**ELEC 204 Digital Design Lab Report**

Lab 3

Name: Mehmet Enes Erciyes

Date: 11/10/2019

\*Please delete the highlighted lines and write your own parts for the report.

\*\*Reminder: Your lab grade is a weighted average of your performance before, during and after the lab: **Total lab grade = Preliminary work\* (30%) + Lab interview and demo\* (40%) + Lab report (30%)**

**\*\*\*Please make sure the lab report does not exceed 4 A4 pages.**

\*\*\*\*Please make sure you indicate the name of your lab collaborator (if there is any) who you worked together to solve the lab questions. Do not change your lab collaborator throughout the semester.

1. **Introduction and objectives**

Main objective of this lab is to be able to use simple combinational logic circuit elements to build modules and use these modules to build hierarchical architectures.

In this experiment, our code first implements a one bit ALU, a unit consisting of an arithmetic and logic unit. In this part, we get one bit inputs and map them to both an arithmetic output and a logic output.

Second part of the code is cascading the one bit ALUs to build a complete 4 bit ALU. After computing every output from F(3) – F(0). In this part, we also shift our numbers from 2’s complement to standard binary. We determine if our number should have a negative sign. Then, according to the M(mode) selection, we select which output to show and turn off the other output. If arithmetic mode is selected, seven segment is turned on, otherwise LEDs work.

At the last part, we combine the output, LEDs and Seven Segment Display, with the result of the 4-bit ALU. We connect the inputs and outputs to correct pins with the ucf file.

1. **Methods**

Explain the inputs (how many bits, names of the inputs),

Explain the outputs (how many bits, names of the outputs),

Explain what the VHDL code must do

Explain how your code works

Provide the truth table

1. **Problems encountered, errors and warnings resolved**

Explain what problems you encountered while writing your code.

Explain what synthesis errors and warnings you observed.

Explain what problems you had to solve (or could not) on your board even if your code could be synthesized successfully.

1. **Conclusion**

Provide a 1 paragraph summary of the lab and explain what you learned from this lab.

References

1. Please cite any resource (web site, book, youtube video) you used for this lab.

**Appendix 1. Lab source code**

**Appendix 2. RTL schematics**

**Appendix 3. FPGA Board photos showing working code**

**Appendix 4. Screenshots from Xilinx for the errors and other board issues**

**ALU- one bit**

entity ALU\_onebit is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

s : in STD\_LOGIC\_VECTOR (1 downto 0);

c\_in : in STD\_LOGIC;

F\_logic : out STD\_LOGIC;

F\_arithmetic : out STD\_LOGIC;

c\_out : out STD\_LOGIC);

end ALU\_onebit;

architecture Behavioral of ALU\_onebit is

signal x,y : std\_logic;

begin

process

begin

if s = "00" then

F\_logic <= A and B;

elsif s = "01" then

F\_logic <= A or B;

elsif s = "10" then

F\_logic <= A xor B;

elsif s = "11" then

F\_logic <= A xnor B;

end if;

x <= (((not (s(1) and s(0))) and A) or ((s(1) and s(0)) and (not A)));

y <= (s(0) and B) or (s(1) and (not s(0)) and not B);

F\_arithmetic <= x xor y xor c\_in;

c\_out <= ((x xor y) and c\_in) or (x and y);

end process;

end Behavioral;

**ALU- 4bit**

entity ALU is

Port ( A : in STD\_LOGIC\_VECTOR (3 downto 0);

B : in STD\_LOGIC\_VECTOR (3 downto 0);

M : in STD\_LOGIC;

s : in STD\_LOGIC\_VECTOR (1 downto 0);

c0 : in STD\_LOGIC;

F\_logic: out STD\_LOGIC\_VECTOR (3 downto 0);

F\_arithmetic: out STD\_LOGIC\_VECTOR (3 downto 0);

neg\_sign: out STD\_LOGIC\_VECTOR (3 downto 0);

c4 : out STD\_LOGIC);

end ALU;

architecture Behavioral of ALU is

COMPONENT ALU\_onebit

PORT(

A : IN std\_logic;

B : IN std\_logic;

s : IN std\_logic\_vector(1 downto 0);

c\_in : IN std\_logic;

F\_logic : OUT std\_logic;

F\_arithmetic : OUT std\_logic;

c\_out : OUT std\_logic

);

