Oakland University

School of Engineering And Computer Science

CSI 5760: Embedded System Design with FPGA

Winter 2023

Professor Ganesan

Hansen Shamoon & Deepak Neupane

Lab 4 - Counters

11/05/2023

Submission notes

We created a separate Quartus project for each part of the lab. In the submission, we have included a compressed folder containing all the projects for Lab 4. Included in each project are following items:

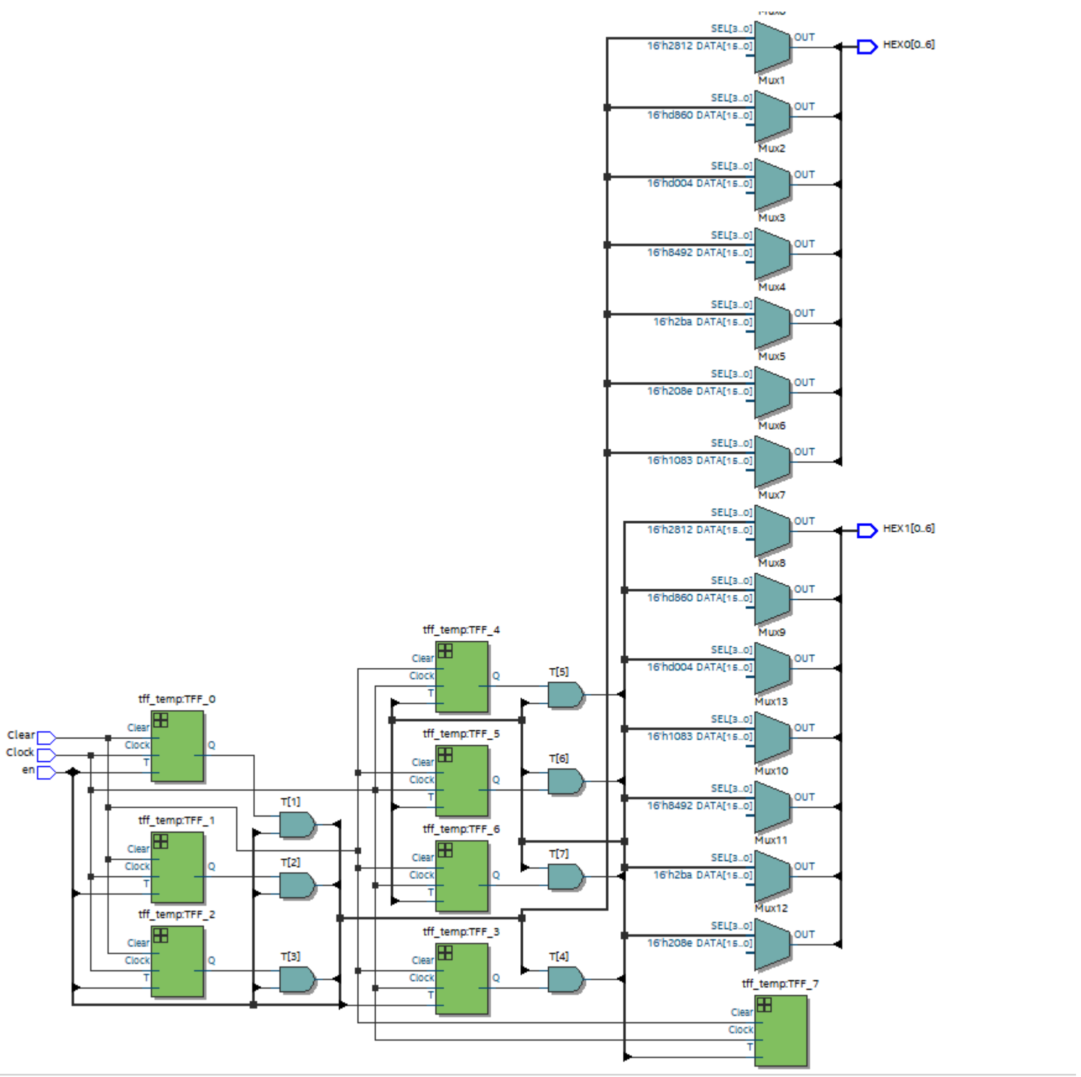
1. Quartus prime project files
2. VHDL code for the lab solution
3. Pin Assignment files exported into csv format

For this lab, we implemented circuits as instructed in the lab manual.

