

Worcester Polytechnic Institute

Electrical and Computer Engineering Department

ECE3801 – Lab1 (Seven Segment Display)

Copyright: Feb, 2005

Written by Ahmad Hatami
Modified by Samant Kakarla

Modified 11/2006: S. Jarvis

Please have all individual components created in this lab available to all parties working together to complete the lab. Make sure everyone also has the components available individually from previous labs. It is a good idea to backup your data on another media apart from the network space you have available.

Objective

In this lab you will design a circuit that drives the seven segment displays on Spartan-3 Starter Board. You will learn how to:

- Use Webpack to capture a schematic design.
- Do a hierarchical design
- Use MSI circuits in your designs

Big Picture

Background

In digital systems, when we want to represent numbers as human readable information, we use an arrangement of LED's in a shape called seven segment displays. Figure 1 shows such an arrangement. In order to represent any hex number between 0000-1111 we can illuminate a combination of these LEDs as shown in Figure 1. In most cases all LED's in a seven segment are common cathode. To illuminate an LED you will impose a logic one voltage to its input.

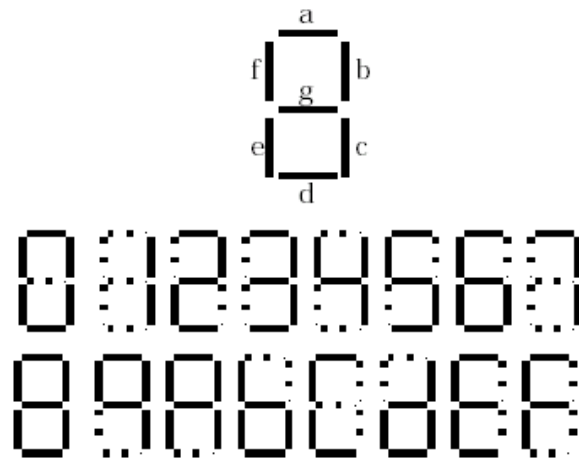


Figure 1

Seven Segment Display on Spartan3 Board

The Spartan-3 Starter Kit board has a four-character, seven-segment LED display controlled by FPGA user-I/O pins, as shown in Figure 2 Each digit shares eight common control signals to light individual LED segments. Each individual character has a separate

anode control input. A detailed schematic for the display appears in [1]. To light an individual signal, drive the individual segment control signal Low along with the associated anode control signal for the individual character. In Figure 2 for example, the left-most character displays the value '2'. The digital values driving the display in this example are shown in blue. The AN3 anode control signal is Low, enabling the control inputs for the left-most character. The segment control inputs, A through G and DP, drive the individual segments that comprise the character. A Low value lights the individual segment, a High turns off the segment. A Low on the A input signal, lights segment 'a' of the display. The anode controls for the remaining characters, AN[2:0] are all High, and these characters ignore the values presented on A through G and DP.

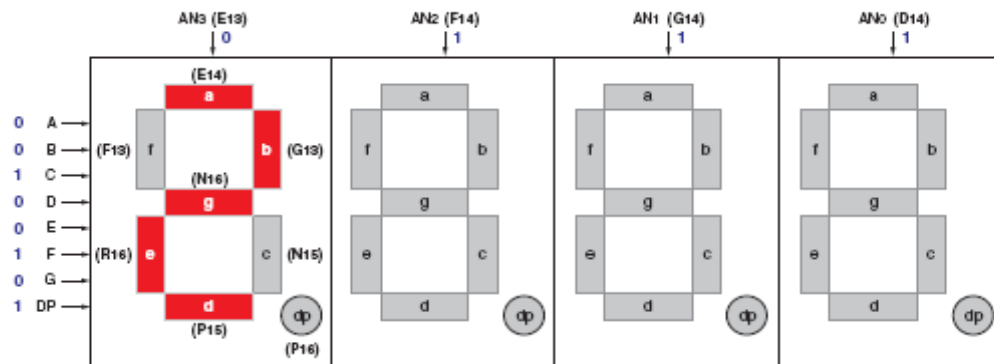


Figure 2

Figure 3 shows the input/output interface of your design. Four 4-bits hexadecimal inputs ($W(3:0), X(3:0), Y(3:0), Z(3:0)$) with their associated decimal points (DP_W, DP_X, DP_Y, DP_Z) must be displayed on the four digit seven segment display of the board from left to right respectively. Sometimes you need to display a negative sign on a seven segment. Your system provides additional 4-bit sign inputs (S_W, S_X, S_Y, S_Z) for that purpose. An activated sign bit for a segment supersedes its hexadecimal value. As an example if S_W is activated the system displays a minus sign in the leftmost seven segment display and ignores the inputs on $W(3:0)$.

