hybrid jump counter

Unger, *The Essence of Logic Circuits*, Prentice-Hall, Englewood Cliffs, NJ, 1989; and C. H. Roth, Jr., *Fundamentals of Logic Design*, West Publishing Co., St. Paul, MN, 1985.

Exercises

Nothing should be made by man's labor which is not worth making, or which must be made by labor degrading to the makers.

—William Morris

2.1 *(Gate Logic)* Draw schematics for the following functions in terms of AND, OR, and inverter gates.

a. $X\,(Y+Z)$

b. $X\,Y+X\,Z$

c. $\overline{X(Y+Z)}$

d. $\overline{X}+\overline{YZ}$

e. $W\,(X+Y\,Z)$

2.2 *(Gate Logic)* Draw the schematics for the following functions using NOR gates and inverters only:

a. $\overline{[\overline{X}+(\overline{Y+Z})]}$

b. $\overline{[\,\overline{(\overline{X}+\overline{Y})}+\overline{(\overline{X}+\overline{Z})}\,]}$

2.3 *(Gate Logic)* Draw the schematics for the following functions using NAND gates and inverters only:

a. $\overline{[X(\overline{\overline{Y}\,\overline{Z}})]}$

b. $XY+XZ$

2.4 *(Gate Logic)* Draw switch networks for the XOR and XNOR Boolean operators.

2.5 *(Gate Logic)* Design a hall light circuit to the following specification. There is a switch at either end of a hall that controls a single light. If the light is off, changing the position of either switch causes the light to turn on. Similarly, if the light is on, changing the position of either switch causes the light to turn off. Write your assumptions, derive a truth table, and describe how to implement this function in terms of logic gates or switching networks.

2.6 *(Laws and Theorems of Boolean Algebra)* Use switching diagrams to demonstrate the validity of the following simplification theorems:

a. $X+X\,Y=X$

b. $(X+\overline{Y})\,Y = XY$

c. $\overline{(X+Y)} = \overline{X}\bullet\overline{Y}$

2.7 *(Laws and Theorems of Boolean Algebra)* Prove the following simplification theorems using the first eight laws of Boolean algebra:

a. $(X+Y)\,(X+\overline{Y}) = X$

b. $X\,(X+Y)=X$

c. $(X+\overline{Y})\,Y = XY$

d. $(X+Y)\,(\overline{X}+Z) = XZ+\overline{X}Y$

2.8 *(Laws and Theorems of Boolean Algebra)* Verify that

a. OR and AND are duals of each other.

b. NOR and NAND are duals of each other.

c. XNOR and XOR are duals of each other.

d. XNOR is the complement of XOR:

$$\overline{(X\overline{Y}+\overline{X}Y)} = \overline{X}\,\overline{Y}+XY$$

2.9 *(Laws and Theorems of Boolean Algebra)* Prove, using truth tables, that

$$XY+YZ+\overline{X}Z = XY+\overline{X}Z$$

2.10 *(Laws and Theorems of Boolean Algebra)* Use DeMorgan's theorem to compute the complement of the following Boolean expressions:

a. $A\,(B+CD)$

b. $ABC+B\,(\overline{C}+\overline{D})$

c. $\overline{X}+\overline{Y}$

d. $X+Y\overline{Z}$

e. $(X+Y)\,Z$

f. $X+\overline{YZ}$

g. $X(Y+Z\overline{W}+\overline{V}S)$

2.11 *(Laws and Theorems of Boolean Algebra)* Form the complement of the following functions:

a. $f(A,B,C,D)=[A+\overline{BCD}]\;[\overline{AD}+B\,(\overline{C}+A)]$

b. $f(A,B,C,D)=A\overline{B}C+(\overline{A}+B+D)\;(AB\overline{D}+\overline{B})$

2.12 *(Laws and Theorems of Boolean Algebra)* Using Boolean algebra, verify that the schematic of Figure Ex2.12 implements an XOR function.

hybrid jump counter

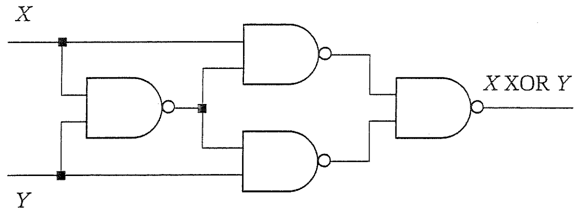

Figure Ex2.12 XOR implemented by NAND gates.

