4 BOOLEAN ALGEBRA
AND
LOGIC SIMPLIFICATION

BOOLEAN OPERATIONS AND EXPRESSIONS

Variable, complement, and literal are terms used in Boolean algebra. A variable is a symbol used to represent a logical quantity. Any single variable can have a 1 or a 0 value. The complement is the inverse of a variable and is indicated by a bar over variable (overbar). For example, the complement of the variable A is $\overline{A}$. If A = 1, then $\overline{A} = 0$. If A = 0, then $\overline{A} = 1$. 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. A literal is a variable or the complement of a variable.

Boolean Addition

Recall from part 3 that Boolean addition is equivalent to the OR operation. 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 $A + B$, $A + \overline{B}$, $A + B + \overline{C}$, and $\overline{A} + B + C + \overline{D}$.

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

Determine the values of A, B, C, and D that make the sum term

$A + \overline{B} + C + \overline{D}$ equal to 0.

Boolean Multiplication

Also recall from part 3 that Boolean multiplication is equivalent to the AND operation. 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, $A\overline{B}$, ABC, and $\overline{A}BC\overline{D}$.

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

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

LAWS AND RULES OF BOOLEAN ALGEBRA

■ ***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.

Commutative Laws

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

A+B = B+A

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. Fig.(4-1) 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 ≡ means "equivalent to.").

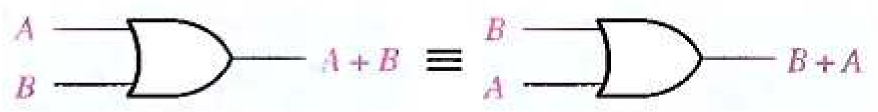

Fig.(4-1) Application of commutative law of addition.

►The commutative law of multiplication for two variables is

A.B = B.A

This law states that the order in which the variables are ANDed makes no difference. Fig.(4-2), il1ustrates this law as applied to the AND gate.

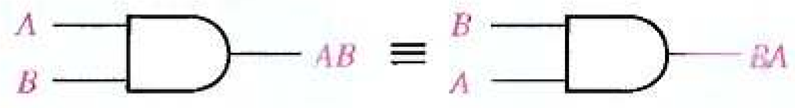

Fig.(4-2) Application of commutative law of multiplication.

Associative Laws :

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

A + (B + C) = (A + B) + C

This law states that when ORing more than two variables, the result is the same regardless of the grouping of the variables. Fig.(4-3), illustrates this law as applied to 2-input OR gates.

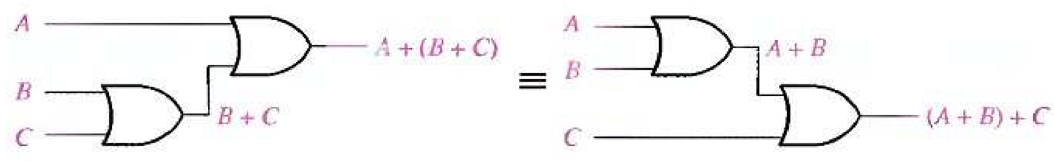

Fig.(4-3) Application of associative law of addition.

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

A(BC) = (AB)C

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

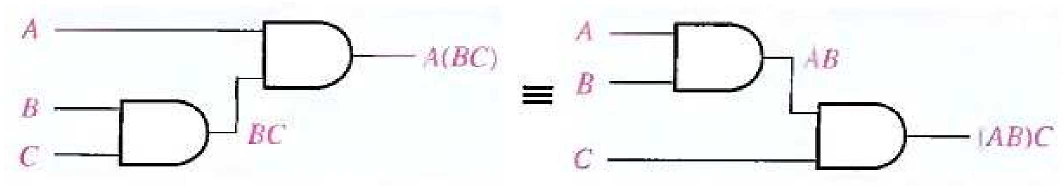

Fig.(4-4) Application of associative law of multiplication.

Distributive Law:

►The distributive law is written for three variables as follows:

A(B + C) = AB + AC

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).

Fig.(4-5) illustrates the distributive law in terms of gate implementation.

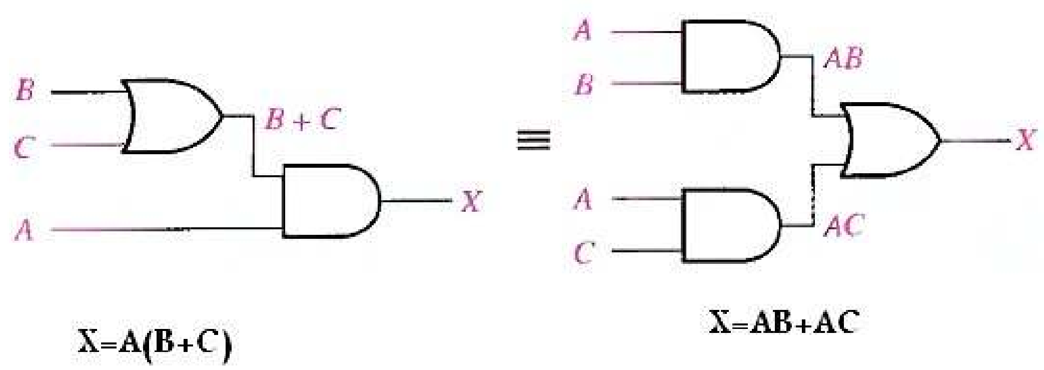

Fig.(4-5) Application of distributive law.

■ ***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.** $A \cdot \overline{A} = 0$
3. $A \cdot 0 = 0$	**9.** $\overline{\overline{A}} = A$
4. $A \cdot 1 = A$	**10.** $A + AB = A$
5. $A + A = A$	**11.** $A + \overline{A}B = A + B$
6. $A + \overline{A} = 1$	**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 Fig.(4-6), where the lower input is fixed at 0.

A+0=A

Fig.(4-6)