 0 to 8, 11 to 13, L, T, O) (initial value = 0x0B) × ADCLK + 0.5 ADCLK | | | |
| | Time for conversion by successive approximation | 12-bit conversion accuracy | | $t_{SAM}$ | 13 ADCLK | | | |
| | | 10-bit conversion accuracy | | | 11 ADCLK | | | |
| | | 8-bit conversion accuracy | | | 9 ADCLK | | | |
| Scan end processing time*1 | | | $t_{ED}$ | | 1 PCLKA + 3 ADCLK<br>2 PCLKA + 3 ADCLK*5 | | | |

Note 1. See Figure 37.26 and Figure 37.27 for an illustration of times $t_D$, $t_{SPLSH}$, $t_{DIAG}$, $t_{CONV}$, and $t_{ED}$.
Note 2. This is the maximum time required from software writing or trigger input to A/D conversion start.
Note 3. The sampling time setting must satisfy the electrical characteristics.
Note 4. This does not include the time consumed in the path from timer output to trigger input.
Note 5. If ADCLK is faster than PCLKA (PCLKA to ADCLK frequency ratio = 1:2 or 1:4), the scan end processing time changes.

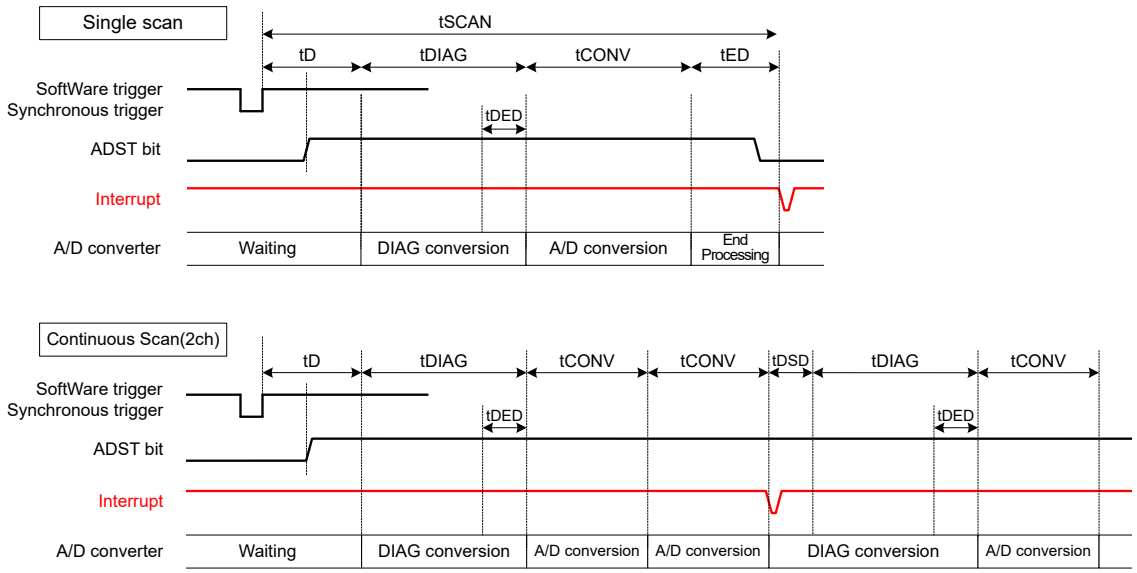



**Figure 37.26 Scan conversion timing when activated by software or a synchronous trigger input (ELC)**

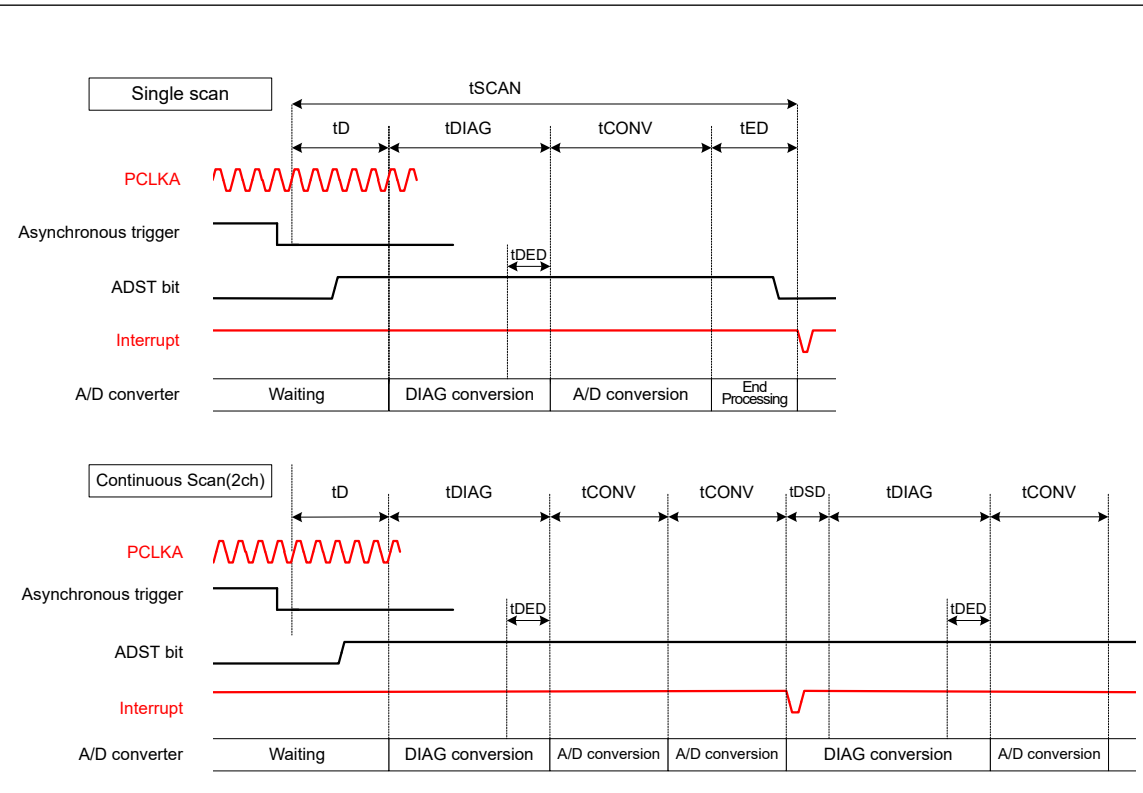



**Figure 37.27 Scan conversion timing when activated by an asynchronous trigger input (ADTRG0)**

## 37.3.7 Usage Example of A/D Data Register Automatic Clearing Function

Setting the ADCER.ACE bit to 1 automatically clears the A/D data registers (ADDRy, ADRD, ADDBLDR, ADDBLDRA, ADDBLDRB, ADTSDR, ADOCDR) to 0x0000 when the A/D data registers are read by the CPU or DTC or DMAC.

This function enables detection of update failures of the A/D data registers (ADDRy, ADRD, ADDBLDR, ADDBLDRA, ADDBLDRB, ADTSDR, ADOCDR). This section describes examples in which the function to automatically clear the ADDRy register is enabled and disabled.

- If the ADCER.ACE bit is 0 (automatic clearing is disabled) and for some reason, if the A/D conversion result (0x0222) is not written to the ADDRy register, the ADDRy value retains the old data (0x0111). In addition, if this ADDRy value is read into a general-purpose register using an A/D scan end interrupt, the old data (0x0111) can be saved in the general-purpose register. When checking whether there is an update failure, it is necessary to frequently save the old data in SRAM or in a general-purpose register.
- If the ADCER.ACE bit is 1 (automatic clearing is enabled), when ADDRy = 0x0111 is read by the CPU or DTC or DMAC, ADDRy is automatically set to 0x0000. Next, if the A/D conversion result of 0x0222 cannot be transferred to ADDRy for some reason, the cleared data (0x0000) remains as the ADDRy value. If this ADDRy value is read into a general-purpose register using an A/D scan end interrupt, 0x0000 is saved in the general-purpose register. Occurrence of an ADDRy update failure can be determined by checking that the read data value is 0x0000.

### 37.3.8 A/D-Converted Value Addition/Average Mode

A/D-converted value addition/average mode can be used when A/D conversion of the analog input of the selected channels, the temperature sensor output, the internal reference voltage is selected.

In A/D-converted value addition mode, the same channel is A/D-converted 1, 2, 3, 4, or 16 consecutive times, and the sum of the converted values is stored in the data register. The conversion count of the addition function can be set to 16 only when 12-bit accuracy is selected. In A/D-converted value average mode, the same channel is A/D-converted 2 or 4 consecutive times, and the mean of the converted values is stored in the data register. The use of the average of these results can improve the accuracy of A/D conversion, depending on the types of noise components that are present. This function, however, cannot always guarantee an improvement in A/D conversion accuracy.

The A/D-converted value addition/average function can be used when A/D conversion of the analog inputs of the selected channels or A/D conversion of the temperature sensor output or A/D conversion of the internal reference voltage is selected. The A/D-converted value addition/average function can also be used for channels for which the double-trigger function is selected.

The addition function for self-diagnosis is not provided.

### 37.3.9 Disconnection Detection Assist Function

The ADC12 incorporates a function to fix the charge for sampling capacitance to the specified state VREFH0 or VREFL0 before the start of A/D conversion. This function enables disconnection detection in wiring of analog inputs.

Figure 37.28 shows the A/D conversion operation when the disconnection detection assist function is used. Figure 37.29 shows an example of disconnection detection when precharge is selected. Figure 37.30 shows an example of disconnection detection when discharge is selected.

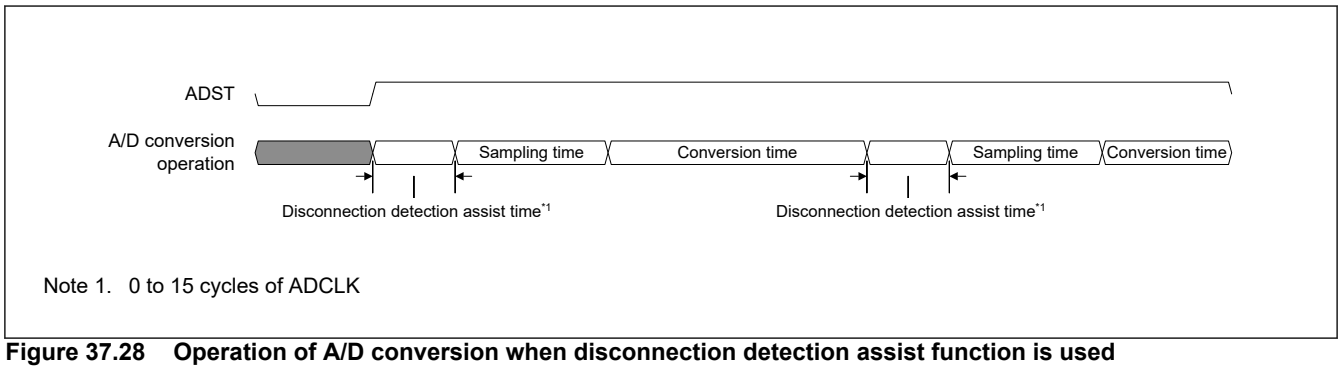
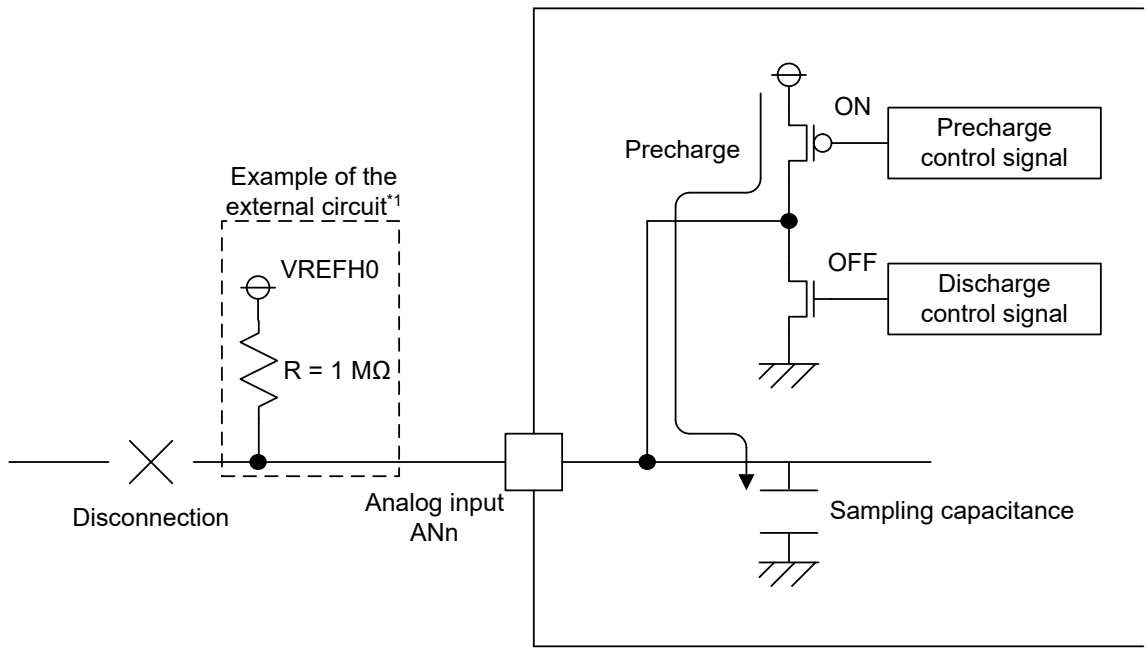
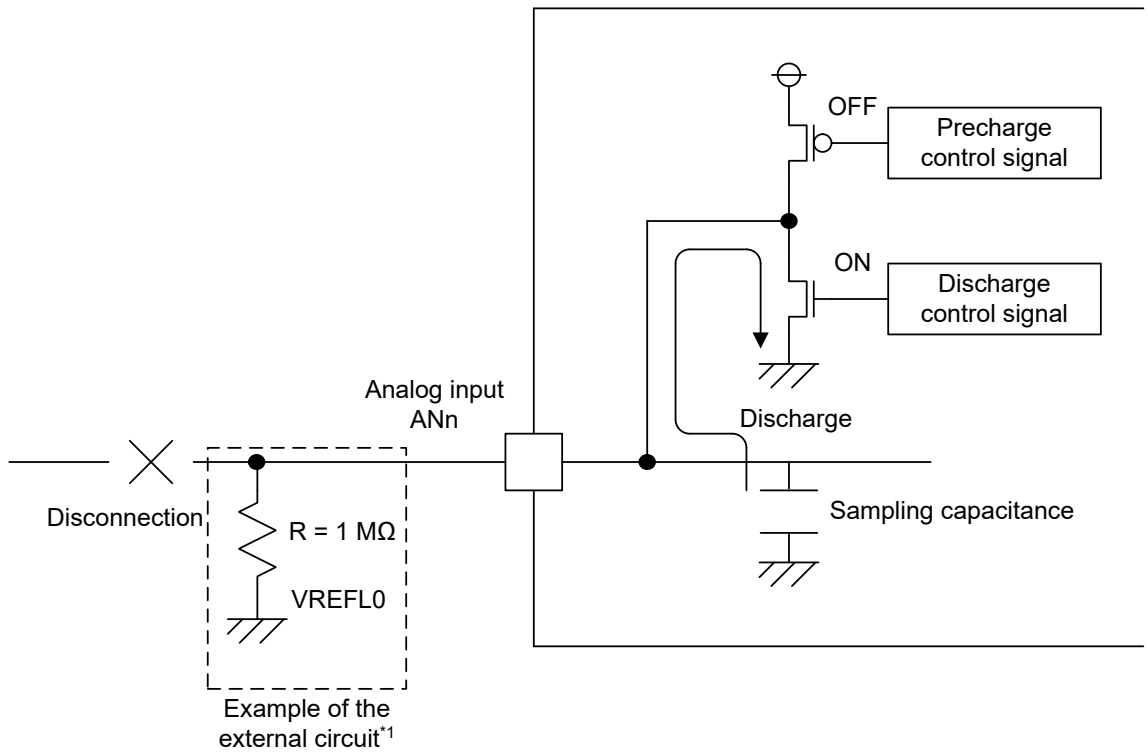



**Figure 37.28 Operation of A/D conversion when disconnection detection assist function is used**




Note 1. The converted result should be used after it is fully evaluated because the result data when disconnection occurs varies depending on the external circuit.

**Figure 37.29 Example of disconnection detection when precharge is selected**



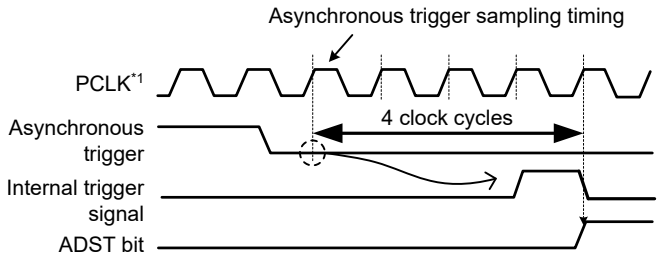

Note 1. The converted result should be used after it is fully evaluated because the result data when disconnection occurs varies depending on the external circuit.

**Figure 37.30 Example of disconnection detection when discharge is selected**

## 37.3.10 Starting A/D Conversion with an Asynchronous Trigger

A/D conversion can be started by the input of an asynchronous trigger. To start A/D conversion by an asynchronous trigger, set the pin function in the PmnPFS register, set the A/D Conversion Start Trigger Select bits (ADSTRGR.TRSA[5:0]) to 0x00, then input a high-level signal to the asynchronous trigger (ADTRG0 pin). Finally, set both the ADCSR.TRGE and ADCSR.EXTRG bits to 1. Figure 37.31 shows timing of the asynchronous trigger input.

An asynchronous trigger cannot be selected in the A/D conversion start trigger for group B used in group scan mode. For details on setting the pin function, see section 19, I/O Ports.




Note 1. PCLKA

**Figure 37.31 Asynchronous trigger input timing**

### 37.3.11 Starting A/D Conversion with a Synchronous Trigger from a Peripheral Module

A/D conversion can be started by a synchronous trigger (ELC). To do this, set the ADCSR.TRGE bit to 1 and the ADCSR.EXTRG bit to 0, and select the relevant sources in the ADSTRGR.TRSA[5:0] bits and ADSTRGR.TRSB[5:0] bits.

### 37.3.12 Using Data Buffers

This IP is provided with a ring buffer function consisting of 16 A/D data buffers. This function sequentially stores A/D conversion results other than self-diagnosis result (including addition/average results) in data buffers (ADBUFn, n = 0 to 15).

Each conversion result is stored at the timing when the A/D conversion result is stored in the data register, and most recent 16 conversion result data are retained.

The figure-below shows the schematic of data buffers, pointer, and overflow flag operations. When the BUFEN bit is set to 1, the A/D conversion result is transferred at each end of A/D conversion. The pointer indicates the number of data buffer to which the next transferred data is to be written. When data is written to up to buffer 15, the pointer is reset to 0000b and the overflow flag is set to 1. Subsequently transferred data overwrites the previously written data.

The overflow flag is reset to the initial value by writing 0x00 to the ADBUFPTR register.

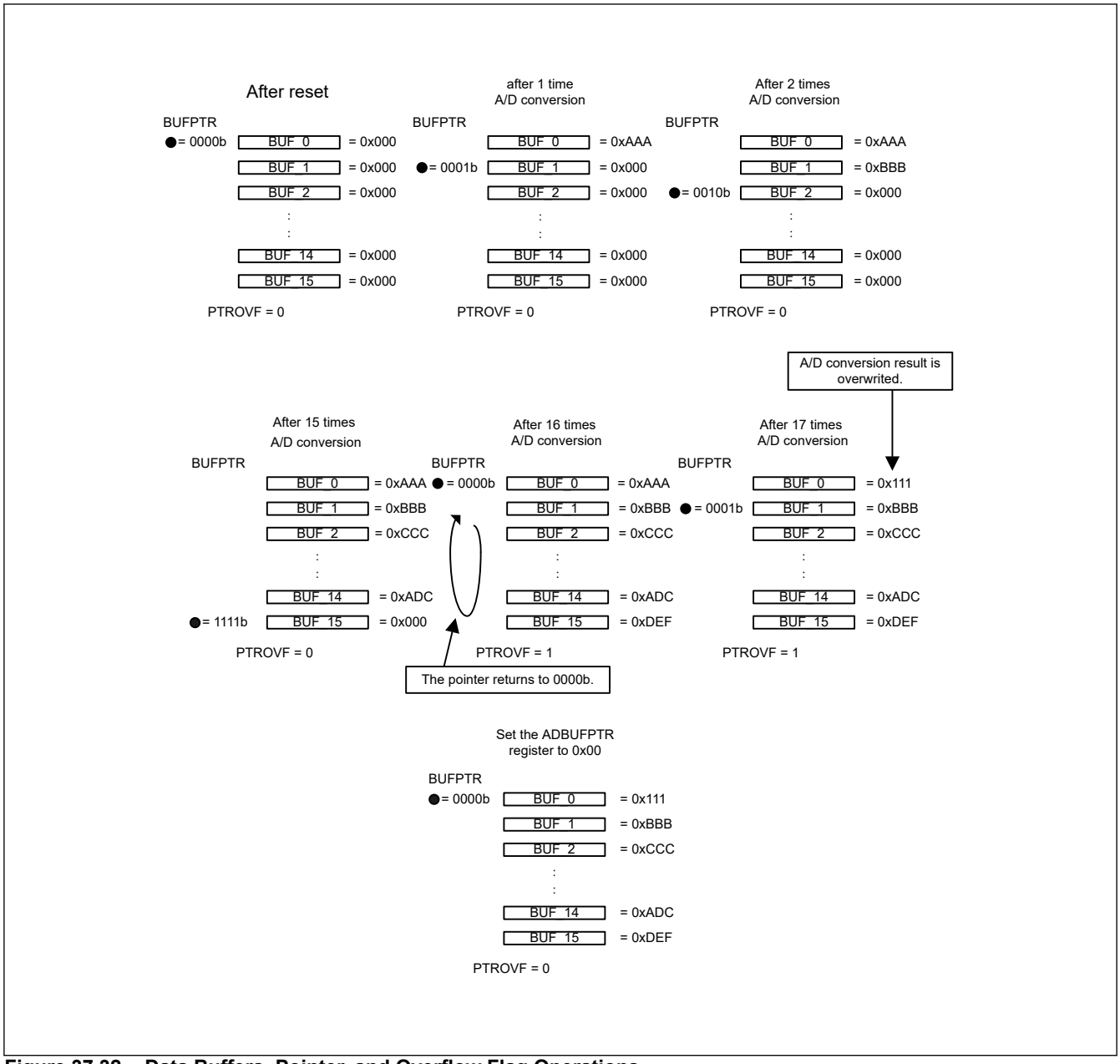



**Figure 37.32 Data Buffers, Pointer, and Overflow Flag Operations**

## 37.4 Interrupt Sources and DTC, DMAC Transfer Requests

### 37.4.1 Interrupt Requests

The ADC12 can send scan end interrupt requests ADC120_ADI and ADC120_GBADI to the CPU. The ADC12 also generates the ADC120_CMPAI/ADC120_CMPBI interrupt for the CPU in response to matches with a condition for comparison.

An ADC120_ADI interrupt is always generated. An ADC120_GBADI interrupt can be generated by setting the ADCSR.GBADIE bit to 1. Similarly, ADC120_CMPAI and ADC120_CMPBI interrupts can be generated by setting the ADCMPCR.CMPAIE and ADCMPCR.CMPBIE bit to 1.

In addition, the DTC or DMAC can be started when an ADC120_ADI or an ADC120_GBADI interrupt is generated. Using an ADC120_ADI or ADC120_GBADI interrupt to activate the DTC or DMAC to read the converted data enables continuous conversion without a burden on software.

Table 37.26 describes the interrupt sources and ELC events available for the ADC12.

**Table 37.26 The interrupt source and ELC event of ADC12 (1 of 2)**

| Operation | | | Interrupt request or ELC event | Interrupt request | DTC or DMAC activation | ELC event request | Function |
|---|---|---|---|---|---|---|---|
| Scan mode | Double trigger mode | Compare function Window A/B | | | | | |
| Single scan mode | Deselected | Deselected | ADC120_ADI | ✓ | ✓ | ✓ | ADC120_ADI generated at the end of single scan |
| | | Selected | ADC120_ADI | ✓ | ✓ | ✓ | ADC120_ADI generated at the end of single scan |
| | | | ADC120_CMPAI | ✓ | — | — | ADC120_CMPAI generated on a match comparison condition of Window A |
| | | | ADC120_CMPBI | ✓ | — | — | ADC120_CMPBI generated on a match comparison condition of Window B |
| | | | ADC120_WCMPM | — | ✓ | ✓ | ADC120_WCMPM generated on a match condition of the Window A/B compare function |
| | | | ADC120_WCMPUM | — | ✓ | ✓ | ADC120_WCMPUM generated on a mismatch condition of the Window A/B compare function |
| | Selected | Deselected | ADC120_ADI | ✓ | ✓ | ✓ | ADC120_ADI generated at the end of scans in the even numbered times |
| Continuous scan mode | Deselected | Deselected | ADC120_ADI | ✓ | ✓ | ✓ | ADC120_ADI generated at the end of scan of all selected channels |
| | | Selected | ADC120_CMPAI | ✓ | — | — | ADC120_CMPAI generated on a match comparison condition of Window A |
| | | | ADC120_CMPBI | ✓ | — | — | ADC120_CMPBI generated on a match comparison condition of Window B |

**Table 37.26 The interrupt source and ELC event of ADC12 (2 of 2)**

| Operation | | | Interrupt request or ELC event | Interrupt request | DTC or DMAC activation | ELC event request | Function |
|---|---|---|---|---|---|---|---|
| Scan mode | Double trigger mode | Compare function Window A/B | | | | | |
| Group scan mode | Deselected | Deselected | ADC120_ADI | ✓ | ✓ | ✓ | ADC120_ADI generated at the end of group A scan |
| | | | ADC120_GBADI | ✓ | ✓ | — | ADC120_GBADI dedicated to group B generated at the end of group B scan |
| | | Selected | ADC120_ADI | ✓ | ✓ | ✓ | ADC120_ADI generated at the end of group A scan |
| | | | ADC120_GBADI | ✓ | ✓ | — | ADC120_GBADI dedicated to group B generated at the end of group B scan |
| | | | ADC120_CMPAI | ✓ | — | — | ADC120_CMPAI generated on a match comparison condition of Window A |
| | | | ADC120_CMPBI | ✓ | — | — | ADC120_CMPBI generated on a match comparison condition of Window B |
| | Selected | Deselected | ADC120_ADI | ✓ | ✓ | ✓ | ADC120_ADI generated at the end of group A scans in the even-numbered times |
| | | | ADC120_GBADI | ✓ | ✓ | — | ADC120_GBADI dedicated to group B generated at the end of group B scan |

Note: ✓ available
—: unavailable

For details on DTC settings, see section 17, Data Transfer Controller (DTC).

## 37.5 Event Link Function

### 37.5.1 Event Output to the ELC

The ELC uses the ADC120_ADI interrupt request signal as an event signal ADC120_ADI, enabling link operation for the preset module. The ADC120_GBADI interrupt and ADC120_CMPAI/ADC120_CMPBI interrupts cannot be used as an event signal. For details, see Table 37.26.

An event signal can be output regardless of the settings of the corresponding interrupt request enable bits. For the scan end event(ADC120_ADI), a high-level pulse for one PCLKA cycle is output at the same output timing as the interrupt output (ADC120_ADI) shown in Table 37.26. For a compare function match (ADC120_WCMPM) and mismatch event (ADC120_WCMPUM) to the ELC, a high-level pulse for one PCLKA cycle is output at the timing delayed by one cycle (PCLKA) from the interrupt output (ADC120_ADI) shown in Table 37.26.

To use compare function match (ADC120_WCMPM) or mismatch event (ADC120_WCMPUM) to the ELC, specify single-scan mode.

### 37.5.2 ADC12 Operation through an Event from the ELC

The ADC12 can start A/D conversion by the preset event specified in the ELSRn settings for the ELC as follows:

- Select the ELC_AD00 signal in the ELC.ELSR8 register
- Select the ELC_AD01 signal in the ELC.ELSR9 register

If an ELC event occurs during A/D conversion, the event is disabled.

## 37.6 Usage Notes

### 37.6.1 Constraints on Setting the Registers

Set each register while the ADCSR.ADST bit is 0.

### 37.6.2 Constraints on Reading the Data Registers

The following registers must be read in halfword units:

- A/D Data Registers
- A/D Data Duplexing Register
- A/D Data Duplexing Register A
- A/D Data Duplexing Register B
- A/D Temperature Sensor Data Register
- A/D Internal Reference Voltage Register
- A/D Self-Diagnosis Data Register
- A/D Data Buffer Registers n (N = 0 to 15)

If a register is read twice in byte units, that is, the upper byte and lower byte are read separately, the A/D-converted value read initially might disagree with the A/D-converted value read subsequently. To prevent this, never read the data registers in byte units.

### 37.6.3 Constraints on Stopping A/D Conversion

#### (1) A/D Conversion Stop Procedure

To stop A/D conversion when an asynchronous trigger or a synchronous trigger is selected as the condition for starting A/D conversion, follow the procedure shown in Figure 37.33.

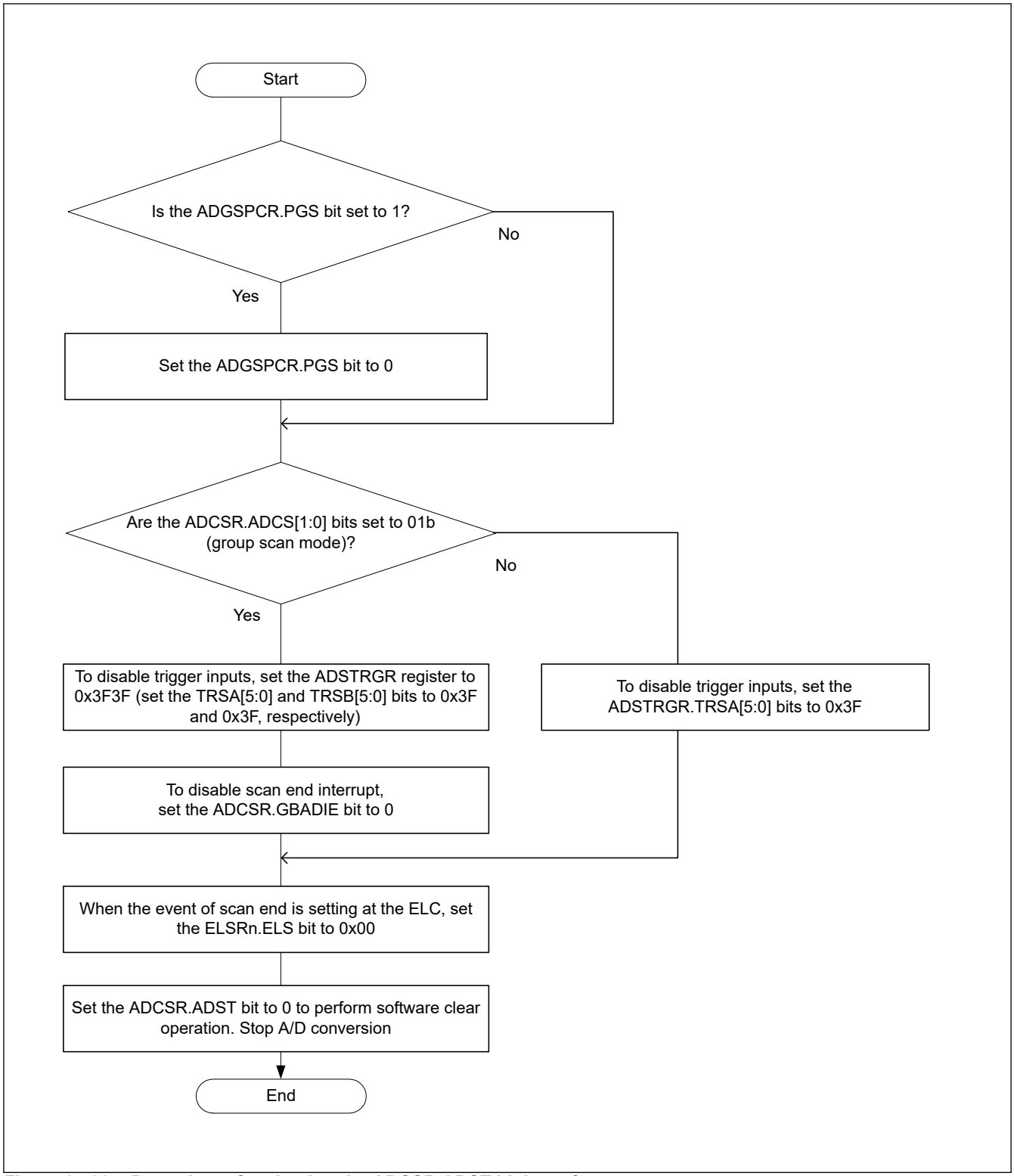



**Figure 37.33 Procedures for clearing the ADCSR.ADST bit by software**

To specify the following settings after performing the clear operation by software, provide a wait period for at least two ADCLK cycles.

- Enabling scan end interrupts
- Enabling scan end events for the event link controller
- Starting A/D conversion by software
- Enabling trigger input

### (2) Notes on Modes and Status Bits

If necessary, individually initialize or set again the voltage status for self-diagnosis, the judgment of the even number or odd number specified for double-trigger mode, and the monitor flags of the compare function.

- To set again the voltage status for self-diagnosis, set the ADCER.DIAGLD bit to 1 and then set a desired value in the ADCER.DIAGVAL[1:0] bits.
- If the setting of the ADCSR.DBLE bit is changed from 0 to 1, the double-trigger mode operation starts from the first scanning.
- To initialize the monitor flags of the compare function (MONCMPA, MONCMPB, and MONCOMB), set the ADCMPCR.CMPAE and ADCMPCR.CMPBE bits to 0.

## 37.6.4 A/D Conversion Restart and Termination Timing

A maximum of 6 ADCLK cycles is required for the idle analog unit of the ADC12 to restart on setting the ADCSR.ADST bit to 1. A maximum of 2 ADCLK cycles is required for the operating analog unit of the ADC12 to terminate on setting the ADCSR.ADST bit to 0.

## 37.6.5 Constraints on Scan End Interrupt Handling

When scanning the same analog input twice using any trigger, the first A/D-converted data is overwritten with the second A/D-converted data. This occurs when the CPU does not complete the reading of the A/D-converted data by the time the A/D conversion of the first analog input for the second scan ends after the first scan end interrupt is generated.

## 37.6.6 Settings for the Module-Stop Function

The Module Stop Control Register can enable or disable ADC12 operation. The ADC12 is initially stopped after a reset. The registers become accessible on release from the module-stop state. After release from the module-stop state, wait for at least 1 µs before starting A/D conversion. For details, see section 10, Low Power Modes.

## 37.6.7 Notes on Entering the Low-Power States

Before entering the module-stop state or Software Standby mode, be sure to stop A/D conversion. Set the ADCSR.ADST bit in ADCSR to 0 and secure certain period until the analog unit of the ADC12 stops. Follow the procedure shown in Figure 37.33 to clear the ADCSR.ADST bit with software. Then, wait for 2 clock cycles of ADCLK before entering the module-stop state or Software Standby mode.

## 37.6.8 Error in Absolute Accuracy When Disconnection Detection Assistance Is in Use

Using disconnection detection assistance leads to an error in absolute accuracy of the ADC12. This error arises because an erroneous voltage is input to the analog input pins due to the resistive voltage division between the pull-up or pull- down resistor (Rp) and the resistance of the signal source (Rs). This error in absolute accuracy is calculated from the following formula:

$$\text{Maximum error in absolute accuracy (LSB)} = \left(2^{\text{Resolution}} - 1\right) \times \text{Rs} / \left(\text{Rs} + \text{Rp}\right)$$

Only use disconnection detection assistance after thorough evaluation.

## 37.6.9 Constraints on Operating Modes and Status Bits

Initialize or set again individually, if necessary, the voltage values in self-diagnosis, the value of the first scan or second scan in double trigger mode, the data buffer pointer, and status monitor in the compare function.

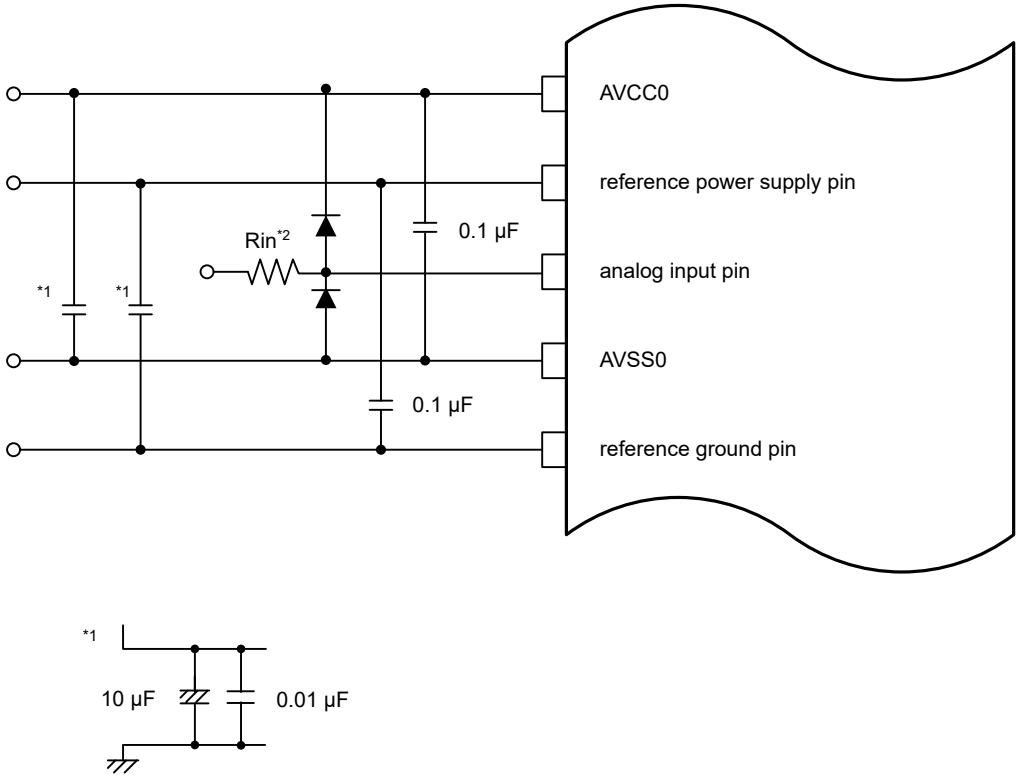
- Select the voltage values in self-diagnosis (ADCER.DIAGVAL[1:0]) after setting ADCER.DIAGLD to 1.
- Double-trigger mode operates as the first scan after setting ADCSR.DBLE from 0 to 1.
- The status monitor bits (MONCMPA, MONCMPB, MONCOMB) in the compare function are initialized after setting ADCMPCR.CMPAE and ADCMPCR.CMPBE to 0.

## 37.6.10 Notes on Board Design

The board should be designed so that digital circuits and analog circuits are separated from each other as far as possible. In addition, digital circuit signal lines and analog circuit signal lines should not intersect or be placed near each other. If these rules are not followed, noise can occur on analog signals and A/D conversion accuracy is affected. The analog input pins, reference power supply pin (VREFH0), reference ground pin (VREFL0), and analog power supply (AVCC0) should be separated from digital circuits using the analog ground (AVSS0). The analog ground (AVSS0) should be connected to a stable digital ground (VSS) on the board (single-point ground plane connection).

## 37.6.11 Constraints on Noise Prevention

To prevent the analog input pins from being destroyed by abnormal voltage such as excessive surge, insert a capacitor between AVCC0 and AVSS0 and between VREFH0 and VREFL0. Additionally, connect a protection circuit to protect the analog input pins as shown in Figure 37.34.




Note 1. The values shown here are reference values.
Note 2. Rin: Signal source impedance

**Figure 37.34 Example protection circuit for analog inputs**

## 37.6.12 Port Settings When Using the ADC12 Input

When using the high-precision channels, do not use PORT0 as general I/O. Renesas recommends that you do not use the digital output that is also used as the AD analog input if normal-precision channel is used. If the digital output that is also used as the AD analog input is used for output signals, perform A/D conversion several times, eliminate the maximum and minimum values, and obtain the average of the other results.

## 37.6.13 Notes on Canceling Software Standby Mode

After software standby mode is canceled, wait at least 1 µs after the stabilization time for the oscillator elapses and before starting A/D conversion. For details, see section 10, Low Power Modes

## 37.6.14 Calculation for Sampling Time

The sampling time can be easily estimated by the following figure and formula. This is the time to reach the voltage within 1/4 LSB.

$t_{SPL} = (R_{EXT} + R_{AD}) \times (C_{EXT} + C_{AD}) \times \ln (C_{AD} / (C_{EXT} + C_{AD}) \times 2^{N+2})$

$R_{EXT}$ shows external signal source impedance
$C_{EXT}$ shows external capacitance (pin capacitance*1 + PCB parasitic capacitance)
N = 12, 10 or 8 (conversion resolution)
$C_{AD}$ = 5 pF (internal capacitance)
$R_{AD}$ = 1.0 kΩ (internal resistance, case of high-speed channels)
$R_{AD}$ = 2.0 kΩ (internal resistance, case of normal-speed channels)

Note 1. Typical value of analog input pin is 5 pF

For example, if $R_{EXT}$ is 1 kΩ, $C_{EXT}$ is 10 pF and N is 12 bits, $t_{SPL}$ of high-speed channel is 258 ns.

This formula simplifies the general use case. This formula is not guaranteed and should be used only for estimation.

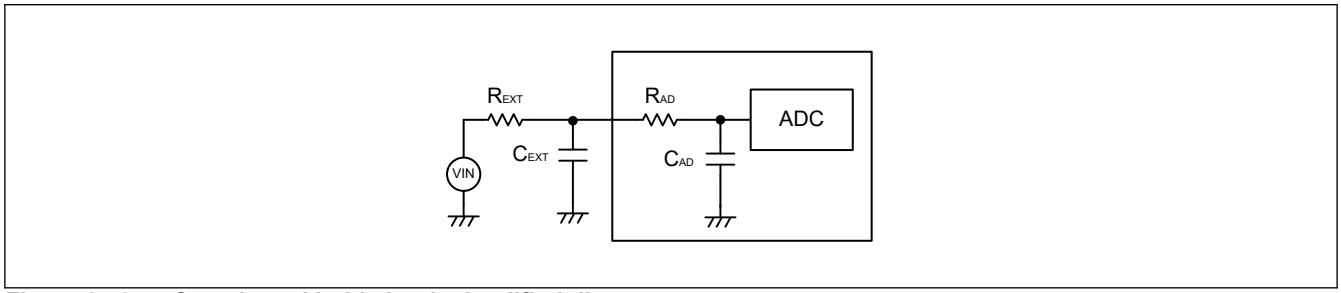



**Figure 37.35 Sample and hold circuit simplified diagram**

# 38. 12-Bit D/A Converter (DAC12)

## 38.1 Overview

The MCU provides a 12-bit D/A Converter (DAC12) with an output amplifier. Table 38.1 lists the DAC12 specifications, Figure 38.1 shows a block diagram, and Table 38.2 lists the I/O pins.

**Table 38.1 DAC12 specifications**

| Parameter | Specifications |
|---|---|
| Resolution | 12 bits |
| Output channels | 2 channels |
| Interference reduction between analog modules | Methods provided to minimize interference between D/A and A/D conversion:<br>• D/A converted data update timing is controlled by the synchronous D/A conversion enable input signal from the ADC12<br>• Degradation of A/D conversion accuracy caused by interference is reduced by controlling the DAC12 inrush current generation timing with the enable signal. |
| Module-stop function | Module-stop state can be set to reduce power consumption |
| Event link function (input) | The DA0 and DA1 conversion can be started on input of an event signal |
| D/A output amplifier control function | Controls whether the output amplifier (for both amplifier-through and amplifier-bias controls) is used |
| TrustZone Filter | Security attribution can be set |

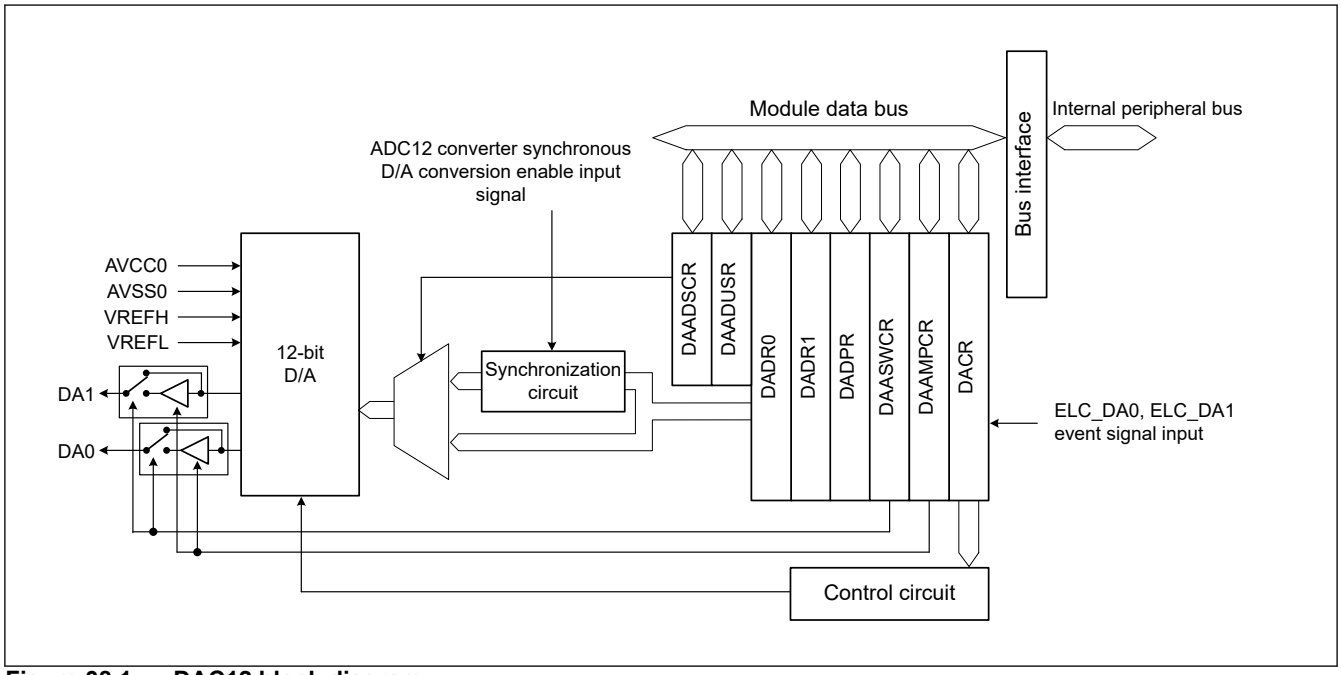



**Figure 38.1 DAC12 block diagram**

Table 38.2 lists the pin configuration of the DAC12.

**Table 38.2 DAC12 I/O pins (1 of 2)**

| Pin name | I/O | Function |
|---|---|---|
| AVCC0 | Input | • Analog power and analog reference top voltage supply pin for ADC12 and DAC12.<br>• Connect to VCC when these modules are not used. |
| AVSS0 | Input | • Analog ground and analog reference ground supply pin for ADC12 and DAC12.<br>• Connect to VSS when these modules are not used. |
| VREFH | Input | Analog reference top voltage supply pin for the DAC12 |
| VREFL | Input | Analog reference ground pin for the DAC12 |

**Table 38.2 DAC12 I/O pins (2 of 2)**

| Pin name | I/O | Function |
|---|---|---|
| DA0 | Output | Channel 0 output pin for the analog signals processed by the DAC12 |
| DA1 | Output | Channel 1 output pin for the analog signals processed by the DAC12 |

## 38.2 Register Descriptions

### 38.2.1 DADRn : D/A Data Register n (n = 0, 1)

Base address: DAC12 = 0x4017_1000

Offset address: 0x00 + 0x02 × n

| Bit position: | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

DADRn register is 16-bit read/write registers that store data for D/A conversion. When an analog output is enabled, the values in DADRn are converted and output to the analog output pins.

12-bit data can be formatted as left- or right-justified in the DADPR.DPSEL bit setting. In right-justified format (DADPR.DPSEL = 0), the lower 12 bits, [11:0], are valid. In left-justified format (DADPR.DPSEL = 1), the upper 12 bits, [15:4], are valid.

### 38.2.2 DACR : D/A Control Register

Base address: DAC12 = 0x4017_1000

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DAOE1 | DAOE0 | DAE | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | — | These bits are read as 1. The write value should be 1. | R/W |
| 5 | DAE*1 | D/A Enable<br>0: Control D/A conversion of channels 0 and 1 individually<br>1: Control D/A conversion of channels 0 and 1 collectively | R/W |
| 6 | DAOE0 | D/A Output Enable 0<br>0: Disable analog output of channel 0 (DA0)<br>1: Enable D/A conversion of channel 0 (DA0) | R/W |
| 7 | DAOE1 | D/A Output Enable 1<br>0: Disable analog output of channel 1 (DA1)<br>1: Enable D/A conversion of channel 1 (DA1) | R/W |

Note 1. This bit controls D/A conversion in combination with the DAOEi bit (i = 0, 1), which controls the output of the conversion results. For details, see Table 38.3.

**Table 38.3 D/A conversion controls**

| DAE | DAOE1 | DAOE0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | Disable D/A conversion and analog output pins (DA0, DA1)*1 |
| | | 1 | • Enable D/A conversion of channel 0 and disable D/A conversion of channel 1<br>• Enable analog output of channel 0 (DA0) and disable analog output of channel 1 (DA1)*1 |
| | 1 | 0 | • Disable D/A conversion of channel 0 and enable D/A conversion of channel 1<br>• Disable analog output of channel 0 (DA0)*1 and enable analog output of channel 1 (DA1) |
| | | 1 | • Enable D/A conversion of channels 0 and 1<br>• Enable analog output of channels 0 and 1 (DA0, DA1) |
| 1 | x | x | • Enable D/A conversion of channels 0 and 1<br>• Collective enable analog output of channels 0 and 1 (DA0, DA1) |

Note: x: Don't care
Note 1. When analog output is disabled, the analog output signal is placed in the Hi-Z state.

Only set this register while the ADC12 is halted when the DAADSCR.DAADST bit is 1 (interference reduction between D/A and A/D conversion is enabled). Only set DACR while the ADCSR.ADST bit is 0 and after selecting the software trigger, for the ADC12 trigger to securely stop the ADC12.

### DAE bit (D/A Enable)

The DAE bit controls D/A conversion, amplifier operation, and analog output in combination with the DAOEi bit (i = 0, 1) and the DAAMPCR.DAAMPi bit (i = 0, 1). See Table 38.4.

When interference reduction between D/A and A/D conversions is enabled (DAADSCR.DAADST = 1), set the ADCSR.ADST bit of the ADC12 to 0. Then, select the software trigger for the ADC12 trigger to securely stop the ADC12.

### DAOEi bit (D/A Output Enable i)

The DAOEi bit (i = 0, 1) controls D/A conversion, amplifier operation, and analog output in combination with the DAE bit and DAAMPCR.DAAMPi bit (i = 0, 1). See Table 38.4.

When both the DAOEi bit (i = 0, 1) and DAE bit are 0, D/A conversion of channel i (i = 0, 1) is not processed, and no conversion result is output.

When interference reduction between D/A and A/D conversions is enabled (DAADSCR.DAADST = 1), set the DAOEi bit while the ADCSR.ADST bit of the ADC12 is set to 0. Then, select the software trigger for the ADC12 trigger to securely stop the ADC12.

The event link function can be used to set the DAOEi bit to 1. The DAOE0 bit is set to 1 when the event specified in the ELSR12 register of the ELC (ELC_DA0 event) occurs, and output of the D/A conversion results starts. The DAOE1 bit is set to 1 when the event specified in the ELSR13 register of the ELC (ELC_DA1 event) occurs, and output of the D/A conversion results starts.

**Table 38.4 D/A conversion and analog output control**

| DACR | | DAAMPCR | | | |
|---|---|---|---|---|---|
| DAE | DAOEi | DAAMPi | Channel i operation | Amplifier operation of channel i | Analog output of channel i |
| 0 | 0 | 0 | Stop | Stop | Hi-Z |
| | | 1 | Stop | Stop | Hi-Z |
| | 1 | 0 | Run | Stop | Amplifier-through |
| | | 1 | Run | Run | Amplifier output |
| 1 | 0 | 0 | Run | Stop | Amplifier-through |
| | | 1 | Run | Run | Amplifier output |
| | 1 | 0 | Run | Stop | Amplifier-through |
| | | 1 | Run | Run | Amplifier output |

Note: i = 0, 1

### 38.2.3 DADPR : DADRn Format Select Register

Base address: DAC12 = 0x4017_1000

Offset address: 0x05

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DPSEL | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | DPSEL | DADRn Format Select<br>0: Right-justified format<br>1: Left-justified format | R/W |

### 38.2.4 DAADSCR : D/A A/D Synchronous Start Control Register

Base address: DAC12 = 0x4017_1000

Offset address: 0x06

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DAADST | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | DAADST | D/A A/D Synchronous Conversion<br>0: Do not synchronize DAC12 with ADC12 operation (disable interference reduction between D/A and A/D conversion).<br>1: Synchronize DAC12 with ADC12 operation (enable interference reduction between D/A and A/D conversion). | R/W |

To minimize interference between D/A and A/D conversion, the DAADSCR register enables synchronization of the start timing of D/A conversion with the ADC12 synchronous D/A conversion enable input signal.

Only set this register while the ADC12 is halted, that is, while the ADCSR.ADST bit is 0 after selecting the software trigger as the ADC12 trigger.

Select the target ADC12 unit before setting the DAADST bit to 1. Set DAADUSR[0] bit to 1 to select unit 0.

#### DAADST bit (D/A A/D Synchronous Conversion)

Setting the DAADST bit to 0 allows the DADRn register value to be converted into analog data at any time. Setting the DAADST bit to 1 allows synchronization of D/A conversion with the synchronous D/A conversion enable input signal from the ADC12. With this bit set, D/A conversion does not start until the ADC12 completes A/D conversion, even when the DADRn register is changed.

Set this bit while the ADCSR.ADST bit is set to 0. Then, select the software trigger for the ADC12 trigger to securely stop the ADC12. Set the DAADUSR.AMADSEL0 bit to 1 before setting the DAADST bit to 1.

The event link function cannot be used when the DAADST bit is set to 1. Stop the event link function by setting the ELSR12 and ELSR13 registers of the ELC. The setting of the DAADST bit is shared by channels 0 and 1 of the DAC12.

## 38.2.5 DAAMPCR : D/A Output Amplifier Control Register

Base address: DAC12 = 0x4017_1000

Offset address: 0x08

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DAAMP1 | DAAMP0 | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | DAAMP0 | Amplifier Control 0<br>0: Do not use channel 0 output amplifier<br>1: Use channel 0 output amplifier | R/W |
| 7 | DAAMP1 | Amplifier Control 1<br>0: Do not use channel 1 output amplifier<br>1: Use channel 1 output amplifier | R/W |

The DAAMPCR register selects D/A output with or without using the amplifier.

### DAAMP0 bit (Amplifier Control 0)

When the DAAMP0 bit is 0, analog values are output for D/A output of channel 0 without using the amplifier. When the DAAMP0 bit is 1, analog values are output for D/A output of channel 0 through the amplifier.

When both the DACR.DAE and DACR.DAOE0 bits are 0, the amplifier is not used regardless of the setting of the DAAMP0 bit. See Table 38.4 for details.

### DAAMP1 bit (Amplifier Control 1)

When the DAAMP1 bit is 0, analog values are output for D/A output of channel 1 without using the amplifier. When the DAAMP1 bit is 1, analog values are output for D/A output of channel 1 through the amplifier.

When both the DACR.DAE and DACR.DAOE1 bits are 0, the amplifier is not used regardless of the setting of the DAAMP1 bit. See Table 38.4 for details.

## 38.2.6 DAASWCR : D/A Amplifier Stabilization Wait Control Register

Base address: DAC12 = 0x4017_1000

Offset address: 0x1C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | DAASW1 | DAASW0 | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | DAASW0 | D/A Amplifier Stabilization Wait 0<br>0: Amplifier stabilization wait off (output) for channel 0<br>1: Amplifier stabilization wait on (high-Z) for channel 0 | R/W |
| 7 | DAASW1 | D/A Amplifier Stabilization Wait 1<br>0: Amplifier stabilization wait off (output) for channel 1<br>1: Amplifier stabilization wait on (high-Z) for channel 1 | R/W |

The DAASWCR register controls D/A output with the output amplifier. This register is used in the initialization procedure to wait for stabilization of the D/A output amplifier. Each bit in DAASWCR should be set to 1 when both the DACR.DAE bit and the DACR.DAOEi (i = 0, 1) bit are 0 . See section 38.6.5. Initialization Procedure with the Output Amplifier.

### DAASW0 bit (D/A Amplifier Stabilization Wait 0)

Set the DAASW0 bit to 1 in the initialization procedure to wait for the stabilization of the D/A channel 0 output amplifier. When DAASW0 is set to 1, D/A conversion operates, but the conversion result of D/A is not output from channel 0. When the DAASW0 bit is 0, the stabilization wait time stops, and the D/A conversion result of channel 0 is output through the output amplifier.

### DAASW1 bit (D/A Amplifier Stabilization Wait 1)

Set the DAASW1 bit to 1 in the initialization procedure to wait for the stabilization of the D/A channel 1 output amplifier. When DAASW1 is set to 1, D/A conversion operates, but the conversion result of D/A is not output from channel 1. When the DAASW1 bit is 0, the stabilization wait time stops, and the D/A conversion result of channel 1 is output through the output amplifier.

## 38.2.7 DAADUSR : D/A A/D Synchronous Unit Select Register

Base address: DAC12 = 0x4017_1000

Offset address: 0x10C0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | AMADSEL0 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | AMADSEL0 | A/D Unit 0 Select<br>0: Do not select unit 0<br>1: Select unit 0 | R/W |
| 1 | — | This bit is read as 0. The write value should be 0. | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

The DAADUSR register selects the target ADC12 unit for D/A and A/D synchronous conversions. Set the AMADSEL0 bit to 1 to select unit 0 as the target synchronous unit for the MCU. When setting the DAADSCR.DAADST bit to 1 for synchronous conversions, select the target unit in this register in advance.

Only set the DAADUSR register while the ADCSR.ADST bit of the ADC12 is set to 0 and the DAADSCR.DAADST bit is set to 0.

## 38.3 Operation

The DAC12 includes D/A conversion circuits for two channles, each of which can operate independently. When the DAOEn bit (n = 0, 1) in DACR is set to 1, DAC12 is enabled and the conversion result is output.

This following example shows D/A conversion on channel 0. Figure 38.2 shows the timing of this operation.

To process D/A conversion on channel 0:

1. Set the data for D/A conversion in the DADR0 register and the data format in the DADPR.DPSEL bit.
2. Set the DACR.DAOE0 bit to 1 to start D/A conversion. The conversion result is output from the analog output pin DA0 after the conversion time $t_{DCONV}$ elapses. The conversion result continues to be output until DADR0 is written to again or the DAOE0 bit is set to 0. The output value (reference) is expressed by the following formula:

$$\frac{\text{Setting in DADR0}}{4096} \times \text{VREFH}$$

3. To start conversion again, write another value to DADR0. The conversion result is output after the conversion time $t_{DCONV}$ elapses.
   When the DAADSCR.DAADST bit is 1 (interference reduction between D/A and A/D conversion is enabled), a maximum of one A/D conversion time is required for D/A conversion to start. When ADCLK is faster than the peripheral clock, a longer time might be required.
4. To disable analog output, set the DAOE0 bit to 0.

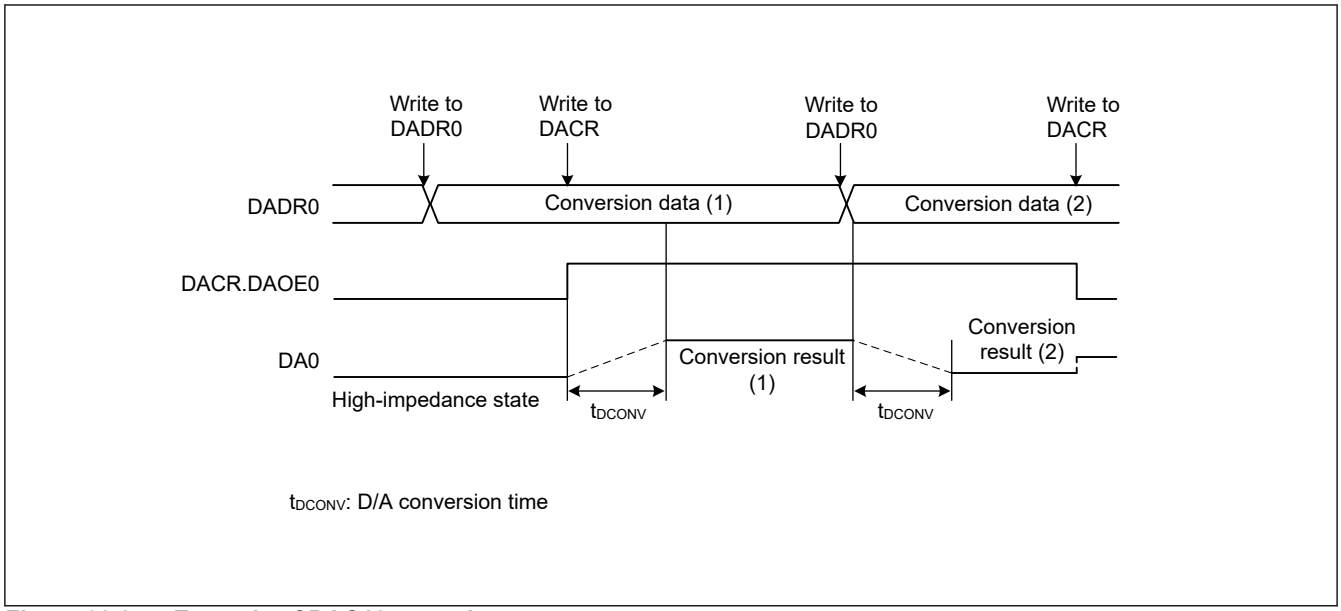



**Figure 38.2 Example of DAC12 operation**

### 38.3.1 Reducing Interference between D/A and A/D Conversion

When D/A conversion starts, the DAC12 generates inrush current. Because the DAC12 and ADC12 share the same analog power supply, the generated inrush current can interfere with ADC12 operation.

While the DAADSCR.DAADST bit is 1, D/A conversion does not start immediately on updating the DADRm register. Instead:

- If the DADRm register data is modified while the ADC12 is halted, D/A conversion starts in 1 PCLKA cycle.
- If the DADRm register data is modified while the ADC12 is performing a 12-bit A/D conversion, D/A conversion starts on A/D conversion completion. Therefore, it takes up to one A/D conversion time period for the DADRm register data update to be reflected as the D/A conversion circuit output. Until the D/A conversion completes, the DADRm register value does not correspond to the analog output value.

When the DAADSCR.DAADST bit is 1, it is not possible to check through software whether the DADRm register value was D/A-converted.

The following sequence provides an example of channel 0 D/A conversion, in which the DAC12 is synchronized with the ADC12. Figure 38.3 shows the timing of this operation.

To perform D/A conversion on channel 0 in synchronization with the ADC12:

1. Confirm that the ADC12 is halted and set the DAADUSR.AMADSEL0 bit to 1.
2. Confirm that the ADC12 is halted and set the DAADSCR.DAADST bit to 1.
3. Confirm that the ADC12 is halted and set the DACR.DAOE0 bit to 1.
4. Set the DADR0 register. If ADCLK is faster than the peripheral clock, D/A conversion might be delayed for longer than one A/D conversion time.
   - If the ADC12 is halted (ADCSR.ADST = 0) when the DADR0 register is modified, D/A conversion starts in 1 PCLKA cycle.
   - If the 12-bit A/D conversion is in progress (ADCSR.ADST = 1) when the DADR0 register is modified, D/A conversion starts on A/D conversion completion. If the DADR0 register is modified twice during A/D conversion, the first update might not be converted.

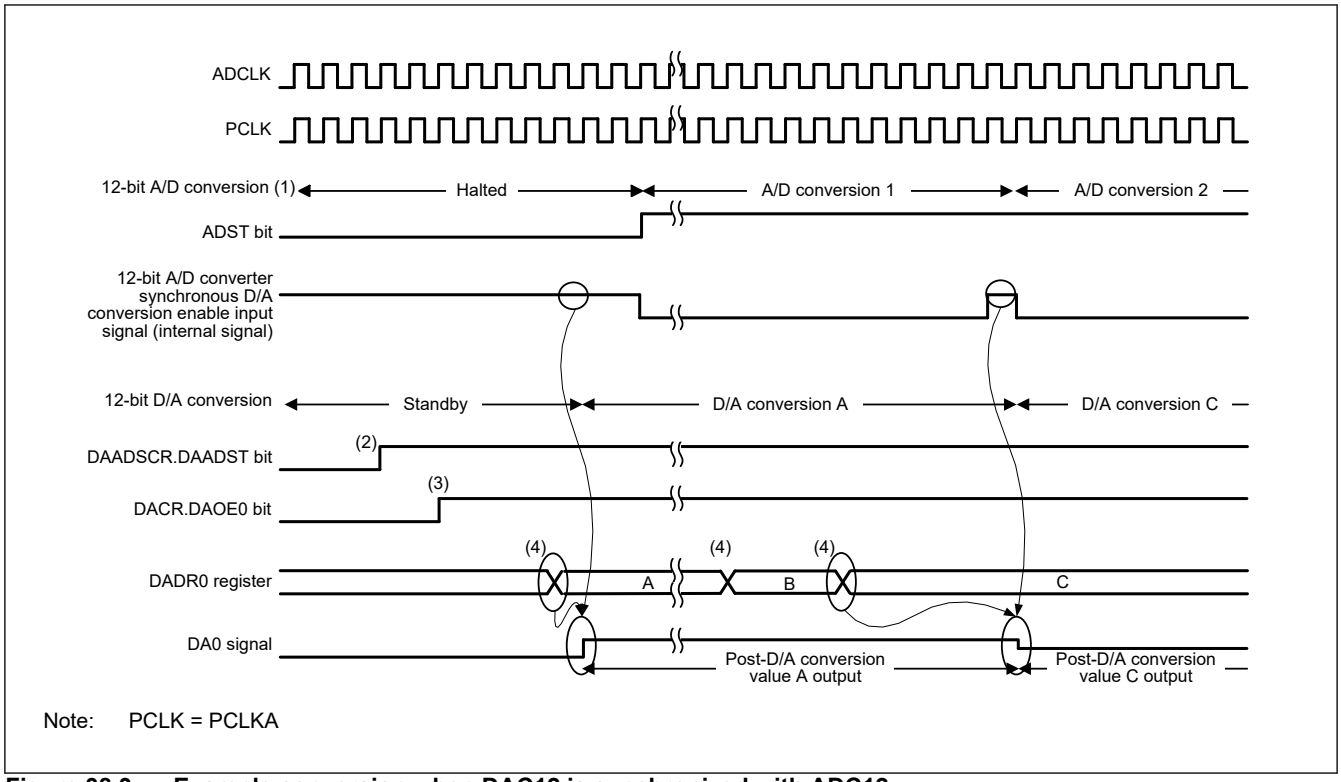



**Figure 38.3 Example conversion when DAC12 is synchronized with ADC12**

When ADCLK is faster than PCLKA, the DAC12 might not be able to capture the synchronous D/A conversion enable input signal from the ADC12 during the 1 ADCLK cycle that is output between A/D conversion 1 and A/D conversion 2, as shown in Figure 38.4. In this case, post-D/A conversion value A is continuously output as the DA0 signal.

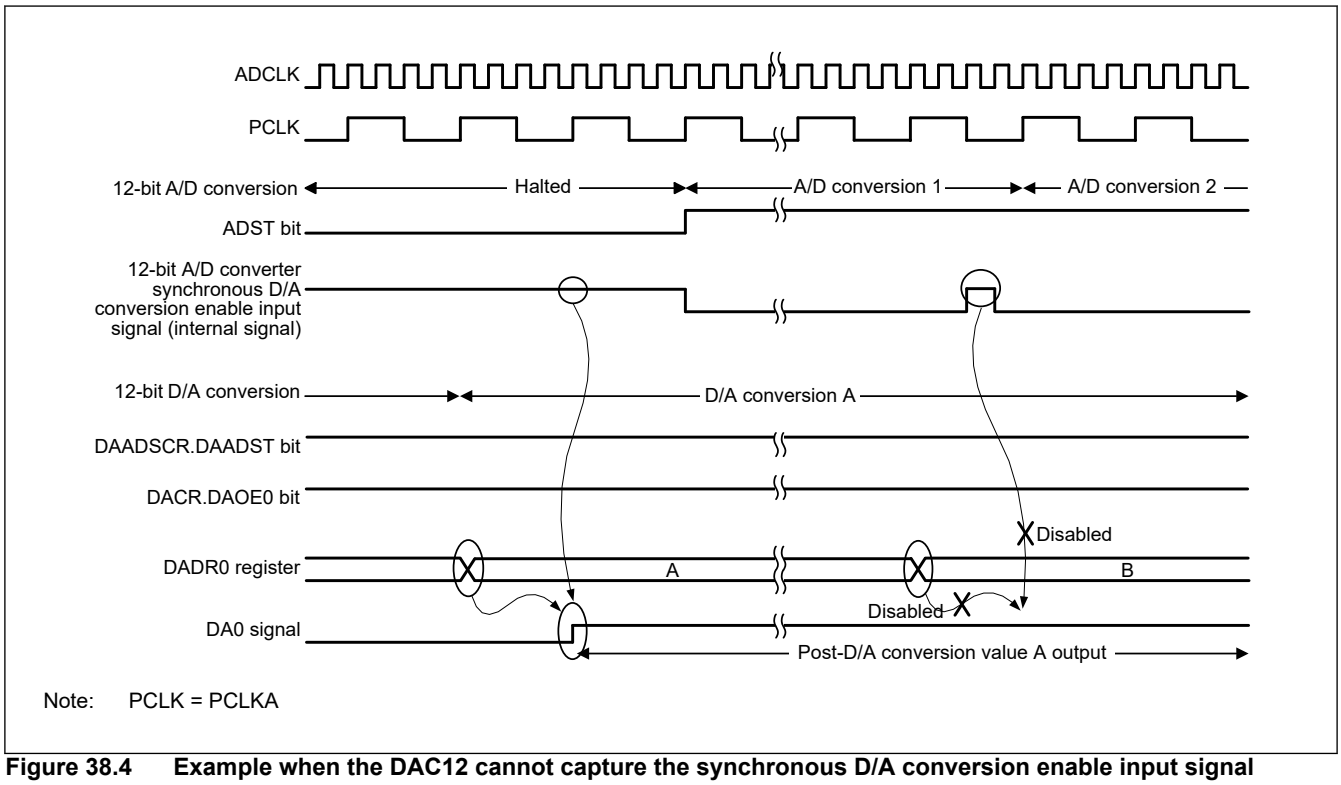



**Figure 38.4 Example when the DAC12 cannot capture the synchronous D/A conversion enable input signal from the ADC12**

## 38.4 Event Link Operation Setting Procedure

This section describes the procedures used in event link operation.

### 38.4.1 DA0 Event Link Operation Setting Procedure

To set up DA0 event link operation:

1. Set the DADPR.DPSEL bit and the data for D/A conversion in the DADR0 register.
2. Set the ELC_DA0 event signal to be linked to each peripheral module in the ELSR12 register.
3. Set the ELCR.ELCON bit to 1. This enables event link operation for all modules with the event link function selected.
4. Set the event output source module to activate the event link. After the event is output from the module, the DACR.DAOE0 bit becomes 1, and D/A conversion starts on channel 0.
5. Set the ELSR12 register to 0x0000 to stop event link operation of DAC12 channel 0. All event link operation is stopped when the ELCR.ELCON bit is set to 0.

### 38.4.2 DA1 Event Link Operation Setting Procedure

To set up DA1 event link operation:

1. Set the DADPR.DPSEL bit and set the data for D/A conversion in the DADR1 register.
2. Set the ELC_DA1 event signal to be linked to each peripheral module in the ELSR13 register.
3. Set the ELCR.ELCON bit to 1. This enables the event link operation for all modules with the event link function selected.
4. Set the event output source module to activate the event link. After the event is output from the module, the DACR.DAOE1 bit is becomes 1, and D/A conversion starts on channel 1.
5. Set the ELSR13 register to 0x0000 to stop event link operation on DAC12 channel 1. All event link operation is stopped when the ELCR.ELCON bit is set to 0.

## 38.5 Usage Notes on Event Link Operation

- When the event link function is used, do not use the amplifier output function.
- When the event link function is used, set the DACR.DAE bit to 0.
- When the event specified for the ELC_DA0 event signal is generated while a write to the DACR.DAOE0 bit is performed, the write cycle is stopped, and the generated event takes precedence in setting the bit to 1.
- When the event specified for the ELC_DA1 event signal is generated while a write to the DACR.DAOE1 bit is performed, the write cycle is stopped, and the generated event takes precedence in setting the bit to 1.
- Use of the event link function is prohibited when the DAADSCR.DAADST bit is set to 1 to reduce interference between D/A and A/D conversions.

## 38.6 Usage Notes

### 38.6.1 Settings for the Module-Stop Function

DAC12 operation can be disabled or enabled using the Module Stop Control Register. The DAC12 is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

### 38.6.2 DAC12 Operation in the Module-Stop State

When the MCU enters the module-stop state with D/A conversion enabled, the D/A outputs are retained, and the analog power supply current is the same as during D/A conversion. If the analog power supply current must be reduced in the module-stop state, disable D/A conversion by setting the DACR.DAOE1, DAOE0, and DAE bits to 0.

### 38.6.3 DAC12 Operation in Software Standby Mode

When the MCU enters Software Standby mode with D/A conversion enabled, the D/A outputs are retained, and the analog power supply current is the same as during D/A conversion. If the analog power supply current must be reduced in Software Standby mode, disable D/A conversion by setting the DACR.DAOE1, DAOE0, and DAE bits to 0.

### 38.6.4 Constraint on Entering Deep Software Standby Mode

When the MCU enters Deep Software Standby mode with D/A conversion enabled, the outputs of the DAC12 are placed in a high impedance state.

### 38.6.5 Initialization Procedure with the Output Amplifier

Use the following initialization procedures with the output amplifier. The example shows the case for channel 0.

To initialize the DAC12 with the output amplifier:

1. Write 0x0000 to the DADR0 register.
2. Set the DAASWCR.DAASW0 bit to 1.
3. Set the DAAMPCR.DAAMP0 bit to 1.
4. Set the DACR.DAE bit or the DACR.DAOE0 bit to 1 to start operation of the amplifier.
5. Clear the DAASWCR.DAASW0 bit to 0 after waiting for the duration of D/A conversion time $t_{DCONV}$.
6. Write the value to be converted in the DADR0 register.

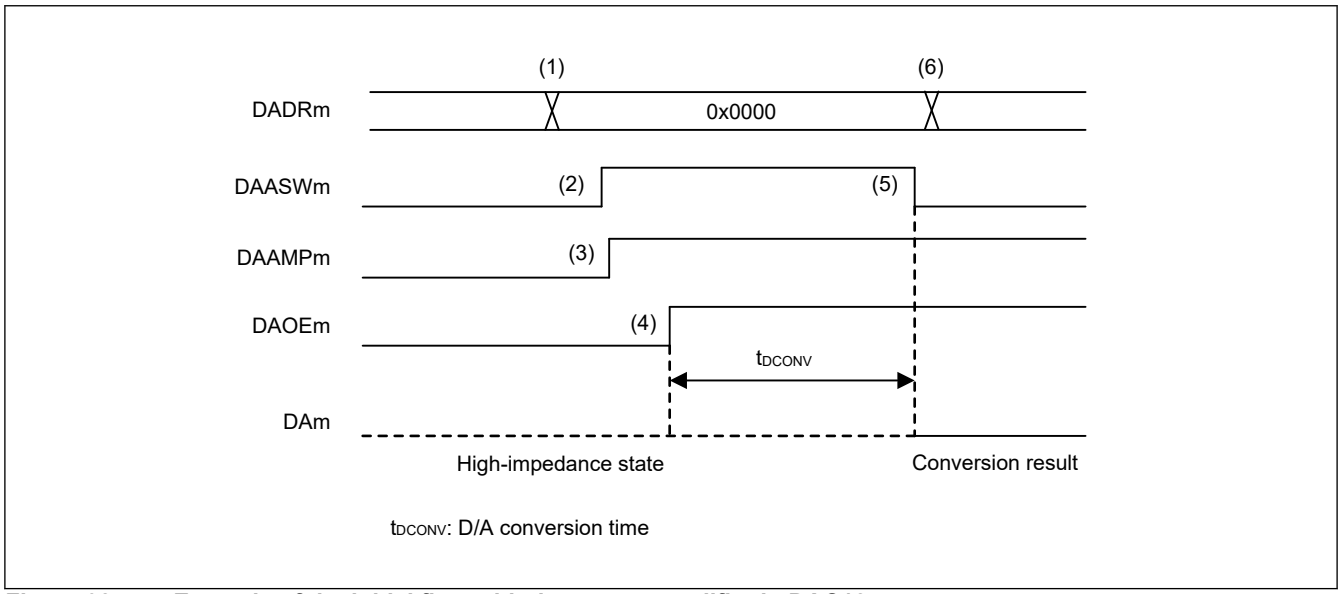



**Figure 38.5 Example of the initial flow with the output amplifier in DAC12**

While the amplifier is running, clearing the DACR.DAE and DACR.DAOE0 bits to 0 allows the amplifier to stop operation. To use the amplifier again, repeat steps 1 to 6.

### 38.6.6 Constraint on Usage When Interference Reduction between D/A and A/D Conversion Is Enabled

When the DAADSCR.DAADST bit is 1, enabling interference reduction between D/A and A/D conversion, do not place the ADC12 in the module-stop state. Doing so can halt D/A conversion in addition to A/D conversion.

# 39. Temperature Sensor (TSN)

## 39.1 Overview

The on-chip Temperature Sensor (TSN) determines and monitors the die temperature for reliable operation of the device. The sensor outputs a voltage directly proportional to the die temperature, and the relationship between the die temperature and the output voltage is fairly linear. The output voltage is provided to the ADC12 for conversion and can be further used by the end application.

Table 39.1 lists the TSN specifications, and Figure 39.1 shows a block diagram.

**Table 39.1 TSN specifications**

| Item | Description |
|---|---|
| Temperature sensor voltage output | Temperature sensor outputs a voltage to the 12-bit A/D converter |
| Module-stop function | Module-stop state can be set to reduce power consumption |
| Temperature sensor calibration data | Reference data measured for each chip at factory shipment is stored in a register |
| TrustZone Filter | Security attribution can be set |

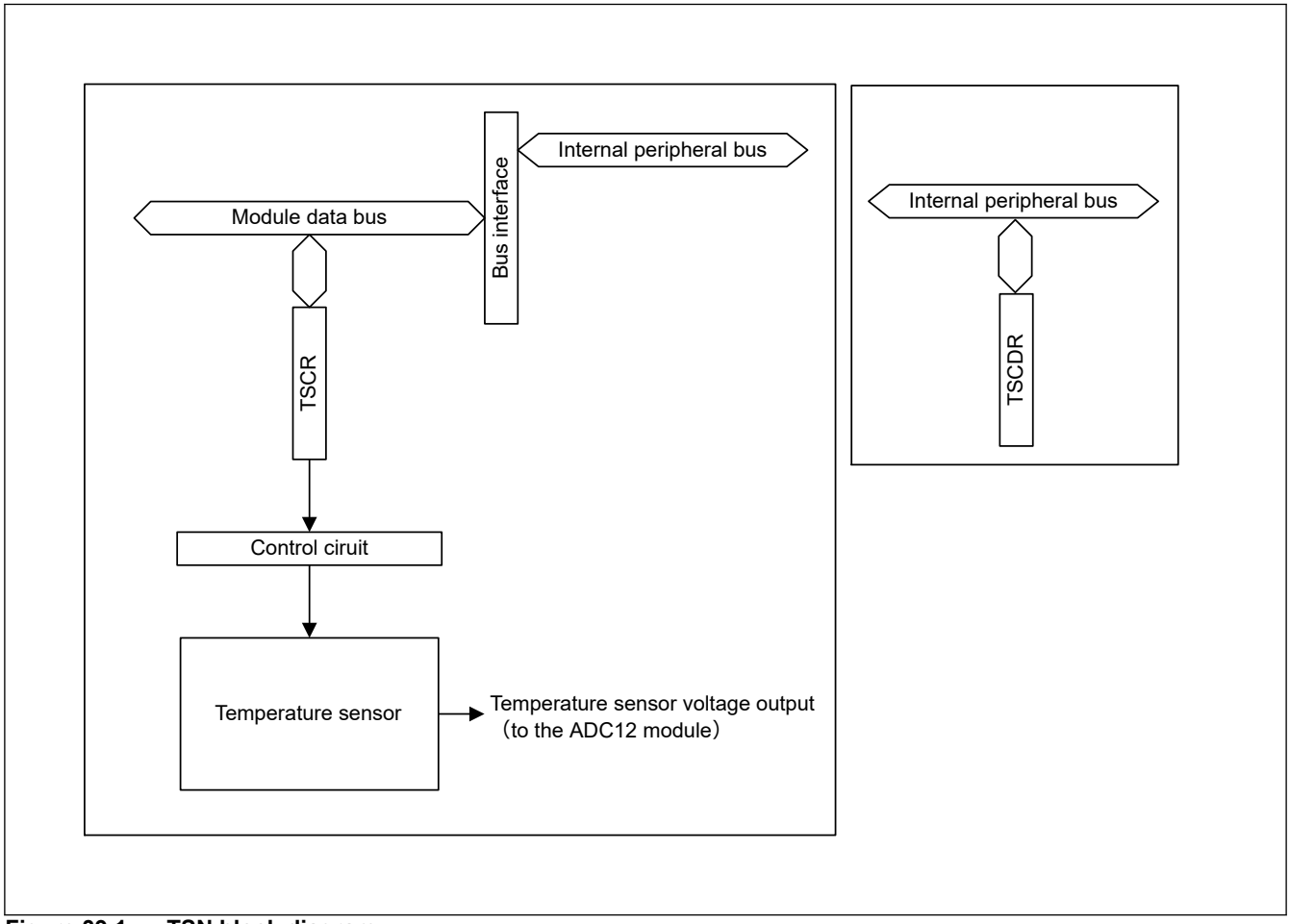



**Figure 39.1 TSN block diagram**

## 39.2 Register Descriptions

### 39.2.1 TSCR : Temperature Sensor Control Register

Base address: TSN = 0x400F_3000

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | TSEN | — | — | TSOE | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 4 | TSOE | Temperature Sensor Output Enable<br>0: Disable output from the temperature sensor to the ADC12<br>1: Enable output from the temperature sensor to the ADC12 | R/W |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | TSEN | Temperature Sensor Enable<br>0: Stop the temperature sensor<br>1: Start the temperature sensor. | R/W |

The TSCR is a register which controls the temperature sensor. The timing constraints shown in Figure 39.3 apply to the settings of the TSCR register.

#### TSOE bit (Temperature Sensor Output Enable)

The TSOE bit enables or disables the temperature sensor output to ADC12.

#### TSEN bit (Temperature Sensor Enable)

The TSEN bit starts or stops the temperature sensor.

### 39.2.2 TSCDR : Temperature Sensor Calibration Data Register

Base address: TSD = 0x407F_B000

Offset Address: 0x017C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Bit field: | TSCDR[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | Chip-specific value | | | | | | | | | | | | | | | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | TSCDR[15:0] | Temperature Sensor Calibration Data<br>Chip-specific value | R |
| 31:16 | — | These bits are read as 0. | R |

The TSCDR register stores temperature sensor calibration data measured for each chip at factory shipment.

Temperature sensor calibration data is the output voltage of the temperature sensor under the conditions Tj = 127°C and AVCC0 = VREFH0 = 3.3 V converted to a digital value by the 12-bit A/D converter.

The TSCDR register is a read-only 32-bit register. Read from this register in 32-bit units.

Temperature sensor calibration data is stored in the lower 12 bits of the TSCDR register.

## 39.3 Using the Temperature Sensor

The temperature sensor outputs a voltage that varies with the temperature. This voltage is converted to a digital value by the 12-bit A/D converter. To obtain the die temperature, convert this value into the temperature.

### 39.3.1 Preparation for Using the Temperature Sensor

The ambient temperature (T) is proportional to the temperature sensor voltage output (Vs), so ambient temperature is calculated with the following formula:

T = (Vs - V1) / slope + T1

- T: Ambient temperature of MCU as calculation result (°C)
- Vs: Voltage output by the temperature sensor on temperature measurement (V)
- T1: Temperature experimentally measured at one point (°C)
- V1: Voltage output by the temperature sensor on measurement of T1 (V)
- T2: Temperature experimentally measured at a second point (°C)
- V2: Voltage output by the temperature sensor on measurement of T2 (V)
- Slope: Temperature gradient of the temperature sensor (V / °C), slope = (V2 - V1) / (T2 -T1)

Characteristics vary between sensors, so Renesas recommends measuring two different sample temperatures as follows:

1. Use the 12-bit A/D converter to measure the voltage V1 output by the temperature sensor at temperature T1.
2. Again use the 12-bit A/D converter to measure the voltage V2 output by the temperature sensor at a different temperature T2.
3. Obtain the temperature gradient (slope = (V2 - V1) / (T2 - T1)) from these results.
4. Subsequently, obtain temperatures by substituting the slope into the formula for the temperature characteristic (T = (Vs - V1) / slope + T1).

If you are using the temperature gradient given in section 47, Electrical Characteristics, use the A/D converter to measure the voltage V1 output by the temperature sensor at temperature T1, then calculate the temperature characteristic using the following formula:

T = (Vs - V1) / slope + T1

Note: This method produces less accurate temperatures than measurement at two points.

In this MCU, the TSCDR register stores the temperature value (CAL127) of the temperature sensor measured under the condition Ta = Tj = 127°C and AVCC0 = VREFH0 = 3.3 V. If you use this value as the sample measurement result at the first point, you can omit the preparation before using the temperature sensor.

V1 is calculated from CAL127:

V1 = 3.3 × CAL127 / 4096 [V] (In case of 12 bit accuracy)

Using this value, the measured temperature can be calculated according to the following formula:

T = (Vs - V1) / slope + 127 [°C]

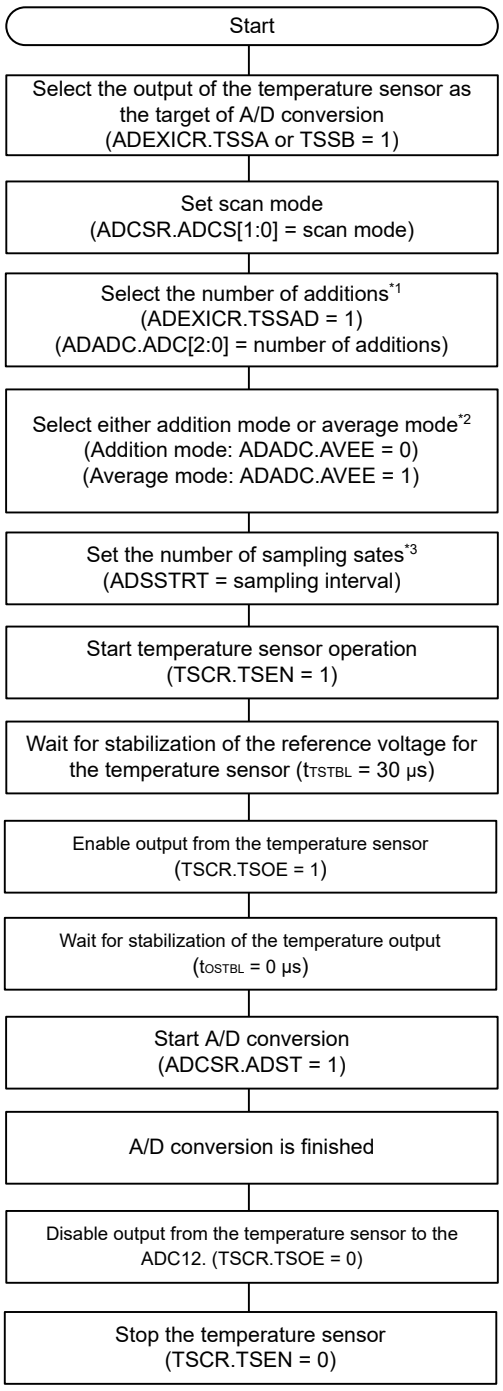
- T: Ambient temperature of MCU as calculation result (°C)
- Vs: Voltage output by the temperature sensor when the temperature is measured (V)
- V1: Voltage output by the temperature sensor when Ta = Tj = 127°C and AVCC0 = VREFH0 = 3.3 V (V)
- Slope: Temperature gradient of the temperature sensor*1 / 1000 (V/°C)

Note 1. See section 47, Electrical Characteristics

### 39.3.2 Procedures for Using the Temperature Sensor

Figure 39.2 shows the procedure for using the TSN.

For details, see section 37, 12-Bit A/D Converter (ADC12).




Note 1. This setting is not required if addition/average mode is not set.
Note 2. The ADADC.ADC[2:0] setting is limited to some values in additions/average mode. For details on the available ADADC.ADC[2:0] settings in additions/average mode, see section 37, 12-Bit A/D Converter (ADC12).
Note 3. Set the sampling time to more than the value described in section 47, Electrical Characteristics.

**Figure 39.2 Procedure example for using the TSN**

Figure 39.3 shows the timing from the start of temperature sensor operation until the completion of A/D conversion when the ADC12 is in single scan mode (the conversion target is the temperature sensor output only). The times shown in the figure are described in Table 39.2

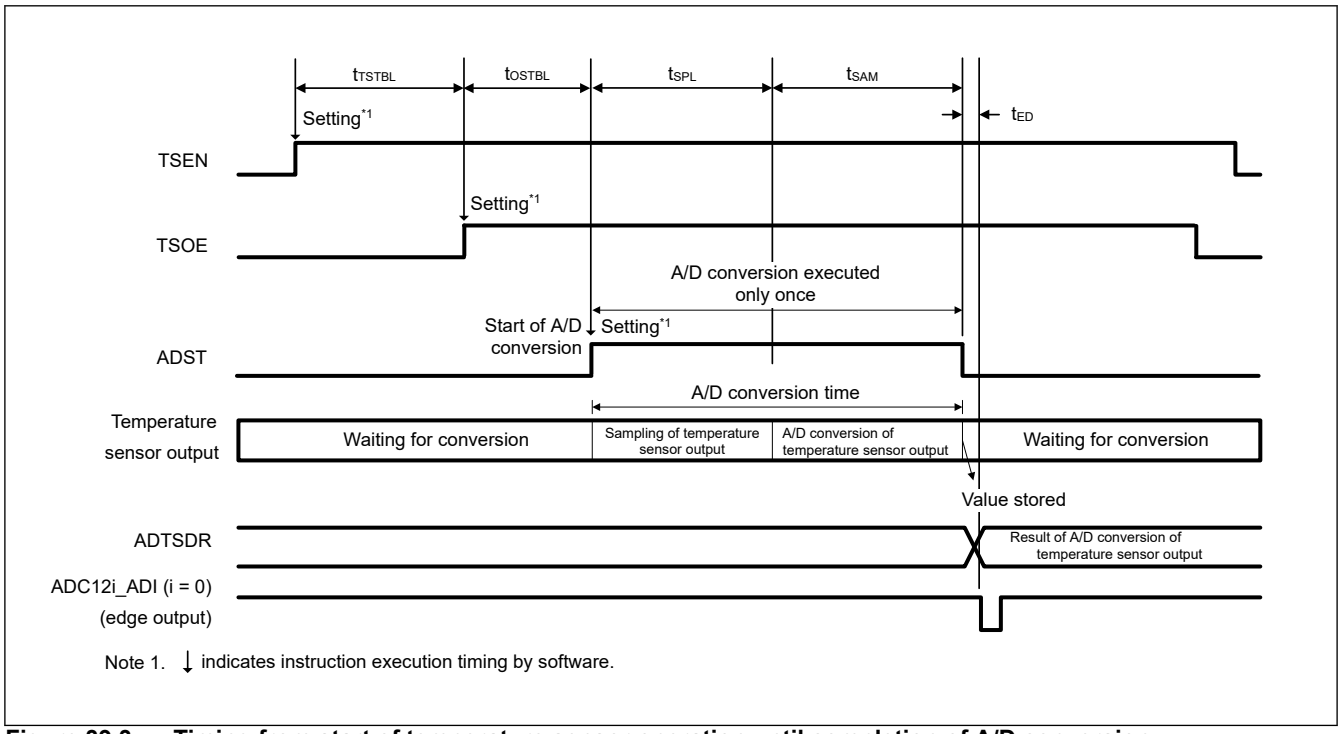



**Figure 39.3 Timing from start of temperature sensor operation until completion of A/D conversion**

**Table 39.2 Time until completion of A/D conversion after start of temperature sensor operation**

| Parameter | Symbol | Time |
|---|---|---|
| Wait time for temperature sensor reference voltage stabilization | $t_{TSTBL}$ | 30 µs (min) |
| Wait time for temperature sensor output stabilization | $t_{OSTBL}$ | 0 µs (min) |
| A/D converter input sampling time | $t_{SPL}$ | ADSSTRn setting × ADCLK cycles |
| A/D conversion time | $t_{SAM}$ | See the table in section 37.3.6. Analog Input Sampling and Scan Conversion Time. |
| Scan conversion end delay | $t_{ED}$ | |

## 39.4 Usage Notes

### 39.4.1 Settings for the Module-Stop Function

TSN operation can be disabled or enabled using the associated bit in Module Stop Control Register D (MSTPCRD). The TSN is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

# 40. Capacitive Touch Sensing Unit (CTSU)

## 40.1 Overview

The Capacitive Touch Sensing Unit (CTSU) measures the electrostatic capacitance of the touch sensor. Changes in the electrostatic capacitance are determined by software that enables the CTSU to detect whether a finger is in contact with the touch sensor. The electrode surface of the touch sensor is usually enclosed with an electrical conductor so that a finger does not come into direct contact with the electrode.

As Figure 40.1 shows, electrostatic capacitance (parasitic capacitance) exists between the electrode and the surrounding conductors. Because the human body is an electrical conductor, when a finger is placed close to the electrode, the value of electrostatic capacitance increases.

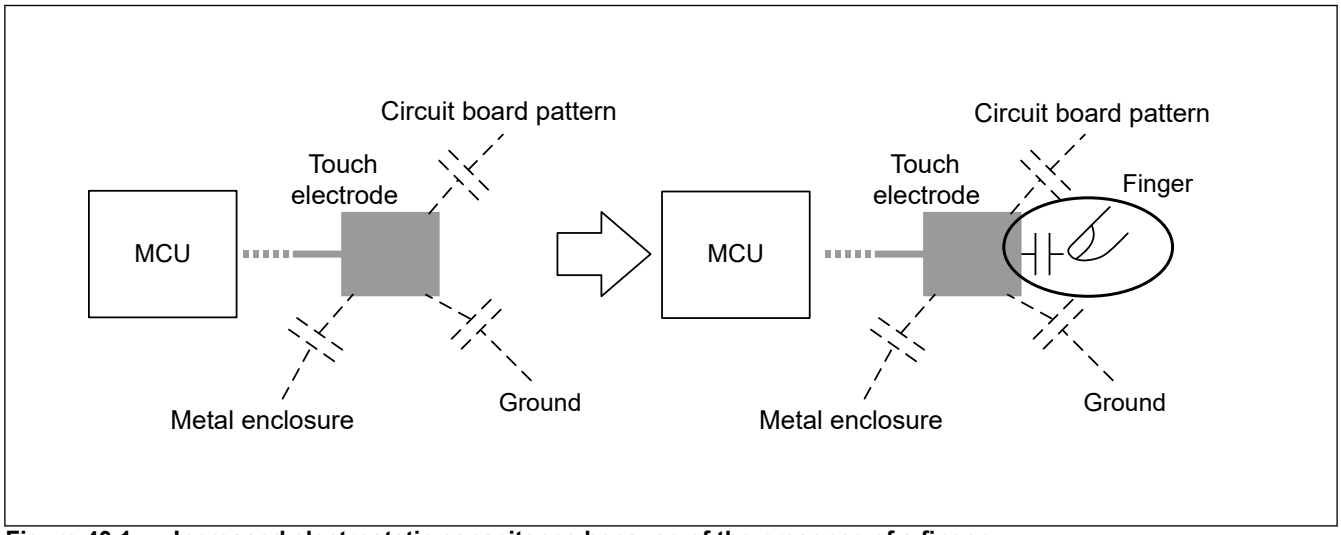



**Figure 40.1 Increased electrostatic capacitance because of the presence of a finger**

Electrostatic capacitance is detected by the self-capacitance and mutual capacitance methods. In the self-capacitance method, the CTSU detects electrostatic capacitance generated between a finger and a single electrode. In the mutual capacitance method, two electrodes are used one as a transmit electrode and the other as a receive electrode, and the CTSU detects the change in the electrostatic capacitance generated between the two when a finger is placed close to them.

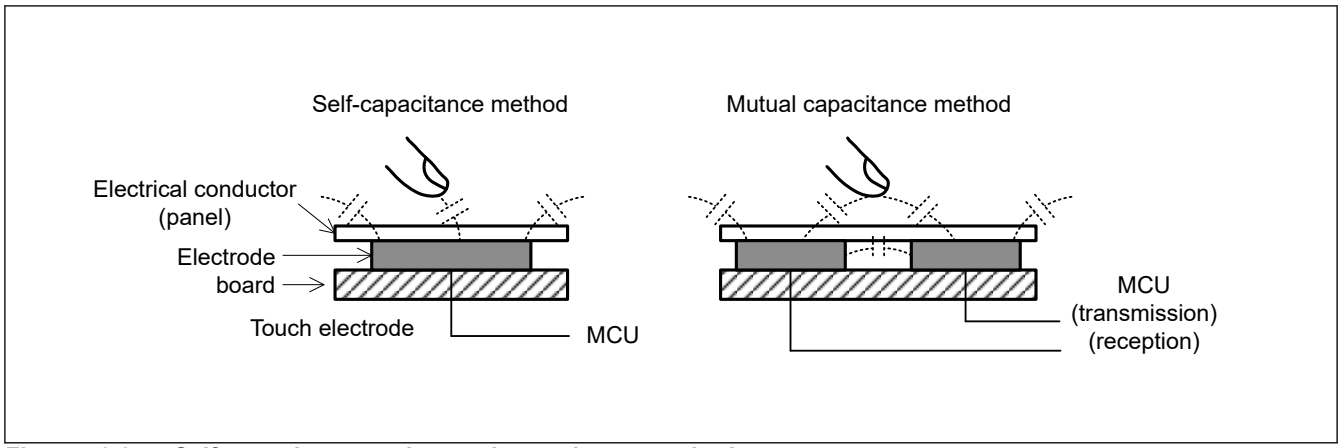



**Figure 40.2 Self-capacitance and mutual capacitance methods**

Electrostatic capacitance is measured by counting a clock signal whose frequency changes according to the amount of charged or discharged current, for a specified period. For details on the measurement principles of the CTSU, see section 40.3.1. Principles of Measurement Operation.

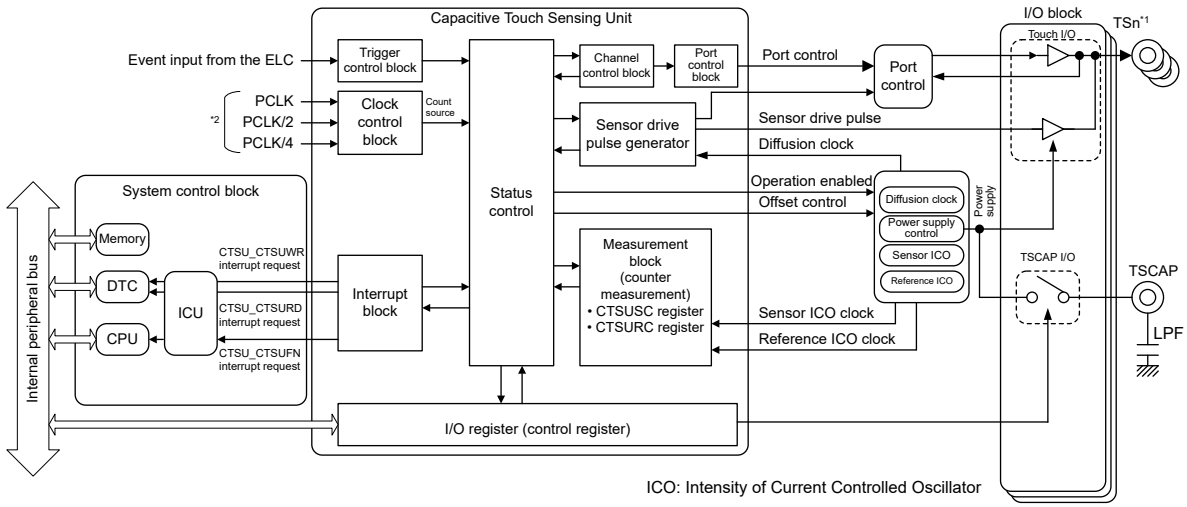
Table 40.1 lists the CTSU specifications and Figure 40.3 shows a block diagram.

**Table 40.1 CTSU specifications**

| Parameter | | Specifications |
|---|---|---|
| Operating clock | | PCLKB, PCLKB/2, or PCLKB/4 |
| Pins | Electrostatic capacitance measurement | 12 channels |
| | TSCAP | Low Pass Filter (LPF) connection pin |
| Measurement modes | Self-capacitance single scan mode | Electrostatic capacitance is measured on one channel using the self-capacitance method. |
| | Self-capacitance multiscan mode | Electrostatic capacitance is measured successively on multiple channels using the self-capacitance method. |
| | Mutual capacitance full scan mode | Electrostatic capacitance is measured successively on multiple channels using the mutual capacitance method. |
| Calibration mode | | Improvement of measured value accuracy by calibration of sensor ICO |
| Noise prevention | | Synchronous noise prevention, high-pass noise prevention |
| Measurement start conditions | | • Software trigger<br>• External trigger (ELC_CTSU from the Event Link Controller (ELC)) |
| TrustZone Filter | | Security attribution can be set |

As Figure 40.3 shows, the CTSU consists of the following components:

- Status control block
- Trigger control block
- Clock control block
- Channel control block
- Port control block
- Sensor drive pulse generator
- Measurement block
- Interrupt block
- I/O registers




Note 1. TSn = TS01 to TS12
Note 2. PCLK = PCLKB

**Figure 40.3 CTSU block diagram**

**Table 40.2 CTSU I/O pins**

| Pin name | I/O | Function |
|---|---|---|
| TS01 to TS12 | Input | Electrostatic capacitive measurement pins (touch pins) |
| TSCAP | — | LPF connection pin |

## 40.2 Register Descriptions

### 40.2.1 CTSUCR0 : CTSU Control Register 0

Base address: CTSU = 0x400D_0000

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUTXVSEL | — | — | CTSUINIT | CTSUIOC | CTSUSNZ | CTSUCAP | CTSUSTRT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CTSUSTRT | CTSU Measurement Operation Start<br>0: Stop measurement operation*1<br>1: Start measurement operation | R/W |
| 1 | CTSUCAP | CTSU Measurement Operation Start Trigger Select<br>0: Software trigger<br>1: External trigger | R/W |
| 2 | CTSUSNZ | CTSU Wait State Power-Saving Enable<br>0: Disable power-saving function during wait state<br>1: Enable power-saving function during wait state | R/W |
| 3 | CTSUIOC | CTSU Transmit Pin Control<br>0: The TS pins are driven low<br>1: The TS pins are driven high | R/W |
| 4 | CTSUINIT | CTSU Control Block Initialization<br>Writing 1 to this bit initializes the CTSU control block and CTSUSC, CTSURC, CTSUMCH0, CTSUMCH1, and CTSUST registers. This bit is read as 0. | W |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | CTSUTXVSEL | CTSU Transmission Power Supply Select<br>0: TSCAP is selected as the power-supply voltage for the transmission pins.<br>1: VCL is selected as the power-supply voltage for the transmission pins. | R/W |

Note 1. When the CTSU is not used, fix the value of this bit to 0.

Only set the CTSUCAP, CTSUSNZ, CTSUTXVSEL2 and CTSUTXVSEL bits when the CTSUSTRT bit is 0. These bits can be set at the same time that measurement operation starts.

#### CTSUSTRT bit (CTSU Measurement Operation Start)

The CTSUSTRT bit specifies whether CTSU operation starts or stops. When the CTSUSTRT bit is 0, measurement starts when the software writes 1 to the CTSUSTRT bit (software trigger) and stops when the hardware clears the CTSUSTRT bit to 0. When the CTSUSTRT bit is 1, the CTSU waits for an external trigger by writing 1 to the CTSUSTRT bit, and measurement starts on the rising edge of the external trigger. When measurement is stopped, the CTSU waits for the next external trigger and operation continues.

