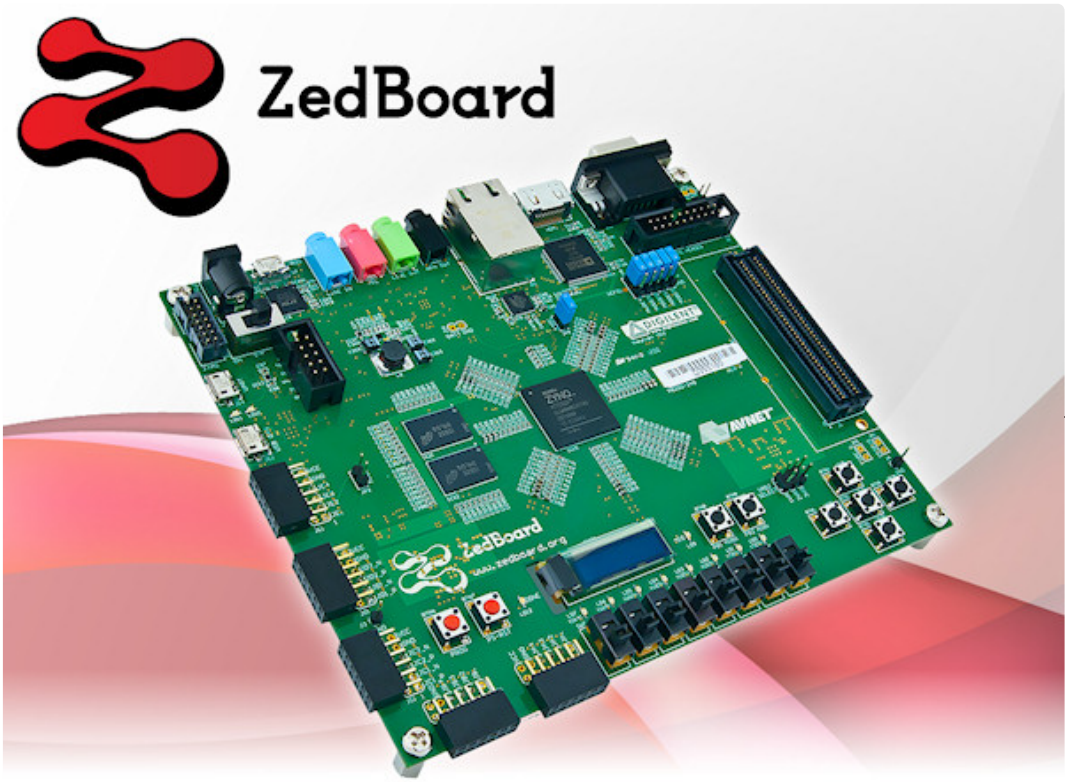
Comments for vivado 2020.1 and vitis 2020.1

Getting Started with Zynq



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/zedboard-obl-bg-600.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

Overview

This guide will provide a step by step walk-through of creating a hardware design using the Vivado IP Integrator for the Zedboard.

At the end of this tutorial you will have:

- Created a simple hardware design incorporating the on board LEDs and switches.
- Created a .C project in XIlinx Vivado SDK tieing the on board LEDs and switches together using the hardware design shown in the previous step.

Prerequisites

Hardware

- **Digilent's Zedboard Development Board and a Micro USB cable for UART communication and JTAG programming**

Software

- **Xilinx Vivado 2015.X with the SDK package.**

Board Support Files

- **Zedboard Support Files**
  - *These files will describe GPIO () interfaces on your board and make it easier to select your board in the initial design setup and add GPIO () IP blocks in the block design*
  - *Follow this Wiki guide Vivado Board Files for Digilent 7-Series FPGA Boards (https://reference.digilentinc.com/vivado/boardfiles) on how to install Board Support Files for Vivado 2015.X*

General Design Flow

I. Vivado

- Open Vivado and select Zedboard
- Create an new Vivado Project
- Create empty block design workspace inside the new project

- Add required IP blocks using the IP integrator tool and build Hardware Design
- Validate and save block design
- Create HDL system wrapper
- Run design Synthesis and Implementation
- Generate Bit File
- Export Hardware Design including the generated bit stream file to SDK tool
- Launch SDK

Now the Hardware design is exported to the SDK tool. The Vivado to SDK hand-off is done internally through Vivado. We will use SDK to create a Software application that will use the customized board interface data and FPGA hardware configuration by importing the hardware design information from Vivado.

II. SDK

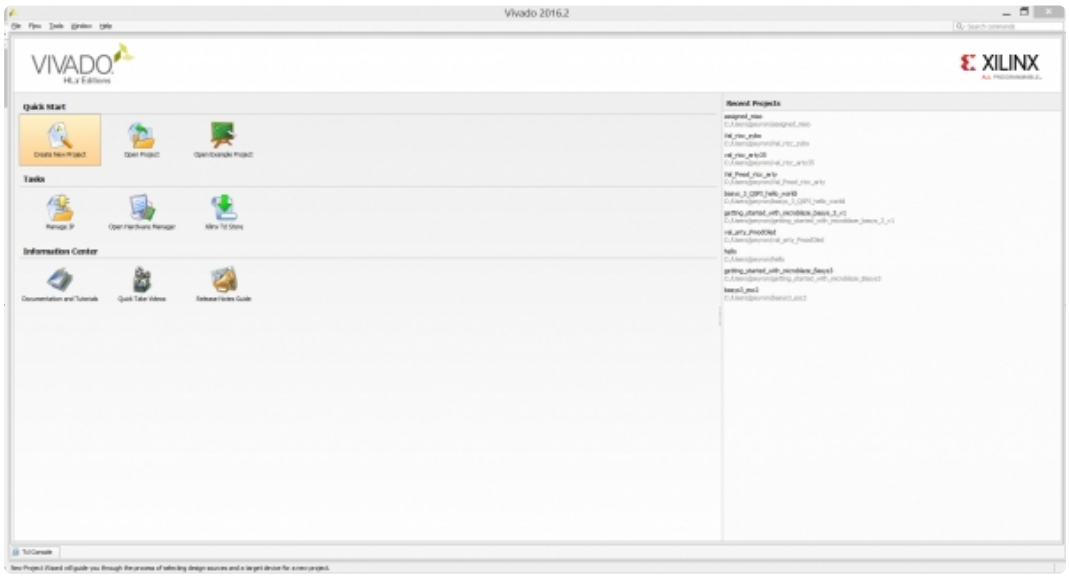
- Create new application project and select default Hello World template
- Program FPGA and run application

—

1. Creating a New Project

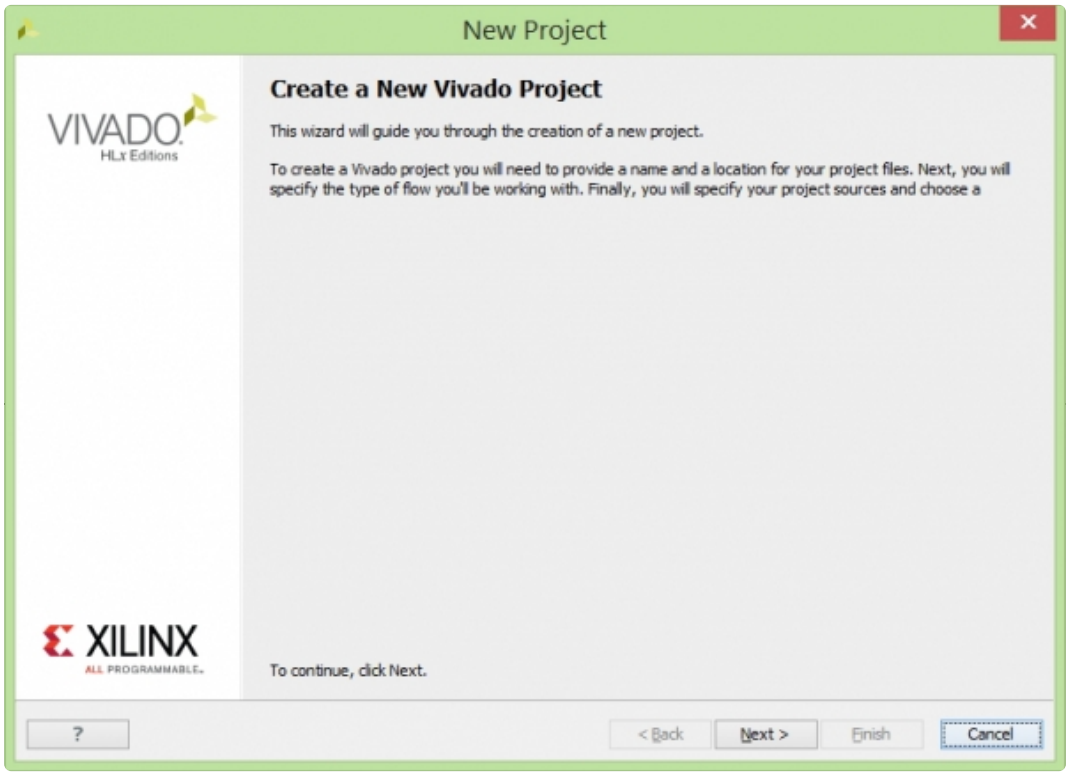
When you first run Vivado this will be the main start window where you can create a new project or open a recent one.

1.1) Click on **Create New Project**.