Implementation

In order to implement a complicated project we need to use a divide and conquer approach. We start with a simple component and build a hierarchy of simple components to complete our overall design.

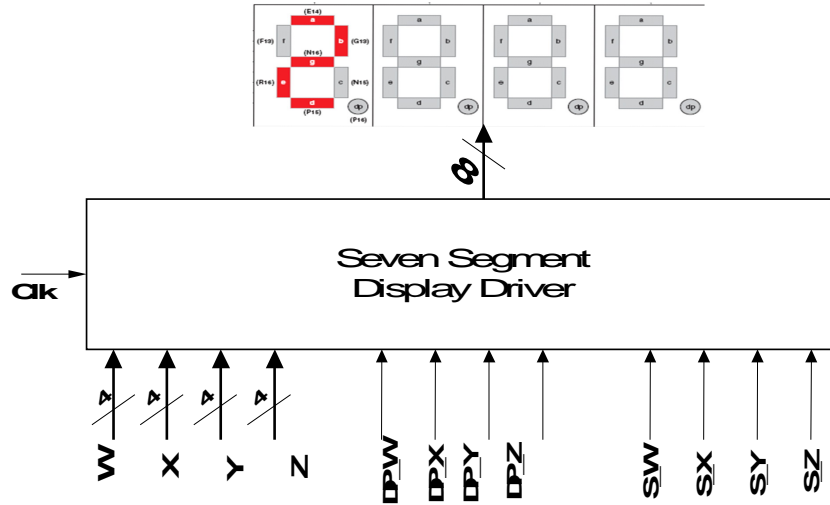


Figure 3

Pre-Lab

Let's start with a simple building block as depicted in Figure 1 This block converts a 4-bit hex number to a seven segment decoded value. You need to implement seven different functions of four input digits ($D3, D2, D1, D0$), based on the truth table depicted in Table 1. Each output drives one segment of the display. Note that seven segments of Spartan3 board are active low.

Character	Inputs				Outputs						
	D3	D2	D1	D0	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	1	1	0	0	1	1	1	1
2	0	0	1	0	0						
3	0	0	1	1	0						
4	0	1	0	0	1						
5	0	1	0	1	0						
6	0	1	1	0	0						
7	0	1	1	1	0						
8	1	0	0	0	0						
9	1	0	0	1	0						
A	1	0	1	0	0						
B	1	0	1	1	1						
C	1	1	0	0	0						
D	1	1	0	1	1						
E	1	1	1	0	0						
F	1	1	1	1	0						

Table 1

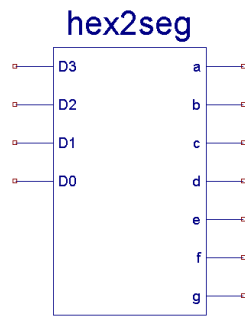


Figure 4

1. Complete the Table 1 based on Figures 1-3
2. Pick two segments from (*a, b ... g*) and find the minimized function for them.
3. Review the lab tutorial provided in the course web page.

In Lab

1. Open the project navigator and create a new project.
2. Click on add a new source.
3. Select VHDL module as the new source file type.
4. Since some of the components that you design in this lab provide the same function of your tutorial projects, we suggest to add a prefix 'v_' to the file names to distinguish between the files that are designed by VHDL and those designed previously. So name your file 'v_hex2seg' and click on next.
5. On this dialog box you can define the IO interface of your design. Use Fig. 5 as a guideline and click on next and click on finish on the next screen.

