MicroZed/PicoZed: Hello World

15 July 2015
Version 2015_2.01

Overview

Once a Zynq Hardware Platform is created and exported from Vivado, the next step is to create an application targeted at the platform and see it operating in hardware. This tutorial will show how to do that with the simplest of all software applications – Hello World.

Objectives

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

- Import a Zynq Hardware Platform into SDK
- Create a BSP
- Add a new application based on a Xilinx-provided template in SDK
- Run the application on the MicroZed or PicoZed hardware

Experiment Setup

Software

The software used to test this reference design is:

- Windows-7 64-bit
- Xilinx SDK 2015.2
- Silicon Labs CP201x USB-to-UART Bridge Driver
    - www.microzed.org → Support → Documentation → MicroZed Silicon Labs CP210x USB-to-UART Setup Guide
    - Note that MicroZed and the PicoZed FMC Carrier both use the same Silicon Labs CP2104 device, so the setup is the same.

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
    - 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
- One of the following:
    - Avnet MicroZed 7010 or 7020
    - Avnet PicoZed 7010, 7015, 7020, or 7030 with the PicoZed FMC Carrier
- USB cable (Type A to Micro-USB Type B) – *one included in kit*
- JTAG Programming Cable (Platform Cable, Digilent HS1, HS2, or HS3 cable)
    - If you don't already have a JTAG Cable, Avnet recommends the Digilent HS3 Cable
    - http://www.em.avnet.com/jtaghs3

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

Experiment 1: Import the Hardware Platform

The first requirement within SDK is to import a hardware platform.

1. Launch SDK by selecting **Start → All Programs → Xilinx Design Tools → SDK 2015.2 → Xilinx SDK 2015.2**.
2. Select a workspace. Click **OK**.

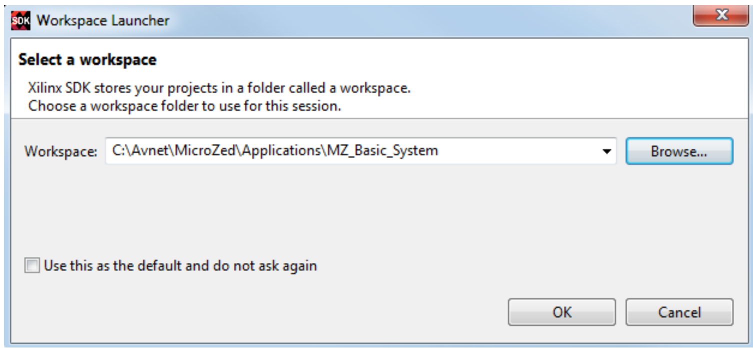


Figure 1 – SDK Workspace

3. If at any time you get a Windows Security Alert, select the first two checkboxes, then click **Allow access**, then click **Yes**.

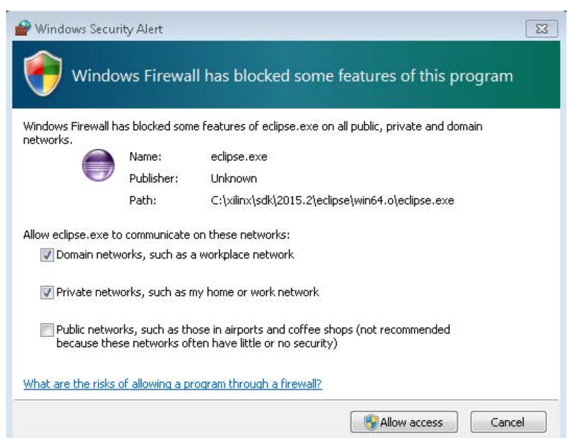


Figure 2 – Windows Security Alert from SDK

4. Close the *Welcome* screen by clicking the ☒ next to *Welcome* on the tab.
5. Select **File → New → Project**.

6. Expand the *Xilinx* item, and select **Hardware Platform Specification**. Click **Next >**.

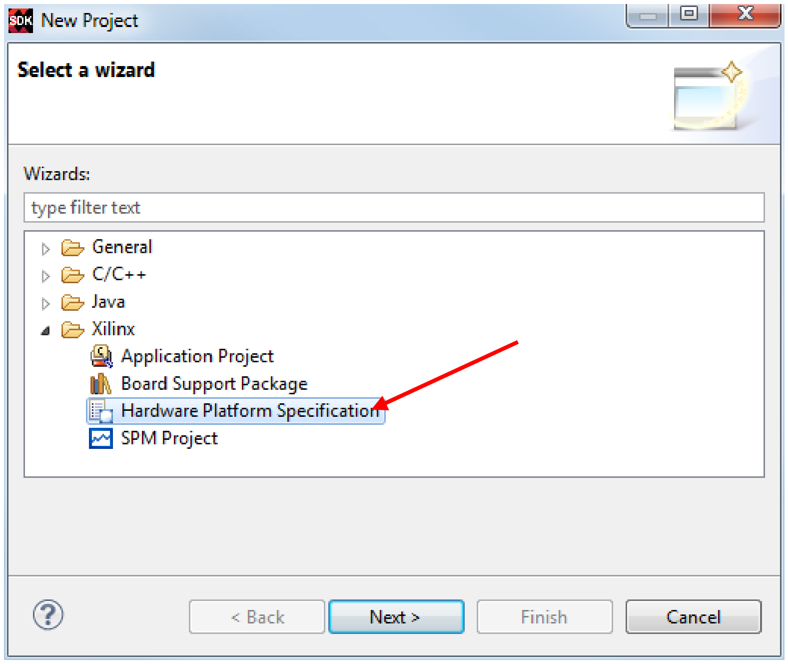


Figure 3 – Creating a New Hardware Platform

If you had simply launched SDK from Vivado, it would have automatically named and imported your hardware platform for you. For consistency, we will use the same default name that Vivado would have used.

7. Insert **hw_platform_0** for the *Project name*.
8. Click **Browse** and select the `System_wrapper.hdf` file generated during the Export process from Vivado. This will be included in the archive provided by the hardware engineer. Or, if you are continuing from the first tutorial, you will find it in a similar location as here:

   `C:\Avnet\MicroZed\Projects\MZ_Basic_System\MZ_Basic_System.sdk\`

9. After selecting `System_wrapper.hdf`, click **Open**.
10. The Bitstream is embedded in the HDF, so it is not separately specified here. Click **Finish**.