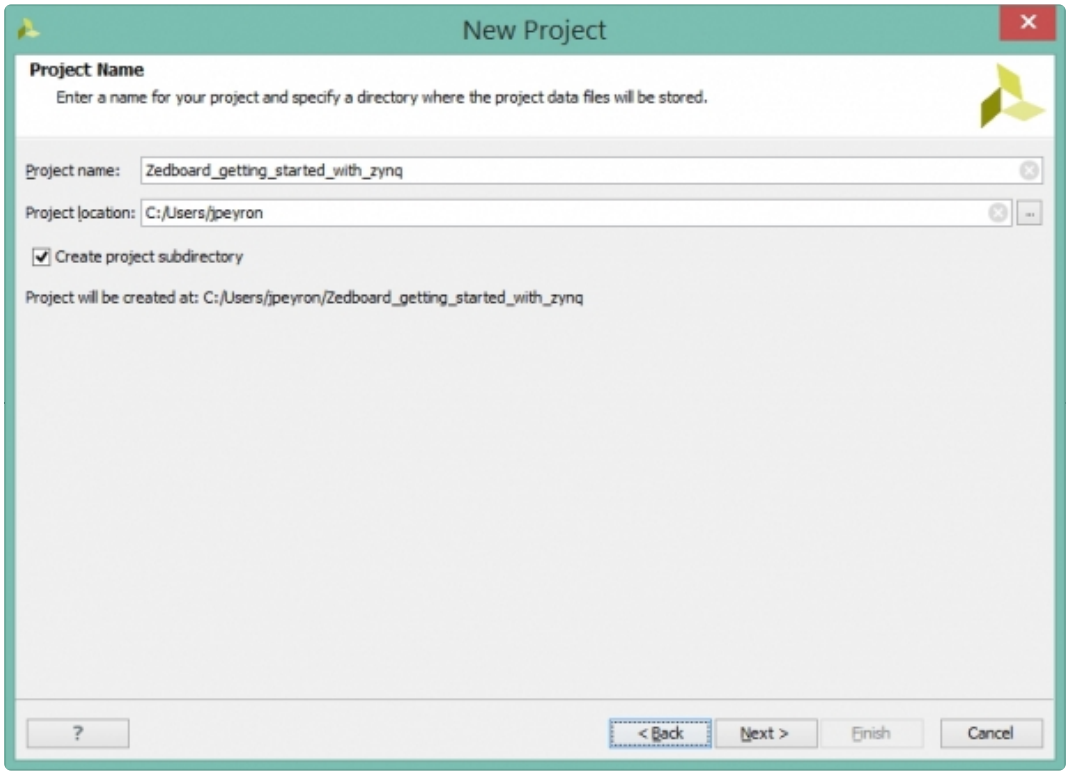
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_1.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

1.2) You will be presented with the project creation wizard. Click **Next**.



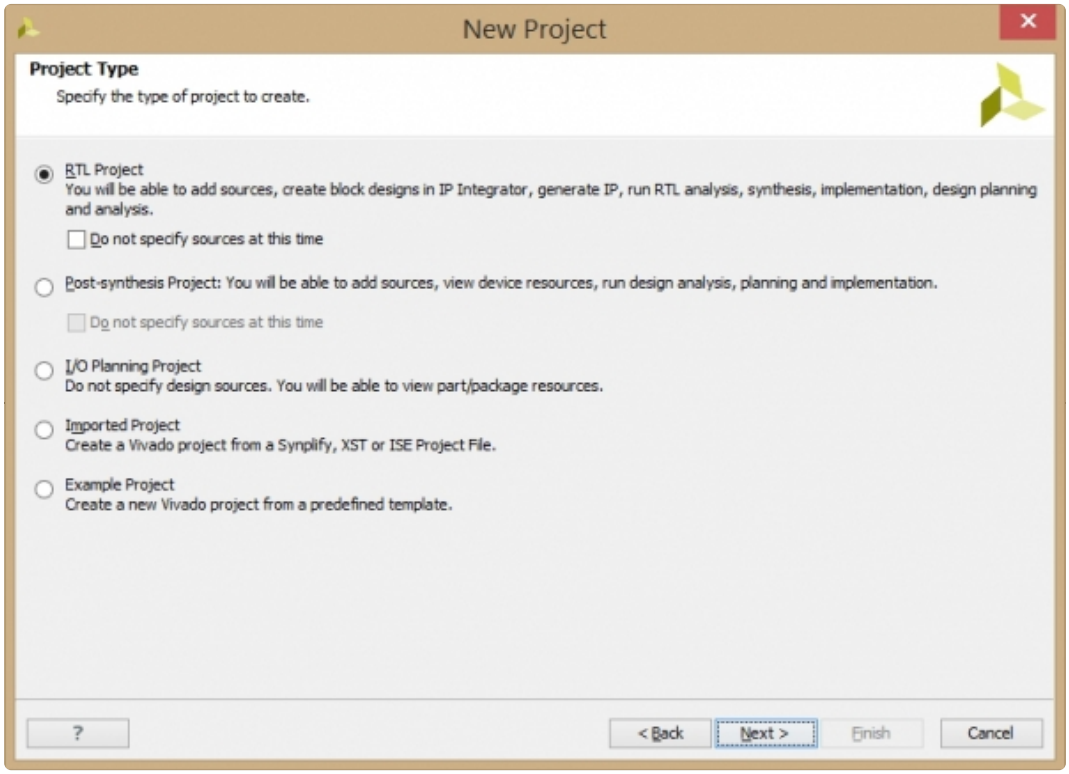
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_2.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

1.3) Enter a project name and location the click **Next**.



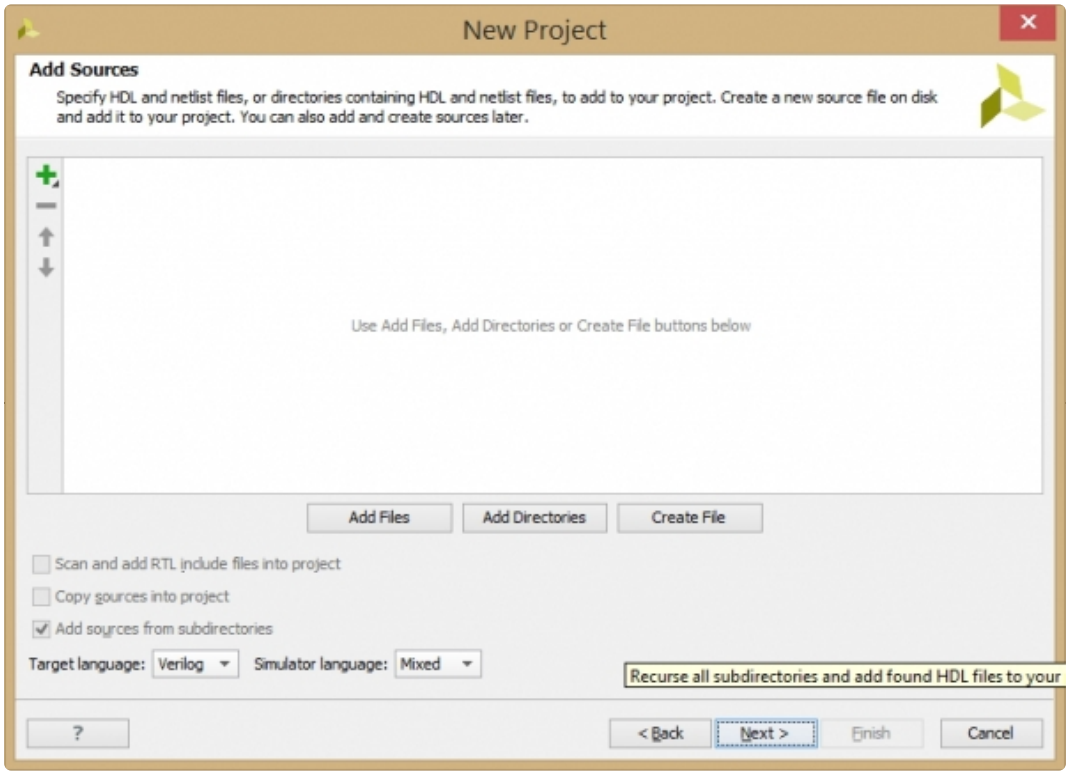
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_3.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

1.4) Select **RTL Project** and click **Next**.



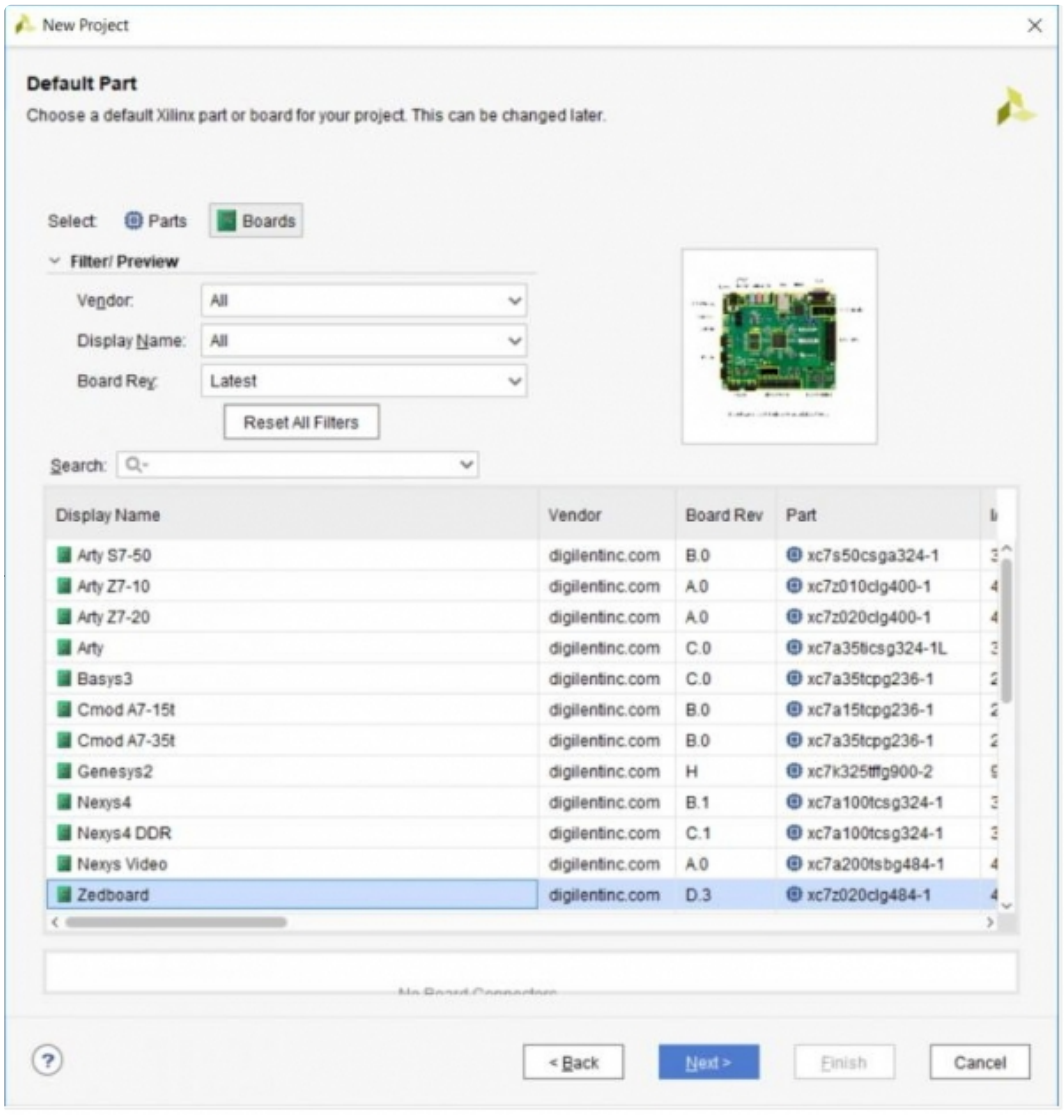
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_4.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

1.5) This demo does not use any existing sources, existing IP or constraints. Click through the next three screens.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_5.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

1.6) Select **Boards** and select the **Zedboard** board file. Click **Next** and then **Finish**.

