



A Practical Guide to Getting Started with Xilinx SDSoC

| | |
|-------------------|------------------|
| Tools: | 2018.3 |
| Training Version: | v4 |
| Date: | 27 February 2019 |

© 2019 Avnet. All rights reserved. All trademarks and registered trademarks are the property of their respective owners. All specifications are subject to change without notice.

NOTICE OF DISCLAIMER: Avnet is providing this design, code, or information "as is." By providing the design, code, or information as one possible implementation of this feature, application, or standard, Avnet makes no representation that this implementation is free from any claims of infringement. You are responsible for obtaining any rights you may require for your implementation. Avnet expressly disclaims any warranty whatsoever with respect to the adequacy of the implementation, including but not limited to any warranties or representations that this implementation is free from claims of infringement and any implied warranties of merchantability or fitness for a particular purpose.

Introduction

Using the instructions contained herein, you shall learn how to install custom platforms as well as what an output of SDSoC is using the provided example project. Using this base knowledge, you will be able to better understand the needs of SDSoC while leveraging this platform for your own projects. There are some places where we will accept some acceleration (use of Board Presets), however there is no reason that you would be required to do such. For instance, if you were generating the SDSoC platform for a custom board.

Designed by Avnet

MiniZed™ is a Zynq® 7Z007S single-core development board. With the advent of the latest cost-optimized portfolio from Xilinx, this board targets entry-level Zynq developers with a low-cost prototyping platform.

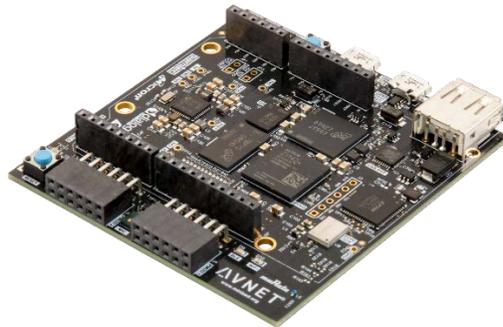


Figure 1 - MiniZed

Ultra96 is the first 96boards development board with 64bit ARM and programmable logic. Using the Zynq UltraScale+™ MPSoC XCZU3EG multi-core SoC with accelerators, this makes a perfect platform for starting out with highly complex SDSoC applications. This board targets entry-level Zynq UltraScale+ developers with a low-cost 96boards compatible prototyping platform.



Figure 2 – Ultra96

UltraZed-EG SOM is a highly integrated System-on-Module (SOM) based on the powerful Xilinx Zynq UltraScale+ MPSoC family of devices. Designed in a small form factor, the UltraZed-EG SOM packages all the necessary functions such as system memory, Ethernet, USB, and configuration memory needed for an embedded processing system. While this SOM shares the same family as the Ultra96v1 and Ultra96v2, the Xilinx XCZU3EG-1SFVA625 device provides many optimizations and breakout capabilities that are not available on the much smaller package contained in the Ultra96.

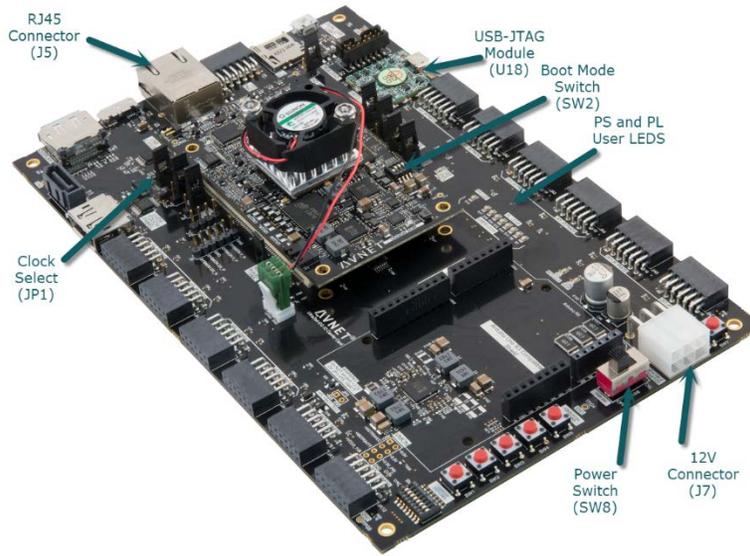


Figure 3 – UltraZed-EG Starter Kit

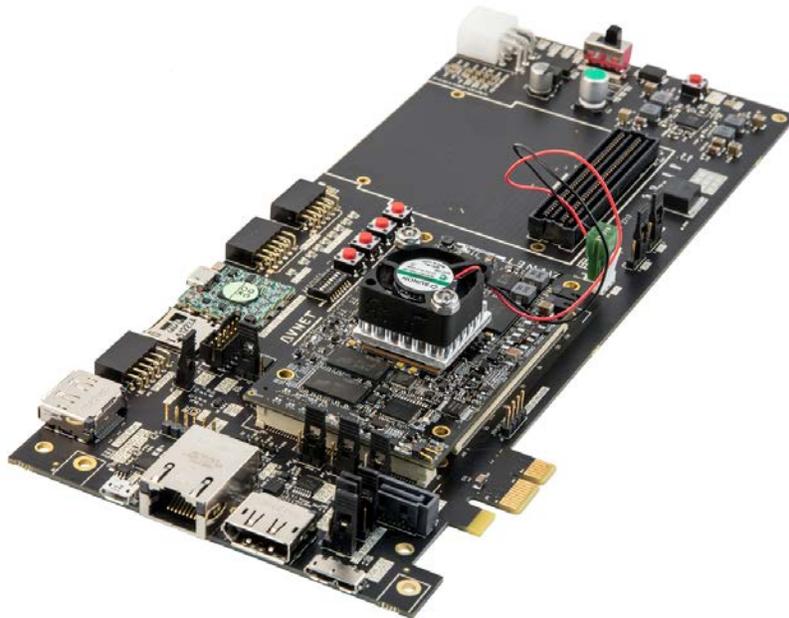


Figure 4 – UltraZed-EG PCIe Carrier Card with SOM

UltraZed-EV is a Zynq UltraScale+ MPSoC multi-core SOM which is centered around the Xilinx XCZU7EV. This high performance, full featured SOM mates with the Avnet UltraZed-EV carrier card, which breaks out the FBVB900 package to connect to many transceivers (PS and PL), many video standards, GigE, SATA 3.0, USB 2.0/3.0, PCIe Gen 2 Root Complex, as well as a FMC-HPC allowing access to the PCIe Gen 3 core through the PL interfaces. Being the MPSoC is a 7EV, this means this SOM also include the new H.264/H.265 video codec. This Video Codec Unit (VCU), can do simultaneous 4K2K encode and decode up to 60FPS! This board targets high performance-level Zynq MPSoC developers with a full featured media prototyping platform.

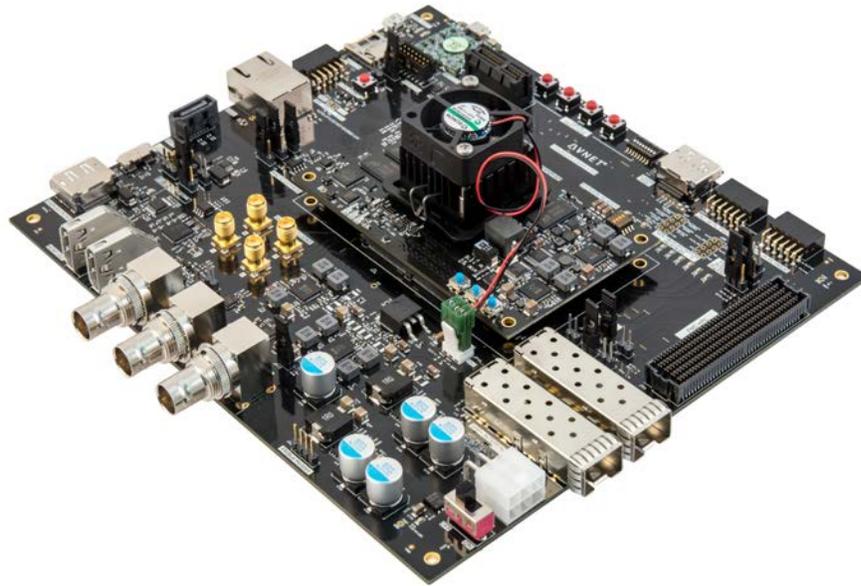


Figure 5 – UltraZed-EV Starter Kit

Please contact your local Avnet FAE for further details with any of these kits.

Lab 1 Design Objectives

Lab 1 offers system developers an example of how to:

- Work through and become familiar with the SDSoC 2018.3 tool flow, from a designer's perspective
- Demonstrate a Matrix Multiply Example output on one of the development kits listed using a pre-built SDSoC platform (you will learn later how to create that platform)
- Learn how to install custom SDSoC Platforms

In order to install a custom platform, you will first need to generate the platform. In this case, Xilinx and Avnet has multiple resources available. The main reference you should use will be the Xilinx User Guide 1146. The SDSoC Environment Platform Development Guide v2018.3 document has all the proper references to the details one would need to create their own platform. Avnet and Xilinx also both offer trainings. Avnet has a training guide which steps the user through creation of a custom platform for the MiniZed development board. This SpeedWay is called "A Practical Guide to Getting Started with Xilinx SDSoC".

