Configuration and Booting

Introduction

This lab guides you through creating a bootable system capable of booting from the SD card or the QSPI flash memory located on the board. It also demonstrates how different bitstreams can be loaded in the PL section after the board is booted up and the corresponding application can be executed.

Objectives

After completing this lab, you will be able to:

- Create a bootable system capable of booting from the SD card
- Create a bootable system capable of booting from the QSPI flash
- Load the bitstream stored on the SD card or in the QSPI flash memory
- Configure the PL section using the stored bitstream through the PCAP resource
- Execute the corresponding application

Procedure

This lab is separated into steps that consist of general overview statements that provide information on the detailed instructions that follow. Follow these detailed instructions to progress through the lab.

Design Description

In this lab, you will design just the PS based embedded system consists of ARM Cortex-A9 processor SoC. The SDIO and QSPI interfaces are included in the base design.

The base design will then load the user selected design, consisting of both different hardware and software, and execute it. The following diagram represents the completed design (**Figure 1**).

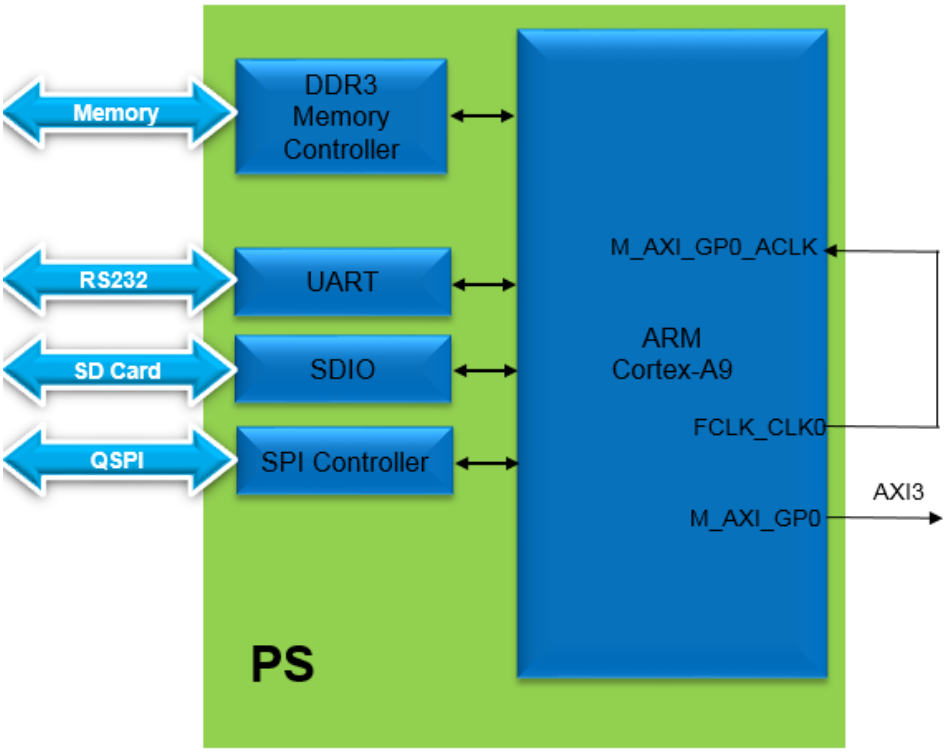
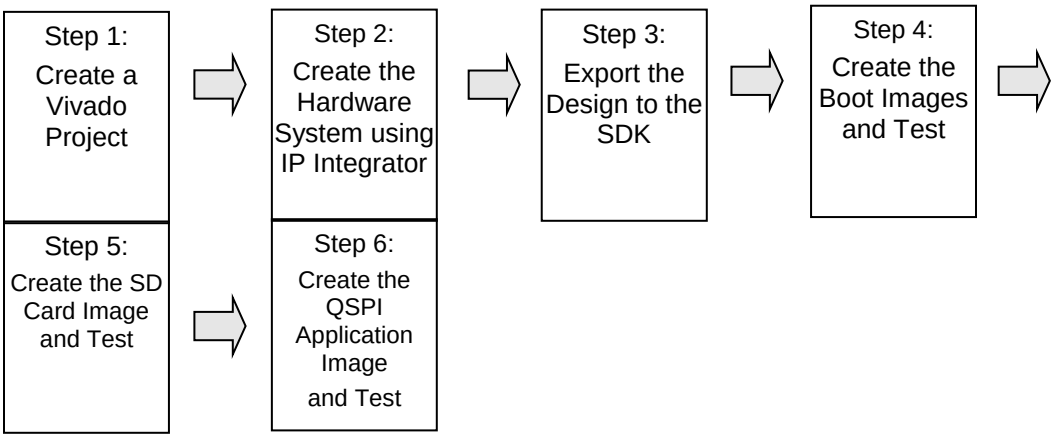


