

Digital Logic & Design (Theory)

Lecture No.5

CHAPTER OUTLINE

- 4-1 Boolean Operations and Expressions
- 4-2 Laws and Rules of Boolean Algebra
- 4-3 DeMorgan's Theorems
- 4-4 Boolean Analysis of Logic Circuits
- 4-5 Logic Simplification Using Boolean Algebra
- 4-6 Standard Forms of Boolean Expressions

4-1 Boolean Operations and Expressions

Boolean algebra is the mathematics of digital logic. A basic knowledge of Boolean algebra is indispensable to the study and analysis of logic circuits. In the last chapter, Boolean operations and expressions in terms of their relationship to NOT, AND, OR, NAND, and NOR gates were introduced.

Variable, complement, and literal are terms used in Boolean algebra. A **variable** is a symbol (usually an italic uppercase letter or word) used to represent an action, a condition, or data. Any single variable can have only a 1 or a 0 value. The **complement** is the inverse of a variable and is indicated by a bar over the variable (overbar). For example, the complement

of the variable A is \bar{A} . If $A = 1$, then $\bar{A} = 0$. If $A = 0$, then $\bar{A} = 1$. The complement of the variable A is read as “not A ” or “ A bar.” Sometimes a prime symbol rather than an overbar is used to denote the complement of a variable; for example, B' indicates the complement of B . In this book, only the overbar is used. A **literal** is a variable or the complement of a variable.

Boolean Addition

Recall from Chapter 3 that **Boolean addition** is equivalent to the OR operation. The basic rules are illustrated with their relation to the OR gate in Figure 4-1.

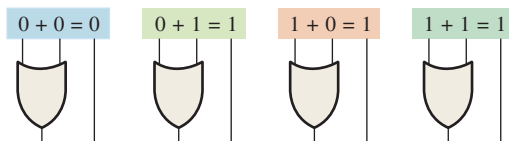


FIGURE 4-1

In Boolean algebra, a **sum term** is a sum of literals. In logic circuits, a sum term is produced by an OR operation with no AND operations involved. Some examples of sum terms are $A + B$, $A + \bar{B}$, $A + B + \bar{C}$, and $\bar{A} + B + C + \bar{D}$.

A sum term is equal to 1 when one or more of the literals in the term are 1. A sum term is equal to 0 only if each of the literals is 0.

EXAMPLE 4-1

Determine the values of A , B , C , and D that make the sum term $A + \bar{B} + C + \bar{D}$ equal to 0.

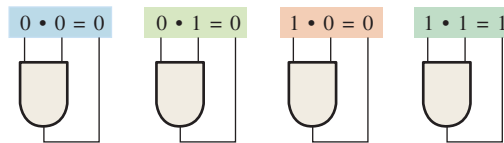
Solution

For the sum term to be 0, each of the literals in the term must be 0. Therefore, $A = 0$, $B = 1$ so that $\bar{B} = 0$, $C = 0$, and $D = 1$ so that $\bar{D} = 0$.

$$A + \bar{B} + C + \bar{D} = 0 + \bar{1} + 0 + \bar{1} = 0 + 0 + 0 + 0 = 0$$

Boolean Multiplication

Also recall from Chapter 3 that **Boolean multiplication** is equivalent to the AND operation. The basic rules are illustrated with their relation to the AND gate in Figure 4-2.

**FIGURE 4-2**

In Boolean algebra, a **product term** is the product of literals. In logic circuits, a product term is produced by an AND operation with no OR operations involved. Some examples of product terms are AB , $A\bar{B}$, ABC , and $A\bar{B}\bar{C}\bar{D}$.

A product term is equal to 1 only if each of the literals in the term is 1. A product term is equal to 0 when one or more of the literals are 0.

EXAMPLE 4-2

Determine the values of A , B , C , and D that make the product term $A\bar{B}\bar{C}\bar{D}$ equal to 1.

Solution

For the product term to be 1, each of the literals in the term must be 1. Therefore, $A = 1$, $B = 0$ so that $\bar{B} = 1$, $C = 1$, and $D = 0$ so that $\bar{D} = 1$.

$$A\bar{B}\bar{C}\bar{D} = 1 \cdot \bar{0} \cdot \bar{1} \cdot \bar{0} = 1 \cdot 1 \cdot 1 \cdot 1 = 1$$

4-2 Laws and Rules of Boolean Algebra

As in other areas of mathematics, there are certain well-developed rules and laws that must be followed in order to properly apply Boolean algebra. The most important of these are presented in this section.

Laws of Boolean Algebra

The basic laws of Boolean algebra—the **commutative laws** for addition and multiplication, the **associative laws** for addition and multiplication, and the **distributive law**—are the same as in ordinary algebra. Each of the laws is illustrated with two or three variables, but the number of variables is not limited to this.

Commutative Laws

The *commutative law of addition* for two variables is written as

$$A + B = B + A$$

Equation 4-1

This law states that the order in which the variables are ORed makes no difference. Remember, in Boolean algebra as applied to logic circuits, addition and the OR operation are the same. Figure 4-3 illustrates the commutative law as applied to the OR gate and shows that it doesn't matter to which input each variable is applied. (The symbol \equiv means "equivalent to.")

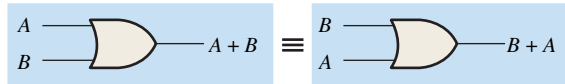


FIGURE 4-3 Application of commutative law of addition.

The *commutative law of multiplication* for two variables is

$$AB = BA \quad \text{Equation 4-2}$$

This law states that the order in which the variables are ANDed makes no difference. Figure 4-4 illustrates this law as applied to the AND gate. Remember, in Boolean algebra as applied to logic circuits, multiplication and the AND function are the same.

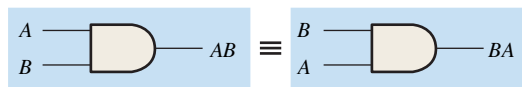


FIGURE 4-4 Application of commutative law of multiplication.

Associative Laws

The *associative law of addition* is written as follows for three variables:

$$A + (B + C) = (A + B) + C \quad \text{Equation 4-3}$$

This law states that when ORing more than two variables, the result is the same regardless of the grouping of the variables. Figure 4-5 illustrates this law as applied to 2-input OR gates.

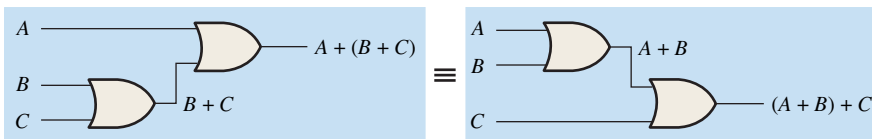


FIGURE 4-5 Application of associative law of addition. Open file F04-05 to verify. *A Multisim tutorial is available on the website.*

The *associative law of multiplication* is written as follows for three variables:

$$A(BC) = (AB)C \quad \text{Equation 4-4}$$

This law states that it makes no difference in what order the variables are grouped when ANDing more than two variables. Figure 4-6 illustrates this law as applied to 2-input AND gates.

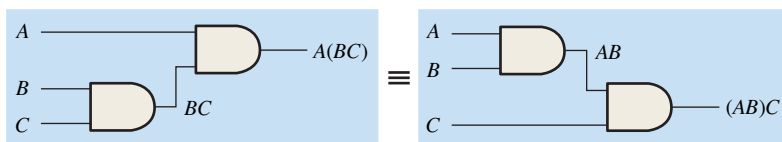


FIGURE 4-6 Application of associative law of multiplication. Open file F04-06 to verify.