Table 40.3 lists the CTSU states.

**Table 40.3 CTSU status (1 of 2)**

| CTSUSTRT bit | CTSUCAP bit | CTSU state |
|---|---|---|
| 0 | 0 | Stopped |
| 0 | 1 | Stopped |

**Table 40.3 CTSU status (2 of 2)**

| CTSUSTRT bit | CTSUCAP bit | CTSU state |
|---|---|---|
| 1 | 0 | Measurement in progress |
| 1 | 1 | Measurement in progress and waiting for an external trigger*1 |

Note 1. The state can be read from the CTSUST.CTSUSTC[2:0] flags as follows:
During measurement: CTSUST.CTSUSTC[2:0] flags ≠ 000b
While waiting for an external trigger: CTSUST.CTSUSTC[2:0] flags = 000b
When the CTSU is not used, fix the value of this bit to 0.

If the software sets the CTSUSTRT bit to 1 when the bit is already 1, the write is ignored and operation continues. To force operation to stop through the software when the CTSUSTRT bit is 1, set the CTSUSTRT bit to 0 and the CTSUINIT bit to 1 at the same time.

### CTSUCAP bit (CTSU Measurement Operation Start Trigger Select)

The CTSUCAP bit specifies the measurement start condition. For details, see CTSUSTRT bit (CTSU Measurement Operation Start).

### CTSUSNZ bit (CTSU Wait State Power-Saving Enable)

The CTSUSNZ bit enables or disables power-saving operation during a wait state. It can also suspend the CTSU power supply, which decreases power consumption during the wait state. In the suspended state, the CTSU power supply is turned off while the external TSCAP is still charged.

Table 40.4 shows the CTSU power supply state control.

**Table 40.4 CTSU power supply state control**

| CTSUCR1.CTSUPON bit | CTSUSNZ bit | CTSUCAP bit | CTSUSTRT bit | CTSU power supply state |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Stopped |
| 1 | 0 | — | — | Operating |
| 1 | 1 | 0 | 0 | Suspended |

Note: Other settings are prohibited.

To start measurement from the suspended state, set the CTSUSNZ bit to 0, and then set the CTSUSTRT bit to 1. To suspend the module after measurement stops, set the CTSUSNZ bit to 1.

### CTSUIOC bit (CTSU Transmit Pin Control)

The CTSUIOC bit selects the logic level of the TS pin when the CTSUERRS.CTSUTSOD bit is set to 1. This bit setting is ignored when the CTSUTSOD bit is 0.

### CTSUINIT bit (CTSU Control Block Initialization)

Write 1 to the CTSUINIT bit to initialize the internal control registers. To force the current operation to stop, set the CTSUSTRT bit to 0 and the CTSUINIT bit to 1 at the same time. This stops the operation and initializes the internal control registers.

Do not write 1 to the CTSUINIT bit when the CTSUSTRT bit is 1.

### CTSUTXVSEL bit (CTSU Transmission Power Supply Select)

The CTSUTXVSEL bit is used to switch the power supply for the transmit buffer in mutual capacitance full scan mode.

Set this bit to 0 for any other mode or when the VCC voltage is lower than 2.7 V. This bit switches the power supply for touch I/O, which is set for transmission by the CTSUCHTRCn registers. Table 40.5 lists the power supply for the TSm pin. When the VCC voltage fluctuates greatly due to switching of the output buffer, switching to the internal logic power supply can reduce the effect on the voltage fluctuation.

**Table 40.5 Power supplied to the TSm pins**

| Setting of CTSUCHTRCn Register | CTSUTXVSEL bit | Power supply of TSm pins |
|---|---|---|
| 0 (Reception) | * | VCC |
| 1 (Transmission) | 0 (VCC) | |
| | 1 (Internal logic power supply) | Internal logic power supply |

## 40.2.2 CTSUCR1 : CTSU Control Register 1

Base address: CTSU = 0x400D_0000

Offset address: 0x01

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUMD[1:0] | | CTSUCLK[1:0] | | CTSU ATUNE1 | — | CTSU CSW | CTSU PON |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CTSUPON | CTSU Power Supply Enable<br>0: Power off the CTSU<br>1: Power on the CTSU | R/W |
| 1 | CTSUCSW | CTSU LPF Capacitance Charging Control<br>This bit controls charging of the LPF capacitance connected to the TSCAP pin.<br>0: Turn off capacitance switch<br>1: Turn on capacitance switch | R/W |
| 2 | — | This bit is read as 0. The write value should be 0. | R/W |
| 3 | CTSUATUNE1 | CTSU Power Supply Capacity Adjustment<br>0: Normal output<br>1: High-current output | R/W |
| 5:4 | CTSUCLK[1:0] | CTSU Operating Clock Select<br>0 0: PCLKB<br>0 1: PCLKB/2 (PCLKB divided by 2)<br>1 0: PCLKB/4 (PCLKB divided by 4)<br>1 1: Setting prohibited | R/W |
| 7:6 | CTSUMD[1:0] | CTSU Measurement Mode Select<br>0 0: Self-capacitance single scan mode<br>0 1: Self-capacitance multiscan mode<br>1 0: Setting prohibited<br>1 1: Mutual capacitance full scan mode | R/W |

Only set the CTSUCR1 register when the CTSUCR0.CTSUSTRT bit is 0.

### CTSUPON bit (CTSU Power Supply Enable)

The CTSUPON bit controls the power supply to the CTSU. Set the CTSUPON and CTSUCSW bits to the same value.

### CTSUCSW bit (CTSU LPF Capacitance Charging Control)

The CTSUCSW bit controls charging of the LPF capacitor connected to the TSCAP pin by turning the capacitance switch on or off. After the capacitance switch is turned on, wait until the capacitance connected to the TSCAP pin is charged for the specified time before starting measurement by setting CTSUCR0.CTSUSTRT to 1. Before starting measurement, use an I/O port to output low to the TSCAP pin, and discharge the existing LPF capacitance. Set the CTSUPON and CTSUCSW bits to the same value.

### CTSUATUNE1 bit (CTSU Power Supply Capacity Adjustment)

The CTSUATUNE1 bit sets the capacity of the CTSU power supply. Normally, set this bit to 0.

### CTSUCLK[1:0] bits (CTSU Operating Clock Select)

The CTSUCLK[1:0] bits select the operating clock.

### CTSUMD[1:0] bits (CTSU Measurement Mode Select)

The CTSUMD[1:0] bits set the measurement mode. For details, see section 40.3.2. Measurement Modes.

## 40.2.3 CTSUSDPRS : CTSU Synchronous Noise Reduction Setting Register

Base address: CTSU = 0x400D_0000

Offset address: 0x02

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | CTSUSOFF | CTSUPRMODE[1:0] | | CTSUPRRATIO[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | CTSUPRRATIO[3:0] | CTSU Measurement Time and Pulse Count Adjustment<br>These bits set the measurement time and the measurement pulse count setting.<br>Recommended setting: 3 (0011b) | R/W |
| 5:4 | CTSUPRMODE[1:0] | CTSU Base Period and Pulse Count Setting<br>These bits set the base pulse count.<br>0 0: 510 pulses<br>0 1: 126 pulses<br>1 0: 62 pulses (recommended setting)<br>1 1: Setting prohibited | R/W |
| 6 | CTSUSOFF | CTSU High-Pass Noise Reduction Function Off Setting<br>This bit turns spectrum diffusion on or off to reduce high-pass noise.<br>0: Turn on<br>1: Turn off | R/W |
| 7 | — | This bit is read as 0. The write value should be 0. | R/W |

Only set the CTSUSDPRS register when the CTSUCR0.CTSUSTRT bit is 0.

### CTSUPRRATIO[3:0] bits (CTSU Measurement Time and Pulse Count Adjustment)

The CTSUPRRATIO[3:0] bits are used to determine the measurement time and the measurement pulse count. These values are calculated using the following formulas, where the base pulse count is determined by the CTSUPRMODE[1:0] setting:

$$\text{Measurement pulse count} = \text{base pulse count} \times (\text{CTSUPRRATIO[3:0] bits} + 1)$$

$$\text{Measurement time} = (\text{base pulse count} \times (\text{CTSUPRRATIO[3:0] bits} + 1) + (\text{base pulse count} - 2) \times 0.25) \times \text{base clock cycle}$$

Note: For details on the base clock cycle, see section 40.2.15. CTSUSO1 : CTSU Sensor Offset Register 1.

### CTSUPRMODE[1:0] bits (CTSU Base Period and Pulse Count Setting)

The CTSUPRMODE[1:0] bits select the number of base pulses that occur during measurement.

### CTSUSOFF bit (CTSU High-Pass Noise Reduction Function Off Setting)

The CTSUSOFF bit turns on or off the function for reducing high-pass noise. Set this bit to 1 to turn the function off.

## 40.2.4 CTSUSST : CTSU Sensor Stabilization Wait Control Register

Base address: CTSU = 0x400D_0000

Offset address: 0x03

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUSST[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | CTSUSST[7:0] | CTSU Sensor Stabilization Wait Control<br>Fix the value of these bits to 00010000b. | R/W |

Only set the CTSUSST register when the CTSUCR0.CTSUSTRT bit is 0.

### CTSUSST[7:0] bits (CTSU Sensor Stabilization Wait Control)

The CTSUSST[7:0] bits set the stabilization wait time for the TSCAP pin voltage. Always fix these bits to 00010000b. If these bits are not set, the TSCAP voltage will be unstable at the start of measurement, and the CTSU will be unable to obtain correct touch measurement results.

## 40.2.5 CTSUMCH0 : CTSU Measurement Channel Register 0

Base address: CTSU = 0x400D_0000

Offset address: 0x04

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | CTSHMCH0[4:0] | | | | |
| Value after reset: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | CTSHMCH0[4:0] | CTSU Measurement Channel 0<br>In self-capacitance single scan mode, these bits set the channel to be measured.<br>Other than when specified, starting measurement operation by setting CTSUCR0.CTSUSTRT to 1 is prohibited after these bits are set.<br>In other measurement modes, these bits indicate the channel that is currently being measured.<br>0x01: TS01<br>0x02: TS02<br>⋮<br>0x0B: TS11<br>0x0C: TS12<br>0x1F: Measurement is being stopped. (Except in self-capacitance single scan mode) | R/W*1 |
| 7:5 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Writing to these bits is only enabled in self-capacitance single scan mode (CTSUCR1.CTSUMD[1:0] bits = 00b).

Only set the CTSUMCH0 register when the CTSUCR0.CTSUSTRT bit is 0.

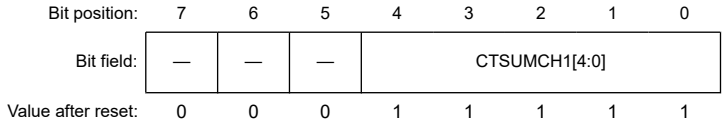
### CTSHMCH0[4:0] bits (CTSU Measurement Channel 0)

In self-capacitance single scan mode, the CTSHMCH0[4:0] bits set the channel to be measured. In this mode, only specify enabled channels (TS01 to TS12). In other modes, these indicate the receive channel that is being measured, and writing to these bits has no effect.

## 40.2.6 CTSUMCH1 : CTSU Measurement Channel Register 1

Base address: CTSU = 0x400D_0000

Offset address: 0x05




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 4:0 | CTSUMCH1[4:0] | CTSU Measurement Channel 1<br>0x01: TS01<br>0x02: TS02<br>⋮<br>0x0B: TS11<br>0x0C: TS12<br>0x1F: Measurement is being stopped. | R |
| 7:5 | — | These bits are read as 0. | R |

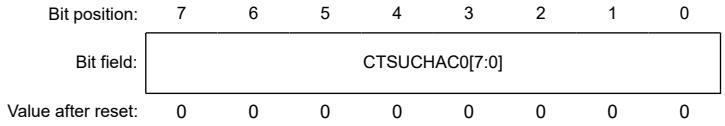
### CTSUMCH1[4:0] bits (CTSU Measurement Channel 1)

In full scan mode, the CTSUMCH1[4:0] bits indicate the transmit channel that is being measured. They are always 0x1F when measurement is stopped, or in self-capacitance single scan and multi-scan modes.

## 40.2.7 CTSUCHAC0 : CTSU Channel Enable Control Register 0

Base address: CTSU = 0x400D_0000

Offset address: 0x06




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | CTSUCHAC0[7:0] | CTSU Channel Enable Control 0<br>These bits select whether the associated TS pin is measured. bit1 is associated with TS01 and bit7 with TS07.<br>0: Do not measure<br>1: Measure | R/W |

Only set the CTSUCHAC0 register when the CTSUCR0.CTSUSTRT bit is 0.

### CTSUCHAC0[7:0] bits (CTSU Channel Enable Control 0)

The CTSUCHAC0[7:0] bits select the receive and transmit pins whose electrostatic capacitance is to be measured.

## 40.2.8 CTSUCHAC1 : CTSU Channel Enable Control Register 1

Base address: CTSU = 0x400D_0000

Offset address: 0x07

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUCHAC1[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | CTSUCHAC1[7:0] | CTSU Channel Enable Control 1<br>These bits select whether the associated TS pin is measured. bit0 is associated with TS08 and bit4 with TS12.<br>0: Do not measure<br>1: Measure | R/W |

Only set the CTSUCHAC1 register when the CTSUCR0.CTSUSTRT bit is 0.

### CTSUCHAC1[7:0] bits (CTSU Channel Enable Control 1)

The CTSUCHAC1[7:0] bits select the receive and transmit pins whose electrostatic capacitance is to be measured.

## 40.2.9 CTSUCHTRC0 : CTSU Channel Transmit/Receive Control Register 0

Base address: CTSU = 0x400D_0000

Offset address: 0x0B

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUCHTRC0[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | CTSUCHTRC0[7:0] | CTSU Channel Transmit/Receive Control 0<br>bit1 is associated with TS01 and bit7 with TS07.<br>0: Reception<br>1: Transmission | R/W |

Only set the CTSUCHTRC0 register when the CTSUCR0.CTSUSTRT bit is 0.

### CTSUCHTRC0[7:0] bits (CTSU Channel Transmit/Receive Control 0)

In full scan mode, the CTSUCHTRC0[7:0] bits allocate reception or transmission to the associated TS pins. The setting is ignored in self-capacitance single scan and multi-scan modes.

## 40.2.10 CTSUCHTRC1 : CTSU Channel Transmit/Receive Control Register 1

Base address: CTSU = 0x400D_0000

Offset address: 0x0C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUCHTRC1[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | CTSUCHTRC1[7:0] | CTSU Channel Transmit/Receive Control 1<br>bit0 is associated with TS08 and bit4 with TS12.<br>0: Reception<br>1: Transmission | R/W |

Only set the CTSUCHTRC1 register when the CTSUCR0.CTSUSTRT bit is 0.

#### CTSUCHTRC1[7:0] bits (CTSU Channel Transmit/Receive Control 1)

In full scan mode, the CTSUCHTRC1[7:0] bits allocate reception or transmission to the associated TS pins. The setting is ignored in self-capacitance single scan and multiscan modes.

## 40.2.11 CTSUDCLKC : CTSU High-Pass Noise Reduction Control Register

Base address: CTSU = 0x400D_0000

Offset address: 0x10

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | CTSUSSCNT[1:0] | | — | — | CTSUSSMOD[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | CTSUSSMOD[1:0] | CTSU Diffusion Clock Mode Select<br>Set these bits to 00b. | R/W |
| 3:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5:4 | CTSUSSCNT[1:0] | CTSU Diffusion Clock Mode Control<br>Set these bits to 11b. | R/W |
| 7:6 | — | These bits are read as 0. The write value should be 0. | R/W |

Only set the CTSUDCLKC register when the CTSUCR0.CTSUSTRT bit is 0.

#### CTSUSSMOD[1:0] bits (CTSU Diffusion Clock Mode Select)

The CTSUSSMOD[1:0] bits set the mode of the spectrum diffusion clock for high-pass noise reduction. When using the high-pass function, always fix these bits to 00b. If these bits are not set, the CTSU will be unable to effectively reduce high-pass noise.

#### CTSUSSCNT[1:0] bits (CTSU Diffusion Clock Mode Control)

The CTSUSSCNT[1:0] bits adjust the amount of spectrum diffusion applied to reduce high-pass noise. When using the high-pass noise reduction function, always fix these bits to 11b. If these bits are not set, touch measurement might be performed incorrectly.

## 40.2.12 CTSUST : CTSU Status Register

Base address: CTSU = 0x400D_0000

Offset address: 0x11

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUPS | CTSUROVF | CTSUSOVF | CTSUDTSR | — | CTSUSTC[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | CTSUSTC[2:0] | CTSU Measurement Status Counter<br>These counters indicate the current measurement status.<br>0 0 0: Status 0<br>0 0 1: Status 1<br>0 1 0: Status 2<br>0 1 1: Status 3<br>1 0 0: Status 4<br>1 0 1: Status 5 | R |
| 3 | — | This bit is read as 0. The write value should be 0. | R/W |
| 4 | CTSUDTSR | CTSU Data Transfer Status Flag<br>This flag indicates whether the measurement result stored in the sensor counter and the reference counter was read.<br>0: Read<br>1: Not read | R |
| 5 | CTSUSOVF | CTSU Sensor Counter Overflow Flag<br>This flag indicates an overflow on the sensor counter.<br>0: No overflow occurred<br>1: Overflow occurred | R/W |
| 6 | CTSUROVF | CTSU Reference Counter Overflow Flag<br>This flag indicates an overflow on the reference counter.<br>0: No overflow occurred<br>1: Overflow occurred | R/W |
| 7 | CTSUPS | CTSU Mutual Capacitance Status Flag<br>This flag indicates the measurement status in mutual capacitance full scan mode.<br>0: First measurement<br>1: Second measurement | R |

When using the CTSUCR0.CTSUINIT bit to clear an overflow flag, make sure that the CTSUCR0.CTSUSTRT bit is 0.

### CTSUSTC[2:0] flags (CTSU Measurement Status Counter)

The CTSUSTC[2:0] flags are a counter indicating the current measurement status. For details on each status, see section 40.3.2.2. Status counter.

### CTSUDTSR flag (CTSU Data Transfer Status Flag)

The CTSUDTSR flag indicates whether the measurement result stored in the sensor counter and the reference counter was read. The flag sets to 1 when measurement completes and 0 when the reference counter is read by software or the DTC. The flag can also be cleared using the CTSUCR0.CTSUINIT bit.

### CTSUSOVF flag (CTSU Sensor Counter Overflow Flag)

The CTSUSOVF flag sets to 1 when the sensor counter, CTSUSC, overflows. On overflow, the counter value reads as 0xFFFF. Measurement processing continues for the specified period.

No interrupt occurs on an overflow. To determine the channel on which the overflow occurred, read the measurement result of each channel after measurement completes, signaled by a measurement end interrupt.

This flag is cleared when 0 is written after 1 is read by software. It can also be cleared using the CTSUCR0.CTSUINIT bit.

### CTSUROVF flag (CTSU Reference Counter Overflow Flag)

The CTSUROVF flag sets to 1 when the reference counter, CTSURC, overflows. On overflow, the counter value reads as 0xFFFF. Measurement processing continues for the specified period.

No interrupt occurs on an overflow. To determine the channel on which the overflow occurred, read the measurement result of each channel after measurement completes, signaled by a measurement end interrupt.

This flag is cleared when 0 is written after 1 is read by software. It can also be cleared using the CTSUCR0.CTSUINIT bit.

### CTSUPS flag (CTSU Mutual Capacitance Status Flag)

In mutual capacitance full scan mode, when CTSUCR1.CTSUMD[1:0] = 11b, the CTSUPS flag indicates whether the measurement is the first or second of two measurements for each channel. When measurement is stopped or in other measurement modes, this flag is always 0.

## 40.2.13 CTSUSSC : CTSU High-Pass Noise Reduction Spectrum Diffusion Control Register

Base address: CTSU = 0x400D_0000

Offset address: 0x12

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | CTSUSSDIV[3:0] | | | | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 11:8 | CTSUSSDIV[3:0] | CTSU Spectrum Diffusion Frequency Division Setting<br>These bits specify the spectrum diffusion frequency division setting based on the base clock frequency division setting. | R/W |
| 15:12 | — | These bits are read as 0. The write value should be 0. | R/W |

### CTSUSSDIV[3:0] bits (CTSU Spectrum Diffusion Frequency Division Setting)

The CTSUSSDIV[3:0] bits specify the spectrum diffusion frequency derived from the base clock frequency division setting. To calculate the correct setting for CTSUSSDIV[3:0], see the relationship between base clock frequencies and the settings in Table 40.6.

**Table 40.6 Relationship between base clock frequencies and CTSUSSDIV[3:0] bits settings**

| Base clock frequency fb (MHz) | CTSUSSDIV[3:0] bit setting |
|---|---|
| 4.00 ≤ fb | 0000b |
| 2.00 ≤ fb < 4.00 | 0001b |
| 1.33 ≤ fb < 2.00 | 0010b |
| 1.00 ≤ fb < 1.33 | 0011b |
| 0.80 ≤ fb < 1.00 | 0100b |
| 0.67 ≤ fb < 0.80 | 0101b |
| 0.57 ≤ fb < 0.67 | 0110b |
| 0.50 ≤ fb < 0.57 | 0111b |
| 0.44 ≤ fb < 0.50 | 1000b |
| 0.40 ≤ fb < 0.44 | 1001b |
| 0.36 ≤ fb < 0.40 | 1010b |
| 0.33 ≤ fb < 0.36 | 1011b |
| 0.31 ≤ fb < 0.33 | 1100b |
| 0.29 ≤ fb < 0.31 | 1101b |
| 0.27 ≤ fb < 0.29 | 1110b |
| fb < 0.27 | 1111b |

## 40.2.14 CTSUSO0 : CTSU Sensor Offset Register 0

Base address: CTSU = 0x400D_0000

Offset address: 0x14

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUSNUM[5:0] | | | | | | CTSUSO[9:0] | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9:0 | CTSUSO[9:0] | CTSU Sensor Offset Adjustment<br>These bits adjust the electronic capacitance when the electrode is not being touched.<br>0x000: Current offset is 0.<br>0x001: Current offset is 1.<br>0x002: Current offset is 2.<br>⋮ ⋮<br>0x3FE: Current offset is 1022.<br>0x3FF: Current offset is maximum. | R/W |
| 15:10 | CTSUSNUM[5:0] | CTSU Measurement Count Setting<br>These bits set the number of measurements. | R/W |

### CTSUSO[9:0] bits (CTSU Sensor Offset Adjustment)

The CTSUSO[9:0] bits offset the sensor ICO input current generated from electrostatic capacitance during touch measurement when the electrode is not being touched. This prevents the CTSU sensor counter from overflowing. Set the TS pin that is to be measured next after a CTSU_CTSUWR interrupt occurs.

### CTSUSNUM[5:0] bits (CTSU Measurement Count Setting)

The CTSUSNUM[5:0] bits specify how many times the measurement pulse count specified in the CTSUSDPRS.CTSUPRRATIO[3:0] and CTSUSDPRS.CTSUPRMODE[1:0] bits is repeated during the measurement time. The measurement pulse count is repeated (CTSUSNUM[5:0] bits + 1) times. Set the TS pin that is to be measured next after a CTSU_CTSUWR interrupt is generated.

## 40.2.15 CTSUSO1 : CTSU Sensor Offset Register 1

Base address: CTSU = 0x400D_0000

Offset address: 0x16

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | CTSUICOG[1:0] | | CTSUSDPA[4:0] | | | | | CTSURICOA[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | CTSURICOA[7:0] | CTSU Reference ICO Current Adjustment<br>These bits adjust the input current of the reference ICO.<br>0x00: Current offset is 0<br>0x01: Current offset is 1<br>0x02: Current offset is 2<br>⋮<br>0xFE: Current offset is 254<br>0xFF: Current offset is maximum. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 12:8 | CTSUSDPA[4:0] | CTSU Base Clock Setting<br>These bits are used to generate the base clock.<br>0x00: Operating clock divided by 2*1<br>0x01: Operating clock divided by 4<br>0x02: Operating clock divided by 6<br>0x03: Operating clock divided by 8<br>0x04: Operating clock divided by 10<br>0x05: Operating clock divided by 12<br>0x06: Operating clock divided by 14<br>0x07: Operating clock divided by 16<br>0x08: Operating clock divided by 18<br>0x09: Operating clock divided by 20<br>0x0A: Operating clock divided by 22<br>0x0B: Operating clock divided by 24<br>0x0C: Operating clock divided by 26<br>0x0D: Operating clock divided by 28<br>0x0E: Operating clock divided by 30<br>0x0F: Operating clock divided by 32<br>0x10: Operating clock divided by 34<br>0x11: Operating clock divided by 36<br>0x12: Operating clock divided by 38<br>0x13: Operating clock divided by 40<br>0x14: Operating clock divided by 42<br>0x15: Operating clock divided by 44<br>0x16: Operating clock divided by 46<br>0x17: Operating clock divided by 48<br>0x18: Operating clock divided by 50<br>0x19: Operating clock divided by 52<br>0x1A: Operating clock divided by 54<br>0x1B: Operating clock divided by 56<br>0x1C: Operating clock divided by 58<br>0x1D: Operating clock divided by 60<br>0x1E: Operating clock divided by 62<br>0x1F: Operating clock divided by 64 | R/W |
| 14:13 | CTSUICOG[1:0] | CTSU ICO Gain Adjustment<br>These bits adjust the output frequency gain of the sensor ICO and the reference ICO.<br>0 0: 100% gain<br>0 1: 66% gain<br>1 0: 50% gain<br>1 1: 40% gain | R/W |
| 15 | — | This bit is read as 0. The write value should be 0. | R/W |

Note 1. Do not set the CTSUSDPA[4:0] bits set to 00000b while the high-pass noise reduction function is turned off (CTSUSDPRS.CTSUSOFF bit = 1) in mutual capacitance full scan mode (CTSUCR1.CTSUMD[1:0] bits = 11b).

After a CTSU_CTSUWR interrupt occurs, write first to the CTSUSSC register, next to the CTSUSO0 register, and then to the CTSUSO1 register. The write to the CTSUSO1 register causes a transition to Status 3. (See Table 40.7 and Table 40.8.) Set all of the bits in a single operation when writing to the CTSUSO1 register.

### CTSURICOA[7:0] bits (CTSU Reference ICO Current Adjustment)

The CTSURICOA[7:0] bits adjust the oscillation frequency using the input current of the reference ICO.

### CTSUSDPA[4:0] bits (CTSU Base Clock Setting)

The CTSUSDPA[4:0] bits select the base clock used as the source for the sensor drive pulse by dividing the operating clock. For details on the setting procedure, see section 40.3.2.1. Initial settings flow.

### CTSUICOG[1:0] bits (CTSU ICO Gain Adjustment)

The CTSUICOG[1:0] bits adjust the output frequency gain of the sensor ICO and the reference ICO. Normally, set these bits to 00b for the maximum gain. If changes in the capacitance between when the electrode is touched and when it is not touched greatly exceed the dynamic range of the sensor ICO, adjust the gain appropriately with this setting.

## 40.2.16 CTSUSC : CTSU Sensor Counter

Base address: CTSU = 0x400D_0000

Offset address: 0x18

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUSC[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | CTSUSC[15:0] | CTSU Sensor Counter<br>These bits indicate the measurement result of the sensor ICO. They read 0xFFFF when an overflow occurs. | R |

After a CTSU_CTSURD interrupt occurs, read first from the CTSUSC counter and then from the CTSURC counter.

### CTSUSC[15:0] bits (CTSU Sensor Counter)

The CTSUSC[15:0] bits are configured as an increment counter for the sensor ICO clock. Read these bits after a CTSU_CTSURD interrupt occurs. After the CTSURC counter is read, these bits are cleared immediately before the CTSU measurement status counter value changes to Status 4 (the CTSUST.CTSUSTC[2:0] flags changes to 100b) in the next measurement. They can also be cleared using the CTSUCR0.CTSUINIT bit.

## 40.2.17 CTSURC : CTSU Reference Counter

Base address: CTSU = 0x400D_0000

Offset address: 0x1A

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSURC[15:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | CTSURC[15:0] | CTSU Reference Counter<br>These bits indicate the measurement result of the reference ICO. They read 0xFFFF when an overflow occurs. | R |

After a CTSU_CTSURD interrupt occurs, read first from the CTSUSC counter and then from the CTSURC counter.

Status 3 continues until the CTSURC counter is read, even if the stabilization time specified for Status 3 elapses.

### CTSURC[15:0] bits (CTSU Reference Counter)

The CTSURC[15:0] bits are configured as an increment counter for the reference ICO clock. The reference ICO optimizes touch measurement performed by the sensor ICO. There is some deviation depending on the internal sensor ICO and the reference ICO in the CTSU, but both ICOs have almost the same characteristics, including the dynamic range and the current-to-frequency characteristics. The range of current amount that can be set in the reference ICO current adjustment bits is about the same as the dynamic range of both ICOs, and the current amount input to the sensor ICO must be within this dynamic range. To ensure this, use the reference ICO to check the differences between the ICOs and measure the current-to-oscillation frequency characteristics. Because the reference ICO oscillation frequency can be obtained from the reference ICO counter, the ICO oscillation frequency (counter value/measurement time) for the input current amount can be measured by setting the value in the reference ICO current adjustment bits and measuring the reference ICO counter. The reference ICO counter value measured using the maximum value in the reference ICO current adjustment bits is the maximum value of the ICO dynamic range. The current to the sensor ICO must be offset in the offset adjustment bits so that the sensor ICO counter value does not exceed this value.

Read the CTSURC[15:0] bits after a CTSU_CTSURD interrupt occurs. After these bits are read, they are cleared immediately before the CTSU measurement status counter value changes to Status 4 (the CTSUST.CTSUSTC[2:0] flags changes to 100b) in the next measurement. They can also be cleared using the CTSUCR0.CTSUINIT bit.

## 40.2.18 CTSUERRS : CTSU Error Status Register

Base address: CTSU = 0x400D_0000

Offset address: 0x1C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | CTSUICOMP | — | — | — | — | — | — | — | CTSUTSOC | CTSUCLKSEL1 | — | — | CTSUDRV | CTSUTSOD | CTSUSPMD[1:0] | |
| Value after reset: | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | CTSUSPMD[1:0] | Calibration Mode<br>0 0: Capacitance measurement mode<br>1 0: Calibration mode<br>Others: Seting prohibited | R/W |
| 2 | CTSUTSOD | TS Pin Fixed Output<br>0: Capacitance measurement mode<br>1: TS pins are forced to be high or low | R/W |
| 3 | CTSUDRV | Calibration Setting 1<br>0: Capacitance measurement mode<br>1: Calibration setting 1 | R/W |
| 5:4 | — | These bits are read as 0. The write value should be 0. | R/W |
| 6 | CTSUCLKSEL1 | Calibration Setting 3<br>0: Capacitance measurement mode<br>1: Calibration setting 3 | R/W |
| 7 | CTSUTSOC | Calibration Setting 2<br>0: Capacitance measurement mode<br>1: Calibration setting 2 | R/W |
| 14:8 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15 | CTSUICOMP | TSCAP Voltage Error Monitor<br>0: Normal TSCAP voltage<br>1: Abnormal TSCAP voltage*1 | R |

Note 1. When the CTSUCR1.CTSUPON bit is 0, this bit is set to 1.

### CTSUSPMD[1:0] bits (Calibration Mode)

The CTSUSPMD[1:0] bits are used to calibrate the CTSU. When measuring the capacitance, set these bits to 00b.

### CTSUTSOD bit (TS Pin Fixed Output)

The CTSUTSOD bit is used to calibrate the CTSU. When setting this bit to 1, the TS pins are forced to the logic level specified by the CTSUCR0.CTSUIOC bit. When measuring the capacitance, set this bit to 0.

### CTSUDRV bit (Calibration Setting 1)

The CTSUDRV bit is used to calibrate the CTSU. When measuring the capacitance, set this bit to 0.

### CTSUCLKSEL1 bit (Calibration Setting 3)

The CTSUCLKSEL1 bit is used to calibrate the CTSU. When measuring the capacitance, set this bit to 0.

### CTSUTSOC bit (Calibration Setting 2)

The CTSUTSOC bit is used to calibrate the CTSU. When measuring capacitance, set this bit to 0.

**CTSUICOMP bit (TSCAP Voltage Error Monitor)**

The CTSUICOMP bit monitors the TSCAP voltage and this bit is set to 1 if the voltage becomes abnormal.

If the offset current specified in the CTSUSO0 register exceeds the sensor ICO input current during touch measurement, the TSCAP voltage becomes abnormal and touch measurement cannot be performed correctly.

If the TSCAP voltage becomes abnormal, the sensor ICO counter value becomes undefined, but touch measurement completes normally, so it is difficult to detect an abnormality by reading the sensor ICO counter value. If the CTSU reference ICO current adjustment bits (CTSURICOA[7:0]) in the CTSUSO1 register are set to any value other than 0, always check this bit when touch measurement completes.

This bit is cleared by writing 0 to the CTSUCR1.CTSUPON bit and turning off the power supply.

## 40.2.19 CTSUTRMR : CTSU Reference Current Calibration Register

Base address: CTSU = 0x400D_0000

Offset address: 0x20

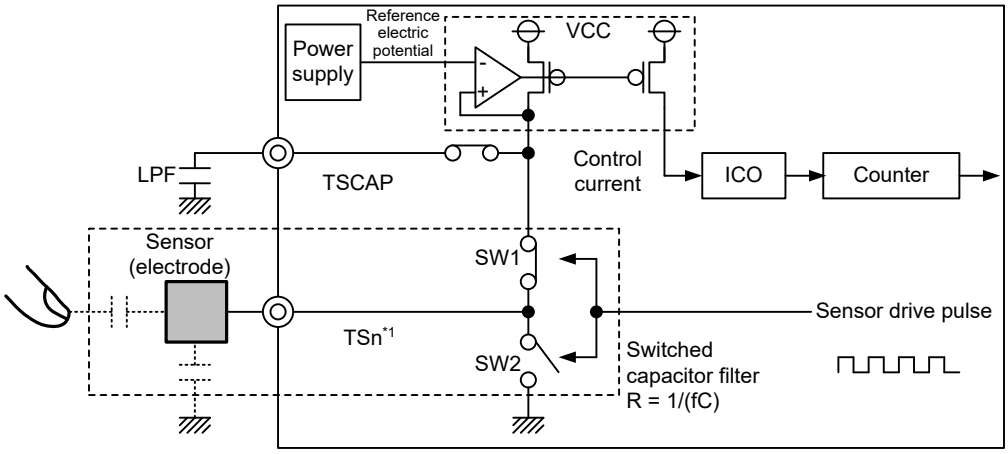
| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | |
| Value after reset: | Unique value for each chip | | | | | | | |

The CTSUTRMR resister stores a reference current value calibrated under the specified condition for each chip at factory shipment. When rewriting this register, set the CTSUERRS.CTSUSPMD[1:0] bits to 10b (calibration mode). When resetting the MCU, the values returns to the factory setting value. Do not rewrite this register when the CTSUSPMD[1:0] bits are 00b (capacitance measurement mode).

# 40.3 Operation

## 40.3.1 Principles of Measurement Operation

Figure 40.4 shows the measurement circuit.



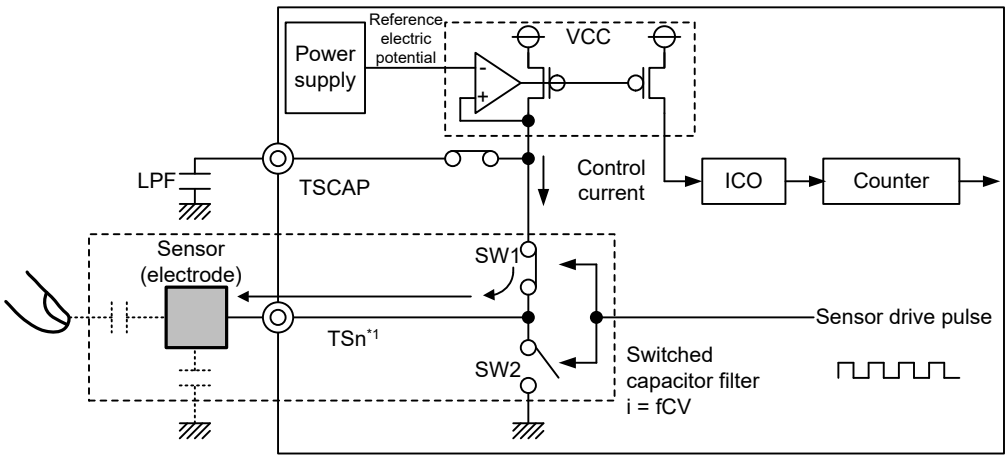

Note 1. TSn = TS01 to TS12

**Figure 40.4 Measurement circuit**

Figure 40.5 to Figure 40.7 explain the electrostatic capacitance measurement operation principles of the CTSU current frequency conversion. The operation is as follows:

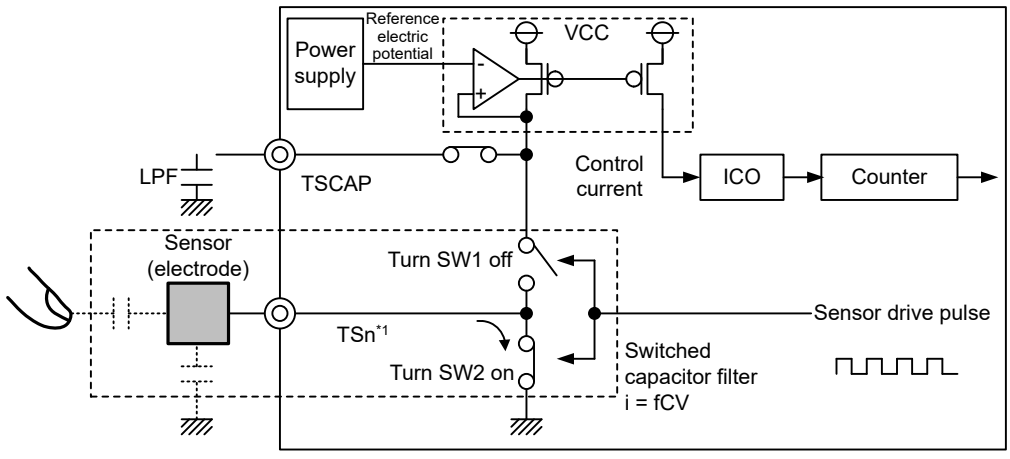
1. The electrostatic capacitance of the electrode is charged by turning SW1 on and SW2 off (Figure 40.5).

2. The charged capacitance is discharged by turning SW1 off and SW2 on (Figure 40.6).
3. Current flows to the switched capacitor filter by switching between charging and discharging. At this point, if a finger is in close proximity, the capacitance and the flowing current change. A clock is generated by supplying the control current, which is proportional to the amount of current flowing through the switched capacitor filter, from the circuit that generates the TSCAP power supply to the ICO. The counter measures the clock frequency that changes depending on whether a finger is in close proximity. The software uses the value read from the counter to determine contact with a finger (Figure 40.7).




Note 1. TSn = TS01 to TS12

**Figure 40.5 Charging operation**




Note 1. TSn = TS01 to TS12

**Figure 40.6 Discharging operation**

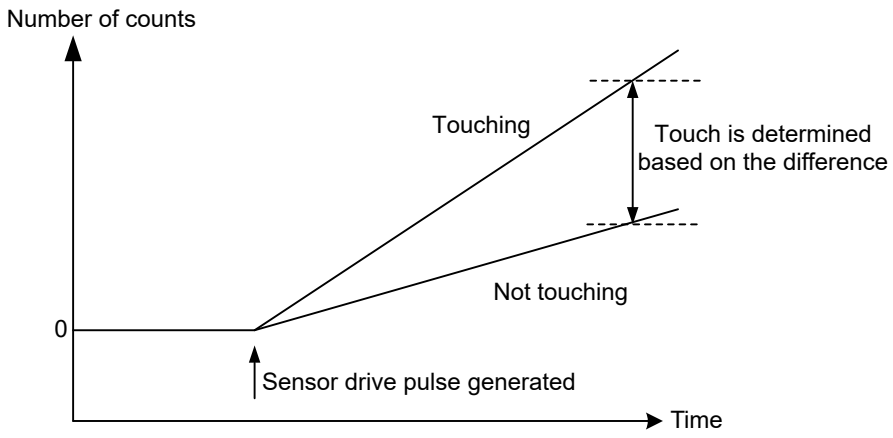
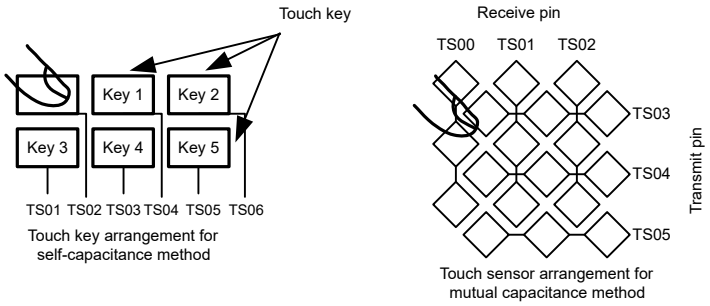



**Figure 40.7 Change in measured value when finger is touching and not touching**

## 40.3.2 Measurement Modes

The CTSU supports self-capacitance and mutual capacitance methods. Figure 40.8 illustrates these methods.




Note: This figure indicates a usage example. About available pins, see Table 40.2

**Figure 40.8 Overview of self-capacitance method and mutual capacitance method**

In the self-capacitance method, a single touch pin is allocated to a single touch key to measure individual electrostatic capacitance when a finger is in close proximity. In this method, capacitance can be measured in both single scan and multiscan modes.

In the mutual capacitance method, the capacitance between two opposing electrodes (transmit and receive pins) is measured.

### 40.3.2.1 Initial settings flow

Figure 40.9 shows the flow for the CTSU initial settings.

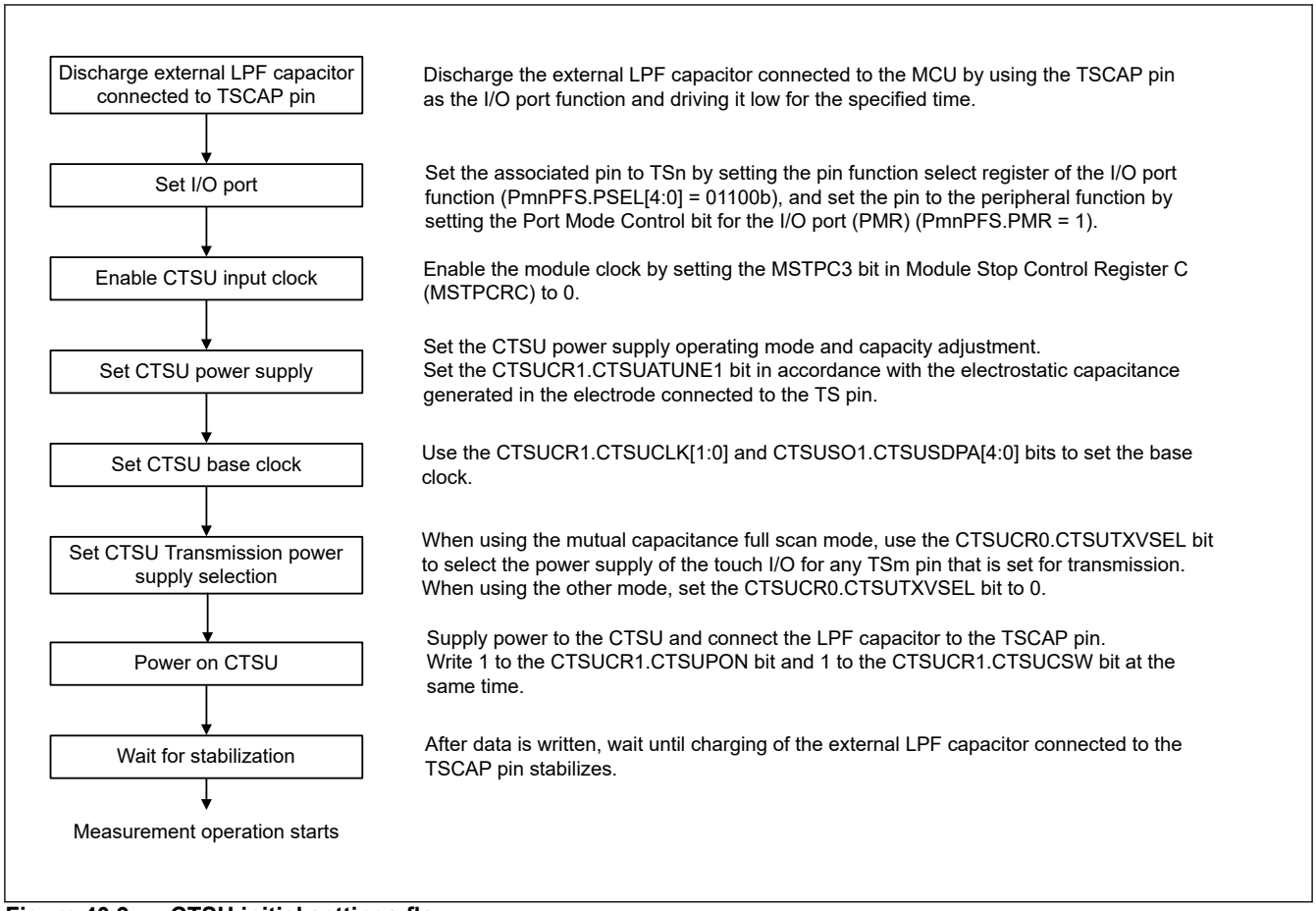



**Figure 40.9 CTSU initial settings flow**

Figure 40.10 shows the flow for stopping CTSU operation and invoking the standby state.

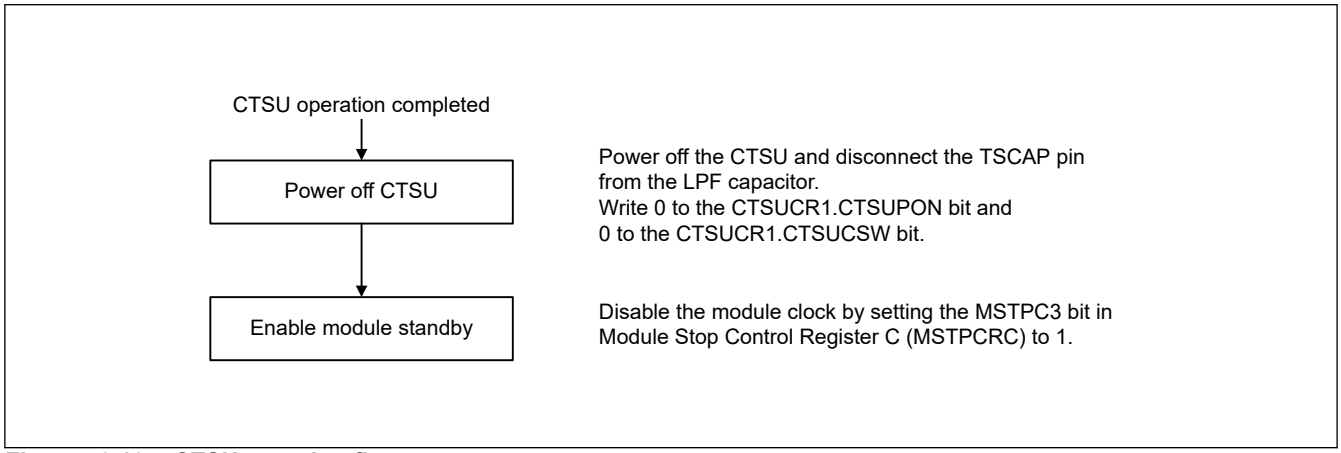
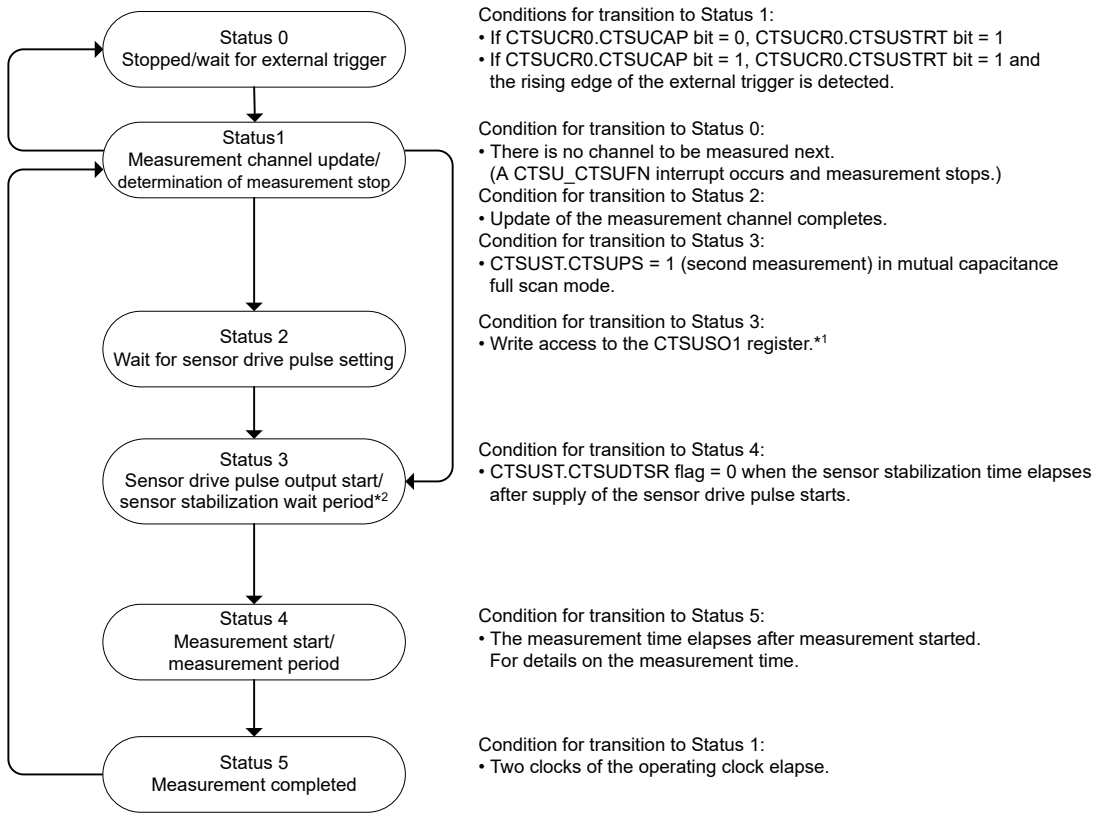



**Figure 40.10 CTSU stopping flow**

When restarting operation after it stops, follow the initial settings flow shown in Figure 40.9.

### 40.3.2.2 Status counter

The measurement status counter of the CTSU Status Register (CTSUST) indicates the current measurement status. The measurement status is shared by all three modes. Figure 40.11 shows the status operation transitions.




Note 1. When using the DTC or ICU to set the registers during CTSU_CTSUWR interrupt handling, write to the CTSUSO1 register last.
Note 2. If the CTSUST.CTSUDTSR flag = 1, wait until the previous measurement result is transferred.

**Figure 40.11 Status operation transitions**

The status counter transitions to Status 0 when all of the specified measurement channels are measured.

The CTSUCR0.CTSUSTRT bit is set to 0 by hardware when a software trigger is used. When an external trigger is used, the value of 1 is retained, and the CTSU waits for the next trigger.

When operation is forced to stop during measurement or the trigger wait state, by a simultaneous 0 write to the CTSUCR0.CTSUSTRT bit and a 1 write to the CTSUCR0.CTSUINIT bit, the status transitions to Status 0 and measurement stops.

If the channel to be measured is not set in the CTSUCHAC0 to CTSUCHAC1 or CTSUCHTRC0 to CTSUCHTRC1 registers, a CTSU_CTSUFN interrupt occurs immediately after a transition to Status 1, and then the status transitions to Status 0.

In the following situations, there is no channel to be measured:

- No measurement target channel is specified in the CTSUCHAC0 to CTSUCHAC1 registers.
- In self-capacitance single scan mode, the channel specified in the CTSUMCH0 register is not a measurement target in the CTSUCHAC0 to CTSUCHAC1 registers.
- In full scan modes, there is no transmit channel or receive channel to be measured based on the combined settings of the CTSUCHAC0 to CTSUCHAC1 and CTSUCHTRC0 to CTSUCHTRC1 registers.

If there is no channel to be measured based on these settings, a CTSU_CTSUFN interrupt occurs immediately after a transition to Status 1, and the counter transitions to Status 0.

### 40.3.2.3 Self-capacitance single scan mode operation

In self-capacitance single scan mode, electrostatic capacitance on one channel is measured. Figure 40.12 shows the software flow and an operation example, and Figure 40.13 shows the timing.

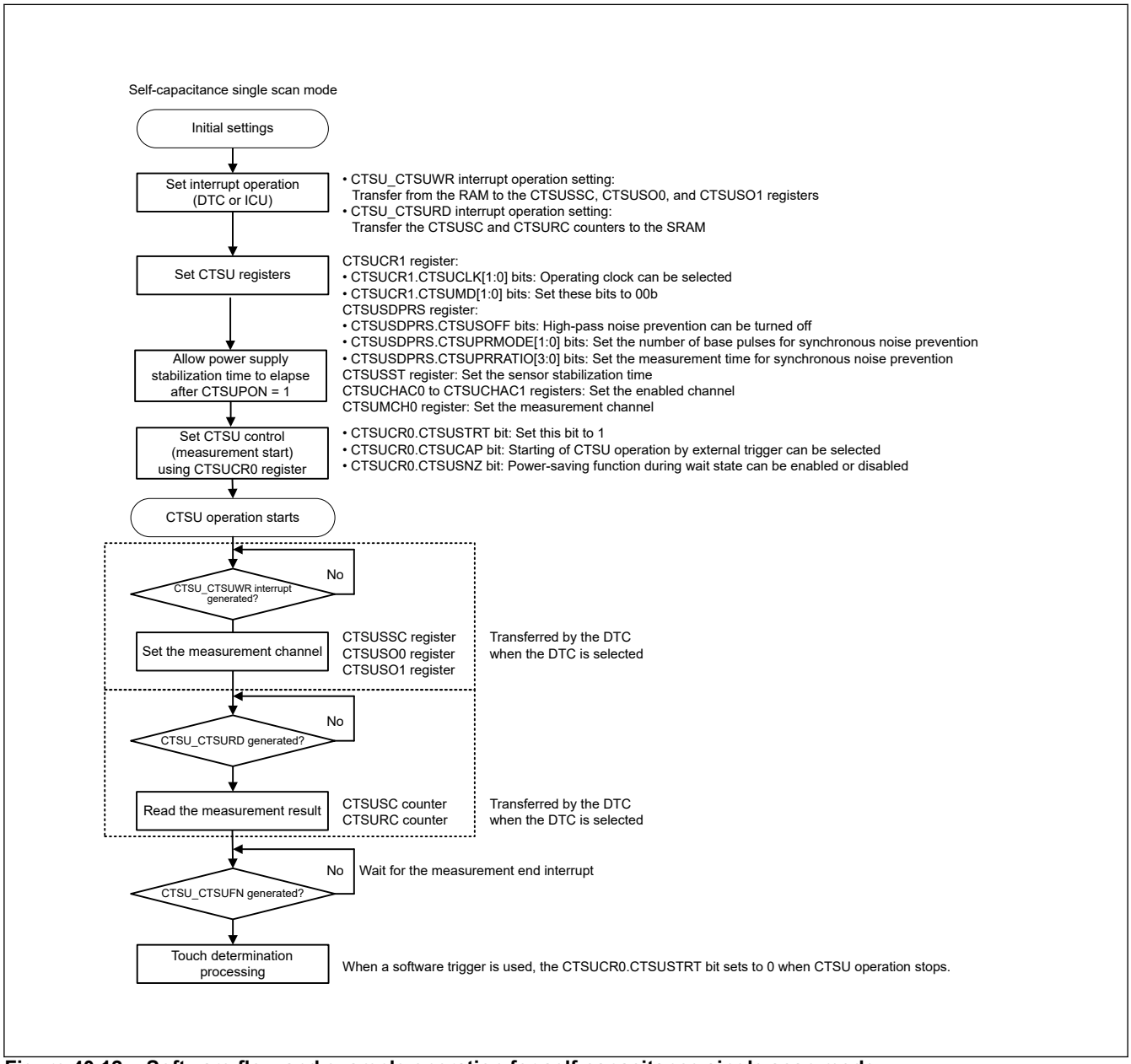



**Figure 40.12 Software flow and example operation for self-capacitance single scan mode**

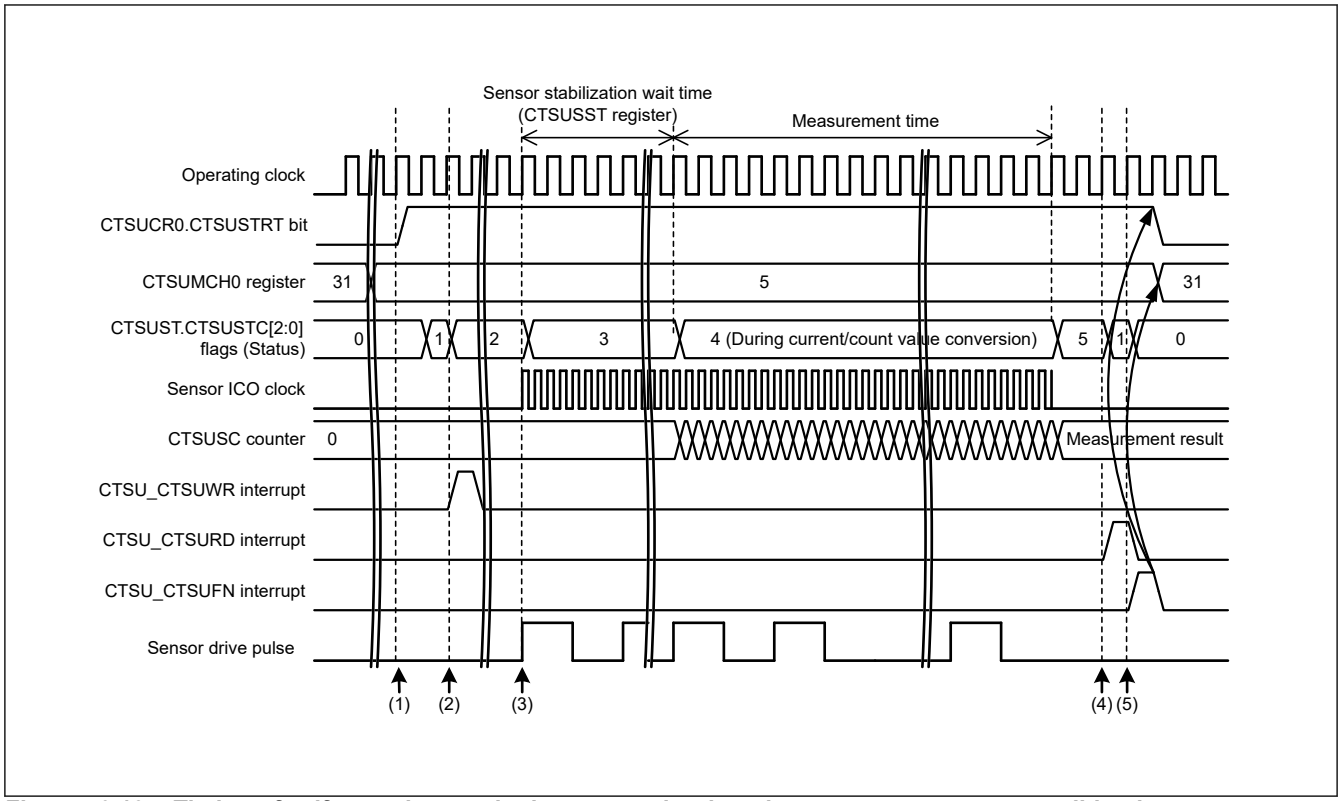



**Figure 40.13 Timing of self-capacitance single scan mode when the measurement start condition is a software trigger**

The following describes the operation shown in Figure 40.13.

1. After initial settings are made, operation is started by writing 1 to the CTSUCR0.CTSUSTRT bit.
2. After the channel to be measured is determined in accordance with the preset conditions, a request for setting the channel (CTSU_CTSUWR) is output.
3. On completion of writing the measurement channel settings (CTSUSSC, CTSUSO0, and CTSUSO1 registers), the sensor drive pulse is output and the sensor ICO clock and reference ICO clock operate.
4. After the sensor stabilization wait time and the measurement time elapse, and measurement stops, a measurement result read request (CTSU_CTSURD) is output.
5. A measurement end interrupt (CTSU_CTSUFN) is output and measurement stops (transition to Status 0).

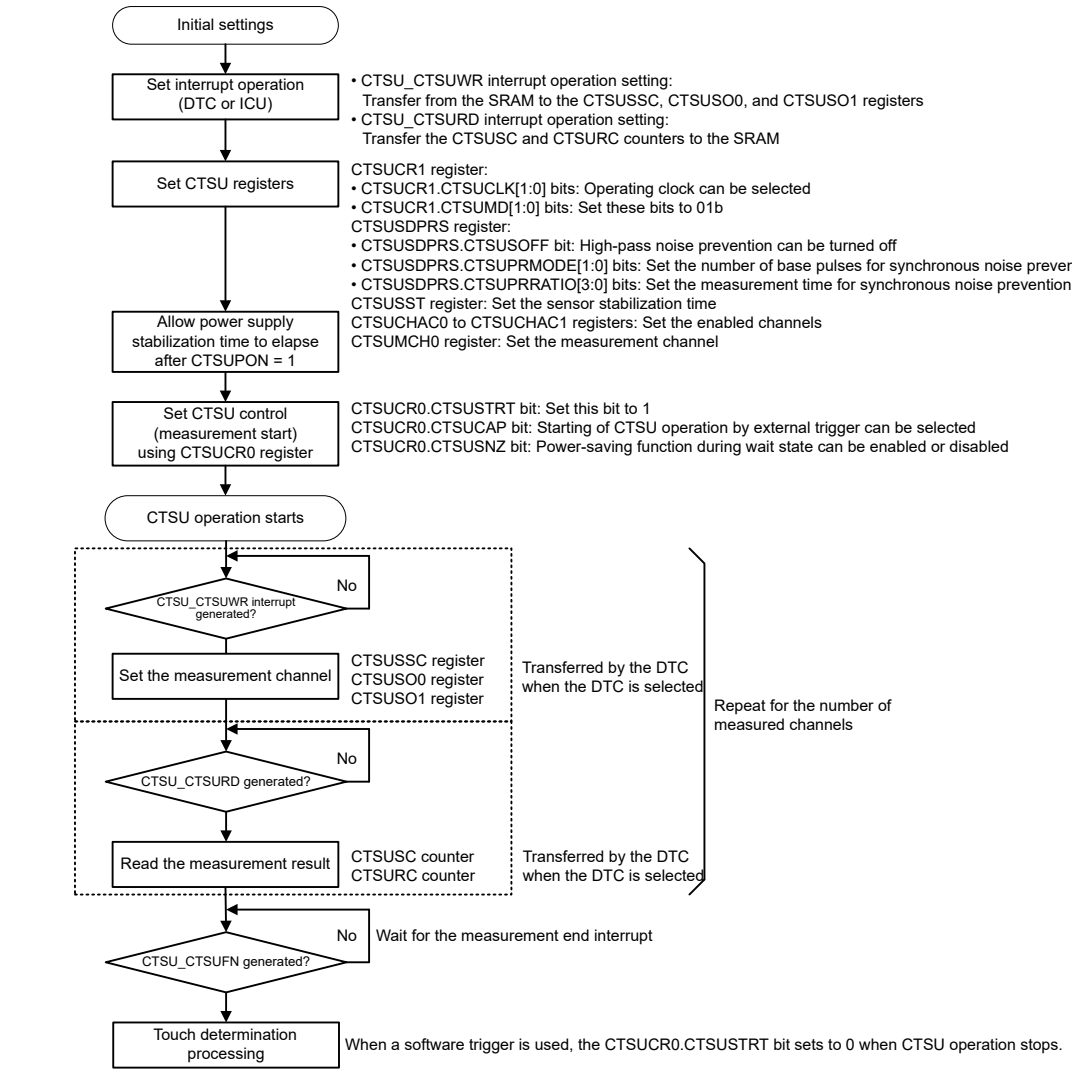
Table 40.7 lists the touch pin states in self-capacitance single scan mode.

**Table 40.7 Touch pin states in self-capacitance single scan mode**

| Status | Touch pin | |
|---|---|---|
| | Measured channel | Non-measured channel |
| 0 | Low | Low |
| 1 | Low | Low |
| 2 | Low | Low |
| 3 | Pulse | Low |
| 4 | Pulse | Low |
| 5 | Low | Low |

### 40.3.2.4 Self-capacitance multi-scan mode operation

In self-capacitance multi-scan mode, electrostatic capacitance on all channels that are specified as measurement targets in the CTSUCHAC0 to CTSUCHAC1 registers is measured sequentially in ascending order. Figure 40.14 shows the software flow and an operation example, and Figure 40.15 shows the timing.




Channel measurement sequence in self-capacitance multiscan mode

Settings:
• Select self-capacitance multiscan mode (CTSUCR1.CTSUMD[1:0] bits = 01b)
• Set channels 0, 2, 5, and 6 to enabled channels (CTSUCHAC0.CTSUCHAC0[7:0] bits = 01100101b)

| Receive Channels | | | | | | | |
|---|---|---|---|---|---|---|---|
| Channel 7 | Channel 6 | Channel 5 | Channel 4 | — | Channel 2 | — | Channel 0 |
| | (4) | (3) | | | (2) | | (1) |

**Figure 40.14 Software flow and example operation for self-capacitance multi-scan mode**

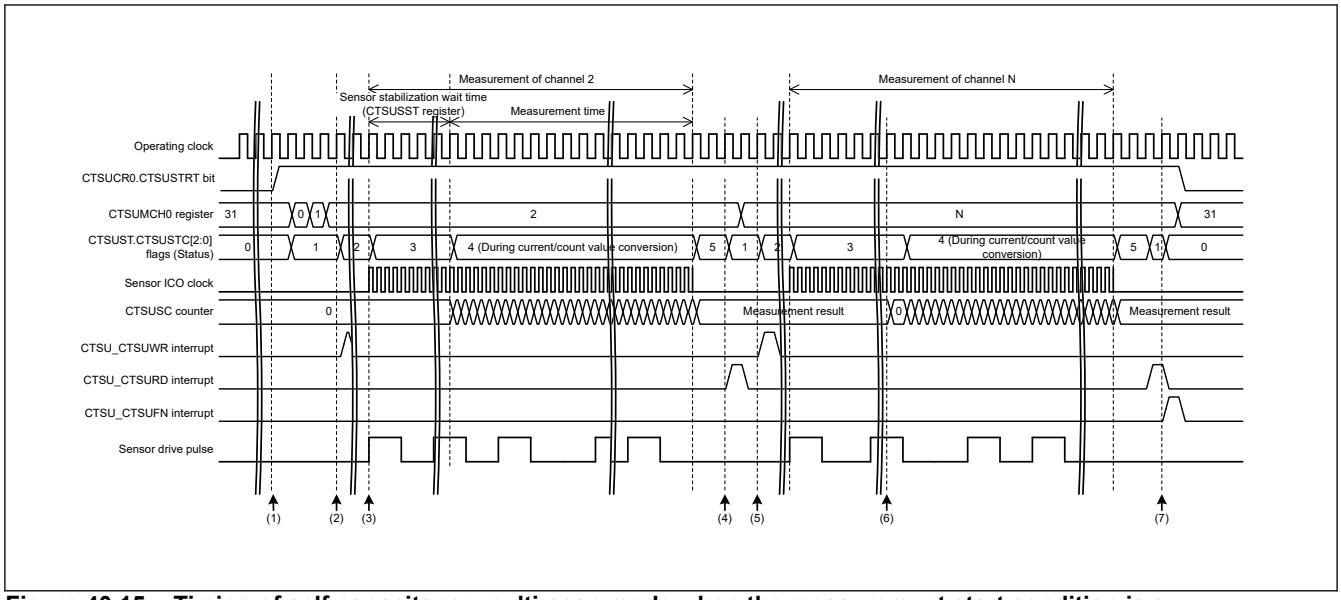



**Figure 40.15 Timing of self-capacitance multi-scan mode when the measurement start condition is a software trigger**

The following describes the operation shown in Figure 40.15:

1. After initial settings are made, operation is started by writing 1 to the CTSUCR0.CTSUSTRT bit.
2. After the channel to be measured is determined in accordance with the preset conditions, a request for setting the channel (CTSU_CTSUWR) is output.
3. On completion of writing the measurement channel settings (CTSUSSC, CTSUSO0, and CTSUSO1 registers), the sensor drive pulse is output and the sensor ICO clock and the reference ICO clock operate.
4. After the sensor stabilization wait time and the measurement time elapse, and measurement stops, a measurement result read request (CTSU_CTSURD) is output.
5. After the channel to be measured next is determined, a measurement channel setting request (CTSU_CTSUWR) is output.
6. After the stabilization wait time elapses and when the previous measurement is read, the result is cleared and measurement starts.
7. On completion of all measurement channels, a measurement end interrupt (CTSU_CTSUFN) is output and measurement is stopped (transition to Status 0).

Table 40.8 lists the touch pin states in self-capacitance multi-scan mode.

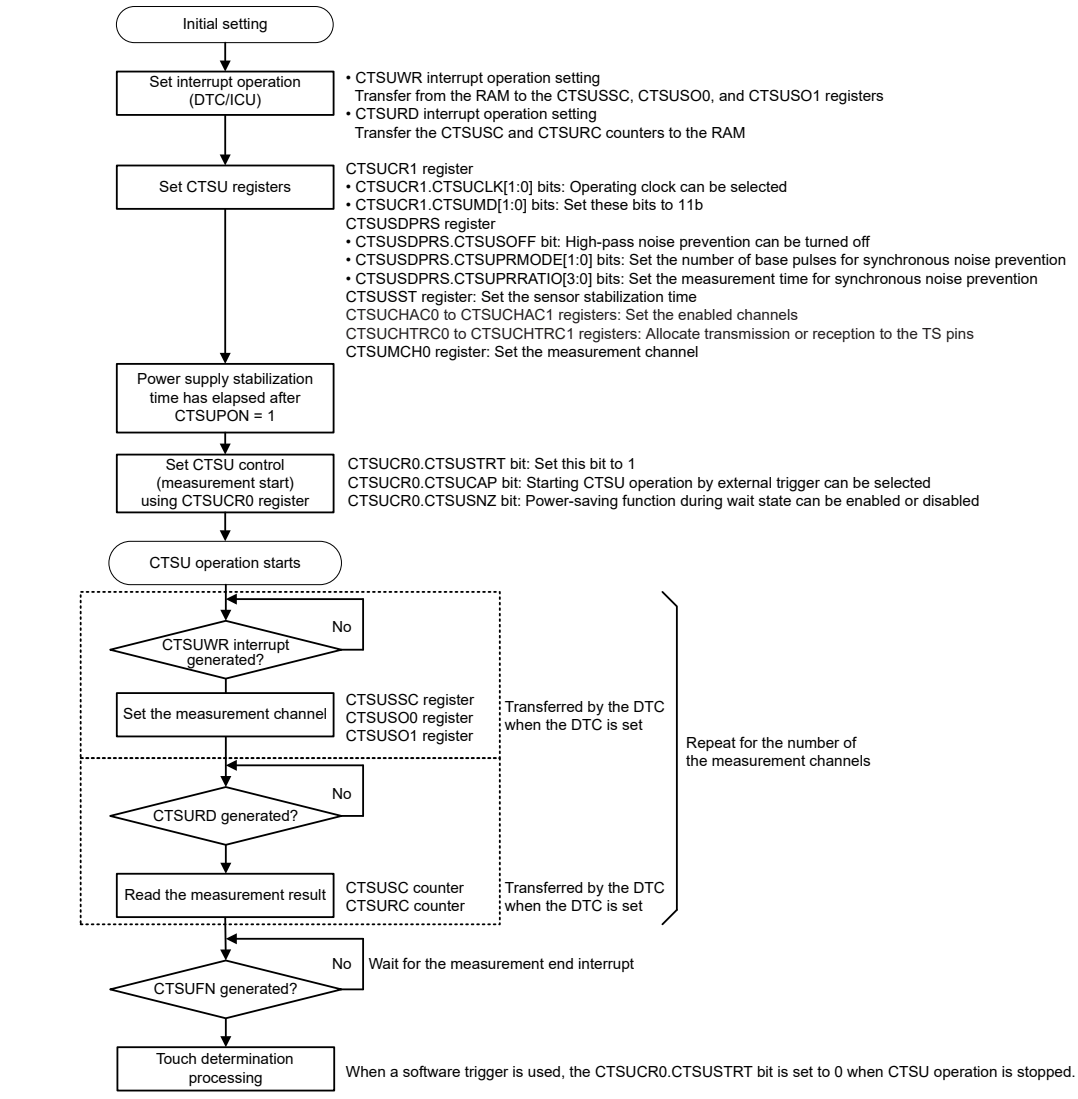
**Table 40.8 Touch pin states in self-capacitance multi-scan mode**

| Status | Touch pin | |
|---|---|---|
| | Measured channel | Non-measured channel |
| 0 | Low | Low |
| 1 | Low | Low |
| 2 | Low | Low |
| 3 | Pulse | Low |
| 4 | Pulse | Low |
| 5 | Low | Low |

### 40.3.2.5 Mutual capacitance full scan mode operation

In mutual capacitance full scan mode, measurement is performed during the high-level period of the sensor drive pulse on the receive channel by applying the edge to the target transmit channel to be measured. A single measurement target is measured twice, on the rising and falling edges. The difference between the data of these two measurements determines whether or not the electrode was touched, which results in a higher touch sensitivity.

Electrostatic capacitance is measured sequentially on channels set to transmission or reception in the CTSUCHTRC0 to CTSUCHTRC1 registers, and specified as measurement targets in the CTSUCHAC0 to CTSUCHAC1 registers. The capacitance is measured by combining these signals. Figure 40.16 shows the software flow and an operation example, and Figure 40.17 shows the timing.




Channel measurement sequence in mutual capacitance full scan mode

Settings:
- Select mutual capacitance full scan mode (CTSUCR1.CTSUMD[1:0] = 11b)
- Set channels 0, 2, 5, and 6 to enabled channels (CTSUCHAC0.CTSUCHAC0[7:0] bits = 01100101b)
- Set channels 0, 2 to receive channels and channels 4 to 7 to transmit channels (CTSUCHTRC0.CTSUCHTRC0[7:0] bits = 11110000b)

| | | Receive Channels | | | |
|---|---|---|---|---|---|
| | | — | Channel 2 | — | Channel 0 |
| Transmit channels | Channel 4 | | | | |
| | Channel 5 | | (3) | | (1) |
| | Channel 6 | | (4) | | (2) |
| | Channel 7 | | | | |

**Figure 40.16 Software flow and example operation for mutual capacitance full scan mode**

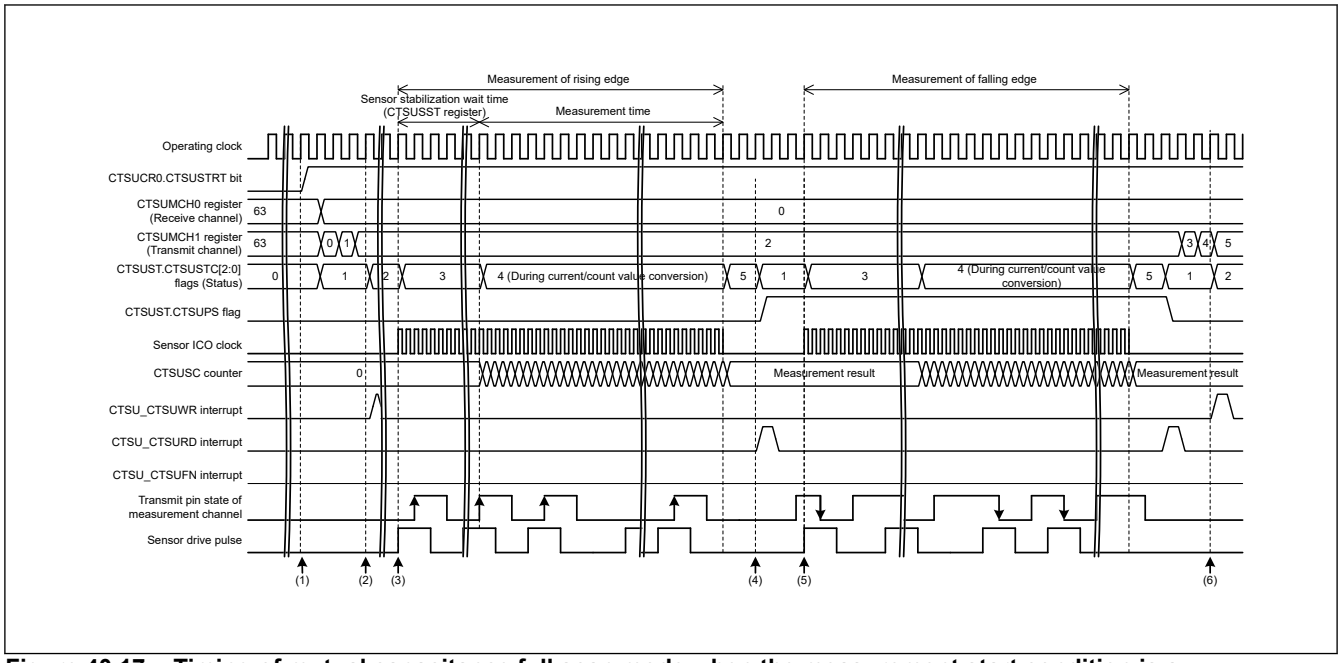



**Figure 40.17 Timing of mutual capacitance full scan mode when the measurement start condition is a software trigger**

The following describes the operation shown in Figure 40.17

1. After initial settings are made, operation is started by writing 1 to the CTSUCR0.CTSUSTRT bit.
2. After the channel to be measured is determined in accordance with the preset conditions, a request for setting the channel (CTSU_CTSUWR) is output.
3. On completion of writing the measurement channel settings (CTSUSSC, CTSUSO0, and CTSUSO1 registers), the sensor drive pulse is output and the sensor ICO clock and the reference ICO clock operate. At the same time, a pulse detected on the rising edge is output to the transmit pin on the measurement channel during the high-level period of the sensor drive pulse.
4. After the sensor stabilization wait time and the measurement time elapsed and measurement stops, a measurement result read request (CTSU_CTSURD) is output.
5. The same channel is measured by outputting a pulse detected on the falling edge during the high-level period of the sensor drive pulse.
6. After the same channel is measured twice, the channel to be measured next is determined and measured in the same way.
7. On completion of all measurement channels, a measurement end interrupt (CTSU_CTSUFN) is output and measurement stops (transition to Status 0).

The mutual capacitance measurement status flag (CTSUST.CTSUPS bit) changes when Status 5 transitions to Status 1. Table 40.9 lists the touch pin states in mutual capacitance full scan mode.

**Table 40.9 Touch pin states in mutual capacitance full scan mode (1 of 2)**

| Status | Touch pins for receive channels | | Touch pins for transmit channels | | Remarks |
|---|---|---|---|---|---|
| | Measured channel | Non-measured channel | Measured channel | Non-measured channel | |
| 0 | Low | Low | Low | Low | — |
| 1 | Low | Low | Low/high | Low | — |
| 2 | Low | Low | Low | Low | — |
| 3 | Pulse | Low | Pulse | Low | The phase pulse is the same as that of the receive channel on the first measurement and opposite on the second measurement. |
| 4 | Pulse | Low | Pulse | Low | — |

**Table 40.9 Touch pin states in mutual capacitance full scan mode (2 of 2)**

| Status | Touch pins for receive channels | | Touch pins for transmit channels | | Remarks |
|---|---|---|---|---|---|
| | Measured channel | Non-measured channel | Measured channel | Non-measured channel | |
| 5 | Low | Low | Low | Low | — |

## 40.3.3 Parameters Common to Multiple Modes

### 40.3.3.1 Sensor stabilization wait time and measurement time

Figure 40.18 shows the timing of the sensor stabilization wait and measurement.

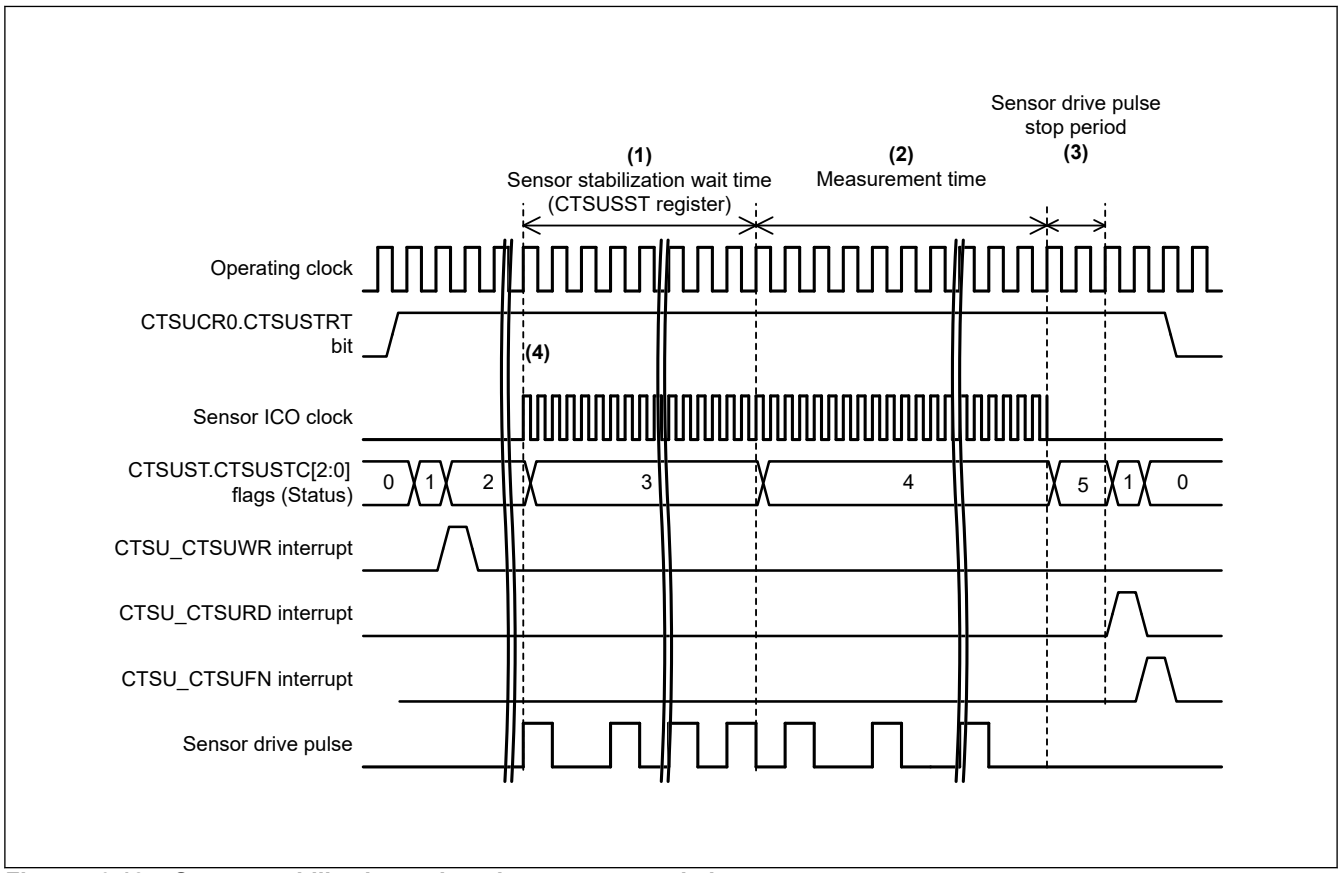



**Figure 40.18 Sensor stabilization wait and measurement timing**

The following describes the operation shown in Figure 40.18:

1. In response to the CTSU_CTSUWR interrupt request, output of the sensor drive pulse is started by a write access to the CTSUSO1 register. The CTSU waits for the stabilization time set in the CTSUSST register.
2. When the sensor stabilization time elapses and the CTSUST.CTSUDTSR flag clears to 0, measurement starts on transition to Status 4. The measurement time is determined by the base clock cycle setting and the CTSUSDPRS.CTSUPRMODE[1:0], CTSUPRRATIO[3:0], and CTSUSO0.CTSUSNUM[5:0] bits. When the measurement time elapses, measurement of the channel stops.
3. After the measurement time elapses, the status transitions to Status 1 after two operating clock cycles, and a CTSU_CTSURD interrupt occurs. Read the data from the CTSUSC and CTSURC counters. At this time, the sensor drive pulse is output at the low level. When measurement of all specified channels is completed, the CTSUCR0.CTSUSTRT bit clears to 0.
4. The sensor ICO clock oscillates while the CTSUST.CTSUSTC[2:0] flags = 011b (Status 3) or 100b (Status 4).

### 40.3.3.2 Interrupts

The CTSU supports the following interrupts:

- Write request interrupt for setting registers for each channel (CTSU_CTSUWR)
- Measurement data transfer request interrupt (CTSU_CTSURD)
- Measurement end interrupt (CTSU_CTSUFN)

#### (1) Write request interrupt for setting registers for each channel (CTSU_CTSUWR)

Store the settings for each measurement channel in the SRAM, and set up the DTC or ICU transfer associated with the CTSU_CTSUWR interrupt in advance. The CTSU_CTSUWR interrupt is output when Status 1 transitions to Status 2. Write the settings for the selected channel from the SRAM to the CTSUSSC, CTSUSO0, and CTSUSO1 registers (Figure 40.19). Because write access to the CTSUSO1 register controls the transition to the next status, you must set this register last.

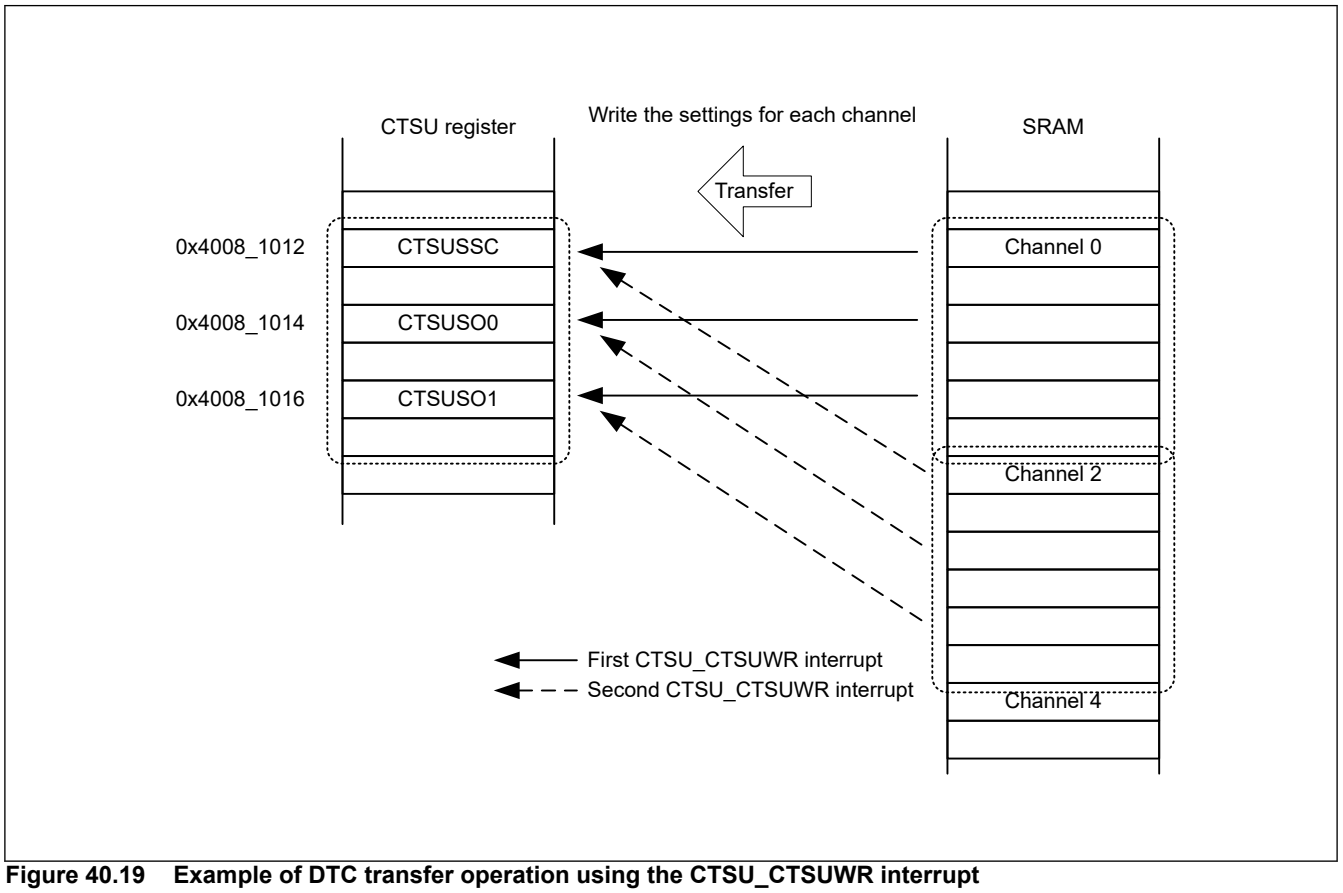



**Figure 40.19 Example of DTC transfer operation using the CTSU_CTSUWR interrupt**

The registers to be set (CTSUSSC, CTSUSO0, and CTSUSO1) are allocated at sequential addresses. On CTSU_CTSUWR interrupt generation, set up the operation as follows:

- Transfer destination address: CTSUSSC register address
- Handling at the transfer destination address: Transfer 2-byte data three times with a single interrupt. The address of the start byte is fixed.
- Transfer source address: CTSUSSC register data storage address for the lowest number channel in the settings stored in the SRAM
- Handling at the transfer source address: Transfer 2-byte data three times with a single interrupt. The address of the first byte is continued from the previous interrupt handling.
- Number of transfers per interrupt: Specify the number of measurements.

### (2) Measurement data transfer request interrupt (CTSU_CTSURD)

Set up the DTC or ICU transfer associated with the CTSU_CTSURD interrupt in advance. The CTSU_CTSURD interrupt is output when Status 5 transitions to Status 1. Read the measurement result from the CTSUSC and CTSURC counters (Figure 40.20).

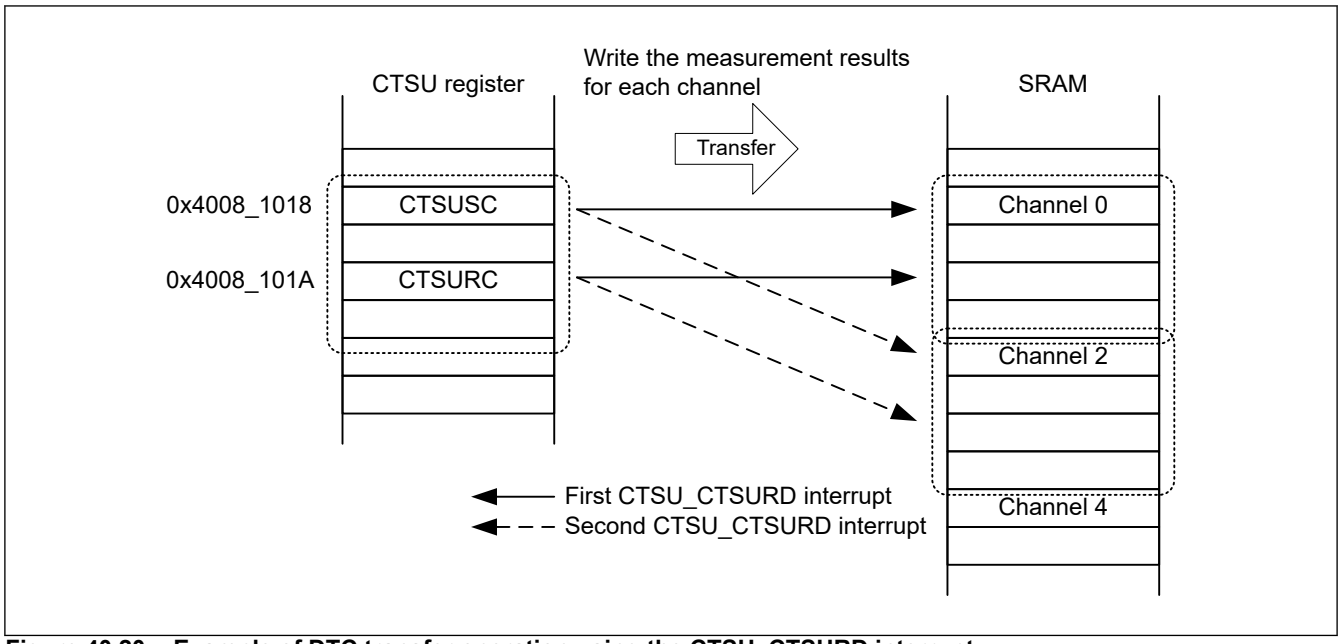



**Figure 40.20 Example of DTC transfer operation using the CTSU_CTSURD interrupt**

The measurement result registers (CTSUSC and CTSURC counters) used as transfer sources are allocated at sequential addresses. On CTSU_CTSURD interrupt generation, set up the operation as follows:

- Transfer source address: CTSUSC counter address
- Handling at the transfer source address: Transfer 2-byte data twice with a single interrupt. The start address is fixed.
- Transfer destination address: CTSUSC counter data storage address for the lowest number channel in the settings stored in the SRAM
- Handling at the transfer destination address: Transfer 2-byte data twice with a single interrupt. The start address is continued from the previous interrupt handling.
- Number of transfers per interrupt: Specify the number of measurements.

### (3) Measurement end interrupt (CTSU_CTSUFN)

When all channels are measured, an interrupt occurs when Status 1 transitions to Status 0. In the software, check the overflow flags (CTSUST.CTSUSOVF and CTSUROVF) and read the measurement results to determine whether or not the electrode was touched. Interrupt requests are accepted or disabled in the interrupt control block.

## 40.4 Usage Notes

### 40.4.1 Measurement Result Data (CTSUSC and CTSURC Counters)

Read access during measurement is prohibited. If the measurement result data is accessed, an incorrect value might be read because of asynchronous operation.

### 40.4.2 Constraint on Software Trigger

When 10b (PCLKB/4) is selected in the CTSUCR1.CTSUCLK[1:0] bits, to restart measurement by writing 1 to the CTSUR0.CTSUSTRT bit after measurement is complete, wait for at least 3 cycles to elapse after an interrupt occurs, and then write to the CTSUCR0.CTSUSTRT bit.

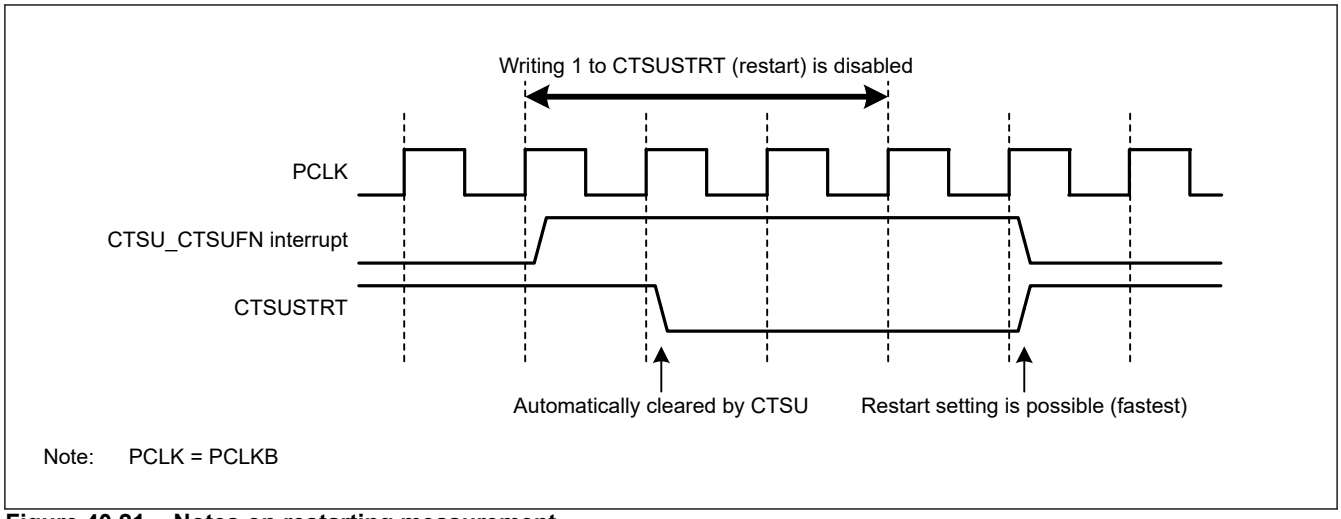



**Figure 40.21 Notes on restarting measurement**

### 40.4.3 Constraints on External Triggers

- If an external trigger is input during the measurement time, measurement does not start. The next external event is enabled after 1 cycle of the operating clock when a CTSU_CTSUFN interrupt occurs.
- To stop external trigger mode, write 0 to the CTSUCR0.CTSUSTRT bit and 0 to the CTSUCR0.CTSUINIT bit at the same time (forced stop).

### 40.4.4 Constraints on Forced Stops

To force the current operation to stop, write 0 to the CTSUCR0.CTSUSTRT bit and 1 to the CTSUCR0.CTSUINIT bit at the same time. After this setting, the operation is stopped and the internal control registers are initialized.

When the CTSUCR0.CTSUINIT bit is used for initialization, the following registers are initialized in addition to the initialization of the internal measurement state.

- CTSUMCH0 register
- CTSUMCH1 register
- CTSUST register
- CTSUSC counter
- CTSURC counter

If operation is forced to stop, an interrupt request might be generated depending on the internal state. After a forced stop, also perform the processing for stopping and disabling the DTC or ICU. If a DTC transfer is stopped in an installed system for some reason, also perform the processing for forcing stop to and initializing the CTSU.

### 40.4.5 TSCAP Pin

The TSCAP pin requires an external decoupling capacitor to stabilize CTSU internal voltage. The traces between the TSCAP pin and the capacitor, and the capacitor and ground should be as short and wide as physically possible.

The capacitor connected to the TSCAP pin should be fully discharged using I/O port control to output a low level, before turning on the switch (CTSUCR1.CTSUCSW bit = 1) to establish a connection.

### 40.4.6 Constraints on Measurement Operation (CTSUCR0.CTSUSTRT Bit = 1)

During measurement (CTSUCR0.CTSUSTRT bit = 1), do not use the settings for stopping the peripheral clock or changing the port settings related to the touch pins (TSn and TSCAP pins) and Transmission power supply selection (CTSUCR0.CTSUTXVSEL) in the higher layers of the system.

If control settings non-compliant with these constraints are made, operation is forced to stop (CTSUCR0.CTSUSTRT bit = 0 and CTSUCR0.CTSUINIT bit = 1), write 0 to the CTSUCR1.CTSUPON bit and 0 to the CTSUCR1.CTSUCSW bit at the same time, and set the CTSUCR0.CTSUSNZ bit to 0. Next, restart from the initial settings flow shown in Figure 40.9.

# 41. Data Operation Circuit (DOC)

## 41.1 Overview

The Data Operation Circuit (DOC) compares, adds, and subtracts 16-bit data. When a selected condition applies, 16-bit data is compared and an interrupt can be generated. Table 41.1 lists the DOC specifications and Figure 41.1 shows a block diagram.

**Table 41.1 DOC specifications**

| Item | Description |
|---|---|
| Data operation function | 16-bit data comparison, addition, and subtraction |
| Module-stop function | The module-stop state can be set to reduce power consumption. |
| Interrupts and event link function (DOC_DOPCI) | An interrupt occurs on the following conditions:<br>• The compared values either match or mismatch<br>• The result of data addition is greater than 0xFFFF<br>• The result of data subtraction is less than 0x0000 |
| TrustZone Filter | Security attribution can be set |

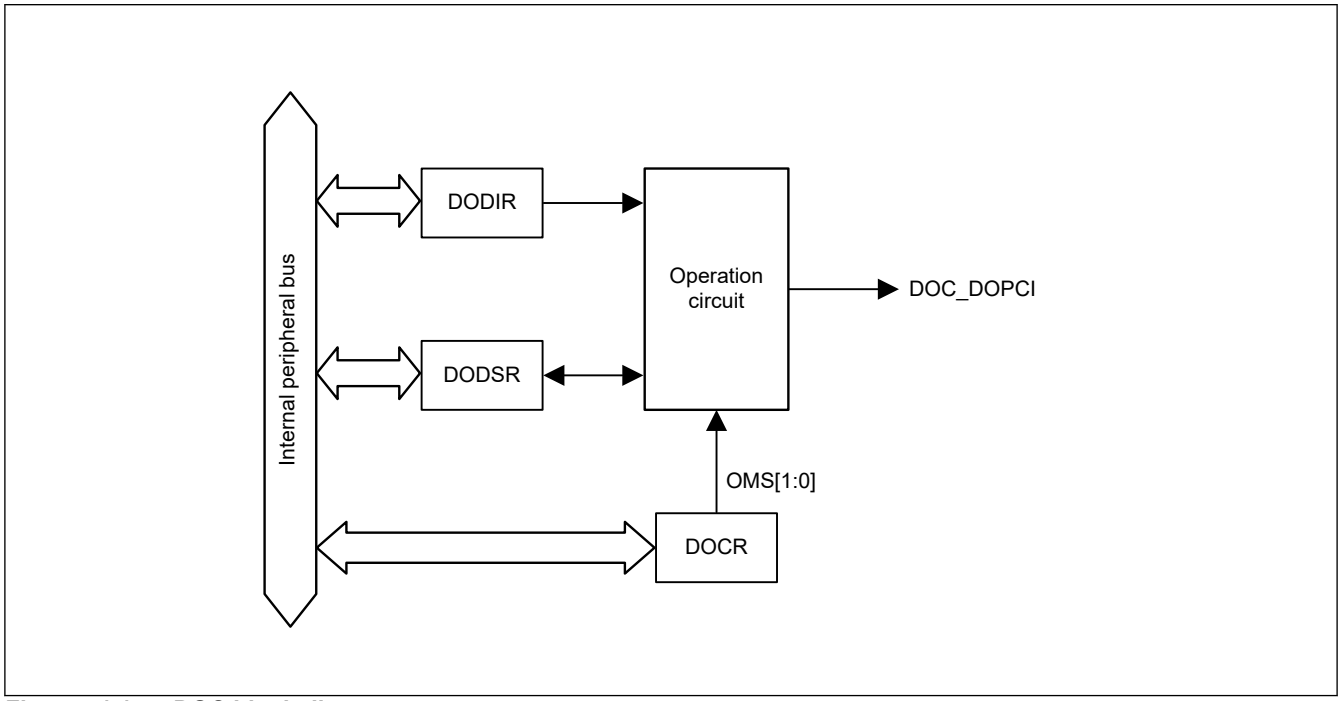



**Figure 41.1 DOC block diagram**

## 41.2 DOC Register Descriptions

### 41.2.1 DOCR : DOC Control Register

Base address: DOC = 0x4010_9000

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | DOPCFCL | DOPCF | — | — | DCSEL | OMS[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | OMS[1:0] | Operating Mode Select<br>0 0: Data comparison mode<br>0 1: Data addition mode<br>1 0: Data subtraction mode<br>1 1: Setting prohibited | R/W |
| 2 | DCSEL*1 | Detection Condition Select<br>0: Set DOPCF flag when data mismatch is detected<br>1: Set DOPCF flag when data match is detected | R/W |
| 4:3 | — | These bits are read as 0. The write value should be 0. | R/W |
| 5 | DOPCF | DOC Flag<br>Indicates the result of an operation. | R |
| 6 | DOPCFCL | DOPCF Clear<br>0: Retain DOPCF flag state<br>1: Clear DOPCF flag | R/W |
| 7 | — | These bits are read as 0. The write value should be 0. | R/W |

Note 1. Only valid when data comparison mode is selected.

#### OMS[1:0] bits (Operating Mode Select)

The OMS[1:0] bits select the operating mode of the DOC.

#### DCSEL bit (Detection Condition Select)

The DCSEL bit selects the detection condition in data comparison mode. This bit is only valid when data comparison mode is selected.

#### DOPCF flag (DOC Flag)

The DOPCF flag indicates the result of an operation.

[Setting conditions]

- The result of data comparison matches the condition selected in the DCSEL bit
- A data addition result is greater than 0xFFFF
- A data subtraction result is less than 0x0000

[Clearing condition]

- Writing 1 to the DOPCFCL bit

#### DOPCFCL bit (DOPCF Clear)

Setting the DOPCFCL bit to 1 clears the DOPCF flag. This bit is read as 0.