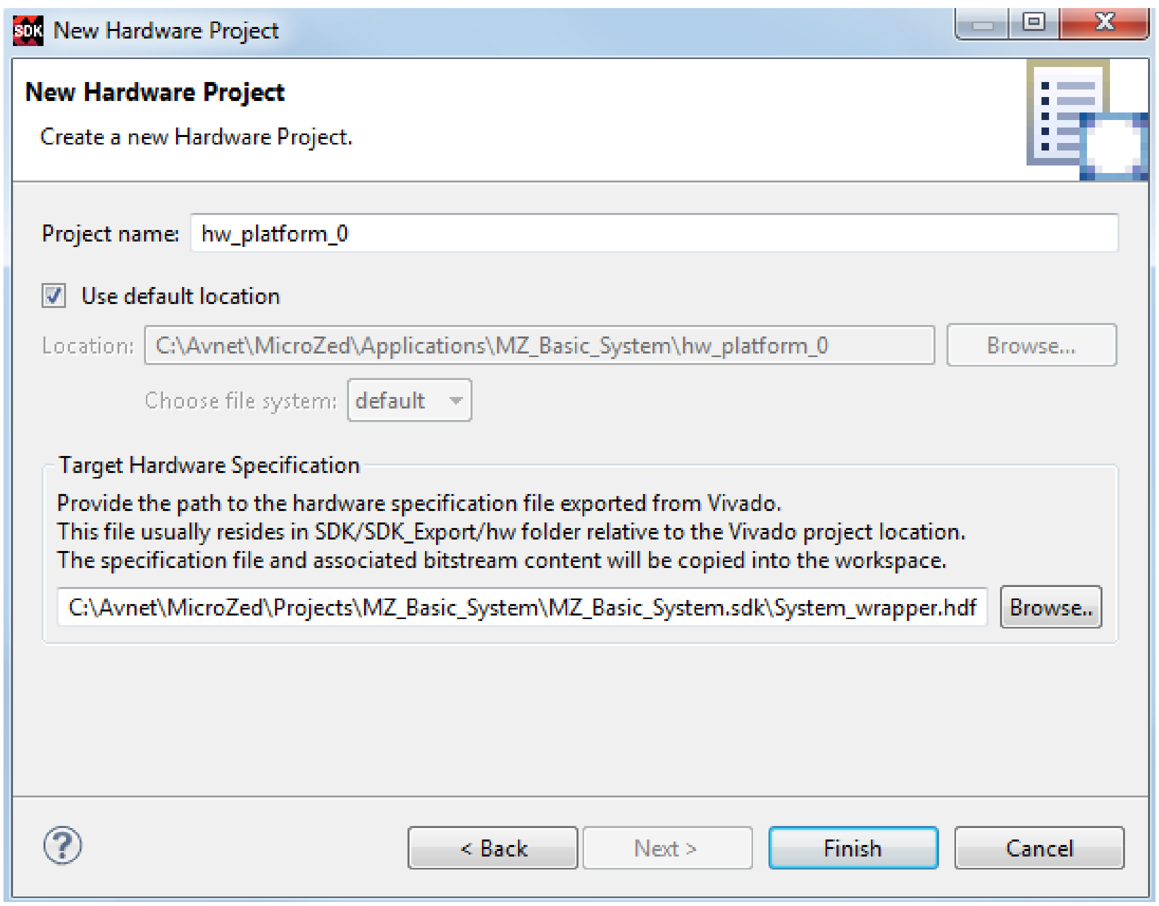


Figure 4 – Import Hardware Platform from Vivado

11. Notice the PS7 Zynq hardware platform is now visible in the *Project Explorer*.

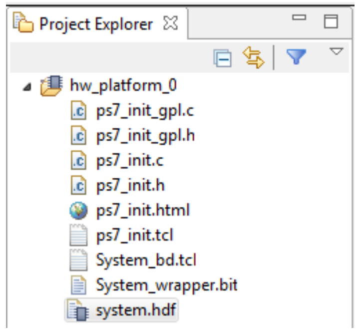
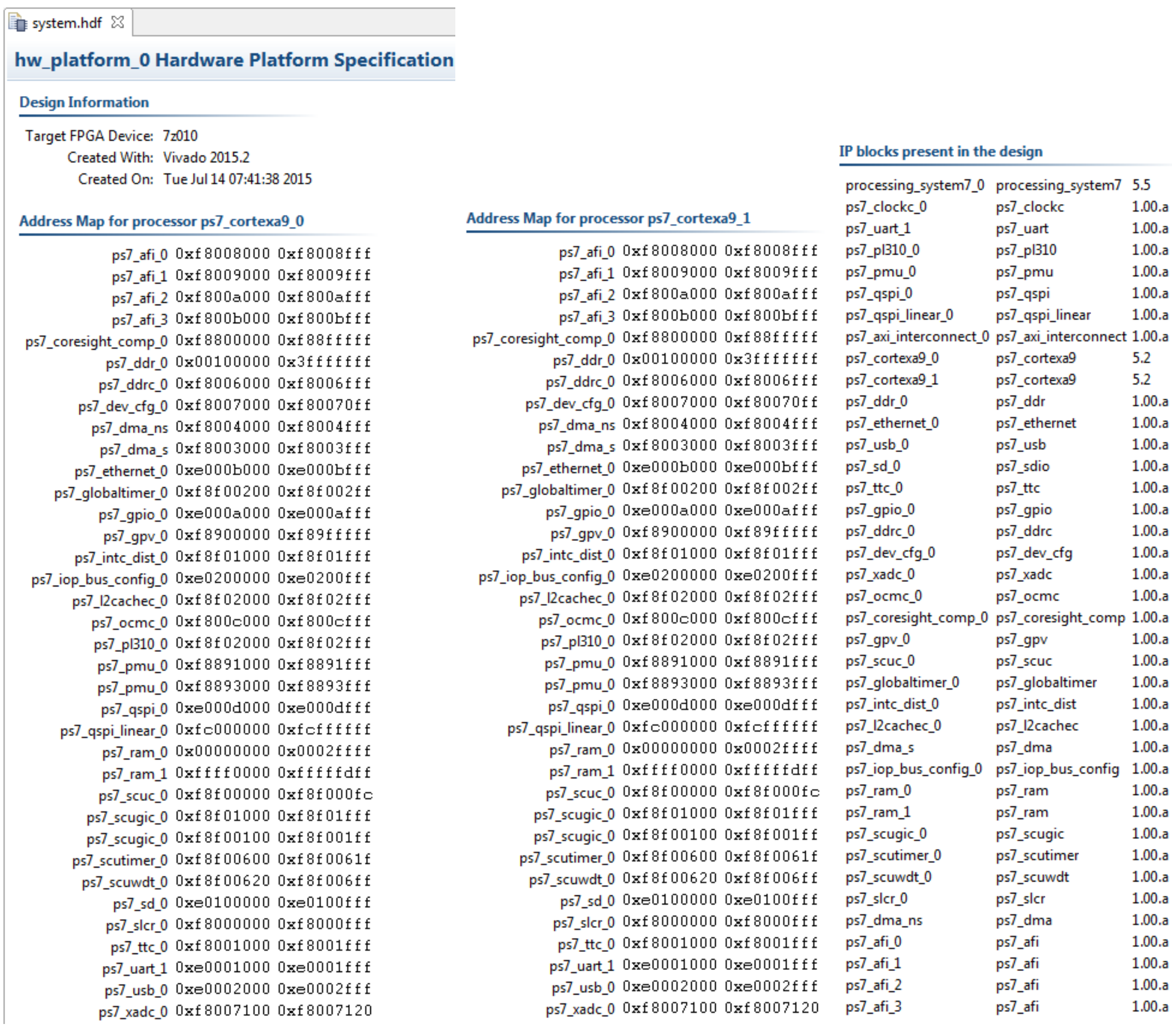


Figure 5 – Hardware Platform Imported and Ready for Use

If you select the HDF file, SDK will show you information about the hardware platform (not the HDF raw code itself).



system.hdf

hw_platform_0 Hardware Platform Specification

Design Information

Target FPGA Device: 7z010
Created With: Vivado 2015.2
Created On: Tue Jul 14 07:41:38 2015

Address Map for processor ps7_cortexa9_0

ps7_afi_0	0xf8008000	0xf8008fff
ps7_afi_1	0xf8009000	0xf8009fff
ps7_afi_2	0xf800a000	0xf800afff
ps7_afi_3	0xf800b000	0xf800bfff
ps7_coresight_comp_0	0xf8800000	0xf88fffff
ps7_ddr_0	0x00100000	0x3fffffff
ps7_ddrc_0	0xf8006000	0xf8006fff
ps7_dev_cfg_0	0xf8007000	0xf80070ff
ps7_dma_ns	0xf8004000	0xf8004fff
ps7_dma_s	0xf8003000	0xf8003fff
ps7_ethernet_0	0xe000b000	0xe000bfff
ps7_globaltimer_0	0xf8f00200	0xf8f002ff
ps7_gpio_0	0xe000a000	0xe000afff
ps7_gpv_0	0xf8900000	0xf89fffff
ps7_intc_dist_0	0xf8f01000	0xf8f01fff
ps7_iop_bus_config_0	0xe0200000	0xe0200fff
ps7_l2cachec_0	0xf8f02000	0xf8f02fff
ps7_ocmc_0	0xf800c000	0xf800cfff
ps7_pl310_0	0xf8f02000	0xf8f02fff
ps7_pmu_0	0xf8891000	0xf8891fff
ps7_pmu_0	0xf8893000	0xf8893fff
ps7_qspi_0	0xe000d000	0xe000dfff
ps7_qspi_linear_0	0xfc000000	0xfcffffff
ps7_ram_0	0x00000000	0x0002ffff
ps7_ram_1	0xffff0000	0xfffffdff
ps7_scuc_0	0xf8f00000	0xf8f000fc
ps7_scugic_0	0xf8f01000	0xf8f01fff
ps7_scugic_0	0xf8f00100	0xf8f001ff
ps7_scutimer_0	0xf8f00600	0xf8f0061f
ps7_scuwdt_0	0xf8f00620	0xf8f006ff
ps7_sd_0	0xe0100000	0xe0100fff
ps7_slcr_0	0xf8000000	0xf8000fff
ps7_ttc_0	0xf8001000	0xf8001fff
ps7_uart_1	0xe0001000	0xe0001fff
ps7_usb_0	0xe0002000	0xe0002fff
ps7_xadc_0	0xf8007100	0xf8007120

