1. (MK 3-1) Determine the Boolean functions for outputs $X$ and $Y$ as a function of the four inputs in the circuit in Figure 3-52.

\[
T_1 = D' \quad T_2 = BC \quad T_3 = (T_1 + A)' = A'D \\
T_4 = (T_1 \cdot T_3)' = T_1' + T_3' = D' + (A'D)' = D' + A + D' = 1 \\
T_5 = T_1 + T_2 = D' + BC
\]

\[
X = T_4T_5 = D' + BC \quad Y = T_3 \cdot T_5 = A'D'(D' + BC) = A'BCD
\]

2. (MK 3-2) Obtain the truth table for the circuit shown in Figure 3-53. Draw an equivalent circuit for $F$ with fewer NAND gates.
3. (MK 3-4) Find the simplified Boolean functions for outputs \( F \) and \( G \) of the circuit in Figure 3-55.

\[
\begin{array}{|c|c|c|c|c|c|c|}
\hline
XYZ & l_1=(XY)' & l_2=(X\bar{Y})' & l_3=(\bar{X}Y)' & l_4=(l_2 \bar{Z}') & l_5=(l_3 \bar{Z}') & t=(l_4 l_5)' \\
\hline
000 & 1 & 1 & 1 & 1 & 1 & 0 \\
001 & 1 & 1 & 0 & 1 & 0 & 1 \\
010 & 1 & 1 & 0 & 0 & 1 & 0 \\
011 & 1 & 1 & 0 & 0 & 0 & 1 \\
100 & 1 & 0 & 1 & 1 & 0 & 0 \\
101 & 1 & 0 & 1 & 1 & 0 & 1 \\
110 & 0 & 0 & 1 & 1 & 0 & 1 \\
111 & 0 & 0 & 1 & 1 & 0 & 1 \\
\hline
\end{array}
\]

Equivalent Circuit

\[ T_1 = (ABD)' \quad T_2 = (AC)' \quad T_3 = (BCD)' \]
\[ T_4 = (AT_1 T_2)' = A' + ABD + AC = A' + BD + C = F \]
\[ T_5 = (T_3 B T_3)' = AD + B' + CD \quad T_6 = (T_2 T_3 C)' = A + BD + C' \]
\[ G = (T_4 T_5 T_6)' = A (B' + D') C' + (A' + D') B (C' + D') + A' (B' + D') C = AB'C' + A'B'C + A'B'D + BD \]
4. (MK 3-5) Find the truth table for \( F \) and \( G \) of the circuit in Figure 3-55 by using logic simulation.
5. (MK 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.

<table>
<thead>
<tr>
<th>XYZ</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0010</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0011</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0100</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0101</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0110</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0111</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1001</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1010</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1011</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1100</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1101</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1110</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1111</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

6. (MK 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?

\[ M = A \overline{B} + A \overline{C} + B \overline{C} \]

This is the same function implemented by the carry output of the full adder.
7. (MK 3-12) Complete the design of the BCD-to-seven-segment decoder by performing the following steps.

(a) Plot the seven maps for the outputs, 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 (b).

\[
a = A'B'C' + B'C'D' + A'BD + A'C
\]

\[
b = A'B' + B'C' + A'C'D' + A'CD
\]

\[
c = A'B + B'C' + A'D
\]

\[
d = B'C'D' + A'B'C' + A'BC'D + A'B'C + A'CD'
\]

\[
e = B'C'D' + A'CD'
\]
(b) The sum of products expressions are given with the maps above.

If each expression is implemented separately, then

for a we have 4 ANDs plus 1 OR
for b we have 4 ANDs plus 1 OR
for c we have 3 ANDs plus 1 OR
for d we have 5 ANDs plus 1 OR
for e we have 2 ANDs plus 1 OR
for f we have 4 ANDs plus 1 OR
for g we have 4 ANDs plus 1 OR
this gives a total of 26 ANDs and 7 ORs

By sharing gates for common product terms, we can save some gates.