### 41.2.2 DODIR : DOC Data Input Register

Base address: DOC = 0x4010_9000

Offset address: 0x02

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | It stores 16-bit data used in the operations. | R/W |

### 41.2.3 DODSR : DOC Data Setting Register

Base address: DOC = 0x4010_9000

Offset address: 0x04

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 15:0 | n/a | It stores 16-bit data used as a reference in data comparison mode. This register also stores the results of operations in data addition and subtraction modes. | R/W |

## 41.3 Operation

### 41.3.1 Data Comparison Mode

Figure 41.2 shows an example operation in data comparison mode operation by the DOC. The following sequence is an example operation when DCSEL is set to 0 (data mismatch is detected as a result of data comparison):

1. Write 00b to the DOCR.OMS[1:0] bits to select data comparison mode.
2. Set 16-bit reference data in DODSR.
3. Write the 16-bit data for comparison to DODIR.
4. Continue writing the 16-bit data until all data for comparison is written to DODIR.
5. If a value written to DODIR does not match that in DODSR, the DOCR.DOPCF flag is set to 1.

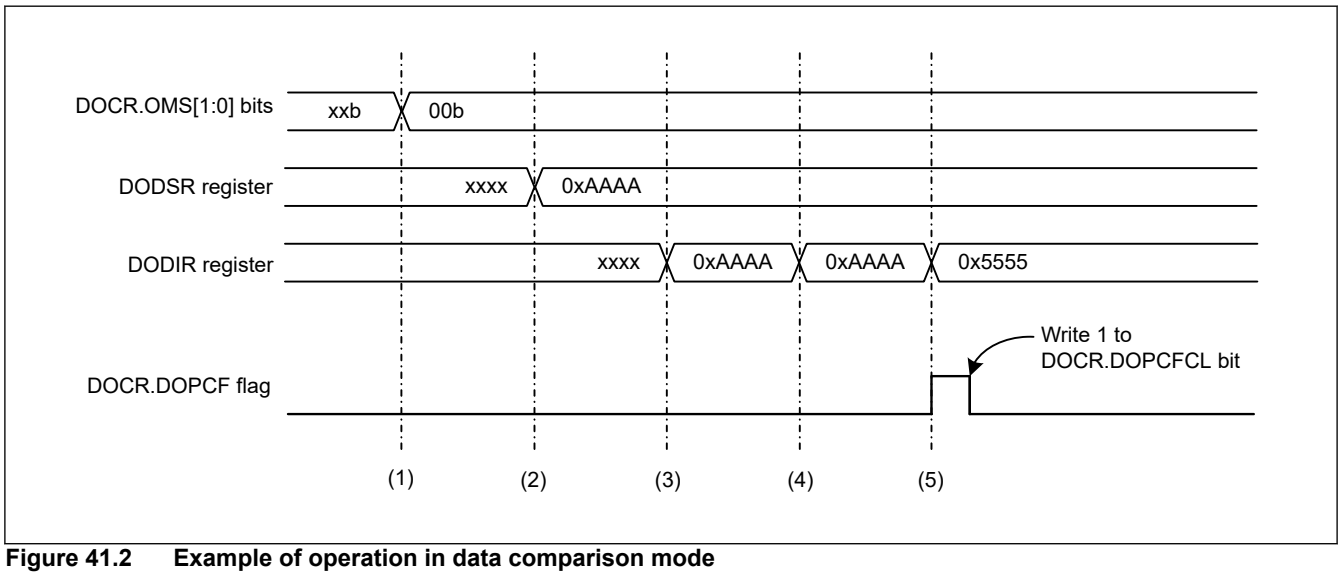



**Figure 41.2 Example of operation in data comparison mode**

### 41.3.2 Data Addition Mode

Figure 41.3 shows an example operation in data addition mode. The steps are as follows:

1. Write 01b to the DOCR.OMS[1:0] bits to select data addition mode.
2. Set 16-bit data as the initial value in the DODSR register.
3. Write the 16-bit data to be added to the DODIR register. The result of the operation is stored in the DODSR register.
4. Continue writing the 16-bit data until all data to be added is written to the DODIR.
5. If the result of an operation is greater than 0xFFFF, the DOCR.DOPCF flag is set to 1.

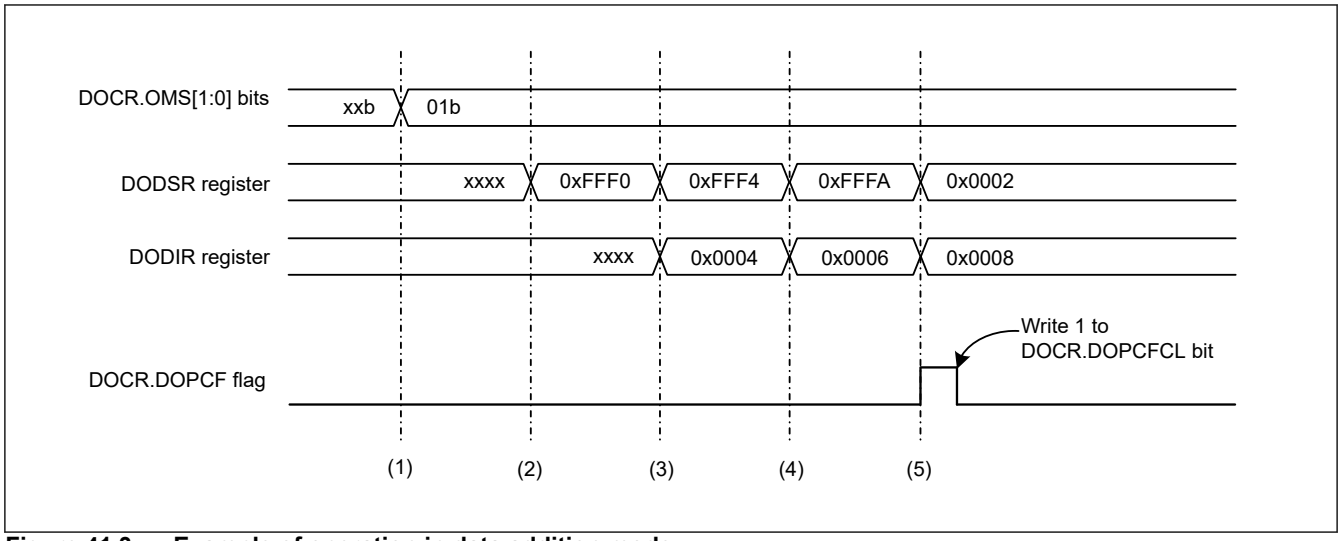



**Figure 41.3 Example of operation in data addition mode**

### 41.3.3 Data Subtraction Mode

Figure 41.4 shows an example operation in data subtraction mode. The steps are as follows:

1. Write 10b to the DOCR.OMS[1:0] bits to select data subtraction mode.
2. Set 16-bit data as the initial value in the DODSR register.
3. Write the 16-bit data to be subtracted to the DODIR register. The result of the operation is stored in DODSR.
4. Continue writing the 16-bit data to the DODIR register until all data to be subtracted is written.
5. If the result of an operation is less than 0x0000, the DOCR.DOPCF flag is set to 1.

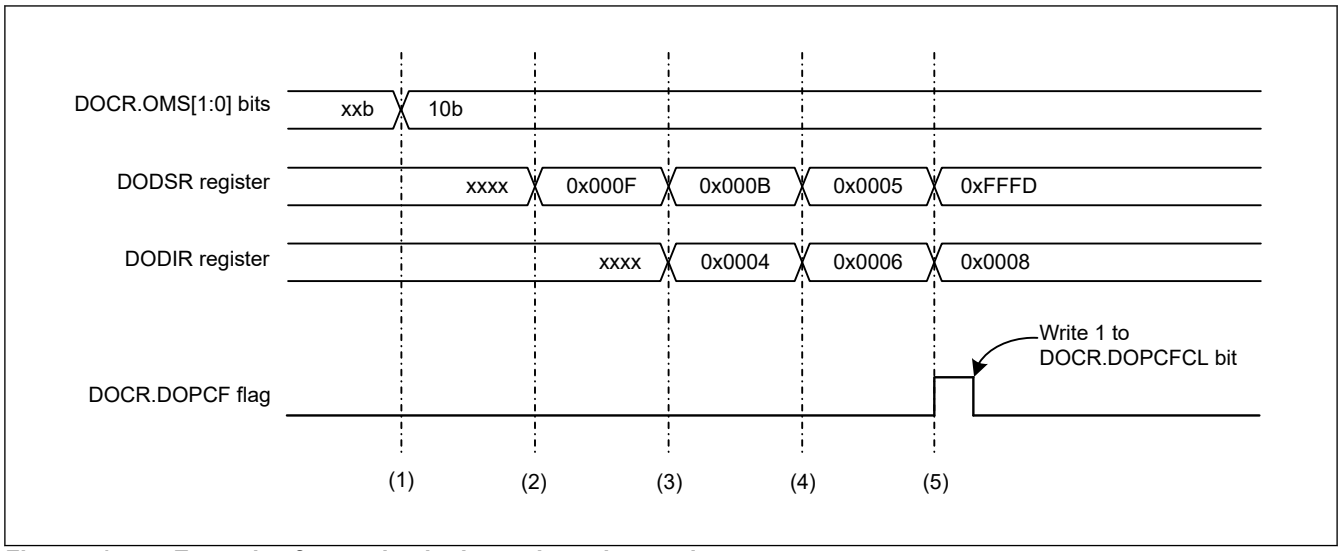



**Figure 41.4 Example of operation in data subtraction mode**

## 41.4 Interrupt Source

The DOC generates the DOC interrupt (DOC_DOPCI) as an interrupt request. Table 41.2 describes the DOC interrupt request.

**Table 41.2 Interrupt request from DOC**

| Interrupt request | Status flag | Interrupt source |
|---|---|---|
| DOC interrupt | DOPCF | • The result of data comparison matches the condition selected in the DOCR.DCSEL bit.<br>• The result of data addition is greater than 0xFFFF.<br>• The result of data subtraction is less than 0x0000. |

## 41.5 Output of an Event Signal to the Event Link Controller (ELC)

The DOC outputs an event signal for the ELC under the following conditions:

- The compared values either match or mismatch
- The data addition result is greater than 0xFFFF
- The data subtraction result is less than 0x0000

This signal can be used to initiate operations by other modules selected in advance and can also be used as an interrupt request. When an event signal is generated, the DOC Flag (DOCR.DOPCF) is set to 1.

## 41.6 Usage Notes

### 41.6.1 Settings for the Module-Stop State

The module Stop Control Register C (MSTPCRC) can enable or disable DOC operation. The DOC is initially stopped after reset. Releasing the module-stop state enables access to the registers. For details, see section 10, Low Power Modes.

# 42. SRAM

## 42.1 Overview

The MCU provides an on-chip, high-density SRAM module with either parity-bit checking or Error Correction Code (ECC). The first 64 KB area of the SRAM0 is the ECC. Parity check is performed on the other areas.

Table 42.1 lists the SRAM specifications.

**Table 42.1 SRAM specifications**

| Parameter | Without ECC | With ECC |
|---|---|---|
| SRAM capacity | SRAM0: 64 KB | SRAM0: 64 KB |
| SRAM address | SRAM0: 0x2001_0000 to 0x2001_FFFF | SRAM0: 0x2000_0000 to 0x2000_FFFF |
| Access | Can access with no wait. One wait access is set at initial state.<br>For details, see section 42.3.9. Access Cycle | |
| Data retention function | Not available in Deep Software Standby mode | |
| Module-stop function | Module-stop state can be set to reduce power consumption | |
| Parity | Even parity with 8-bit data and 1-bit parity | No parity |
| Error checking | even-parity（Date:8bit, parity:1bit） | SEC-DED (Single-Error Correction and Double-Error Detection Code) |
| Security | TrustZone Filter is integrated for memory access and SFR access. Access to the memory space is controlled by setting the memory Security Attribution (SA). And, access to I/O space (SFR) space is controlled by setting the register SA. See section 42.3.6. TrustZone Filter function. | |

## 42.2 Register Descriptions

### 42.2.1 SRAMSAR : SRAM Security Attribution Register

Base address: CPSCU = 0x4000_8000

Offset address: 0x10

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | SRAMSA2 | SRAMSA1 | SRAMSA0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SRAMSA0 | Security attributes of registers for SRAM Protection<br>0: Secure<br>1: Non-Secure | R/W |
| 1 | SRAMSA1 | Security attributes of registers for SRAM Protection 2<br>0: Secure<br>1: Non-Secure | R/W |
| 2 | SRAMSA2 | Security attributes of registers for ECC Relation<br>0: Secure<br>1: Non-Secure | R/W |
| 31:3 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

**SRAMSA0 bit (Security attributes of registers for SRAM Protection)**

Security attributes of registers for SRAM Protection. The target registers are as follow.

- PARIOAD
- SRAMPRCR

**SRAMSA1 bit (Security attributes of registers for SRAM Protection 2)**

Security attributes of registers for SRAM Protection 2. The target registers are as follow.

- SRAMWTSC
- SRAMPRCR2

**SRAMSA2 bit (Security attributes of registers for ECC Relation)**

Security attributes of registers for ECC Relation. The target registers are as follow.

- ECCMODE
- ECC2STS
- ECC1STSEN
- ECC1STS
- ECCPRCR
- ECCPRCR2
- ECCETST
- ECCOAD

## 42.2.2 PARIOAD : SRAM Parity Error Operation After Detection Register

Base address: SRAM = 0x4000_2000

Offset address: 0x00

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | OAD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | OAD | Operation After Detection<br>0: Non-maskable interrupt<br>1: Reset | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The PARIOAD register controls the operation on detection of a parity error. The SRAM Protection Register (SRAMPRCR) protects this register against writes. Always set the SRAMPRCR bit in SRAMPRCR to 1 before writing to this bit. Do not write to the PARIOAD register while accessing the SRAM.

**OAD bit (Operation After Detection)**

The OAD bit specifies the generation of either a reset or non-maskable interrupt when a parity error is detected. The OAD bit is commonly used for SRAM0 (without ECC)/Standby SRAM.

## 42.2.3 SRAMPRCR : SRAM Protection Register

Base address: SRAM = 0x4000_2000

Offset address: 0x04

| Bit position: | 7 | | | | | | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | KW[6:0] | | | | | | | SRAMPRCR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SRAMPRCR | Register Write Control<br>0: Disable writes to protected registers<br>1: Enable writes to protected registers | R/W |
| 7:1 | KW[6:0] | Write Key Code<br>These bits enable or disable writes to the SRAMPRCR bit | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### SRAMPRCR bit (Register Write Control)

The SRAMPRCR bit controls the write mode of the PARIOAD register. Setting the bit to 1 enables writes to the PARIOAD register. When you write to this bit, always write 0x78 to KW[6:0] bits simultaneously.

### KW[6:0] bits (Write Key Code)

The KW[6:0] bits enable or disable writes to the SRAMPRCR bit. When you write to the SRAMPRCR bit, always write 0x78 to these bits simultaneously. When a value other than 0x78 is written to KW[6:0], the SRAMPRCR bit is not updated. The KW[6:0] bits are always read as 0x00.

## 42.2.4 SRAMWTSC : SRAM Wait State Control Register

Base address: SRAM = 0x4000_2000

Offset address: 0x08

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | SRAM0WTEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SRAM0WTEN | SRAM0 wait enable<br>0: No wait<br>1: Add wait state in read access cycle to SRAM0 | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

This register can be rewritten only when the SRAMPRCR2 bit in the SRAMPRCR2 register is 1.

The protection register (SRAMPRCR2) protects this register against writing. Change the effective bit in the protection register (SRAMPRCR2) to write in this register.

Do not write to SRAMWTSC while access to SRAM is in progress.

### SRAM0WTEN bit (SRAM0 wait enable)

This bit sets the wait cycle to the operation region in SRAM0 (Both parity and ECC areas). When it is set 1 in the SRAM0WTEN bit, 1 wait cycle is inserted into the read cycle of operation region in SRAM0. And 1 wait cycle is also inserted between the “write to read/write” sequential cycle in the same region of SRAM0.

## 42.2.5 SRAMPRCR2 : SRAM Protection Register 2

Base address: SRAM = 0x4000_2000

Offset address: 0x0C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | KW[6:0] | | | | | | | SRAMPRCR2 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SRAMPRCR2 | Register Write Control<br>0: Disable writes to the protectedregisters<br>1: Enable writes to the protected registers | R/W |
| 7:1 | KW[6:0] | Write Key Code<br>These bits enable or disable writes to the SRAMPRCR2 bit | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### SRAMPRCR2 bit (Register Write Control)

The SRAMPRCR2 bit controls the write mode of the SRAMWTSC register. Setting the bit to 1 enables writes to the SRAMWTSC register. When you write to this bit, always write 0x78 to KW[6:0] at the same time.

### KW[6:0] bits (Write Key Code)

The KW[6:0] bits enable or disable writes to the SRAMPRCR2 bit. When you write to SRAMPRCR2 bit, always write 0x78 to these bits at the same time. When a value other than 0x78 is written to KW[6:0], the SRAMPRCR2 bit is not updated. The KW[6:0] bits are always read as 0x00.

## 42.2.6 ECCMODE : ECC Operating Mode Control Register

Base address: SRAM = 0x4000_2000

Offset address: 0xC0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | ECCMOD[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | ECCMOD[1:0] | ECC Operating Mode Select<br>0 0: Disable ECC function<br>0 1: Setting prohibited<br>1 0: Enable ECC function without error checking<br>1 1: Enable ECC function with error checking | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:

- Secure and Non-secure access are allowed.

The ECCMODE register specifies the ECC operating mode. The ECC Protection Register (ECCPRCR) protects this register against writes. Before writing to this register, set the ECCPRCR bit in the ECCPRCR register to 1 (write protection disabled). Do not write to the ECCMODE register while accessing the SRAM.

### ECCMOD[1:0] bits (ECC Operating Mode Select)

The ECCMOD[1:0] bits set the access mode to the ECC area in SRAM0.

## 42.2.7 ECC2STS : ECC 2-Bit Error Status Register

Base address: SRAM = 0x4000_2000

Offset address: 0xC1

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | ECC2ERR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ECC2ERR | ECC 2-Bit Error Status<br>0: No 2-bit ECC error occurred<br>1: 2-bit ECC error occurred | R/W*1 |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. Only 0 can be written to clear the bit.

### ECC2ERR bit (ECC 2-Bit Error Status)

The ECC2ERR bit indicates whether a 2-bit ECC error occurred in the ECC area of SRAM0. When a 2-bit error is detected while ECC operations are enabled and error checking is selected, theECC2ERR bit is set to 1. The SRAM error signal is also asserted at this time. The 2-bit ECC error can be cleared by writing 0 to the ECC2ERR bit.

The SRAM error can be specified as a non-maskable interrupt or a reset in the ECCOAD register. Do not access the ECC area in SRAM0 while writing 0 to this register.

## 42.2.8 ECC1STSEN : ECC 1-Bit Error Information Update Enable Register

Base address: SRAM = 0x4000_2000

Offset address: 0xC2

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | E1STSEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | E1STSEN | ECC 1-Bit Error Information Update Enable<br>0: Disable updating of 1-bit ECC error information<br>1: Enable updating of 1-bit ECC error information | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The ECC1STSEN register enables or disables updating of the ECC 1-bit Error Status Register (ECC1STS) in response to a 1-bit error ECC error in the SRAM0 (ECC area).

The ECC Protection Register (ECCPRCR) protects this register against writes. Before writing to this bit, set the ECCPRCR bit in the ECCPRCR register to 1 (write protection disabled).

### E1STSEN bit (ECC 1-Bit Error Information Update Enable)

The E1STSEN bit enables or disables updating of the SRAM (ECC area) 1-Bit Error Status Register (ECC1STS) in response to a 1-bit error in the ECC area of SRAM0. This register also functions as an interrupt or a reset mask.

## 42.2.9 ECC1STS : ECC 1-Bit Error Status Register

Base address: SRAM = 0x4000_2000

Offset address: 0xC3

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | ECC1ERR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ECC1ERR | ECC 1-Bit Error Status<br>0: No 1-bit ECC error occurred<br>1: 1-bit ECC error occurred | R/(W)*1 |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

Note 1. Only 0 can be written to clear the bit.

### ECC1ERR bit (ECC 1-Bit Error Status)

The ECC1ERR bit indicates whether a 1-bit ECC error occurred in the ECC area of SRAM0. When a 1-bit error is detected while ECC operations are enabled and error checking is selected, theECC1ERR bit is set to 1. The SRAM error signal is also asserted at this time. The 1-bit ECC error can be cleared by writing 0 to the ECC1ERR bit.

The SRAM error can be specified as a non-maskable interrupt or a reset in the ECCOAD register. Do not access the ECC area in SRAM0 while writing 0 to this register.

## 42.2.10 ECCPRCR : ECC Protection Register

Base address: SRAM = 0x4000_2000

Offset address: 0xC4

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | KW[6:0] | | | | | | | ECCPRCR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ECCPRCR | Register Write Control<br>0: Disable writes to the protected registers<br>1: Enable writes to the protected registers | R/W |
| 7:1 | KW[6:0] | Write Key Code<br>0x78: Enable write to the ECCPRCR bit<br>Others: Disable write to the ECCPRCR bit | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed

- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:

- Secure and Non-secure access are allowed.

### ECCPRCR bit (Register Write Control)

The ECCPRCR bit controls the write of the ECCMODE, ECC1STSEN, and ECCOAD registers. When this bit is set to 1, writing to the ECCMODE, ECC1STSEN, and ECCOAD registers is enabled. When writing to this bit, write 0x78 to the KW[6:0] bits at the same time.

### KW[6:0] bits (Write Key Code)

The KW[6:0] bits enable or disable writes to the ECCPRCR bit. When writing toECCPRCR bit, write 0x78 to the KW[6:0] bits at the same time. When a value other than 0x78 is written to theKW[6:0] bits, the ECCPRCR bit is not updated. The KW[6:0] bits are always read as 0x00.

## 42.2.11 ECCPRCR2 : ECC Protection Register 2

Base address: SRAM = 0x4000_2000

Offset address: 0xD0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | KW2[6:0] | | | | | | | ECCPRCR2 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ECCPRCR2 | Register Write Control<br>0: Disable writes to the protected registers<br>1: Enable writes to the protected registers | R/W |
| 7:1 | KW2[6:0] | Write Key Code<br>0x78: Enable write to the ECCPRCR2 bit<br>Others: Disable write to the ECCPRCR2 bit | W |

Note: If the security attribution is configured as Secure:

- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:

- Secure and Non-secure access are allowed.

### ECCPRCR2 bit (Register Write Control)

The ECCPRCR2 bit controls the write mode of the ECCETST register. When the ECCPRCR2 bit is set to 1, writes to the ECCETST register is enabled. When writing to this bit, write 0x78 to the KW2[6:0] bits at the same time.

### KW2[6:0] bits (Write Key Code)

The KW2[6:0] bits enable or disable writes to the ECCPRCR2 bit. When writing to ECCPRCR2 bit, write 0x78 to the KW2[6:0] bits at the same time. When a value other than 0x78 is written to the KW2[6:0] bits, the ECCPRCR2 bit is not updated. The KW2[6:0] bits are always read as 0x00.

## 42.2.12 ECCETST : ECC Test Control Register

Base address: SRAM = 0x4000_2000

Offset address: 0xD4

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | TSTBYP |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TSTBYP | ECC Bypass Select<br>0: Disable ECC bypass<br>1: Enable ECC bypass | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The ECC Protection Register 2 (ECCPRCR2) protects this register against writes. Before writing to this bit, set the ECCPRCR2 bit in the ECCPRCR2 register to 1 (write protection disabled). Do not write to the ECCETST register while accessing the SRAM.

#### TSTBYP bit (ECC Bypass Select)

The TSTBYP bit enables direct access to the ECC code by bypassing the ECC function. When the ECC bypass function is used, the ECCMOD[1:0] bits in the ECCMODE register are set to 00b. The ECC must be accessed in 32 bits using the same address for 32-bit data. The ECC code is assigned to the lower 7 bits of the 32-bit data. When writing the ECC code, the upper 25 bits are ignored. When reading the ECC code, the upper 25 bits are undefined.

Note: For details of ECC test, see section 42.3.4. ECC Decoder Testing.

### 42.2.13 ECCOAD : SRAM ECC Error Operation After Detection Register

Base address: SRAM = 0x4000_2000

Offset address: 0xD8

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | OAD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | OAD | Operation After Detection<br>0: Non-maskable interrupt<br>1: Reset | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

The ECC Protection Register (ECCPRCR) protects this register against writes. Before writing to this bit, set the ECCPRCR bit in the ECCPRCR register to 1 (write protection disabled). Do not write to the ECCOAD register while accessing the SRAM.

#### OAD bit (Operation After Detection)

The OAD bit selects whether to generate a reset or a non-maskable interrupt when an ECC error is detected. The OAD bit in the ECCOAD register is used for SRAM0 (ECC area).

## 42.3 Operation

### 42.3.1 Module Stop Function

Power consumption can be reduced by setting module stop control register A (MSTPCRA) to stop supply of the clock signal to SRAM.

SRAM0 is controlled by SRAM0 bit in MSTPCRA register and, in the case of 1, SRAM0 becomes the clock stop state.

The SRAM is thus placed in the module-stop state by stopping supply of the clock signals. The SRAM operates after a reset.

SRAM is not accessible if it is in the module-stop state. A transition to the module-stop state should not be made while access to SRAM is in progress.

Access to the SRAM in the module-stop state is prohibited. If access is attempted, correct operation is not guaranteed.

For details on the MSTPCRA register, see section 10, Low Power Modes.

### 42.3.2 Correction of ECC errors

Enabling and disabling of ECC error correction can be selected through ECCMODE register setting. In the initial state, ECC error correction is disabled. The ECC check type is SEC-DED (Single-Error Correction and Double-Error Detection Code).

When ECC function is enabled, 7-bit check bits are appended to 32-bit data for writing. For reading, 39-bit (data: 32 bits, check bits: 7 bits) data is read out from the SRAM (ECC area).

When ECC function is enabled and error checking is selected by setting ECCMOD[1:0] in the ECCMODE register to 00b, error correction is done if a 1-bit error occurs and the ECC1ERR bit in the ECC1STS register is set to 1 if the E1STSEN bit in the ECC1STSEN register is 1. If a 2-bit error occurs, error detection is done and the ECC2ERR bit in the ECC2STS register is set to 1, though error correction is not performed.

When ECC function is enabled and the error checking is disable, error correction is done if a 1-bit error occurs but ECC1ERR bit in the ECC1STS register is not updated although E1STSEN bit in the ECC1STSEN register is 1. If a 2-bit error occurs, this error is detected but the ECC2ERR bit in the ECC2STS register is not updated, and error correction is not performed.

When ECC function is disable, neither error correction nor error detection is done although 1-bit or 2-bit error occur.

So ECC1ERR bit and ECC2ERR bit are not updated.

There is no way to confirm the location where the error was found. Therefore, when after the occurrence of an error, update all the data.

When updating all the data after the occurrence of an error, the 32-bit data writing is only supported.

Since the SRAM data is undefined after power on and release from Deep Software Standby mode, accessing the SRAM when ECC function is enabled and error checking is selected causes an ECC error to occur. Therefore, before using ECC function, initial writing with 32-bit data size to the area to be used in the SRAM should be done.

When a read access is executed in a row after a write access, read access is executed with priority. Therefore, during initialization, do not perform the read access in a row after the write access.

### 42.3.3 ECC Error Interrupt Function

When ECC function is enabled and error checking is applied to the SRAM (ECC area), an ECC error occurs when either the ECC2ERR bit in the ECC2STS register or the ECC1ERR bit in the ECC1STS register becomes 1 to indicate that ECC checking revealed a 2-bit error or a 1-bit error, respectively.

An ECC erroris output with a pulse width of ICLK.When the ECC 1-bit error is to be masked, set the ECC1STSEN.E1STSEN bit to 0 to disable updating of the ECC1ERR bit. An ECC error will not be generated while ECC function is disabled or when ECC function is enabled but error checking is not selected.

ECC error can choose non maskable interrupt or reset by ECCOAD register. When set 1 in the OAD bit of the ECCOAD register, ECC error is output to the Reset function. When set 0 in the OAD bit of the ECCOAD register, ECC Error interrupt is output to the ICU as non-maskable interrupt.

### 42.3.4 ECC Decoder Testing

Figure 42.1 shows the ECC decoder testing.

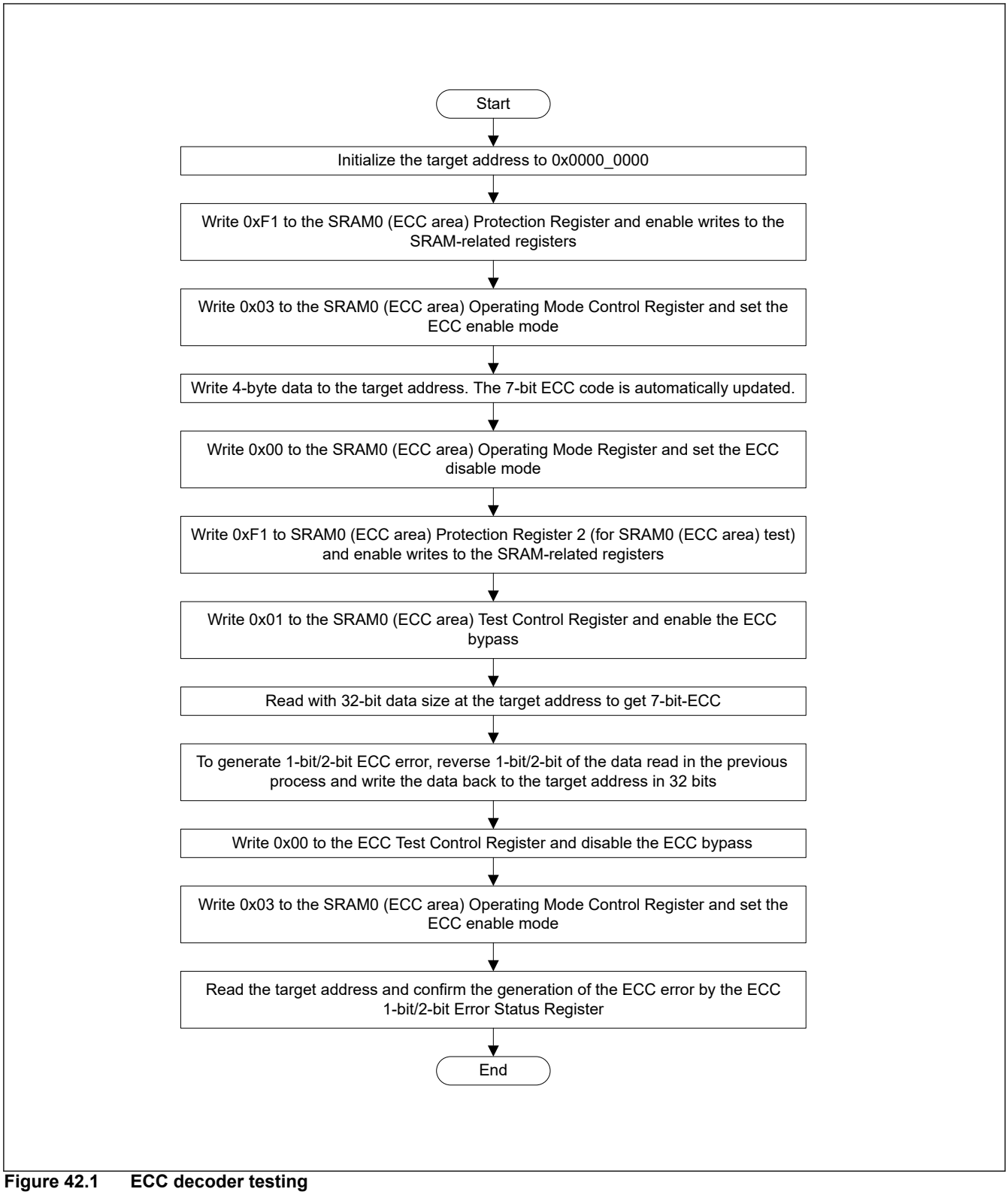



**Figure 42.1 ECC decoder testing**

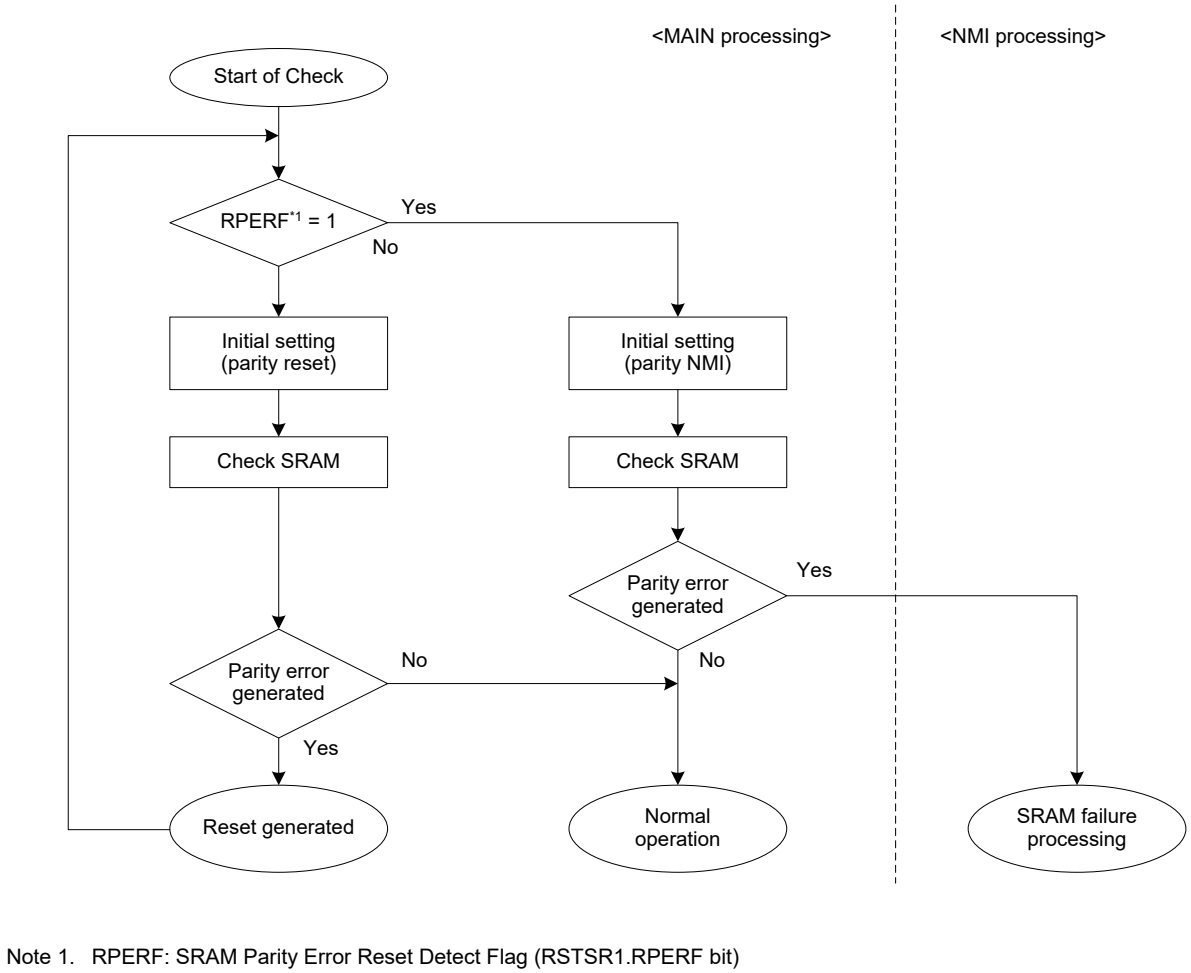
### 42.3.5 Parity Calculation Function

The IEC60730 standard requires the checking of SRAM data. When data is written, a parity bit is added to every 8-bit data in the SRAM which has 32-bit data width, and when data is read, the parity is checked. When a parity error occurs, a parity-error notification is generated. This function can also be used to trigger a reset.

The parity-error notification can be specified as a non-maskable interrupt or a reset in the OAD bit of the PARIOAD register. When the OAD bit is set to 1, a parity error is output to the reset function. When the OAD bit is set to 0, a parity error is output to the ICU as a non-maskable interrupt.

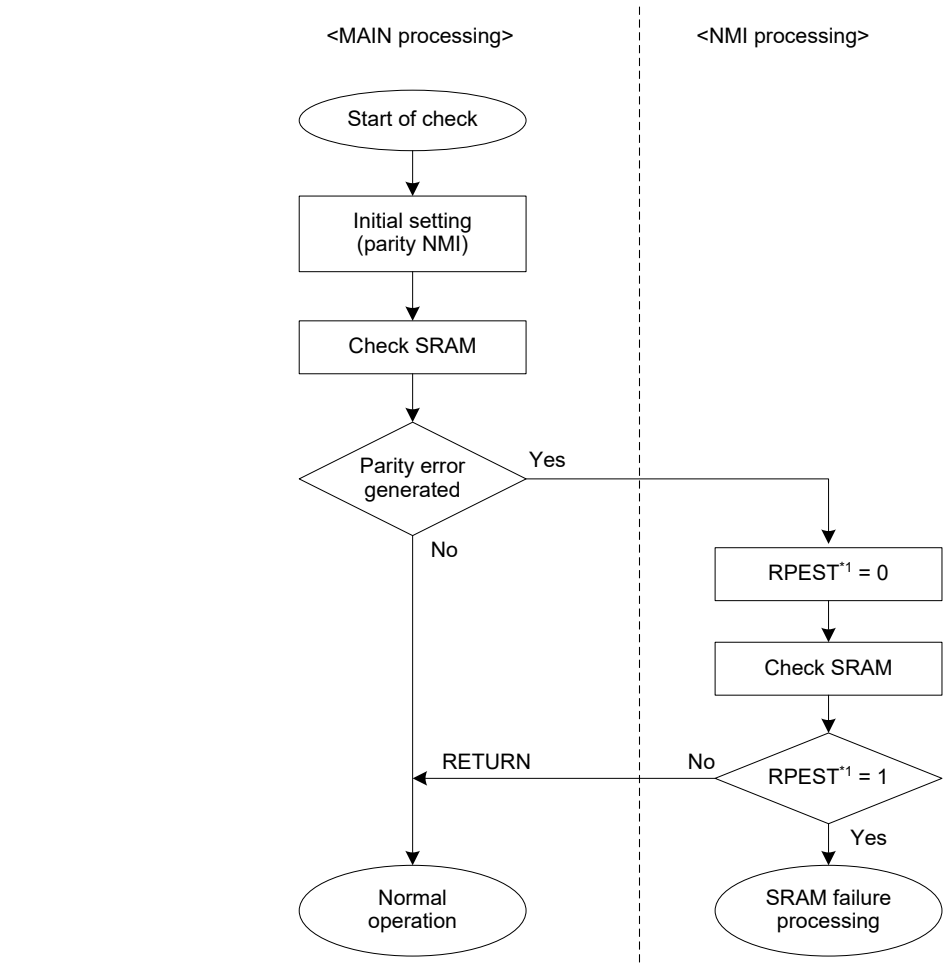
Parity errors can be occasionally caused by noise. To confirm whether the cause of the parity error is noise or corruption, follow the parity check flows shown in Figure 42.2 and Figure 42.3.

When a read access is executed in a row after a write access, read access is executed with priority. Therefore, during initialization, do not perform the read access in a row after the write access.




Note 1. RPERF: SRAM Parity Error Reset Detect Flag (RSTSR1.RPERF bit)

**Figure 42.2 Flow of SRAM parity check when SRAM parity reset is enabled**




Note 1. RPEST: SRAM Parity Error Interrupt Status Flag (NMISR.RPEST bit)

**Figure 42.3 Flow of SRAM parity check when SRAM parity interrupt is enabled**

### 42.3.6 TrustZone Filter function

There are two types of TrustZone Filter function for SRAM.

- TrustZone Filter for SRAM register protection
- TrustZone Filter for SRAM memory protection

#### 42.3.6.1 TrustZone Filter for SRAM register protection

SRAM registers can be protected with a Security Attribution (SA) from Non-secure access. When SA indicates that SRAM registers are secure status, non-secure access cannot overwrite them because TrustZone Filter detects finds an error and protects the write access. SA for SRAM registers is just one to be used commonly among SRAM registers.

**Table 42.2 Register protection (1 of 2)**

| SA | Access status | Write access | Read access |
|---|---|---|---|
| Secure | Secure | Permit | Permit |
| | Non-secure | TrustZone Filter error Protected | Permit |

**Table 42.2 Register protection (2 of 2)**

| SA | Access status | Write access | Read access |
|---|---|---|---|
| Non-secure | Secure | Permit | Permit |
| | Non-secure | Permit | Permit |

When TrustZone Filter error for SRAM register access occurs, no error notification and no error response occurs.

### 42.3.6.2 TrustZone Filter for SRAM memory protection

SRAM memory, e.g. SRAM0 include ECC region and Parity can be divided into Secure/Non secure callable/Non secure status with Memory Security Attribution (MSA) and can be protected from Non-secure access. When MSA indicates that SRAM memory region are Secure or Non secure callable status, Non-secure access can't overwrite them.

**Table 42.3 Memory protection**

| SA | Access status | Write access | Read access |
|---|---|---|---|
| Secure / Non secure callable | Secure | Permit | Permit |
| | Non-secure | TrustZone Filter error<br>• Protected<br>• Error response occurs | TrustZone Filter error<br>• Read data is 0<br>• Error response occurs |
| Non-secure | Secure | Permit | Permit |
| | Non-secure | Permit | Permit |

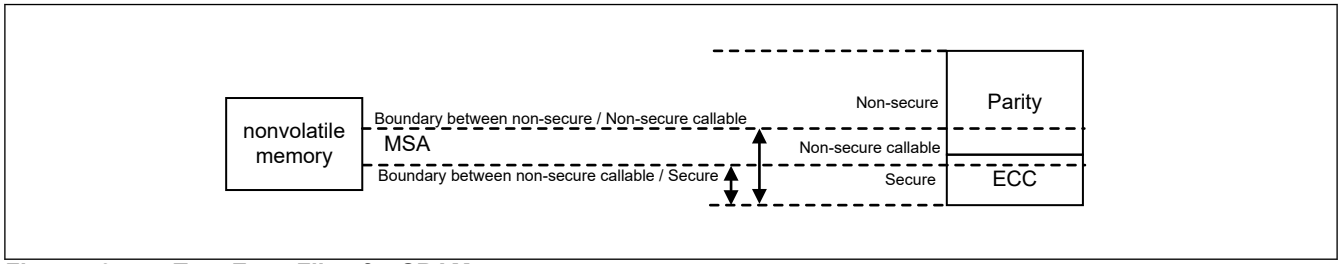



**Figure 42.4 TrustZone Filter for SRAM memory**

When TrustZone Filter error for SRAM memory access occurs, an error notification which become Reset request or NMI request occurs. See section 46.2. Arm TrustZone Security .

### 42.3.7 Interrupt Source

The SRAM interrupt source includes an ECC error, Parity error and TrustZone filter error. ECC error and Parity error can choose non-maskable interrupt or reset by OAD bit. When the debugger is connected, reset and non-maskable interrupt are maskable. Also, if these masks are set by the debugger, each status register is not set even if an ECC error occurs. For details on the debug mode, see section 2, CPU.

**Table 42.4 SRAM Interrupt Source**

| Name | Interrupt Source | DTC Activation | DMAC Activation |
|---|---|---|---|
| ECCERR | ECC error (ECC operation region in SRAM0) | Not possible | Not possible |
| PARITYERR | Parity error | Not possible | Not possible |
| TZFLT | TrustZone filter error | Not possible | Not possible |

### 42.3.8 Wait state

Depending on the operating frequency of ICLK, the WAIT setting for SRAM access has the following conditions.

[ICLK frequency] (SRAM0):

- 100 MHz ≥ ICLK = No-wait

### 42.3.9 Access Cycle

■ Number of cycles from the CPU

**Table 42.5 SRAM0 (ECC Area)**

| Register Setting | | Read (cycles) | | Write (cycles) | |
|---|---|---|---|---|---|
| | | Word access | Half-word/Byte access | Word access | Half-word/Byte access |
| ECC Off ECCMOD[1] = 0 | SRAM0WTEN = 0 | 1 | | 1*1 | |
| | SRAM0WTEN = 1 | 2 | | 1*1 | |
| ECC On ECCMOD[1] = 1 | SRAM0WTEN = 0 | 1 | | 1*1 | 3 |
| | SRAM0WTEN = 1 | 2 | | 1*1 | 3 |

Note 1. For efficiency of the access, when read access occurs to the same memory after write, memory write by the precedent write command delays it until for the next idle cycle or the next write access. When read continues, it is given priority to read.

**Table 42.6 SRAM0 (Parity Area)**

| Register Setting | Read (cycles) | | Write (cycles) | |
|---|---|---|---|---|
| | Word access | Half-word/Byte access | Word access | Half-word/Byte access |
| SRAM0WTEN = 0 | 1 | | 1*1 | |
| SRAM0WTEN = 1 | 2 | | 1*1 | |

Note 1. For efficiency of the access, when read access occurs to the same memory after write, memory write by the precedent write command delays it until for the next idle cycle or the next write access. When read continues, it is given priority to read.

### 42.3.10 ECC encode specification

The following table shows ECC encoding specifications. Add the ECC cord (eout [6:0]) formed by the following calculating formula to higher 7 bits (din [38:32]) of write data and write in it at SRAM.

**Table 42.7 ECC encode**

| ECC code | calculation formula |
|---|---|
| eout[6] | (din[13] ^ din[12] ^ din[11] ^ din[10] ^ din[9] ^ din[8] ^ din[7] ^ din[6] ^ din[5] ^ din[4] ^ din[3] ^ din[2] ^ din[1] ^ din[0]) |
| eout[5] | (din[23] ^ din[22] ^ din[21] ^ din[20] ^ din[19] ^ din[18] ^ din[17] ^ din[16] ^ din[15] ^ din[14] ^ din[3] ^ din[2] ^ din[1] ^ din[0]) |
| eout[4] | (din[29] ^ din[28] ^ din[27] ^ din[26] ^ din[25] ^ din[24] ^ din[17] ^ din[16] ^ din[15] ^ din[14] ^ din[7] ^ din[6] ^ din[5] ^ din[4]) |
| eout[3] | (din[31] ^ din[30] ^ din[26] ^ din[25] ^ din[24] ^ din[20] ^ din[19] ^ din[18] ^ din[14] ^ din[10] ^ din[9] ^ din[8] ^ din[4] ^ din[0]) |
| eout[2] | (din[31] ^ din[30] ^ din[28] ^ din[27] ^ din[24] ^ din[22] ^ din[21] ^ din[18] ^ din[15] ^ din[12] ^ din[11] ^ din[8] ^ din[5] ^ din[1]) |
| eout[1] | ~(din[30] ^ din[29] ^ din[27] ^ din[25] ^ din[23] ^ din[21] ^ din[19] ^ din[16] ^ din[13] ^ din[11] ^ din[9] ^ din[6] ^ din[2] ^ din[0]) |
| eout[0] | ~(din[31] ^ din[29] ^ din[28] ^ din[26] ^ din[23] ^ din[22] ^ din[20] ^ din[17] ^ din[13] ^ din[12] ^ din[10] ^ din[7] ^ din[3] ^ din[0]) |

Note: eout[6:0] = ECC code, din[31:0] = write data

# 43. Standby SRAM

## 43.1 Overview

An on-chip SRAM is provided to retain data in Deep Software Standby mode. Table 43.1 lists the Standby SRAM specifications.

**Table 43.1 Standby SRAM specifications**

| Item | Description |
|---|---|
| SRAM capacity | 1 KB |
| SRAM address | 0x2800_0000 to 0x2800_03FF |
| Access | Standby SRAM clock is the same clock as the PCLKB. See section 43.3.5. Access Cycle for details. |
| Data retention function | Data can be retained in Deep Software Standby mode. See section 43.3.1. Data Retention for details. |
| parity | Even parity (data: 8 bits, parity: 1 bit) |
| Module-stop function | Module-stop state can be set to reduce power consumption. See section 43.3.2. Setting for the Module-stop Function for details. |
| Security | Permits the read and write operations to Standby RAM following TrustZone Filter function. See section 43.3.4. TrustZone Filter function for details. |

## 43.2 Register Descriptions

### 43.2.1 STBRAMSAR : Standby RAM memory Security Attribution Register

Base address: CPSCU = 0x4000_8000

Offset address: 0x014

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | NSBSTBR[3:0] | | | |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | NSBSTBR[3:0] | Security attributes of each region for Standby RAM<br>0x0: Region7-0 are all Secure.<br>0x1: Region7 is Non-secure. Region6-0 are Secure<br>0x2: Region7-6 are Non-secure. Region5-0 are Secure.<br>0x3: Region7-5 are Non-secure. Region4-0 are Secure.<br>0x4: Region7-4 are Non-secure. Region 3-0 are Secure.<br>0x5: Region7-3 are Non-secure. Region 2-0 are Secure.<br>0x6: Region7-2 are Non-secure. Region 1-0 are Secure.<br>0x7: Region7-1 are Non-Secure. Region0 is Secure.<br>Others: Region7-0 are all Non-Secure. | R/W |
| 31:4 | — | This bit is read as 1. | R |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

**NSBSTBR[3:0] bit (Security attributes of each region for Standby RAM)**

Standby RAM is divided into 8 regions. Each region can be set as Secure or Non-secure state with NSBSTBR[3:0]

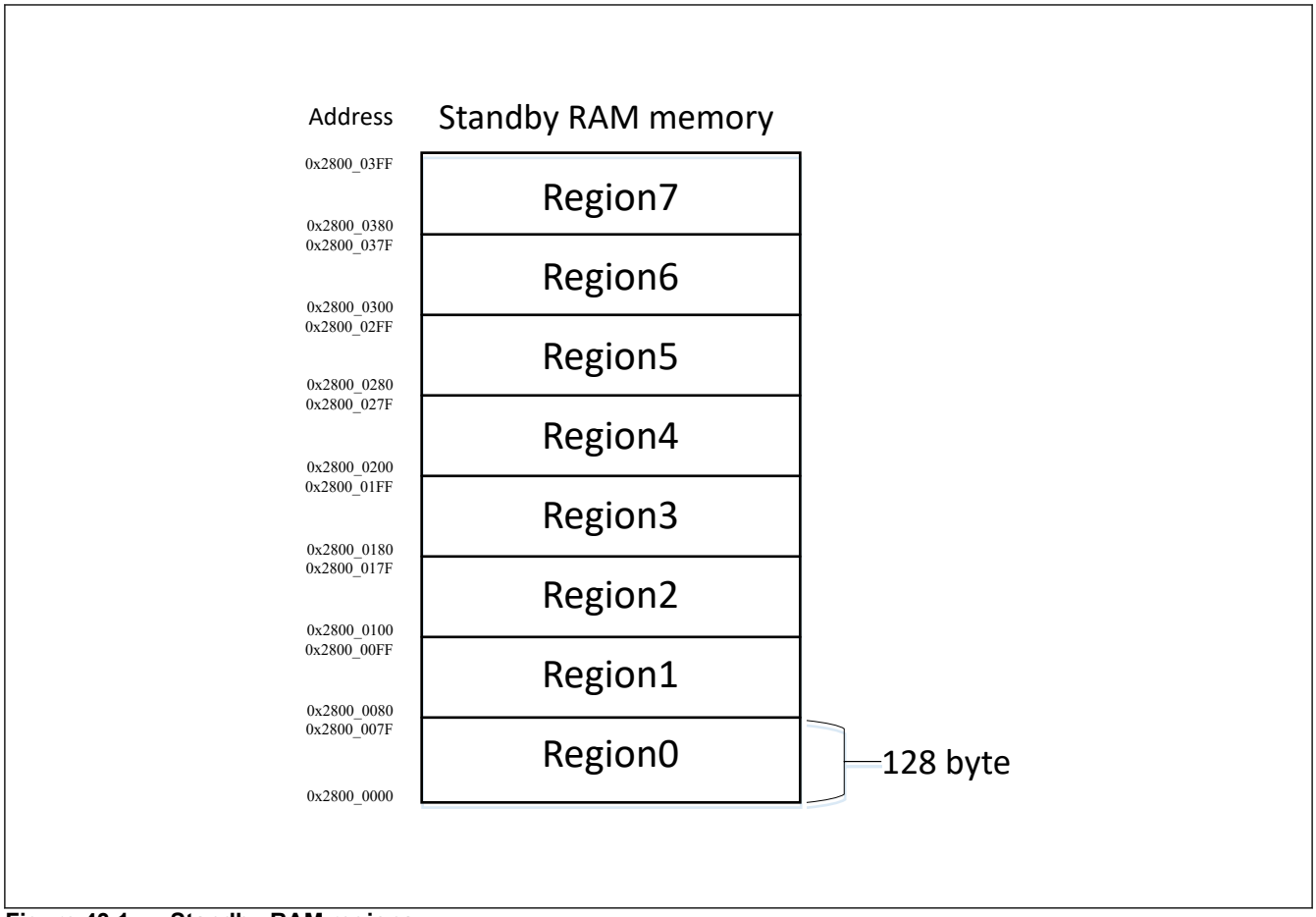



**Figure 43.1 Standby RAM regions**

## 43.3 Operation

### 43.3.1 Data Retention

The power supply to the Standby SRAM in Deep Software Standby mode is enabled by the DPSBYCR.DEEPCUT[1:0] bits. If the DPSBYCR.DEEPCUT[1:0] bits are set to 00b, data in the Standby SRAM is retained in Deep Software Standby mode. For details on the DPSBYCR.DEEPCUT[1:0] bits, see section 10, Low Power Modes.

### 43.3.2 Setting for the Module-stop Function

Power consumption can be reduced by setting module stop control register A (MSTPCRA) to stop supply of the clock signal to SRAM.

If the Standby SRAM bit in MSTPCRA is set to 1, supply of the clock signal to the Standby SRAM is stopped.

The Standby SRAM is thus placed in the module-stop state by stopping supply of the clock signals. The Standby SRAM operates after a reset.

The Standby SRAM is not accessible if it is in the module-stop state. A transition to the module-stop state should not be made while access to the standby SRAM is in progress.

For details on the MSTPCRA register, see section 10, Low Power Modes.

### 43.3.3 Parity Calculation Function

The parity calculation Function for Standby SRAM is as same as SRAM without ECC.

See section 42.3.5. Parity Calculation Function and section 42.3.7. Interrupt Source.

OAD bit in PARIOAD register is commonly used for SRAM0 (without ECC) / Standby SRAM.

### 43.3.4 TrustZone Filter function

There is only one type of TrustZone Filter function for Standby SRAM and that is, TrustZone Filter for SRAM memory protection

#### 43.3.4.1 TrustZone Filter for Standby SRAM Memory Protection

Standby SRAM memory can be divided into 8 regions, 128 bytes each with a Security Attribution (SA) to be protected from Non-secure access. When SA indicates that the region in Standby SRAM is secure status, non-secure access cannot overwrite them because TrustZone Filter detects finds an error and protects the write access.

**Table 43.2 Security Attribution and Access status**

| SA | Access status | Write access | Read access |
|---|---|---|---|
| Secure | Secure | Permit | Permit |
| | Non-secure | TrustZone Filter error<br>- Protected | TrustZone Filter error<br>- Read data is 0 |
| Non-secure | Secure | Permit | Permit |
| | Non-secure | Permit | Permit |

When TrustZone Filter error for Standby SRAM access occurs, no error notification and no error response occurs.

### 43.3.5 Access Cycle

Number of cycles from the CPU.

**Table 43.3 Standby SRAM (parity area 0x2800_0000 to 0x2800_03FF)**

| | Read cycle | | Write cycle | |
|---|---|---|---|---|
| | Word access | Halfword/Byte access | Word access | Halfword/Byte access |
| ICLK ≥ PCLKB | Min.: 2 PCLKB<br>Max.: (n -1) ICLK + 2 PCLKB | | Min.: 1 PCLKB<br>Max.: (n -1) ICLK + 1 PCLKB | |

Note: When the frequency ratio of ICLK : PCLKB is n : 1.

## 43.4 Usage Notes

### 43.4.1 Instruction Fetch from the Standby SRAM Area

When using Standby SRAM to operate a program, initialize the Standby SRAM area so that the CPU can correctly prefetch data. A parity error might occur if the CPU prefetches from an area that is not initialized. Initialize the additional 12-byte area from the end address of the program with the 4-byte boundary. Renesas recommends using the NOP instruction for data initialization.

# 44. Flash Memory

This MCU incorporates code flash memory, data flash memory, and option-setting memory. The code flash memory stores instructions and operands, and the data flash memory stores data. For option-setting memory, see section 6, Option-Setting Memory.

## 44.1 Overview

Table 44.1 lists the specifications of the flash memory, and Figure 44.1 is block diagrams of the flash memory related modules.

The I/O pins used in boot mode, see Table 44.27.

The FCU (flash control unit) controls programming and erasure of the flash memory. The FACI (flash application command interface) controls the FCU according to the specified FACI commands.

Regarding the configuration of the code flash memory, see Figure 44.2, and for the configuration of the data flash memory, see Figure 44.3.

**Table 44.1 Specifications of flash memory (1 of 2)**

| Item | Code flash memory | Data flash memory |
|---|---|---|
| Memory capacity | User area: 512 Kbytes max | Data area: 8 Kbytes |
| Read cycle | See section 44.16.3. Access Cycle | See section 44.16.3. Access Cycle |
| Value after erasure | 0xFF | Undefined |
| Programming/erasing method | • Programming and erasing the code flash memory and data flash memory, and programming the option-setting memory are handled by the FACI commands specified in the FACI command issuing area (0x407E_0000) (self-programming).<br>• Programming/erasure through transfer by a serial-programmer via a serial interface (serial programming) | |
| Protection | Protects against erroneous rewriting of the flash memory | |
| Background operations (BGOs)*1 | • The data flash memory can be read while the code flash memory is being programmed or erased.<br>• The code flash memory can be read while the data flash memory is being programmed or erased. | |
| Units of programming and erasure | • Units of programming for the user area: 128 bytes<br>• Units of erasure for the user area: Block units | • Unit of programming for the data area:4/8/16 bytes<br>• Unit of erasure for the data area:64/128/256 bytes |
| Other functions | Interrupts can be accepted during self-programming.<br>In the initial settings of this MCU, an expansion area of the option-setting memory can be set. | |
| On-board programming (four types) | Programming/erasure in boot mode (for the SCI interface)<br>• The asynchronous serial interface (SCI9) is used.<br>• The transfer rate is adjusted automatically.<br>Programming/erasure in boot mode (for the USB interface)<br>• USBFS is used.<br>• Dedicated hardware is not required, so direct connection to a PC is possible.<br>Programming/erasure in On-chip debug mode<br>• JTAG or SWD interface is used<br>Programming and erasure by self-programming<br>• This allows code flash memory programming/erasure without resetting the system. | |
| Unique ID | A 16-byte ID code provided for each MCU | |
| FACI command | Program : 128 bytes<br>Block erase: 1 block (8 KB or 32 KB)<br>P/E suspend<br>P/E resume<br>Forced Stop<br>Status Clear<br>Configuration set (16 bytes) | Program: 4/8/16 bytes<br>Block Erase: 1 block (64 bytes)<br>Multi Block Erase: 64/128/256 bytes<br>P/E suspend<br>P/E resume<br>Forced Stop<br>Blank Check: 4 bytes to data flash memory capacity<br>Status Clear |

**Table 44.1 Specifications of flash memory (2 of 2)**

| Item | Code flash memory | Data flash memory |
|---|---|---|
| Security function | Protects against illicit tampering with or reading out of data in flash memory<br>Startup area select setting protection<br>● BTFLG and FSUACR registers are protected by the FSPR bit.<br>Permanent block protect setting protection<br>● Code flash memory is permanently protected from programming/erasure operation by the permanent block protect function.<br>Flash memory protection for TrustZone<br>● Protection for flash memory area (P/E)<br>● Protection for flash memory area (read)<br>● Protection for register<br>● Protection during FACI command operation.<br>● Code flash P/E mode entry protection | |
| Safety function | Software protection<br>● FACI command protection by FENTRYR register.<br>● Flash memory is protected by FWEPROR register<br>● The user area is protected by the block protect setting<br>Error protection<br>● Error is detected when unintended commands or prohibited settings occur. The FACI command is not accepted after an error detection.<br>Boot area protection<br>● The start-up area select function allows customer to safely update the boot firmware. The size of the start-up area is 8 KB. | |
| Interrupt request | ● FRDYI (flash sequencer ready (processing end)) : Enabled by FRDYIE bit.<br>● FIFERR (flash sequencer error) : Enabled by CFAEIE/CMDLKIE/DFAEIE bits | |
| Address conversion | ● Start-up area select function is supported | |

Note 1. Limitations apply to the combinations of the address ranges for programming/erasure process and reading process: see Table 44.29.

Figure 44.1 shows how modules related to flash memory can be configured. The flash sequencer is configured with the FCU and FACI. The FCU executes basic control for rewriting of the flash memory. The FACI receives FACI commands using peripheral bus, and controls FCU operations accordingly.

In response to a reset, the FACI transfers data from the flash memory to the option byte storage registers.

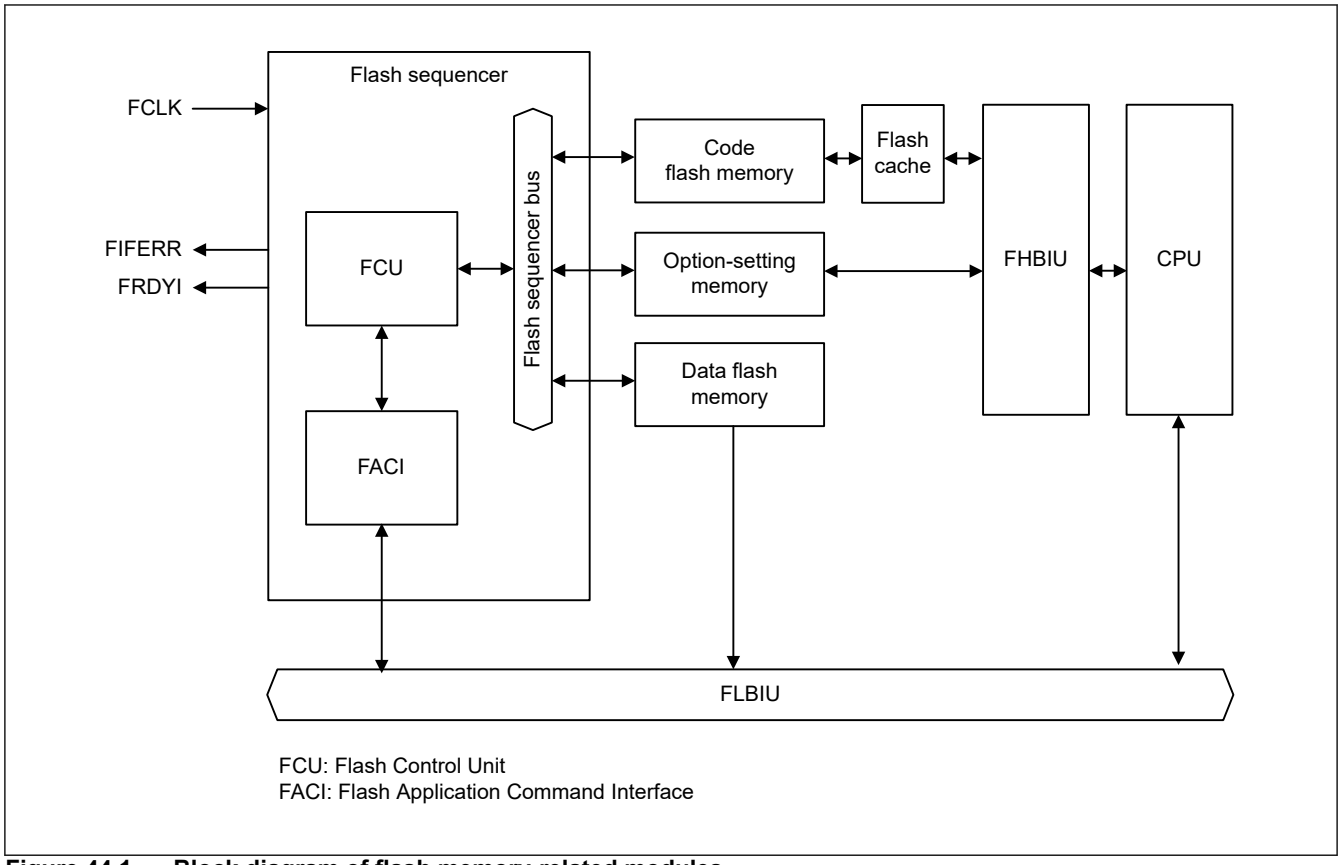



**Figure 44.1 Block diagram of flash memory-related modules**

## 44.2 Structure of Memory

Figure 44.2 shows the memory map of code flash memory.

The user area of the code flash memory in this MCU is divided into 8- and 32-Kbyte blocks, which serve as the units of erasure.

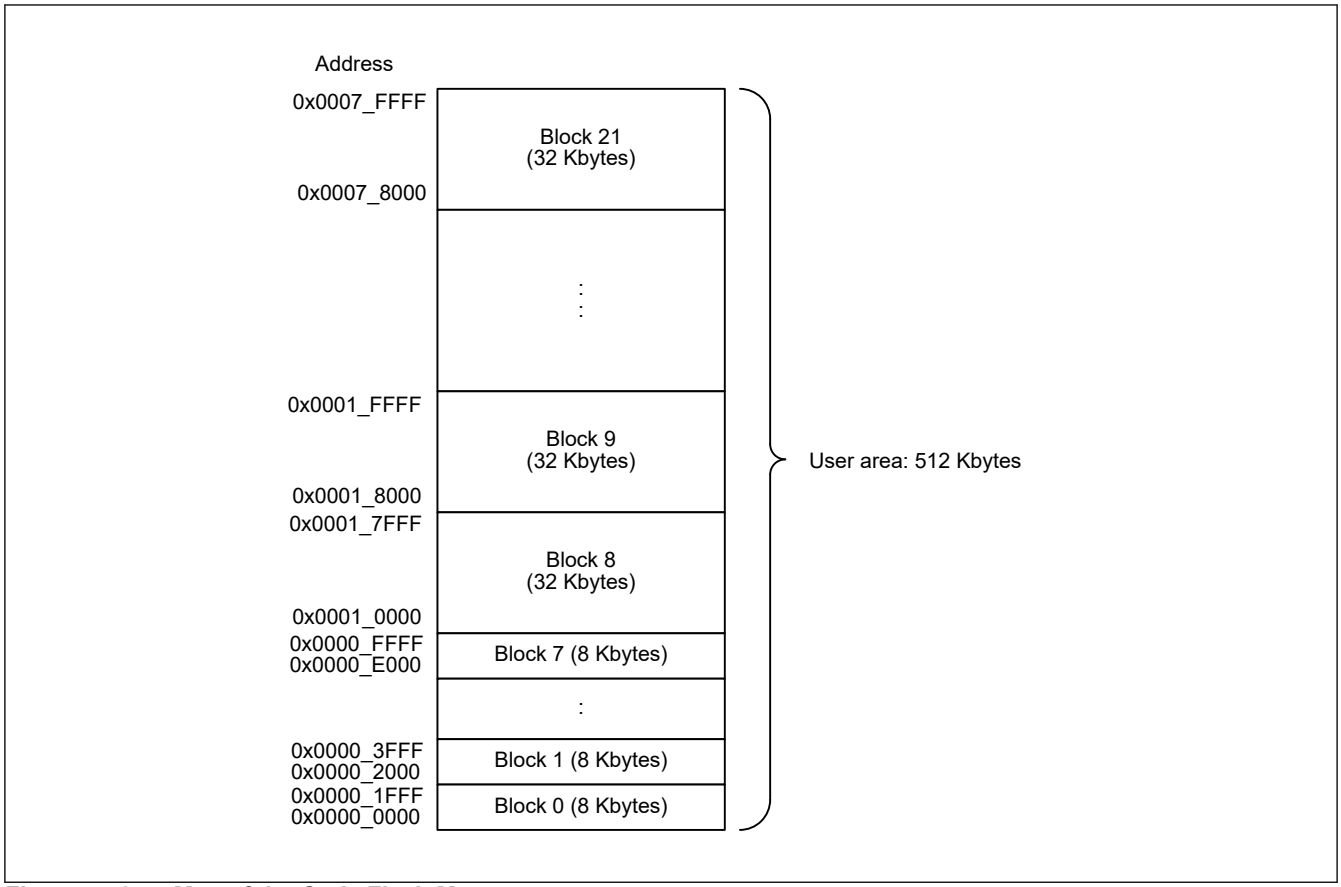



**Figure 44.2 Map of the Code Flash Memory**

**Table 44.2 Read and programming/erasure address by product for the code flash memory**

| Product | Address | Number of blocks |
|---|---|---|
| 512 Kbytes product | 0x0000_0000 to 0x0007_FFFF | 0 to 21 |
| 384 Kbytes product | 0x0000_0000 to 0x0005_FFFF | 0 to 17 |
| 256 Kbytes product | 0x0000_0000 to 0x0003_FFFF | 0 to 13 |

The data area of the data flash memory in this MCU is divided into 64-byte blocks, with each being a unit for erasure. Figure 44.3 shows the mapping of the data flash memory.

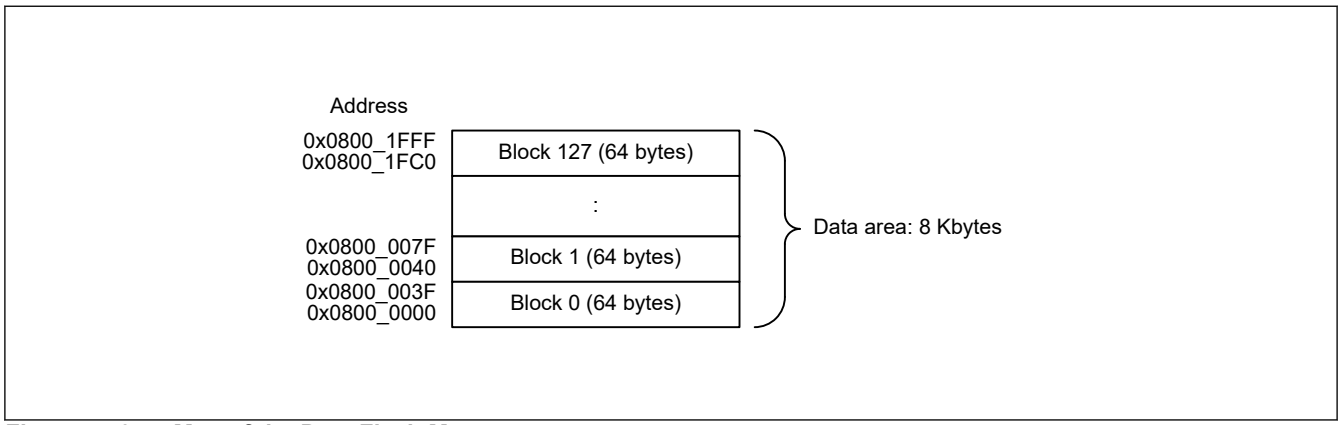



**Figure 44.3 Map of the Data Flash Memory**

## 44.3 Address Space

Using the hardware interface with flash memory requires access to all registers of the hardware, which is for issuing FACI commands. Table 44.3 provides information about the hardware interface.

**Table 44.3 Information on the hardware interface area**

| Area | Address | Capacity |
|---|---|---|
| Area containing various registers of the hardware | See section 44.4. Register Descriptions. | See section 44.4. Register Descriptions. |
| FACI command-issuing area | 0x407E_0000 | 4 bytes |

For the address information of the flash memory, see Figure 44.2.

## 44.4 Register Descriptions

### 44.4.1 FCACHEE : Flash Cache Enable Register

Base address: FCACHE = 0x4001_C100

Offset address: 0x000

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | FCACHEEN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FCACHEEN | Flash Cache Enable<br>0: FCACHE is disabled<br>1: FCACHE is enabled | R/W |
| 15:1 | — | These bits are read as 0. The write value should be 0. | R/W |

This register is not controlled by any security attribute register.

**FCACHEEN bit (Flash Cache Enable)**

FCACHEE.FCACHEEN bit enable and disables the function of Flash Cache of FCACHE1, FCACHE2 and FLPF.

FCACHEE.FCACHEEN bit dose not influence for FCACHEIV.FCACHEIV.

When FCACHE is enabled, it works for accesses marked as cacheable.

It is prohibited to disable FCACHE after enabling.

### 44.4.2 FCACHEIV : Flash Cache Invalidate Register

Base address: FCACHE = 0x4001_C100

Offset address: 0x004

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | FCACHEIV |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FCACHEIV | Flash Cache Invalidate<br>0: Read: Do not invalidate.<br>Write: The setting is ignored.<br>1: Invalidate<br>FCACHE is invalidated. | R/W |
| 15:1 | — | These bits are read as 0. The write value should be 0. | R/W |

This register is not controlled by any security attribute register.

### FCACHEIV bit (Flash Cache Invalidate)

When 1 is written to FCACHEIV.FCACHEIV bit, the Flash cache data of FCACHE1, FCACHE2 and FLPF is invalidated.

Invalidate FCACHE with keeping FCACHE enabled after programming or erasing the code flash or the option setting memory.

## 44.4.3 FLWT : Flash Wait Cycle Register

Base address: FCACHE = 0x4001_C100

Offset address: 0x01C

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | FLWT[2:0] | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | FLWT[2:0] | Flash Wait Cycle<br>0 0 0: 0 wait (ICLK ≤ 50 MHz)<br>0 0 1: 1 wait (ICLK > 50 MHz)<br>0 1 0: 2 wait Not specified<br>0 1 1: 3 wait Not specified<br>Others: Setting prohibited | R/W |
| 7:3 | — | These bits are read as 0. The write value should be 0. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### FLWT[2:0] bits (Flash Wait Cycle)

The Flash Wait Cycle Register (FLWT) sets the access wait count for the flash memory.

For faster clock frequencies, set FLWT.FLWT before changing the clock frequency. For slower clock frequencies, set FLWT.FLWT after changing the clock frequency.

For information on the frequency setting, see section 8, Clock Generation Circuit.

## 44.4.4 FSAR : Flash Security Attribution Register

Base address: FCACHE = 0x4001_C100

Offset address: 0x040

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | FCKMHZSA | — | — | — | — | — | — | — | FLWTSA |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FLWTSA | FLWT Security Attribution<br>Target register : FLWT<br>0: Secure<br>1: Non-Secure | R/W |
| 7:1 | — | These bits are read as 1. The write value should be 1. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 8 | FCKMHZSA | FCKMHZ Security Attribution<br>Target register : FCKMHZ<br>0: Secure<br>1: Non-Secure | R/W |
| 15:9 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

Write access is invalid when PRCR.PRC4 bit is 0. See section 12, Register Write Protection.

#### FLWTSA bit (FLWT Security Attribution)

This bit sets the security attribute of FLWT.

#### FCKMHZSA bit (FCKMHZ Security Attribution)

This bit sets the security attribute of FCKMHZ.

### 44.4.5 UIDRn : Unique ID Registers n (n = 0 to 3)

Address: 0x0100_8190 + n × 4

| Bit position: | 31 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | UID | | | | | | | | | | | | | | | |
| Value after reset: | Unique value for each chip | | | | | | | | | | | | | | | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | UID | Unique ID | R |

The UIDRn is a read-only register that stores a 16-byte ID code (unique ID) for identifying the individual MCU. The UIDRn register should be read in 32-bit units. When reading by the signature request command of the serial programming interface, the data is read in order from the data with the large address. That is, the data in 0x0100_819F is read first, and in 0x0100_8190 is read last.

### 44.4.6 PNRn : Part Numbering Register n (n = 0 to 3)

Address: 0x0100_80F0 + n × 4

| Bit position: | 31 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PNR | | | | | | | | | | | | | | | |
| Value after reset: | Unique value for each chip | | | | | | | | | | | | | | | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | PNR | Part Number | R |

The PNRn is a read-only register that stores a 16-byte part numbering. The PNRn register should be read in 32-bit units. Each byte corresponds to the ASCII code representation of the product part number as detailed in Table 1.14. The first character ("R", 0x52 in ASCII code) of the part number is stored in the byte with the smallest address (0x0100_80F0). When reading by the signature request command of the serial programming interface, the data is read in order from the data with the small address. That is, the data in 0x0100_80F0 is read first, and in 0x0100_80FF is read last.

## 44.4.7 MCUVER : MCU Version Register

Address: 0x0100_81B0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | MCUVE | | | | | | | |
| Value after reset: | Value depend on the chip | | | | | | | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | MCUVE | MCU Version | R |

The MCUVER is a read-only register that stores a MCU version. The MCUVER register should be read in 8-bit units.

## 44.4.8 FWEPROR : Flash P/E Protect Register

Base address: SYSC = 0x4001_E000

Offset address: 0x416

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | FLWE[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | FLWE[1:0] | Flash Programming and Erasure<br>0 0: Prohibits Program, Block Erase, Multi Block Erase, Blank Check, and Configuration set command processing.<br>0 1: Permits Program, Block Erase, Multi Block Erase, Blank Check, and Configuration set command processing.<br>1 0: Prohibits Program, Block Erase, Multi Block Erase, Blank Check, and Configuration set command processing.<br>1 1: Prohibits Program, Block Erase, Multi Block Erase, Blank Check, and Configuration set command processing. | R/W |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |

It is possible that Program, Block Erase, Multi Block Erase, Blank Check, and Configuration set command processing are prohibited by software.

The FWEPROR register is initialized by a reset from the following:

- All reset source
- Transition to Deep Software Standby mode
- Transition to Software Standby mode.

### FLWE[1:0] bits (Flash Programming and Erasure)

The FLWE[1:0] bits are used to set the flash P/E protection. The value after reset is 10b.

If these bits are set to other than 01b that does not allow programming and erasure of the flash memory, the following commands cannot be executed. Issuing any of the following commands leads to setting of the FLWEERR bit in the FSTATR register to 1.

Program / Block Erase / Multi Block Erase / Blank Check / Configuration set command

## 44.4.9 FASTAT : Flash Access Status Register

Base address: FACI = 0x407F_E000

Offset address: 0x10

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CFAE | — | — | CMDLK | DFAE | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 3 | DFAE | Data Flash Memory Access Violation Flag<br>0: No data flash memory access violation has occurred<br>1: A data flash memory access violation has occurred. | R/W*1 |
| 4 | CMDLK | Command Lock Flag<br>0: The flash sequencer is not in the command-locked state<br>1: The flash sequencer is in the command-locked state. | R |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | CFAE | Code Flash Memory Access Violation Flag<br>0: No code flash memory access violation has occurred<br>1: A code flash memory access violation has occurred. | R/W*1 |

Note 1. Only 0 can be written to clear the flag after 1 is read.

The FASTAT register indicates whether a code flash or data flash memory access violation has occurred. If any of the CFAE, CMDLK, and DFAE bits is set to 1, the flash sequencer enters the command-locked state (see section 44.11.2. Error Protection). To release it from the command-locked state, issue a status clear command or Forced Stop command to the flash sequencer.

### DFAE bit (Data Flash Memory Access Violation Flag)

The DFAE bit indicates whether a data flash memory access violation occurred. When this bit is set to 1, the ILGLERR bit in the FSTATR register is set to 1, placing the flash sequencer in the command-locked state.

[Setting conditions]

FACI commands issued in the data flash P/E mode are as follows:

- The setting of the FSADDR or FEADDR register is the reserved portion of the data area
- FACI commands of non-secure access are issued while the setting of the FSADDR or FEADDR register is the secure region address.

[Clearing conditions]

- When this bit is written as 0 after it is set to 1
- When the flash sequencer starts to process the Status Clear or Forced Stop command.

### CMDLK bit (Command Lock Flag)

The CMDLK bit indicates that the flash sequencer is in the command-locked state.

[Setting conditions]

- The flash sequencer detects an error and enters the command-locked state.

[Clearing conditions]

- When the flash sequencer starts to process the Status Clear or Forced Stop command.

### CFAE bit (Code Flash Memory Access Violation Flag)

The CFAE bit indicates whether a code flash memory access violation has occurred. When this bit is set to 1, the ILGLERR bit in the FSTATR register is set to 1, placing the flash sequencer in the command-locked state.

[Setting conditions]

FACI commands issued in the code flash P/E mode are as follows:

- The setting of the FSADDR register is the reserved portion of the user area
- The Configuration set command is issued while the setting of the FSADDR register is from 0x0000A100 to 0x0000A2F0 in self-programming mode
- FACI commands of non-secure access are issued while the setting of the FSADDR register is the secure region address.

[Clearing conditions]

- When this bit is written as 0 after it is set to 1
- When the flash sequencer starts to process the Status Clear or Forced Stop command.

## 44.4.10 FAEINT : Flash Access Error Interrupt Enable Register

Base address: FACI = 0x407F_E000

Offset address: 0x14

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | CFAEIE | — | — | CMDLKIE | DFAEIE | — | — | — |
| Value after reset: | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 2:0 | — | These bits are read as 0. The write value should be 0. | R/W |
| 3 | DFAEIE | Data Flash Memory Access Violation Interrupt Enable<br>0: Generation of an FIFERR interrupt request is disabled when FASTAT.DFAE is set to 1<br>1: Generation of an FIFERR interrupt request is enabled when FASTAT.DFAE is set to 1. | R/W |
| 4 | CMDLKIE | Command Lock Interrupt Enable<br>0: Generation of an FIFERR interrupt request is disabled when FASTAT.CMDLK is set to 1<br>1: Generation of an FIFERR interrupt request is enabled when FASTAT.CMDLK is set to 1. | R/W |
| 6:5 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | CFAEIE | Code Flash Memory Access Violation Interrupt Enable<br>0: Generation of an FIFERR interrupt request is disabled when FASTAT.CFAE is set to 1<br>1: Generation of an FIFERR interrupt request is enabled when FASTAT.CFAE is set to 1. | R/W |

The FAEINT register enables or disables generation of a flash access error (FIFERR) interrupt request.

### DFAEIE bit (Data Flash Memory Access Violation Interrupt Enable)

The DFAEIE bit enables or disables generation of an FIFERR interrupt request when a data flash memory access violation occurs, setting the DFAE bit in the FASTAT register to 1.

### CMDLKIE bit (Command Lock Interrupt Enable)

The CMDLKIE bit enables or disables generation of an FIFERR interrupt request when the flash sequencer enters the command-locked state, setting the CMDLK bit in the FASTAT register to 1.

### CFAEIE bit (Code Flash Memory Access Violation Interrupt Enable)

The CFAEIE bit enables or disables generation of an FIFERR interrupt request when a code flash memory access violation occur, setting the CFAE bit in the FASTAT register to 1.

## 44.4.11 FRDYIE : Flash Ready Interrupt Enable Register

Base address: FACI = 0x407F_E000

Offset address: 0x18

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | FRDYIE |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FRDYIE | Flash Ready Interrupt Enable<br>0: Generation of an FRDY interrupt request is disabled<br>1: Generation of an FRDY interrupt request is enabled. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

The FRDYIE register enables or disables generation of a flash ready (FRDY) interrupt request.

### FRDYIE bit (Flash Ready Interrupt Enable)

The FRDYIE bit enables or disables generation of an FRDY interrupt request when the FRDY bit in the FSTATR register is changed from 0 to 1 on completion of processing by the flash sequencer of the Program, Block Erase, Multi Block Erase, Blank Check, and Configuration set command.

## 44.4.12 FSADDR : FACI Command Start Address Register

Base address: FACI = 0x407F_E000

Offset address: 0x30

| Bit position: | 31 | 0 |
|---|---|---|
| Bit field: | FSADDR[31:0] | |
| Value after reset: | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | FSADDR[31:0] | Start Address for FACI Command Processing | R/W*1 |

Note 1. These bits can be written when the FRDY bit in the FSTATR register is 1. Writing to these bits are ignored when the FRDY bit is 0. Note that b0 and b1 are read-only.

**Table 44.4 FACI command addres boundary**

| Command | Address Boundary |
|---|---|
| Program (code flash memory) | 128-byte |
| Program (data flash memory) | 4, 8, 16 -byte |
| Block Erase (code flash memory) | 8-KB or 32-KB |
| Block Erase (data flash memory) | 64-byte |
| Multi Block Erase (data flash memory) | 64-byte |
| Blank Check (data flash memory) | 4-byte |
| Configuration set | 16-byte |

The FSADDR register specifies the address where the target area for command processing starts when the FACI command for Program, Block Erase, Multi Block Erase, Blank Check, or Configuration set is issued.

The FSADDR value is initialized when the SUINIT bit in the FSUINITR register is set to 1. It is also initialized by a reset.

### FSADDR[31:0] bits (Start Address for FACI Command Processing)

The FSADDR[31:0] bits specify the start address for FACI command processing. Bits [31:24] are ignored in FACI command processing for the code flash memory. Bits [31:17] are ignored in FACI command processing for the data flash memory. Bits associated with the address bits of lower order than the address boundary listed in Table 44.4 are also ignored.

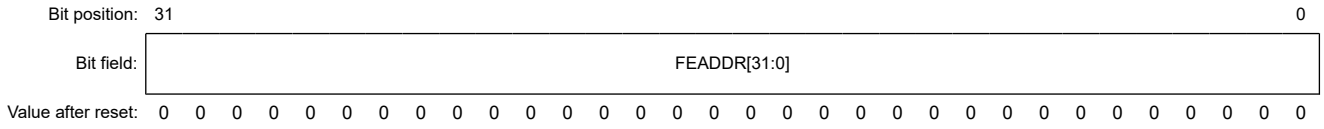
For information on the addresses of the code flash memory and the data flash memory, see section 44.2. Structure of Memory.

For information on the addresses of the configuration setting, see section 44.9.3.15. Configuration Set Command.

## 44.4.13 FEADDR : FACI Command End Address Register

Base address: FACI = 0x407F_E000

Offset address: 0x34




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 31:0 | FEADDR[31:0] | End Address for FACI Command Processing | R/W*1 |

Note 1. These bits can be written when the FRDY bit in the FSTATR register is 1. Writing to these bits are ignored when the FRDY bit is 0. Note that bit [0] and bit [1] are read-only.

The FEADDR register specifies the end address of the target area for Multi Block Erase and Blank Check command processing. When incremental mode is selected as the addressing mode for Blank Checking (when FBCCNT.BCDIR = 0), the address specified in the FSADDR register should be equal to or smaller than the address in the FEADDR register. Conversely, the address in the FSADDR register should be equal to or larger than the address in the FEADDR register when decremental mode is selected as the addressing mode for Blank Check (i.e. when FBCCNT.BCDIR = 1). If the BCDIR, FSADDR, and FEADDR bit settings are inconsistent with the specified rules, the flash sequencer enters the command-locked state (see section 44.11.2. Error Protection).

The FEADDR value is initialized when the SUINIT bit in the FSUINITR register is set to 1. It is also initialized by a reset.

### FEADDR[31:0] bits (End Address for FACI Command Processing)

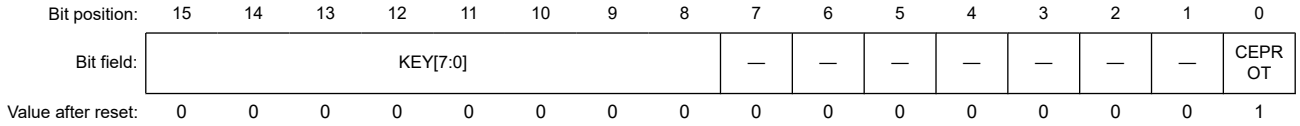
The FEADDR[31:0] bits specify the end address for Multi Block Erase and Blank Check command processing. In command processing, bits 31 to 17 and any bits that do not reach the address boundaries listed in the section 44.4.12. FSADDR : FACI Command Start Address Register are ignored.

For information on the addresses of the flash memory, see section 44.2. Structure of Memory.

## 44.4.14 FMEPROT : Flash P/E Mode Entry Protection Register

Base address: FACI = 0x407F_E000

Offset address: 0x44




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | CEPROT | Code Flash P/E Mode Entry Protection<br>0: FENTRYC bit is not protected<br>1: FENTRYC bit is protected. | R/W*1 *2 *4 |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code | W*3 |

Note 1. Writing to this bit is only possible when the FRDY bit in the FSTATR register is 1. Writing to this bit while the FRDY bit = 0 is ignored.
Note 2. Writing to this bit is only possible when 16 bits are written and the value written to the KEY bits is 0xD9.
Note 3. Written values are not retained by these bits (always read as 0x00).
Note 4. Only secure access can write to this register. Both secure access and non-secure read access are allowed. Non-secure write access is denied, but TrustZone access error is not generated.

### CEPROT bit (Code Flash P/E Mode Entry Protection)

The CEPROT bit specifies the protection setting of the FRNTRYC bit in the FENTRYR register.

[Setting condition]

- 1 being written to the CEPROT bit while writing to FMEPROT is enabled.

[Clearing condition]

- 0 being written to the CEPROT bit while writing to FMEPROT is enabled.

## 44.4.15 FBPROT0 : Flash Block Protection Register

Base address: FACI = 0x407F_E000

Offset address: 0x78

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | BPCN |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BPCN0 | Block Protection for Non-secure Cancel<br>0: Block protection is enabled<br>1: Block protection is disabled. | R/W*1 *2 |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code | W*3 |

Note 1. This bit can be written when the FRDY bit in the FSTATR register is 1. Writing to this bit is ignored when the FRDY bit is 0.
Note 2. Writing to this bit is only possible when 16 bits are written and the value written to the KEY[7:0] bits is 0x78.
Note 3. Written values are not retained by these bits (always read as 0x00).

The FBPROT0 register is used to disable the block protect function for non-secure. When the block protect setting is locked by the permanent block setting, it cannot be disabled by this register.

The FBPROT0 value is initialized when the SUINIT bit in the FSUINITR is set to 1, because the FENTRYR value is initialized to 0x0000. It is also initialized by a reset.

### BPCN0 bit (Block Protection for Non-secure Cancel)

The BPCN0 bit disables the block protect setting for non-secure function.

[Setting condition]

- When the write-enabling conditions are satisfied and the FENTRYR is not 0x0000, write 1 to this bit.

[Clearing conditions]

- 8 bits being written to FBPROT0 while the FRDY bit is 1.
- A value other than 0x78 specified in the KEY bits and 16 bits are written to FBPROT0 while the FRDY bit is 1.
- 0 being written to the BPCN0 bit while writing to FBPROT0 is enabled.
- The FENTRYR register value is 0x0000.

## 44.4.16 FBPROT1 : Flash Block Protection for Secure Register

Base address: FACI = 0x407F_E000

Offset address: 0x7C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | BPCN1 |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BPCN1 | Block Protection for Secure Cancel<br>0: Block protection is enabled<br>1: Block protection is disabled. | R/W*1 *2 |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code | W*3 |

Note 1. Writing to this bit is only possible when the FRDY bit in the FSTATR register is 1. Writing to this bit while FRDY bit = 0 is ignored.
Note 2. Writing to this bit is only possible when 16 bits are written and the value written to the KEY[7:0] bits is 0xB1.
Note 3. Written values are not retained by these bits (always read as 0x00).

The FBPROT1 register is used to disable the block protect function for secure developer. When the block protect setting is locked by the permanent block setting, it cannot be disabled by this register.

The FBPROT1 value is initialized when the SUINIT bit in the FSUINITR is set to 1, because the FENTRYR value is initialized to 0x0000. It is also initialized by a reset.

### BPCN1 bit (Block Protection for Secure Cancel)

The BPCN1 bit disables the block protect setting for secure function.

[Setting condition]

- When the write-enabling conditions are satisfied and the FENTRYR is not 0x0000, write 1 to BPCN1.

[Clearing conditions]

- 8 bits being written to FBPROT1 while the FRDY bit is 1.
- A value other than 0xB1 specified in the KEY bits and 16 bits are written to FBPROT1 while the FRDY bit is 1.
- 0 being written to the BPCN1 bit while writing to FBPROT1 is enabled.
- The FENTRYR register value is 0x0000.

## 44.4.17 FSTATR : Flash Status Register

Base address: FACI = 0x407F_E000

Offset address: 0x80

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | ILGCOMERR | FESETERR | SECERR | OTERR | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | FRDY | ILGLERR | ERSERR | PRGERR | SUSRDY | DBFULL | ERSSPD | PRGSPD | — | FLWERR | — | — | — | — | — | — |
| Value after reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 5:0 | — | These bits are read as 0. The write value should be 0. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 6 | FLWEERR | Flash Write/Erase Protect Error Flag<br>0: An error has not occurred<br>1: An error has occurred. | R |
| 7 | — | These bits are read as 0. The write value should be 0. | R/W |
| 8 | PRGSPD | Programming Suspend Status Flag<br>0: The flash sequencer is not in the programming suspension processing state or programming suspended state<br>1: The flash sequencer is in the programming suspension processing state or programming suspended state. | R |
| 9 | ERSSPD | Erasure Suspend Status Flag<br>0: The flash sequencer is not in the erasure suspension processing state or the erasure suspended state<br>1: The flash sequencer is in the erasure suspension processing state or the erasure suspended state. | R |
| 10 | DBFULL | Data Buffer Full Flag<br>0: The data buffer is empty<br>1: The data buffer is full. | R |
| 11 | SUSRDY | Suspend Ready Flag<br>0: The flash sequencer cannot receive P/E suspend commands<br>1: The flash sequencer can receive P/E suspend commands. | R |
| 12 | PRGERR | Programming Error Flag<br>0: Programming has completed successfully<br>1: An error has occurred during programming. | R |
| 13 | ERSERR | Erasure Error Flag<br>0: Erasure has completed successfully<br>1: An error has occurred during erasure. | R |
| 14 | ILGLERR | Illegal Command Error Flag<br>0: The flash sequencer has not detected an illegal FACI command or illegal flash memory access<br>1: The flash sequencer has detected an illegal FACI command or illegal flash memory access. | R |
| 15 | FRDY | Flash Ready Flag<br>0: Program, Block Erase, Multi Block Erase, P/E suspend, P/E resume, Forced Stop, Blank Check, or Configuration set command processing is in progress<br>1: None of the above is in progress. | R |
| 19:16 | — | These bits are read as 0. The write value should be 0. | R/W |
| 20 | OTERR | Other Error<br>0: A status clear or forced stop command processing is complete<br>1: An error has occurred. | R |
| 21 | SECERR | Security Error<br>0: A status clear or forced stop command processing is complete<br>1: An error has occurred. | R |
| 22 | FESETERR | FENTRY Setting Error<br>0: A status clear or forced stop command processing is complete<br>1: An error has occurred. | R |
| 23 | ILGCOMERR | Illegal Command Error<br>0: A status clear or forced stop command processing is complete<br>1: An error has occurred. | R |
| 31:24 | — | These bits are read as 0. The write value should be 0. | R/W |

The FSTATR register indicates the state of the flash sequencer.

### FLWEERR flag (Flash Write/Erase Protect Error Flag)

The FLWEERR flag indicates a violation of the flash memory overwrite protection setting in the FWEPROR register. When this flag is 1, the flash sequencer is in the command-locked state.

[Setting condition]

- An error has occurred.

[Clearing condition]

- The flash sequencer starts processing the Forced Stop command.

### PRGSPD flag (Programming Suspend Status Flag)

The PRGSPD flag indicates that the flash sequencer is in the programming suspension processing state or programming suspended state.

[Setting condition]

- The flash sequencer starts processing in response to the programming suspend command.

[Clearing conditions]

- Reception of the P/E resume command by the flash sequencer (after write access to the FACI command-issuing area is complete)
- The flash sequencer starts processing the Forced Stop command.

### ERSSPD flag (Erasure Suspend Status Flag)

The ERSSPD flag indicates that the flash sequencer is in the erasure suspension processing state or erasure suspended state.

[Setting condition]

- The flash sequencer starts processing in response to an erasure suspend command.

[Clearing condition]

- Reception of the P/E resume command by the flash sequencer (after write access to the FACI command-issuing area is complete)
- The flash sequencer starts processing of the Forced Stop command.

### DBFULL flag (Data Buffer Full Flag)

The DBFULL flag indicates the state of the data buffer when the program command is issued. The flash sequencer incorporates a buffer for write data (data buffer). When data for writing to the flash memory are written to the FACI command-issuing area while the data buffer is full, the flash sequencer inserts a wait cycle in the peripheral bus.

[Setting condition]

- The data buffer becomes full while program commands are issued.

[Clearing condition]

- The data buffer becomes empty.

### SUSRDY flag (Suspend Ready Flag)

The SUSRDY flag indicates whether the flash sequencer can receive a P/E suspend command.

[Setting condition]

- After starting programming/erasure processing, the flash sequencer enters a state in which P/E suspend commands can be received.

[Clearing conditions]

- Reception of the P/E suspend command or Forced Stop command by the flash sequencer (after write access to the FACI command-issuing area is complete)
- During programming or erasure, the flash sequencer enters the command-locked state
- Programming or erasure has completed.

### PRGERR flag (Programming Error Flag)

The PRGERR flag indicates the result of programming of the flash memory. When this flag is 1, the flash sequencer is in the command-locked state.

[Setting condition]

- An error has occurred during programming.

[Clearing condition]

- The flash sequencer starts processing of the Status Clear or Forced Stop command.

#### ERSERR flag (Erasure Error Flag)

The ERSERR flag indicates the result of erasure of the flash memory. When this flag is 1, the flash sequencer is in the command-locked state.

[Setting condition]

- An error has occurred during erasure.

[Clearing condition]

- The flash sequencer starts processing of the Status Clear or Forced Stop command.

#### ILGLERR flag (Illegal Command Error Flag)

The ILGLERR flag indicates that the flash sequencer has detected an illegal FACI command or flash memory access. If this flag is 1, the flash sequencer is in the command-locked state.

[Setting conditions]

- See section 44.11.2. Error Protection.

[Clearing condition]

- The flash sequencer starts processing of the Status Clear or Forced Stop command.

#### FRDY flag (Flash Ready Flag)

The FRDY flag indicates the command processing state of the flash sequencer.

[Setting conditions]

- The flash sequencer completes command processing
- The flash sequencer receives a P/E suspend command and suspends programming of the flash memory
- The flash sequencer received the Forced Stop command and ended command processing.

[Clearing conditions]

- The flash sequencer received an FACI command
- For Program and Configuration setting, the first write access to the FACI command-issuing area
- For other commands, the last write access to the FACI command-issuing area.

#### OTERR flag (Other Error)

See Table 44.21. When this flag is 1, the flash sequencer is in the command-lock state.

[Setting condition]

- An error has occurred.

[Clearing condition]

- The status clear or forced stop command processing is complete.

#### SECERR flag (Security Error)

See Table 44.21. When this flag is 1, the flash sequencer is in the command-lock state.

[Setting condition]

- An error has occurred.

[Clearing condition]

- The status clear or forced stop command processing is complete.

### FESETERR flag (FENTRY Setting Error)

See Table 44.21. When this flag is 1, the flash sequencer is in the command-lock state.

[Setting condition]

- An error has occurred.

[Clearing condition]

- The status clear or forced stop command processing is complete.

### ILGCOMERR flag (Illegal Command Error)

See Table 44.21. When this flag is 1, the flash sequencer is in the command-lock state.

[Setting condition]

- An error has occurred.

[Clearing condition]

- The status clear or forced stop command processing is complete.

## 44.4.18 FENTRYR : Flash P/E Mode Entry Register

Base address: FACI = 0x407F_E000

Offset address: 0x84

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | FENTRYD | — | — | — | — | — | — | FENTRYC |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | FENTRYC | Code Flash P/E Mode Entry<br>0: Code flash is in read mode<br>1: Code flash is in P/E mode. | R/W*1*2 |
| 6:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 7 | FENTRYD | Data Flash P/E Mode Entry<br>0: Data flash is in read mode<br>1: Data flash is in P/E mode. | R/W*1*2 |
| 15:8 | KEY[7:0] | Key Code | W*3 |

Note 1. These bits can be written when the FRDY bit in the FSTATR register is 1. Writing to these bits are ignored when the FRDY bit is 0.
Note 2. Writing to these bits is only possible when 16 bits are written and the value written to the KEY[7:0] bits is 0xAA.
Note 3. Written values are not retained by these bits (always read 0x00).

FENTRYR is used to specify code flash P/E mode or data flash P/E mode. To specify the code flash P/E mode or data flash P/E mode so that the flash sequencer can receive FACI commands, set either the FENTRYD or FENTRYC bit to 1 to place the flash sequencer in P/E mode.

FENTRYR is initialized when the SUINIT bit in FSUINITR is set to 1. It is also initialized by a reset.

Note: Writing a value of 0XAA81 to this register causes the ILGLERR bit in the FSTATR register to be set to 1, resulting in the flash sequencer being placed in the command-locked state.

### FENTRYC bit (Code Flash P/E Mode Entry)

The FENTRYC bit specifies P/E mode for the code flash memory.

[Setting condition]

- Write 1 to the FENTRYC bit while writing to FENTRYR is enabled and FENTRYR is 0x0000.

[Clearing conditions]

- Write 8 bits to FENTRYR while the FRDY bit is 1
- A value other than 0xAA is specified in the KEY[7:0] bits and 16 bits are written to FENTRYR while the FRDY bit is 1
- Write 0 to the FENTRYC bit while writing to FENTRYR is enabled
- Write to FENTRYR while writing is enabled and its value is other than 0x0000
- The protection of FMEPROT register is enabled.

#### FENTRYD bit (Data Flash P/E Mode Entry)

The FENTRYD bit specifies P/E mode for the data flash memory.

[Setting condition]

- Write 1 to the FENTRYD bit while writing to FENTRYR is enabled and FENTRYR is 0x0000.

[Clearing conditions]

- Write 8 bits to FENTRYR while the FRDY bit is 1
- Writing of 16 bits to FENTRYR with a value other than 0xAA specified for the KEY[7:0] bits while the FRDY bit is 1
- Write 0 to the FENTRYD bit while writing to FENTRYR is enabled
- Write to FENTRYR while writing is enabled and its value is other than 0x0000.

#### KEY[7:0] bits (Key Code)

The KEY[7:0] bits control writing permission to the FENTRYD or FENTRYC bits.

### 44.4.19 FSUINITR : Flash Sequencer Setup Initialization Register

Base address: FACI = 0x407F_E000

Offset address: 0x8C

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | SUINIT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | SUINIT | Set-Up Initialization<br>0: The FSADDR, FEADDR, FBPROT0, FBPROT1, FENTRYR, FBCCNT, and FCPSR flash sequencer setup registers keep their current values<br>1: The FSADDR, FEADDR, FBPROT0, FBRPOT1, FENTRYR, FBCCNT, and FCPSR flash sequencer setup registers are initialized. | R/W*1*2 |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code | W*3 |

Note 1. This bit can be written when the FRDY bit in the FSTATR register is 1. Writing to this bit is ignored when the FRDY bit is 0.
Note 2. Writing to these bits is only possible when 16 bits are written and the value written to the KEY[7:0] bits is 0x2D.
Note 3. Written values are not retained by these bits (always read 0x00).

FSUINITR is used for initialization of the flash sequencer setup.

#### SUINIT bit (Set-Up Initialization)

The SUINIT bit initializes the following flash sequencer setup registers:

- FSADDR
- FEADDR
- FBPROT0
- FBPROT1

- FENTRYR
- FBCCNT
- FCPSR.

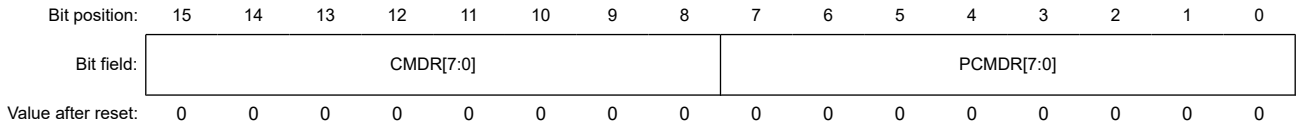
### KEY[7:0] bits (Key Code)

The KEY[7:0] bits control writing permission to the SUINIT bit.

## 44.4.20 FCMDR : FACI Command Register

Base address: FACI = 0x407F_E000

Offset address: 0xA0




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | PCMDR[7:0] | Pre-command Flag<br>The command just before the latest command is stored. | R |
| 15:8 | CMDR[7:0] | Command Flag<br>The latest command is stored. | R |

FCMDR records the two most recent commands accepted by the flash sequencer.

### PCMDR[7:0] bits (Pre-command Flag)

The PCMDR[7:0] bits indicate the command received immediately before the latest command received by the flash sequencer.

### CMDR[7:0] bits (Command Flag)

The CMDR[7:0] bits indicate the latest command received by the flash sequencer.

**Table 44.5 States of FCMDR after receiving commands**

| Command | CMDR | PCMDR |
|---|---|---|
| Program | 0xE8 | Previous command |
| Block erase | 0xD0 | 0x20 |
| Multi block erase | 0xD0 | 0x21 |
| P/E suspend | 0xB0 | Previous command |
| P/E resume | 0xD0 | Previous command |
| Status Clear | 0x50 | Previous command |
| Forced Stop | 0xB3 | Previous command |
| Blank Check | 0xD0 | 0x71 |
| Configuration set | 0x40 | Previous command |

## 44.4.21 FBCCNT : Blank Check Control Register

Base address: FACI = 0x407F_E000

Offset address: 0xD0

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | BCDIR |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BCDIR | Blank Check Direction<br>0: Blank checking is executed from the lower addresses to the higher addresses (incremental mode)<br>1: Blank checking is executed from the higher addresses to the lower addresses (decremental mode). | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

FBCCNT specifies the addressing mode in processing the Blank Check command. FBCCNT is initialized when the SUINIT bit in FSUINITR is set to 1. It is also initialized by a reset.

