BOOLEAN ALGEBRA

2.1 Introduction

Binary logic deals with variables that have two discrete values: 1 for *TRUE* and 0 for *FALSE*. A simple switching circuit containing active elements such as a diode and transistor can demonstrate the binary logic, which can either be ON (switch closed) or OFF (switch open). Electrical signals such as voltage and current exist in the digital system in either one of the two recognized values, except during transition.

The switching functions can be expressed with Boolean equations. Complex Boolean equations can be simplified by a new kind of algebra, which is popularly called Switching Algebra or Boolean Algebra, invented by the mathematician George Boole in 1854. Boolean Algebra deals with the rules by which logical operations are carried out.

2.2 BASIC DEFINITIONS

Boolean algebra, like any other deductive mathematical system, may be defined with a set of elements, a set of operators, and a number of assumptions and postulates. A set of elements means any collection of objects having common properties. If S denotes a set, and X and Y are certain objects, then $X \in S$ denotes X is an object of set S, whereas $Y \notin S$ denotes Y is not the object of set S. A binary operator defined on a set S of elements is a rule that assigns to each pair of elements from S a unique element from S. As an example, consider this relation X*Y = Z. This implies that * is a binary operator if it specifies a rule for finding Z from the objects (X, Y) and also if all X, Y, and Z are of the same set S. On the other hand, * can not be binary operator if X and Y are of set S and Z is not from the same set S.

The postulates of a mathematical system are based on the basic assumptions, which make possible to deduce the rules, theorems, and properties of the system. Various algebraic structures are formulated on the basis of the most common postulates, which are described as follows:

1. Closer: A set is closed with respect to a binary operator if, for every pair of elements of S, the binary operator specifies a rule for obtaining a unique element of S. For example, the set of natural numbers N = {1, 2, 3, 4, ...} is said to be closed with respect to the binary operator plus (+) by the rules of arithmetic addition, since for any $X,Y \in N$ we obtain a unique element $Z \in N$ by the operation X + Y = Z. However, note that the set of natural numbers is not closed with respect to the binary operator minus (–) by the rules of arithmetic subtraction because for 1 – 2 = –1, where –1 is not of the set of naturals numbers.

2. Associative Law: A binary operator * on a set S is said to be associated whenever

(A*B)*C = A*(B*C) for all A,B,C ∈ S.

3. Commutative Law: A binary operator * on a set S is said to be commutative Whenever A*B = B*A for all A,B ∈ S.

4. Identity Element: A set S is to have an identity element with respect to a binary operation * on S, if there exists an element E ∈ S with the property E*A = A*E = A.

Example: The element 0 is an identity element with respect to the binary operator + on the set of integers I = {.... –4, –3, –2, –1, 0, 1, 2, 3, 4, ...} as
A + 0 = 0 + A = A.
Similarly, the element 1 is the identity element with respect to the binary operator × as A × 1 = 1 × A = A.

5. Inverse: If a set S has the identity element E with respect to a binary operator *, there exists an element B ∈ S, which is called the inverse, for every A ∈ S, such that A*B = E.

Example: In the set of integers I with E = 0, the inverse of an element A is (-A) since A + (–A) = 0.

6. Distributive Law: If * and (.) are two binary operators on a set S, * is said to be distributive over (.), whenever
A*(B.C) = (A*B).(A*C).
If summarized, for the field of real numbers, the operators and postulates have the following meanings:
The binary operator + defines addition.
The additive identity is 0.
The additive inverse defines subtraction.
The binary operator (.) defines multiplication.
The multiplication identity is 1.
The multiplication inverse of A is 1/A, defines division *i.e.*, A. 1/A = 1.
The only distributive law applicable is that of (.) over +
A . (B + C) = (A . B) + (A . C)

2.3 DEFINITION OF BOOLEAN ALGEBRA

In 1854 George Boole introduced a systematic approach of logic and developed an algebraic system to treat the logic functions, which is now called Boolean algebra. In 1938 C.E. Shannon developed a two-valued Boolean algebra called Switching algebra, and demonstrated that the properties of two-valued or bistable electrical switching circuits can be represented by this algebra. The postulates formulated by E.V. Huntington in 1904 are employed for the formal definition of Boolean algebra. However, Huntington postulates are not unique for defining Boolean algebra and other postulates are also used. The following Huntington postulates are satisfied for the definition of Boolean algebra on a set of elements S together with two binary operators (+) and (.).