Figure 1. Completed Design

General Flow for this Lab



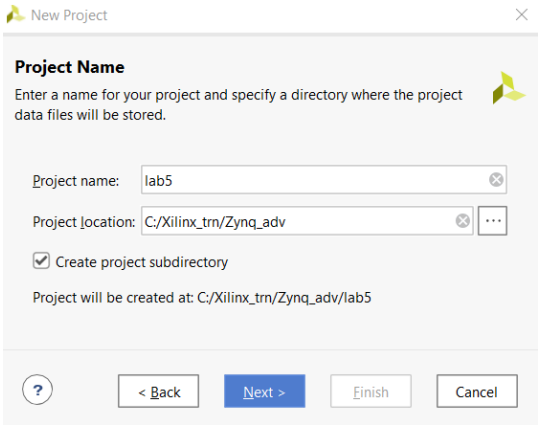
In the instructions below;
{***sources***} refers to: C:\Xilinx_trn\Zynq_adv\lab_sources
{***labs***} refers to : C:\Xilinx_trn\Zynq_adv

Create a Vivado Project Step 1

1-1. Launch Vivado and create an empty project, called lab5, using the Verilog language.

1-1-1. Open Vivado and click **Create New Project** and click **Next**.

1-1-2. Click the Browse button of the *Project Location* field of the **New Project** form, browse to **C:\Xilinx_trn\Zynq_adv**, and click **Select**.



1-1-3. Enter **lab5** in the *Project Name* field. Make sure that the *Create Project Subdirectory* box is checked. Click **Next**.

1-1-4. Select the **RTL Project** option in the *Project Type* form, and click **Next**.

1-1-5. Select **Verilog** as the *Target Language* in the *Add Sources* form, and click **Next**.

1-1-6. Click **Next**.

1-1-7. Click *Boards*, and select the **ZedBoard** and click **Next**.

1-1-8. Click **Finish** to create an empty Vivado project.

Creating the Hardware System Using IP Integrator Step 2

2-1. Create a block design in the Vivado project using IP Integrator to generate the ARM Cortex-A9 processor based hardware system.

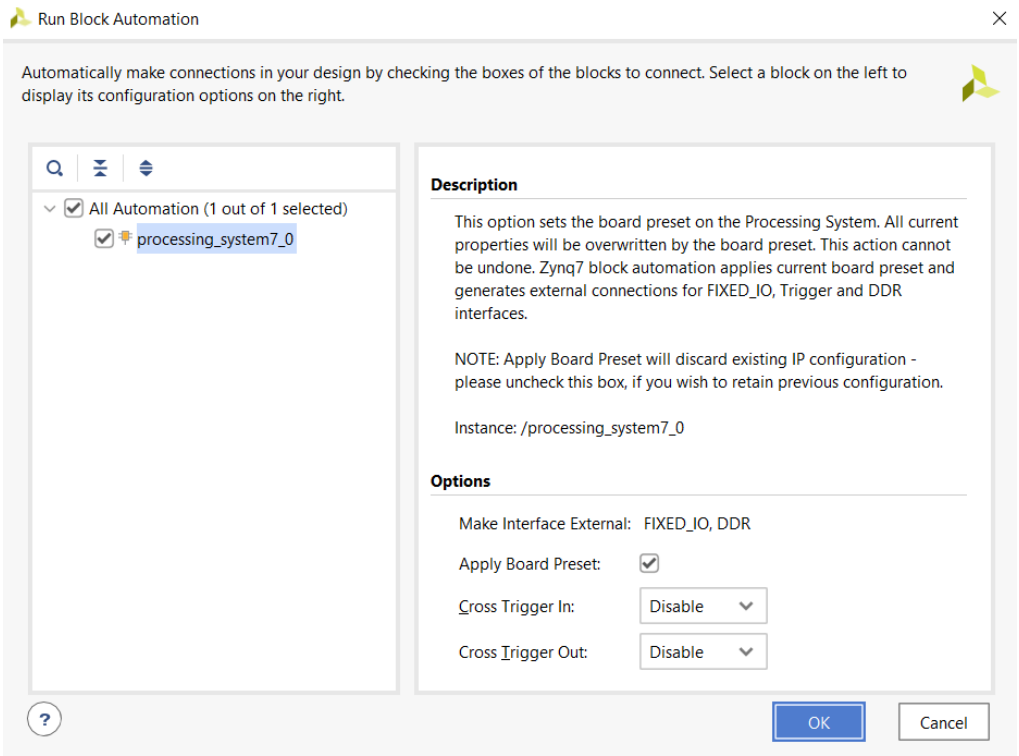
2-1-1. In the Flow Navigator, click **Create Block Design** under IP Integrator.

