MicroZed/PicoZed: Creating a Zynq Hardware Platform in Vivado

27 March 20177

Version 2016_4.01

Overview

With a traditional processor, the hardware platform is pre-defined. The manufacturer selected the processor parameters and built-in peripherals when the chip was designed. To make use of this pre-defined processor, you need only target that specific hardware platform in the software development tools.

The Zynq-7000 All Programmable SoC is different. Zynq provides multiple building blocks and leaves the definition to you as the design engineer. This adds flexibility, but it also means that a bit of work needs to be done up front before any software development can take place.

The first step in completing a Zynq design is to define and build the hardware platform. The purpose of this tutorial is to show you how to quickly and easily create a base hardware platform for MicroZed or PicoZed .

Objectives

When this tutorial is complete, you will be able to:

- Create a new project in Vivado, targeting MicroZed or PicoZed
- Create a block based design to insert an ARM processor core
- Import the MicroZed or PicoZed Zynq PS Preset settings
- Build and export the hardware platform

Experiment Setup

Software

The software used to test this reference design is:

- Windows-7 64-bit
- Xilinx Vivado 2016.4
- *Board Definition Install for Vivado 2015.3, 2015.4, 2016.1, 2016.2. 2016.4 26 Jan 2017*
    - MicroZed: http://microzed.org/support/documentation/1519
    - PicoZed + FMC Carrier V1: http://picozed.org/support/documentation/4701
    - PicoZed + FMC Carrier V2: http://picozed.org/support/documentation/13076

Hardware

The hardware setup used to test this reference design includes:

- Win-7 PC with the following recommended memory[1]:
    - 1.6 GB RAM available for the Xilinx tools to complete a XC7Z010 design
    - 2.3 GB RAM available for the Xilinx tools to complete a XC7Z015 design
    - 1.9 GB RAM available for the Xilinx tools to complete a XC7Z020 design
    - 2.7 GB RAM available for the Xilinx tools to complete a XC7Z030 design

[1] Refer to www.xilinx.com/design-tools/vivado/memory.htm

Experiment 1: Create a New Zynq Project in Vivado

The MicroZed Evaluation Kit is supported by Vivado WebPack (which is free). It supports MicroZed 7010 and 7020 as well as all four PicoZed versions (7010/7015/7020/7030). The Zynq Processing System (PS) may be used without anything programmed in the Programmable Logic (PL). This PS-only style is the simplest way to use Zynq, so that is what we will do during this lab. However, the power of Zynq is found in using soft IP in the PL, interconnecting PS to PL, and routing extra PS built-in peripherals through EMIO to PL I/Os, and then programming of the PL is required.

This tutorial will take advantage of built-in 3rd-party board definition files. The board definition archive contains all the files for both MicroZed and PicoZed. The identical archive is posted in three places.

- www.microzed.org → Support → Documentation → MicroZed → *MicroZed Board Definition Install for Vivado 2015.3, 2015.4, 2016.1, 2016.2, 2016.4*
- www.picozed.org → Support → Documentation → PicoZed FMC Carrier Card V1 (under Related parts for PicoZed) → *PicoZed Board Definition Install for Vivado 2015.3, 2015.4 2016.1, 2016.2, 2016.4*
- www.picozed.org → Support → Documentation → PicoZed FMC Carrier Card V2 (under Related parts for PicoZed) → *PicoZed Board Definition Install for Vivado 2015.3, 2015.4 2016.1, 2016.2, 2016.4*

1. If not previously completed, download the MicroZed/PicoZed Board Definition Install for Vivado 2016.4 archive and follow the instructions to install the board definitions.

2. Launch Vivado by selecting **Start → All Programs → Xilinx Design Tools → Vivado 2016.4 → Vivado 2016.4**.

3. Select **File → New Project** or click on **Create New Project** under *Quick Start*.

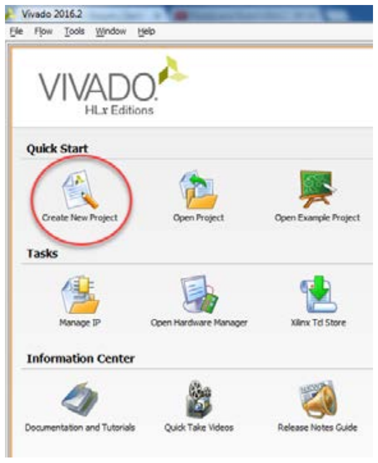
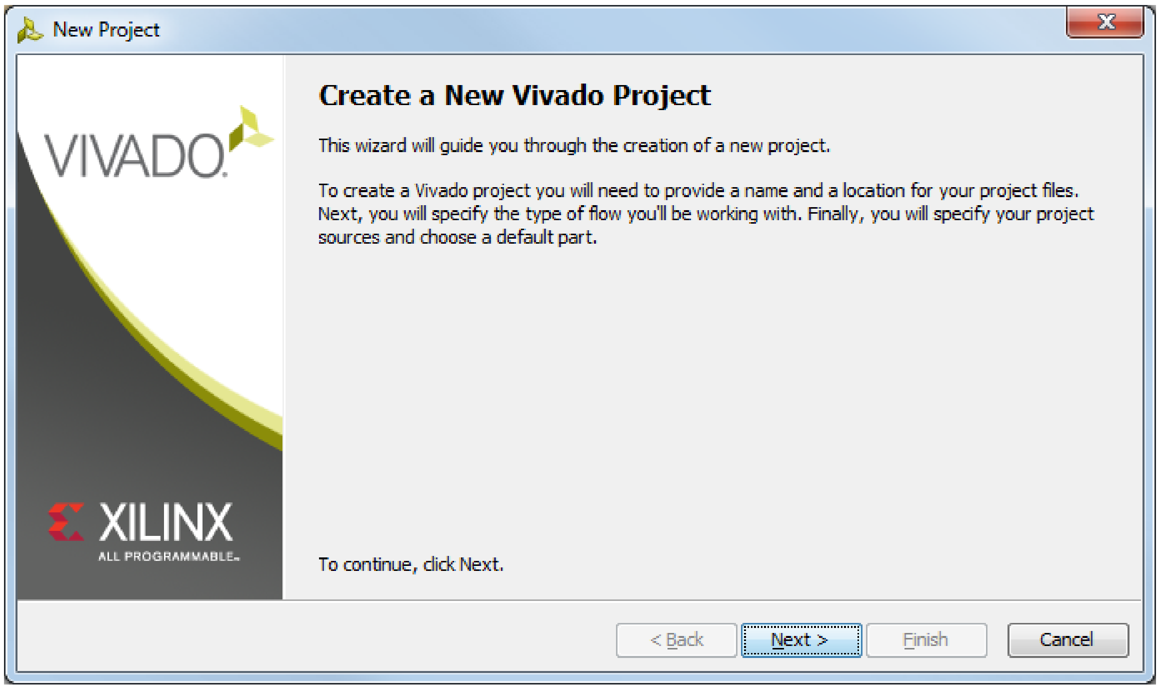


Figure 1 – Vivado Launched

4. Click **Next >**.



New Project

Create a New Vivado Project

This wizard will guide you through the creation of a new project.

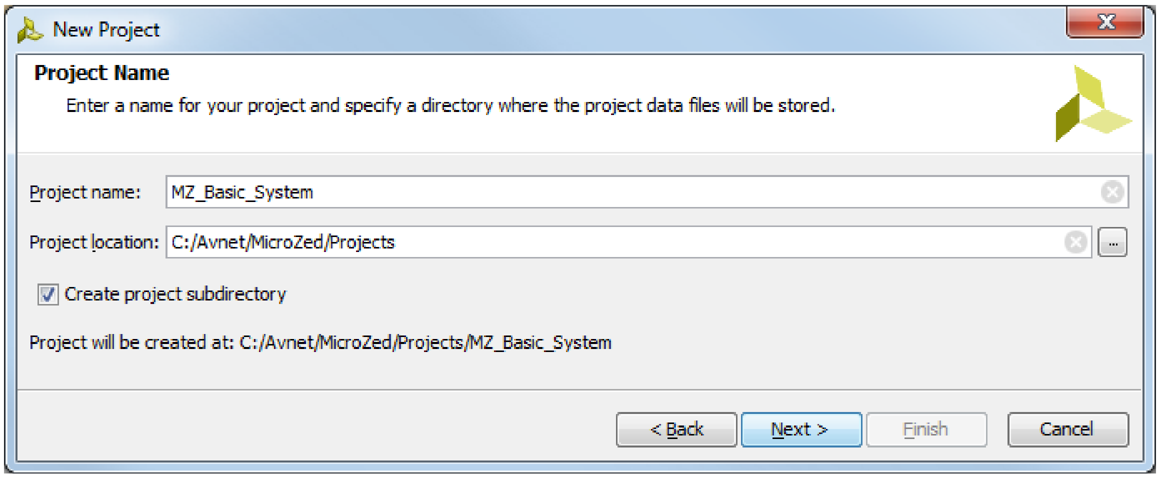
To create a Vivado project you will need to provide a name and a location for your project files. Next, you will specify the type of flow you'll be working with. Finally, you will specify your project sources and choose a default part.

To continue, click Next.

Figure 2 – New Vivado Project Wizard Launched

5. Click the browse icon ... . Browse to set the *Project location* to your desired project location and click **Select**.

6. Set the *Project name* to **`MZ_Basic_System`** or **`PZ_Basic_System`**. Also verify the ***`Create project subdirectory`*** checkbox is selected. Click **Next >**.



New Project

Project Name

Enter a name for your project and specify a directory where the project data files will be stored.

