**Digital Design (CSCE 2114)**

**Lab 6 - Part I**

# **Objective:**

1. Learn how to write VHDL code to implement some simple combinational circuits.
2. Learn how to simulate your design before implementing it on the FPGA to verify that it works.

**VHDL Code Hints:**

1. VHDL is a **case insensitive** and **strongly typed** language.
2. Comments start with two adjacent hyphens **--** and end at end of line.
3. Every line ends with **;** like C language.
4. Please, refer to “[VHDL Notes](http://www.csce.uark.edu/~jparkers/CSCE2114-spring2017/index.html)“, “[VHDL Data Types](http://www.csce.uark.edu/~jparkers/CSCE2114-spring2017/index.html)”, and “[VHDL Operators](http://www.csce.uark.edu/~jparkers/CSCE2114-spring2017/VHDL-Operators.pdf)” on class website for more examples and syntax.

**Exercises**

1. **3-Input Function**
   1. In Quartus, create a new project named lab6\_a .
      1. **Please refer to earlier labs** if you don’t know how to perform this step.
      2. (Note: If prompted for license, use: [1800@csce-licsrv.ddns.uark.edu](mailto:1800@csce-licsrv.ddns.uark.edu)).
   2. Click on File -> New and select VHDL File.
   3. Write VHDL code of the combinational function:
      1. **F(x1, x2, x3) = Sm(0, 2, 3, 4, 5, 6).** 
         1. You are free to optimize your circuit and then implement it.
         2. You can use simple operators like: **not**, **and**, **or**.
         3. Use parenthesis to prioritize the operators (refer to the VHDL\_note.ppt for some examples).
   4. Code template:

------------------------------------

-- Project Name:

-- Target Devices:

-- Description:

-- Revision:

-- Additional Comments:

------------------------------------

LIBRARY ieee ;

USE ieee.std\_logic\_1164.all ;

ENTITY lab6\_a IS

PORT ( x1,x2,x3 : in std\_logic;

f : out std\_logic);

END lab6\_a ;

ARCHITECTURE Behavior OF lab6\_a IS

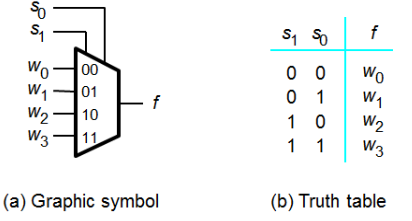
BEGIN

f<= …...................; --*function logic goes here.*

END Behavior ;

* 1. After finishing your code, compile your project and correct any errors.
  2. Add a waveform file and add all the 3 inputs and one output nodes to your waveform list as described in an earlier lab. Apply all 8 possible inputs with 100ns intervals.
     1. 0 -100ns : x1=0; x2=0; x3=0;
     2. 100ns - 200ns : x1=0; x2=0; x3=1;
     3. 200ns - 300ns : x1=0; x2=1; x3=0;
     4. 300ns - 400ns : x1=0; x2=1; x3=1;
     5. 400ns - 500ns : x1=1; x2=0; x3=0;
     6. 500ns - 600ns : x1=1; x2=0; x3=1;
     7. 600ns - 700ns : x1=1; x2=1; x3=0;
     8. 700ns - 800ns : x1=1; x2=1; x3=1;
  3. Run the simulation and check the output/results. Your simulation results should show that the output of your design matches the truth table of function f.
  4. Save a screenshot of both your VHDL code and your waveform results. Show your work to the TA before you close your project and move on to the next exercise.

1. **Implementing a Multiplexer**
   1. In Quartus, create a new project named lab6\_b .
   2. Click on File -> New and select VHDL File
   3. Write the VHDL code for a 4-1 Multiplexer. You can refer to the VHDL\_note.ppt for more help.



* 1. Code template:

------------------------------------

-- Project Name:

-- Target Devices:

-- Description:

-- Revision:

-- Additional Comments:

------------------------------------

LIBRARY ieee ;

USE ieee.std\_logic\_1164.all ;

ENTITY lab6\_b IS

PORT ( w0, w1, w2, w3 : in std\_logic;

s : IN STD\_LOGIC\_vector (1 downto 0) ;

f : OUT STD\_LOGIC ) ;

END lab6\_b ;

ARCHITECTURE Behavior OF lab6\_b IS

BEGIN

PROCESS ( w0, w1,w2, w3, s)

BEGIN

CASE s IS

WHEN '00' =>

f <= w0 ;

...............

WHEN OTHERS =>

f <= w3 ;

END CASE ;

END PROCESS ;

END Behavior ;

* 1. After finishing your code, compile your project and correct any errors.
  2. Add a waveform file and add all the inputs and outputs to your waveform list. Assign ‘1’ to *W0*, ‘X’ to *W1*, ‘Z’ to *W2* and ‘0’ to *W3*. Check output *f* for all possible *S0, S1* inputs.
     1. 0 -100ns : s=00;
     2. 100ns - 200ns : s=01;
     3. 200ns - 300ns : s=10;
     4. 300ns - 400ns : s=11;
     5. 400ns - 500ns : s=00;
     6. 500ns - 600ns : s=01;
     7. 600ns - 700ns : s=10;
     8. 700ns - 800ns : s=11;
  3. Run the simulation and check the output/results. Your simulation results should show that the output of your design matches the truth table of the multiplexer.
  4. Save a screenshot of both your VHDL code and your waveform results. Show your work to the TA before you close the project.

**Lab Report**

There will be no lab report due next week. The lab report will be due one week after we complete part 2 of this lab; however, you should include these things in your final lab report.

1. Screenshot of both of your VHDL code and your waveforms results for each exercise.
2. List and describe steps involved in VHDL code compilation in the Quartus tool.
3. How does this compilation process differ from traditional C programs compilation?
4. What could be improved about this lab?