Distributive Law

The distributive law is written for three variables as follows:

$$A(B + C) = AB + AC \quad \text{Equation 4-5}$$

This law states that ORing two or more variables and then ANDing the result with a single variable is equivalent to ANDing the single variable with each of the two or more variables and then ORing the products. The distributive law also expresses the process of *factoring* in which the common variable A is factored out of the product terms, for example, $AB + AC = A(B + C)$. Figure 4-7 illustrates the distributive law in terms of gate implementation.

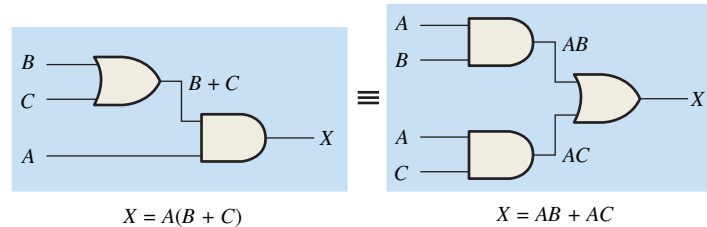


FIGURE 4-7 Application of distributive law. Open file F04-07 to verify.

Rules of Boolean Algebra

Table 4-1 lists 12 basic rules that are useful in manipulating and simplifying **Boolean expressions**. Rules 1 through 9 will be viewed in terms of their application to logic gates. Rules 10 through 12 will be derived in terms of the simpler rules and the laws previously discussed.

TABLE 4-1

Basic rules of Boolean algebra.

1. $A + 0 = A$	7. $A \cdot A = A$
2. $A + 1 = 1$	8. $A \cdot \bar{A} = 0$
3. $A \cdot 0 = 0$	9. $\bar{\bar{A}} = A$
4. $A \cdot 1 = A$	10. $A + AB = A$
5. $A + A = A$	11. $A + \bar{A}B = A + B$
6. $A + \bar{A} = 1$	12. $(A + B)(A + C) = A + BC$

A , B , or C can represent a single variable or a combination of variables.

Rule 1: $A + 0 = A$ A variable ORed with 0 is always equal to the variable. If the input variable A is 1, the output variable X is 1, which is equal to A . If A is 0, the output is 0, which is also equal to A . This rule is illustrated in Figure 4-8, where the lower input is fixed at 0.

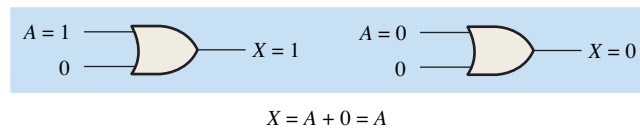


FIGURE 4-8

Rule 2: $A + 1 = 1$ A variable ORed with 1 is always equal to 1. A 1 on an input to an OR gate produces a 1 on the output, regardless of the value of the variable on the other input. This rule is illustrated in Figure 4-9, where the lower input is fixed at 1.

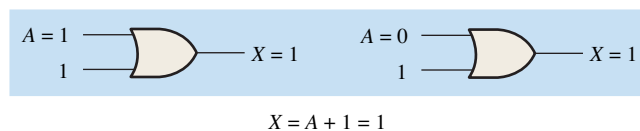


FIGURE 4-9

Rule 3: $A \cdot 0 = 0$ A variable ANDed with 0 is always equal to 0. Any time one input to an AND gate is 0, the output is 0, regardless of the value of the variable on the other input. This rule is illustrated in Figure 4–10, where the lower input is fixed at 0.

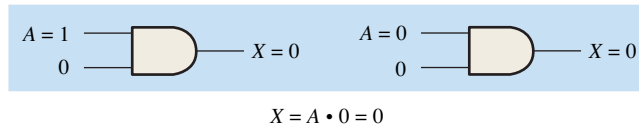


FIGURE 4-10

Rule 4: $A \cdot 1 = A$ A variable ANDed with 1 is always equal to the variable. If A is 0, the output of the AND gate is 0. If A is 1, the output of the AND gate is 1 because both inputs are now 1s. This rule is shown in Figure 4–11, where the lower input is fixed at 1.

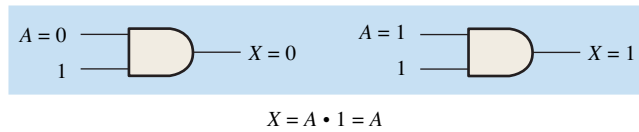


FIGURE 4-11

Rule 5: $A + A = A$ A variable ORed with itself is always equal to the variable. If A is 0, then $0 + 0 = 0$; and if A is 1, then $1 + 1 = 1$. This is shown in Figure 4–12, where both inputs are the same variable.

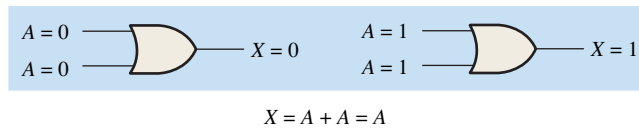


FIGURE 4-12

Rule 6: $A + \bar{A} = 1$ A variable ORed with its complement is always equal to 1. If A is 0, then $0 + \bar{0} = 0 + 1 = 1$. If A is 1, then $1 + \bar{1} = 1 + 0 = 1$. See Figure 4–13, where one input is the complement of the other.

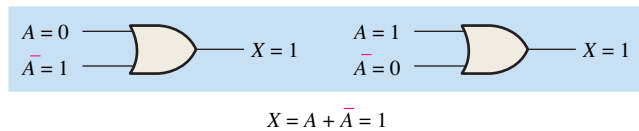


FIGURE 4-13

Rule 7: $A \cdot A = A$ A variable ANDed with itself is always equal to the variable. If $A = 0$, then $0 \cdot 0 = 0$; and if $A = 1$, then $1 \cdot 1 = 1$. Figure 4–14 illustrates this rule.

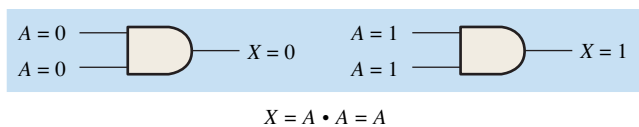


FIGURE 4-14

Rule 8: $A \cdot \bar{A} = 0$ A variable ANDed with its complement is always equal to 0. Either A or \bar{A} will always be 0; and when a 0 is applied to the input of an AND gate, the output will be 0 also. Figure 4–15 illustrates this rule.

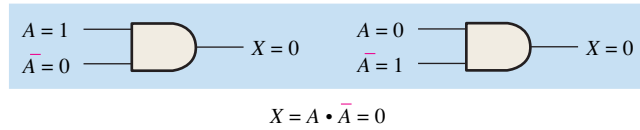


FIGURE 4-15

Rule 9: $\bar{\bar{A}} = A$ The double complement of a variable is always equal to the variable. If you start with the variable A and complement (invert) it once, you get \bar{A} . If you then take \bar{A} and complement (invert) it, you get A , which is the original variable. This rule is shown in Figure 4–16 using inverters.