Address Map for processor ps7_cortexa9_1

ps7_afi_0	0xf8008000	0xf8008fff
ps7_afi_1	0xf8009000	0xf8009fff
ps7_afi_2	0xf800a000	0xf800afff
ps7_afi_3	0xf800b000	0xf800bfff
ps7_coresight_comp_0	0xf8800000	0xf88fffff
ps7_ddr_0	0x00100000	0x3fffffff
ps7_ddrc_0	0xf8006000	0xf8006fff
ps7_dev_cfg_0	0xf8007000	0xf80070ff
ps7_dma_ns	0xf8004000	0xf8004fff
ps7_dma_s	0xf8003000	0xf8003fff
ps7_ethernet_0	0xe000b000	0xe000bfff
ps7_globaltimer_0	0xf8f00200	0xf8f002ff
ps7_gpio_0	0xe000a000	0xe000afff
ps7_gpv_0	0xf8900000	0xf89fffff
ps7_intc_dist_0	0xf8f01000	0xf8f01fff
ps7_iop_bus_config_0	0xe0200000	0xe0200fff
ps7_l2cachec_0	0xf8f02000	0xf8f02fff
ps7_ocmc_0	0xf800c000	0xf800cfff
ps7_pl310_0	0xf8f02000	0xf8f02fff
ps7_pmu_0	0xf8891000	0xf8891fff
ps7_pmu_0	0xf8893000	0xf8893fff
ps7_qspi_0	0xe000d000	0xe000dfff
ps7_qspi_linear_0	0xfc000000	0xfcffffff
ps7_ram_0	0x00000000	0x0002ffff
ps7_ram_1	0xffff0000	0xfffffdff
ps7_scuc_0	0xf8f00000	0xf8f000fc
ps7_scugic_0	0xf8f01000	0xf8f01fff
ps7_scugic_0	0xf8f00100	0xf8f001ff
ps7_scutimer_0	0xf8f00600	0xf8f0061f
ps7_scuwdt_0	0xf8f00620	0xf8f006ff
ps7_sd_0	0xe0100000	0xe0100fff
ps7_slcr_0	0xf8000000	0xf8000fff
ps7_ttc_0	0xf8001000	0xf8001fff
ps7_uart_1	0xe0001000	0xe0001fff
ps7_usb_0	0xe0002000	0xe0002fff
ps7_xadc_0	0xf8007100	0xf8007120

IP blocks present in the design

processing_system7_0	processing_system7	5.5
ps7_clockc_0	ps7_clockc	1.00.a
ps7_uart_1	ps7_uart	1.00.a
ps7_pl310_0	ps7_pl310	1.00.a
ps7_pmu_0	ps7_pmu	1.00.a
ps7_qspi_0	ps7_qspi	1.00.a
ps7_qspi_linear_0	ps7_qspi_linear	1.00.a
ps7_axi_interconnect_0	ps7_axi_interconnect	1.00.a
ps7_cortexa9_0	ps7_cortexa9	5.2
ps7_cortexa9_1	ps7_cortexa9	5.2
ps7_ddr_0	ps7_ddr	1.00.a
ps7_ethernet_0	ps7_ethernet	1.00.a
ps7_usb_0	ps7_usb	1.00.a
ps7_sd_0	ps7_sdio	1.00.a
ps7_ttc_0	ps7_ttc	1.00.a
ps7_gpio_0	ps7_gpio	1.00.a
ps7_ddrc_0	ps7_ddrc	1.00.a
ps7_dev_cfg_0	ps7_dev_cfg	1.00.a
ps7_xadc_0	ps7_xadc	1.00.a
ps7_ocmc_0	ps7_ocmc	1.00.a
ps7_coresight_comp_0	ps7_coresight_comp	1.00.a
ps7_gpv_0	ps7_gpv	1.00.a
ps7_scuc_0	ps7_scuc	1.00.a
ps7_globaltimer_0	ps7_globaltimer	1.00.a
ps7_intc_dist_0	ps7_intc_dist	1.00.a
ps7_l2cachec_0	ps7_l2cachec	1.00.a
ps7_dma_s	ps7_dma	1.00.a
ps7_iop_bus_config_0	ps7_iop_bus_config	1.00.a
ps7_ram_0	ps7_ram	1.00.a
ps7_ram_1	ps7_ram	1.00.a
ps7_scugic_0	ps7_scugic	1.00.a
ps7_scutimer_0	ps7_scutimer	1.00.a
ps7_scuwdt_0	ps7_scuwdt	1.00.a
ps7_slcr_0	ps7_slcr	1.00.a
ps7_dma_ns	ps7_dma	1.00.a
ps7_afi_0	ps7_afi	1.00.a
ps7_afi_1	ps7_afi	1.00.a
ps7_afi_2	ps7_afi	1.00.a
ps7_afi_3	ps7_afi	1.00.a

Figure 6 – system.hdf Report on Hardware Specification

Experiment 2: BSP

Next, we will create a bare metal board support package, which Xilinx calls Standalone. This will assemble and compile various drivers that relate to the peripherals in the hardware platform for later use in our applications.

