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)

March 26, 2020

Pre Lab #7

**Introduction**

This prelab deals with ALU. ALU is short for arithmetic logic unit. These are used to do operations such as Addition, subtract, and OR. And much more. Last week, we learned about state machines. Another name for these state machines are FSM or finite state machines. For the prelab we are going to learn how the FSM and ALU would interact. The state machine is considered the brains of the operation while the ALU is the body. The state machine directs what operations the ALU will complete.

**Background, Design Solution and Results**

This prelab provides us some insight into what our ALU should be able to complete. The ALU should be able to do at least 4 of the functions below:

XOR

NAND

Adds 4 to the input

Subtract 3 from the input

Shift left by 2

Shift right by 3

We are to build a 4-bit ALU that will be able to implement at least 4 of these functions presented.

Problem 1:

The problem is asking what would it take to selection certain functions within your ALU that you want to perform. For example, you want to add 2 4-bit numbers together within your ALU to yield results. Each operation would be dictated by a state. For example, if I were to pick XOR, NAND, ADD, and SUB. My state machine would be around 4 states long. XOR would be state 1, NAND would be state 2, ADD would be state 3, and SUB would be state 4. In order to complete the ADD function desired I would need to progress 3 states in my machine to state 3 to execute this. This can be done by setting up my state machine to automatically go state by state or I could add feedback which would make the state machine follow a path of execution rather than it being completely linear.

Problem 2:

Choosing the method on the ALU should be on a state by state basis. What I mean by that is depending on the state an operation should be called. We can control this by assigning each state a number based on the amount of states there are present. If there are 4 states, the first function used should be declared by 00 then 01 for the second and etc. until 11 is reached to represent that last state needed to represent the functions/instructions. For my ALU, the value of 00 will be my XOR, 01 will be my NAND, ADD will be my 10, and 11 will be my SUB operations. This will allow for selections of each of the states in the ALU. As well in the actual ALU there is always a selection bit that helps with this process.

Problem 3

Entity moorealu is

Port( reset, clk, input: in std\_logic;

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

End;

Architecture beh of moorealu is

Signal cs,ns: std\_logic\_vector( 1 downto 0);

Begin

Process(clk.reset)

Begin

If (reset=’1’) then cs<=”00”;

elsif(clk' event and clk ='1') then

cs<=ns;

end if;

end process;

--this process will loop through all the states

process ( cs)

begin

case cs is

when ("00") =>-- state 0

if (input=’1’)

output <= "00";

ns<= ("01");

else

output<=”00”;

ns<=(“00”);

when "01" =>-- state 1

if (input=’1’)

output <= "01";

ns<= "10";

else

output<=”01”;

ns<=(“01”);

when "10" =>--state 2

if (input=’1’)

output <= "10";

ns<= "11";

else

output<=”10”;

ns<=(“10”);

when "11" =>-- state 3

if (input=’1’)

output <= "11";

ns<= "11";

else

output<=”11”;

ns<=(“00”);

when others =>

Null;

end case;

end process;

end;

inputs

ALU  
with XOR,NAND, AND, SUB

FSM

output

**Conclusion**

After doing the prelab, I understand the relationship between an ALU and the fsm. The Fsm is a brains behind the alu which allows it to moves from state to state and preform key operations.