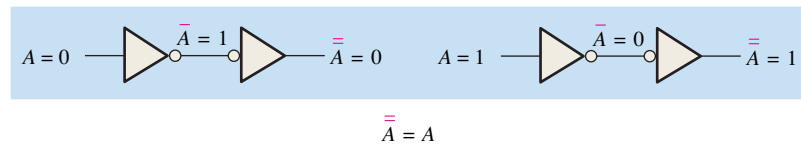


FIGURE 4-16

Rule 10: $A + AB = A$ This rule can be proved by applying the distributive law, rule 2, and rule 4 as follows:

$$\begin{aligned}
 A + AB &= A \cdot 1 + AB && \text{Factoring (distributive law)} \\
 &= A \cdot 1 && \text{Rule 2: } (1 + B) = 1 \\
 &= A && \text{Rule 4: } A \cdot 1 = A
 \end{aligned}$$

The proof is shown in Table 4–2, which shows the truth table and the resulting logic circuit simplification.

TABLE 4-2

Rule 10: $A + AB = A$. Open file T04-02 to verify.

A	B	AB	$A + AB$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

↑ equal ↑

Rule 11: $A + \bar{A}B = A + B$ This rule can be proved as follows:

$$\begin{aligned}
 A + \bar{A}B &= (A + AB) + \bar{A}B && \text{Rule 10: } A = A + AB \\
 &= (AA + AB) + \bar{A}B && \text{Rule 7: } A = AA \\
 &= AA + AB + A\bar{A} + \bar{A}B && \text{Rule 8: adding } A\bar{A} = 0 \\
 &= (A + \bar{A})(A + B) && \text{Factoring} \\
 &= 1 \cdot (A + B) && \text{Rule 6: } A + \bar{A} = 1 \\
 &= A + B && \text{Rule 4: drop the 1}
 \end{aligned}$$

The proof is shown in Table 4–3, which shows the truth table and the resulting logic circuit simplification.

TABLE 4-3

Rule 11: $A + \bar{A}B = A + B$. Open file T04-03 to verify.

A	B	$\bar{A}B$	$A + \bar{A}B$	$A + B$
0	0	0	0	0
0	1	1	1	1
1	0	0	1	1
1	1	0	1	1

↑ equal ↑

Rule 12: $(A + B)(A + C) = A + BC$ This rule can be proved as follows:

$$\begin{aligned}
 (A + B)(A + C) &= AA + AC + AB + BC && \text{Distributive law} \\
 &= A + AC + AB + BC && \text{Rule 7: } AA = A \\
 &= A(1 + C) + AB + BC && \text{Factoring (distributive law)} \\
 &= A \cdot 1 + AB + BC && \text{Rule 2: } 1 + C = 1 \\
 &= A(1 + B) + BC && \text{Factoring (distributive law)} \\
 &= A \cdot 1 + BC && \text{Rule 2: } 1 + B = 1 \\
 &= A + BC && \text{Rule 4: } A \cdot 1 = A
 \end{aligned}$$

The proof is shown in Table 4-4, which shows the truth table and the resulting logic circuit simplification.

TABLE 4-4

Rule 12: $(A + B)(A + C) = A + BC$. Open file T04-04 to verify.

A	B	C	$A + B$	$A + C$	$(A + B)(A + C)$	BC	$A + BC$
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	1	1	1	1
1	0	0	1	1	1	0	1
1	0	1	1	1	1	0	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

↑ equal ↑

4-3 DeMorgan's Theorems

DeMorgan, a mathematician who knew Boole, proposed two theorems that are an important part of Boolean algebra. In practical terms, DeMorgan's theorems provide mathematical verification of the equivalency of the NAND and negative-OR gates and the equivalency of the NOR and negative-AND gates, which were discussed in Chapter 3.

DeMorgan's first theorem is stated as follows:

The complement of a product of variables is equal to the sum of the complements of the variables.

Stated another way,

The complement of two or more ANDed variables is equivalent to the OR of the complements of the individual variables.

The formula for expressing this theorem for two variables is

$$\overline{XY} = \overline{X} + \overline{Y} \quad \text{Equation 4-6}$$

DeMorgan's second theorem is stated as follows:

The complement of a sum of variables is equal to the product of the complements of the variables.

Stated another way,

The complement of two or more ORed variables is equivalent to the AND of the complements of the individual variables.

The formula for expressing this theorem for two variables is

$$\overline{X + Y} = \overline{X} \overline{Y} \quad \text{Equation 4-7}$$

Figure 4-17 shows the gate equivalencies and truth tables for Equations 4-6 and 4-7.

As stated, DeMorgan's theorems also apply to expressions in which there are more than two variables. The following examples illustrate the application of DeMorgan's theorems to 3-variable and 4-variable expressions.

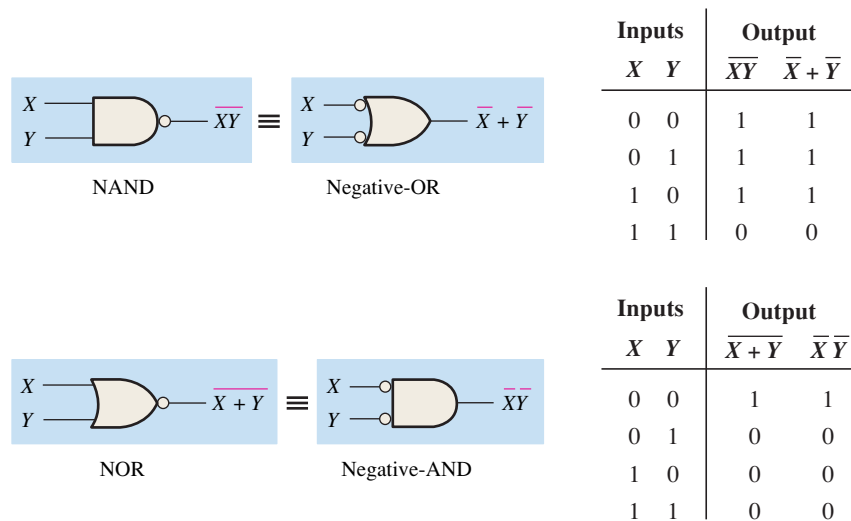


FIGURE 4-17 Gate equivalencies and the corresponding truth tables that illustrate DeMorgan's theorems. Notice the equality of the two output columns in each table. This shows that the equivalent gates perform the same logic function.

EXAMPLE 4-3

Apply DeMorgan's theorems to the expressions \overline{XYZ} and $\overline{X + Y + Z}$.

Solution

$$\overline{XYZ} = \overline{X} + \overline{Y} + \overline{Z}$$

$$\overline{X + Y + Z} = \overline{X} \overline{Y} \overline{Z}$$

EXAMPLE 4-4

Apply DeMorgan's theorems to the expressions \overline{WXYZ} and $\overline{W + X + Y + Z}$.