Example Design Requirements

Software

The software used to test this reference design is:

- Xilinx SDx / SDSoC 2018.3 (SDSoC License Required)
- Platform archive
- MiniZed, Ultra96v1, Ultra96v2, UltraZed-EG with Carrier, or UltraZed-EV Board Definition for Vivado

Hardware

The hardware setup used to test this reference design includes:

- Lenovo ThinkPad T420 Laptop
 - Intel® Core i5-2540M CPU - 2.60 GHz
 - 4GB DDR3 Memory
 - SD card slot on PC or external USB-based SD card reader
- Avnet MiniZed (AES-MINIZED-7Z007-G)
 - Or
- Avnet Ultra96v1 (AES-ULTRA96-G)
 - Or
- Avnet Ultra96v2 (PN TBD)
 - Or
- Avnet UltraZed-EG Starter Kit (AES-ZU3EG-1-SK-G)
 - Or
- Avnet UltraZed-EG SOM with PCIECC (AES-ZU3EG-1-SOM-G and AES-ZU-PCIECC-G)
 - Or
- Avnet UltraZed-EV Starter Kit (AES-ZU7EV-1-SK-G)
- 1 - USB cable (Type A to Micro-USB Type B)

Experiment Set Up

You must have installed the Xilinx tools and properly licensed them. The Board Definition files should be installed into both your SDx based Vivado installation. Instructions for installing board definitions are located on the UltraZed.org website.

You will also need an unzip tool, such as 7-zip.

Experiment 1: Install the Pre-Built SDx Hardware Platform

(mz_avnet, u96v1_avnet, u96v2_avnet, uzegiocc_avnet, uzegpcie_avnet, uzev_avnet)

SDSoC comes pre-installed with many platforms that allow you to immediately use many Xilinx boards. The listed development boards are not one of them. For this experiment, you will use pre-built hardware platforms called **<developmentBoard>_avnet** so that you can quickly begin using SDx. You will later learn how to build that hardware platform.

Installation of an SDx hardware platform is a two-step process:

- Copy the hardware platform files to a repository. For our purposes today, we will use `C:\Avnet\platforms` as our repository location.
- Setup SDx to use a repository, pointed at the repository location

We'll start this experiment by creating a new project so that you can see the pre-installed platforms. Then you will add the three Avnet platforms.

1. Launch the SDx IDE by clicking **Start → Xilinx Design Tools → SDx 2018.2 → SDx IDE 2018.2**

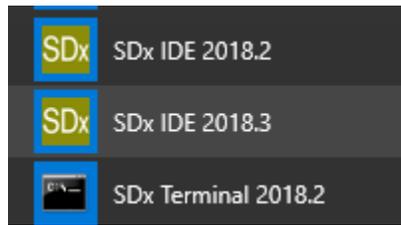
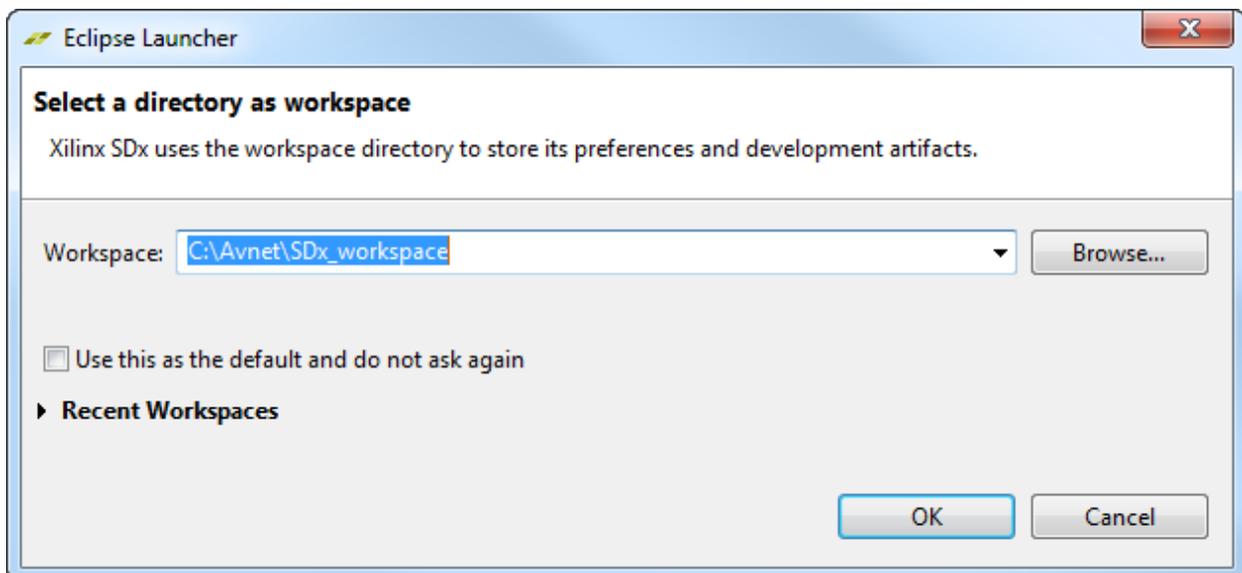


Figure 6 – Launch SDx IDE

2. SDSoc requires a workspace. Due to Windows path length constraints. It is important that this path be short. Please enter this specific path for consistency through these labs and then click **OK**.

C:\Avnet\SDx_workspace



3. If you see a security alert, select **Allow Access**.

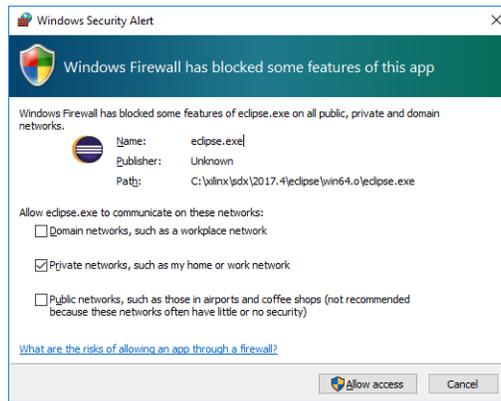


Figure 7 – Security Alert

The SDx IDE will launch and validate your license. You should see the Welcome dashboard as shown below.

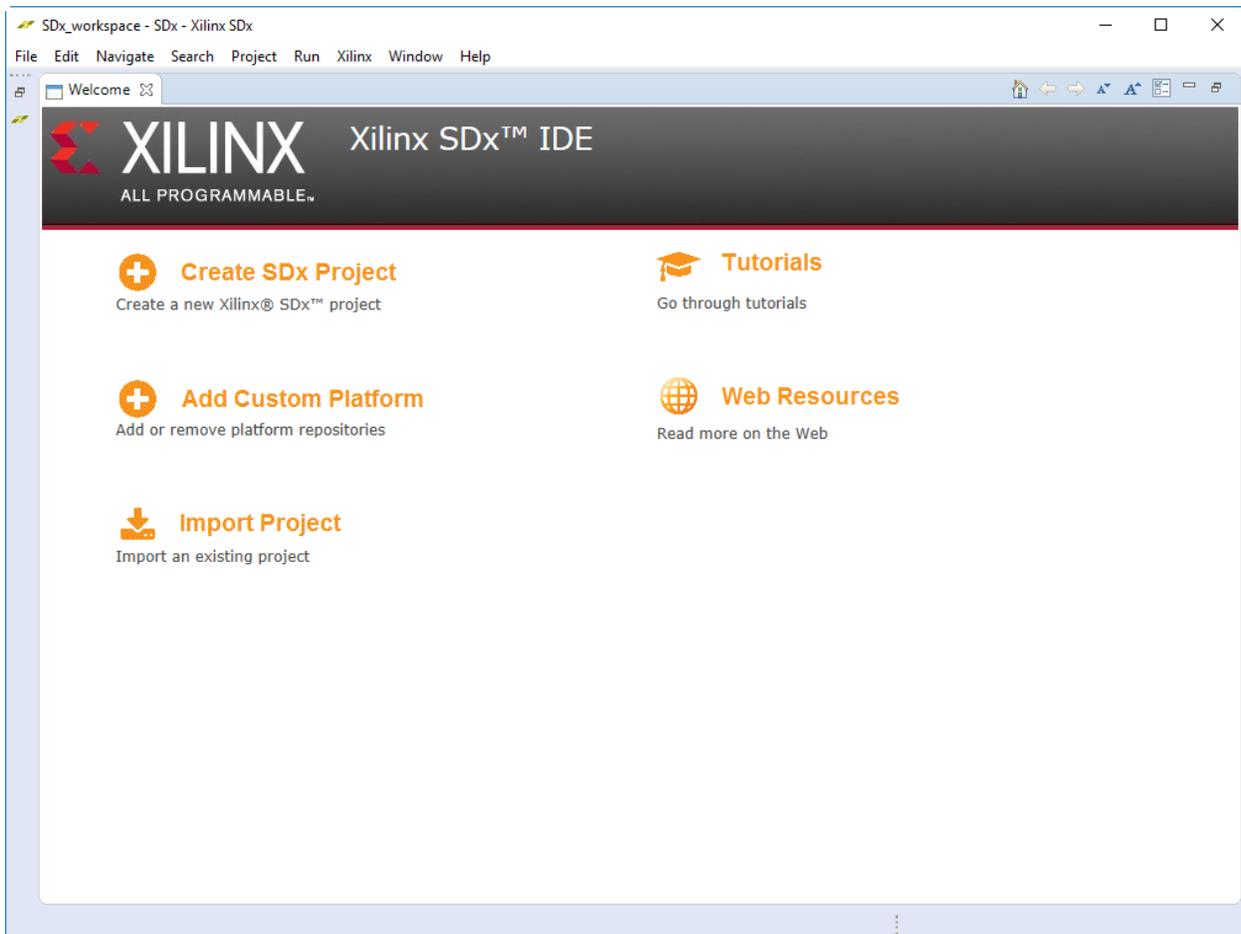


Figure 8 – SDx IDE Dashboard

4. Begin by creating a new Xilinx SDx project
 - a. Select **File** → **New** → **SDx Application Project...** or click **Create SDx Project** on Welcome screen