1. In SDK select **File → New → Board Support Package**.

2. Accept the default settings for the standalone BSP OS. Click **Finish**.

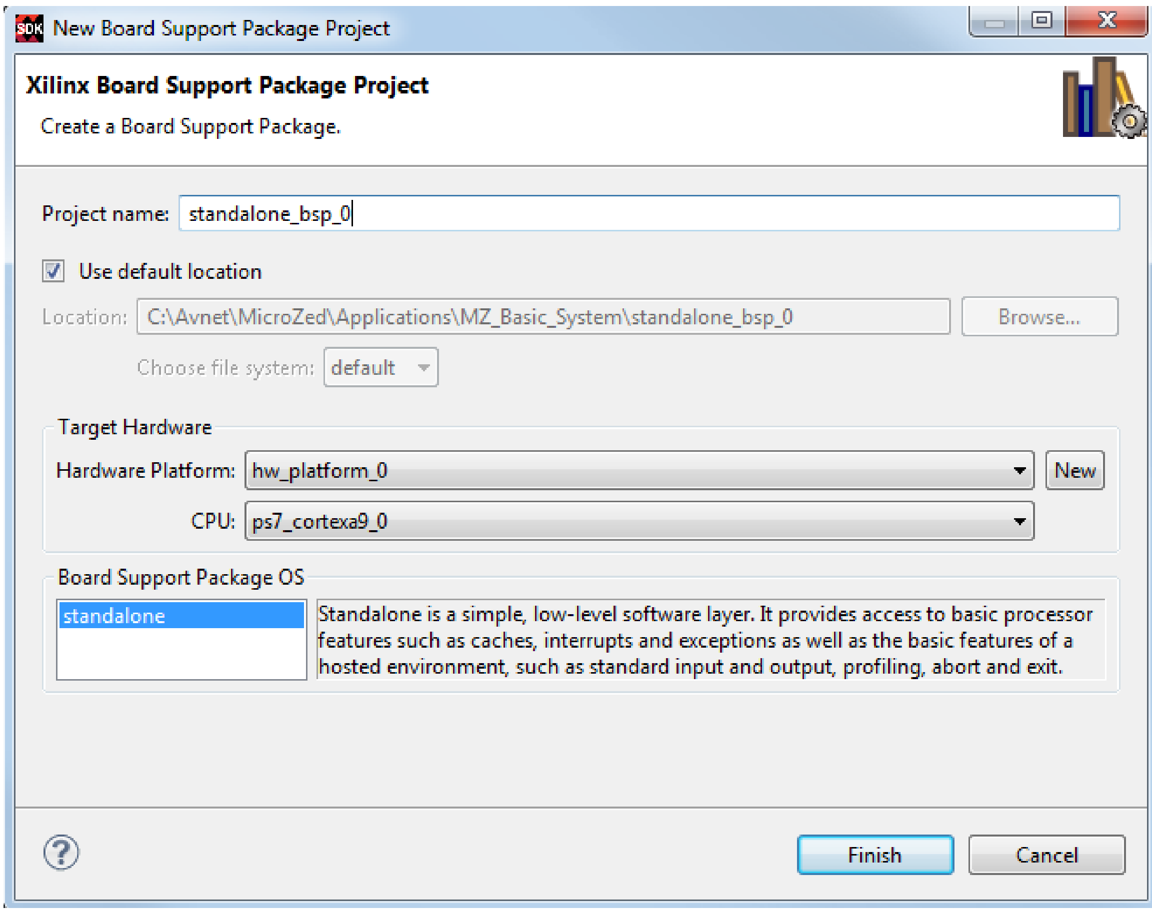


Figure 7 – Standalone BSP

3. In the *Board Support Package Settings*, accept the default settings. Click **OK**.

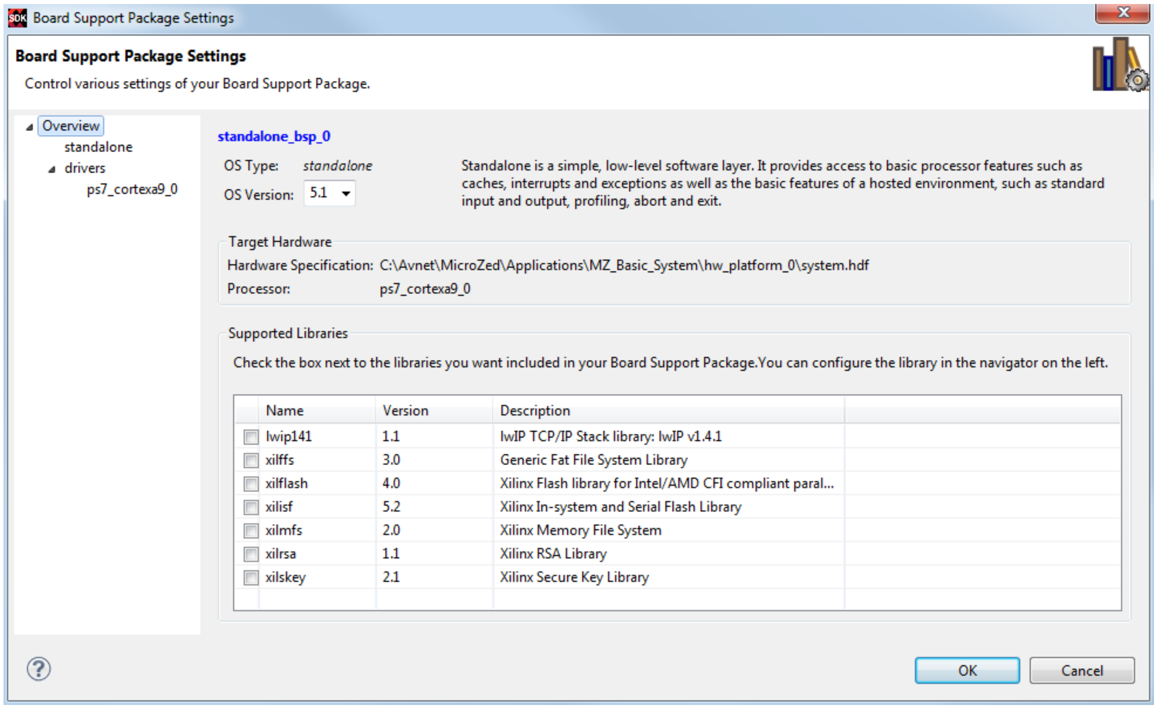


Figure 8 – Board Support Package Settings

Based on the default settings in SDK, the BSP will automatically be built once it is added to the project. This may take a minute to compile the new BSP. The progress may be seen in the *Console* tab.

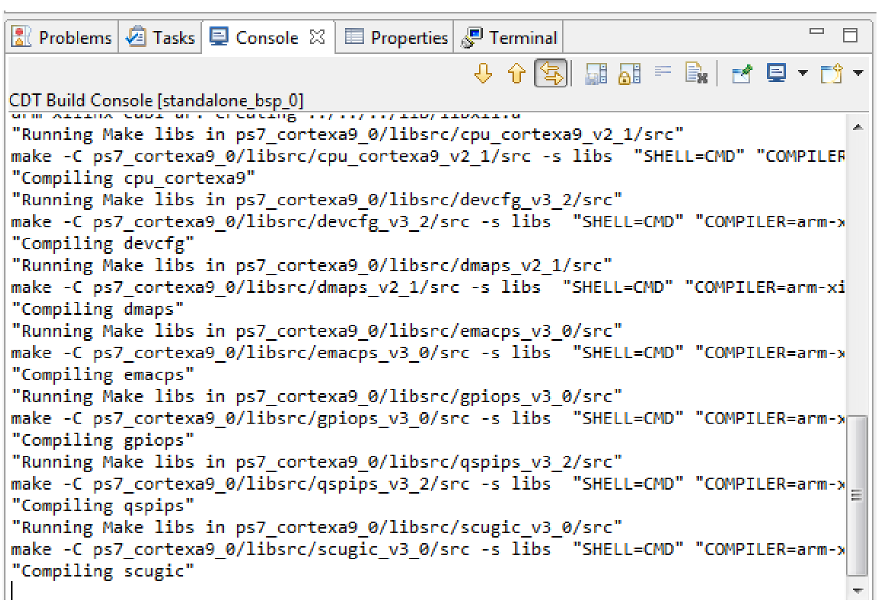


Figure 9 – BSP Building

The standalone_bsp_0 is now visible in the *Project Explorer*.

4. Expand **standalone_bsp_0** under the *Project Explorer*.

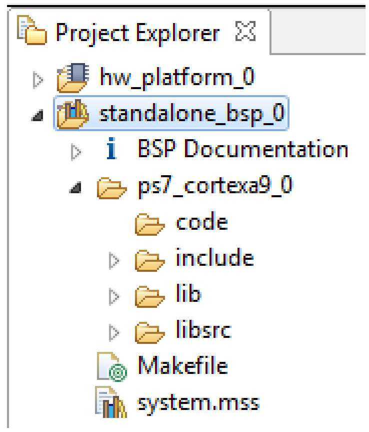


Figure 10 **– BSP Added to the Project**

Experiment 3: Add Application

With a Hardware Platform and BSP, we are now ready to add an application and run something on the board.

1. In SDK, select **File → New → Application Project**.

2. In the **Project Name** field type in `Hello_Zed`. Change the **BSP** to the existing StandAlone BSP. Click **Next >**.