Solution

$$\begin{aligned}\overline{WXYZ} &= \overline{W} + \overline{X} + \overline{Y} + \overline{Z} \\ \overline{W + X + Y + Z} &= \overline{W}\overline{X}\overline{Y}\overline{Z}\end{aligned}$$

Each variable in DeMorgan's theorems as stated in Equations 4-6 and 4-7 can also represent a combination of other variables. For example, X can be equal to the term $AB + C$, and Y can be equal to the term $A + BC$. So if you can apply DeMorgan's theorem for two variables as stated by $\overline{XY} = \overline{X} + \overline{Y}$ to the expression $\overline{(AB + C)(A + BC)}$, you get the following result:

$$\overline{(AB + C)(A + BC)} = \overline{AB + C} + \overline{A + BC}$$

Notice that in the preceding result you have two terms, $\overline{AB + C}$ and $\overline{A + BC}$, to each of which you can again apply DeMorgan's theorem $X + Y = \overline{X}\overline{Y}$ individually, as follows:

$$\overline{AB + C} + \overline{A + BC} = \overline{AB}\overline{C} + \overline{A}\overline{BC}$$

Notice that you still have two terms in the expression to which DeMorgan's theorem can again be applied. These terms are \overline{AB} and \overline{BC} . A final application of DeMorgan's theorem gives the following result:

$$\overline{AB}\overline{C} + \overline{A}\overline{BC} = \overline{A + B}\overline{C} + \overline{A}\overline{B + C}$$

Although this result can be simplified further by the use of Boolean rules and laws, DeMorgan's theorems cannot be used any more.

Applying DeMorgan's Theorems

The following procedure illustrates the application of DeMorgan's theorems and Boolean algebra to the specific expression

$$\overline{\overline{A + BC} + \overline{D(E + F)}}$$

Step 1: Identify the terms to which you can apply DeMorgan's theorems, and think of each term as a single variable. Let $\overline{A + BC} = X$ and $\overline{D(E + F)} = Y$.

Step 2: Since $\overline{X + Y} = \overline{X}\overline{Y}$,

$$\overline{(\overline{A + BC}) + (\overline{D(E + F)})} = \overline{\overline{A + BC}}\overline{\overline{D(E + F)}}$$

Step 3: Use rule 9 ($\overline{\overline{A}} = A$) to cancel the double bars over the left term (this is not part of DeMorgan's theorem).

$$\overline{(\overline{A + BC})}\overline{\overline{D(E + F)}}$$

Step 4: Apply DeMorgan's theorem to the second term.

$$\overline{(\overline{A + BC})}\overline{\overline{D(E + F)}} = \overline{(\overline{A + BC})}(\overline{D} + \overline{E + F})$$

Step 5: Use rule 9 ($\overline{\overline{A}} = A$) to cancel the double bars over the $E + \overline{F}$ part of the term.

$$\overline{(\overline{A + BC})}(\overline{D} + \overline{E + F}) = \overline{(\overline{A + BC})}(\overline{D} + E + \overline{F})$$

The following three examples will further illustrate how to use DeMorgan's theorems.

EXAMPLE 4-5

Apply DeMorgan's theorems to each of the following expressions:

- (a) $\overline{(A + B + C)D}$
- (b) $\overline{ABC + DEF}$
- (c) $\overline{AB} + \overline{CD} + EF$

Solution

- (a) Let $A \pm B \pm C = X$ and $D = Y$. The expression $\overline{(A + B + C)}D$ is of the form $\overline{XY} = \overline{X + Y}$ and can be rewritten as

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

Next, apply DeMorgan's theorem to the term $\overline{A + B + C}$.

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

- (b) Let $ABC = X$ and $DEF = Y$. The expression $\overline{ABC} + \overline{DEF}$ is of the form $\overline{X + Y} = \overline{X} \overline{Y}$ and can be rewritten as

$$\overline{ABC} + \overline{DEF} = \overline{(ABC)} \overline{(DEF)}$$

Next, apply DeMorgan's theorem to each of the terms \overline{ABC} and \overline{DEF} .

$$\overline{(ABC)} \overline{(DEF)} = (\overline{A} + \overline{B} + \overline{C})(\overline{D} + \overline{E} + \overline{F})$$

- (c) Let $\overline{AB} = X$, $\overline{CD} = Y$, and $EF = Z$. The expression $\overline{\overline{AB} + \overline{CD} + EF}$ is of the form $\overline{X + Y + Z} = \overline{X} \overline{Y} \overline{Z}$ and can be rewritten as

$$\overline{\overline{AB} + \overline{CD} + EF} = \overline{(\overline{AB})} \overline{(\overline{CD})} \overline{(EF)}$$

Next, apply DeMorgan's theorem to each of the terms $\overline{\overline{AB}}$, $\overline{\overline{CD}}$, and \overline{EF} .

$$\overline{(\overline{AB})} \overline{(\overline{CD})} \overline{(EF)} = (\overline{A} + \overline{B})(\overline{C} + \overline{D})(\overline{E} + \overline{F})$$

EXAMPLE 4-6

Apply DeMorgan's theorems to each expression:

- (a) $\overline{(A + B) + C}$
 (b) $\overline{(\overline{A} + B) + CD}$
 (c) $\overline{(A + B)\overline{CD} + E + \overline{F}}$

Solution

- (a) $\overline{(A + B) + C} = \overline{(\overline{A + B})} \overline{C} = (A + B)\overline{C}$
 (b) $\overline{(\overline{A} + B) + CD} = \overline{(\overline{A} + B)} \overline{CD} = (\overline{\overline{A}})\overline{(C + D)} = A\overline{(C + D)}$
 (c) $\overline{(A + B)\overline{CD} + E + \overline{F}} = \overline{((A + B)\overline{CD})} \overline{(E + \overline{F})} = (\overline{A + B} + C + D)\overline{EF}$

EXAMPLE 4-7

The Boolean expression for an exclusive-OR gate is $\overline{A}B + A\overline{B}$. With this as a starting point, use DeMorgan's theorems and any other rules or laws that are applicable to develop an expression for the exclusive-NOR gate.

Solution

Start by complementing the exclusive-OR expression and then applying DeMorgan's theorems as follows:

$$\overline{\overline{A}B + A\overline{B}} = \overline{(\overline{A}B)} \overline{(A\overline{B})} = (\overline{\overline{A}} + \overline{B})(\overline{A} + \overline{\overline{B}}) = (A + \overline{B})(\overline{A} + B)$$

Next, apply the distributive law and rule 8 ($A \cdot \overline{A} = 0$).

$$(\overline{A} + B)(A + \overline{B}) = \overline{A}A + \overline{A}\overline{B} + AB + B\overline{B} = \overline{A}\overline{B} + AB$$

The final expression for the XNOR is $\overline{A}\overline{B} + AB$. Note that this expression equals 1 any time both variables are 0s or both variables are 1s.

4-4 Boolean Analysis of Logic Circuits

Boolean algebra provides a concise way to express the operation of a logic circuit formed by a combination of logic gates so that the output can be determined for various combinations of input values.

Boolean Expression for a Logic Circuit

To derive the Boolean expression for a given combinational logic circuit, begin at the left-most inputs and work toward the final output, writing the expression for each gate. For the example circuit in Figure 4-18, the Boolean expression is determined in the following three steps:

1. The expression for the left-most AND gate with inputs C and D is CD .
2. The output of the left-most AND gate is one of the inputs to the OR gate and B is the other input. Therefore, the expression for the OR gate is $B + CD$.
3. The output of the OR gate is one of the inputs to the right-most AND gate and A is the other input. Therefore, the expression for this AND gate is $A(B + CD)$, which is the final output expression for the entire circuit.

