Introduction to Systems-on-Chip
Development of a Basic System-on-Chip Architecture in an FPGA

Abigail Butka, Muhammed Kawser Ahmed, Christophe Bobda

March 29, 2024

1 Objective

The learning objective of this module is to gain hands-on experience in the basic concepts of System-on-Chip design. In this lab you will use the Xilinx Zynq ARM Cortex-A9 hard-core processor and Xilinx IP cores to build your own SoC project in Vivado. Then, you will write the C applications that will run on the Zynq processor in Vitis.

For more conceptual details of this module, please refer to the accompanying PowerPoint presentation and video.
In the unfortunate case that the following tutorial no longer produces a bitstream due to software updates to Vivado, the bitstreams for each task are provided in the accompanying zip file for both the ZYBO Z7-10 and the PYNQ-Z2.

This tutorial assumes some familiarity with Vivado.

2 Equipment and Software Needed for this Module

1. Xilinx Vivado 2021.1.
2. A Xilinx FPGA.
   (a) This lab was tested using the PYNQ-Z2 and ZYBO Z7-10
3. A USB-UART cable to program the board and display the output in the terminal

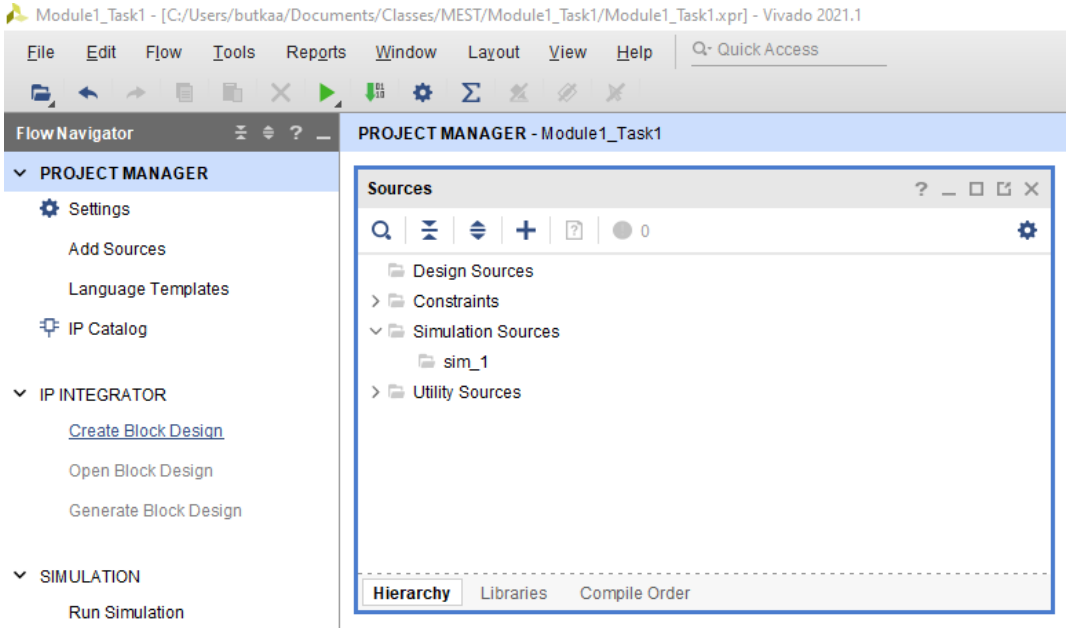
3 Module Description

3.1 Task 1: Getting Started with Zynq

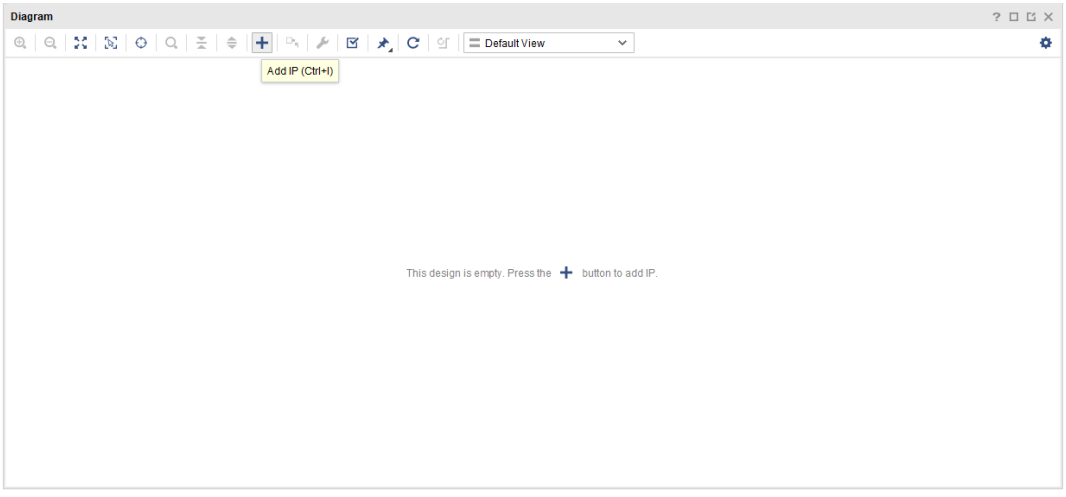
This task assumes some familiarity with Vivado. If you are unfamiliar with Vivado, please watch the accompanying video, or perform some RTL-level tutorials before beginning.