Project name: MZ_Basic_System

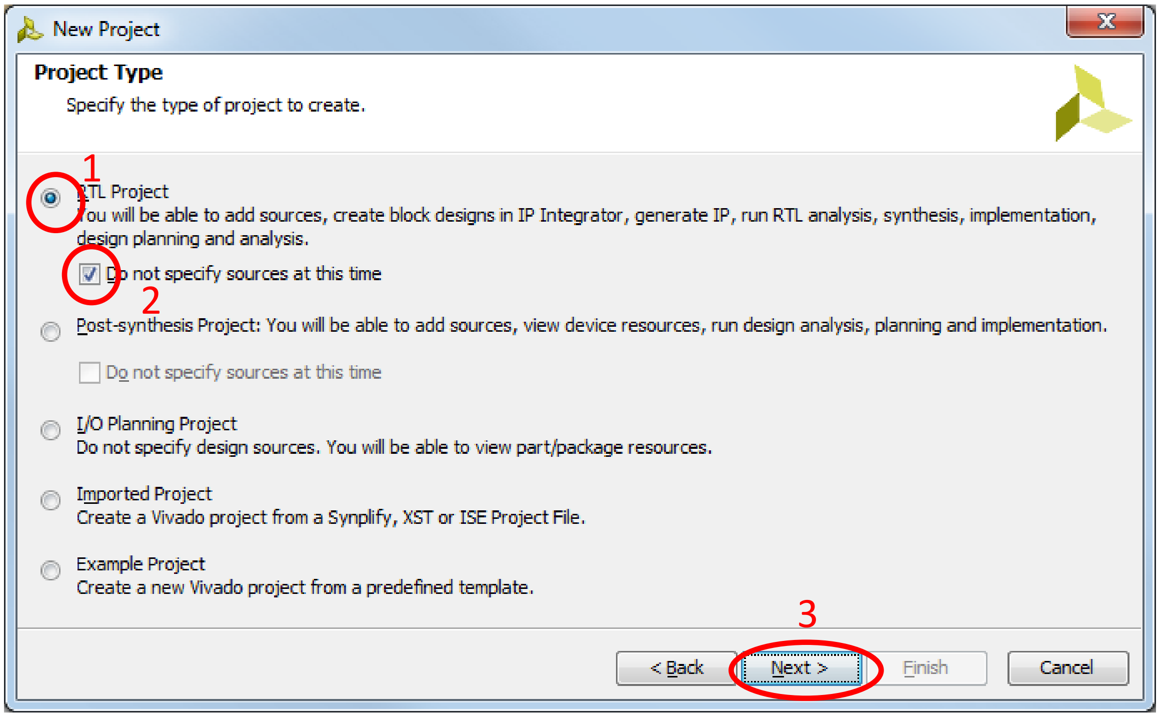
Project location: C:/Avnet/MicroZed/Projects

[✓] Create project subdirectory

Project will be created at: C:/Avnet/MicroZed/Projects/MZ_Basic_System

Figure 3 – Set Project Name and Location

7. The project will be RTL based, so leave the radio button for *RTL Project* selected. Since this is a brand new project, check the box for ***Do not specify sources at this time***. Click **Next >**.



New Project

Project Type

Specify the type of project to create.

[● RTL Project] You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.

[✓ Do not specify sources at this time]

Post-synthesis Project: You will be able to add sources, view device resources, run design analysis, planning and implementation.

Do not specify sources at this time

I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.

Imported Project
Create a Vivado project from a Synplify, XST or ISE Project File.

Example Project
Create a new Vivado project from a predefined template.

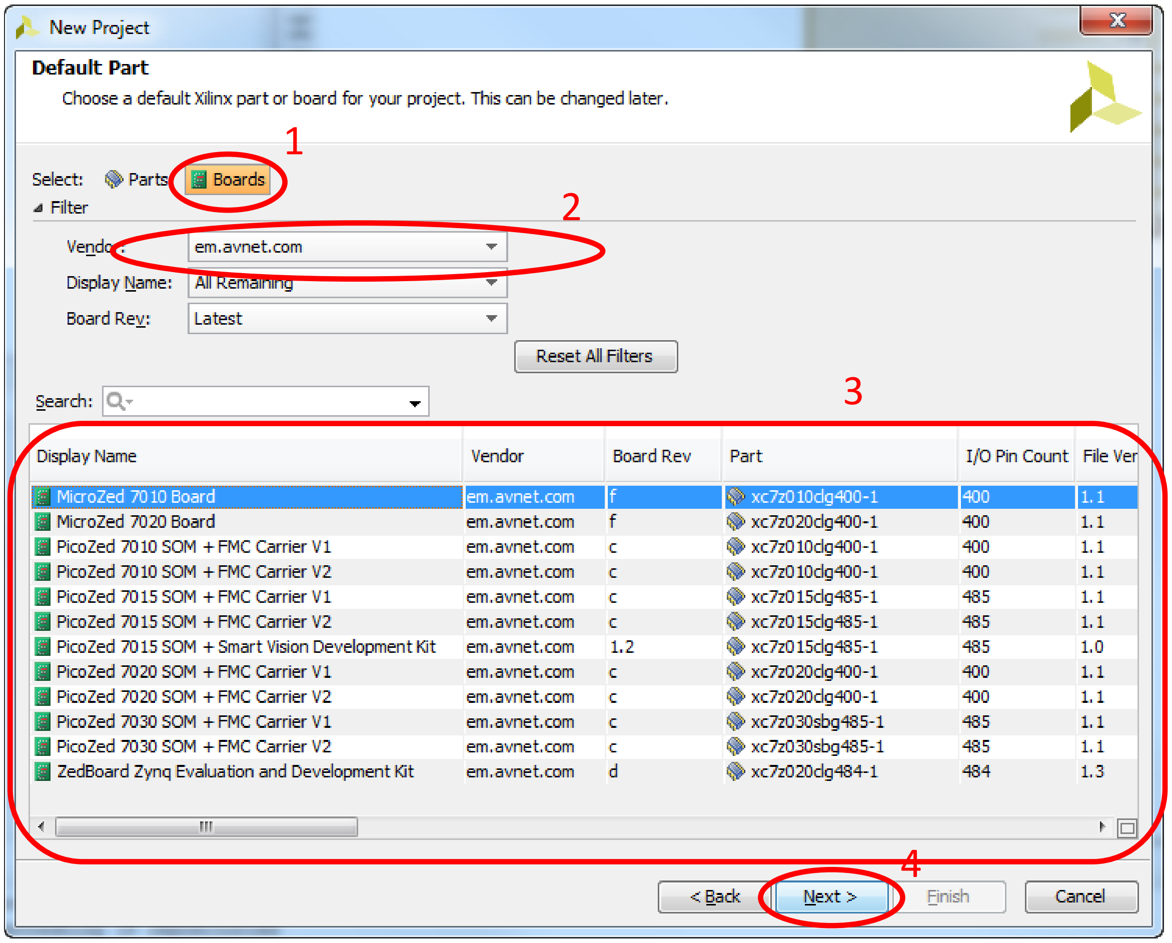
Figure 4 – Set Project Type

Next, the **Default Part** is selected. This can be done by specifying a specific part or by selecting a board. If you have installed the Board Definition archive correctly, you will have access to all MicroZed and PicoZed products.

8. In the *Select* area, select **Boards**.

9. Set the *Vendor* to **em.avnet.com**. This should leave only seven boards in the table.

10. Single-click the **MicroZed** or **PicoZed Board** that matches your configuration. Note that the MicroZed Rev F board may be selected for any revision B, C, F, or G. Click **Next >**.



Default Part

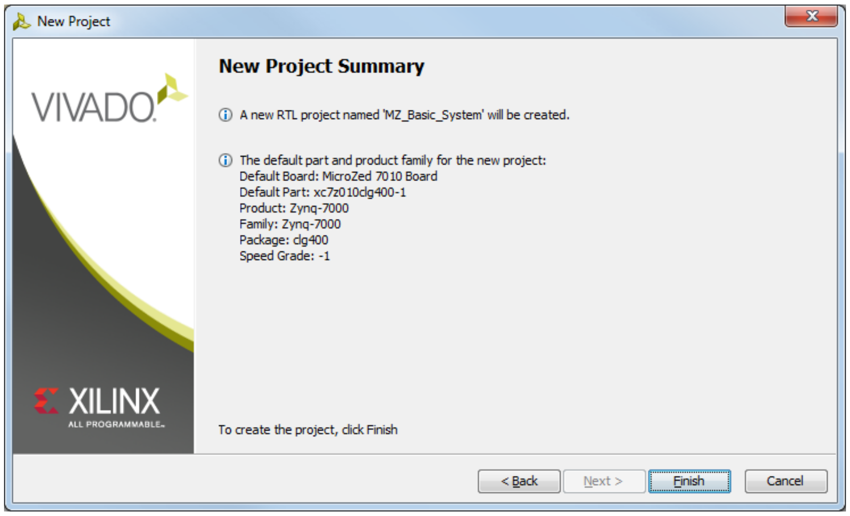
Choose a default Xilinx part or board for your project. This can be changed later.

Vendor: em.avnet.com
Display Name: All Remaining
Board Rev: Latest

Display Name	Vendor	Board Rev	Part	I/O Pin Count	File Ver
MicroZed 7010 Board	em.avnet.com	f	xc7z010clg400-1	400	1.1
MicroZed 7020 Board	em.avnet.com	f	xc7z020clg400-1	400	1.1
PicoZed 7010 SOM + FMC Carrier V1	em.avnet.com	c	xc7z010clg400-1	400	1.1
PicoZed 7010 SOM + FMC Carrier V2	em.avnet.com	c	xc7z010clg400-1	400	1.1
PicoZed 7015 SOM + FMC Carrier V1	em.avnet.com	c	xc7z015clg485-1	485	1.1
PicoZed 7015 SOM + FMC Carrier V2	em.avnet.com	c	xc7z015clg485-1	485	1.1
PicoZed 7015 SOM + Smart Vision Development Kit	em.avnet.com	1.2	xc7z015clg485-1	485	1.0
PicoZed 7020 SOM + FMC Carrier V1	em.avnet.com	c	xc7z020clg400-1	400	1.1
PicoZed 7020 SOM + FMC Carrier V2	em.avnet.com	c	xc7z020clg400-1	400	1.1
PicoZed 7030 SOM + FMC Carrier V1	em.avnet.com	c	xc7z030sbg485-1	485	1.1
PicoZed 7030 SOM + FMC Carrier V2	em.avnet.com	c	xc7z030sbg485-1	485	1.1
ZedBoard Zynq Evaluation and Development Kit	em.avnet.com	d	xc7z020clg484-1	484	1.3

Figure 5 – Select the Target Board

11. A project summary is displayed. Click **Finish**. The Vivado cockpit is now displayed.



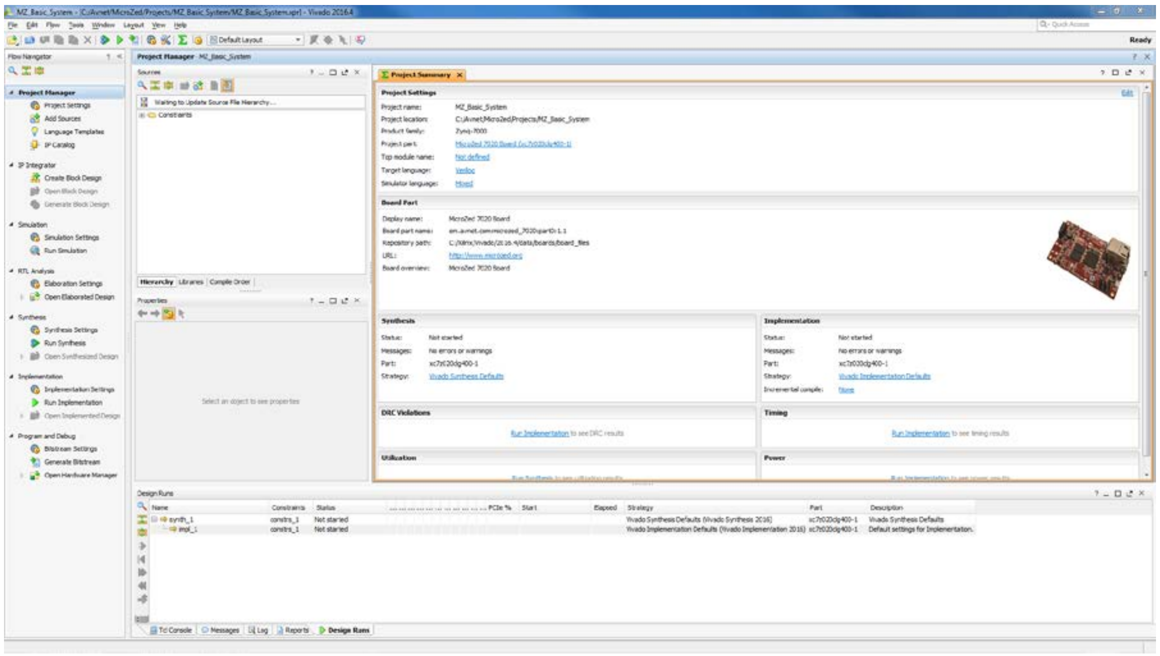
New Project Summary

A new RTL project named 'MZ_Basic_System' will be created.

