Grace Pope

CPE 166 – 03 Monday 2-4:40

Advanced Logic Design Lab

Lab 4

**Contents**

[Introduction 3](#_gjdgxs)

[Part 1 – Simplified Microprocessor Design 3](#_30j0zll)

[Design Purpose 3](#_1fob9te)

[Engineering Data 3](#_3znysh7)

[Source Code 5](#_tyjcwt)

[User Constraint File 19](#_3dy6vkm)

[Simulation Waveforms 19](#_1t3h5sf)

[Results Discussion 20](#_4d34og8)

[Conclusion 20](#_2s8eyo1)

# Introduction

This lab was to design a microprocessor that was a simplified version of a microprocessor that could be used in other devices. It would perform the function R[2] = M0 + (not M1) + Cin by using a finite state machine to control the logic necessary to perform the arithmetic function as specified in the lab manual.

# Part 1 – Simplified Microprocessor Design

## Design Purpose

This lab was intended to show us how to design a microprocessor. It used a finite state machine to control a datapath circuit that consisted of multiplexors, d flip flops, and an arithmetic logic unit. This microprocessor was representative of more complex microprocessors used in other devices.

## Engineering Data

There are two modules within my top level microprocessor design, the fsm and the datapath module. These modules are connected by the top level module. The block diagram for the top level module is in figure 1. The datapath module includes three 2-to-1 multiplexors, four d flip flops with asynchronous clear and clock enable, one 4-to-1 multiplexor, and one arithmetic logic unit. The design for the datapath module is in figure 2.

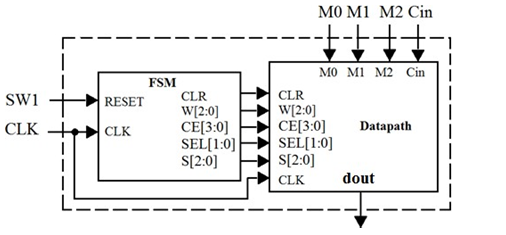


Figure 1: Top level module diagram

The main module in the datapath cirucit was the arithmatic logic unit. It took three inputs, S, to determine what function it should do. S was a three bit number, so the alu would be able to do eight different functions on the other inputs, A, B, and Cin. The truth table for the alu is in table 1. The most difficult part of designing the alu was figuring out how to do the addition. I kept getting an error because two 1 bit numbers can have a carry bit which would add up to a 2 bit number. This was a problem for me because VHDL wants the size of the numbers being added to be the same size of the result. To solve this problem, I needed to concatenate A, B, and Cin with a 0 to make them a 2 bit number. Since the alu only needed to output a 1 bit number, the output for the alu would only take the last bit of the added number as the output. All of the other operations for the alu were 1 bit numbers, so I had no difficulty with them when creating the alu.

|  |  |  |  |
| --- | --- | --- | --- |
| **S2** | **S1** | **S0** | **F** |
| 0 | 0 | 0 | F=A+B+Cin |
| 0 | 0 | 1 | F=A+B’+Cin |
| 0 | 1 | 0 | F= B |
| 0 | 1 | 1 | F= A NAND B |
| 1 | 0 | 0 | F= A AND B |
| 1 | 0 | 1 | F= A OR B |
| 1 | 1 | 0 | F= A’ |
| 1 | 1 | 1 | F = A XOR B |

Table 1: ALU truth table

The other modules within the datapath circuit were d flip flops and multiplexors. For the design of those, I just copied the modules from previous labs. The only modification needed was to include a clock enable signal in the d flip flops since most of the other labs also used flip flops with asynchronous reset.

The next module to be connected was the complete datapath circuit. This was its own sub-top level module. It connected all of the d flip flops, multiplexors, and the alu together. The reference I used to connect the datapath module is in figure 2.