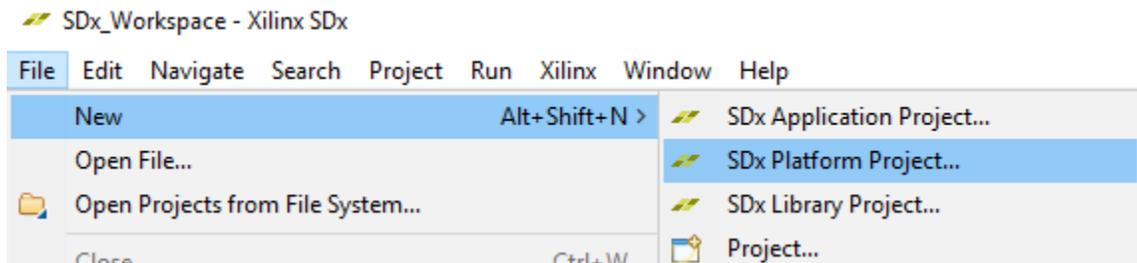


Figure 9 – Xilinx SDx Project Creation

5. For Project Name, enter **MiniZedMM, U96V1MM, U96V2MM, UZEGIOCCMM, UZEGPCIECCMM** or **UZEVMM**
 - b. NOTE: the location of the project

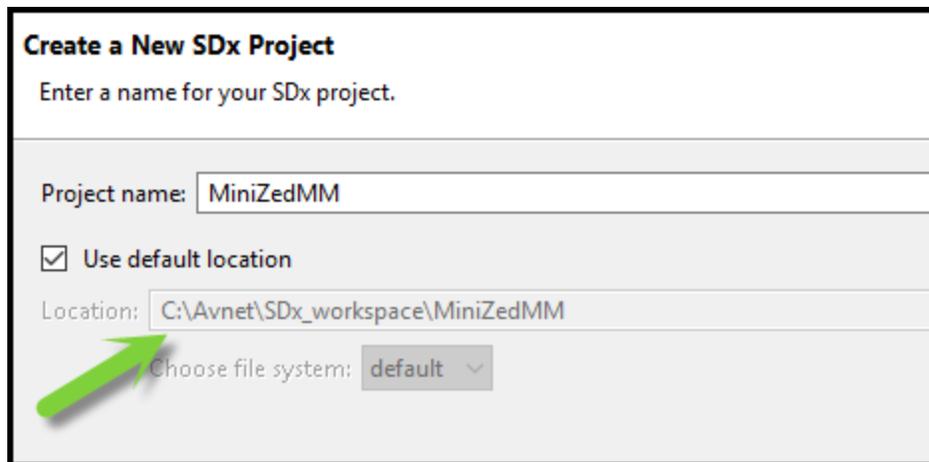


Figure 10 – Project Location

6. Click **Next >**

Here you will see the list of pre-installed Hardware Platforms. These are all Zynq-based boards. Notice that MiniZed is not in the list.

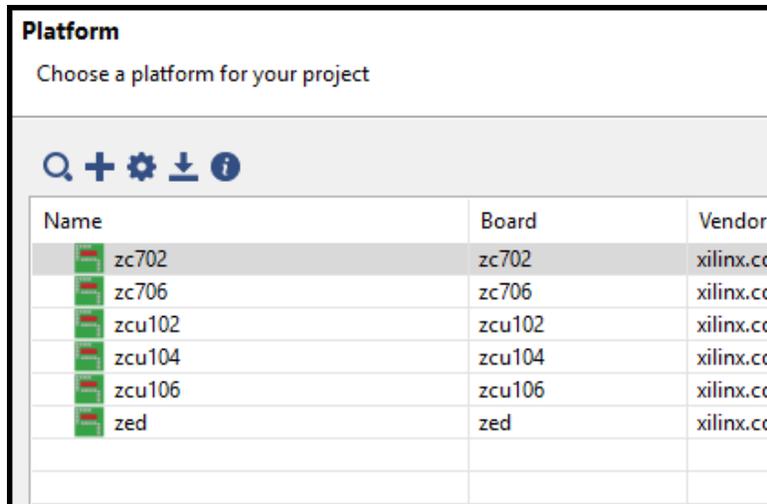


Figure 11 – Pre-installed Hardware Platforms in SDx 2018.3

7. Open Windows Explorer and create the directory C:\Avnet\platforms
8. Now unzip the SDSoc_Platform_v2018p3.zip archive into the repository at C:\Avnet\platforms

When complete you should have a directory structure as shown below:

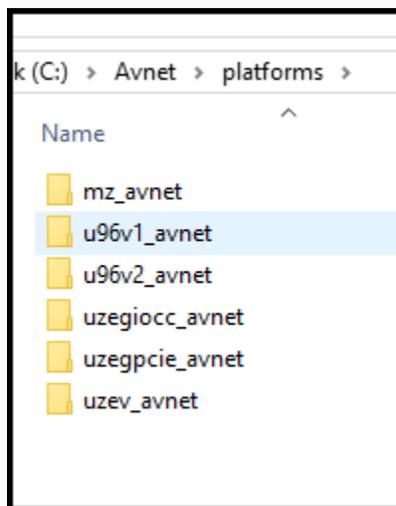


Figure 12 – SDx Platform Repository Folder

9. In the SDx New Project dialog, click on the + button

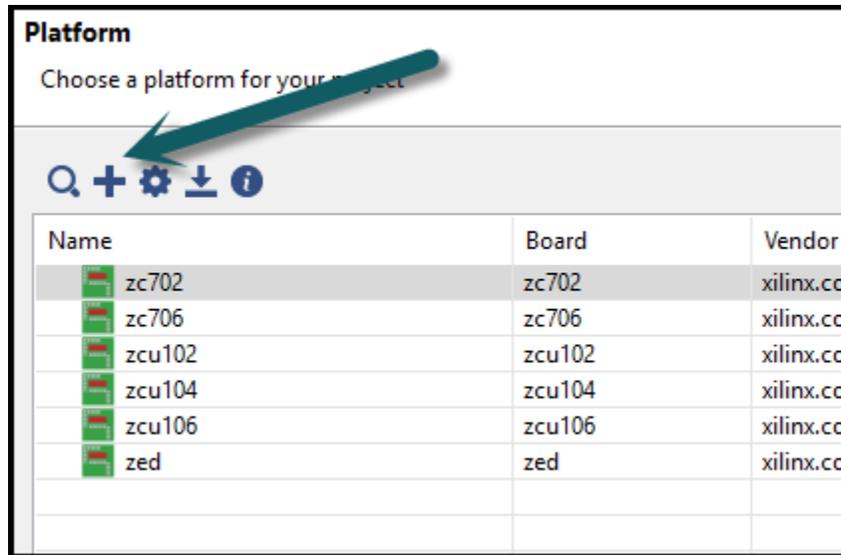


Figure 13 – Manage Repositories

10. In the next dialog, click on the green plus sign on the left pane.

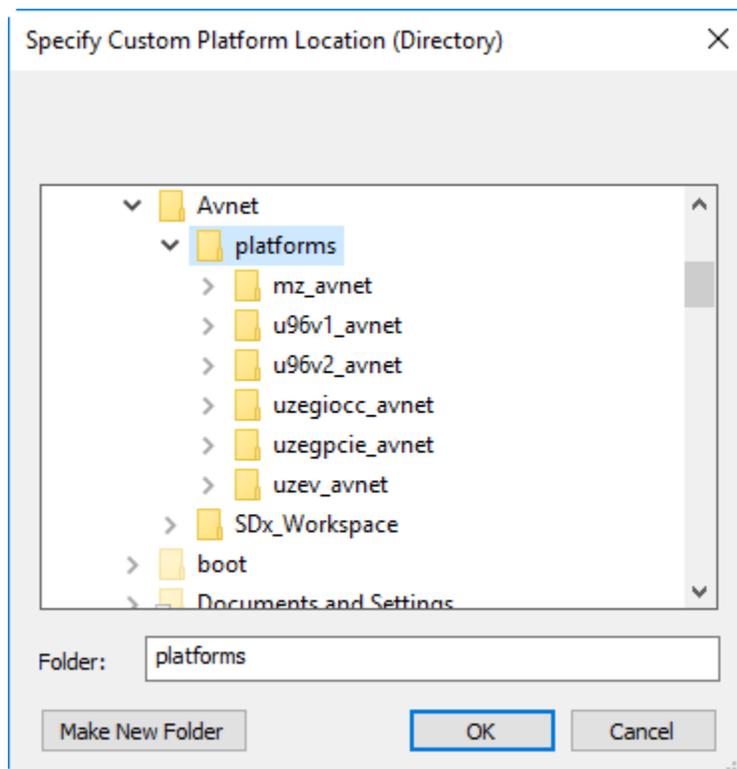
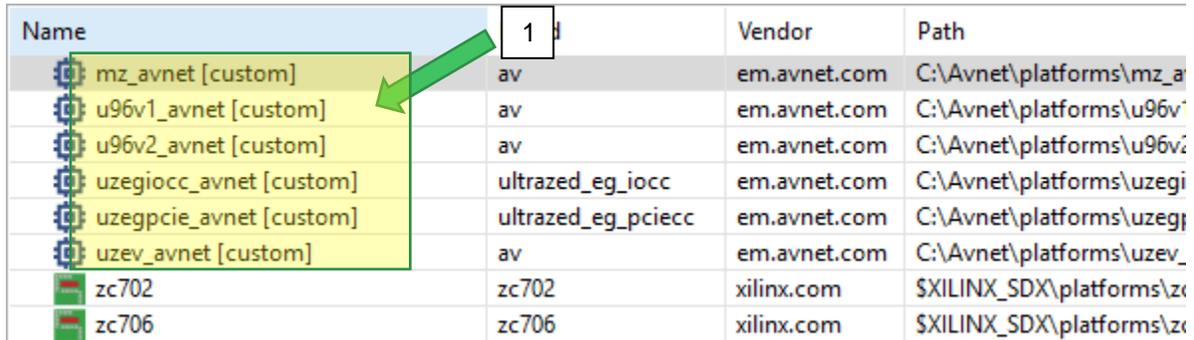


Figure 14 – Add Custom Platform Location

11. Navigate to C:\Avnet\platforms. Select **platforms** and click **OK**.

12. Notice that now the Platform List is updated.

Notice that in the **Choose Hardware Platform** is updated with the custom platforms and noted with a [custom] tag



| Name | 1 | Vendor | Path |
|-------------------------|--------------------|--------------|---------------------------|
| mz_avnet [custom] | av | em.avnet.com | C:\Avnet\platforms\mz_a |
| u96v1_avnet [custom] | av | em.avnet.com | C:\Avnet\platforms\u96v1 |
| u96v2_avnet [custom] | av | em.avnet.com | C:\Avnet\platforms\u96v2 |
| uzegiocc_avnet [custom] | ultrazed_eg_iocc | em.avnet.com | C:\Avnet\platforms\uzegi |
| uzegpcie_avnet [custom] | ultrazed_eg_pciecc | em.avnet.com | C:\Avnet\platforms\uzegp |
| uzev_avnet [custom] | av | em.avnet.com | C:\Avnet\platforms\uzev_ |
| zc702 | zc702 | xilinx.com | \$XILINX_SDX\platforms\zc |
| zc706 | zc706 | xilinx.com | \$XILINX_SDX\platforms\zc |

Figure 15 – mz_avnet (custom) Now an Available Platform

You have now installed Avnet's custom platforms.