### BCDIR bit (Blank Check Direction)

The BCDIR bit specifies the addressing mode for Blank Check.

## 44.4.22 FBCSTAT : Blank Check Status Register

Base address: FACI = 0x407F_E000

Offset address: 0xD4

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | BCST |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | BCST | Blank Check Status Flag<br>0: The target area is in the non-programmed state, that is, the area has been erased but has not yet been reprogrammed<br>1: The target area has been programmed with 0s or 1s. | R |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |

FBCSTAT stores the results of checking in response to the Blank Check command.

### BCST flag (Blank Check Status Flag)

The BCST flag indicates the results of checking in response to the Blank Check command.

## 44.4.23 FPSADDR : Data Flash Programming Start Address Register

Base address: FACI = 0x407F_E000

Offset address: 0xD8

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | PSADR[16:0] | | | | | | | | | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 16:0 | PSADR[16:0] | Programmed Area Start Address<br>The address of the first programmed area | R |
| 31:17 | — | These bits are read as 0. The write value should be 0. | R/W |

FPSADDR indicates the address of the first programmed area found in processing of the Blank Check command.

### PSADR[16:0] bits (Programmed Area Start Address)

The PSADR[16:0] bits indicate the address of the first programmed area found in processing of the Blank Check command. The address is an offset from the address where the data flash memory starts. The setting of these bits is only valid if the BCST bit in the FBCSTAT register is 1 and while the FRDY bit in the FSTATR register is 1. When the BCST bit in the FBCSTAT register is 0, the PSADR[16:0] bits hold the address produced by the previous check.

## 44.4.24 FSUASMON : Flash Startup Area Select Monitor Register

Base address: FACI = 0x407F_E000

Offset address: 0xDC

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | BTFLG | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0/1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | FSPR | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0/1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14:0 | — | These bits are read as 0. The write value should be 0. | R |
| 15 | FSPR | Protection Programming Flag to set Boot Flag and Startup Area Control<br>0: Protected state<br>1: Non-protected state. | R |
| 30:16 | — | These bits are read as 0. The write value should be 0. | R |
| 31 | BTFLG | Flag of Startup Area Select for Boot Swap<br>0: The startup area is the alternate block (block 1)<br>1: The startup area is the default block (block 0). | R |

### FSPR bit (Protection Programming Flag to set Boot Flag and Startup Area Control)

The FSPR bit indicates the protection state against the configuration set command for the BTFLG bit, and FSUACR Register.

In response to a reset or configuration set command, the FACI transfers data from flash memory to this register.

### BTFLG bit (Flag of Startup Area Select for Boot Swap)

The BTFLG bit indicates whether the address of the startup area is exchanged for the boot swap function or not.

In response to a reset or configuration set command, the FACI transfers data from flash memory to this register.

## 44.4.25 FCPSR : Flash Sequencer Processing Switching Register

Base address: FACI = 0x407F_E000

Offset address: 0xE0

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | ESUSPMD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | ESUSPMD | Erasure Suspend Mode<br>0: Suspension priority mode<br>1: Erasure priority mode. | R/W |
| 15:1 | — | These bits are read as 0. The write value should be 0. | R/W |

FCPSR selects the erasure suspension mode. FCPSR is initialized when the SUINIT bit in FSUINITR is set to 1. It is also initialized by a reset.

### ESUSPMD bit (Erasure Suspend Mode)

The ESUSPMD bit selects the erasure suspension mode when a P/E suspend command is issued while the flash sequencer is executing erasure processing (see section 44.9.3.10. P/E Suspend Command). This bit should be set before issuing Block Erase or Multi Block Erase command.

## 44.4.26 FPCKAR : Flash Sequencer Processing Clock Notification Register

Base address: FACI = 0x407F_E000

Offset address: 0xE4

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | PCKA[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | PCKA[7:0] | Flash Sequencer Operating Clock Notification<br>These bits are used to set the operating frequency of the flash sequencer while processing FACI commands. | R/W*1*2 |
| 15:8 | KEY[7:0] | Key Code | W*3 |

Note 1. This bit can be written when the FRDY bit in the FSTATR register is 1. Writing to this bit is ignored when the FRDY bit is 0.
Note 2. Writing to these bits is only possible when 16 bits are written and the value written to the KEY[7:0] bits is 0x1E.
Note 3. Written values are not retained by these bits (always read 0x00).

FPCKAR specifies the operating frequency of the flash sequencer while processing FACI commands. The highest operating frequency for the given product is set as the initial value.

### PCKA[7:0] bits (Flash Sequencer Operating Clock Notification)

The PCKA[7:0] bits specify the operating frequency of the flash sequencer while processing FACI commands. Set the desired frequency for these bits before issuing an FACI command. Specifically, convert the frequency in MHz to a binary number and set it for these bits.

Example:

Frequency is 35.9 MHz (PCKA = 0x24)

Round up the first decimal place of 35.9 MHz to a whole number (= 36) and convert it into a binary number.

If the value set in these bits is smaller than the actual operating frequency of the flash sequencer, the flash memory programming/erasure characteristics cannot be guaranteed. If the value set in these bits is greater than the actual operating frequency of the flash sequencer, the flash memory programming/erasure characteristics can be guaranteed but the FACI

command processing time such as the time programming/erasure takes will increase. The minimum FACI command processing time is obtained when the operating frequency of the flash sequencer is the same as the PCKA value.

**KEY[7:0] bits (Key Code)**

The KEY[7:0] bits control writing permission to the PCKA bit.

## 44.4.27 FSUACR : Flash Startup Area Control Register

Base address: FACI = 0x407F_E000

Offset address: 0xE8

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | SAS[1:0] | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1:0 | SAS[1:0] | Startup Area Select<br>0 0: Startup area is selected by BTFLG bit<br>0 1: Startup area is selected by BTFLG bit<br>1 0: Startup area is temporarily switched to the default area (block 0)<br>1 1: Startup area is temporarily switched to the alternate area (block 1). | R/W*1 *3 |
| 7:2 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | Key Code | W*2 |

Note 1. Following described the write condition of thisese bits (these conditions are required at the same time).
1. Access size to this register is 16 bits
2. The value of KEY[7:0] is 0x66
3. The FSPR bit is 1.

Note 2. Written values are not retained by these bits (always read 0x00).

Note 3. Only secure access can write to this register. Both secure access and non-secure read access are allowed. Non-secure write access is denied, but TrustZone access error is not generated.

FSUACR sets the startup area for the boot swap function.

**SAS[1:0] bits (Startup Area Select)**

The SAS[1:0] bits select the startup area. Three methods are available for changing the startup area.

**KEY[7:0] bits (Key Code)**

The KEY[7:0] bits control writing permission to the SAS [1:0] bits.

## 44.4.28 FCKMHZ : Data Flash Access Frequency Register

Base address: FLAD = 0x407F_C000

Offset address: 0x40

| Bit position: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit field: | FCKMHZ[7:0] | | | | | | | |
| Value after reset: | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 7:0 | FCKMHZ[7:0] | Data Flash Access Frequency Register<br>These bits optimize the speed of reading the data flash memory. | R/W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

This register optimizes the speed of reading the data flash memory.

Set the frequency of the peripheral module clock (FCLK) of internal peripheral bus which is the clock for access to the data flash memory, in MHz units. For example, 35.9 MHz should be rounded up and set the frequency to 36. Number of cycles required for access to the data flash memory are inserted according to the frequency. When changing the frequency of the FCLK, follow the procedure below to modify the value of the data flash access frequency register (FCKMHZ) in either of the following ways according to whether operation is at a lower frequency before or after the change.

- When changing the speed from low to high: Modify FCKMHZ. After confirming the change by reading FCKMHZ, change the frequency.
- When changing the speed from high to low: Change the frequency. After the frequency is changed, modify FCKMHZ.

## 44.5 Flash Cache

### 44.5.1 Feature of flash cache

The FCACHE (Flash Cache) speeds up read access from bus master to the flash memory. The FCACHE includes:

- FCACHE1, for CPU instruction fetches
- FCACHE2, for CPU operand access
- FLPF, for the prefetch access in CPU instruction fetches

**Table 44.6 Flash Cache 1 (FCACHE1) overview**

| | |
|---|---|
| Cache Target Region | 0x0000_0000 - 0x007F_FFFF |
| Target Bus Master | CPU instruction Fetch |
| Capacity | 256 Bytes |
| Associativity | 8WAY set associative |
| | 128 bits/entry (128 bit aligned data), 2 entries/way |
| Access Cycle | Cache Hit : 0 wait<br>Cache Miss : wait number of Flash Wait Cycle Register |

**Table 44.7 Flash Cache 2 (FCACHE2) overview**

| | |
|---|---|
| Cache Target Region | 0x0000_0000 - 0x007F_FFFF |
| Target Bus Master | CPU Operand Access |
| Capacity | 16 Bytes |
| Associativity | Full Associative |
| | 128 bits/entry (128 bit aligned data), 1 entry |
| Access Cycle | Cache Hit : 0 wait<br>Cache Miss : wait number of Flash Wait Cycle Register |

**Table 44.8 Prefetch Buffer (FLPF) overview**

| | |
|---|---|
| Cache Target Region | 0x0000_0000 - 0x007F_FFFF |
| Capacity | 32 Bytes |
| Associativity | Full Associative |
| | 128 bits/entry (128 bit aligned data), 2 entries |
| Request Address | Next address of previous CPU Instruction |
| Access Cycle | Cache Hit : 0 wait<br>Cache Miss : wait number of Flash Wait Cycle Register |

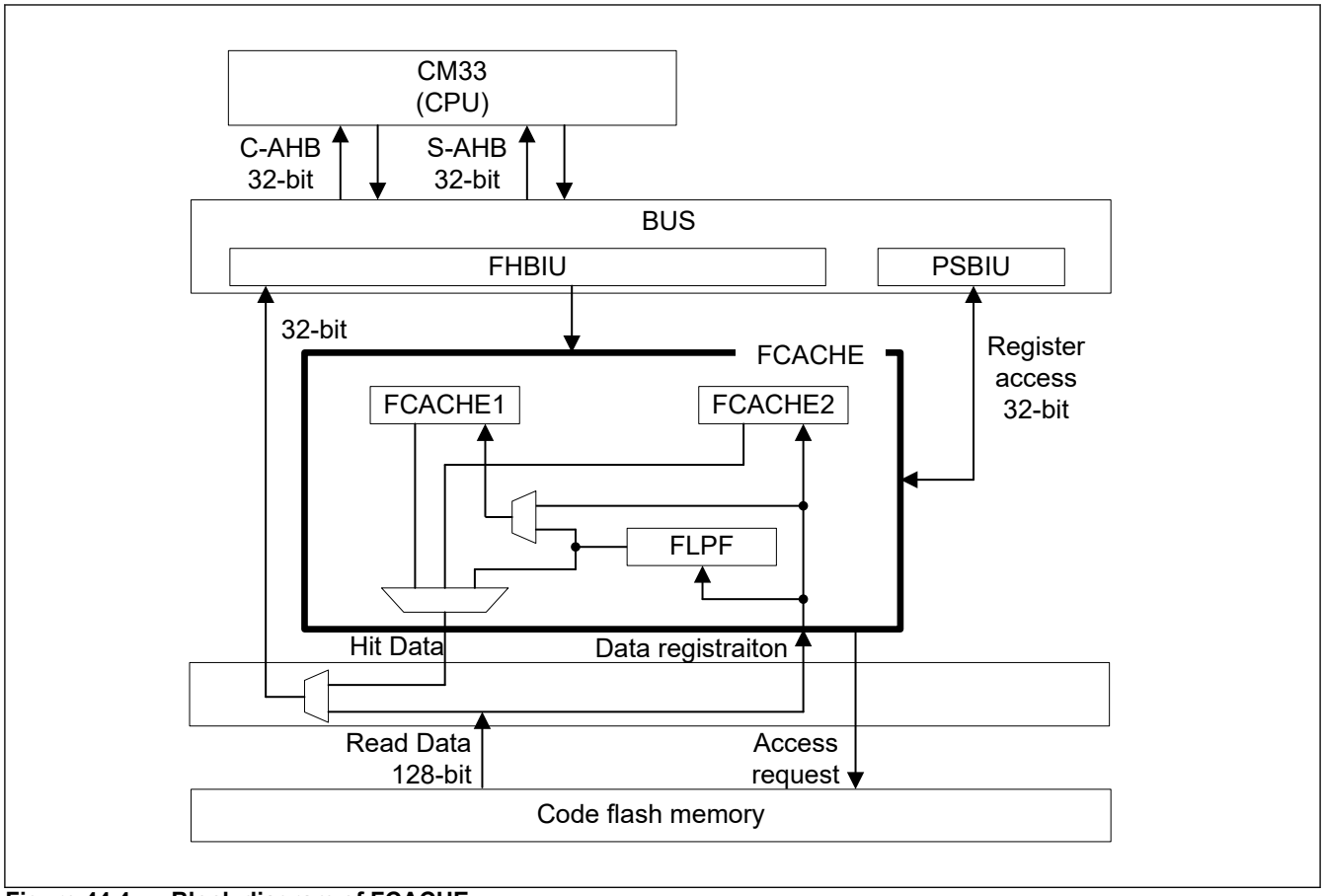



**Figure 44.4 Block diagram of FCACHE**

## 44.6 Operating Modes Associated with Flash Memory

Figure 44.5 is a diagram of the mode transitions associated with the flash memory. For the procedures for setting the modes, see section 6, Option-Setting Memory.

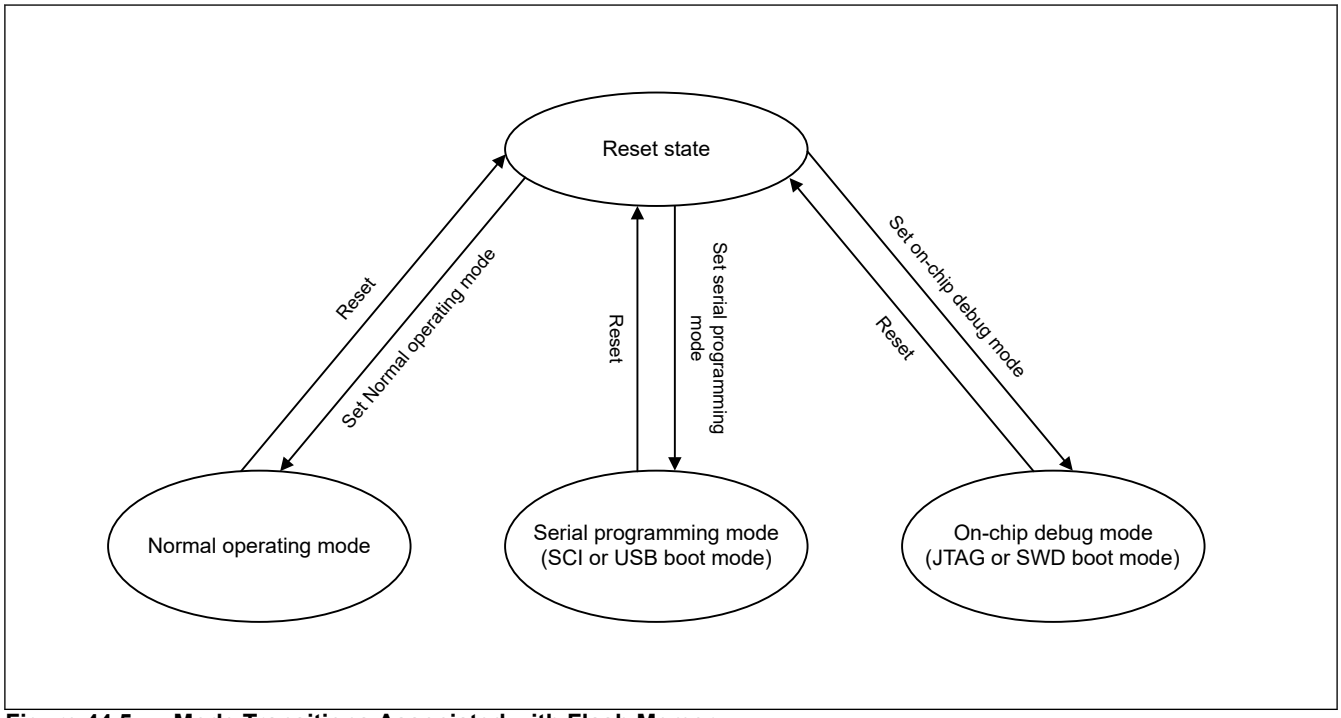



**Figure 44.5 Mode Transitions Associated with Flash Memory**

The flash memory area where programming and erasure are permitted and the boot program after a reset are different according to each mode. The differences between modes are listed in Table 44.9.

**Table 44.9 Differences between Modes**

| Parameter | Normal operating mode | Serial programming mode (SCI or USB boot mode) | On-chip debug mode (JTAG or SWD boot mode) |
|---|---|---|---|
| Programmable and erasable areas | • Code flash memory<br>• Data flash memory<br>• Option setting memory (programming only) | • Code flash memory<br>• Data flash memory<br>• Option-setting memory (programming only) | • Code flash memory<br>• Data flash memory<br>• Option setting memory (programming only) |
| Erasure in block units | Possible | Possible | Possible |
| Boot program at a reset | User area program | Embedded program for serial programming | Depends on debug command |

## 44.7 Overview of Functions

By using a dedicated flash-memory programmer to program the flash memory through a serial interface (serial programming) or JTAG/SWD interface (on-chip debug mode), the device can be rewritten regardless of whether this is before or after it is mounted on the target system.

Furthermore, security functions to prohibit rewriting or reading of the user program written to the flash memory are incorporated, and this can prevent falsification and illicit reading of the programs by third parties.

Programming by the user program (self-programming) is available to suit applications where the application on the target system may require updating after manufacturing or shipment. Protection features for the safe rewriting of the flash memory are also incorporated. Furthermore, interrupt processing during self-programming is supported, so programming can proceed at the same time as processing for external communications, etc., and this is the case in various situations. Table 44.10 lists the overview of the methods of programming and the corresponding operating modes.

**Table 44.10 Programming methods**

| Programming method | Functional overview | Operating mode |
|---|---|---|
| Serial programming | A dedicated flash-memory programmer through the SCI or USBFS interface enables on-board programming of the flash memory after the device is mounted on the target system. | Serial programming mode |
| | A dedicated flash-memory programmer through the SCI or USBFS interface and a dedicated programming adapter board allow off-board programming of the flash memory, for example, programming of the device before it is mounted on the target system. | |
| Self-programming | A user program written to memory in advance of serial programming execution can also program the flash memory. The background operation capability makes it possible to fetch instructions or otherwise read data from the code flash memory while the data flash memory is programmed. As a result, a program resident in the code flash memory is able to program the data flash memory.<br>For background operations that are not possible, instructions in the code flash memory cannot be fetched and data cannot be accessed while the code flash memory is being programmed by self-programming. In such cases, a program for programming from the internal SRAM must be transferred in advance and executed. | Normal operating mode |
| JTAG or SWD programming | A dedicated flash-memory programmer or an on-chip debugger through JTAG or SWD enables on-board programming of the flash memory after the device is mounted on the target system.<br>A dedicated flash-memory programmer or an on-chip debugger through JTAG or SWD and a dedicated programming adapter board allow off-board programming of the flash memory, for example, programming of the device before it is mounted on the target system. | On-chip debug mode |

Table 44.11 lists the functions of the flash memory. Serial programmer commands realize each function of serial programming, while reading of the flash memory by an FACI command or the user program realizes each function of self-programming.

**Table 44.11 Basic Functions**

| Function | Functional overview | Availability | |
|---|---|---|---|
| | | Serial programming | Self-programming |
| Blank check | Checks a specified block to ensure that writing to it has not already proceeded. Results of reading from data flash memory to which nothing is written after erasure are not guaranteed, so use blank checking to confirm that writing to memory has not proceeded after erasure. | Not supported | Supported (data flash programming only) |
| Block erasure | Erases the memory contents in the specified block | Supported | Supported |
| Programming | Writes to the specified address | Supported | Supported |
| CRC | Calculates the CRC in the specified range of the flash memory and transfers it to the flash programmer | Supported | Non supported |
| Read | Reads data programmed in the flash memory | Supported | Not supported (read by user program is possible) |
| Start-up program protection functions | Configures the start-up program protection functions | Supported | Supported |
| Option function selection | Selects the option function, and modifies the initial setting of this MCU | Supported | Supported |
| Block protection | Setting block protection | Supported | Supported |
| Device lifecycle transition | Transitions the device lifecycle | Supported | Not supported |
| Memory security attribution | Setting the memory security attribution | Supported | Not supported |
| Key | Injects key | Supported | Supported (except the key related to device lifecycle transition) |
| All erasure | Erase the flash memory to the state after shipment | Supported | Not supported |

The flash memory supports various security functions.

Table 44.12 lists the security functions supported by the flash memory.

**Table 44.12 Lists of Security Functions**

| Function | Description |
|---|---|
| Security flag for Start-up area select | Start-up area selection can be protected by setting of security flag (FSPR). |
| Permanently block protection | Programming or erasure of each block of code flash memory can be protected permanently. |
| Protection for TrustZone | Programming or erasure area, readable area, register access, and FACI command operation are protected by ARM TrustZone security. |
| Programming or erasure mode protection | Only secure developer can enter the programming or erasure mode for code flash. |

## 44.8 Operating Modes of the Flash Sequencer

The flash sequencer has three operating modes as shown in Figure 44.6. Transitions between modes are initiated by changing the value of the FENTRYR register.

When the value of the FENTRYR register is 0x0000, the flash sequencer is in read mode. In this mode, it does not receive FACI commands. The code flash memory and data flash memory are both readable.

When the value of the FENTRYR register is 0x0001, the flash sequencer is in code flash P/E mode where the code flash memory can be programmed or erased by FACI commands. In this mode, the data flash memory is readable.

When the value of the FENTRYR register is 0x0080, the flash sequencer is in data flash P/E mode where the data flash memory can be programmed or erased by FACI commands. In this mode, the data flash memory is not readable. However, the code flash memory is readable.

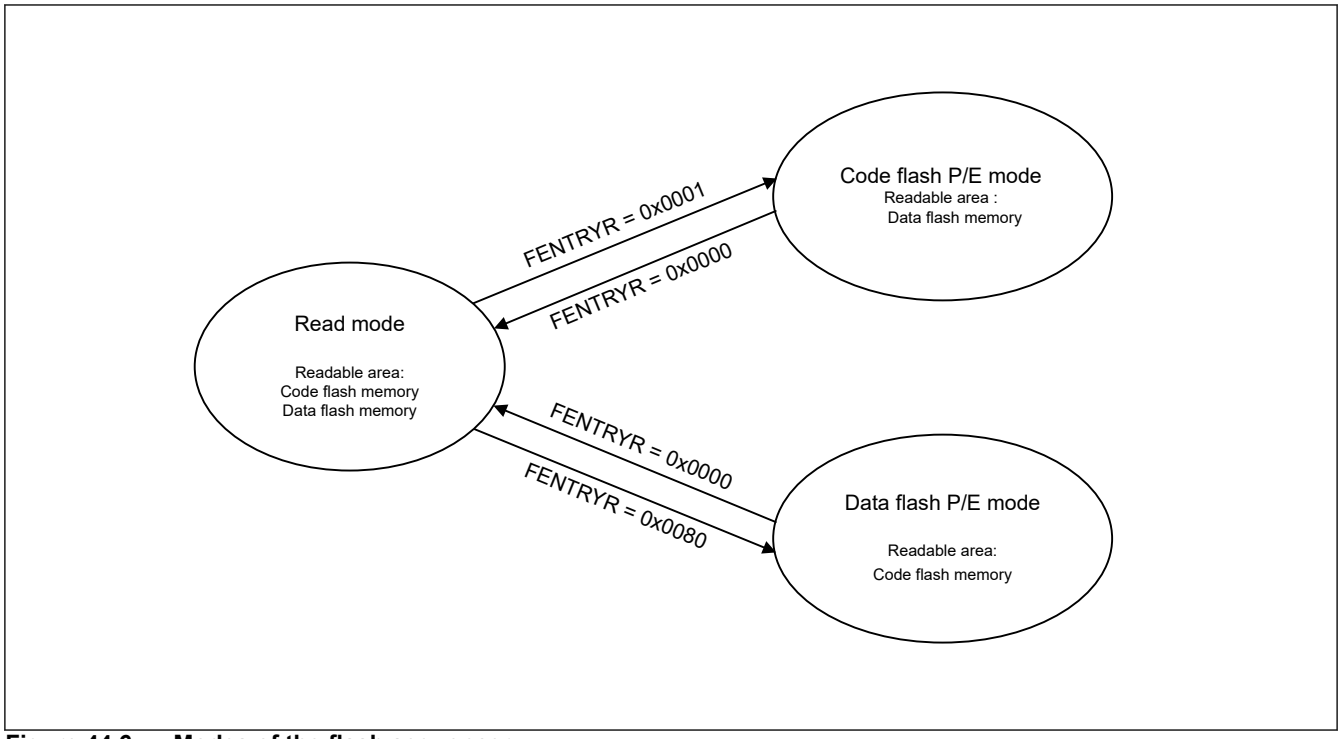



**Figure 44.6 Modes of the flash sequencer**

## 44.9 FACI Commands

### 44.9.1 List of FACI Commands

The FACI controls the FCU according to the specified FACI commands.

This section describes information about the FACI commands and Table 44.13 lists the FACI commands.

**Table 44.13 FACI commands**

| FACI command | Function |
|---|---|
| Program | Programs the user area and data area.<br>Units of programming are 128 bytes for the user area and 4, 8, or 16 bytes for the data area. |
| Block erase | Erases user area and data area.<br>The erase unit is 8 KB or 32 KB for user area, and 64 bytes for data flash. |
| Multi block erase | Erases data area.<br>The erase unit is 64, 128, 256 bytes for data flash. |
| P/E suspend | Suspends programming or erasure processing. |
| P/E resume | Resumes suspended programming or erasure processing. |
| Status clear | Initializes the ILGLERR, ERSERR, PRGERR, ILGCOMERR, FESETERR, SECERR, and OTERR bits in the FSTATR register and the CMDLK, CFAE, and DFAE bits in the FASTAT register, and the flash sequencer released from command-locked state. |
| Forced stop | Forcibly stops processing of FACI commands and initializes the FSTATR and FASTAT registers. |
| Blank check | Checks if data areas are blank.<br>Units of Blank Check: 4 bytes to data flash memory capacity (specified in 4-byte units). |
| Configuration set | Sets the option-setting memory.<br>Units of setting: 16 bytes. |

The FACI commands are issued by writing to the FACI command-issuing area (see Table 44.3). When write access as shown in Table 44.14 proceeds in the specified state, the flash sequencer executes the processing associated with the given command (see section 44.9.2. Relationship between the Flash Sequencer State and FACI Commands).

**Table 44.14 FACI command formats**

| FACI commands | Number of write access | Write data to the FACI command-issuing area | | | |
|---|---|---|---|---|---|
| | | 1st access | 2nd access | 3rd to (N+2)th access | (N+3)th access |
| Program (user area)<br>N = 64 | 67 | 0xE8 | 0x40 (=N) | WD1 to WD64 | 0xD0 |
| Program (data area)<br>4-byte programming: N = 2<br>8-byte programming: N = 4<br>16-byte programming: N = 8 | N+3 | 0xE8 | 0x02 (=N)<br>0x04 (=N)<br>0x08 (=N) | WD1 to WDN | 0xD0 |
| Block Erase<br>(user area 8K/32K Bytes) | 2 | 0x20 | 0xD0 | — | — |
| Block Erase<br>(data area 64 bytes) | 2 | 0x20 | 0xD0 | — | — |
| Multi block erase<br>(data area 64/128/256 bytes) | 2 | 0x21 | 0xD0 | — | — |
| P/E suspend | 1 | 0xB0 | — | — | — |
| P/E resume | 1 | 0xD0 | — | — | — |
| Status Clear | 1 | 0x50 | — | — | — |
| Forced Stop | 1 | 0xB3 | — | — | — |
| Blank Check | 2 | 0x71 | 0xD0 | — | — |
| Configuration set<br>N = 8 | 11 | 0x40 | 0x08 (=N) | WD1 to WD8 | 0xD0 |

Note: WDN (N = 1, 2, …): Nth 16-bit data to be programmed.

The flash sequencer clears the FSTATR.FRDY bit to 0 at the start of a command processing other than the Status Clear command, and sets this bit to 1 on completion.

If the FRDYIE.FRDYIE bit setting is 1, a flash ready (FRDY) interrupt is generated when the FSTATR.FRDY bit is set to 1.

## 44.9.2 Relationship between the Flash Sequencer State and FACI Commands

The FACI commands are accepted according to the mode/state of the flash sequencer. FACI commands should be issued after transitioning of the flash sequencer to the code flash P/E mode or data flash P/E mode and after checking the state of the flash sequencer.

Use the FSTATR and FASTAT registers to check the state of the flash sequencer. In addition, the occurrence of errors in general can be checked by reading the CMDLK bit in the FASTAT register. The value of the CMDLK bit is the logical OR of the following bits in the FSTATR register:

- ILGLERR
- ILGCOMERR
- FESETERR
- SECERR
- OTERR
- ERSERR
- PRGERR
- FLWEERR.

Table 44.15 lists the available FACI commands in each operating mode.

**Table 44.15 Operating mode and available FACI commands**

| Operating mode | FENTRYR | Available FACI commands |
|---|---|---|
| Read mode | 0x0000 | None |
| Code flash P/E mode | 0x0001 | Program<br>Block erase<br>P/E suspend<br>P/E resume<br>Status Clear<br>Forced Stop<br>Configuration set |
| Data flash P/E mode | 0x0080 | Program<br>Block erase<br>Multi block erase<br>P/E suspend<br>P/E resume<br>Status Clear<br>Forced Stop<br>Blank Check |

Table 44.16 shows the state of the flash sequencer and acceptable FACI commands. An appropriate mode is assumed to have been set before the commands are executed.

**Table 44.16 Acceptable FACI commands and state of the flash sequencer**

| | Program, block erase or multi block erase command processing | Configuration set command processing | Program, block erase or multi block erase command suspension processing | Blank check command processing | Programming suspended | Erasure suspended | Programming while erasure is suspended | Command-locked state (FRDY = 1) | Command-locked state (FRDY = 0) | Processing of forced stop command | Other state |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FRDY bit | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| SUSRDY bit | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ERSSPD bit | 0 | 0 | 0/1 | 0/1 | 0 | 1 | 1 | 0/1 | 0/1 | 0 | 0 |
| PRGSPD bit | 0 | 0 | 0/1 | 0/1 | 1 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |
| CMDLK bit | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| Program | X | X*4 | X | X | X | O*3 | X | X | X | X | O |
| Block erase or multi block erase | X | X*4 | X | X | X | X | X | X | X | X | O |
| P/E suspend | O | X*4 | X | X | X | X | X | — | X | X | — |
| P/E resume | X | X*4 | X | X | O | O | X | X | X | X | X |
| Status clear | X | X*4 | X | X | O | O | X | O | X | X | O |
| Forced stop | O | O*4 | O | O | O | O | O | O | O | O | O |
| Blank check | X | X*4 | X | X | O*1 | O*1 | X | X | X | X | O*1 |
| Configuration set | X | X*4 | X | X | X | X | X | X | X | X | O*2 |

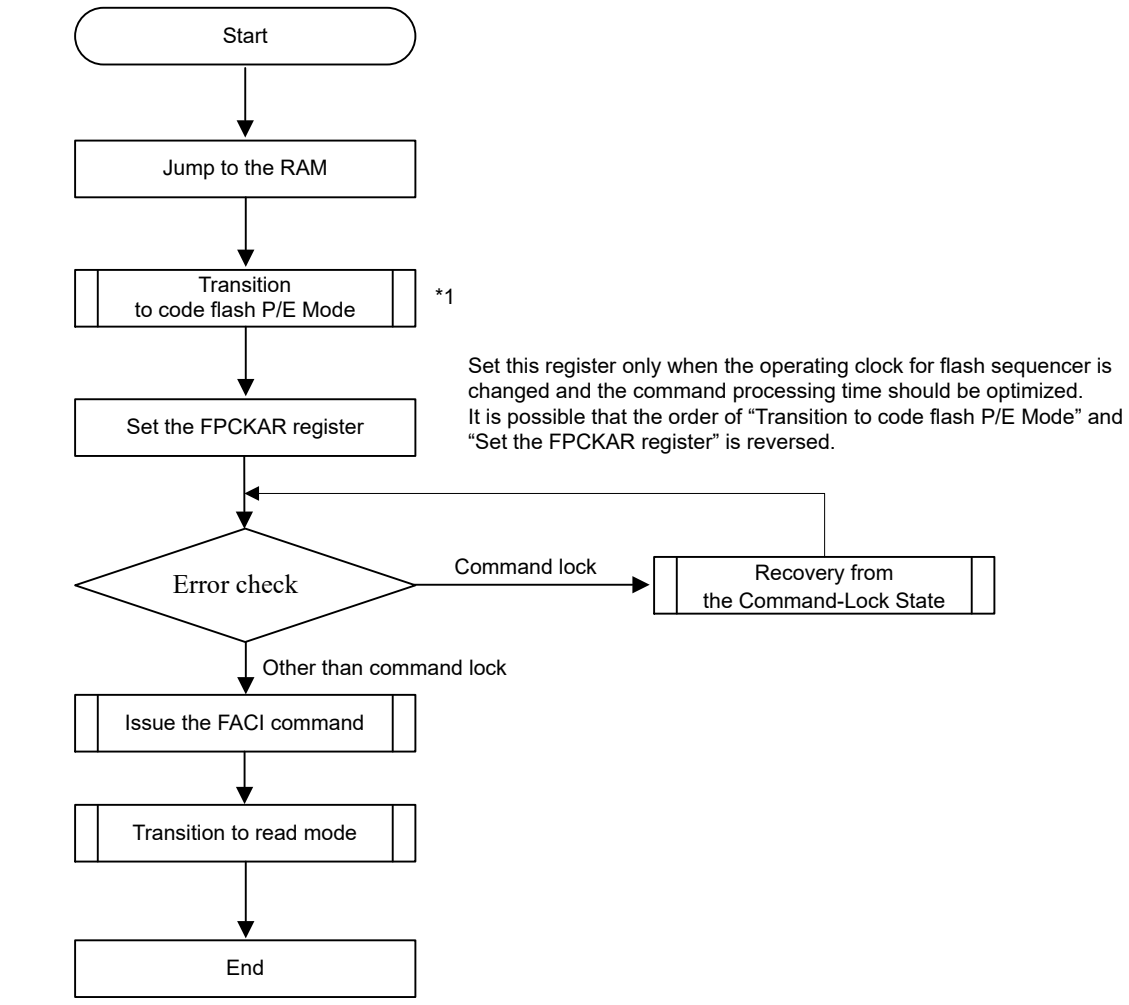
Note: O: Acceptable
X: Not acceptable (places the sequencer in the command-locked state)

—: Ignored
Note 1. Only acceptable in data flash P/E mode.
Note 2. Only acceptable in code flash P/E mode
Note 3. Acceptable when programming area is other than erase suspending block.
Note 4. When configuration set is processing and when FSTATR.DBFULL bit is 1, do not issue this command.

## 44.9.3 Usage of FACI Commands

### 44.9.3.1 Overview of Command Usage in Code Flash P/E Mode

Figure 44.7 show an overview of FACI command usage in code flash P/E mode. For the available commands in code flash P/E mode, see Table 44.15.

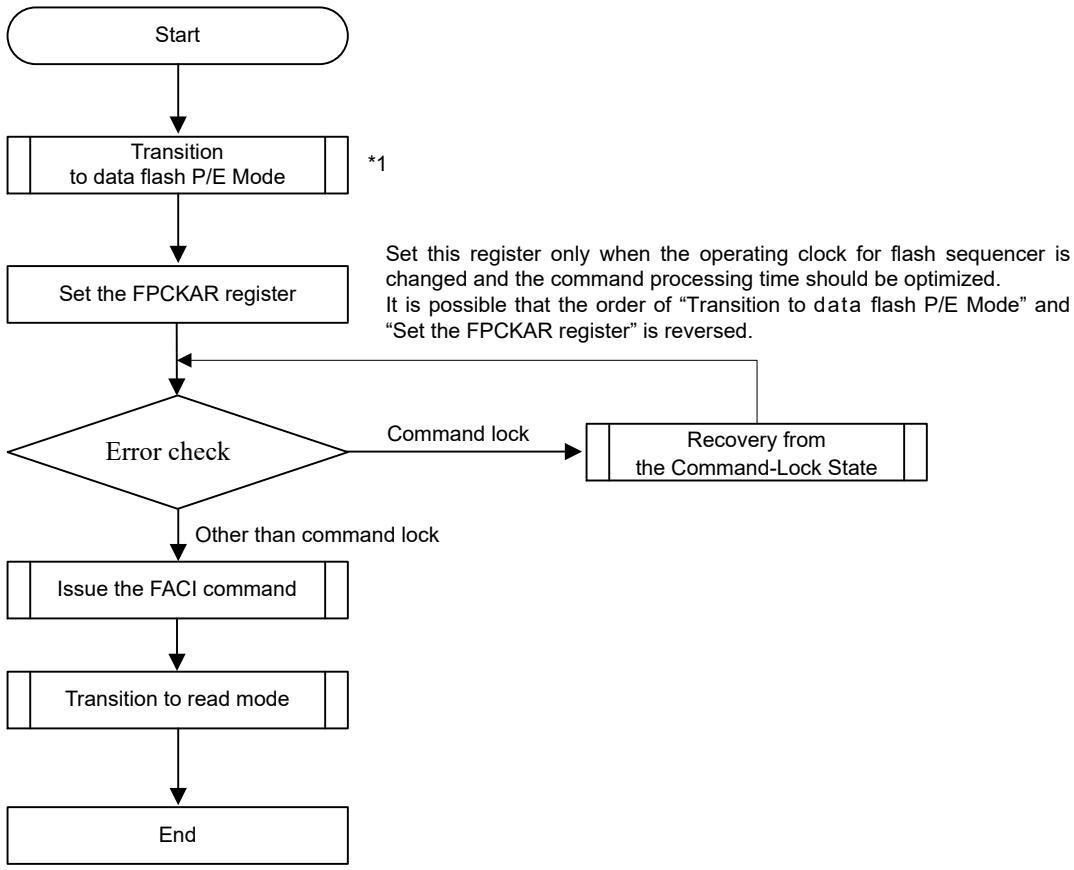



Note 1. See section 44.9.3.3. Transition to Code Flash P/E Mode

**Figure 44.7 Overview of command usage in code flash P/E mode**

### 44.9.3.2 Overview of Command Usage in Data Flash P/E Mode

Figure 44.8 shows an overview of FACI command usage in data flash P/E and Table 44.15 lists the available commands in data flash P/E mode.




Note 1. See section 44.9.3.4. Transition to Data Flash P/E Mode

**Figure 44.8 Overview of command usage in data flash P/E mode**

### 44.9.3.3 Transition to Code Flash P/E Mode

To issue FACI commands for the code flash memory, a transition to code flash P/E mode is required by setting the FENTRYC bit in the FENTRYR register to 1.

Figure 44.9 shows the procedure to transition to code flash P/E mode.

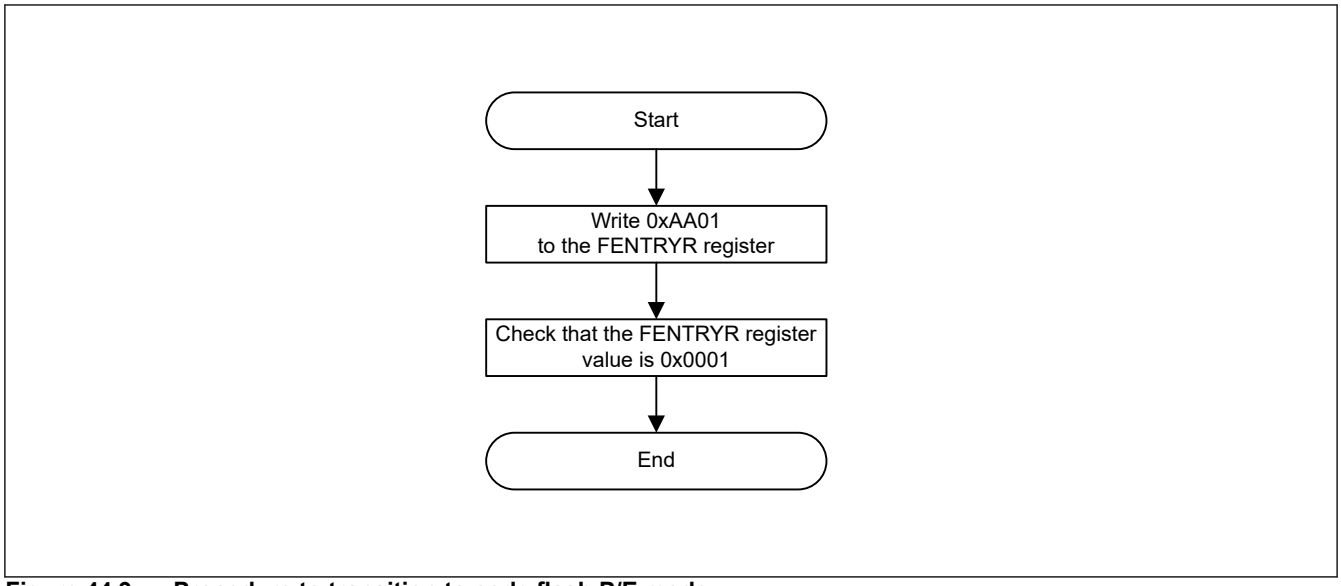



**Figure 44.9 Procedure to transition to code flash P/E mode**

### 44.9.3.4 Transition to Data Flash P/E Mode

To issue FACI commands for the data flash memory, a transition to data flash P/E mode is required by setting the FENTRYD bit in the FENTRYR register to 1.

Figure 44.10 shows the procedure to transition to data flash P/E mode.

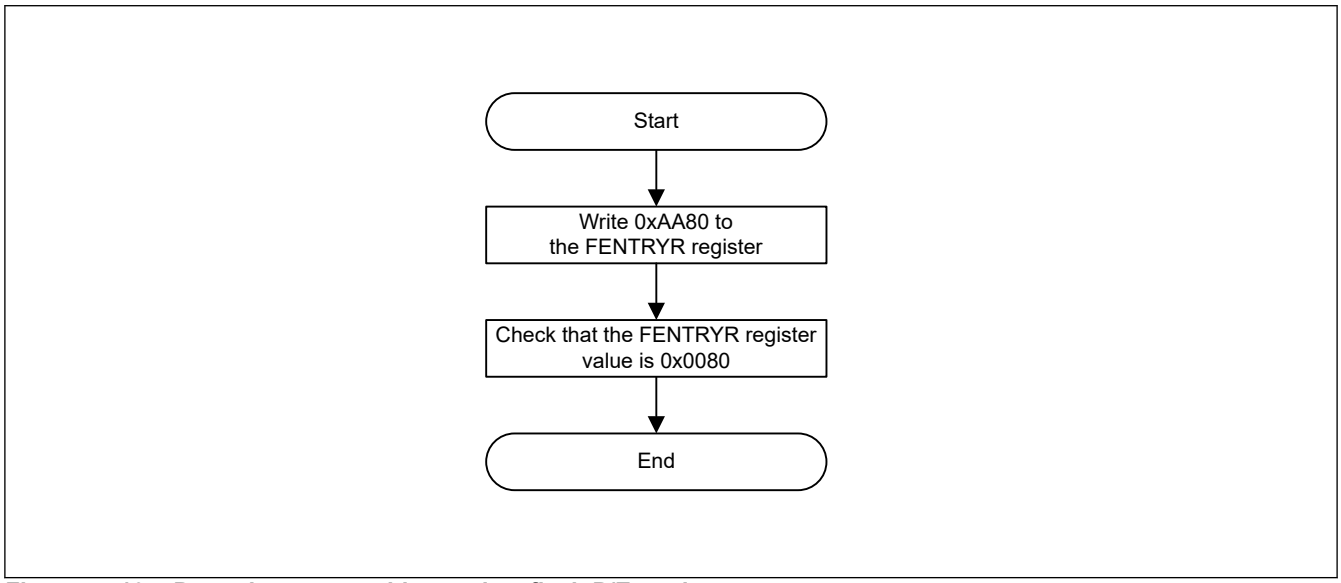
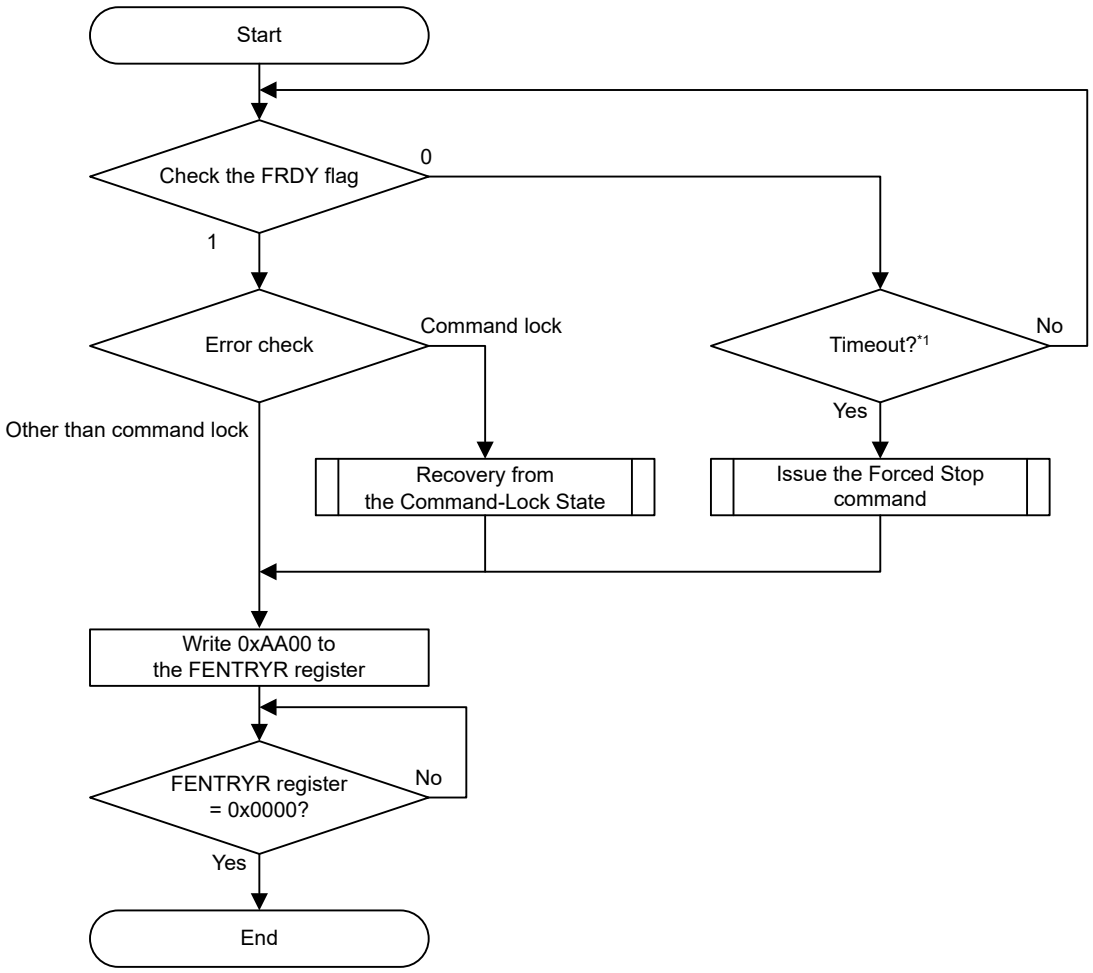



**Figure 44.10 Procedure to transition to data flash P/E mode**

### 44.9.3.5 Transition to Read Mode

To read the flash memory, a transition to read mode is required by setting the FENTRYR register to 0x0000. The transition to read mode should be made after the flash sequencer completes the processing and while operation is not in the command-locked state.

Figure 44.11 shows the procedure to transition to read mode.




Note 1. Judgment of the timeout is based on 1.1 times the maximum time for processing of the FACI command that is in progress (see section 47, Electrical Characteristics)

**Figure 44.11 Procedure to transition to read mode**

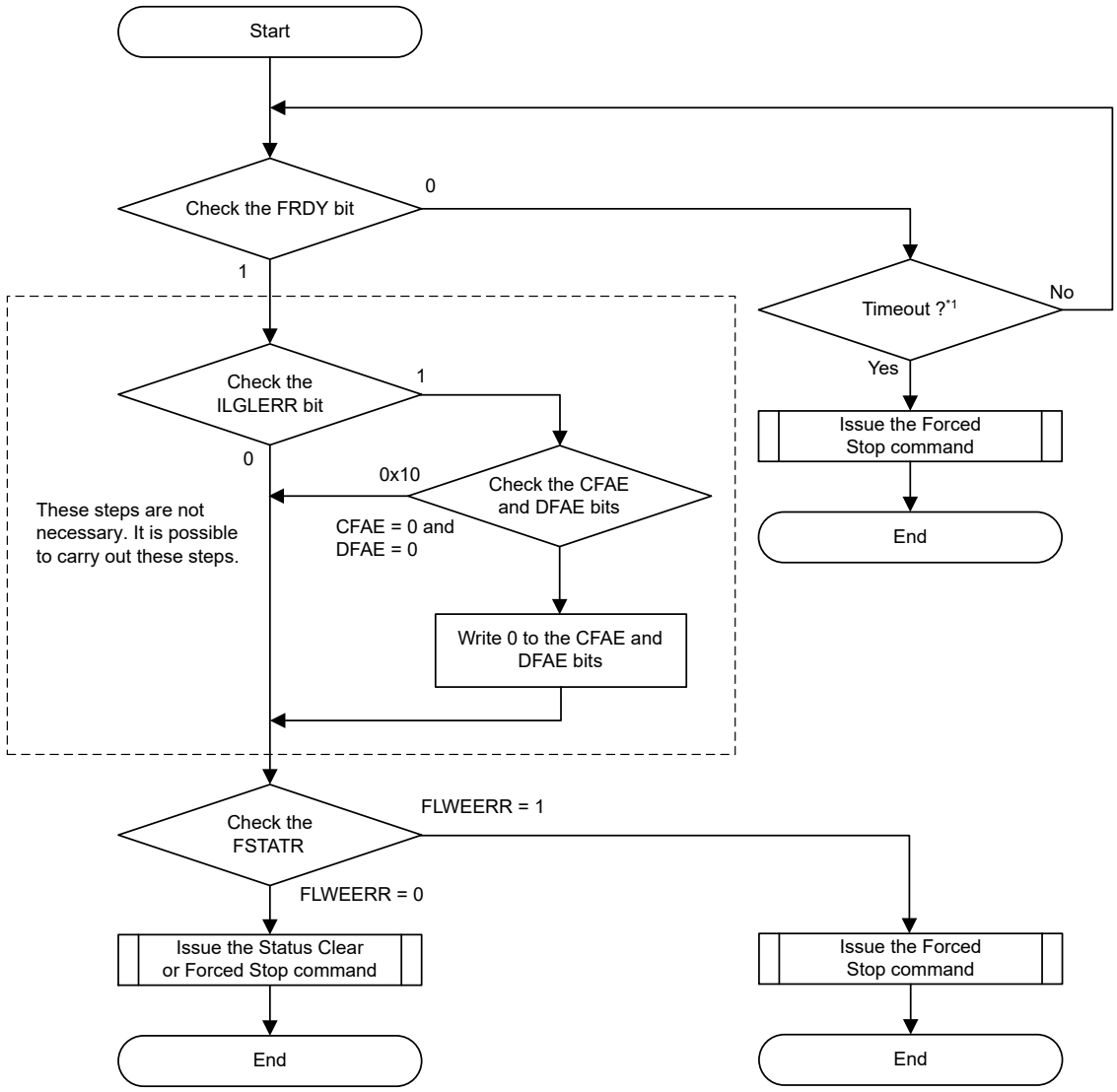
### 44.9.3.6 Recovery from the Command-Locked State

When the flash sequencer enters the command-locked state, FACI commands cannot be accepted. To release the sequencer from the command-locked state, use the status clear command, forced stop command, or FASTAT register.

When the command-locked state is detected by checking for an error before issuing the P/E suspend command, the FRDY bit in the FSTATR register might be 0 even though command processing has not completed. If processing is not complete by the maximum programming/erasure time specified in the electrical characteristics, this is a timeout and the flash sequencer must be stopped with the forced stop command.

The FLWEERR bit in the FSTATR register does not change from 1 to 0 with the status clear command. When these bits are set to 1, use the forced stop command to release from the command-locked state. Bits other than FRDY and FLWEERR in FSTATR register that indicate the command-locked state can be changed from 1 to 0 with the status clear or forced stop command.

Figure 44.12 shows the recovery flow from the command-locked state.




Note 1. Judgment of the timeout is based on 1.1 times the maximum time for processing of the FACI command that is in progress (see section 47, Electrical Characteristics)

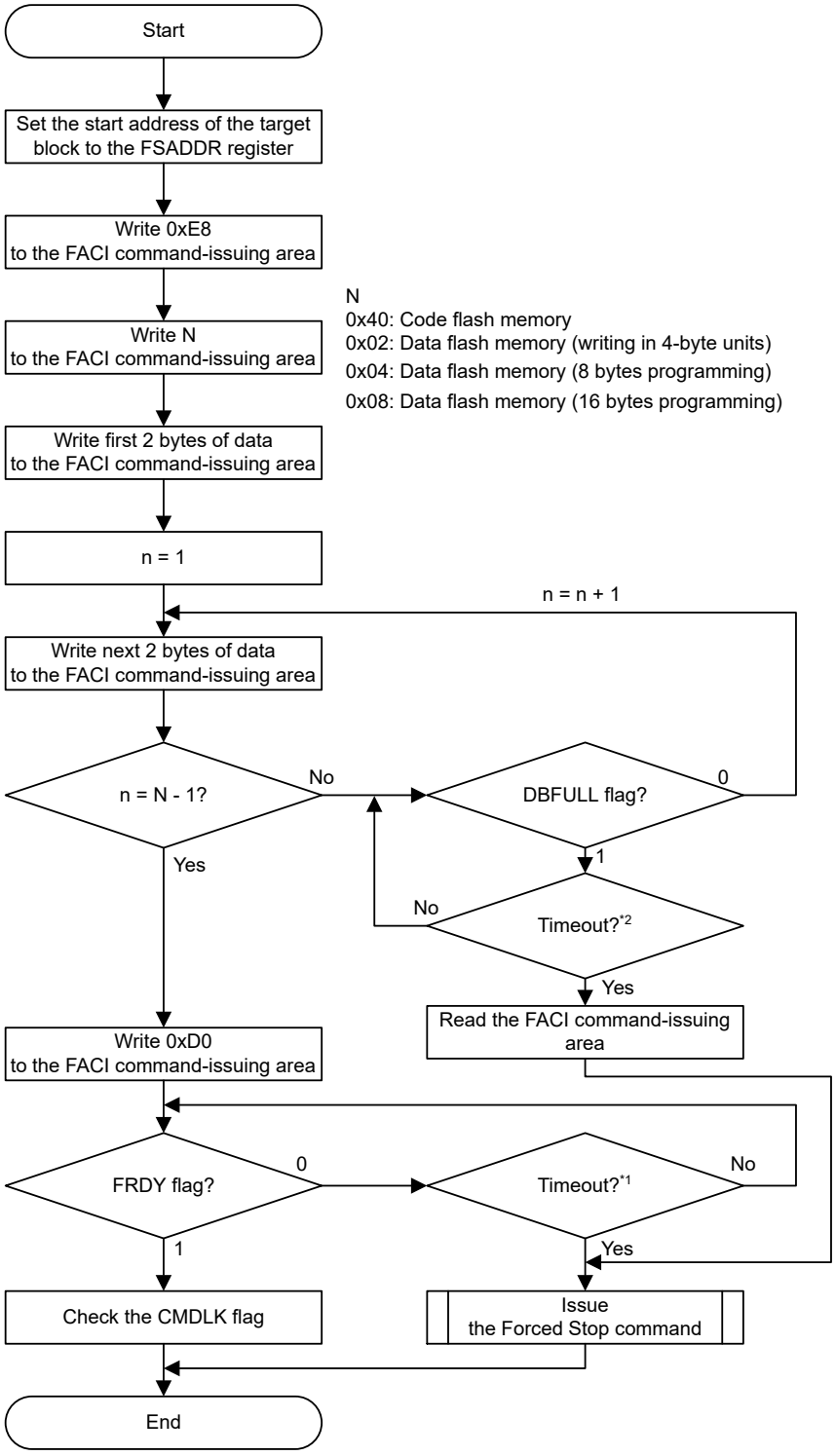
**Figure 44.12 Recovery flow from the command-locked state**

### 44.9.3.7 Program Command

The program command is used for writing to the user area and data area. Before issuing the FACI program command, set the first address of the target block in the FSADDR register. Writing 0xD0 at the final access of the FACI command-issuing area starts the program command processing. If the target area of program command processing contains area that are not for writing, write 0xFFFF to the corresponding area.

Issuing the program command while the FACI internal data buffer is full leads to a wait on the peripheral bus that might affect communications performance of other peripheral modules. To avoid a wait, set the DBFULL bit in the FSTATR register to 0 when issuing the FACI command. Writing to the data area does not lead to the data buffer becoming full.

Figure 44.13 shows the usage of the program command.




Note 1. Judgment of the timeout is based on 1.1 times the maximum time of programming. See section 47, Electrical Characteristics about those time

Note 2. 1.4 µs ( $f_{FCLK}$ ≥ 20 MHz)

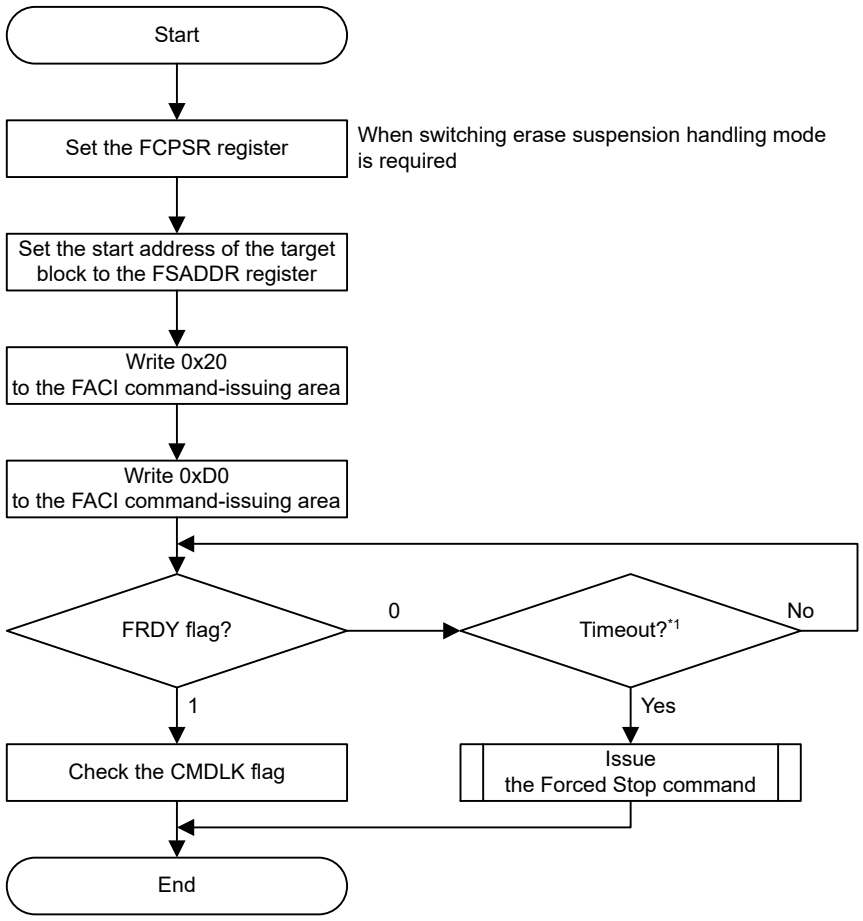
**Figure 44.13 Usage flow of the program command**

### 44.9.3.8 Block Erase Command

The block erase command is used for erasing user area or data area. The erase unit is one block. Before issuing a block erase command, set the first address of the target block to FSADDR register. Writing 0xD0 at the second write access of the FACI command triggers the FACI to start the block erase command processing. Completion of command processing can be confirmed with the FRDY bit of FSTATR register.

Set the FCPSR registers before issuing the block erase command. Additionally, FCPSR must be set when the erasure-suspended mode is to be switched.

Figure 44.14 shows the usage of the block erase command.




Note 1. Judgment of the timeout is based on 1.1 times the maximum time of erasure. See section 47, Electrical Characteristics.

**Figure 44.14 Usage flow of the block erase command**

### 44.9.3.9 Multi Block Erase Command

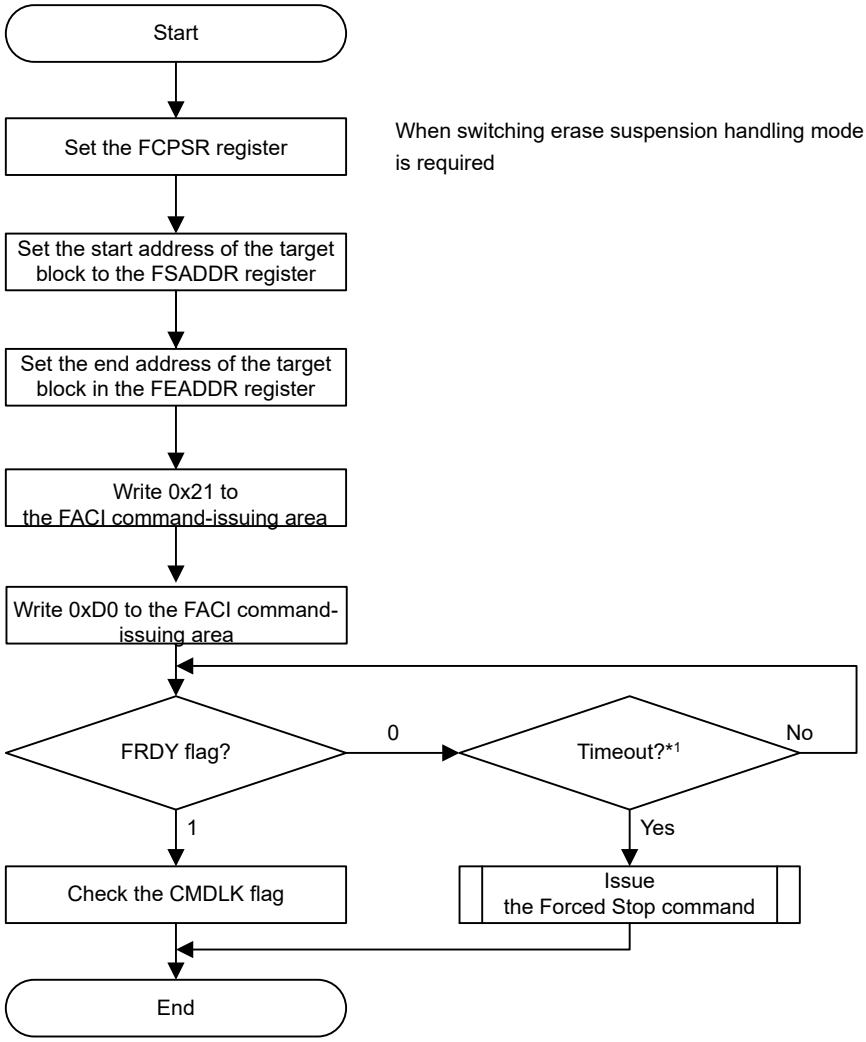
The multi block erase command is used for erasing data area. The erase unit is 64, 128, or 256 bytes. Before issuing the multi block erase command, set the start address to FSADDR register and the end address to FEADDR register. Writing 0xD0 at the second write access of the FACI command triggers FACI to start the multi block erase command processing. Completion of command processing can be confirmed with the FRDY bit of FSTATR register.

Set the FCPSR registers before issuing the multi block erase command. Additionally, FCPSR must be set when the erasure-suspended mode is to be switched.

The erase size is specified by both the FSADDR and FEADDR settings. Table 44.17 describes how to set the FSADDR and FEADDR.

**Table 44.17 Settings for the erase size**

| Erase size | FSADDR | FEADDR |
|---|---|---|
| 64 bytes | FSA0 to FSA5 = 0 (64 byte-boundary) | FSADDR + 0x3C |
| 128 bytes | FSA0 to FSA6 = 0 (128 byte-boundary) | FSADDR + 0x7C |
| 256 bytes | FSA0 to FSA7 = 0 (256 byte-boundary) | FSADDR + 0xFC |




Note 1. Judgment of the timeout is based on 1.1 times the maximum time of erasure. See section 47, Electrical Characteristics.

**Figure 44.15 Usage flow of the multi block command**

### 44.9.3.10 P/E Suspend Command

The P/E suspend command is used to suspend programming/erasure. Before issuing a P/E suspend command, check that the CMDLK bit in the FASTAT register is 0, and that the execution of programming/erasure is performed normally. To confirm that the P/E suspend command can be received, check that the SUSRDY bit in the FSTATR register is 1. After issuing a P/E suspend command, read the CMDLK bit to confirm that no error occurs.

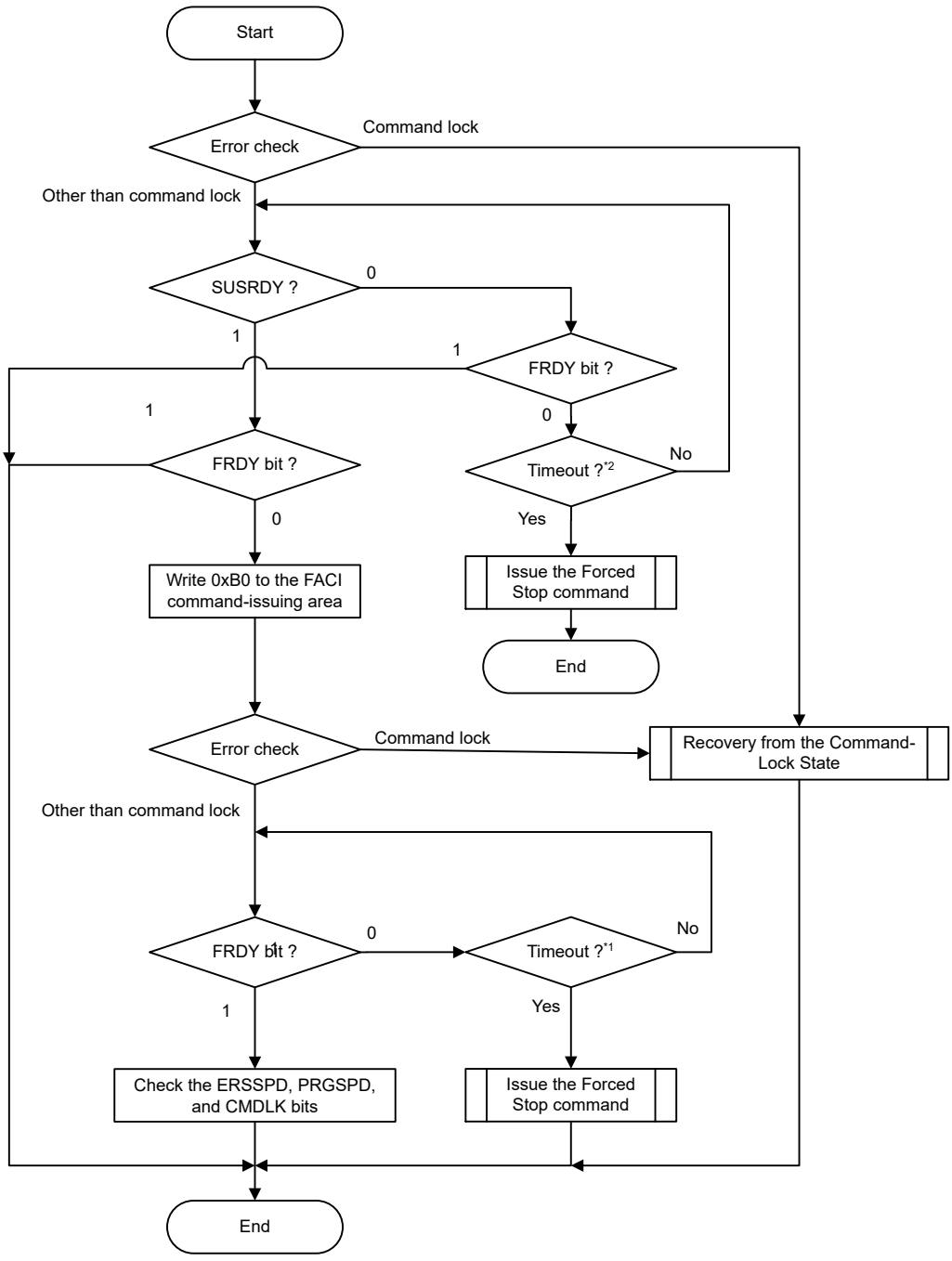
If an error occurs during programming/erasure, the CMDLK bit is set to 1. When programming/erasure processing has finished from the time when the SUSRDY bit is 1 to when the P/E suspend command is received, no error occurs and the

suspended state is not entered (the FRDY bit in the FSTATR register is 1 and the ERSSPD and PRGSPD bits in FSTATR are 0).

When a P/E suspend command is received and the programming/erasure suspend processing finishes normally, the flash sequencer enters the suspended state, the FRDY bit is set to 1, and the ERSSPD or PRGSPD bit is 1. After issuing a P/E suspend command, check that the ERSSPD or PRGSPD bit is 1 and the suspended state is entered, then proceed with the subsequent flow. If a P/E resume command is issued in the subsequent flow even when the suspended state is not entered, an illegal command error occurs and the flash sequencer shifts to the command-locked state (see section 44.11.2. Error Protection).

If the erasure suspended state is entered, programming to blocks other than an erasure target block can be performed. Additionally, the programming and erasure suspended states can shift to read mode by clearing the FENTRYR register.

Figure 44.16 shows the usage of the P/E suspend command.




Note 1. Judgment of the timeout is based on 1.1 times the maximum time of suspend delay during erasure for both code flash and data flash.
See section 47, Electrical Characteristics.
Note 2. 4.4µs ($f_{FCLK}$ >= 20MHz)

**Figure 44.16 Usage flow of the P/E suspend command**

## (1) Suspension during Programming

When issuing a P/E suspend command during flash memory programming, the flash sequencer suspends programming processing. Figure 44.17 shows the suspend programming operation. When receiving programming-related command, the flash sequencer clears the FRDY bit in the FSTATR register to 0 to start programming. If the flash sequencer enters the state

in which the P/E suspend command can be received after programming starts, it sets the SUSRDY bit in the FSTATR register to 1.

When a P/E suspend command is issued, the flash sequencer receives the command and clears the SUSRDY bit to 0. If the flash sequencer receives a P/E suspend command while a programming pulse is applied, the flash sequencer continues with the pulse. After the specified pulse application time, the flash sequencer finishes pulse application, starts the programming suspend processing, and sets the PRGSPD bit in the FSTATR register to 1.

When a suspended processing finishes, the flash sequencer sets the FRDY bit to 1 to enter the programming suspended state. When receiving a P/E resume command in the programming suspended state, the flash sequencer clears the FRDY and PRGSPD bits to 0 and resumes programming.

Figure 44.17 shows the timing for suspension during programming.

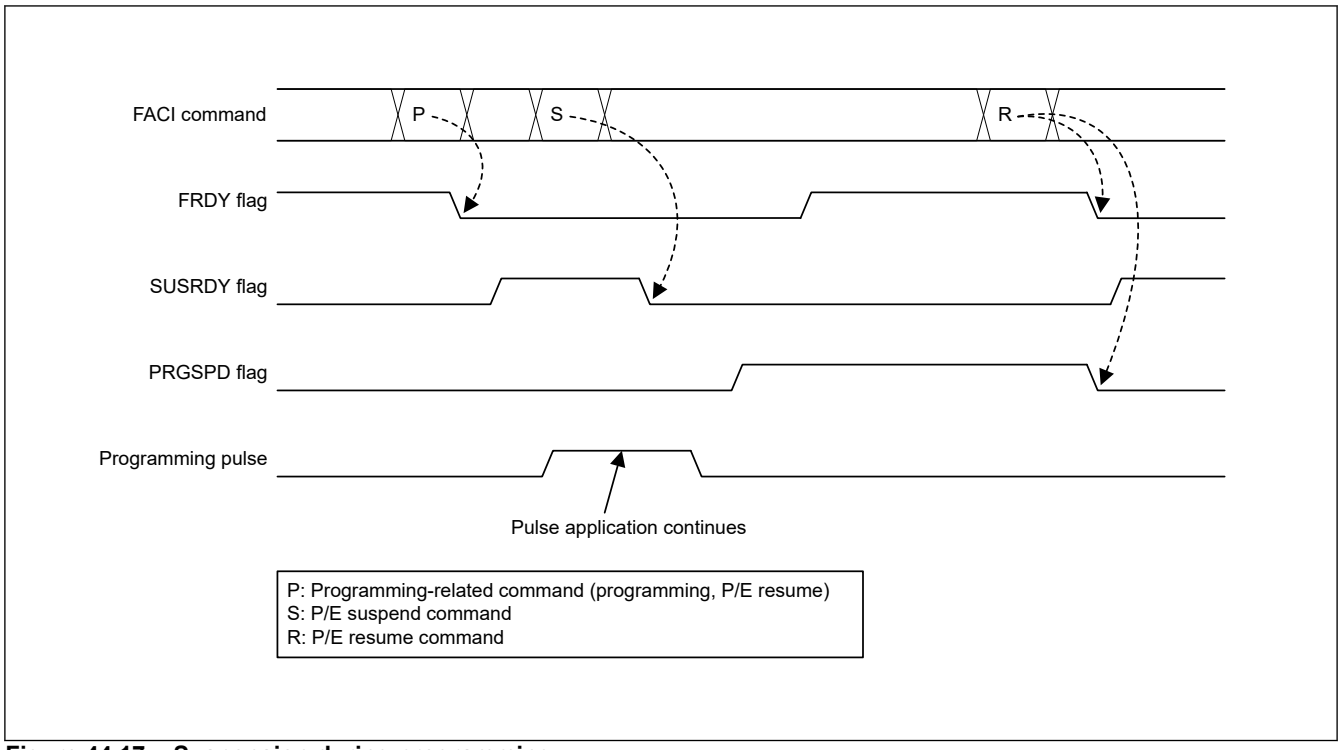



**Figure 44.17 Suspension during programming**

### (2) Suspension during Erasure (Suspension Priority Mode)

The flash sequencer has a suspension priority mode for the suspension of erasure. Figure 44.18 shows the suspend operation of erasure when the erasure suspend mode is set to the suspension priority mode (the ESUSPMD bit in the FCPSR register is 0).

When receiving an erasure-related command, the flash sequencer clears the FRDY bit in the FSTATR register to 0 to start erasure. If the flash sequencer enters the state in which the P/E suspend command can be received after erasure starts, it sets the SUSRDY bit in the FSTATR register to 1.

When a P/E suspend command is issued, the flash sequencer receives the command and clears the SUSRDY bit to 0.

When receiving a suspend command during erasure, the flash sequencer starts the suspend processing and sets the ERSSPD bit in the FSTATR register to 1 even when it is applying an erasure pulse. When the suspended processing finishes, the flash sequencer sets the FRDY bit to 1 to enter the erasure suspended state. When receiving a P/E resume command in the erasure suspended state, the flash sequencer clears the FRDY and ERSSPD bits to 0 and resumes erasure. Operations of the FRDY, SUSRDY, and ERSSPD bits at the suspension and resumption of erasure are the same, regardless of the erasure suspend mode.

The setting of the erasure suspend mode affects the control method of erasure pulses. In suspension priority mode, when receiving a P/E suspend command while erasure pulse A that has not been previously suspended is being applied, the flash sequencer suspends the application of erasure pulse A and enters the erasure suspended state. When receiving a P/E suspend command while reapplying erasure pulse A after erasure is resumed with a P/E resume command, the flash sequencer

continues to apply erasure pulse A. After the specified pulse application time, the flash sequencer finishes erasure pulse application and enters the erasure suspended state.

When the flash sequencer receives a P/E resume command next and erasure pulse B is being applied, the flash sequencer receives a P/E suspend command again, and the application of erasure pulse B is then suspended. In suspension priority mode, delays due to suspension can be minimized because the application of an erasure pulse is suspended once per pulse, and priority is given to the suspend processing.

If the interval of suspension after resume is longer than $t_{REST1}$ (Resume time: priority on suspension, resume after the 1st suspend for the same pulse), suspend delay will be always $t_{SESD1}$ (Suspend delay: priority on suspension, the 1st suspend for the same pulse).

If the interval of suspension after resume is shorter than $t_{REST1}$, suspend delay becomes either $t_{SESD1}$ or $t_{SESD2}$ (Suspend delay: priority on suspension, the 2nd suspend for the same pulse).

(The value of $t_{REST1}$ / $t_{SESD1}$ / $t_{SESD2}$, see section 47, Electrical Characteristics.)

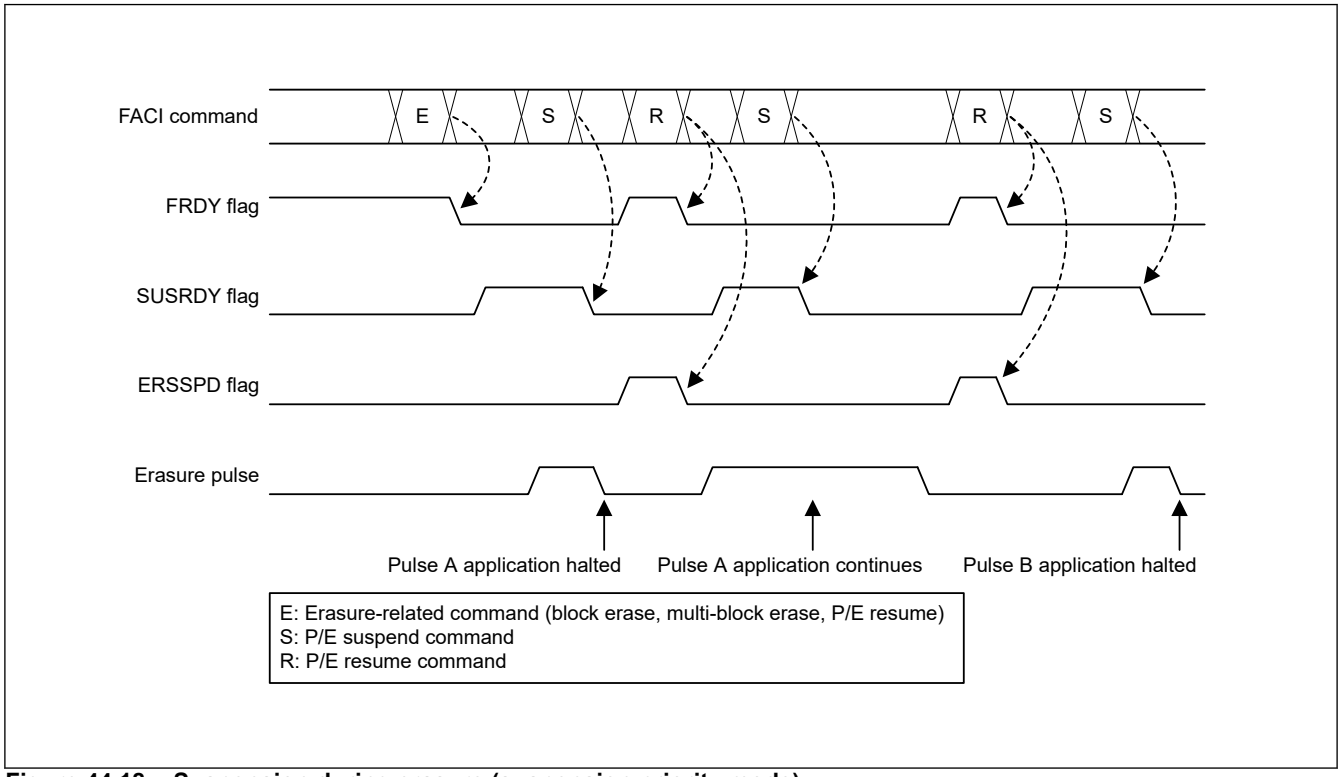



**Figure 44.18 Suspension during erasure (suspension priority mode)**

## (3) Suspension during Erasure (Erasure Priority Mode)

The flash sequencer has an erasure priority mode for the suspension of erasure. Figure 44.19 shows the suspend operation of erasure when the erasure suspend mode is set to the erasure priority mode (the ESUSPMD bit in the FCPSR register is 1). The control method of erasure pulses in erasure priority mode is the same as that of programming pulses for the programming suspend processing.

If the flash sequencer receives a P/E suspend command while an erasure pulse is applied, the flash sequencer continues to apply the pulse. In this mode, the required time for the erasure processing can be reduced compared to the suspension priority mode because the re-application of erasure pulses does not occur when a P/E resume command is issued.

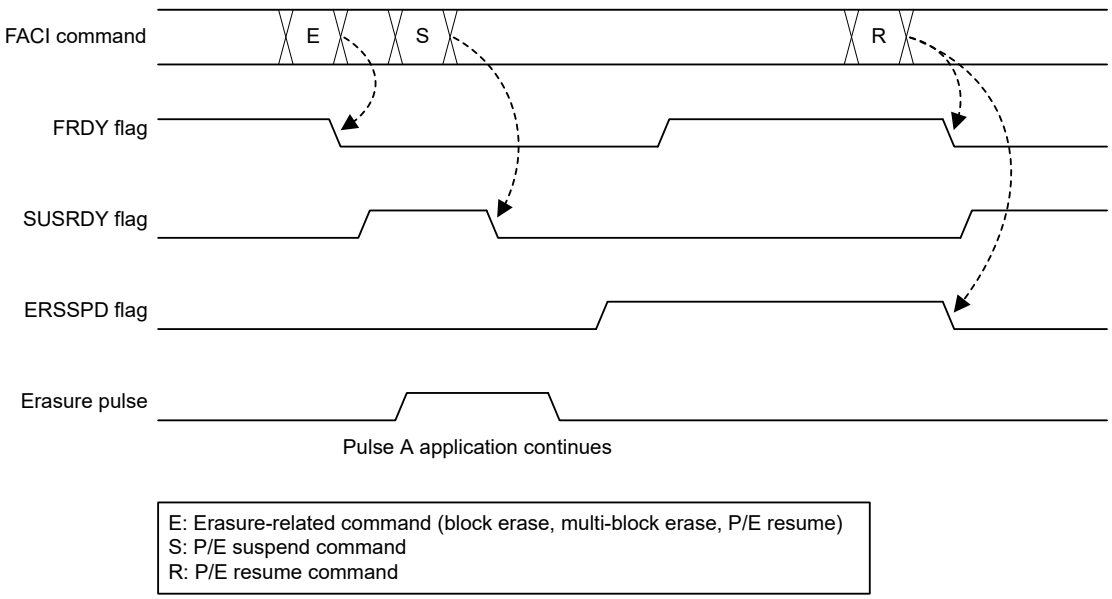
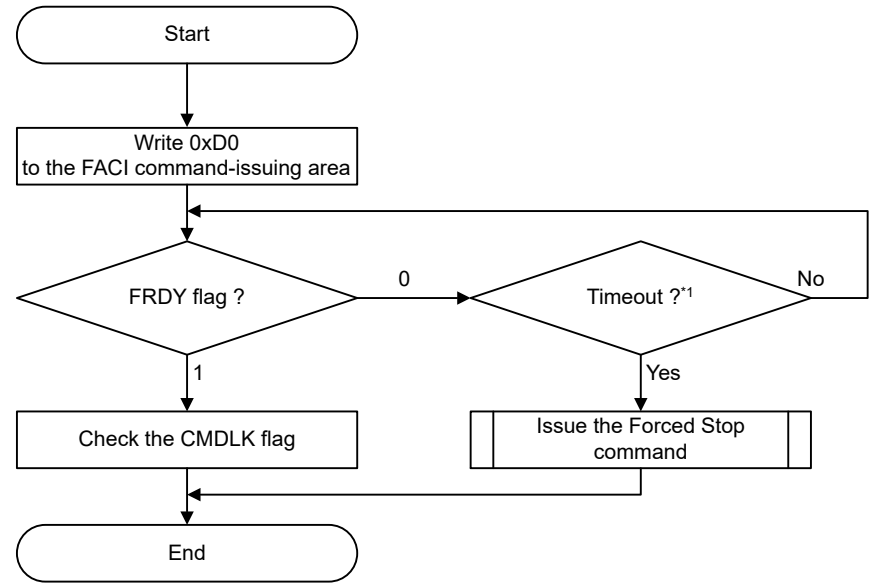



**Figure 44.19 Suspension during erasure (erasure priority mode)**

### 44.9.3.11 P/E Resume Command

The P/E resume command is used to resume suspended programming or erasure. If the FENTRYR setting has been modified during suspension, issue a P/E resume command only after resetting FENTRYR to the previous value that was held before the P/E suspend command was issued. Figure 44.20 shows usage of the P/E resume command.




Note 1. Judgment of the timeout is based on 1.1 times the maximum time for halted processing of the FACI command. See section 47, Electrical Characteristics.

**Figure 44.20 Usage flow of the P/E resume command**

### 44.9.3.12 Status Clear Command

The status clear command is used to clear the command-locked state (see section 44.9.3.6. Recovery from the Command-Locked State).

You can use the status clear command to clear the following bits in the FSTATR register in the command-locked state:

- ILGLERR
- ILGCOMERR
- FESETERR
- SECERR
- OTERR
- ERSERR
- PRGERR

Figure 44.21 shows usage of the status clear command.

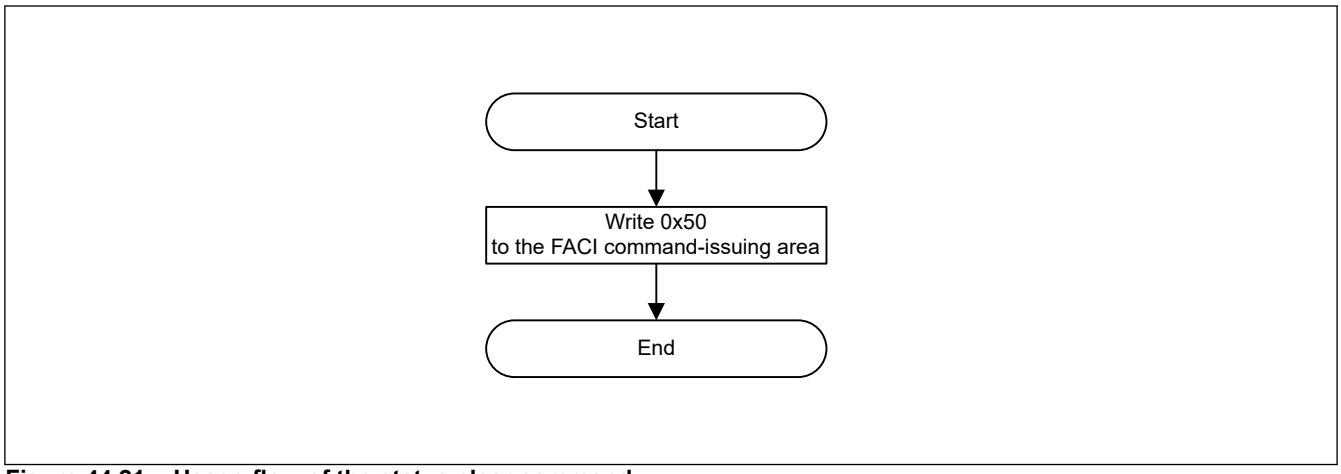


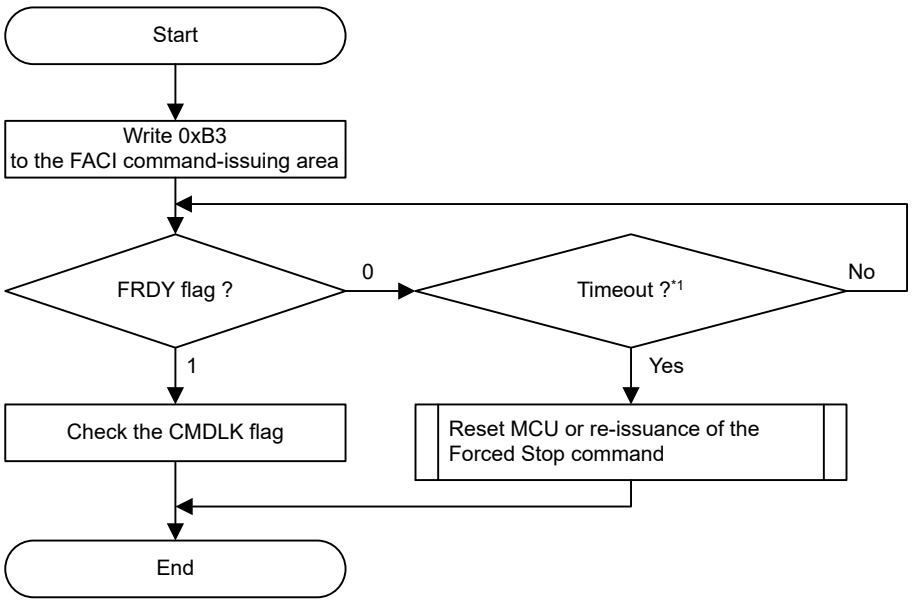

**Figure 44.21 Usage flow of the status clear command**

### 44.9.3.13 Forced Stop Command

The forced stop command is used to forcibly end command processing by the flash sequencer. Although this command halts command processing more quickly than the P/E suspension command, values from the programming or erasure that are in progress are not guaranteed. Additionally, resumption of processing is not possible. Processing of programming or erasure that is halted by the forced stop command is also defined as one programming round.

Executing the forced stop command also initializes part of the FACI, the whole FCU, the FSTATR and FASTAT registers. This command can be used in the procedure for recovery from the command-locked state and for processing in response to a timeout of the flash sequencer (see section 44.9.3.6. Recovery from the Command-Locked State).

Figure 44.22 shows usage of the forced stop command.




Note 1. Judgment of the timeout is based on 1.1 times the maximum time of the Forced Stop command latency. See section 47, Electrical Characteristics.

**Figure 44.22 Usage flow of the forced stop command**

### (1) Notes on Using the Forced Stop Command during Command Issue

When using the forced stop command at the timeout occurrence by DBFULL bit of the program command, writing in the FACI command-issuing area is sometimes processed as writing in data of the program command. See Table 44.3 in section 44.3. Address Space for information on the FACI command-issuing area to force a command lock. Then issue a forced stop command with return method from the command lock status (see Figure 44.13). Locking commands is possible in any case where the unit for reading the FACI command issuing area is 8, 16, or 32 bits.

## 44.9.3.14 Blank Check Command

The blank check command is used to confirm that an area is in the non-programmed state. Values read from the data flash memory that have been erased but not yet programmed again that is in the non-programmed state, are undefined.

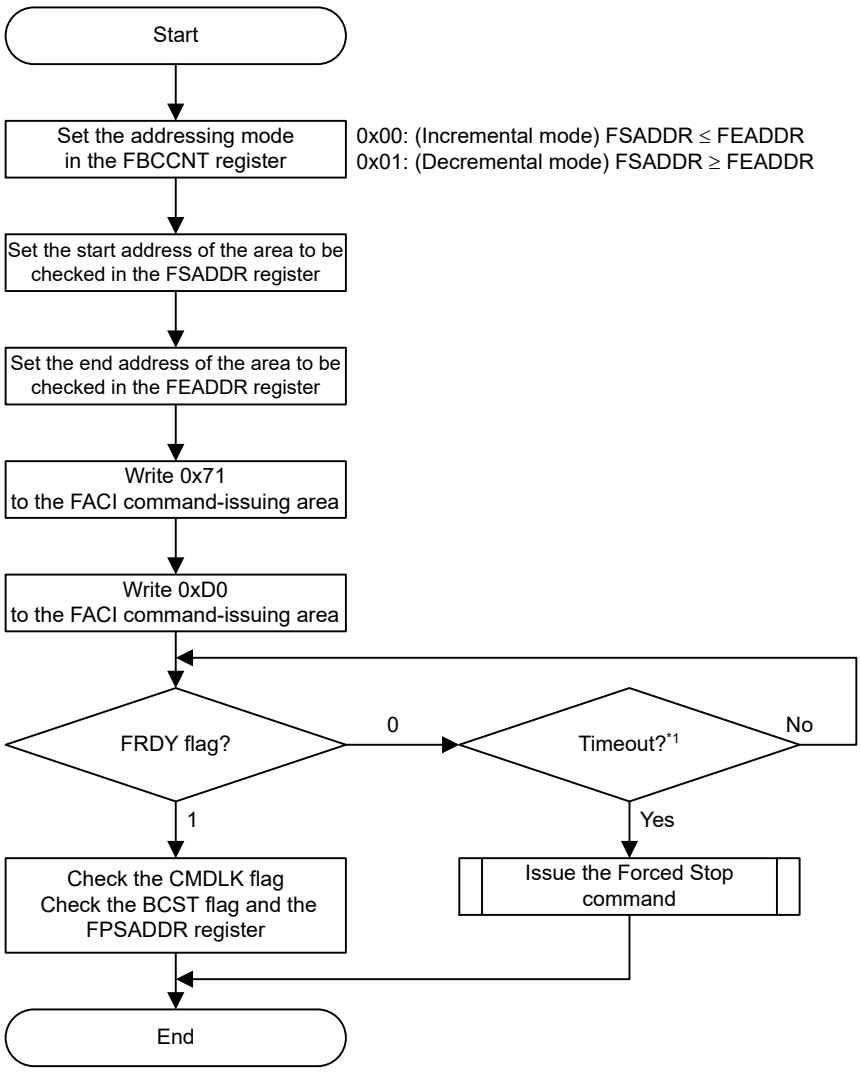
Before issuing the Blank Check command, set addressing mode, start address, and end address of the target area for Blank Check to the FBCCNT, FSADDR, and FEADDR registers. When Blank Check addressing mode is set to decremental mode (i.e. FBCCNT.BCDIR = 1), address specified in FSADDR should be equal to or larger than address in FEADDR.

On the other hand, the address in FSADDR should be equal to or smaller than address in FEADDR when Blank Check addressing mode is set to incremental mode (i.e. FBCCNT.BCDIR = 0).

If the settings of the BCDIR bit, FSADDR, and FEADDR are inconsistent, the flash sequencer enters the command-locked state. The size of the target area for Blank Check is in the range from 4 bytes to the data flash memory capacity and is set in units of 4 bytes.

Write 0x71 and 0xD0 to the FACI command-issuing area to start Blank Check. Completion of processing can be confirmed by the FRDY bit of the FSTATR register. At the end of processing, the result of Blank Check is stored in the BCST bit in the FBCSTAT register. If non-programmed data exists within the target area for Blank Check, flash sequencer stops Blank Check command operation. In this case, address of non-programmed data is indicated to FPSADDR register.

Figure 44.23 shows usage of the blank check command.



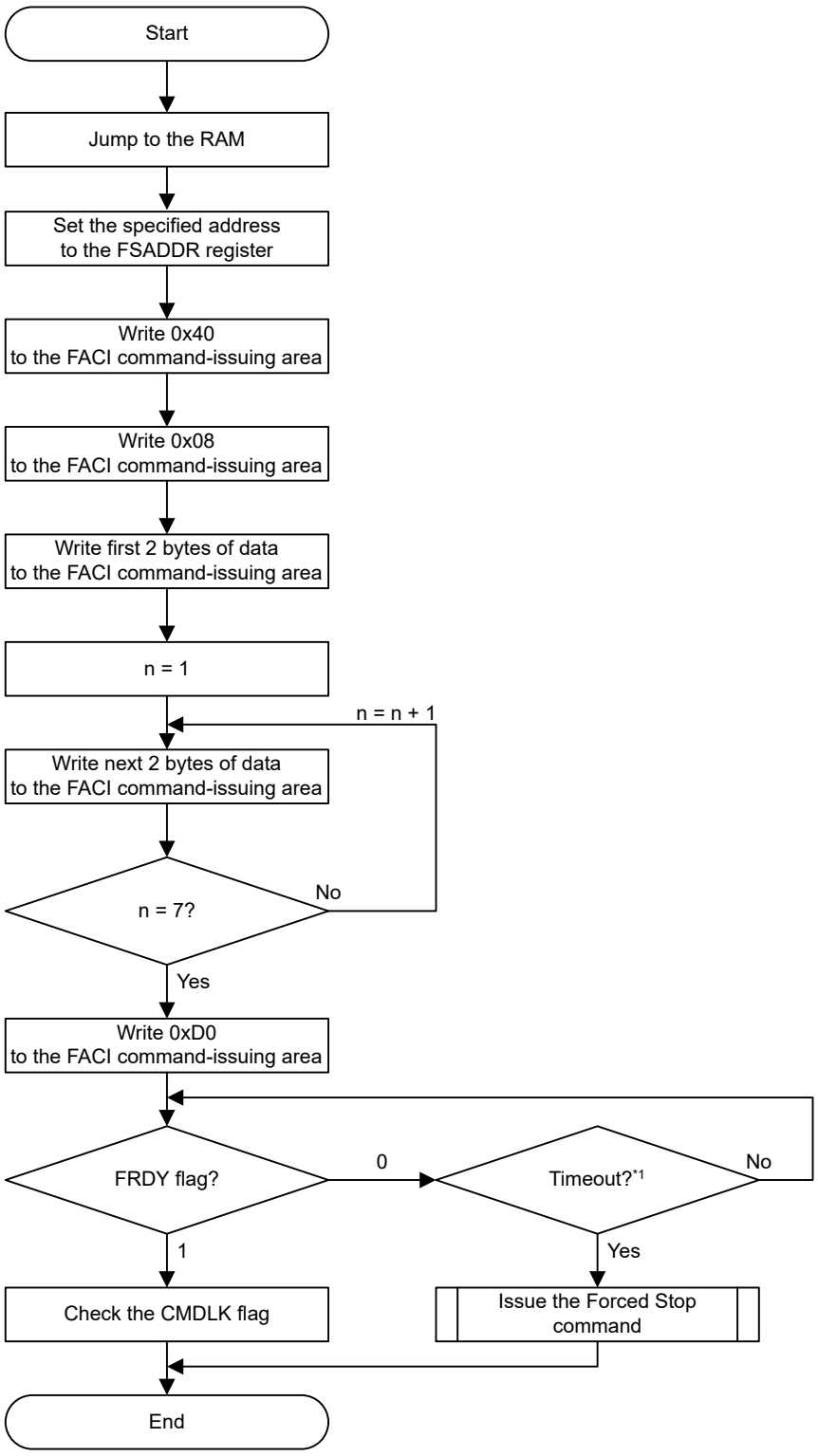

Note 1. Judgment of the timeout is based on 1.1 times the maximum time of Blank Checking for the whole target range. See section 47, Electrical Characteristics.

**Figure 44.23 Usage flow of the blank check command**

### 44.9.3.15 Configuration Set Command

The Configuration set command is used to set option-setting memory. Before issuing the Configuration set command, set the specified address (shown in Table 44.18) in the FSADDR register. Writing 0xD0 to the FACI command-issuing area in the final access for issuing the FACI command starts FACI processing of the Configuration set command.

Figure 44.24 shows usage of the configuration set command.




Note 1. Judgment of the timeout is based on 1.1 times the maximum time of programing in option-setting memory. See section 47, Electrical Characteristics.

**Figure 44.24 Usage flow of the configuration set command**

The correspondence between the possible target data for configuration setting and the address value set in the FSADDR register is shown in Table 44.18. For details, see section 44.4.12. FSADDR : FACI Command Start Address Register.

**Table 44.18 Address Used by Configuration Set Command**

| Address | FSADDR Register Value | Setting Data | Operation of additional writing | | Timing when the Setting is Enabled |
|---|---|---|---|---|---|
| | | | SAS.FSPR bit is 1 | SAS.FSPR bit is 0 | |
| 0x0100_A100 | 0x0100_A100 | Option Function Select Register 0 (OFS0) | Writable | Writable | At a reset |
| 0x0100_A134 | 0x0100_A130 | Start-up Area Setting Register (SAS) | Writable | Not writable*1 | When a reset or command is executed |
| 0x0100_A180 | 0x0100_A180 | Option Function Select Register 1 (OFS1) | Writable | Writable | At a reset |
| 0x0100_A1C0 | 0x0100_A1C0 | Block Protect Setting Register (BPS) | Writable*2 | Writable*2 | When a reset or command is executed |
| 0x0100_A1E0 | 0x0100_A1E0 | Permanent Block Protect Setting Register (PBPS) | Writable*3 (from 1 to 0 only) | Writable*3 (from 1 to 0 only) | When a reset or command is executed |
| 0x0100_A200 | 0x0100_A200 | Option Function Select Register 1 Secure (OFS1_SEC) | Writable | Writable | At a reset |
| 0x0100_A240 | 0x0100_A240 | Block Protect Setting Register Secure (BPS_SEC) | Writable*4 | Writable*4 | When a reset or command is executed |
| 0x0100_A260 | 0x0100_A260 | Permanent Block Protect Setting Register Secure (PBPS_SEC) | Writable*5 (from 1 to 0 only) | Writable*5 (from 1 to 0 only) | When a reset or command is executed |
| 0x0100_A280 | 0x0100_A280 | Option Function Select Register 1 Select (OFS1_SEL) | Writable | Writable | At a reset |
| 0x0100_A2C0 | 0x0100_A2C0 | Block Protect Setting Register Select (BPS_SEL) | Writable | Writable | At a reset |

Note 1. The SAS.FSPR bit cannot be restored to 1 by using the Configuration set command once it is set to 0. Therefore, setting the start-up area select flags again becomes impossible. (when the Configuration set command is issued to the address of 0x0100A134, the command is locked.) Exercise extra caution when handling the SAS.FSPRbit.
Note 2. Once PBPS[n] bit is set to 0, the BPS[n] bit cannot be restored to 1 by using the Configuration set command.
Note 3. Once these bits are set to 0, the bits cannot be restored to 1 by using the Configuration set command. The PBPS[n] bit cannot be set to 0 by using the Configuration set command when the BPS[n] bit is 1.
Note 4. Once PBPS_SEC[n] bit is set to 0, the BPS_SEC[n] bit cannot be restored to 1 by using the Configuration set command.
Note 5. Once these bits are set to 0, the bits cannot be restored to 1 by using the Configuration set command. The PBPS_SEC[n] bit cannot be set to 0 by using the Configuration set command when the BPS_SEC[n] bit is 1.

## 44.10 Suspend Operation

Reading from the flash memory is not possible during programming or erasure if the conditions for background operation given in Table 44.29 are not satisfied. When a P/E suspend command is issued to suspend the programming or erasure of the flash memory, reading from the flash memory is enabled. Regarding P/E suspend commands, there are one suspend command mode for programming and two suspend command modes for erasure (suspension priority mode and erasure priority mode). To resume suspended programming or erasure, the P/E resume command is available. For details on the suspend operation, see Figure 44.16.

## 44.11 Protection Function

### 44.11.1 Software Protection

Software protection disables programming and erasure of the code flash memory through the settings of control registers and block protect setting in the user area. If an attempt is made to issue an FACI command against software protection, the flash sequencer enters the command-locked state.

#### 44.11.1.1 Protection through FWEPROR

Unless the FWEPROR.FLWE[1:0] bits are set to 01b, programming cannot proceed in any mode.

#### 44.11.1.2 Protection by FENTRYR

When the FENTRYR register is set to 0x0000, the flash sequencer enters read mode. In read mode, FACI commands cannot be accepted. If an attempt is made to issue an FACI command in read mode, the flash sequencer enters the command-locked state.

#### 44.11.1.3 Protection by Block Protect Setting

Each block in user area has the block protect setting (BPS or BPS_SEC). When the FBPROT0 or FBPROT1 register is 0x0000 and the block protect bit is 0, issuing the Program or Block Erase command to user area of the code flash causes the command-locked state. To program or erase the block whose block protect bit is 0, set the FBPROT0 or FBPROT1 register to 0x0001.

The block protect setting can be locked by the permanent block protect setting (PBPS or PBPS_SEC). When the permanent block protect setting and the block protect setting are 0, issuing a Program or Block erase command to user area of the code flash causes the flash sequencer to enter the command-locked state regardless of the FBPROT0 and FBPROT1 register settings.

Valid block protect setting (BPS or BPS_SEC) depends on the Block Protect Select bit (BPS_SEL).

See section 44.12.2. Permanent Block Protect Setting for details of the block protect setting and permanent block protect setting. See section 44.4.15. FBPROT0 : Flash Block Protection Register and section 44.4.16. FBPROT1 : Flash Block Protection for Secure Register for more information.

For details of block protect setting (BPS or BPS_SEC) and block protect select (BPS_SEL), see section 6, Option-Setting Memory.

The protected area by the block protect setting is always determined by the address of the FSADDR register setting regardless of the address swapping function setting (the startup area select). Table 44.19 to Table 44.20 show the relation of user area and the block protect setting in each function setting.

