CSC258H Lab 2: Circuit Creation

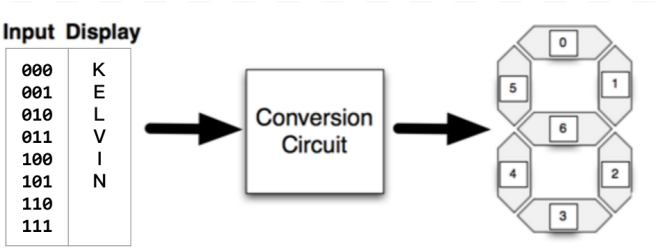
1 Introduction

This week, we'll practice designing circuits made of logic gates and will connect these circuits to switches and a 7-segment display on the DE-2. You'll need to be familiar with the software tools we used last week; refer back to last week's lab if you need to refresh your memory.

Before the lab, read through the lab handout so that you are familiar with the procedure, and then complete the Karnaugh-maps for the circuits described in the next section. The TA will be available to answer questions during lab or to help you if you get stuck, but you must have your circuit expressions by the time lab starts to have time to complete the lab.

2 Driving a 7-Segment Display

A 7-segment display takes a 7-bit number as input. The figure below is of a 7 segment display; the number on each segment is the index of the bit that determines whether it is lit ("on") or dark ("off").



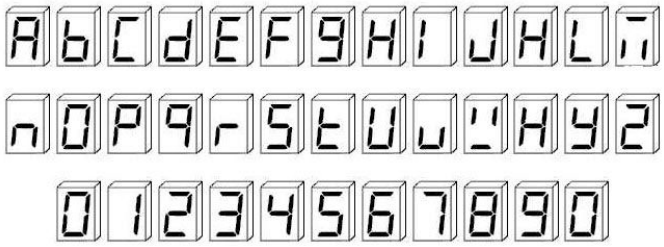
We would like a circuit that can be used to greet your TA on the 7 segment display. Since the lengths of messages will be 8 letters (including the space), the circuit needs three input bits. The figure above also shows an example how the inputs could match to the letters. In the example, the input `000` should display the letter `K`. Therefore, the circuit you will design should produce the 7-bit output `0001001`. (7-seg LEDs use negative logic, where *0* means "on".) You may, if you wish, rearrange the assignment of letters to inputs to create a simpler circuit.

To repeat: 7-seg LEDs use *negative logic*, so *0* means "ON" and *1* means "OFF."

Below are the messages that you need to show to your TA for each lab section. Note that you must also properly display the space character, i.e., all segments should be off when you switch to the space.

- PRA0101, 0102: HEY NICK
- PRA0103, 0104: SUP ALEX
- PRA0106: HI JACOB
- PRA0107, 0108: RAJDEEP
- PRA0109: HEY CARL

Below is a picture that shows how each English letter should be displayed on the 7-segment display.



First, **before the lab**, generate the expressions for the seven optimized circuits – one for each output (segment on the display). To do so, you'll need to create a truth-table with 3 input bits and 7 output bits. Then, you should use seven K-maps – one for each output – to generate the optimal expression that expresses it.

Once you arrive in lab, compare your final circuit expressions with someone else in class. If they differ, determine why and copy down the correct expressions. Then, create a new Quartus project, create a new schematic in the project, and implement your fully optimized circuits. Make sure to create a test vector to verify that the circuit works as you expect.

Once your simulation results convince you that your design is working correctly, load your circuit onto the FPGA. Connect the inputs to switches 0-2 (PIN_N25, PIN_N26, PIN_P25) and the outputs to 7-segment digit 0 (PIN_AF10, PIN_AB12, PIN_AC12, PIN_AD11, PIN_AE11, PIN_V14, PIN_V13).

No, I've no idea where they came up with these pin names. However, there's a faster way to do it: importing pin assignments. Carl Marquez has created instructions that are linked on the course webpage.

3 Summary of TODOs

Below is a short summary of the steps to be completed for this lab:

1. Before the lab, generate the optimized circuit expressions using K-maps.
2. In the lab, implement the circuit using Quartus.
3. Simulate the circuit. Once you are convinced it is working correctly, show the waveforms to your TA and explain why it demonstrates your circuit is correct.
4. (Optional) Load the circuit to a DE-2 board and greet your TA with it.

Evaluation (3 marks in total): 1 mark for bringing k-maps to class; 1 mark for building the circuit; 1 mark for demonstrating a test waveform.