Lab 8: Registers

COEN/ELEN 21L

Alex Heiler and Francesca Narea

3/10/17

Introduction:

In this lab, we translated our logic from the pre lab into Quartus schematics and created four different registers. We were challenged to work with the Altera FPGA to display inputs, outputs and used switches as well as push button to perform different operations. As well, we oversaw the use of debugging LEDs as a replacement for the waveforms we have used in previous labs. The purpose of this lab was to apply our knowledge in order to create a simple calculator.

Prelabs:

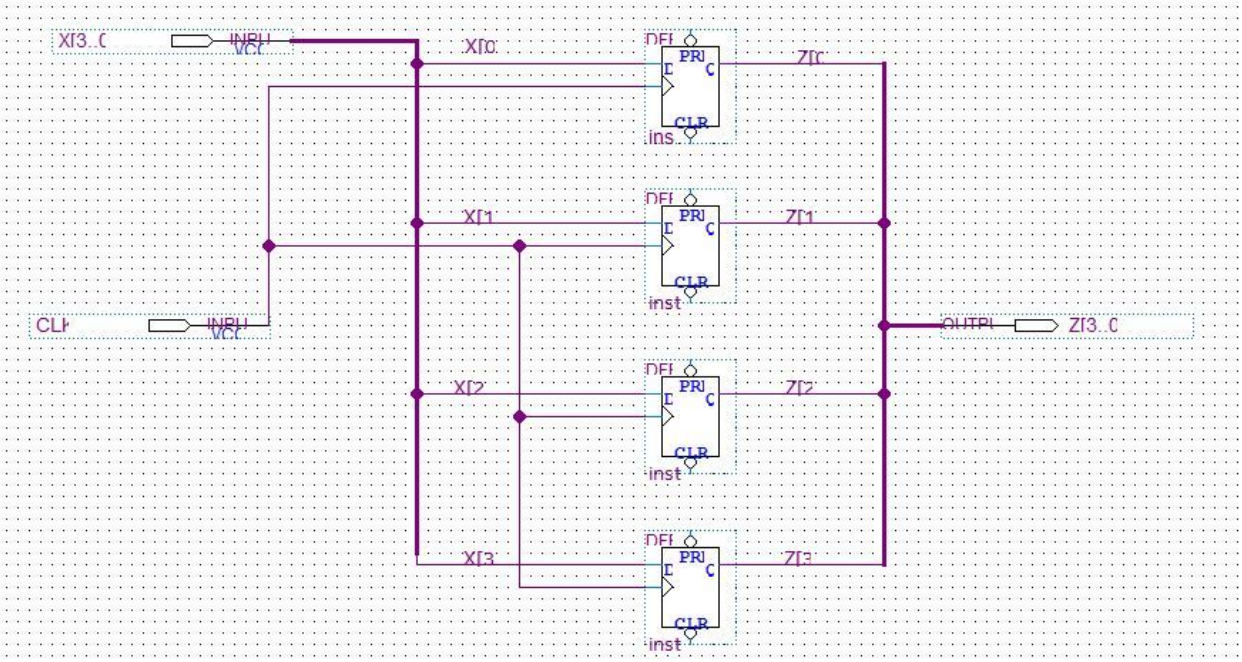

Figure 1.1: Pre lab for Alex (page 1)