Part 1: Implement a T-type flip flop

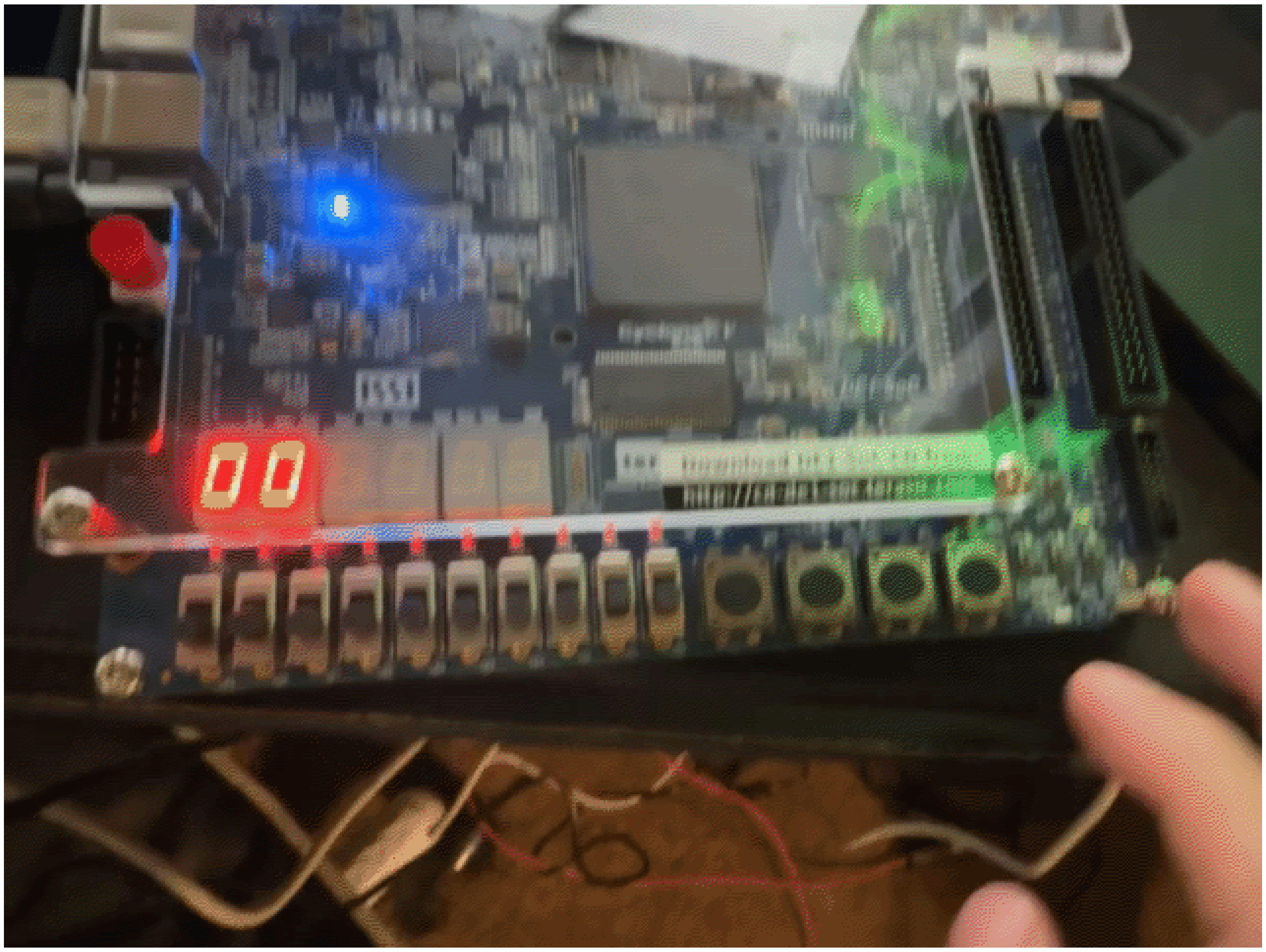
In this part, we are to implement an 8 bit synchronous counter that uses four T-Type flip flops. The counter increments its value on each positive edge of the clock signal if the enable signal is high. The counter is reset to 0 if the synchronous clear input is low.