2-1-2. Name the block **system** and click **OK**.

2-1-3. In the *Diagram* panel, click the + button.

2-1-4. Once the IP Catalog is open, type ***zy*** into the Search bar, and double click on **ZYNQ7 Processing System** entry to add it to the design.

2-1-5. Click on *Run Block Automation* in the message at the top of the *Diagram* panel.



2-1-6. Leave the default option of *Apply Board Preset* checked, and click **OK**.

2-1-7. Double click on the Zynq block to open the *Customization* window.

A block diagram of the Zynq should now be open, showing various configurable blocks of the Processing System.

2-2. Configure the I/O Peripherals block to only have QSPI, UART 1, GPIO, EMIO with 1 bit, and SD 0 support. Deselect TTC device.

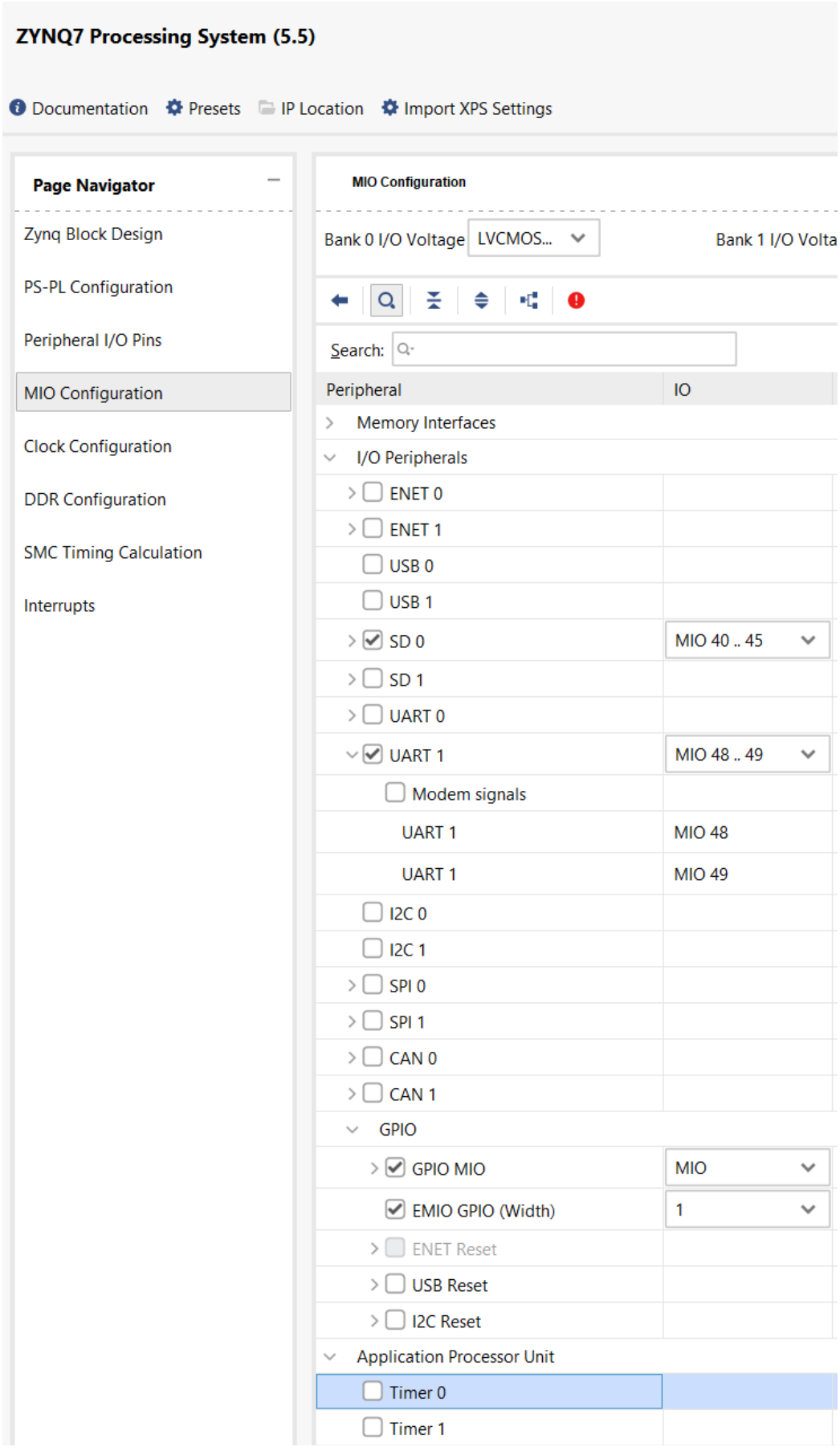
2-2-1. Click on the *MIO Configuration* panel to open its configuration form.

