CE100 Lab Report 4

16-bit Counter

Student Name: Armando Silva

Tutor: Dawn Hustig-Schultz

Lab Sec: TTH 2:00pm-4:00pm

Date: January 26th, 2016

Description:

The purpose of this lab was to get familiar with the implementation of positive edge-trigged filp-flops as well as using system clocks to sequential components. Using the logic learned in class, I was expected to build a 16-bit binary counter with Count Enable, Terminal Counts, an edge-detector, a ring counter to control the 7-segment displays, a selector to choose one out of four 4-bit buses, and a 70sement display module written in Verilog. Once we have all these components, I assembled a 16-bit counter that displays in hex four 7-segment digits that counts up each time Push Button 0 is pressed, and counts up continuously while holding Push Button 2 except in the range FFF8 to FFFF.

Part A- Counter Design

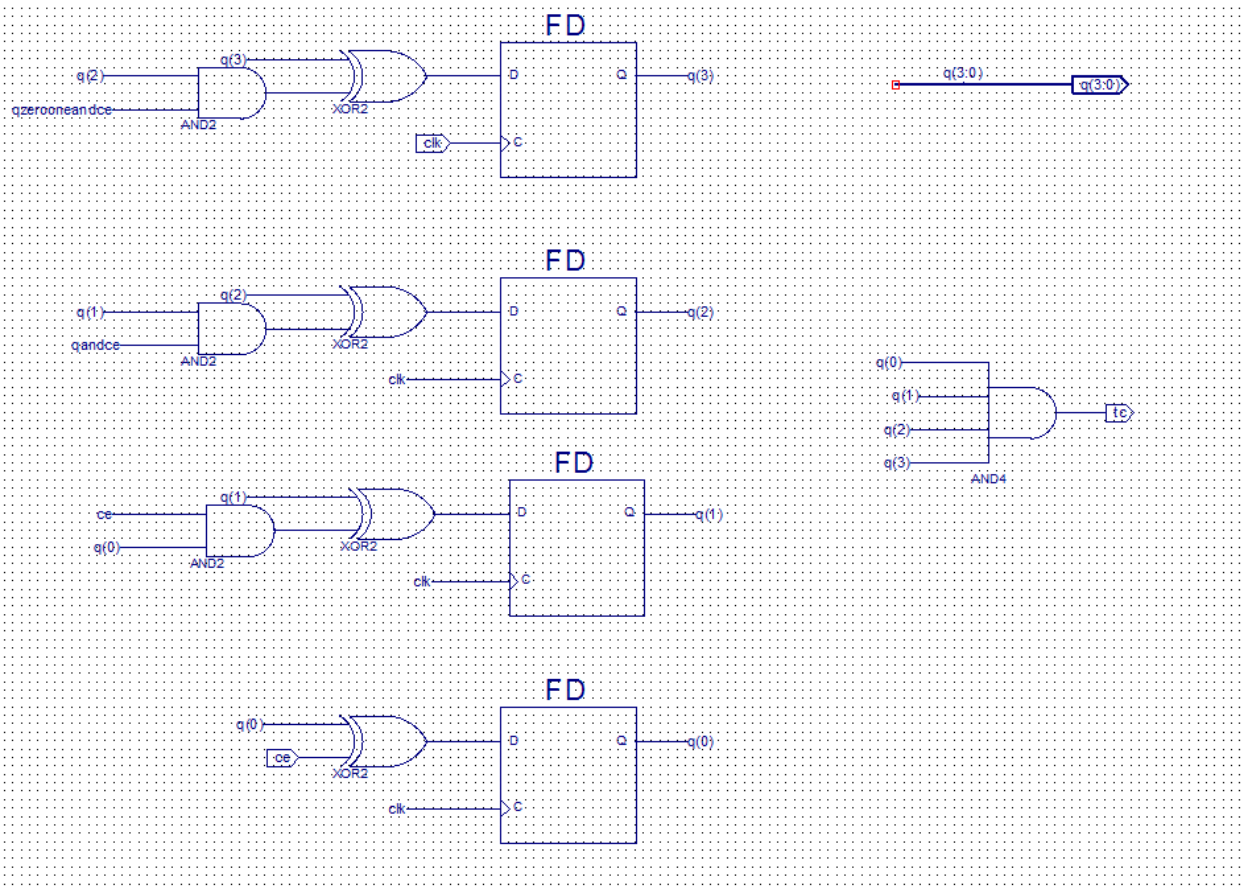


Figure 1: The Four Bit Counter