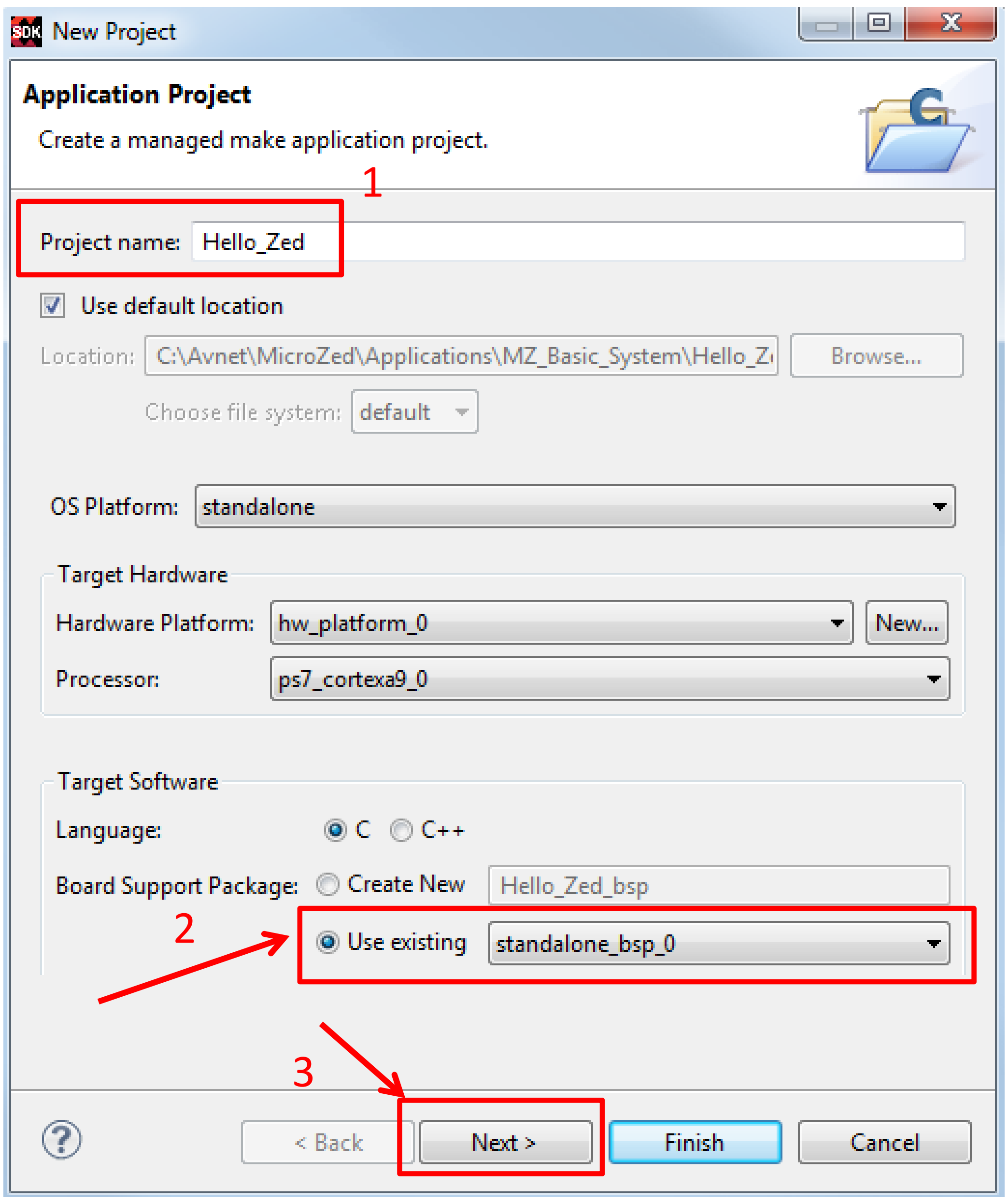


Figure 11 - New Application Wizard

3. Select **Hello World** from the *Available Templates* field. Click **Finish**.

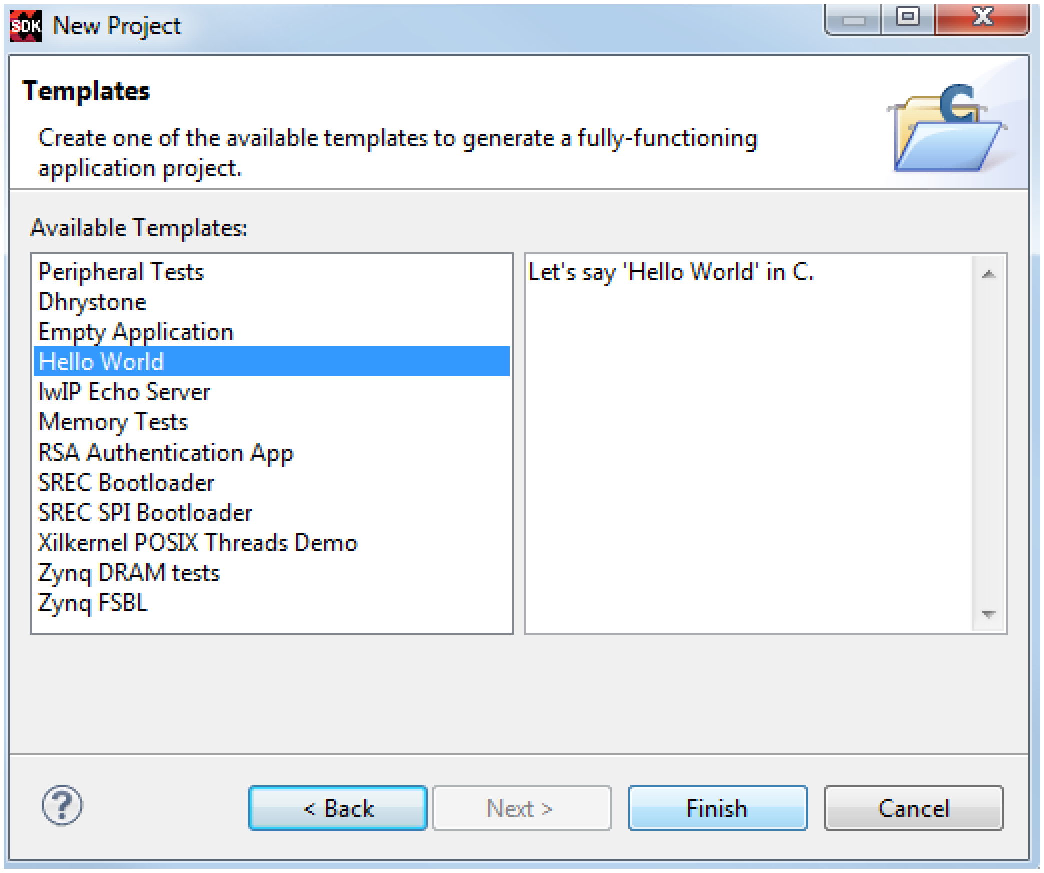


Figure 12 – New Application Project: Hello World

4. Notice that the Hello_Zed application is now visible in *Project Explorer*. By default, SDK will build the application automatically after it is added.