Experiment 2: Create the mz_avnet Matrix Multiply Project

- Now, we can choose the platform you are targeting, here we choose the mz_avnet platform by selecting **mz_avnet [custom]** then click **Next >** to continue. You can also choose one of the other Avnet platforms.

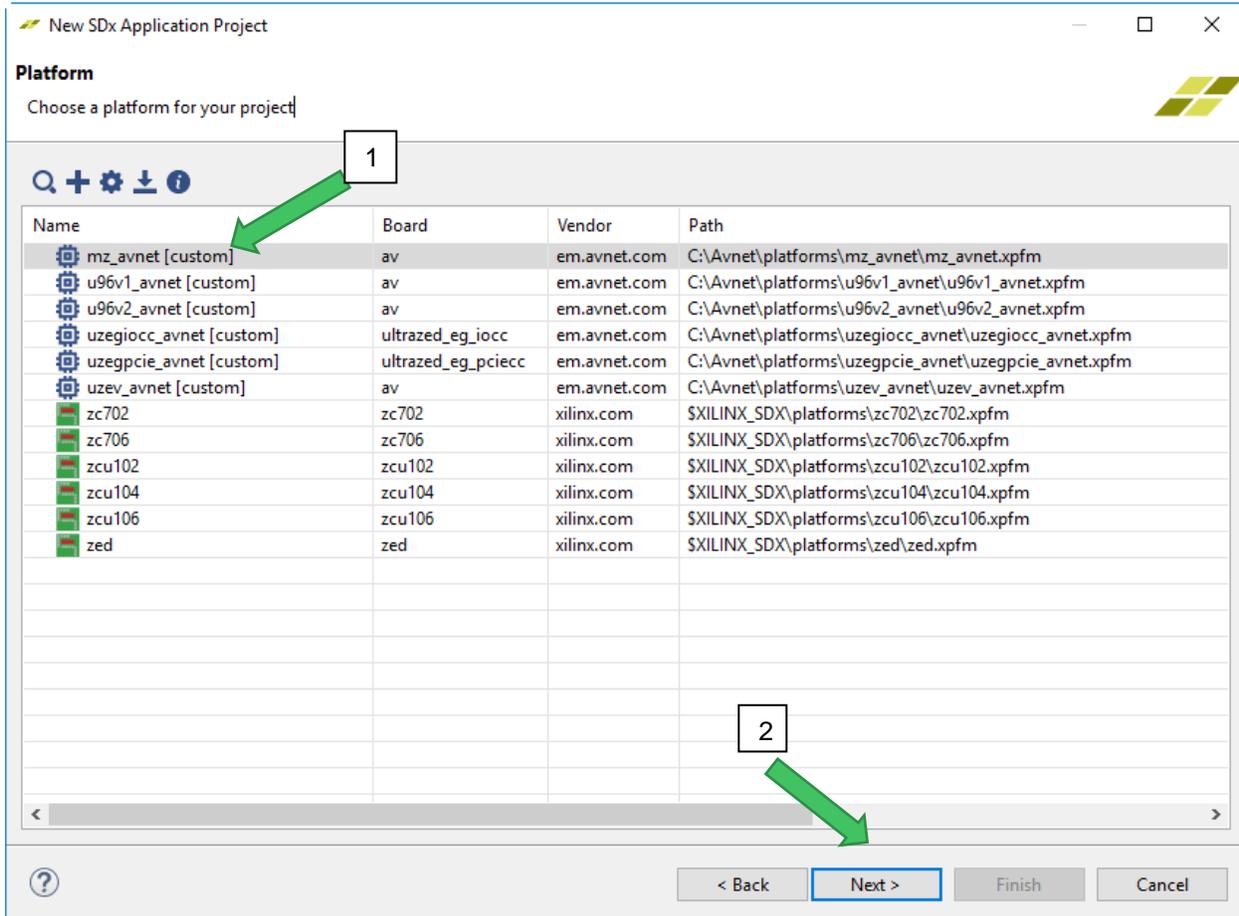
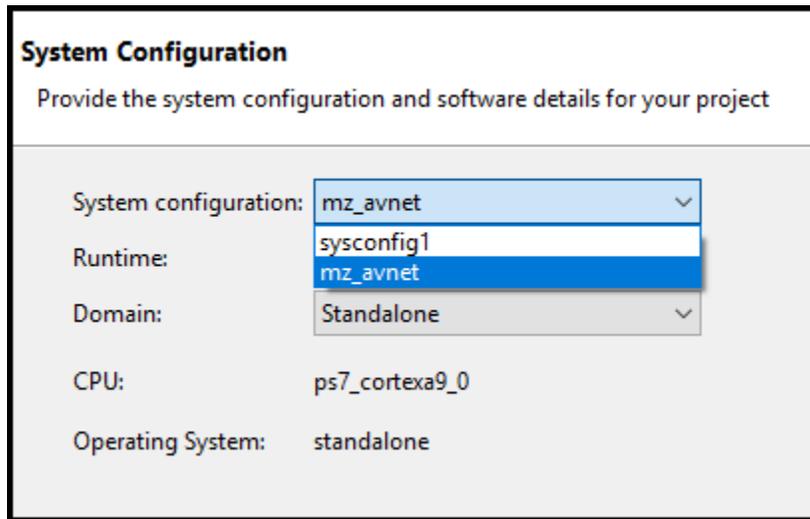


Figure 16 – Choose mz_avnet Hardware Platform

14. Due to HOW the platform was created, we now have two System Configurations. You should select `mz_avnet` from the System configuration drop down. Note that this is also where you can choose to generate C-Callable Libraries, a new feature in current tools.



System Configuration
Provide the system configuration and software details for your project

System configuration: mz_avnet

Runtime: sysconfig1
mz_avnet

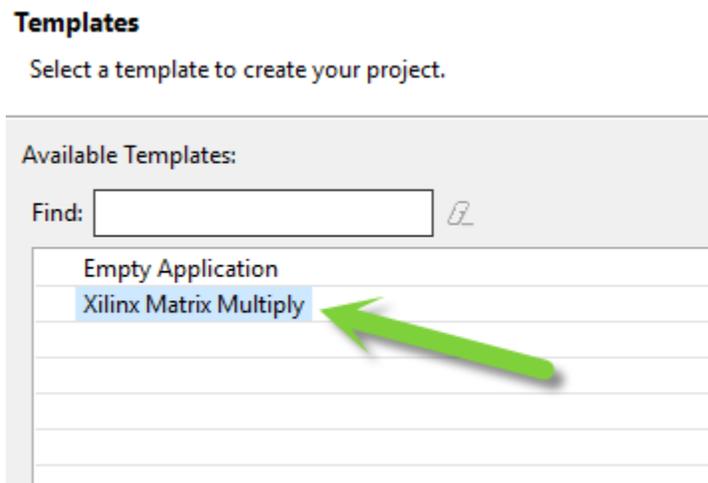
Domain: Standalone

CPU: ps7_cortexa9_0

Operating System: standalone

Figure 17 – Software Platform

15. Click **Next >** to continue.
16. Select the provided Xilinx Matrix Multiply example as shown in Figure 18



Templates
Select a template to create your project.

Available Templates:

Find: 

Empty Application

Xilinx Matrix Multiply

Figure 18 – Example Selection

17. Click **Finish** to create a new Empty application.

Now the SDx Project Explorer will have the MiniZedMM from which we can generate SDSoc projects.

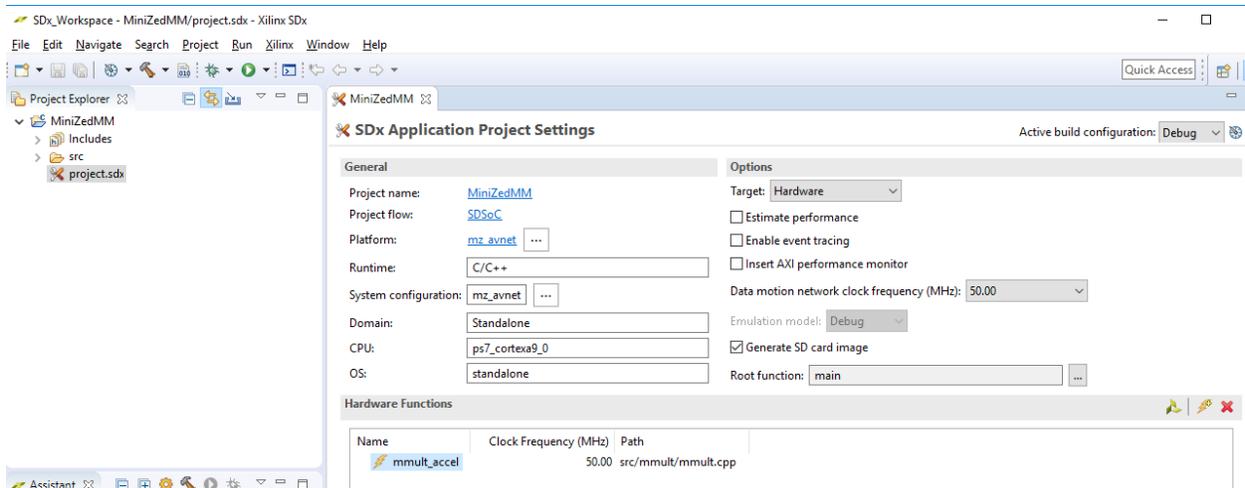


Figure 19 – MiniZedMM added to SDx Project Explorer

For the purposes of this experiment, we will use Matrix Multiply Example code.