As we start designing the 16-bit counter, we must start by proceeding with a 4-bit counter. To do this we use logic gates and D flip flops. The logic behind these flip flops is that we are moving from one state to another where we xor the Qth state with an AND of CE and Qn -1 to Q0th states. We also had a TC so that we can input a high only when the counter bits are all 1's independently. The final product was the result of Figure 1, as seen above.

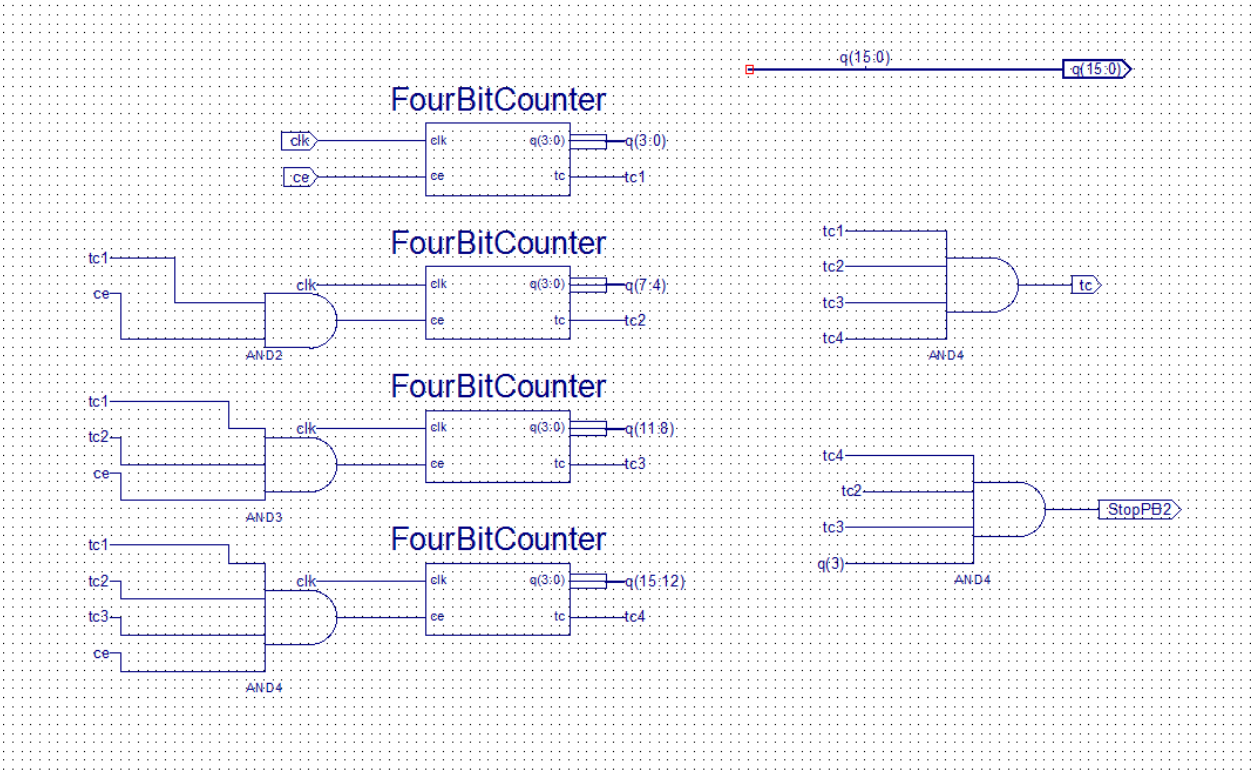


Figure 2: The 16-bit Counter

The ultimate goal of this symbol was to create a 16-bit counter, were we are assuming at this point that I have created a Four Bit Counter as seen in Figure 1. I then connect four of these 4-bit counters together to make a 16-bit counter. My 16-bit counter has three outputs: q(15:0), which is a 16-bit bus, 4 bits from each counter, a final terminal, the AND of all the TCs, and StopPB2, which is when my signal is at FFF8. From here, I have the Figure 3 as my product and create the 16-bit symbol to use for the top schematic.