2.13 *(Boolean Simplification)* Simplify the following functions using the theorems of Boolean algebra. Write the particular law or theorem you are using in each step. For each simplified function you derive, how many literals does it have?

a. $f(X,Y) = XY + X\overline{Y}$

b. $f(X,Y) = (X+Y)\ (X+\overline{Y})$

c. $f(X,Y,Z) = Y\overline{Z} + \overline{X}YZ + XYZ$

d. $f(X,Y,Z) = (X+Y)\ (\overline{X}+Y+Z)\ (\overline{X}+Y+\overline{Z})$

e. $f(W,X,Y,Z) = X + XYZ + \overline{X}YZ + \overline{X}Y + WX + \overline{W}X$

2.14 *(Boolean Simplification)* Consider the function $f(A,B,C,D) = (AD + \overline{A}C)\ [\overline{B}(C + B\overline{D})]$.

a. Draw its schematic using AND, OR, and inverter gates.

b. Using Boolean algebra, put the function into its minimized form and draw the resulting schematic.

2.15 *(Canonical Forms)* Consider the function $f(A,B,C,D) = \Sigma\, m(0,1,2,7,8,9,10,15)$.

a. Write this as a Boolean expression in canonical minterm form.

b. Rewrite the expression in canonical maxterm form.

c. Write the complement of f in "little m" notation and as a canonical minterm expression.

d. Write the complement of f in "big M" notation and as a canonical maxterm expression.

2.16 *(Canonical Forms and Boolean Simplification)* Given the following function in product of sums form, not necessarily minimized:

$$F(W,X,Y,Z) = (W + \overline{X} + \overline{Y})\ (\overline{W} + \overline{Z})\ (W + Y)$$

a. Express the function in the canonical sum of products form. Use "little m" notation.

b. Reexpress the function in minimized sum of products form.

c. Express $\overline{F}$ in minimized sum of products form.

d. Reexpress $\overline{F}$ in minimized product of sums form.

2.17 *(Boolean Simplification)* Using K-maps, find the following:

a. Minimum sum of products form for the function and its complement given in Exercise 2.15.

b. Minimum product of sums form for the function and its complement given in Exercise 2.15.

2.18 *(Boolean Simplification)* Use Karnaugh maps (K-maps) to simplify the following functions in sum of products form. How many literals appear in your minimized solutions?

a. $f(X,Y,Z) = \Pi\ M(0,1,6,7)$

b. $f(W,X,Y,Z) = \Pi\ M(1,3,7,9,11,15)$

c. $f(V,W,X,Y,Z) = \Pi\ M(0,4,18,19,22,23,25,29)$

d. $f(A,B,C,D) = \Sigma\ m(0,2,4,6)$

e. $f(A,B,C,D) = \Sigma\ m(0,1,4,5,12,13)$

f. $f(A,B,C,D,E) = \Sigma\ m(0,4,18,19,22,23,25,29)$

g. $f(A,B,C,D,E,F) = \Sigma\ m(3,7,12,14,15,19,23,27,28,29,31,35,39,44,45,46,48,49,50,52,53,55,56,57,59)$.

2.19 *(Boolean Simplification)* Determine the minimized realization of the following functions in the sum of products form:

a. $f(W,X,Y,Z) = \Sigma\ m(0,2,8,9) + \Sigma\ d(1,3)$

b. $f(W,X,Y,Z) = \Sigma\ m(1,7,11,13) + \Sigma\ d(0,5,10,15)$

c. $f(V,W,X,Y,Z) = \Sigma\ m(2,8,9,10,13,15,16,18,19,23) + \Sigma\ d(3,11,17,22)$

d. $f(V,W,X,Y,Z) = \Sigma\ m(0,1,2,9,13,16,18,24,25) + \Sigma\ d(8,10,17,19)$

2.20 *(Boolean Simplification)* Use the K-map method to find the minimized product of sums expressions for the following Boolean functions:

a. $f(A,B,C) = A \oplus B \oplus C$

b. $f(A,B,C) = AB + BC + AC$

c. $f(A,B,C,D) = \Sigma\ m(1,3,5,7,9) + \Sigma\ d(6,12,13)$

d. $f(A,B,C,D) = \Pi\ M(0,1,6,7)$

e. $f(A,B,C,D) = \Sigma\ m(0,2,4,6)$

2.21 *(Positive and Negative Logic)* Show the following:

a. A positive logic AND is equivalent to a negative logic OR.

b. A positive logic NOR is equivalent to a negative logic NAND.

hybrid jump counter

c. A positive logic XOR is equivalent to a negative logic XNOR.

d. A positive logic XNOR is equivalent to a negative logic XOR.