After implementing the flip flop and the counter we can see the circuit synthesized is as below:



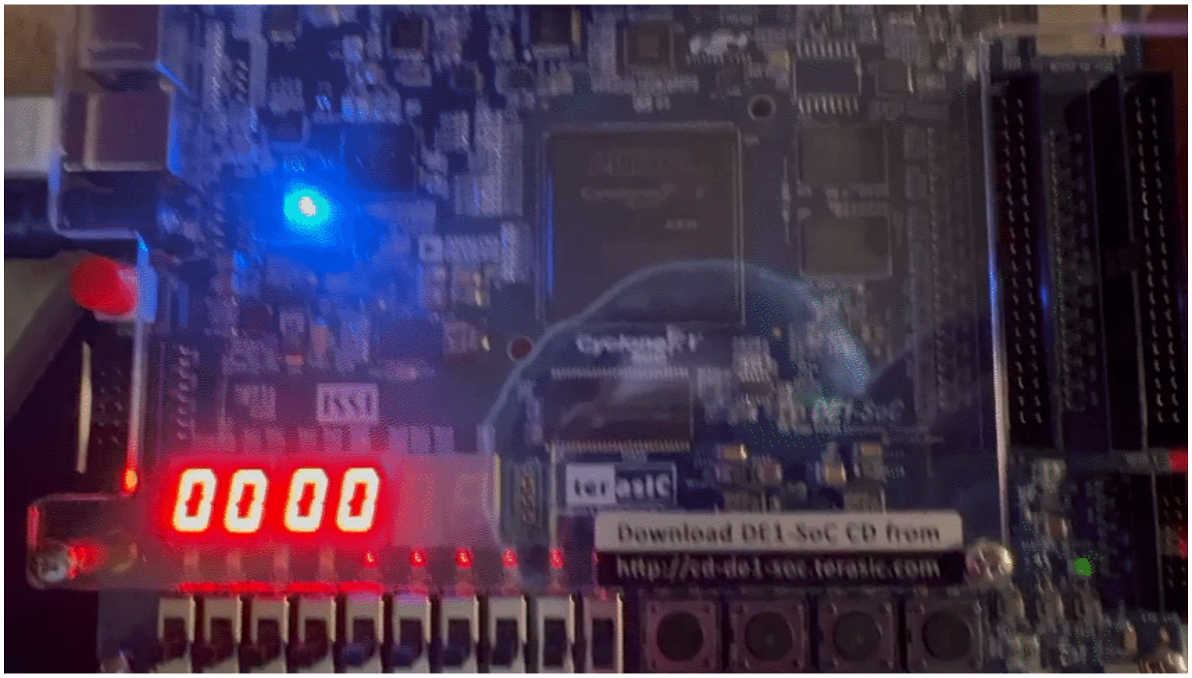
After mapping the outputs and inputs to the switches and hex displays, the expected behavior of the circuit was to display the counter incremented with button click and the reset to reset the counter back to 0. However, we could not observe the expected behavior. We saw the increment in 4s and while the output is shown, the counter is not incrementing as expected.

Shown below is the how the circuit behaves after programming the board:



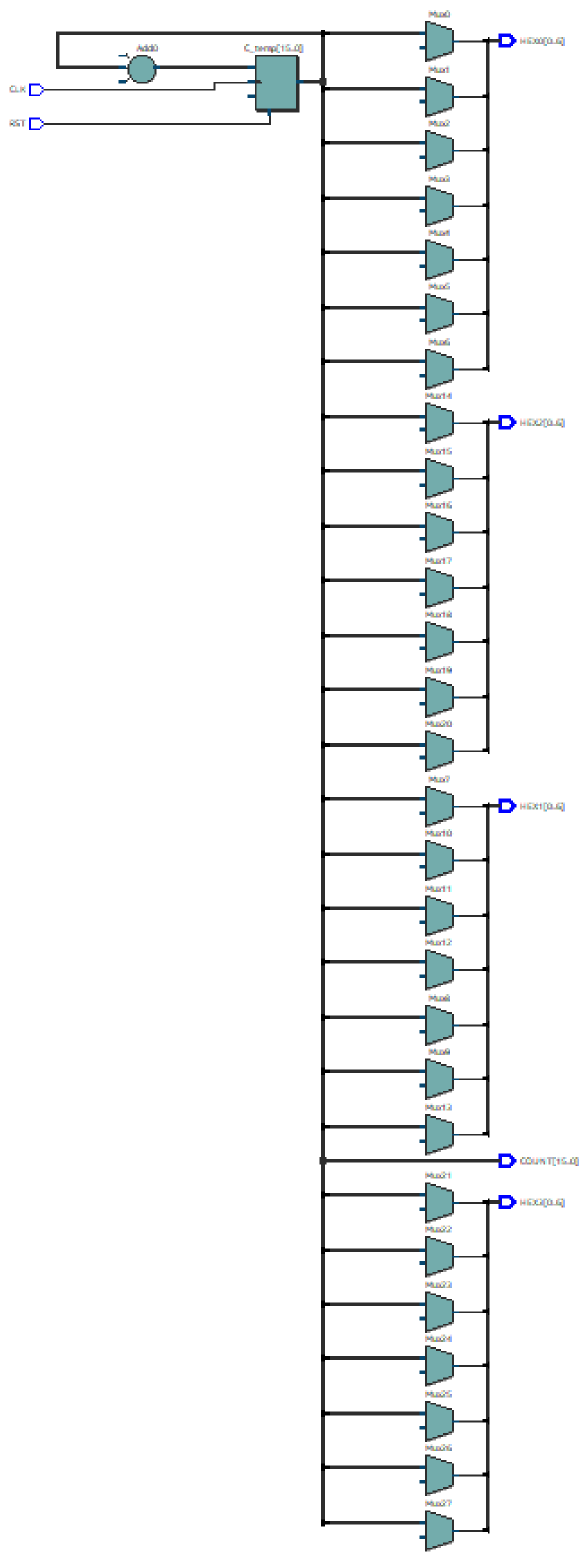
Part 2: Alternate way to implement a counter

In this part of the lab, we implement a counter by using a register and adding 1 to its value. This is just another way to implement the counter we did for part 1. We use the key on the De1SoC board to increment the counter and a clear switch to reset the counter to 0.



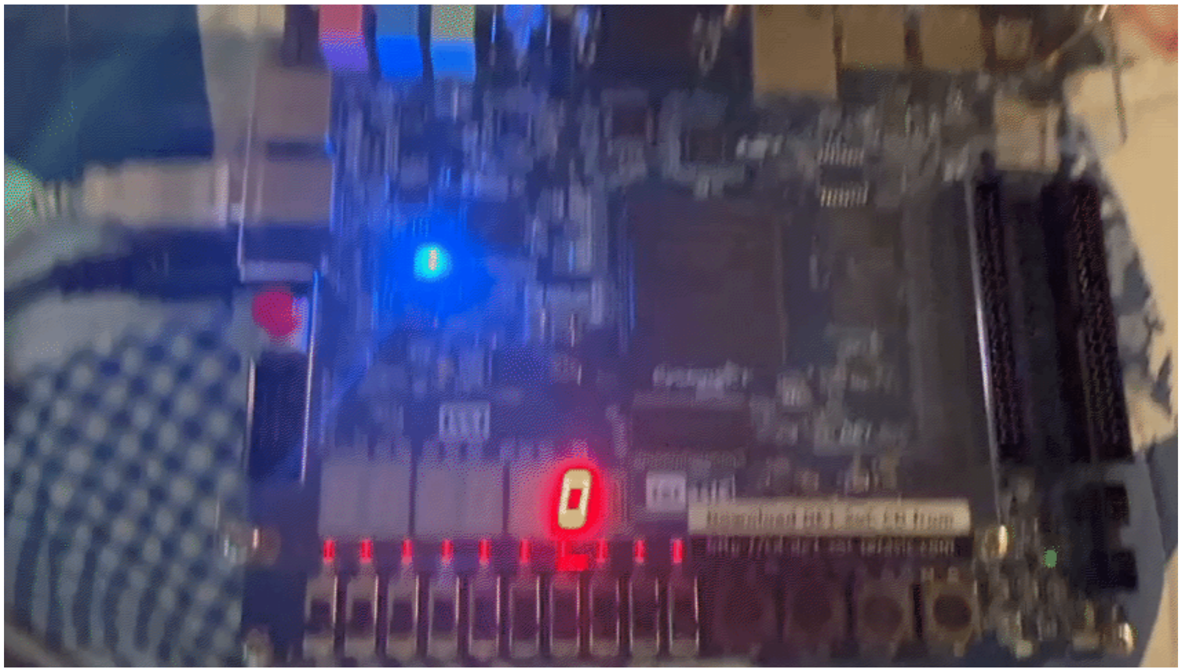
In this part we switched from HEX0-3 to HEX2-5 because HEX1 in our board is defective. Reset was set to switch 9.