2-2-2. Expand the *IO Peripherals* on the right and uncheck *ENET 0*, *USB 0* **leaving *UART 1* and *SD 0* selected.**

2-2-3. Expand the *IO Peripherals>GPIO and* uncheck *USB Reset*, *I2C reset* **leaving *GPMIO* selected.**

2-2-4. Select EMIO GPIO and set it to 1.

2-2-5. Expand the *IO Peripherals>Application Processing Unit* and uncheck the **Timer 0**.


ZYNQ7 Processing System (5.5)
Documentation
Presets
IP Location
Import XPS Settings
Page Navigator
Zynq Block Design
PS-PL Configuration
Peripheral I/O Pins
MIO Configuration
Clock Configuration
DDR Configuration
SMC Timing Calculation
Interrupts
MIO Configuration
Bank 0 I/O Voltage
LVCMOS...
Bank 1 I/O Volta
Search:
Peripheral
IO
Memory Interfaces
I/O Peripherals
ENET 0
ENET 1
USB 0
USB 1
SD 0
MIO 40 .. 45
SD 1
UART 0
UART 1
MIO 48 .. 49
Modem signals
UART 1
MIO 48
UART 1
MIO 49
I2C 0
I2C 1
SPI 0
SPI 1
CAN 0
CAN 1
GPIO
GPIO MIO
MIO
EMIO GPIO (Width)
1
ENET Reset
USB Reset
I2C Reset
Application Processor Unit
Timer 0
Timer 1

2-2-6. Click **OK**.

The configuration form will close and the block diagram will be updated.

2-2-7. Using wiring tool, connect FCLK_CLK0 to M_AXI_GP0_ACLK. The block diagram will look as shown below.

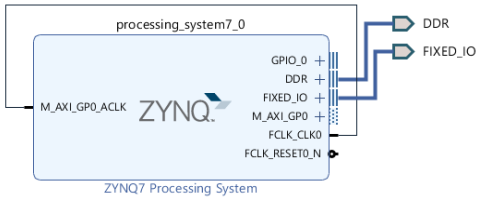


Figure 2. ZYNQ7 Processing System configured block

2-2-8. In the *Diagram* pannel click on the (Validate Design) button to make sure that there are no errors.

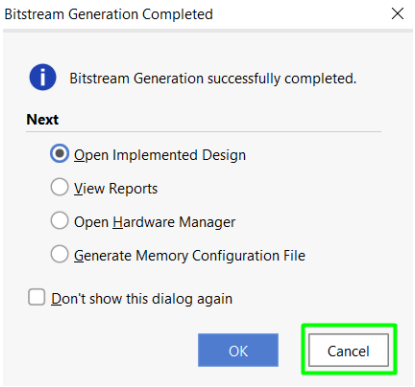
Export the Design to the SDK and create the software projects Step 3

3-1. Create the top-level HDL of the embedded system, and generate the bitstream.

3-1-1. In Vivado, select the *Sources tab*, expand the *Design Sources,* right-click the *system.bd* and select **Create HDL Wrapper** and click **OK.**

3-1-2. Click on **Generate Bitstream** and **Yes** if prompted to run the processes. Click **OK** to launch the runs.

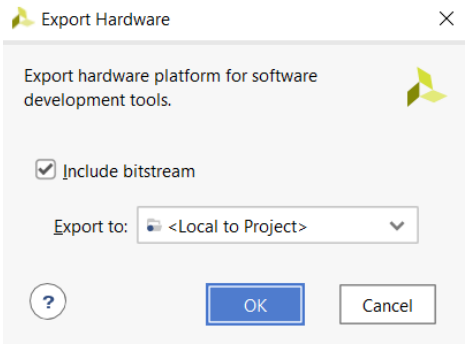
3-1-3. Wait a completion of the process.



3-1-4. When the bitstream generation process has completed successfully, click **Cancel**.

3-2. Export the design to the SDK and create the Hello World application.