The example code that you choose here is not really relevant as we are interested in the outputs of SDSoC as well as the platform itself. This should be seen as a black box where any accelerator (ex. video processing, LTE engine, AES engine, etc) could be inserted.

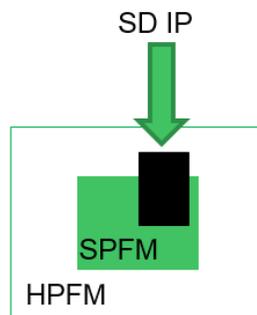


Figure 20 – Target Code is Black Box

This is one of the most powerful abilities of SDSoC! Since we are using a provided platform, simply imported the example template straight from the platform folder.

Instructions on how to create your own Sample Template is located in UG1146.

If you need to import files, the process is the same as with Xilinx SDK. To learn how to import the matrix multiply files from within the SDSoC installation see the Appendix.

Experiment 3: Build the mz_avnet Matrix Multiply Project

18. Next, back in SDx Project Explorer right click the MiniZedMM project name, click on **Build Project**, remember, you can follow this same flow with any of the provided platforms

19. While this is building, notice the SDx project Settings. This is a great place to see the overview of the settings that are going into building this project

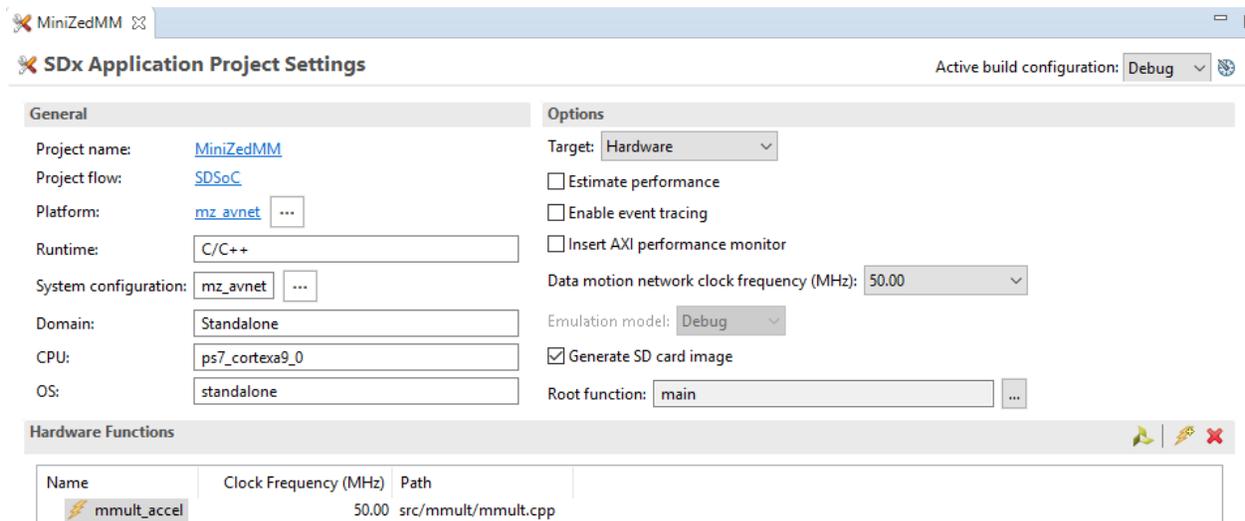


Figure 21 – Project Settings

We left the project as Debug and did not select Release. If we had selected Release, the tool would have performed additional runtime optimizations, which would have increased our build time.

The above steps can be seen in more detail in UG1028 – SDSoC Intro Tutorial v2018.3.

NOTE: Use DocNav or search directly from Xilinx.com as there is a good chance you can end up with an older version of the documentation.

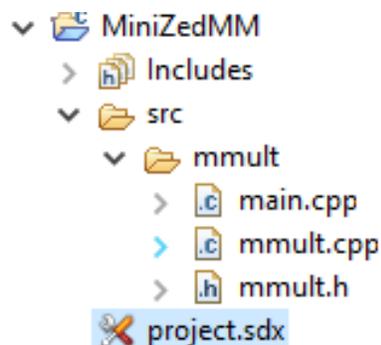


Figure 22 – MiniZedMM Sources

20. Notice that SDSoC generates a layout that looks very similar to the Xilinx SDK. This was a design decision to ensure a familiarity to the tools including the process flow.

While we wait for the build, let's explore. What is happening while this is building?

- The tool copies the Vivado project from the platform into your local build area
- The tool analyses the provided C code, including pragmas, and builds an internal data motion graph – what connects to what, how, etc. It will make decisions at this stage based on your memory configuration, buffer sizes (if known), etc. to determine interfaces, data movers, etc.
- The C code moving to hardware is synthesized by HLS
- SDSoC updates the BD to incorporate the data motion infrastructure and the generated HLS IPs
- The tool updates your C code to seamlessly call the accelerator instead of the C function (you can see the results of this in the `_sds` directory in the project build area)
- Generate a bitstream
- Combine the bitstream into BOOT.BIN using the FSBL, BIF, etc. that you provide as part of the platform

21. Navigate to `C:\Avnet\SDx_workspace\MinizedMM\src\mmult`. Notice the 3 source files there.
Note: Swap the bolded project folder name for the one you chose to build.

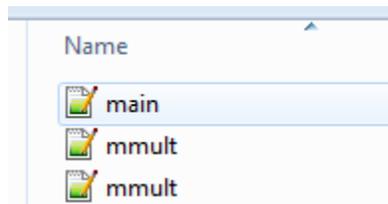


Figure 23 – MinizedMM Source files

More notes regarding the files and the file structure

- Main.cpp runs the functions
 - Mmult.cpp has pragmas listed
 - Mmult.h has pragmas listed
22. While this is building, open the Xilinx Software Command Line Tool. If you are not using a MiniZed, you can skip to step 23, as the other kits have SDCARD slots and the boot.bin can be directly loaded on to the SDCARD for testing.
- a. We will need this to jtag the BOOT.BIN file onto our MiniZed.

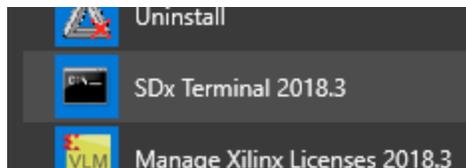


Figure 24 – Open Xilinx Software Command Line Tool

Experiment 4: Set Up Your MiniZed

While the project is building, you can also set up your MiniZed.

23. First configure the boot jumper. For non-MiniZed kits, refer to the Quick Start Card or Manual for how to configure the Kit for SDCARD booting. If not using a MiniZed, skip to the next step.

For MiniZed, you can select between FLASH and JTAG booting. We want to ensure switch 1 is set towards the F or PS_Button.

- a. Note: From the Factory the switch's protective film should be removed and already set to F. If it is not, the switch will look similar to Figure 25.

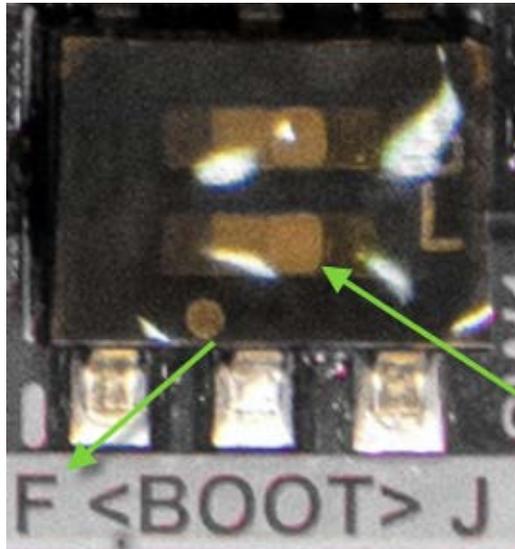


Figure 25 - Untouched Boot Switch

- b. If your MiniZed boot configuration switch is similar to the above, remove the protective film and slide switch 1 (indicated by the silkscreen DASH above the F) to be toggled to FLASH Booting (F).
24. Next plug the Development Kit into your PC in order to register the board with a COM port
 - a. Note: Windows 10 has been known to create two COM ports when plugging the MiniZed into the PC.
 - b. Note: Ultra96 will need the Debug Adapter Board, refer to the Manual for more information
 - c. For a MiniZed, with a factory fresh board, open two instances of Tera Term, one for each COM port; 8,N,1,115200
 - d. Reboot your MiniZed using the Reset button (see Figure 31 – Reset Button)
 - e. In the Windows Device Manager (see Experiment 5, step 27) remove the COM port that did NOT have text appear

Experiment 5: Run the Design

25. Insert one USB cable into the MiniZed USB JTAG/UART MicroUSB connector
 - a. For other kits, plug one MicroUSB cable into the UART USB connector
26. Plug the other end into your PC
27. If this is the first time you have connected your MiniZed, locate the COM Port assigned, if you are not using a MiniZed, skip to step 28
 - a. Right click on “My Computer” and choose Manage
 - b. From here, select Device Manager
 - c. Under the Ports section, locate the USB Serial Port Assignment

