California State University, Northridge
ECE 520L - System on Chip
Fall 2022

# Lab 2: Vivado IP Integrator

September 13, 2022
Instructor: Saba Janamian

Troy Israel

**Introduction**

The purpose of this experiment was to further our understanding of how to use Vivado and to further understand how to implement IP design blocks. We followed a Xilinx document for setting up AXI IP blocks and connecting them together to create a whole AXI block design.

**Procedure**

Firstly, I created a new Vivado Project. I then when to the IP block creator and add the AXI Quad SPI block directed by the Xilinx document. Next I added the following IPs from the IP block list: AXI UART Lite, AXI Block RAM Controller, AXI Interrupt Controller and the AXI Interconnect. At this point, there are five IP blocks in the vivado window. The next is task is to create interface ports which would be used as connection points if the design that we are building would actually be produced. Inside the interface port we need to modify its mode as well as change its name. Following the interface port, is the task of creating normal ports which will also be modified and changed to clock and reset.

After the ports are created, then we can start connecting ports to the AXI Interconnect block. Because of the name chosen, ACLK gets connected to all the ACLK ports and same goes for the Reset. The AXI Interconnect then gets modified so that the number of master interfaces is expanded from 2 to 5 and the CLK and RESET ports need to also be connected to these new ports. The BRAM controller also gets connected to the clock and reset ports. The other AXI blocks also got their CLK and RESET connected to the main ports.

Next we run the connection automation and select all external interfaces and the BRAM controller for auto connection. Doing so created external ports and a Block Memory Generator. Another external port was needed to be made for the AXI Quad SPI that connects to the external SPI clk. The interrupt signals on the various IPs will be connected together in the concatenation signal block. The 4th input to the concat block will be given a constant value of 0. Inside the address editor we assign all the ports and set the range to 64K. Now we Validate the design and generate the Output Products followed by the HDL wrapper file.

## Test Strategy

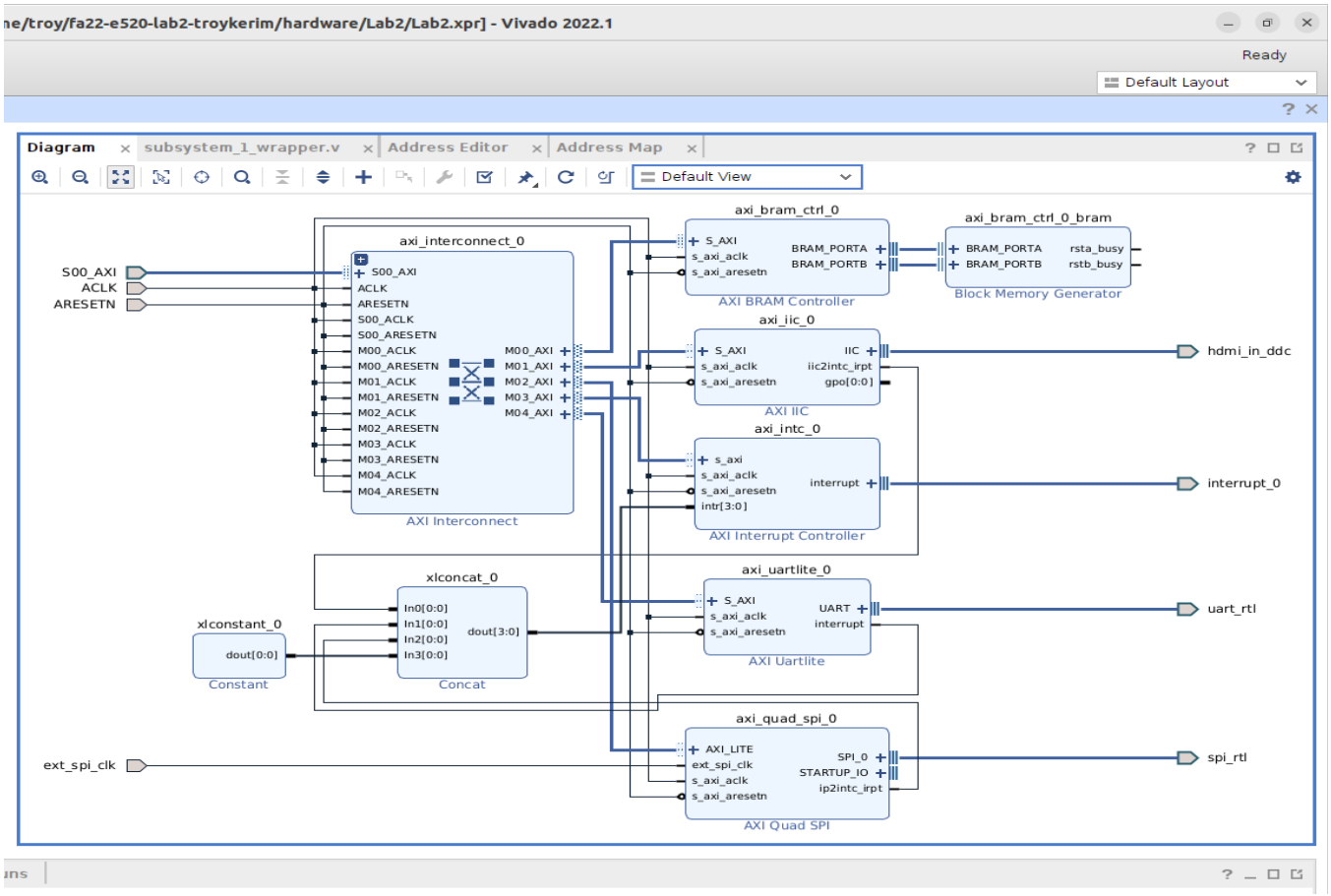


Fig 2.1 Full Block Design

In Fig 2.1, it shows the final overall block design after following the documentation. Initially, only five IP blocks were made. The AXI Interconnect, BRAM controller, IIC, Interrupt, Uart and SPI. The AXI interconnect by default is set to 2 inputs for ACLK and ARESETN but since we have more than two blocks we can expand the Interconnect to accommodate more IP blocks.