3-2-1. Export the hardware configuration by clicking **File > Export > Export Hardware…**

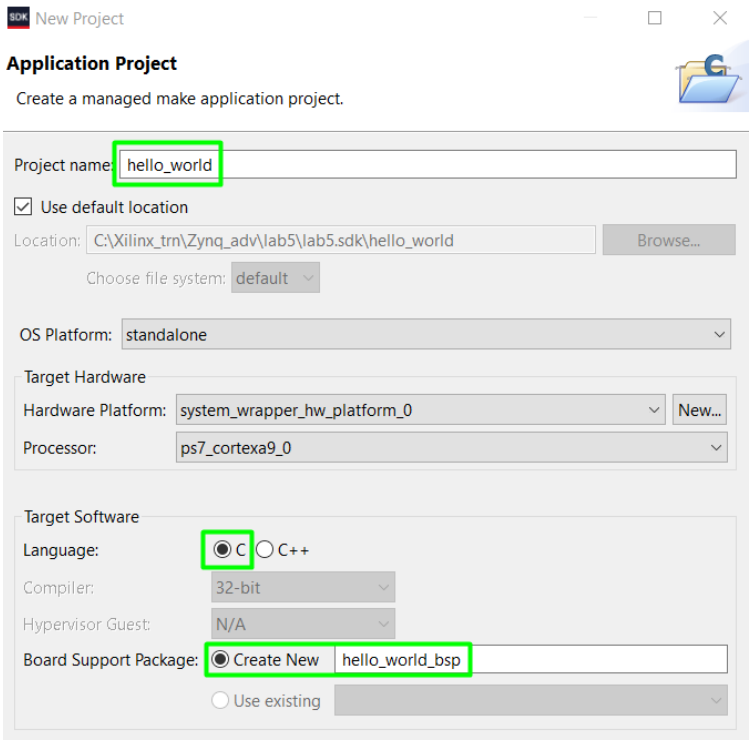


3-2-2. Click the box to *Include Bitstream*, then click **OK**

3-2-3. Launch SDK by clicking **File > Launch SDK** and click **OK**

3-2-4. In SDK, select **File** > **New** > **Application Project.**

3-2-5. Enter **hello_world** in the project name field, and leave all other settings as default.



3-2-6. Click **Next** and make sure that the *Hello World* application template is selected, and click **Finish** to generate the project *hello_world* and board support package *hello_world_bsp*.

3-2-7. Right click on hello_world_bsp and click **Board Support Package Settings**

3-2-8. Tick to include *xilffs* click **OK** (This is required for the next step to create the FSBL).

3-3. Create a first stage bootloader (FSBL).

3-3-1. Select **File** > **New > Application Project.**

3-3-2. Enter **zynq_fsbl** as the project name, select the *Use existing* standalone Board Support Package option with **hello_world_bsp**.

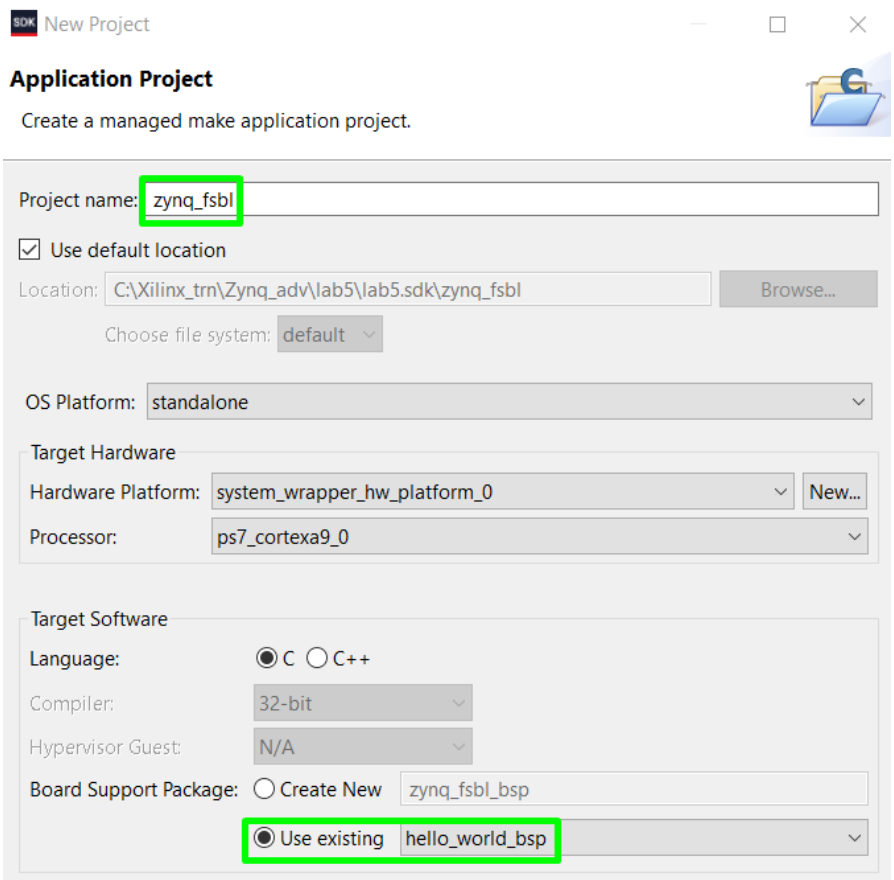


Figure 3. Creating the FSBL Application

3-3-3. Click **Next.**

3-3-4. Select *Zynq FSBL* in the **Available Templates** pane and click **Finish**.