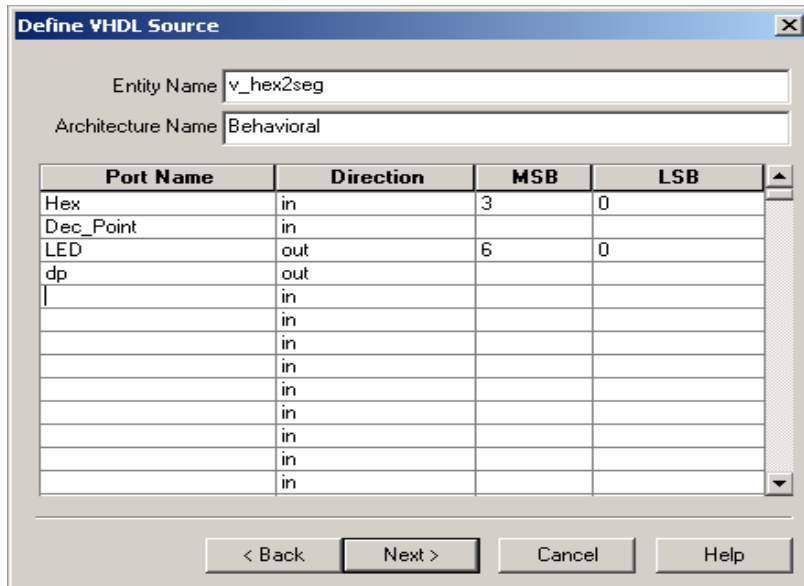


Figure 5

6. Xilinx generates a VHDL file with appropriate entity declaration and a template for the architecture of your design.
7. Use a conditional signal assignment (WITH-SELECT-WHEN) to generate the appropriate outputs for the 'LED' output. Table 2 shows one case when the input is equal to zero.
8. Use a Boolean equation to generate the output for the 'dp' output.
9. Save your file.
10. Make sure that your new VHDL file is selected in the project navigator.
11. On the process window double click on 'View RTL Schematic' under 'Synthesize' section of the project navigator.
12. If there is no syntax error in your file this opens a schematic window describing your VHDL design. Push into the generated symbol and see how Xilinx has implemented your design.
13. Create a test bench waveform for this file and make sure that your design functions correctly.
14. You can create a symbol for this design and use it in your schematics by double clicking on 'Create Schematic Symbol' under 'Design Utilities' on the

project navigator. Now when you create a schematic, the symbol of this design should be available under the directory you saved it in.

```

WITH Hex SELECT
  LED <= "0000001" WHEN "0000",
  .
  .
  " _ _ _ _ _ " WHEN OTHERS

```

Table 2

1. Figure 6 shows a more advanced implementation of 7-seg decoder. Here are the description for each pin:
 - a. **CS** (Chip Select): When activated the chip decodes the input values otherwise the chip is not active
 - b. **Sign**: You may want to use a single display to show a negative sign when dealing with signed numbers. When sign is activated (High) the output should display a minus sign.
 - c. **Point**: When activated you want to activate the decimal point of the display.
 - d. **D(3:0)**: 4-bit hexadecimal number to be decoded

CS	Sign	Point	D(3:0)	a – g	DP
0	X	X	X X X X	H H H H H H H	H
1	0	0	D ₃ D ₂ D ₁ D ₀	Decoded value for D(3:0)	H
1	0	1	D ₃ D ₂ D ₁ D ₀	Decoded value for D(3:0)	L
1	1	0	X X X X	H H H H H H L	H
1	1	1	X X X X	H H H H H H L	L

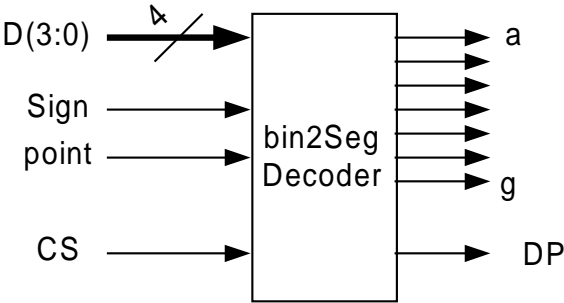


Figure 6

2. Create a new schematic file, and design a circuit for the new seven segment decoder. You can use your previous design generated in VHDL and build on it. If you want, you can try to do this in VHDL instead (you can recycle/modify much of your previous code).
3. Check your schematic, and save it.
4. Create a test bench waveform and verify your design with different input combinations.
5. Create a symbol for your design and save it under bin2seg.
6. Next you want to download your design into the board and verify its behavior.
7. Create a new schematic similar to Figure 7.
8. Check your schematic and save the file.
9. Create a user constraints file for your new schematic as did in the tutorial.

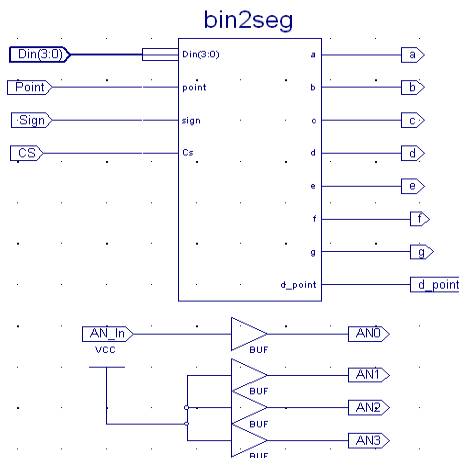


Figure 7

10. Your constraint file should be similar to Table 3 (the pin locations are given the User Manual).
11. Save the users constraint file and generate the programming file.
12. Download your design into the board.
13. Make sure you understand the concept of this file.
14. Verify your design by testing all possible input combinations.

