CECS341_Lab#2

Twan Tran, Brian Thi, Jae Bum Jang

Group# 14

CECS 341

Lab 2 - Adder - Subtractor

With the basic knowledge learned from Lab 1, prelab tutorial, and Into to VHDL Lecture, we managed to complete block design. Next we went on to develop a test bench to verify our block design. It was quite challenging at first for us, but with some research and help from the TA, we got it running.

For the Prelab Part 1, we learned how to develop a separate behavioral VHDL program. We created an ENTITY statement that specifies inputs and output of a circuit that uses PORT statements to define those inputs and outputs. Then created the ARCHITECTURE statement, which describes the behavior of the circuit. In this case we use Boolean expressions such as: AND, OR, and NOT.

We also learned how to develop a TestBench that verifies the functionality of a Design, in this case our VHDL program. In the Testbench we created a Component that has port A,B,C,F as well as the input and output. Then created functional descriptions in the architecture body. There we also created a "wait statement" that waits for 10 nanoseconds before execution. Ran the simulation and observed the waveform result.

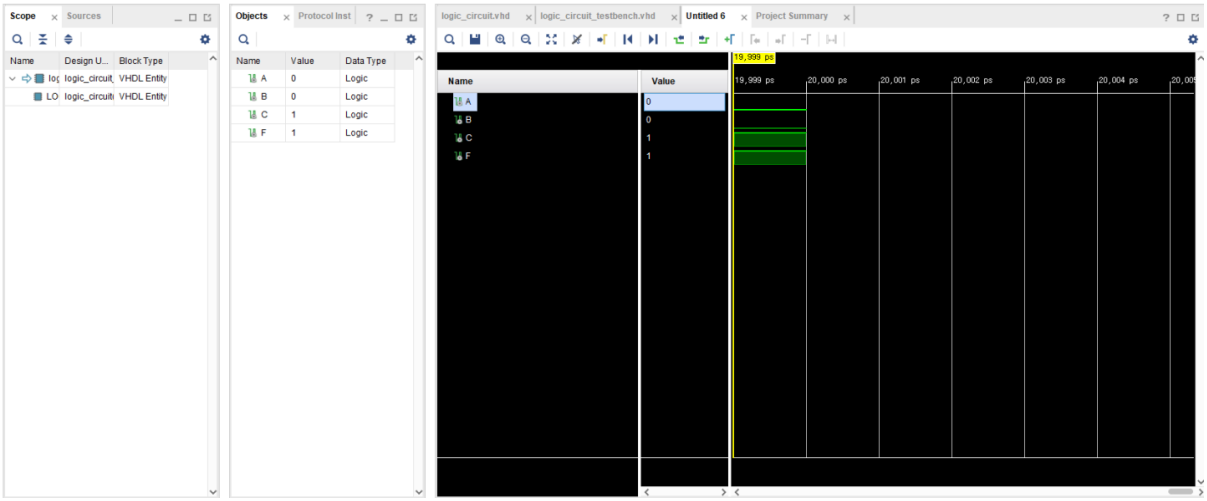
With Part 2, first we had to utilize the block designing knowledge from Lab#1, to make a new form. From Lab#1, it was basically 2 bit adder from a 1 bit full adder. In this Lab#2, we add 2 more adders in order to create a 4 bits adder block design. It was almost similar concept with Lab#1, but with additional slices and adders per each ports. Plus, we used utility vector that connenct with input of Port B, and C to make a connection on the full adder along with input of Port A. Detailed description of the block design attached below.

On simulation source, the test bench assigned values to A, B, C to output Sum. A and B are the 4 bits binary numbers, and C is the Carry. The tests are on the simple operation of adding/subtracting 2 of 4-bits binary numbers, while C = 0 or C = 1. It also tested the design on overflow cases such as ‘1111 + 1111’.