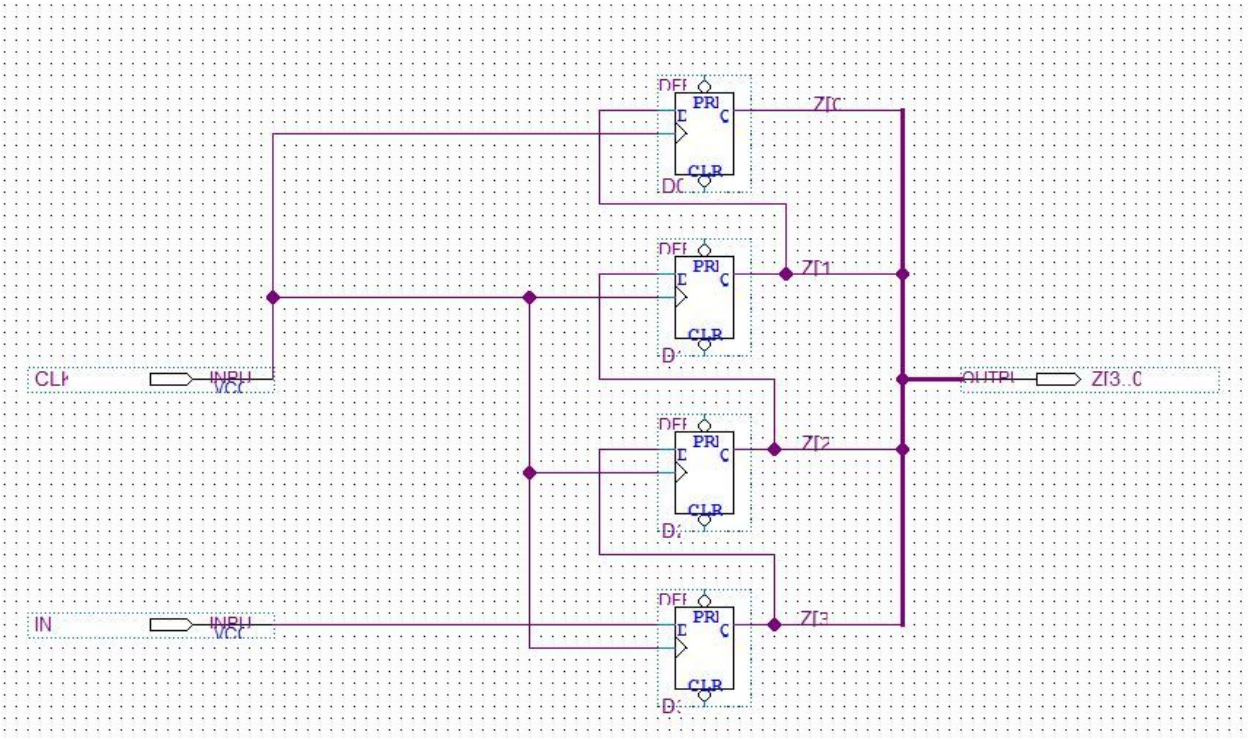

Figure 1.2: Pre lab for Alex (page 2)

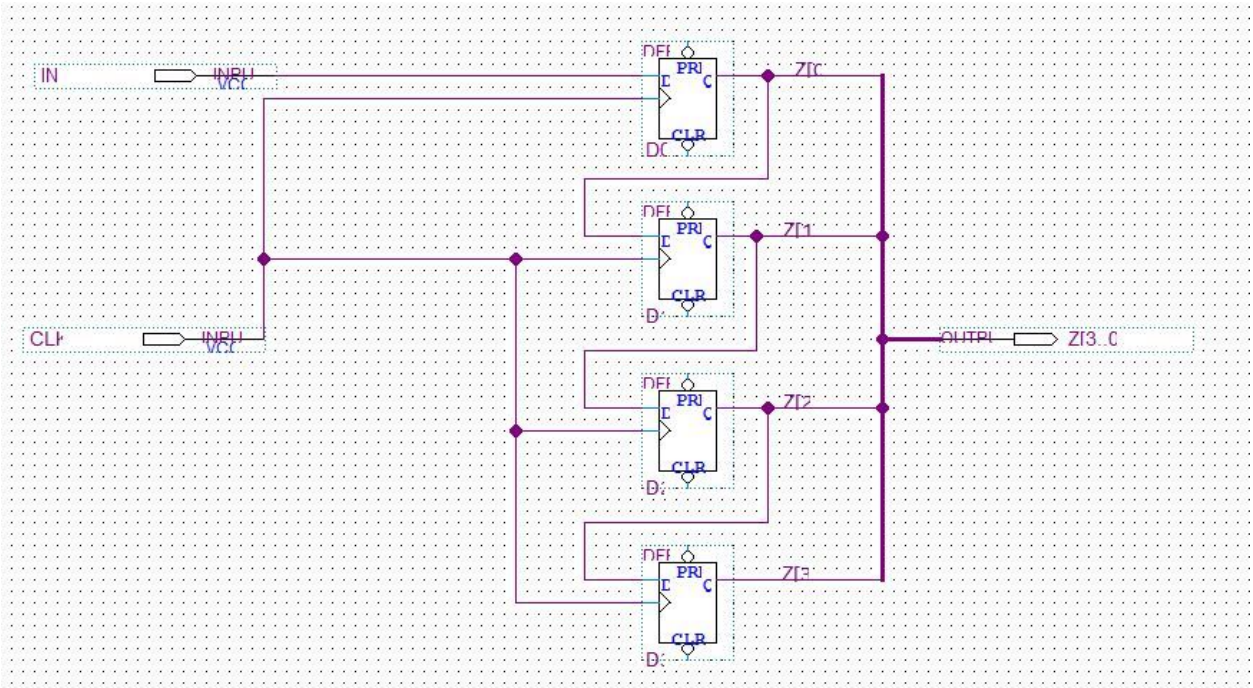

Figure 1.3: Pre lab for Alex (page 3)