1. First, create a Vivado project, make sure you choose the board you have. Do not add any sources or constraints.

2. Next, create the Block Design by clicking the blue text seen in the image below.

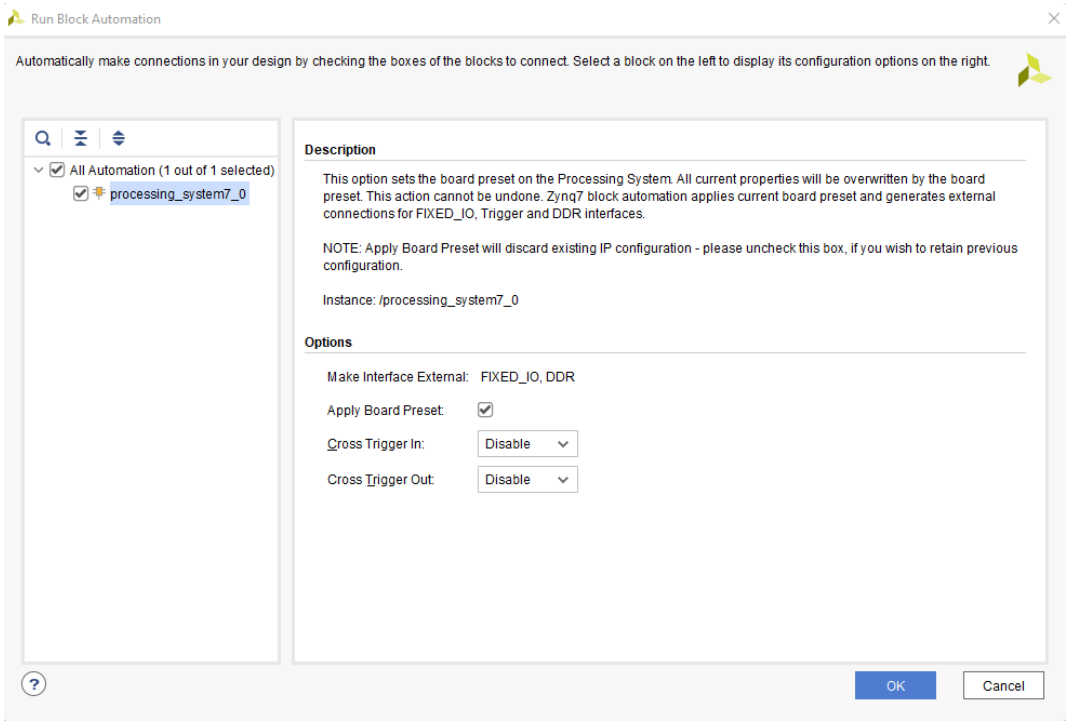


3. Add the IP ZYNQ7 Processing System to the Block Diagram by clicking either plus sign within the Block Diagram and searching for Zynq.

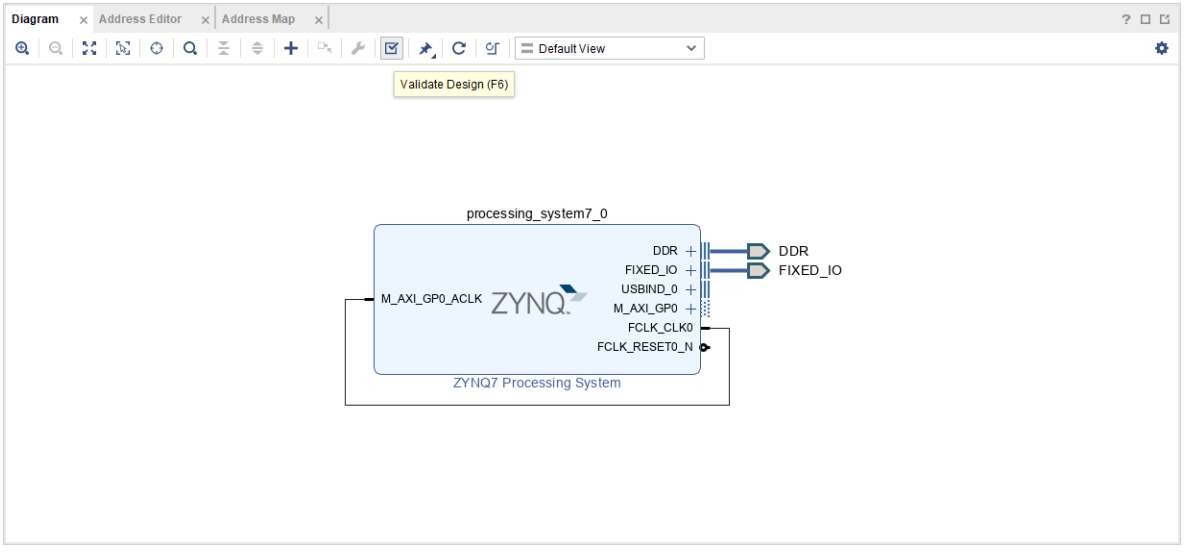


4. Next, connect M_AXI_GP0_ACLK to FCLK_CLK0 of the Zynq IP by clicking and dragging.

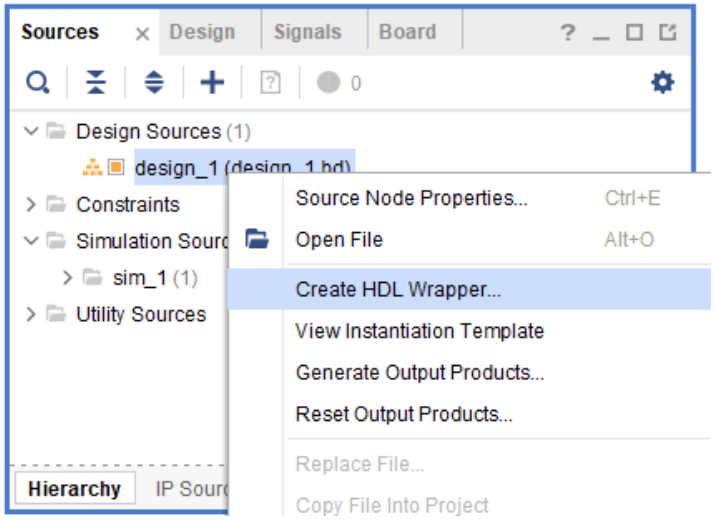
5. Then, note that once you added the ZYNQ7 to the block diagram, a green flag stating "Run Block Automation" appeared. Click Run Block Automation, ensure all boxes on the left-hand side are checked, and click OK.



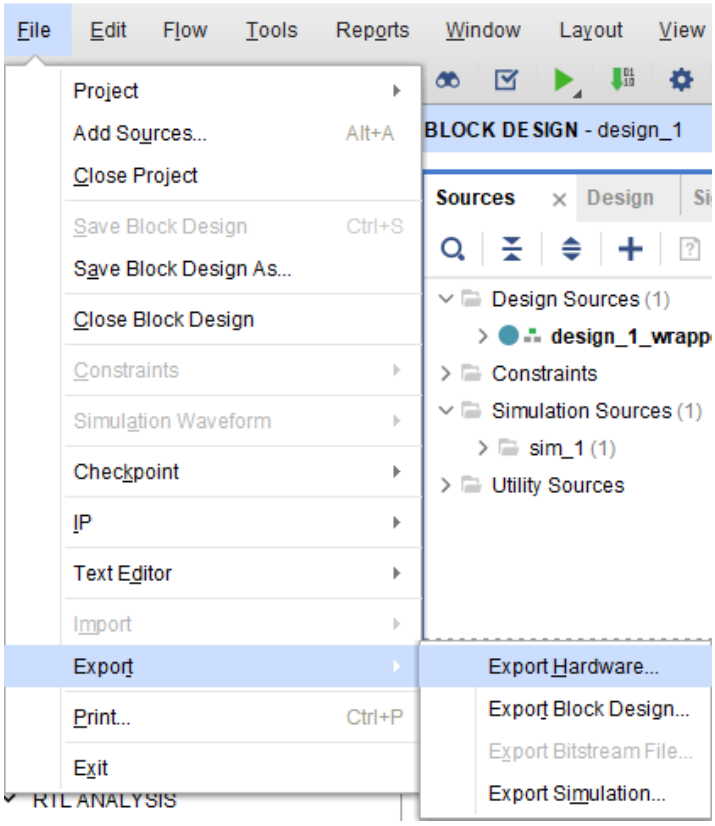
6. Next, locate and click the checkbox in the Block Diagram, Validate Design. There may be some warnings. This is OK. At this point, your Block Design should look like the image below.



