Sum-of-Products (SOP) Expressions

Most Boolean reductions result in an equation in one of two forms:

1. *Product-of-sums* (POS) expression
2. *Sum-of-products* (SOP) expression

The POS expression usually takes the form of two or more ORed variables within parentheses ANDed with two or more other variables within parentheses. Examples of POS expressions are:

$$X = (A + \overline{B}) \cdot (B + C)$$
$$X = (B + \overline{C} + \overline{D}) \cdot (BC + \overline{E})$$
$$X = (A + \overline{C}) \cdot (\overline{B} + E) \cdot (C + B)$$

The SOP expression usually takes the form of two or more variables ANDed together ORed with two or more other variables ANDed together. Examples of SOP expressions are:

$$X = A\overline{B} + AC + \overline{A}BC$$
$$X = AC\overline{D} + \overline{C}D + B$$
$$X = B\overline{C}\,\overline{D} + A\overline{B}DE + CD$$

The SOP expression is used most often because it lends itself nicely to the development of truth tables, timing diagrams, and Karnaugh maps.

For example, let's work with the equation

$$X = \overline{A\overline{B} + \overline{C}D}$$

Using DeMorgan's theorem yields

$$X = \overline{A\overline{B}} \cdot \overline{\overline{C}D}$$

Using DeMorgan's theorem again puts it into a POS format:

$$X = (\overline{A} + B) \cdot (C + \overline{D}) \leftarrow \text{POS}$$

Using the distributive law produces an equation in the SOP format:

$$X = \overline{A}C + \overline{A}\,\overline{D} + BC + B\overline{D} \leftarrow \text{SOP}$$

Now, to fill in a truth table for X using the *SOP* expression, we would put a 1 at X for $A = 0, C = 1$; and for $A = 0, D = 0$; and for $B = 1, C = 1$; and for $B = 1, D = 0$.

3–5 KARNAUGH MAPPING

We learned in previous sections that by using Boolean algebra and DeMorgan's theorem we can minimize the number of gates that are required to implement a particular logic function. This is very important for the reduction of circuit cost, physical size, and gate failures. You may have found some of the steps in the Boolean reduction process to require ingenuity on your part, and a lot of practice.

Karnaugh mapping, named for its originator, is another method of simplifying logic circuits. It still requires that you reduce the equation to an SOP form, but from there you follow a *systematic approach* that will always produce the simplest configuration possible for the logic circuit.

A Karnaugh map (K-map) is similar to a truth table in that it graphically shows the output level of a Boolean equation for each of the possible input variable combinations. Each output level is placed in a separate *cell* of the K-map. K-maps can be used to simplify equations having two, three, four, five, or six different input variables. Solving five- and six-variable K-maps is extremely cumbersome and can be more practically solved using advanced computer techniques. In this book we will solve two-, three-, and four-variable K-maps.

Determining the number of cells in a K-map is the same as finding the number of combinations or entries in a truth table. A two-variable map will require $2^2 = 4$ cells. A three-variable map will require $2^3 = 8$ cells. A four-variable map will require $2^4 = 16$ cells. The three different K-maps are shown in Figure 3–36.

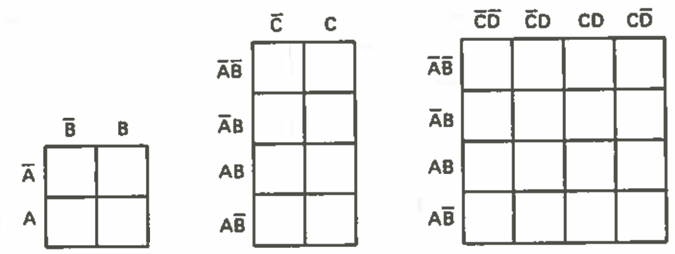

Figure 3–36 Two-variable, three-variable, and four-variable Karnaugh maps.

Each cell within the K-map corresponds to a particular combination of the input variables. For example, in the two-variable K-map, the upper-left cell corresponds to $\overline{A}\,\overline{B}$, the lower-left cell is $A\overline{B}$, the upper-right cell is $\overline{A}B$, and the lower-right cell is AB.

Also notice that when moving from one cell to an adjacent cell, only one variable changes. For example, look at the three-variable K-map. The upper-left cell is $\overline{A}\,\overline{B}\,\overline{C}$, the adjacent cell just below it is $\overline{A}B\overline{C}$. In that case the $\overline{A}\,\overline{C}$ remained the same and only the $\overline{B}$ changed to B. The same holds true for each adjacent cell.

To use the K-map reduction procedure, you must perform the following steps:

1. Transform the Boolean equation to be reduced into an SOP expression.
2. Fill in the appropriate cells of the K-map.
3. Encircle adjacent cells in groups of two, four, or eight. (The more adjacent cells encircled, the simpler the final equation.)
4. Find each term of the final SOP equation by determining which variables remain constant within each circle.

Now, let's consider the equation

$$X = \overline{A}(\overline{B}C + \overline{B}\,\overline{C}) + \overline{A}B\overline{C}$$

First, transform the equation to an SOP expression:

$$X = \overline{A}\,\overline{B}C + \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C}$$

The terms of that SOP expression can be put into a truth table, then transferred to a K-map, as shown in Figure 3–37. Working with the K-map, we will now encircle adjacent 1s in groups of two, four, or eight. We end up with two circles of two cells each, as shown in Figure 3–38. The first circle surrounds the two 1s at the top of the K-map and the second circle surrounds the two 1s in the left column of the K-map.

Once the circles have been drawn encompassing all the 1s in the map, the final simplified equation is obtained by determining *which variables remain the same within each*