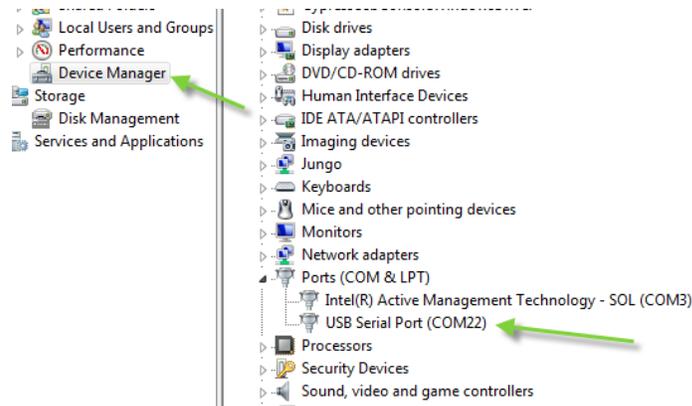


Figure 26 - Device Manager Used to Find USB Serial Port COM Port

28. Open a terminal program such as TeraTerm. Configure it to connect to the COM Port we found in the previous step using 115200/8/n/1/n as settings.
 - a. NOTE: if you open C:\Program Files (x86)\teraterm\TERATERM.INI you can change the defaults so you do not have to continuously modify the settings each time you open a serial port

```
; Serial port parameters
; Port number
ComPort=1
; Baud rate
BaudRate=9600
; Parity (even/odd/none/mark/space)
Parity=none

; Serial port parameters
; Port number
ComPort=4
; Baud rate
BaudRate=115200
; Parity (even/odd/none/mark/space)
Parity=none
```

Figure 27 – Changing TeraTerm Defaults

If this is the first time you have powered up your MiniZed, you will see a LOT of text as the default factory boot image is PetaLinux.

SDSoC builds BOOT.BIN files. While the intention is to be able to use these with SDCARD boot, such as with devices like Ultra96 and UltraZed-EV, this is just a standard BOOT.BIN file and can be treated as such for devices like MiniZed, which do not have an SDCARD cage.

29. For products with an SDCARD cage, skip to step 36

30. Using explorer, navigate to the below and inspect for files. They will appear once the build is complete

C:\Avnet\SDx_workspace\MiniZedMM\Debug\sd_card

31. Once the build is complete, in the **SDK** Xilinx Software Command Line Tool 2018.3, change the working directory to the location of BOOT.BIN

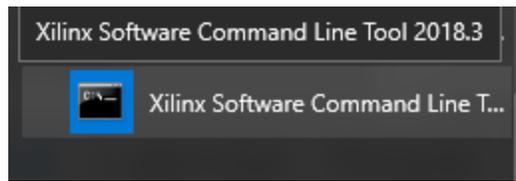


Figure 28 – Xilinx Software Command Line Tool (SDK)

```
cd C:/Avnet/SDx_workspace/MiniZedMM/Debug/sd_card
```

32. Once at the xsct% prompt, execute the command to program the QSPI over JTAG in the MiniZed.

```
exec program_flash -f boot.bin -fsbl  
C:/Avnet/platforms/mz_avnet/sw/mz_avnet/boot/fsbl.elf -flash_type  
qspi_single
```

a. If you get a firewall/security message, accept it, as the JTAG process uses IP to interact with the Zynq chip (below shown with older tool version)

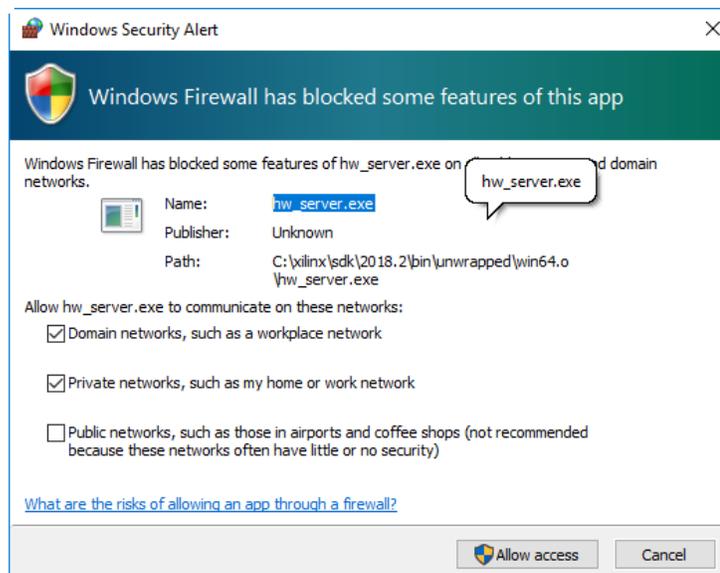
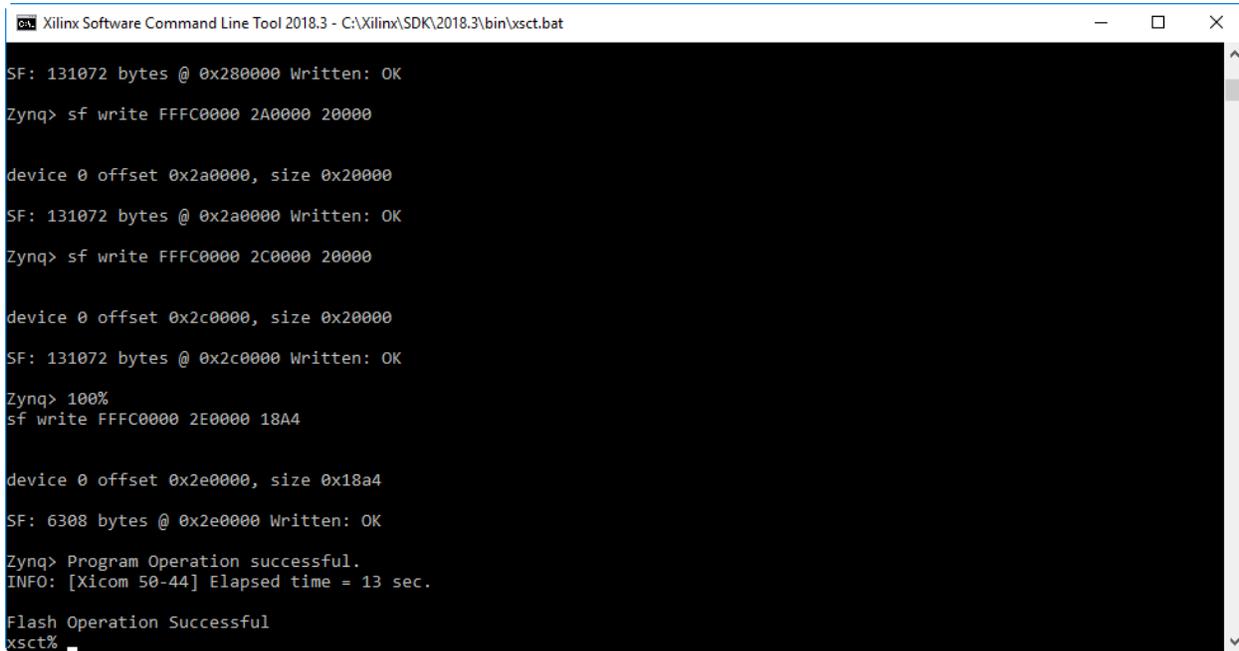


Figure 29 – JTAG Security Message

- b. Notice the blue DONE LED turns off. This indicates the image is being programmed, do not interrupt this process!
- c. When complete, you should see the message, "Flash Operation Successful"



```
Xilinx Software Command Line Tool 2018.3 - C:\Xilinx\SDK\2018.3\bin\xsct.bat
SF: 131072 bytes @ 0x280000 Written: OK
Zynq> sf write FFFC0000 2A0000 20000
device 0 offset 0x2a0000, size 0x20000
SF: 131072 bytes @ 0x2a0000 Written: OK
Zynq> sf write FFFC0000 2C0000 20000
device 0 offset 0x2c0000, size 0x20000
SF: 131072 bytes @ 0x2c0000 Written: OK
Zynq> 100%
sf write FFFC0000 2E0000 18A4
device 0 offset 0x2e0000, size 0x18a4
SF: 6308 bytes @ 0x2e0000 Written: OK
Zynq> Program Operation successful.
INFO: [Xicom 50-44] Elapsed time = 13 sec.
Flash Operation Successful
xsct%
```

Figure 30 – Programming MiniZed QSPI

- 33. Close XSCT and the Explorer window. Note that if you were to rebuild and the two windows were left open, the SDx tools will COMPLETE all building, BIT FILE, HLS IPI generation, etc., and JUST as it is ready to combine into a boot.bin, it will fail the build as XCST% will hold the directory while the SDx tool tried to delete – thus failing after the 20 minute+ build. Therefore, closing XSCT is critical.
- 34. Once you get the successful message, press the RESET button on the MiniZed