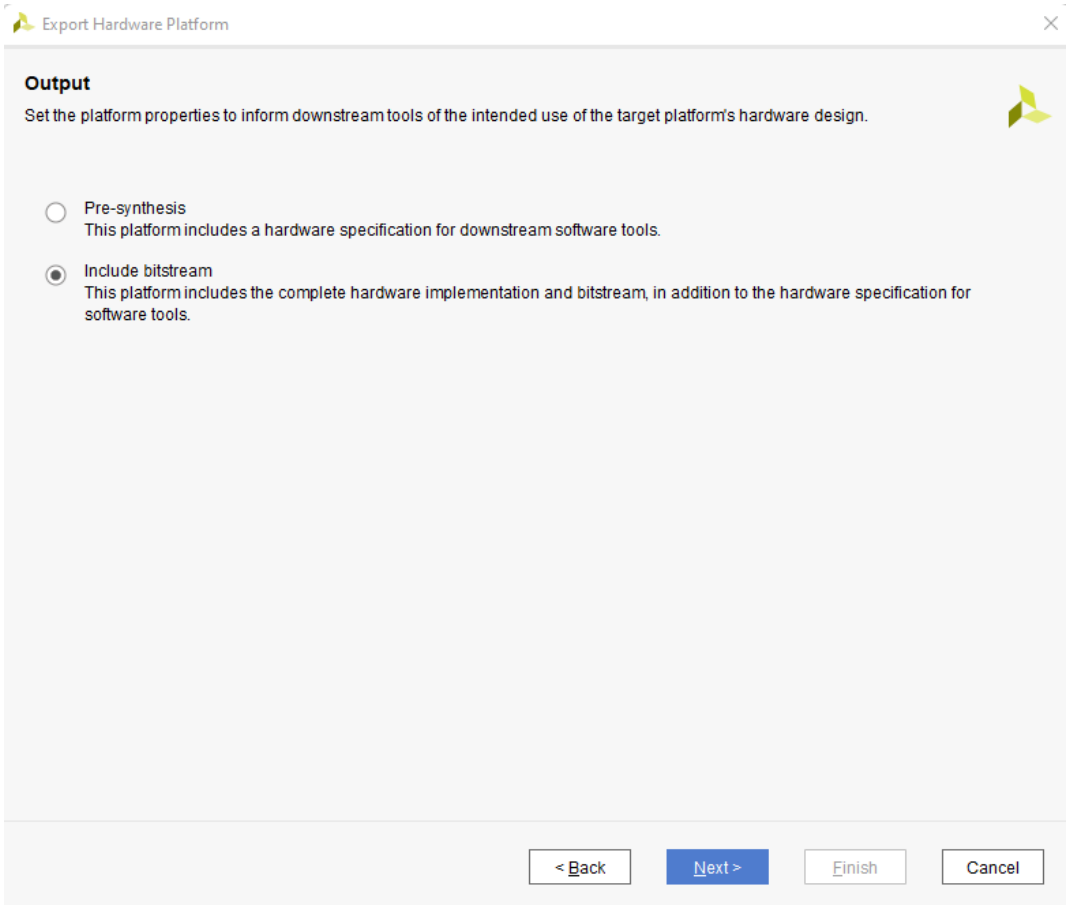
7. Next, navigate to the Sources tab and expand Design Sources.



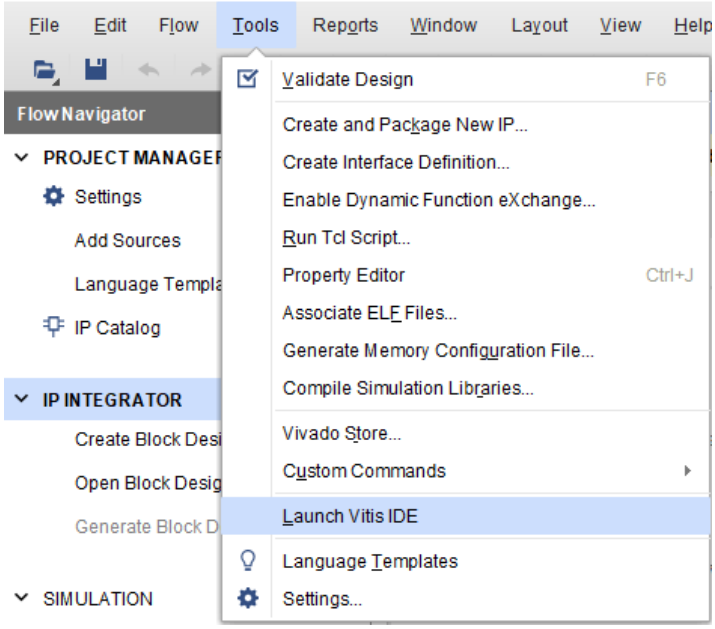
8. Right-click design_1.bd and click Create HDL Wrapper. Then select Let Vivado Manage Wrapper and Auto-Update and click OK.
9. This will create a design_1_wrapper.v file, that will now be the top file of the design.
10. Finally, generate the bitstream by clicking on Generate Bitstream in the Flow Navigator.
11. When the window "Bitstream Generation successfully completed" pops up, click cancel.
    (a) This will open the implemented design which shows the pin placements on the FPGA.
    (b) We do not need to see this information now.
12. Next, click on File/Export/Export Hardware.



13. Click Next, then ensure that Include Bitstream is selected. and click Next again.

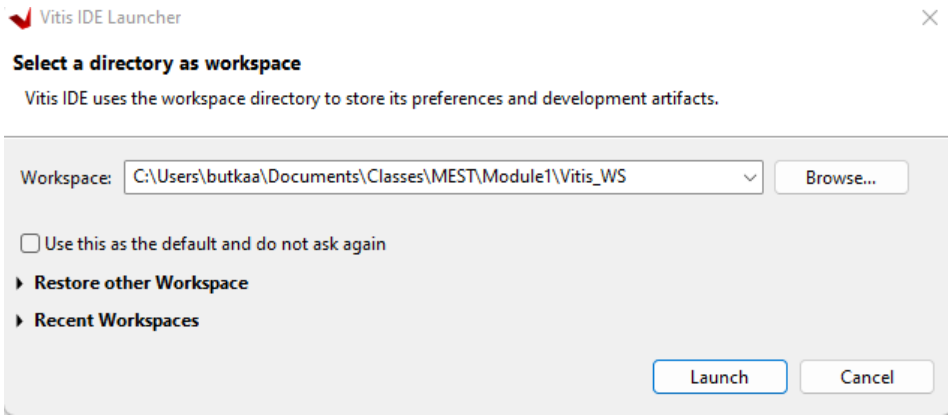


14. Leave the XSA file named design_1_wrapper, then click Next and Finish.
15. Navigate to and click on Tools/Launch Vitis IDE.