Shown below is the screenshot of the circuit as shown in the RTL viewer:


Add0
C_temp[15..0]
CLK
RST
Mux0
HEX0[0..6]
Mux1
Mux2
Mux3
Mux4
Mux5
Mux6
Mux14
HEX2[0..6]
Mux15
Mux16
Mux17
Mux18
Mux19
Mux20
Mux7
HEX1[0..6]
Mux10
Mux11
Mux12
Mux8
Mux9
Mux13
COUNT[15..0]
Mux21
HEX3[0..6]
Mux22
Mux23
Mux24
Mux25
Mux26
Mux27

Part 3: Display digit using counter

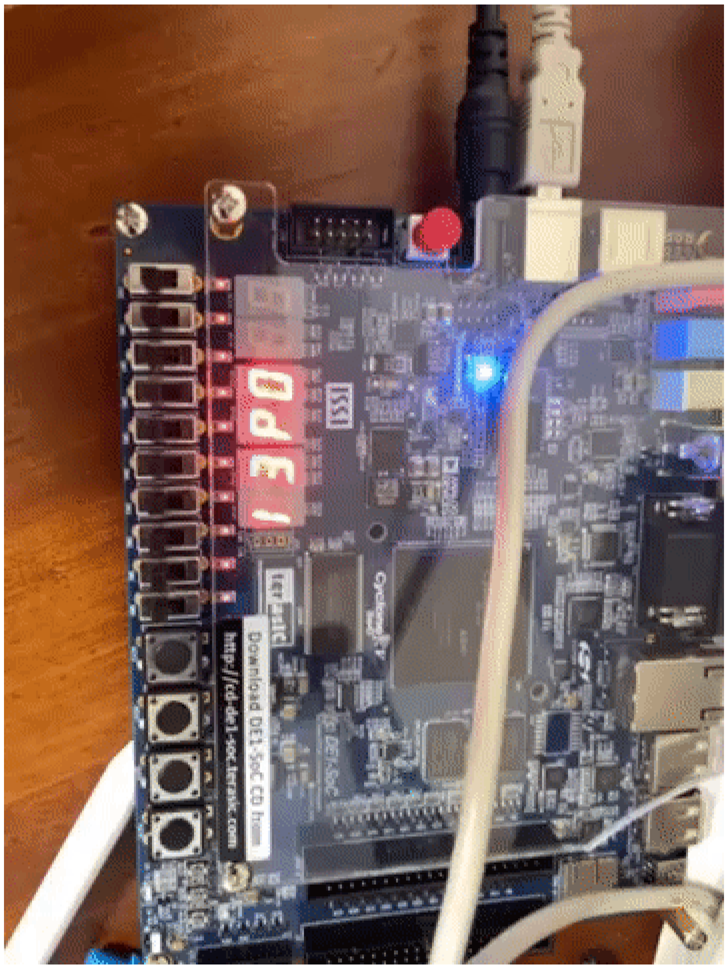
In this part, we utilize the internal clock CLOCK_50 (50 Mhz clock) to create a slower counter to increment a counter from 0 to 9 and display the counter in the hex display. A 50 Mhz clock increments a counter to 50 million. We use this logic to increment the display digit every 50 million counts and reset the counter for the next iteration.