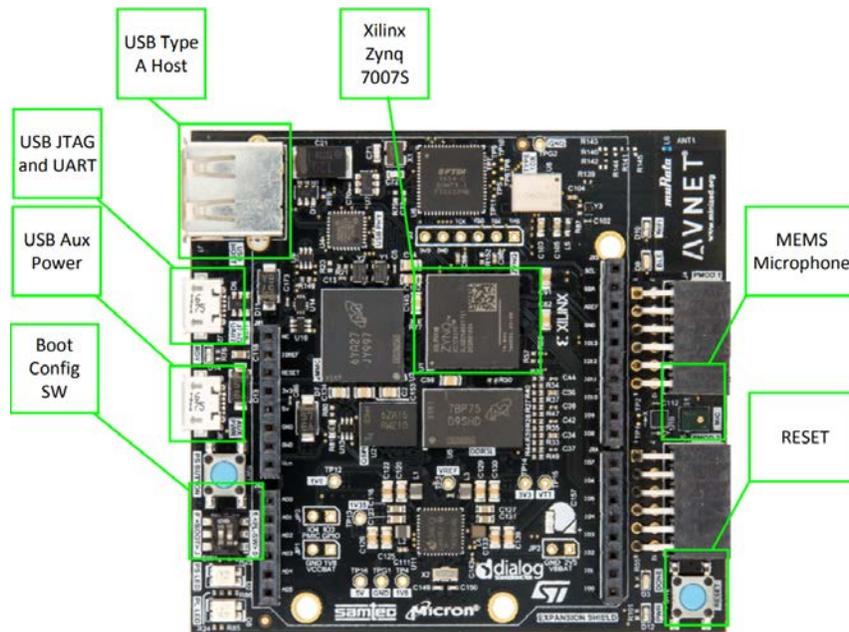


Figure 31 – Reset Button

35. The BLUE Done should light up again and you will see after each press of the reset, the Matrix Multiply will run.
36. For MiniZed, skip this step. For other products with an SDCARD cage, copy the `boot.bin` file to your included SDCARD. The file is located:
 - C:\Avnet\SDx_workspace\U96V1MM\Debug\sd_card
 - or
 - C:\Avnet\SDx_workspace\U96V2MM\Debug\sd_card
 - or
 - C:\Avnet\SDx_workspace\UZEGIOCCMM\Debug\sd_card
 - or
 - C:\Avnet\SDx_workspace\UZEGPCIEMM\Debug\sd_card
 - or
 - C:\Avnet\SDx_workspace\UZEVMM\Debug\sd_card
37. For MiniZed, skip this step. For other products with an SDCARD cage, insert the SDCARD into the SDCARD cage. Turn on the power and you should observe a similar display, as shown in Figure 33 for Ultra96 and for Figure 37 UltraZed-EV.
 - a. Note that with MiniZed, due to the reduced resources in a Zynq ZC7007s, the Matrix Multiply has been reduced to a 24x24 array. For Ultra96 and UltraZed-EV, you will see a 32x32 array
 - b. Note that the acceleration that MiniZed provides is based on a 50MHz clock. The Ultra96 and all UltraZed types utilize a 100MHz clock. Acceleration results will be higher on these platforms.
38. You can now CLOSE the SDx IDE.

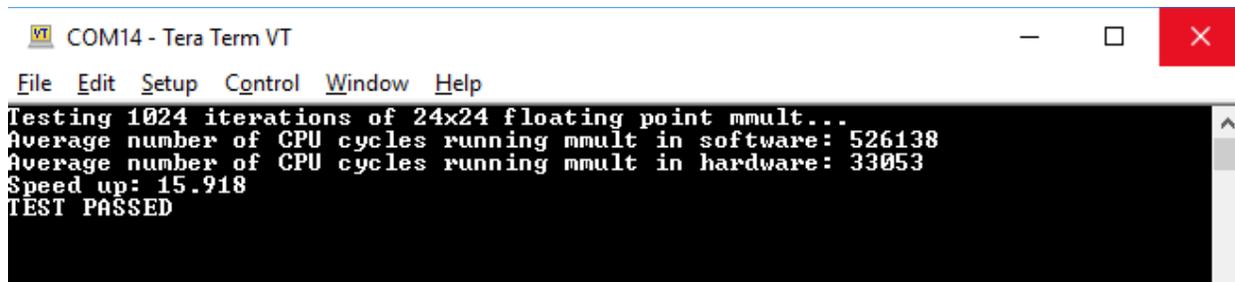


Figure 32 – Matrix Multiply Running in Both Software and Hardware, MiniZed

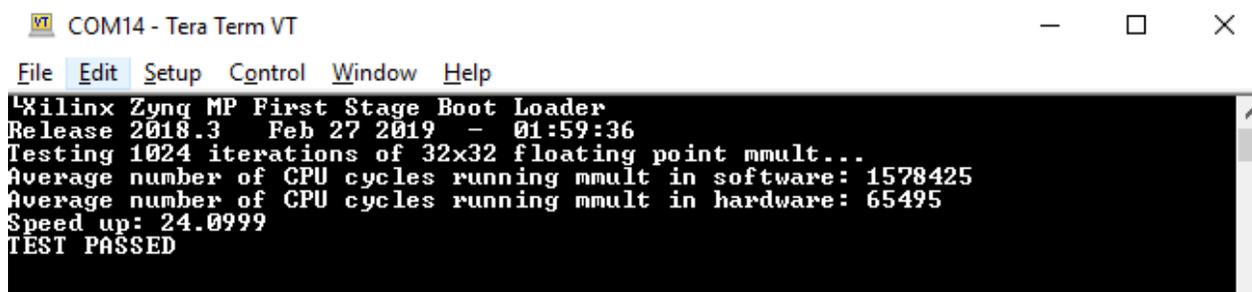


Figure 33 – Matrix Multiply Running in Both Software and Hardware, Ultra96

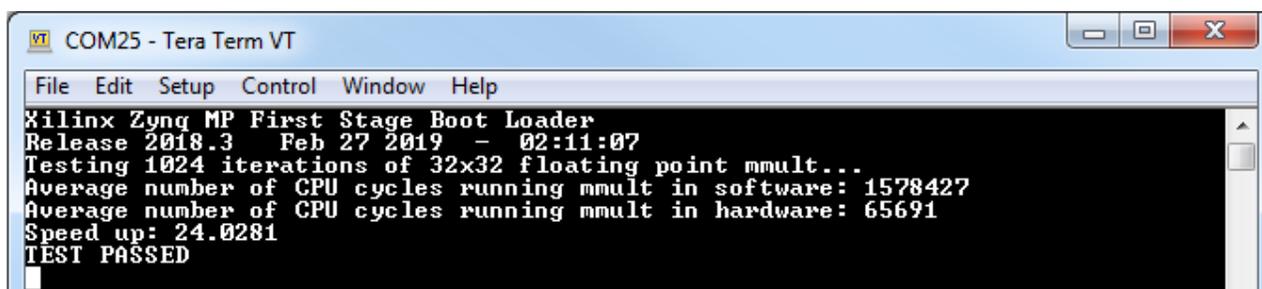


Figure 34 – Matrix Multiply Running in Both Software and Hardware, Ultra96v2

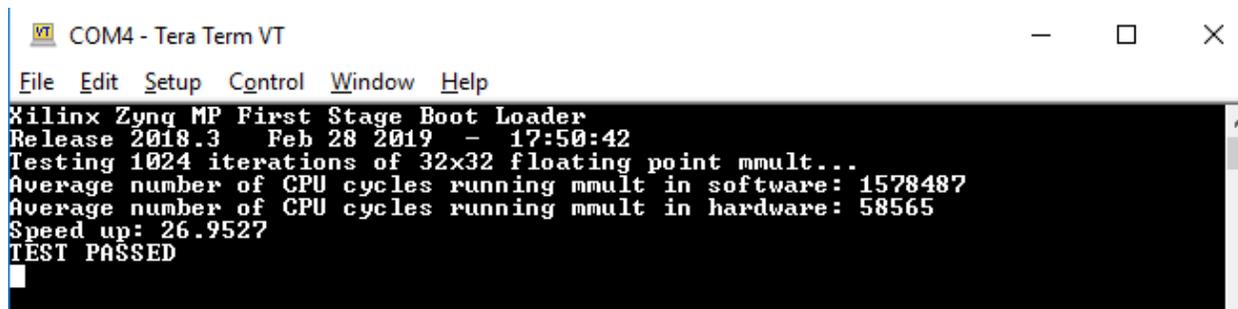


Figure 35 – Matrix Multiply Running in Both Software and Hardware, UltraZed-EG IOCC

```
COM10 - Tera Term VT
File Edit Setup Control Window Help
Xilinx Zynq MP First Stage Boot Loader
Release 2018.3 Feb 28 2019 - 18:05:38
Testing 1024 iterations of 32x32 floating point mmult...
Average number of CPU cycles running mmult in software: 1578503
Average number of CPU cycles running mmult in hardware: 58495
Speed up: 26.9853
TEST PASSED
```

Figure 36 – Matrix Multiply Running in Both Software and Hardware, UltraZed-EG PCIeCC

```
COM6 - Tera Term VT
File Edit Setup Control Window Help
Xilinx Zynq MP First Stage Boot Loader
Release 2018.3 Feb 27 2019 - 02:20:31
Testing 1024 iterations of 32x32 floating point mmult...
Average number of CPU cycles running mmult in software: 1578501
Average number of CPU cycles running mmult in hardware: 54156
Speed up: 29.1473
TEST PASSED
```

Figure 37 – Matrix Multiply Running in Both Software and Hardware, on UltraZed-EV

Appendix A: Importing Files

In order to import source files into our project, the operational flow is the same as with Vivado SDK. These basic steps are laid out in the following procedure. This is only necessary if you are importing files instead of using the example built into the provided platform.