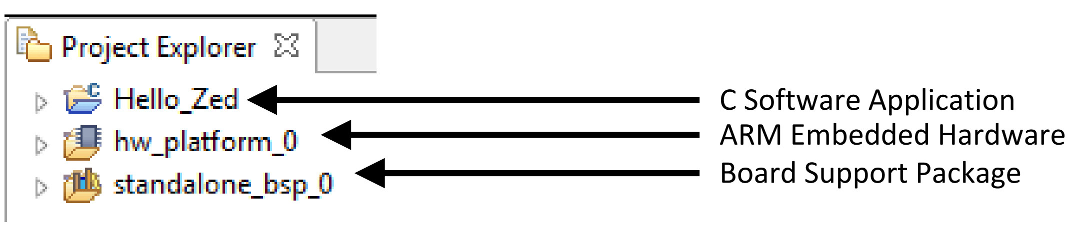


Figure 13 – Project Explorer View with Hello World C Application Added

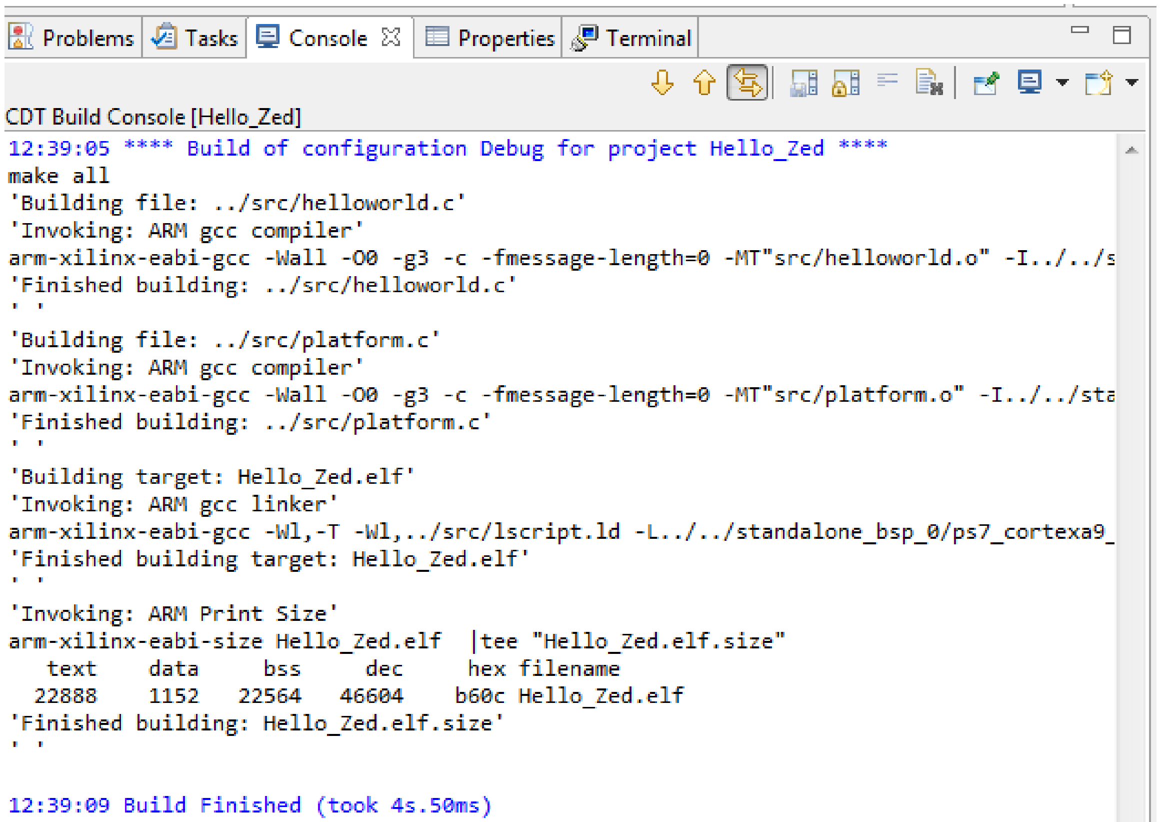


Figure 14 – Hello World Application Automatically Built

Experiment 4: Run on Hardware

1. Set the Boot Mode jumpers to Cascaded JTAG Mode.

    a. MicroZed: MIO[5:2] = GND. Set JP3, JP2, and JP1 to positions 1-2.

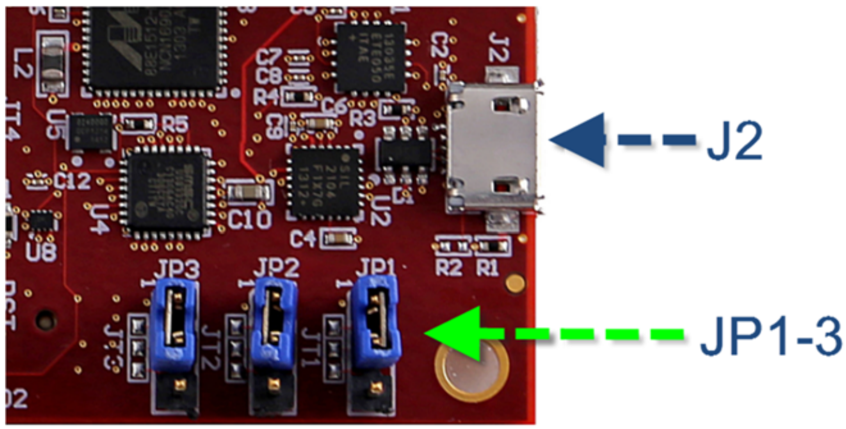


Figure 15 – Cascaded JTAG Boot Mode on MicroZed

    b. PicoZed: Set both switches on SW1 on the SOM away from the SW1 silkscreen

Neither MicroZed nor PicoZed has on-board USB JTAG programming. Thus it requires an external JTAG programmer, such as the Digilent HS3.

2. Connect a Platform Cable or Digilent Programming cable from your PC to the 2x7 JTAG socket.
    a. MicroZed:
        i. Use J3
    b. PicoZed FMC Carrier:
        i. Use J12 – PC4_JTAG

3. Power the board and connect a USB cable from your PC to the USB-UART port.
    a. MicroZed:
        i. Plug the micro-USB into J2.
        ii. The USB cable will power MicroZed. You should see the Green Power Good LED (D5) and the Red User LED (D3) light.

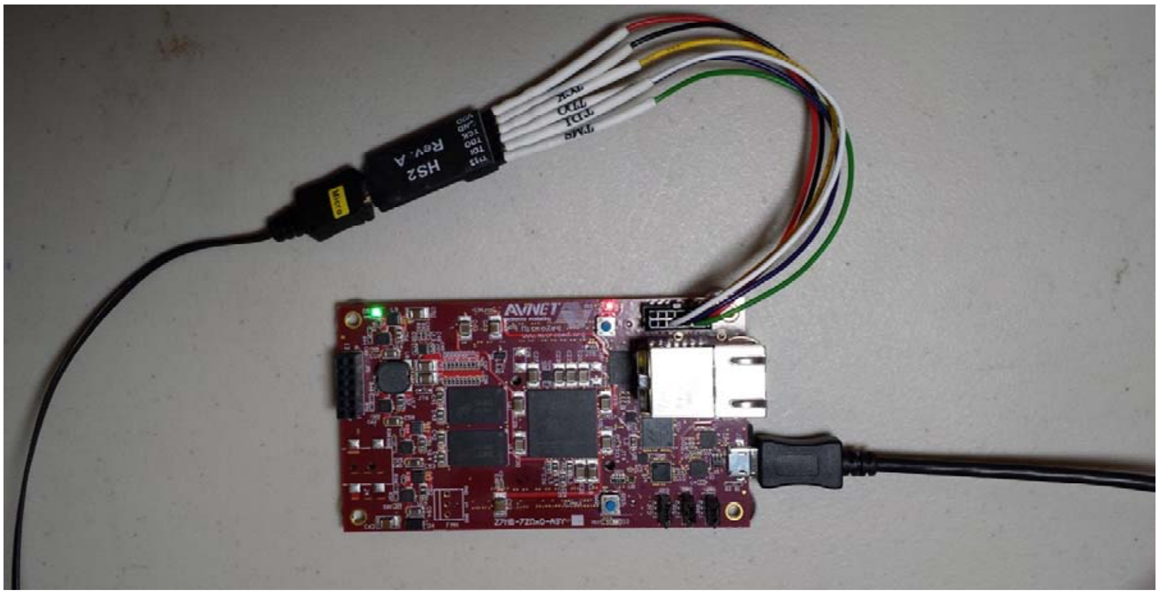


Figure 16 – MicroZed Powered and Connected to Digilent HS2 and USB-UART

b. PicoZed:
   i. Make sure the PZCC-FMC power switch (SW7) is OFF.
   ii. Insert the PicoZed module onto the PZCC-FMC.
   iii. Set the on-board jumpers as follows
      1. JP1 is open
      2. JP3 is closed in position 1-2
      3. JP4 is closed
      4. JP6 is open
      5. J9 is closed in positions 3-5 and 4-6
      6. CON2 is open, which sets V_ADJ to 1.8V
   iv. Insert the appropriate country plug into the 12V AC/DC adapter. Plug it into the J14 2x3 power connector. (NOTE – this 2x3 connector is NOT compatible with ATX power supplies.)
   v. Turn the PZCC-FMC power switch (SW7) to the ON position.
   vi. Plug in the micro-USB cable to PZCC-FMC USB-UART port (J6). (The reason for waiting until AFTER power is applied to the board is explained in the PZCC-FMC Errata.)
   vii. After 1-2 seconds, you will notice five LEDs that are lit:
      a. D1 (green) on PicoZed, indicating Power Good
      b. D19 (green) on PZCC-FMC, indicating Vin is on
      c. D14 (green) on PZCC-FMC, which is the PG_MODULE handshake between the SOM and the Carrier indicating that the SOM power is good
      d. D21 (blue) on PZCC-FMC indicating that the Zynq PL configuration is DONE
      e. D6 (amber) indicating the USB-UART is connected