The Concatenation block is concatenating all of the interrupt output ports of the IIC, Uartlite and the SPI blocks and then feeding them into the Interrupt controller. Initially only three interrupt ports are given to the concat block. Therefore a Constant block is made and given to the forth input of the concat block.

**Results**

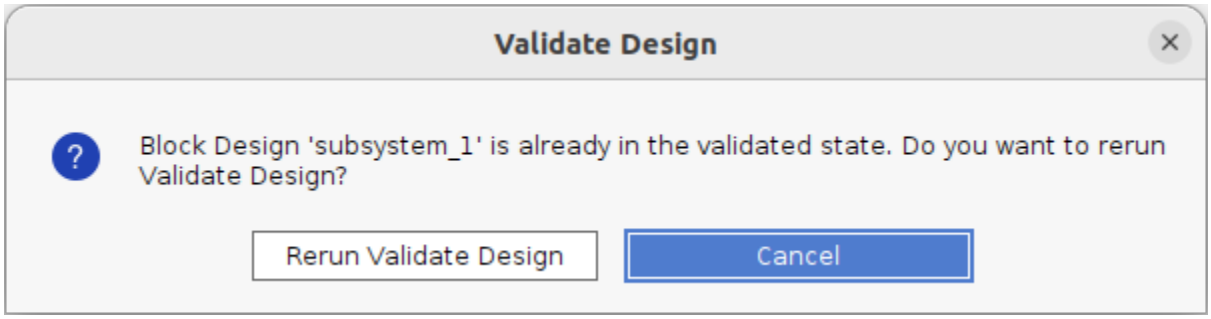


Fig 2.2 Design Validation

Once all the IP blocks have been connected, the design needs to be validated. Fig 2.2 shows that my design was validated. Now that the design was validated, I can create the HDL wrapper file after generating the output product. The next figure shows the one of the last steps I need to do before the HDL file.

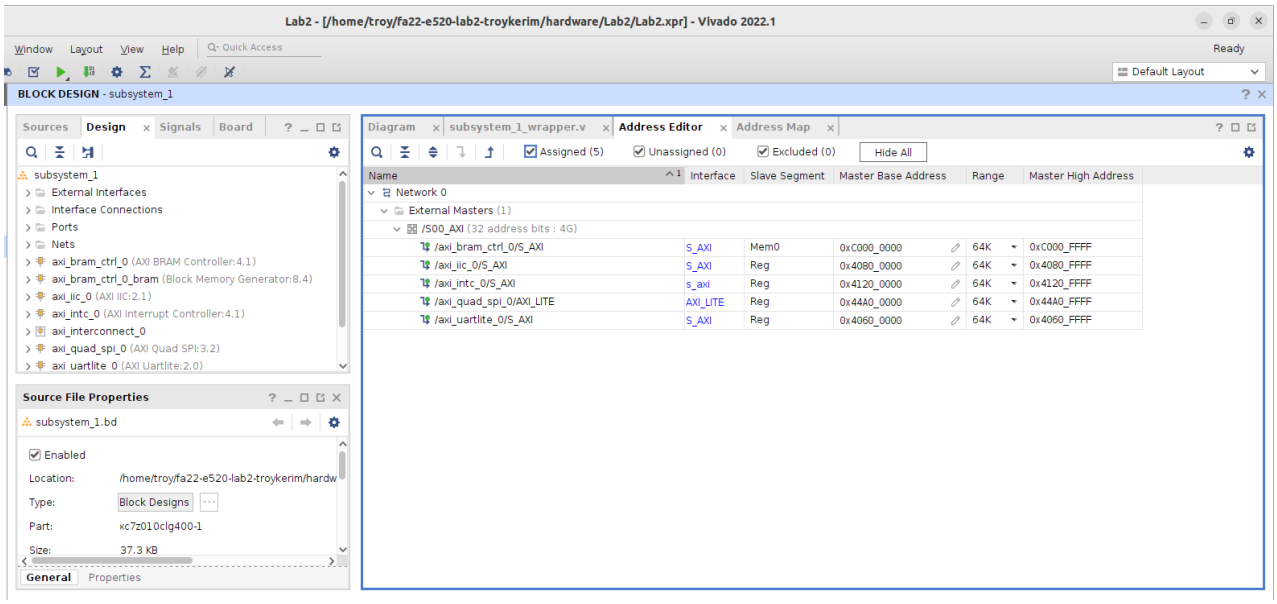


Fig 2.3 Address Editor

Fig 2.3 shows the Address Editor that was set up before validation. Initially one of the values was not set 64k, most likely because of vivado but it is an easy fix. After the Address Editor was created, I ran the Generate Output Product. When this finishes I can then create the HDL wrapper file and the lab is overall finished. There is the step of generating the Bitstream but this lab we do not have access to the XDC file. This lab is originally from xilinx and their own documentation. Therefore, they used a specific XDC file which we do not have. So no bitstream for this lab.

**Conclusion**

In this lab we learned how to create and add IP design blocks as well as learned how to create input and output ports. With regard to IO ports we also learned the difference between Interface port and traditional port that vivado supports. An interface port allows a system that we design to be created as a bus instead of a regular port which is just a wire.