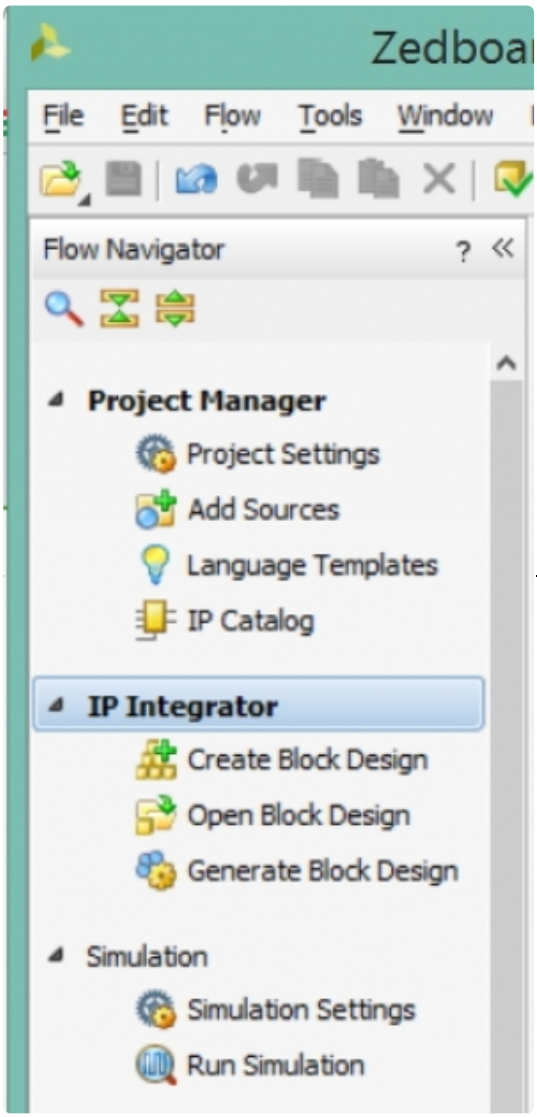


(https://reference.digilentinc.com/_detail/learn/programmable-logic/tutorials/zedboard-getting-started-with-zynq/board_file_zedboard.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

Make sure to select the board file made by Digilent.

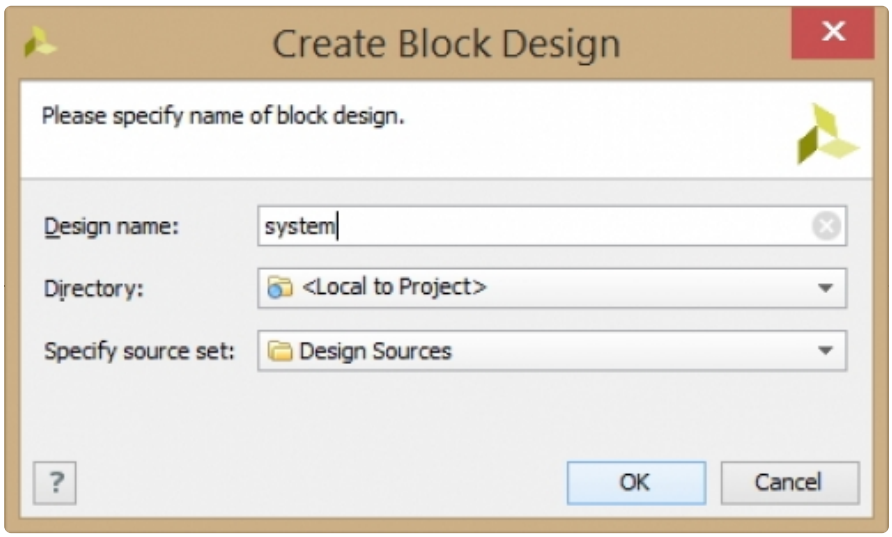
2. Creating a New Block Design

2.1) Once the process has completed, click **Create Block Design** in the flow navigator.



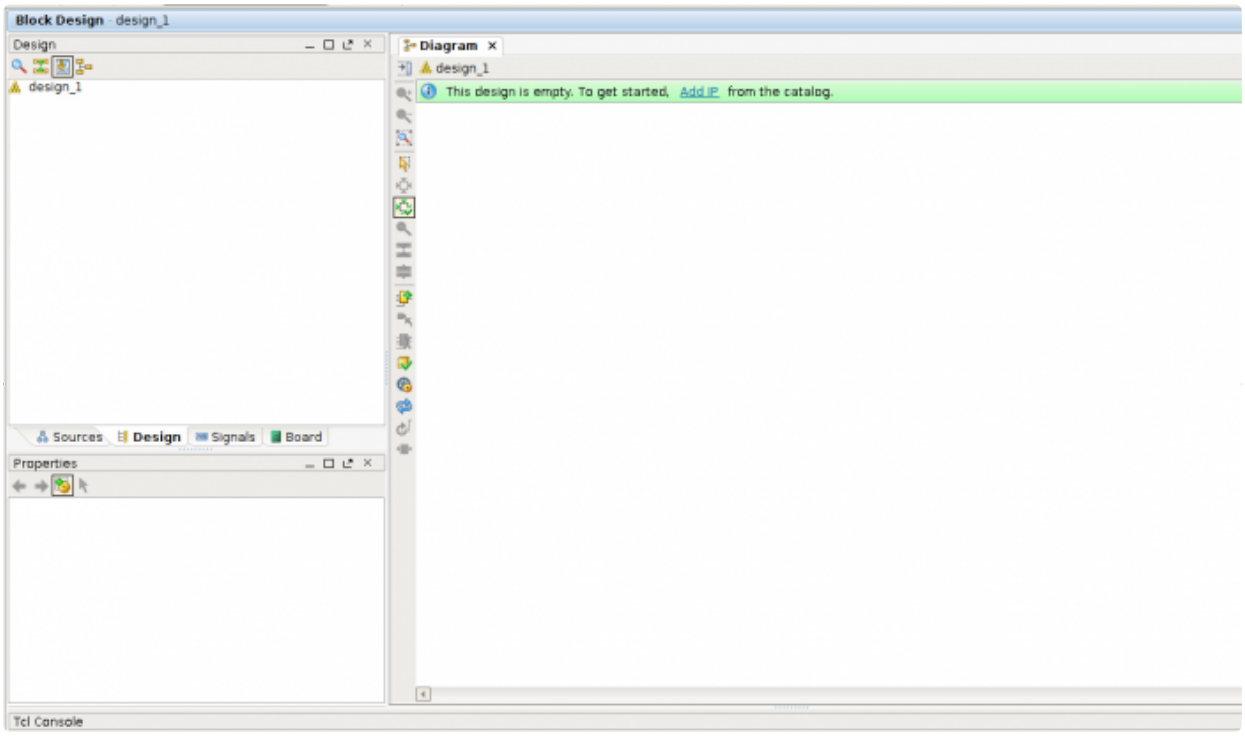
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_7.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

2.2) Click **OK**.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_8.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

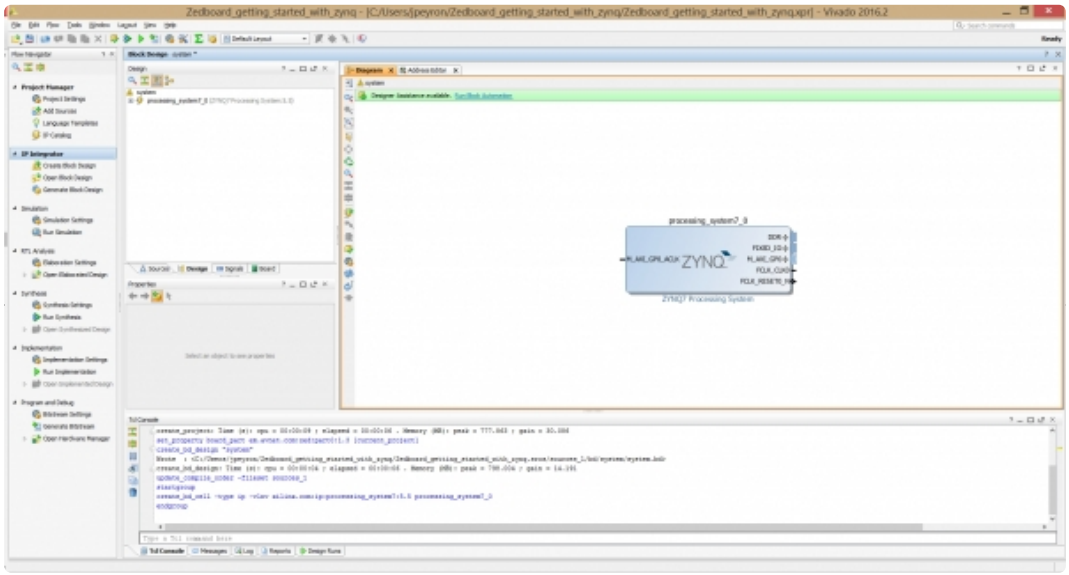
2.3) A blank Block Design will open up.