The default part and product family for the new project:
Default Board: MicroZed 7010 Board
Default Part: xc7z010clg400-1
Product: Zynq-7000
Family: Zynq-7000
Package: clg400
Speed Grade: -1

To create the project, click Finish

Figure 6 – New Project Summary



Project Settings

Project name:	MZ_Basic_System
Project location:	C:/Avnet/MicroZed/Projects/MZ_Basic_System
Product family:	Zynq-7000
Project part:	MicroZed 7020 Board (xc7z020clg400-1)
Top module name:	Not defined
Target language:	Verilog
Simulator language:	Mixed

Board Part

Display name:	MicroZed 7020 Board
Board part name:	em.avnet.com:microzed_7020:part0:1.1
Repository path:	C:/Xilinx/Vivado/2016.4/data/boards/board_files
URL:	http://www.microzed.org
Board overview:	MicroZed 7020 Board

Synthesis

Status:	Not started
Messages:	No errors or warnings
Part:	xc7z020clg400-1
Strategy:	Vivado Synthesis Defaults

Implementation

Status:	Not started
Messages:	No errors or warnings
Part:	xc7z020clg400-1
Strategy:	Vivado Implementation Defaults
Incremental compile:	None

DRC Violations: Run Implementation to see DRC results

Timing: Run Implementation to see timing results

Design Runs

Name	Constraints	Status	PCIe %	Start	Elapsed	Strategy	Part	Description
synth_1	constrs_1	Not started				Vivado Synthesis Defaults (Vivado Synthesis 2016)	xc7z020clg400-1	Vivado Synthesis Defaults
impl_1	constrs_1	Not started				Vivado Implementation Defaults (Vivado Implementation 2016)	xc7z020clg400-1	Default settings for Implementation.

Figure 7 – Vivado Cockpit

Experiment 2: Create and Edit a Block Design

The current project is blank. To access the ARM processing system, we will add an embedded source to the Vivado project using IP Integrator.

1. The recommended way to add an embedded processor is through the Block Design method via IP Integrator. Select **Create Block Design**.

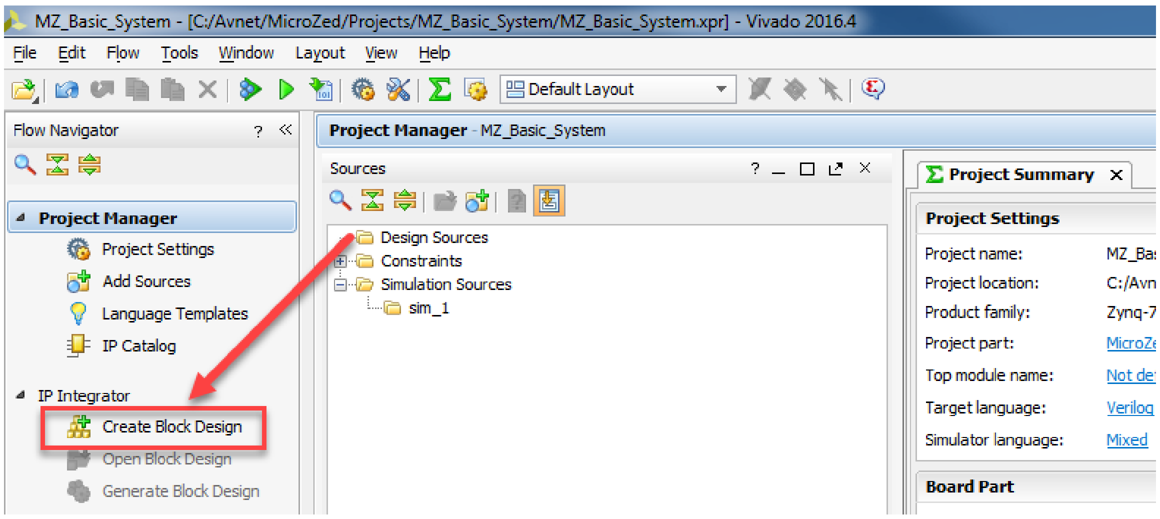
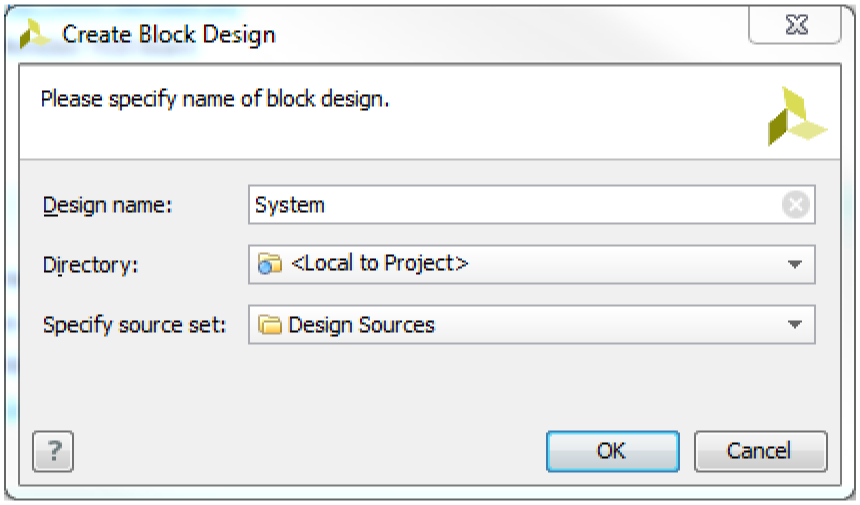


Figure 8 – Create Block Design

2. Give the Block Design a name. *System* is commonly used. Click **OK**.


Create Block Design

Please specify name of block design.

Design name: System

Directory: <Local to Project>

Specify source set: Design Sources

Figure 9 – Block Design Name

3. In the Diagram window, click the **Add IP** text icon in either location.

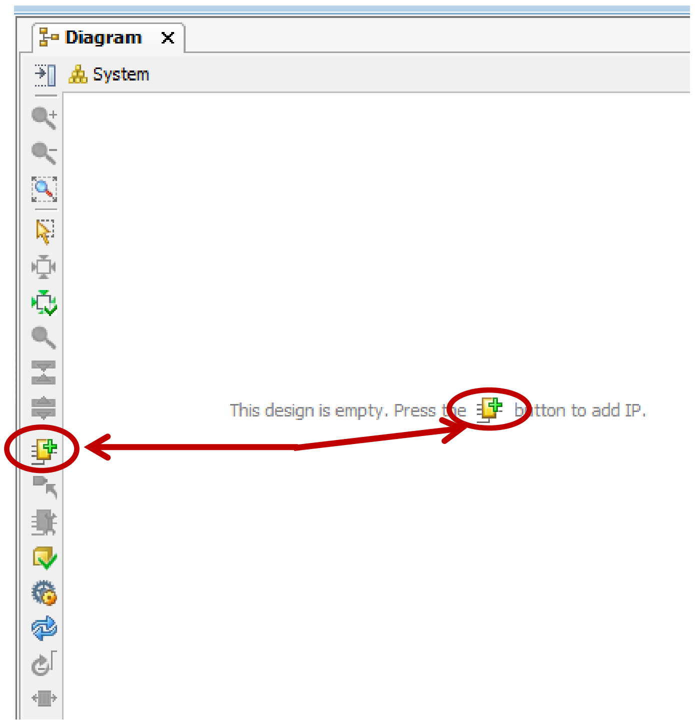


Figure 10 – Add IP to the Block Design

4. The *Add Sources* window opens. Start typing “Zynq” in the search window. Find the **ZYNQ7 Processing System** IP. Either double-click this or drag and drop to the *Diagram* window.

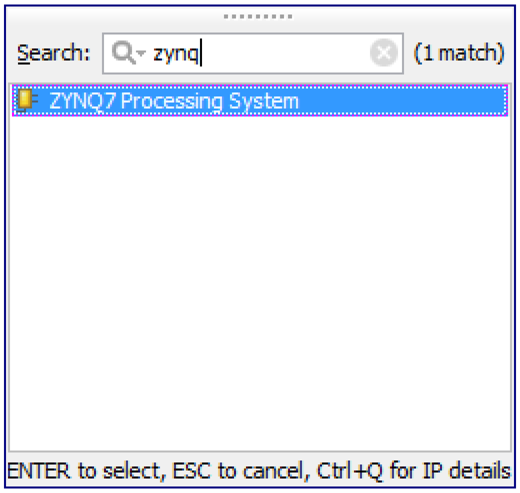


Figure 11 – Add IP Window

The Zynq Processing system will appear in the *Diagram* window. Also a new tab will appear labeled **Address Editor**.

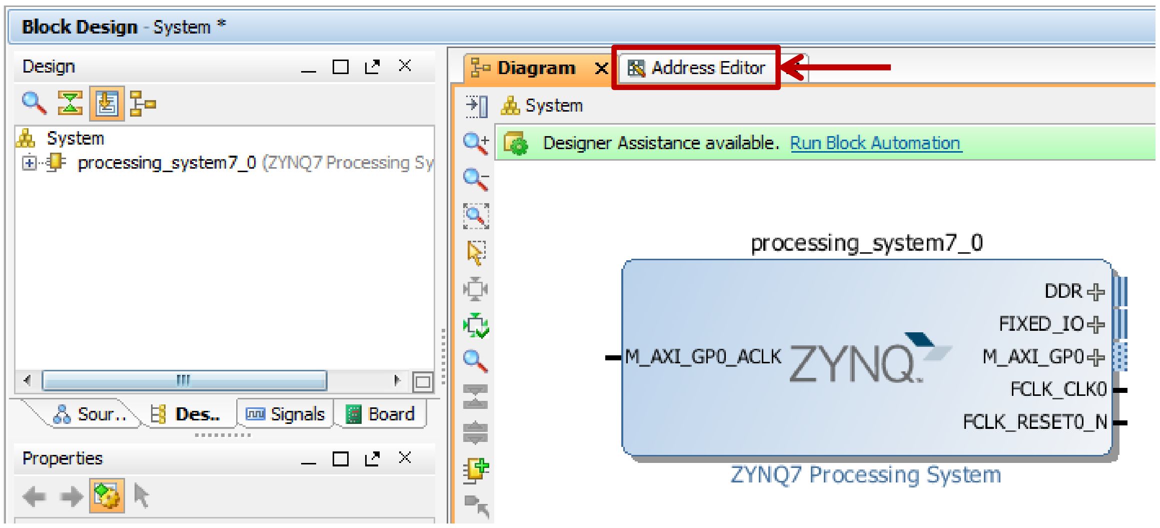


Figure 12 – Updated Block Diagram

5. Similar to the *Add IP* prompt in the previous step, notice now that the *Designer Assistance* has provided the hint to *Run Block Automation*. Click the **Run Block Automation** link at the top of the window.