Part B – 7-Segment Display

```
19  //
20  //////////////////////////////////////////////////////////////////////////////////
21  module hex7seg(
22      input n0,
23      input n1,
24      input n2,
25      input n3,
26      output CA,
27      output CB,
28      output CC,
29      output CD,
30      output CE,
31      output CF,
32      output CG
33      );
34
35
36
37     //sop for CA
38
39     assign CA = ((~n3&~n2&~n1&n0) | (~n3&n2&~n1&~n0) | (n3&~n2&n1&n0) | (n3&n2&~n1&n0));
40
41     //sop for CB
42
43     assign CB = ((~n3&n2&~n1&n0) | (~n3&n2&n1&~n0) | (n3&~n2&n1&n0) | (n3&n2&~n1&~n0) | (n3&n2&n1&~n0) | (n3&n2&n1&n0));
44
45     //sop for CC
46
47     assign CC = ((~n3&~n2&n1&~n0) | (n3&n2&~n1&~n0) | (n3&n2&n1&~n0) | (n3&n2&n1&n0));
48
49     //sop for CD
50
51     assign CD = ((~n3&~n2&~n1&n0) | (~n3&n2&~n1&~n0) | (~n3&n2&n1&n0) | (n3&~n2&n1&~n0) | (n3&n2&n1&n0));
52
53     //sop for CE
54
55     assign CE = ((~n3&~n2&~n1&n0) | (~n3&~n2&n1&n0) | (~n3&n2&~n1&~n0) | (~n3&n2&~n1&n0) | (~n3&n2&n1&n0) | (n3&~n2&~n1&n0));
56
57     //sop for CF
58
59     assign CF = ((~n3&~n2&~n1&n0) | (~n3&~n2&n1&~n0) | (~n3&~n2&n1&n0) | (~n3&n2&n1&n0) | (n3&n2&~n1&n0));
60
61     //sop for CG
62
63     assign CG = ((~n3&~n2&~n1&~n0) | (~n3&~n2&~n1&n0) | (~n3&n2&n1&n0) | (n3&n2&~n1&~n0));
64
65  endmodule
```

Figure 3: Verilog of 7-Segment Display

For this part, we were to assemble a 7-Segment display that takes in a 4-bit vector to produce the same 7 segment LEDS as the previous labs, except that it was to be done in Verilog. For this, I used the sum of product I wrote out for lab 2 and wrote it in Verilog as seen in Figure 3. Then after creating it, I saved the file and created the symbol to use in my top schematic.