A *zynq_fsbl* project will be created. It will be used in creating the BOOT.bin file.

The BOOT.bin file will be stored on the SD card which will be used to boot the board.

Create the Boot Images and Test Step 4

4-1. You will create the BOOT.bin file using the FSBL, system_wrapper.bit and hello_world.elf files.

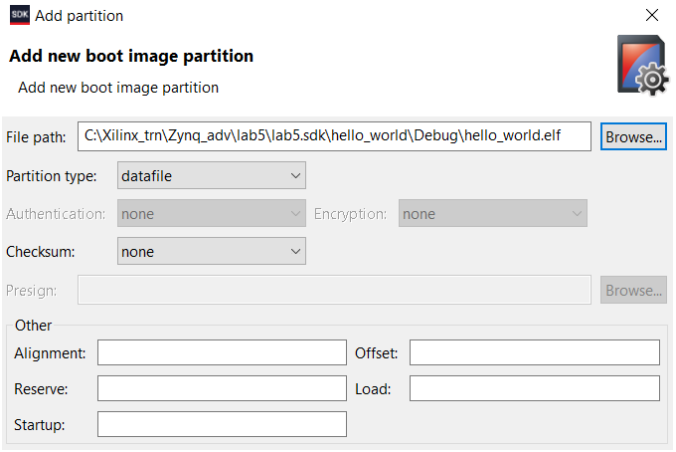
4-1-1. In the SDK, right click on zynq_fsbl and select **Create Boot Image**.

4-1-2. In the window appeared pay attention on *Boot image partitions* pane and make sure that

the First Boot Loader**, zynq_fsbl.elf**, file from **C:\Xilinx_trn\Zynq_adv\lab5\lab5.sdk\zynq_fsbl\Debug** directory (this is where the FSBL was created) and the bitstream, **system_wrapper.bit**, file from **C:\Xilinx_trn\Zynq_adv\lab5\lab5.sdk\system_wrapper_hw_platform_0** were added automatically.

4-1-3. Click on the **Add** button of the *Boot image partitions,*

4-1-4. Click the Browse button in the Add Partition form, browse to **C:\Xilinx_trn\Zynq_adv\lab5\lab5.sdk\hello_world\Debug** and add the software application, **hello_world.elf**.



4-1-5. Click **OK**.

4-1-6. Check settings comparing it with the following figure.