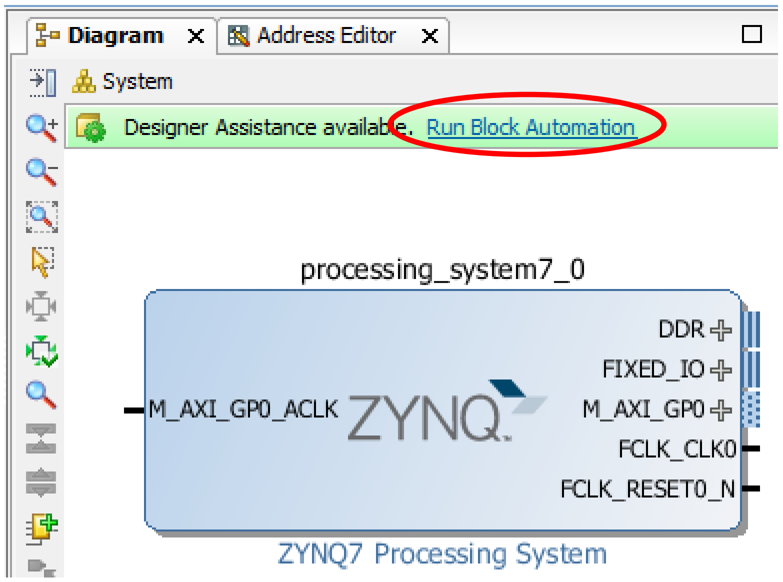


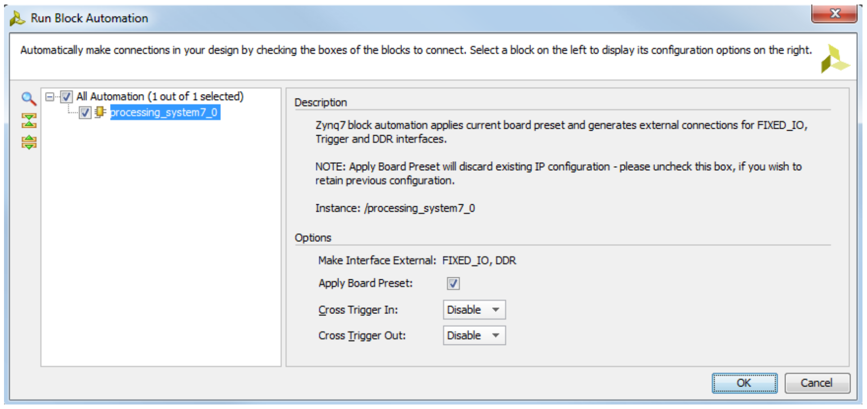
Figure 13 - Run Block Automation

6. Notice the block automation wizard has identified two sources of I/O that need to be made external. One is obvious, the **DDR** interface. The other is labeled **FIXED_IO**. FIXED_IO is basically the MIO pin connections. They are labeled FIXED_IO because you cannot change their assignments in this window.

   The *Apply Board Preset* checkbox applies the Preset TCL that was included as part of the board definition archive. Leave this checked. For details about how to build a system manually, please see the *Avnet Zynq Hardware Development Speedway*.

   The Cross Trigger options may be left Disabled.

   Click **OK** to connect these external signals.



Run Block Automation

Automatically make connections in your design by checking the boxes of the blocks to connect. Select a block on the left to display its configuration options on the right.

All Automation (1 out of 1 selected)
processing_system7_0

Description

Zynq7 block automation applies current board preset and generates external connections for FIXED_IO, Trigger and DDR interfaces.

NOTE: Apply Board Preset will discard existing IP configuration - please uncheck this box, if you wish to retain previous configuration.

Instance: /processing_system7_0

Options

Make Interface External: FIXED_IO, DDR
Apply Board Preset: ✓
Cross Trigger In: Disable
Cross Trigger Out: Disable

Figure 14 – Run Block Automation

7. You will now see the Zynq block with external I/O.

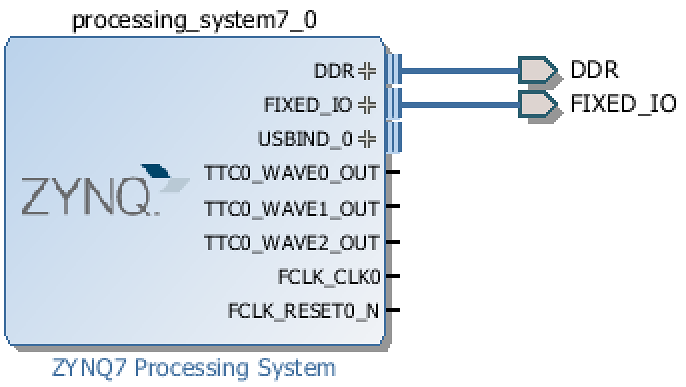


Figure 15 - Zynq Block Diagram for MicroZed and PZ7010/FMC-V1 with External I/O

The block for all of the PicoZeds (with the exception of the 7010 with the FMC-V1 Carrier) look a little bit different since those board definitions enable PL GPIO peripherals. You will notice the addition of the enabled M_AXI_GP0 port and accompanying clock M_AXI_GP0_ACLK.

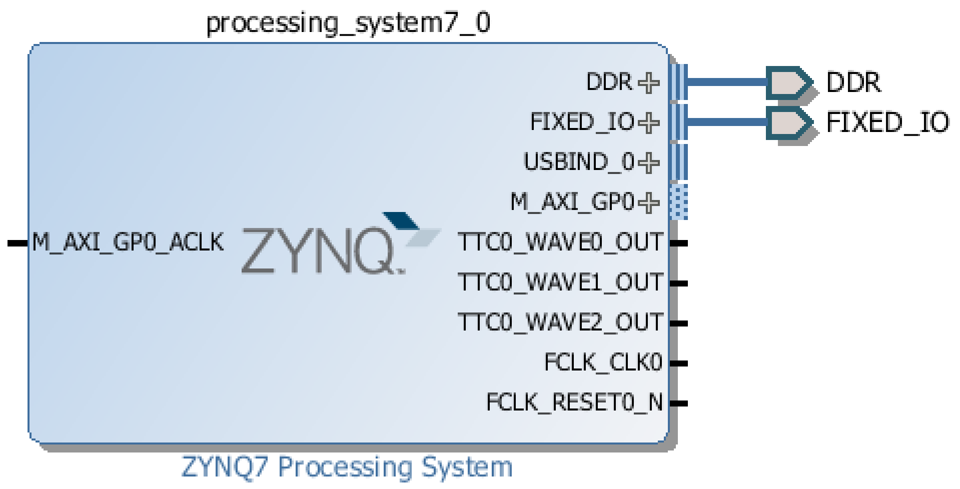
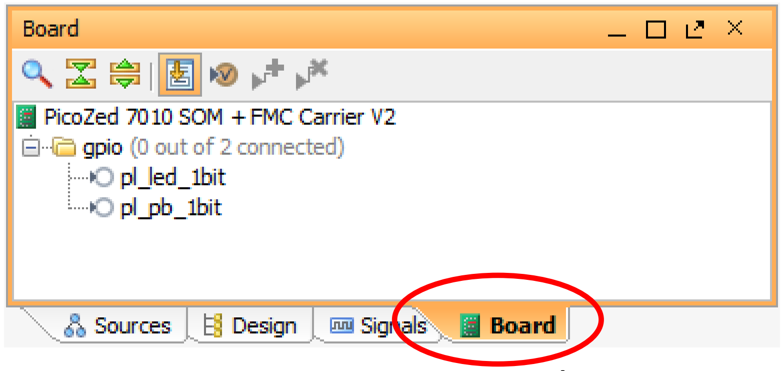


Figure 16 – Zynq Block Diagram for All PicoZeds Except PZ7010/FMC-V1

For MicroZed or PZ7010/FMC-V1 designs, skip to **Experiment 2, Step 15** on Page 18.

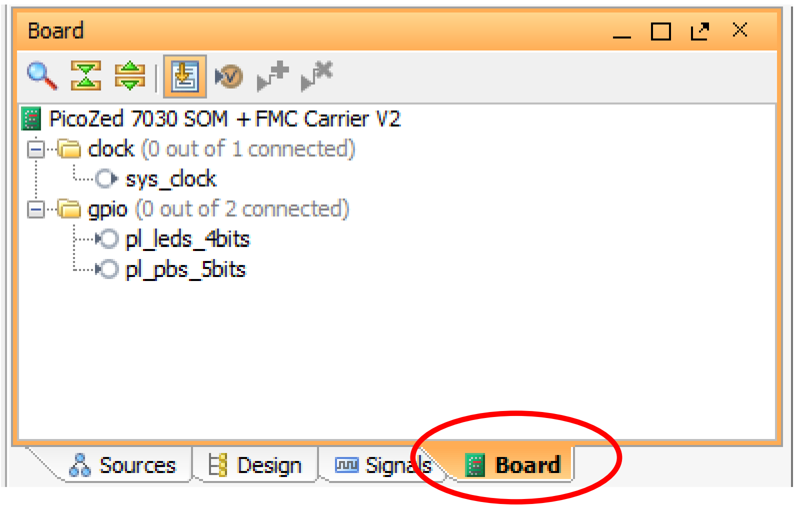
For all other PicoZed designs, continue with the next steps to connect the PL GPIO.

8. Click the Board tab. It will looks similar to one of the images below, depending on the number of LEDs and push buttons (PBs) connected.