Part C- Selector

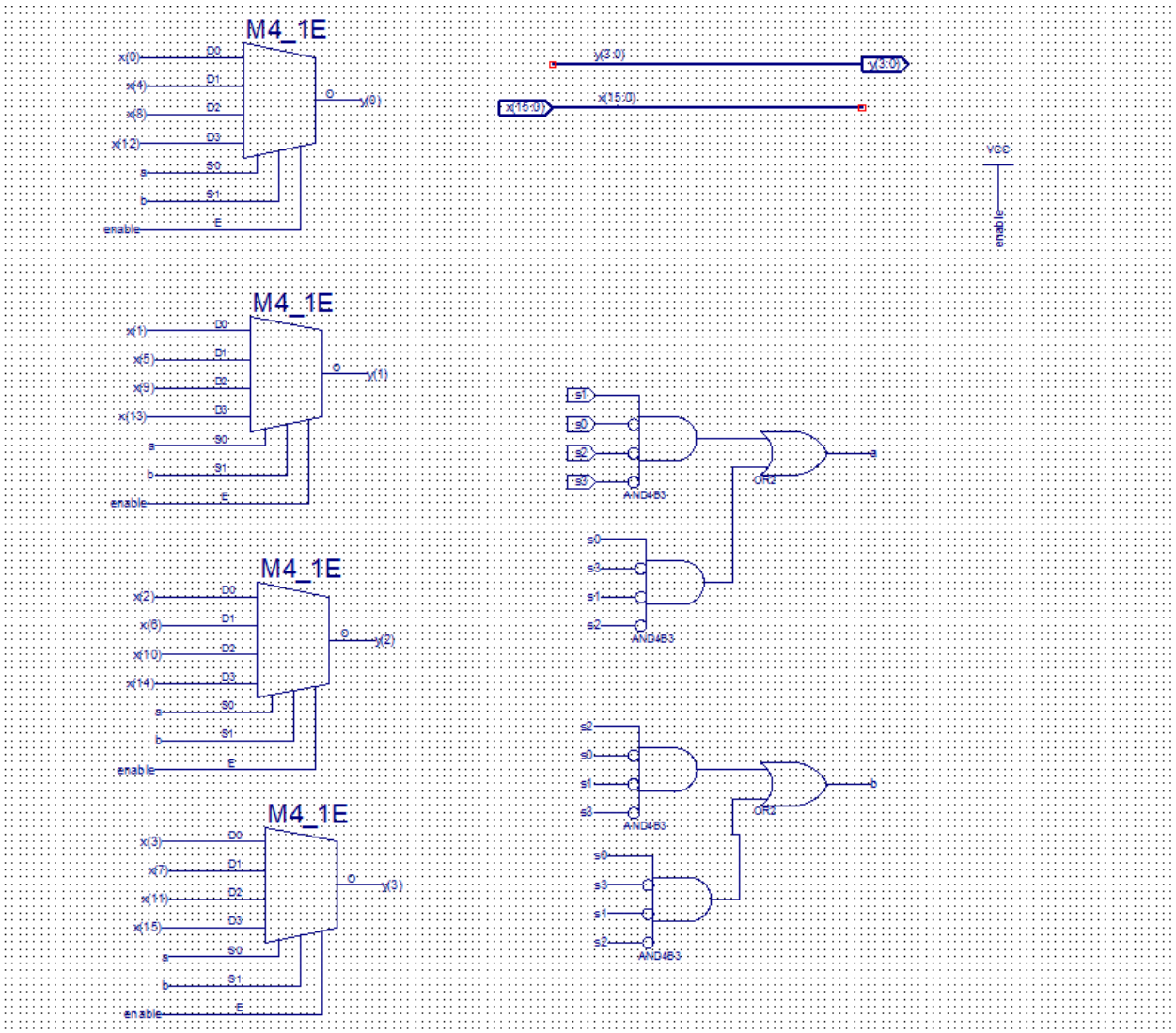


Figure 4: The Selector Schematic

In this part we were to assemble the selector. Its functionality was to not use a hex 7-symbol for each digit, but rather be economical and only use one of these symbols by feeding it the 5-bit values at one time. Then, the selector controls the inputs, with a 16-bit bus, and output the correct 4-bit bus. Figure 4 was derived from Figure 5, the table has its inputs as S0 through S3 and A, B as selectors. I used sum of product for each selector and ultimately took the four-bit output into one output bus. The final sum of product for Figure 5 was: A= (~S0S1~S2~S3) +(S0~S1~S2S3) and B = (~S0~S1S2~S3) +(S0~S1~S2~S3)

S0	S1	S2	S3	A	B
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Figure 5: Table for Multiplier of Selector

Part D: The Ring Counter

The next step in this lab was to create a 4-bit Ring Counter to allow the FPGA board to display two LEDs at once (however in reality, the switching between two LEDs is so fast that it seems that the LEDs are on at once, but only one is on). To design this we use only D Flip-Flops and invert the first input to allow for an initial start. Here my product can be seen in Figure 6.