1. (*a*) Closer with respect to the operator (+).

(*b*) Closer with respect to the operator (.).

2. (*a*) An identity element with respect to + is designated by 0 *i.e.*, A + 0 = 0 + A = A.

(*b*) An identity element with respect to . is designated by 1 *i.e.*, A.1 = 1. A = A.

3. (*a*) Commutative with respect to (+), *i.e.*, A + B = B + A.

(*b*) Commutative with respect to (.), *i.e.*, A.B = B.A.

4. (*a*) (.) is distributive over (+), *i.e.*, A . (B+C) = (A . B) + (A . C).

(*b*) (+) is distributive over (.), *i.e.*, A + (B .C) = (A + B) . (A + C).

5. For every element A ∈ S, there exists an element A′ ∈ S (called the complement of A) such that A + A′ = 1 and A . A′ = 0.

6. There exists at least two elements A,B ∈ S, such that A is not equal to B.

Comparing Boolean algebra with arithmetic and ordinary algebra (the field of real numbers), the following differences are observed:

1. Huntington postulates do not include the associate law. However, Boolean algebra follows the law and can be derived from the other postulates for both operations.

2. The distributive law of (+) over (.) *i.e.*, A+ (B.C) = (A+B) . (A+C) is valid for Boolean algebra, but not for ordinary algebra.

3. Boolean algebra does not have additive or multiplicative inverses, so there are no subtraction or division operations.

4. Postulate 5 defines an operator called Complement, which is not available in ordinary algebra.

5. Ordinary algebra deals with real numbers, which consist of an infinite set of elements. Boolean algebra deals with the as yet undefined set of elements S, but in the two valued Boolean algebra, the set S consists of only two elements: 0 and 1.

Boolean algebra is very much similar to ordinary algebra in some respects. The symbols (+) and (.) are chosen intentionally to facilitate Boolean algebraic manipulations by persons already familiar to ordinary algebra. Although one can use some knowledge from ordinary algebra to deal with Boolean algebra, beginners must be careful not to substitute the rules of ordinary algebra where they are not applicable.

2.5.2 DeMorgan's Theorem

Two theorems that were proposed by DeMorgan play important parts in Boolean algebra.

The first theorem states that the complement of a product is equal to the sum of the complements. That is, if the variables are A and B, then:

$$(A.B)' = A' + B'$$

The second theorem states that the complement of a sum is equal to the product of the complements. In equation form, this can be expressed as:

$$(A + B)' = A' . B'$$

The complements of Boolean logic function or a logic expression may be simplified or expanded by the following steps of DeMorgan's theorem.

(*a*) Replace the operator (+) with (.) and (.) with (+) given in the expression.

(*b*) Complement each of the terms or variables in the expression.

DeMorgan's theorems are applicable to any number of variables. For three variables A, B, and C, the equations are:

$$(A.B.C)' = A' + B' + C' \text{ and}$$

$$(A + B + C)' = A'.B'.C'$$

2.5.3 Other Important Theorems

Theorem 1(*a*): $A + A = A$

$A + A = (A + A).1$	by postulate 2(*b*)
$= (A + A) . (A + A')$	by postulate 5
$= A + A.A'$	
$= A + 0$	by postulate 4
$= A$	by postulate 2(*a*)

Theorem 1(*b*): $A . A = A$

$A . A = (A . A) + 0$	by postulate 2(a)
$= (A . A) + (A . A')$	by postulate 5
$= A (A + A')$	
$= A . 1$	by postulate 4
$= A$	by postulate 2(b)

Theorem 2(*a*): $A + 1 = 1$

Theorem 2(*b*): $A . 0 = 0$

Theorem 3(*a*): $A + A.B = A$

$A + A.B = A . 1 + A.B$	by postulate 2(b)
$= A (1 + B)$	by postulate 4(a)
$= A . 1$	by postulate 2(a)
$= A$	by postulate 2(b)

Theorem 3(*b*): $A (A + B) = A$ by duality

The following is the complete list of postulates and theorems useful for two-valued Boolean algebra.