PicoZed 7010 SOM + FMC Carrier V2
gpio (0 out of 2 connected)
pl_led_1bit
pl_pb_1bit

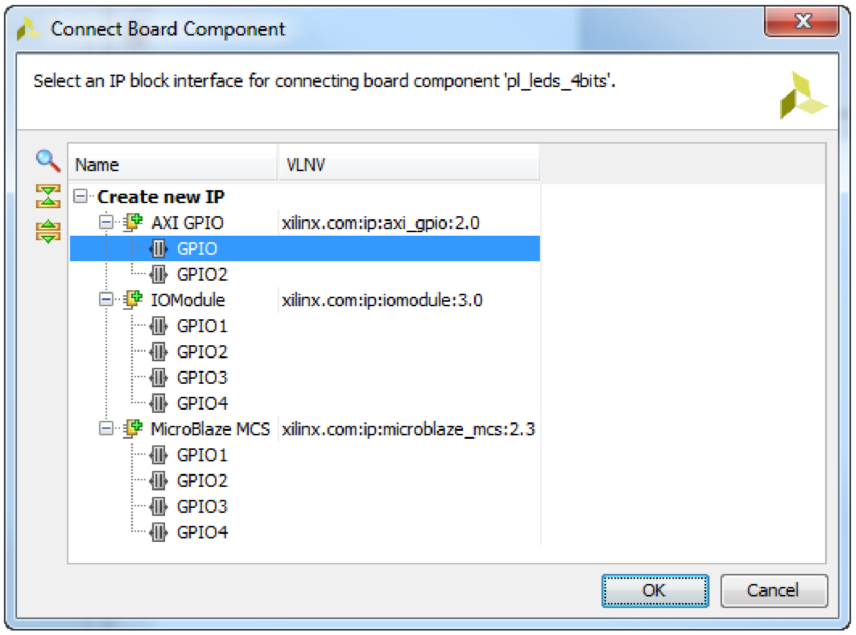
Figure 17 – Board Tab for PZ7010/FMC-V2



PicoZed 7030 SOM + FMC Carrier V2
clock (0 out of 1 connected)
sys_clock
gpio (0 out of 2 connected)
pl_leds_4bits
pl_pbs_5bits

Figure 18 – Board Tab for PZ7030/FMC-V2

9. Double-click pl_leds_Xbits. Select GPIO under Create new IP. Click **OK**.


Connect Board Component

Select an IP block interface for connecting board component 'pl_leds_4bits'.

Name	VLNV
Create new IP	
AXI GPIO	xilinx.com:ip:axi_gpio:2.0
GPIO	
GPIO2	
IOModule	xilinx.com:ip:iomodule:3.0
GPIO1	
GPIO2	
GPIO3	
GPIO4	
MicroBlaze MCS	xilinx.com:ip:microblaze_mcs:2.3
GPIO1	
GPIO2	
GPIO3	
GPIO4	

Figure 19 – Connect the PL LEDs to an AXI GPIO Peripheral

10. Ignore the *Run Connection Automation* for now. Click to optimize the routing.

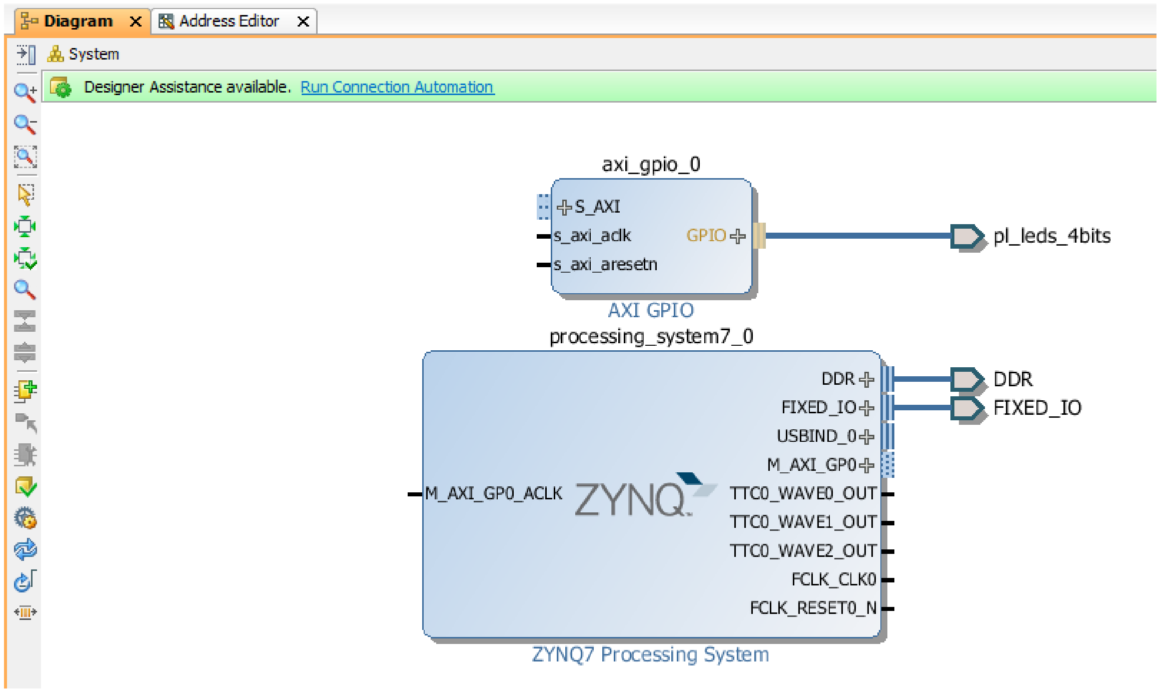
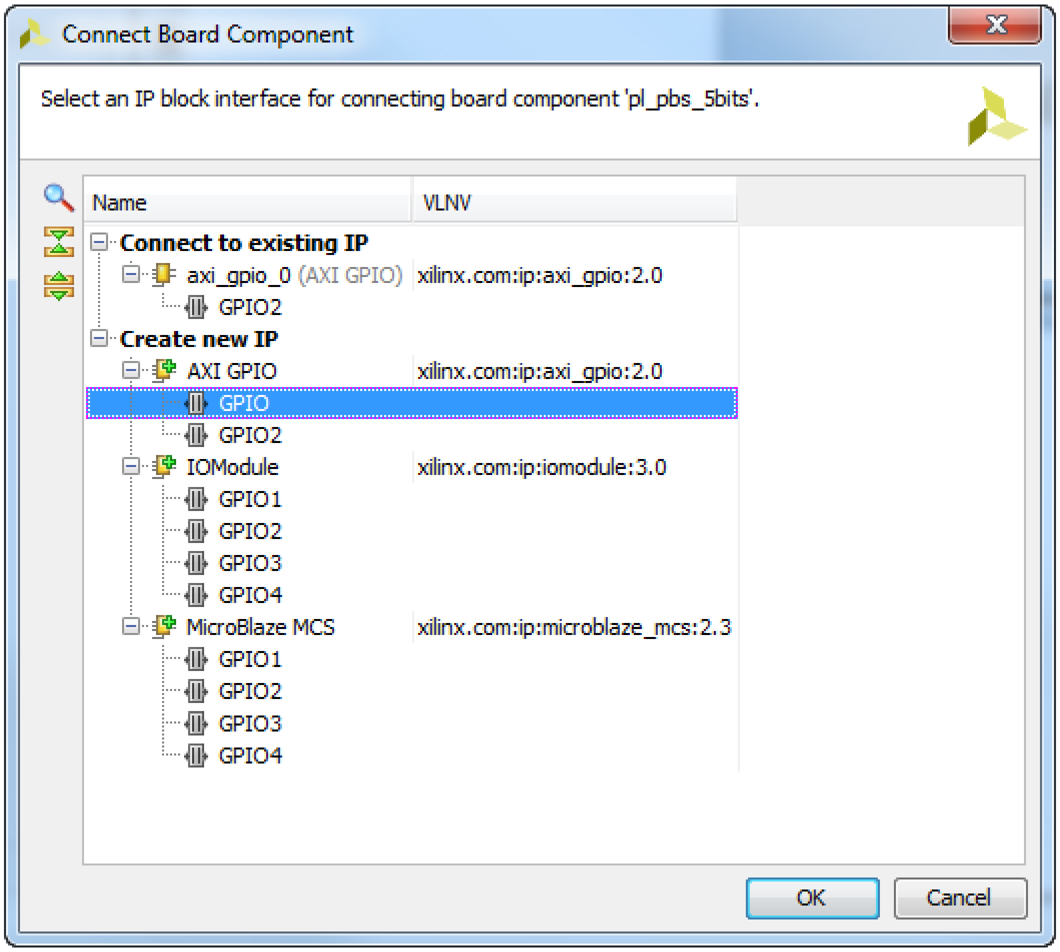


Figure 20 – PL LED AXI GPIO Peripheral Added to Block Design

11. Double-click pl_pbs_Xbits. Select GPIO under **Create new IP**. Click **OK**.



Connect Board Component

Select an IP block interface for connecting board component 'pl_pbs_5bits'.

Name	VLNV
Connect to existing IP	
axi_gpio_0 (AXI GPIO)	xilinx.com:ip:axi_gpio:2.0
GPIO2	
Create new IP	
AXI GPIO	xilinx.com:ip:axi_gpio:2.0
GPIO	
GPIO2	
IOModule	xilinx.com:ip:iomodule:3.0
GPIO1	
GPIO2	
GPIO3	
GPIO4	
MicroBlaze MCS	xilinx.com:ip:microblaze_mcs:2.3
GPIO1	
GPIO2	
GPIO3	
GPIO4	