(https://reference.digilentinc.com/_detail/zybo/blank_block_design.png?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

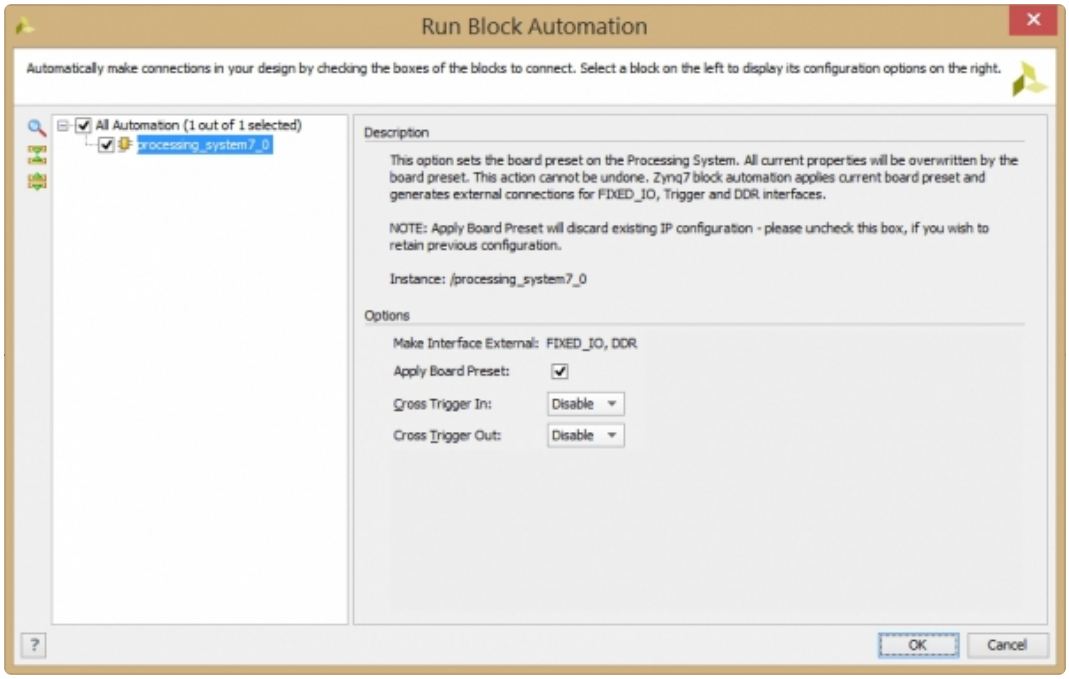
3. Add the Zynq IP & GPIO Blocks

3.1) Click the **Add IP** button and search for ZYNQ. Double click on **ZYNQ7 Processing System** to place the bare Zynq block.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_9.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

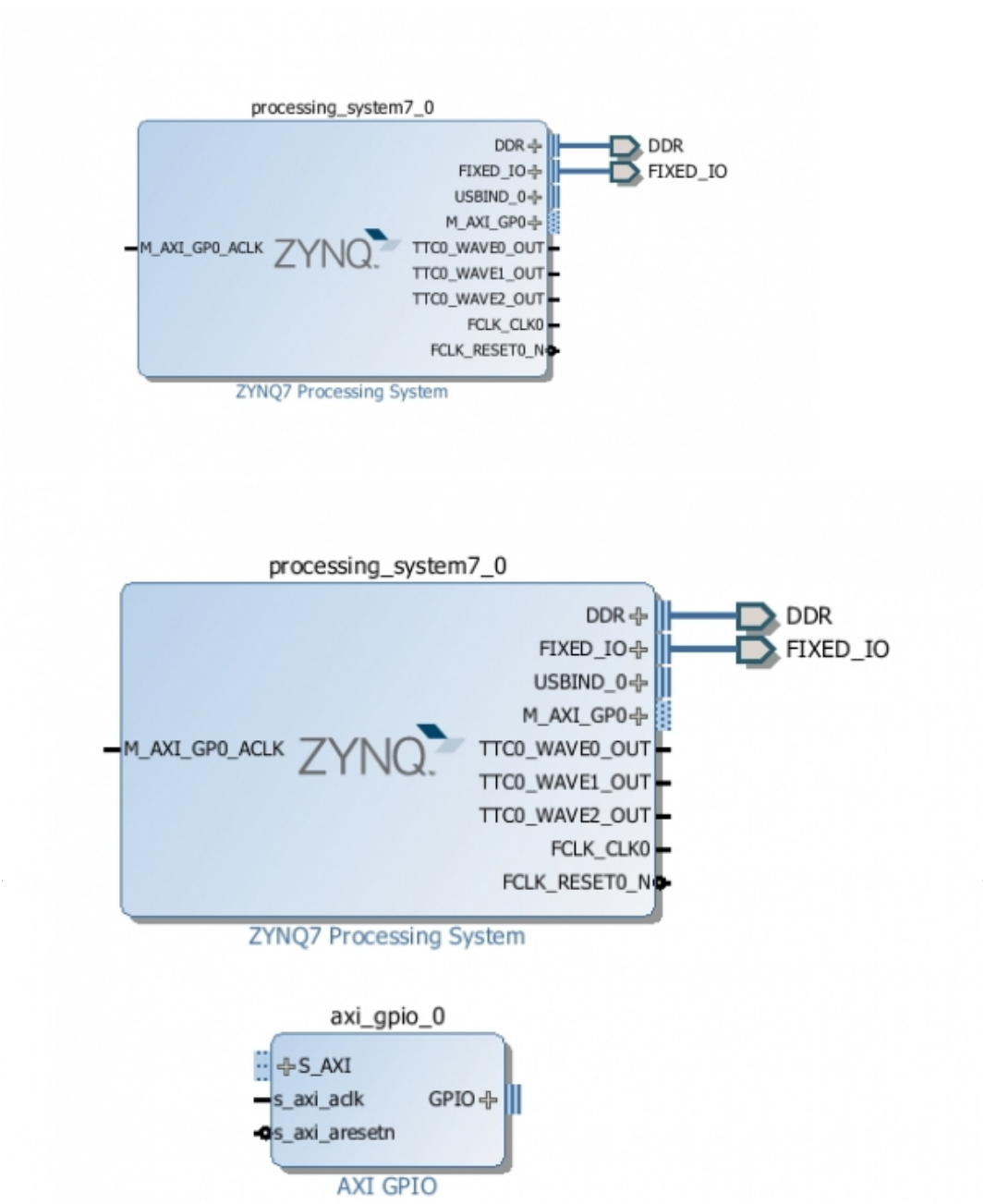
3.2) Click the **Run Block Automation** link



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_10.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

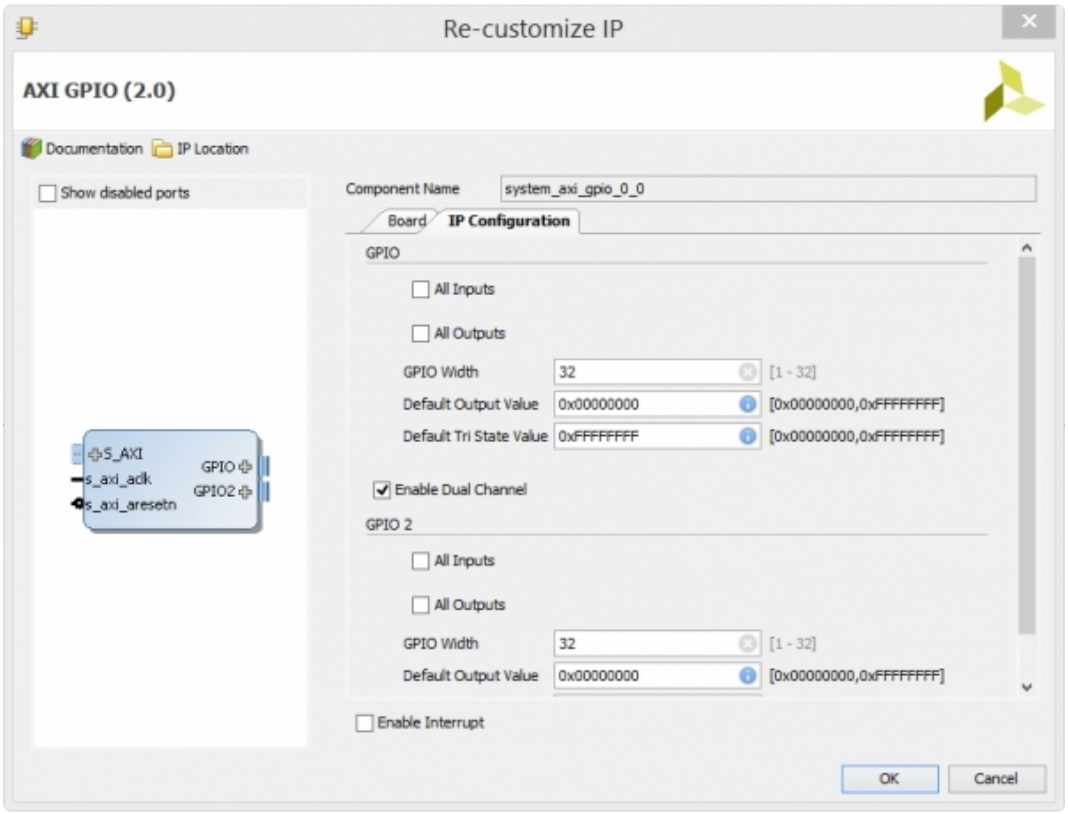
Your Zynq block should now look like the picture below.

3.3) Click the **Add IP** icon again, this time search for "gpio" and add the **AXI GPIO ()** core.



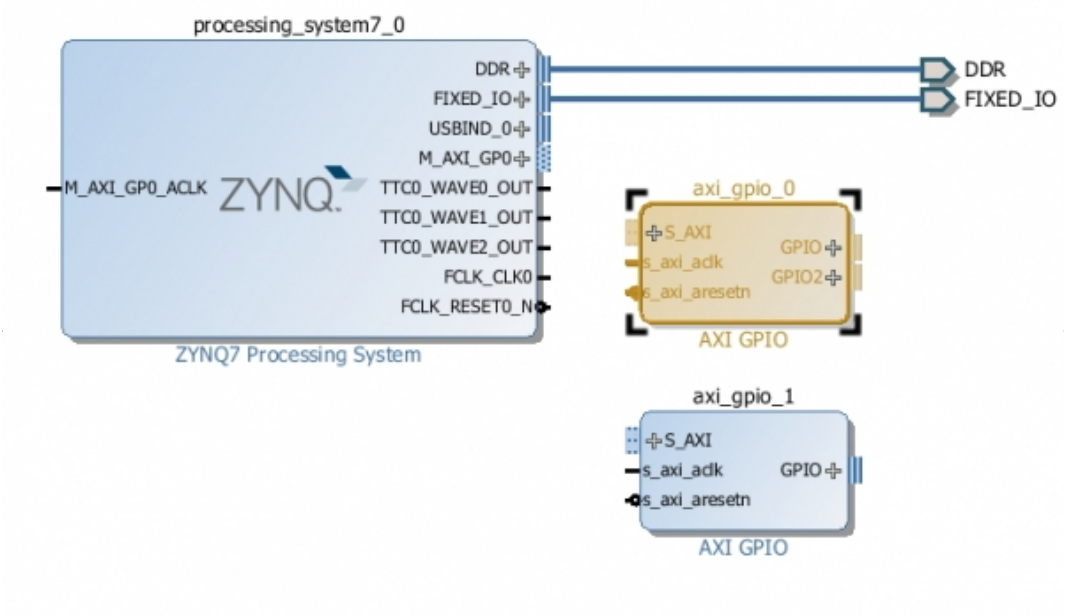
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_12.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

3.4) Double-click on new **axi_gpio_0** core that was just added to bring up the customizing window. Under the **IP Configuration** tab check the **Enable Dual Channel** box. Click **OK**.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_13.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