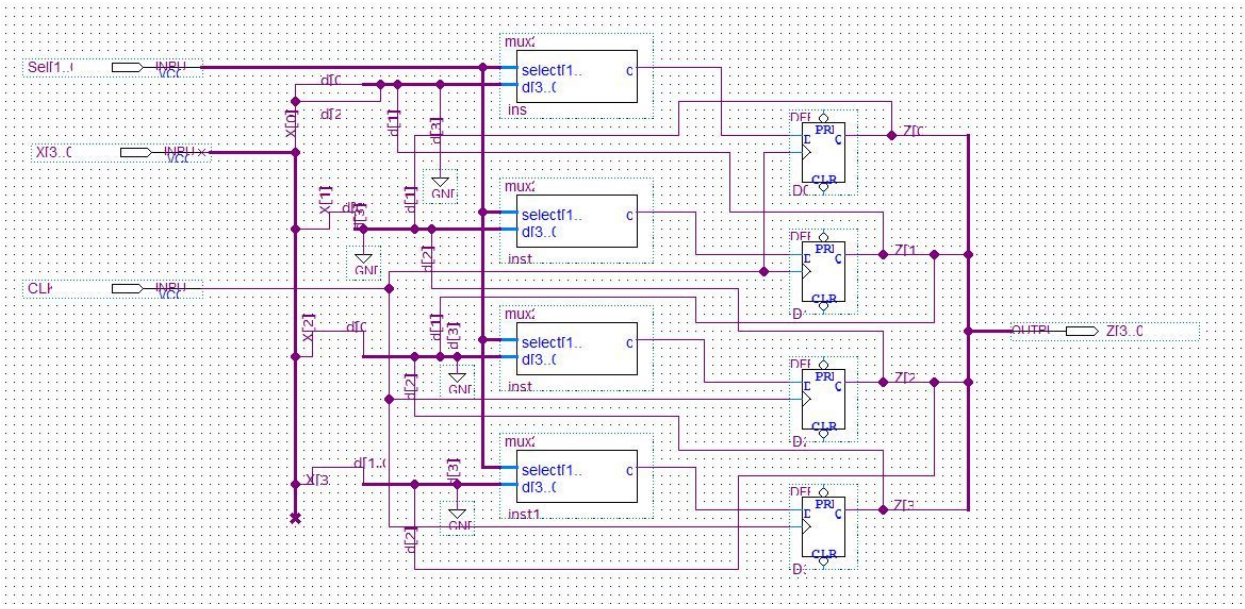

Figure 1.4: Pre lab for Alex (page 4)

