CET4805 **Component and Subsystem Design II**

EXPERIMENT # 4: <u>7-segment Display with VHDL</u>

Name: ____Date: _______________

Equipment/Parts Needed:

Quartus II[R] Web Edition V9.1 SP2 software by Altera Corporation
USB drive to save your files

Objective:

- Use simple VHDL assignment statements to represent a function table
- Introduce the selected signal assignment WHEN-ELSE clause
- Display hexadecimal numbers (0 through F) on the 7-segment LED of the DE-2 board

Discussion:

Seven segment LED displays are often found in clock radios, VCRs, microwave ovens, toys and many other household items. They are primarily used to display decimal numbers but they can also display a few alphabets and other characters. This experiment describes interfacing a seven segment LED display to the DE-board. You will display a hexadecimal value from 0 (00h) to 15 (0Fh) on the seven segment LED display.

A seven segment LED display is a special arrangement of 7 LED elements to form a rectangular shape using two vertical segments on each side with one horizontal segment on the top, middle, and bottom. By individually turning the segments on or off, numbers from 0 to 9 and some letters can be displayed. Seven segment displays sometime also have an eighth segment to display the decimal point. Therefore, a seven-segment display will require seven outputs from the schematic design to display a number, and one more output if the decimal point is to be displayed too. (This experiment ignores the decimal point.)

The 7 LEDs inside the display can be arranged with a common cathode or common anode configuration. With a common cathode display, the cathodes of all the segment LEDs are tied together and this common point must be connected to the ground. A required LED segment is then turned on by applying a logic 1 to its anode. In common anode displays, all the anodes are tied together and the common anode is connected to the supply voltage Vcc. Individual segments are turned on by applying logic 0 to their cathodes. Since D2-2 has a common anode 7-segment, displaying a number requires turning on and off the proper segment LEDs. For example, to display a number 7, only segments a, b, and c should be turned on, which means their logic levels should be low.

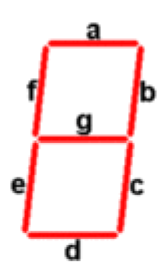

Figure 4-1 7-segment LED

CET4805 **Component and Subsystem Design II**

Let’s assign a bus value using D3, D2, D1, and D0 to represented each bit position in a 4-bit bus D (also represented by D[3…0]. Also, the outputs will be represented by a 7-bit bus, S[6..0], as shown in Table 4-1. Assuming the buses are properly defined in the Entity Declaration, we can say that bus S will take on the 7-bit value that corresponds to the value of bus D.

Symbol	D3	D2	D1	D0	Shape	S6	S5	S4	S3	S2	S1	S0
0	0	0	0	0		1	0	0	0	0	0	0
1	0	0	0	1		1	1	1	1	0	0	1
2	0	0	1	0		0	1	0	0	1	0	0
3	0	0	1	1		0	1	1	0	0	0	0
4	0	1	0	0		0	0	1	1	0	0	1
5	0	1	0	1		0	0	1	0	0	1	0
6	0	1	1	0		0	0	0	0	0	1	0
7	0	1	1	1		1	1	1	1	0	0	0
8	1	0	0	0		0	0	0	0	0	0	0
9	1	0	0	1		0	0	1	0	0	0	0
A	1	0	1	0								
B	1	0	1	1								
C	1	1	0	0								
D	1	1	0	1								
E	1	1	1	0								
F	1	1	1	1								