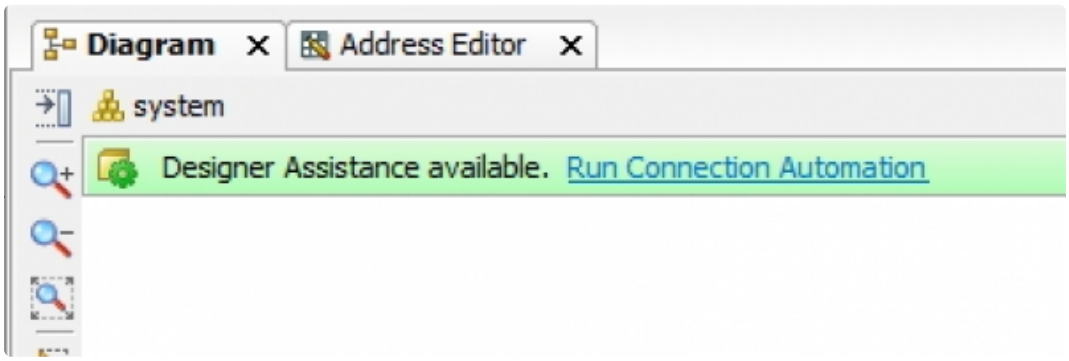
3.5) Add another GPIO () core by repeating step 3.4 but do not enable dual channel.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_14.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

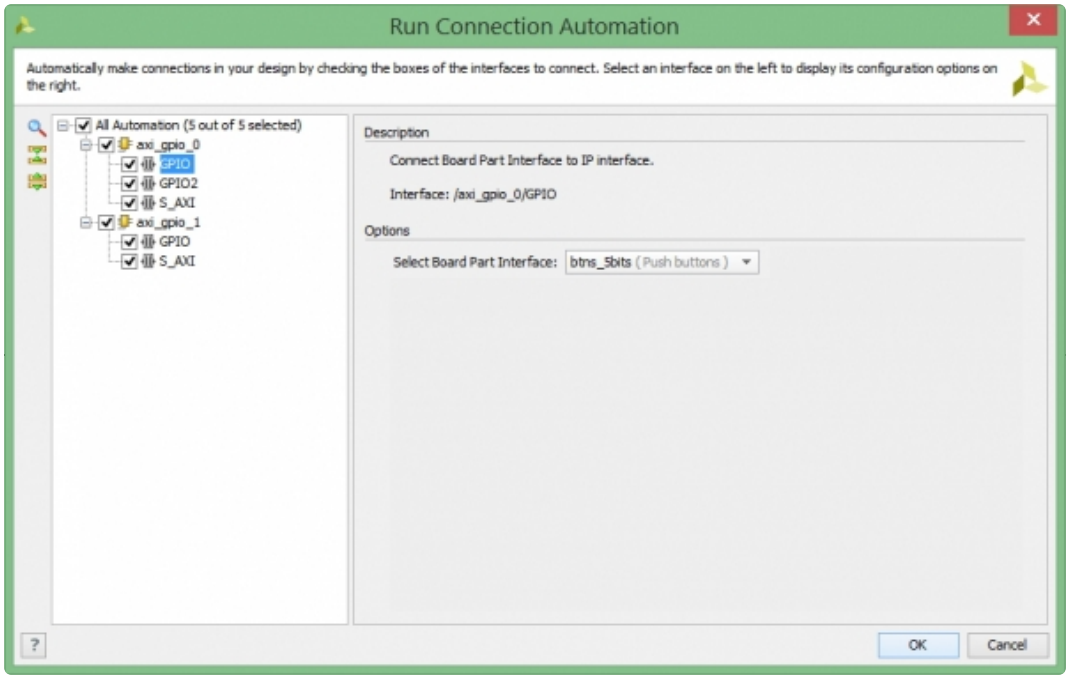
4. Run the Connection Automation Tool

4.1) The connection automation tool will add the required logic blocks for the demo. Select **Run Connection Automation** highlighted in blue.



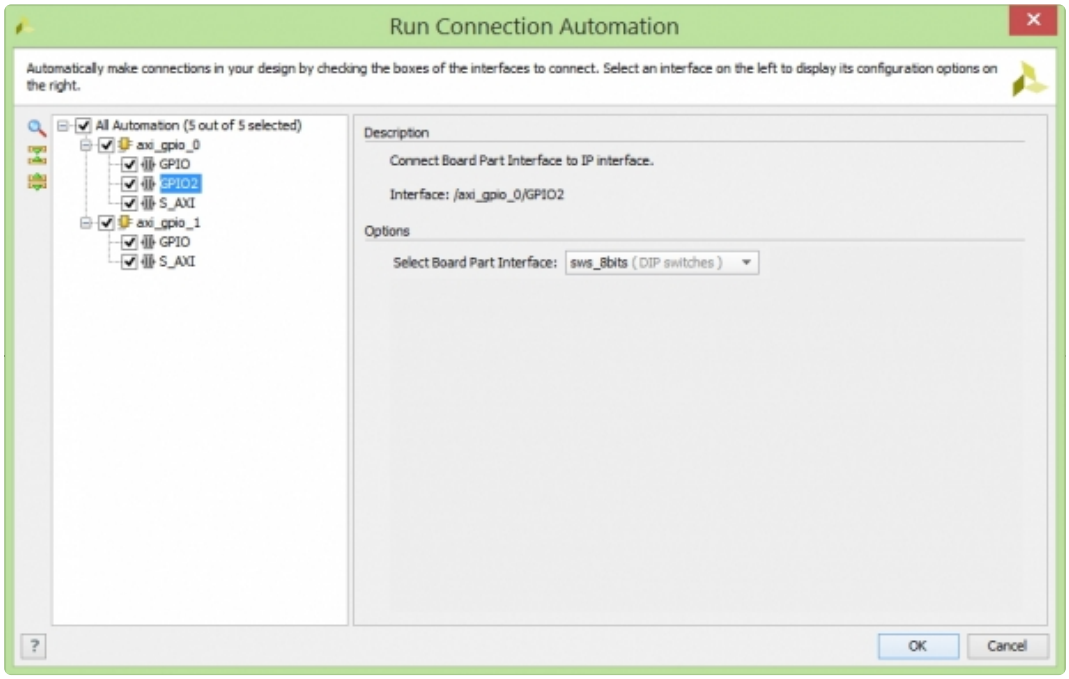
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_15.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

4.2) Check the box by **All Automation**. Select **GPIO ()** under **axi_gpio_0** and select **btns_5bits** in the Board Part Interface drop-down box.



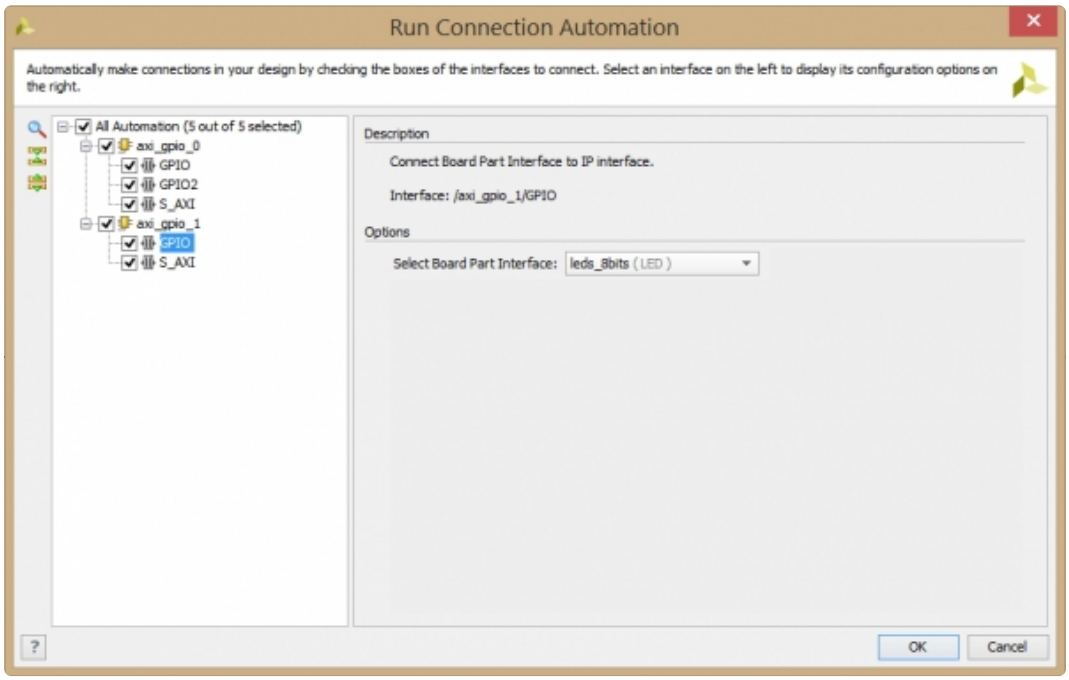
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_16.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

4.3) Select **GPIO2** under **axi_gpio_0** and select **swts_8bits** in the drop-down box.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_17.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

4.4) Select **GPIO ()** under **axi_gpio_1** and select **leds_8bits** in the drop-down box and hit **OK**.

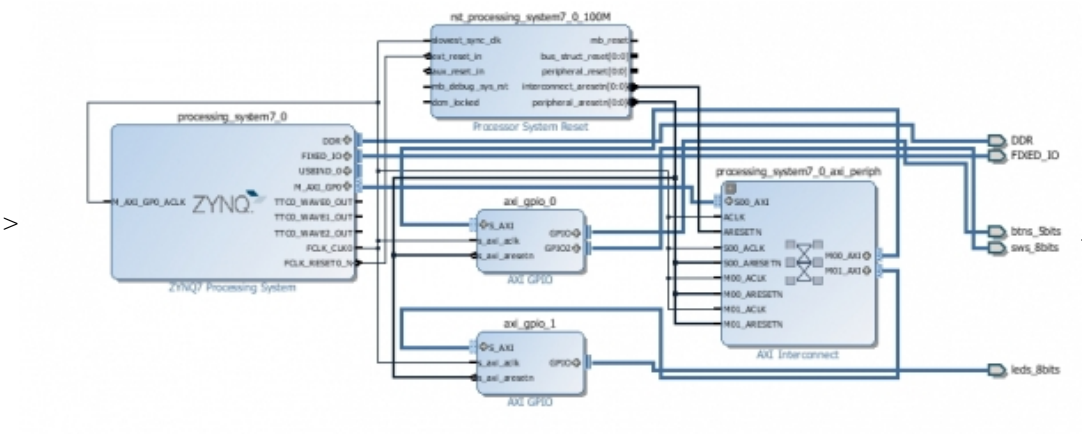


(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_18.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

4.5) This process will add:

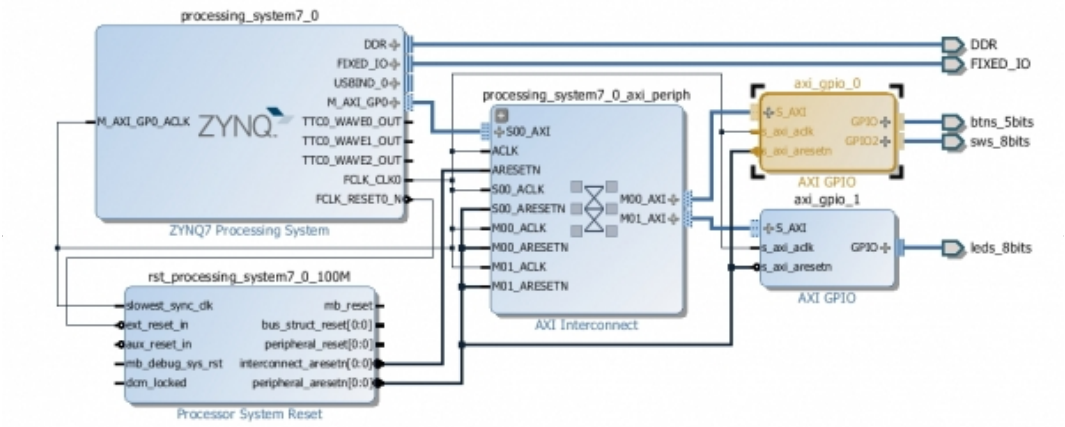
```
* The AXI interconnect
* Processor System Reset
* The board parts for the buttons, switches and LEDs.
```

>



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_19.1.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

4.6) Next let's clean up our Block Design. Click the **Regenerate Layout** button to rearrange your block design.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_19.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

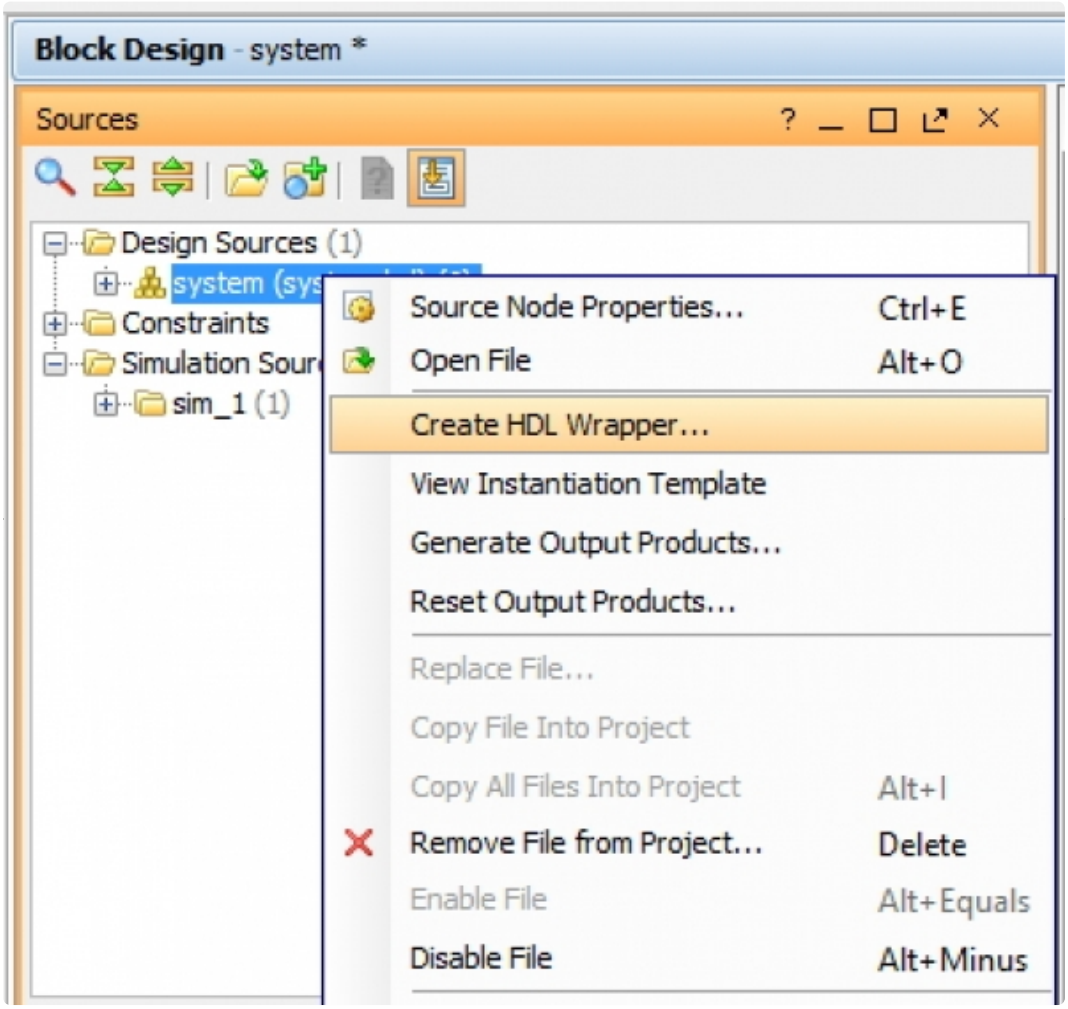
---

5. Generate HDL Wrapper and Validate Design

5.1) Select **Validate Design**. This will check for design and connection errors.

Not found in vivado 2020.1 but not needed

5.2) After the design validation step we will proceed with creating a HDL System Wrapper. In the block design window, under the **Design Sources** tab, right-click on the block diagram file. We labeled it "design_1.bd" and select **Create HDL Wrapper**.

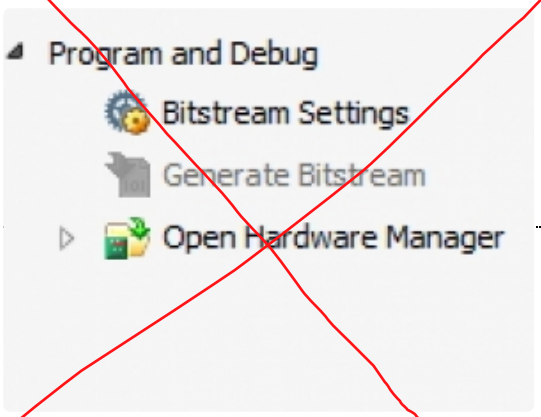


(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_20.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

This will create a top module in VHDL and will allow you to generate a bitstream.

6. Generate the Bitstream

6.1) Click on **Generate Bitstream** at the bottom of the Flow Navigator. Wait for the process to complete and click OK.

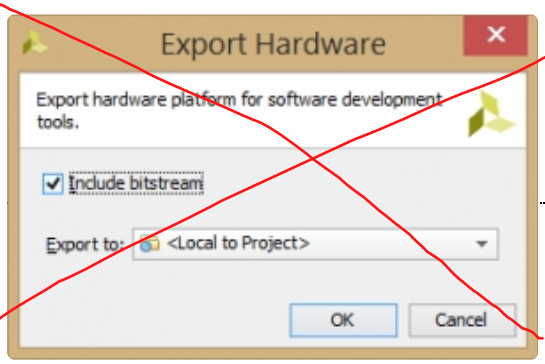


For vivado 2020.1: (found on left side)
* Run Synthesis
* Run Implementation
* Generate Bitstream

(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_22.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

7. Export hardware files for SDK

7.1) Go to file→Export→Export Hardware… Make sure to check the box for **Include bitstream** then click **OK**.



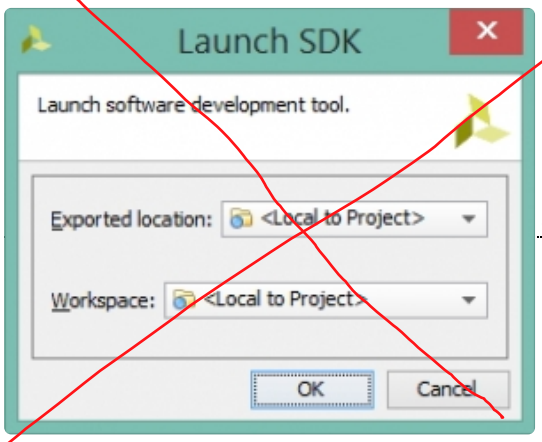
Vivado 2020.1:
File->Export->Export Hardware
* choose Platform type: "Fixed"
* click Next
* choose: Include bitstream
* choose: Any name for bitfile

(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_23.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

8. Launch SDK

Openup Vitis

8.1) Go to **File→Launch SDK** and click **OK**.



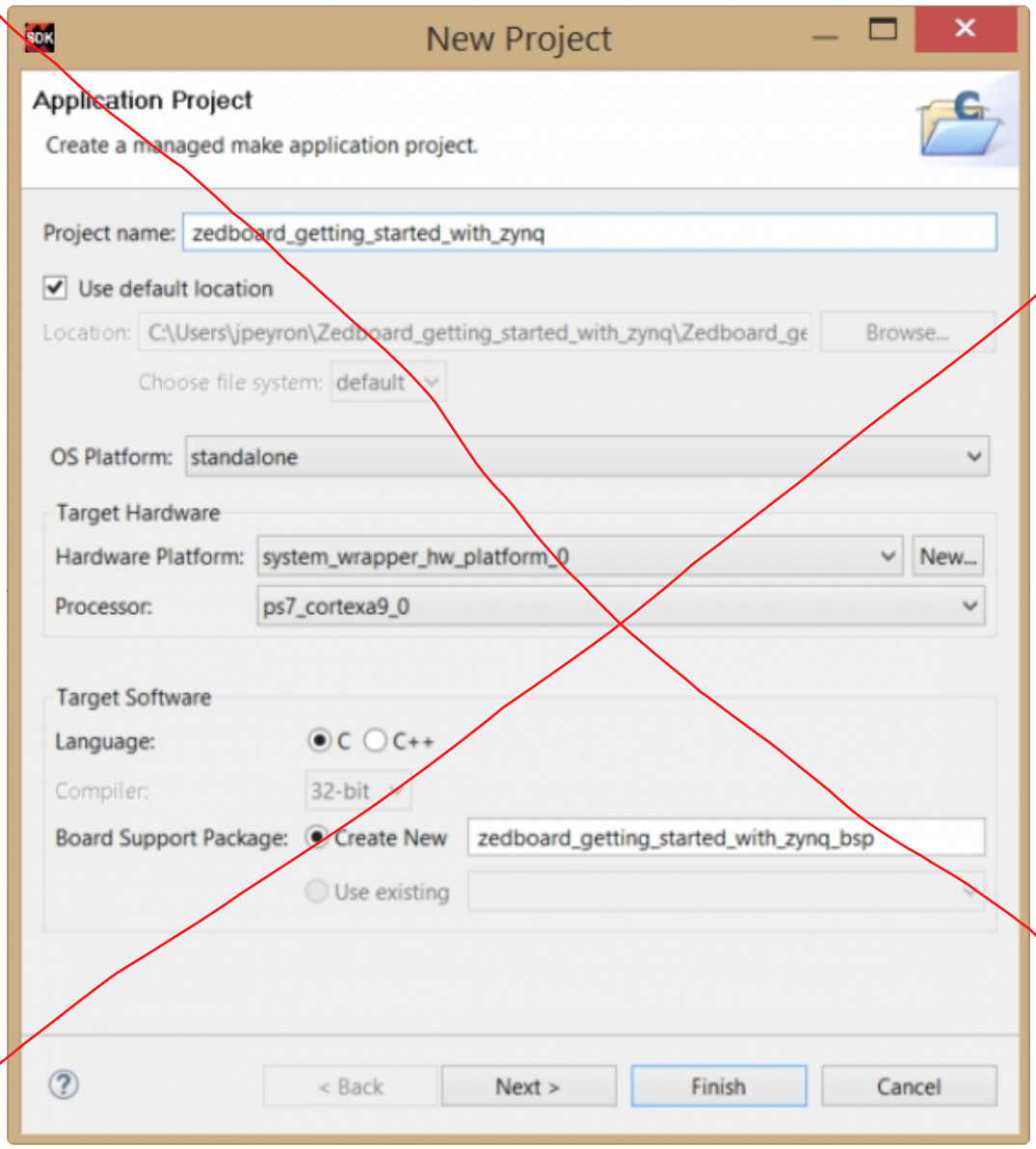
(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_24.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