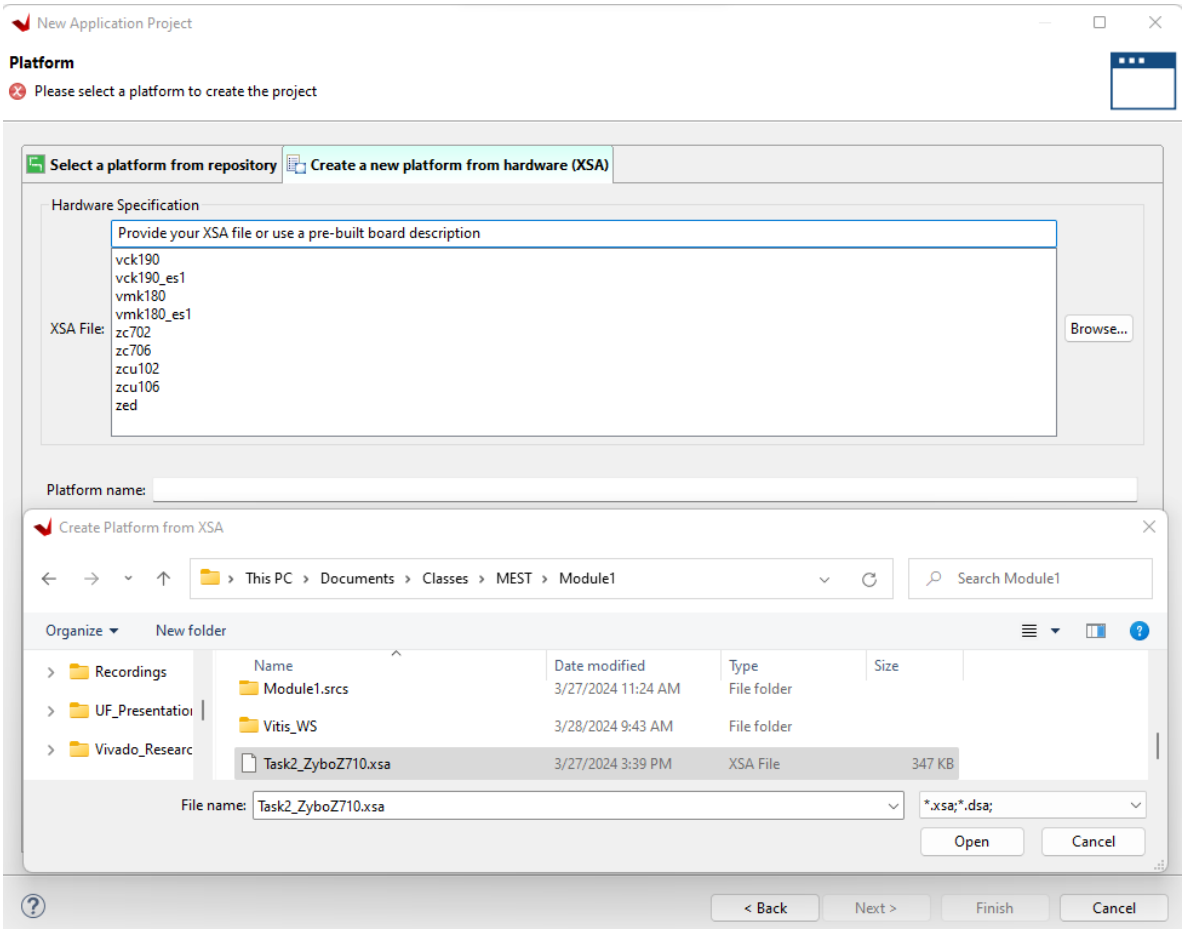


16. When the new window loads, “Select a directory as workspace”, click on browse.
17. Browse to the location of your Vivado project, you should see design_1_wrapper.xsa.

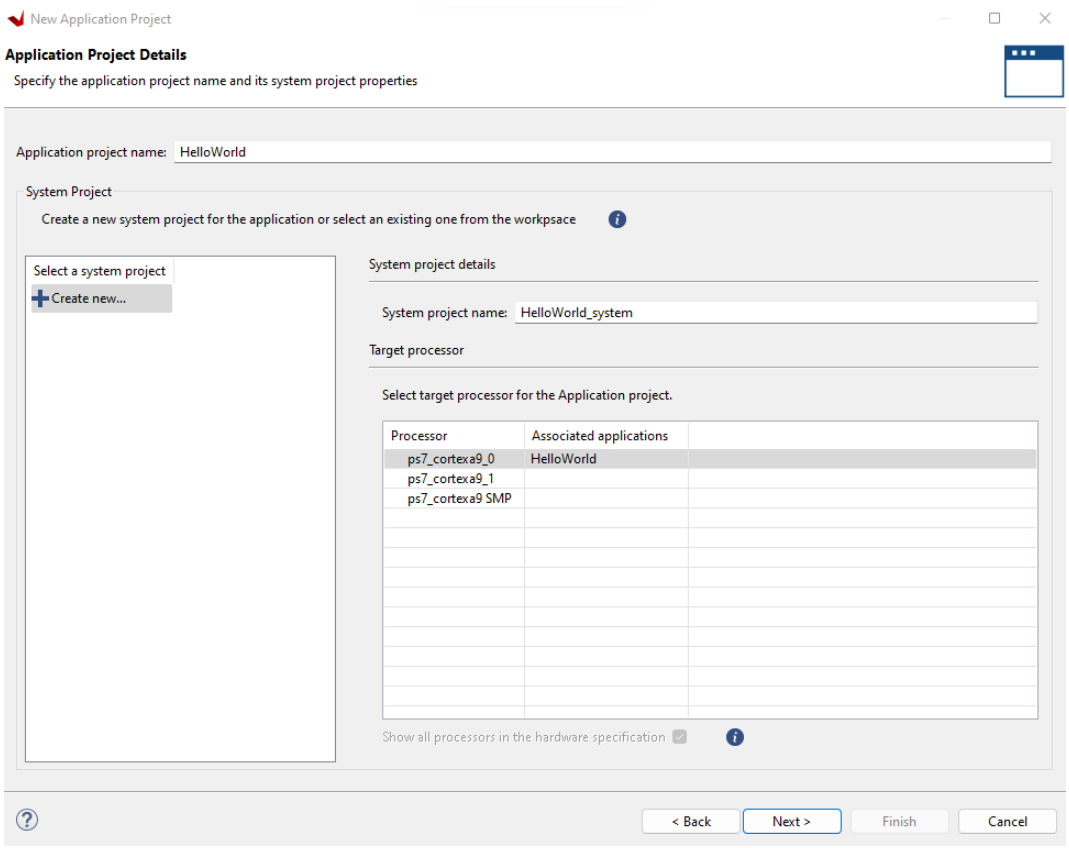
18. Right-click and create a folder titled “Vitis_WS.” This is where all your Vitis projects will be stored.