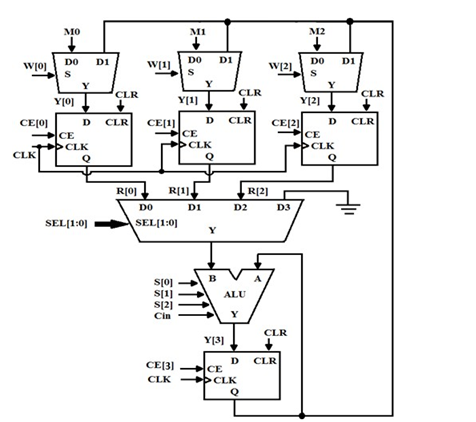


Figure 2: Datapath circuit diagram

Once the datapath module was completed, I had a good idea of what my state machine needed to do to control the circuit. My state machine used five states, including a reset state. The Reset state would clear all of the flip flops and make everything zero. If the reset switch was high, it would go right into the second state, S1, where it would enable the last multiplexor. The signal to the multiplexor was W, and it was always 000 because the data from the multiplexor should always be the D0 input. The CE input would need to be 0001 because we only wanted the last flip flop to be enabled to be chosen from the first 4-to-1 multiplexor. The third state, S2, was to enable the 4-to-1 multiplexor to allow the data to pass through to the alu, where it would choose to output F= B. The fourth state, S3, would enable the second d flip flop to take the data from the second 2-to-1 multiplexor and send it to the 4-to-1 multiplexor. The last state would enable the final d flip flop to allow the Qout to be the answer to the design problem, R[2] = M0 + (not M1) + Cin. The datapath module would output Qout to the top level module, where it would be displayed on the fpga. The state diagram I used is in figure 3.