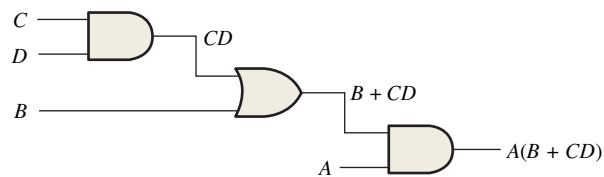


FIGURE 4-18 A combinational logic circuit showing the development of the Boolean expression for the output.

Constructing a Truth Table for a Logic Circuit

Once the Boolean expression for a given logic circuit has been determined, a truth table that shows the output for all possible values of the input variables can be developed. The procedure requires that you evaluate the Boolean expression for all possible combinations of values for the input variables. In the case of the circuit in Figure 4-18, there are four input variables (A , B , C , and D) and therefore sixteen ($2^4 = 16$) combinations of values are possible.

Evaluating the Expression

To evaluate the expression $A(B + CD)$, first find the values of the variables that make the expression equal to 1, using the rules for Boolean addition and multiplication. In this case, the expression equals 1 only if $A = 1$ and $B + CD = 1$ because

$$A(B + CD) = 1 \cdot 1 = 1$$

Now determine when the $B + CD$ term equals 1. The term $B + CD = 1$ if either $B = 1$ or $CD = 1$ or if both B and CD equal 1 because

$$B + CD = 1 + 0 = 1$$

$$B + CD = 0 + 1 = 1$$

$$B + CD = 1 + 1 = 1$$

The term $CD = 1$ only if $C = 1$ and $D = 1$.

To summarize, the expression $A(B + CD) = 1$ when $A = 1$ and $B = 1$ regardless of the values of C and D or when $A = 1$ and $C = 1$ and $D = 1$ regardless of the value of B . The expression $A(B + CD) = 0$ for all other value combinations of the variables.

Putting the Results in Truth Table Format

The first step is to list the sixteen input variable combinations of 1s and 0s in a binary sequence as shown in Table 4-5. Next, place a 1 in the output column for each combination of input variables that was determined in the evaluation. Finally, place a 0 in the output column for all other combinations of input variables. These results are shown in the truth table in Table 4-5.

TABLE 4-5
Truth table for the logic circuit in Figure 4-18.

Inputs				Output
A	B	C	D	$A(B + CD)$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

EXAMPLE 4-8

Use Multisim to generate the truth table for the logic circuit in Figure 4-18.

Solution

Construct the circuit in Multisim and connect the Multisim Logic Converter to the inputs and output, as shown in Figure 4-19. Click on the $\Rightarrow \rightarrow \overline{0|1}$ conversion bar, and the truth table appears in the display as shown.

You can also generate the simplified Boolean expression from the truth table by clicking on $\overline{0|1} \Rightarrow \text{IMP} \ A|B$.

Truth table

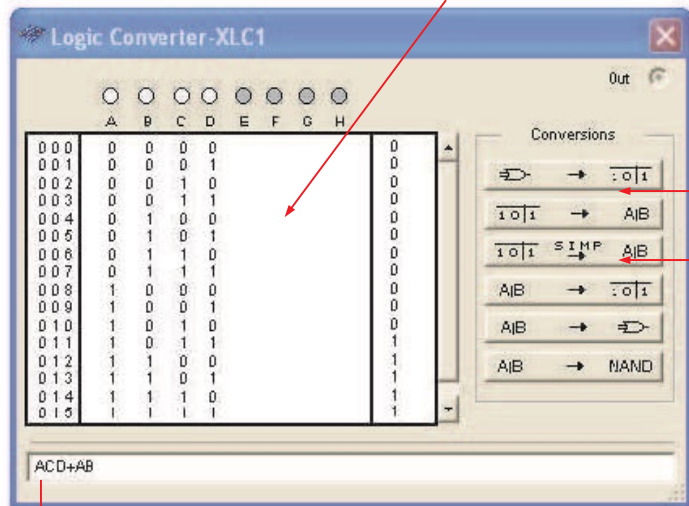
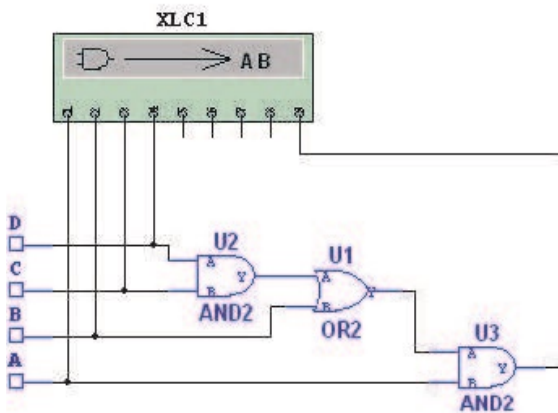


FIGURE 4-19

Boolean expression

4-5 Logic Simplification Using Boolean Algebra

A logic expression can be reduced to its simplest form or changed to a more convenient form to implement the expression most efficiently using Boolean algebra. The approach taken in this section is to use the basic laws, rules, and theorems of Boolean algebra to manipulate and simplify an expression. This method depends on a thorough knowledge of Boolean algebra and considerable practice in its application, not to mention a little ingenuity and cleverness.

A simplified Boolean expression uses the fewest gates possible to implement a given expression. Examples 4-9 through 4-12 illustrate Boolean simplification.

EXAMPLE 4-9

Using Boolean algebra techniques, simplify this expression:

$$AB + A(B + C) + B(B + C)$$

Solution

The following is not necessarily the only approach.

Step 1: Apply the distributive law to the second and third terms in the expression, as follows:

$$AB + AB + AC + BB + BC$$

Step 2: Apply rule 7 ($BB = B$) to the fourth term.

$$AB + AB + AC + B + BC$$

Step 3: Apply rule 5 ($AB + AB = AB$) to the first two terms.

$$AB + AC + B + BC$$

Step 4: Apply rule 10 ($B + BC = B$) to the last two terms.

$$AB + AC + B$$

Step 5: Apply rule 10 ($AB + B = B$) to the first and third terms.

$$B + AC$$

At this point the expression is simplified as much as possible. Once you gain experience in applying Boolean algebra, you can often combine many individual steps.

Figure 4-20 shows that the simplification process in Example 4-9 has significantly reduced the number of logic gates required to implement the expression. Part (a) shows that five gates are required to implement the expression in its original form; however, only two gates are needed for the simplified expression, shown in part (b). It is important to realize that these two gate circuits are equivalent. That is, for any combination of levels on the A , B , and C inputs, you get the same output from either circuit.

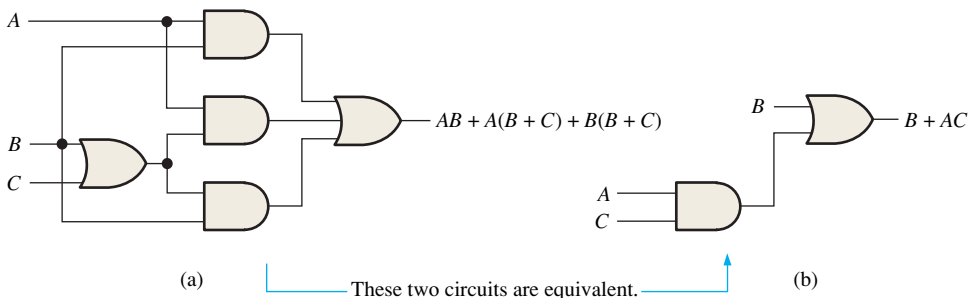


FIGURE 4-20 Gate circuits for Example 4-9. Open file F04-20 to verify equivalency.