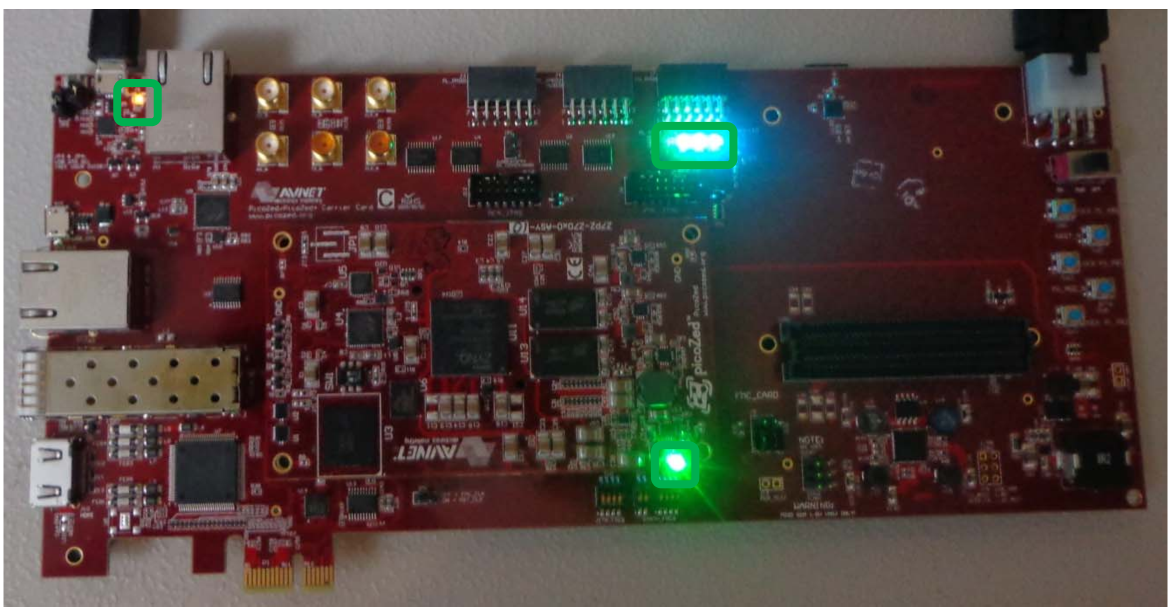


Figure 17 – PicoZed / PZCC-FMC Powered On with LEDs

If this is the first time you've connected the MicroZed, PicoZed, and/or the JTAG cable to this computer, you may see Windows install device drivers for the USB-UART and/or the JTAG cable. You should have previously installed the driver for the Silicon Labs CP2104 USB-UART. The Platform Cable and Digilent USB-JTAG drivers were installed during the Xilinx tool installation.

4. Use Device Manager to determine the COM port for the Silicon Labs CP201x USB-UART. In Windows 7, click **Start → Control Panel**, and then click **Device Manager**. Click **Yes** to confirm.

5. Expand ***Ports***. Note the COM port number for the SiLabs Serial device. This example shows COM4.

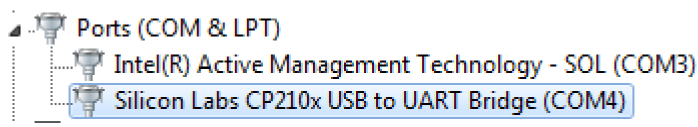


Figure 18 – Find the COM port number for the SiLabs USB-UART device

6. Open a serial communication utility for the COM port assigned on your system. SDK provides a serial terminal utility, which will be used throughout the tutorial; select **Window → Show View → Terminal** to open it.

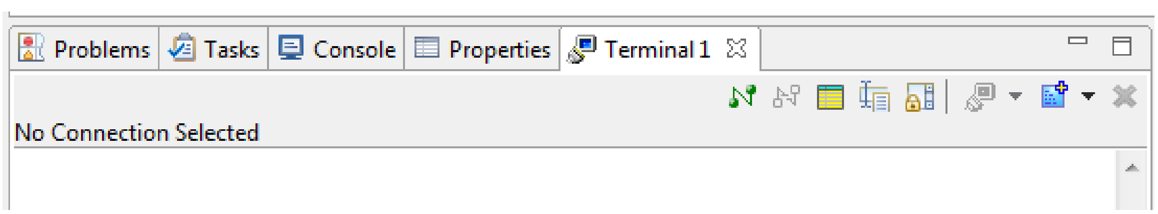


Figure 19 – Terminal Window Header Bar

7. Click the **Settings** button to open the Terminal Settings dialog box.
8. Change the settings as shown below. Click **OK**.

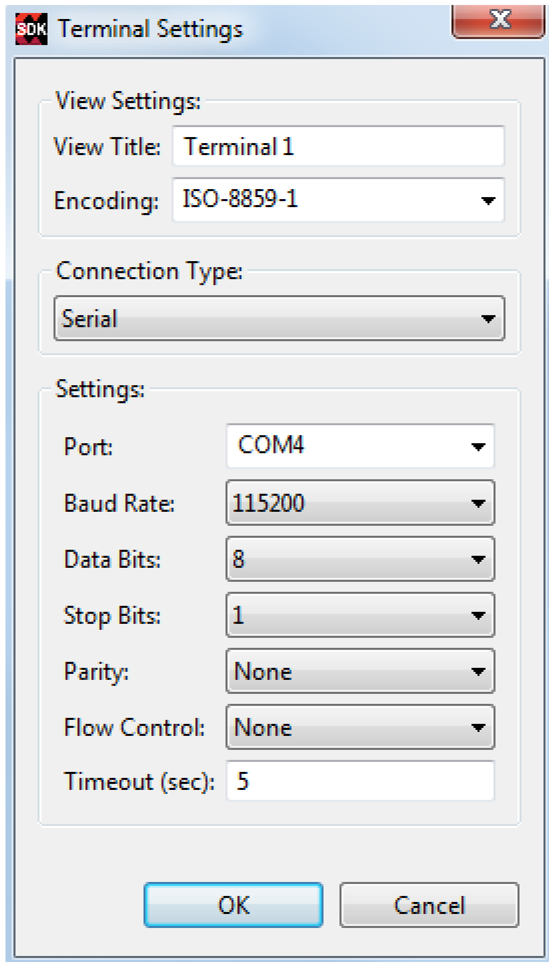


Figure 20 – Terminal Settings Dialog Box

Note, once the settings are set, clicking the **Connect** button will connect afterward.

9. Program the PL first by clicking the icon or selecting **Xilinx Tools → Program FPGA**. The default options are acceptable. Click **Program**. When complete, the Blue DONE LED should light.