Figure 3: State diagram for lab 4

## Source Code

|  |
| --- |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : top.vhd  -- Purpose of Code : top level module  -- Project Part Number : Lab 4.A  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  library IEEE;  use IEEE.STD\_LOGIC\_1164.ALL;  entity top is  Port ( clk, SW1, cin: in std\_logic;  M: in std\_logic\_vector(2 downto 0);  Led: out std\_logic  );  end top;  architecture Behavioral of top is  component fsm is  port(clk,sw1: IN STD\_LOGIC;  clr: out std\_logic;  W,S: out std\_logic\_vector(2 downto 0);  CE: out std\_logic\_vector(3 downto 0);  sel: out std\_logic\_vector(1 downto 0)  );  end component;  component datapath is  Port ( M, W, S: in std\_logic\_vector(2 downto 0);  clk, clr, Cin: in std\_logic;  sel: in std\_logic\_vector(1 downto 0);  CE: in std\_logic\_vector(3 downto 0);  dout: out std\_logic  );  end component;  signal clr: std\_logic;  signal W,S: std\_logic\_vector(2 downto 0);  signal CE: std\_logic\_vector(3 downto 0);  signal sel: std\_logic\_vector(1 downto 0);  signal dout, gnd: std\_logic;  begin  gnd <='0';  uut: fsm port map(clk=>clk, sw1=>sw1, clr=>clr, W=>W, S=>S, CE=>CE, sel=>sel);  uut2: datapath port map(M=>M, W=>W, S=>S, clk=>clk, clr=>clr, Cin=>Cin, sel=>sel, CE=>CE, dout=>dout);  Led <= dout  end Behavioral; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : fsm.vhd  -- Purpose of Code : finite state machine for control of datapath  -- Project Part Number : Lab 4.B  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  library IEEE;  use IEEE.STD\_LOGIC\_1164.ALL;  entity fsm is  port(clk,sw1: IN STD\_LOGIC;  clr: out std\_logic;  W,S: out std\_logic\_vector(2 downto 0);  CE: out std\_logic\_vector(3 downto 0);  sel: out std\_logic\_vector(1 downto 0)  );  end fsm;  architecture Behavioral of fsm is  type state\_type is (s0,s1,s2,s3,s4);  signal cs, ns: state\_type;  begin  Process(clk, sw1)  Begin  if (sw1 = '1') then  cs <= s0;  elsif (rising\_edge(clk)) then  if (sw1='0') then  cs<=ns;  end if;  end if;  End process;    process(cs, sw1)  begin  case (cs) is  when s0 =>  if sw1='1' then  ns <= s0;  else  ns <= s1;  end if;  when s1 =>  if sw1='1' then  ns <= s0;  else  ns <= s2;  end if;  when s2 =>  if sw1='1' then  ns <= s0;  else  ns <= s3;  end if;  when s3 =>  if sw1='1' then  ns <= s0;  else  ns <= s4;  end if;  when s4 =>  if sw1='1' then  ns <= s0;  else  ns <= s4;  end if;  when others => ns <=s0;  end case;  end process;    Process(cs) --outputs  Begin  Case (cs) is  When S0 =>  CE <= "0000";  sel <= "11";  clr <= '1';  W <= "000";  S <= "000";  When S1 =>  CE <= "0001";  sel <= "00";  clr <= '0';  W <= "000";  S <= "010";  When S2 =>  CE <= "1000";  sel <= "00";  clr <= '0';  W <= "000";  S <= "010";  When S3 =>  CE <= "0010";  sel <= "10";  clr <= '0';  W <= "010";  S <= "001";  When S4 =>  CE <= "1000";  sel <= "10";  clr <= '0';  W <= "000";  S <= "001";  When others=>  CE <= "0000";  sel <= "11";  clr <= '1';  W <= "000";  S <= "000";  end case;  end process;  end Behavioral; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : datapath.vhd  -- Purpose of Code : Top level module for datapath circuit  -- Project Part Number : Lab 4.C  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  library IEEE;  use IEEE.STD\_LOGIC\_1164.ALL;  entity datapath is  Port ( M, W, S: in std\_logic\_vector(2 downto 0);  clk, clr, Cin: in std\_logic;  sel: in std\_logic\_vector(1 downto 0);  CE: in std\_logic\_vector(3 downto 0);  dout: out std\_logic  );  end datapath;  architecture Behavioral of datapath is  component mux is  PORT (A,B : IN STD\_LOGIC;  SEL: IN STD\_LOGIC;  Y : OUT STD\_LOGIC  );  end component;  component dff is  port( clk,clr,CE: in std\_logic;  regDin: in std\_logic;  regQ: out std\_logic  );  end component;  component mux4 is  PORT ( A,B,C,D : IN STD\_LOGIC;  SEL : IN STD\_LOGIC\_VECTOR(1 downto 0);  Y : OUT STD\_LOGIC  );  end component;  component alu is  port ( A, B, Cin : in std\_logic;  s : in std\_logic\_vector(2 downto 0);  F : out std\_logic  );  end component;  signal R: std\_logic\_vector(2 downto 0);  signal Ytmp: std\_logic\_vector(3 downto 0);  signal muxy, Qout: std\_logic;  signal gnd: std\_logic;  begin  gnd<='0';  uut: mux port map(A=>M(0), B=>Qout, sel=>W(0), Y=>Ytmp(0));  uut2: mux port map(A=>M(1), B=>Qout, sel=>W(1), Y=>Ytmp(1));  uut3: mux port map(A=>M(2), B=>Qout, sel=>W(2), Y=>Ytmp(2));  uut4: dff port map(clk=>clk, clr=>clr, CE=>CE(0), regDin=>ytmp(0), regQ=>R(0));  uut5: dff port map(clk=>clk, clr=>clr, CE=>CE(1), regDin=>ytmp(1), regQ=>R(1));  uut6: dff port map(clk=>clk, clr=>clr, CE=>CE(2), regDin=>ytmp(2), regQ=>R(2));  uut7: mux4 port map(A=>R(0), B=>R(1), C=>R(2), D=>gnd, SEL=>sel, Y=>muxy);  uut8: alu port map(A=>Qout, B=>muxy, Cin=>Cin, s=>S, F=>ytmp(3));  uut9: dff port map(clk=>clk, clr=>clr, CE=>CE(3), regDin=>ytmp(3), regQ=>Qout);  dout <= Qout;  end Behavioral; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : mux.vhd  -- Purpose of Code : 2-to-1 multiplexor  -- Project Part Number : Lab 4.D  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  LIBRARY IEEE;  USE IEEE.STD\_LOGIC\_1164.ALL;  ENTITY mux IS  PORT (A,B : IN STD\_LOGIC;  SEL: IN STD\_LOGIC;  Y : OUT STD\_LOGIC  );  END mux;  ARCHITECTURE design OF mux IS  BEGIN  Y <= a WHEN SEL = '0' ELSE  b ;  END design; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : dff.vhd  -- Purpose of Code : D flip flop with clr and clock enable  -- Project Part Number : Lab 4.E  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  library ieee;  use ieee.std\_logic\_1164.all;  entity dff is  port( clk,clr,CE: in std\_logic;  regDin: in std\_logic;  regQ: out std\_logic  );  end dff;  architecture beh of dff is  begin  process(clk)  begin  if(rising\_edge(clk)) then  if(clr = '1') then  regQ <= '0';  elsif(CE = '1') then  regQ <= regDin;  end if;  end if;  end process;  end beh; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : mux4.vhd  -- Purpose of Code : 4-to-1 Multiplexor  -- Project Part Number : Lab 4.F  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  LIBRARY IEEE;  USE IEEE.STD\_LOGIC\_1164.ALL;  ENTITY mux4 IS  PORT ( A,B,C,D : IN STD\_LOGIC;  SEL : IN STD\_LOGIC\_VECTOR(1 downto 0);  Y : OUT STD\_LOGIC  );  END mux4;  ARCHITECTURE design OF mux4 IS  BEGIN  Y <= A WHEN SEL = "00" ELSE  B WHEN SEL = "01" ELSE  C WHEN SEL = "10" ELSE  D;  END design; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : alu.vhd  -- Purpose of Code : Arithmetic Logic Unit  -- Project Part Number : Lab 4.G  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  library IEEE;  use IEEE.STD\_LOGIC\_1164.ALL;  use ieee.std\_logic\_unsigned.all;  use IEEE.STD\_LOGIC\_ARITH.ALL;  entity alu is  port ( A, B, Cin : in std\_logic;  s : in std\_logic\_vector(2 downto 0);  F : out std\_logic  );  end alu;  architecture beh of alu is  signal fadd,fadd2: std\_logic\_vector(1 downto 0);  signal atmp, btmp, ctmp: std\_logic\_vector(1 downto 0);  begin  atmp<='0' & (a);  btmp<='0' & (b);  ctmp<='0' & cin;  fadd <= atmp + btmp +Ctmp;  fadd2 <= atmp +(not Btmp) + Ctmp;  process(s)  begin  if (s="000") then  F <= fadd(0);  elsif (s="001") then  F <= fadd2(0);  elsif (s="010") then  F <= B;  elsif s="011" then  F <= (A nand B);  elsif s="100" then  F <= (A and B);  elsif s="101" then  F <= (A or B);  elsif s="110" then  F <= (not A);  elsif s="111" then  F <= (A xor B);  end if;  end process;  end beh; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : top\_tb.vhd  -- Purpose of Code : test bench for top level module  -- Project Part Number : Lab 4.H  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  LIBRARY IEEE;  USE IEEE.STD\_LOGIC\_1164.ALL;  ENTITY top\_tb IS  END