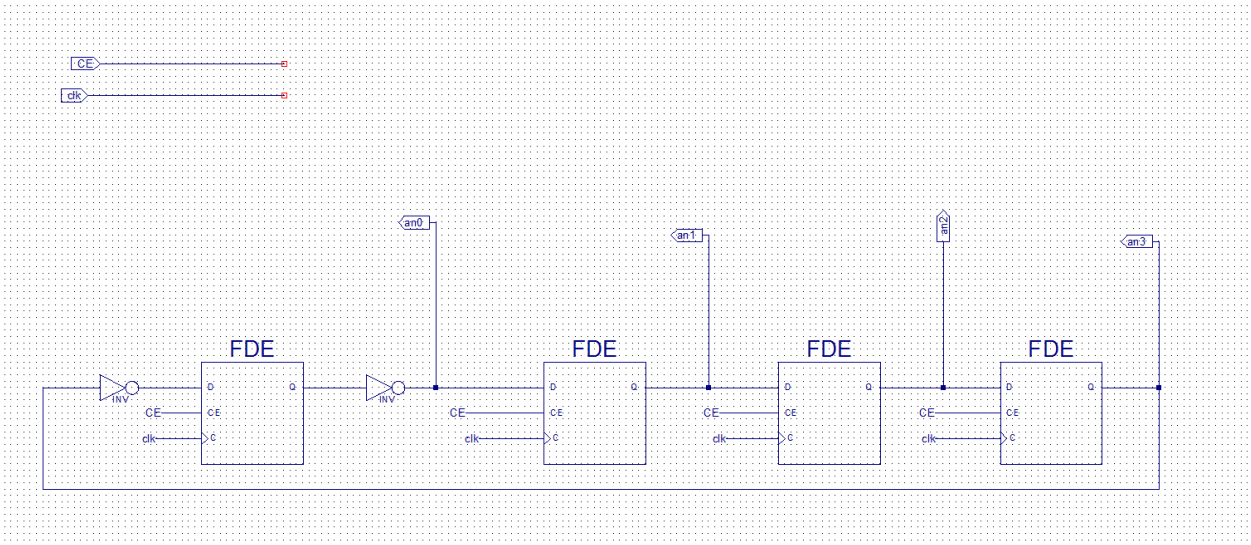


Figure 6: The Ring Counter

Part E: Edge Detector

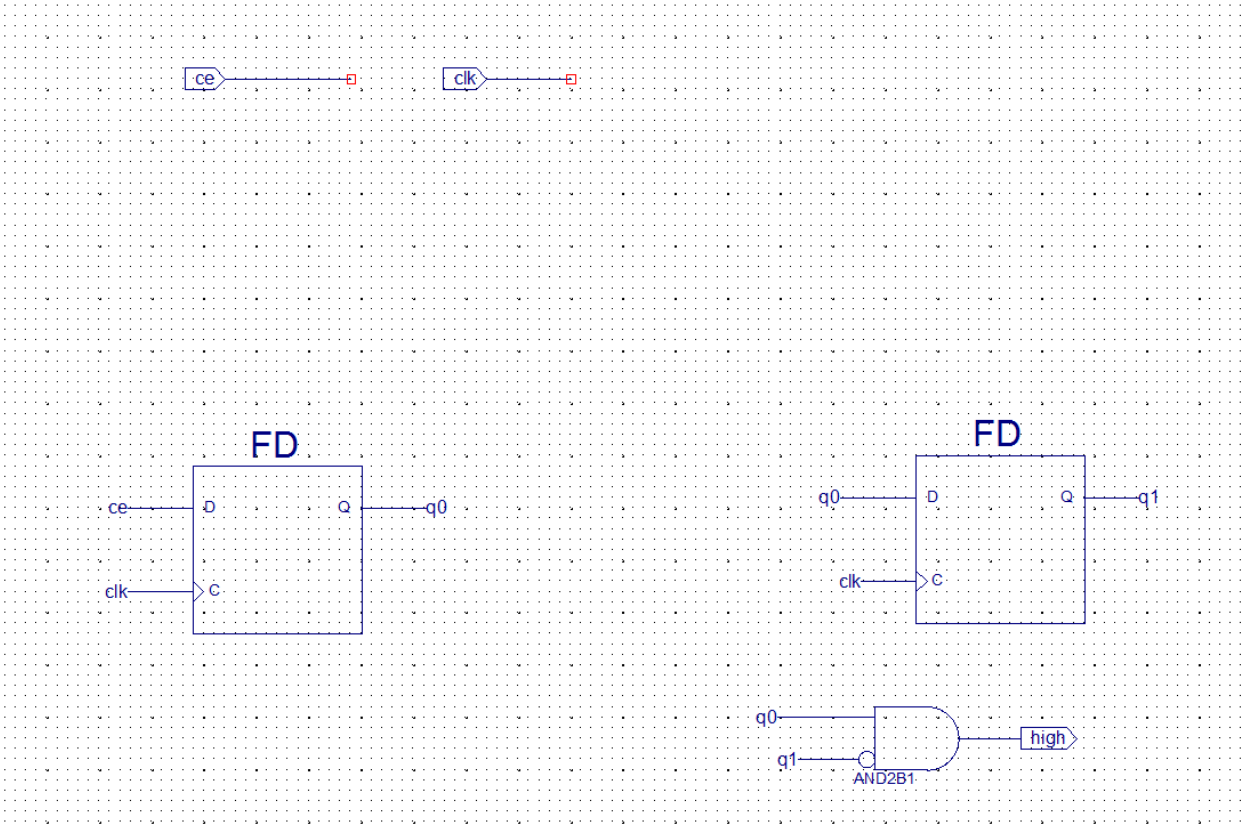


Figure 7: The Edge Detector

For this part, I designed an Edge Detector that generates a high value for one clock cycle if the past two inputs consist of a 0 followed by a 1. We can observe this from the logic gate ~Q0 & Q1.

Part abcdefg

The final part, the moment I waited for hours into the lab, the Top Level Schematic. After assuming to finish the parts before this in the lab report, we enter this top level schematic with all the pieces to assemble our 16 bit counter. I add a eclk given by Martine Schlag, and connect the symbols together as seen in Figure 8.