1. Right click on the MiniZedMM src folder and select **Import...**

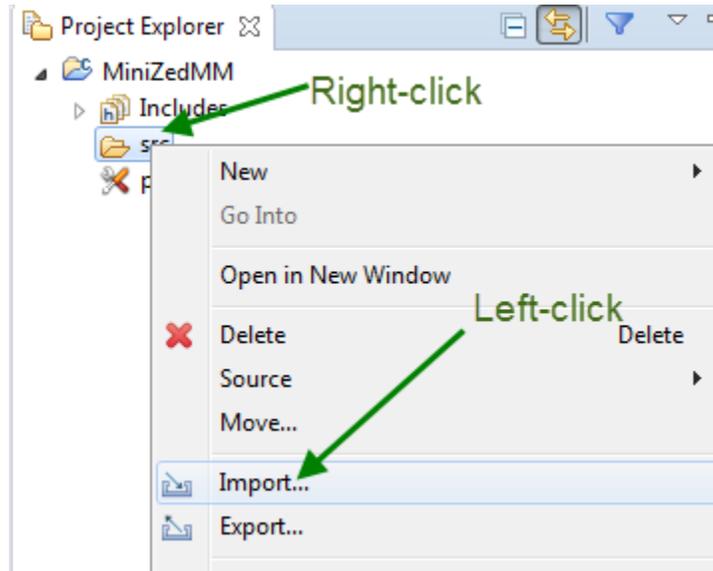


Figure 38 – Import Source Files

2. Select **General**, then **File System** and **Next >**.

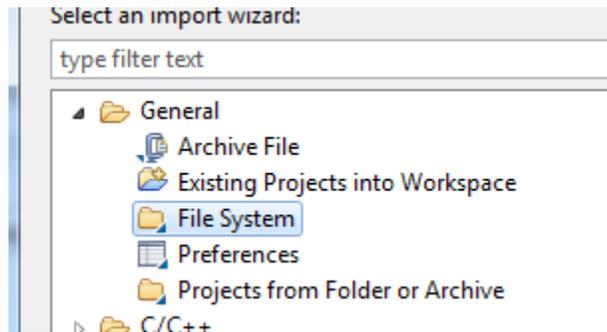


Figure 39 – Import from the File System

3. For the “From Directory” field, either copy/paste the location below or click on **Browse** then browse to the following example location:

C:\Xilinx\SDx\2018.2\samples\mmult

- Next select three checkboxes next to the .cpp and .h files. Ensure that the **Into folder** box shows MiniZedMM/src as well as the checkboxes matches the screen in the below figure.

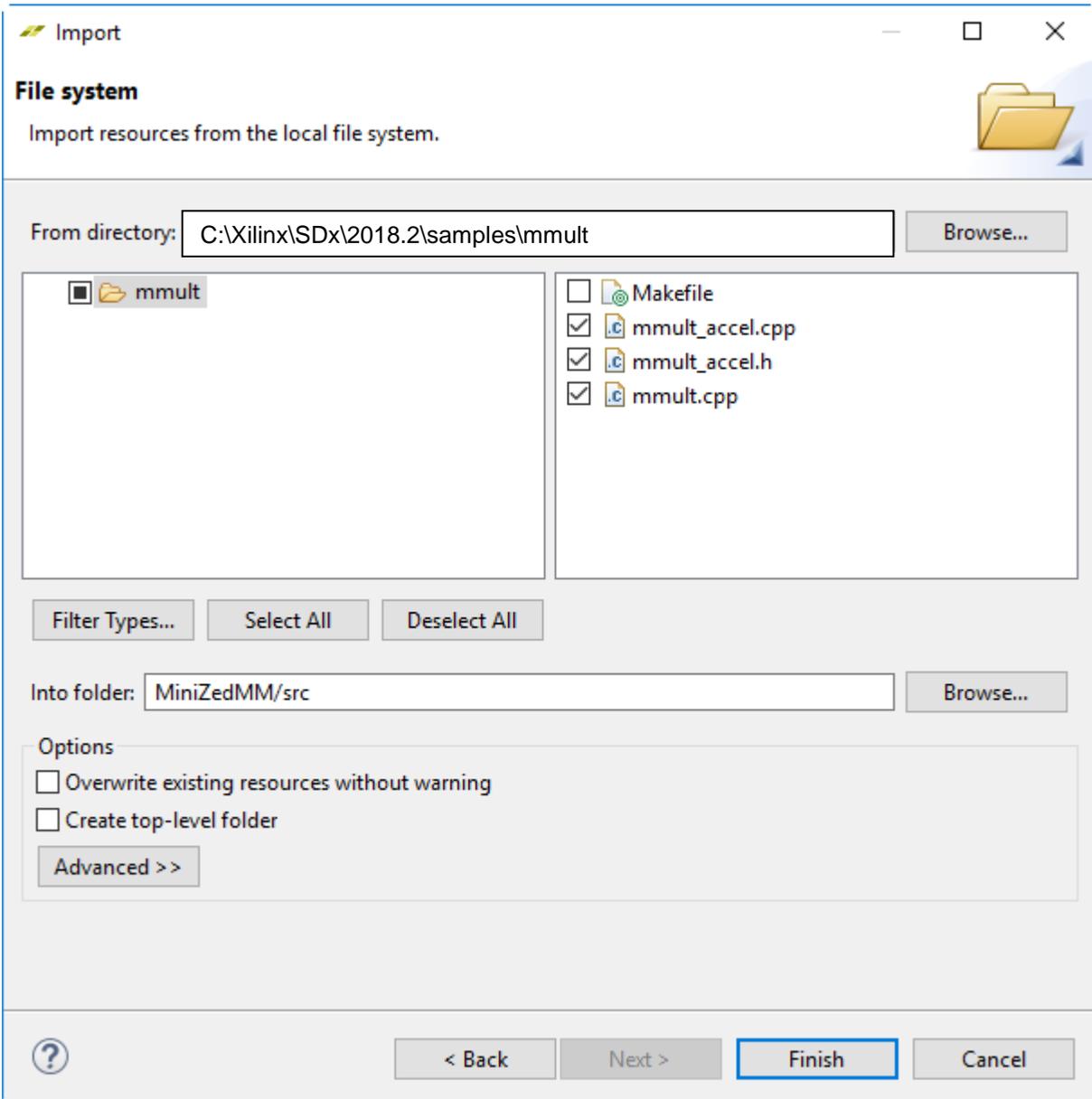


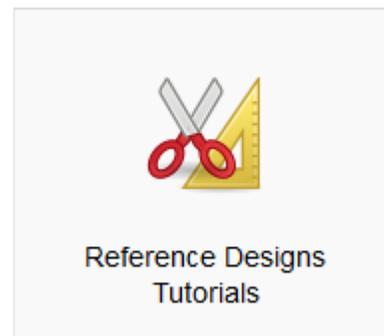
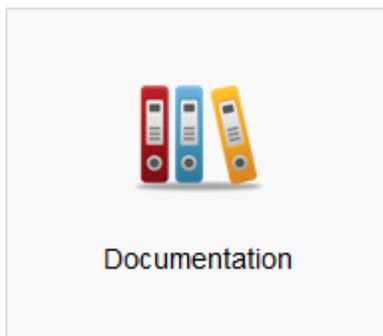
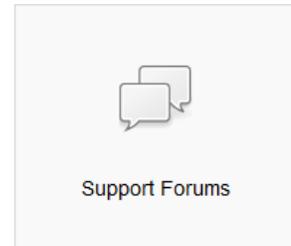
Figure 40 – Import Example Files

- Click **Finish**. You should now see the three sources added in Project Explorer.

Appendix B: Getting Support

Avnet Support

- Technical support is offered online through the minized.org website support forums. MiniZed users are encouraged to participate in the forums and offer help to others when possible.
- To access the most current collateral for the MiniZed, visit the community support page (www.minized.org/content/support) and click one of the icons shown below:



- MiniZed Documentation
<http://minized.org/support/documentation/18891>
- MiniZed Reference Designs
<http://minized.org/support/design/18891/146>
- Ultra96 Documentation
<http://ultra96.org/support/documentation/24166>
- Ultra96 Reference Designs
<http://ultra96.org/support/design/24166/156>
- UltraZed-EV Documentation
(Starter Kit) <http://ultrazed.org/support/documentation/22596>
(SOM) <http://ultrazed.org/support/documentation/25481>
(Carrier Card) <http://ultrazed.org/support/documentation/22581>
- UltraZed-EV Reference Designs
<http://ultrazed.org/support/design/22581/166>

Xilinx Support

The following technical support options are available to Xilinx customers:

- Technical information is available online 24 hours a day from the [Support website](#)
- Technical Support staff are available to respond to your questions in the [Community Forums](#)
- Individual assistance from Xilinx Technical Support **may** be available through [Service Portal](#)
- Phone support is **only** available with an active open case number

Global Phone Number

| Region | Language | Phone** | Support Hours* |
|--------------------------------|-------------------|---|----------------------|
| North America | EN | 1 800-255-7778 or +1 408-879-5199 | M-F 7:00 -17:00 PST |
| Europe, Middle East and Africa | EN, DE, FR | 00 800-5152-5152 or +353 1-461-5700 | M-F 8:00 -17:00 GMT |
| China | CH (Mandarin), EN | +86 800 988 0218 +86 400 880 0218 (Mobile Phone) | M-F* 9:00 -18:00 CST |
| Taiwan | CH (Mandarin), EN | +886 2-8176-1060 | M-F 9:00 -18:00 CST |
| Hong Kong | CH (Mandarin), EN | +852 3187-3855 | M-F 9:00 -18:00 CST |

* Support hours listed apply for both standard and daylight savings (summer) time. Please check the [Technical Support Holiday Calendar 2016](#) for support availability during holidays in your region.

** 00 800-5152-5152 is a international free phone (toll free) number available in the following countries: Austria, Belgium, Denmark, Finland, France, Germany, Ireland Israel, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland, and United Kingdom. All other countries must use +353 1-461-5700.

** For the numbers listed, '+' represents the International Direct Dialing (IDD) prefix of the country from which you are calling. Please consult your local telephone service provider for more information on specific IDD instructions.

Revision History

| Date | Version | Revision |
|-------------|----------------|--|
| 25 Aug 17 | 00 | Preliminary release |
| 28 Aug 17 | 00 | Edits per Author |
| 17 Sep 17 | 1 | First public release |
| 19 Apr 18 | 2 | Updated to SDSoC 2017.4 |
| 18 Aug 18 | 3 | Updated to SDSoC 2018.2, added UltraZed-EV and Ultra96 |
| 19 Oct 18 | 3.1 | Updated for new Ultra96 BDF name |
| 25 Oct 18 | 3.2 | Added in UltraZed-EG IOCC and PCIECC |
| 27 Feb 19 | 4 | Updated to SDSoC v2018.3 |