Practice Midterm exam

ECE 303: Advanced Digital Logic Design
Suggested time: 75 minutes

You may not refer to your book or notes during this exam.

Please look over the whole exam before starting work. If there is time pressure, it is better to almost finish 100% of the problems than to totally finish 20% of one problem and not start the rest. Making a mistake early in the design process can cause huge problems later on. Although the graders will give partial credit, making a mistake early in a multi-part question can also cause problems later on.

Do all the required problems and one of the two optional problems.

1. Manual minimization **(20 points, required)**

   Consider the following equation

   $$f(a,b,c,d) = \sum(0,1,2,5,7,10,12,13,15)$$

   (a) Write out an expression for f as a sum of minterms. How many literals does this expression have?

   (b) In one sentence, explain why is literal count is useful.

   (c) Use the Quine-McCluskey method to find a minimal two-level SOP expression for f. I suggest starting this sub-problem on a left page in your solutions book.

   (d) How many literals does this expression have?

   (e) Is the Quine-McCluskey solution guaranteed to be optimal?

   (f) What is the proper name for the second phase of the Quine-McCluskey method?

   (g) Please comment on the relationship between input size and worst-case execution time for all existing algorithms that optimally solve the problem encountered in the second phase of the Quine-McCluskey method.

   (h) Starting from the Quine-McCluskey simplified SOP form, show the PLA implementation of the function. You may use the short-hand or standard representation to show AND-plane gate inputs.

2. Advanced manual minimization and implementation **(20 points, optional)**

   (a) Starting from the Quine-McCluskey simplified SOP form derived in question 1, use iterated kernel division to find a good multi-level implementation for f. Each time you select a kernel, chose one that will result in a minimal number of literals in the resulting expression. Ties can be broken arbitrarily.

   (b) How many literals does this expression have?

   (c) Is the multi-level solution guaranteed to be optimal?

   (d) Draw an AND/OR gate-level diagram for the multi-level solution.

   (e) How many levels of logic exist in this diagram?

   (f) Derive a NAND/NOR gate-level diagram using bubble pushing. Try to be neat when showing your steps.

   (g) How many configurable logic blocks would this function require to implement in a Xilinx FPGA?

3. Heuristic two-level minimization **(20 points, required)**

   Consider the cover composed of the following cubes: $\overline{c}\,\overline{d}$, $\overline{a}$, bd, $a\overline{c}$, and cd

   (a) Explain, in one sentence, what Espresso's *Irredundant Cover* move does.

   (b) Irredundant cover needs to determine which cubes are relatively essential, partially redundant, and totally redundant. Describe a fast method (at least as fast as that used in Espresso) of checking to see whether a given cube is relatively essential.

   (c) Apply this method to cube $X1X1$.

   (d) Apply this method to cube cd.

   (e) Given a cover, and a list of the relatively essential cubes in the cover, explain how to determine which other cubes are totally redundant. For the sake of simplicity, assume that the cover contains no don't-care cubes.

4. Hierarchical composition **(20 points, required)**

   You have a technology library containing only 2:1 multiplexors.

   (a) Give a block diagram of a circuit composed of gates in the technology library that is capable of realizing an arbitrary function of three variables by changing the variables and data on the block's inputs.

   (b) How many 2:1 multiplexors are required to implement any function of two (yes, two) variables?

   (c) Let $M(n)$ be the number of 2:1 multiplexors required to implement a function of n variables. Assuming termination of the $M(2)$ case you answered in the previous subquestion, write $M(n)$ as a function of $M(n-1)$.

   (d) What function best *approximates* $M(n)$?

      i. $M(n) \approx 2$

      ii. $M(n) \approx 2n$

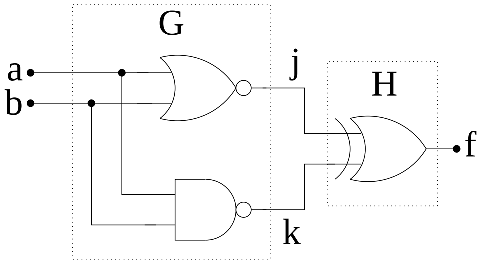
      iii. $M(n) \approx n^2$

      iv. $M(n) \approx 2^n$

      v. $M(n) \approx n^n$

5. Don't-cares **(20 points, required)**

   Consider the following circuit



   (a) Give a truth table and minimal (you may choose the method of minimization but your solution should be optimal) SOP expressions for $j(a,b)$ and $k(a,b)$. Don't use don't-cares.

   (b) Give a truth table and minimal SOP expression for $f(j,k)$. Don't use don't-cares.

   (c) Write $f(a,b)$ in multilevel, factored, form. Don't use don't-cares. How many literals are required?

   (d) Define "observability don't-care".

   (e) Define "satisfiability don't-care".

   (f) Give the truth tables for G and H, using don't-care's when appropriate. Label each don't-care with its type.

   (g) Find minimal SOP expressions for $j(a,b), k(a,b)$, and $f(j,k)$.

   (h) Write $f(a,b)$ in multilevel, factored, form. How many literals are required?

   (i) Give a truth table and minimal SOP expression for $f(a,b)$. How many literals are required? Don't use don't-cares.

6. CMOS design **(20 points, optional)**

   Consider the following function, specified by its *minterms*

$$f(a,b,c,d) = \sum(0,2,4,7,9,11,14) + d(1,5,12,13)$$

   (a) Use a K-map to get a minimal POS (yes, *POS*) expression for f.

   (b) Draw a circuit diagram based on this POS expression.

   (c) Assuming access to the complemented and uncomplemented literals for each variable, manipulate the circuit diagram until it can easily and efficiently be implemented in CMOS.

   (d) For each type of gate used in the circuit diagram, draw a transistor-level diagram composed of nets, NMOS transistors, and PMOS transistors. You don't need to redo the whole diagram at transistor level. However, you do need to show that you understand how to build each gate you used, starting from transistors.