- BPS[0] to BPS[n] or BPS_SEC[0] to BPS_SEC[n] are assigned to the block of user area (for example, address is 0x00_0000 to the last block address).
- BPS[0]/BPS_SEC[0] and BPS[1]/BPS_SEC[1] are assigned to the block of user area depending on the startup area select setting (SAS.BTFLG bit). (See section 44.11.3. Start-Up Program Protection).

Table 44.19 shows the block protect setting when the startup area select is disabled (not swapping).

Table 44.20 show example of the block protect setting when the address conversion function is used.

**Table 44.19 Example of Block Protect setting when SAS.BTFLG is 1**

| FSADDR[23:0] | Block size | Block protect setting | User area block number | Notes |
|---|---|---|---|---|
| The last block address | 32 KB | BPS[n] or BPS_SEC[n] | Block n | — |
| ⋮ | ⋮ | ⋮ | ⋮ | — |
| 0x01_8000 to 0x01_FFFF | 32 KB | BPS[9] or BPS_SEC[9] | Block 9 | — |
| 0x01_0000 to 0x01_7FFF | 32 KB | BPS[8] or BPS_SEC[8] | Block 8 | — |
| 0x00_E000 to 0x00_FFFF | 8 KB | BPS[7] or BPS_SEC[7] | Block 7 | — |
| 0x00_C000 to 0x00_DFFF | 8 KB | BPS[6] or BPS_SEC[6] | Block 6 | — |
| ⋮ | ⋮ | ⋮ | ⋮ | — |
| 0x00_2000 to 0x00_3FFF | 8 KB | BPS[1] or BPS_SEC[1] | Block 1 | Not swap block 0 and block 1 in this startup area select setting |
| 0x00_0000 to 0x00_1FFF | 8 KB | BPS[0] or BPS_SEC[0] | Block 0 | Not swap block 0 and block 1 in this startup area select setting |

**Table 44.20 Example of Block Protect setting when SAS.BTFLG is 0**

| FSADDR[23:0] | Block size | Block protect setting | User area block number | Notes |
|---|---|---|---|---|
| The last block address | 32 KB | BPS[n] or BPS_SEC[n] | Block n | — |
| ⋮ | ⋮ | ⋮ | ⋮ | — |
| 0x01_8000 to 0x01_FFFF | 32 KB | BPS[9] or BPS_SEC[9] | Block 9 | — |
| 0x01_0000 to 0x01_7FFF | 32 KB | BPS[8] or BPS_SEC[8] | Block 8 | — |
| 0x00_E000 to 0x00_FFFF | 8 KB | BPS[7] or BPS_SEC[7] | Block 7 | — |
| 0x00_C000 to 0x00_DFFF | 8 KB | BPS[6] or BPS_SEC[6] | Block 6 | — |
| ⋮ | ⋮ | ⋮ | ⋮ | — |
| 0x00_2000 to 0x00_3FFF | 8 KB | BPS[1] or BPS_SEC[1] | Block 0 | Swap block 0 and block 1 in this startup area select setting |
| 0x00_0000 to 0x00_1FFF | 8 KB | BPS[0] or BPS_SEC[0] | Block 1 | Swap block 0 and block 1 in this startup area select setting |

## 44.11.2 Error Protection

Error protection detects the issuing of illegal FACI commands, illegal access, and flash sequencer malfunction. FACI command acceptance is disabled (command-locked state) in response to the detection of these errors. The flash memory cannot be programmed or erased while the flash sequencer is in the command-locked state. For release from the command-locked state, issue the Status Clear or Forced Stop command. The Status Clear command can only be used while the FRDY bit in the FSTATR register is 1. The Forced Stop command can be used regardless of the value of the FRDY bit. While the CMDLKIE bit in the FAEINT register is 1, a flash access error (FIFERR) interrupt is generated if the flash sequencer enters the command-locked state (the CMDLK bit in the FASTAT register is set to 1).

If the flash sequencer enters the command-locked state in response to a command other than the P/E suspend command during programming or erasure processing, the flash sequencer continues the processing for programming or erasure. In this state, the P/E suspend command cannot be used to suspend the processing for programming or erasure. If a command is issued in the command-locked state, the ILGLERR bit becomes 1 and the other bits retain the values set from previous error detection.

Table 44.21 shows the error protection types and status bit values after error detections.

**Table 44.21 Error protection type (1 of 3)**

| Error type | Description | ILGCOMERR | FESETERR | SECERR | OTERR | ILGLERR | ERSERR | PRGERR | FLWEERR | CFAE | DFAE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FENTRYR setting error | The value set in FENTRYR is not 0x0000, 0x0001, or 0x0080 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | The FENTRYR setting at suspension is different from that at resumption | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

**Table 44.21 Error protection type (2 of 3)**

| Error type | Description | ILGCOMERR | FESETERR | SECERR | OTERR | ILGLERR | ERSERR | PRGERR | FLWEERR | CFAE | DFAE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Illegal command error | An undefined size is specified in the first cycle of the command. (not byte-write) | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | An undefined code is written in the first access of the FACI command | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | The value specified in the last access of the multiple-access FACI command is not 0xD0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | The value (N) specified in the second write access of the FACI command in the program or configuration set command is wrong | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | Blank Check command is issued with inconsistent BCDIR, FSADDR, and FEADDR settings (see section 44.4.13. FEADDR : FACI Command End Address Register) | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0/1 *1 |
| | A multi block erase command is issued with inconsistent FSADDR and FEADDR settings.<br>● FSADDR > FEADDR<br>● FEADDR is set to reserved area. | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0/1 *1 |
| | An FACI command not acceptable in each mode is issued (see Table 44.15) | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | An FACI command is issued when command acceptance conditions are not satisfied (see Table 44.16) | 0/1 | 0/1 | 0/1 | 0/1 | 1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| | A program or block erase command is issued against the area protected by the block protect setting (see section 44.11.1.3. Protection by Block Protect Setting) | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | A program command is issued against the erase area in erase suspend | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Erasure error | An error occurs during erasure | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Programming error | An error occurs during programming | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Code flash memory access violation | An FACI command is issued to the reserved portion of the user area in code flash P/E mode | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| | Configuration set command is issued to the reserved option-setting memory | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| | Configuration set command of non-secure access is issued to the secure region of TrustZone in the code flash | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| | Program or block erase command of non-secure access is issued to the secure region of user area. | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |

**Table 44.21 Error protection type (3 of 3)**

| Error type | Description | ILGCOMERR | FESETERR | SECERR | OTERR | ILGLERR | ERSERR | PRGERR | FLWEERR | CFAE | DFAE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Data flash memory access violation | A program or block erase command is issued to the reserved data area in data flash P/E mode | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| | A multi block erase command is issued to the reserved data area in data flash P/E mode.<br>(FSADDR is set to reserved data area). | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| | Blank Check command is issued to reserved data area in data flash P/E mode.<br>(FSADDR is set to reserved data area ). | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| | A program, block erase, multi block erase, or blank check command of non-secure access is issued to the secure region of data area. | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| Security error | Configuration set command for the SAS.BTFLG bit setting is issued when the SAS.FSPR bit is 0 (see section 44.9.3.15. Configuration Set Command) | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Others | An FACI command-issuing area is accessed in read mode | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| | An FACI command-issuing area is read in code flash P/E mode or data flash P/E mode | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Flash write erase protection error | A flash memory write protection error is detected by the FWEPROR register setting*2 during command processing by the flash sequencer | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 1 | 0 | 0 |

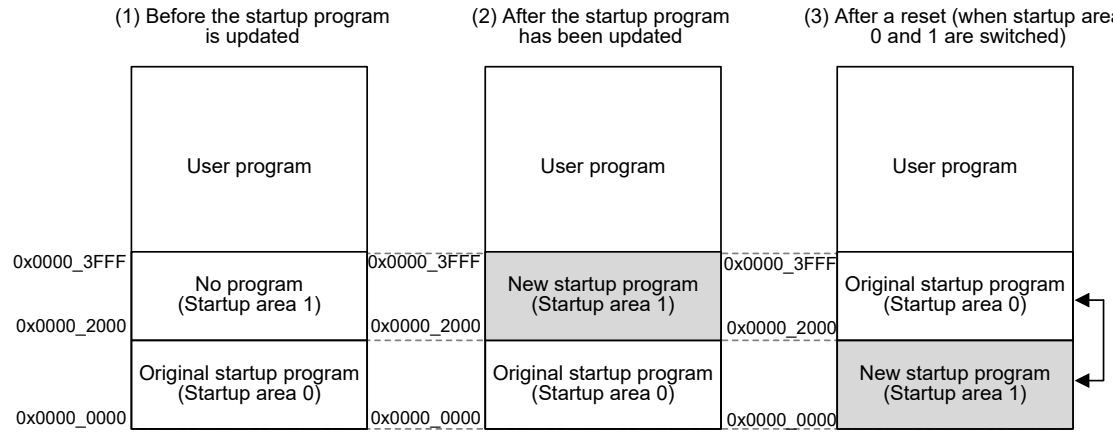
Note 1. DFAE value depends on the FSADDR setting.
Note 2. For details on the FWEPROR register, see section 44.4.8. FWEPROR : Flash P/E Protect Register.

## 44.11.3 Start-Up Program Protection

Protection of the startup program is for protection of the program to be started after a reset (the startup program). This function provides a way to safely update the startup program when rewriting is suspended during a reset.

The startup area is 8 Kbytes in size and is assigned to the user area in the code flash memory. This function uses the values of the SAS.BTFLG bit and the FSUACR.SAS[1:0] bits to change the area where the startup program is stored in block units (see Figure 44.25 to Figure 44.28).

In protection of the startup program, the state of the selection of the startup area can be fixed by the FSPR bit. However, the SAS.FSPR bit never be restored to 1 once the flag is set to 0. Exercise extra caution when handling the SAS.FSPR bit.




Note: The addresses are used for reading, programming, and erasure. Startup area 0 is erased when Block Erase command is issued for the address 0x00000000 with allocation shown under 3.

1. Before the startup program is updated
   When the SAS.BTFLG bit is 1 and the FSUACR.SAS[1:0] bits are 0xb, the address range from 0x00002000 to 0x00003FFF is selected as startup area 1 and the address range from 0x00000000 to 0x00001FFF is selected as startup area 0.
2. After the startup program has been updated
   A new startup program can be written to startup area 1. At this time, startup area 0 will still contain the previous startup program, so rewriting by starting up from startup area 0 remains possible if rewriting to the startup area 1 fails.
3. After a reset (when startup areas 0 and 1 are switched)
   When rewriting startup area 1 is successfully completed, switch the assignment of startup areas 0 and 1 by rewriting the startup area selection bit (the SAS.BTFLG bit) with Configuration set command. Afterwards, the program is started up from startup area 1 after a reset.

**Figure 44.25 Concept of Protection of the Startup Program**

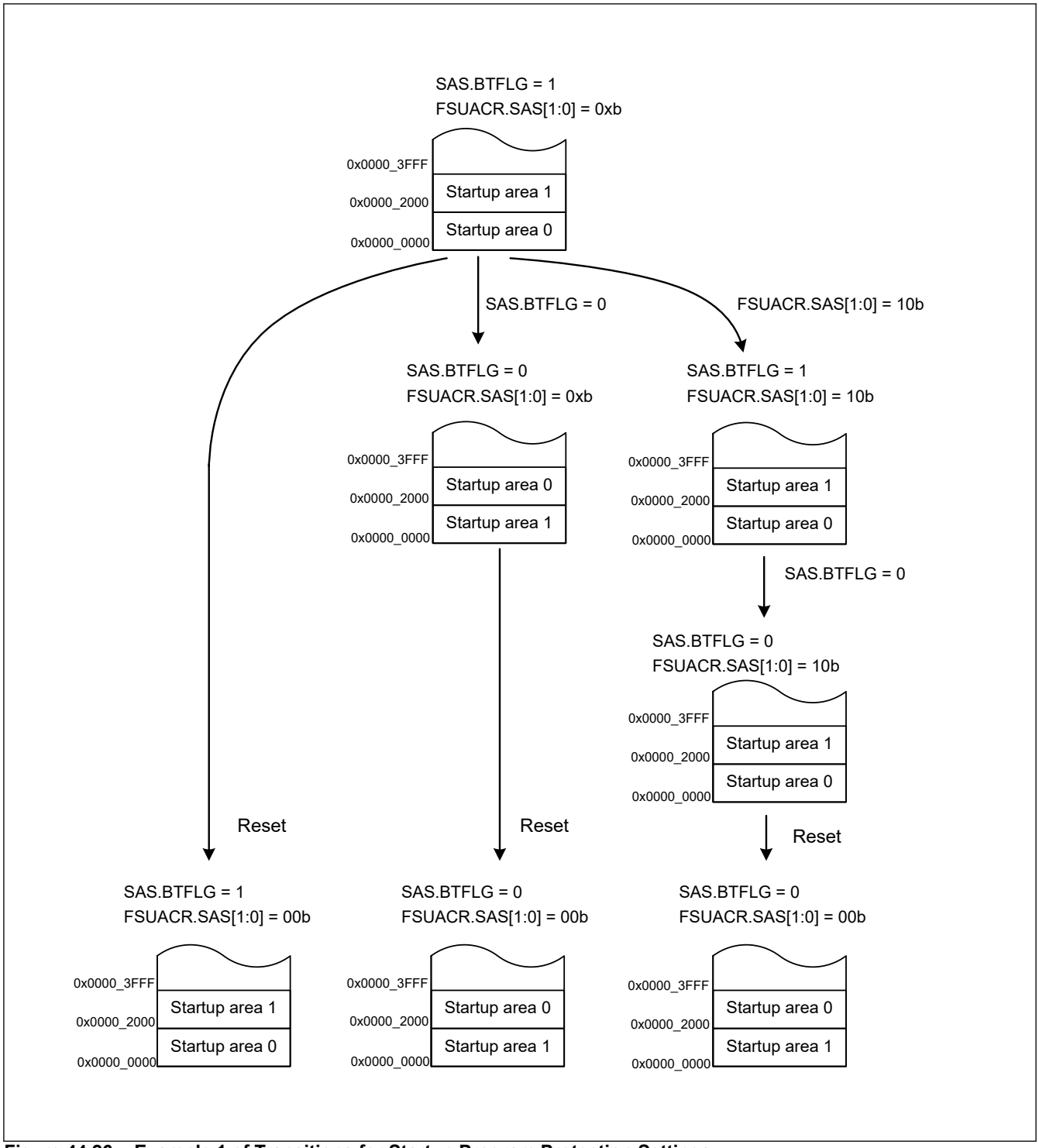



**Figure 44.26 Example 1 of Transitions for Startup Program Protection Settings**

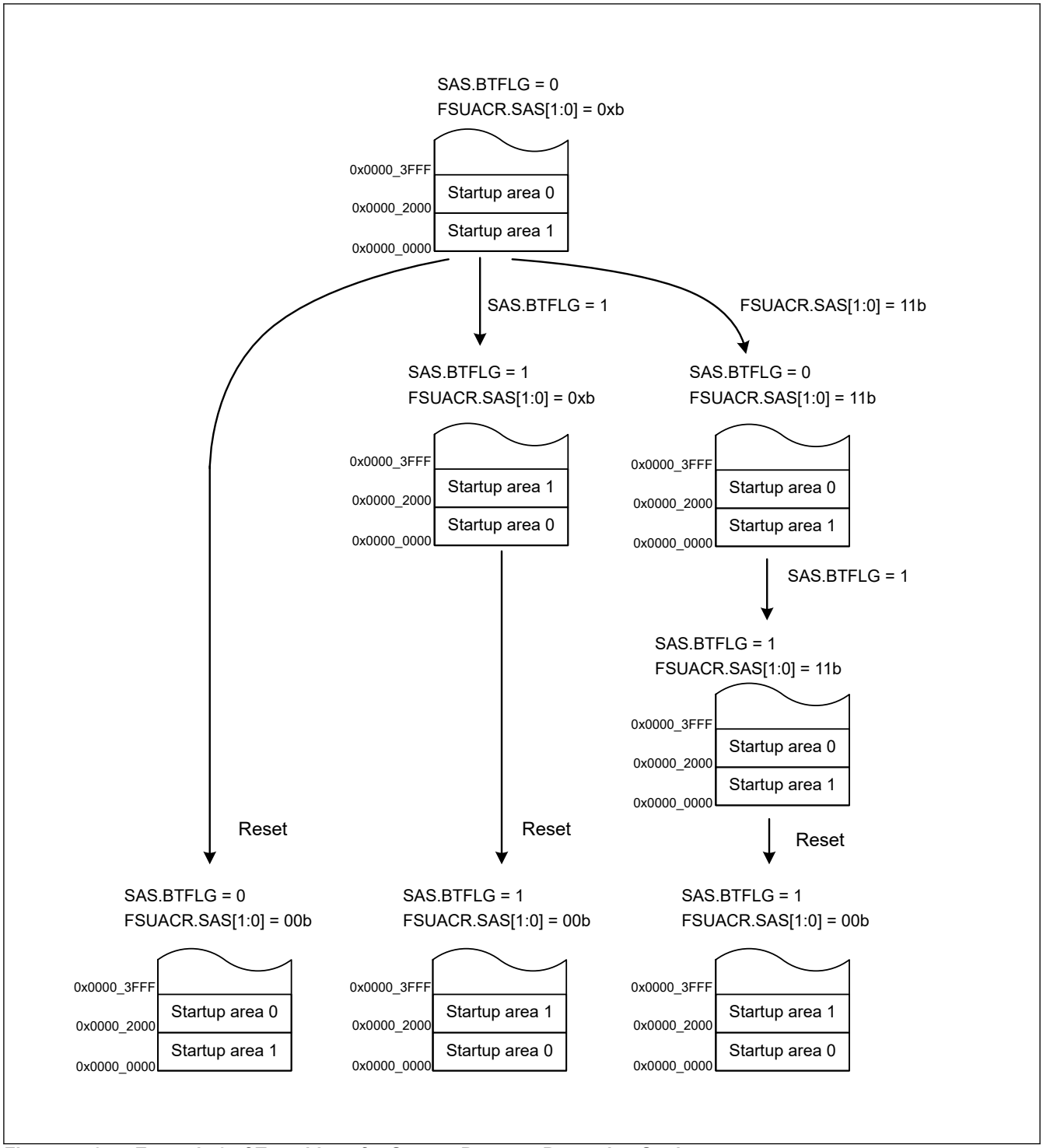
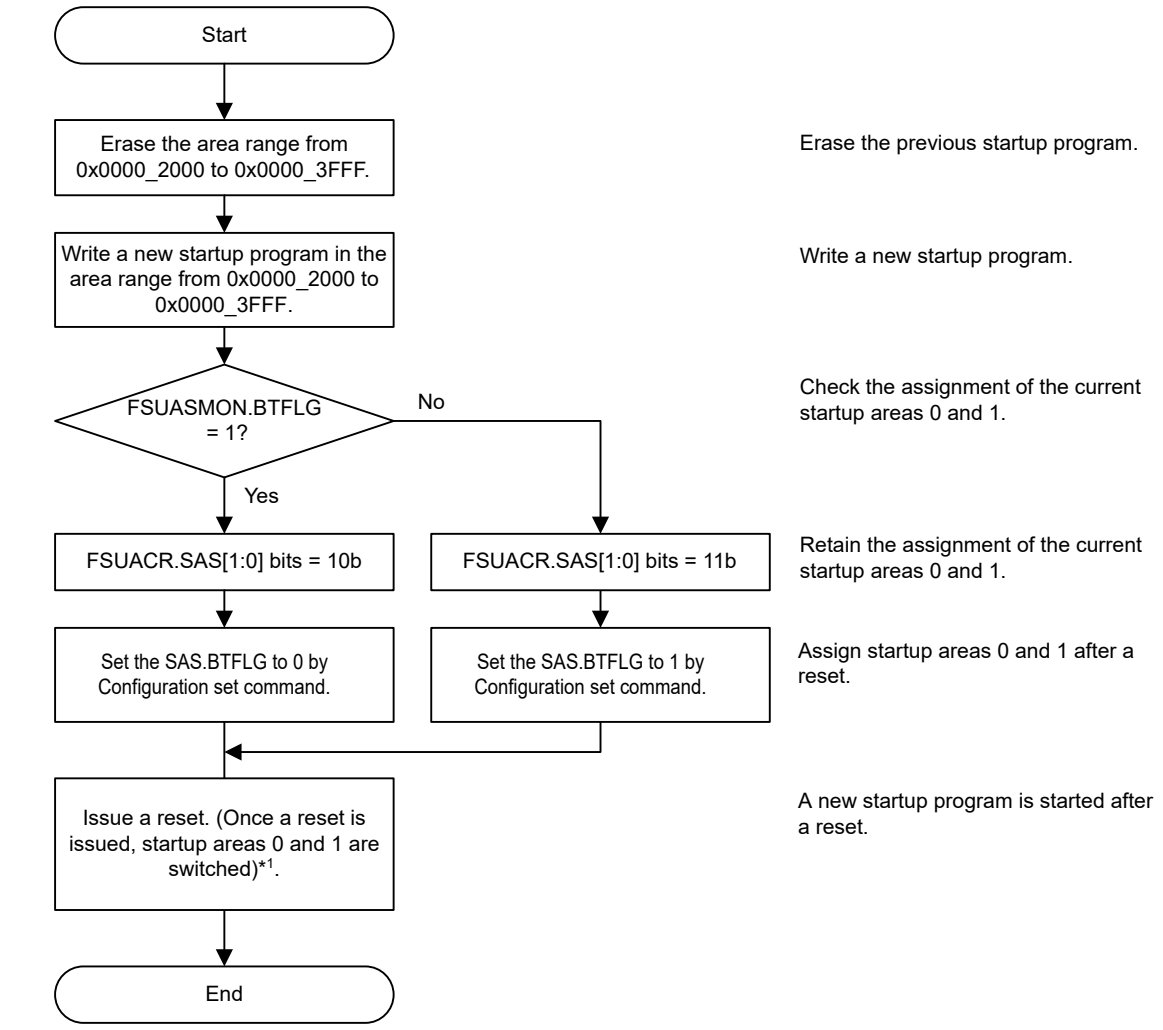



**Figure 44.27 Example 2 of Transitions for Startup Program Protection Settings**




Note 1. For details, see Figure 44.26 and Figure 44.27.

**Figure 44.28 Concept of Protection of the Startup Program**

## 44.12 Security Function

The flash sequencer supports the following security functions:

- Security flag for startup area
- Permanent block protect setting
- Flash memory protection for TrustZone

### 44.12.1 Security Flag for Startup Area Select

The security flag (SAS.FSPR) for the startup area is located in the option-setting memory.

When the SAS.FSPR bit is 0, issuing the configuration set command to change the SAS.BTFLG bit causes the flash sequencer to be in the command-locked state. Also, when the SAS.FSPR bit is 0, it is invalid to write to the Startup Area Select bits SAS[1:0] in the FSUACR register. The SAS.FSPR bit enables protection.

## 44.12.2 Permanent Block Protect Setting

The permanent block protect setting is the clear protection for the block protection setting. User area cannot be permanently updated by the FACI command when the permanent block protect setting is enabled. See section 44.11.1.3. Protection by Block Protect Setting for more details.

The block protect setting and the permanent block protect setting have the write/clear protection against the configuration set command. The flash sequencer does not detect an error when the configuration set command is issued to the write/clear protected settings.

Figure 44.29 and Table 44.22 show the write/clear protection against the block protect setting (BPS[n]) and the permanent protect setting (PBPS[n]). Figure 44.30 and Table 44.23 show the write/clear protection against the block protect setting for secure (BPS_SEC[n]) and permanent protect setting for secure (PBPS_SEC[n]).

Effective permanent block protect setting (PBPS or PBPS_SEC) depends on block protect select (BPS_SEL). For details of permanent block protect setting (PBPS or PBPS_SEC) and block protect select (BPS_SEL), see section 6, Option-Setting Memory.

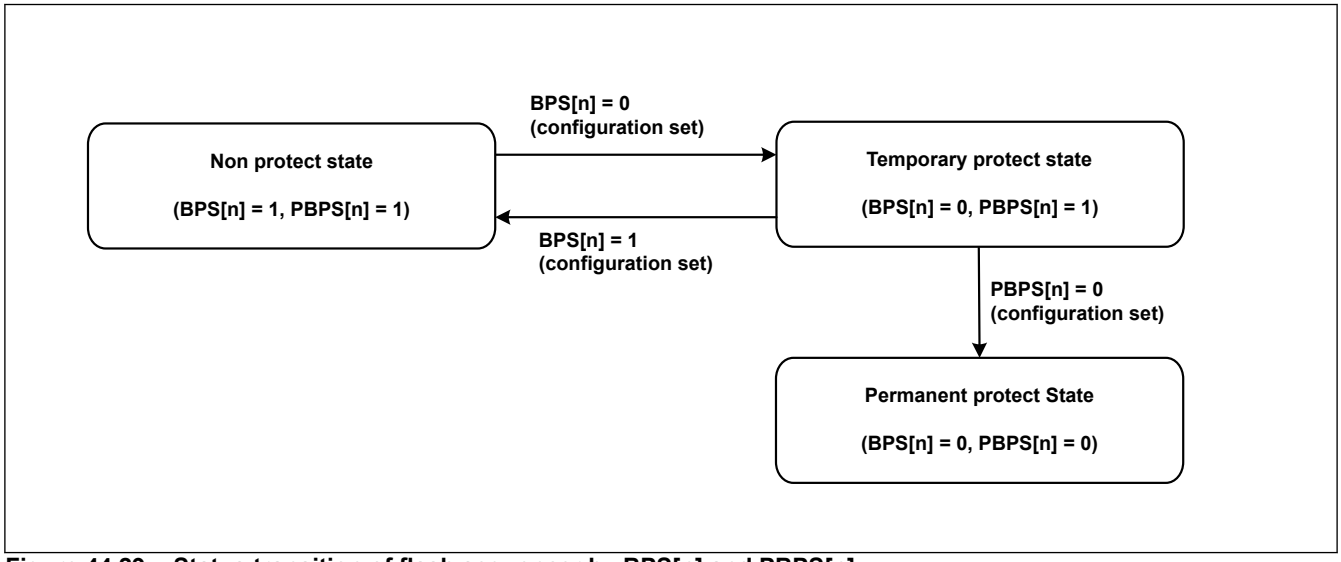



**Figure 44.29 Status transition of flash sequencer by BPS[n] and PBPS[n]**

**Table 44.22 Write/clear protection of BPS[n] and PBPS[n]**

| Current state | | Updatable state by configuration set command | | | |
|---|---|---|---|---|---|
| BPS[n] | PBPS[n] | BPS[n] = 1 | BPS[n] = 0 | PBPS[n] = 1 | PBPS[n] = 0 |
| 1 | 1 | ✓ | ✓ | ✓ | X |
| 1 | 0 | — | — | — | — |
| 0 | 1 | ✓ | ✓ | ✓ | ✓ |
| 0 | 0 | X | ✓ | X | ✓ |

Note:
- ✓ indicates updatable by configuration set command.
- X indicates not updatable by configuration set command (error does not occur).
- — indicates not reaching to this state.

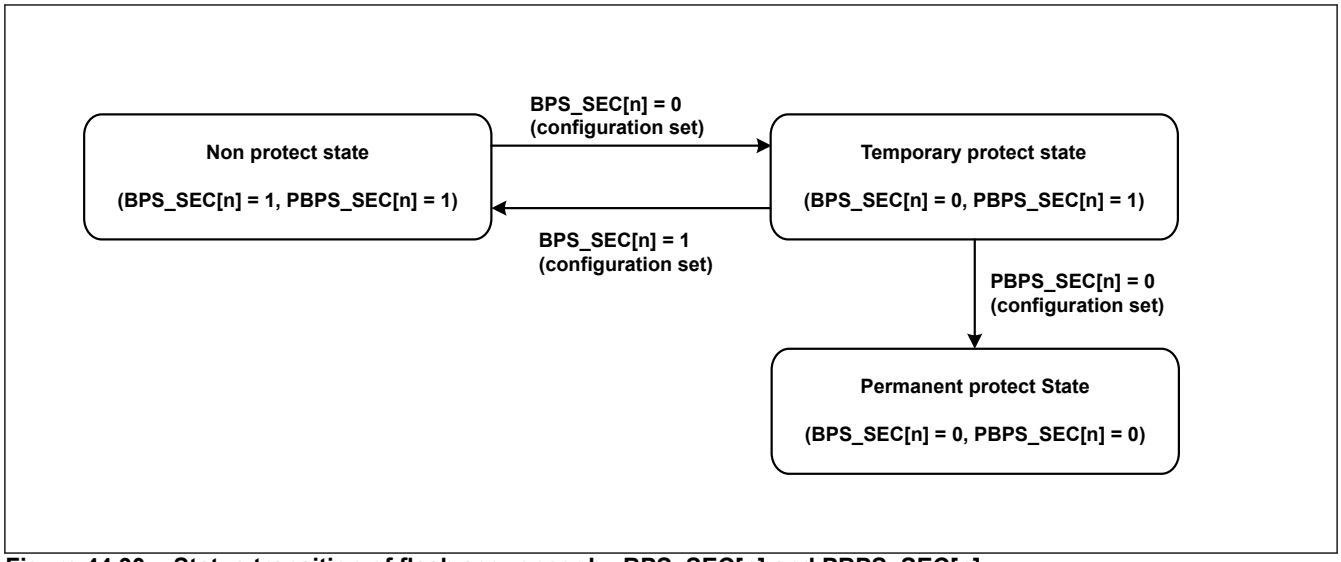



**Figure 44.30 Status transition of flash sequencer by BPS_SEC[n] and PBPS_SEC[n]**

**Table 44.23 Write/clear protection of BPS_SEC[n] and PBPS_SEC[n]**

| Current state | | Updatable state by configuration set command | | | |
|---|---|---|---|---|---|
| BPS_SEC[n] | PBPS_SEC[n] | BPS_SEC[n] = 1 | BPS_SEC[n] = 0 | PBPS_SEC[n] = 1 | PBPS_SEC[n] = 0 |
| 1 | 1 | ✓ | ✓ | ✓ | X |
| 1 | 0 | — | — | — | — |
| 0 | 1 | ✓ | ✓ | ✓ | ✓ |
| 0 | 0 | X | ✓ | X | ✓ |

Note:
- ✓ indicates updatable by configuration set command.
- X indicates not updatable by configuration set command (error does not occur).
- — indicates not reaching to this state.

## 44.12.3 Flash Memory Protection for TrustZone

Information in this section focuses on the flash sequencer operation.

The flash memory provides the following types of protect function against non-secure access:

- Protection for flash memory area (P/E)
- Protection for flash memory area (read)
- Protection for registers
- Protection during FACI command operation
- Code flash P/E mode entry protection

### 44.12.3.1 Protection for Flash Memory Area (P/E)

This function protects the secure region of the code flash and data flash from FACI commands of non-secure access. The condition of protection depends on the FACI command, the access attribution, and the memory boundary setting.

For details of secure region, see section 46, Security Features .

See Table 44.24 for information on protection of the flash memory area (P/E).

**Table 44.24 Protection for the flash memory area (P/E)**

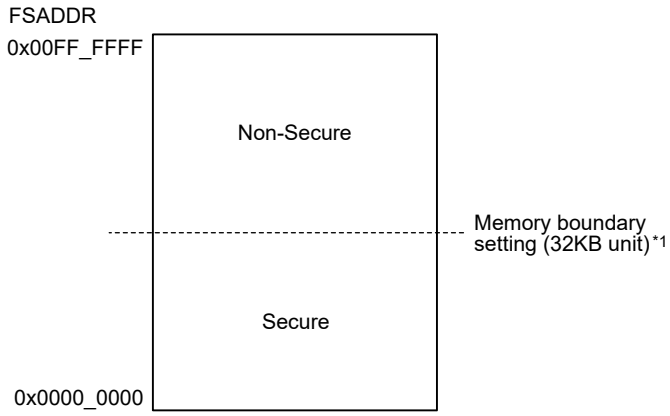
| FACI command | Target area | | Issuing of FACI command by non-secure access | Issuing of FACI command by secure access |
|---|---|---|---|---|
| Program<br>Block erase | Code flash memory | User area (non-secure area) | ✓ | ✓ |
| | | User area (secure area) | X | ✓ |
| | Data flash memory | Data area (non-secure area) | ✓ | ✓ |
| | | Data area (secure area) | X | ✓ |
| Multi block erase<br>Blank check | Data flash memory | Data area (non-secure area) | ✓ | ✓ |
| | | Data area (secure area) | X | ✓ |
| Configuration set | Code flash memory | option-setting memory (non-secure area) | ✓ | ✓ |
| | | option-setting memory (secure area) | X | ✓ |

Note:
- ✓ indicates FACI command operation is not prohibited.
- X indicates FACI command operation is prohibited. Error occurs when the area is selected, and the FACI command is executed.

When the target area of FACI command is the user area of code flash, the flash sequencer compares the FSADDR register setting with the memory boundary setting of the code flash and determines whether the target area is in the secure region.

The memory boundary can be set to 0x0000_0000 to 0x00FF_8000 in 32 KB unit.

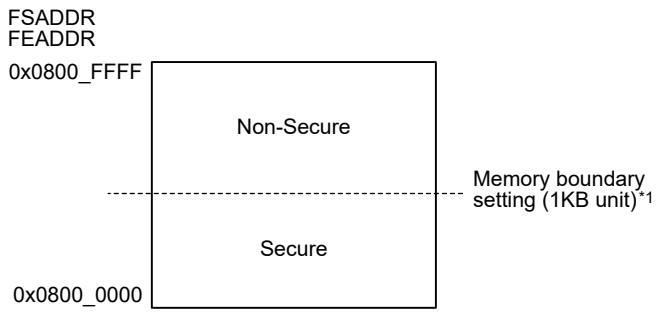
Figure 44.31 shows details of the non-secure/secure attribute of user area in the code flash.




Note 1. Comparing FSADDR with memory boundary setting of user area and judges whether the target area is in secure area.

**Figure 44.31 Secure/non-secure region in user area**

When the target area of the issuing FACI command is the data area of data flash, the flash sequencer compares the FSADDR/FEADDR register setting with the memory boundary setting of the data flash and determines whether the target area is in secure region. The memory boundary can be set to 0x0800_0000 to 0x0800_FC00 in 1 KB unit. Figure 44.32 shows details of the non-secure/secure attribute of data area in the data flash.




Note 1. Comparing FSADDR/FEADDR with memory boundary setting of data area and judges whether the target area is in secure area.

**Figure 44.32 Secure/non-secure region in data area**

See Figure 44.33 for details of non-secure/secure region of option-setting memory. The flash sequencer judges that target area is secure region from the FSADDR register setting.

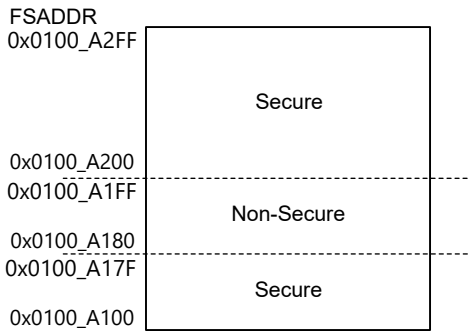



**Figure 44.33 Secure/non-secure region in option-setting memory**

### 44.12.3.2 Protection for Flash Memory Area (Read)

This function protects the secure region of code flash and data flash from non-secure bus access.

For details of secure region, see section 46, Security Features .

### 44.12.3.3 Protection for Register

The flash sequencer registers have write-access protection against non-secure access. Table 44.25 shows details of the protected registers of the flash sequencer.

**Table 44.25 Protected registers of the flash sequencer for TrustZone (1 of 2)**

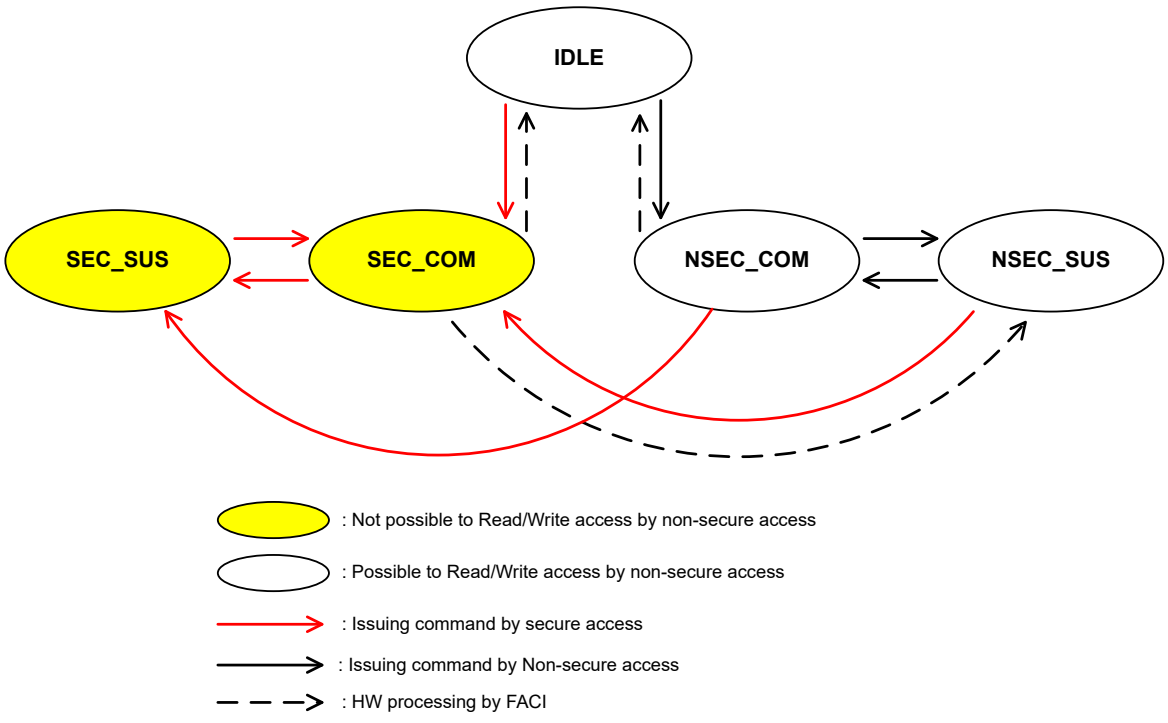
| Protection target register | Security attribute setting | Notes |
|---|---|---|
| FCKMHZ | Security attribution register setting (FSAR.FCKMHZSA) | See section 44.4.4. FSAR : Flash Security Attribution Register |
| FMEPROT | Always secure | See section 44.4.14. FMEPROT : Flash P/E Mode Entry Protection Register |

**Table 44.25 Protected registers of the flash sequencer for TrustZone (2 of 2)**

| Protection target register | Security attribute setting | Notes |
|---|---|---|
| FBPROT1 | Always secure | See section 44.4.16. FBPROT1 : Flash Block Protection for Secure Register |
| FSUACR | Always secure | See section 44.4.27. FSUACR : Flash Startup Area Control Register |
| FACI command-issuing area and all registers of FACI (Base address is FACI) and FWEPROR register | During FACI command processing by secure access | See section 44.12.3.4. Protection during FACI Command Operation |

### 44.12.3.4 Protection during FACI Command Operation

This function protects read/write access to the FACI command-issuing area, including all registers of FACI (Base address is FACI) and FWEPROR register by the non-secure access during the FACI command processing of the secure access. The protect condition includes the suspending period of the program, block erase, or multi block erase command by the P/E suspend command of the secure access. See Figure 44.34 and Table 44.26 for details of the protection during the FACI command operation.




Note: IDLE: State of not executing FACI command
SEC_COM: State of executing FACI command by secure access
SEC_SUS: State of "P/E Suspend" by secure access
NSEC_COM: State of executing FACI command by non-secure access
NSEC_SUS: State of P/E Suspend by non-secure access

**Figure 44.34 State of protection during FACI command operation**

**Table 44.26 Protection during FACI command operation**

| | Flash sequencer is not operating | Program, Block erase, Multi block erase, Blank check, or Configuration set command processing | | Command lock state | | Forced stop command processing | | While suspend Program, Block erase, or Multi block erase command | | Program command processing while suspend Block erase or Multi block erase command by secure access | | Program command processing while suspend Block erase or Multi block erase command by non-secure access | | P/E resume command processing while suspend Program, Block erase, or Multi block erase command by secure access | | P/E Resume command processing while suspend Program, Block erase, or Multi block erase command by non-secure access | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FACI command attribute | — | S | NS | S | NS | S | NS | S | NS | S | NS *1 | S | NS | S | NS *1 | S | NS |
| FRDY bit | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| PRGSPD or ERSSPD bit | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| CMDLK bit | 0 | 0 | 0 | 1 | 1 | 1/0 | 1/0 | 1/0 | 1/0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Non-secure access | ✓ | X | ✓ | ✓ | ✓ | X | ✓ | X | ✓ | X | ✓ | X | ✓ | X | ✓ | X | ✓ |

Note:
- S indicates the FACI command by the secure access.
- NS indicates the FACI command by the non-secure access.
- ✓ indicates read/write access is possible by the non-secure access.
- X indicates read/write access is not possible by the non-secure access. Write data is ignored and read data is always 0.

Note 1. The FACI command issued by the non-secure access is not allowed.

Code flash programming/erasure can be protected by the FMEPROT register of secure function. Therefore, it does not assume that secure function issues P/E suspend command during code flash programming/erasure of non-secure function.

Data flash programming/erasure of non-secure can be suspended by secure function. If secure function issues P/E suspend command during data flash programming/erasure of non-secure function, secure function should issue P/E resume command. When secure function issues P/E resume command, secure function should notify non-secure function that data flash programming/erasure is complete and return to non-secure function. See Figure 44.35 and Figure 44.36 in example of issuing P/E suspend of secure function during programming/erasure of non-secure function.

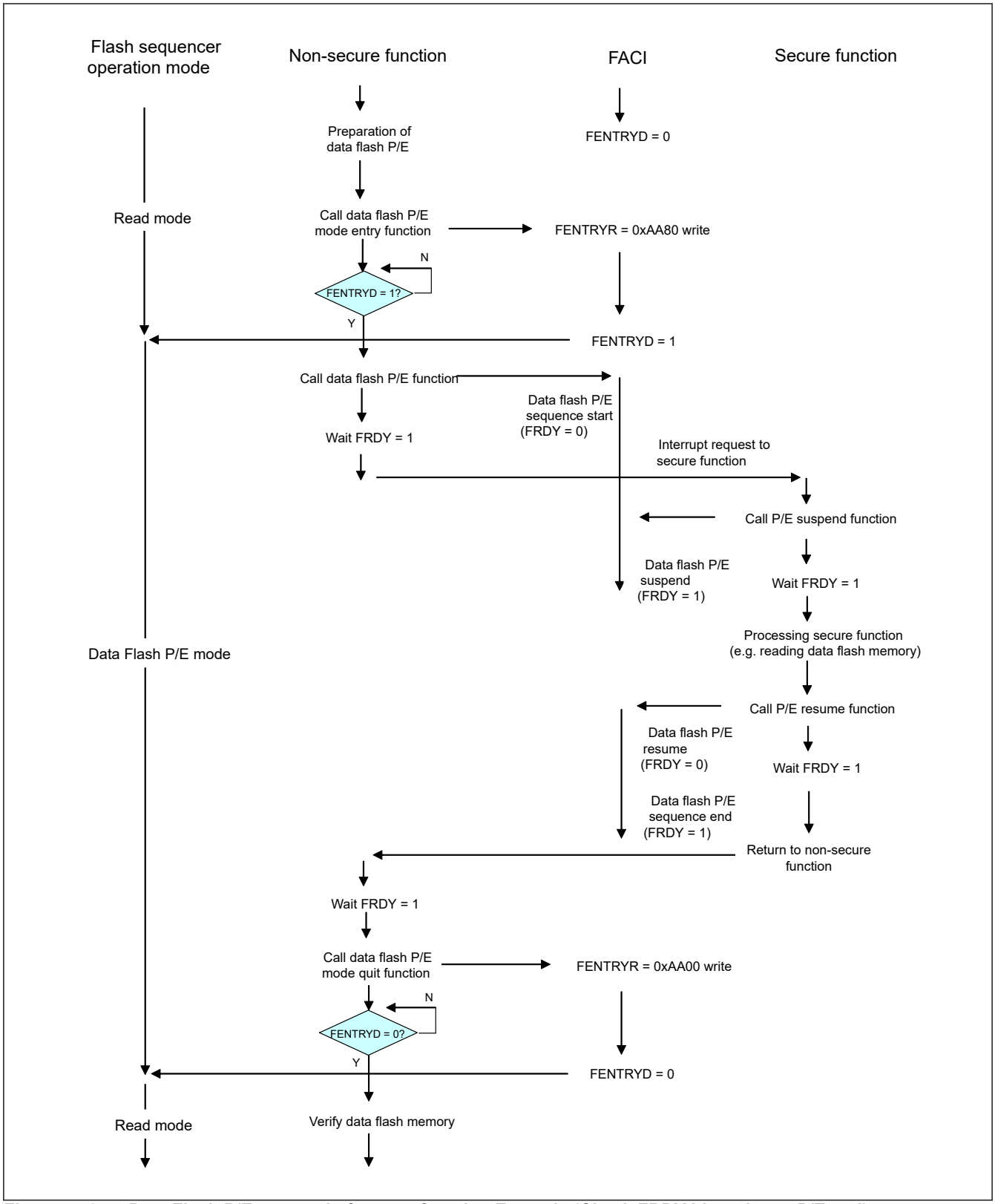



**Figure 44.35 Data Flash P/E suspend of secure function Example (Check FRDY bit to detect P/E end)**

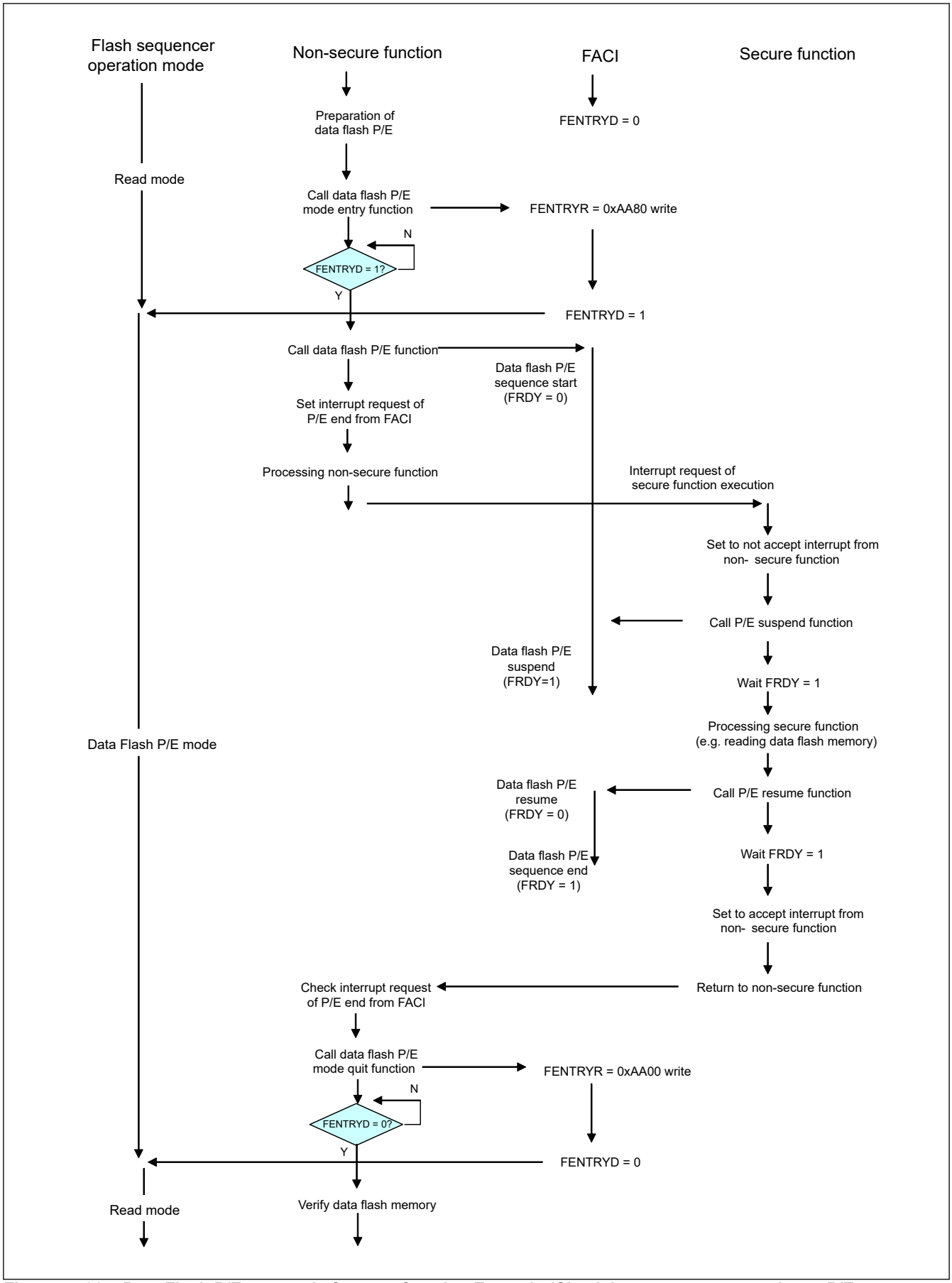



**Figure 44.36 Data Flash P/E suspend of secure function Example (Check interrupt request to detect P/E end)**

#### 44.12.3.5 Code Flash P/E Mode Entry Protection

The flash sequencer has protection function of code flash P/E by the FMEPROT register for the secure developer. Secure function can prevent disturbance of reading code flash memory by this protection function. See section 44.4.14. FMEPROT : Flash P/E Mode Entry Protection Register.

For applications that do not require non-secure region programming/erasure other than from secure function, it is recommended to always disable non-secure function of code flash programming/erasure by enabling the protection function of FMEPROT register.

For details, see Figure 44.37 of the code flash P/E sequence example by non-secure function.

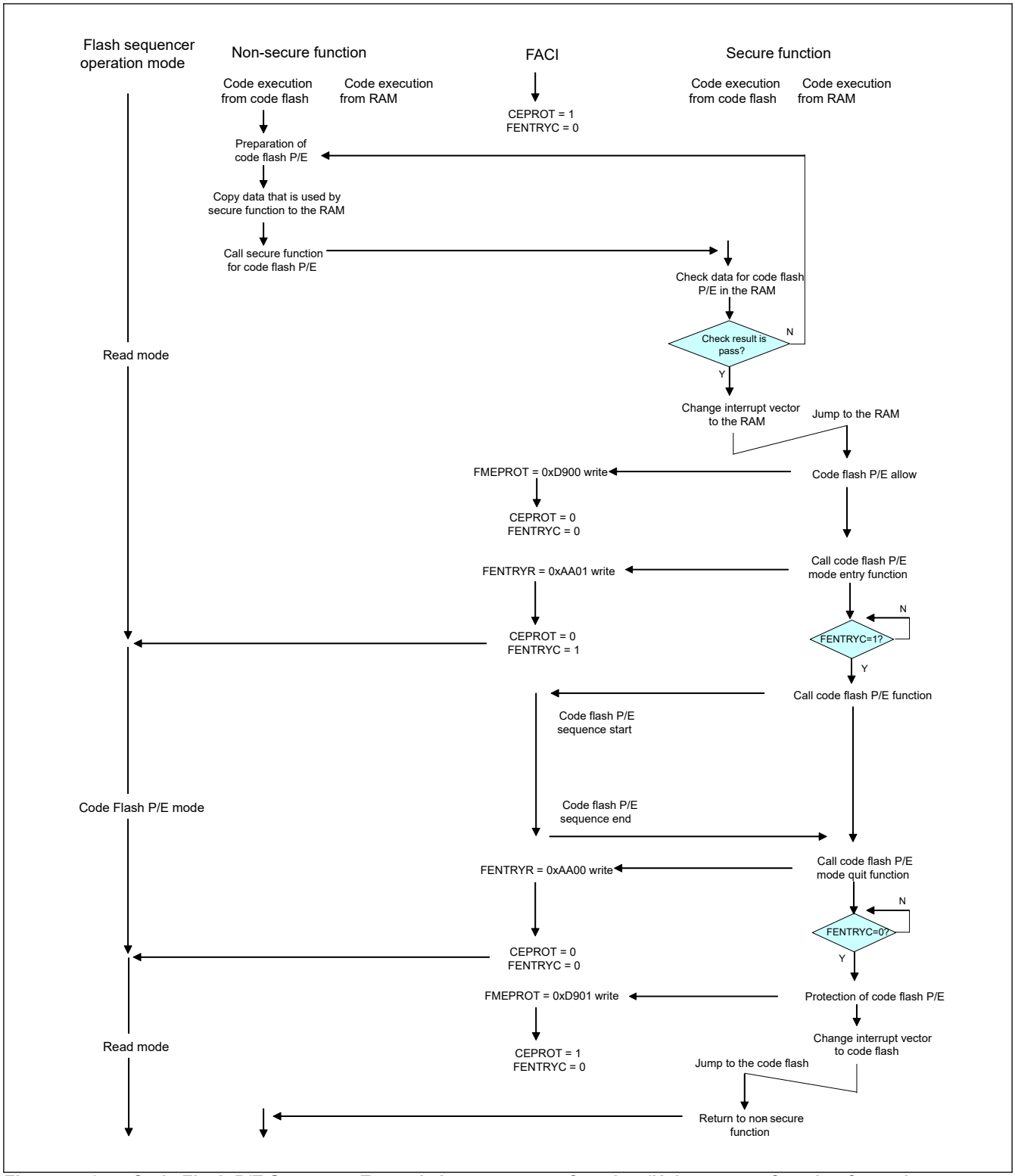



**Figure 44.37 Code Flash P/E Sequence Example by non-secure function (Using secure function for code flash P/E)**

## 44.13 Boot Mode

There are two serial programming modes; the boot mode (for the SCI interface) with SCI9 and the boot mode (for the USB interface) with USBFS. Table 44.27 lists the I/O pins used in boot mode. Table 44.28 lists the available communication interface used in the boot mode.

**Table 44.27 I/O Pins Used in Boot Mode**

| Pin Name | I/O | Mode to be Used | Use |
|---|---|---|---|
| MD | Input | Boot mode (for the SCI interface) Boot mode (for the USB interface) | Selection of operating mode |
| P110/RXD9 | Input | Boot mode (for the SCI interface) | For host communication (to receive data through SCI) |
| P109/TXD9 | Output | | For host communication (to transmit data through SCI) |
| USB_DP, USB_DM | I/O | Boot mode (for the USB interface) | Data input/output of USB |
| USB_VBUS | Input | | Detection of connection and disconnection of USB cables |

**Table 44.28 Available Communication Interface Used in Boot Mode**

| | | | |
|---|---|---|---|
| Main clock oscillator or external clock is connected | Yes | No | No |
| Sub clockoscillator is connected*1 | Yes or No | Yes | No |
| Available interface | SCI or USB | SCI or USB | SCI |
| Tool connection time*2 | Up to 1 second | Up to 2 seconds | Up to 3 seconds |

Note 1. The drive capability of the sub clock oscillator is set to standard by SOMCR.SODRV bit. Note that if you use the crystal corresponding the low drive capability on your board, the crystal may not oscillate in the boot mode.
Note 2. See the boot firmware manual for the detail connection time.

## 44.13.1 Boot Mode (for the SCI Interface)

In boot mode (for the SCI interface), the host sends control commands and data for programming, and the flash memory is programmed or erased accordingly. An on-chip SCI handles transfer between the host and this MCU in asynchronous mode. Tools for transmission of control commands and the data for programming must be prepared in the host.

When this MCU is activated in boot mode (for the SCI interface), the program on the dedicated area the MCU is executed. The boot program automatically adjusts the bit rate of the SCI and controls programming/erasure by receiving control commands from the host.

Figure 44.38 shows the system configuration for operations in boot mode (for the SCI interface).

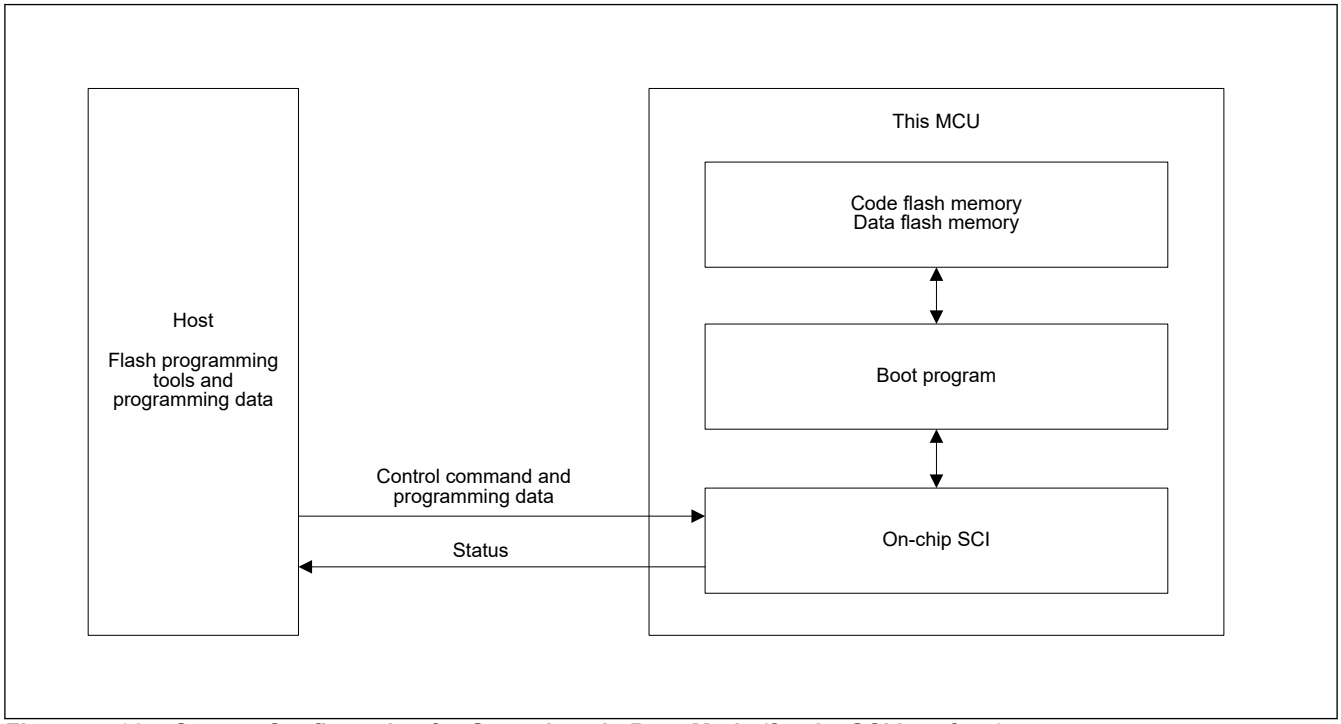



**Figure 44.38 System Configuration for Operations in Boot Mode (for the SCI Interface)**

### 44.13.2 Boot Mode (for the USB Interface)

In boot mode (with the USB interface), the flash memory can be programmed or erased by sending control commands and program data from the host. An on-chip USB is used for communications between the host and this MCU. The host requires tools for sending control commands and data for programming. Figure 44.39 shows the configuration of a system for use in boot mode (for the USB interface). The USB cable must be connected on reset release.

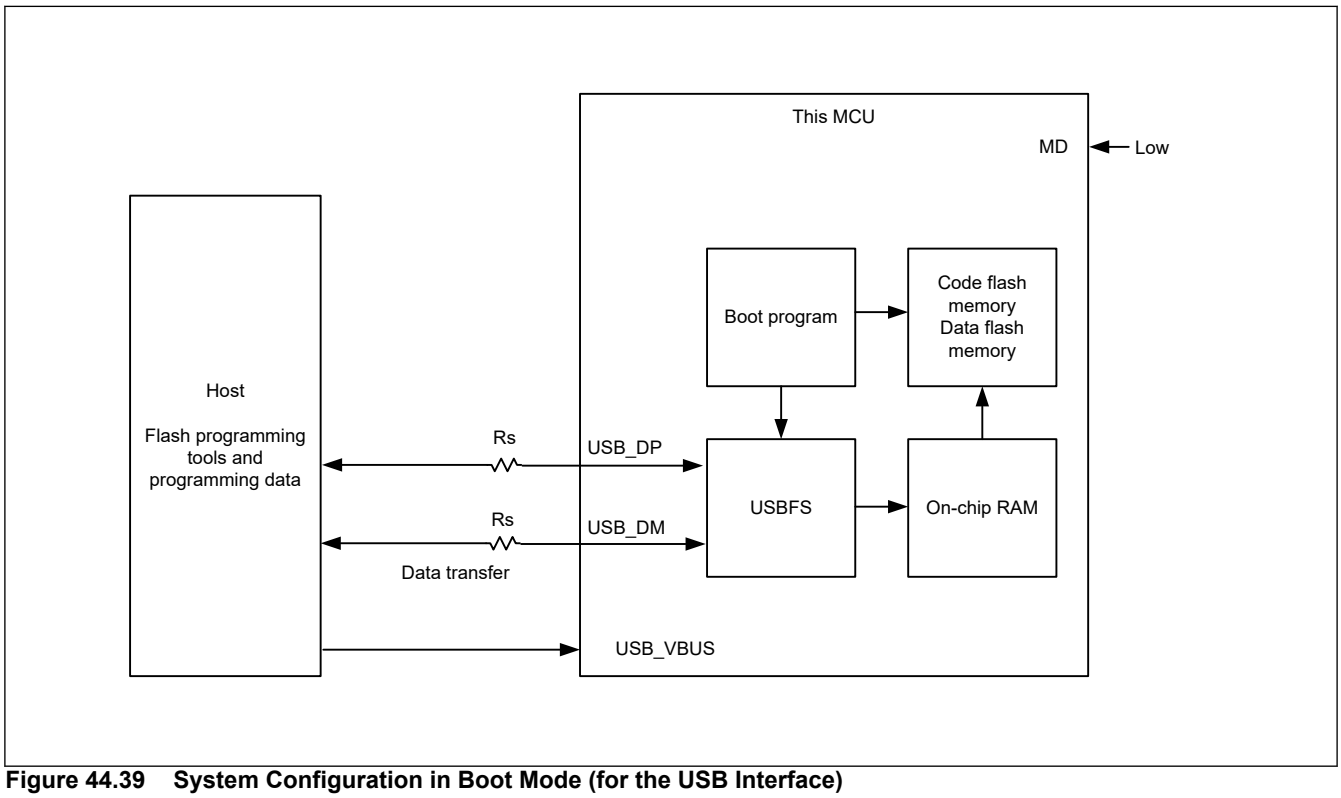



**Figure 44.39 System Configuration in Boot Mode (for the USB Interface)**

## 44.14 Using the Serial Programmer for Rewriting

A serial programmer can be used to rewrite flash memory in boot mode.

### (1) Serial Programming

This MCU is mounted on the system board at the time of serial programming. Providing a connector to the board enables rewriting of this MCU by the serial programmer to proceed.

### 44.14.1 Environments for Serial Programming

The recommended environments for rewriting the flash memory of the MCU with data are described below.

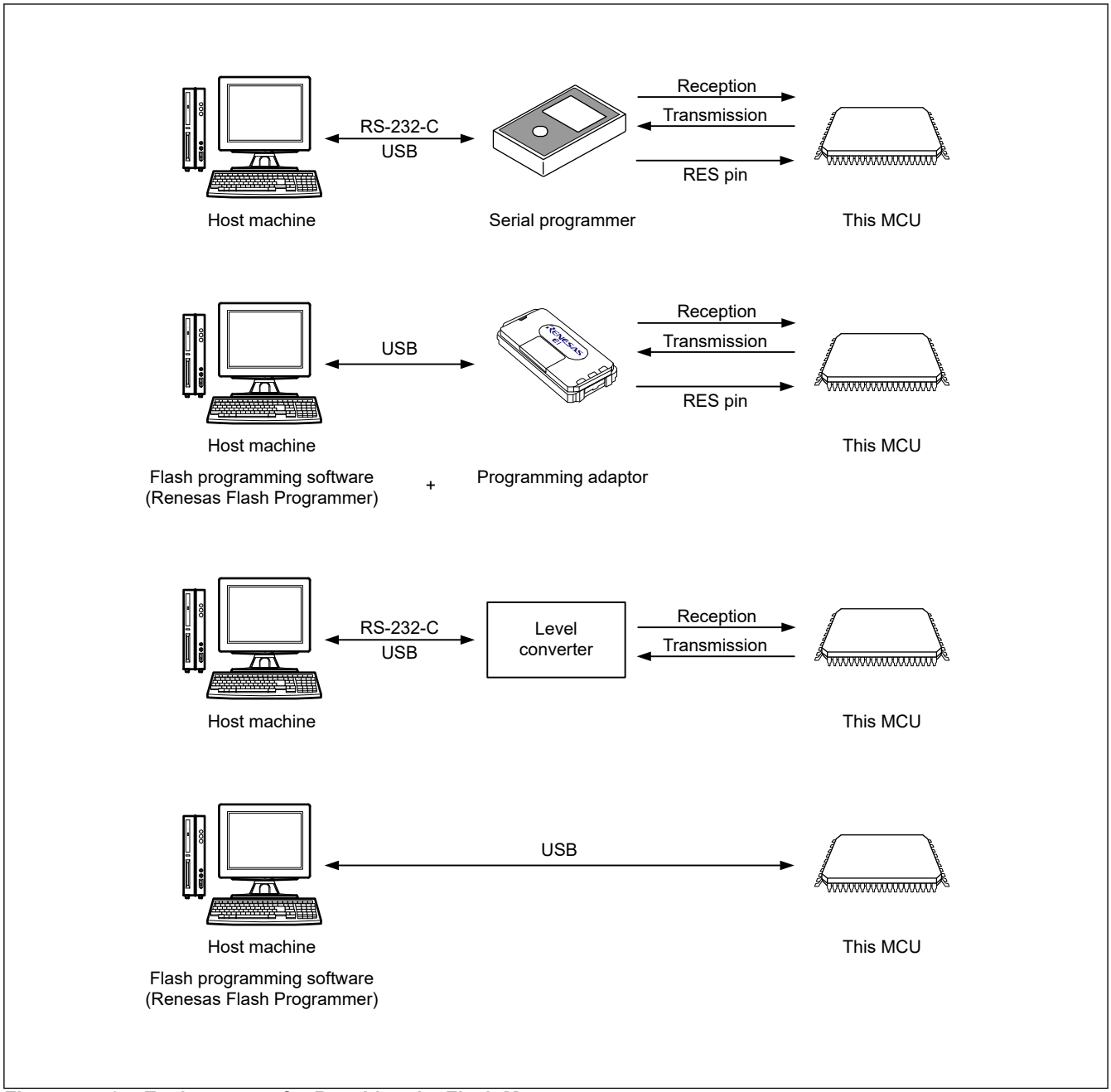



**Figure 44.40 Environments for Rewriting the Flash Memory**

## 44.15 Programming through Self-Programming

### 44.15.1 Overview

This MCU supports programming of the flash memory by the user program itself. The FACI commands can be used with user programs for writing to the flash memory. This allows upgrading of user programs and rewriting of constant data fields.

The program for rewriting must be transferred to the internal RAM in advance when the BGO is not available or when rewriting the option-setting memory.

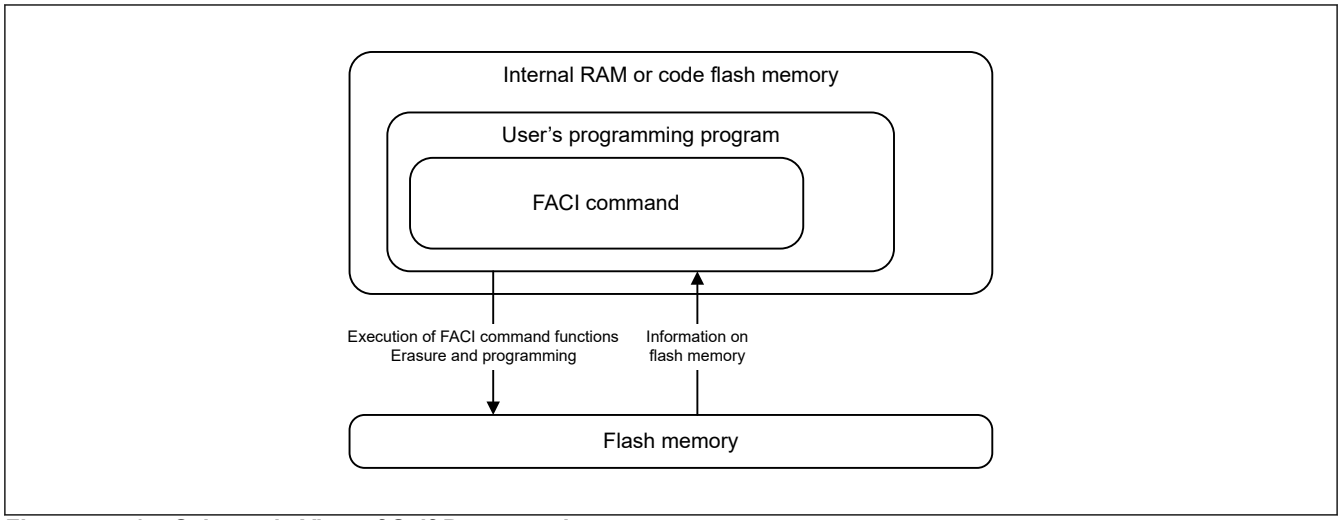



**Figure 44.41 Schematic View of Self-Programming**

For comprehensive information on the self-programming, see section 44.9. FACI Commands.

### 44.15.2 Background Operation

The background operation (BGO) can be used to execute the flash rewrite routine on the code flash memory when the data flash memory is rewritten.

Background operations can be used when the combination of the flash memory for rewriting and the flash memory for reading is any of those listed below.

**Table 44.29 Conditions under which Background Operation is Usable**

| | Range for rewriting | Range for reading |
|---|---|---|
| Common | Code flash memory | Data flash memory |
| | Data flash memory | Code flash memory |

## 44.16 Reading Flash Memory

### 44.16.1 Reading Code Flash Memory

Special settings are not required to read code flash memory after release from the reset state. Data can simply be read out through access to addresses in the code flash memory.

When reading code flash memory that has been erased but not yet been programming again (i.e. that is in the non-programmed state), all bits are read as 1.

### 44.16.2 Reading Data Flash Memory

Special settings are not required to read data flash memory after release from the reset state. Data can simply be read out through access to addresses in the data flash memory.

Values read from data flash memory that has been erased but not yet been programming again (i.e. that is in the non-programmed state) are undefined. Use blank checking when you need to confirm that an area is in the non-programmed state.

### 44.16.3 Access Cycle

**Table 44.30 Code Flash Memory (1 of 2)**

| Flash Cache Operation | FLWT Register Setting | Read cycle (ICLK) |
|---|---|---|
| enable and hit | — | 1 |

**Table 44.30 Code Flash Memory (2 of 2)**

| Flash Cache Operation | FLWT Register Setting | Read cycle (ICLK) |
|---|---|---|
| disable or miss | 0x00 | 1 |
| | 0x01 | 2 |

**Table 44.31 Data Flash Memory**

| FCKMHZ Register Setting | Read cycle |
|---|---|
| 0x00 to 0x09 | Min: 3 FCLK<br>Max: (n - 1) ICLK + 3 FCLK |
| 0x0A to 0x13 | Min: 4 FCLK<br>Max: (n - 1) ICLK + 4 FCLK |
| 0x14 to 0x1D | Min: 5 FCLK<br>Max: (n - 1) ICLK + 5 FCLK |
| 0x1E to 0x27 | Min: 6 FCLK<br>Max: (n - 1) ICLK + 6 FCLK |
| 0x28 to 0x31 | Min: 7 FCLK<br>Max: (n - 1) ICLK + 7 FCLK |
| 0x32 | Min: 8 FCLK<br>Max: (n - 1) ICLK + 8 FCLK |

Note: When the frequency ratio of ICLK : FCLK is n : 1

## 44.17 Usage Notes

### (1) Reading Area Where Programming/Erasure was Interrupted and Area Targeted for Suspension

The data stored in the area where programming or erasure has been suspended or the area where programming or erasure has been suspended by using the suspend command are undefined. To avoid faulty operation caused by reading undefined data, take care not to fetch instructions or read data from areas where programming or erasure was suspended and where programming or erasure was suspended by using the suspend command.

### (2) Suspension During Programming/Erasure

When processing of programming/erasure is stopped by issuing the P/E suspend command, the programming/erasure processing can be resumed by issuing the P/E resume command. If the flash sequencer enters the command-locked state for any reason and issues the forced stop command after the suspended processing is normally completed and the ERSSPD flag or PRGSPD flag is set to 1, the suspended processing cannot be resumed. In addition, the values in the area where the processing was suspended are not guaranteed. Erase that area

### (3) Prohibition of Additional Programming

Programming a given area of the code flash memory or data flash memory twice is not possible. To program the code flash memory or data flash memory where has been programed, erase the target area. Programming can be added to the option-setting memory.

### (4) Resets During Programming/Erasure, or Blank Checking

In the case of a reset due to the signal on the RES pin during programming/erasure, or blank checking of the flash memory, wait for at least $t_{RESW}$ (see section 47, Electrical Characteristics) of the reset input period once the operating voltage is within the range stipulated in the electrical characteristics, then release the device from the reset state.

### (5) Allocation of Vectors for Interrupts and Other Exceptions During Programming/Erasure

Generation of an interrupt or other exception during programming/erasure may lead to fetching of the vector from the code flash memory. Under conditions where BGO cannot be used, set the address of the vector to an address that is not in the code flash memory. Alternatively, make sure that no handling of interrupts or exceptions proceeds during programming/erasure.

### (6) Items Prohibited During Programming/Erasure, or Blank Checking

High voltage is applied to the flash memory during programming/erasure, or blank checking. To prevent damage to the flash memory, do not perform the following operations.

- Have the operating voltage from the power supply go beyond the permitted range.
- Change the FWEPROR.FLWE[1:0]bits.
- Change the OPCCR.OPCM[2:0] and SOPCCR.SOPCM bits.
- Change the SCKDIVCR.FCK[2:0]bits.
- Change the SCKSCR.CKSEL[2:0]bits.
- Transition to the software standby mode, or Deep Software Standby mode.

### (7) Programming/Erasure in Low-Speed Modes and Subosc-Speed Mode

Do not programming/erasure the flash memory when low-speed mode or subosc-speed mode is selected with the operating power control register (OPCCR or SOPCCR).

### (8) Emulator Connection

Renesas provides the emulator which supports both debugging using SWD or JTAG communication and serial programming using SCI communication. This emulator makes it easy to switch between debugging and serial programming.

Table 44.32 shows the pinout of 10 pin or 20 pin socket pinouts when using this emulator. The pinout of SWD and JTAG is ARM standard, and MD, TXD, RXD pins are added for the serial programming using SCI communication.

The serial programming interface must be used to program the TrustZone IDAU boundary register settings.

It is recommended to connect P300/SWCLK/TCK and P201/MD pins using wired OR circuit on the board to use both debugging and serial programming

**Table 44.32 Pin assign for emulator**

| Pin No. | SWD | JTAG | Serial Programming using SCI |
|---|---|---|---|
| 1 | VCC | VCC | VCC |
| 2 | P108/SWDIO | P108/TMS | NC |
| 4 | P300/SWCLK<br>Wired OR with P201/MD | P300/TCK<br>Wired OR with P201/MD | P201/MD |
| 6 | P109/SWO/TXD9 | P109/TDO/TXD9 | P109/TXD9 |
| 8 | P110/RXD9 | P110/TDI/RXD9 | P110/RXD9 |
| 9 | GNDdetect | GNDdetect | GNDdetect |
| 10 | nRESET | nRESET | nRESET |
| 12 | P214/TCLK | P214/TCLK | NC |
| 14 | P211/TDATA[0] | P211/TDATA[0] | NC |
| 16 | P210/TDATA[1] | P210/TDATA[1] | NC |
| 18 | P209/TDATA[2] | P209/TDATA[2] | NC |
| 20 | P208/TDATA[3] | P208/TDATA[3] | NC |
| 3, 5, 15, 17, 19 | GND | GND | GND |
| 7 | NC | NC | NC |
| 11, 13 | NC | NC | NC |

# 45. Internal Voltage Regulator

## 45.1 Overview

The MCU includes one internal voltage regulator:

- Linear regulator (LDO)

This regulator supplies voltage to all internal circuits and memory except for I/O, analog, USB, and battery backup power domains.

## 45.2 Operation

Table 45.1 lists the LDO mode pin settings, and Figure 45.1 shows the LDO mode settings. In LDO mode, the internal voltage is generated from VCC.

**Table 45.1 LDO mode pin**

| Pins | Setting descriptions |
|---|---|
| All VCC | • Connect each pin to the system power supply.<br>• Connect each pin to VSS through a 0.1-μF multilayer ceramic capacitor. Place the capacitor close to the pin. |
| VCL | Connect the each pin to VSS through a 0.22-μF multilayer ceramic capacitor. Place the capacitor close to the pin. |

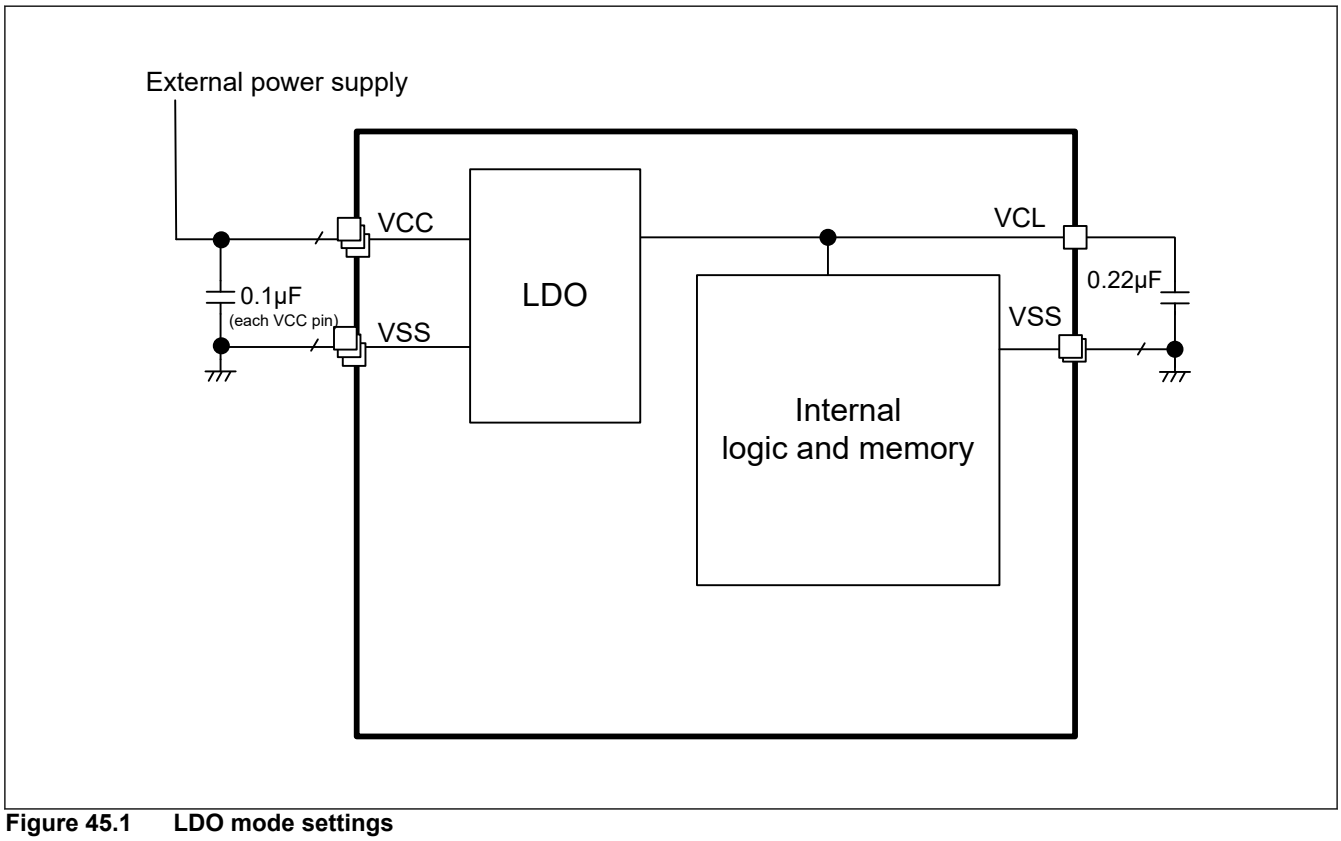



**Figure 45.1 LDO mode settings**

# 46. Security Features

## 46.1 Features

- ARMv8-M TrustZone security
  - Eight regions IDAU for memory space
    - Up to three regions for the code flash
    - Up to two regions for the data flash
    - Up to three regions for the SRAM
    - IDAU setting is common for the CPU, DMAC, and DTC
  - SAU is not implemented
  - Secure or Non-secure region for the Standby SRAM
  - Secure or Non-secure region for the VBATT backup registers
  - Individual Secure or Non-secure security attribution for each peripheral
  - Some peripherals support both Secure and Non-secure security attributions
- Device lifecycle management
- Three debug access levels
  - DBG2: The debugger connection is allowed, and no restriction to access memories and peripherals
  - DBG1: The debugger connection is allowed, and restricted to access only non-secure memory regions and peripherals
  - DBG0: The debugger connection is not allowed
- Key injection
- Cryptographic accelerator
  - See section 36, Secure Cryptographic Engine (SCE9)
- Secure pin multiplexing
  - All I/O port pins can be configured individually as secure or non-secure
  - Pin functions of SCI3, SPI0, IIC0, GPT321 and GPT165 can be configured as secure pin
  - See section 19, I/O Ports

## 46.2 Arm TrustZone Security

### 46.2.1 Arm TrustZone Technology

Arm TrustZone technology divides the system and the application into Secure and Non-secure domains. Secure application can access both Secure and Non-secure memory and resources. Non-secure application can only access Non-secure memory and resources.

The system starts up in Secure state by default. The security state of CPU can be either Secure or Non-secure.

### 46.2.2 Memory Security Attribution

The code flash, the data flash, and the SRAM are divided into Secure (S), Non-secure (NS) and Non-secure callable (NSC) regions. These memory security attributions are set into the nonvolatile memory by the serial programming command when the device lifecycle is SSD state. These memory security attributions are loaded into the IDAU and the memory controller before application execution. These memory security attributions cannot be updated by application but can be through the dedicated registers.

The code flash can be divided in up to three regions. The data flash can be divided in up to two regions. The SRAM can be divided in up to three regions. Figure 46.1 shows the memory mapping. Table 46.1 shows the size of memory region.

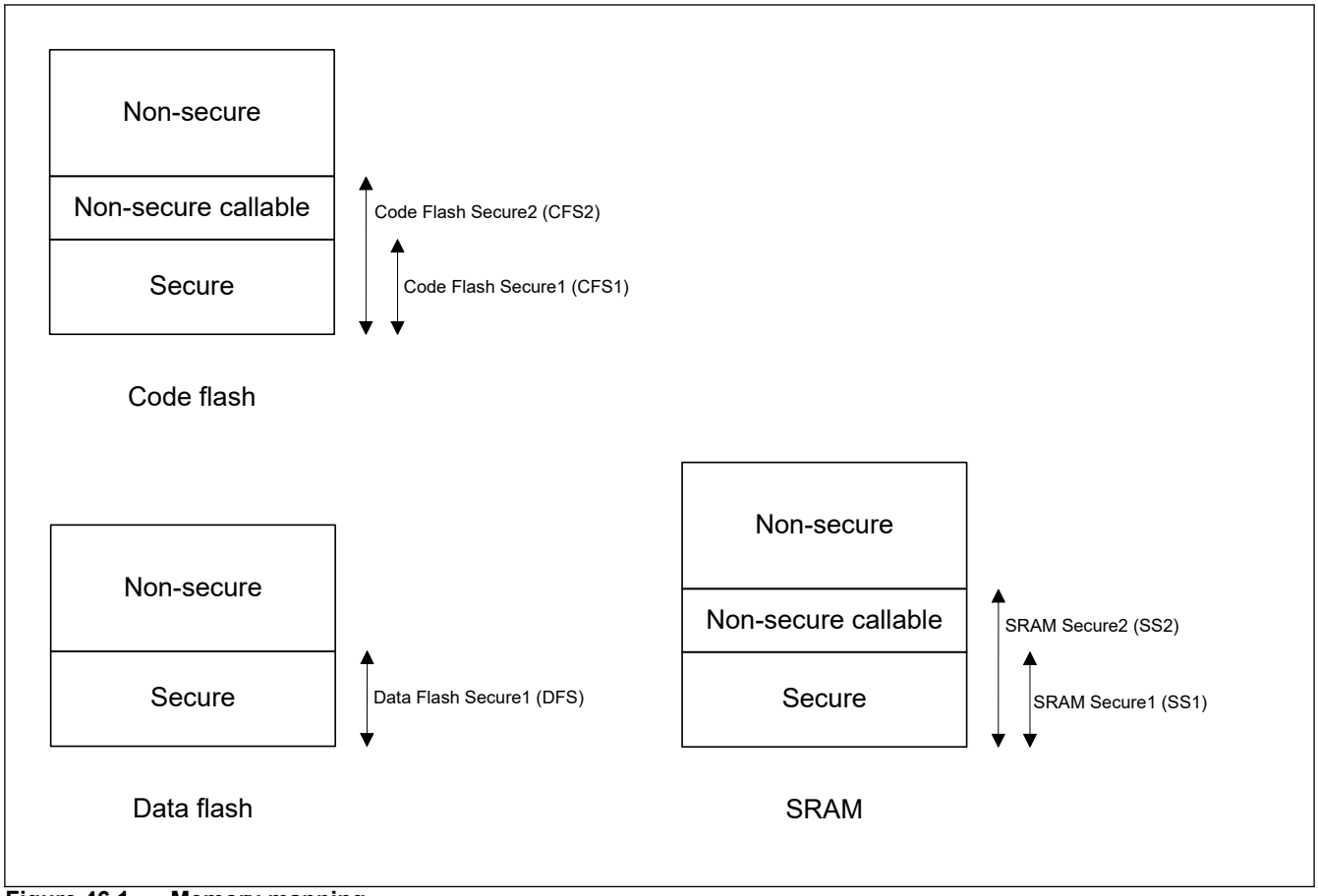



**Figure 46.1 Memory mapping**

**Table 46.1 Memory Region Size**

| Memory Region | Start Address | Size |
|---|---|---|
| Code flash secure | 0x0000_0000 | CFS1 × 1 KB |
| Code flash non-secure callable | CFS1 × 1 KB | CFS2 × 32 KB - CFS1 × 1 KB |
| Code flash non-secure | CFS2 × 32 KB | Code flash size - CFS2 × 32 KB |
| Data flash secure | 0x0800_0000 | DFS × 1 KB |
| Data flash non-secure | 0x0800_0000 + DFS × 1 KB | Data flash size - DFS × 1 KB |
| SRAM secure | 0x2000_0000 | SS1 × 1 KB |
| SRAM non-secure callable | 0x2000_0000 + SS1 × 1 KB | SS2 × 8 KB - SS1 × 1 KB |
| SRAM non-secure | 0x2000_0000 + SS2 × 8 KB | SRAM size - SS2 × 8 KB |

The Standby SRAM is divided 8 regions. Security attribution can be set for each region, but both secure region and non-secure region must be contiguous. In other words, the Standby SRAM can have one contiguous secure region and one contiguous non-secure region. The Standby SRAM security attribution is set to the dedicated register by the secure application. See section 43, Standby SRAM for the details.

The VBATT backup register is divided 8 regions. Security attribution can be set for each region, but both secure region and non-secure region must be contiguous. In other words, the VBATT backup register can have one contiguous Secure region and one contiguous Non-secure region. The VBATT backup register security attribution is set to the dedicated register by the secure application. See section 11, Battery Backup Function for the details.

Table 46.2 shows the access permission of the memory.

**Table 46.2 Access Permission of Memory**

| Memory | Secure access | Non-secure access |
|---|---|---|
| Code flash, Data flash, SRAM configured as Secure or Non-secure callable | allowed | Write ignored / Read ignored<br>TrustZone Access error is generated |
| Code flash, Data flash, SRAM configured as non-secure | allowed | allowed |
| Standby SRAM, VBATT backup register configured as Secure | allowed | Write ignored / Read 0x00<br>TrustZone Access error is not generated |
| Standby SRAM, VBATT backup register configured as Non-secure | allowed | allowed |

### 46.2.3 Peripheral Security Attribution

Each peripheral can be configured to be Secure or Non-secure.

Peripherals are divided into two types.

Type-1 peripherals has the one security attribution. Access to all registers is controlled by one security attribution. Type-1 peripheral security attribution is set to the PSARx (x = B to E) register by the secure application.

Type-2 peripherals has the security attribution for each register or for each bit. Access to each register or bit field is controlled according to these security attributions. Type-2 peripheral security attribution is set to the Security Attribution register in each module by the secure application. For the Security Attribution register, see sections in the user manual for each peripheral.

Table 46.3 shows the classification of peripheral type.

**Table 46.3 Peripheral Type Classification**

| Type | Peripheral |
|---|---|
| Type-1 | SCI, SPI, USBFS, CAN, IIC, SCE9, DOC, SDHI, SSIE, CTSU, CRC, CAC, TSN, ADC12, DAC12, POEG, AGT, GPT, RTC, IWDT, WDT |
| Type-2 | System control (Resets, LVD, Clock Generation Circuit, Low Power Modes, Battery Backup Function), FLASH CACHE, SRAM controller, DMAC, DTC, ICU, MPU, BUS, Security setting, ELC, I/O ports |
| Always Non-secure | QSPI |

Table 46.4 shows the access permission of type-1 peripherals. The access permission of type-2 peripherals is different by peripherals. See section Register Description of each peripherals.

**Table 46.4 The access permission of type-1 peripherals**

| Permission | Secure access | Non-secure access |
|---|---|---|
| Peripheral configured as secure | allowed | Write ignored / Read ignored<br>TrustZone Access error is generated |
| Peripheral configured as non-secure | allowed | allowed |

### 46.2.4 Flash Sequencer Security Attribution

The flash sequencer is used to program or erase the flash.

The flash sequencer has the special security attribution. Table 46.5 shows the access permission of flash sequencer.

**Table 46.5 Access Permissions of Flash Sequencer**

| | Secure access | Non-secure access |
|---|---|---|
| FACI command issuing area | allowed | When the FACI command is issued to the secure region of code flash, data flash and option-setting memory<br>● Issued FACI command is invalid<br>● Flash sequencer error is generated<br>When the FACI command is issued to the non-secure region of code flash, data flash and option-setting memory<br>● Issued FACI command is valid |
| FBPROT1, FSUACR, FMEPROT registers | allowed | Write ignored / Readable<br>TrustZone Access error is not generated |
| FCKMHZ register | allowed | Configured by Flash Security Attribution register<br>When configured as Secure,<br>● Write ignored / Readable<br>● TrustZone Access error is not generated.<br>When configured as Non-secure<br>● allowed |
| Other registers | allowed | During programming/erasure or during suspend programming/erasure by secure application<br>● Write ignored / Read 0x00<br>● TrustZone Access error is not generated<br>In other state<br>● allowed |

## 46.2.5 Address Space Security Attribution

Table 46.6 shows the security attribution of the address space.

**Table 46.6 Address Space Security Attribution**

| Region | Attribution |
|---|---|
| Code flash secure | Secure |
| Code flash non-secure callable | Non-secure callable |
| Code flash non-secure | Non-secure |
| Data flash secure | Secure |
| Data flash non-secure | Non-Secure |
| SRAM secure | Secure |
| SRAM non-secure callable | Non-secure callable |
| SRAM non-secure | Non-secure |
| Peripherals | Exempt |
| Other area | Exempt |

Note: Exempt: No check will be done. All bus transactions are propagated.

## 46.2.6 TrustZone Access Error

Table 46.7 shows the behavior when TrustZone access error. The behavior varies depending on the master or slave area to be accessed.

**Table 46.7 The Behavior When TrustZone Access Error**

| Area | CPU | DMAC/DTC |
|---|---|---|
| Code flash, Data flash, SRAM | Detect SecureFault exception*2 | ● Transfer does not start<br>● Occur NMI or reset*1<br>● Occur interrupt (DMA_TRANSERR) |
| Other area | ● Detect BusFault exception*2 *3<br>● Occur NMI or reset*1*2 *3 | ● Stop transfer*4<br>● Occur NMI or reset *1 *4<br>● Occur interrupt (DMA_TRANSERR)*4 |

Note 1. NMI or reset is selected with TZFOAD.OAD bit.

Note 2. When TrustZone access error occurs by the debugger access, exception, NMI, or reset does not occurs.Only the error response is returned.
Note 3. These error behaviors does not occur for write access to the PHBIU or PLBIU address space which memory attribute is set to "Early Write Acknowledgment" by the ARM MPU.
Note 4. These error behaviors does not occer for write access from DMAC to the PHBIU or PLBIU address space when the bufferable write is enabled by DMBWR.BWE.

## 46.3 Device Lifecycle Management

Device lifecycle identifies the current phase of the device and controls the capabilities of the debug interface, the serial programming interface and Renesas test mode. Figure 46.2 is the illustration of the device lifecycle. Table 46.8 shows the lifecycle definition and capability in each lifecycle.

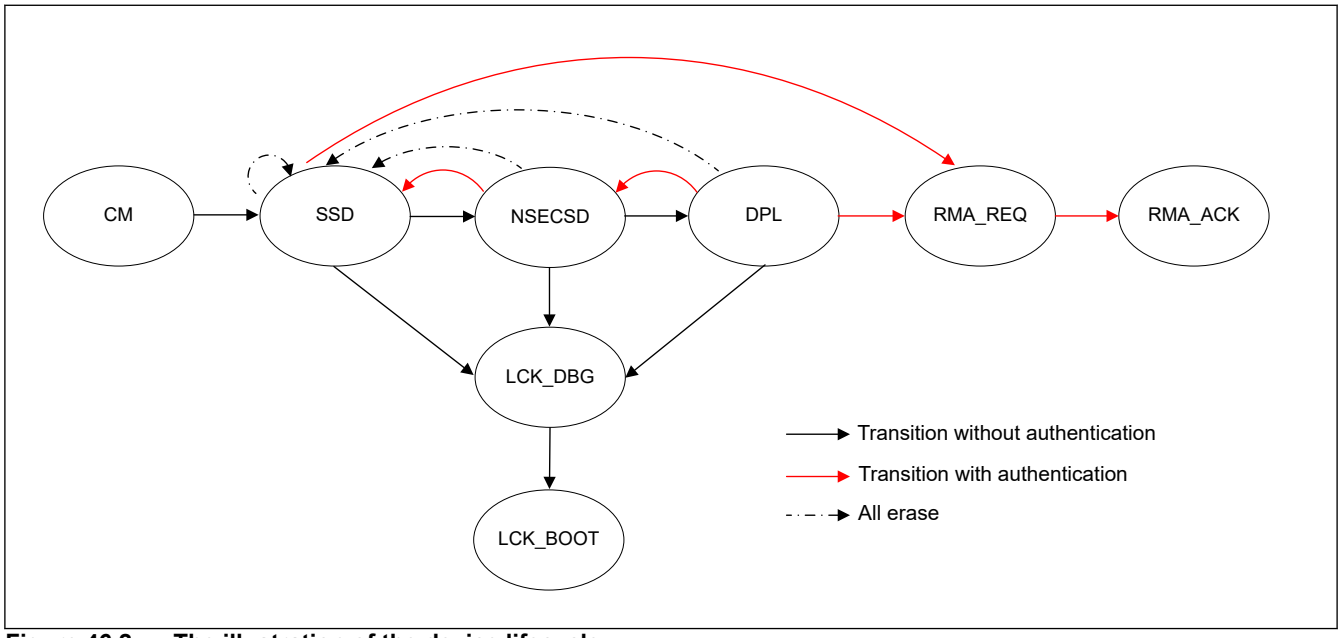



**Figure 46.2 The illustration of the device lifecycle**

**Table 46.8 The lifecycle definition and the capability can be used in each lifecycle (1 of 2)**

| Lifecycle | Definition | Debug level | Serial programming | Renesas test mode |
|---|---|---|---|---|
| CM | “Chip Manufacturing”<br>The device is in Renesas factory. The state when the customer received the device. | DBG2 | Available<br>cannot access code/data flash area | Not available |
| SSD | “Secure Software Development”<br>The secure part of application is being developed. | DBG2 | Available<br>can program/erase/read all code/data flash area | Not available |
| NSECSD | “Non-SECure Software Development”<br>The non-secure part of application is being developed. | DBG1 | Available<br>can program/erase/read only non-secure code/data flash area | Not available |
| DPL | “DePLoyed”<br>The device is in-field. | DBG0 | Available<br>cannot access code/data flash area | Not available |
| LCK_DBG | “LoCKed DeBuG”<br>The debug interface is permanently disabled. | DBG0 | Available<br>cannot access code/data flash area | Not available |
| LCK_BOOT | “LoCKed BOOT interface”<br>The debug interface and the serial programming interface are permanently disabled. | DBG0 | Not available | Not available |

**Table 46.8 The lifecycle definition and the capability can be used in each lifecycle (2 of 2)**

| Lifecycle | Definition | Debug level | Serial programming | Renesas test mode |
|---|---|---|---|---|
| RMA_REQ | “Return Material Authorization REQuest”<br>Request for RMA. The customer must send the device to Renesas in this state. | DBG0 | Available<br>cannot access code/data flash area | Not available |
| RMA_ACK | “Return Material Authorization ACKnowledged”<br>Failure analysis in Renesas | DBG2 | Available<br>cannot access code/data flash area | Available |

### 46.3.1 Changing the Lifecycle State

Use the serial programming commands to change the device lifecycle state. See the boot firmware application note for the detail of command. The lifecycle cannot be updated by application but can read through the dedicated registers.

As shown in Figure 46.2, there are three types lifecycle transition.

The first one is to change to lower debug access level or restrict the serial programming mode. This change can be done with no restriction.

Note: The debug interface is permanently disabled in LCK_DBG. After changed to LCK_DBG, the debug interface cannot be used forever.

Note: The debug interface and serial programming interface are permanently disabled in LCK_BOOT. After changed to LCK_BOOT, the debug interface and the serial programming interface cannot be used forever.

The second one is to change to higher debug access level or request for RMA. This change needs key authentication. The key length is 128 bits. The secure developer needs to inject two keys when the lifecycle is SSD state. One is “SECDBG_KEY” which is used to authentication to change the lifecycle from NSECSD to SSD. Other one is “RMA_KEY” which is used to authentication to change the lifecycle from SSD or DPL to RMA_REQ. The non-secure developer needs to inject one key when the lifecycle is NSECSD state. This is “NONSECDBG_KEY” which is used to authentication to change the lifecycle from DPL to NSECSD. See section 46.4. Key Injection for the detail of how to inject the key. The key authentication uses a challenge and response authentication or authentication using the unique ID. The authentication using the unique ID is available only transition to RMA_REQ. The following is the process of how to calculate the response, challenge and response authentication, or the authentication code using unique ID.

Response = HMAC-SHA256 (KEY, 128bits challenge || fixed value (256bits))

Authentication code = HMAC-SHA256 (KEY, 128bits unique ID || fixed value (256 bits))

fixed value = 9e56dc41 cf0c9648 1b811141 f8f9ba1e 4dd77746 6d403593 17f46d64 fe64fdf6

Note: That "||" represents concatenation, not logical or.

Note: In case the key is not injected, these lifecycle changes cannot be done.

Note: In the lifecycle transition from NSECSD to SSD or form DPL to NSECSD, the contents on the flash memory are not erased.

Note: MCU does not respond after changing to higher debug access level or RMA_REQ. If you continue to use the serial programming commands, need to re-enter the boot mode after a reset. See the boot firmware application note for the detail.

Note: In the lifecycle transition to RMA_REQ, the contents on the flash memory except permanently locked block or setting or BPS_SEL register are erased. The contents in the permanently locked block or register can be read by Renesas at failure analysis. Permanently locked block means the block which programming and erasure is disabled permanently by PBPS, PBPS_SEC and BPS_SEL register. Permanently locked register means SAS register which programming and erasure is disabled permanently by FSPR bit.

The third one is all erase. This is done by an initialize command unless an initialize command itself is disabled. The lifecycle is back to SSD and the contents on the flash memory is erased. If there is permanently locked block or register, an initialize command does not execute. In case of the all bits of PBPS and PBPS_SEC register are 1 and FSPR bit is1, an initialize command is executable.

Note: The initialize command can be issued by everyone, so contents on the flash memory are easily erased. Developers who do not want this can invalidate the initialize command permanently by parameter setting command.

Note: MCU does not respond after executing the initialize command. If you continue to use the serial programming commands, need to re-enter the boot mode after a reset. See the boot firmware application note for the detail.

### 46.3.2 Debug access level

There are three debug access levels, and the debug access level changes according to the lifecycle state.

- DBG2: The debugger connection is allowed, and no restriction to access memories and peripherals
- DBG1: The debugger connection is allowed, and restricted to access only non-secure memory regions and peripherals
- DBG0: The debugger connection is not allowed

### 46.3.3 Serial Programming

Whether a serial programmer can be connected and the range of flash memory that can be accessed depends on the lifecycle state as shown in Table 46.8. And the accepted serial programming command differs depending on the lifecycle state. See the boot firmware application note for the detail of command.

### 46.3.4 Lifecycle changing example

The following is a typical lifecycle changing example.

**Secure developer**

- Change the lifecycle from CM to SSD by using the serial programming command.
- Set the memory security attribution of the code flash, data flash and SRAM by using the serial programming command.
- Program the secure application by using the serial programming interface and debug the secure application. Debug is possible if the lifecycle is CM, but it is impossible to set the memory security attribution in CM state. If the memory security attribution is not set, all area of the code flash, data flash and SRAM is Secure.

Note: Need to configure the registers listed in Table 46.10 as Non-secure only in NSECSD state. See section 46.6.1. Restrictions on setting the security attribution for details.

- Inject “SECDBG_KEY” and “RMA_KEY” by using the serial programming command (if need).
- Inject AES, RSA, ECC, HMAC keys listed in Table 46.9 (if need).
- Disable the all erase by using the serial programming command (if need).
- Change the lifecycle from SSD to NSECSD by using the serial programming command.

**Non-secure developer**

- Program the Non-secure application by using the serial programming interface and debug the Non-secure application.
- Inject “NONSECDBG_KEY” by using the serial programming command (if need).
- Inject AES, RSA, ECC, HMAC keys listed in Table 46.9 (if need).
- Disable the all erase by using the serial programming command (if need).
- Change the lifecycle to DPL by using the serial programming command.

### 46.3.5 Failure analysis

If the customer requests the failure analysis to Renesas, it is necessary to send the device after changing the lifecycle to RMA_REQ. If the lifecycle is not RMA_REQ, Renesas cannot do the failure analysis. Because RMA_REQ is permanent state, it cannot back to the another state after changing to RMA_REQ. It is assumed to change to SSD or NSECSD and analyze before changing to RMA_REQ.

Devices sent to Renesas will not be returned to customers. The device will be discarded.

Note: As described in the section 46.3.1. Changing the Lifecycle State, RMA_KEY is needed to change the lifecycle to RMA_REQ. If the customer forgets the RMA_KEY, Renesas cannot do the failure analysis.

## 46.4 Key Injection

There are three steps required to inject a user key into the MCU.

First, the customer needs to create the 256 bits installation key. This key is called User Factory Programming Key (UFPK) and used to encrypt a user key. The customer gets the key of the wrapped version (W-UFPK) through the Renesas Key Wrapping Service.

Second the customer encrypts the user key using UFPK as the AES key.

Last the customer sends W-UFPK and the encrypted user key to the MCU by using serial programming interface. The sent user key is decrypted, wrapped with the hardware unique key, and then stored in the nonvolatile memory.

Figure 46.3 is the illustration of key injection. Table 46.9 shows the keys that can be injected by serial programming interface.

User Key is used for authentication during the life cycle transition.