Table 4-1

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY vhdl4_1 IS
      PORT   ( D        :IN std_logic_vector(3 downto 0);
               S        :OUT std_logic_vector (6 downto 0) );
END vhdl4_1;
ARCHITECTURE behavior OF vhdl4_1 IS
      Begin
            S <=  "1000000" when D="0000" else
                  "1111001" when D="0001" else
                  "0100100" when D="0010" else
                  "0110000" when D="0011" else
                  "0011001" when D="0100" else
                  "0010010" when D="0101" else
                  "0000010" when D="0110" else
                  "1111000" when D="0111" else
                  "0000000" when D="1000" else
                  "0010000" ;
END behavior;
```

Text Box 4-1

CET4805 **Component and Subsystem Design II**

Part 1 Procedure

Creating a New Project

1. Open the Quartus II software. Select **File – New Project Wizard**. Enter the appropriate drive letter for the designated storage area on the computer you are using followed by the working directory C**:\altera\91sp2\quartus\kwon\Lab4**. You need to go through the step from 1 through 8 in the Part 1 of **Lab1** manual. Don't forget to create the folder **Lab4** under the subfolder of your last name. Assign the project name **Lab4_1**, assign ***Cyclone II*** for the device family, and select the ***EP2C35F672C6*** chip in the Family & device settings [page of 3 of 5].

Creating a VHDL File (bdf)

2. Open a new VHDL Device Design file (**File > New**) by highlighting VHDL File. Type the VHDL codes shown in Text Box 4-1.

3. Save the VHDL file as **vhdl4_1.vhd** as part of our project under your subfolder. Place a check mark in the space labeled Add file to current project and press Save.

4. Select **File > Create/Update > Create Symbol Files for Current File** to create a symbol file for the VHDL code entered. A display window should soon appear stating that the **Create Symbol File was (or not) successful**. Click **OK** and close the Compilation Report window.

5. Open a new Schematic file (**File > New**) by highlighting **Block Diagram/Schematic File**. And click **OK**. And construct the circuit shown in Figure 4-2 using the symbols you just created. Each symbol should be available in the Project Library in the Symbol diagonal box.

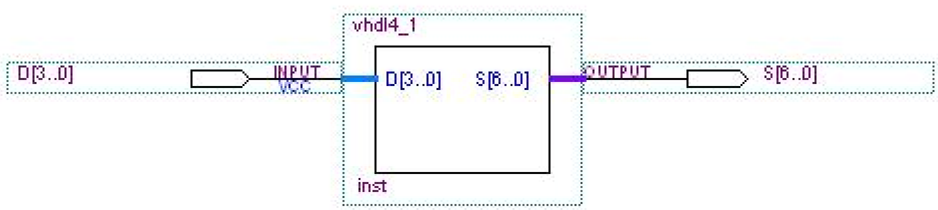

Figure 4-2

6. Before compiling this *bdf* file, we need to name this *bdf* file and save it as part of our project under your subfolder. Choose **File > Save As** and enter **File name** as *lab4_1*. Place a *check mark* in the space labeled **Add file to current project** and press **Save**.

7. Compile the project by selecting **Processing** > **Start Compilation**, or press Ctrl-L, or use the Compilation button in the toolbar. The compilation takes several seconds. When it is complete it should give a message that indicates, "Full compilation was successful". Press **OK**. If unsuccessful, correct all errors and try to re-compile.

Simulating a Vector Waveform File (vwf)

8. As you have done step 23 through 28 in the **Part 1** of Lab1, you need to create a **Vector Waveform File (*vwf*)** to simulate a design(*bdf*) file. Add all inputs and output, specify an end time of 1 μs and a grid size of 100 ns for our waveform display, and then save it as lab4_1.vwf.

9. When creating the D[3..0] bus, enter D for the bus name, select Hexadecimal for the Radix, and enter 4 for the Bus Width in the Node Properties window. When created, the D waveform will appear with a plus sign implying that it can be ungrouped to show the individual bits, D[3], D[2], D[1], and D[0].

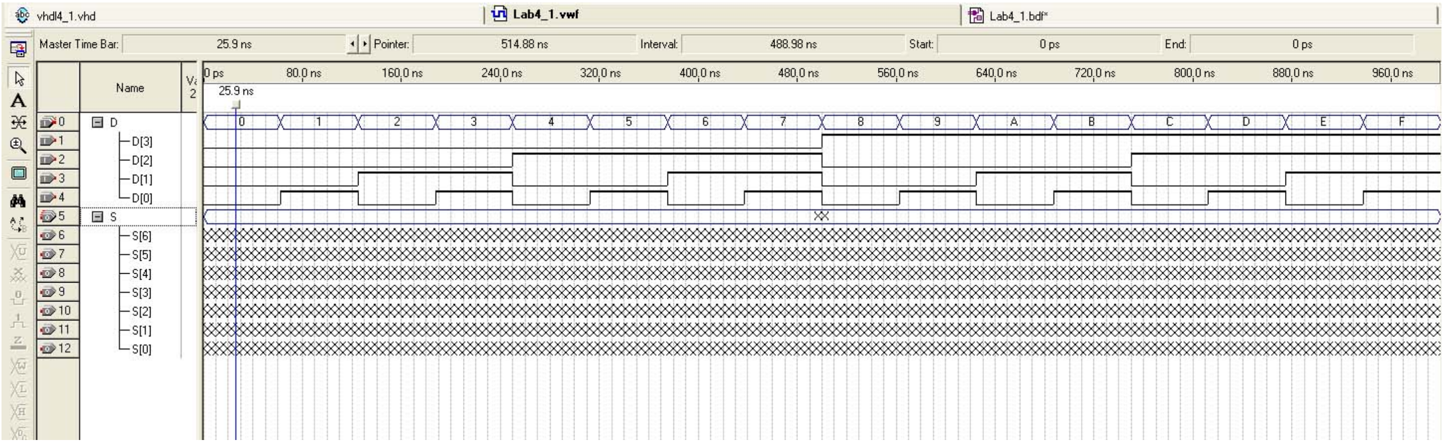

Figure 4-3

10. Select **Processing – Start Simulation**, or press Ctrl-I, or use the Simulation button in the toolbar. After a few moments a message stating "Simulation was successful" should appear. Click **OK**.

11. The Simulation Waveforms appear in the Simulation Report. You may have to expand the size of the Simulation Waveforms to suit your need and choose **View > Fit in Window** to see the entire waveform.

12. Program your schematic design into DE-2 board to display the values from '0' through '9'.

Input	Component	PIN	Output	Component	PIN
D[0]	SW[0]	PIN_N25	S[0]	HEX0[0]	
D[1]	SW[1]	PIN_N26	S[1]	HEX0[1]	
D[2]	SW[2]	PIN_P25	S[2]	HEX0[2]	
D[3]	SW[3]	PIN_AE14	S[3]	HEX0[3]	
			S[4]	HEX0[4]	
			S[5]	HEX0[5]	
			S[6]	HEX0[6]	

Table 4-2 Pin assignments

Part 2 Practice

1. Complete the Table 4-1 and display the value from 0 through F on the 7-segment of DE-2 board.

   1) Create the VHDL codes for the values from 'A' through 'F'.
   2) Create a Block Design File (*bdf* file) for X using the symbol created from the *vhdl* file.
   3) Create a Vector Waveform File (*vwf*) for X. The simulation should show all possible combination of inputs.
   4) Include the copies of *vhdl* codes and *bdf* file and *vwf* file as well as your pin assignment in the lab report.