top\_tb;  ARCHITECTURE beh OF top\_tb IS  component top  Port ( clk, SW1, cin: in std\_logic;  M: in std\_logic\_vector(2 downto 0);  Led: out std\_logic  );  end component;  signal clk, SW1, cin: std\_logic;  signal M: std\_logic\_vector(2 downto 0);  signal Led: std\_logic;  BEGIN  Uut: top port map ( clk => clk, sw1 => sw1, cin => cin, M=>M, led=>led);  process  begin  clk <= '0'; wait for 10 ns;  clk <= '1'; wait for 10 ns;  end process;  process  begin  sw1 <= '1'; cin <='0'; M <="101";  wait for 40 ns;  sw1<='0';  wait for 160 ns;  sw1 <= '1'; cin <='1'; M <="101";  wait for 40 ns;  sw1<='0';  wait for 160 ns;  sw1 <= '1'; cin <='1'; M <="011";  wait for 40 ns;  sw1<='0';  wait for 160 ns;  sw1 <= '1'; cin <='0'; M <="011";  wait for 40 ns;  sw1<='0';  wait for 160 ns;  sw1 <= '1'; cin <='1'; M <="000";  wait for 40 ns;  sw1<='0';  wait for 160 ns;  sw1 <= '1'; cin <='0'; M <="000";  wait for 40 ns;  sw1<='0';  wait for 160 ns;  sw1 <= '1'; cin <='0'; M <="110";  wait for 40 ns;  sw1<='0';  wait for 160 ns;  sw1 <= '1'; cin <='1'; M <="110";  wait for 40 ns;  sw1<='0';  wait for 160 ns;  wait;  end process;  END beh; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : datapath\_tb.vhd  -- Purpose of Code : test bench for datapath circuit  -- Project Part Number : Lab 4.I  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  library IEEE;  use IEEE.STD\_LOGIC\_1164.ALL;  entity datapath\_tb is  end datapath\_tb;  architecture Behavioral of datapath\_tb is  component datapath is  Port ( M, W, S: in std\_logic\_vector(2 downto 0);  clk, clr, Cin: in std\_logic;  sel: in std\_logic\_vector(1 downto 0);  CE: in std\_logic\_vector(3 downto 0);  dout: out std\_logic  );  end component;  signal m,w,s: std\_logic\_vector(2 downto 0);  signal clk, clr, Cin: std\_logic;  signal sel: std\_logic\_vector(1 downto 0);  signal CE: std\_logic\_vector(3 downto 0);  signal dout: std\_logic;  begin  uut: datapath port map(m=>m, w=>w, s=>s, clk=>clk, clr=>clr, cin=>cin, sel=>sel, CE=>ce, dout=>dout);  process  begin  clk <= '0'; wait for 10 ns;  clk <= '1'; wait for 10 ns;  end process;  process  begin  CE <= "0000"; sel <= "11"; clr <= '1'; w <= "000"; S <= "000"; cin <= '0'; M <= "101";  wait for 40 ns;  CE <= "0001"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "1000"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "0010"; sel <= "10"; clr <= '0'; w <= "010"; S <= "001";  wait for 40 ns;  CE <= "1000"; sel <= "10"; clr <= '0'; w <= "000"; S <= "001";  wait for 60 ns;  CE <= "0000"; sel <= "11"; clr <= '1'; w <= "000"; S <= "000"; cin <= '1'; M <= "101";  wait for 40 ns;  CE <= "0001"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "1000"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "0010"; sel <= "10"; clr <= '0'; w <= "010"; S <= "001";  wait for 40 ns;  CE <= "1000"; sel <= "10"; clr <= '0'; w <= "000"; S <= "001";  wait for 60 ns;  CE <= "0000"; sel <= "11"; clr <= '1'; w <= "000"; S <= "000"; cin <= '1'; M <= "011";  wait for 40 ns;  CE <= "0001"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "1000"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "0010"; sel <= "10"; clr <= '0'; w <= "010"; S <= "001";  wait for 40 ns;  CE <= "1000"; sel <= "10"; clr <= '0'; w <= "000"; S <= "001";  wait for 60 ns;  CE <= "0000"; sel <= "11"; clr <= '1'; w <= "000"; S <= "000"; cin <= '0'; M <= "011";  wait for 40 ns;  CE <= "0001"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "1000"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "0010"; sel <= "10"; clr <= '0'; w <= "010"; S <= "001";  wait for 40 ns;  CE <= "1000"; sel <= "10"; clr <= '0'; w <= "000"; S <= "001";  wait for 60 ns;  CE <= "0000"; sel <= "11"; clr <= '1'; w <= "000"; S <= "000"; cin <= '1'; M <= "000";  wait for 40 ns;  CE <= "0001"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "1000"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "0010"; sel <= "10"; clr <= '0'; w <= "010"; S <= "001";  wait for 40 ns;  CE <= "1000"; sel <= "10"; clr <= '0'; w <= "000"; S <= "001";  wait for 60 ns;  CE <= "0000"; sel <= "11"; clr <= '1'; w <= "000"; S <= "000"; cin <= '0'; M <= "000";  wait for 40 ns;  CE <= "0001"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "1000"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "0010"; sel <= "10"; clr <= '0'; w <= "010"; S <= "001";  wait for 40 ns;  CE <= "1000"; sel <= "10"; clr <= '0'; w <= "000"; S <= "001";  wait for 60 ns;  CE <= "0000"; sel <= "11"; clr <= '1'; w <= "000"; S <= "000"; cin <= '0'; M <= "110";  wait for 40 ns;  CE <= "0001"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "1000"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "0010"; sel <= "10"; clr <= '0'; w <= "010"; S <= "001";  wait for 40 ns;  CE <= "1000"; sel <= "10"; clr <= '0'; w <= "000"; S <= "001";  wait for 60 ns;  CE <= "0000"; sel <= "11"; clr <= '1'; w <= "000"; S <= "000"; cin <= '1'; M <= "110";  wait for 40 ns;  CE <= "0001"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "1000"; sel <= "00"; clr <= '0'; w <= "000"; S <= "010";  wait for 40 ns;  CE <= "0010"; sel <= "10"; clr <= '0'; w <= "010"; S <= "001";  wait for 40 ns;  CE <= "1000"; sel <= "10"; clr <= '0'; w <= "000"; S <= "001";  wait for 60 ns;  wait;  end process;  end Behavioral; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : fsm\_tb.vhd  -- Purpose of Code : Test bench for finite state machine  -- Project Part Number : Lab 4.J  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  LIBRARY IEEE;  USE IEEE.STD\_LOGIC\_1164.ALL;  ENTITY fsm\_tb IS  END fsm\_tb;  ARCHITECTURE beh OF fsm\_tb IS  component fsm  port(clk,sw1: IN STD\_LOGIC;  clr: out std\_logic;  W,S: out std\_logic\_vector(2 downto 0);  CE: out std\_logic\_vector(3 downto 0);  sel: out std\_logic\_vector(1 downto 0)  );  end component;  signal clk, sw1, clr: STD\_LOGIC;  signal W,S: std\_logic\_vector(2 downto 0);  signal CE: std\_logic\_vector(3 downto 0);  signal sel: std\_logic\_vector(1 downto 0);  BEGIN  Uut: fsm port map ( clk => clk, sw1 => sw1, clr => clr, W=>W, S=>S, CE=>CE, sel=>sel);  process  begin  clk <= '0'; wait for 10 ns;  clk <= '1'; wait for 10 ns;  end process;  process  begin  sw1 <= '1';  wait for 40 ns;  sw1<='0';  wait for 40 ns;  sw1<='0';  wait for 40 ns;  sw1<='0';  wait for 40 ns;  sw1<='0';  wait for 40 ns;  wait;  end process;  END beh; |
| -----------------------------------------------------------  -- Author : Grace Pope  -- Date : 12/5/18  -- File Name : alu\_tb.vhd  -- Purpose of Code : Test bench for ALU module  -- Project Part Number : Lab 4.K  --Hardware FPGA/CPLD Device : xc7a100tcsg324-1  -------------------------------------------------------------  LIBRARY IEEE;  USE IEEE.STD\_LOGIC\_1164.ALL;  ENTITY alu\_tb IS  END alu\_tb;  ARCHITECTURE beh OF alu\_tb IS  component alu  port ( A, B, Cin : in std\_logic;  s : in std\_logic\_vector(2 downto 0);  F : out std\_logic  );  end component;  signal A, B, Cin: STD\_LOGIC;  signal S: std\_logic\_vector(2 downto 0);  signal F: std\_logic;  BEGIN  Uut: alu port map ( A => A, B => B, Cin => Cin, S=>S, F=>F);  process  begin  A <= '0'; B<='0'; Cin <= '0'; S <= "000";  wait for 20 ns;  S <="111";  wait for 20 ns;  S <="011";  wait for 20 ns;  A <= '0'; B<='0'; Cin <= '1'; S <= "001";  wait for 20 ns;  A <= '0'; B<='1'; Cin <= '0'; S <= "010";  wait for 20 ns;  A <= '0'; B<='1'; Cin <= '1'; S <= "011";  wait for 20 ns;  A <= '1'; B<='0'; Cin <= '0'; S <= "100";  wait for 20 ns;  A <= '1'; B<='0'; Cin <= '1'; S <= "101";  wait for 20 ns;  A <= '1'; B<='1'; Cin <= '0'; S <= "110";  wait for 20 ns;  S <="001";  wait for 20 ns;  S <="100";  wait for 20 ns;  A <= '1'; B<='1'; Cin <= '1'; S <= "111";  wait for 20 ns;  wait;  end process;  END beh; |