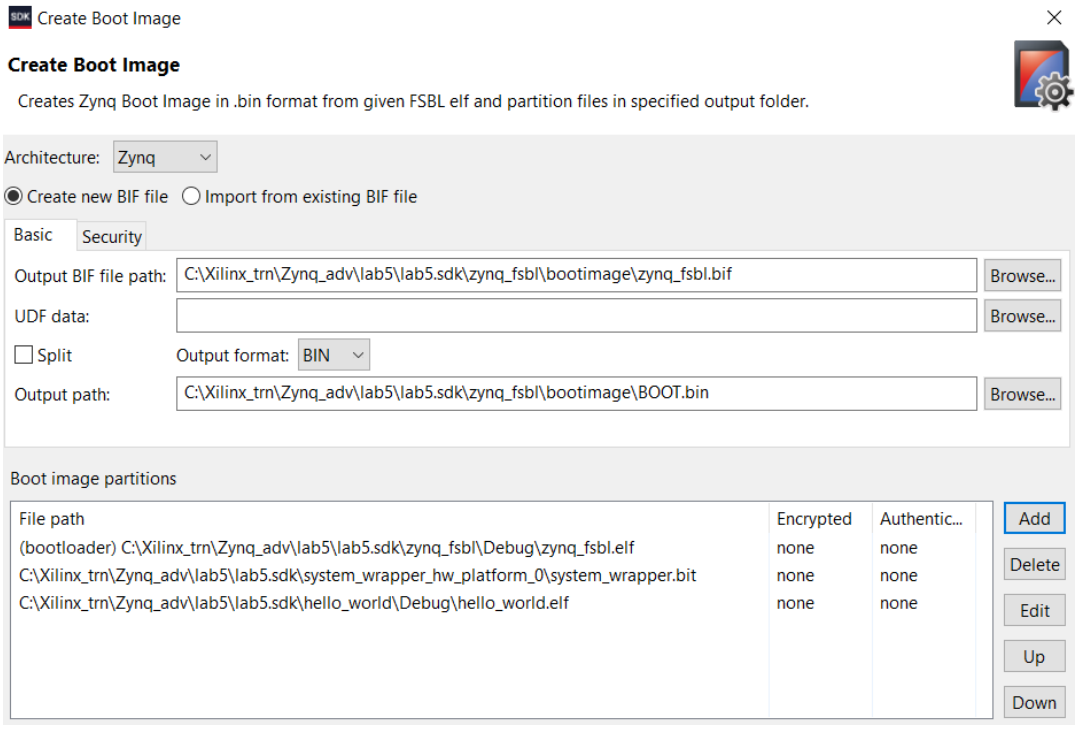


Figure 5. Creating BOOT.bin image file

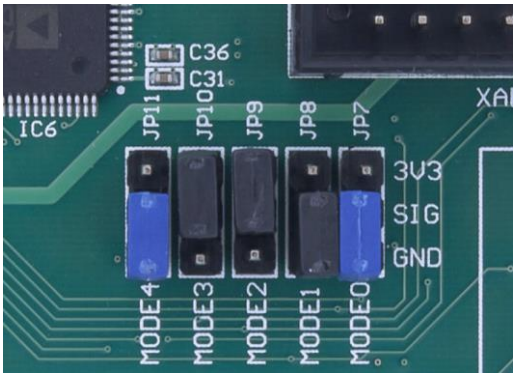
4-1-7. Click the **Create Image** button.

The *BOOT.bin* and the *zynq_fsbl.bif* files will be created in the **C:\Xilinx_trn\Zynq_adv\lab5\lab5.sdk\zynq_fsbl\bootimage** folder. We will use the BOOT.bin for the SD card boot up.

4-1-8. Insert a blank SD/MicroSD card (FAT32 formatted) in a Card reader, and using the Windows Explorer, copy the BOOT.bin file from the **C:\Xilinx_trn\Zynq_adv\lab5\lab5.sdk\zynq_fsbl\bootimage** folder in to the SD/MicroSD card.

4-2. Set the board in SD card boot mode. Test the functionality by starting a Terminal emulator program and powering ON the board.

4-2-1. Set the board in the SD card boot mode (For Zedboard, set the mode pins JP11- JP7 as GND-SIG, SIG-3V3, SIG-3V3, GND-SIG, GND-SIG).



4-2-2. Insert the SD/MicroSD card into the board.

4-2-3. Connect your PC to the UART port with the provided micro-USB cable.

4-2-4. Power ON the board.

4-2-5. Start Terminal or SDK Terminal in SDK or a Terminal emulator program setting it to the current COM port and 115200 baud rate.

4-2-6. Press the PS_RST (Red button) button on the board.

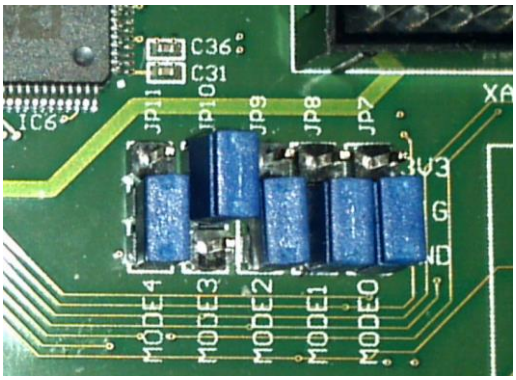
4-2-7. You should see the **Hello World** message in the terminal emulator window. If you don't see it, then press the PS_RST (right Red button) button on the board.

4-2-8. Once satisfied: disconnect Terminal and power OFF the board.

4-2-9. Remove the SD card.

4-3. Set the board in the QSPI boot mode. Power ON the board, Program the QSPI using the Flash Writer utility, and test by pressing PS-RST button.

4-3-1. Set the board in the QSPI mode (For Zedboard set the mode pins JP11- JP7 as GND-SIG SIG-3V3, GND-SIG, GND-SIG, GND-SIG).



4-3-2. Connect your PC to the UART port with the provided micro-USB cable.

4-3-3. Power ON the board.

4-3-4. Start a Terminal emulator program setting it to the current COM port and an 115200 baud rate.

4-3-5. Select **Xilinx > Program Flash**.

4-3-6. In the *Program Flash Memory* window, in the field Image File, click the **Browse** button, and browse to the **C:\Xilinx_trn\Zynq_adv\lab5\lab5.sdk\zynq_fsbl\bootimage** folder, select **BOOT.bin** file, and click **Open**.

4-3-7. In the *Program Flash Memory* window, in the *Offset* field enter **0.**

4-3-8. In the *Program Flash Memory* window, in the field FSBL File, click the **Browse** button, and browse to the **C:\Xilinx_trn\Zynq_adv\lab5\lab5.sdk\zynq_fsbl\Debug** folder, select **zynq_fsbl.elf** file, and click **Open**.

4-3-9. Check box *Blank check after erase*.

4-3-10. Check box *Verify after flash*.