15. Show your working circuit to the TA.

```
NET "a" LOC = "E14";
NET "b" LOC = "G13";
NET "c" LOC = "N15";
NET "d" LOC = "P15";
NET "e" LOC = "R16";
NET "f" LOC = "F13";
NET "g" LOC = "N16";
NET "d_point" LOC = "P16";

NET "AN0" LOC = "D14";
NET "AN1" LOC = "G14";
NET "AN2" LOC = "F14";
NET "AN3" LOC = "E13";

NET "Din<0>" LOC = "F12";
NET "Din<1>" LOC = "G12";
NET "Din<2>" LOC = "H14";
NET "Din<3>" LOC = "H13";
```

Table 3

Multiple Digits Display

In order to display multiple digits on the board you need to implement a time multiplexing scheme among the control signals AN3-AN0. In this section you will design the circuit for doing that.

1. Read chapter 3 of [1].

Clock

You need to have a clock in your design. Spartan3 board has an internal 50 MHz clock which is very fast for our purpose. You are going to down convert that for this project.

2. Download the files 'clk_convrt.vhd' and 'clk_convrt.sym' from the course website and add them to your project by right-clicking on the project name and add source.

3. This block (clk_convrt) down converts the 50 MHz clock to three different clock signals (1 Hz, 10 HZ, and 10 KHz) (Figure 8).

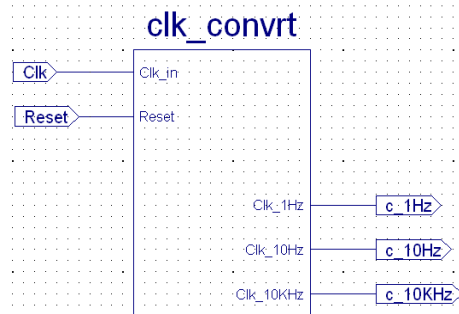


Figure 8

4. Here you will create a simple schematic to understand the behavior of this block.
5. Create a new schematic file similar to Figure 8.
6. Create a user constraints file for this schematic similar to Table 4
7. Generate the bitstream file and download it into the board. You should see LD0, LD1, and LD2 blinking with the frequency of 1, 10, and 10 KHz respectively.

```

NET "Clk" LOC = "T9";
NET "c_1Hz" LOC = "K12";
NET "c_10Hz" LOC = "P14";
NET "c_10KHz" LOC = "L12";

```

Table 4

Seven Segment Driver Unit

Using the clock signal you need to develop a system that time multiplexes among signals AN3 – AN0 as depicted in Figure 9. Figure 10 shows a block, which provides the desired signals. Here we describe all the related signals.

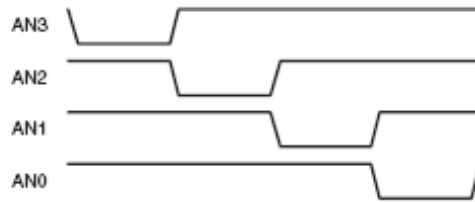


Figure 9

Clk: Clock for this component typically 1 KHz is a reasonable choice.

Reset: Resets this block

E: Enables this component

$O_1 O_0$: binary representation of the activated output.

During each clock period only one of the outputs ($AN3-AN0$) is activated and $O_1 O_0$ corresponds to the active signal. So, after $O_1 O_0 = 11$ is generated on the fourth clock cycle, it should go back to 00.

8. Design a circuit that generates these outputs. Your design should include a sequencer. You can implement the sequencer by using two flip flops and some combinational gates, or alternatively with a two bit binary counter (For example CB2RE).
9. Check the schematic of your design and save it.
10. Create a test bench waveform and test your design with simulation (Figure 11).
11. Create a symbol for your design.

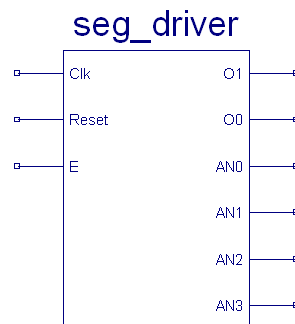


Figure 10



Figure 11

12. Create a schematic file similar to Figure 12 for testing most of the components you have designed so far.
13. Create a user constraints file similar to Table (again, use the Spartan-3 User Manual).
14. Generate a programming file for this design and describe the behavior of this system. What will happen if you use the 10 KHz output of the clock converter component instead of the 1 Hz output?