## User Constraint File

|  |
| --- |
| ##-----------------------------------------------------------  ##-- Author : Grace Pope  ##-- Date : 12/5/18  ##-- File Name : lab4.xdc  ##-- Purpose of Code : Constraint file for lab 4  ##-- Project Part Number : Lab 4.L  ##--Hardware FPGA/CPLD Device : xc7a100tcsg324-1  ##-------------------------------------------------------------  ## Clock signal  set\_property -dict { PACKAGE\_PIN E3 IOSTANDARD LVCMOS33 } [get\_ports { clk }]; #IO\_L12P\_T1\_MRCC\_35 Sch=clk100mhz  create\_clock -add -name sys\_clk\_pin -period 10.00 -waveform {0 5} [get\_ports {clk}];  ##Switches  set\_property -dict { PACKAGE\_PIN J15 IOSTANDARD LVCMOS33 } [get\_ports { SW1 }]; #IO\_L24N\_T3\_RS0\_15 Sch=sw[0]  set\_property -dict { PACKAGE\_PIN L16 IOSTANDARD LVCMOS33 } [get\_ports { cin }]; #IO\_L3N\_T0\_DQS\_EMCCLK\_14 Sch=sw[1]  set\_property -dict { PACKAGE\_PIN M13 IOSTANDARD LVCMOS33 } [get\_ports { M[0] }]; #IO\_L6N\_T0\_D08\_VREF\_14 Sch=sw[2]  set\_property -dict { PACKAGE\_PIN R15 IOSTANDARD LVCMOS33 } [get\_ports { M[1] }]; #IO\_L13N\_T2\_MRCC\_14 Sch=sw[3]  set\_property -dict { PACKAGE\_PIN R17 IOSTANDARD LVCMOS33 } [get\_ports { M[2] }]; #IO\_L12N\_T1\_MRCC\_14 Sch=sw[4]  ## LEDs  set\_property -dict { PACKAGE\_PIN H17 IOSTANDARD LVCMOS33 } [get\_ports { Led }]; #IO\_L18P\_T2\_A24\_15 Sch=led[0] |