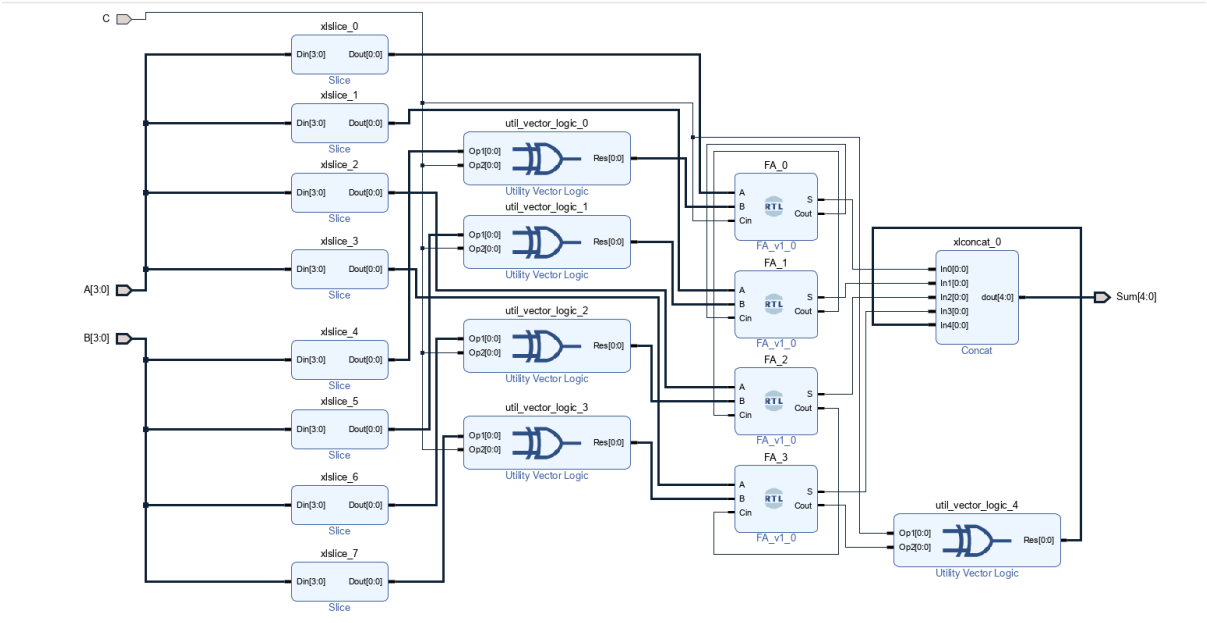
We have learned more about the fundamental of adding 4-bits binary numbers, how to use Vivado and VHDL more effectively, creating Designs then made up test cases

for them.

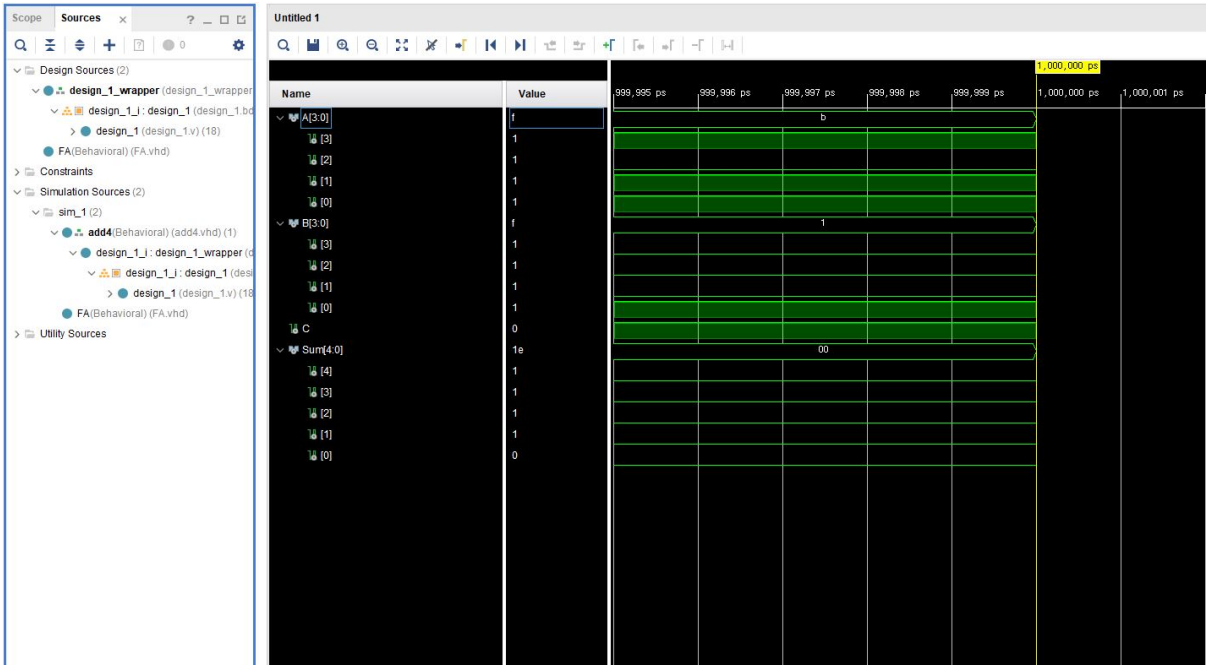
<Part 1, Waveform>



<Part 2, Block Design>



<Part 2, Waveform>


Scope
Sources
Design Sources (2)
design_1_wrapper (design_1_wrapper
design_1_i : design_1 (design_1.bd
design_1 (design_1.v) (18)
FA(Behavioral) (FA.vhd)
Constraints
Simulation Sources (2)
sim_1 (2)
add4(Behavioral) (add4.vhd) (1)
design_1_i : design_1_wrapper (d
design_1_i : design_1 (desi
design_1 (design_1.v) (18
FA(Behavioral) (FA.vhd)
Utility Sources
Untitled 1
Name
Value
A[3:0]
[3]
[2]
[1]
[0]
B[3:0]
[3]
[2]
[1]
[0]
C
Sum[4:0]
[4]
[3]
[2]
[1]
[0]
f
1
1
1
1
f
1
1
1
1
0
1e
1
1
1
1
0
1,000,000 ps
999,995 ps
999,996 ps
999,997 ps
999,998 ps
999,999 ps
1,000,000 ps
1,000,001 ps
b
1
00