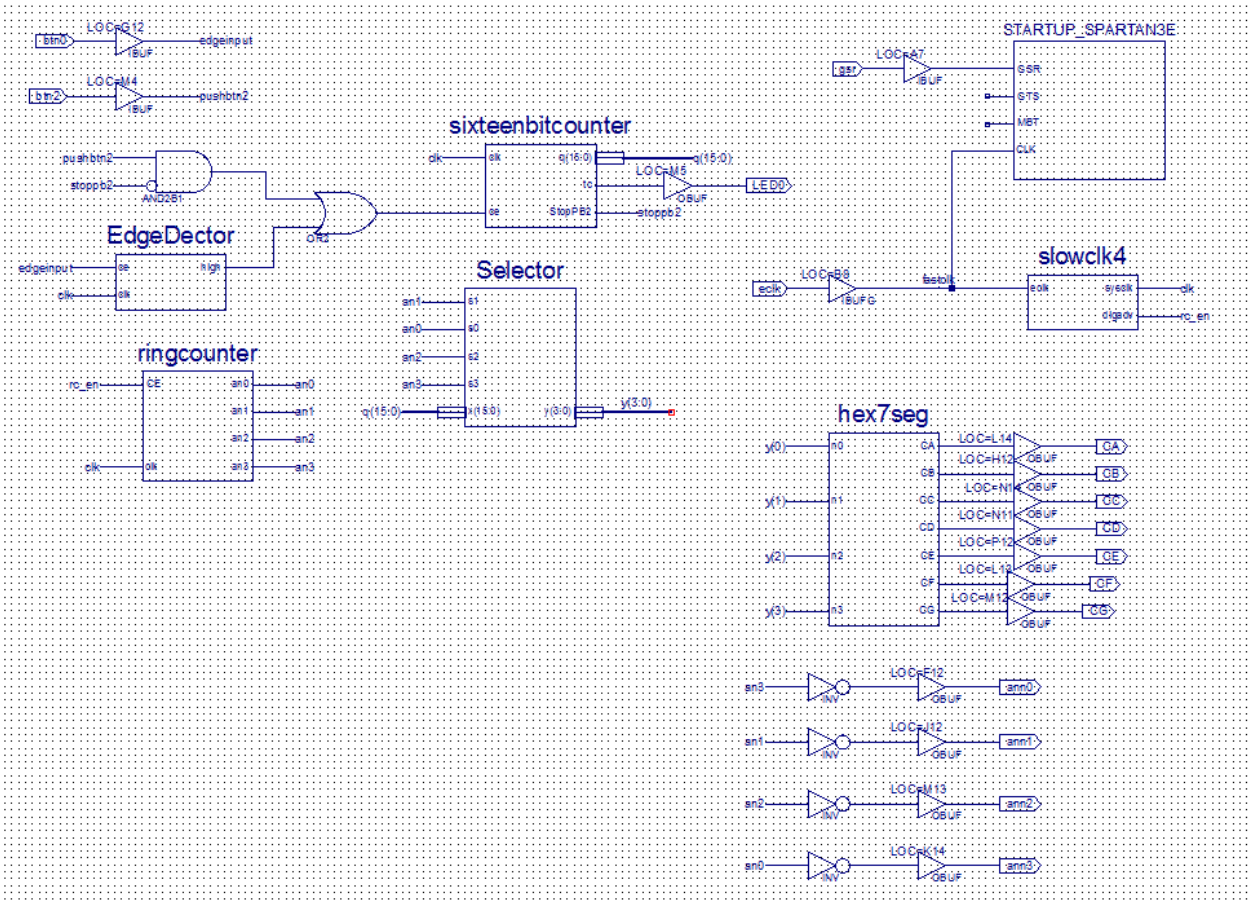


Figure 8: The Top Level Schematic

After finishing my design, configuring the FPGA and demonstrating to my TA, I followed the last two experiments of the lab. The questions were as followed: What happens when hold down Push Button 3? When holding down Push Button 3, one of my anodes showed me a 0. This may be because my gsr which is connected to my clk is always being inputted and reset. The second part was connecting the clock inputs to signal coming out of the ibufg rather than clk, by doing this it makes the display go to FFF8 when PB2 is pressed. This is because the clock has more frequency and speed of the counter.

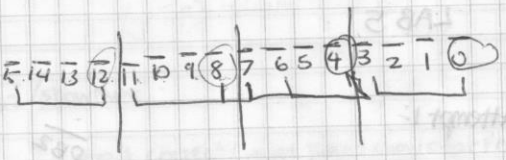
Conclusion:

Ultimately this lab showed me how to comprehend clocks and selectors. With this I was able to process the logic behind buses and counters, along with clock cycles and edge detectors.

LAB 4

Q_3	Q_2	Q_1	Q_0	CE=0	CE=1
0	0	0	0	0000	0001
0	0	0	1	0001	0010
0	0	1	0	0010	0011
0	0	1	1	0011	0100
0	1	0	0	0100	0101
0	1	0	1	0101	0110
0	1	1	0	0110	0111
0	1	1	1	0111	1000
1	0	0	0	1000	1001
1	0	0	1	1001	1010
1	0	1	0	1010	1011
1	0	1	1	1011	1100
1	1	0	0	1100	1101
1	1	0	1	1101	1110
1	1	1	0	1110	1111
1	1	1	1	1111	0000

S_0	S_1	S_2	S_3	A	B
0	0	0	1	0	0
0	0	1	0	0	1
0	0	0	0	1	0
1	0	0	0	1	1



$A = (\bar{S}_0 S_1 \bar{S}_2 \bar{S}_3) + (S_0 \bar{S}_1 \bar{S}_2 S_3)$

$B = (\bar{S}_0 \bar{S}_1 S_2 \bar{S}_3) + (S_0 \bar{S}_1 \bar{S}_2 \bar{S}_3)$

Signal		PIN
PushBtn	0	= G12
	2	= M4
CA		= L14
CB		= H12
CC		= N14
CD		= N11
CE		= P12
CF		= L13
CG		= M12
LD0		= M5
An0		= F12
An1		= J12
An2		= M13
An3		= K14

LAB 4

Rebecca

17:47

2 FEB 2016

2618