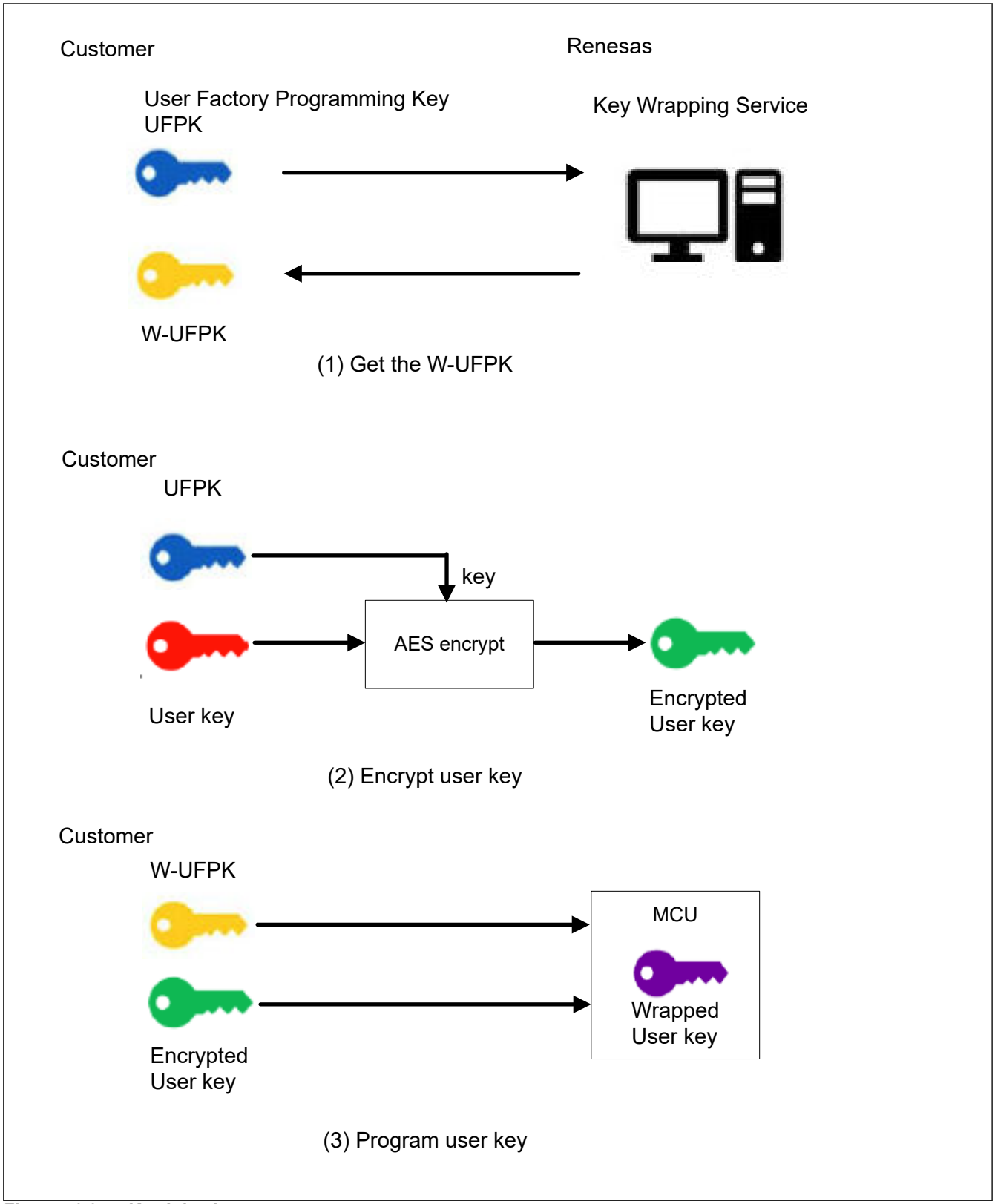



**Figure 46.3 Key Injection**

**Table 46.9 The keys that can be injected by serial programming (1 of 2)**

| | |
|---|---|
| **Lifecycle transition** | SECDBG_KEY, NONSECDBG_KEY, RMA_KEY |
| **AES** | AES-128, AES-192, AES-256, AES-XTS-128, AES-XTS-256 |
| **RSA** | RSA-1024, RSA-2048, RSA-3072, RSA-4096 (Public and Private) |

**Table 46.9 The keys that can be injected by serial programming (2 of 2)**

| | |
|---|---|
| **ECC** | ECC P192, ECC P224, ECC P256, ECC P384, ECC P512 (Public and Private) |
| **HMAC** | HMAC-SHA224, HMAC-SHA256 |

## 46.5 Register Description

### 46.5.1 PSARB : Peripheral Security Attribution Register B

Base address: PSCU = 0x400E_0000

Offset address: 0x04

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PSARB31 | PSARB30 | PSARB29 | PSARB28 | PSARB27 | — | — | — | — | PSARB22 | — | — | PSARB19 | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | PSARB11 | — | PSARB9 | PSARB8 | — | PSARB6 | — | — | — | PSARB2 | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | — | This bit is read as 1. The write value should be 1. | R/W |
| 1 | — | This bit is read as 1. The write value should be 1. | R/W |
| 2 | PSARB2 | CAN0 and the MSTPCRB.MSTPB2 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 3 | — | This bit is read as 1. The write value should be 1. | R/W |
| 5:4 | — | These bits are read as 1. The write value should be 1. | R/W |
| 6 | PSARB6 | QSPI and the MSTPCRB.MSTPB6 bit security attribution<br>This bit is read as 1 (non-secure). | R |
| 7 | — | This bit is read as 1. The write value should be 1. | R/W |
| 8 | PSARB8 | IIC1 and the MSTPCRB.MSTPB8 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 9 | PSARB9 | IIC0 and the MSTPCRB.MSTPB9 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 10 | — | This bit is read as 1. The write value should be 1. | R/W |
| 11 | PSARB11 | USBFS and the MSTPCRB.MSTPB11 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 12 | — | This bit is read as 1. The write value should be 1. | R/W |
| 14:13 | — | These bits are read as 1. The write value should be 1. | R/W |
| 15 | — | This bit is read as 1. The write value should be 1. | R/W |
| 16 | — | This bit is read as 1. The write value should be 1. | R/W |
| 17 | — | This bit is read as 1. The write value should be 1. | R/W |
| 18 | — | This bit is read as 1. The write value should be 1. | R/W |
| 19 | PSARB19 | SPI0 and the MSTPCRB.MSTPB19 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 21:20 | — | These bits are read as 1. The write value should be 1. | R/W |
| 22 | PSARB22 | SCI9 and the MSTPCRB.MSTPB22 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 23 | — | This bit is read as 1. The write value should be 1. | R/W |
| 24 | — | This bit is read as 1. The write value should be 1. | R/W |
| 25 | — | This bit is read as 1. The write value should be 1. | R/W |
| 26 | — | This bit is read as 1. The write value should be 1. | R/W |
| 27 | PSARB27 | SCI4 and the MSTPCRB.MSTPB27 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 28 | PSARB28 | SCI3 and the MSTPCRB.MSTPB28 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 29 | PSARB29 | SCI2 and the MSTPCRB.MSTPB29 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 30 | PSARB30 | SCI1 and the MSTPCRB.MSTPB30 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 31 | PSARB31 | SCI0 and the MSTPCRB.MSTPB31 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |

Note: A bit undefined in this table is reserved bit. The reserved bit should be kept the initial value.
Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

The PSARB specifies the security attribution for each module and the corresponding bit in Module Stop Control Register.

## 46.5.2 PSARC : Peripheral Security Attribution Register C

Base address: PSCU = 0x400E_0000

Offset address: 0x08

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PSARC31 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | PSARC13 | PSARC12 | — | — | — | PSARC8 | — | — | — | — | PSARC3 | — | PSARC1 | PSARC0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PSARC0 | CAC and the MSTPCRC.MSTPC0 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 1 | PSARC1 | CRC and the MSTPCRC.MSTPC1 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 2 | — | This bit is read as 1. The write value should be 1. | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | PSARC3 | CTSU and the MSTPCRC.MSTPC3 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 7:4 | — | These bits are read as 1. The write value should be 1. | R/W |
| 8 | PSARC8 | SSIE0 and the MSTPCRC.MSTPC8 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 11:9 | — | These bits are read as 1. The write value should be 1. | R/W |
| 12 | PSARC12 | SDHI0 and the MSTPCRC.MSTPC12 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 13 | PSARC13 | DOC and the MSTPCRC.MSTPC13 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 19:14 | — | These bits are read as 1. The write value should be 1. | R/W |
| 20 | — | This bit is read as 1. The write value should be 1. | R/W |
| 21 | — | This bit is read as 1. The write value should be 1. | R/W |
| 26:22 | — | These bits are read as 1. The write value should be 1. | R/W |
| 27 | — | These bits are read as 1. The write value should be 1. | R/W |
| 30:28 | — | These bits are read as 1. The write value should be 1. | R/W |
| 31 | PSARC31 | SCE9 and the MSTPCRC.MSTPC31 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

The PSARC specifies the security attribution for each module and the corresponding bit in Module Stop Control Register.

## 46.5.3 PSARD : Peripheral Security Attribution Register D

Base address: PSCU = 0x400E_0000

Offset address: 0x0C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | PSARD22 | — | PSARD20 | — | — | — | PSARD16 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | PSARD14 | PSARD13 | PSARD12 | PSARD11 | — | — | — | — | — | — | — | PSARD3 | PSARD2 | PSARD1 | PSARD0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PSARD0 | AGT3 and the MSTPCRD.MSTPD0 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 1 | PSARD1 | AGT2 and the MSTPCRD.MSTPD1 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 2 | PSARD2 | AGT1 and the MSTPCRD.MSTPD2 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3 | PSARD3 | AGT0 and the MSTPCRD.MSTPD3 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 10:4 | — | These bits are read as 1. The write value should be 1. | R/W |
| 11 | PSARD11 | POEG Group D and the MSTPCRD.MSTPD11 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 12 | PSARD12 | POEG Group C and the MSTPCRD.MSTPD12 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 13 | PSARD13 | POEG Group B and the MSTPCRD.MSTPD13 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 14 | PSARD14 | POEG Group A and the MSTPCRD.MSTPD14 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 15 | — | These bits are read as 1. The write value should be 1. | R/W |
| 16 | PSARD16 | ADC120 and the MSTPCRD.MSTPD16 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 19:17 | — | These bits are read as 1. The write value should be 1. | R/W |
| 20 | PSARD20 | DAC12 and the MSTPCRD.MSTPD20 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 21 | — | This bit is read as 1. The write value should be 1. | R/W |
| 22 | PSARD22 | TSN and the MSTPCRD.MSTPD22 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 31:23 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

The PSARD specifies the security attribution for each module and the corresponding bit in Module Stop Control Register.

## 46.5.4 PSARE : Peripheral Security Attribution Register E

Base address: PSCU = 0x400E_0000

Offset address: 0x10

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PSARE31 | PSARE30 | PSARE29 | PSARE28 | PSARE27 | PSARE26 | PSARE25 | PSARE24 | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | PSARE15 | PSARE14 | — | — | — | — | — | — | — | — | — | — | — | PSARE2 | PSARE1 | PSARE0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PSARE0 | WDT security attribution<br>0: Secure<br>1: Non-secure | R/W |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 1 | PSARE1 | IWDT security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 2 | PSARE2 | RTC security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 13:3 | — | These bits are read as 1. The write value should be 1. | R/W |
| 14 | PSARE14 | AGT5 and the MSTPCRE.MSTPE14 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 15 | PSARE15 | AGT4 and the MSTPCRE.MSTPE15 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 21:16 | — | These bits are read as 1. The write value should be 1. | R/W |
| 22 | — | This bit is read as 1. The write value should be 1. | R/W |
| 23 | — | This bit is read as 1. The write value should be 1. | R/W |
| 24 | PSARE24 | GPT7 and the MSTPCRE.MSTPE24 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 25 | PSARE25 | GPT6 and the MSTPCRE.MSTPE25 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 26 | PSARE26 | GPT5 and the MSTPCRE.MSTPE26 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 27 | PSARE27 | GPT4 and the MSTPCRE.MSTPE27 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 28 | PSARE28 | GPT3 and the MSTPCRE.MSTPE28 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 29 | PSARE29 | GPT2 and the MSTPCRE.MSTPE29 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 30 | PSARE30 | GPT1 and the MSTPCRE.MSTPE30 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 31 | PSARE31 | GPT0, GPT_OPS and the MSTPCRE.MSTPE31 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.
Note: This register is write-protected by PRCR register.

The PSARE specifies the security attribution for each module and the corresponding bit in Module Stop Control Register.

## 46.5.5 MSSAR : Module Stop Security Attribution Register

Base address: PSCU = 0x400E_0000

Offset address: 0x14

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | MSSAR3 | MSSAR2 | MSSAR1 | MSSAR0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | MSSAR0 | The MSTPCRC.MSTPC14 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 1 | MSSAR1 | The MSTPCRA.MSTPA22 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 2 | MSSAR2 | The MSTPCRA.MSTPA7 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 3 | MSSAR3 | The MSTPCRA.MSTPA0 bit security attribution<br>0: Secure<br>1: Non-secure | R/W |
| 31:4 | — | These bits are read as 1. The write value should be 1. | R/W |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

The MSSAR specifies the security attribution for the corresponding bit in Module Stop Control Register.

## 46.5.6 CFSAMONA : Code Flash Security Attribution Monitor Register A

Base address: PSCU = 0x400E_0000

Offset address: 0x18

| Bit position: | 31 | | | | | | | | 23 | | | | | | | | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | CFS2[8:0] | | | | | | | | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | The value set by user | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

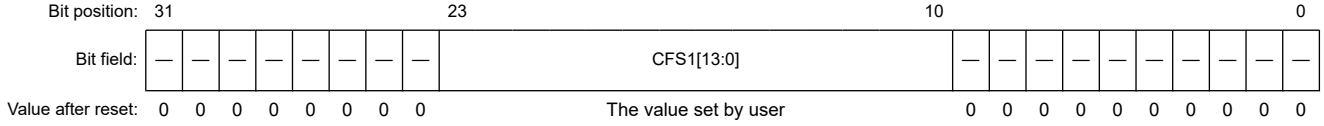
| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 14:0 | — | These bits are read as 0. | R |
| 23:15 | CFS2[8:0] | Code Flash Secure area 2<br>Indicate the total area of secure region and non-secure callable region for code flash. | R |
| 31:24 | — | These bits are read as 0. | R |

Note: The CFSAMONA does not have security attribution.

## 46.5.7 CFSAMONB : Code Flash Security Attribution Monitor Register B

Base address: PSCU = 0x400E_0000

Offset address: 0x1C



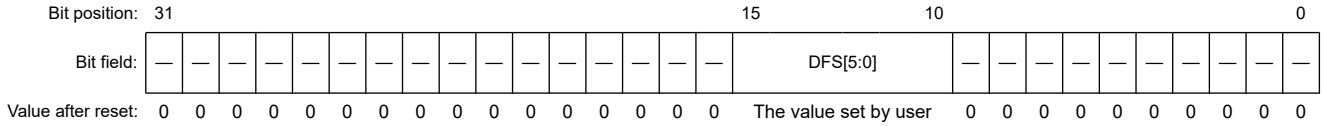

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9:0 | — | These bits are read as 0. | R |
| 23:10 | CFS1[13:0] | Code Flash Secure area 1<br>Indicate the area of secure region for code flash. | R |
| 31:24 | — | These bits are read as 0. | R |

Note: The CFSAMONB does not have security attribution.

## 46.5.8 DFSAMON : Data Flash Security Attribution Monitor Register

Base address: PSCU = 0x400E_0000

Offset address: 0x20



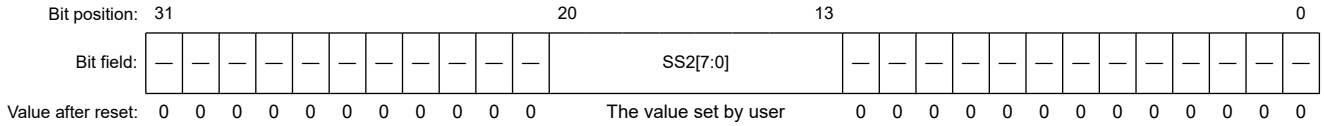

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9:0 | — | These bits are read as 0. | R |
| 15:10 | DFS[5:0] | Data flash Secure area<br>Indicate the area of Secure region for data flash. | R |
| 31:16 | — | These bits are read as 0. | R |

Note: The DFSAMON does not have security attribution.

## 46.5.9 SSAMONA : SRAM Security Attribution Monitor Register A

Base address: PSCU = 0x400E_0000

Offset address: 0x24




| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 12:0 | — | These bits are read as 0. | R |
| 20:13 | SS2[7:0] | SRAM Secure area 2<br>Indicate the total area of Secure region and non-secure callable region for SRAM. | R |
| 31:21 | — | These bits are read as 0. | R |

Note: The SSAMONA does not have security attribution.

## 46.5.10 SSAMONB : SRAM Security Attribution Monitor Register B

Base address: PSCU = 0x400E_0000

Offset address: 0x28

| Bit position: | 31 | | | | | | | | | | | 20 | | | | | | | | | | 10 | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | SS1[10:0] | | | | | | | | | | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | The value set by user | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 9:0 | — | These bits are read as 0. | R |
| 20:10 | SS1[10:0] | SRAM secure area 1<br>Indicate the area of secure region for SRAM. | R |
| 31:21 | — | These bits are read as 0. | R |

Note: The SSAMONB does not have security attribution.

## 46.5.11 DLMMON : Device Lifecycle Management State Monitor Register

Base address: PSCU = 0x400E_0000

Offset address: 0x2C

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | DLMMON[3:0] | | | |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DLM status value | | | |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 3:0 | DLMMON[3:0] | Device Lifecycle Management State Monitor<br>0x1: CM<br>0x2: SSD<br>0x3: NSECSD<br>0x4: DPL<br>0x5: LCK_DBG<br>0x6: LCK_BOOT<br>0x7: RMA_REQ<br>0x8: RMA_ACK<br>Others: Reserved | R |
| 31:4 | — | These bits are read as 0. The write value should be 0. | R |

Note: The DLMMON does not have security attribution.

## 46.5.12 TZFSAR : TrustZone Filter Security Attribution Register

Base address: CPSCU = 0x4000_8000

Offset address: 0x180

| Bit position: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | TZFSA0 |
| Value after reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | TZFSA0 | Security attributes of registers for TrustZone Filter<br>0: Secure<br>1: Non-secure | R/W |
| 31:1 | — | These bits are read as 1. | R |

Note: Only Secure access can write to this register. Both Secure access and Non-secure read access are allowed but Non-secure write access is not allowed and TrustZone access error is not generated.

Note: This register is write-protected by PRCR register.

### TZFSA0 bit (Security attributes of registers for TrustZone Filter)

Security attributes of register for TZFOAD and TZFPT registers.

## 46.5.13 TZFOAD : TrustZone Filter Operation After Detection Register

Base address: TZF = 0x4000_0E00

Offset address: 0x00

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | OAD |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | OAD | Operation after detection<br>0: Non-maskable interrupt<br>1: Reset | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | KeyCode<br>This bit is used to enable or disable writing of the OAD bit. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### OAD bit (Operation after detection)

The OAD bit is specified to generate either reset or non-maskable interrupt when the access to the protect region is detected by the TrustZone Filter.

When the OAD bit is set, write 0xA5 in the KEY[7:0] bits at the same time.

### KEY[7:0] bits (KeyCode)

The KEY[7:0] bits are used to enable or disable writing of the OAD bit. When writing the OAD bit, write 0xA5 in the KEY[7:0] bits at the same time.

When the KEY[7:0] bits value except 0xA5 is written in, the OAD bit is not updated.

The KEY[7:0] bits are read always as 0x00.

## 46.5.14 TZFPT : TrustZone Filter Protect Register

Base address: TZF = 0x4000_0E00

Offset address: 0x04

| Bit position: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit field: | KEY[7:0] | | | | | | | | — | — | — | — | — | — | — | PROTECT |
| Value after reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Symbol | Function | R/W |
|---|---|---|---|
| 0 | PROTECT | Protection of register<br>0: All Bus TrustZone Filter register writing is protected. Read is possible.<br>1: All Bus TrustZone Filter register writing is possible. | R/W |
| 7:1 | — | These bits are read as 0. The write value should be 0. | R/W |
| 15:8 | KEY[7:0] | KeyCode<br>This bit is used to enable or disable writing of the PROTECT bit. | W |

Note: If the security attribution is configured as Secure:
- Secure access and Non-secure read access are allowed
- Non-secure write access is ignored, and TrustZone access error is not generated.

If the security attribution is configured as Non-secure:
- Secure and Non-secure access are allowed.

### PROTECT bit (Protection of register)

The PROTECT bit controls enable or disable writing to the corresponding registers to be protected. TZFOAD register is protected by PROTECT.

When the PROTECT bit is set, write 0xA5 in the KEY[7:0] bits at the same time.

### KEY[7:0] bits (KeyCode)

The KEY[7:0] bits are used to enable or disable writing of the PROTECT bit. When writing the PROTECT bit, write 0xA5 in the KEY[7:0] bits at the same time.

When the KEY[7:0] bits value except 0xA5 is written in, the PROTECT bit is not updated.

The KEY[7:0] bits are read always as 0x00.

## 46.6 Usage Notes

### 46.6.1 Restrictions on setting the security attribution

To set the software breakpoint, the debugger need to re-program the flash. Table 46.10 shows the registers that the debugger sets to re-program the flash. If the security attribution of the register listed in Table 46.10 is configured as secure, the debugger cannot set the software breakpoint in NSECSD state because the debugger cannot change the register setting. Secure developper need to configure the registers listed in Table 46.10 as non-secure only in NSECSD state.

**Table 46.10 The registers that the debugger sets to re-program the flash**

| Function name | Register name |
|---|---|
| Clock Generation Circuit | SCKDIVCR, SCKCR, PLLCCR, PLLCR, HOCOCR, MOCOCR |
| Low-Power modes | OPCCR, SOPCCR |

### 46.6.2 SAU setting

After reset, all of address space is marked as Secure by SAU default setting. SAU_CTRL register should be set to 0x2 to enable the IDAU security attribution. That is, after setting SAU_CTRL register to 0x2, the address space security attribution becomes as shown in Table 46.6.

### 46.6.3 Non-secure exception during the setting of FACI registers

As shown in Table 46.5, the registers related to FACI are protected from non-secure access only during programming/erasure or during suspend programming/erasure. Outside of this state, the access from non-secure region is not protected. For example, when programming by the secure user, the non-secure user can rewrite the FSADDR if a non-secure exception occurs immediately after “Set the start address of the target block to the FSADDR register” flow in Figure 44.13. If the FACI command is issued after the non-secure exception processing is completed and the CPU state returns to the secure state, data will be programmed to an address not intended by the secure user.

To prevent such a things, secure user needs to set not to accept the non-secure exception during the following period.

- Set not to accept the non-secure exeption before setting FWEPROR to 0x01 or setting FENTRYR to other than 0x0000, that is before lereasing the protection of FWEPROR or FENTRYR.
- Set to accept the non-secure exception after all write access to the FACI command-issuing area is completed.

### 46.6.4 FCU interrupt usage

It is recommended that secure users do not use the FCU interrupts, but rather use the register polling. Because non-secure users can program/erase the data flash without calling the secure gateway, if secure user uses FCU interrupts, the unintentional exception handliing may be exeuted when data flash is programmed/erased by a non-secure user.

# 47. Electrical Characteristics

Supported peripheral functions and pins differ from one product name to another.

Unless otherwise specified, the electrical characteristics of the MCU are defined under the following conditions:

- VCC = AVCC0 = VCC_USB = VBATT = 2.7 to 3.6 V
- 2.7 ≤ VREFH0/VREFH ≤ AVCC0
- VSS = AVSS0 = VREFL0/VREFL = VSS_USB = 0 V
- $T_a = T_{opr}$

Figure 47.1 shows the timing conditions.

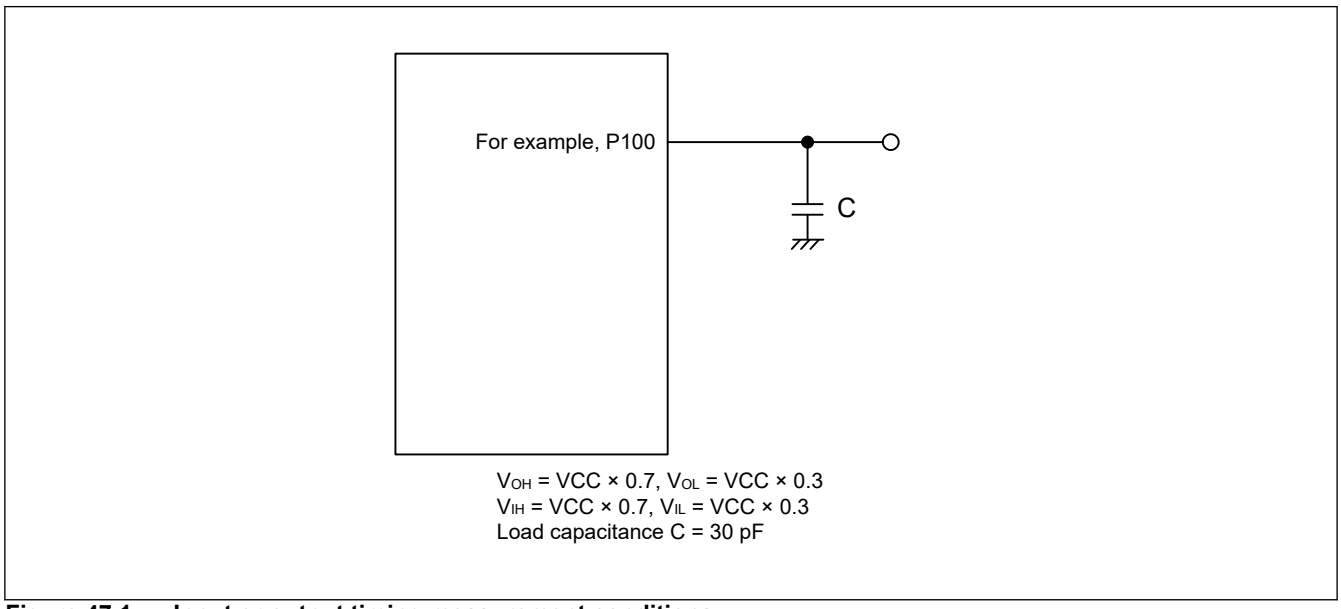



**Figure 47.1 Input or output timing measurement conditions**

The recommended measurement conditions for the timing specification of each peripheral provided are for the best peripheral operation. Make sure to adjust the driving abilities of each pin to meet your conditions.

## 47.1 Absolute Maximum Ratings

**Table 47.1 Absolute maximum ratings**

| Parameter | Symbol | Value | Unit |
|---|---|---|---|
| Power supply voltage | VCC, VCC_USB*2 | –0.3 to +4.0 | V |
| VBATT power supply voltage | VBATT | –0.3 to +4.0 | V |
| Input voltage (except for 5 V-tolerant ports*1) | $V_{in}$ | –0.3 to VCC + 0.3 | V |
| Input voltage (5 V-tolerant ports*1) | $V_{in}$ | –0.3 to + VCC + 4.0 (max. 5.8) | V |
| Reference power supply voltage | VREFH/VREFH0 | –0.3 to VCC + 0.3 | V |
| Analog power supply voltage | AVCC0*2 | –0.3 to +4.0 | V |
| Analog input voltage | $V_{AN}$ | –0.3 to AVCC0 + 0.3 | V |
| Operating temperature*3 *4 | $T_{opr}$ | –40 to +105 | °C |
| Storage temperature | $T_{stg}$ | –55 to +125 | °C |

Note 1. Ports P205, P206, P400, P401, P407 to P415, and P708 are 5 V tolerant.
Note 2. Connect AVCC0 and VCC_USB to VCC.
Note 3. See section 47.2.1. Tj/Ta Definition.
Note 4. Contact a Renesas Electronics sales office for information on derating operation when Ta = +85°C to +105°C. Derating is the systematic reduction of load for improved reliability.

**Caution: Permanent damage to the MCU might result if absolute maximum ratings are exceeded.**

**Table 47.2 Recommended operating conditions**

| Parameter | Symbol | Value | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Power supply voltages | VCC | When USB is not used | 2.7 | — | 3.6 | V |
| | | When USB is used | 3.0 | — | 3.6 | V |
| | VSS | | — | 0 | — | V |
| USB power supply voltages | VCC_USB | | — | VCC | — | V |
| | VSS_USB | | — | 0 | — | V |
| VBATT power supply voltage | VBATT | | 1.8 | — | 3.6 | V |
| Analog power supply voltages | AVCC0*1 | | — | VCC | — | V |
| | AVSS0 | | — | 0 | — | V |

Note 1. Connect AVCC0 to VCC. When the A/D converter and the D/A converter are not in use, do not leave the AVCC0, VREFH/VREFH0, AVSS0, and VREFL/VREFL0 pins open. Connect the AVCC0 and VREFH/VREFH0 pins to VCC, and the AVSS0 and VREFL/VREFL0 pins to VSS, respectively.

## 47.2 DC Characteristics

### 47.2.1 Tj/Ta Definition

**Table 47.3 DC characteristics**

Conditions: Products with operating temperature ($T_a$) -40 to +105°C

| Parameter | Symbol | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|
| Permissible junction temperature | $T_j$ | — | 125 | °C | High-speed mode<br>Low-speed mode<br>Subosc-speed mode |

Note: Make sure that $T_j = T_a + \theta ja \times$ total power consumption (W), where total power consumption = $(VCC - V_{OH}) \times \Sigma I_{OH} + V_{OL} \times \Sigma I_{OL} + I_{CC}max \times VCC$.

### 47.2.2 I/O $V_{IH}$, $V_{IL}$

**Table 47.4 I/O $V_{IH}$, $V_{IL}$ (1 of 2)**

| Parameter | | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Input voltage (except for Schmitt trigger input pins) | Peripheral function pin | EXTAL (external clock input), SPI (except RSPCK) | $V_{IH}$ | VCC × 0.8 | — | — | V |
| | | | $V_{IL}$ | — | — | VCC × 0.2 | |
| | | IIC (SMBus) | $V_{IH}$ | 2.1 | — | VCC + 3.6 (max 5.8) | |
| | | | $V_{IL}$ | — | — | 0.8 | |

**Table 47.4 I/O $V_{IH}$, $V_{IL}$ (2 of 2)**

| Parameter | | | | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|---|---|
| Schmitt trigger input voltage | Peripheral function pin | IIC (except for SMBus) | | | $V_{IH}$ | VCC × 0.7 | — | VCC + 3.6 (max 5.8) | V |
| | | | | | $V_{IL}$ | — | — | VCC × 0.3 | |
| | | | | | $\Delta V_T$ | VCC × 0.05 | — | — | |
| | | 5 V-tolerant ports*1 *5 | | | $V_{IH}$ | VCC × 0.8 | — | VCC + 3.6 (max 5.8) | |
| | | | | | $V_{IL}$ | — | — | VCC × 0.2 | |
| | | | | | $\Delta V_T$ | VCC × 0.05 | — | — | |
| | | RTCIC0, RTCIC1, RTCIC2 | When using the Battery Backup Function | When VBATT power supply is selected | $V_{IH}$ | $V_{BATT}$ × 0.8 | — | $V_{BATT}$ + 0.3 | |
| | | | | | $V_{IL}$ | — | — | $V_{BATT}$ × 0.2 | |
| | | | | | $\Delta V_T$ | $V_{BATT}$ × 0.05 | — | — | |
| | | | | When VCC power supply is selected | $V_{IH}$ | VCC × 0.8 | — | Higher voltage either VCC + 0.3 V or $V_{BATT}$ + 0.3 V | |
| | | | | | $V_{IL}$ | — | — | VCC × 0.2 | |
| | | | | | $\Delta V_T$ | VCC × 0.05 | — | — | |
| | | | When not using the Battery Backup Function | | $V_{IH}$ | VCC × 0.8 | — | VCC + 0.3 | |
| | | | | | $V_{IL}$ | — | — | VCC × 0.2 | |
| | | | | | $\Delta V_T$ | VCC × 0.05 | — | — | |
| | | Other input pins*2 | | | $V_{IH}$ | VCC × 0.8 | — | — | |
| | | | | | $V_{IL}$ | — | — | VCC × 0.2 | |
| | | | | | $\Delta V_T$ | VCC × 0.05 | — | — | |
| | Ports | 5 V-tolerant ports*3 *5 | | | $V_{IH}$ | VCC × 0.8 | — | VCC + 3.6 (max 5.8) | V |
| | | | | | $V_{IL}$ | — | — | VCC × 0.2 | |
| | | Other input pins*4 | | | $V_{IH}$ | VCC × 0.8 | — | — | |
| | | | | | $V_{IL}$ | — | — | VCC × 0.2 | |

Note 1. RES and peripheral function pins associated with Ports P205, P206, P400, P401, P407 to P415, and P708（total 15 pins).
Note 2. All input pins except for the peripheral function pins already described in the table.
Note 3. Ports P205, P206, P400, P401, P407 to P415, and P708（total 14 pins).
Note 4. All input pins except for the ports already described in the table.
Note 5. When VCC is less than 2.7 V, the input voltage of 5 V-tolerant ports should be less than 3.6 V, otherwise breakdown may occur because 5 V-tolerant ports are electrically controlled so as not to violate the break down voltage.

## 47.2.3 I/O $I_{OH}$, $I_{OL}$

**Table 47.5 I/O $I_{OH}$, $I_{OL}$**

| Parameter | | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Permissible output current (average value per pin) | Ports P000 to P008, P013 to P015, P201 | — | $I_{OH}$ | — | — | –2.0 | mA |
| | | | $I_{OL}$ | — | — | 2.0 | mA |
| | Ports P205, P206, P407 to P415, P708 (total 12 pins) | Low drive*1 | $I_{OH}$ | — | — | –2.0 | mA |
| | | | $I_{OL}$ | — | — | 2.0 | mA |
| | | Middle drive*2 | $I_{OH}$ | — | — | –4.0 | mA |
| | | | $I_{OL}$ | — | — | 4.0 | mA |
| | | High drive*3 | $I_{OH}$ | — | — | –20 | mA |
| | | | $I_{OL}$ | — | — | 20 | mA |
| | Other output pins*4 | Low drive*1 | $I_{OH}$ | — | — | –2.0 | mA |
| | | | $I_{OL}$ | — | — | 2.0 | mA |
| | | Middle drive*2 | $I_{OH}$ | — | — | –4.0 | mA |
| | | | $I_{OL}$ | — | — | 4.0 | mA |
| | | High drive*3 | $I_{OH}$ | — | — | –16 | mA |
| | | | $I_{OL}$ | — | — | 16 | mA |
| Permissible output current (max value per pin) | Ports P000 to P008, P013 to P015, P201 | — | $I_{OH}$ | — | — | –4.0 | mA |
| | | | $I_{OL}$ | — | — | 4.0 | mA |
| | Ports P205, P206, P407 to P415, P708 (total 12 pins) | Low drive*1 | $I_{OH}$ | — | — | –4.0 | mA |
| | | | $I_{OL}$ | — | — | 4.0 | mA |
| | | Middle drive*2 | $I_{OH}$ | — | — | –8.0 | mA |
| | | | $I_{OL}$ | — | — | 8.0 | mA |
| | | High drive*3 | $I_{OH}$ | — | — | –40 | mA |
| | | | $I_{OL}$ | — | — | 40 | mA |
| | Other output pins*4 | Low drive*1 | $I_{OH}$ | — | — | –4.0 | mA |
| | | | $I_{OL}$ | — | — | 4.0 | mA |
| | | Middle drive*2 | $I_{OH}$ | — | — | –8.0 | mA |
| | | | $I_{OL}$ | — | — | 8.0 | mA |
| | | High drive*3 | $I_{OH}$ | — | — | –32 | mA |
| | | | $I_{OL}$ | — | — | 32 | mA |
| Permissible output current (maxvalue of total of all pins) | Maximum of all output pins | | $\Sigma I_{OH\ (max)}$ | — | — | –80 | mA |
| | | | $\Sigma I_{OL\ (max)}$ | — | — | 80 | mA |

Note 1. This is the value when low driving ability is selected in the Port Drive Capability bit in the PmnPFS register. The selected driving ability is retained in Deep Software Standby mode.
Note 2. This is the value when middle driving ability is selected in the Port Drive Capability bit in the PmnPFS register. The selected driving ability is retained in Deep Software Standby mode.
Note 3. This is the value when high driving ability is selected in the Port Drive Capability bit in the PmnPFS register. The selected driving ability is retained in Deep Software Standby mode.
Note 4. Except for P200, which is an input port.

**Caution: To protect the reliability of the MCU, the output current values should not exceed the values in this table. The average output current indicates the average value of current measured during 100 µs.**

## 47.2.4 I/O $V_{OH}$, $V_{OL}$, and Other Characteristics

**Table 47.6 I/O $V_{OH}$, $V_{OL}$, and other characteristics**

| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| Output voltage | IIC | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 3.0 mA |
| | | $V_{OL}$ | — | — | 0.6 | | $I_{OL}$ = 6.0 mA |
| | IIC*1 | $V_{OL}$ | — | — | 0.4 | | $I_{OL}$ = 15.0 mA (ICFER.FMPE = 1) |
| | | $V_{OL}$ | — | 0.4 | — | | $I_{OL}$ = 20.0 mA (ICFER.FMPE = 1) |
| | Ports P205, P206, P407 to P415, P708 (total 12 pins)*2 | $V_{OH}$ | VCC – 1.0 | — | — | | $I_{OH}$ = –20 mA<br>VCC = 3.3 V |
| | | $V_{OL}$ | — | — | 1.0 | | $I_{OL}$ = 20 mA<br>VCC = 3.3 V |
| | Other output pins | $V_{OH}$ | VCC – 0.5 | — | — | | $I_{OH}$ = –1.0 mA |
| | | $V_{OL}$ | — | — | 0.5 | | $I_{OL}$ = 1.0 mA |
| Input leakage current | RES | $\|I_{in}\|$ | — | — | 5.0 | µA | $V_{in}$ = 0 V<br>$V_{in}$ = 5.5 V |
| | Port P200 | | — | — | 1.0 | | $V_{in}$ = 0 V<br>$V_{in}$ = VCC |
| Three-state leakage current (off state) | 5 V-tolerant ports | $\|I_{TSI}\|$ | — | — | 5.0 | µA | $V_{in}$ = 0 V<br>$V_{in}$ = 5.5 V |
| | Other ports (except for port P200) | | — | — | 1.0 | | $V_{in}$ = 0 V<br>$V_{in}$ = VCC |
| Input pull-up MOS current | Ports P0 to P7 | $I_p$ | –300 | — | –10 | µA | VCC = 2.7 to 3.6 V<br>$V_{in}$ = 0 V |
| Input capacitance | USB_DP, USB_DM, and ports P014, P015, P400, P401 | $C_{in}$ | — | — | 16 | pF | Vbias = 0 V<br>Vamp = 20 mV<br>f = 1 MHz<br>Ta = 25°C |
| | Other input pins | | — | — | 8 | | |

Note 1. SCL0_A, SDA0_A (total 2 pins).
Note 2. This is the value when high driving ability is selected in the Port Drive Capability bit in the PmnPFS register. The selected driving ability is retained in Deep Software Standby mode.

## 47.2.5 Operating and Standby Current

**Table 47.7 Operating and standby current (1 of 2)**

| Parameter | | | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|---|
| Supply current*1 | High-speed mode | Maximum*2 *13 | | $I_{CC}$*3 | — | — | 65 | mA | ICLK = 100 MHz<br>PCLKA = 100 MHz<br>PCLKB = 50 MHz<br>PCLKC = 50 MHz<br>PCLKD = 100 MHz<br>FCLK = 50 MHz |
| | | CoreMark®*5 *6 *12 *14 | | | — | 8.1 | — | | |
| | | Normal mode | All peripheral clocks enabled, while (1) code executing from flash*4 *12 | | — | 15.4 | — | | |
| | | | All peripheral clocks disabled, while (1) code executing from flash*5 *6 *12 *14 | | — | 6.1 | — | | |
| | | Sleep mode*5 *14 | | | — | 4.4*6 *12 | 25*7 *13 | | |
| | | Increase during BGO operation | Data flash P/E | | — | 6 | — | | |
| | | | Code flash P/E | | — | 8 | — | | |
| | Low-speed mode*5 *10 | | | | — | 0.8 | — | | ICLK = 1 MHz |
| | Subosc-speed mode*5 *11 | | | | — | 0.7 | — | | ICLK = 32.768 kHz |
| | Software Standby mode | | SNZCR.RXDREQEN = 1 | | — | — | 14 | | — |
| | | | SNZCR.RXDREQEN = 0 | | — | 0.7 | — | | — |
| | Deep Software Standby mode | Power supplied to Standby SRAM and USB resume detecting unit | | | — | 16 | 96 | μA | — |
| | | Power not supplied to SRAM or USB resume detecting unit | Power-on reset circuit low power function disabled | | — | 12 | 27 | | — |
| | | | Power-on reset circuit low power function enabled | | — | 5 | 17 | | — |
| | | Increase when the RTC and AGT are operating | When the low-speed on-chip oscillator (LOCO) is in use | | — | 4.4 | — | | — |
| | | | When a crystal oscillator for low clock loads is in use | | — | 1.0 | — | | — |
| | | | When a crystal oscillator for standard clock loads is in use | | — | 1.6 | — | | — |
| | RTC operating while VCC is off (with the battery backup function, only the RTC and sub-clock oscillator operate) | | When a crystal oscillator for low clock loads is in use | | — | 0.6 | — | | $V_{BATT}$ = 1.8 V, VCC = 0 V |
| | | | | | — | 1.2 | — | | $V_{BATT}$ = 3.3 V, VCC = 0 V |
| | | | When a crystal oscillator for standard clock loads is in use | | — | 1.1 | — | | $V_{BATT}$ = 1.8 V, VCC = 0 V |
| | | | | | — | 1.8 | — | | $V_{BATT}$ = 3.3 V, VCC = 0 V |
| | Inrush current on returning from Deep Software Standby mode | | Inrush current*8 | $I_{RUSH}$ | — | 160 | — | mA | |
| | | | Energy of inrush current*8 | $E_{RUSH}$ | — | 1.0 | — | μC | |

**Table 47.7 Operating and standby current (2 of 2)**

| Parameter | | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|
| Analog power supply current | During 12-bit A/D conversion | | $AI_{CC}$ | — | 0.8 | 1.1 | mA | — |
| | Temperature sensor | | | — | 0.1 | 0.2 | mA | — |
| | During D/A conversion (per unit) | Without AMP output | | — | 0.1 | 0.2 | mA | — |
| | | With AMP output | | — | 0.6 | 1.1 | mA | — |
| | Waiting for A/D, D/A conversion (all units) | | | — | 0.5 | 1.0 | mA | — |
| | ADC12, DAC12 in standby modes (all units)*9 | | | — | 0.4 | 4.0 | µA | — |
| Reference power supply current (VREFH0) | During 12-bit A/D conversion (unit 0) | | $AI_{REFH0}$ | — | 70 | 120 | µA | — |
| | Waiting for 12-bit A/D conversion (unit 0) | | | — | 0.07 | 0.5 | µA | — |
| | ADC12 in standby modes (unit 0) | | | — | 0.07 | 0.5 | µA | — |
| Reference power supply current (VREFH) | During D/A conversion (per unit) | Without AMP output | $AI_{REFH}$ | — | 0.1 | 0.4 | mA | — |
| | | With AMP ouput | | — | 0.1 | 0.4 | mA | — |
| | Waiting for D/A (all units) conversion | | | — | 0.07 | 0.8 | µA | — |
| USB operating current | Low speed | USB | $I_{CCUSBLS}$ | — | 3.5 | 6.5 | mA | VCC_USB |
| | Full speed | USB | $I_{CCUSBFS}$ | — | 4.0 | 10.0 | mA | VCC_USB |
| LDO operating current (1 unit)*15 | | | $I_{CCLDO}$ | — | 0.18 | — | mA | — |
| PLL2-LDO operating current | | | $I_{CCPLL2LDO}$ | — | 0.21 | — | mA | — |

Note 1. Supply current values are with all output pins unloaded and all input pull-up MOSs in the off state.
Note 2. Measured with clocks supplied to the peripheral functions. This does not include the BGO operation.
Note 3. $I_{CC}$ depends on f (ICLK) as follows.
$I_{CC}$ Max. = 0.53 × f + 12 (max. operation in high-speed mode)
$I_{CC}$ Typ. = 0.05 × f + 1.85 (normal operation in high-speed mode, all peripheral clocks disabled)
$I_{CC}$ Typ. = 0.12 × f + 0.69 (low-speed mode)
$I_{CC}$ Max. = 0.13 × f + 12 (sleep mode)
Note 4. This does not include the BGO operation.
Note 5. Supply of the clock signal to peripherals is stopped in this state. This does not include the BGO operation.
Note 6. FCLK, PCLKA, PCLKB, PCLKC, and PCLKD are set to divided by 64 (1.563 MHz).
Note 7. FCLK, PCLKA, PCLKB, PCLKC, and PCLKD are set to divided by 64 (3.125 MHz).
Note 8. Reference value
Note 9. When the MCU is in Software Standby mode or the MSTPCRD.MSTPD16 (12-Bit A/D Converter 0 Module Stop bit) and MSTPCRD.MSTPD20 (12-bit D/A converter module stop bit) are in the module-stop state.
Note 10. FCLK, PCLKA, PCLKB, PCLKC, and PCLKD are set to divided by 64 (15.6 kHz).
Note 11. PCLKA, PCLKB, PCLKC, and PCLKD are set to divided by 64 (512 Hz). FCLK is the same frequency as that of ICLK.
Note 12. PLL output frequency = 100MHz.
Note 13. PLL output frequency = 200MHz.
Note 14. PLL2-LDO disabled.
Note 15. n = 0, 1

**Table 47.8 Coremark and normal mode current**

| Parameter | | | Symbol | Typ | Unit | Test conditions |
|---|---|---|---|---|---|---|
| Supply Current*1 | Coremark*2 *3 *4 | | $I_{CC}$ | 81 | µA/MHz | ICLK = 100MHz<br>PCLKA<br>= PCLKB<br>= PCLKC<br>= PCLKD<br>= FCLK<br>= 1.56 MHz |
| | Normal mode | All peripheral clocks disabled, cache on, while (1) code executing from flash*2 *3 *4 | | 60 | | |
| | | All peripheral clocks disabled, cache off, while (1) code executing from flash*2 *3 *4 | | 118 | | |

Note 1. Supply current values are with all output pins unloaded and all input pull-up MOSs in the off state.
Note 2. Supply of the clock signal to peripherals is stopped in this state. This does not include the BGO operation.
Note 3. PLL output frequency = 100MHz.
Note 4. PLL2-LDO disabled.

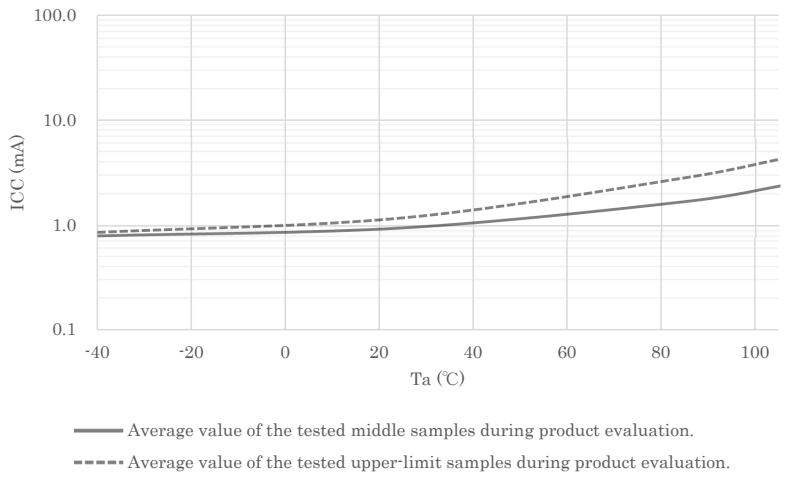



**Figure 47.2 Temperature dependency in Software Standby mode (reference data)**

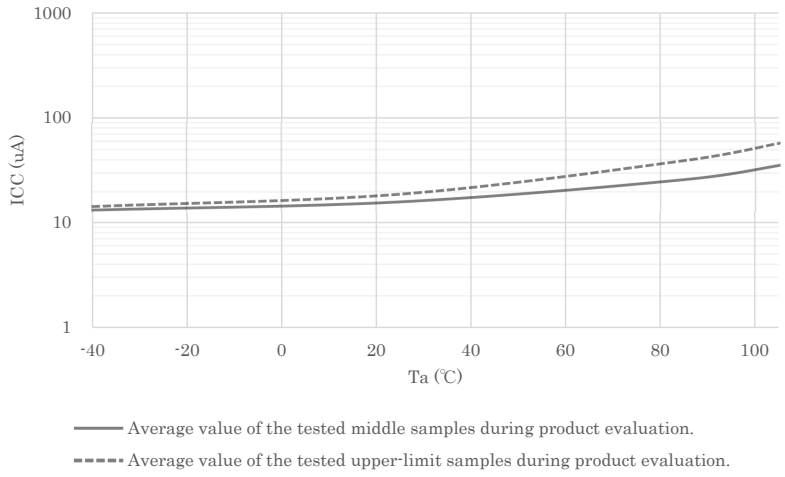



**Figure 47.3 Temperature dependency in Deep Software Standby mode, power supplied to standby SRAM and USB resume detecting unit (reference data)**

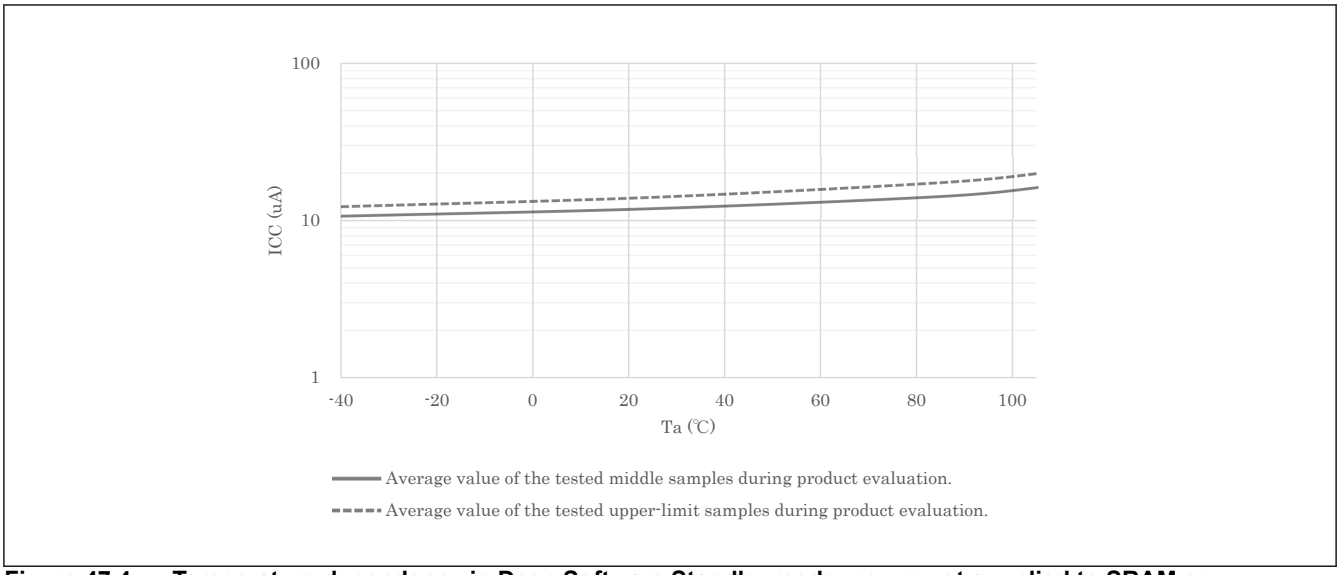



**Figure 47.4 Temperature dependency in Deep Software Standby mode, power not supplied to SRAM or USB resume detecting unit, power-on reset circuit low power function disabled (reference data)**

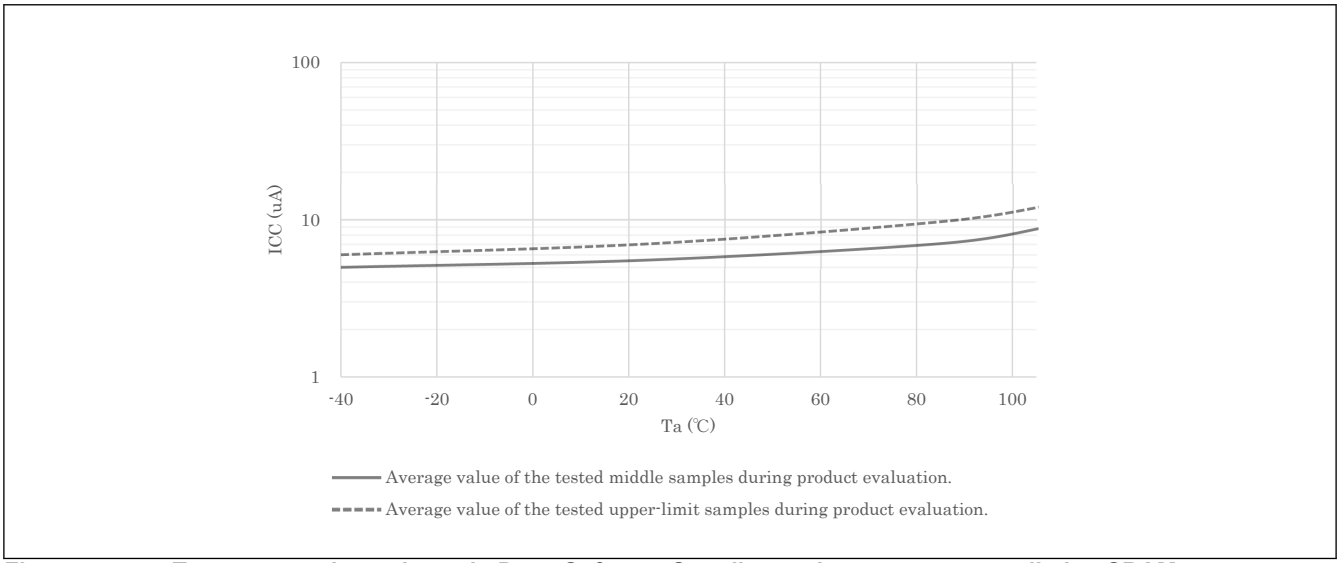



**Figure 47.5 Temperature dependency in Deep Software Standby mode, power not supplied to SRAM or USB resume detecting unit, power-on reset circuit low power function enabled (reference data)**

## 47.2.6 VCC Rise and Fall Gradient and Ripple Frequency

**Table 47.9 Rise and fall gradient characteristics**

| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| VCC rising gradient | Voltage monitor 0 reset disabled at startup | SrVCC | 0.0084 | — | 20 | ms/V | — |
| | Voltage monitor 0 reset enabled at startup | | 0.0084 | — | — | | — |
| | SCI/USB boot mode*1 | | 0.0084 | — | 20 | | — |
| VCC falling gradient*2 | | SfVCC | 0.0084 | — | — | ms/V | — |

Note 1. At boot mode, the reset from voltage monitor 0 is disabled regardless of the value of the OFS1.LVDAS bit.
Note 2. This applies when VBATT is used.

**Table 47.10 Rising and falling gradient and ripple frequency characteristics**

The ripple voltage must meet the allowable ripple frequency $f_{r(VCC)}$ within the range between the VCC upper limit (3.6 V) and lower limit (2.7 V). When the VCC change exceeds VCC ±10%, the allowable voltage change rising and falling gradient dt/dVCC must be met.

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| Allowable ripple frequency | $f_{r\ (VCC)}$ | — | — | 10 | kHz | Figure 47.6<br>$V_{r\ (VCC)} \leq VCC \times 0.2$ |
| | | — | — | 1 | MHz | Figure 47.6<br>$V_{r\ (VCC)} \leq VCC \times 0.08$ |
| | | — | — | 10 | MHz | Figure 47.6<br>$V_{r\ (VCC)} \leq VCC \times 0.06$ |
| Allowable voltage change rising and falling gradient | dt/dVCC | 1.0 | — | — | ms/V | When VCC change exceeds VCC ±10% |

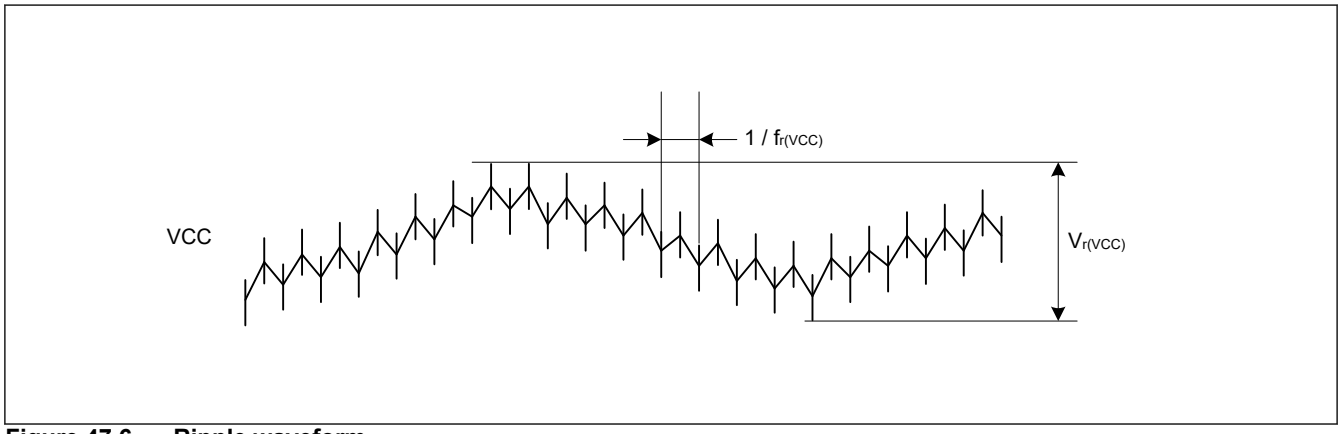



**Figure 47.6 Ripple waveform**

## 47.2.7 Thermal Characteristics

Maximum value of junction temperature (Tj) must not exceed the value of "section 47.2.1. Tj/Ta Definition".

Tj is calculated by either of the following equations.

- Tj = Ta + θja × Total power consumption
- Tj = Tt + Ψjt × Total power consumption
  - Tj : Junction Temperature (°C)
  - Ta : Ambient Temperature (°C)
  - Tt : Top Center Case Temperature (°C)
  - θja : Thermal Resistance of "Junction"-to-"Ambient" (°C/W)
  - Ψjt : Thermal Resistance of "Junction"-to-"Top Center Case" (°C/W)
- Total power consumption = Voltage × (Leakage current + Dynamic current)
- Leakage current of IO = Σ ($I_{OL} \times V_{OL}$) /Voltage + Σ ($|I_{OH}| \times |VCC - V_{OH}|$) /Voltage
- Dynamic current of IO = Σ IO ($C_{in} + C_{load}$) × IO switching frequency × Voltage
  - $C_{in}$: Input capacitance
  - $C_{load}$: Output capacitance

Regarding θja and Ψjt, refer to Table 47.11.

**Table 47.11 Thermal Resistance**

| Parameter | Package | Symbol | Value*1 | Unit | Test conditions |
|---|---|---|---|---|---|
| Thermal Resistance | 48-pin QFN (PWQN0048KC-A) | θja | 23.9 | °C/W | JESD 51-2 and 51-7 compliant |
| | 48-pin LQFP (PLQP0048KB-B) | | 62.1 | | |
| | 64-pin LQFP (PLQP0064KB-C) | | 54.6 | | |
| | 100-pin LQFP (PLQP0100KB-B) | | 55.1 | | |
| | 48-pin QFN (PWQN0048KC-A) | Ψjt | 0.28 | °C/W | JESD 51-2 and 51-7 compliant |
| | 48-pin LQFP (PLQP0048KB-B) | | 2.39 | | |
| | 64-pin LQFP (PLQP0064KB-C) | | 1.90 | | |
| | 100-pin LQFP (PLQP0100KB-B) | | 1.90 | | |

Note 1. The values are reference values when the 4-layer board is used. Thermal resistance depends on the number of layers or size of the board. For details, refer to the JEDEC standards.

### 47.2.7.1 Calculation guide of $I_{CC}$max

Table 47.12 shows the power consumption of each unit.

**Table 47.12 Power consumption of each unit (1 of 2)**

| Dynamic current/ Leakage current | MCU Domain | Category | Item | Frequency [MHz] | Current [uA/MHz] | Current*1 [mA] |
|---|---|---|---|---|---|---|
| Leakage current | Analog | LDO and Leak*2 | Ta = 75 °C*3 | — | — | 7.82 |
| | | | Ta = 85 °C*3 | — | — | 9.13 |
| | | | Ta = 95 °C*3 | — | — | 11.08 |
| | | | Ta = 105 °C*3 | — | — | 14.33 |

**Table 47.12 Power consumption of each unit (2 of 2)**

| Dynamic current/ Leakage current | MCU Domain | Category | Item | Frequency [MHz] | Current [uA/MHz] | Current[*1] [mA] |
|---|---|---|---|---|---|---|
| Dynamic current | CPU | Operation with Flash and SRAM | Coremark | 100 | 55.556 | 5.56 |
| | Peripheral Unit | Timer | GPT16 (4ch)[*4] | 100 | 3.575 | 0.36 |
| | | | GPT32 (4ch)[*4] | 100 | 4.230 | 0.42 |
| | | | POEG (4 Groups) | 50 | 1.361 | 0.07 |
| | | | AGT (6ch)[*4] | 50 | 9.228 | 0.46 |
| | | | RTC | 50 | 4.277 | 0.21 |
| | | | WDT | 50 | 0.764 | 0.04 |
| | | | IWDT | 50 | 0.339 | 0.02 |
| | | Communication interfaces | USBFS | 50 | 9.385 | 0.47 |
| | | | SCI (6ch)[*4] | 100 | 18.715 | 1.87 |
| | | | IIC (2ch)[*4] | 50 | 3.367 | 0.16 |
| | | | CAN | 50 | 1.898 | 0.09 |
| | | | SPI | 100 | 3.024 | 0.30 |
| | | | QSPI | 100 | 2.051 | 0.21 |
| | | | SSIE | 50 | 3.208 | 0.16 |
| | | | SDHI | 50 | 6.341 | 0.32 |
| | | Analog | ADC12 | 100 | 2.287 | 0.23 |
| | | | DAC12 (2ch)[*4] | 100 | 0.869 | 0.09 |
| | | | TSN | 50 | 0.166 | 0.01 |
| | | Human machine interfaces | CTSU | 50 | 0.605 | 0.03 |
| | | Event link | ELC | 50 | 0.865 | 0.04 |
| | | Security | SCE9 | 100 | 218.100 | 21.81 |
| | | Data processing | CRC | 100 | 0.600 | 0.06 |
| | | | DOC | 100 | 0.388 | 0.04 |
| | | System | CAC | 50 | 0.844 | 0.04 |
| | | DMA | DMAC | 100 | 4.479 | 0.45 |
| | | | DTC | 100 | 4.274 | 0.43 |

Note 1. The values are guaranteed by design.
Note 2. LDO and Leak are internal voltage regulator's current and MCU's leakage current.
It is selected according to the temperature of Ta.
Note 3. Δ(Tj-Ta) = 20 °C is considered to measure the current.
Note 4. To determine the current consumption per channel, group or unit, divide Current [mA] by the number of channels, groups or units.

Table 47.13 shows the outline of operation for each unit.

**Table 47.13 Outline of operation for each unit (1 of 2)**

| Peripheral | Outline of operation |
|---|---|
| GPT | Operating modes is set to saw-wave PWM mode.<br>GPT is operating with PCLKD. |
| POEG | Only clear module stop bit. |
| AGT | AGT is operating with PCLKB. |
| RTC | RTC is operating with LOCO. |

**Table 47.13 Outline of operation for each unit (2 of 2)**

| Peripheral | Outline of operation |
|---|---|
| WDT | WDT is operating with PCLKB. |
| IWDT | IWDT is operating with IWDTCLK. |
| USBFS | Transfer types is set to bulk transfer.<br>USBFS is operating using Full-speed transfer (12 Mbps). |
| SCI | SCI is transmitting data in clock synchronous mode. |
| IIC | Communication format is set to I2C-bus format.<br>IIC is transmitting data in master mode. |
| CAN | CAN is transmitting and receiving data in self-test mode 1. |
| SPI | SPI mode is set to SPI operation (4-wire method).<br>SPI master/slave mode is set to master mode.<br>SPI is transmitting 8-bit width data. |
| QSPI | QSPI is issuing Fast Read Quad I/O Instruction. |
| SSIE | Communication mode is set to Master.<br>System word length is set to 32 bits.<br>Data word length is set to 20 bits.<br>SSIE is transmitting data using I2S format. |
| SDHI | Transfer bus mode is set to 4-bit wide bus mode.<br>SDHI is issuing CMD24 (single-block write). |
| ADC12 | Resolution is set to 12-bit accuracy.<br>Data registers is set to A/D-converted value addition mode.<br>ADC12 is converting the analog input in continuous scan mode. |
| DAC12 | DAC12 is outputting the conversion result while updating the value of data register. |
| TSN | TSN is operating. |
| CTSU | CTSU is operating in self-capacitance single scan mode. |
| ELC | Only clear module stop bit. |
| SCE9 | SCE9 is executing built-in self test. |
| CRC | CRC is generating CRC code using 32-bit CRC32-C polynomial. |
| DOC | DOC is operating in data addition mode. |
| CAC | Measurement target clocks is set to PCLKB.<br>Measurement reference clocks is set to PCLKB.<br>CAC is measuring the clock frequency accuracy. |
| DMAC | Bit length of transfer data is set to 32 bits.<br>Transfer mode is set to block transfer mode.<br>DMAC is transferring data from SRAM0 to SRAM0. |
| DTC | Bit length of transfer data is set to 32 bits.<br>Transfer mode is set to block transfer mode.<br>DTC is transferring data from SRAM0 to SRAM0. |

### 47.2.7.2 Example of $T_j$ calculation

Assumption :

- Package 100-pin LQFP : θja = 55.1 °C/W
- Ta = 100 °C
- $I_{CC}$max = 40 mA
- VCC = 3.5 V (VCC = AVCC = VCC_USB)
- $I_{OH}$ = 1 mA, $V_{OH}$ = VCC – 0.5 V, 12 Outputs
- $I_{OL}$ = 20 mA, $V_{OL}$ = 1.0 V, 8 Outputs
- $I_{OL}$ = 1 mA, $V_{OL}$ = 0.5 V, 12 Outputs

- $C_{in}$ = 8 pF, 16 pins, Input frequency = 10 MHz
- $C_{load}$ = 30 pF, 16 pins, Output frequency = 10 MHz

Leakage current of IO = Σ ($V_{OL}$ × $I_{OL}$) / Voltage + Σ ((VCC - $V_{OH}$) × $I_{OH}$) / Voltage
= (20 mA × 1 V) × 8 / 3.5 V + (1 mA × 0.5 V) × 12 / 3.5 V + ((VCC - (VCC - 0.5 V)) × 1 mA) × 12 / 3.5 V
= 45.7 mA + 1.71 mA + 1.71 mA
= 49.1 mA

Dynamic current of IO = Σ IO ($C_{in}$ + $C_{load}$) × IO switching frequency × Voltage
= ((8 pF × 16) × 10 MHz + (30 pF × 16) × 10 MHz) × 3.5 V
= 21.3 mA

Total power consumption = Voltage × (Leakage current + Dynamic current)
= (40 mA × 3.5 V) + (49.1 mA + 21.3 mA) × 3.5 V
= 386 mW (0.386 W)

Tj = Ta + θja × Total power consumption
= 100 °C + 55.1 °C/W × 0.386W
= 121.3 °C

## 47.3 AC Characteristics

### 47.3.1 Frequency

**Table 47.14 Operation frequency value in high-speed mode**

| Parameter | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Operation frequency | System clock (ICLK*2) | f | — | — | 100 | MHz |
| | Peripheral module clock (PCLKA)*2 | | — | — | 100 | |
| | Peripheral module clock (PCLKB)*2 | | — | — | 50 | |
| | Peripheral module clock (PCLKC)*2 | | —*3 | — | 50 | |
| | Peripheral module clock (PCLKD)*2 | | — | — | 100 | |
| | Flash interface clock (FCLK)*2 | | —*1 | — | 50 | |

Note 1. FCLK must run at a frequency of at least 4 MHz when programming or erasing the flash memory.
Note 2. See section 8, Clock Generation Circuit for the relationship between the ICLK, PCLKA, PCLKB, PCLKC, PCLKD, and FCLK frequencies.
Note 3. When the ADC12 is used, the PCLKC frequency must be at least 1 MHz.

**Table 47.15 Operation frequency value in low-speed mode**

| Parameter | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Operation frequency | System clock (ICLK)*2 | f | — | — | 1 | MHz |
| | Peripheral module clock (PCLKA)*2 | | — | — | 1 | |
| | Peripheral module clock (PCLKB)*2 | | — | — | 1 | |
| | Peripheral module clock (PCLKC)*2 *3 | | —*3 | — | 1 | |
| | Peripheral module clock (PCLKD)*2 | | — | — | 1 | |
| | Flash interface clock (FCLK)*1 *2 | | — | — | 1 | |

Note 1. Programming or erasing the flash memory is disabled in low-speed mode.
Note 2. See section 8, Clock Generation Circuit for the relationship between the ICLK, PCLKA, PCLKB, PCLKC, PCLKD, and FCLK frequencies.
Note 3. When the ADC12 is used, the PCLKC frequency must be set to at least 1 MHz.

**Table 47.16 Operation frequency value in Subosc-speed mode**

| Parameter | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Operation frequency | System clock (ICLK)*2 | f | 29.4 | — | 36.1 | kHz |
| | Peripheral module clock (PCLKA)*2 | | — | — | 36.1 | |
| | Peripheral module clock (PCLKB)*2 | | — | — | 36.1 | |
| | Peripheral module clock (PCLKC)*2 *3 | | — | — | 36.1 | |
| | Peripheral module clock (PCLKD)*2 | | — | — | 36.1 | |
| | Flash interface clock (FCLK)*1 *2 | | 29.4 | — | 36.1 | |

Note 1. Programming or erasing the flash memory is disabled in Subosc-speed mode.
Note 2. See section 8, Clock Generation Circuit for the relationship between the ICLK, PCLKA, PCLKB, PCLKC, PCLKD, and FCLK frequencies.
Note 3. The ADC12 cannot be used.

## 47.3.2 Clock Timing

**Table 47.17 Clock timing except for sub-clock oscillator (1 of 2)**

| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| EXTAL external clock input cycle time | | $t_{EXcyc}$ | 41.66 | — | — | ns | Figure 47.7 |
| EXTAL external clock input high pulse width | | $t_{EXH}$ | 15.83 | — | — | ns | |
| EXTAL external clock input low pulse width | | $t_{EXL}$ | 15.83 | — | — | ns | |
| EXTAL external clock rise time | | $t_{EXr}$ | — | — | 5.0 | ns | |
| EXTAL external clock fall time | | $t_{EXf}$ | — | — | 5.0 | ns | |
| Main clock oscillator frequency | | $f_{MAIN}$ | 8 | — | 24 | MHz | — |
| Main clock oscillation stabilization wait time (crystal)*1 | | $t_{MAINOSCWT}$ | — | — | —*1 | ms | Figure 47.8 |
| LOCO clock oscillation frequency | | $f_{LOCO}$ | 29.4912 | 32.768 | 36.0448 | kHz | — |
| LOCO clock oscillation stabilization wait time | | $t_{LOCOWT}$ | — | — | 60.4 | µs | Figure 47.9 |
| ILOCO clock oscillation frequency | | $f_{ILOCO}$ | 13.5 | 15 | 16.5 | kHz | — |
| MOCO clock oscillation frequency | | $f_{MOCO}$ | 6.8 | 8 | 9.2 | MHz | — |
| MOCO clock oscillation stabilization wait time | | $t_{MOCOWT}$ | — | — | 15.0 | µs | — |
| HOCO clock oscillator oscillation frequency | Without FLL | $f_{HOCO16}$ | 15.78 | 16 | 16.22 | MHz | –20 ≤ Ta ≤ 105°C |
| | | $f_{HOCO18}$ | 17.75 | 18 | 18.25 | | |
| | | $f_{HOCO20}$ | 19.72 | 20 | 20.28 | | |
| | | $f_{HOCO16}$ | 15.71 | 16 | 16.29 | | –40 ≤ Ta ≤ –20°C |
| | | $f_{HOCO18}$ | 17.68 | 18 | 18.32 | | |
| | | $f_{HOCO20}$ | 19.64 | 20 | 20.36 | | |
| | With FLL | $f_{HOCO16}$ | 15.960 | 16 | 16.040 | | –40 ≤ Ta ≤ 105°C<br>Sub-clock frequency accuracy is ±50 ppm. |
| | | $f_{HOCO18}$ | 17.955 | 18 | 18.045 | | |
| | | $f_{HOCO20}$ | 19.950 | 20 | 20.050 | | |
| HOCO clock oscillation stabilization wait time*2 | | $t_{HOCOWT}$ | — | — | 64.7 | µs | — |
| HOCO period jitter | | — | — | ±85 | — | ps | — |
| FLL stabilization wait time | | $t_{FLLWT}$ | — | — | 1.8 | ms | — |
| PLL clock frequency | | $f_{PLL}$ | 100 | — | 200 | MHz | — |
| PLL2 clock frequency | | $f_{PLL2}$ | 120 | — | 240 | MHz | — |
| PLL/PLL2 clock oscillation stabilization wait time | | $t_{PLLWT}$ | — | — | 174.9 | µs | Figure 47.10 |

**Table 47.17 Clock timing except for sub-clock oscillator (2 of 2)**

| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| PLL/PLL2 period jitter | $f_{PLL}$, $f_{PLL2}$ ≥ 120MHz | — | — | ±100 | — | ps | — |
| | $f_{PLL}$, $f_{PLL2}$ < 120MHz | — | — | ±120 | — | ps | — |
| PLL/PLL2 long term jitter | | — | — | ±300 | — | ps | Term: 1μs, 10μs |

Note 1. When setting up the main clock oscillator, ask the oscillator manufacturer for an oscillation evaluation, and use the results as the recommended oscillation stabilization time. Set the MOSCWTCR register to a value equal to or greater than the recommended value.
After changing the setting in the MOSCCR.MOSTP bit to start main clock operation, read the OSCSF.MOSCSF flag to confirm that it is 1, and then start using the main clock oscillator.

Note 2. This is the time from release from reset state until the HOCO oscillation frequency ($f_{HOCO}$) reaches the range for guaranteed operation.

**Table 47.18 Clock timing for the sub-clock oscillator**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| Sub-clock frequency | $f_{SUB}$ | — | 32.768 | — | kHz | — |
| Sub-clock oscillation stabilization wait time | $t_{SUBOSCWT}$ | — | — | —*1 | s | Figure 47.11 |

Note 1. When setting up the sub-clock oscillator, ask the oscillator manufacturer for an oscillation evaluation and use the results as the recommended oscillation stabilization time.
After changing the setting in the SOSCCR.SOSTP bit to start sub-clock operation, only start using the sub-clock oscillator after the sub-clock oscillation stabilization time elapses with an adequate margin. A value that is two times the value shown is recommended.

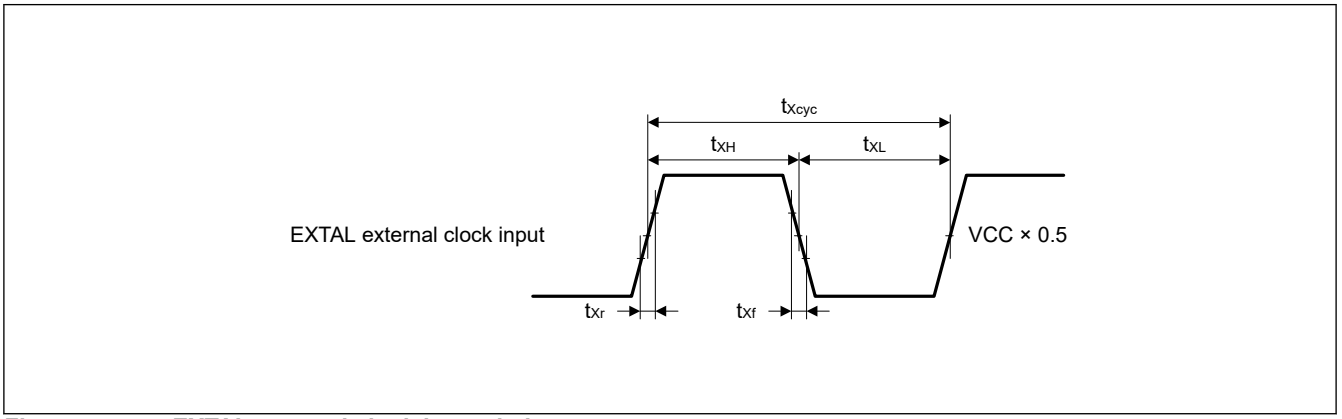



**Figure 47.7 EXTAL external clock input timing**

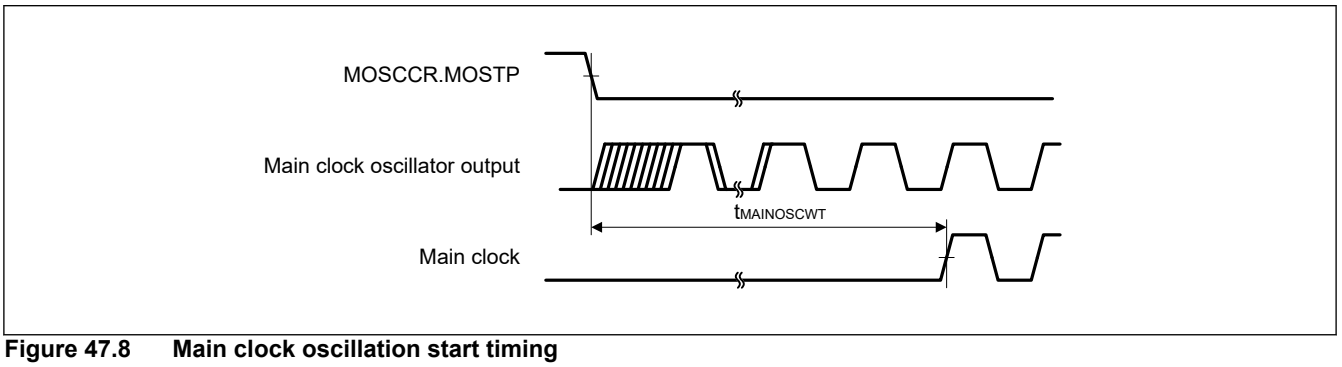



**Figure 47.8 Main clock oscillation start timing**

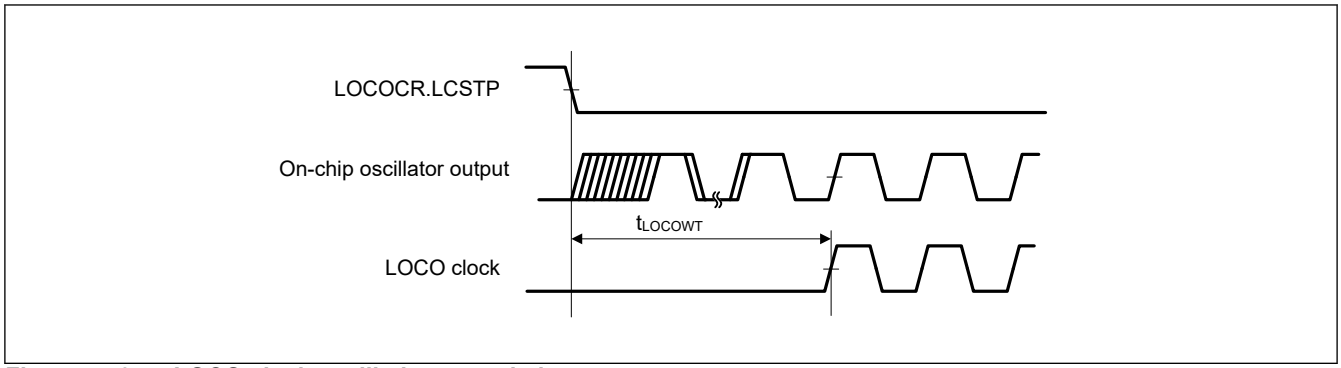



**Figure 47.9 LOCO clock oscillation start timing**

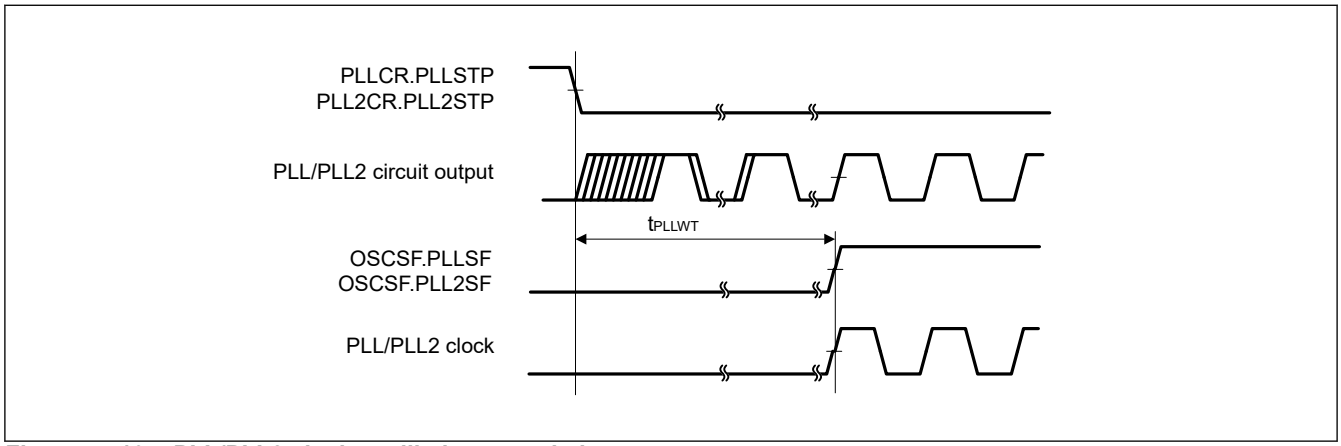



**Figure 47.10 PLL/PLL2 clock oscillation start timing**

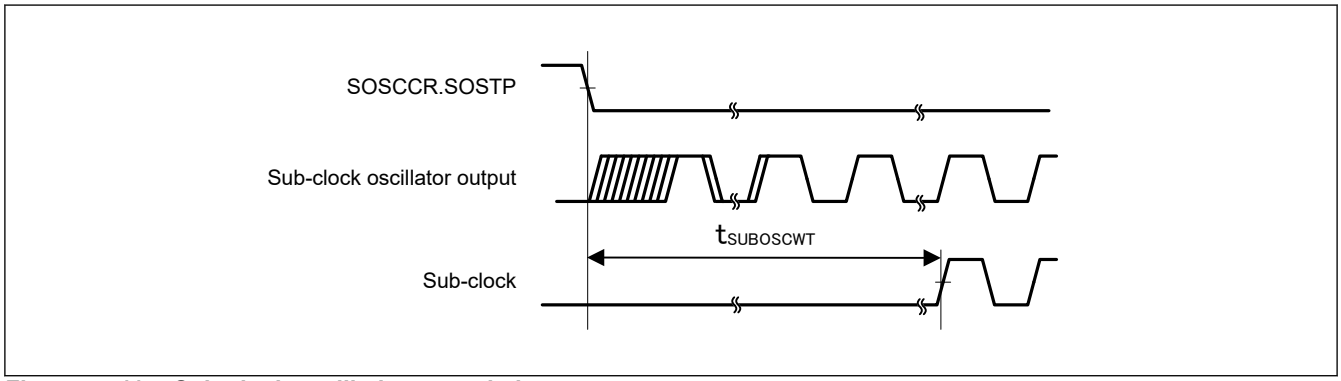



**Figure 47.11 Sub-clock oscillation start timing**

## 47.3.3 Reset Timing

**Table 47.19 Reset timing (1 of 2)**

| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| RES pulse width | Power-on | $t_{RESWP}$ | 0.7 | — | — | ms | Figure 47.12 |
| | Deep Software Standby mode | $t_{RESWD}$ | 0.6 | — | — | ms | Figure 47.13 |
| | Software Standby mode, Subosc-speed mode | $t_{RESWS}$ | 0.3 | — | — | ms | |
| | All other | $t_{RESW}$ | 200 | — | — | µs | |
| Wait time after RES cancellation | | $t_{RESWT}$ | — | 37.3 | 41.2 | µs | Figure 47.12 |

**Table 47.19 Reset timing (2 of 2)**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| Wait time after internal reset cancellation (IWDT reset, WDT reset, software reset, SRAM parity error reset, SRAM ECC error reset, bus master MPU error reset, TrustZone error reset) | $t_{RESW2}$ | — | 324 | 397.7 | µs | — |

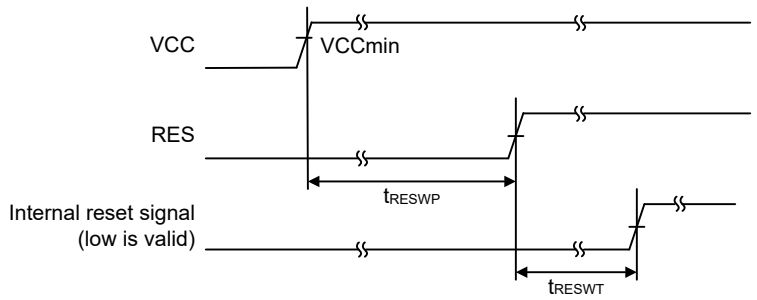



**Figure 47.12 RES pin input timing under the condition that VCC exceeds $V_{POR}$ voltage threshold**

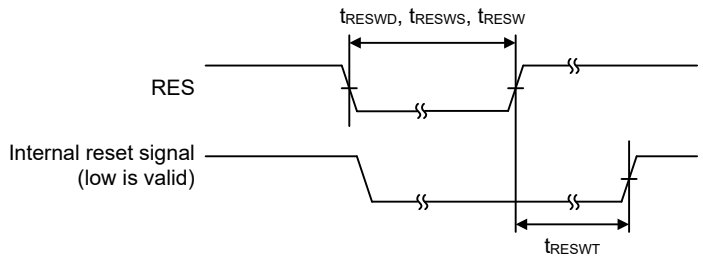



**Figure 47.13 Reset input timing**

## 47.3.4 Wakeup Timing

**Table 47.20 Timing of recovery from low power modes (1 of 2)**

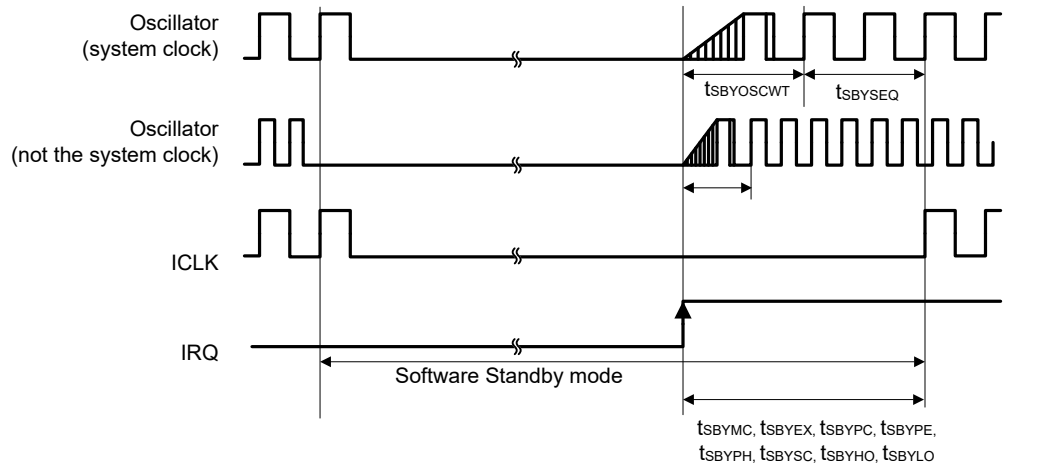
| Parameter | | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|
| Recovery time from Software Standby mode*1 | Crystal resonator connected to main clock oscillator | System clock source is main clock oscillator*2 | $t_{SBYMC}$*13 | — | 2.1 | 2.4 | ms | Figure 47.14 The division ratio of all oscillators is 1. |
| | | System clock source is PLL with main clock oscillator*3 | $t_{SBYPC}$*13 | — | 2.2 | 2.6 | ms | |
| | External clock input to main clock oscillator | System clock source is main clock oscillator*4 | $t_{SBYEX}$*13 | — | 45 | 125 | µs | |
| | | System clock source is PLL with main clock oscillator*5 | $t_{SBYPE}$*13 | — | 170 | 255 | µs | |
| | System clock source is sub-clock oscillator*6 *11 | | $t_{SBYSC}$*13 | — | 0.7 | 0.8 | ms | |
| | System clock source is LOCO*7 *11 | | $t_{SBYLO}$*13 | — | 0.7 | 0.9 | ms | |
| | System clock source is HOCO clock oscillator*8 | | $t_{SBYHO}$*13 | — | 55 | 130 | µs | |
| | System clock source is PLL with HOCO*9 | | $t_{SBYPH}$*13 | — | 175 | 265 | µs | |
| | System clock source is MOCO clock oscillator*10 | | $t_{SBYMO}$*13 | — | 35 | 65 | µs | |

**Table 47.20 Timing of recovery from low power modes (2 of 2)**

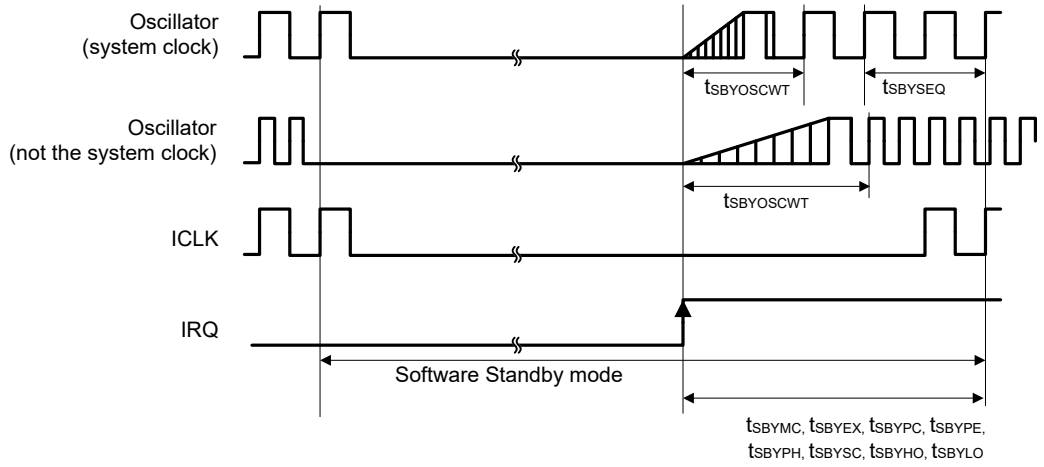
| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| Recovery time from Deep Software Standby mode | DPSBYCR.DEEPCUT[1] = 0 and DPSWCR.WTSTS[5:0] = 0x0E | $t_{DSBY}$ | — | 0.38 | 0.54 | ms | Figure 47.15 |
| | DPSBYCR.DEEPCUT[1] = 1 and DPSWCR.WTSTS[5:0] = 0x19 | $t_{DSBY}$ | — | 0.55 | 0.73 | ms | |
| Wait time after cancellation of Deep Software Standby mode | | $t_{DSBYWT}$ | 56 | — | 57 | $t_{cyc}$ | |
| Recovery time from Software Standby mode to Snooze mode | High-speed mode when system clock source is HOCO (20 MHz) | $t_{SNZ}$ | — | 35*12 | 70*12 | µs | Figure 47.16 |
| | High-speed mode when system clock source is MOCO (8 MHz) | $t_{SNZ}$ | — | 11*12 | 14*12 | µs | |

Note 1. The recovery time is determined by the system clock source. When multiple oscillators are active, the recovery time can be determined with the following equation:
Total recovery time = recovery time for an oscillator as the system clock source + the longest tSBYOSCWT in the active oscillators - tSBYOSCWT for the system clock + 2 LOCO cycles (when LOCO is operating) + Subosc is oscillating and MSTPC0 = 0 (CAC module stop))
Note 2. When the frequency of the crystal is 24 MHz (Main Clock Oscillator Wait Control Register (MOSCWTCR) is set to 0x05) and the greatest value of the internal clock division setting is 1.
Note 3. When the frequency of PLL is 200 MHz (Main Clock Oscillator Wait Control Register (MOSCWTCR) is set to 0x05) and the greatest value of the internal clock division setting is 4.
Note 4. When the frequency of the external clock is 24 MHz (Main Clock Oscillator Wait Control Register (MOSCWTCR) is set to 0x00) and the greatest value of the internal clock division setting is 1.
Note 5. When the frequency of PLL is 200 MHz (Main Clock Oscillator Wait Control Register (MOSCWTCR) is set to 0x00) and the greatest value of the internal clock division setting is 4.
Note 6. The Sub-clock oscillator frequency is 32.768 KHz and the greatest value of the internal clock division setting is 1.
Note 7. The LOCO frequency is 32.768 kHz and the greatest value of the internal clock division setting is 1.
Note 8. The HOCO frequency is 20 MHz and the greatest value of the internal clock division setting is 1.
Note 9. The PLL frequency is 200 MHz and the greatest value of the internal clock division setting is 4.
Note 10. The MOCO frequency is 8 MHz and the greatest value of the internal clock division setting is 1.
Note 11. In Subosc-speed mode, the sub-clock oscillator or LOCO continues oscillating in Software Standby mode.
Note 12. When the SNZCR.RXDREQEN bit is set to 0, the following time is added as the power supply recovery time: 16 µs (typical), 48 µs (maximum).
Note 13. The recovery time can be calculated with the equation of tSBYOSCWT + tSBYSEQ. And they can be determined with the following value and equation. For n, the greatest value is selected from among the internal clock division settings.

| Wakeup time | TYP | | MAX | | Unit |
|---|---|---|---|---|---|
| | tSBYOSCWT | tSBYSEQ | tSBYOSCWT | tSBYSEQ | |
| tSBYMC | (MSTS[7:0]*32 + 3) / 0.262 | 35 + 18 / fICLK + 4n / fMAIN | (MSTS[7:0]*32 + 14 / 0.236 | 62 + 18 / fICLK + 4n / fMAIN | µs |
| tSBYPC | (MSTS[7:0]*32 + 34) / 0.262 | 35 + 18 / fICLK + 4n / fPLL | (MSTS[7:0]*32 + 45) / 0.236 | 62 + 18 / fICLK + 4n / fPLL | µs |
| tSBYEX | 10 | 35 + 18 / fICLK + 4n / fEXMAIN | 62 | 62 + 18 / fICLK + 4n / fEXMAIN | µs |
| tSBYPE | 135 | 35 + 18 / fICLK + 4n / fPLL | 192 | 62 + 18 / fICLK + 4n / fPLL | µs |
| tSBYSC | 0 | 35 + 18 / fICLK + 4n / fSUB | 0 | 62 + 18 / fICLK + 4n / fSUB | µs |
| tSBYLO | 0 | 35 + 18 / fICLK + 4n / fLOCO | 0 | 62 + 18 / fICLK + 4n / fLOCO | µs |
| tSBYHO | 20 | 35 + 18 / fICLK + 4n / fHOCO | 67 | 62 + 18 / fICLK + 4n / fHOCO | µs |
| tSBYPH | 140 | 35 + 18 / fICLK + 4n / fPLL | 202 | 62 + 18 / fICLK + 4n / fPLL | µs |
| tSBYMO | 0 | 35 + 18 / fICLK + 4n / fMOCO | 0 | 62 + 18 / fICLK + 4n / fMOCO | µs |




When stabilization of the system clock oscillator is slower




When stabilization of an oscillator other than the system clock is slower

**Figure 47.14 Software Standby mode cancellation timing**

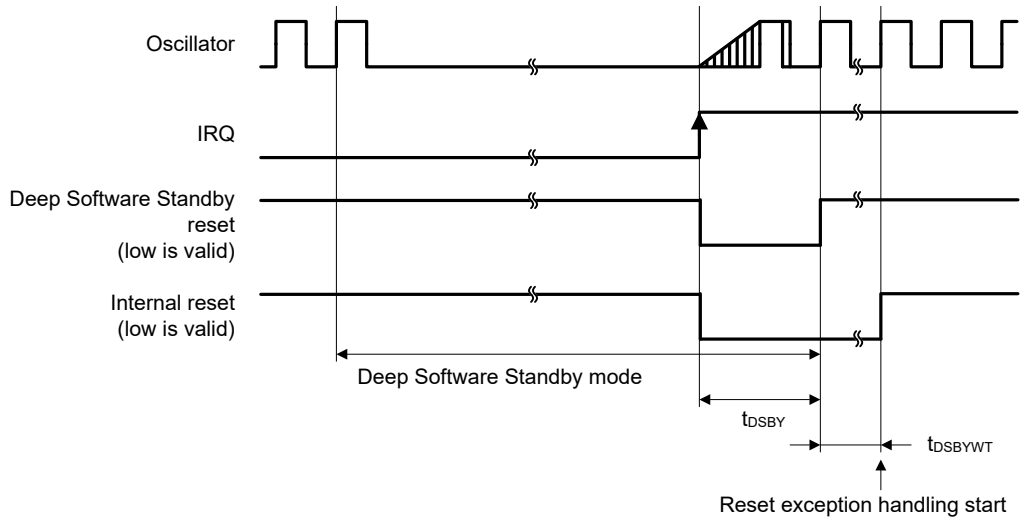
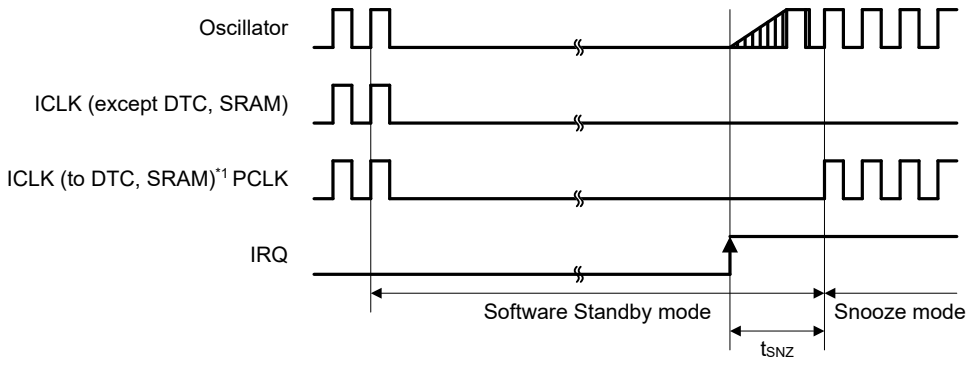



**Figure 47.15 Deep Software Standby mode cancellation timing**




Note 1. When SNZCR.SNZDTCEN bit is set to 1, ICLK is supplied to DTC and SRAM.

**Figure 47.16 Recovery timing from Software Standby mode to Snooze mode**

## 47.3.5 NMI and IRQ Noise Filter

**Table 47.21 NMI and IRQ noise filter**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions | |
|---|---|---|---|---|---|---|---|
| NMI pulse width | $t_{NMIW}$ | 200 | — | — | ns | NMI digital filter disabled | $t_{Pcyc} \times 2 \leq 200$ ns |
| | | $t_{Pcyc} \times 2$*1 | — | — | | | $t_{Pcyc} \times 2 > 200$ ns |
| | | 200 | — | — | | NMI digital filter enabled | $t_{NMICK} \times 3 \leq 200$ ns |
| | | $t_{NMICK} \times 3.5$*2 | — | — | | | $t_{NMICK} \times 3 > 200$ ns |
| IRQ pulse width | $t_{IRQW}$ | 200 | — | — | ns | IRQ digital filter disabled | $t_{Pcyc} \times 2 \leq 200$ ns |
| | | $t_{Pcyc} \times 2$*1 | — | — | | | $t_{Pcyc} \times 2 > 200$ ns |
| | | 200 | — | — | | IRQ digital filter enabled | $t_{IRQCK} \times 3 \leq 200$ ns |
| | | $t_{IRQCK} \times 3.5$*3 | — | — | | | $t_{IRQCK} \times 3 > 200$ ns |

Note: 200 ns minimum in Software Standby mode.
Note: If the clock source is switched, add 4 clock cycles of the switched source.
Note 1. $t_{Pcyc}$ indicates the PCLKB cycle.
Note 2. $t_{NMICK}$ indicates the cycle of the NMI digital filter sampling clock.
Note 3. $t_{IRQCK}$ indicates the cycle of the IRQi digital filter sampling clock.

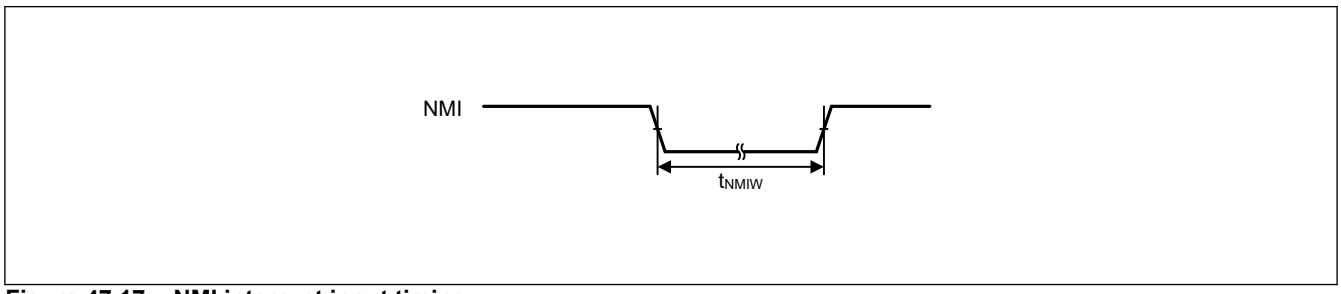



**Figure 47.17 NMI interrupt input timing**

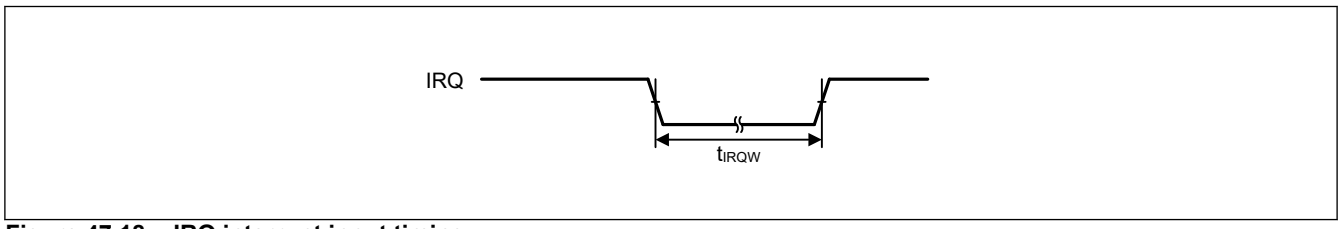



**Figure 47.18 IRQ interrupt input timing**

## 47.3.6 I/O Ports, POEG, GPT, AGT, and ADC12 Trigger Timing

**Table 47.22 I/O ports, POEG, GPT, AGT, and ADC12 trigger timing**

GPT32 Conditions:
High drive output is selected in the Port Drive Capability bit in the PmnPFS register.
AGT Conditions:
Middle drive output is selected in the Port Drive Capability bit in the PmnPFS register.

| Parameter | | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| I/O ports | Input data pulse width | | $t_{PRW}$ | 1.5 | — | $t_{Pcyc}$ | Figure 47.19 |
| POEG | POEG input trigger pulse width | | $t_{POEW}$ | 3 | — | $t_{Pcyc}$ | Figure 47.20 |
| GPT | Input capture pulse width | Single edge | $t_{GTICW}$ | 1.5 | — | $t_{PDcyc}$ | Figure 47.21 |
| | | Dual edge | | 2.5 | — | | |
| | GTIOCxY output skew (x = 0 to 3, Y = A or B) | Middle drive buffer | $t_{GTISK}$*1 | — | 4 | ns | Figure 47.22 |
| | | High drive buffer | | — | 4 | | |
| | GTIOCxY output skew (x = 4 to 7, Y = A or B) | Middle drive buffer | | — | 4 | | |
| | | High drive buffer | | — | 4 | | |
| | GTIOCxY output skew (x = 0 to 7, Y = A or B) | Middle drive buffer | | — | 6 | | |
| | | High drive buffer | | — | 6 | | |
| | OPS output skew GTOUUP, GTOULO, GTOVUP, GTOVLO, GTOWUP, GTOWLO | | $t_{GTOSK}$ | — | 5 | ns | Figure 47.23 |
| AGT | AGTIO, AGTEE input cycle | | $t_{ACYC}$*2 | 100 | — | ns | Figure 47.24 |
| | AGTIO, AGTEE input high width, low width | | $t_{ACKWH}$, $t_{ACKWL}$ | 40 | — | ns | |
| | AGTIO, AGTO, AGTOA, AGTOB output cycle | | $t_{ACYC2}$ | 62.5 | — | ns | |
| ADC12 | ADC12 trigger input pulse width | | $t_{TRGW}$ | 1.5 | — | $t_{Pcyc}$ | Figure 47.25 |

Note: $t_{Pcyc}$: PCLKB cycle, $t_{PDcyc}$: PCLKD cycle.
Note 1. This skew applies when the same driver I/O is used. If the I/O of the middle and high drivers is mixed, operation is not guaranteed.
Note 2. Constraints on input cycle:
When not switching the source clock: $t_{Pcyc} \times 2 < t_{ACYC}$ should be satisfied.

When switching the source clock: $t_{Pcyc} \times 6 < t_{ACYC}$ should be satisfied.