9. Create a new Hello World Application Project

For vitis:
* File -> New -> Application Project
* (skip welcome page)
* click on "Create a new platform from hardware (XSA)" and browse for xilinx project and locate the .xsa file in the root directory -> Next
* choose "ps7_cortexa9_0" as processor type and enter any project name -> Next
* (skip "Domain" page with Next)
* Choose "Hello World" -> Finish

* Navigate: "any project name" -> src -> helloworld.c
* Continue from 10.1

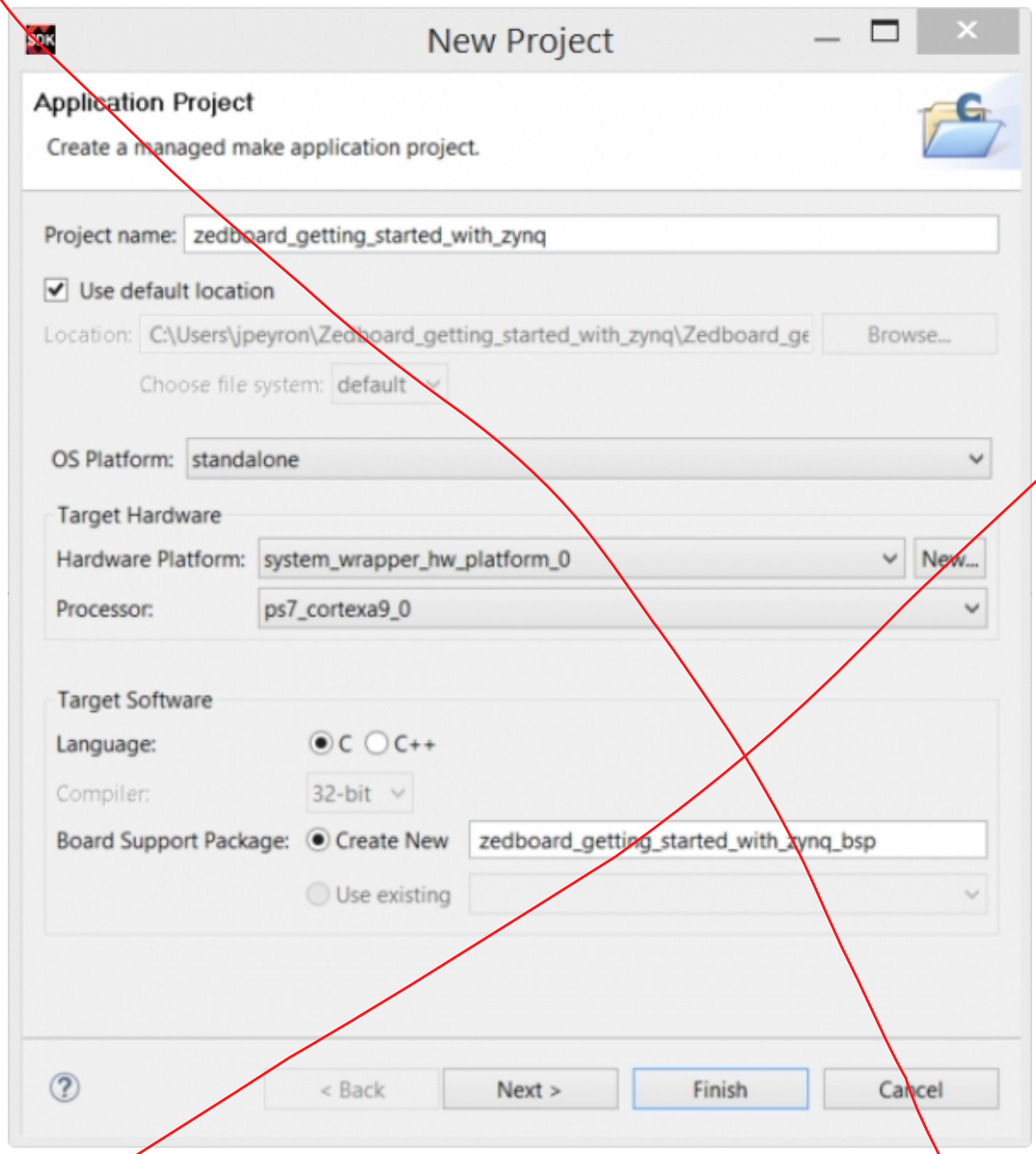
9.1) Go to **File→New→Application Project**.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_25.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

9.2) Enter the project details:

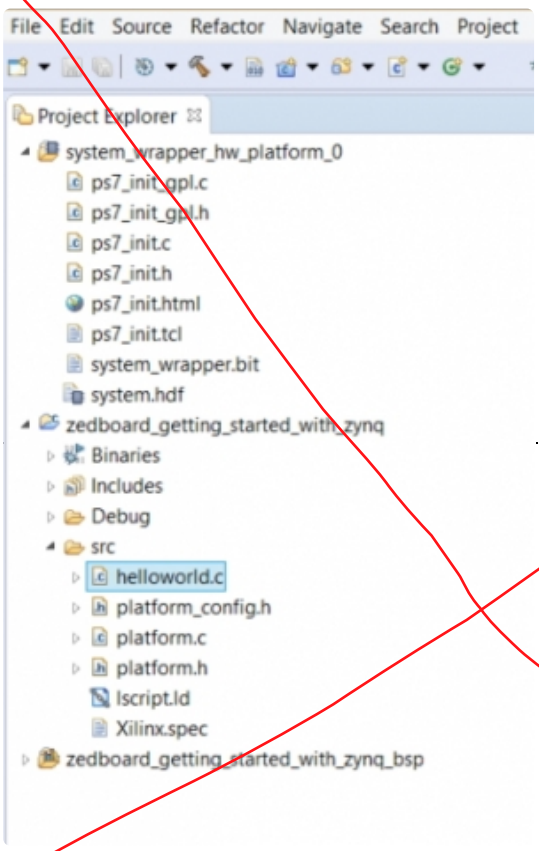
```
* Project name: "zedboard_getting_started_with_zynq"
* Hardware Platform: design_1_wrapper_hw_platform_0
* Processor: ps7_cortexas9_0
* Languate: C
* OS Platform: standalone
* Board Support Package: Create New (leave default name)
```



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_26.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

9.3) The Hello World demo is a good starting point for this demo. Click **Next**, select **Hello World** and click **Finish**. This process will add two directories to the project explorer.

9.4) Expand **zedboard_getting_started_with_zynq** then open **src** and double click on “helloworld.c”. This is the default Hello World C code.



(https://reference.digilentinc.com/_detail/reference/programmable-logic/zedboard-getting-started-with-zynq/image_27.jpg?id=learn%3Aprogrammable-logic%3Atutorials%3Azedboard-getting-started-with-zynq%3Astart)

10.1) Copy and paste the code below into the helloworld.c file.

