3.12 Simplify the following Boolean functions:

(a)* $F(A, B, C, D) = \Pi(1, 3, 5, 7, 13, 15)$

(b) $F(A, B, C, D) = \Pi(1, 3, 6, 9, 11, 12, 14)$

3.13 Simplify the following expressions to (1) sum-of-products and (2) products-of-sums:

(a)* $x'z' + y'z' + yz' + xy$

(b) $ACD' + C'D + AB' + ABCD$

(c) $(A' + B + D')(A' + B' + C')(A' + B' + C)(B' + C + D')$

(d) $BCD' + ABC' + ACD$

3.14 Give three possible ways to express the following Boolean function with eight or fewer literals:

$$F = A'BC'D + AB'CD + A'B'C' + ACD'$$

3.15 Simplify the following Boolean function F, together with the don't-care conditions d, and then express the simplified function in sum-of-minterms form:

(a) $F(x, y, z) = \Sigma(0, 1, 4, 5, 6)$
$d(x, y, z) = \Sigma(2, 3, 7)$

(b)* $F(A, B, C, D) = \Sigma(0, 6, 8, 13, 14)$
$d(A, B, C, D) = \Sigma(2, 4, 10)$

(c) $F(A, B, C, D) = \Sigma(5, 6, 7, 12, 14, 15,)$
$d(A, B, C, D) = \Sigma(3, 9, 11, 15)$

(d) $F(A, B, C, D) = \Sigma(4, 12, 7, 2, 10,)$
$d(A, B, C, D) = \Sigma(0, 6, 8)$

3.16 Simplify the following functions, and implement them with two-level NAND gate circuits:

(a) $F(A, B, C, D) = AC'D' + A'C + ABC + AB'C + A'C'D'$

(b) $F(A, B, C, D) = A'B'C'D + CD + AC'D$

(c) $F(A, B, C) = (A' + C' + D')(A' + C')(C' + D')$

(d) $F(A, B, C, D) = A' + B + D' + B'C$

3.17* Draw a NAND logic diagram that implements the complement of the following function:

$$F(A, B, C, D) = \Sigma(0, 1, 2, 3, 6, 10, 11, 14)$$

3.18 Draw a logic diagram using only two-input NOR gates to implement the following function:

$$F(A, B, C, D) = (A \oplus B)'(C \oplus D)$$

3.19 Simplify the following functions, and implement them with two-level NOR gate circuits:

(a)* $F = wx' + y'z' + w'yz'$

(b) $F(w, x, y, z) = \Sigma(0, 3, 12, 15)$

(c) $F(x, y, z) = [(x + y)(x = z)]'$

3.20 Draw the multiple-level NOR circuit for the following expression:

$$CD(B + C)A + (BC' + DE')$$

3.21 Draw the multiple-level NAND circuit for the following expression:

$$w(x + y + z) + xyz$$

3.22 Convert the logic diagram of the circuit shown in Fig. 4.4 into a multiple-level NAND circuit.

3.23 Implement the following Boolean function F, together with the don't-care conditions d, using no more than two NOR gates:

$$F(A, B, C, D) = \Sigma(2, 4, 10, 12, 14,)$$
$$d(A, B, C, D) = \Sigma(0, 1, 5, 8)$$

Assume that both the normal and complement inputs are available.

3.24 Implement the following Boolean function F, using the two-level forms of logic (a) NAND-AND, (b) AND-NOR, (c) OR-NAND, and (d) NOR-OR:

$$F(A, B, C, D) = \Sigma(0, 4, 8, 9, 10, 11, 12, 14)$$

3.25 List the eight degenerate two-level forms and show that they reduce to a single operation. Explain how the degenerate two-level forms can be used to extend the number of inputs to a gate.

3.26 With the use of maps, find the simplest sum-of-products form of the function $F = fg$, where

$$f = abc' + c'd + a'cd' + b'cz'$$

and

$$g = (a + b + c' + d')(b' + c' + d)(a' + c + d')$$

3.27 Show that the dual of the exclusive-OR is also its complement.

3.28 Derive the circuits for a three-bit parity generator and four-bit parity checker using an odd parity bit.

3.29 Implement the following four Boolean expressions with three half adders:

$$D = A \oplus B \oplus C$$
$$E = A'BC + AB'C$$
$$F = ABC' + (A' + B')C$$
$$G = ABC$$

3.30* Implement the following Boolean expression with exclusive-OR and AND gates:

$$F = AB'CD' + A'BCD' + AB'C'D + A'BC'D$$

3.31 Write a Verilog gate-level description of the circuit shown in

(a) Fig. 3.20(a) (b) Fig. 3.20(b) (c) Fig. 3.21(a)
(d) Fig. 3.21(b) (e) Fig. 3.24 (f) Fig. 3.25

3.32 Using continuous assignment statements, write a Verilog description of the circuit shown in

(a) Fig. 3.20(a) (b) Fig. 3.20(b) (c) Fig. 3.21(a)
(d) Fig. 3.21(b) (e) Fig. 3.24 (f) Fig. 3.25

3.33 The exclusive-OR circuit of Fig. 3.30(a) has gates with a delay of 3 ns for an inverter, a 6 ns delay for an AND gate, and a 8 ns delay for an OR gate. The input of the circuit goes from $xy = 00$ to $xy = 01$.