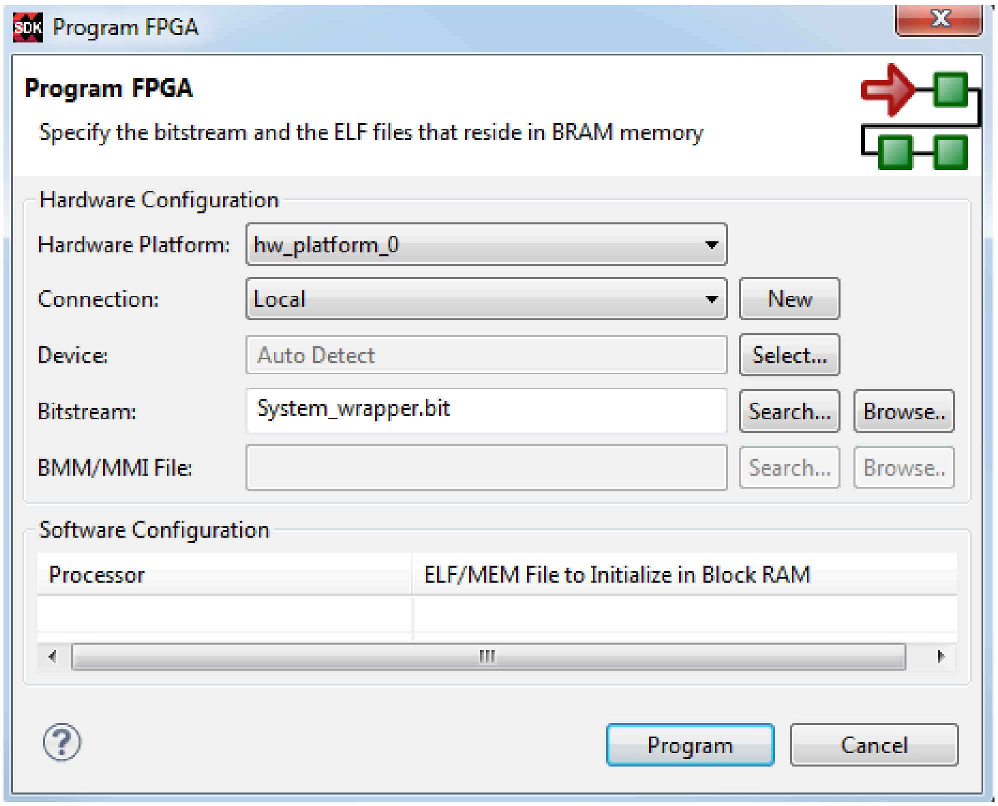


Figure 21 – Program FPGA

10. Right-click on the Hello_Zed application and select **Run As → Launch on Hardware (GDB)**. Notice that it is the 4th option down.

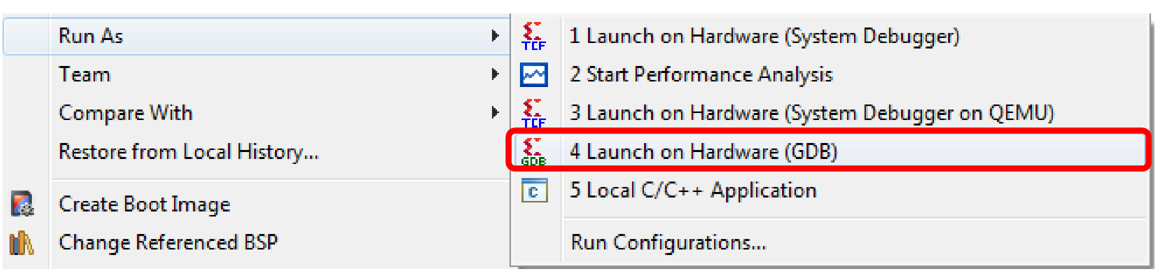


Figure 22 – Launch on Hardware (GDB)

11. The tools will now initialize the processor, download the Hello_Zed.elf to DDR, and then run Hello_Zed. This takes a few seconds to complete, depending on the USB traffic in your system. You can follow the progress in the lower right corner of SDK.

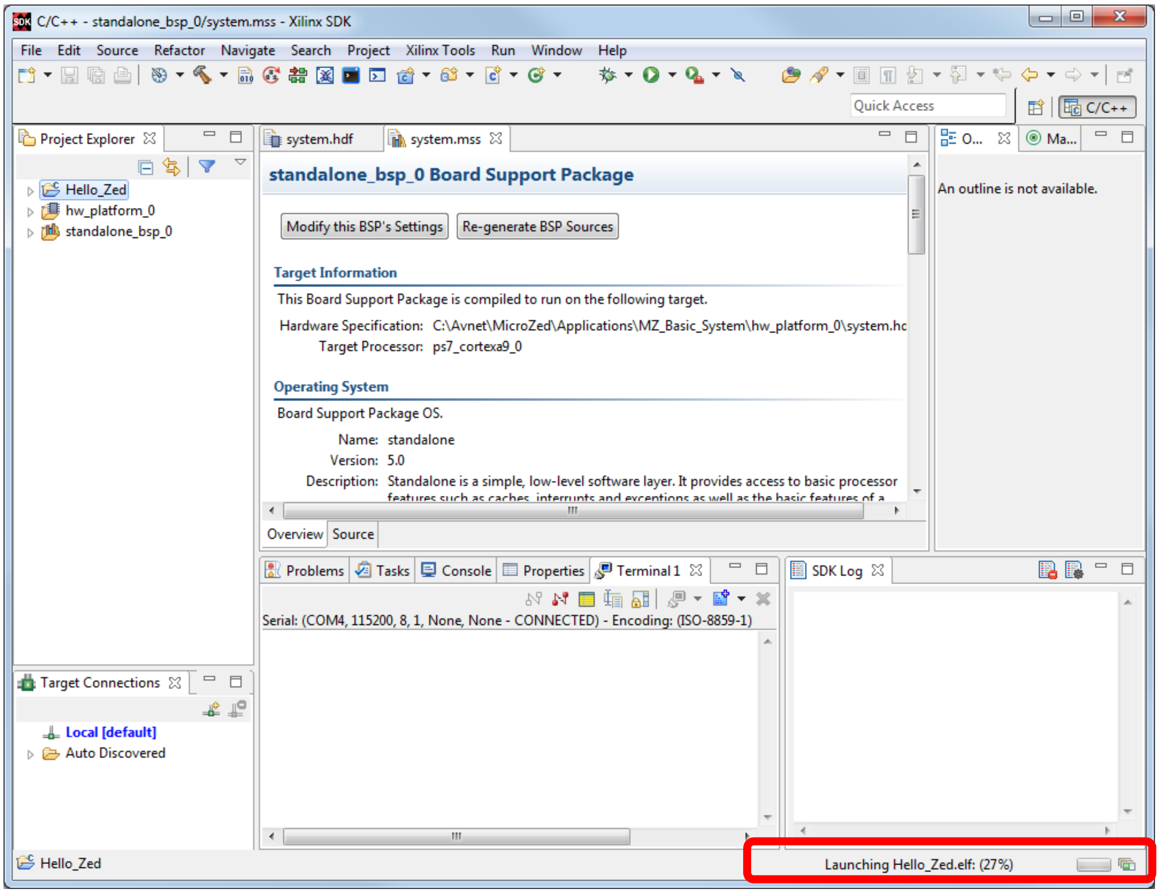


Figure 23 – Launching Hello_Zed Progress

SDK will download the Hello World ELF to the DDR3, and the ARM cpu0 begins executing the code. On MicroZed, you will notice that the Red User LED will go out, which is expected since that GPIO is now properly configured without a pull-up. The application standard output is displayed in the SDK Terminal. If SDK automatically switches to the *Console* tab, click on the *Terminal* tab to see the output.

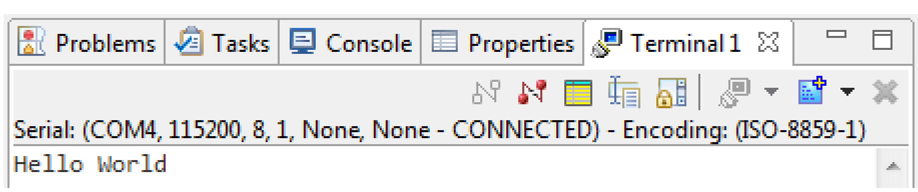


Figure 24 – Hello Zed Complete

You have now booted Zynq hardware on MicroZed or PicoZed!

Revision History

Date	Version	Revision
23 Aug 2013	2013_2.01	Initial Avnet release for Vivado 2013.2
09 Jun 2014	2014_1.01	Update to 2014.1
11 Jun 2014	2014_2.01	Update to 2014.2
29 Jun 2015	2015_1.01	Update to 2015.1. Add support for PicoZed.
15 Jul 2015	2015_2.01	Update to 2015.2.