SFWR ENG 2DA4 Lab 2

First lab Week of: Sep 30, 2019
Prep Due week of: (8:40/14:40), Oct. 7, 2019
Demo Due Week of: (11:20/17:20), Oct. 7, 2019
Assigment due in class: 17:40, Thu Oct. 10, 2019

Announcements:

Note 1: all lab due dates/times are for your scheduled lab in the indicated week.

As you will be doing a fair number of k-maps in this lab, I recommend dividing the work up equally between each partner. Make sure you each do an equal share for both circuits, as well as both types of k-maps (s-of-p and p-of-s).

For this and every lab: after you have demonstrated your lab to a TA , you must e-mail (as per instructions in lab 1) the final versions of your Verilog files (the ones demoed to the TA) as attachements to the following e-mail address: `rlta1@cas.mcmaster.ca`

Part 1: Assignment

Note: You must put your lab section on the top of your assignment as they will be handed back during your lab.

Note: It is unlikely that Assign 2 will be marked and returned before the midterm. I recommend that you make a copy of your assignment before handing it in so you can check it against the solns as part of your midterm prep.

The following are relevant (related to this lab and the midterm!) textbook questions to be handed in 17:40, Thu Oct. 10, 2019:

1) For Part 4.i below, find the minimal s-of-p and p-of-s equation for LED segment "0." (note, only characters 1, 4, b, and d will not have this segment lit). Use k-maps. Which is lowest cost?

2) Derive a minimum-cost circuit that implements the function $f(x_1, x_2, x_3, x_4) = \Sigma m(4, 7, 8, 11) + D(12, 15)$.

3-6) Appendix B # 1 , 3, 36, 47 (Label as per the instructions in question B.44. You do not have to label the outputs of the LUT with the logic expressions. (hint. consider starting by factoring out common terms)).

Part 2: More Complex Logic Design: Seven-Segment Displays

1. Purpose

   The purpose of this lab is to build several more complex logic circuits and to gain increased familiarity with the Altera Quartus Prime software and the DE1-SoC boards.

2. Background

A seven-segment display is often used on computers, watches, VCRs and many electronic devices to display numbers and some characters. It consists of seven independent lights (light emitting diodes (LEDs)) in an "8" configuration as shown below in Figure 1. By turning on different segments, you can display different numbers and some letters.

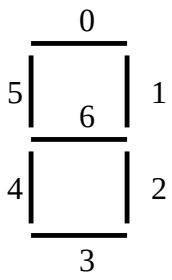

Figure 1: 7-segment Display

The Altera DE1-SoC board has six seven-segment displays (HEX0-HEX5 with HEX5 the one closest to the edge of the board) attached directly to the pins of the Cyclone V FPGA. Figure 1 shows the naming of each segment. See Table 3-9 on Page 27 of the *DE1-SoC User Manual*, available in PDF from the URL: http://www.cas.mcmaster.ca/~leduc/slides2d04/labs/DE1-SoC_User_manual_ref.pdf for the connections between segments and pin assignments.

NOTE: For these displays, to turn a segment on, you must drive the associated pin with a "0". To turn it off, drive it with a "1".

For the Karnaugh-map method, see sections 2.11-2.14 of text as well as lecture notes. See Appendix A, Section *A.12,* for details on how to use subcircuits.

3. Preparation

Hand in as your prep the code and simulation output for all circuits in step 4 below. Hand in the k-maps for segments "3" and "5" for both parts of step 4. You should arrive at the lab with your preparation ready to hand in. Do not try to print it out at the start of the lab period.

4. Problems

You are to create two logic circuits that would allow you to drive one of the seven-segment displays on an Altera DE1-SoC board.

i) Design a subcircuit that takes a four bit (x_3, x_2, x_1, x_0) input and drives a seven segment display as described in the table below. Note that for the letters, some are capitalized and some are not. (The reason is that a capital B, for example, would come out the same as an 8 on a seven-segment display, so we will display a lower case b instead).

Determine the equations for the seven-segment display segments, and minimize them using the Karnaugh-map method. Because there are so many functions to minimize, you are not required to do the K-map for both sum-of-products and product-of-sums. Just choose one for each output to do. Make sure you do about 50% of each type (the same goes for the "`name.v`" circuit below).

$x_3x_2x_1x_0$	Display
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

$x_3x_2x_1x_0$	Display
1000	8
1001	9
1010	A
1011	b
1100	C
1101	d
1110	E
1111	F

Create a Verilog file called `hexdisp.v` containing the logic function for each segment as a boolean equation (with AND, OR, NOT, etc.). Simulate and test your equations.

ii) Design a similar subcircuit to the one given in part 1, except that the four inputs will be used to generate the letters of one of your last names. For example, if your last name is Parnas, you could use code 0000 to display a P, 0001 to display an A, 0010 to display a R (or something as close to an R as you can get).

Note that you can choose any code you wish for each letter . Notice also that you do not need to create two codes for a letter that appears twice in your name. For example, the letter A appears twice in Suvanthingham, but you only need to produce one code for the letter A - the intent is that you will be able to spell out your last name on the seven-segment display by entering the codes for the letters one at a time.

Fairness alert: If your last name is longer than 9 unique letters, you need only do 9 unique characters of it. If your last name is shorter than 9 characters, you must add more letters from your first name until you have coded at least 9 unique characters.

Since you will only need to do 9 codes, you can leverage the don't cares that will be available in the truth tables, when using the Karnaugh map method.

Create a Verilog file called `name.v` containing your design, then simulate and test your equations.

iii) Combine both of these subcircuits in a single project file call `lab2demo.v`. Compile this design for simulation using the Cyclone V FPGA (5CSEMA5F31C6) device and use the following pin assignments:

Connect the inputs for the component based on the subcircuit from (i) to toggle switches SW[9], SW[8], SW[7] and SW[5] and use this component's outputs to drive HEX5 on the Altera board. Similarly for the component based on the subcircuit in (ii), connect the inputs to SW[4], SW[3], SW[2]and SW[1] and connect the outputs to HEX4 on the Altera board.

Download this file to the DE1-SoC board and demonstrate it to the TA.

The last two items will be needed in the next lab. Download and demonstrate them to a TA.

iv) Build and simulate an edge triggered D flip-flop as described in the Verilog Reference guide, Section A.14.2 (see Figure A.27) of Appendix A of the text. Use a pushbutton (KEY[0]-KEY[3] - see page 25 of DE1-SoC user manual) as clock, and toggle switches as inputs. Use a segment of the LED display for output. **Please note**, the push button generates a low logic level when it is pressed. When it is not pressed, it generates a high logic level.

Please also note that for the current version of Quartus Prime and for the DE1-SoC board, timing simulation will give the same result as functional simulation.

v) In the same fashion, build and simulate a three-bit D register with an asynchronous reset (clear) signal as described in Section A.14.3 and Section A.14.4 (see Figure A.30).