END COMPONENT;

signal F\_logictemp : std\_logic\_vector(3 downto 0);

signal F\_arithmetictemp : std\_logic\_vector(3 downto 0);

signal F\_arithmetictemp\_binary: std\_logic\_vector(3 downto 0);

signal c\_out1temp : std\_logic;

signal c\_out2temp : std\_logic;

signal c\_out3temp : std\_logic;

signal c\_out4temp : std\_logic;

begin

first: ALU\_onebit PORT MAP(

A => A(3),

B => B(3),

s => s,

c\_in => c0,

F\_logic => F\_logictemp(0),

F\_arithmetic => F\_arithmetictemp(0),

c\_out => c\_out1temp

);

second: ALU\_onebit PORT MAP(

A => A(2),

B => B(2),

s => s,

c\_in => c\_out1temp,

F\_logic => F\_logictemp(1),

F\_arithmetic => F\_arithmetictemp(1),

c\_out => c\_out2temp

);

third: ALU\_onebit PORT MAP(

A => A(1),

B => B(1),

s => s,

c\_in => c\_out2temp,

F\_logic => F\_logictemp(2),

F\_arithmetic => F\_arithmetictemp(2),

c\_out => c\_out3temp

);

fourth: ALU\_onebit PORT MAP(

A => A(0),

B => B(0),

s => s,

c\_in => c\_out3temp,

F\_logic => F\_logictemp(3),

F\_arithmetic => F\_arithmetictemp(3),

c\_out => c\_out4temp

);

with F\_arithmetictemp select F\_arithmetictemp\_binary <=

"0000" when "0000",

"0001" when "0001",

"0010" when "0010",

"0011" when "0011",

"0100" when "0100",

"0101" when "0101",

"0110" when "0110",

"0111" when "0111",

"1000" when "1000",

"0111" when "1001",

"0110" when "1010",

"0101" when "1011",

"0100" when "1100",

"0011" when "1101",

"0010" when "1110",

"0001" when others;

process

begin

if M = '0' then

F\_logic <= F\_logictemp;

F\_arithmetic <= x"f";

neg\_sign <= x"f";

c4 <= '0';

else

F\_arithmetic <= F\_arithmetictemp\_binary;

if F\_arithmetictemp(3) = '1' then

neg\_sign <= x"a";

else

neg\_sign <= x"f";

end if;

c4 <= c\_out4temp;

F\_logic <= x"0";

end if;

end process;

end Behavioral;

**Main**

entity main is

Port ( A : in STD\_LOGIC\_VECTOR (3 downto 0);

B : in STD\_LOGIC\_VECTOR (3 downto 0);

s : in STD\_LOGIC\_VECTOR (1 downto 0);

M : in STD\_LOGIC;

c\_in : in STD\_LOGIC;

LEDs : out STD\_LOGIC\_VECTOR (3 downto 0);

c\_out : out STD\_LOGIC;

clk: in std\_logic;

SevenSegControl: out std\_logic\_vector (7 downto 0):= x"ff";

SevenSegBus: out std\_logic\_vector (7 downto 0)

);

end main;

architecture Behavioral of main is

COMPONENT sevenSegment

PORT(

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

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

C : IN std\_logic\_vector(3 downto 0);

D : IN std\_logic\_vector(3 downto 0);

E : IN std\_logic\_vector(3 downto 0);

F : IN std\_logic\_vector(3 downto 0);

G : IN std\_logic\_vector(3 downto 0);

H : IN std\_logic\_vector(3 downto 0);

clk : IN std\_logic;

SevenSegControl : OUT std\_logic\_vector(7 downto 0);

SevenSegBus : OUT std\_logic\_vector(7 downto 0)

);

END COMPONENT;

COMPONENT ALU

PORT(

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

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

M : IN std\_logic;

s : IN std\_logic\_vector(1 downto 0);

c0 : IN std\_logic;

F\_logic : OUT std\_logic\_vector(3 downto 0);

F\_arithmetic : OUT std\_logic\_vector(3 downto 0);

neg\_sign : OUT std\_logic\_vector(3 downto 0);

c4 : OUT std\_logic

);

END COMPONENT;

signal alu\_logicout: std\_logic\_vector(3 downto 0);

signal alu\_arithmeticout : std\_logic\_vector(3 downto 0);

signal negsigntemp : std\_logic\_vector(3 downto 0);

begin

ALU\_main: ALU PORT MAP(

A => A,

B => B,

M => M,

s => s,

c0 => c\_in,

F\_logic => alu\_logicout,

F\_arithmetic => alu\_arithmeticout,

neg\_sign => negsigntemp,

c4 => c\_out

);

sevenSegmentInst: sevenSegment PORT MAP(

A => alu\_arithmeticout,

B => negsigntemp,

C => x"f",

D => x"f",

E => x"f",

F => x"f",

G => x"f",

H => x"f",

clk => clk,

SevenSegControl => SevenSegControl,

SevenSegBus => SevenSegBus

);

LEDs <= alu\_logicout;

end Behavioral;