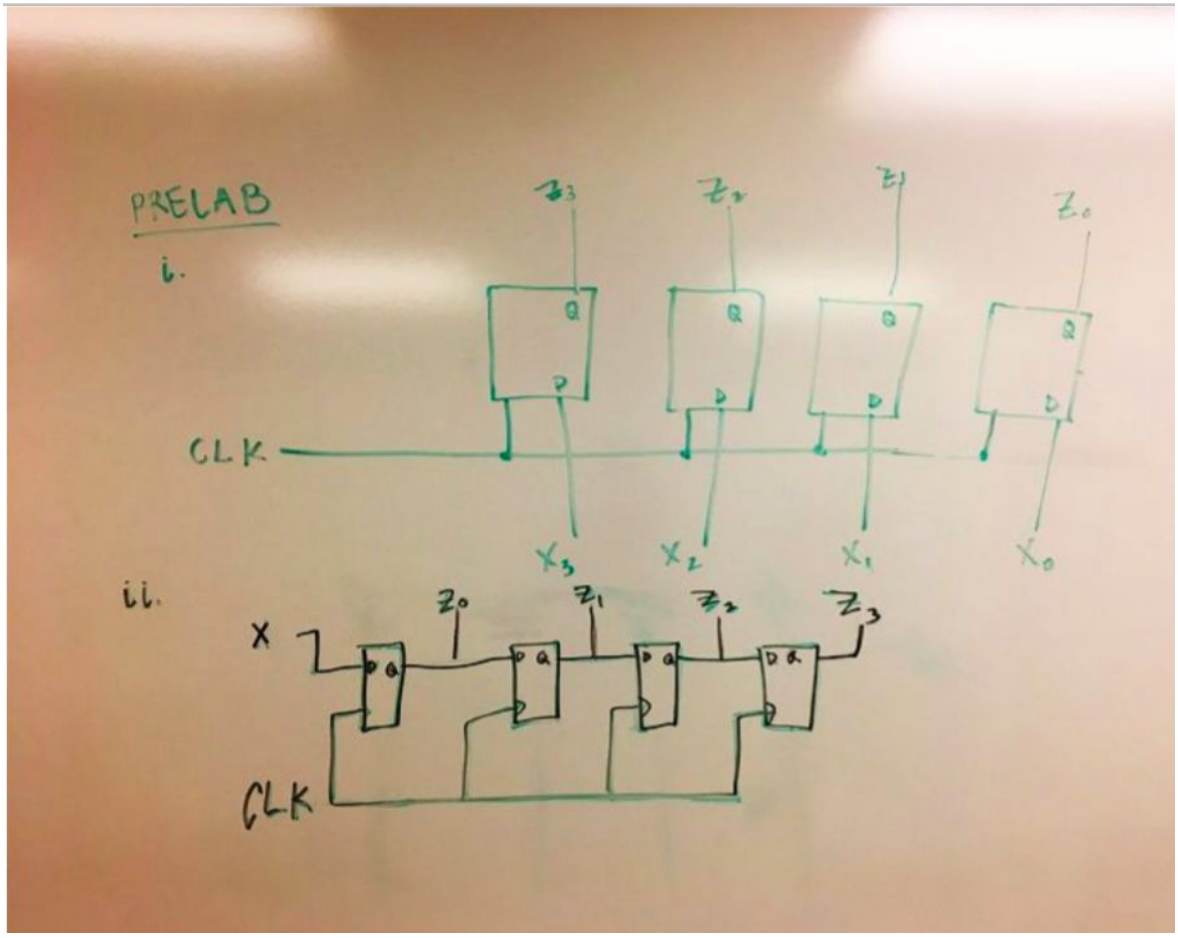

Figure 1.5: Pre lab for Francesca (page 1)

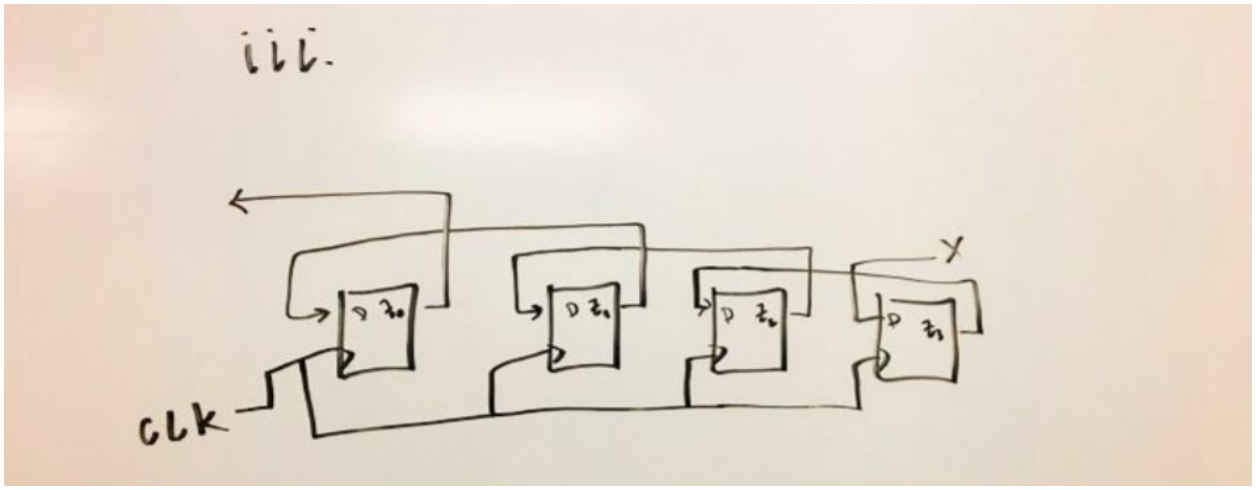

Figure 1.6: Pre lab for Francesca (page 2)