2.22 *(Quine-McCluskey Method)* Use the Quine-McCluskey method to find the minimum equivalent forms for the following Boolean expressions.

a. $f(X,Y,Z) = \Sigma\, m(2,3,4,5)$

b. $f(A,B,C,D) = \Sigma\, m(0,1,4,5,12,13)$

2.23 *(Quine-McCluskey Method)* Given the function $F(A,B,C,D) = \Sigma\, m(1,5,7,8,9,13,15) + d(4,14)$, find the minimum sum of products form using the Quine-McCluskey method. Show your process of deriving the prime implicants. Include the implication chart from which your minimum sum of products form is derived.

2.24 *(Karnaugh Map Method)* There may be more than one true minimum equivalent form for a given Boolean expression. Demonstrate this by drawing a four-variable K-map that has two different minimized forms for the same Boolean expression, each with the same number of terms and literals.

2.25 *(Karnaugh Map Method)* Using a four-variable K-map, fill it with 1's and 0's to find a function that illustrates the following points. Write the expressions for each of the requested forms and count the number of terms and literals for each one:

a. The minimized sum of products and product of sums forms have the same number of terms and literals.

b. The minimized sum of products form has fewer terms and literals than the minimized product of sums form.

c. The minimized product of sums form has fewer terms and literals than the minimized sum of products form.

2.26 *(Combinational Logic Design)* Consider a five-input Boolean function that is asserted whenever exactly two of its inputs are asserted.

a. Construct its truth table.

b. What is the function in sum of products form, using "little *m*" notation?

c. What is the function in product of sums form, using "big *M*" notation?

d. Use the Karnaugh map method to simplify the function in sum of products form.

2.27 *(Combinational Logic Design)* In this chapter, we've examined the BCD increment by 1 function. Now consider a binary increment by 1 function defined over the 4-bit binary numbers 0000 through 1111.

a. Fill in the truth table for the function.

b. Fill in the four 4-variable K-maps, and find the minimized sum of products for each output function.

c. Repeat the process for the minimized product of sums form. Which leads to the simpler implementation in terms of the number of literals?

2.28 *(Combinational Logic Design)* In this chapter, we've examined a 2-bit binary adder circuit. Now consider a 2-bit binary subtractor, defined as follows. The inputs A, B and C, D form the two 2-bit numbers N_1 and N_2. The circuit will form the difference $N_1 - N_2$ on the output bits F (most significant) and G (least significant). Assume that the circuit never sees an input combination in which N_1 is less than N_2. The output bits are don't cares in these cases.

a. Fill in the four-variable truth table for F and G.

b. Fill in the K-map for the minimum sum of products expression for the functions F and G.

c. Repeat to find the minimum product of sums expression for F and G.

2.29 *(Combinational Logic Design)* Consider a four-input function that outputs a 1 whenever an odd number of its inputs are 1.

a. Fill in the truth table for the function.

b. Fill in the K-map to find the minimum sum of products expression for the function. What is it? Can the function be minimized using the K-map method?

c. Can you think of a more economical way to implement this function if XOR gates are allowed? (*Warning*: It will be very tedious to try to simplify this function using Boolean algebra, so think about the question first!)

2.30 *(Combinational Logic Design)* Design a combinational circuit with three data inputs D_2, D_1, D_0, two control inputs C_1, C_0, and two outputs R_1, R_0. R_1 and R_0 should be the remainder after dividing the binary number formed from D_2, D_1, D_0 by the number formed by C_1, C_0. For example, if D_2, D_1, $D_0 = 111$ and C_1,

hybrid jump counter

$C_0 = 10$, then R_1, $R_0 = 01$ (that is, the remainder of 7 divided by 2 is 1). Note that division by zero will never be requested.

a. Fill in truth tables for the combinational logic functions R_1 and R_0.

b. Derive minimized sum of product realizations of these functions using the Karnaugh map method.

c. Draw a circuit schematic that implements R_1 and R_0 using NAND gates only. You may assume any fan-in gates that you need.