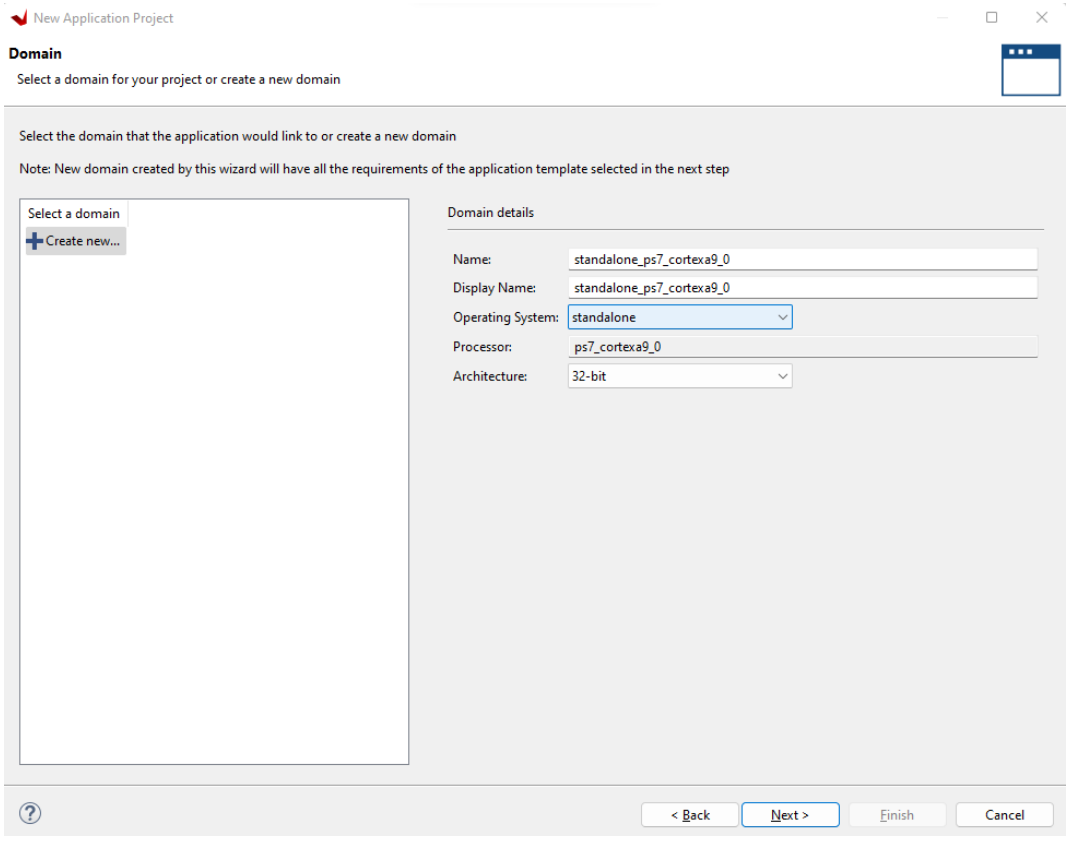
19. Click Launch.
20. When Vitis loads, click Create Application Project
21. This will open the Welcome Page of Vitis, click next
22. You will now be on the Platform Page. Click the “Create a new platform from hardware (XSA)” tab.
23. Click browse, navigate to your Vivado project again, and select the design_1_wrapper.xsa file, then click Next.



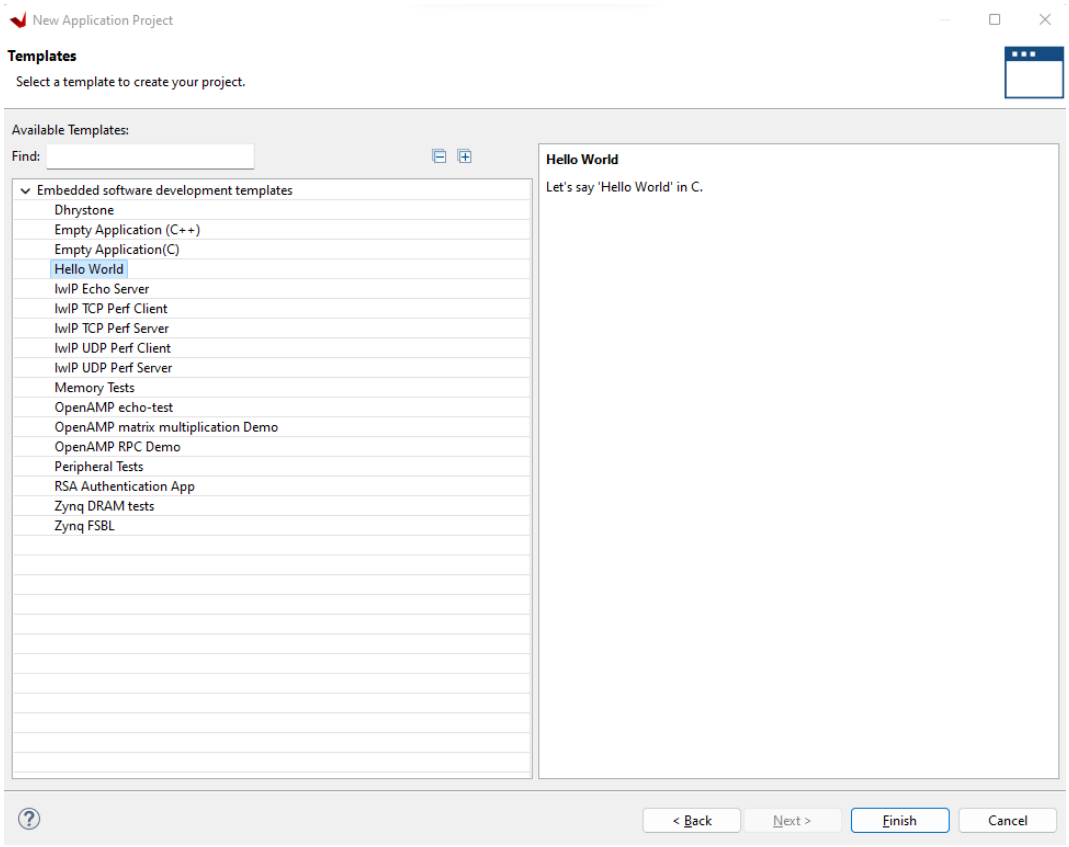
24. Name the project HelloWorld, then click Next