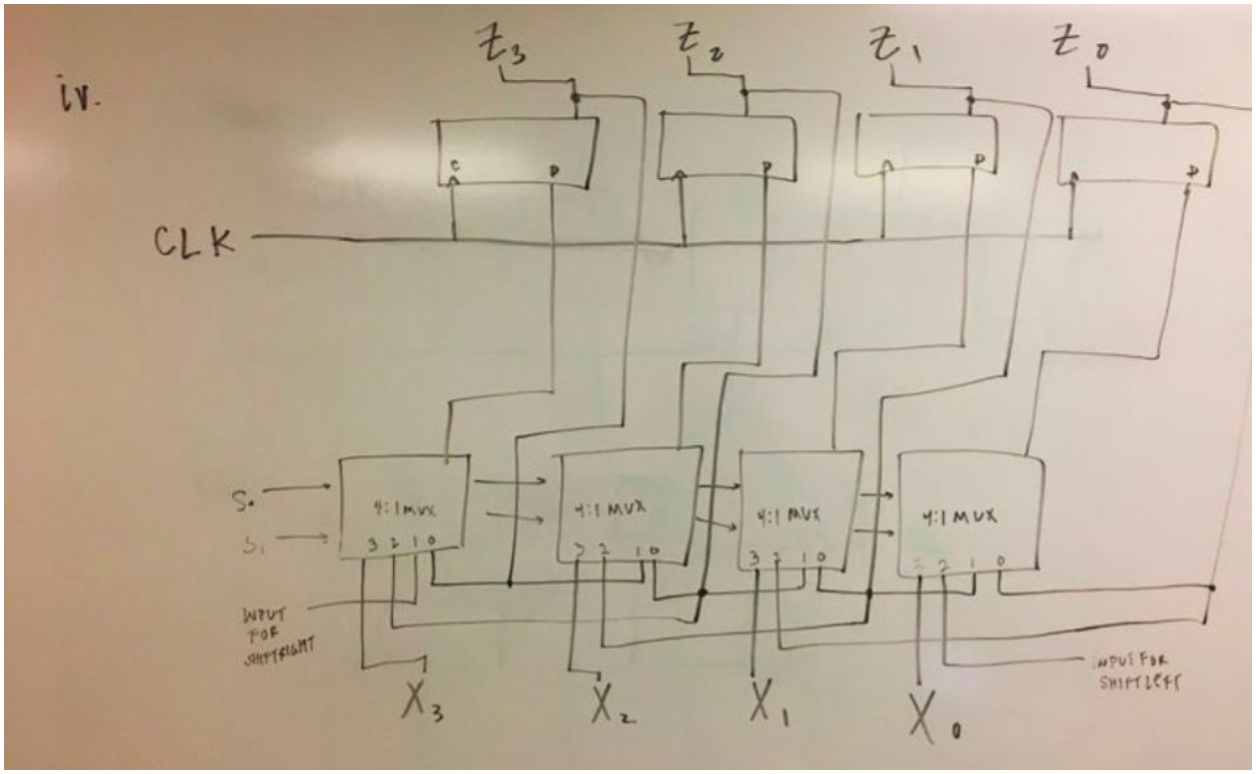

Figure 1.7: Pre lab for Francesca (page 3)

Procedures:

First, we drew our schematics in Quartus for the three different registers from our pre lab--the load, shift-right and shift-left. In Part 1, with the data storage register, we connected our 4-bit input to switches and the clock input to a push button. In order to observe the outputs, we connected LEDs and the seven segment hexadecimal display. Next, we connected our inputs and outputs in the same way for our right-shift and left-shift register. Lastly, we designed the schematic for the Universal Register using muxes.

Load Register

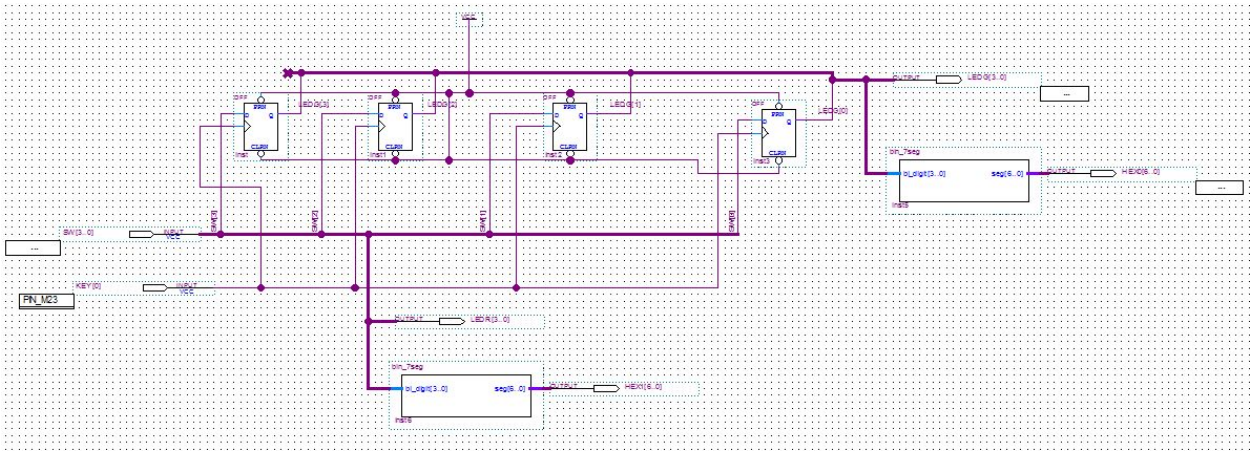

Figure 2.1: Load Register Schematic

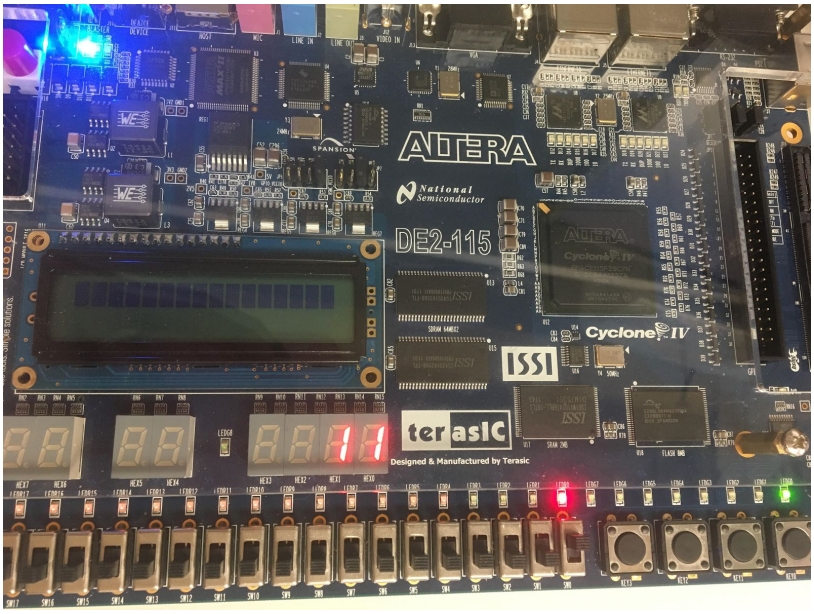

Figure 2.2: Working load register on Altera FPGA

Test Plan:

1. Change input to 0000 and press the button
2. Verify output remains 0000
3. Change input to 1111 and press the button
4. Verify output changes to 1111

Right-Shift Register

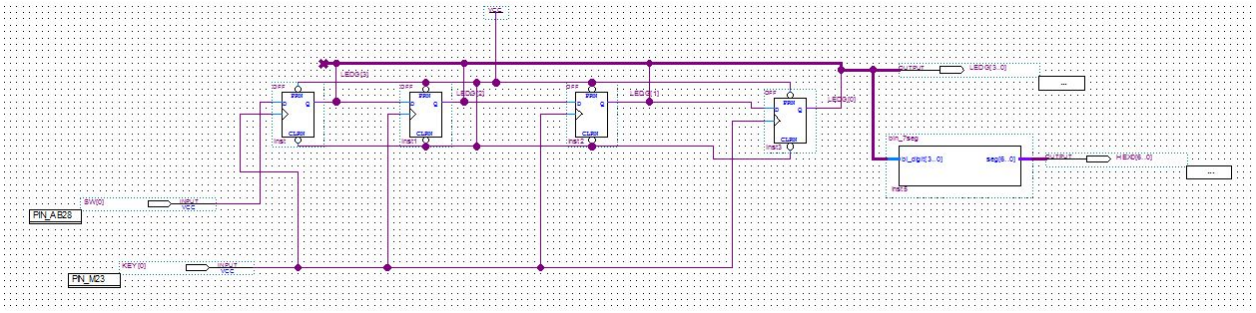

Figure 3.1: Right-Shift Register Schematic

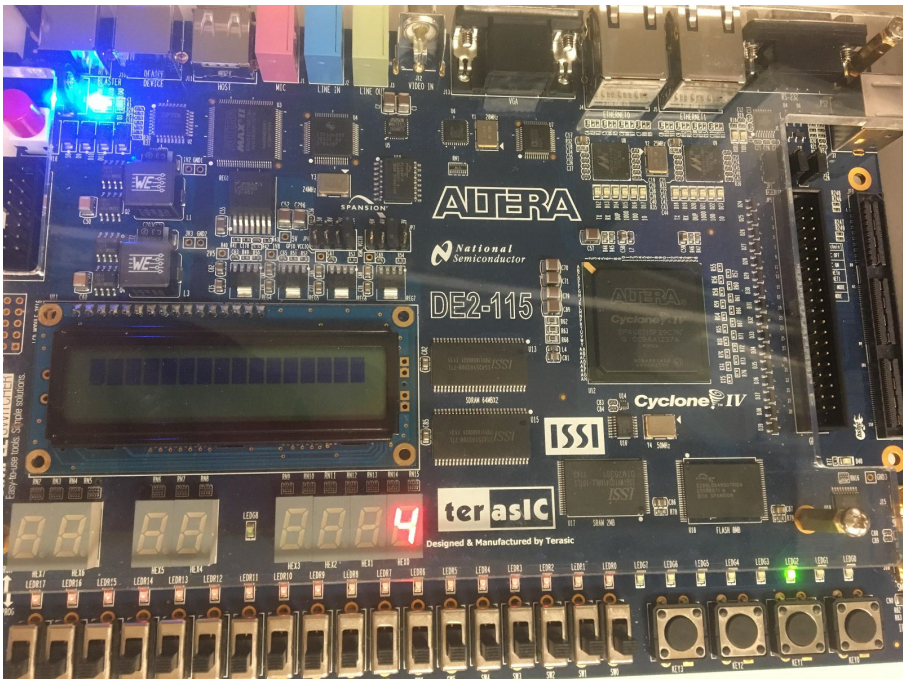

Figure 3.2: Working left-shift register on Altera FPGA

Test Plan:

1. Change input switch to 0 and press the button
2. Verify output remains 0000
3. Change input switch to 1 and press button
4. Verify output is 0001
5. Change input switch back to 0, press button multiple times
6. Verify output cycles to 0010, 0100, 1000, then 0000