Figure 21 – Connect the PL PBs to an AXI GPIO Peripheral

12. Optimize the routing again, and you should see a similar image as below.

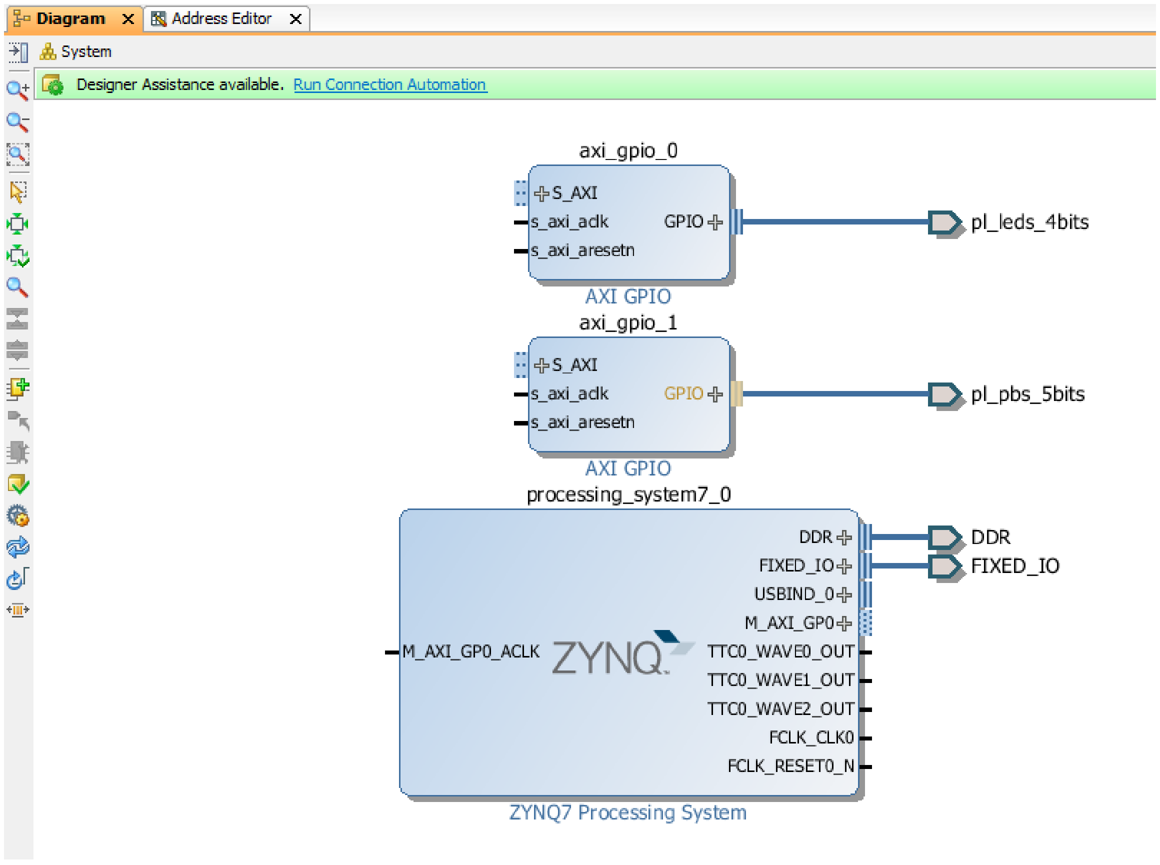
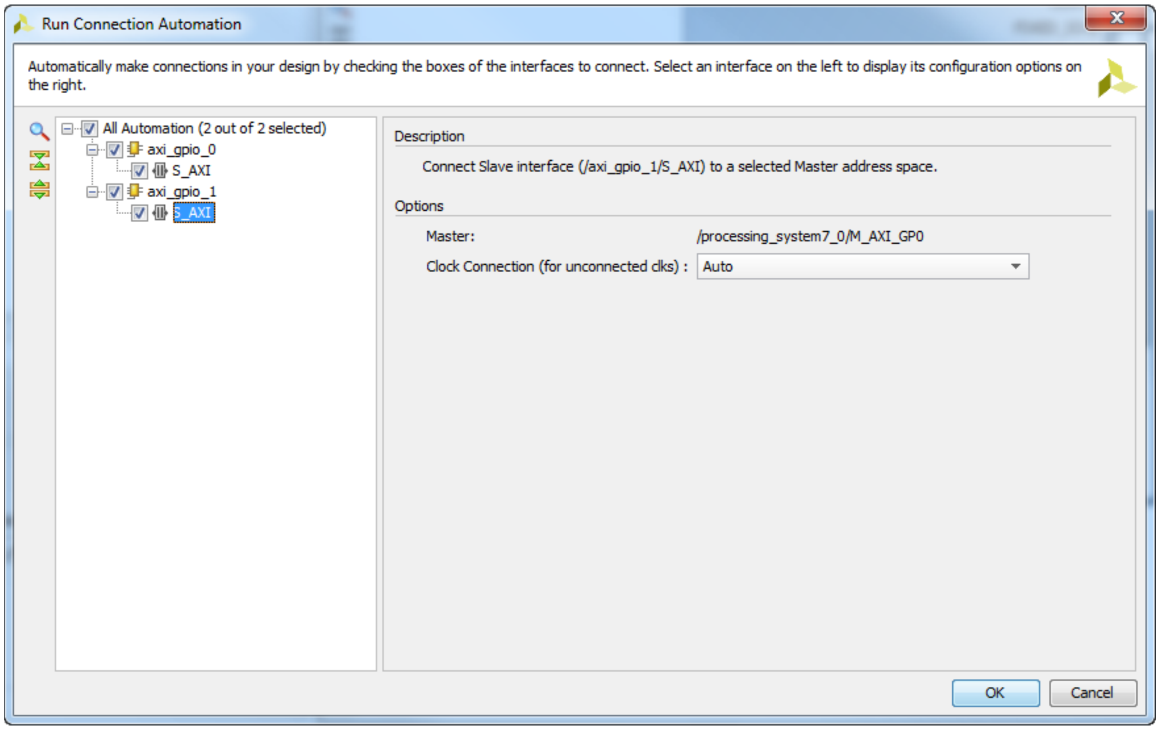


Figure 22 – Two AXI GPIO Peripherals Added to Block Design

13. Click on **Run Connection Automation then** select **S_AXI** for both axi_gpio_0 and axi_gpio_1. Click **OK**.


Run Connection Automation

Automatically make connections in your design by checking the boxes of the interfaces to connect. Select an interface on the left to display its configuration options on the right.

- [✓] All Automation (2 out of 2 selected)
  - [✓] axi_gpio_0
    - [✓] S_AXI
  - [✓] axi_gpio_1
    - [✓] S_AXI

Description

Connect Slave interface (/axi_gpio_1/S_AXI) to a selected Master address space.

Options

Master: /processing_system7_0/M_AXI_GP0

Clock Connection (for unconnected clks) : Auto

Figure 23 – Connect AXI GPIO Peripherals to AXI Slave Interconnect

14. Regenerate the layout by clicking .

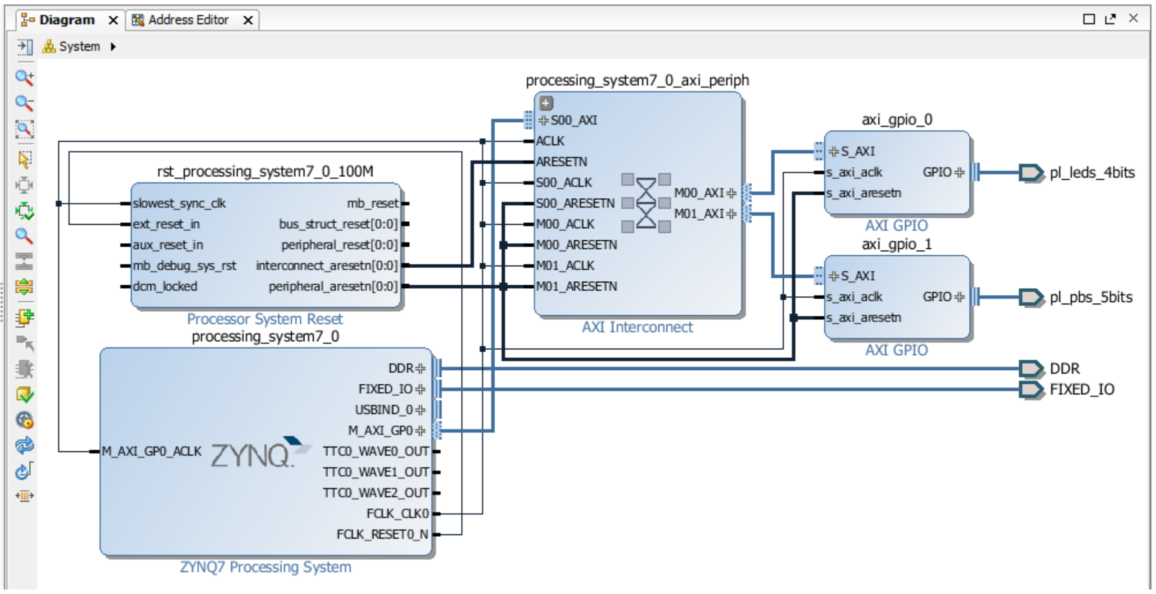


Figure 24 – Regenerated Layout with Two AXI GPIOs with Interconnect and RST Blocks

15. At this point, we can **validate** our design. Click the Validate Design icon ✓. A successful validation window will appear. Click **OK**.

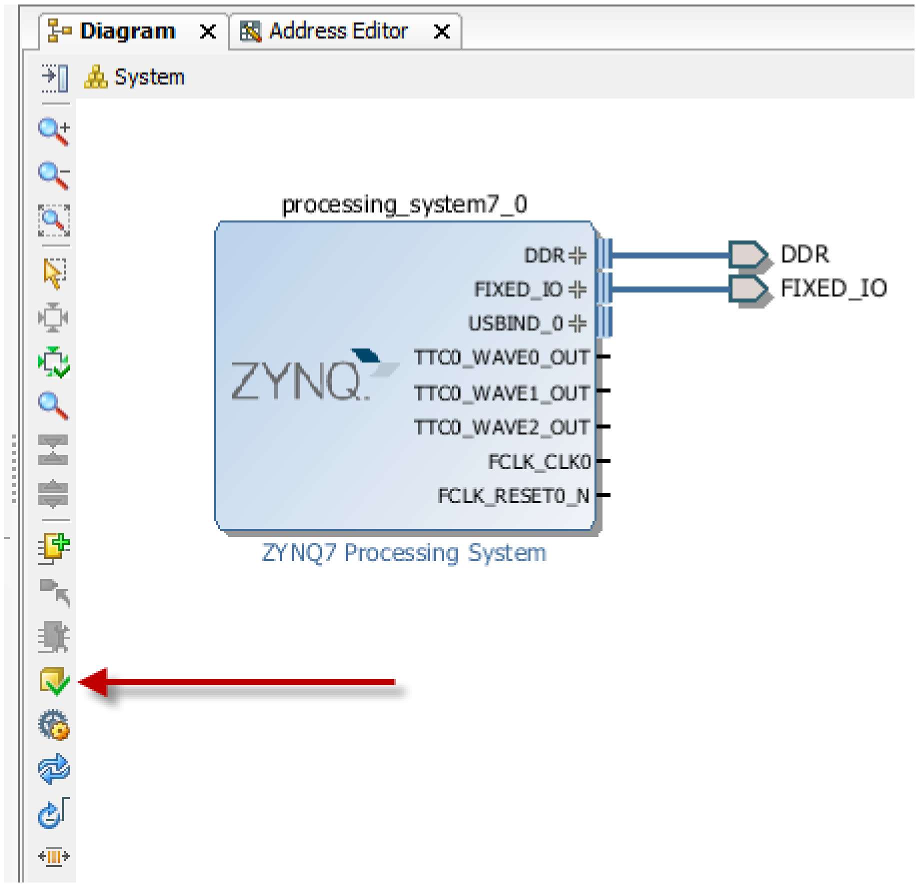
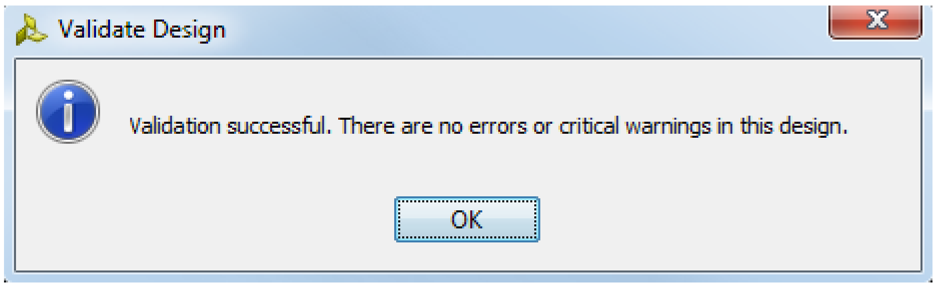


Figure 25 - Validate Zynq Block Design


Validate Design

Validation successful. There are no errors or critical warnings in this design.