Part 4: Rotate the word “dE10” on four hex displays

In this part, we took the implementation of lab 1 and modified it such that the word “dE10” rotates in the four hex displays. We achieved this by creating a counter that increments at every clock high. We used the internal CLOCK_50 to keep a counter and used the counter value to rotate the word at every second. The clock provided to us is 50 MHz, which means that there is 50 million cycles per second. So, a counter incremented at every cycle would make the counter 50 million in one second. We used this logic to implement the timer and rotate the word every second.

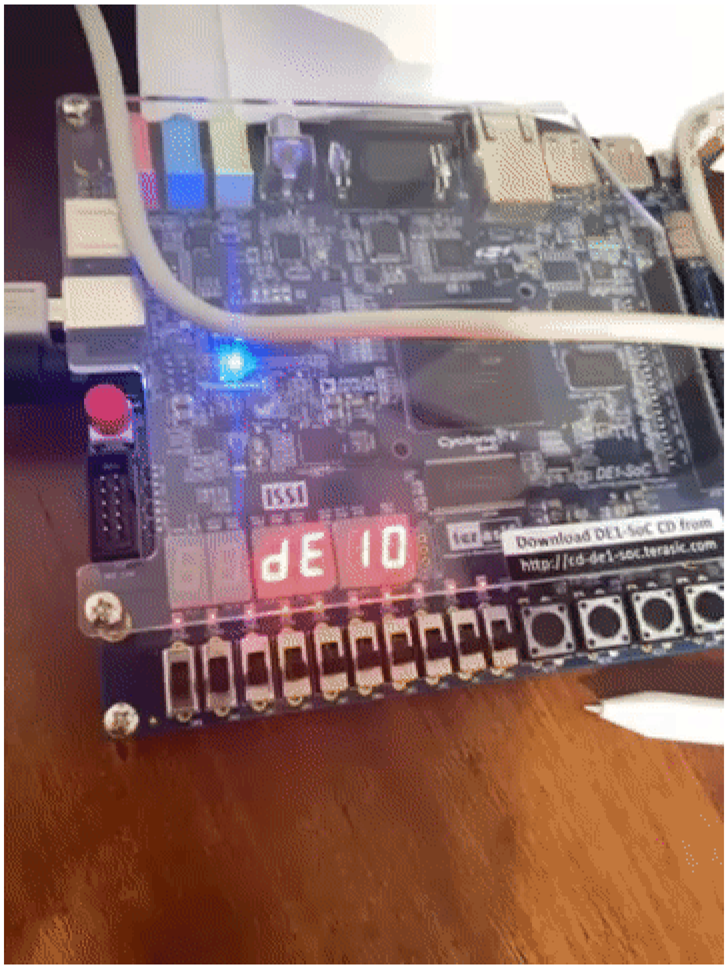
We were able to program and observe the output on the board as well. Added below is the .gif saved version of the program running on the board:



Part 5: Rotate word “dE10” on all six hex displays

In this part, we extended the circuit from part 4 and made the word “dE10” rotate on all six displays. The implementation for this was also very similar to that of part 4. We implemented a slower counter to keep track of the time and used that counter to change the display value on each HEX display after one second. We used the clock input similar to part 4.

We were able to compile and program the board and observe the letters rotate. Added below is a short image of the code running on the board in gif format:



Conclusion

In this lab, we learned how to implement counters in VHDL in various ways. We also learned how to use the clock input that is available in the SoC board to create slower clocks for various other purposes. In some parts of this lab, we made a button act as a clock input and in other parts we utilized the board's built in clock as the clock input for our circuits. However, we could not get part 1 to behave as expected on the board and could not figure out the reason. The remaining of this lab was implemented successfully and the circuits along with the board behaved as expected. Overall, this was a very helpful lab in understanding the utilization of clock.