Left-Shift Register

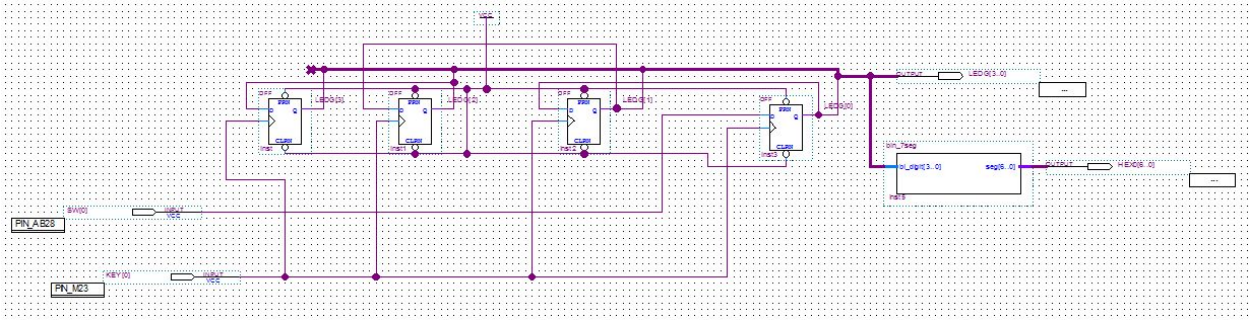

Figure 4.1: Left-Shift Register Schematic

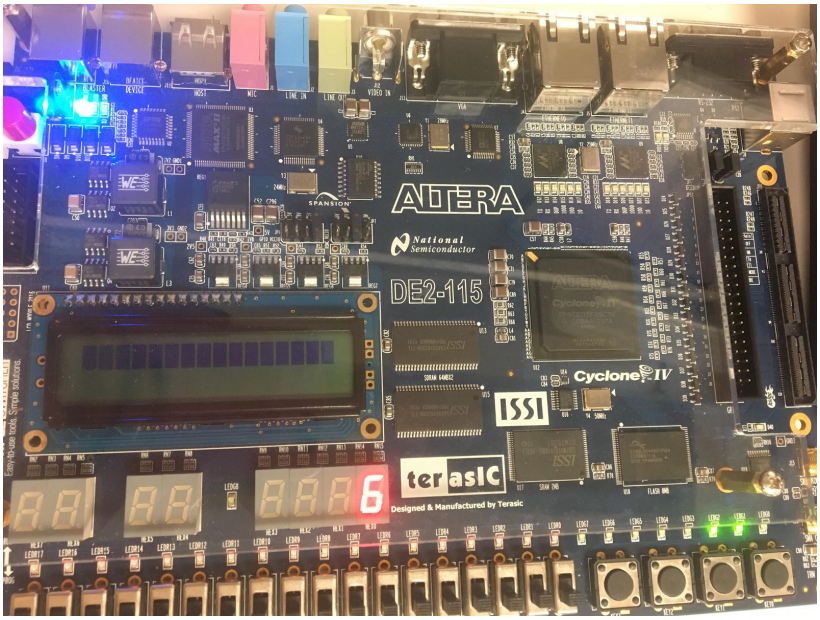

Figure 4.2: Working right-shift register on Altera FPGA

Test Plan:

1. Change input switch to 0 and press the button
2. Verify output remains 0000
3. Change input switch to 1 and press the button
4. Verify output changes to 1000
5. Change input switch back to 0 and press the button multiple times
6. Verify output cycles to 0100, 0010, 0001, then 0000