EXAMPLE 4-10

Simplify the following Boolean expression:

$$[\overline{A}\overline{B}(C + BD) + \overline{A}\overline{B}]C$$

Note that brackets and parentheses mean the same thing: the term inside is multiplied (ANDed) with the term outside.

Solution

Step 1: Apply the distributive law to the terms within the brackets.

$$(\overline{A}\overline{B}C + \overline{A}\overline{B}BD + \overline{A}\overline{B})C$$

Step 2: Apply rule 8 ($\overline{B}B = 0$) to the second term within the parentheses.

$$(\overline{A}\overline{B}C + A \cdot 0 \cdot D + \overline{A}\overline{B})C$$

Step 3: Apply rule 3 ($A \cdot 0 \cdot D = 0$) to the second term within the parentheses.

$$(\overline{A}\overline{B}C + 0 + \overline{A}\overline{B})C$$

Step 4: Apply rule 1 (drop the 0) within the parentheses.

$$(\overline{A}\overline{B}C + \overline{A}\overline{B})C$$

Step 5: Apply the distributive law.

$$\overline{A}\overline{B}CC + \overline{A}\overline{B}C$$

Step 6: Apply rule 7 ($CC = C$) to the first term.

$$\overline{A}\overline{B}C + \overline{A}\overline{B}C$$

Step 7: Factor out $\overline{B}C$.

$$\overline{B}C(A + \overline{A})$$

Step 8: Apply rule 6 ($A + \overline{A} = 1$).

$$\overline{B}C \cdot 1$$

Step 9: Apply rule 4 (drop the 1).

$$\overline{B}C$$

EXAMPLE 4-11

Simplify the following Boolean expression:

$$\overline{A}BC + \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + ABC$$

Solution

Step 1: Factor BC out of the first and last terms.

$$BC(\overline{A} + A) + \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C$$

Step 2: Apply rule 6 ($\overline{A} + A = 1$) to the term in parentheses, and factor $\overline{A}\overline{B}$ from the second and last terms.

$$BC \cdot 1 + \overline{A}\overline{B}(\overline{C} + C) + \overline{A}\overline{B}C$$

Step 3: Apply rule 4 (drop the 1) to the first term and rule 6 ($\overline{C} + C = 1$) to the term in parentheses.

$$BC + \overline{A}\overline{B} \cdot 1 + \overline{A}\overline{B}C$$

Step 4: Apply rule 4 (drop the 1) to the second term.

$$BC + \overline{A}\overline{B} + \overline{A}\overline{B}C$$

Step 5: Factor \overline{B} from the second and third terms.

$$BC + \overline{B}(A + \overline{A}\overline{C})$$

Step 6: Apply rule 11 ($A + \overline{A}\overline{C} = A + \overline{C}$) to the term in parentheses.

$$BC + \overline{B}(A + \overline{C})$$

Step 7: Use the distributive and commutative laws to get the following expression:

$$BC + \overline{A}\overline{B} + \overline{B}\overline{C}$$

EXAMPLE 4-12

Simplify the following Boolean expression:

$$\overline{AB + AC} + \overline{A}BC$$

Solution

Step 1: Apply DeMorgan's theorem to the first term.

$$(\overline{AB})(\overline{AC}) + \overline{A}BC$$

Step 2: Apply DeMorgan's theorem to each term in parentheses.

$$(\overline{A} + \overline{B})(\overline{A} + \overline{C}) + \overline{A}BC$$

Step 3: Apply the distributive law to the two terms in parentheses.

$$\overline{A}\overline{A} + \overline{A}\overline{C} + \overline{A}B + \overline{B}\overline{C} + \overline{A}BC$$

Step 4: Apply rule 7 ($\overline{AA} = \overline{A}$) to the first term, and apply rule 10 [$\overline{A}B + A\overline{B}C = \overline{A}B(1 + C) = \overline{A}B$] to the third and last terms.

$$\overline{A} + \overline{A}\overline{C} + \overline{A}B + \overline{B}\overline{C}$$

Step 5: Apply rule 10 [$\overline{A} + \overline{A}\overline{C} = \overline{A}(1 + \overline{C}) = \overline{A}$] to the first and second terms.

$$\overline{A} + \overline{A}B + \overline{B}\overline{C}$$

Step 6: Apply rule 10 [$\overline{A} + \overline{A}B = \overline{A}(1 + B) = \overline{A}$] to the first and second terms.

$$\overline{A} + \overline{B}\overline{C}$$

4-6 Standard Forms of Boolean Expressions

All Boolean expressions, regardless of their form, can be converted into either of two standard forms: the sum-of-products form or the product-of-sums form. Standardization makes the evaluation, simplification, and implementation of Boolean expressions much more systematic and easier.

The Sum-of-Products (SOP) Form

A product term was defined in Section 4-1 as a term consisting of the product (Boolean multiplication) of literals (variables or their complements). When two or more product terms are summed by Boolean addition, the resulting expression is a **sum-of-products (SOP)**. Some examples are

$$\begin{aligned} AB + ABC \\ ABC + CDE + \overline{BCD} \\ \overline{AB} + \overline{ABC} + AC \end{aligned}$$

Also, an SOP expression can contain a single-variable term, as in $A + \overline{A}BC + BCD$. Refer to the simplification examples in the last section, and you will see that each of the final expressions was either a single product term or in SOP form. In an SOP expression, a single overbar cannot extend over more than one variable; however, more than one variable in a term can have an overbar. For example, an SOP expression can have the term \overline{ABC} but not \overline{ABC} .

Domain of a Boolean Expression

The **domain** of a general Boolean expression is the set of variables contained in the expression in either complemented or uncomplemented form. For example, the domain of the expression $\overline{AB} + \overline{ABC}$ is the set of variables A, B, C and the domain of the expression $\overline{ABC} + \overline{CDE} + \overline{BCD}$ is the set of variables A, B, C, D, E .

AND/OR Implementation of an SOP Expression

Implementing an SOP expression simply requires ORing the outputs of two or more AND gates. A product term is produced by an AND operation, and the sum (addition) of two or more product terms is produced by an OR operation. Therefore, an SOP expression can be implemented by AND-OR logic in which the outputs of a number (equal to the number of product terms in the expression) of AND gates connect to the inputs of an OR gate, as shown in Figure 4–22 for the expression $AB + BCD + AC$. The output X of the OR gate equals the SOP expression.

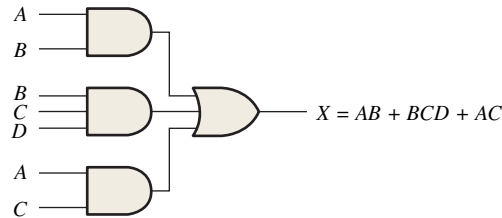


FIGURE 4–22 Implementation of the SOP expression $AB + BCD + AC$.

NAND/NAND Implementation of an SOP Expression

NAND gates can be used to implement an SOP expression. By using only NAND gates, an AND/OR function can be accomplished, as illustrated in Figure 4–23. The first level of NAND gates feed into a NAND gate that acts as a negative-OR gate. The NAND and negative-OR inversions cancel and the result is effectively an AND/OR circuit.