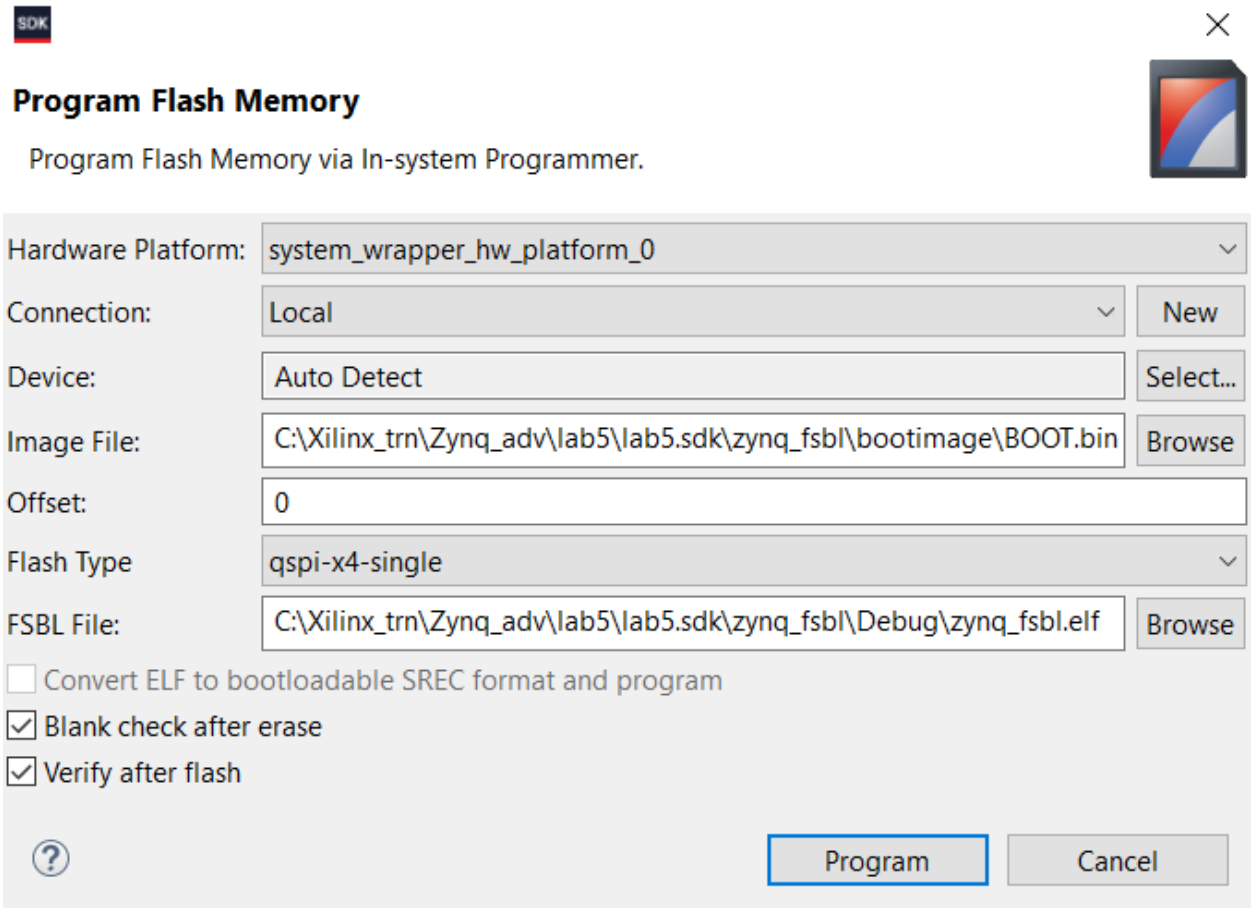
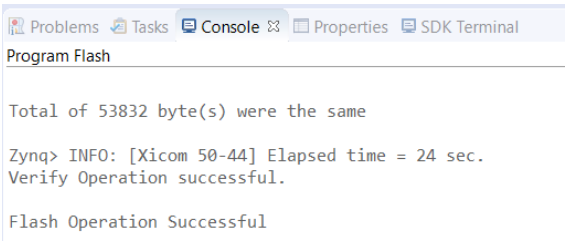


Figure 6. Program Flash Memory form

4-3-11. Click the **Program** button.

The QSPI flash will be programmed.



4-3-12. Connect the Terminal window.

4-3-13. Press the PS-RST (on the ZedBoard) (right Red button) button on the board to see the **Hello World** message in the terminal emulator window.

4-3-14. Once satisfied: disconnect Terminal and power OFF the board.

Conclusion

This lab led you through creating the boot images, which were capable of booting standalone applications from either the SD card or the QSPI flash memory.