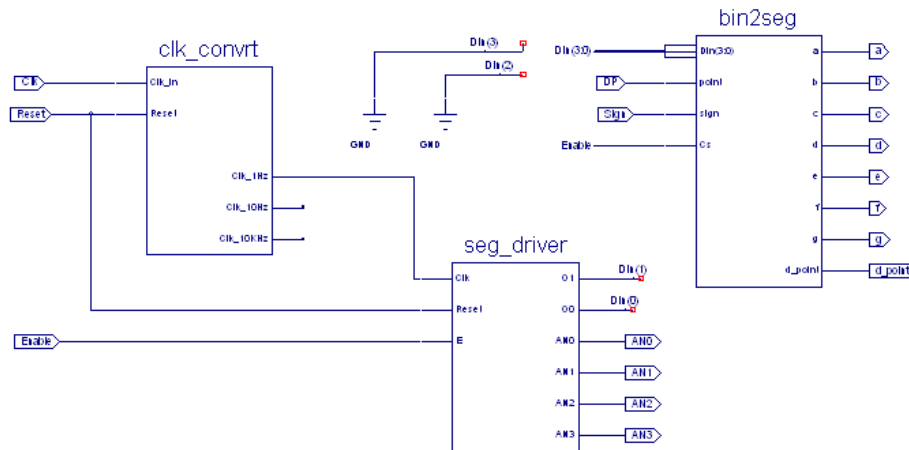


Figure 12

NET "Clk" LOC = "T9";
NET "Reset" LOC = "L14";
NET "Enable" LOC = "L13";
NET "Sign" LOC = "k13";
NET "DP" LOC = "K14";
NET "a" LOC = "E14";
NET "b" LOC = "G13";
NET "c" LOC = "N15";
NET "d" LOC = "P15";

Table 5

Input Multiplexing

If you take a look at Figure 3 again, you can realize two distinct multiplexing in that design:

Input Multiplexing: At the input of the system at each moment we want to display one of the 4-bit numbers (W , X , Y , Z) and its associated sign and decimal points.

Output Multiplexing: At the output of the system we have four seven segment displays and at each time interval we want to activate one of those displays and show the corresponding input number.

In the previous section you have designed and tested the output multiplexer. Now you want to design the input multiplexing unit.

15. Create a new Schematic file for input multiplexing.
16. Figure 13 shows the basic function for this unit. Four 4-bit inputs (W , X , Y , Z) and their associated signs and decimal points are multiplexed to the outputs based on control inputs $A1$ $A0$. Design a circuit for this purpose. You can use a single bit, four to one multiplexer unit (M4_1E) provided by the library.
17. Check your schematic and save your file.
18. Create a test bench waveform and test the functionality of your design.
19. Create a symbol for this component.

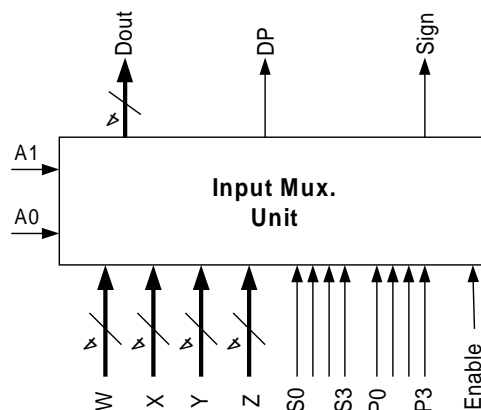


Figure 13

Putting the Generic Four Digit Display Unit together

20. Create a new schematic file and put the circuit in Figure 14 together.
21. Explain the behavior of this system.
22. Save your design.
23. Create a test bench waveform and make sure your circuit is functional.
24. Create a symbol for your design.