Figure 26 – Validation Successful

16. Click **Save Block Design** icon, , to save the project.

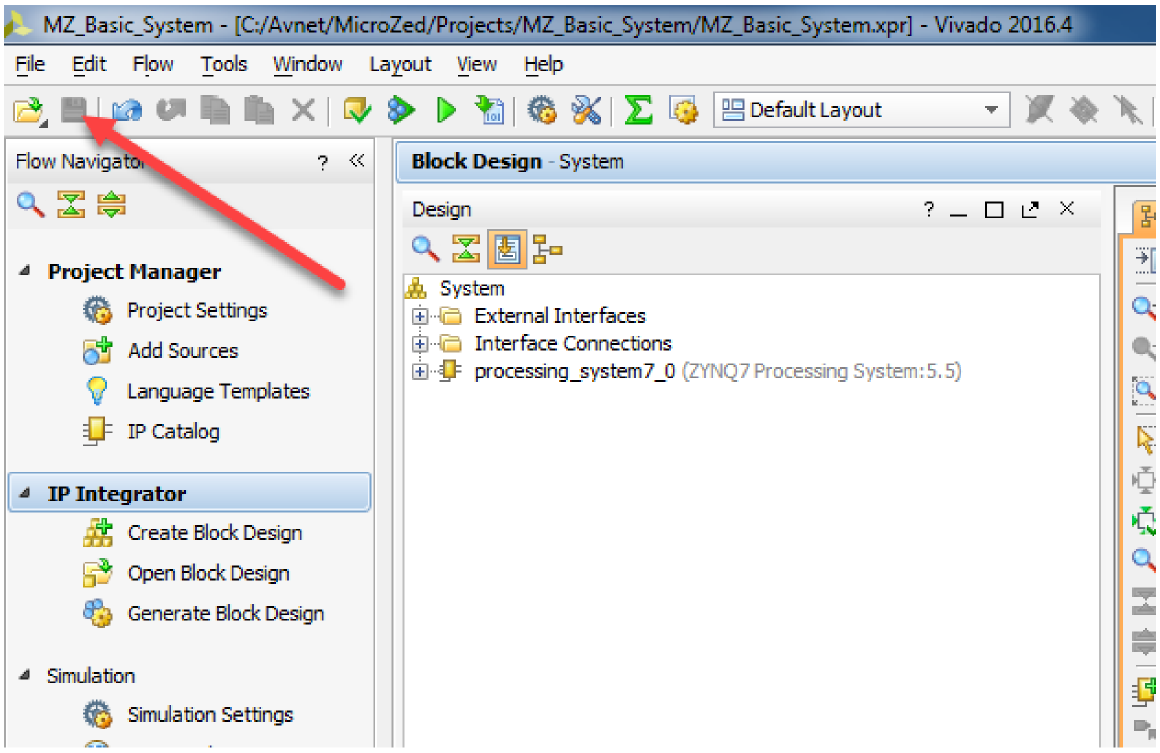


Figure 27 - Save Block Design

17. Switch to the **Sources** tab by clicking on it.

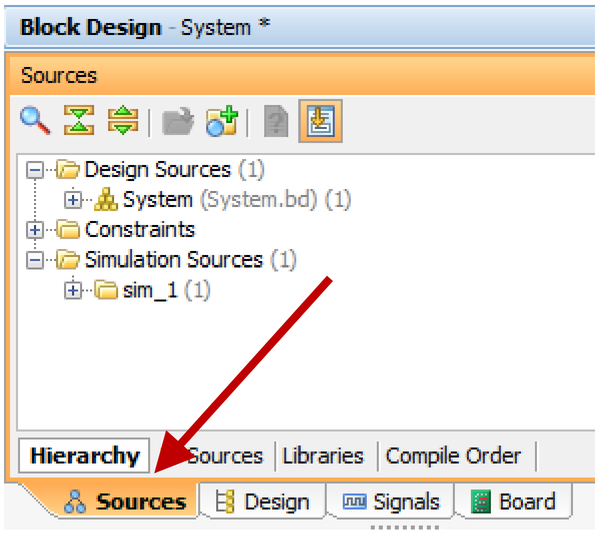


Figure 28 – Sources Tab

18. Right-click on **System (System.bd)** and select **Create HDL wrapper**.

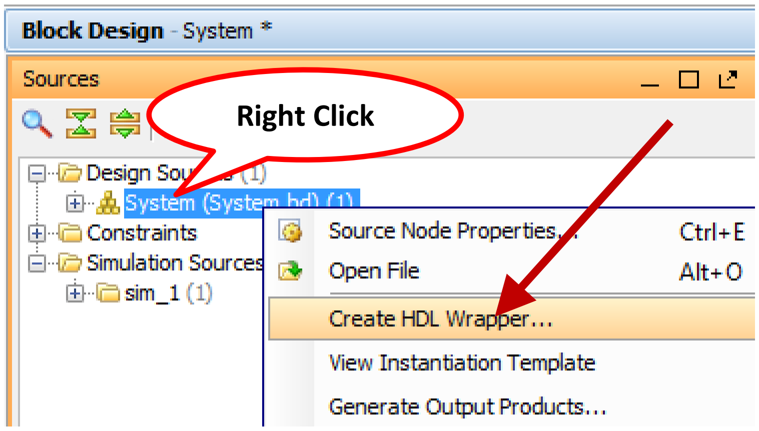
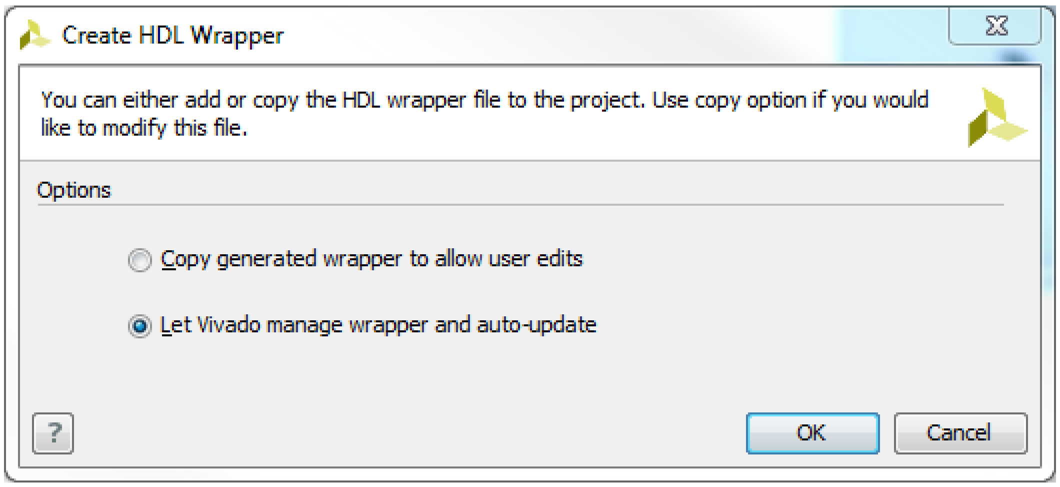


Figure 29 - Create Top Level HDL Wrapper

19. For now, leave the option selected to *Let Vivado manage wrapper and auto-update*. Click **OK**.


Create HDL Wrapper

You can either add or copy the HDL wrapper file to the project. Use copy option if you would like to modify this file.

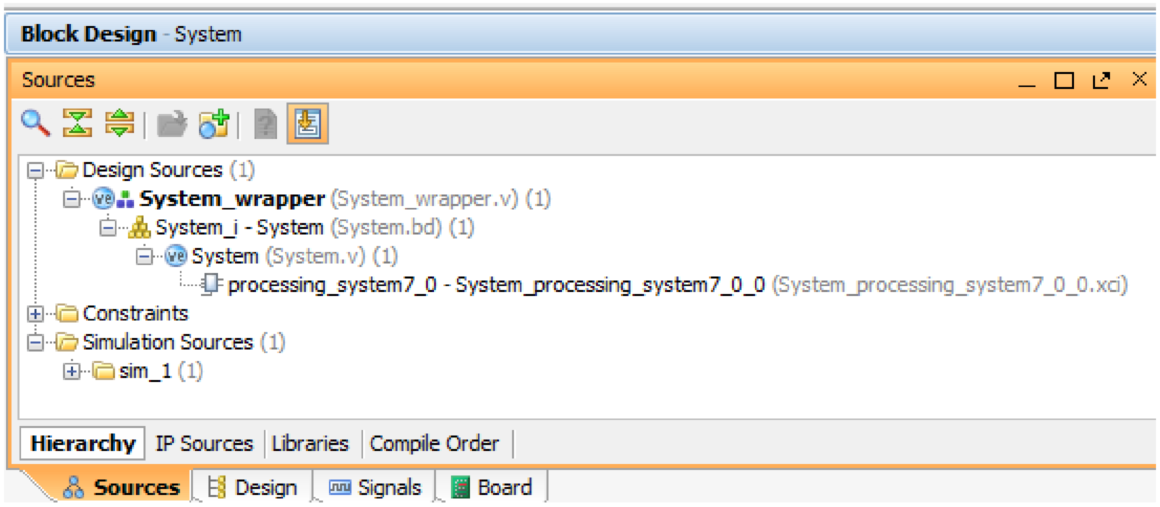
Options

( ) Copy generated wrapper to allow user edits

[✓ Let Vivado manage wrapper and auto-update]

Figure 30 – Let Vivado Manage Wrapper

20. Once the top-level wrapper is created, you can see the design hierarchy in the *Sources* tab. Notice that **System_wrapper.v** is the top-level HDL wrapper that was created. **System.bd** is the Block Design.



Design Sources (1)
System_wrapper (System_wrapper.v) (1)
System_i - System (System.bd) (1)
System (System.v) (1)
processing_system7_0 - System_processing_system7_0_0 (System_processing_system7_0_0.xci)
Constraints
Simulation Sources (1)
sim_1 (1)

Figure 31 – System_wrapper.v Generated

21. Click **Generate Bitstream** in the *Flow Navigator* window.

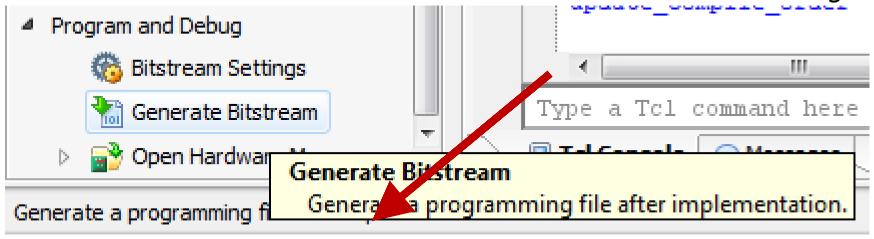
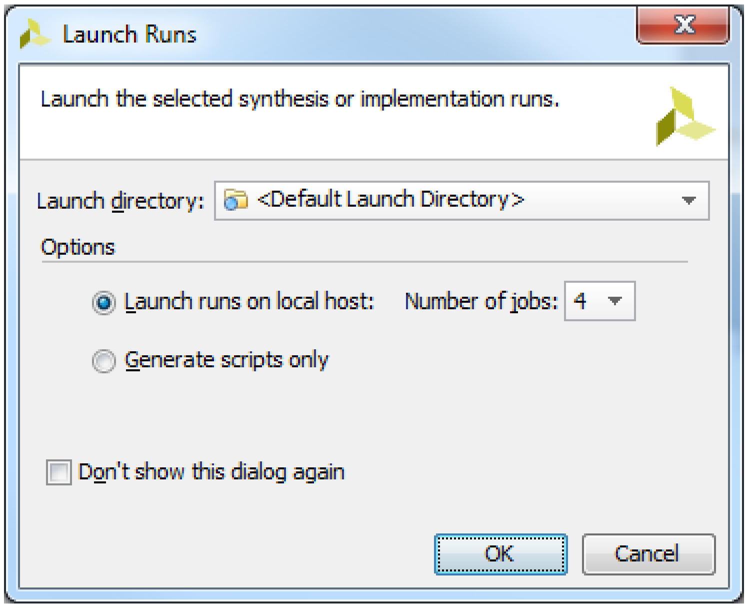


Figure 32 – Generate Bitstream

22. Launch runs window will open **Select** Ok for the default configurations. Click **Yes** to start Synthesis and Implementation flows. *Check the upper right-hand corner of the tool for a status bar.*



Launch Runs

Launch the selected synthesis or implementation runs.

