EECS 140/141 Introduction to Digital Logic Design

Spring Semester 2020

Assignment **#7** Due **17 March 2020**

Reading: Sections 1.6, 5.1, 5.2, and 5.3.1 in Brown/Vranesic

All logic networks on this (and every other assignment) *must* be drawn using a logic template. Points will be deducted for failure to do this!

1. If you are in EECS 140, use the function given in problem 4.20 (p. 242) for this problem. If you are in EECS 141, use the function given in problem 4.22 (p. 242).
    a. Find the min-cost SoP synthesis. Don't forget that part of this process is to list all the Prime Implicants (PIs) and then identify which are Essential (EPIs).
    b. Use factoring to produce a multi-level synthesis that is different from the min-cost SoP synthesis. This multi-level synthesis may be more or less costly than the min-cost SoP synthesis.
    c. Use functional decomposition on the factored synthesis to simplify the circuit further. In particular, look for some expression in your factored synthesis that could be the complement of some other expression in that synthesis, then use that expression and its complement in the synthesis.
    d. Draw the logic diagram corresponding to your *final* synthesis (from the previous part) using AND/OR/NOT gates.
    e. Give the cost, the number of levels, and the maximum fan-in for each syntheses from parts (a) through (c).
2. Consider the logic circuit shown in Figure P4.2 on p. 245, and in particular, the function g implemented by this circuit. Label the internal points of the g implementation P_1 to P_5 from bottom (output of 2-input NOR) to top (output of 4-input AND).
    a. In a fashion similar to Example 4.10 on p. 202, give the logic expression for each of P_1 to P_5 and then give an expression for g in terms of these internal points.
    b. Use the logic expressions for P_1 to P_5 and Boolean Algebra properties to write an expression for g in Sum of Products form, where each product term contains only literals.
    c. Use the SoP expression from the previous part to draw the K-map for this function.
    d. From the K-map, give an expression for g as a sum of minterms. Use the $g = \Sigma m(\ldots)$ form.
3. Represent the following (unsigned) binary integers in the following 3 forms: (i) decimal, (ii) (unsigned) octal, and (iii) (unsigned) hexadecimal. Do each conversion directly from the binary form to the other form. You *must* show all the steps in converting from the binary form to each other form. Note that the spaces are included for clarity only, much like commas are used in decimal (such as 100,000).
    a. 0000 0000

b. 0101 0011

c. 1111 0000

4. Represent the following decimal integers in (unsigned) binary, octal, and hexadecimal forms. Do each conversion directly from the decimal form to the other form. You *must* show all the steps in converting from the decimal form to each other form.
   a. 85 b. 32 c. 211

5. Represent the following numbers in (unsigned) binary form and in decimal form. Show the conversion steps.

   a. $(BE)_{16}$

   b. $(85)_{16}$

   c. $(17)_{16}$

   d. $(17)_8$

   e. $(32)_8$

   f. $(211)_8$

6. What are the smallest and largest decimal integers that can represented in an 8-bit (unsigned) binary form? Repeat for a 16-bit (unsigned) binary form?

7. Other characters besides numbers can be represented in binary. For example, how many bits are required for a binary representation of:

   a. The 26 upper-case letters in the English alphabet?

   b. All of the upper-case and lower-case letters in the English alphabet?

   c. All of the upper-case and lower-case letters in the English alphabet, the 10 numerals, and 15 punctuation marks?

8. The binary form can be used to represent all sorts of things. For example, suppose four judges can vote either yes or no on an issue.

   a. Find the *minimum* number of bits required to represent the collective vote, that is, to record all of the individual votes.

   b. Repeat for the situation in which each judge can vote yes, no, or abstain.

9. Repeat the entire last problem (both parts) under the constraint that each judge is assigned a *separate* binary field in the representation. That is, each judge's vote is represented as a binary number, and then those 4 binary representations are concatenated (placed side-by-side) to form the representation of the collective vote.