ECEN 429: Introduction to Digital Systems Design Laboratory

North Carolina Agricultural and Technical State University

Department of Electrical and Computer Engineering

Freddie Boadu (Reporter)

Mike Umelo (Partner)

April 2, 2020

Pre Lab #8

**Introduction**

This prelab deals with dealing with registers. We are learning for lab 8 how to use the registers in the lab for this week.

**Background, Design Solution and Results**

This prelab provides us some insight into what our registers that we are using will do and look like when they are synced with a clock.

Problem 1:

The code is enabled with a rising edge clock signal. The size of the register is 3 bit. The size of the register would change by adding more bits to the signal nxt. The contents of the register would have to have another function in there to clear the bits. This would have to be something along the lines of if reset is 1 then make the contents of the register go to one.

Problem 2:

The counter is implemented using the clock signal and reset within the code. It starts with making a signal of nxt. A different value can be done by changing the generic integer on the counter.

Problem 3

The inputs of the clk means just the clock of the code, the reset means resetting the shift register contents, and the sin is for shifting left in this case. The output of buffer is to output the contents based on the number of integers stated wanting to be outputted minus one. The shift is showed by moving the output (n-2 downto 0) and anding it with the amounted needed to shift left which is showed by sin. To make it count in the other direction, you would do (n+2 downto 0) for the output to yield a different result. The enable signal would be put on 1.

Problem 4:

The fsm would need to generate signals to move to the next state to shift right and signals that go to the current state or previous. The amount of state would be based on the amount of shifts wanted to be completed.

Problem 5:

Entity reg is

Port(

D: in std\_logic\_vector(1 downto 0);

clk, reset, load: std\_logic;

Q: out std\_logic\_vector(1 downto 0));

End;

architecture Register\_with\_sync\_load\_behavior of Register\_with\_sync\_load is

begin

process (clk)

begin

if rising\_edge(clk) then

if (load = ‘1’) then

Q <= D;

end if;

end if;

end process;

end Register\_with\_sync\_load\_behavior;

**Conclusion**

After doing the prelab, I understand the relationship between the fsm and shift registers operations.