## Simulation Waveforms

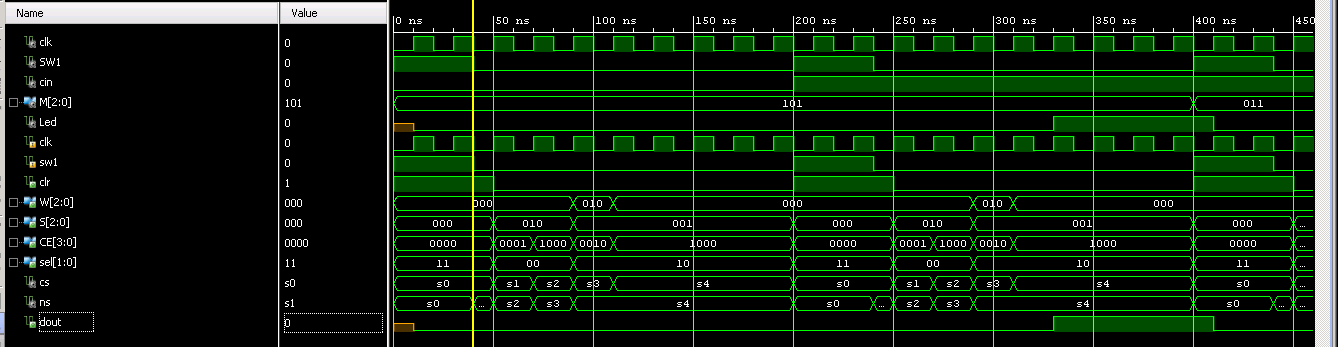


Figure 4: Top level simulation

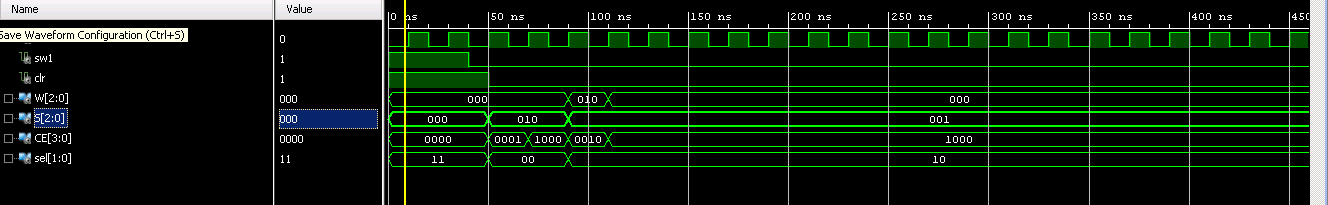


Figure 5: FSM simulation

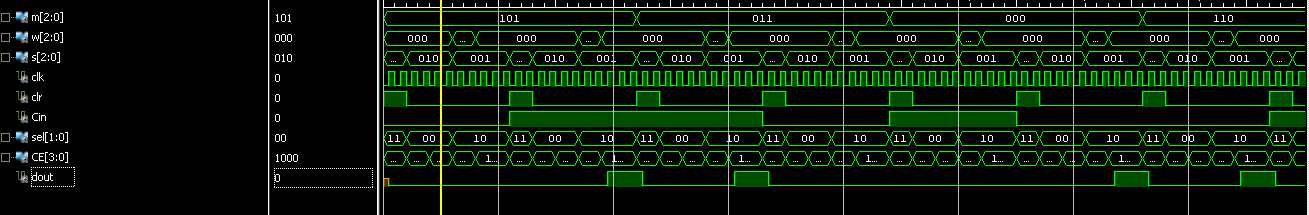


Figure 6: Datapath simulation

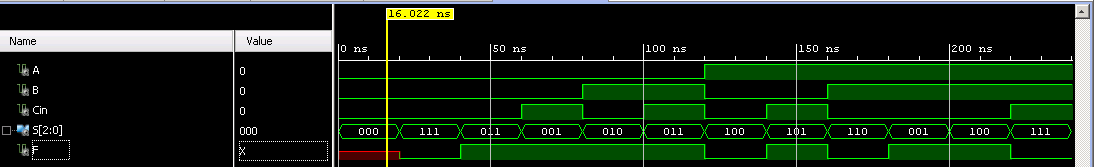


Figure 7: ALU simulation

## Results Discussion

I was able to display the results I wanted when simulating, but when I uploaded my code to the fpga, the led was never completely turned off except when the reset was high. I know there was some kind of error in my code, but I did not have time to find where the error was, nor did I know how to fix it if I did find it. Even though the LED never turned off completely, it would still turn on brighter if the actual result was correct. I think the way I designed my state machine was not correct or something was mislabeled within my other modules. Despite this error, the simplified microprocessor still performed the functions and displayed the results on the fpga.

# Conclusion

This project concludes the labs for this class. I think it was a good way to end the labs because it seemed like a useful application to all of the projects we have been working on in this class. When first starting to design the microprocessor, I was confused about all of the extra wires and inputs described in the figure, but I was able to figure it out and complete the design. The design of the microprocessor was not difficult because we knew all of the modules that the datapath circuit would include. If I had more time, I would figure out why my LED was never turned off, but I am satisfied that I was able to show the final results on the fpga.