Khoa A. Vu

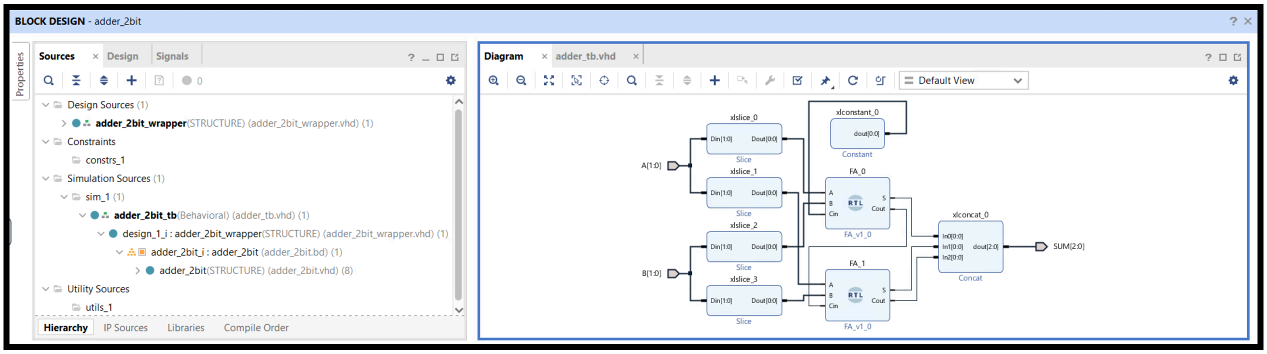
Professor Sayadi

CECS 341 (Sec. 01)

08 March 2024

Lab 1

Generally speaking, Lab 1 promotes the use of the Vivado tool and IP Integrator to complete all the given tasks and instructions. In doing so, I applied and learned how to use a block design to portray how the processor components are connected, extracting a subset of bus and using different types of IPs to slice and concatenate, along with establishing a VHDL module and HDL wrapper that instantiated in test-bench and simulated it. With that being said, one key takeaway for me is that Vivado is a very intricate, detail-oriented tool for constructing hardware and processors. For example, I made a mistake with initializing the incorrect level for the constant generator and it got the program to output the wrong value of the SUM port; thus, to fix the issue, I had to delete the whole constant generator and then recreate another one entirely to reconnect to get it to work. This shows that one tiny mistake can make it difficult for the user to debug and get the expected answer. After getting the program to work, the following snapshots are the results of my lab:



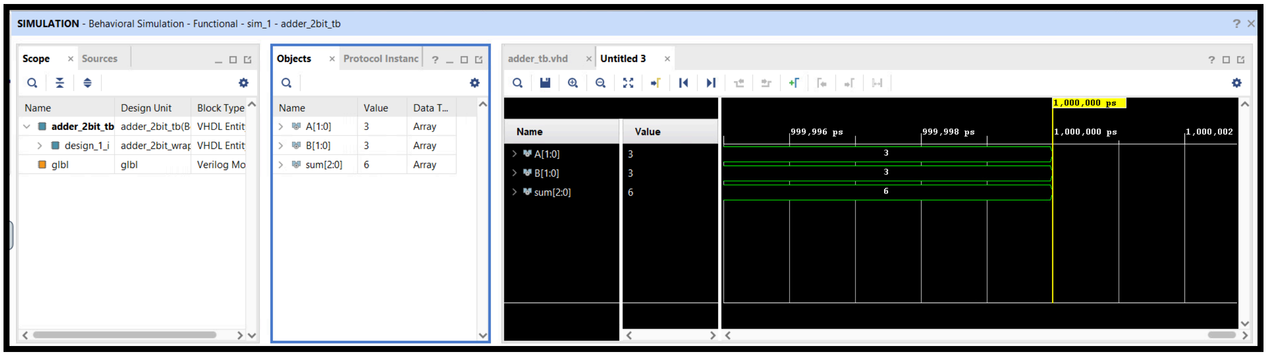
BLOCK DESIGN - adder_2bit

Sources | Design | Signals

- Design Sources (1)
  - **adder_2bit_wrapper**(STRUCTURE) (adder_2bit_wrapper.vhd) (1)
- Constraints
  - constrs_1
- Simulation Sources (1)
  - sim_1 (1)
    - **adder_2bit_tb**(Behavioral) (adder_tb.vhd) (1)
      - design_1_i : adder_2bit_wrapper(STRUCTURE) (adder_2bit_wrapper.vhd) (1)
        - adder_2bit_i : adder_2bit (adder_2bit.bd) (1)
          - adder_2bit(STRUCTURE) (adder_2bit.vhd) (8)
- Utility Sources
  - utils_1

Block Design

On the surface, the diagram of the block design showcases a simple building of the processor components using block designs and IPs. However, the Source displays the files stacked on top of each other to get the program and generate the simulation. Starting with the wrapper in the test bench that is at the top, the program traverses through the file and the wrapper then calls on the Simulation Source to run. In the Simulation Source, the program traverses the adder_2bit_tb file for the information code and then uses that information to generate the output in the simulation.



SIMULATION - Behavioral Simulation - Functional - sim_1 - adder_2bit_tb

Scope

Name	Design Unit	Block Type
adder_2bit_tb	adder_2bit_tb(B	VHDL Entit
design_1_i	adder_2bit_wrap	VHDL Entit
glbl	glbl	Verilog Mo

Objects

Name	Value	Data T...
A[1:0]	3	Array
B[1:0]	3	Array
sum[2:0]	6	Array

Untitled 3

Name	Value
A[1:0]	3
B[1:0]	3
sum[2:0]	6

Simulation Waveform

Once the Block design was constructed, I ran the simulation function to get the following waveform above in the snapshot. In the simulation, the waveform generated the outputs of 3, 3, and 6 as a result of the inputs that came in from the codes of adder_2bit_tb files to ports A and B.

The information that ports A and B received was delivered throughout the processor components to be sliced and concatenated to produce the outputs above. Another thing to notice while running the simulation is that the adder_2bit_tb file needs to be in VHDL file type for the program to run the simulation smoothly.

In conclusion, despite encountering some limitations along the way such as making the program run and generating the correct outputs, I was able to overcome these challenges and learned more about how Vivado worked, especially its block designs, IP Integrator, and simulation.