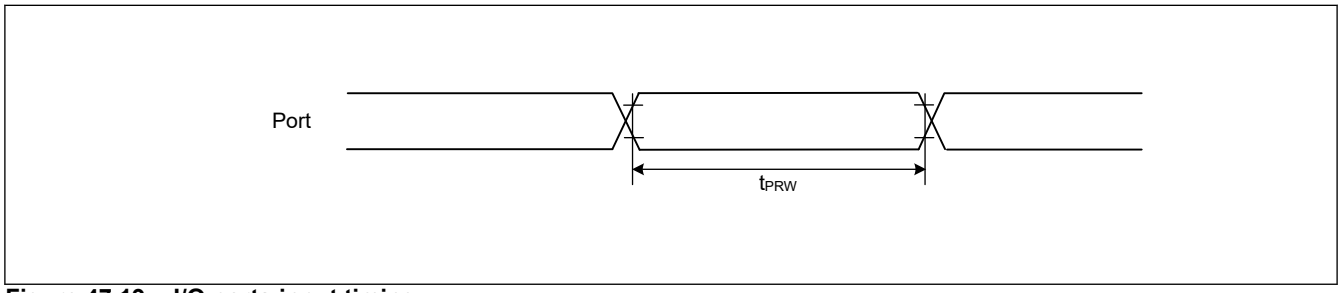



**Figure 47.19 I/O ports input timing**

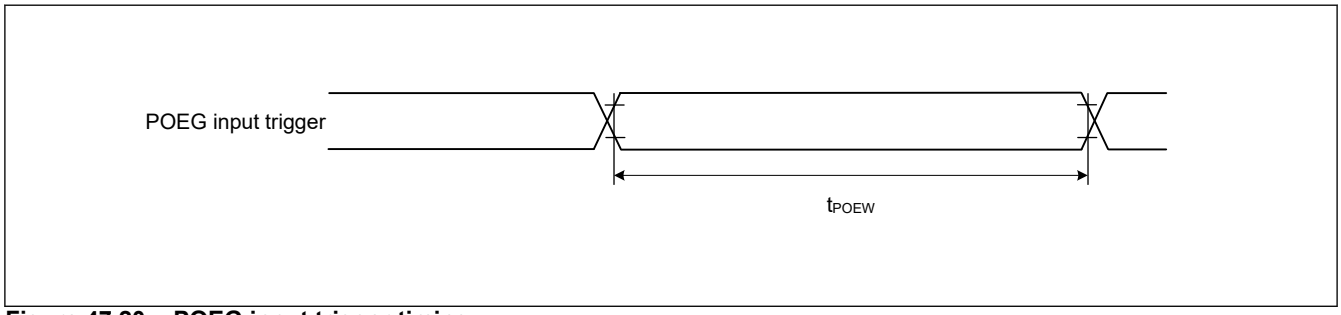



**Figure 47.20 POEG input trigger timing**

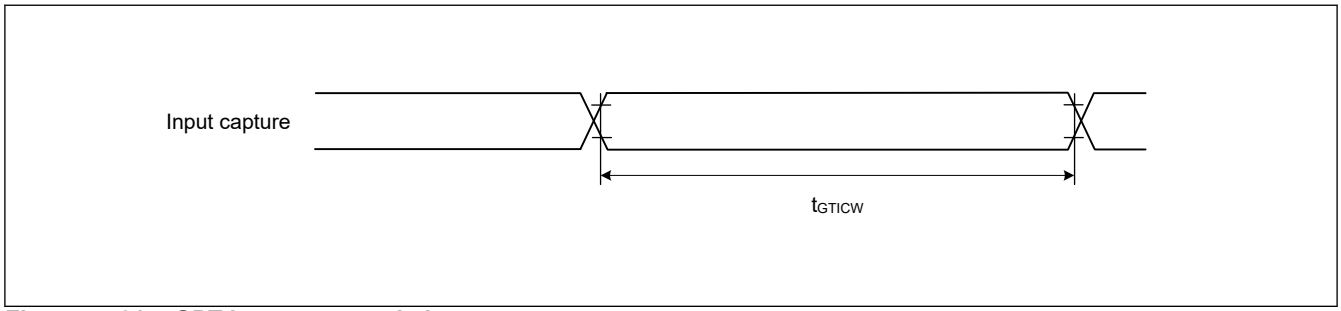



**Figure 47.21 GPT input capture timing**

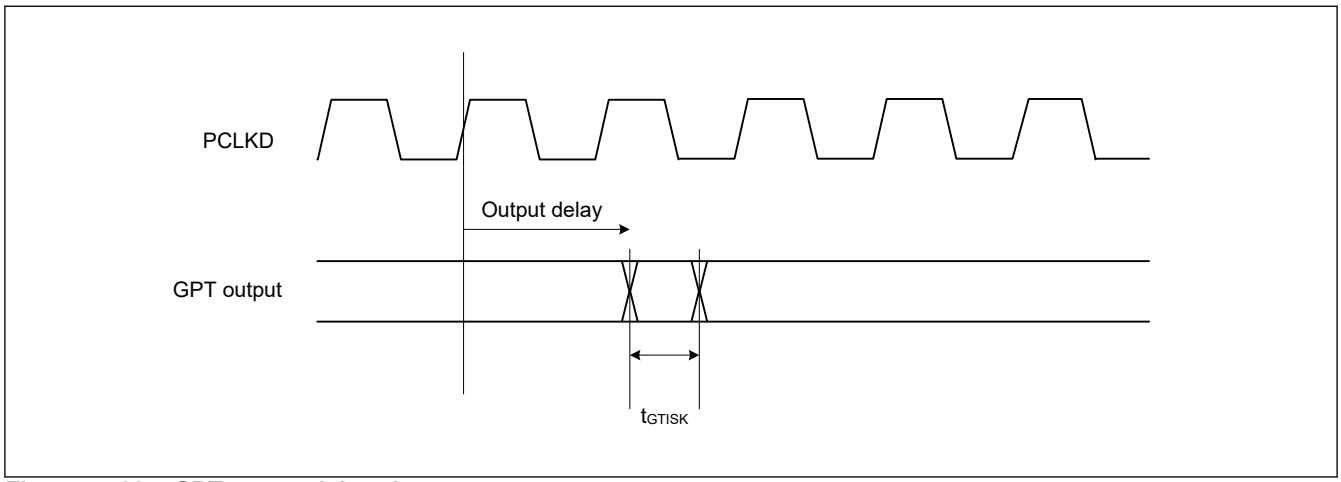



**Figure 47.22 GPT output delay skew**

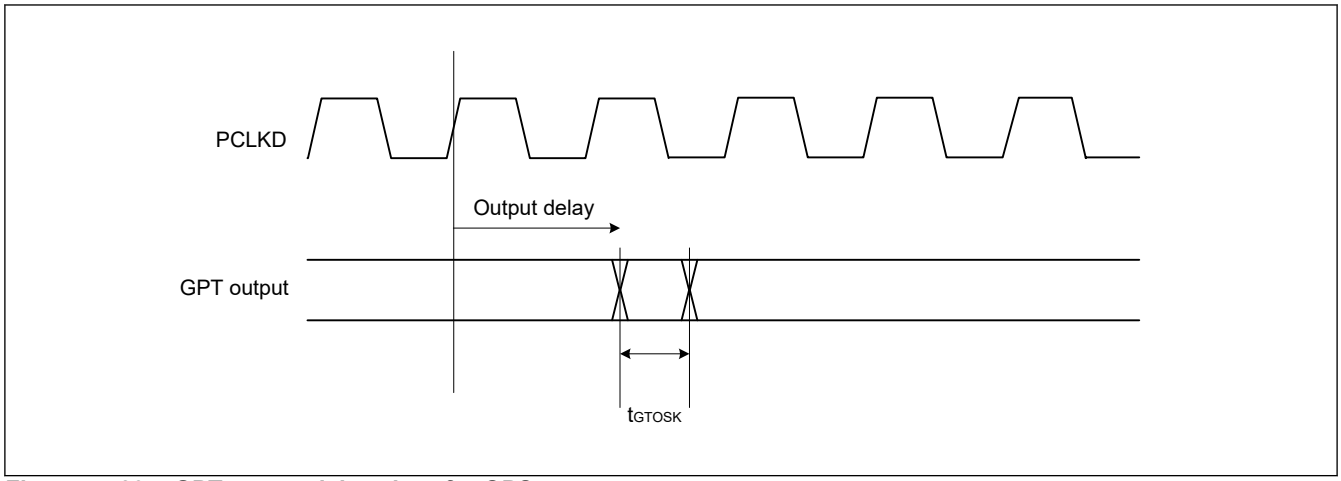



**Figure 47.23 GPT output delay skew for OPS**

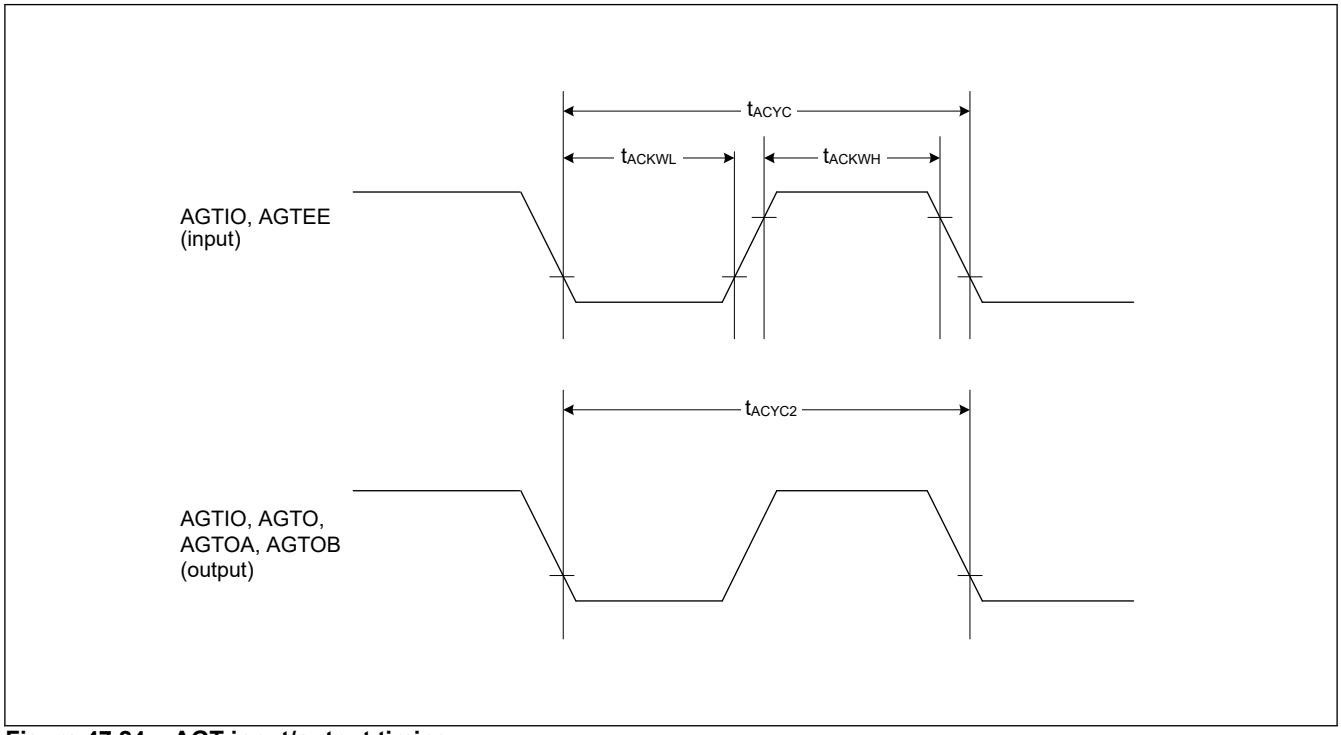



**Figure 47.24 AGT input/output timing**

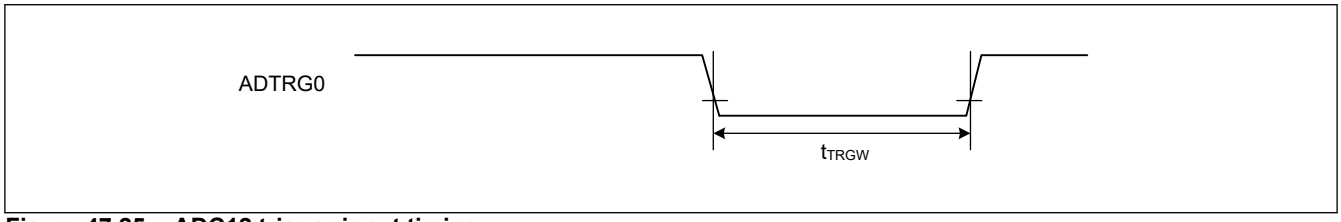



**Figure 47.25 ADC12 trigger input timing**

## 47.3.7 CAC Timing

**Table 47.23 CAC timing**

| Parameter | | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|
| CAC | CACREF input pulse width | $t_{PBcyc} \leq t_{cac}$ *1 | $t_{CACREF}$ | $4.5 \times t_{cac} + 3 \times t_{PBcyc}$ | — | — | ns | — |
| | | $t_{PBcyc} > t_{cac}$ *1 | | $5 \times t_{cac} + 6.5 \times t_{PBcyc}$ | — | — | ns | |

Note: $t_{PBcyc}$: PCLKB cycle.
Note 1. $t_{cac}$: CAC count clock source cycle.

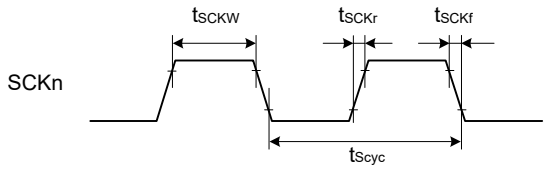
## 47.3.8 SCI Timing

**Table 47.24 SCI timing (1)**

Conditions: High drive output is selected in the Port Drive Capability bit in the PmnPFS register.

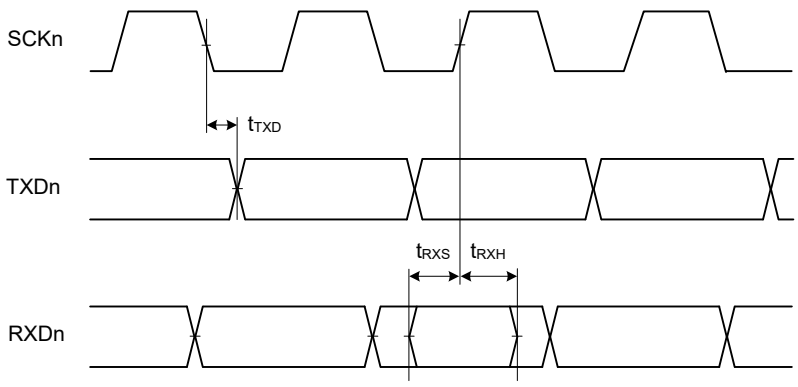
| Parameter | | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| SCI | Input clock cycle | Asynchronous | $t_{Scyc}$ | 4 | — | $t_{Pcyc}$ | Figure 47.26 |
| | | Clock synchronous | | 6 | — | | |
| | Input clock pulse width | | $t_{SCKW}$ | 0.4 | 0.6 | $t_{Scyc}$ | |
| | Input clock rise time | | $t_{SCKr}$ | — | 5 | ns | |
| | Input clock fall time | | $t_{SCKf}$ | — | 5 | ns | |
| | Output clock cycle | Asynchronous | $t_{Scyc}$ | 6 (other than SCI1, SCI2)<br>8 (SCI1, SCI2) | — | $t_{Pcyc}$ | |
| | | Clock synchronous | | 4 | — | | |
| | Output clock pulse width | | $t_{SCKW}$ | 0.4 | 0.6 | $t_{Scyc}$ | |
| | Output clock rise time | | $t_{SCKr}$ | — | 5 | ns | |
| | Output clock fall time | | $t_{SCKf}$ | — | 5 | ns | |
| | Transmit data delay | Clock synchronous master mode (internal clock) | $t_{TXD}$ | — | 5 | ns | Figure 47.27 |
| | | Clock synchronous slave mode (external clock) | $t_{TXD}$ | — | 25 | ns | |
| | Receive data setup time | Clock synchronous master mode (internal clock) | $t_{RXS}$ | 15 | — | ns | |
| | | Clock synchronous slave mode (external clock) | $t_{RXS}$ | 5 | — | ns | |
| | Receive data hold time | Clock synchronous | $t_{RXH}$ | 5 | — | ns | |

Note: $t_{Pcyc}$: PCLKA cycle.




Note: n = 0 to 4, 9

**Figure 47.26 SCK clock input/output timing**




Note: n = 0 to 4, 9

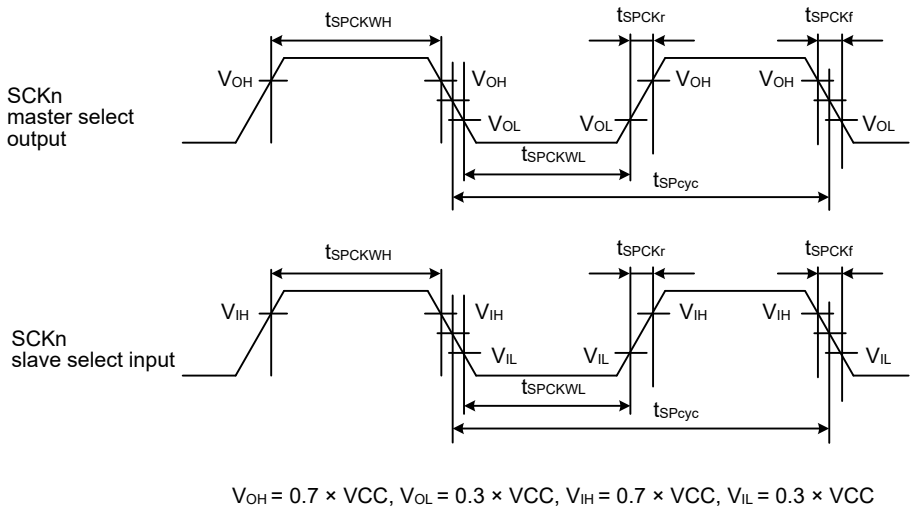
**Figure 47.27 SCI input/output timing in clock synchronous mode**

**Table 47.25 SCI timing (2)**

Conditions: High drive output is selected in the Port Drive Capability bit in the PmnPFS register.

| Parameter | | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| Simple SPI | SCK clock cycle output (master) | | $t_{SPcyc}$ | 4 | 65536 | $t_{Pcyc}$ | Figure 47.28 |
| | SCK clock cycle input (slave) | | | 6 | 65536 | | |
| | SCK clock high pulse width | | $t_{SPCKWH}$ | 0.4 | 0.6 | $t_{SPcyc}$ | |
| | SCK clock low pulse width | | $t_{SPCKWL}$ | 0.4 | 0.6 | $t_{SPcyc}$ | |
| | SCK clock rise and fall time | | $t_{SPCKr}$, $t_{SPCKf}$ | — | 5 | ns | |
| | Data input setup time | master | $t_{SU}$ | 15 | — | ns | Figure 47.29 to Figure 47.32 |
| | | slave | | 5 | — | ns | |
| | Data input hold time | | $t_{H}$ | 5 | — | ns | |
| | SS input setup time | | $t_{LEAD}$ | 1 | — | $t_{SPcyc}$ | |
| | SS input hold time | | $t_{LAG}$ | 1 | — | $t_{SPcyc}$ | |
| | Data output delay | master | $t_{OD}$ | — | 5 | ns | |
| | | slave | | — | 25 | ns | |
| | Data output hold time | | $t_{OH}$ | -5 | — | ns | |
| | Data rise and fall time | | $t_{Dr}$, $t_{Df}$ | — | 5 | ns | |
| | SS input rise and fall time | | $t_{SSLr}$, $t_{SSLf}$ | — | 5 | ns | |
| | Slave access time | | $t_{SA}$ | — | 3 × $t_{Pcyc}$ + 25 | ns | Figure 47.32 |
| | Slave output release time | | $t_{REL}$ | — | 3 × $t_{Pcyc}$ + 25 | ns | |

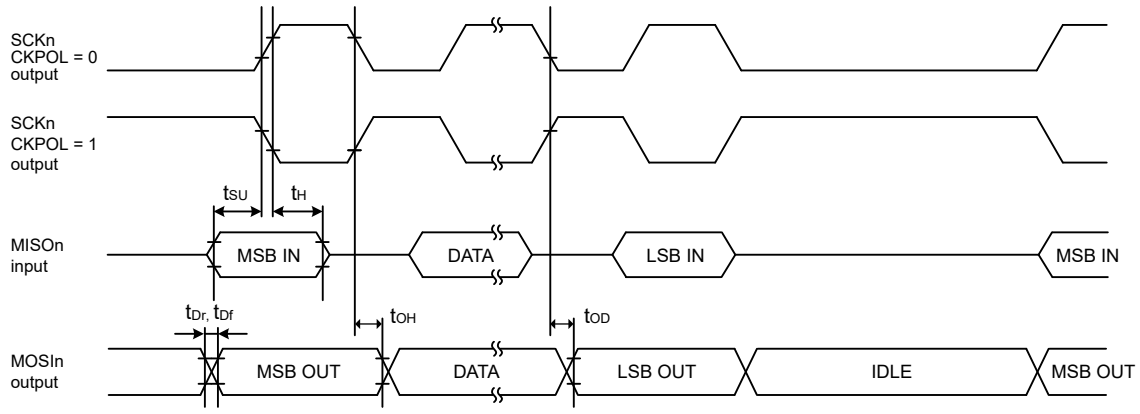
Note: $t_{Pcyc}$: PCLKA cycle.




$V_{OH}$ = 0.7 × VCC, $V_{OL}$ = 0.3 × VCC, $V_{IH}$ = 0.7 × VCC, $V_{IL}$ = 0.3 × VCC

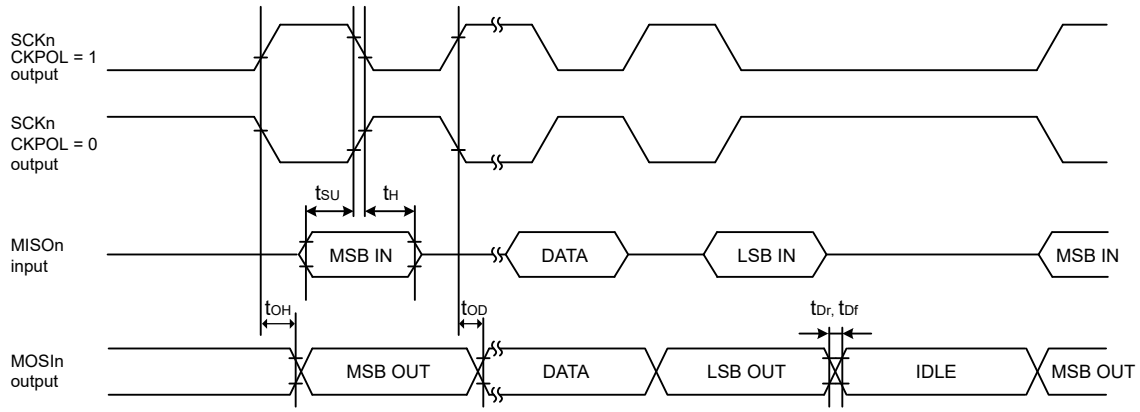
Note: n = 0 to 4, 9

**Figure 47.28 SCI simple SPI mode clock timing**



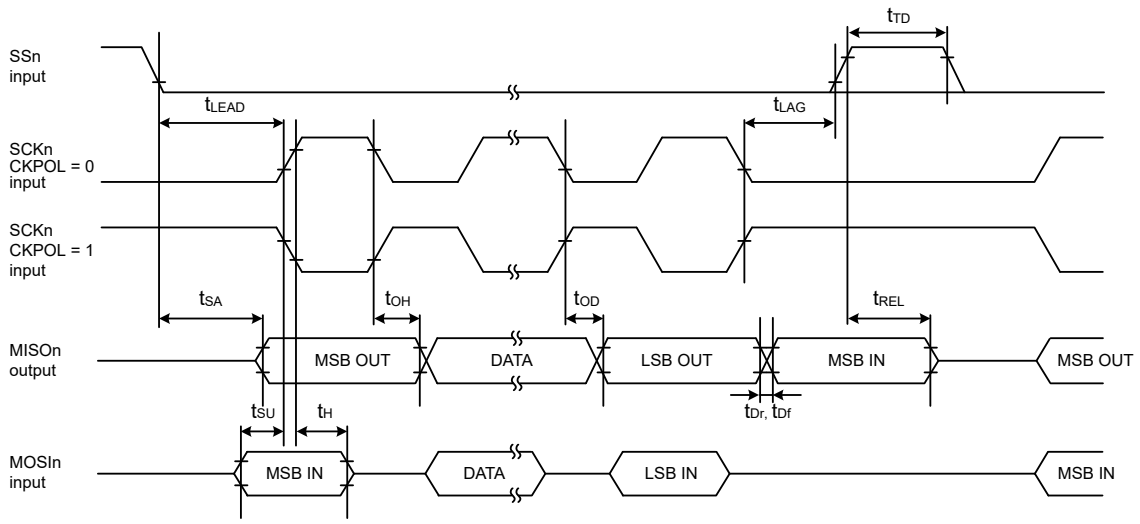

Note: n = 0 to 4, 9

**Figure 47.29 SCI simple SPI mode timing for master when CKPH = 1**



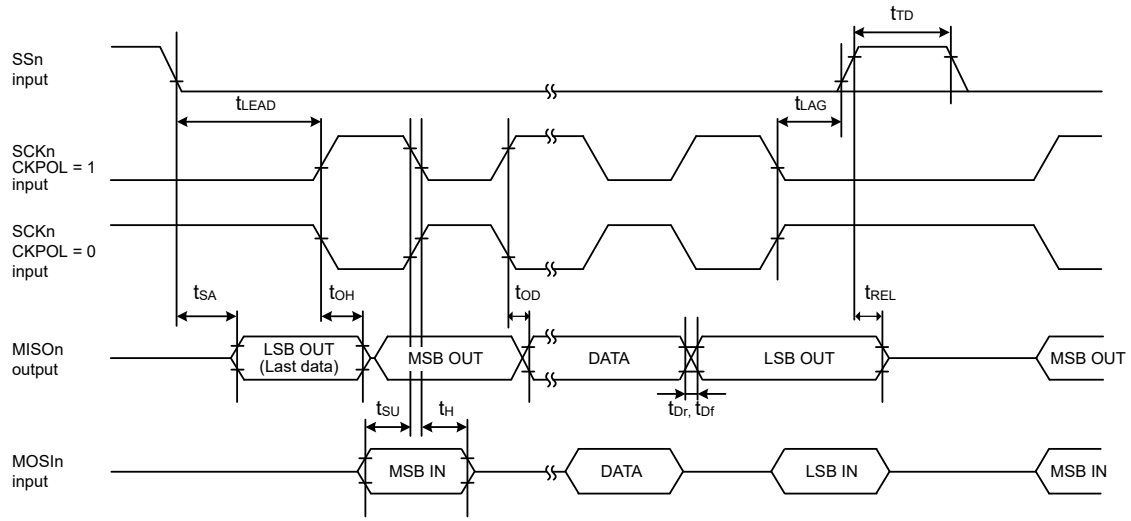

Note: n = 0 to 4, 9

**Figure 47.30 SCI simple SPI mode timing for master when CKPH = 0**




Note: n = 0 to 4, 9

**Figure 47.31 SCI simple SPI mode timing for slave when CKPH = 1**




Note: n = 0 to 4, 9

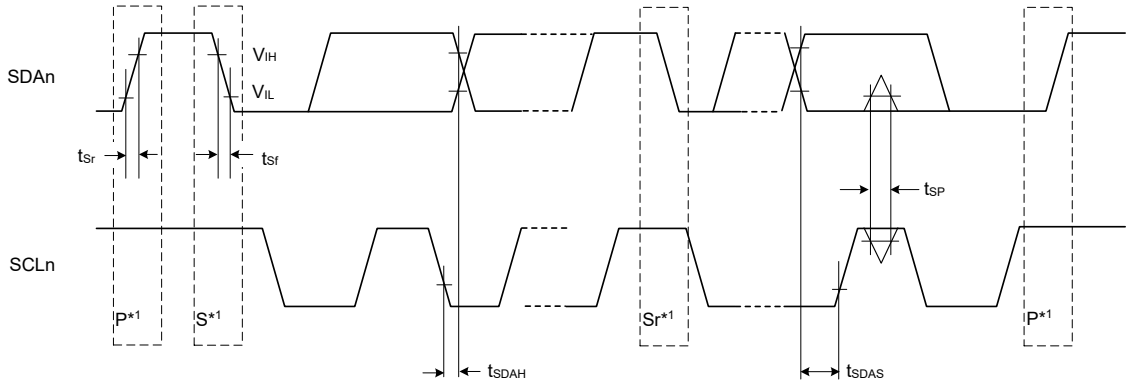
**Figure 47.32 SCI simple SPI mode timing for slave when CKPH = 0**

**Table 47.26 SCI timing (3)**

Conditions: Middle drive output is selected in the Port Drive Capability bit in the PmnPFS register.

| Parameter | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| Simple IIC (Standard mode) | SDA input rise time | $t_{Sr}$ | — | 1000 | ns | Figure 47.33 |
| | SDA input fall time | $t_{Sf}$ | — | 300 | ns | |
| | SDA input spike pulse removal time | $t_{SP}$ | 0 | 4 × $t_{IICcyc}$ | ns | |
| | Data input setup time | $t_{SDAS}$ | 250 | — | ns | |
| | Data input hold time | $t_{SDAH}$ | 0 | — | ns | |
| | SCL, SDA capacitive load | $C_b$*1 | — | 400 | pF | |
| Simple IIC (Fast mode) | SDA input rise time | $t_{Sr}$ | — | 300 | ns | Figure 47.33 |
| | SDA input fall time | $t_{Sf}$ | — | 300 | ns | |
| | SDA input spike pulse removal time | $t_{SP}$ | 0 | 4 × $t_{IICcyc}$ | ns | |
| | Data input setup time | $t_{SDAS}$ | 100 | — | ns | |
| | Data input hold time | $t_{SDAH}$ | 0 | — | ns | |
| | SCL, SDA capacitive load | $C_b$*1 | — | 400 | pF | |

Note: $t_{IICcyc}$: IIC internal reference clock (IICφ) cycle.
Note 1. $C_b$ indicates the total capacity of the bus line.




Test conditions:
$V_{IH}$ = VCC × 0.7, $V_{IL}$ = VCC × 0.3
$V_{OL}$ = 0.6 V, $I_{OL}$ = 6 mA

Note: n = 0 to 4, 9
Note 1. S, P, and Sr indicate the following conditions:
S: Start condition
P: Stop condition
Sr: Restart condition

**Figure 47.33 SCI simple IIC mode timing**

## 47.3.9 SPI Timing

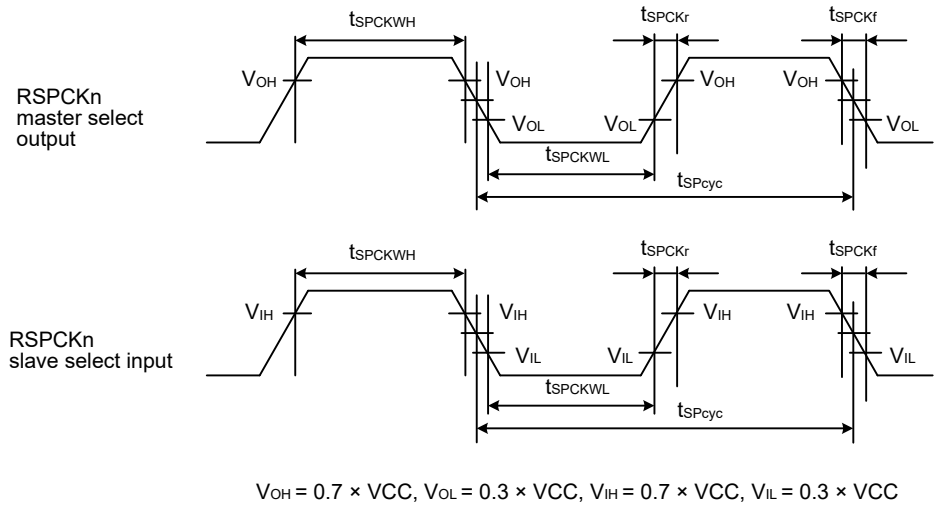
**Table 47.27 SPI timing**

Conditions: High drive output is selected in the Port Drive Capability bit in the PmnPFS register.

| Parameter | | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| SPI | RSPCK clock cycle | Master | $t_{SPcyc}$ | 2 | 4096 | $t_{Pcyc}$ | Figure 47.34 |
| | | Slave | | 4 | 4096 | | |
| | RSPCK clock high pulse width | Master | $t_{SPCKWH}$ | $(t_{SPcyc} - t_{SPCKr} - t_{SPCKf}) / 2 - 3$ | — | ns | |
| | | Slave | | 0.4 | 0.6 | $t_{SPcyc}$ | |
| | RSPCK clock low pulse width | Master | $t_{SPCKWL}$ | $(t_{SPcyc} - t_{SPCKr} - t_{SPCKf}) / 2 - 3$ | — | ns | |
| | | Slave | | 0.4 | 0.6 | $t_{SPcyc}$ | |
| | RSPCK clock rise and fall time | Master | $t_{SPCKr}$, $t_{SPCKf}$ | — | 5 | ns | |
| | | Slave | | — | 1 | µs | |
| | Data input setup time | Master | $t_{SU}$ | 4 | — | ns | Figure 47.35 to Figure 47.40 |
| | | Slave | | 5 | — | | |
| | Data input hold time | Master (PCLKA division ratio set to 1/2) | $t_{HF}$ | 0 | — | ns | |
| | | Master (PCLKA division ratio set to a value other than 1/2) | $t_{H}$ | $t_{Pcyc}$ | — | | |
| | | Slave | $t_{H}$ | 20 | — | | |
| | SSL setup time | Master | $t_{LEAD}$ | $N \times t_{SPcyc} - 10^{*1}$ | $N \times t_{SPcyc} + 100^{*1}$ | ns | |
| | | Slave | | $4 \times t_{Pcyc}$ | — | ns | |
| | SSL hold time | Master | $t_{LAG}$ | $N \times t_{SPcyc} - 10^{*2}$ | $N \times t_{SPcyc} + 100^{*2}$ | ns | |
| | | Slave | | $4 \times t_{Pcyc}$ | — | ns | |
| | Data output delay | Master | $t_{OD1}$ | — | 6.3 | ns | |
| | | | $t_{OD2}$ | — | 6.3 | | |
| | | Slave | $t_{OD}$ | — | 20 | | |
| | Data output hold time | Master | $t_{OH}$ | 0 | — | ns | |
| | | Slave | | 0 | — | | |
| | Successive transmission delay | Master | $t_{TD}$ | $t_{SPcyc} + 2 \times t_{Pcyc}$ | $8 \times t_{SPcyc} + 2 \times t_{Pcyc}$ | ns | |
| | | Slave | | $4 \times t_{Pcyc}$ | | | |
| | MOSI and MISO rise and fall time | Output | $t_{Dr}$, $t_{Df}$ | — | 5 | ns | |
| | | Input | | — | 1 | µs | |
| | SSL rise and fall time | Output | $t_{SSLr}$, $t_{SSLf}$ | — | 5 | ns | |
| | | Input | | — | 1 | µs | |
| | Slave access time | | $t_{SA}$ | — | 25 | ns | Figure 47.39 and Figure 47.40 |
| | Slave output release time | | $t_{REL}$ | — | 25 | | |

Note: $t_{Pcyc}$: PCLKA cycle.

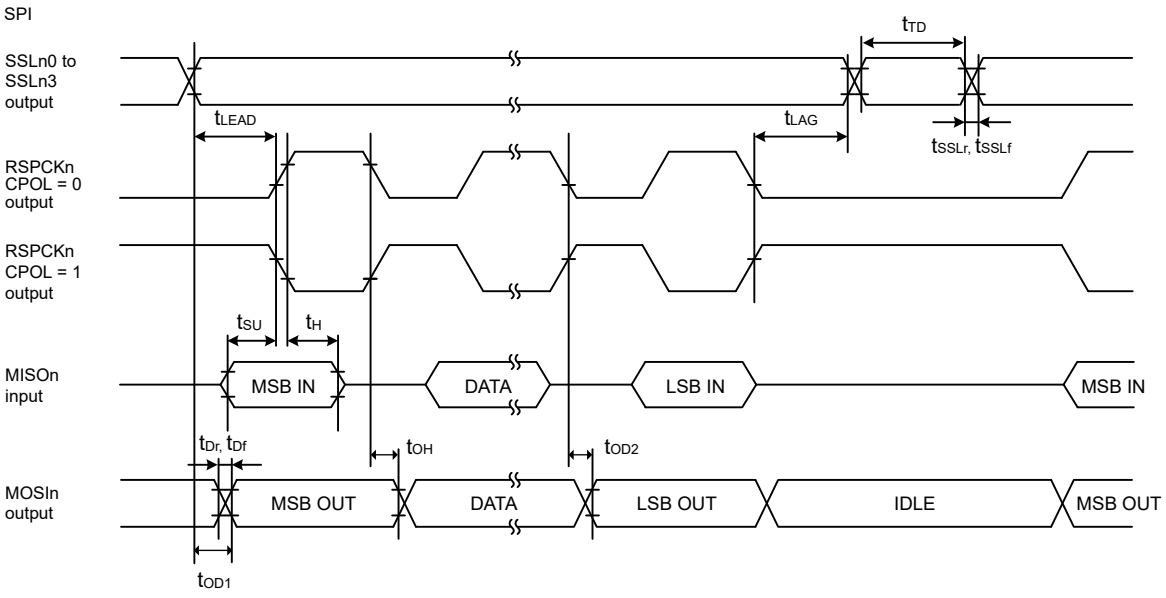
Note: Must use pins that have a letter appended to their name, for instance “_A”, “_B”, to indicate group membership. For the SPI interface, the AC portion of the electrical characteristics is measured for each group.
Note 1. N is set to an integer from 1 to 8 by the SPCKD register.
Note 2. N is set to an integer from 1 to 8 by the SSLND register.




$V_{OH}$ = 0.7 × VCC, $V_{OL}$ = 0.3 × VCC, $V_{IH}$ = 0.7 × VCC, $V_{IL}$ = 0.3 × VCC

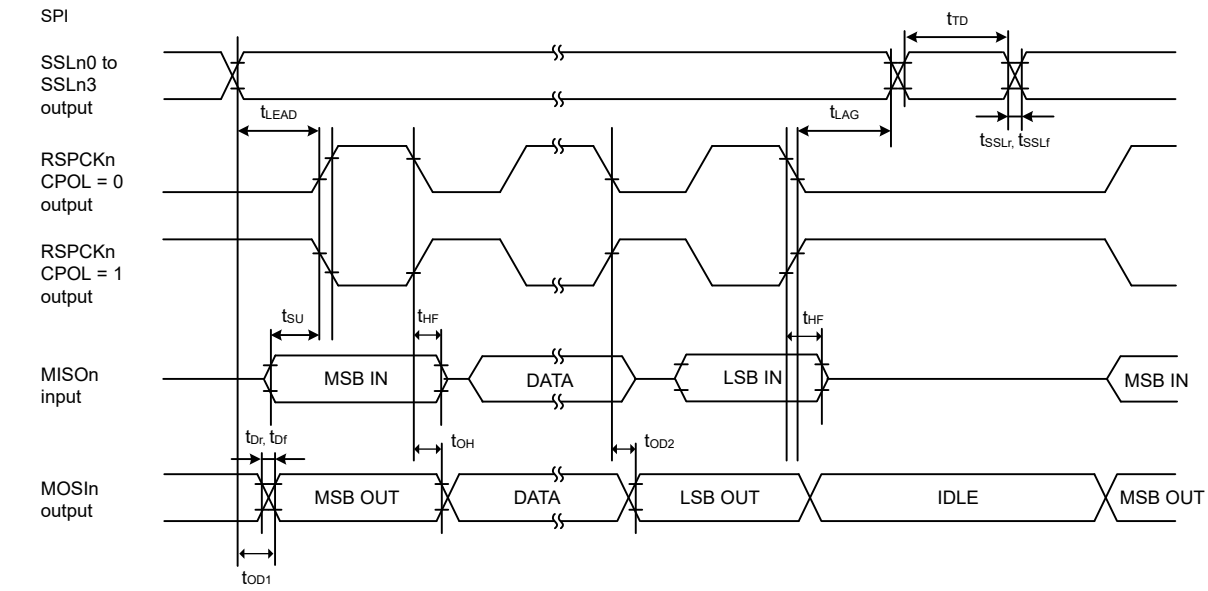
Note: n = A

**Figure 47.34 SPI clock timing**



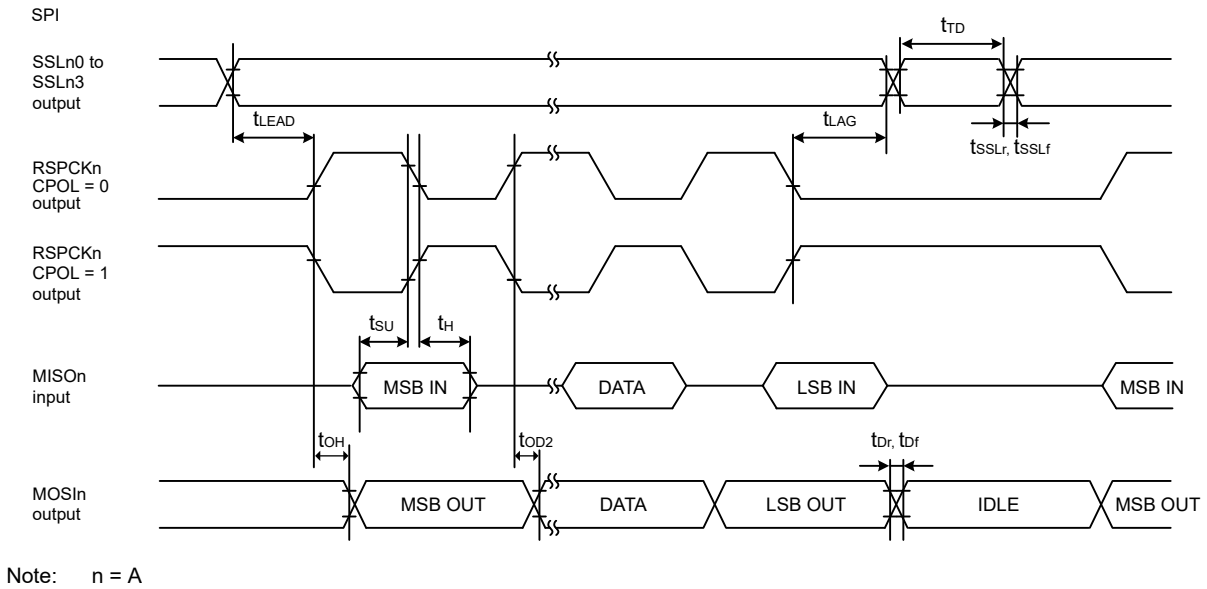

Note: n = A

**Figure 47.35 SPI timing for master when CPHA = 0**




Note: n = A

**Figure 47.36 SPI timing for master when CPHA = 0 and the bit rate is set to PCLKA/2**




Note: n = A

**Figure 47.37 SPI timing for master when CPHA = 1**

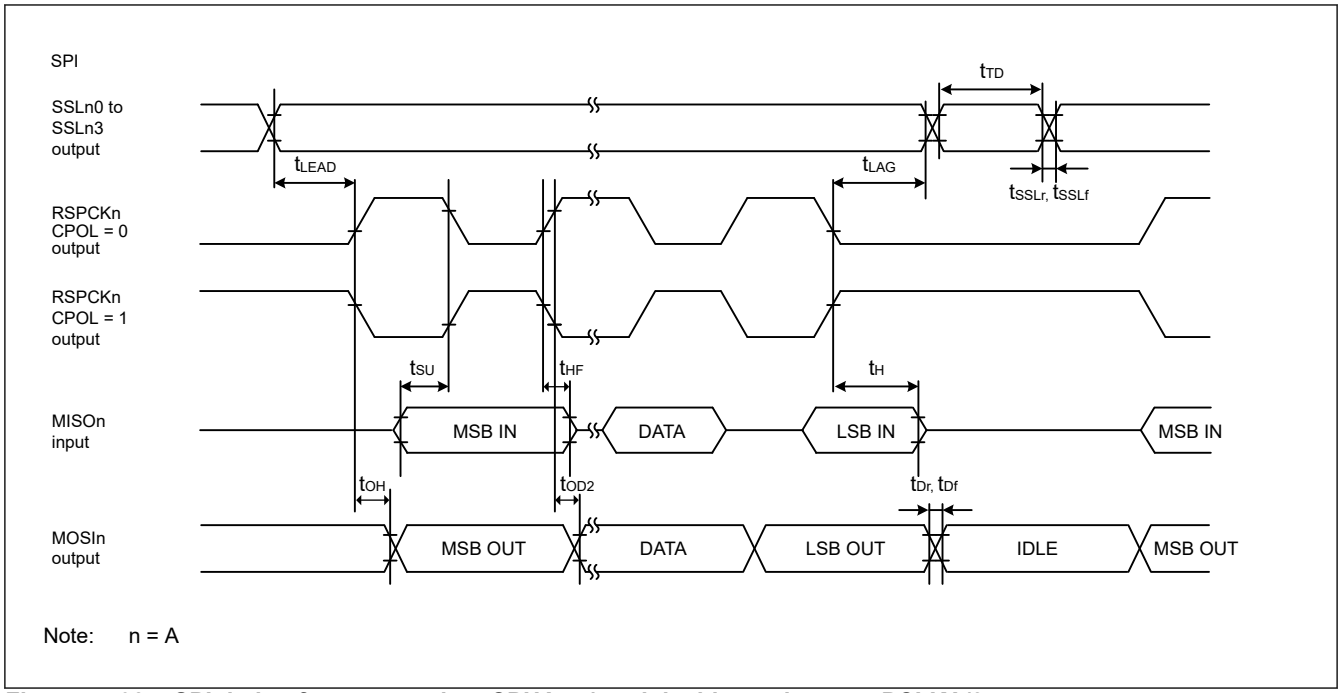



**Figure 47.38 SPI timing for master when CPHA = 1 and the bit rate is set to PCLKA/2**

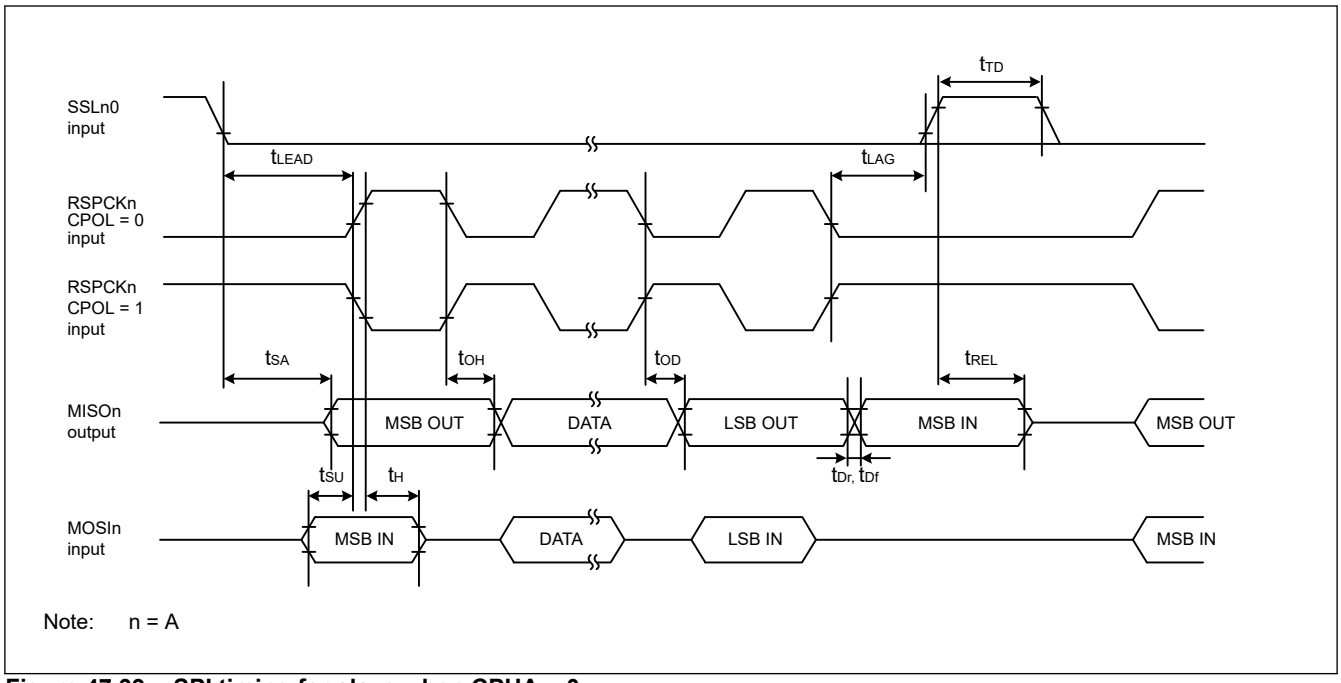
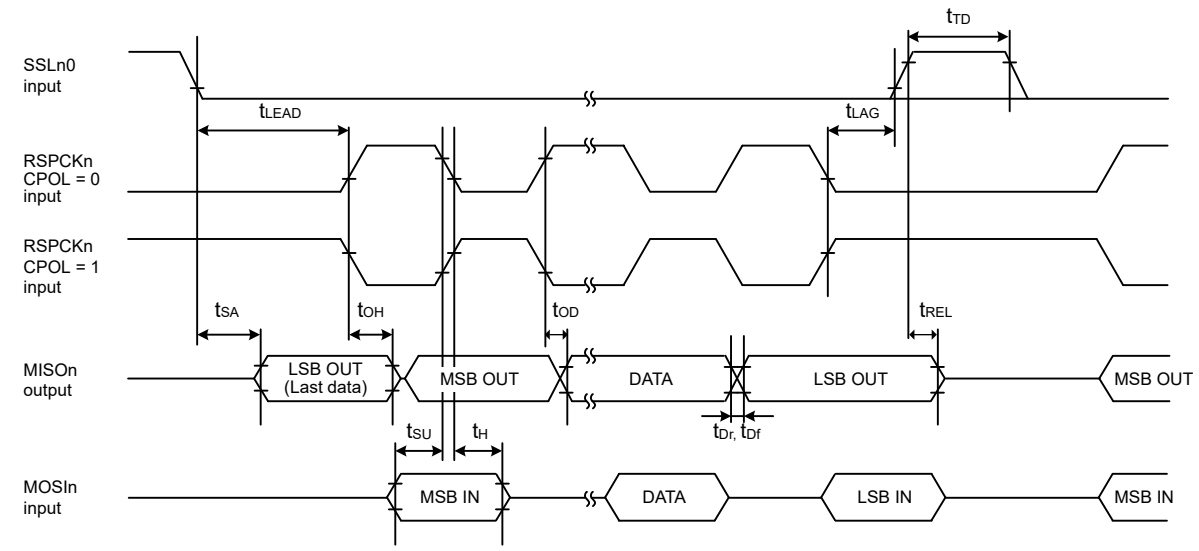



**Figure 47.39 SPI timing for slave when CPHA = 0**




Note: n = A

**Figure 47.40 SPI timing for slave when CPHA = 1**

## 47.3.10 QSPI Timing

**Table 47.28 QSPI timing**

Conditions: High drive output is selected in the Port Drive Capability bit in the PmnPFS register.

| Parameter | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| QSPI | QSPCK clock cycle | $t_{QScyc}$ | 2 | 48 | $t_{Pcyc}$ | Figure 47.41 |
| | QSPCK clock high pulse width | $t_{QSWH}$ | $t_{QScyc}$ × 0.4 | — | ns | |
| | QSPCK clock low pulse width | $t_{QSWL}$ | $t_{QScyc}$ × 0.4 | — | ns | |
| | Data input setup time | $t_{Su}$ | 10 | — | ns | Figure 47.42 |
| | Data input hold time | $t_{IH}$ | 0 | — | ns | |
| | QSSL setup time | $t_{LEAD}$ | (N + 0.5) × $t_{Qscyc}$ - 5*1 | (N + 0.5) × $t_{Qscyc}$ + 100*1 | ns | |
| | QSSL hold time | $t_{LAG}$ | (N + 0.5) × $t_{Qscyc}$ - 5*2 | (N + 0.5) × $t_{Qscyc}$ + 100*2 | ns | |
| | Data output delay | $t_{OD}$ | — | 4 | ns | |
| | Data output hold time | $t_{OH}$ | –3.3 | — | ns | |
| | Successive transmission delay | $t_{TD}$ | 1 | 16 | $t_{QScyc}$ | |

Note: $t_{Pcyc}$: PCLKA cycle.
Note 1. N is set to 0 or 1 in SFMSLD.
Note 2. N is set to 0 or 1 in SFMSHD.

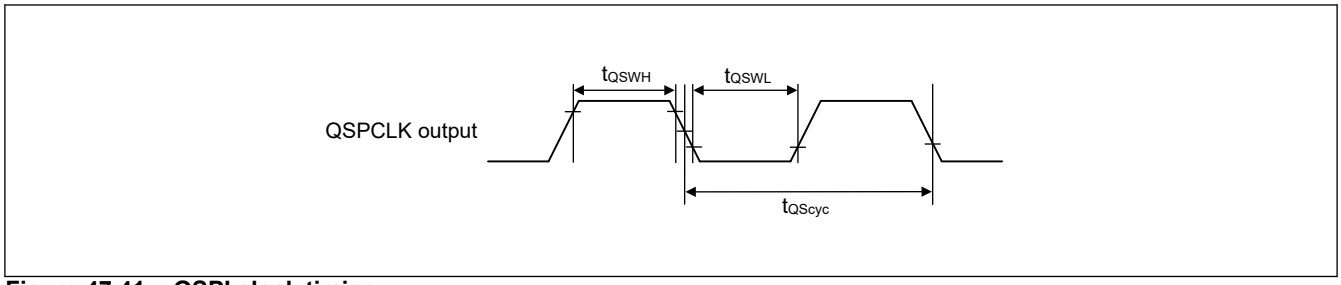



**Figure 47.41 QSPI clock timing**

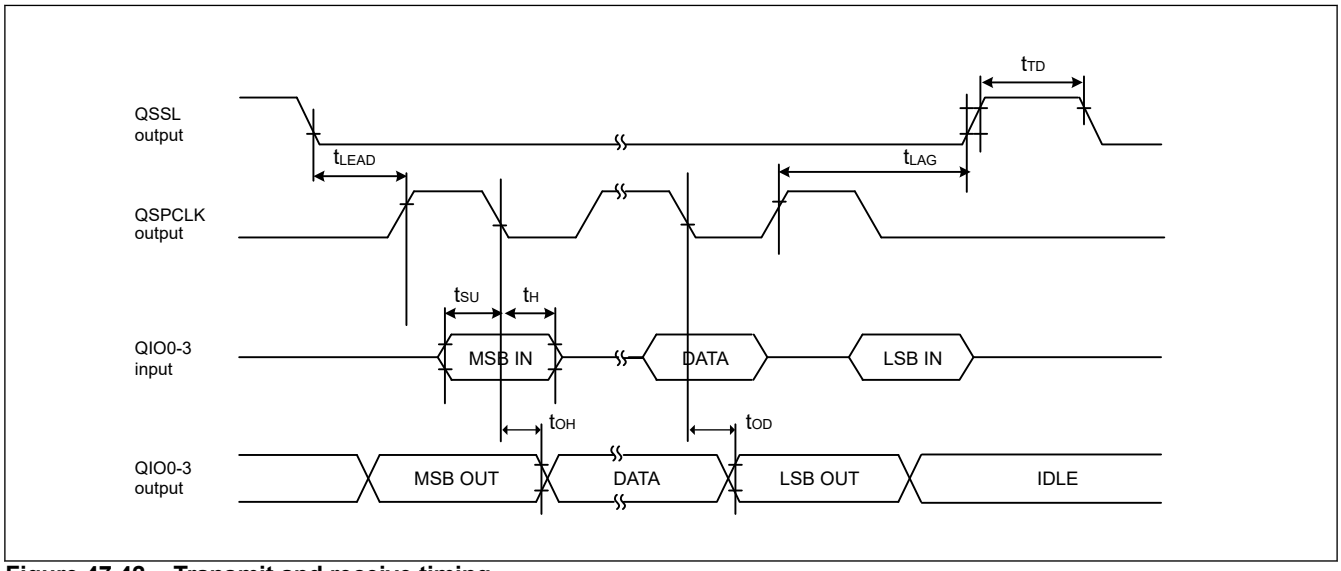



**Figure 47.42 Transmit and receive timing**

## 47.3.11 IIC Timing

**Table 47.29 IIC timing (1) (1 of 2)**

(1) Conditions: Middle drive output is selected in the Port Drive Capability bit in the PmnPFS register for the following pins: SDA0_B, SCL0_B, SDA1_B, SCL1_B.
(2) The following pins do not require setting: SCL0_A, SDA0_A.
(3) Use pins that have a letter appended to their names, for instance "_A" or "_B", to indicate group membership. For the IIC interface, the AC portion of the electrical characteristics is measured for each group.

| Parameter | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| IIC (Standard mode, SMBus) ICFER.FMPE = 0 | SCL input cycle time | $t_{SCL}$ | 6 (12) × $t_{IICcyc}$ + 1300 | — | ns | Figure 47.43 |
| | SCL input high pulse width | $t_{SCLH}$ | 3 (6) × $t_{IICcyc}$ + 300 | — | ns | |
| | SCL input low pulse width | $t_{SCLL}$ | 3 (6) × $t_{IICcyc}$ + 300 | — | ns | |
| | SCL, SDA rise time | $t_{Sr}$ | — | 1000 | ns | |
| | SCL, SDA fall time | $t_{Sf}$ | — | 300 | ns | |
| | SCL, SDA input spike pulse removal time | $t_{SP}$ | 0 | 1 (4) × $t_{IICcyc}$ | ns | |
| | SDA input bus free time when wakeup function is disabled | $t_{BUF}$ | 3 (6) × $t_{IICcyc}$ + 300 | — | ns | |
| | SDA input bus free time when wakeup function is enabled | $t_{BUF}$ | 3 (6) × $t_{IICcyc}$ + 4 × $t_{Pcyc}$ + 300 | — | ns | |
| | START condition input hold time when wakeup function is disabled | $t_{STAH}$ | $t_{IICcyc}$ + 300 | — | ns | |
| | START condition input hold time when wakeup function is enabled | $t_{STAH}$ | 1 (5) × $t_{IICcyc}$ + $t_{Pcyc}$ + 300 | — | ns | |
| | Repeated START condition input setup time | $t_{STAS}$ | 1000 | — | ns | |
| | STOP condition input setup time | $t_{STOS}$ | 1000 | — | ns | |
| | Data input setup time | $t_{SDAS}$ | $t_{IICcyc}$ + 50 | — | ns | |
| | Data input hold time | $t_{SDAH}$ | 0 | — | ns | |
| | SCL, SDA capacitive load | $C_b$ *2 | — | 400 | pF | |

**Table 47.29 IIC timing (1) (2 of 2)**

(1) Conditions: Middle drive output is selected in the Port Drive Capability bit in the PmnPFS register for the following pins: SDA0_B, SCL0_B, SDA1_B, SCL1_B.
(2) The following pins do not require setting: SCL0_A, SDA0_A.
(3) Use pins that have a letter appended to their names, for instance "_A" or "_B", to indicate group membership. For the IIC interface, the AC portion of the electrical characteristics is measured for each group.

| Parameter | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| IIC (Fast mode) | SCL input cycle time | $t_{SCL}$ | 6 (12) × $t_{IICcyc}$ + 600 | — | ns | Figure 47.43 |
| | SCL input high pulse width | $t_{SCLH}$ | 3 (6) × $t_{IICcyc}$ + 300 | — | ns | |
| | SCL input low pulse width | $t_{SCLL}$ | 3 (6) × $t_{IICcyc}$ + 300 | — | ns | |
| | SCL, SDA rise time | $t_{Sr}$ | 20 × (external pullup voltage/5.5V)*1 | 300 | ns | |
| | SCL, SDA fall time | $t_{Sf}$ | 20 × (external pullup voltage/5.5V)*1 | 300 | ns | |
| | SCL, SDA input spike pulse removal time | $t_{SP}$ | 0 | 1 (4) × $t_{IICcyc}$ | ns | |
| | SDA input bus free time when wakeup function is disabled | $t_{BUF}$ | 3 (6) × $t_{IICcyc}$ + 300 | — | ns | |
| | SDA input bus free time when wakeup function is enabled | $t_{BUF}$ | 3 (6) × $t_{IICcyc}$ + 4 × $t_{Pcyc}$ + 300 | — | ns | |
| | START condition input hold time when wakeup function is disabled | $t_{STAH}$ | $t_{IICcyc}$ + 300 | — | ns | |
| | START condition input hold time when wakeup function is enabled | $t_{STAH}$ | 1 (5) × $t_{IICcyc}$ + $t_{Pcyc}$ + 300 | — | ns | |
| | Repeated START condition input setup time | $t_{STAS}$ | 300 | — | ns | |
| | STOP condition input setup time | $t_{STOS}$ | 300 | — | ns | |
| | Data input setup time | $t_{SDAS}$ | $t_{IICcyc}$ + 50 | — | ns | |
| | Data input hold time | $t_{SDAH}$ | 0 | — | ns | |
| | SCL, SDA capacitive load | $C_b$*2 | — | 400 | pF | |

Note: $t_{IICcyc}$: IIC internal reference clock (IICφ) cycle, $t_{Pcyc}$: PCLKB cycle.
Note: Values in parentheses apply when ICMR3.NF[1:0] is set to 11b while the digital filter is enabled with ICFER.NFE set to 1.
Note: Must use pins that have a letter appended to their name, for instance "_A", "_B", to indicate group membership. For the IIC interface, the AC portion of the electrical characteristics is measured for each group.
Note 1. Only supported for SCL0_A and SDA0_A.
Note 2. Cb indicates the total capacity of the bus line.

**Table 47.30 IIC timing (2)**

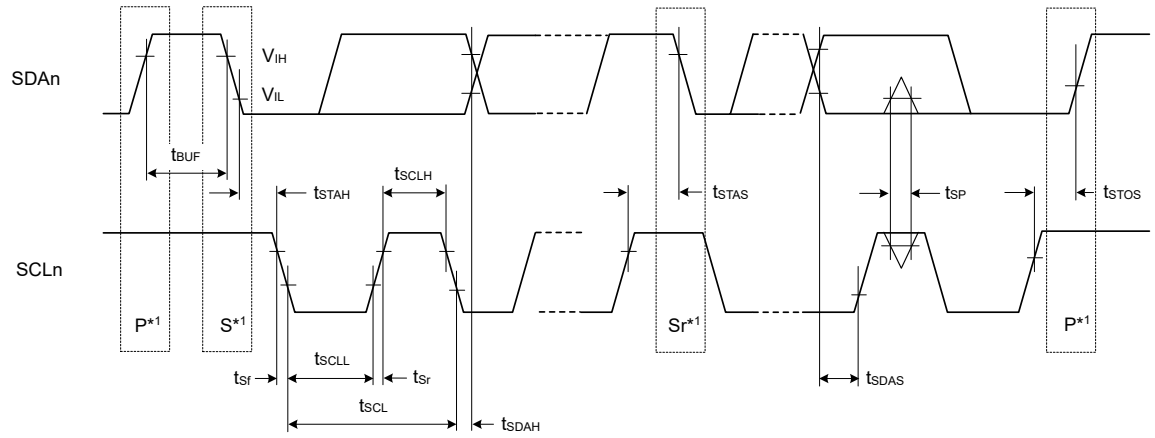
Setting of the SCL0_A, SDA0_A pins is not required with the Port Drive Capability bit in the PmnPFS register.

| Parameter | | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| IIC (Fast-mode+) ICFER.FMPE = 1 | SCL input cycle time | $t_{SCL}$ | 6 (12) × $t_{IICcyc}$ + 240 | — | ns | Figure 47.43 |
| | SCL input high pulse width | $t_{SCLH}$ | 3 (6) × $t_{IICcyc}$ + 120 | — | ns | |
| | SCL input low pulse width | $t_{SCLL}$ | 3 (6) × $t_{IICcyc}$ + 120 | — | ns | |
| | SCL, SDA rise time | $t_{Sr}$ | — | 120 | ns | |
| | SCL, SDA fall time | $t_{Sf}$ | 20 × (external pullup voltage/ 5.5V) | 120 | ns | |
| | SCL, SDA input spike pulse removal time | $t_{SP}$ | 0 | 1 (4) × tIICcyc | ns | |
| | SDA input bus free time when wakeup function is disabled | $t_{BUF}$ | 3 (6) × $t_{IICcyc}$ + 120 | — | ns | |
| | SDA input bus free time when wakeup function is enabled | $t_{BUF}$ | 3 (6) × $t_{IICcyc}$ + 4 × $t_{Pcyc}$ + 120 | — | ns | |
| | Start condition input hold time when wakeup function is disabled | $t_{STAH}$ | $t_{IICcyc}$ + 120 | — | ns | |
| | START condition input hold time when wakeup function is enabled | $t_{STAH}$ | 1 (5) × $t_{IICcyc}$ + $t_{Pcyc}$ + 120 | — | ns | |
| | Restart condition input setup time | $t_{STAS}$ | 120 | — | ns | |
| | Stop condition input setup time | $t_{STOS}$ | 120 | — | ns | |
| | Data input setup time | $t_{SDAS}$ | $t_{IICcyc}$ + 30 | — | ns | |
| | Data input hold time | $t_{SDAH}$ | 0 | — | ns | |
| | SCL, SDA capacitive load | $C_b$ *1 | — | 550 | pF | |

Note: $t_{IICcyc}$: IIC internal reference clock (IICφ) cycle, $t_{Pcyc}$: PCLKB cycle.

Note: Values in parentheses apply when ICMR3.NF[1:0] is set to 11b while the digital filter is enabled with ICFER.NFE set to 1.

Note 1. Cb indicates the total capacity of the bus line.




Note: n = 0, 1
Note 1. S, P, and Sr indicate the following conditions:
S: Start condition
P: Stop condition
Sr: Restart condition

**Figure 47.43 I²C bus interface input/output timing**

## 47.3.12 SSIE Timing

**Table 47.31 SSIE timing**

(1) High drive output is selected with the Port Drive Capability bit in the PmnPFS register.
(2) Use pins that have a letter appended to their names, for instance "_A" or "_B" to indicate group membership. For the SSIE interface, the AC portion of the electrical characteristics is measured for each group.

| Parameter | | | Symbol | Target specification | | Unit | Comments |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Max. | | |
| SSIBCK0 | Cycle | Master | $t_O$ | 80 | — | ns | Figure 47.44 |
| | | Slave | $t_I$ | 80 | — | ns | |
| | High level/ low level | Master | $t_{HC}/t_{LC}$ | 0.35 | — | $t_O$ | |
| | | Slave | | 0.35 | — | $t_I$ | |
| | Rising time/ falling time | Master | $t_{RC}/t_{FC}$ | — | 0.15 | $t_O$ / $t_I$ | |
| | | Slave | | — | 0.15 | $t_O$ / $t_I$ | |
| SSILRCK0/ SSIFS0, SSITXD0, SSIRXD0 | Input set up time | Master | $t_{SR}$ | 12 | — | ns | Figure 47.46, Figure 47.47 |
| | | Slave | | 12 | — | ns | |
| | Input hold time | Master | $t_{HR}$ | 8 | — | ns | |
| | | Slave | | 15 | — | ns | |
| | Output delay time | Master | $t_{DTR}$ | -10 | 5 | ns | |
| | | Slave | | 0 | 20 | ns | Figure 47.46, Figure 47.47 |
| | Output delay time from SSILRCK0/ SSIFS0 change | Slave | $t_{DTRW}$ | — | 20 | ns | Figure 47.48*1 |
| GTIOC2A, AUDIO_CLK | Cycle | | $t_{EXcyc}$ | 20 | — | ns | Figure 47.45 |
| | High level/ low level | | $t_{EXL}/t_{EXH}$ | 0.4 | 0.6 | $t_{EXcyc}$ | |

Note 1. For slave-mode transmission, SSIE has a path, through which the signal input from the SSILRCK0/SSIFS0 pin is used to generate transmit data, and the transmit data is logically output to the SSITXD0 pin.

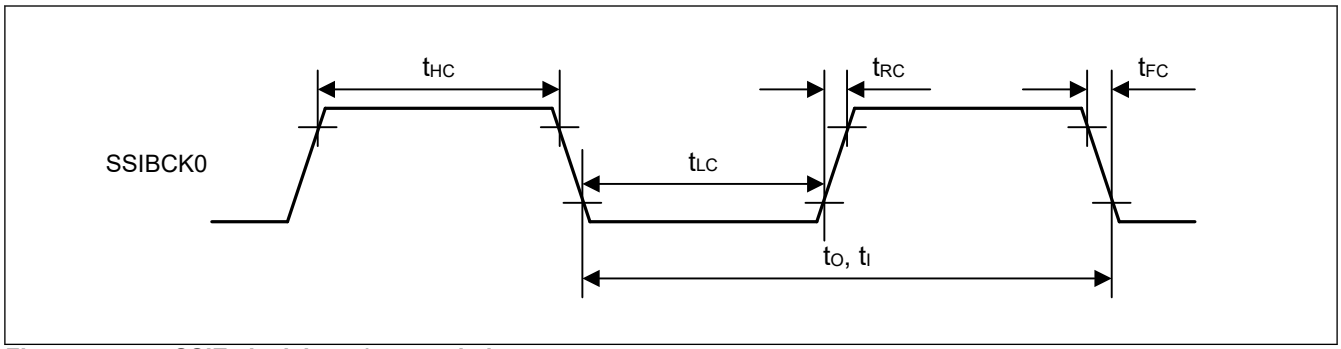



**Figure 47.44 SSIE clock input/output timing**

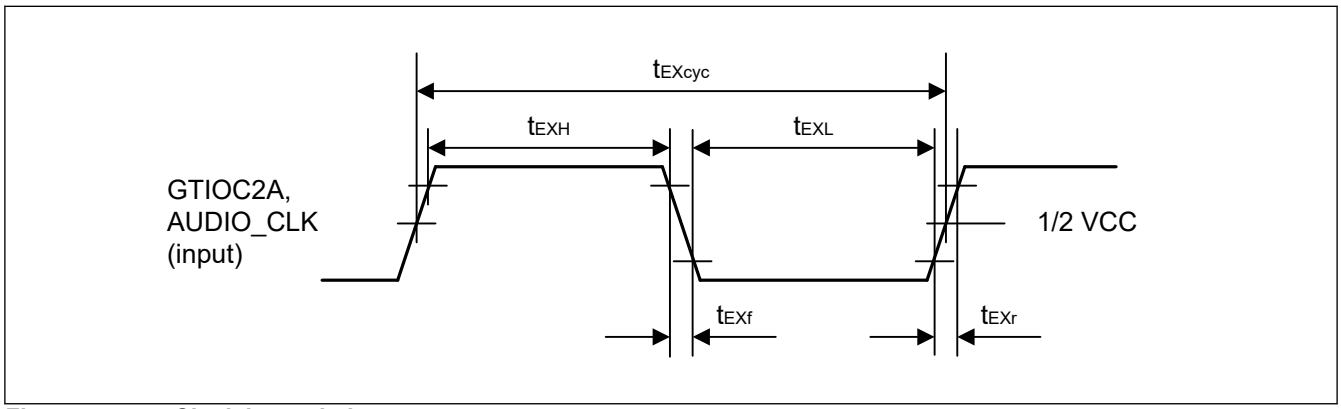



**Figure 47.45 Clock input timing**

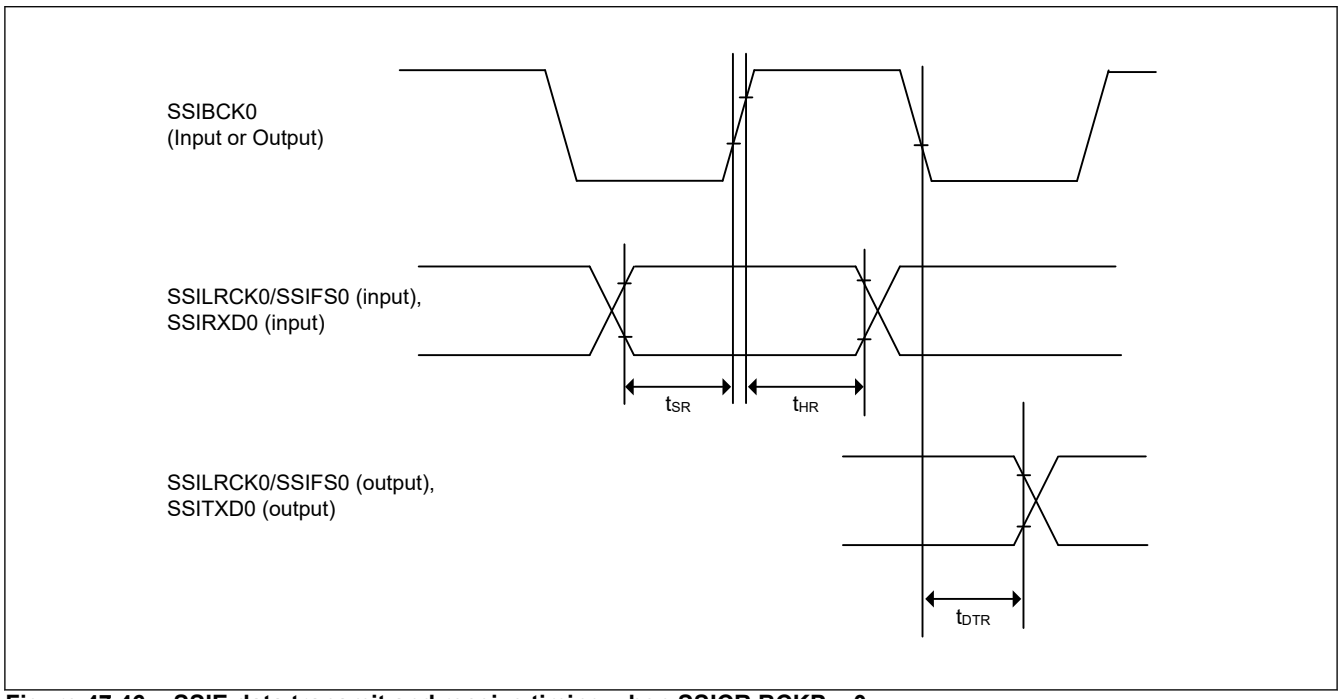



**Figure 47.46 SSIE data transmit and receive timing when SSICR.BCKP = 0**

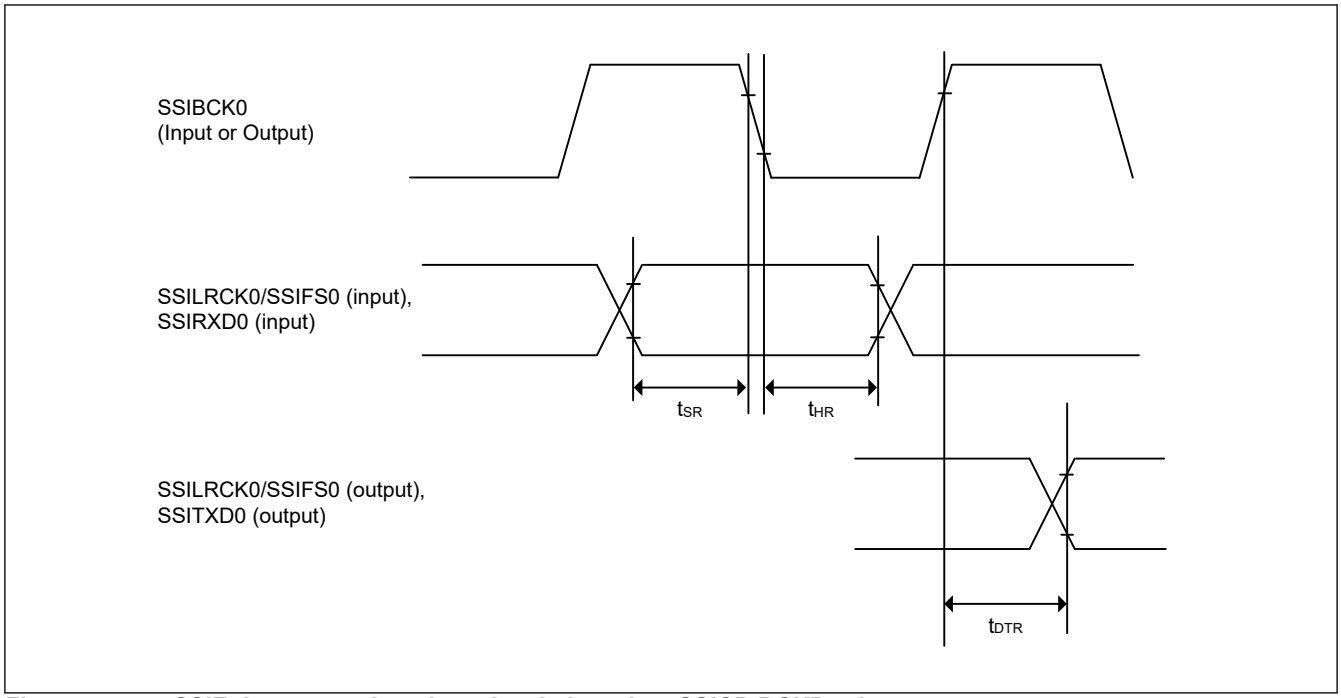



**Figure 47.47 SSIE data transmit and receive timing when SSICR.BCKP = 1**

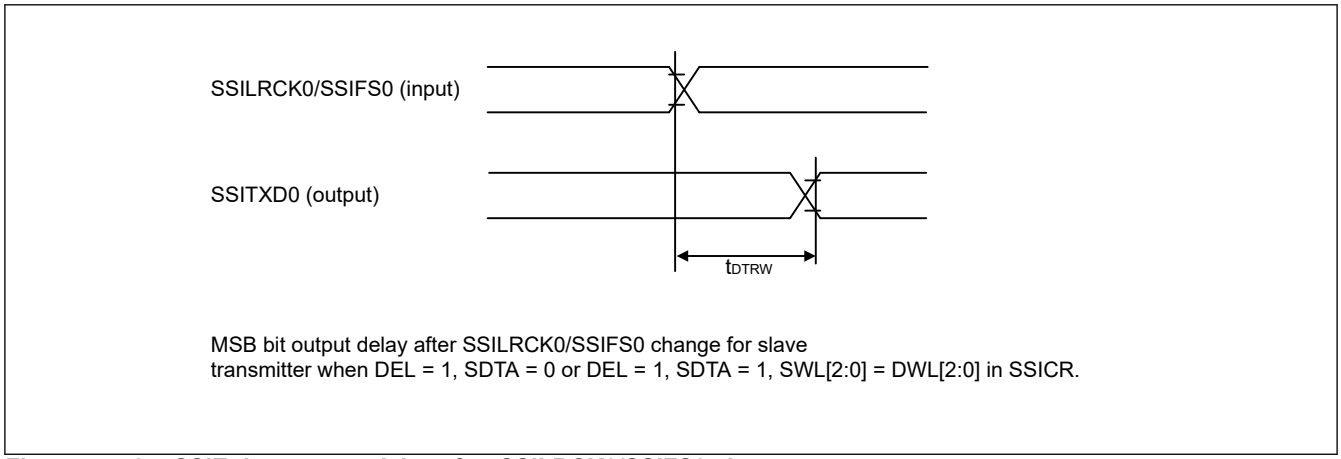



**Figure 47.48 SSIE data output delay after SSILRCK0/SSIFS0 change**

## 47.3.13 SD/MMC Host Interface Timing

**Table 47.32 SD/MMC Host Interface signal timing**

Conditions: High drive output is selected in the Port Drive Capability bit in the PmnPFS register.
Clock duty ratio is 50%.

| Parameter | Symbol | Min | Max | Unit | Test conditions |
|---|---|---|---|---|---|
| SDCLK clock cycle | $T_{SDCYC}$ | 20 | — | ns | Figure 47.49 |
| SDCLK clock high pulse width | $T_{SDWH}$ | 6.5 | — | ns | |
| SDCLK clock low pulse width | $T_{SDWL}$ | 6.5 | — | ns | |
| SDCLK clock rise time | $T_{SDLH}$ | — | 3 | ns | |
| SDCLK clock fall time | $T_{SDHL}$ | — | 3 | ns | |
| SDCMD/SDDAT output data delay | $T_{SDODLY}$ | –7 | 4 | ns | |
| SDCMD/SDDAT input data setup | $T_{SDIS}$ | 4.5 | — | ns | |
| SDCMD/SDDAT input data hold | $T_{SDIH}$ | 1.5 | — | ns | |

Note: Must use pins that have a letter appended to their name, for instance “_A”, “_B”, to indicate group membership. For the SD/MMC Host interface, the AC portion of the electrical characteristics is measured for each group.

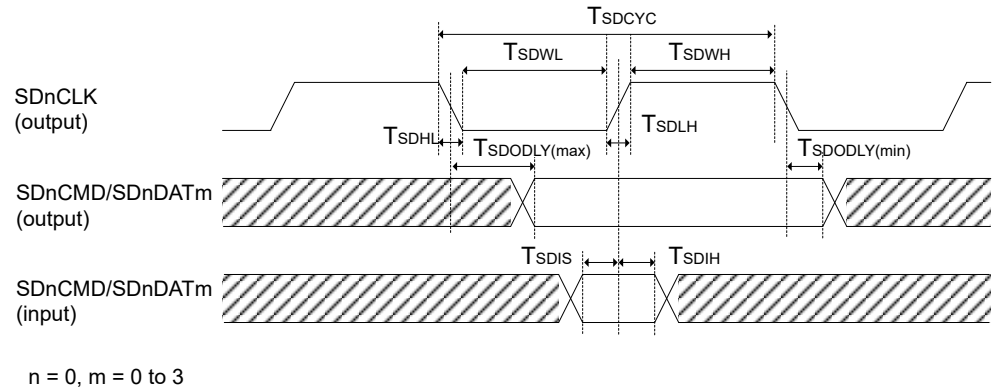



**Figure 47.49 SD/MMC Host Interface signal timing**

## 47.4 USB Characteristics

### 47.4.1 USBFS Timing

**Table 47.33 USBFS low-speed characteristics for host only (USB_DP and USB_DM pin characteristics)**

Conditions: VCC = AVCC0 = VCC_USB = VBATT = 3.0 to 3.6V, 2.7 ≤ VREFH0/VREFH ≤ AVCC0, USBCLK = 48 MHz

| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| Input characteristics | Input high voltage | $V_{IH}$ | 2.0 | — | — | V | — |
| | Input low voltage | $V_{IL}$ | — | — | 0.8 | V | — |
| | Differential input sensitivity | $V_{DI}$ | 0.2 | — | — | V | \| USB_DP - USB_DM \| |
| | Differential common-mode range | $V_{CM}$ | 0.8 | — | 2.5 | V | — |
| Output characteristics | Output high voltage | $V_{OH}$ | 2.8 | — | 3.6 | V | $I_{OH}$ = –200 µA |
| | Output low voltage | $V_{OL}$ | 0.0 | — | 0.3 | V | $I_{OL}$ = 2 mA |
| | Cross-over voltage | $V_{CRS}$ | 1.3 | — | 2.0 | V | Figure 47.50 |
| | Rise time | $t_{LR}$ | 75 | — | 300 | ns | |
| | Fall time | $t_{LF}$ | 75 | — | 300 | ns | |
| | Rise/fall time ratio | $t_{LR}$ / $t_{LF}$ | 80 | — | 125 | % | $t_{LR}$/ $t_{LF}$ |
| Pull-up and pull-down characteristics | USB_DP and USB_DM pull-down resistance in host controller mode | $R_{pd}$ | 14.25 | — | 24.80 | kΩ | — |

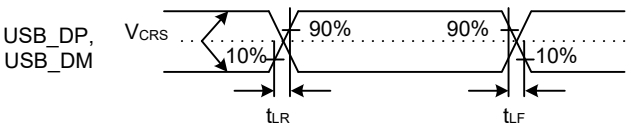



**Figure 47.50 USB_DP and USB_DM output timing in low-speed mode**

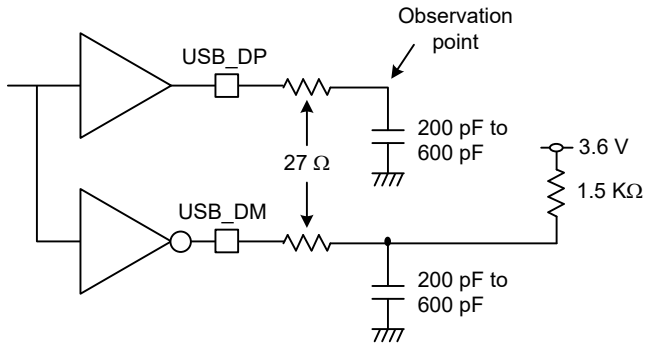



**Figure 47.51 Test circuit in low-speed mode**

**Table 47.34 USBFS full-speed characteristics (USB_DP and USB_DM pin characteristics)**

Conditions: VCC = AVCC0 = VCC_USB = VBATT = 3.0 to 3.6 V, 2.7 ≤ VREFH0/VREFH ≤ AVCC0, USBCLK = 48 MHz

| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| Input characteristics | Input high voltage | $V_{IH}$ | 2.0 | — | — | V | — |
| | Input low voltage | $V_{IL}$ | — | — | 0.8 | V | — |
| | Differential input sensitivity | $V_{DI}$ | 0.2 | — | — | V | \| USB_DP - USB_DM \| |
| | Differential common-mode range | $V_{CM}$ | 0.8 | — | 2.5 | V | — |
| Output characteristics | Output high voltage | $V_{OH}$ | 2.8 | — | 3.6 | V | $I_{OH}$ = –200 µA |
| | Output low voltage | $V_{OL}$ | 0.0 | — | 0.3 | V | $I_{OL}$ = 2 mA |
| | Cross-over voltage | $V_{CRS}$ | 1.3 | — | 2.0 | V | Figure 47.52 |
| | Rise time | $t_{LR}$ | 4 | — | 20 | ns | |
| | Fall time | $t_{LF}$ | 4 | — | 20 | ns | |
| | Rise/fall time ratio | $t_{LR}$ / $t_{LF}$ | 90 | — | 111.11 | % | $t_{FR}$/ $t_{FF}$ |
| | Output resistance | $Z_{DRV}$ | 28 | — | 44 | Ω | USBFS: Rs = 27 Ω included |
| Pull-up and pull-down characteristics | DM pull-up resistance in device controller mode | $R_{pu}$ | 0.900 | — | 1.575 | kΩ | During idle state |
| | | | 1.425 | — | 3.090 | kΩ | During transmission and reception |
| | USB_DP and USB_DM pull-down resistance in host controller mode | $R_{pd}$ | 14.25 | — | 24.80 | kΩ | — |

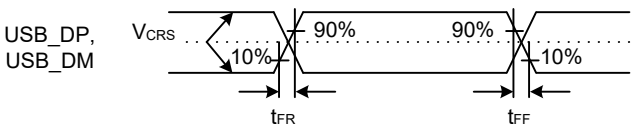



**Figure 47.52 USB_DP and USB_DM output timing in full-speed mode**

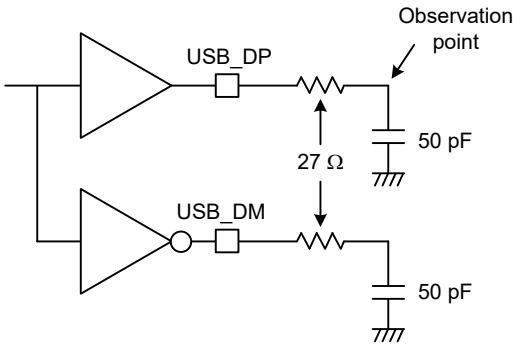



**Figure 47.53 Test circuit in full-speed mode**

**Table 47.35 USBFS characteristics (USB_DP and USB_DM pin characteristics)**

Conditions: VCC = AVCC0 = VCC_USB = VBATT = 3.0 to 3.6 V, 2.7 ≤ VREFH0/VREFH ≤ AVCC0, USBCLK = 48 MHz

| Parameter | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| Battery Charging Specification | D+ sink current | $I_{DP_SINK}$ | 25 | — | 175 | μA | — |
| | D- sink current | $I_{DM_SINK}$ | 25 | — | 175 | μA | — |
| | DCD source current | $I_{DP_SRC}$ | 7 | — | 13 | μA | — |
| | Data detection voltage | $V_{DAT_REF}$ | 0.25 | — | 0.4 | V | — |
| | D+ source voltage | $V_{DP_SRC}$ | 0.5 | — | 0.7 | V | Outout current = 250 μA |
| | D- source voltage | $V_{DM_SRC}$ | 0.5 | — | 0.7 | V | Outout current = 250 μA |

## 47.5 ADC12 Characteristics

**Table 47.36 A/D conversion characteristics for unit 0 (1 of 2)**

Conditions: PCLKC = 1 to 50 MHz

| Parameter | | | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| Frequency | | | 1 | — | 50 | MHz | — |
| Analog input capacitance | | | — | — | 30 | pF | — |
| Quantization error | | | — | ±0.5 | — | LSB | — |
| Resolution | | | — | — | 12 | Bits | — |
| High-precision high-speed channels (AN000 to AN002) | Conversion time[*1] (operation at PCLKC = 50 MHz) | Permissible signal source impedance Max. = 1 kΩ | 0.52 (0.26)[*2] | — | — | μs | Sampling in 13 states |
| | | Max. = 400 Ω | 0.40 (0.14)[*2] | — | — | μs | Sampling in 7 states<br>VCC = AVCC0 = 3.0 to 3.6 V<br>3.0 V ≤ VREFH0 ≤ AVCC0 |
| | Offset error | | — | ±1.0 | ±2.5 | LSB | — |
| | Full-scale error | | — | ±1.0 | ±2.5 | LSB | — |
| | Absolute accuracy | | — | ±2.0 | ±4.5 | LSB | — |
| | DNL differential nonlinearity error | | — | ±0.5 | ±1.5 | LSB | — |
| | INL integral nonlinearity error | | — | ±1.0 | ±2.5 | LSB | — |

**Table 47.36 A/D conversion characteristics for unit 0 (2 of 2)**

Conditions: PCLKC = 1 to 50 MHz

| Parameter | | | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|
| High-precision normal-speed channels (AN003 to AN008, AN011 to AN013) | Conversion time*1 (Operation at PCLKC = 50 MHz) | Permissible signal source impedance Max. = 1 kΩ | 0.92 (0.66)*2 | — | — | µs | Sampling in 33 states |
| | Offset error | | — | ±1.0 | ±2.5 | LSB | — |
| | Full-scale error | | — | ±1.0 | ±2.5 | LSB | — |
| | Absolute accuracy | | — | ±2.0 | ±4.5 | LSB | — |
| | DNL differential nonlinearity error | | — | ±0.5 | ±1.5 | LSB | — |
| | INL integral nonlinearity error | | — | ±1.0 | ±2.5 | LSB | — |
| Normal-precision normal-speed channels (AN016) | Conversion time*1 (Operation at PCLKC = 50 MHz) | Permissible signal source impedance Max. = 1 kΩ | 0.92 (0.66)*2 | — | — | µs | Sampling in 33 states |
| | Offset error | | — | ±1.0 | ±5.5 | LSB | — |
| | Full-scale error | | — | ±1.0 | ±5.5 | LSB | — |
| | Absolute accuracy | | — | ±2.0 | ±7.5 | LSB | — |
| | DNL differential nonlinearity error | | — | ±0.5 | ±4.5 | LSB | — |
| | INL integral nonlinearity error | | — | ±1.0 | ±5.5 | LSB | — |

Note: These specification values apply when there is no access to the external memory during A/D conversion. If access occurs during A/D conversion, values might not fall within the indicated ranges.
The use of PORT0 as digital outputs is not allowed when the 12-Bit A/D converter is used.
The characteristics apply when AVCC0, AVSS0, VREFH0, VREFL0, and 12-bit A/D converter input voltage are stable.

Note 1. The conversion time includes the sampling and comparison times. The number of sampling states is indicated for the test conditions.

Note 2. Values in parentheses indicate the sampling time.

**Table 47.37 A/D internal reference voltage characteristics**

| Parameter | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|
| A/D internal reference voltage | 1.13 | 1.18 | 1.23 | V | — |
| Sampling time | 4.15 | — | — | µs | — |

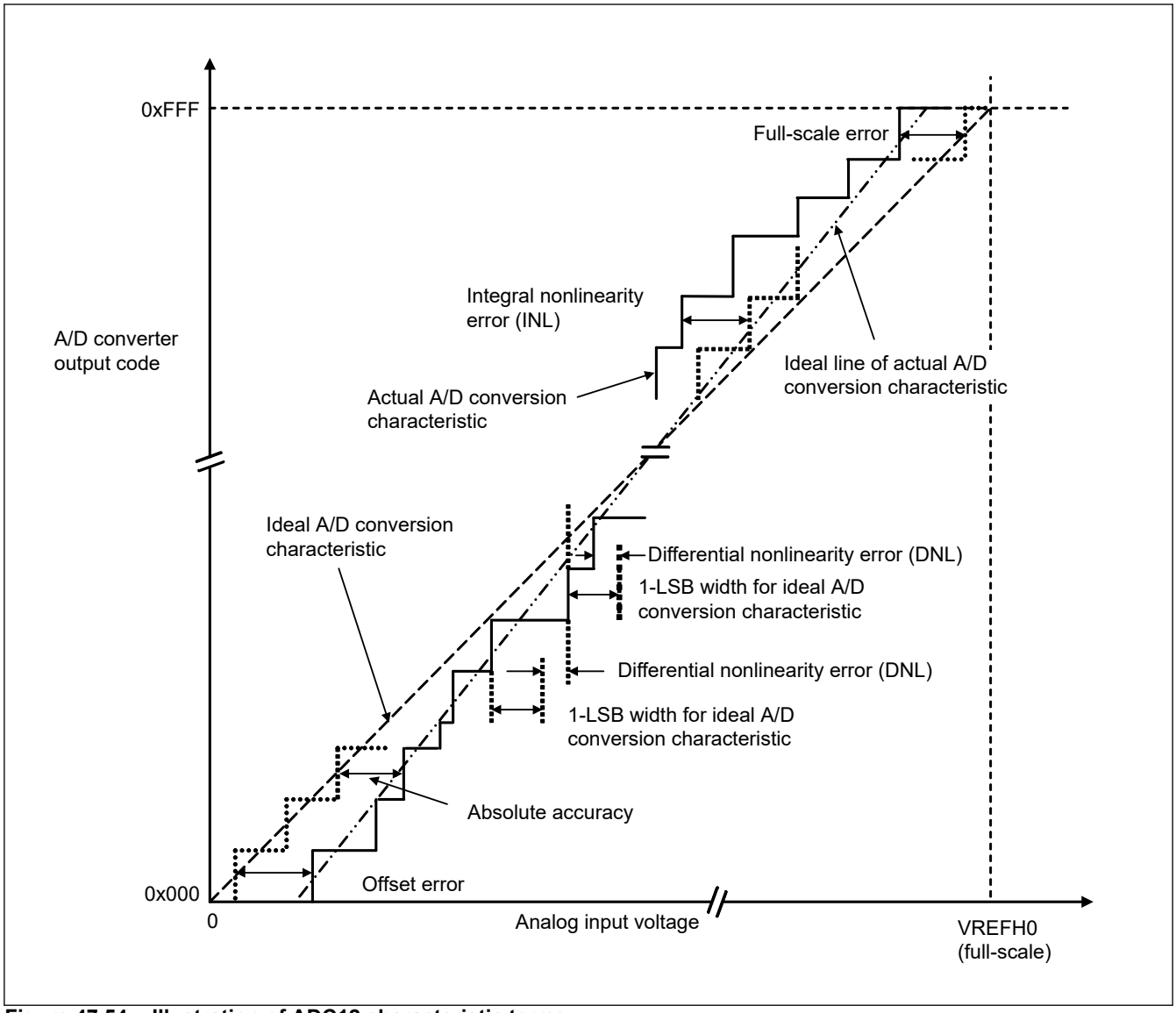



**Figure 47.54 Illustration of ADC12 characteristic terms**

### Absolute accuracy

Absolute accuracy is the difference between output code based on the theoretical A/D conversion characteristics, and the actual A/D conversion result. When measuring absolute accuracy, the voltage at the midpoint of the width of the analog input voltage (1-LSB width), which can meet the expectation of outputting an equal code based on the theoretical A/D conversion characteristics, is used as an analog input voltage. For example, if 12-bit resolution is used and the reference voltage VREFH0 = 3.072 V, then the 1-LSB width becomes 0.75 mV, and 0 mV, 0.75 mV, and 1.5 mV are used as the analog input voltages. If the analog input voltage is 6 mV, an absolute accuracy of ±5 LSB means that the actual A/D conversion result is in the range of 0x003 to 0x00D, though an output code of 0x008 can be expected from the theoretical A/D conversion characteristics.

### Integral nonlinearity error (INL)

Integral nonlinearity error is the maximum deviation between the ideal line when the measured offset and full-scale errors are zeroed, and the actual output code.

### Differential nonlinearity error (DNL)

Differential nonlinearity error is the difference between the 1-LSB width based on the ideal A/D conversion characteristics and the width of the actual output code.

### Offset error

Offset error is the difference between the transition point of the ideal first output code and the actual first output code.

### Full-scale error

Full-scale error is the difference between the transition point of the ideal last output code and the actual last output code.

## 47.6 DAC12 Characteristics

**Table 47.38 D/A conversion characteristics**

| Parameter | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|
| Resolution | — | — | 12 | Bits | — |
| Without output amplifier | | | | | |
| Absolute accuracy | — | — | ±24 | LSB | Resistive load 2 MΩ |
| INL | — | ±2.0 | ±8.0 | LSB | Resistive load 2 MΩ |
| DNL | — | ±1.0 | ±2.0 | LSB | — |
| Output impedance | — | 8.5 | — | kΩ | — |
| Conversion time | — | — | 3 | µs | Resistive load 2 MΩ, Capacitive load 20 pF |
| Output voltage range | 0 | — | VREFH | V | — |
| With output amplifier | | | | | |
| INL | — | ±2.0 | ±4.0 | LSB | — |
| DNL | — | ±1.0 | ±2.0 | LSB | — |
| Conversion time | — | — | 4.0 | µs | — |
| Resistive load | 5 | — | — | kΩ | — |
| Capacitive load | — | — | 50 | pF | — |
| Output voltage range | 0.2 | — | VREFH – 0.2 | V | — |

## 47.7 TSN Characteristics

**Table 47.39 TSN characteristics**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| Relative accuracy | — | — | ± 1.0 | — | °C | — |
| Temperature slope | — | — | 4.0 | — | mV/°C | — |
| Output voltage (at 25 °C) | — | — | 1.24 | — | V | — |
| Temperature sensor start time | $t_{START}$ | — | — | 30 | µs | — |
| Sampling time | — | 4.15 | — | — | µs | — |

## 47.8 OSC Stop Detect Characteristics

**Table 47.40 Oscillation stop detection circuit characteristics**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| Detection time | $t_{dr}$ | — | — | 1 | ms | Figure 47.55 |

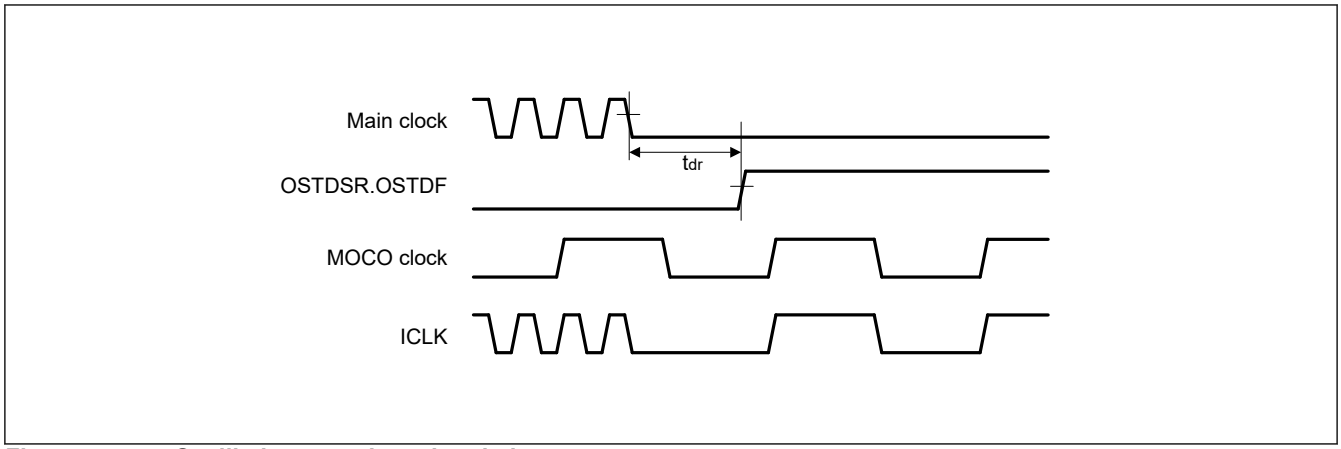



**Figure 47.55 Oscillation stop detection timing**

## 47.9 POR and LVD Characteristics

**Table 47.41 Power-on reset circuit and voltage detection circuit characteristics (1)**

| Parameter | | | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|
| Voltage detection level | Power-on reset (POR) | DPSBYCR.DEEPCUT[1:0] = 00b or 01b. | $V_{POR}$ | 2.5 | 2.6 | 2.7 | V | Figure 47.56 |
| | | DPSBYCR.DEEPCUT[1:0] = 11b. | | 1.8 | 2.25 | 2.7 | | |
| | Voltage detection circuit (LVD0) | | $V_{det0_1}$ | 2.84 | 2.94 | 3.04 | | Figure 47.57 |
| | | | $V_{det0_2}$ | 2.77 | 2.87 | 2.97 | | |
| | | | $V_{det0_3}$ | 2.70 | 2.80 | 2.90 | | |
| | Voltage detection circuit (LVD1) | | $V_{det1_1}$ | 2.89 | 2.99 | 3.09 | | Figure 47.58 |
| | | | $V_{det1_2}$ | 2.82 | 2.92 | 3.02 | | |
| | | | $V_{det1_3}$ | 2.75 | 2.85 | 2.95 | | |
| | Voltage detection circuit (LVD2) | | $V_{det2_1}$ | 2.89 | 2.99 | 3.09 | | Figure 47.59 |
| | | | $V_{det2_2}$ | 2.82 | 2.92 | 3.02 | | |
| | | | $V_{det2_3}$ | 2.75 | 2.85 | 2.95 | | |
| Internal reset time | Power-on reset time | | $t_{POR}$ | — | 4.5 | — | ms | Figure 47.56 |
| | LVD0 reset time | | $t_{LVD0}$ | — | 0.51 | — | | Figure 47.57 |
| | LVD1 reset time | | $t_{LVD1}$ | — | 0.38 | — | | Figure 47.58 |
| | LVD2 reset time | | $t_{LVD2}$ | — | 0.38 | — | | Figure 47.59 |
| Minimum VCC down time*1 | | | $t_{VOFF}$ | 200 | — | — | μs | Figure 47.56, Figure 47.57 |
| Response delay | | | $t_{det}$ | — | — | 200 | μs | Figure 47.57 to Figure 47.59 |
| LVD operation stabilization time (after LVD is enabled) | | | $t_{d(E-A)}$ | — | — | 10 | μs | Figure 47.58, Figure 47.59 |
| Hysteresis width (LVD1 and LVD2) | | | $V_{LVH}$ | — | 70 | — | mV | |

Note 1. The minimum VCC down time indicates the time when VCC is below the minimum value of voltage detection levels $V_{POR}$, $V_{det0}$, $V_{det1}$, and $V_{det2}$ for POR and LVD.