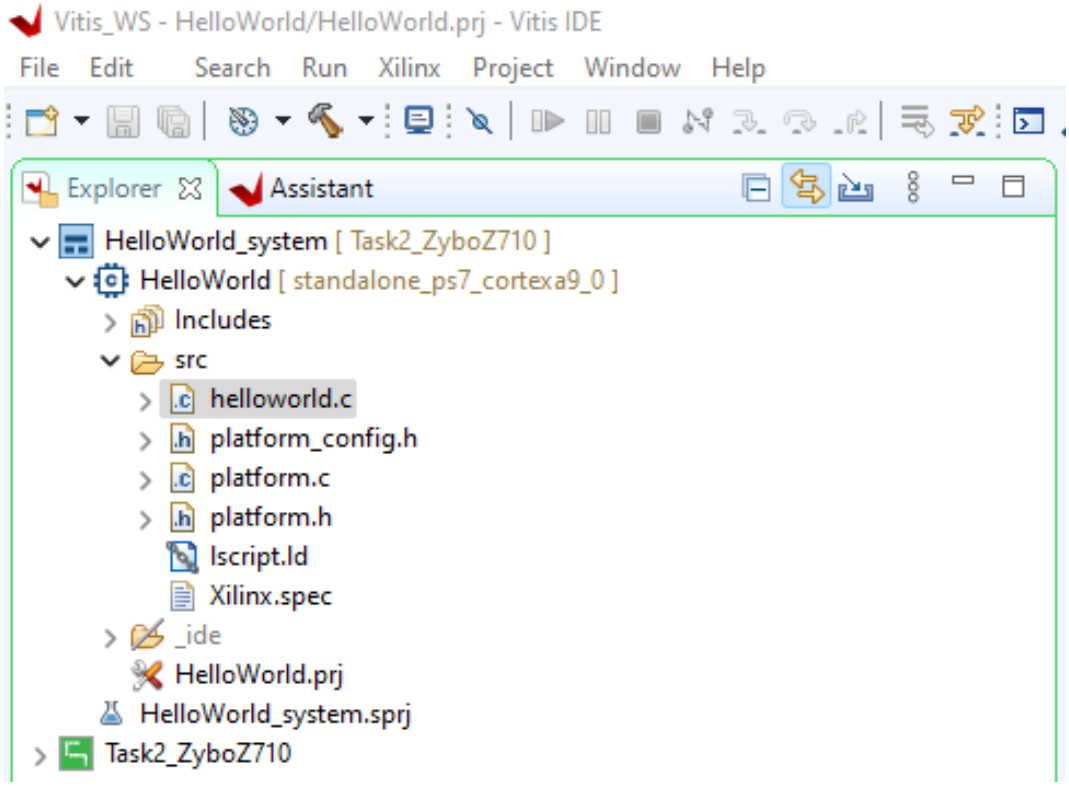
25. On the Domain Page, ensure that the Operating System is set to standalone, then click Next.



26. On the template page, select Hello World, and then click Finish.

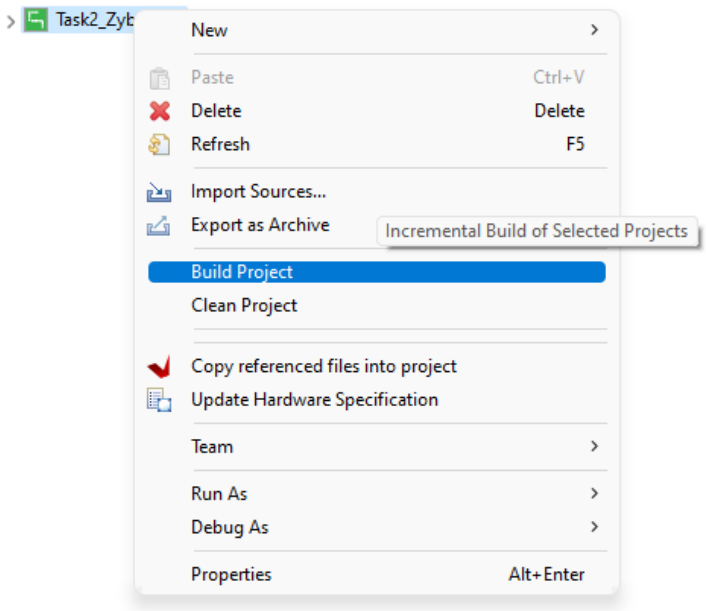


27. This will create both the platform project and application project in your Vitis Workspace.



28. To find the HelloWorld.c file, click on the arrows beside each of the following folders in the Explorer HelloWorld_System/HelloWorld/src/helloworld.c.

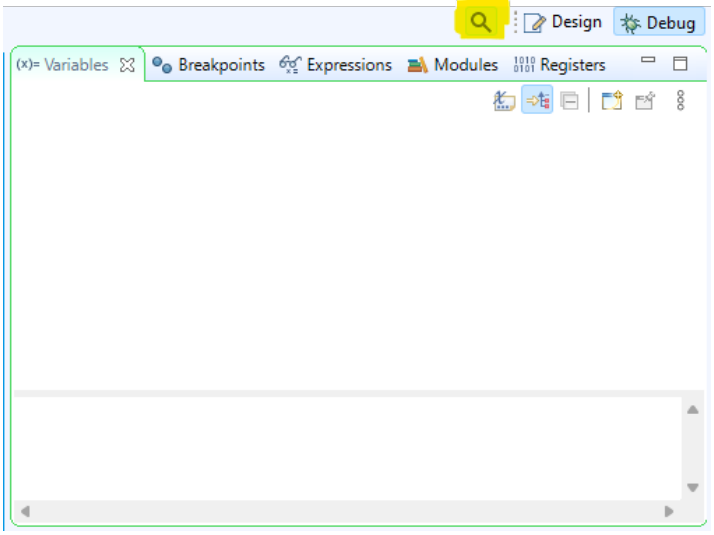
29. Right-click on the green design_1_wrapper platform project and select "Build Project."



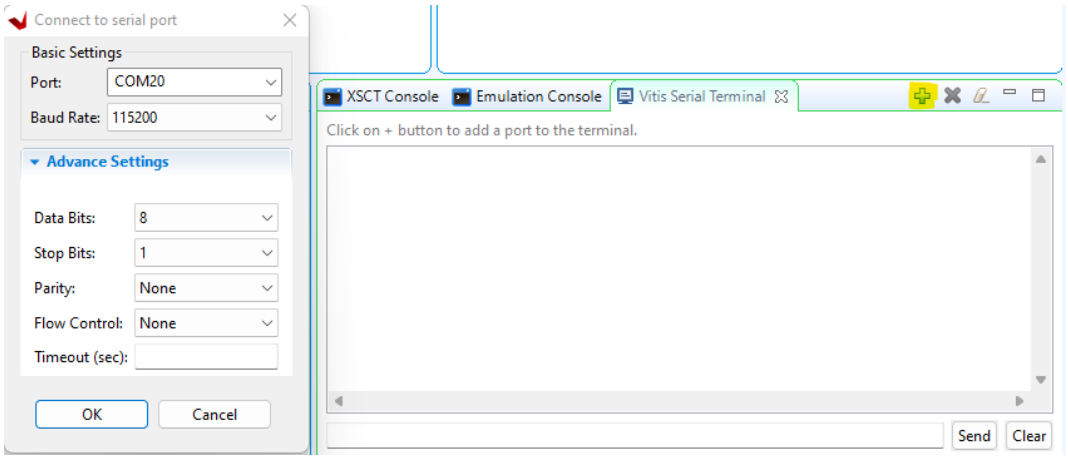
30. Right-click on the blue HelloWorld_system application project and select "Build Project."

    a. If either step 29 or 30 fails with an unexpected error, clean, then build the project again, it should work the second time.