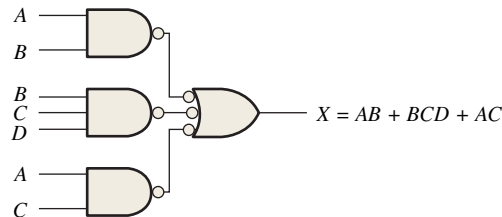


FIGURE 4–23 This NAND/NAND implementation is equivalent to the AND/OR in Figure 4–22.

Conversion of a General Expression to SOP Form

Any logic expression can be changed into SOP form by applying Boolean algebra techniques. For example, the expression $A(B + CD)$ can be converted to SOP form by applying the distributive law:

$$A(B + CD) = AB + ACD$$

EXAMPLE 4–14

Convert each of the following Boolean expressions to SOP form: _____

(a) $AB + B(CD + EF)$ (b) $(A + B)(B + C + D)$ (c) $\overline{\overline{A + B}} + C$

Solution

(a) $AB + B(CD + EF) = AB + BCD + BEF$

(b) $(A + B)(B + C + D) = AB + AC + AD + BB + BC + BD$

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

The Standard SOP Form

So far, you have seen SOP expressions in which some of the product terms do not contain all of the variables in the domain of the expression. For example, the expression $\overline{A}BC + \overline{A}BD + \overline{A}BCD$ has a domain made up of the variables A , B , C , and D . However, notice that the complete set of variables in the domain is not represented in the first two terms of the expression; that is, D or \overline{D} is missing from the first term and C or \overline{C} is missing from the second term.

A *standard SOP expression* is one in which *all* the variables in the domain appear in each product term in the expression. For example, $\overline{A}BCD + \overline{A}BC\overline{D} + \overline{A}BC\overline{D}$ is a standard SOP expression. Standard SOP expressions are important in constructing truth tables, covered in Section 4–7, and in the Karnaugh map simplification method, which is covered in Section 4–8. Any nonstandard SOP expression (referred to simply as SOP) can be converted to the standard form using Boolean algebra.

Converting Product Terms to Standard SOP

Each product term in an SOP expression that does not contain all the variables in the domain can be expanded to standard form to include all variables in the domain and their complements. As stated in the following steps, a nonstandard SOP expression is converted into standard form using Boolean algebra rule 6 ($A + \overline{A} = 1$) from Table 4–1: A variable added to its complement equals 1.

Step 1: Multiply each nonstandard product term by a term made up of the sum of a missing variable and its complement. This results in two product terms. As you know, you can multiply anything by 1 without changing its value.

Step 2: Repeat Step 1 until all resulting product terms contain all variables in the domain in either complemented or uncomplemented form. In converting a product term to standard form, the number of product terms is doubled for each missing variable, as Example 4–15 shows.

EXAMPLE 4–15

Convert the following Boolean expression into standard SOP form:

$$\overline{A}BC + \overline{A}\overline{B} + \overline{A}BC\overline{D}$$

Solution

The domain of this SOP expression is A , B , C , D . Take one term at a time. The first term, $\overline{A}BC$, is missing variable D or \overline{D} , so multiply the first term by $D + \overline{D}$ as follows:

$$\overline{A}BC = \overline{A}BC(D + \overline{D}) = \overline{A}BCD + \overline{A}BC\overline{D}$$

In this case, two standard product terms are the result.

The second term, $\overline{A}\overline{B}$, is missing variables C or \overline{C} and D or \overline{D} , so first multiply the second term by $C + \overline{C}$ as follows:

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

The two resulting terms are missing variable D or \overline{D} , so multiply both terms by $D + \overline{D}$ as follows:

$$\begin{aligned}\overline{A}\overline{B} &= \overline{A}\overline{B}C + \overline{A}\overline{B}\overline{C} = \overline{A}\overline{B}C(D + \overline{D}) + \overline{A}\overline{B}\overline{C}(D + \overline{D}) \\ &= \overline{A}\overline{B}CD + \overline{A}\overline{B}C\overline{D} + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}\overline{C}\overline{D}\end{aligned}$$

In this case, four standard product terms are the result.

The third term, $\overline{A}BC\overline{D}$, is already in standard form. The complete standard SOP form of the original expression is as follows:

$$\overline{A}BC + \overline{A}\overline{B} + \overline{A}BC\overline{D} = \overline{A}BCD + \overline{A}BC\overline{D} + \overline{A}\overline{B}CD + \overline{A}\overline{B}C\overline{D} + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}BC\overline{D}$$

Binary Representation of a Standard Product Term

A standard product term is equal to 1 for only one combination of variable values. For example, the product term $\overline{A}BC\overline{D}$ is equal to 1 when $A = 1, B = 0, C = 1, D = 0$, as shown below, and is 0 for all other combinations of values for the variables.

$$\overline{A}BC\overline{D} = 1 \cdot \overline{0} \cdot 1 \cdot \overline{0} = 1 \cdot 1 \cdot 1 \cdot 1 = 1$$

In this case, the product term has a binary value of 1010 (decimal ten).

Remember, a product term is implemented with an AND gate whose output is 1 only if each of its inputs is 1. Inverters are used to produce the complements of the variables as required.

An SOP expression is equal to 1 only if one or more of the product terms in the expression is equal to 1.

EXAMPLE 4-16

Determine the binary values for which the following standard SOP expression is equal to 1:

$$ABCD + \overline{A}\overline{B}C\overline{D} + \overline{A}\overline{B}\overline{C}\overline{D}$$

Solution

The term $ABCD$ is equal to 1 when $A = 1, B = 1, C = 1$, and $D = 1$.

$$ABCD = 1 \cdot 1 \cdot 1 \cdot 1 = 1$$

The term $\overline{A}\overline{B}C\overline{D}$ is equal to 1 when $A = 1, B = 0, C = 0$, and $D = 1$.

$$\overline{A}\overline{B}C\overline{D} = 1 \cdot \overline{0} \cdot \overline{0} \cdot 1 = 1 \cdot 1 \cdot 1 \cdot 1 = 1$$

The term $\overline{A}\overline{B}\overline{C}\overline{D}$ is equal to 1 when $A = 0, B = 0, C = 0$, and $D = 0$.

$$\overline{A}\overline{B}\overline{C}\overline{D} = \overline{0} \cdot \overline{0} \cdot \overline{0} \cdot \overline{0} = 1 \cdot 1 \cdot 1 \cdot 1 = 1$$

The SOP expression equals 1 when any or all of the three product terms is 1.

The Product-of-Sums (POS) Form

A sum term was defined in Section 4-1 as a term consisting of the sum (Boolean addition) of literals (variables or their complements). When two or more sum terms are multiplied, the resulting expression is a **product-of-sums (POS)**. Some examples are

$$\begin{aligned} &(\overline{A} + B)(A + \overline{B} + C) \\ &(\overline{A} + \overline{B} + \overline{C})(C + \overline{D} + E)(\overline{B} + C + D) \\ &(A + B)(A + \overline{B} + C)(\overline{A} + C) \end{aligned}$$

A POS expression can contain a single-variable term, as in $\overline{A}(A + \overline{B} + C)(\overline{B} + \overline{C} + D)$. In a POS expression, a single overbar cannot extend over more than one variable; however, more than one variable in a term can have an overbar. For example, a POS expression can have the term $\overline{A} + \overline{B} + \overline{C}$ but not $\overline{A + B + C}$.

