

**Department of Electrical Engineering**

**Optional 1: 2-bit Multiplier**

Name: Socheath Sok, ID: 014470701

Professor: Dr. Ray Wang

Class: EE 301

Date Due: December 11, 2017

**Explanation**

The goal of this lab is to design a 2-bit multiplier. This can be accomplished in multiple ways which includes describing the design behavior based on the truth table, making a design with full adders as the component or creating the design using half adders. From the options given, the half-adder design seem to be the most simple and shortest one because there is no complicated connection and less inputs and signals are required since there is no ‘carry in’ input for the half adders. In addition to the adders, four AND gates were also required to combine inputs ‘a’ and ‘b’, which are connected to the switches, before they can be connected to the adders. Lastly, the structural code for the 2-bit multiplier with four input, four output and one wire signal to connect the two half adders was created, tested and implemented on the board.

**VHDL Code for Half Adder**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity Half\_Adder is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

S : out STD\_LOGIC;

C : out STD\_LOGIC);

end Half\_Adder;

architecture Behavioral of Half\_Adder is

begin

S <= A xor B;

C <= A and B;

end Behavioral;

**Structural Code**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity Multiplier is

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

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

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

end Multiplier;

architecture Behavioral of Multiplier is

component Half\_Adder

Port( A : in STD\_LOGIC;

B : in STD\_LOGIC;

S : out STD\_LOGIC;

C : out STD\_LOGIC);

end component;

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

signal carry : std\_logic;

begin

x(0) <= a(0) and b(0);

x(1) <= a(0) and b(1);

x(2) <= a(1) and b(0);

x(3) <= a(1) and b(1);

c(0) <= x(0);

Q1: Half\_Adder Port Map(x(1), x(2), c(1), carry);

Q2: Half\_Adder Port Map(carry, x(3), c(2), c(3));

end Behavioral;

**TestBench**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_unsigned.ALL;

ENTITY Multiplier\_test IS

END Multiplier\_test;

ARCHITECTURE behavior OF Multiplier\_test IS

COMPONENT Multiplier

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

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

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

END COMPONENT;

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

signal a : std\_logic\_vector(1 downto 0) := (others => '0');

signal b : std\_logic\_vector(1 downto 0) := (others => '0');

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

BEGIN

uut: Multiplier PORT MAP (

a => test\_seq(3 downto 2),

b => test\_seq(1 downto 0),

c => c );

stim\_proc: process

BEGIN

test\_seq <= "0000";

wait for 10 ns;

for i in 0 to 15 loop

test\_seq <= test\_seq + "0001";

wait for 10 ns;

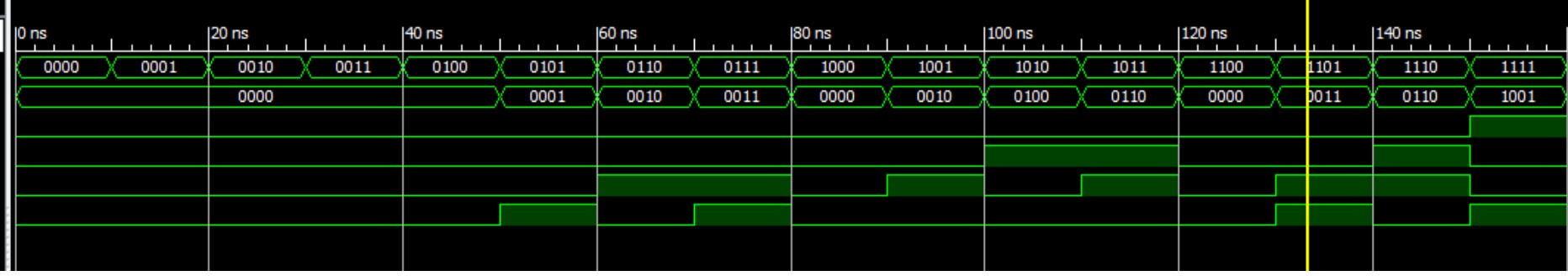
end loop;

wait;

end process;

END;

**WaveForm**



**Constraint File**

Net "a(0)" LOC = "P11";

Net "a(1)" LOC = "L3";

Net "b(0)" LOC = "K3";

Net "b(1)" LOC = "B4";

Net "c(0)" LOC = "M5";

Net "c(1)" LOC = "M11";

Net "c(2)" LOC = "P7";

Net "c(3)" LOC = "P6";