3-6. *Find simplified Boolean equations for the outputs \( F \) and \( G \) of the hierarchical circuit in Figure 3-56.

3-7. Find the truth table for the outputs \( F \) and \( G \) of the hierarchical circuit in Figure 3-57. The symbol shown is for the decoder block in Figure 3-14.

3-8. The logic diagram for a 74HC138 MSI CMOS circuit is given in Figure 3-58. Find the Boolean function for each of the outputs. Describe carefully the circuit function.

3-9. Do Problem 3-8 by using logic simulation to find the output waveforms of the circuit or a partial truth table listing, rather than finding Boolean functions.

3-10. A majority function is generated in a combinational circuit when the output is equal to 1 if the input variables have more 1's than 0's. The output is 0 otherwise. Design a 3-input majority function. What circuit that you are familiar with does this circuit correspond to?

3-11. *Design a combinational circuit that detects an error in the representation of a decimal digit in BCD. In other words, obtain a logic diagram whose output is equal to 1 when the inputs contain any one of the six unused bit combinations in the BCD code.

170 □ CHAPTER 3 / COMBINATIONAL LOGIC DESIGN
\[ \text{FIGURE 3-57} \]
Circuit for Problem 3-7

\[ \text{FIGURE 3-58} \]
Circuit for Problem 3-8 and Problem 3-9
3-12. Complete the design of the BCD-to-seven-segment decoder by performing the following steps:
   (a) Plot the seven maps for each of the outputs for the BCD-to-seven-segment decoder specified in Table 3-3.
   (b) Simplify the seven output functions in sum-of-products form, and determine the total number of gates that will be needed to implement the decoder.
   (c) Verify that the seven output functions listed in the text give a valid simplification. Compare the number of gates with that obtained in part (b).

3-13. *Draw the logic diagram of the BCD-to-seven-segment decoder from the Boolean functions listed in Section 3-4 of the text. Use only 21 NAND gates. Assume that both the complemented and uncomplemented inputs are available.

3-14. Construct a 4-to-16-line decoder with an enable input using five 2-to-4-line decoders with enable inputs.

3-15. *Design a BCD-to-decimal decoder using the unused combinations of the BCD code as don't-care conditions.

3-16. Design an excess-3 to BCD code converter, requiring that all invalid input combinations give 0000 as the output code.

3-17. A combinational circuit is defined by the following three Boolean functions:
   \[ F_1 = X' + Y + XYZ \]
   \[ F_2 = X' + Y + XYZ \]
   \[ F_3 = XYZ + X' + Y \]
   Design the circuit with a decoder and external OR gates.

3-18. A combinational circuit is specified by the following three Boolean functions:
   \[ F_4(A, B, C) = \Sigma m(1, 4, 7) \]
   \[ F_5(A, B, C) = \Sigma m(0, 3, 6) \]
   \[ F_6(A, B, C) = \Pi M(0, 4, 6, 7) \]
   Implement the circuit with a decoder constructed with NAND gates (similar to Figure 3-14) and external NAND gates.

3-19. Draw the logic diagram of a 3-to-8-line decoder with only NOR and NOT gates. Include an enable input.

172  □  CHAPTER 3 / COMBINATIONAL LOGIC DESIGN
3-20. Design a 4-input priority encoder with inputs and outputs as in Table 3-6, but with the truth table representing the case in which input $D_1$ has the highest priority and input $D_2$ has the lowest priority.

3-21. Derive the truth table of an octal-to-binary priority encoder.

3-22. Construct an 8-to-1 line multiplexer with enable input using transmission gates.

3-23. Construct a 10-to-1 line multiplexer with three 4-to-1 line multiplexers. The multiplexers should be interconnected and inputs labeled so that the selection codes 0000 through 1001 can be directly applied to the multiplexer selections inputs without added logic.

3-24. Construct a quad 9-to-1 line multiplexer with four single 8-to-1 line multiplexers and one quadruple 2-to-1 line multiplexer. The multiplexers should be interconnected and inputs labeled so that the selection codes 0000 through 1000 can be directly applied to the multiplexer selection inputs without added logic.

3-25. Construct a 15-to-1 line multiplexer with two 8-to-1 line multiplexers. Interconnect the two multiplexers and label the inputs such that any added logic required to have selection codes 0000 through 1110 is minimized.

3-26. Construct a dual 7-to-1 line multiplexer from two dual 4-to-1 line multiplexers plus minimum added control logic for the selection codes:

<table>
<thead>
<tr>
<th>Selection Code</th>
<th>Data Input Selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_0$ $x_1$ $x_2$ $x_3$</td>
<td>$D_0$ $D_1$ $D_2$ $D_3$ $D_4$ $D_5$ $D_6$</td>
</tr>
<tr>
<td>1 0 0 0</td>
<td>0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 1 0 0</td>
<td>0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 1 0</td>
<td>0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 1</td>
<td>0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 0</td>
<td>1 0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 0</td>
<td>0 1 0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 0</td>
<td>0 0 1 0 0 0 0</td>
</tr>
</tbody>
</table>

3-27. Implement a binary full adder with a dual 4-to-1 line multiplexer and a single inverter.

3-28. Implement the following Boolean function with an 8-to-1 line multiplexer and a single inverter:

$$F(A, B, C, D) = \Sigma m(2, 3, 5, 6, 8, 9, 12, 14)$$

3-29. Implement the Boolean function defined in the truth table of Figure 3-23 with a 4-to-1 line multiplexer and external gates. Connect inputs $A$ and $B$ to the selection lines. The input requirements for the four data lines will be a function of the variables $C$ and $D$. The values of these variables are obtained by