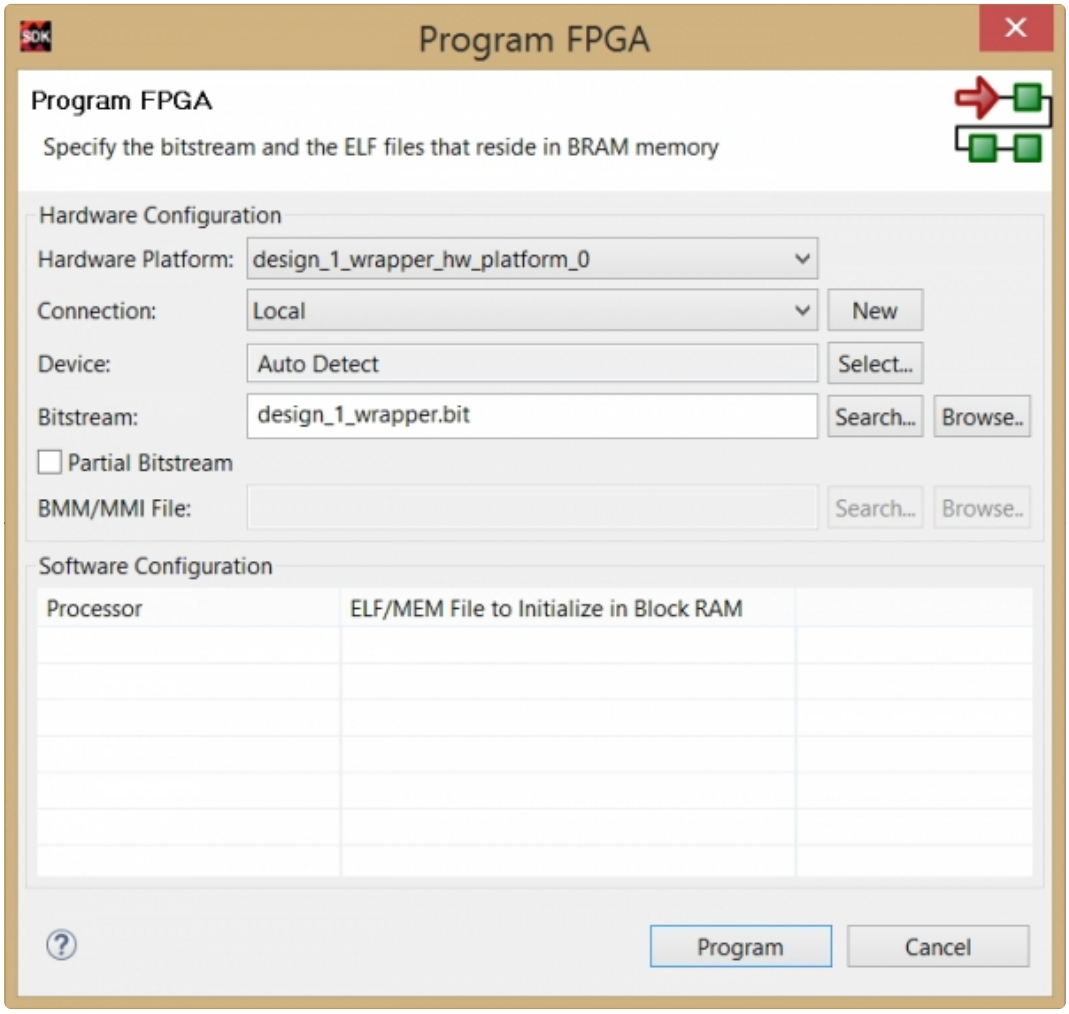
For easy copy-pasting please visit the website!

```
/*****************************************************
Getting Started Guide for Zedboard

This demo displays the status of the switches on the
LEDs and prints a message to the serial communication
when a button is pressed.

Terminal Settings:
   -Baud: 115200
   -Data bits: 8
   -Parity: no
   -Stop bits: 1

7/25/16: Created by JonP
*****************************************************/

#include <stdio.h>
#include "platform.h"
#include <xgpio.h>
#include "xparameters.h"
#include "sleep.h"

int main()
{
   XGpio input, output;
   int button_data = 0;
   int switch_data = 0;

   XGpio_Initialize(&input, XPAR_AXI_GPIO_0_DEVICE_ID); //initialize input XGpio variable
   XGpio_Initialize(&output, XPAR_AXI_GPIO_1_DEVICE_ID);        //initialize output XGpio variable

   XGpio_SetDataDirection(&input, 1, 0xF);                      //set first channel tristate buffer to input
   XGpio_SetDataDirection(&input, 2, 0xF);                      //set second channel tristate buffer to inpu

   XGpio_SetDataDirection(&output, 1, 0x0);             //set first channel tristate buffer to output

   init_platform();

   while(1){
      switch_data = XGpio_DiscreteRead(&input, 2);      //get switch data

      XGpio_DiscreteWrite(&output, 1, switch_data);     //write switch data to the LEDs

      button_data = XGpio_DiscreteRead(&input, 1);      //get button data

      //print message dependent on whether one or more buttons are pressed
      if(button_data == 0b00000){} //do nothing

      else if(button_data == 0b00001)
         xil_printf("button 0 pressed\n\r");

      else if(button_data == 0b00010)
         xil_printf("button 1 pressed\n\r");

      else if(button_data == 0b00100)
         xil_printf("button 2 pressed\n\r");

      else if(button_data == 0b01000)
         xil_printf("button 3 pressed\n\r");

      else if(button_data == 0b10000)
              xil_printf("button 4 pressed\n\r");

      else
         xil_printf("multiple buttons pressed\n\r");
```

```
        usleep(200000);                    //delay

    }
    cleanup_platform();
    return 0;
}
```

11. Run the Project

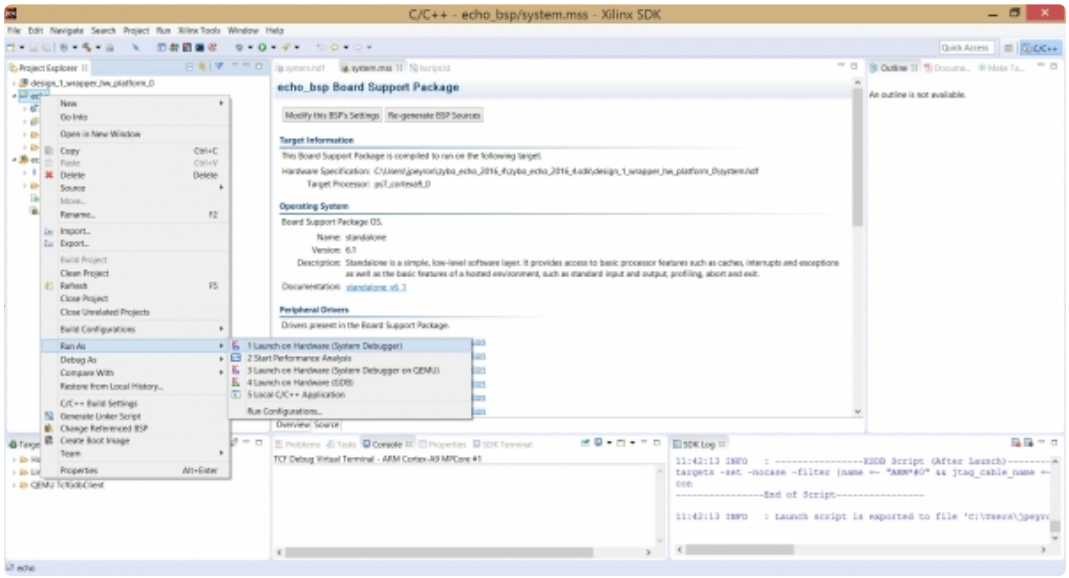
11.1) Make sure that the Zedboard is connected to the host PC via the **UART** USB Port and that JP5 is set to JTAG. To program the FPGA, on the top toolbar, click the **Program FPGA** button. In vitis: Xilinx -> Program FPGA



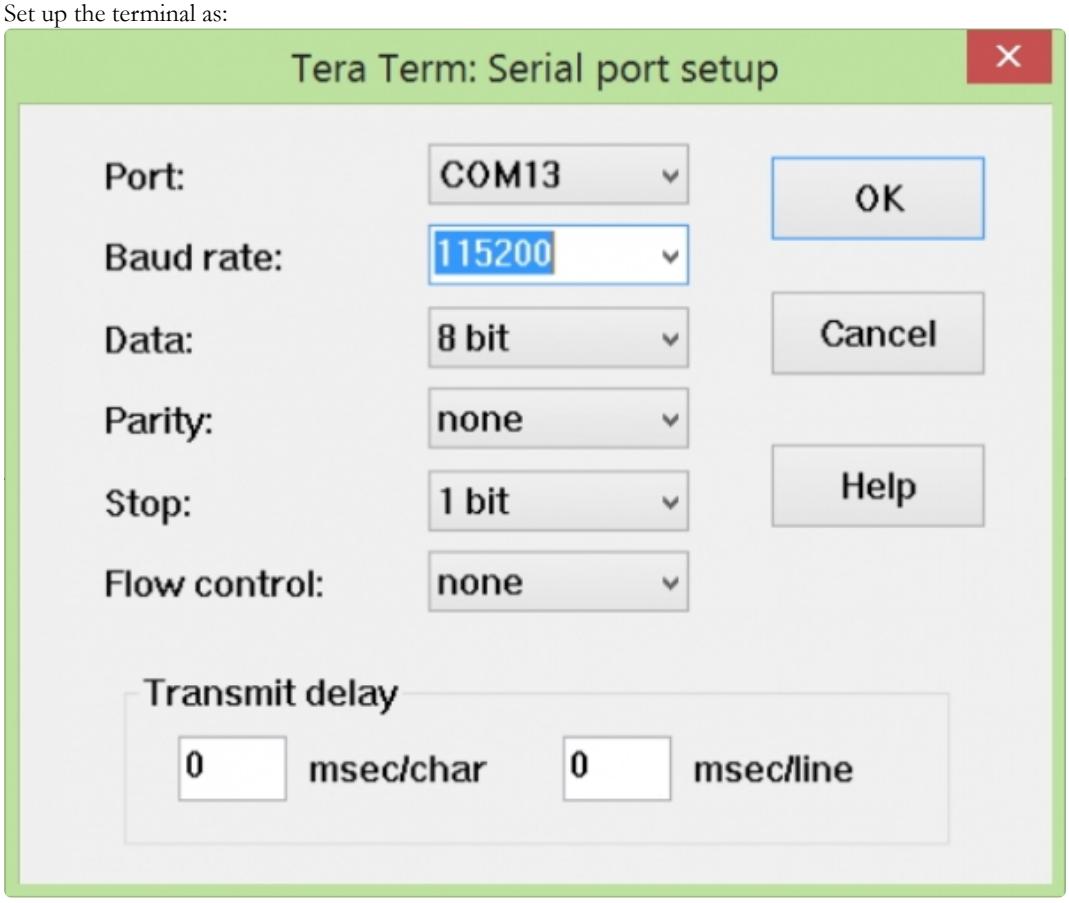
(https://reference.digilentinc.com/_media/learn/programmable-logic/tutorials/zybo-getting-started-with-zynq-server/echo_server_zynq_9.jpg)

11.2) Save the project. The project will automatically build.

11.3) Right click the **zedboard_getting_started_with_zynq** directory and select run as → Launch on Hardware(System Debugger).



(https://reference.digilentinc.com/_media/learn/programmable-logic/tutorials/zybo-getting-started-with-zynq-server/echo_server_zynq_13.jpg)
11.4) The demo will be running on the Zedboard. Try playing around with the 4 switches (labeled SW0-SW3). Doing so should light its respective LED (). Also over the serial port, pressing each button (labeled BTN0-BTN3) will produce the message "button x pressed".
11.5) Tera Term or any serial terminal will work as a Console for displaying the output of the BTN's.
Set up the terminal as:



(https://reference.digilentinc.com/_media/learn/programmable-logic/tutorials/zybo-getting-started-with-zynq-server/echo_server_zynq_10.jpg)

---

learn (https://reference.digilentinc.com/tag/learn?do=showtag&tag=learn), programmable-logic (https://reference.digilentinc.com/tag/programmable-logic?do=showtag&tag=programmable-logic), tutorial (https://reference.digilentinc.com/tag/tutorial?do=showtag&tag=tutorial), zedboard (https://reference.digilentinc.com/tag/zedboard?do=showtag&tag=zedboard)

- Technology Partners (https://store.digilentinc.com/technology-partners/)
- Distributors (https://store.digilentinc.com/our-distributors/)
- Contact Us (https://store.digilentinc.com/contact-us/)
- Store Info (https://store.digilentinc.com/store-info/)
- returns/)
- Legal (https://store.digilentinc.com/
- Jobs (https://store.digilentinc.com/
- Internships (https://store.digilentinc.com/

Connect With Us