31. After building both projects, open the Vitis Serial Terminal by clicking on the magnifying glass in the top right corner and searching for Vitis Serial Terminal.

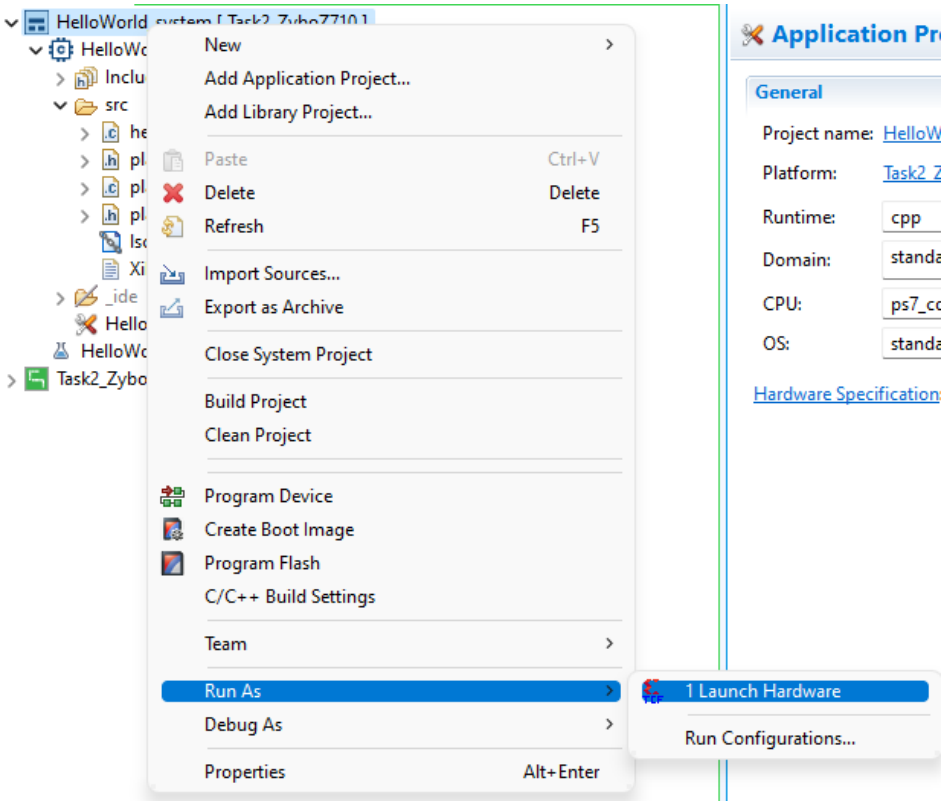


32. When the Vitis Serial Terminal tab opens on the bottom, click the green plus sign to connect to a new serial port.



33. Then, select the Port of your FPGA and ensure the Baud Rate is set to 115200. Click OK.

34. Finally, right-click on the blue HelloWorld_system Application project and select "Run as/ 1. Launch Hardware."



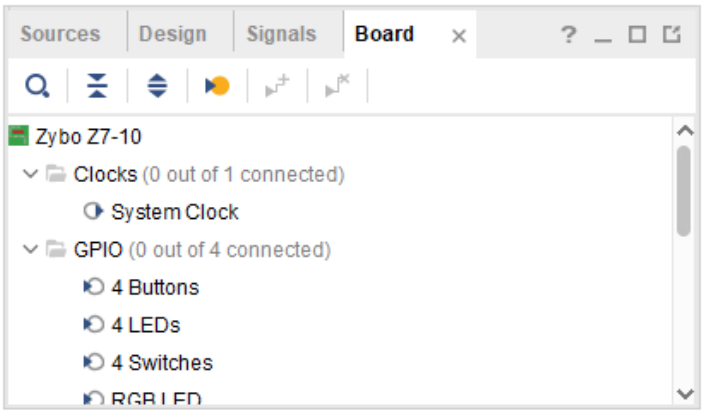
35. Wait for the board to be programmed, then watch the UART output Hello World.

3.2 Task 2: Introduction to GPIO

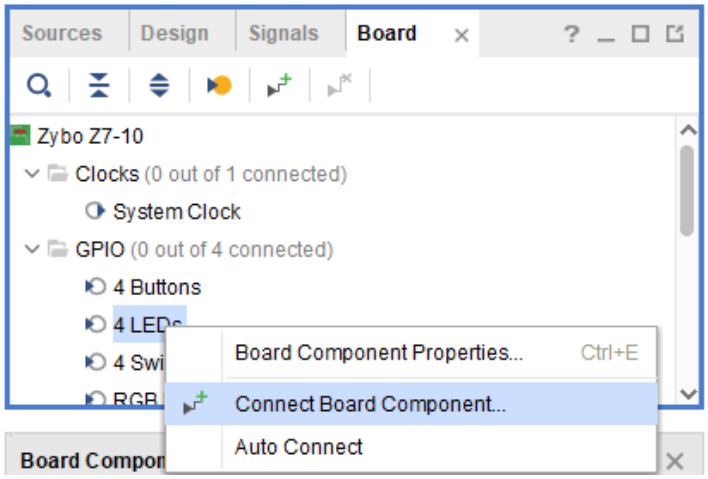
1. To begin, either create a new project using the steps 1-6 of Task 1, or open the pre-existing project.

2. Next, we will begin adding the AXI_GPIO IP cores.

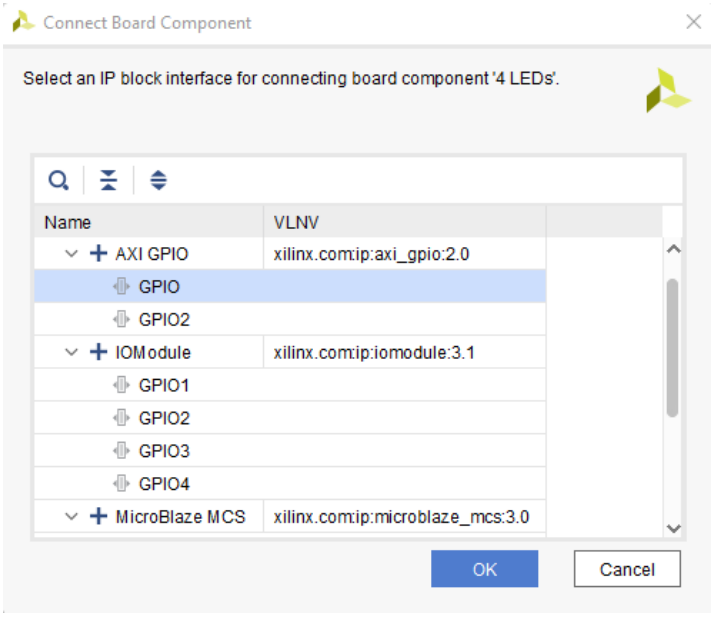
3. Navigate to the Board tab.