Universal Register

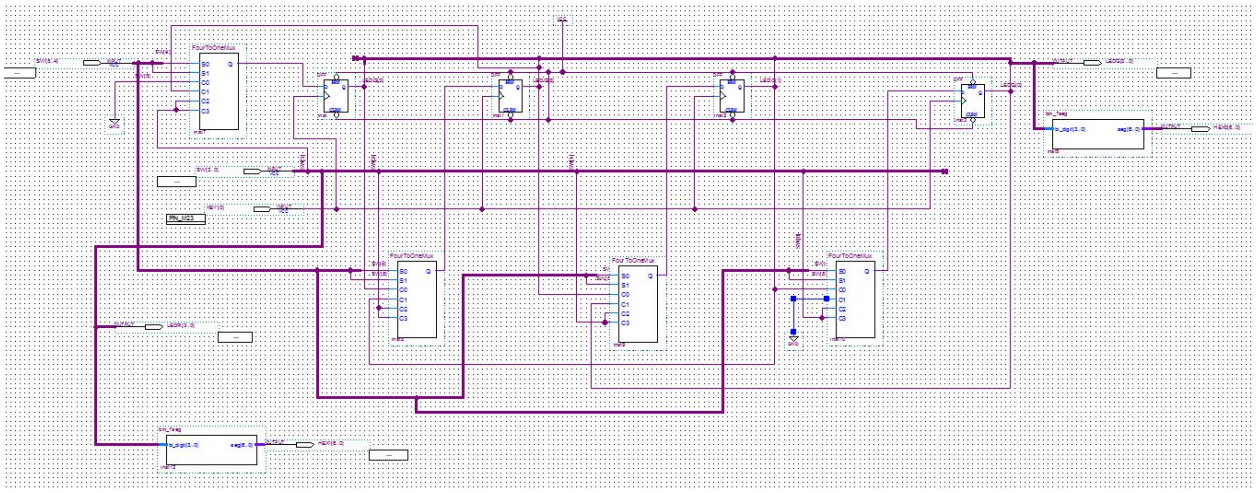

Figure 5.1: Universal Register Schematic

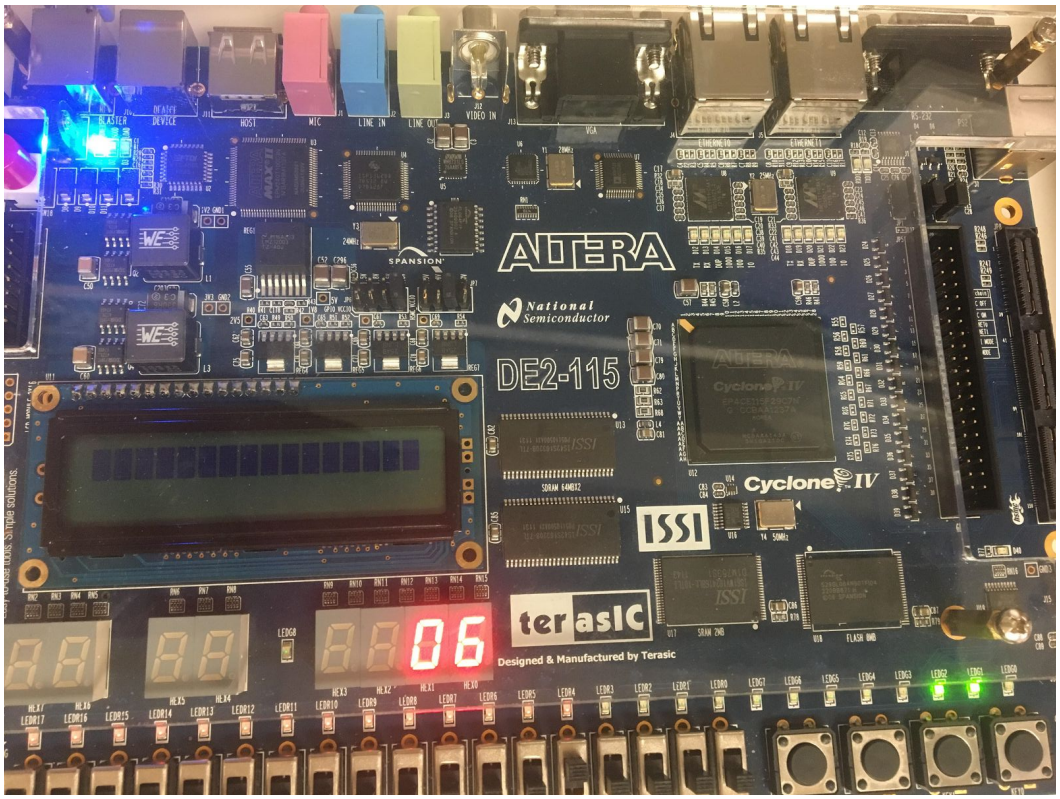

Figure 5.2: Working universal register on Altera FPGA

Test Plan:

1. Change SW[5] to high, use test plan for the load register
2. Change SW[5] to low and SW[4] to high, use test plan for the left-shift register
3. Change SW[5] and SW[4] to low, use test plan for the right-shift register

Conclusion:

During this lab, we worked heavily off of our pre lab schematics in order to design four different types of registers. After having completed the bulk of the work during the pre lab, we were able to then take our logic from the schematics and display this on the Altera FPGA again using a push button, LEDs and the seven segment display, as we have in previous labs. In the end, we successfully created a multiplier/divider through the universal register, which incorporated the other registers we designed working up to the final schematic. We learned about the importance of using test plans to verify that each step of the lab (each register we created) would output the expected result, in order to combine the registers in a more compact program to multiply and divide numbers.

Questions:

3. How would you modify the circuit to allow opcodes for both logical and circular shifts to the left and right? A shift left or right will always leave a "vacant" bit on one side and will shift a bit out to no destination on the other side. A logical shift will shift a zero into the vacant location and lose the bit that is shifted out of the other side of the register. A circular shift will shift the last bit out of one side of the shift register and back into the vacant bit on the other side (like PAC-MAN).

For a circular right-shift, you would connect the output of the least significant register to the input of the most significant register. For a circular left-shift, you would connect the output of the most significant register to the input of the least significant register. This will allow the bits to cycle through.

4. How would you use your circuit to load a 4-bit data word (parallel load) and create a serial bit sequence that could be the input to a serial communication circuit such as USB.

We would load the 4-bit word into the registers and then connected the output of the least significant register to the input of the serial communication circuit. Then, we would right-shift the bits to send one bit at a time.

5. How would you add logic to allow for getting Z= 3X.

Use the left-shift register circuit to multiply the input by 2 and then use a 4-bit adder to add the output of the left-shift register and the initial input of the register. This will give you 2X + X = 3X.