Implementation of a POS Expression

Implementing a POS expression simply requires ANDing the outputs of two or more OR gates. A sum term is produced by an OR operation, and the product of two or more sum terms is produced by an AND operation. Therefore, a POS expression can be implemented by logic in which the outputs of a number (equal to the number of sum terms in the expression) of OR gates connect to the inputs of an AND gate, as Figure 4-24 shows for the expression $(A + B)(B + C + D)(A + C)$. The output X of the AND gate equals the POS expression.

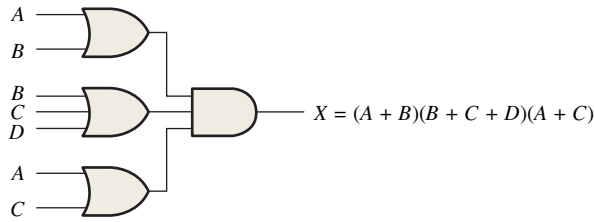


FIGURE 4-24 Implementation of the POS expression $(A + B)(B + C + D)(A + C)$.

The Standard POS Form

So far, you have seen POS expressions in which some of the sum terms do not contain all of the variables in the domain of the expression. For example, the expression

$$(A + \bar{B} + C)(A + B + \bar{D})(A + \bar{B} + \bar{C} + D)$$

has a domain made up of the variables A , B , C , and D . Notice that the complete set of variables in the domain is not represented in the first two terms of the expression; that is, D or \bar{D} is missing from the first term and C or \bar{C} is missing from the second term.

A *standard POS expression* is one in which *all* the variables in the domain appear in each sum term in the expression. For example,

$$(\bar{A} + \bar{B} + \bar{C} + \bar{D})(A + \bar{B} + C + D)(A + B + \bar{C} + D)$$

is a standard POS expression. Any nonstandard POS expression (referred to simply as POS) can be converted to the standard form using Boolean algebra.

Converting a Sum Term to Standard POS

Each sum term in a POS expression that does not contain all the variables in the domain can be expanded to standard form to include all variables in the domain and their complements. As stated in the following steps, a nonstandard POS expression is converted into standard form using Boolean algebra rule 8 ($A \cdot \bar{A} = 0$) from Table 4-1: A variable multiplied by its complement equals 0.

- Step 1:** Add to each nonstandard product term a term made up of the product of the missing variable and its complement. This results in two sum terms. As you know, you can add 0 to anything without changing its value.
- Step 2:** Apply rule 12 from Table 4-1: $A + BC = (A + B)(A + C)$
- Step 3:** Repeat Step 1 until all resulting sum terms contain all variables in the domain in either complemented or uncomplemented form.

EXAMPLE 4-17

Convert the following Boolean expression into standard POS form:

$$(A + \bar{B} + C)(\bar{B} + C + \bar{D})(A + \bar{B} + \bar{C} + D)$$

Solution

The domain of this POS expression is A , B , C , D . Take one term at a time. The first term, $A + \bar{B} + C$, is missing variable D or \bar{D} , so add $D\bar{D}$ and apply rule 12 as follows:

$$A + \bar{B} + C = A + \bar{B} + C + D\bar{D} = (A + \bar{B} + C + D)(A + \bar{B} + C + \bar{D})$$

The second term, $\bar{B} + C + \bar{D}$, is missing variable A or \bar{A} , so add $A\bar{A}$ and apply rule 12 as follows:

$$\bar{B} + C + \bar{D} = \bar{B} + C + \bar{D} + A\bar{A} = (A + \bar{B} + C + \bar{D})(\bar{A} + \bar{B} + C + \bar{D})$$

The third term, $A + \bar{B} + \bar{C} + D$, is already in standard form. The standard POS form of the original expression is as follows:

$$(A + \bar{B} + C)(\bar{B} + C + \bar{D})(A + \bar{B} + \bar{C} + D) = (A + \bar{B} + C + D)(A + \bar{B} + C + \bar{D})(A + \bar{B} + C + \bar{D})(\bar{A} + \bar{B} + C + \bar{D})(A + \bar{B} + \bar{C} + D)$$

Binary Representation of a Standard Sum Term

A standard sum term is equal to 0 for only one combination of variable values. For example, the sum term $A + \bar{B} + C + D$ is 0 when $A = 0$, $B = 1$, $C = 0$, and $D = 1$, as shown below, and is 1 for all other combinations of values for the variables.

$$A + \bar{B} + C + \bar{D} = 0 + \bar{1} + 0 + \bar{1} = 0 + 0 + 0 + 0 = 0$$

In this case, the sum term has a binary value of 0101 (decimal 5). Remember, a sum term is implemented with an OR gate whose output is 0 only if each of its inputs is 0. Inverters are used to produce the complements of the variables as required.

A POS expression is equal to 0 only if one or more of the sum terms in the expression is equal to 0.

EXAMPLE 4-18

Determine the binary values of the variables for which the following standard POS expression is equal to 0:

$$(A + B + C + D)(A + \bar{B} + \bar{C} + D)(\bar{A} + \bar{B} + \bar{C} + \bar{D})$$

Solution

The term $A + B + C + D$ is equal to 0 when $A = 0$, $B = 0$, $C = 0$, and $D = 0$.

$$A + B + C + D = 0 + 0 + 0 + 0 = 0$$

The term $A + \bar{B} + \bar{C} + D$ is equal to 0 when $A = 0$, $B = 1$, $C = 1$, and $D = 0$.

$$A + \bar{B} + \bar{C} + D = 0 + \bar{1} + \bar{1} + 0 = 0 + 0 + 0 + 0 = 0$$

The term $\bar{A} + \bar{B} + \bar{C} + \bar{D}$ is equal to 0 when $A = 1$, $B = 1$, $C = 1$, and $D = 1$.

$$\bar{A} + \bar{B} + \bar{C} + \bar{D} = \bar{1} + \bar{1} + \bar{1} + \bar{1} = 0 + 0 + 0 + 0 = 0$$

The POS expression equals 0 when any of the three sum terms equals 0.

Converting Standard SOP to Standard POS

The binary values of the product terms in a given standard SOP expression are not present in the equivalent standard POS expression. Also, the binary values that are not represented in the SOP expression are present in the equivalent POS expression. Therefore, to convert from standard SOP to standard POS, the following steps are taken:

Step 1: Evaluate each product term in the SOP expression. That is, determine the binary numbers that represent the product terms.

Step 2: Determine all of the binary numbers not included in the evaluation in Step 1.

Step 3: Write the equivalent sum term for each binary number from Step 2 and express in POS form.

EXAMPLE 4-19

Using a similar procedure, you can go from POS to SOP.

Convert the following SOP expression to an equivalent POS expression:

$$\bar{A}\bar{B}\bar{C} + \bar{A}BC + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$$

Solution

The evaluation is as follows:

$$000 + 010 + 011 + 101 + 111$$

Since there are three variables in the domain of this expression, there are a total of eight (2^3) possible combinations. The SOP expression contains five of these combinations, so the POS must contain the other three which are 001, 100, and 110. Remember, these are the binary values that make the sum term 0. The equivalent POS expression is

$$(A + B + \bar{C})(\bar{A} + B + C)(\bar{A} + \bar{B} + C)$$