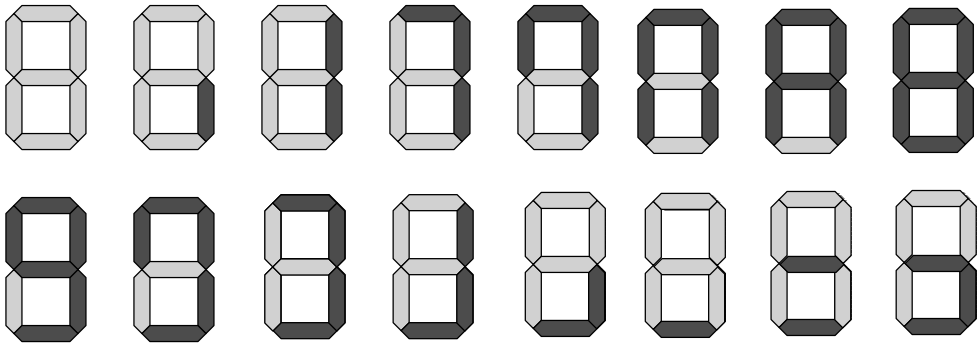
University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Science

EECS 150 R. H. Katz
Spring 2004

Problem Set # 3 (Assigned 5 February, Due 13 February)

1. Design a combinational logic subsystem with three inputs, I_3, I_2, I_1, and two outputs, O_1, O_0, that behaves as follows. The outputs indicate the lowest index of the inputs that is driven high. For example, if I_3 is 0, I_2 is 1, I_1 is 1, then O_1, O_0 would be 01 (i.e., I_1 is the lowest input set to 1).
   (a) Specify the function by filling out a complete truth table.
   (b) Write a specification of this function in Verilog.
   (c) Develop the minimized gate-level implementation using the K-map method.
   (d) Implement this subsystem using 2 x 4:1 multiplexers.
   (e) If implemented using a ROM, what size ROM is required? Why?
   (f) Compare your solutions to parts (c) and (d). Which is simpler and why (i.e., what criteria are using to measure complexity)?
2. The Spirit Rover has discovered that the Martians use a base 16 number system. However, the digits are quite different than the ones to which we are accustomed. The first row below is 0 through 7, and the second row is 8 through F. Your task is to design a combinational logic subsystem to decode a hexadecimal digit in the range of 0 (0000) through F (1111) to drive a seven-segment display for the Martian version of the hexadecimal digits (0-7 in the top row, 8-F in the bottom row).



   (a) Specify the function by filling out a complete truth table for each of the seven segment drivers.
   (b) Write each as a Verilog specification.
   (c) Develop the minimized gate-level implementation using the K-map method, minimizing each K-map independently.
   (d) Repeat (c), but this time, minimize so as to exploit shared product terms wherever possible, as though the implementation target is a PLA.
   (e) How does the complexity of your answers to (c) and (d) compare? Which is simpler and why? What criteria are you using to measure the complexity of these two different implementations?
3. Your task is to design a combinational logic subsystem as part of a larger digital system that makes change from dollar bills. There are three binary counters that keep track of the number of each kind of coin in each reservoir (these are outside your subsystem, and you can treat them as black boxes—but you will need to use their outputs as inputs to the subsystem you are designing). For simplicity, you may assume that the reservoirs contain at most sixteen quarters, sixteen dimes, and sixteen nickels. Your subsystem should give quarters before dimes, and dimes before nickels. For example, if there are at least four quarter left, give all four. If there are three quarters left, give three and two dimes and one nickel (assuming that number of coins are available!), and so on. There is definitely a possibility that

no change can be given because the reservoirs are (almost) exhausted of coins. In such a case, a “no change” sign should be illuminated.

(a) This is a complicated system! To make sure that you understand how it is supposed to function, write down all of the different ways of making change of a dollar given the constraints inherent in the specification above. How many unique ways are there?

(b) Identify your inputs and outputs, and draw a block diagram that shows how the change making subsystem interconnects with the coin counters, the “no change” light, etc.

(c) If you were to implement this subsystem with a ROM, how many words at what width would the ROM have to be? Explain your answer.

(d) Write a Verilog behavioral description of this system.