\[
\begin{align*}
A'B'C' & \text{ appears 2 times, save 1 AND} \\
B'C'D' & \text{ appears 4 times, save 3 ANDs} \\
A'BD & \text{ appears 1 time, save 0 ANDs} \\
A'C & \text{ appears 1 time, save 0 ANDs} \\
A'B & \text{ appears 2 times, save 1 AND} \\
B'C & \text{ appears 2 times, save 1 AND} \\
A'C'D' & \text{ appears 1 time, save 0 ANDs} \\
A'CD & \text{ appears 1 time, save 0 ANDs} \\
A'D & \text{ appears 1 time, save 0 ANDs} \\
A'BC'D & \text{ appears 1 time, save 0 ANDs} \\
A'B'C & \text{ appears 2 times, save 1 AND} \\
A'CD' & \text{ appears 3 times, save 2 ANDs} \\
A'BD' & \text{ appears 1 time, save 0 ANDs} \\
A'BC' & \text{ appears 2 times, save 1 AND}
\end{align*}
\]
So, we get an overall savings of 16 AND gates and 7 OR gates, of 23 altogether.

(c) The number of gates obtained in (b) is two more than in the textbook design. Even though the book uses more product terms for some expressions, there appears to be more overlap in the terms they use, allowing for an overall savings of two gates.

Verification of textbook vs. solution here.

<table>
<thead>
<tr>
<th>Textbook</th>
<th>Derived above</th>
</tr>
</thead>
<tbody>
<tr>
<td>a: $A'C' + A'BD + B'C'D + AB'C'$</td>
<td>$A'B'C' + B'C'D' + A'BD + A'C$</td>
</tr>
<tr>
<td>c: $A'B + A'D + B'C'D' + AB'C'$</td>
<td>$A'B + B'C' + A'D$</td>
</tr>
<tr>
<td>e: $A'CD' + B'C'D$</td>
<td>$B'C'D' + A'CD$</td>
</tr>
<tr>
<td>f: $A'BC' + A'C'D' + A'BD' + AB'C'$</td>
<td>$B'C'D' + A'B'C' + A'BD' + A'BC'$</td>
</tr>
<tr>
<td>g: $A'CD' + A'B'C + A'BC' + A'B'C'$</td>
<td>$A'BC' + A'B'C' + A'B'C + A'CD'$</td>
</tr>
</tbody>
</table>

We can see that in most cases (a,d,e,g) the expressions are identical, except for the order of the terms. For b, the textbook expression has the term $A'B'C'$ while the solution derived above has the term $B'C'$. Since the term $A'B'C'$ is covered by the term $A'B$ in the textbook expression, the two expressions are equivalent. This can also be verified by examining the Karnaugh map for b, above. For c, we can see that each term in the textbook expression is covered by a term in the expression derived above. The term $B'C'$ on the right, does not appear in the textbook expression, but it’s easy to see that all four minterms covered by $B'C'$ are covered by the textbook expression. For f, the difference between the two solutions is that the textbook expression includes the term $A'C'D'$ where the expression derived above has $B'C'D'$. Since both minterms covered by $A'C'D'$ are also covered by terms on the right, and since both minterms covered by $B'C'D'$ are covered by terms on the left, the two expressions are equivalent. This can also be seen by inspection of the Karnaugh map for f.
8. (MK 3-16) Design an excess 3 to BCD code converter, requiring that all invalid input combinations give 0000 as the output code.

There are 12 product terms, only one of which is a repeat. So, the implementation requires 11 AND gates and 4 OR gates, plus 4 inverters.

The schematic and simulation output are shown below.
9. (MK 3-17) A combinational circuit is defined by the following three Boolean functions.

\[ F_1 = (X + Y)' + X Y Z' \]
\[ F_2 = (X + Y)' + X' Y Z \]
\[ F_3 = X Y Z + (X + Y)' \]

Design the circuit with a decoder and external OR gates.

<table>
<thead>
<tr>
<th>XYZ</th>
<th>F_1F_2F_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>1 1 1</td>
</tr>
<tr>
<td>001</td>
<td>1 1 1</td>
</tr>
<tr>
<td>010</td>
<td>0 0 0</td>
</tr>
<tr>
<td>011</td>
<td>0 1 0</td>
</tr>
<tr>
<td>100</td>
<td>0 0 0</td>
</tr>
<tr>
<td>101</td>
<td>0 0 0</td>
</tr>
<tr>
<td>110</td>
<td>1 0 0</td>
</tr>
<tr>
<td>111</td>
<td>0 0 1</td>
</tr>
</tbody>
</table>