CpE Lab

Lab 11: Sequential Logic Design Using Behavioral Modeling

&

Memories and Arithmetic Logic Units

Prepared By:

Olujide Jacobs

Prepared For:

Matt Grubb

November 15, 2013

**Introduction**

The main purpose of this lab is to introduce students to sequential logic, dealing with different states and implementation using combinational logic and flip-flops

**Experiment**

**Part 1 – Pattern recognition using an fsm**

The goal of this part of the lab is to change values randomly relative to a clock and detects the sequence 1101.

**VHDL Code**

**LIBRARY ieee;**

**USE iee.std\_logic\_1164.all;**

**ENTITY fsm IS**

**PORT**

**(**

**clk: in std\_logic;**

**reset: in std\_logic;**

**input1, input2: in std\_logic;**

**output1, output2: in std\_logic;**

**state\_out : out std\_logic**

**);**

**END fsm;**

**ARCHITECTURE rtlof fsm is**

**TYPE state\_type is (state0, state1, state2, state3);**

**SIGNAL state: state\_type;**

**BEGIN**

**PROCESS(clk, reset)**

**BEGIN**

**if reset = '1' then**

**state <= state0;**

**elsif clk'EVENT AND clk = '1' then**

**CASE state is**

**when state0 =>**

**if(input1 = '0') then**

**output1 <= '0';**

**state <= state0;**

**state\_out <= '00'**

**elsif(input2 = '1') then**

**output2 <= '0';**

**state <= state1;**

**state\_out <= '10';**

**END if;**

**when state1 =>**

**if(input1 = '1') then**

**output1 <= '0';**

**state <= state2;**

**state\_out <= '10';**

**END if;**

**when state2 =>**

**if(input1 = '1') then**

**output1 <= '0';**

**state <= state2;**

**state\_out <= '10';**

**elsif(input2 = '0') then**

**output2 <= '0';**

**state <= state3;**

**state\_out <= '00';**

**END if;**

**when state3 =>**

**if(input1 = '1') then**

**output1 <= '1';**

**state <= state1;**

**state\_out <= '11';**

**elsif(input2 = '0') then**

**output2 <= '0';**

**state <= state0;**

**state\_out <= '00';**

**END if;**

**END CASE;**

**END PROCESS;**

The code was compiled, and the input ‘CLK’ was assigned to pin N2, the internal clock of the DE2 board, while input X was assigned to a switch and the outputs ‘q’ were assigned to LEDs.

**Result**

This circuit was tested for three cases: 1001, 0100, and 1101 and it displayed the correct results

**Part II: Arithmetic and logic units**

The assignment here was to design a circuit that uses 4-bit ALU, performs operations between A and B, then display the result.

**VHDL code**

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

USE ieee.std\_logic\_arith.all;

USE ieee.std\_logic\_unsigned.all;

ENTITY alu IS

PORT

(

clk: in std\_logic;

aluOP: in std\_logic\_vector (2 downto 0);

A: in std\_logic\_vector (3 downto 0);

B: in std\_logic\_vector (3 downto 0);

output: out std\_logic\_vector (4 downto 0)

);

END alu;

ARCHITECTURE rtl of alu is

BEGIN

PROCESS(clk, aluOP)

BEGIN

if(aluOP = "000") then

output <= ('0' & A) + ('0' & B);

elsif(aluOP = "001") then

output <= ('0' & A) - ('0' & B);

elsif(aluOP = "010") then

output <= ('0' & A) and ('0' & B);

elsif(aluOP = "011") then

output <= ('0' & A) or ('0' & B);

elsif(aluOP = "100") then

output <= ('0' & B);

elsif(aluOP = "101") then

output <= ('0' & A);

END if;

END PROCESS;

END ARCHITECTURE;

**Result**

3 test cases were examined, 0101 and 1001 gave outputs of 01110, 0001 and 1110 gave an output if 01111, 0110 and 1111 gave output 10101, this was confirmed by calculating the operations A and B by hand.

**Part III: Memory**

The TA provided this part, it basically deals with the storage of data using a memory bank, and then displaying them in the circuit

**Methodology**

**VHDL Code**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity memory is

generic(width:integer:=8;

depth:integer:=32;

addr:integer:=5);

Port(

clk : in std\_logic;

read\_addr: in std\_logic\_vector((addr-1) downto 0);

data\_out: out std\_logic\_vector((width-1) downto 0));

end memory;

architecture behavior of memory is

type ram\_type is array(0 to (depth-1)) of std\_logic\_vector((width-1) downto 0);

signal Z: ram\_type:=("10101110","10110100","10001010","10101010","10101001","00000000","10100101","01010101","10101110","10110100","10001010","10101010","10101001","00000000","10100101","01010101","10101110","10110100","10001010","10101010","10101001","00000000","10100101","01010101","10101110","10110100","10001010","10101010","10101001","00000000","10100101","01010101");

Begin

Process(clk,read\_addr)

Begin

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

Data\_out<=Z(conv\_integer(read\_addr));

end if;

end process;

end;

**Results**

Different circuit styles were displayed with the help of the memory bank, just displaying corresponding data.

**Part IV: Memory II**

This is basically the same thing as the previous part, just more ports added and more conditional if statements

**Methodology**

**VHDL Code**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity memory is

generic(width:integer:=8;

depth:integer:=32;

addr:integer:=5);

Port(

clk : in std\_logic;

read\_addr: in std\_logic\_vector((addr-1) downto 0);

data\_out: out std\_logic\_vector((width-1) downto 0));

end memory;

architecture behavior of memory is

type ram\_type is array(0 to (depth-1)) of std\_logic\_vector((width-1) downto 0);

signal Z: ram\_type:=("10101110","10110100","10001010","10101010","10101001","00000000","10100101","01010101","10101110","10110100","10001010","10101010","10101001","00000000","10100101","01010101","10101110","10110100","10001010","10101010","10101001","00000000","10100101","01010101","10101110","10110100","10001010","10101010","10101001","00000000","10100101","01010101");

Begin

Process(clk,read\_addr)

Begin

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

Data\_out<=Z(conv\_integer(read\_addr));

end if;

end process;

end;

**Results**

The circuit displayed the corresponding data in the memory bank with the switches and leds

**Conclusion**

Although the group encountered some problem along the way, this was still a successful lab. I was able to brush up my circuit design and make it to work.

**Post Lab Questions**