Launch directory: <Default Launch Directory>

Options

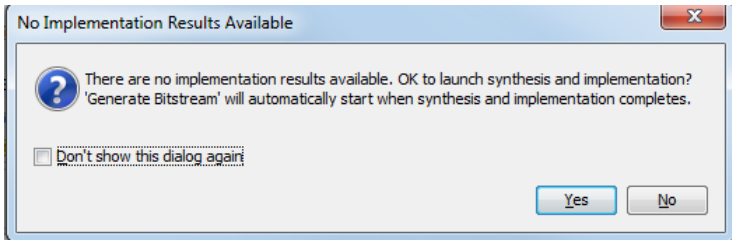
Launch runs on local host: Number of jobs: 4

Generate scripts only

Don't show this dialog again

OK Cancel

Figure 33 – Launch Runs Configurations



No Implementation Results Available

There are no implementation results available. OK to launch synthesis and implementation?
'Generate Bitstream' will automatically start when synthesis and implementation completes.

Don't show this dialog again

Yes No

Figure 34 – Launch Synthesis and Implementation

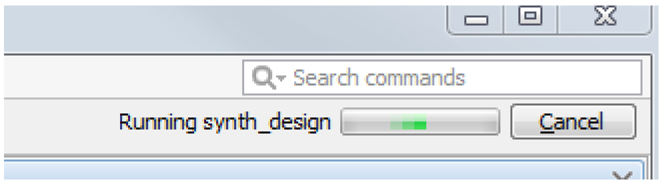
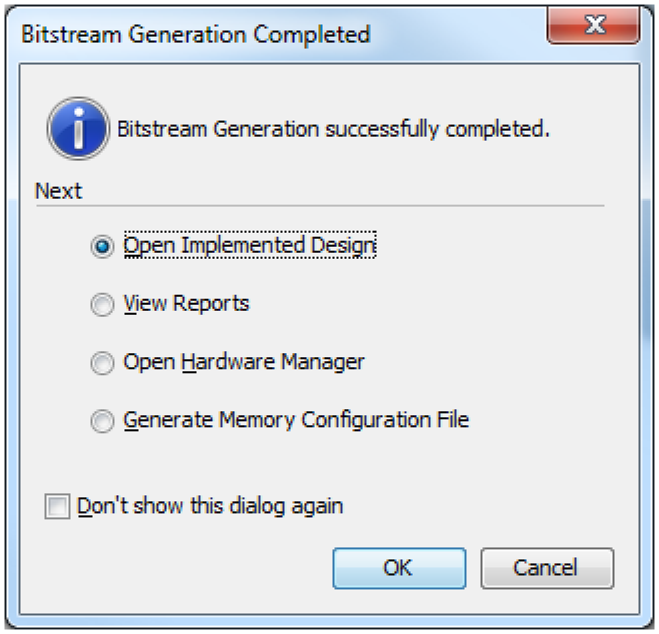


Figure 345 – Progress Status Bar

23. When bitstream generation is completed, click **OK** to *Open Implemented Design*.


Bitstream Generation Completed

Bitstream Generation successfully completed.

Next

(●) Open Implemented Design
( ) View Reports
( ) Open Hardware Manager
( ) Generate Memory Configuration File

☐ Don't show this dialog again

Figure 356 – Open Implemented Design

Experiment 3: Export Hardware Platform to SDK

Now that we've created an embedded system, we must make this platform available to the Software Development Kit (SDK). This is done by exporting the hardware platform.

1. In the Vivado tool, select **File → Export → Export Hardware**. Check the box to ***Include bitstream***. Click **OK**. You could specify a different directory, but for now, leave it as *Local to Project*.

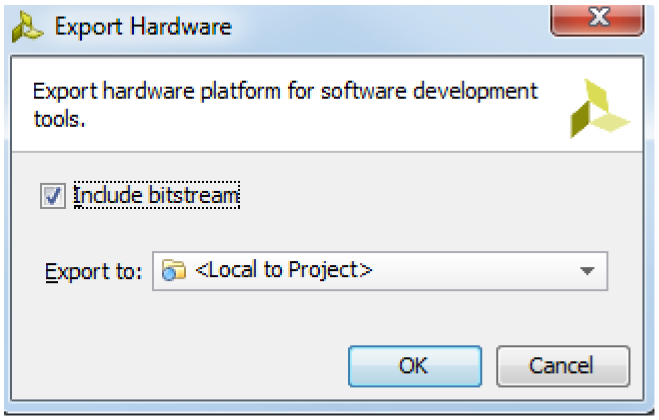
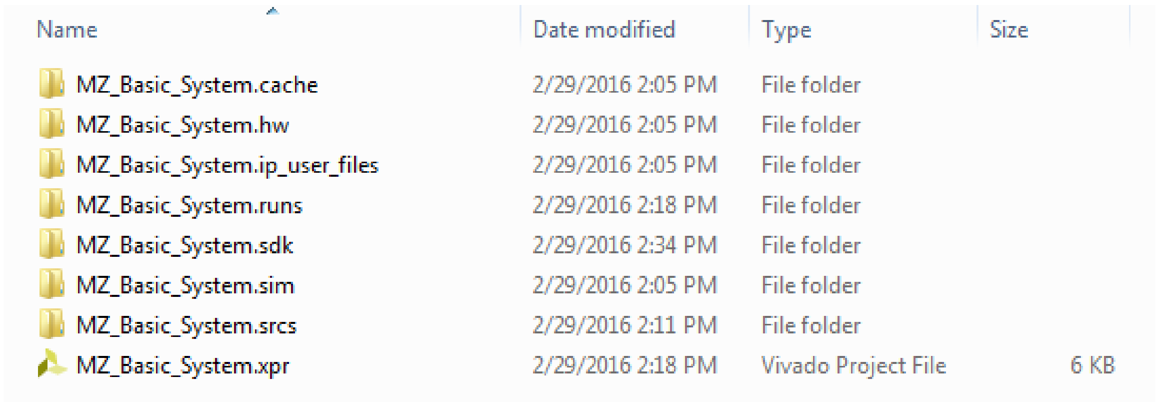


Figure 367 – Export Zynq Hardware Platform

2. We will now explore what you have created. In Windows Explorer, browse to your project directory.



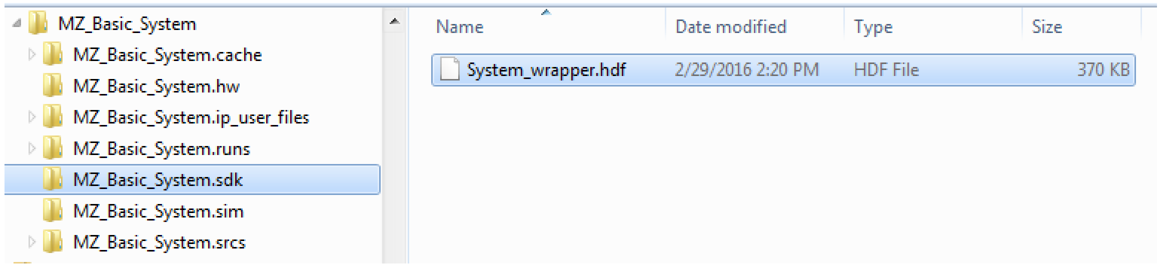
Name	Date modified	Type	Size
MZ_Basic_System.cache	2/29/2016 2:05 PM	File folder	
MZ_Basic_System.hw	2/29/2016 2:05 PM	File folder	
MZ_Basic_System.ip_user_files	2/29/2016 2:05 PM	File folder	
MZ_Basic_System.runs	2/29/2016 2:18 PM	File folder	
MZ_Basic_System.sdk	2/29/2016 2:34 PM	File folder	
MZ_Basic_System.sim	2/29/2016 2:05 PM	File folder	
MZ_Basic_System.srcs	2/29/2016 2:11 PM	File folder	
MZ_Basic_System.xpr	2/29/2016 2:18 PM	Vivado Project File	6 KB

Figure 378 – Project Directory Contents after Export to SDK

You will notice six directories and one file here. The .xpr file is your Vivado Project File and can be used to re-launch your project when you come back to work on it some more.

The .cache, .hw, .runs, .sim, and .srcs directories contain everything related to the hardware design, including the block design source, wrapper HDL, and synthesis/implementation results.

The .sdk folder is the result of the ***Export Hardware*** operation. Everything required for SDK to import the hardware platform is contained inside one file inside this directory. A hardware engineer looking to share the design with the software team could provide this one file. This provides a very compact and portable method to send a Zynq Hardware Platform to a colleague.



Name	Date modified	Type	Size
System_wrapper.hdf	2/29/2016 2:20 PM	HDF File	370 KB

Figure 389 – Zynq Hardware Platform Export for SDK

The next tutorial will show you how to open SDK, import a hardware platform, and run Hello World.

Revision History

Date	Version	Revision
23 Aug 2013	2013_2.01	Initial Avnet release for Vivado 2013.2
02 Jun 2014	2014_1.01	Update for 2014.1 using Avnet MicroZed board definition archive.
11 Jun 2014	2014_2.01	Update for 2014.2. Export hardware now produces HDF file.
29 Jun 2015	2015_1.01	Update for 2015.1. Added support for MicroZed 7020 and PicoZed 7010/15/20/30.
15 Jul 2015	2015_2.01	Update for 2015.2
06 Apr 2016	2015_4.01	Update for 2015.4. Extra connection for eMMC no longer necessary. Added instruction for new PicoZed board definitions to connect the AXI GPIO.
01 Jun 2016	2015_4.02	Update for 2015.4. Added instruction clarification for Run Automation.
15 Sept 2016	2016_2.01	Updated for 2016.2. Minor figure changes.
20 Jan 2017	2016_4.01	Updated to 2016.4.