Postulate 2	(*a*) A + 0 = A	(*b*) A.1 = A
Postulate 5	(*a*) A + A′ = 1	(*b*) A.A′ = 0
Theorem 1	(*a*) A + A = A	(*b*) A.A = A
Theorem 2	(*a*) A + 1 = 1	(*b*) A.0 = 0
Theorem 3, Involution	(A′)′ = A	
Theorem 3, Commutative	(*a*) A + B = B + A	(*b*) A.B = B.A
Theorem 4, Associative	(*a*) A + (B + C) = (A + B) + C	(*b*) A.(B.C) = (A.B).C
Theorem 4, Distributive	(*a*) A(B + C) = A.B + A.C	(*b*) A + B.C = (A + B).(A + C)
Theorem 5, DeMorgan	(*a*) (A + B)′ = A′.B′	(*b*) (A.B)′ = A′ + B′
Theorem 6, Absorption	(*a*) A + A.B = A	(*b*) A.(A + B) = A

Figure 2-6

2.6 BOOLEAN FUNCTIONS

Binary variables have two values, either 0 or 1. A Boolean function is an expression formed with binary variables, the two binary operators AND and OR, one unary operator NOT, parentheses and equal sign. The value of a function may be 0 or 1, depending on the values of variables present in the Boolean function or expression. For example, if a Boolean function is expressed algebraically as:

F = AB′C

then the value of F will be 1, when A = 1, B = 0, and C = 1. For other values of A, B, C the value of F is 0.

Boolean functions can also be represented by truth tables. A *truth table* is the tabular form of the values of a Boolean function according to the all possible values of its variables. For an *n* number of variables, 2^n combinations of 1s and 0s are listed and one column represents function values according to the different combinations. For example, for three variables the Boolean function F = AB + C truth table can be written as below in Figure 2.7

.

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Figure 2-7

A Boolean function from an algebraic expression can be realized to a logic diagram composed of logic gates. Figure 2.8 is an example of a logic diagram realized by the basic gates like AND, OR, and NOT gates.

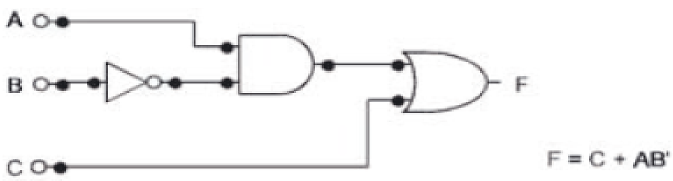

Figure 2-8

2.7 SIMPLIFICATION OF BOOLEAN EXPRESSIONS

When a Boolean expression is implemented with logic gates, each literal in the function is designated as input to the gate. The literal may be a primed or unprimed variable. Minimization of the number of literals and the number of terms leads to less complex circuits as well as less number of gates, which should be a designer's aim. There are several methods to minimize the Boolean function. Here, simplification or minimization of complex algebraic expressions will be shown with the help of postulates and theorems of Boolean algebra.

Example 2.1. *Simplify the Boolean function F=AB+ BC + B′C.*

Solution. F = AB + BC + B′C

= AB + C(B + B′)

= AB + C

Example 2.2. *Simplify the Boolean function F= A + A′B.*

Solution. F = A+ A′B

= (A + A′) (A + B)

= A + B

Example 2.3. *Simplify the Boolean function* $F= A'B'C + A'BC + AB'$.

Solution. $F = A'B'C + A'BC + AB'$

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

$= A'C + AB'$

Example 2.4. *Simplify the Boolean function* $F = AB + (AC)' + AB'C(AB + C)$.

Solution. $F = AB + (AC)' + AB'C(AB + C)$

$= AB + A' + C'+ AB'C.AB + AB'C.C$

$= AB + A' + C' + 0 + AB'C$ ($B.B' = 0$ and $C.C = C$)

$= ABC + ABC' + A' + C' + AB'C$ ($AB = AB(C + C') = ABC + ABC'$)

$= AC(B + B') + C'(AB + 1) + A'$

$= AC + C'+A'$ ($B + B' = 1$ and $AB + 1 = 1$)

$= AC + (AC)'$

$= 1$