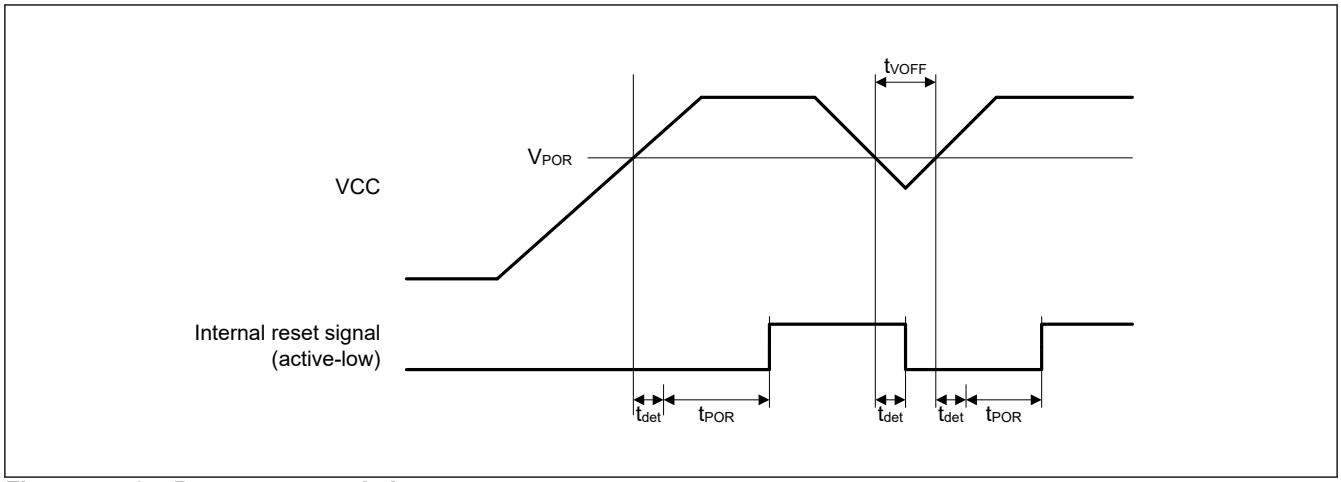



**Figure 47.56 Power-on reset timing**

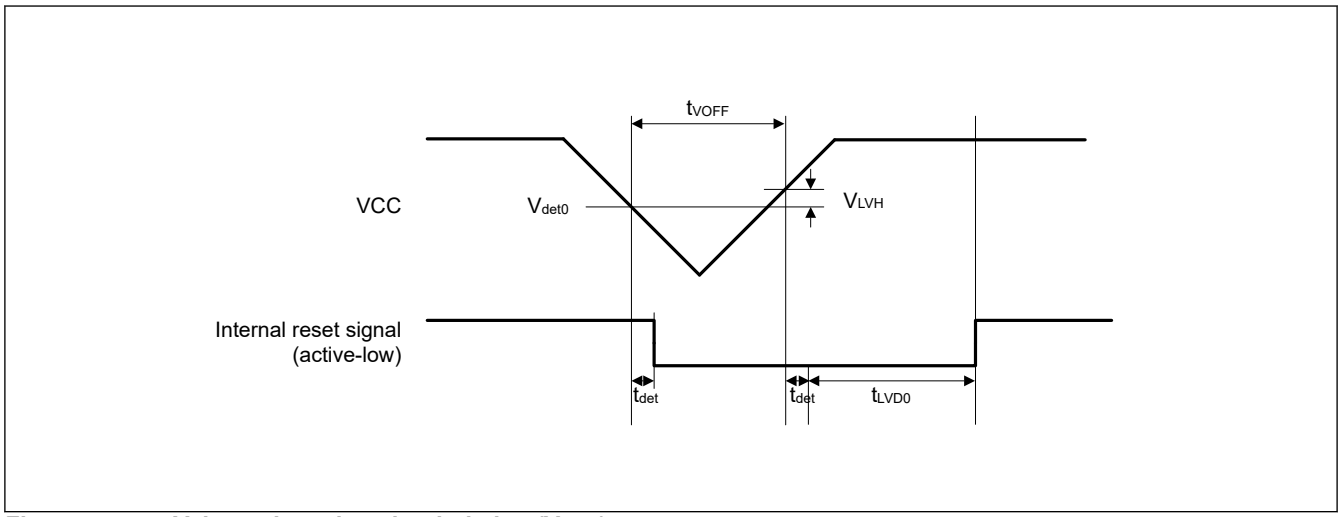



**Figure 47.57 Voltage detection circuit timing ($V_{det0}$)**

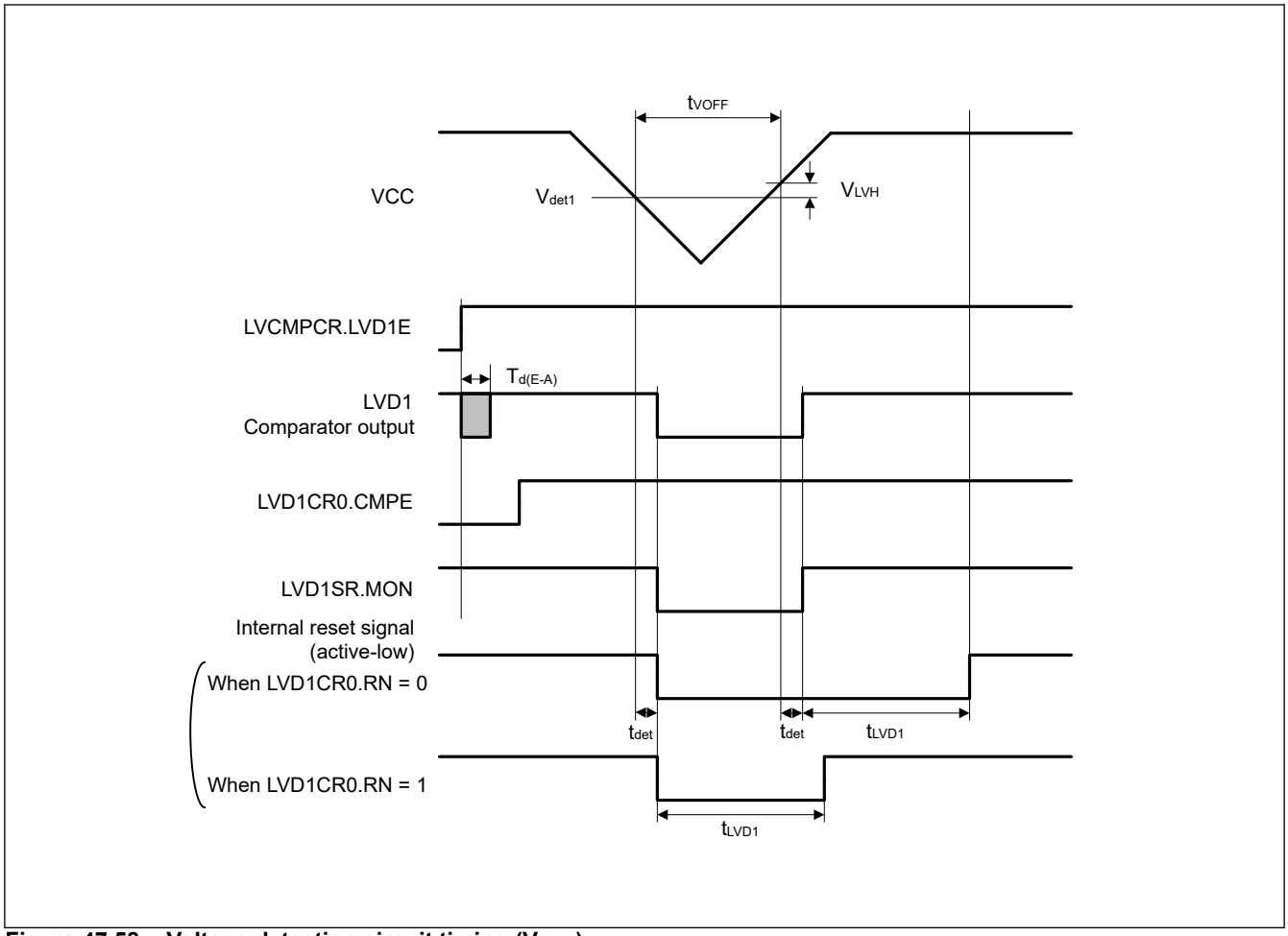



**Figure 47.58 Voltage detection circuit timing ($V_{det1}$)**

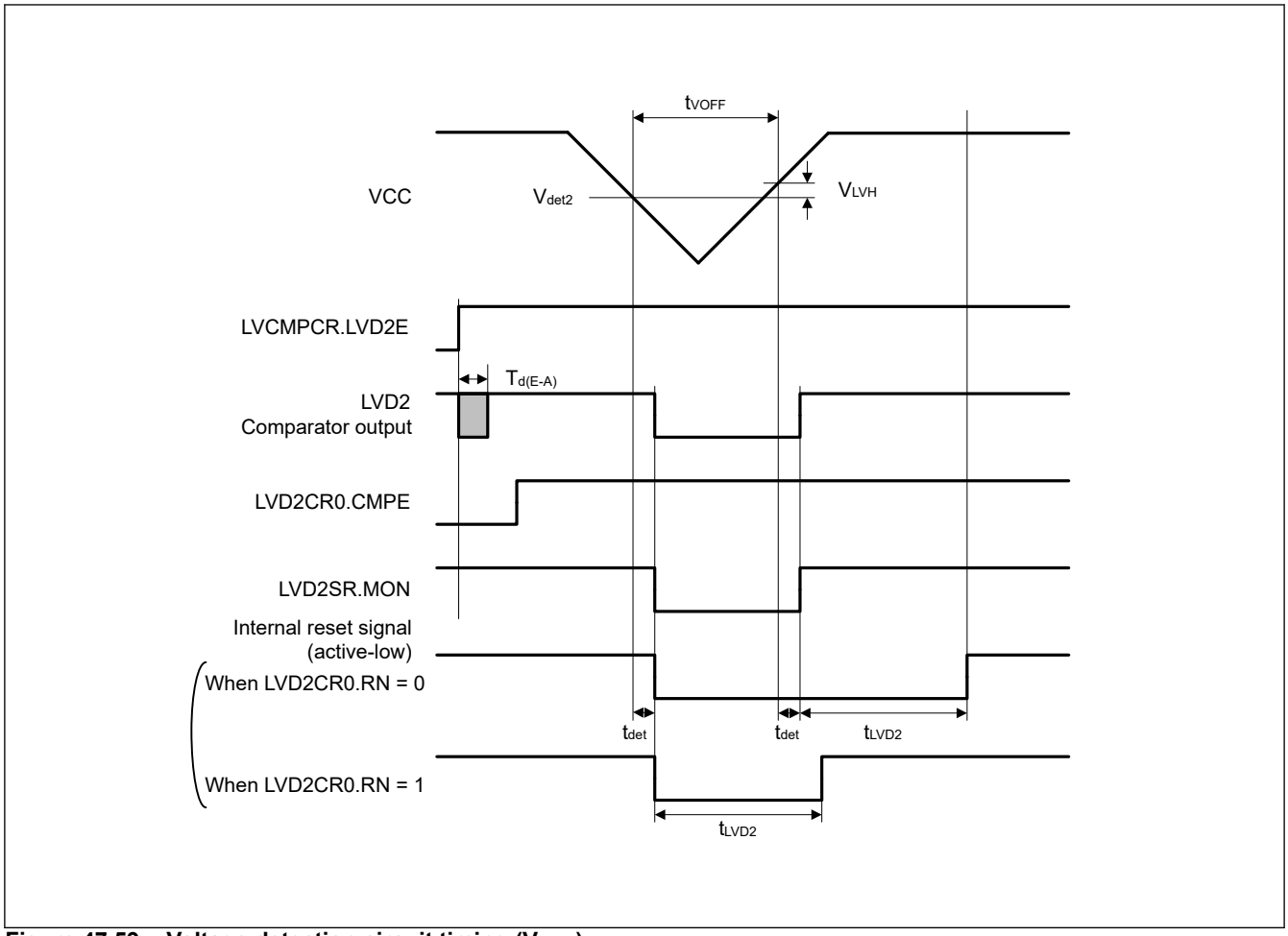



**Figure 47.59 Voltage detection circuit timing ($V_{det2}$)**

## 47.10 VBATT Characteristics

**Table 47.42 Battery backup function characteristics**

Conditions: VCC = AVCC0 = VCC_USB = 2.7 to 3.6 V, 2.7 ≤ VREFH0/VREFH ≤ AVCC0, VBATT = 1.8 to 3.6 V

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| Voltage level for switching to battery backup | $V_{DETBATT}$ | 2.50 | 2.60 | 2.70 | V | Figure 47.60 |
| Lower-limit VBATT voltage for power supply switching caused by VCC voltage drop | $V_{BATTSW}$ | 2.70 | — | — | V | |
| VCC-off period for starting power supply switching | $t_{VOFFBATT}$ | 200 | — | — | µs | |
| VBATT low voltage detection level | $V_{battldet}$ | 1.8 | 1.9 | 2.0 | V | Figure 47.61 |
| Minimum VBATT down time | $t_{BATTOFF}$ | 200 | — | — | µs | |
| Response delay | $t_{BATTdet}$ | — | — | 200 | µs | |
| VBATT monitor operation stabilization time (after VBATTMNSELR.VBATTMNSEL is changed to 1) | $t_{d(E-A)}$ | — | — | 20 | µs | |
| VBATT current increase (when VBATTMNSELR.VBATTMNSEL is 1 compared to the case that VBATTMNSELR.VBATTMNSEL is 0) | $I_{VBATTSEL}$ | — | 140 | 350 | nA | |

Note: The VCC-off period for starting power supply switching indicates the period in which VCC is below the minimum value of the voltage level for switching to battery backup ($V_{DETBATT}$).

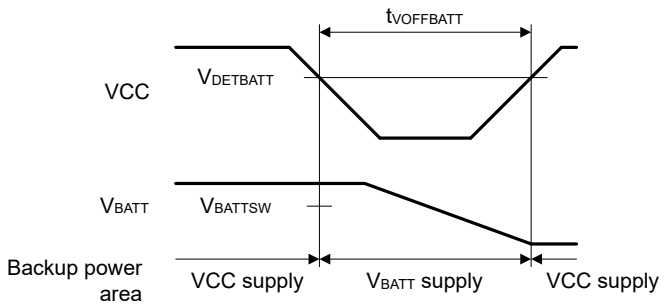



**Figure 47.60 Battery backup function characteristics**

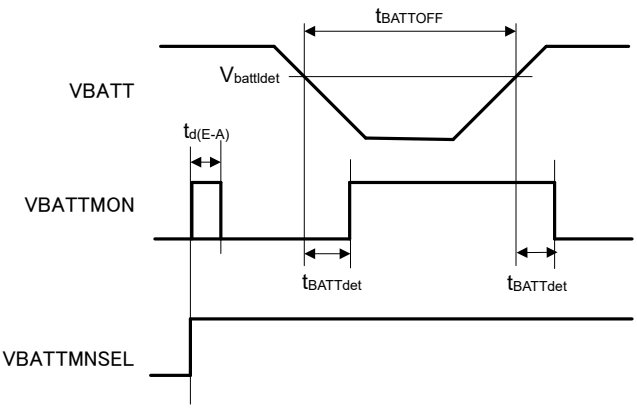



**Figure 47.61 Battery backup function characteristics**

## 47.11 CTSU Characteristics

**Table 47.43 CTSU characteristics**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| External capacitance connected to TSCAP pin | $C_{tscap}$ | 9 | 10 | 11 | nF | — |
| TS pin capacitive load | $C_{base}$ | — | — | 50 | pF | — |
| Permissible output high current | $\Sigma I_{oH}$ | — | — | -40 | mA | When the mutual capacitance method is applied |

## 47.12 Flash Memory Characteristics

### 47.12.1 Code Flash Memory Characteristics

**Table 47.44 Code flash memory characteristics (1 of 2)**

Conditions: Program or erase: FCLK = 4 to 50 MHz
Read: FCLK ≤ 50 MHz

| Parameter | | Symbol | FCLK = 4 MHz | | | 20 MHz ≤ FCLK ≤ 50 MHz | | | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min | Typ*6 | Max | Min | Typ*6 | Max | | |
| Programming time $N_{PEC}$ ≤ 100 times | 128-byte | $t_{P128}$ | — | 0.75 | 13.2 | — | 0.34 | 6.0 | ms | |
| | 8-KB | $t_{P8K}$ | — | 49 | 176 | — | 22 | 80 | ms | |
| | 32-KB | $t_{P32K}$ | — | 194 | 704 | — | 88 | 320 | ms | |

**Table 47.44 Code flash memory characteristics (2 of 2)**

Conditions: Program or erase: FCLK = 4 to 50 MHz
Read: FCLK ≤ 50 MHz

| Parameter | | Symbol | FCLK = 4 MHz | | | 20 MHz ≤ FCLK ≤ 50 MHz | | | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min | Typ*6 | Max | Min | Typ*6 | Max | | |
| Programming time $N_{PEC}$ > 100 times | 128-byte | $t_{P128}$ | — | 0.91 | 15.8 | — | 0.41 | 7.2 | ms | |
| | 8-KB | $t_{P8K}$ | — | 60 | 212 | — | 27 | 96 | ms | |
| | 32-KB | $t_{P32K}$ | — | 234 | 848 | — | 106 | 384 | ms | |
| Erasure time $N_{PEC}$ ≤ 100 times | 8-KB | $t_{E8K}$ | — | 78 | 216 | — | 43 | 120 | ms | |
| | 32-KB | $t_{E32K}$ | — | 283 | 864 | — | 157 | 480 | ms | |
| Erasure time $N_{PEC}$ > 100 times | 8-KB | $t_{E8K}$ | — | 94 | 260 | — | 52 | 144 | ms | |
| | 32-KB | $t_{E32K}$ | — | 341 | 1040 | — | 189 | 576 | ms | |
| Reprogramming/erasure cycle*4 | | $N_{PEC}$ | 10000*1 | — | — | 10000*1 | — | — | Times | |
| Suspend delay during programming | | $t_{SPD}$ | — | — | 264 | — | — | 120 | µs | |
| Programming resume time | | $t_{PRT}$ | — | — | 110 | — | — | 50 | µs | |
| First suspend delay during erasure in suspend priority mode | | $t_{SESD1}$ | — | — | 216 | — | — | 120 | µs | |
| Second suspend delay during erasure in suspend priority mode | | $t_{SESD2}$ | — | — | 1.7 | — | — | 1.7 | ms | |
| Suspend delay during erasure in erasure priority mode | | $t_{SEED}$ | — | — | 1.7 | — | — | 1.7 | ms | |
| First erasing resume time during erasure in suspend priority mode*5 | | $t_{REST1}$ | — | — | 1.7 | — | — | 1.7 | ms | |
| Second erasing resume time during erasure in suspend priority mode | | $t_{REST2}$ | — | — | 144 | — | — | 80 | µs | |
| Erasing resume time during erasure in erasure priority mode | | $t_{REET}$ | — | — | 144 | — | — | 80 | µs | |
| Forced stop command | | $t_{FD}$ | — | — | 32 | — | — | 20 | µs | |
| Data hold time*2 | | $t_{DRP}$ | 10*2 *3 | — | — | 10*2 *3 | — | — | Years | |

Note 1. This is the minimum number of times to guarantee all the characteristics after reprogramming. The guaranteed range is from 1 to the minimum value.
Note 2. This indicates the minimum value of the characteristic when reprogramming is performed within the specified range.
Note 3. This result is obtained from reliability testing.
Note 4. The reprogram/erase cycle is the number of erasures for each block. When the reprogram/erase cycle is n times (n = 10,000), erasing can be performed n times for each block. For example, when 128-byte programming is performed 64 times for different addresses in 8-KB blocks, and then the entire block is erased, the reprogram/erase cycle is counted as one. However, programming the same address several times as one erasure is not enabled. Overwriting is prohibited.
Note 5. Time for resumption includes time for reapplying the erasing pulse (up to one full pulse) that was cut off at the time of suspension.
Note 6. The reference value at VCC = 3.3V and room temperature.

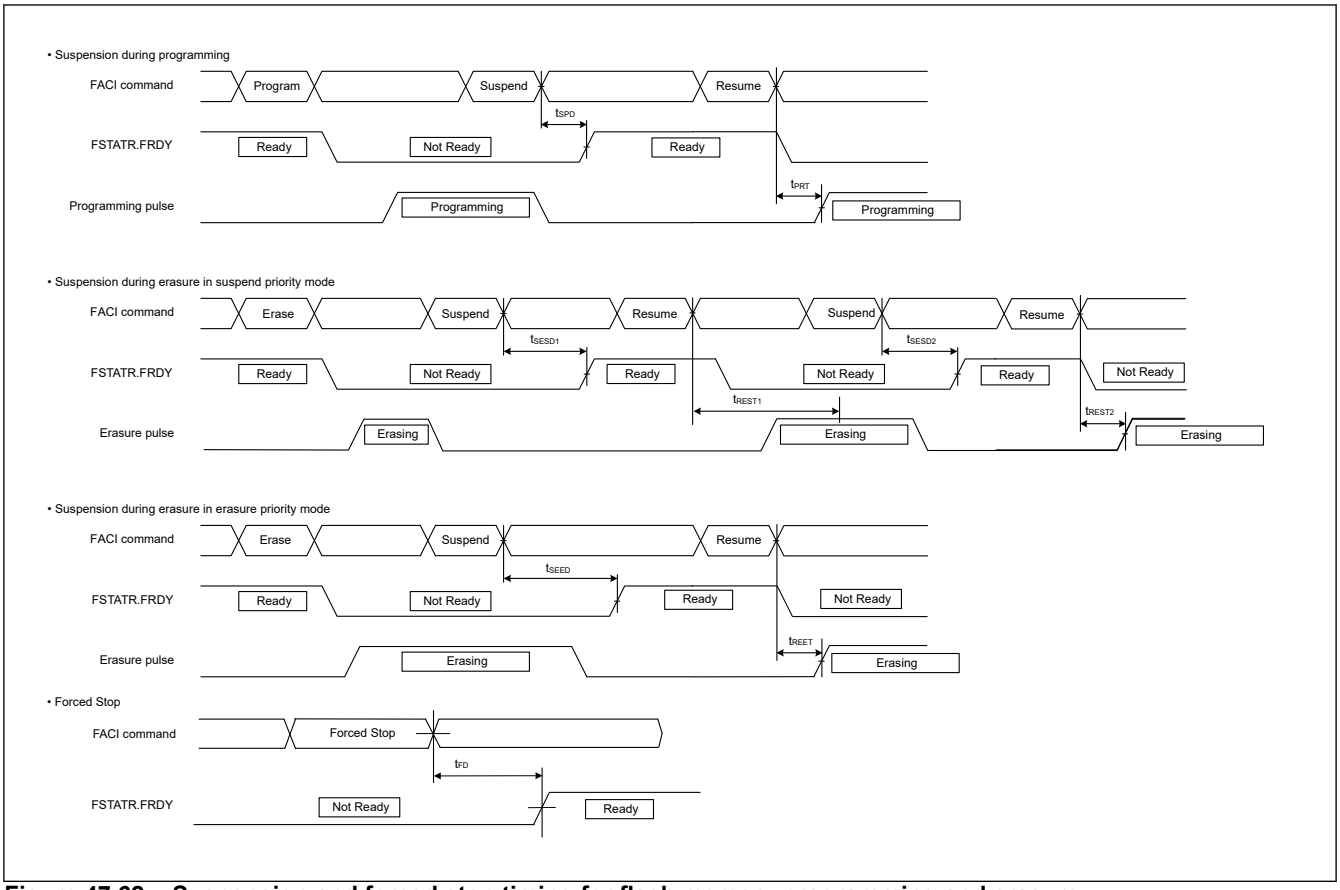



**Figure 47.62 Suspension and forced stop timing for flash memory programming and erasure**

## 47.12.2 Data Flash Memory Characteristics

**Table 47.45 Data flash memory characteristics (1 of 2)**

Conditions: Program or erase: FCLK = 4 to 50 MHz
Read: FCLK ≤ 50 MHz

| Parameter | | Symbol | FCLK = 4 MHz | | | 20 MHz ≤ FCLK ≤ 50 MHz | | | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min | Typ*6 | Max | Min | Typ*6 | Max | | |
| Programming time | 4-byte | $t_{DP4}$ | — | 0.36 | 3.8 | — | 0.16 | 1.7 | ms | |
| | 8-byte | $t_{DP8}$ | — | 0.38 | 4.0 | — | 0.17 | 1.8 | | |
| | 16-byte | $t_{DP16}$ | — | 0.42 | 4.5 | — | 0.19 | 2.0 | | |
| Erasure time | 64-byte | $t_{DE64}$ | — | 3.1 | 18 | — | 1.7 | 10 | ms | |
| | 128-byte | $t_{DE128}$ | — | 4.7 | 27 | — | 2.6 | 15 | | |
| | 256-byte | $t_{DE256}$ | — | 8.9 | 50 | — | 4.9 | 28 | | |
| Blank check time | 4-byte | $t_{DBC4}$ | — | — | 84 | — | — | 30 | µs | |
| Reprogramming/erasure cycle*1 | | $N_{DPEC}$ | 125000*2 | — | — | 125000*2 | — | — | — | |
| Suspend delay during programming | 4-byte | $t_{DSPD}$ | — | — | 264 | — | — | 120 | µs | |
| | 8-byte | | — | — | 264 | — | — | 120 | | |
| | 16-byte | | — | — | 264 | — | — | 120 | | |
| Programming resume time | | $t_{DPRT}$ | — | — | 110 | — | — | 50 | µs | |
| First suspend delay during erasure in suspend priority mode | 64-byte | $t_{DSESD1}$ | — | — | 216 | — | — | 120 | µs | |
| | 128-byte | | — | — | 216 | — | — | 120 | | |
| | 256-byte | | — | — | 216 | — | — | 120 | | |

**Table 47.45 Data flash memory characteristics (2 of 2)**

Conditions: Program or erase: FCLK = 4 to 50 MHz
Read: FCLK ≤ 50 MHz

| Parameter | | Symbol | FCLK = 4 MHz | | | 20 MHz ≤ FCLK ≤ 50 MHz | | | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min | Typ*6 | Max | Min | Typ*6 | Max | | |
| Second suspend delay during erasure in suspend priority mode | 64-byte | $t_{DSESD2}$ | — | — | 300 | — | — | 300 | µs | |
| | 128-byte | | — | — | 390 | — | — | 390 | | |
| | 256-byte | | — | — | 570 | — | — | 570 | | |
| Suspend delay during erasing in erasure priority mode | 64-byte | $t_{DSEED}$ | — | — | 300 | — | — | 300 | µs | |
| | 128-byte | | — | — | 390 | — | — | 390 | | |
| | 256-byte | | — | — | 570 | — | — | 570 | | |
| First erasing resume time during erasure in suspend priority mode*5 | | $t_{DREST1}$ | — | — | 300 | — | — | 300 | µs | |
| Second erasing resume time during erasure in suspend priority mode | | $t_{DREST2}$ | — | — | 126 | — | — | 70 | µs | |
| Erasing resume time during erasure in erasure priority mode | | $t_{DREET}$ | — | — | 126 | — | — | 70 | µs | |
| Forced stop command | | $t_{FD}$ | — | — | 32 | — | — | 20 | µs | |
| Data hold time*3 | | $t_{DRP}$ | 10*3 *4 | — | — | 10*3 *4 | — | — | Year | |

Note 1. The reprogram/erase cycle is the number of erasures for each block. When the reprogram/erase cycle is n times (n = 125,000), erasing can be performed n times for each block. For example, when 4-byte programming is performed 16 times for different addresses in 64-byte blocks, and then the entire block is erased, the reprogram/erase cycle is counted as one. However, programming the same address several times as one erasure is not enabled. Overwriting is prohibited.
Note 2. This is the minimum number of times to guarantee all the characteristics after reprogramming. The guaranteed range is from 1 to the minimum value.
Note 3. This indicates the minimum value of the characteristic when reprogramming is performed within the specified range.
Note 4. This result is obtained from reliability testing.
Note 5. Time for resumption includes time for reapplying the erasing pulse (up to one full pulse) that was cut off at the time of suspension.
Note 6. The reference value at VCC = 3.3 V and room temperature.

## 47.12.3 Option Setting Memory Characteristics

**Table 47.46 Option setting memory characteristics**

Conditions: Program: FCLK = 4 to 50 MHz
Read: FCLK ≤ 50 MHz

| Parameter | Symbol | FCLK = 4 MHz | | | 20 MHz ≤ FCLK ≤ 50 MHz | | | Unit | Test conditions |
|---|---|---|---|---|---|---|---|---|---|
| | | Min | Typ*4 | Max | Min | Typ*4 | Max | | |
| Programming time $N_{OPC}$ ≤ 100 times | $t_{OP}$ | — | 83 | 309 | — | 45 | 162 | ms | |
| Programming time $N_{OPC}$ > 100 times | $t_{OP}$ | — | 100 | 371 | — | 55 | 195 | ms | |
| Reprogramming cycle | $N_{OPC}$ | 20000*1 | — | — | 20000*1 | — | — | Times | |
| Data hold time*2 | $t_{DRP}$ | 10*2 *3 | — | — | 10*2 *3 | — | — | Years | |

Note 1. This is the minimum number of times to guarantee all the characteristics after reprogramming. The guaranteed range is from 1 to the minimum value.
Note 2. This indicates the minimum value of the characteristic when reprogramming is performed within the specified range.
Note 3. This result is obtained from reliability testing.
Note 4. The reference value at VCC = 3.3 V and room temperature.

## 47.13 Boundary Scan

**Table 47.47 Boundary scan characteristics**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| TCK clock cycle time | $t_{TCKcyc}$ | 100 | — | — | ns | Figure 47.63 |
| TCK clock high pulse width | $t_{TCKH}$ | 45 | — | — | ns | |
| TCK clock low pulse width | $t_{TCKL}$ | 45 | — | — | ns | |
| TCK clock rise time | $t_{TCKr}$ | — | — | 5 | ns | |
| TCK clock fall time | $t_{TCKf}$ | — | — | 5 | ns | |
| TMS setup time | $t_{TMSS}$ | 20 | — | — | ns | Figure 47.64 |
| TMS hold time | $t_{TMSH}$ | 20 | — | — | ns | |
| TDI setup time | $t_{TDIS}$ | 20 | — | — | ns | |
| TDI hold time | $t_{TDIH}$ | 20 | — | — | ns | |
| TDO data delay | $t_{TDOD}$ | — | — | 40 | ns | |
| Boundary scan circuit startup time*1 | $T_{BSSTUP}$ | $t_{RESWP}$ | — | — | — | Figure 47.65 |

Note 1. Boundary scan does not function until the power-on reset becomes negative.

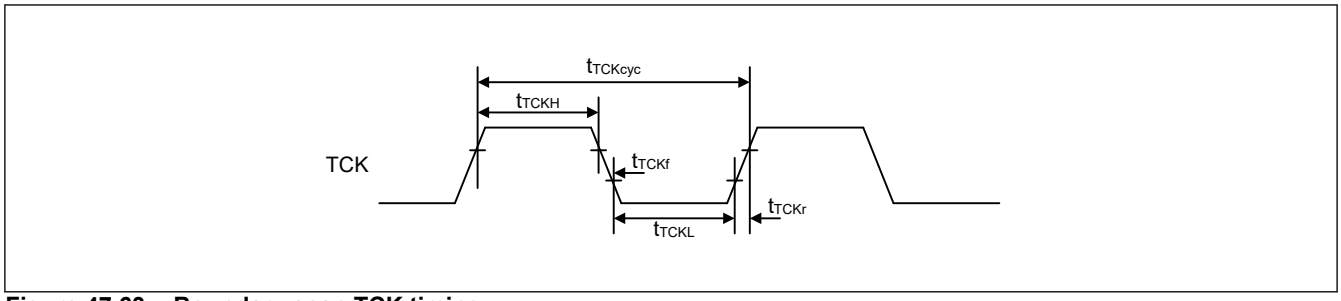



**Figure 47.63 Boundary scan TCK timing**

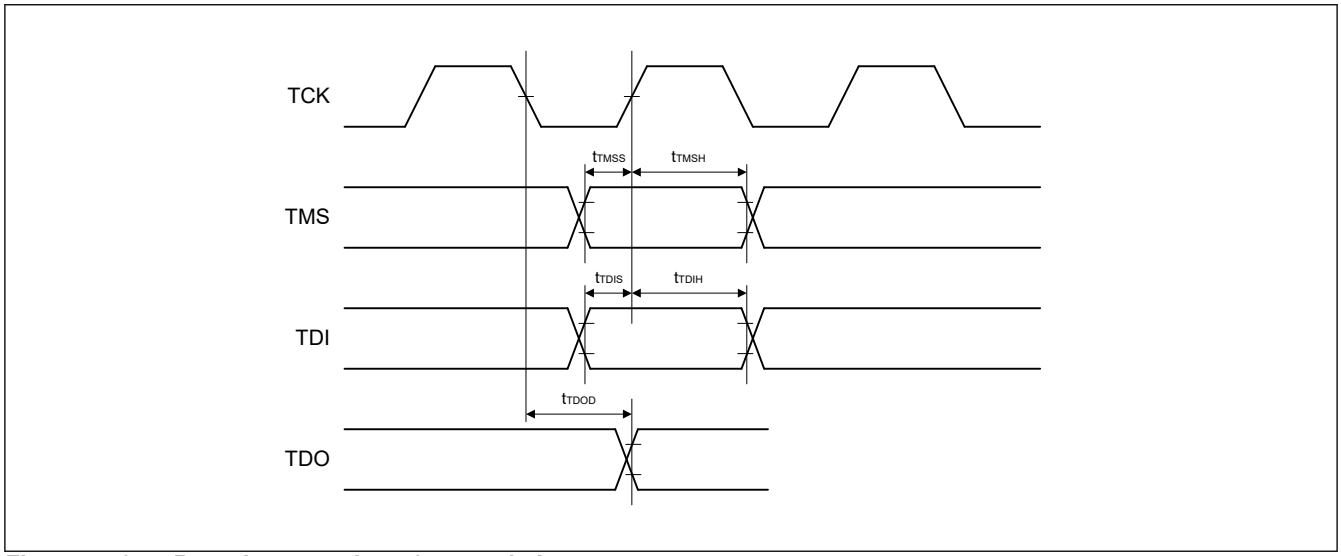



**Figure 47.64 Boundary scan input/output timing**

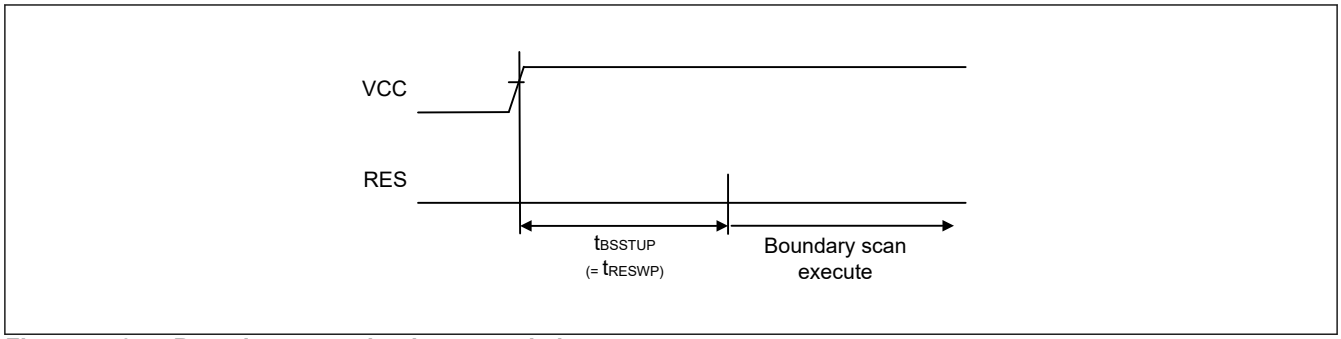



**Figure 47.65 Boundary scan circuit startup timing**

## 47.14 Joint Test Action Group (JTAG)

**Table 47.48 JTAG**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| TCK clock cycle time | $t_{TCKcyc}$ | 40 | — | — | ns | Figure 47.66 |
| TCK clock high pulse width | $t_{TCKH}$ | 15 | — | — | ns | |
| TCK clock low pulse width | $t_{TCKL}$ | 15 | — | — | ns | |
| TCK clock rise time | $t_{TCKr}$ | — | — | 5 | ns | |
| TCK clock fall time | $t_{TCKf}$ | — | — | 5 | ns | |
| TMS setup time | $t_{TMSS}$ | 8 | — | — | ns | Figure 47.67 |
| TMS hold time | $t_{TMSH}$ | 8 | — | — | ns | |
| TDI setup time | $t_{TDIS}$ | 8 | — | — | ns | |
| TDI hold time | $t_{TDIH}$ | 8 | — | — | ns | |
| TDO data delay time | $t_{TDOD}$ | — | — | 20 | ns | |

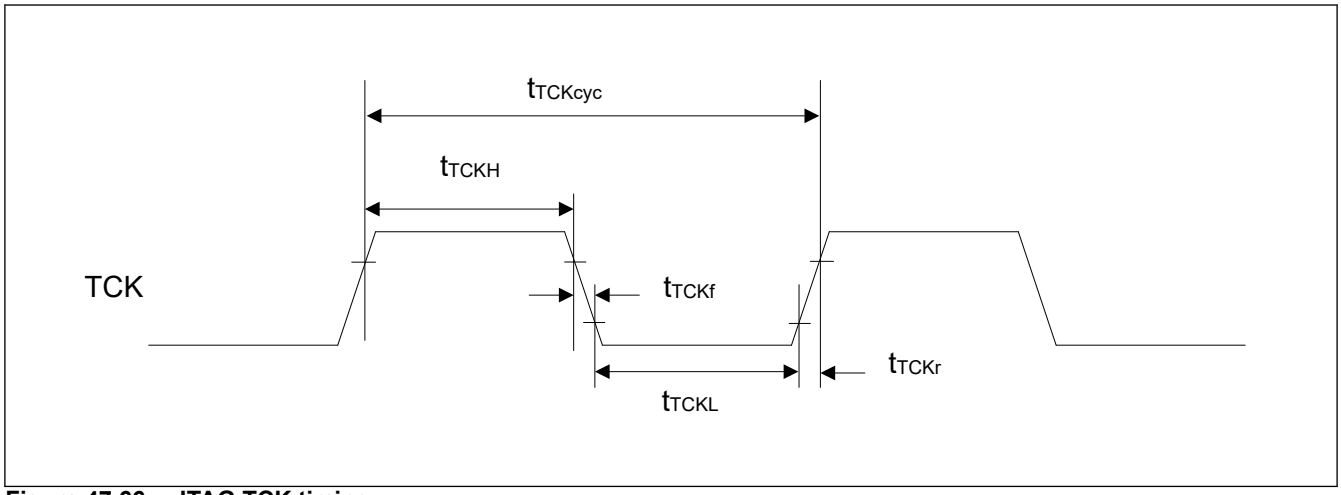



**Figure 47.66 JTAG TCK timing**

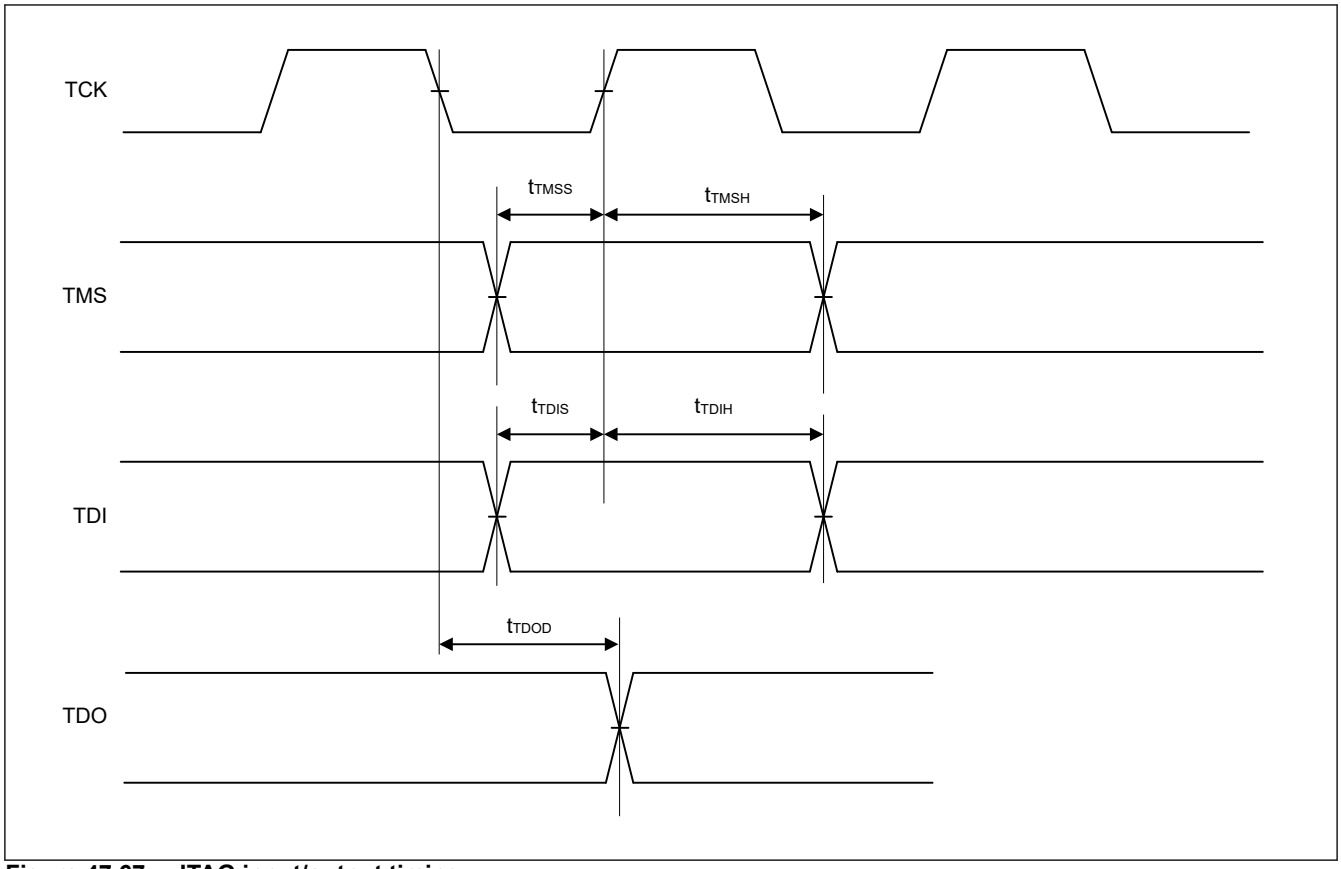



**Figure 47.67 JTAG input/output timing**

## 47.15 Serial Wire Debug (SWD)

**Table 47.49 SWD**

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| SWCLK clock cycle time | $t_{SWCKcyc}$ | 40 | — | — | ns | Figure 47.68 |
| SWCLK clock high pulse width | $t_{SWCKH}$ | 15 | — | — | ns | |
| SWCLK clock low pulse width | $t_{SWCKL}$ | 15 | — | — | ns | |
| SWCLK clock rise time | $t_{SWCKr}$ | — | — | 5 | ns | |
| SWCLK clock fall time | $t_{SWCKf}$ | — | — | 5 | ns | |
| SWDIO setup time | $t_{SWDS}$ | 8 | — | — | ns | Figure 47.69 |
| SWDIO hold time | $t_{SWDH}$ | 8 | — | — | ns | |
| SWDIO data delay time | $t_{SWDD}$ | 2 | — | 28 | ns | |

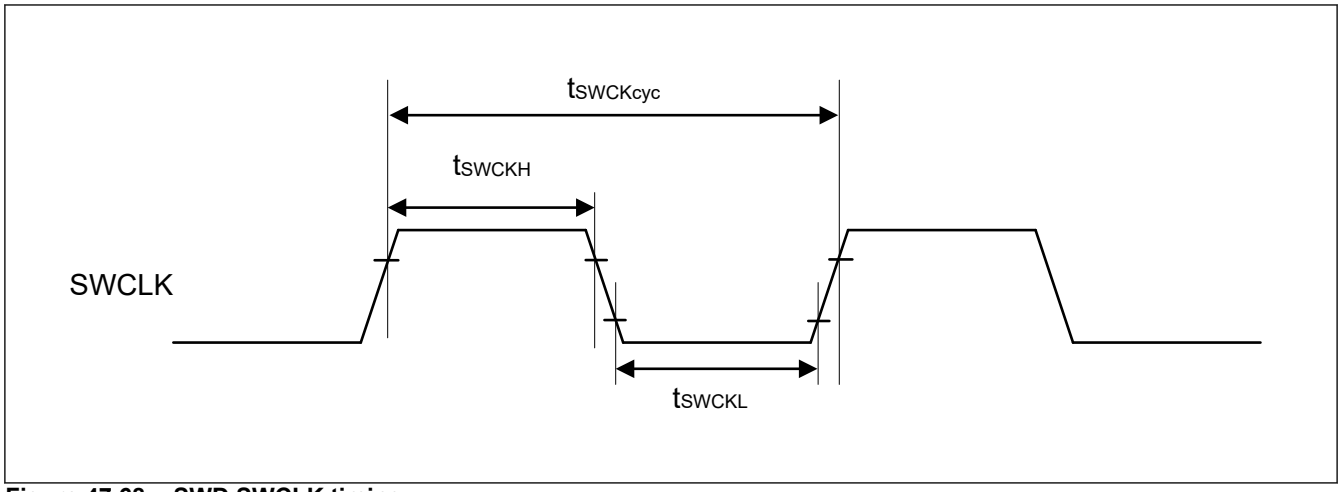



**Figure 47.68 SWD SWCLK timing**

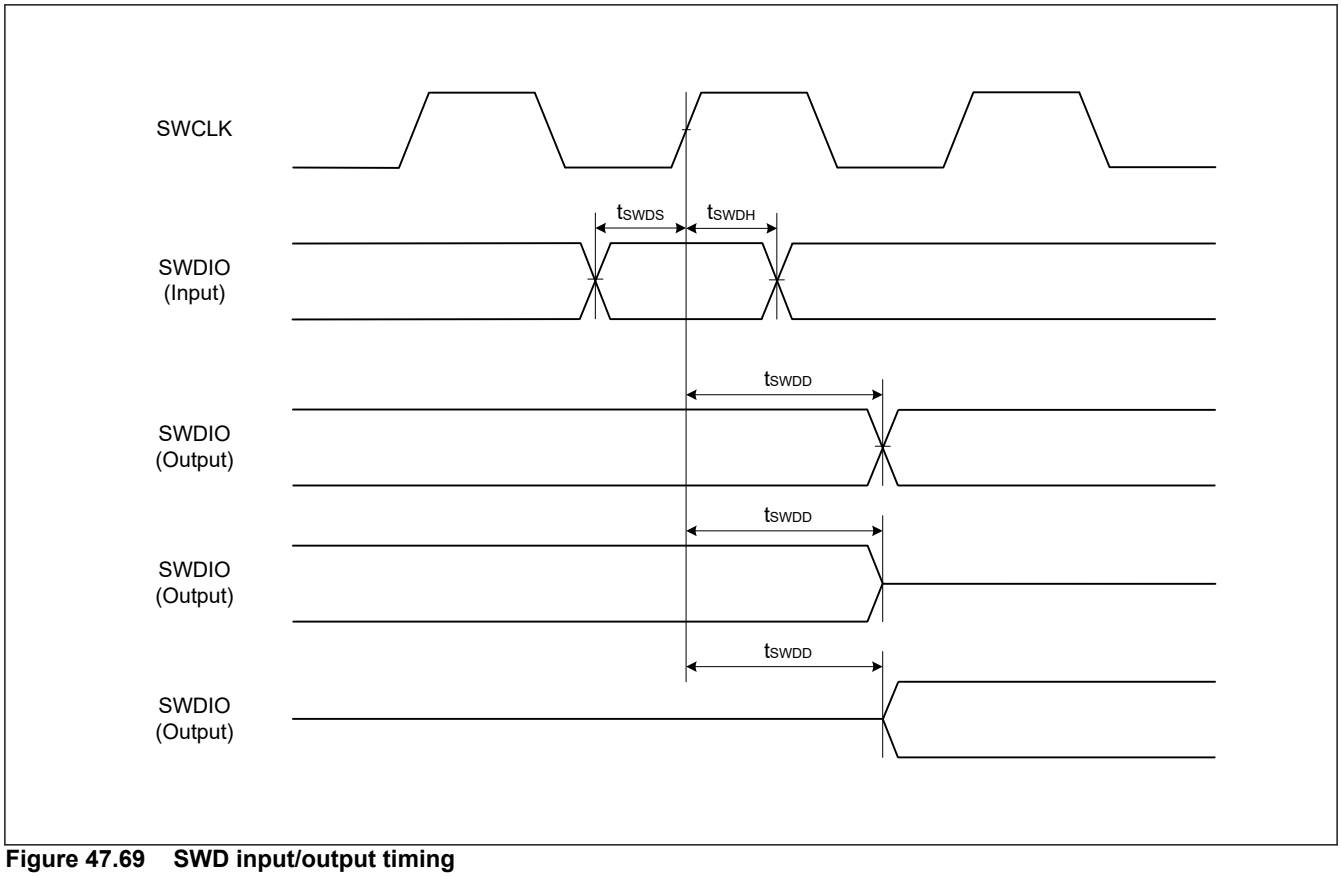



**Figure 47.69 SWD input/output timing**

## 47.16 Embedded Trace Macro Interface (ETM)

**Table 47.50 ETM (1 of 2)**

Conditions: High speed high drive output is selected in the Port Drive Capability bit in the PmnPFS register.

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| TCLK clock cycle time | $t_{TCLKcyc}$ | 40 | — | — | ns | Figure 47.70 |
| TCLK clock high pulse width | $t_{TCLKH}$ | 17 | — | — | ns | |
| TCLK clock low pulse width | $t_{TCLKL}$ | 17 | — | — | ns | |
| TCLK clock rise time | $t_{TCLKr}$ | — | — | 3 | ns | |
| TCLK clock fall time | $t_{TCLKf}$ | — | — | 3 | ns | |

**Table 47.50 ETM (2 of 2)**

Conditions: High speed high drive output is selected in the Port Drive Capability bit in the PmnPFS register.

| Parameter | Symbol | Min | Typ | Max | Unit | Test conditions |
|---|---|---|---|---|---|---|
| TDATA[3:0] output setup time | $t_{TRDS}$ | 3.5 | — | — | ns | Figure 47.71 |
| TDATA[3:0] output hold time | $t_{TRDH}$ | 2.5 | — | — | ns | |

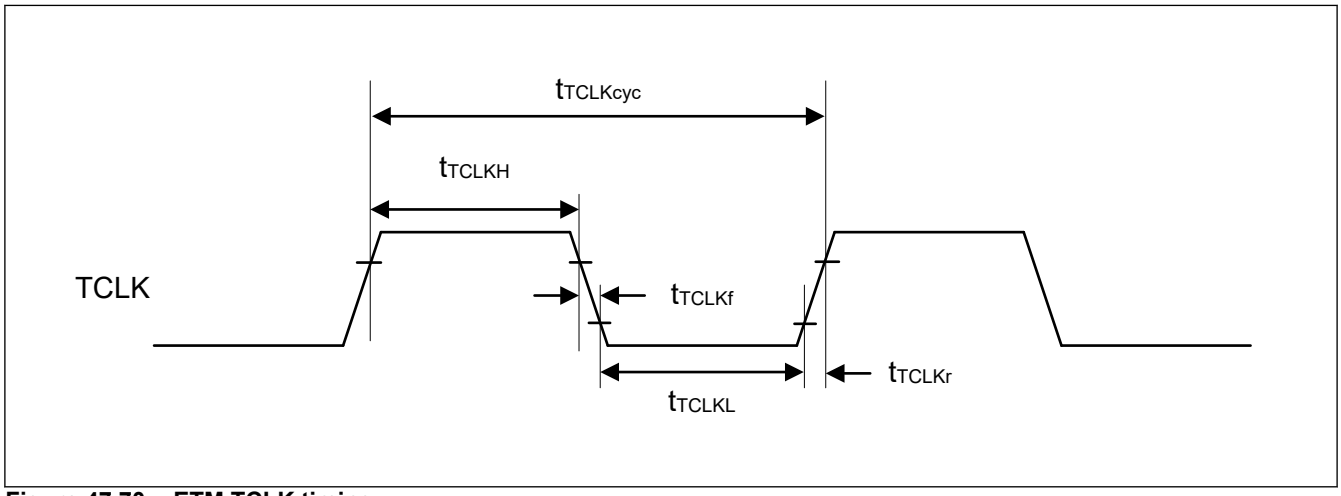



**Figure 47.70 ETM TCLK timing**

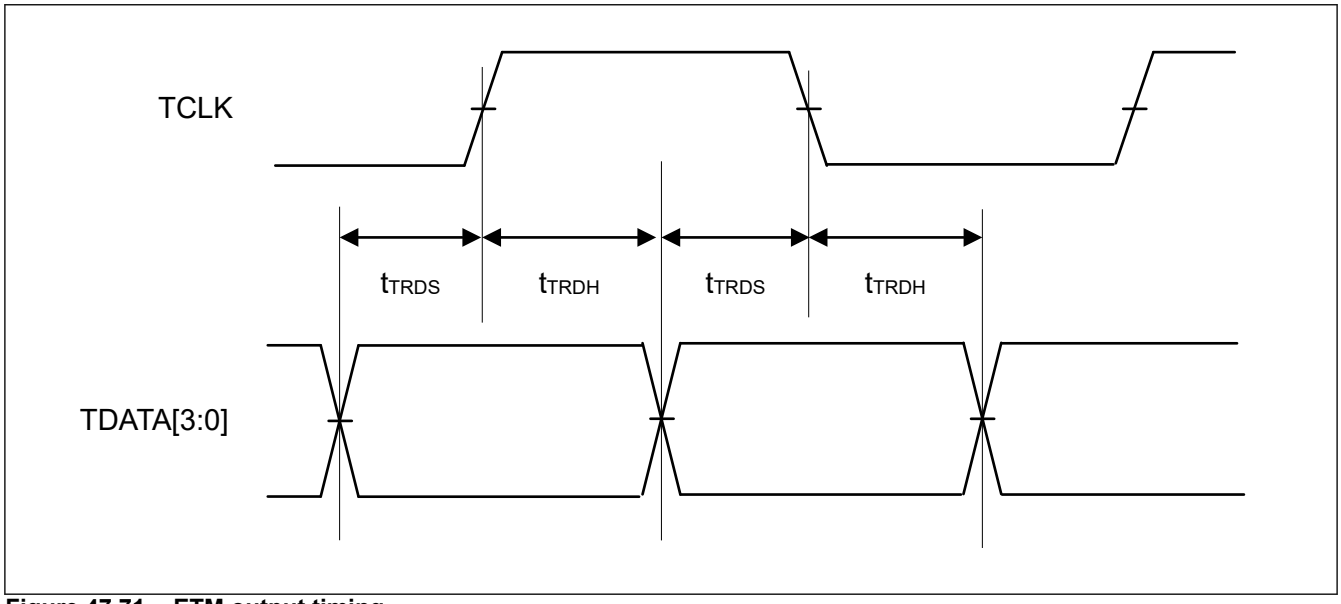



**Figure 47.71 ETM output timing**

# Appendix 1. Port States in Each Processing Mode

| Function | Pin function | Reset | Software Standby mode | Deep Software Standby mode | After Deep Software Standby mode is canceled (return to startup mode) | |
|---|---|---|---|---|---|---|
| | | | | | IOKEEP = 0 | IOKEEP = 1*1 |
| Mode | MD | Pull-up | Keep-O | Keep | Hi-Z | Keep |
| JTAG | TCK/TMS/TDI | Pull-up | Keep-O | Keep | Hi-Z | Keep |
| | TDO | output | Keep-O | Keep | TDO output | Keep |
| IRQ | IRQx | Hi-Z | Keep-O*2 | Keep | Hi-Z | Keep |
| | IRQx-DS | Hi-Z | Keep-O*2 | Keep*3 | Hi-Z | Keep |
| AGT | AGTIOn | Hi-Z | Keep-O*2 | Keep | Hi-Z | Keep |
| | AGTIOn (n=1,3) | Hi-Z | Keep-O*2 | Keep*3 | Hi-Z | Keep |
| SCI | RXD0 | Hi-Z | Keep-O*2 | Keep | Hi-Z | Keep |
| IIC | SCLn/SDAn | Hi-Z | Keep-O*2 | Keep | Hi-Z | Keep |
| USBFS | USB_OVRCURx | Hi-Z | Keep-O*2 | Keep | Hi-Z | Keep |
| | USB_OVRCURx-DS/ USB_VBUS | Hi-Z | Keep-O*2 | Keep*3 | Hi-Z | Keep |
| | USB_DP/USB_DM | Hi-Z | Keep-O*4 | Keep*3 | Hi-Z | Keep |
| RTC | RTCICx | Hi-Z | Keep-O*2 | Keep*3 | Hi-Z | Keep |
| | RTCOUT | Hi-Z | [RTCOUT selected] RTCOUT output | Keep | Hi-Z | Keep |
| CLKOUT | CLKOUT | Hi-Z | [CLKOUT selected] CLKOUT output | Keep | Hi-Z | Keep |
| DAC | DAn | Hi-Z | [DAn output (DAOE = 1)] D/A output retained | Keep | Hi-Z | Keep |
| Others | — | Hi-Z | Keep-O | Keep | Hi-Z | Keep |

Note: H: High-level
L: Low-level
Hi-Z: High-impedance
Keep-O: Output pins retain their previous values. Input pins go to high-impedance.
Keep: Pin states are retained during periods in Software Standby mode.

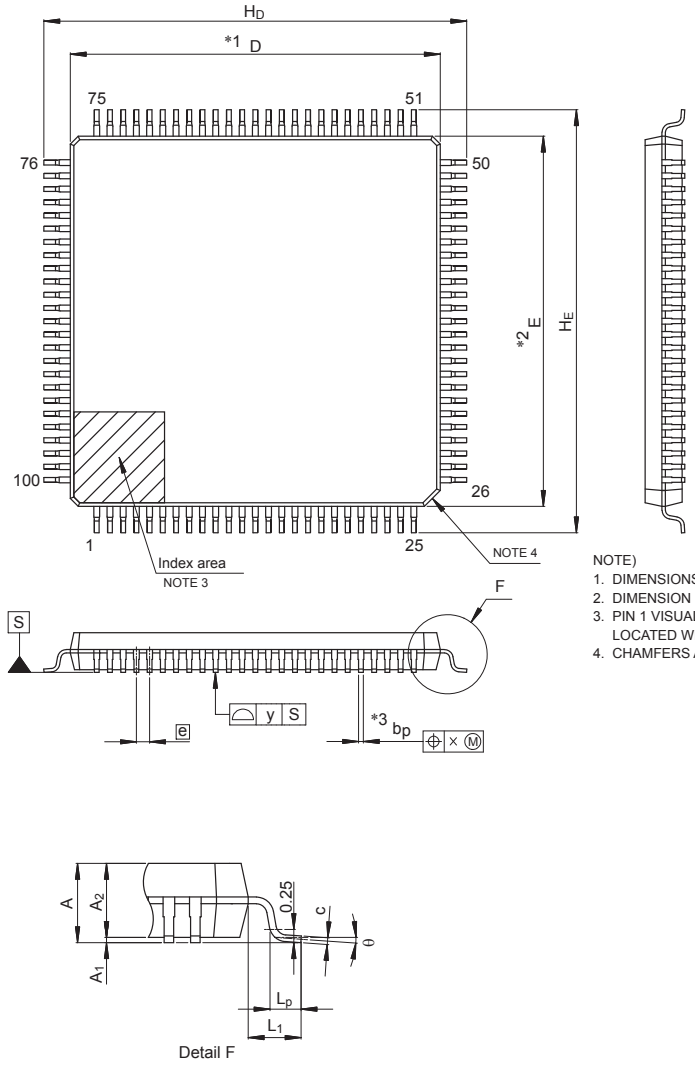
Note 1. Retains the I/O port state until the DPSBYCR.IOKEEP bit is cleared to 0.
Note 2. Input is enabled if the pin is specified as the Software Standby canceling source while it is used as an external interrupt pin.
Note 3. Input is enabled if the pin is specified as the Deep Software Standby canceling source.
Note 4. Input is enabled while the pin is used as an input pin.

# Appendix 2. Package Dimensions

Information on the latest version of the package dimensions or mountings is displayed in “Packages” on the Renesas Electronics Corporation website.

| JEITA Package Code | RENESAS Code | Previous Code | MASS (Typ) [g] |
|---|---|---|---|
| P-LFQFP100-14x14-0.50 | PLQP0100KB-B | — | 0.6 |

Unit: mm







NOTE)
1. DIMENSIONS “*1” AND “*2” DO NOT INCLUDE MOLD FLASH.
2. DIMENSION “*3” DOES NOT INCLUDE TRIM OFFSET.
3. PIN 1 VISUAL INDEX FEATURE MAY VARY, BUT MUST BE LOCATED WITHIN THE HATCHED AREA.
4. CHAMFERS AT CORNERS ARE OPTIONAL, SIZE MAY VARY.

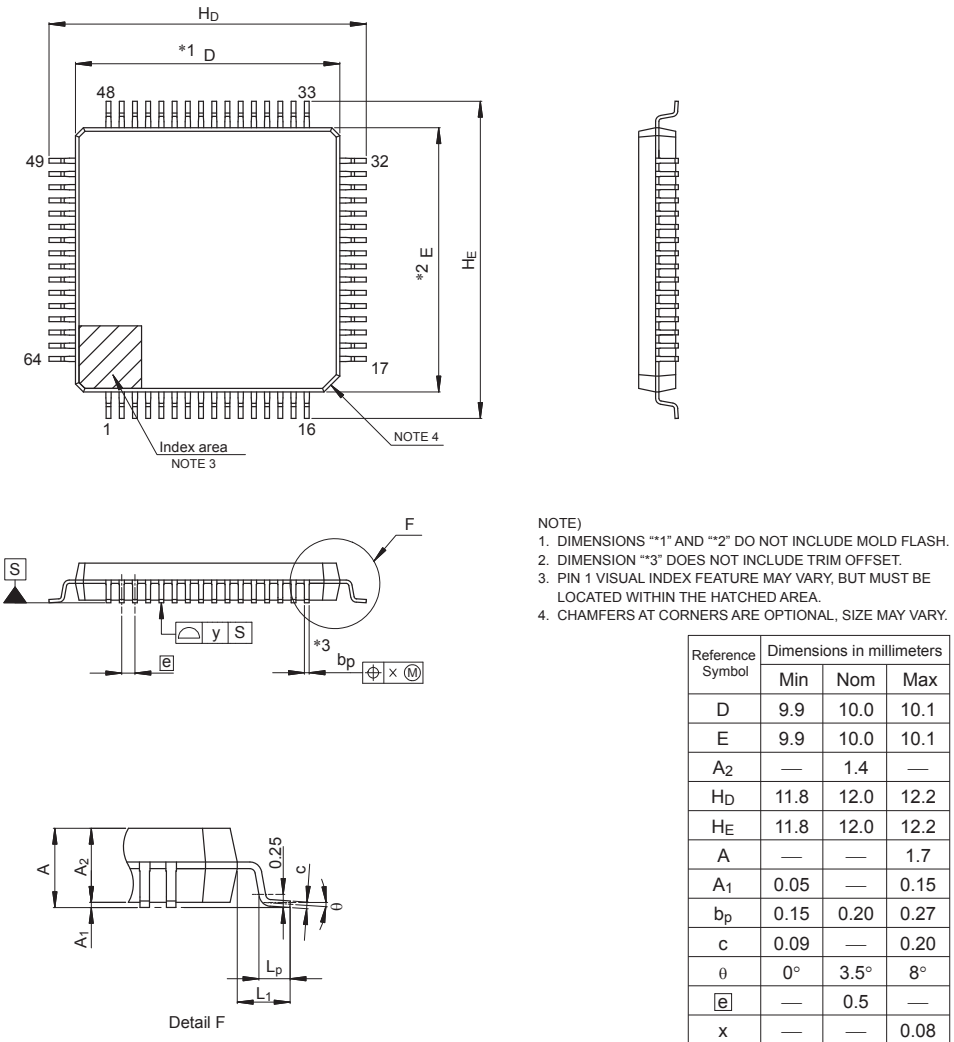
| Reference Symbol | Dimensions in millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | 13.9 | 14.0 | 14.1 |
| E | 13.9 | 14.0 | 14.1 |
| $A_2$ | — | 1.4 | — |
| $H_D$ | 15.8 | 16.0 | 16.2 |
| $H_E$ | 15.8 | 16.0 | 16.2 |
| A | — | — | 1.7 |
| $A_1$ | 0.05 | — | 0.15 |
| $b_p$ | 0.15 | 0.20 | 0.27 |
| c | 0.09 | — | 0.20 |
| θ | 0° | 3.5° | 8° |
| e | — | 0.5 | — |
| x | — | — | 0.08 |
| y | — | — | 0.08 |
| $L_p$ | 0.45 | 0.6 | 0.75 |
| $L_1$ | — | 1.0 | — |

© 2015 Renesas Electronics Corporation. All rights reserved.

**Figure 2.1 LQFP 100-pin**

| JEITA Package Code | RENESAS Code | Previous Code | MASS (Typ) [g] |
|---|---|---|---|
| P-LFQFP64-10x10-0.50 | PLQP0064KB-C | — | 0.3 |

Unit: mm




NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.
3. PIN 1 VISUAL INDEX FEATURE MAY VARY, BUT MUST BE LOCATED WITHIN THE HATCHED AREA.
4. CHAMFERS AT CORNERS ARE OPTIONAL, SIZE MAY VARY.

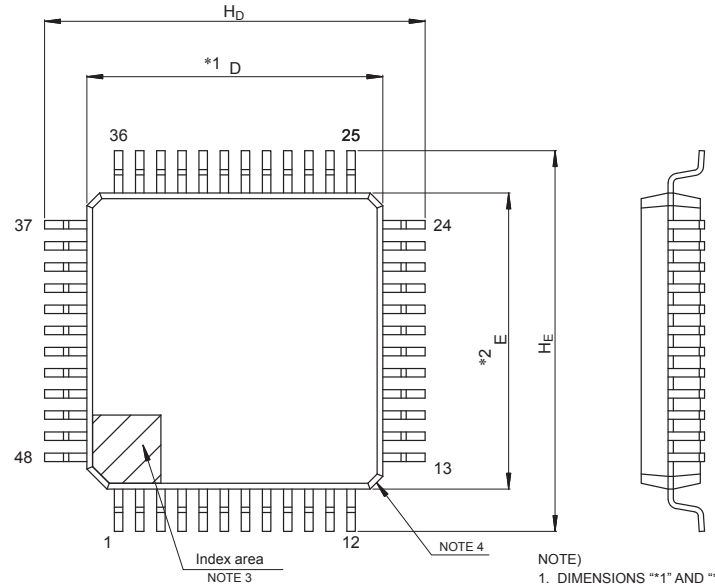
| Reference Symbol | Dimensions in millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | 9.9 | 10.0 | 10.1 |
| E | 9.9 | 10.0 | 10.1 |
| $A_2$ | — | 1.4 | — |
| $H_D$ | 11.8 | 12.0 | 12.2 |
| $H_E$ | 11.8 | 12.0 | 12.2 |
| A | — | — | 1.7 |
| $A_1$ | 0.05 | — | 0.15 |
| $b_p$ | 0.15 | 0.20 | 0.27 |
| c | 0.09 | — | 0.20 |
| θ | 0° | 3.5° | 8° |
| e | — | 0.5 | — |
| x | — | — | 0.08 |
| y | — | — | 0.08 |
| $L_p$ | 0.45 | 0.6 | 0.75 |
| $L_1$ | — | 1.0 | — |

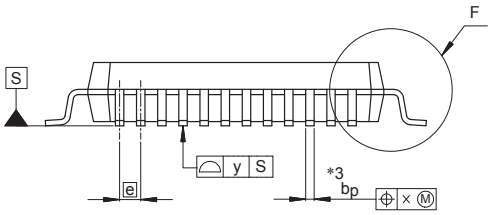
© 2015 Renesas Electronics Corporation. All rights reserved.

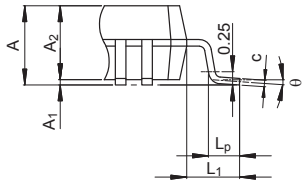
**Figure 2.2 LQFP 64-pin**

| JEITA Package Code | RENESAS Code | Previous Code | MASS (Typ) [g] |
|---|---|---|---|
| P-LFQFP48-7x7-0.50 | PLQP0048KB-B | — | 0.2 |

Unit: mm




Detail F

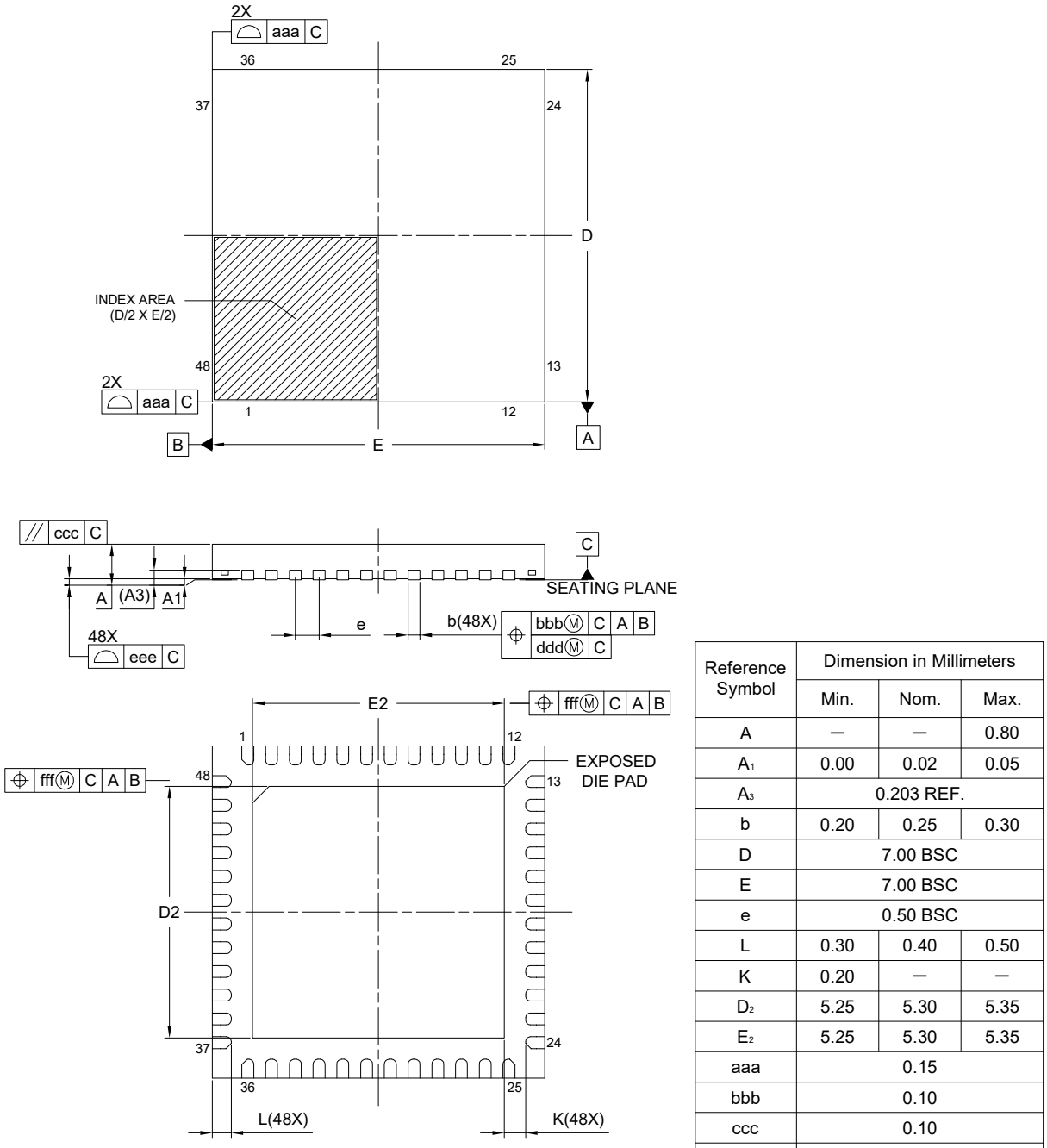
NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.
3. PIN 1 VISUAL INDEX FEATURE MAY VARY, BUT MUST BE LOCATED WITHIN THE HATCHED AREA.
4. CHAMFERS AT CORNERS ARE OPTIONAL, SIZE MAY VARY.

| Reference Symbol | Dimensions in millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | 6.9 | 7.0 | 7.1 |
| E | 6.9 | 7.0 | 7.1 |
| $A_2$ | — | 1.4 | — |
| $H_D$ | 8.8 | 9.0 | 9.2 |
| $H_E$ | 8.8 | 9.0 | 9.2 |
| A | — | — | 1.7 |
| $A_1$ | 0.05 | — | 0.15 |
| $b_p$ | 0.17 | 0.20 | 0.27 |
| c | 0.09 | — | 0.20 |
| θ | 0° | 3.5° | 8° |
| e | — | 0.5 | — |
| x | — | — | 0.08 |
| y | — | — | 0.08 |
| $L_p$ | 0.45 | 0.6 | 0.75 |
| $L_1$ | — | 1.0 | — |

© 2015 Renesas Electronics Corporation. All rights reserved.

**Figure 2.3 LQFP 48-pin**

| JEITA Package code | RENESAS code | MASS(TYP.)[g] |
|---|---|---|
| P-HWQFN048-7x7-0.50 | PWQN0048KC-A | 0.13 g |




| Reference Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min. | Nom. | Max. |
| A | — | — | 0.80 |
| $A_1$ | 0.00 | 0.02 | 0.05 |
| $A_3$ | | 0.203 REF. | |
| b | 0.20 | 0.25 | 0.30 |
| D | | 7.00 BSC | |
| E | | 7.00 BSC | |
| e | | 0.50 BSC | |
| L | 0.30 | 0.40 | 0.50 |
| K | 0.20 | — | — |
| $D_2$ | 5.25 | 5.30 | 5.35 |
| $E_2$ | 5.25 | 5.30 | 5.35 |
| aaa | | 0.15 | |
| bbb | | 0.10 | |
| ccc | | 0.10 | |
| ddd | | 0.05 | |
| eee | | 0.08 | |
| fff | | 0.10 | |

**Figure 2.4 QFN 48-pin**

# Appendix 3. I/O Registers

This appendix describes I/O register address and access cycles by function.

## 3.1 Peripheral Base Addresses

This section provides the base addresses for peripherals described in this manual. Table 3.1 shows the name, description, and the base address of each peripheral.

**Table 3.1 Peripheral base address (1 of 2)**

| Name | Description | Base address |
|---|---|---|
| RMPU | Renesas Memory Protection Unit | 0x4000_0000 |
| TZF | TrustZone Filter | 0x4000_0E00 |
| SRAM | SRAM Control | 0x4000_2000 |
| BUS | BUS Control | 0x4000_3000 |
| DMAC0 | Direct memory access controller 0 | 0x4000_5000 |
| DMAC1 | Direct memory access controller 1 | 0x4000_5040 |
| DMAC2 | Direct memory access controller 2 | 0x4000_5080 |
| DMAC3 | Direct memory access controller 3 | 0x4000_50C0 |
| DMAC4 | Direct memory access controller 4 | 0x4000_5100 |
| DMAC5 | Direct memory access controller 5 | 0x4000_5140 |
| DMAC6 | Direct memory access controller 6 | 0x4000_5180 |
| DMAC7 | Direct memory access controller 7 | 0x4000_51C0 |
| DMA | DMAC Module Activation | 0x4000_5200 |
| DTC | Data Transfer Controller | 0x4000_5400 |
| ICU | Interrupt Controller | 0x4000_6000 |
| CPSCU | CPU System Security Control Unit | 0x4000_8000 |
| DBG | Debug Function | 0x400_1B000 |
| FCACHE | Flash Cache | 0x400_1C100 |
| SYSC | System Control | 0x4001_E000 |
| PORT0 | Port 0 Control Registers | 0x4008_0000 |
| PORT1 | Port 1 Control Registers | 0x4008_0020 |
| PORT2 | Port 2 Control Registers | 0x4008_0040 |
| PORT3 | Port 3 Control Registers | 0x4008_0060 |
| PORT4 | Port 4 Control Registers | 0x4008_0080 |
| PORT5 | Port 5 Control Registers | 0x4008_00A0 |
| PORT6 | Port 6 Control Registers | 0x4008_00C0 |
| PORT7 | Port 7 Control Registers | 0x4008_00E0 |
| PFS | Pmn Pin Function Control Register | 0x4008_0800 |
| ELC | Event Link Controller | 0x4008_2000 |
| RTC | Realtime Clock | 0x4008_3000 |
| IWDT | Independent Watchdog Timer | 0x4008_3200 |
| WDT | Watchdog Timer | 0x4008_3400 |
| CAC | Clock Frequency Accuracy Measurement Circuit | 0x4008_3600 |
| MSTP | Module Stop Control A, B, C, D | 0x4008_4000 |
| POEG | Port Output Enable Module for GPT | 0x4008_A000 |

**Table 3.1 Peripheral base address (2 of 2)**

| Name | Description | Base address |
|---|---|---|
| USBFS | USB 2.0 FS Module | 0x4009_0000 |
| SDHI0 | SD Host Interface 0 | 0x4009_2000 |
| SSIE0 | Serial Sound Interface Enhanced (SSIE) | 0x4009_D000 |
| IIC0 | Inter-Integrated Circuit 0 | 0x4009_F000 |
| IIC0WU | Inter-Integrated Circuit 0 Wake-up Unit | 0x4009_F014 |
| IIC1 | Inter-Integrated Circuit 1 | 0x4009_F100 |
| CAN0 | CAN0 Module | 0x400A_8000 |
| CTSU | Capacitive Touch Sensing Unit | 0x400D_0000 |
| PSCU | Peripheral Security Control Unit | 0x400E_0000 |
| AGT0 | Low Power Asynchronous General purpose Timer 0 | 0x400E_8000 |
| AGT1 | Low Power Asynchronous General purpose Timer 1 | 0x400E_8100 |
| AGT2 | Low Power Asynchronous General purpose Timer 2 | 0x400E_8200 |
| AGT3 | Low Power Asynchronous General purpose Timer 3 | 0x400E_8300 |
| AGT4 | Low Power Asynchronous General purpose Timer 4 | 0x400E_8400 |
| AGT5 | Low Power Asynchronous General purpose Timer 5 | 0x400E_8500 |
| TSN | Temperature Sensor | 0x400F_3000 |
| CRC | CRC Calculator | 0x4010_8000 |
| DOC | Data Operation Circuit | 0x4010_9000 |
| SCI0 | Serial Communication Interface 0 | 0x4011_8000 |
| SCI1 | Serial Communication Interface 1 | 0x4011_8100 |
| SCI2 | Serial Communication Interface 2 | 0x4011_8200 |
| SCI3 | Serial Communication Interface 3 | 0x4011_8300 |
| SCI4 | Serial Communication Interface 4 | 0x4011_8400 |
| SCI9 | Serial Communication Interface 9 | 0x4011_8900 |
| SPI0 | Serial Peripheral Interface 0 | 0x4011_A000 |
| SCE9 | Secure Cryptographic Engine | 0x4016_1000 |
| GPT320 | General PWM 32-Bit Timer 0 | 0x4016_9000 |
| GPT321 | General PWM 32-Bit Timer 1 | 0x4016_9100 |
| GPT322 | General PWM 32-Bit Timer 2 | 0x4016_9200 |
| GPT323 | General PWM 32-Bit Timer 3 | 0x4016_9300 |
| GPT164 | General PWM 16-Bit Timer 4 | 0x4016_9400 |
| GPT165 | General PWM 16-Bit Timer 5 | 0x4016_9500 |
| GPT166 | General PWM 16-Bit Timer 6 | 0x4016_9600 |
| GPT167 | General PWM 16-Bit Timer 7 | 0x4016_9700 |
| GPT_OPS | Output Phase Switching Controller | 0x4016_9A00 |
| ADC120 | 12bit A/D Converter 0 | 0x4017_0000 |
| DAC12 | 12-bit D/A converter | 0x4017_1000 |
| FLAD | Data Flash | 0x407F_C000 |
| FACI | Flash Application Command Interface | 0x407F_E000 |
| QSPI | Quad-SPI | 0x6400_0000 |

Note: Name = Peripheral name
Description = Peripheral functionality
Base address = Lowest reserved address or address used by the peripheral

## 3.2 Access Cycles

This section provides access cycle information for the I/O registers described in this manual.

- Registers are grouped by associated module.
- The number of access cycles indicates the number of cycles based on the specified reference clock.
- In the internal I/O area, reserved addresses that are not allocated to registers must not be accessed, otherwise operations cannot be guaranteed.
- The number of I/O access cycles depends on bus cycles of the internal peripheral bus, divided clock synchronization cycles, and wait cycles of each module. Divided clock synchronization cycles differ depending on the frequency ratio between ICLK and PCLK.
- When the frequency of ICLK is equal to that of PCLK, the number of divided clock synchronization cycles is always constant.
- When the frequency of ICLK is greater than that of PCLK, at least 1 PCLK cycle is added to the number of divided clock synchronization cycles.
- The number of write access cycles indicates the number of cycles obtained by non-bufferable write access.

Note: This applies to the number of cycles when access from the CPU does not conflict with the instruction fetching to the external memory or bus access from other bus masters such as DTC or DMAC.

**Table 3.2 Access cycles (1 of 3)**

| Peripherals | Address | | Number of access cycles | | | | Cycle Unit | Related function |
|---|---|---|---|---|---|---|---|---|
| | | | ICLK = PCLK | | ICLK > PCLK*1 | | | |
| | From | To | Read | Write | Read | Write | | |
| RMPU, TZF, SRAM, BUS, DMACn, DMA, DTC, ICU | 0x4000_0000 | 0x4000_6FFF | 2 | 2 | 2 | 2 | ICLK | Renesas Memory Protection Unit, TrustZone Filter, SRAM Control, BUS Control, Direct memory access controller n, DMAC Module Activation, DTC Control Register, Interrupt Controller |
| CPSCU, DBG, FCACHE | 0x4000_8000 | 0x4001_CFFF | 4 | 3 | 4 | 3 | ICLK | CPU System Security Control Unit, Debug Function, Flash Cache |
| SYSC | 0x4001_E000 | 0x4001_E3FF | 5 | 4 | 5 | 4 | ICLK | System Control |
| SYSC | 0x4001_E400 | 0x4001_E5FF | 9 | 8 | 5 to 8 | 5 to 8 | PCLKB | System Control |
| PORTn, PFS | 0x4008_0000 | 0x4008_0FFF | 5 | 4 | 2 to 5 | 2 to 4 | PCLKB | Port n Control Registers, Pmn Pin Function Control Register |
| ELC, RTC, IWDT, WDT, CAC | 0x4008_2000 | 0x4008_3FFF | 5 | 4 | 3 to 5 | 2 to 4 | PCLKB | Event Link Controller, Realtime Clock, Independent Watchdog Timer, Watchdog Timer, Clock Frequency Accuracy Measurement Circuit |
| MSTP | 0x4008_4000 | 0x4008_4FFF | 5 | 4 | 2 to 5 | 2 to 4 | PCLKB | Module Stop Control |
| POEG | 0x4008_A000 | 0x4008_AFFF | 5 | 4 | 3 to 5 | 2 to 4 | PCLKB | Port Output Enable Module for GPT |
| USBFS | 0x4009_0000 | 0x4009_03FF | 6 | 5 | 3 to 6 | 3 to 5 | PCLKB | USB 2.0 FS Module |
| USBFS | 0x4009_0400 | 0x4009_04FF | 4 | 3 | 1 to 4 | 1 to 3 | PCLKB | USB 2.0 FS Module |

**Table 3.2 Access cycles (2 of 3)**

| Peripherals | Address | | Number of access cycles | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | ICLK = PCLK | | ICLK > PCLK*1 | | Cycle Unit | Related function |
| | From | To | Read | Write | Read | Write | | |
| SDHI0, SSIE0, IICn, IIC0WU | 0x4009_2000 | 0x4009_FFFF | 5 | 4 | 2 to 5 | 2 to 4 | PCLKB | SD Host Interface 0, Serial Sound Interface Enhanced, Inter-Integrated Circuit n, Inter-Integrated Circuit 0 Wake-up Unit |
| CANn | 0x400A_8000 | 0x400A_9FFF | 5 | 4 | 2 to 5 | 2 to 4 | PCLKB | CANn Module |
| CTSU | 0x400D_0000 | 0x400D_FFFF | 4 | 3 | 1 to 4 | 1 to 3 | PCLKB | Capacitive Touch Sensing Unit |
| PSCU | 0x400E_0000 | 0x400E_0FFF | 5 | 4 | 2 to 5 | 2 to 4 | PCLKB | Peripheral Security Control Unit |
| AGTn | 0x400E_8000 | 0x400E_8FFF | 7 | 4 | 5 to 7 | 2 to 4 | PCLKB | Low Power Asynchronous General purpose Timer n |
| TSN | 0x400F_3000 | 0x400F_3FFF | 5 | 4 | 2 to 5 | 2 to 4 | PCLKB | Temperature Sensor |
| CRC, DOC | 0x4010_8000 | 0x4010_9FFF | 5 | 4 | 2 to 5 | 2 to 4 | PCLKA | CRC Calculator, Data Operation Circuit |
| SCIn | 0x4011_8000 | 0x4011_8FFF | 5*2 | 4*2 | 2 to 5*2 | 2 to 4*2 | PCLKA | Serial Communication Interface n |
| SPIn | 0x4011_A000 | 0x4011_AFFF | 5*3 | 4*3 | 2 to 5*3 | 2 to 4*3 | PCLKA | Serial Peripheral Interface n |
| SCE9 | 0x4016_1000 | 0x4016_1FFF | 6 | 4 | 3 to 6 | 2 to 4 | PCLKA | Secure Cryptographic Engine |
| GPT32n, GPT16n, GPT_OPS | 0x4016_9000 | 0x4016_9FFF | 7 | 4 | 4 to 7 | 2 to 4 | PCLKA | General PWM 32-Bit Timer n, General PWM 16-Bit Timer n, Output Phase Switching Controller |
| ADC12n, DAC12 | 0x4017_0000 | 0x4017_2FFF | 5 | 4 | 2 to 5 | 2 to 4 | PCLKA | 12bit A/D Converter n, 12-bit D/A converter |
| QSPI | 0x6400_0000 | 0x6400_000F | 5 | 14 to *4 | 2 to 5 | 14 to *4 | PCLKA | Quad-SPI |
| QSPI | 0x6400_0010 | 0x6400_0013 | 25 to *4 | 6 to *4 | 25 to *4 | 5 to *4 | PCLKA | Quad-SPI |
| QSPI | 0x6400_0014 | 0x6400_0037 | 5 | 14 to *4 | 2 to 5 | 14 to *4 | PCLKA | Quad-SPI |
| QSPI | 0x6400_0804 | 0x6400_0807 | 4 | 3 | 1 to 4 | 1 to 3 | PCLKA | Quad-SPI |

**Table 3.2 Access cycles (3 of 3)**

| Peripherals | Address | | Number of access cycles | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | ICLK = FCLK | | ICLK > FCLK*1 | | Cycle Unit | Related function |
| | From | To | Read | Write | Read | Write | | |
| FLAD, FACI | 0x407F_C000 | 0x407F_EFFF | 3 | 3 | 2 to 3 | 2 to 3 | FCLK | Data Flash, Flash Application Command Interface |

Note 1. If the number of PCLK or FCLK cycles is non-integer (for example 1.5), the minimum value is without the decimal point, and the maximum value is rounded up to the decimal point. For example, 1.5 to 2. 5 is 1 to 3.

Note 2. When accessing a 16-bit register (FTDRHL, FRDRHL, FCR, FDR, LSR, and CDR), access is 2 cycles more than the value shown in Table 3.2. When accessing an 8-bit register (including FTDRH, FTDRL, FRDRH, and FRDRL), the access cycles are as shown in Table 3.2.

Note 3. When accessing the 32-bit register (SPDR), access is 2 cycles more than the value in Table 3.2. When accessing an 8-bit or 16-bit register (SPDR_HA), the access cycles are as shown in Table 3.2.

Note 4. The access cycles depend on the QSPI bus cycles.

# Revision History

**Revision 1.00 — August 06, 2020**

First edition, issued

**Revision 1.01 — August 21, 2020**

**2. CPU:**
- Changed the direction of the inequality sign with the frequency of the CPU clock in the notes of the SysTick timer.

**9. Clock Generation Circuit:**
- Changed the condition of HCSTP bit to start HOCO from 0 to 1.
- Added description of HCSTP bit when switching to Software Standby or Deep Software Standby.

**10. Low Power Modes:**
- Changed CECCLK described in the note of MSCPCRB register to CANCLK. For this reason, the target MSTPBi bit is changed to i = 3.

**31. Quad Serial Peripheral Interface:**
- Added Note 1 to SFMPAE bit.
- Changed Note 1 of SFMCCE bit to Note 2.
- Add note1 about output clock from QSPI.

**Revision 1.10 — January 27, 2021**

**1. Overview:**
- Added note to Figure 1.1 Block diagram.
- Changed pin name in Table 1.15 Pin functions.

**5. Resets:**
- Added Independent watchdog timer and Security Attribute related register to Table 5.3 Module-related registers initialized by each reset source.
- Changed the USBFS, Low power function and Security Attribute description in Table 5.3 Module-related registers initialized by each reset source.

**8. Clock Generation Circuit:**
- Changed register name in Table 8.4 FLL setting flow.
- Changed description of OSCSF.HOCOSF flag.
- Removed Note in 8.5.1 Oscillation Stop Detection and Operation after Detection.

**10. Low Power Modes:**
- Changed Note 1 in MSTPCRB register description.
- Changed description of SNZEDCR0 register.
- Added Note to LDOSCR, PL2LDOSCR register description.
- Changed the IRQMD[1:0] bits setting value in 10.7.3 Example of Software Standby Mode Application.
- Changed description of VBTBER.VBAE bit.
- Added VBTBKRn bit name to VBTBKR register description.

**11.Battery Backup Function:**
- Changed descriptipn of VBAE bit in VBTBER register.
- Changed descriptipn of VBATT Backup Register Usage.

**12. Register Write Protection:**
- Added VBATTMNSELR register to Table 12.1 Association between the bits in the PRCR register and registers to be protected.

**16. DMA Controller:**
- Changed DMSBS.DMSBSL[15:0], DMSBSH[15:0] bit description.
- Added Table 16.3 Available setting for DMSBS register in repeat-block transfer mode.
- Changed DMDBS.DMDBSL[15:0], DMDBSH[15:0] bit description.
- Added Table 16.4 Available setting for DMDBS register in repeat-block transfer mode.

**19. I/O Ports:**
- Removed VREFL pin from Table 19.3 Handling of unused pins.

**20. Port Output Enable for GPT:**
- Removed GTINTAD.GRPDTE bit description from 20.3 Output-Disable Control Operation.
- Removed Note from Figure 20.3 Output-disable release timing for GPT pin outputs.
- Changed PCLK information in Figure 20.4 Output timing of external trigger to the GPT.

**Revision 1.10 — January 27, 2021**

**21. General PWM Timer:**
- Removed description about dead time error in table 21.1 GPT specifications.
- Removed GTINT bit in GTINAD.
- Fixed Interrupt name typo in 21.4.1 Interrupt Sources and 21.5.1 Event Signal Output to ELC.
- Changed register and pin name in Figure 21.3 to 21.10, 21.14 to 21.27, 21.29, 21.46 to 21.56, 21.64, 21.66, 21.67.
- Changed pin name to GTIOCnA and GTIOCnB in Table21.3.

**22. Low Power Asynchronous General Purpose Timer:**
- Changed data described in Figure 22.3 to 22.12.

**23. Realtime Clock:**
- Added RTCCRn.TCEN bit.

**24. Watchdog Timer:**
- Changed WDTCR.RPSS[1:0], RPES[1:0] bits to OFS0.WDTRPSS[1:0], WDTRPES[1:0] bits in 24.3.1.2 Auto start mode.
- Changed description in 24.3.4 Status Flags.

**25. Independent Watchdog Timer:**
- Changed description in 24.3.4 Status Flags.

**26. USB 2.0 Full-Speed Module:**
- Changed descriptipn of BCCTRL1 and BCCTRL2 registers.
- Added TCEN bit in RTCCRn register and description about this bit.
- Added PHYSECTRL register.
- Removed DPUSR0R.DPDDET0, DPUSR1R.DPDDETINT0, DPUSR1R.DPDDETE0 bit and DPBCCTRL register.
- Changed descriptipn of Figure 26.8 to Figure 26.9.
- Deieted interrupt PDDETINT in table 26.16.
- Changed descriptipn of 26.3.15 Battery Charging Detection Processing.
- Changed descriptipn of 26.3.15 Battery Charging Detection Processing.
- Changed descriptipn of 26.3.15.1 Processing in device controller mode.
- Changed description and figure of 26.3.15.2 Processing in host controller mode.

**27. Serial Communications Interface:**
- Removde transmit end interrupt from Table 27.1 SCI specifications.
- Changed channel number in ACTR register.

**28. $I^2C$ Bus Interface:**
- Changed base address in ICWUR and ICWUR2 register.

**30. Serial Peripheral Interface:**
- Changed SSL control function description in Table 30.1 SPI specifications.
- Removed unnecessary terminals of Figure 30.5, Figure 30.6, Figure 30.7, Figure 30.9, Figure 30.11 and Figure 30.12.

**31. Quad Serial Peripheral Interface:**
- Changed address of MCU internal space in Figure 31.2.

**32. Serial Sound Interface Enhanced:**
- Changed description of DWL[2:0] = 0x110 condition in Figure 32.7 Alignment of placement data at transmission.
- Changed description of DWL[2:0] = 0x110 condition in Figure 32.8 Alignment of placement data at reception.

**33. SD/MMC Host Interface:**
- Added Note 1 to value after reset information in SD_STOP.STP bit.
- Added STP bit and SDIF_MODE register to Table 33.4 Bits and flags initialized by SD/MMC Host Interface software reset.
- Added Note 1 to description of SDIF_MODE register.
- Changed register name in Figure 33.5 Read from SD_BUF0.
- Changed to with card condition in Figure 33.7 SD card detect with SDnDAT3.

**36. Secure Cryptographic Engine:**
- Added description about Unique ID and Hardware Unique in Table 36.1 and Figure36.1.

**37. 12-Bit A/D Converter:**
- Changed analog input information in Figure 37.2, 37.7, 37.8, 37.10 to 37.13, 37.18 to 37.25.
- Removed unnecessary pin information in 37.6.12 Port Settings When Using the ADC12 Input.
- Moved Chapter Using Data Buffers from 37.3.12 to 37.3.17

**38. 12-Bit D/A Converter:**
- Changed DAAMPCR and DAASWCR register position in Figure 38.1 DAC12 block diagram.

**39. Temperature Sensor:**
- Changed TSCDR register address.
- Added temperature sensor data information to TSCDR register description.

**43. Stanby SRAM:**
- Changed description of 43.3.5 Access Cycle.

**Revision 1.10 — January 27, 2021**

**44. Flash Memory:**
- Changed function of Bit 7:1 inFSAR register.
- Added description about the ASCII code representation of the product part number in PNRn register.
- Changed reset value from 0 to 1 of MCUVER register.
- Changed read cycle information in Table 44.31 Data Flash Memory.
- Added pin no. 18 information to Table 44.32 Pin assign for emulator.

**46. Security Features:**
- Added some Notes in 46.3.1 Changing the Lifecycle State and 46.3.5 Failure analysis.

**47. Electrical Characteristics:**
- Added information about supported functions and pins.
- Changed LDO to LDOn in Table 47.7 Operating and standby current.
- added Note 15 to PLL2-LDO operating current in Table 47.7 Operating and standby current.
- Removed ADTRG1 in Figure 47.25 ADC12 trigger input timing.

**Appendix:**
- Added QFN-48pin package dimensions in Figure 2.4 QFN 48-pin.

**Revision 1.20 — May 31, 2023**

**1. Overview:**
- Fixed 1.1 Function Outline.
- Fixed 1.3 Part Numbering.
- Fixed 1.5 Pin Functions.
- Fixed 1.6 Pin Assignments.

**2. CPU:**
- Fixed 2.5 Security Attribution for Memory.
- Fixed 2.6.3 Self-Hosted Debug Function.
- Fixed 2.7.4.2 DBGSTOPCR.
- Fixed 2.7.7 Processing on Error response generated by CPU access.

**6. Option-Setting Memory:**
- Fixed 6.2.1 OFS0 : Option Function Select Register 0.

**7. Low Voltage Detection:**
- Fixed 7.2.9 LVD2SR : Voltage Monitor 2 Circuit Status Register.
- Fixed 7.4 Reset from Voltage Monitor 0.
- Fixed 7.6 Interrupt and Reset from Voltage Monitor 2.

**8. Clock Generation Circuit:**
- Fixed Figure 8.1 Clock generation circuit block diagram.
- Fixed 8.2.8 MOSCCR : Main Clock Oscillator Control Register.
- Fixed 8.2.17 OSTDSR : Oscillation Stop Detection Status Register.
- Fixed 8.7.4 USB Clock (USBCLK).

**10. Low Power Modes:**
- Fixed Table 10.1 Specifications of the low power mode functions.
- Fixed 10.2.7 MSTPCRD : Module Stop Control Register D.
- Fixed 10.5 Function for Lower Operating Power Consumption.
- Fixed 10.5.1 Setting Operating Power Control Mode.
- Fixed Figure 10.10 Setting example of using SCI0 in Snooze mode entry.

**11. Battery Backup Function:**
- Fixed 11.2.5 VBTBKR[n].

**13. Interrupt Controller Unit:**
- Fixed 13.2.10 IRQCRi.
- Fixed 13.2.11 NMISR.
- Fixed 13.2.13 NMICLR.
- Fixed 13.3.1 Interrupt Vector Table.

**14. Buses:**
- Fixed Note in Figure 14.1 Bus Connection.
- Fixed 14.2.2 Parallel Operation.
- Fixed 14.3.9 BUSnERRSTAT.
- Fixed 14.3.10 DMACDTCERRSTAT.

**15. Memory Protection Unit:**
- Fixed 15.3 Bus Master MPU.

**Revision 1.20 — May 31, 2023**

**16. DMA Controller:**
- Fixed 16.1 Overview.
- Fixed 16.2.6 DMCRA.
- Fixed 16.2.10 DMAMD.
- Fixed 16.2.18 DMECHR.
- Fixed Table 16.15 Setting of use case: from interval address to single ring buffer.
- Fixed Table 16.16 Setting of use case: from unaligned ring buffer to single ring buffer.
- Fixed Table 16.17 Setting of use case: from single block to multi ring buffer.

**17. Data Transfer Controller:**
- Fixed Figure 17.1 DTC block diagram.
- Fixed 19.2.1 PCNTR1/PODR/PDR.
- Fixed 19.2.3 PCNTR3/PORR/POSR.
- Fixed 19.2.4 PCNTR4/EORR/EOSR.

**20. Port Output Enable for GPT:**
- Fixed 20.3.1.1 Digital Filter

**21. General PWM Timer:**
- Fixed 21.1 Overview.
- Fixed 21.2.1 GTWP.
- Fixed 21.2.4 GTCLR.
- Fixed 21.2.12 GTCR.
- Fixed 21.2.13 GTUDDTYC.
- Fixed 21.2.14 GTIOR.
- Fixed 21.2.15 GTINTAD.
- Fixed 21.2.16 GTST.
- Fixed 21.2.28 OPSCR.
- Fixed 21.3.1.1 Counter operation.
- Fixed 21.7.2 Disabling of Buffer Operation.
- Fixed 21.7.3 GTIOCnm Pin Output Negate Control (n = 0 to 7, m = A, B).
- Fixed 21.9.2 GTCCRn Settings during Compare Match Operation (n = A to F).

**22. Low Power Asynchronous General Purpose Timer:**
- Fixed Table 22.2 AGT I/O pins.
- Fixed 22.2.6 AGTMR2.
- Fixed 22.2.7 AGTIOC.
- Fixed 22.3.1 Reload Register and Counter Rewrite Operation.
- Fixed 22.3.10 Standby Mode.
- Fixed 22.4.3 When Changing Mode.

**23. Realtime Clock:**
- Fixed 23.2 Register Descriptions.
- Fixed 23.2.22 RCR1.
- Fixed 23.3.8.2 Adjustment by software.

**24. Watchdog Timer:**
- Fixed Figure 24.1 WDT block diagram.
- Fixed 24.2.2 WDTCR.
- Fixed 24.2.4 WDTRCR.
- Fixed 24.2.5 WDTCSTPR.
- Fixed 24.3.1.1 Register start mode.
- Fixed 24.3.1.2 Auto start mode.
- Fixed 24.3.8 Association between Option Function Select Register 0 (OFS0) and WDT Registers.

**25. Independent Watchdog Timer:**
- Fixed Figure 25.1 IWDT block diagram.
- Fixed 25.2.3 OFS0.
- Fixed 25.3.1 Auto Start Mode.
- Fixed 25.3.2 Refresh Operation.
- Fixed Figure 25.4 IWDT refresh operation waveforms when OFS0.IWDTCKS[3:0] = 0000b, OFS0.IWDTTOPS[1:0] = 11b.
- Fixed Figure 25.5 Processing for reading IWDT counter value when OFS0.IWDTCKS[3:0] = 0000b, OFS0.IWDTTOPS[1:0] = 11b.

**26. USB 2.0 Full-Speed Module:**
- Fixed 26.2.16 INTSTS0.

**Revision 1.20 — May 31, 2023**

**27. Serial Communications Interface:**
- Fixed 27.1 Overview.
- Fixed 27.4.1 Multi-Processor Serial Data Transmission.
- Fixed 27.4.2 Multi-Processor Serial Data Reception.
- Fixed 27.9.2 SS Function in Master Mode.
- Fixed 27.9.3 SS Function in Slave Mode.
- Fixed 27.10 Bit Rate Modulation Function.
- Fixed 27.15 Noise Cancellation Function.

**28. I2C Bus Interface:**
- Fixed Figure 28.34 Timing of normal wakeup mode 2.
- Fixed Figure 28.37 Timing of command recovery and EEP response modes.

**29. Controller Area Network:**
- Fixed the title of Figure 29.15 Structure of the FIDCRn registers (n = 0).

**31. Quad Serial Peripheral Interface:**
- Changed name of SFMSCK register to SFMSKC in Figure 31.1 QSPI block diagram.
- Updated Figure 31.2 Default area setting and memory map.
- Changed SFMOSW bit name to SFMOHW in 31.5.8 Hold Time for Serial Data Output.
- Updated title of Figure 31.6 Dual SPI protocol example for Fast Read Dual I/O.
- Changed SFMSMD register name to SFMSDC in 31.6.6 Fast Read Quad Output Instruction and in 31.6.7 Fast Read Quad I/O Instruction.

**32. Serial Sound Interface Enhanced:**
- Updated Figure 32.24 Stop/resume of AUDIO_MCK.
- Updated Figure 32.30 Configuration of the transmit FIFO data register and transmit shift register, and FIFO operation example.
- Updated Figure 32.31 Configuration of the receive FIFO data register and receive shift register, and FIFO operation example.
- Updated Figure 32.42 SSIE state transition.
- Updated 32.4.4 SSIFSR : FIFO Status Register.
- Updated 32.4.5 SSIFTDR : Transmit FIFO Data Register.
- Updated 32.4.6 SSIFRDR : Receive FIFO Data Register.
- Changed figure reference from Figure 32.51 to Figure 32.50 in 32.7.2.1 Data communication state.
- Fixed the title of 32.11.1.1 SSIBCK control.
- Fixed the title of 32.11.2.1 AUCKE control.
- Fixed 32.11.3.4 Switching transfer modes.

**33. SD/MMC Host Interface:**
- Fixed Table 33.1 SD/MMC Host Interface specifications.
- Fixed the base address in all registers.
- Fixed 33.2.7 SD_RSPj.
- Fixed 33.2.11 SD_INFO2.

**36. Secure Cryptographic Engine:**
- Fixed 36.1 Overview.
- Fixed Figure 36.3 Encryption and decryption timing (AES).

**37. 12-Bit A/D Converter**
- Fixed 37.1 Overview.
- Fixed 37.2.1 ADDRn.
- Changed the register names of ADANSA0 and ADANSA1 to ADADS0 and ADADS1 in 37.2.7 ADCSR.
- Fixed 37.2.14 ADADC.
- Updated 37.2.28 ADCMPDRn.
- Updated 37.2.29 ADWINnLB.
- Fixed the title of 37.2.18 ADSSTRn/ADSSTRL/ADSSTRT/ADSSTRO.
- Removed 0x3F from the table description for bits [5:0] of 37.2.33 ADCMPBNSR.
- Fixed 37.3.4.3 Group Priority Operation.
- Fixed 37.3.8 A/D-Converted Value Addition/Average Mode.

**42. SRAM:**
- Fixed 42.2.2 PARIOAD.
- Fixed 42.2.3 SRAMPRCR.

**Revision 1.20 — May 31, 2023**

**44. Flash Memory:**
- Fixed 44.4.7 MCUVER.
- Fixed 44.4.16 FBPROT1.
- Fixed 44.4.17 FSTATR.
- Fixed 44.4.18 FENTRYR.
- Fixed 44.4.19 FSUINITR.
- Fixed 44.4.26 FPCKAR.
- Fixed 44.4.27 FSUACR.
- Fixed Table 44.10 Programming methods.
- Fixed 44.9.3.8 Block Erase Command.
- Fixed 44.9.3.9 Multi Block Erase Command.
- Fixed 44.15.1 Overview.

**46. Security Features:**
- Fixed 46.2.2 Memory Security Attribution.
- Fixed Table 46.6 Address Space Security Attribution.
- Fixed 46.3.1 Changing the Lifecycle State.
- Fixed 46.4 Key Injection.

**47. Electrical Characteristics:**
- Fixed 47.2.1 Tj/Ta Definition.
- Fixed Table 47.7 Operating and standby current.

**Appendix 3. I/O Registers:**
- Fixed Table 3.2 Access cycles.

RA4M2 Group User’s Manual: Hardware

Publication Date: Rev.1.20 May 31, 2023
Rev.1.10 Jan 27, 2021

Published by: Renesas Electronics Corporation

# 32-Bit MCU
# RA4M2 Group

Renesas Electronics Corporation

R01UH0892EJ0120