4. Right-click on 4 LEDs under GPIO and select Connect Board Component.



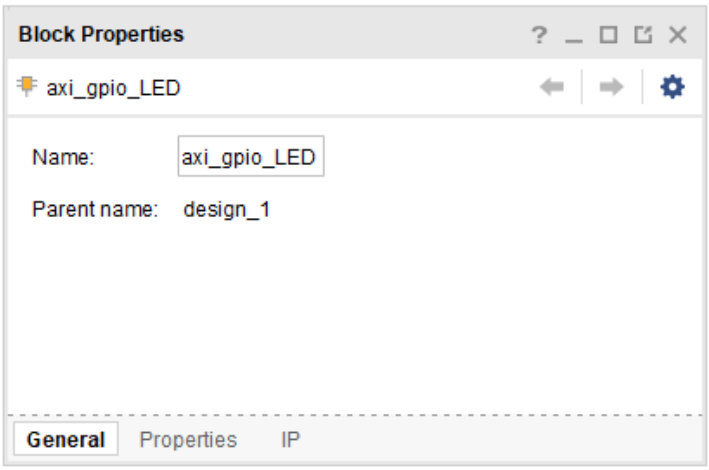
5. This will open a new window titled Connect Board Component. Select AXI GPIO and click OK.



6. Finally, select this new AXI_GPIO IP. Underneath the Board tab, there will be a Block Properties tab.

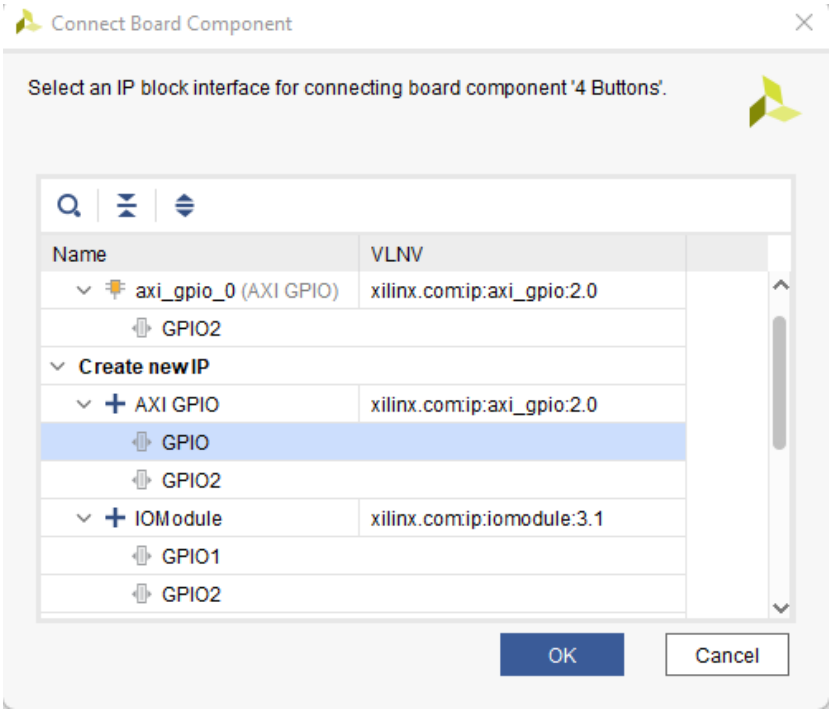
(a) If the Block Properties tab is not there, press CTRL-E.

7. Under Block Properties, change the name of this block from axi_gpio_0 to axi_gpio_LEDS.



8. Next, right-click on 4 Buttons and select Connect Board Component.

9. This prompt will look slightly different than the first. Select AXI GPIO under Create New IP and click OK.



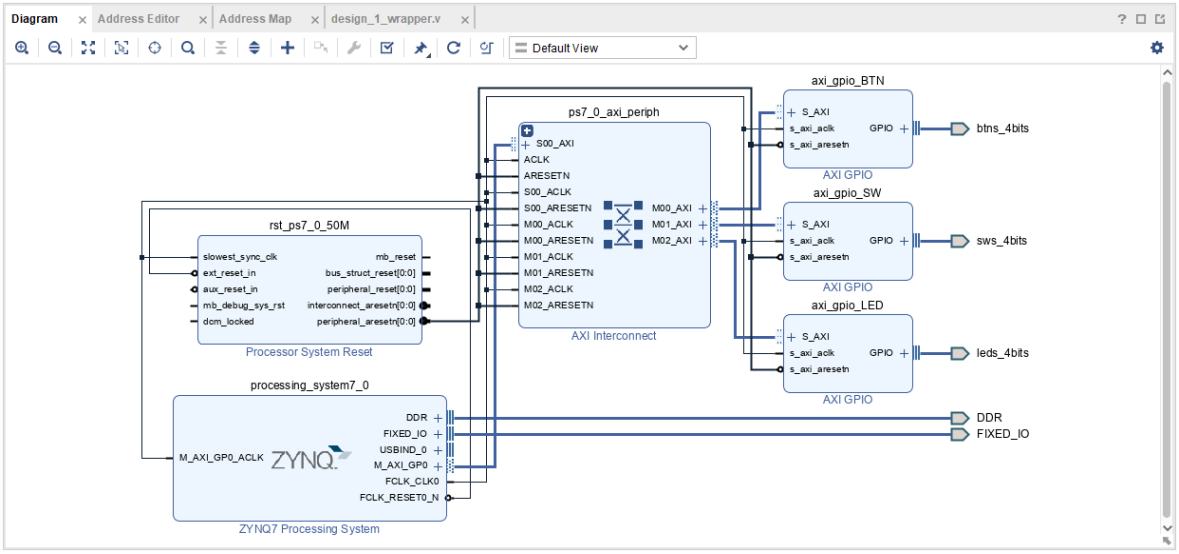
10. Next, there will be two options in the green flag above your block design, Run Block Automation and Run Connection Automation. Run both of these.

11. Change the name of this IP to axi_gpio_BTN.

12. Perform the same steps to add axi_gpio_SW to the design.

13. Regenerate the layout.

14. Now your design should look like the following image:



15. Follow the instructions from Task 1 to generate the bitstream, open Vitis and create a Hello World application and name it Task2, steps x-x.

    (a) If you did not create a new project, you can skip the steps to create a Vitis_WS folder.

16. Open the helloworld.c file and replace all of its code with the code from helloworld_Task2.c in the provided Module's Task2 folder.

17. Right-click on the blue application, Task2_system, and click on build.

18. When the build has completed, right-click the blue application again, and select Run As/1. Launch Application.

19. To confirm the application is running you may either view the text output over the Serial Terminal, or press the left-most button on the Zybo-Z710. This will light up the left-most LED.

    (a) The application reads in the value of the two right-most switches, two left-most buttons, and outputs results to the LEDs.

    (b) The two left-most LEDs show when the left-most buttons are pressed.

    (c) The third LED does not light.

    (d) The fourth LED shows the binary value of the calculation

    (e) Switch Value 0: No calculation

    (f) Switch Value 1: AND

    (g) Switch Value 2: OR

    (h) Switch Value 3: XOR