(a) Determine the signals at the output of each gate from $t = 0$ to $t = 50$ ns.
(b) Write a Verilog gate-level description of the circuit, including the delays.
(c) Write a stimulus module (i.e., a test bench similar to HDL Example 3.3), and simulate the circuit to verify the answer in part (a).

3.34 Using continuous assignments, write a Verilog description of the circuit specified by the following Boolean functions:

$$Out\_1 = (A + B')C'(C + D)$$
$$Out\_2 = (C'D + BCD + CD')(A' + B)$$
$$Out\_3 = (AB + C)D + B'C$$

Write a test bench and simulate the circuit's behavior.

3.35* Find the syntax errors in the following declarations (note that names for primitive gates are optional):

```
module Exmpl-3(A, B, C, D, F)          // Line 1
 inputs      A, B, C, Output D, F,     // Line 2
 output      B                         // Line 3
 and         g1(A, B, D);              // Line 4
 not         (D, A, C),                // Line 5
 OR          (F, B; C);                // Line 6
endmodule;                             // Line 7
```

3.36 Draw the logic diagram of the digital circuit specified by the following Verilog description:

(a)
```
module Circuit_A (A, B, C, D, F);
  input     A, B, C, D;
  output    F;
  wire      w, x, y, z, a, d;
  or        (x, B, C, d);
  and       (y, a ,C);
  and       (w, z ,B);
  and       (z, y, A);
  or        (F, x, w);
  not       (a, A);
  not       (d, D);
endmodule
```

(b)
```
module Circuit_B (F1, F2, F3, A0, A1, B0, B1);
  output    F1, F2, F3;
  input     A0, A1, B0, B1;
  nor       (F1, F2, F3);
  or        (F2, w1, w2, w3);
  and       (F3, w4, w5);
  and       (w1, w6, B1);
  or        (w2, w6, w7, B0);
  and       (w3, w7, B0, B1);
  not       (w6, A1);
  not       (w7, A0);
  xor       (w4, A1, B1);
  xnor      (w5, A0, B0);
endmodule
```

(c)
```
module Circuit_C (y1, y2, y3, a, b);
  output y1, y2, y3;
  input a, b;

  assign y1 = a || b;
  and (y2, a, b);
  assign y3 = a && b;
endmodule
```

3.37 A majority logic function is a Boolean function that is equal to 1 if the majority of the variables are equal to 1, equal to 0 otherwise.
(a) Write a truth table for a four-bit majority function.
(b) Write a Verilog user-defined primitive for a four-bit majority function.

3.38 Simulate the behavior of *Circuit_with_UDP_02467*, using the stimulus waveforms shown in Fig. P3.38.

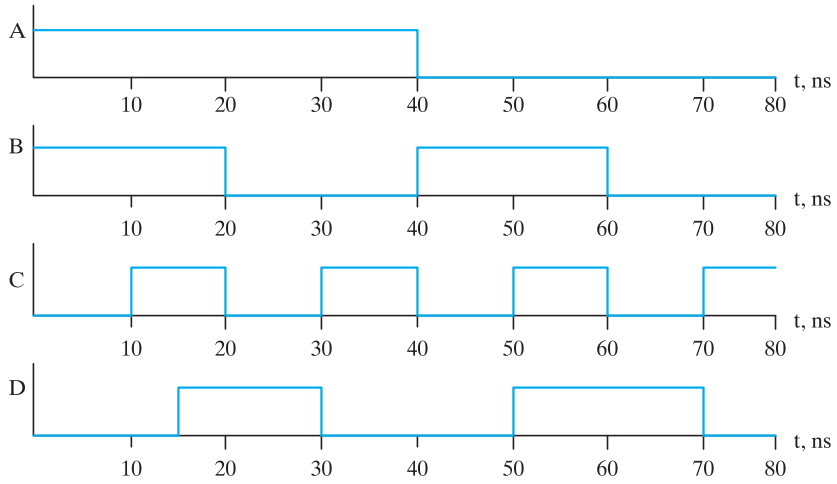

FIGURE P3.38
Stimulus waveforms for Problem 3.38

3.39 Using primitive gates, write a Verilog model of a circuit that will produce two outputs, s and c, equal to the sum and carry produced by adding two binary input bits a and b (e.g., $s = 1$ and $c = 0$ if $a = 0$ and $b = 1$). (*Hint:* Begin by developing a truth table for s and c.)

REFERENCES

1. Bhasker, J. 1997. *A Verilog HDL Primer*. Allentown, PA: Star Galaxy Press.
2. Ciletti, M. D. 1999. *Modeling, Synthesis and Rapid Prototyping with the Verilog HDL*. Upper Saddle River, NJ: Prentice Hall.
3. Hill, F. J., and G. R. Peterson. 1981. *Introduction to Switching Theory and Logical Design*, 3rd ed. New York: John Wiley.
4. *IEEE Standard Hardware Description Language Based on the Verilog Hardware Description Language* (IEEE Std. 1364-1995). 1995. New York: The Institute of Electrical and Electronics Engineers.
5. Karnaugh, M. A Map Method for Synthesis of Combinational Logic Circuits. *Transactions of AIEE, Communication and Electronics*. 72, part I (Nov. 1953): 593–99.
6. Kohavi, Z. 1978. *Switching and Automata Theory*, 2nd ed. New York: McGraw-Hill.