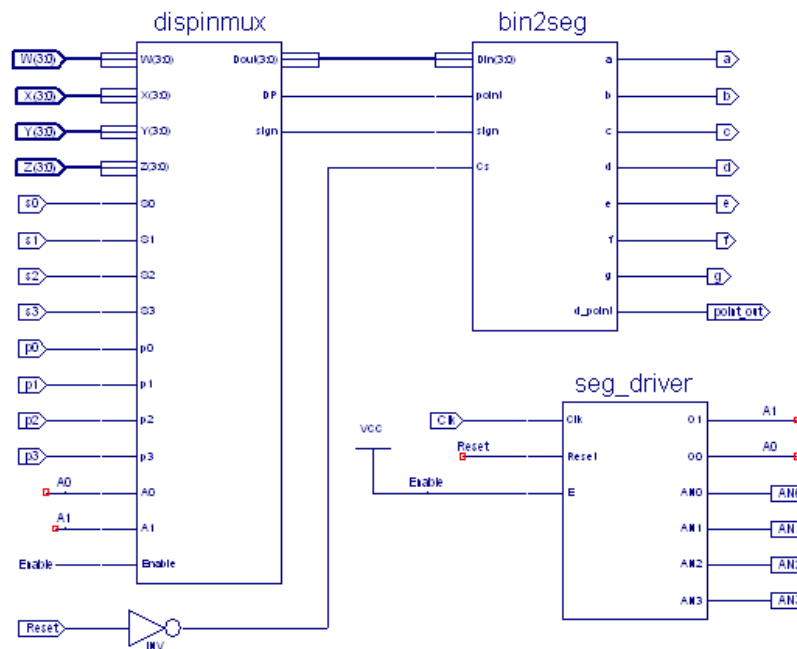


Figure 14

25. Now create a schematic file similar to Figure 15.
26. Check the schematic and save the file.
27. Create a user constraints file that uses (SW7-SW4), (SW3, SW0) as the first and second number respectively. Assign BTN3, BTN2, BTN1, and BTN0 to S2, P2, S1, and P1 respectively.
28. Generate the programming file, download the file into the board, and make sure that your circuit works properly.
29. Show your circuit to the TA.

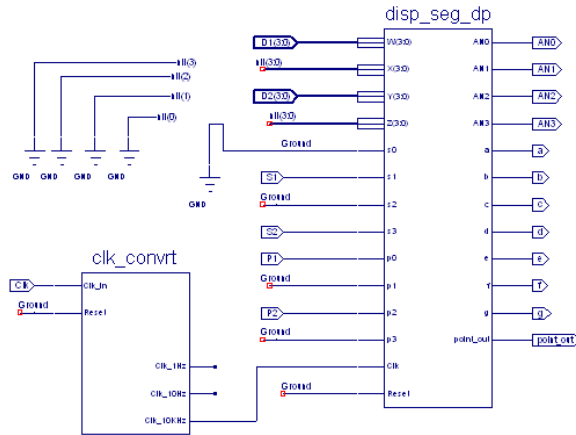


Figure 15

Four Bit Multiplication Display

In this section you will use the multiplier unit that you had created during the first tutorial and the circuit you have just designed to display the result of a four bit multiplication.

30. Create a new schematic file.
31. Design a circuit that multiplies two 4-bit unsigned numbers and displays the result as a hexadecimal number on the seven segment display. (Figure 16)
32. Save your design file.
33. Create a user constraints file assign (SW7-SW4) and (SW3-SW0) to first and second number respectively.
34. Generate the programming file and download that into the board and verify your design.
35. Show your circuit to the TA.

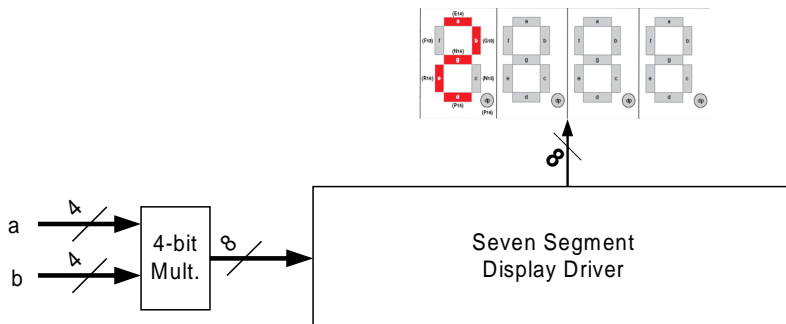


Figure 16

References

- [1] Xilinx, 'Spartan-3 Starter Kit Board User Guide'
- [2] ECE3801 Tutorial - 1

Seven Segment Display Sign-Off Sheet

Make sure lab instructors initialize each part. Attach this page to the end of your report.

Your Name:

Lab Partner:

Date Performed:

Demonstrated that the circuit works correctly:

- One display on seven-segment display works _____
- Multiple digits on the seven-segment display works _____
- Multiplication works _____