5.0 BOOLEAN ALGEBRA AND SIMPLIFICATION TECHNIQUES

Boolean algebra is mathematics of logic. It is one of the most basic tools available to the logic designer and thus can be effectively used for simplification of complex logic expressions. Other useful and widely used techniques based on Boolean theorems(identities) include the use of Karnaugh maps in what is known as the mapping method of logic simplification and the tabular method given by Quine–McCluskey. In this chapter, we will have a closer look at the different postulates and theorems of Boolean algebra and their applications in minimizing Boolean expressions. We will also discuss the mapping method of minimizing fairly complex and large logic expressions.

6.1 Introduction to Boolean Algebra

Boolean algebra, quite interestingly, is simpler than ordinary algebra. It is also composed of a set of symbols and a set of rules to manipulate these symbols. However, this is the only similarity between the two. The differences are many. These include the following:

1. In ordinary algebra, the letter symbols can take on any number of values including infinity. In Boolean algebra, they can take on either of two values, that is, 0 and 1.
2. The values assigned to a variable have a numerical significance in ordinary algebra, whereas in its Boolean counterpart they have a logical significance.
3. While '.' and '+' are respectively the signs of multiplication and addition in ordinary algebra, in Boolean algebra '.' means an AND operation and '+' means an OR operation. For instance, A+B in ordinary algebra is read as A plus B, while the same in Boolean algebra is read as A OR B. More specifically, Boolean algebra captures the essential properties of both logic operations such as AND, OR and NOT and set operations such as intersection, union and complement. As an illustration, the logical assertion that both a statement and its negation cannot be true has a counterpart in set theory, which says that the intersection of a subset and its complement is a null (or empty) set.
5. Boolean algebra may also be defined to be a set A supplied with two binary operations of logical AND ($\wedge$), logical OR (V), a unary operation of logical NOT ($\neg$) and two elements, namely logical FALSE (0) and logical TRUE (1). This set is such that, for all elements of this set, the postulates or axioms relating to the associative, commutative, distributive, absorption and complementation properties of these elements hold good. These postulates are described in the following pages.

6.1.1 Variables, Literals and Terms in Boolean Expressions

Variables are the different symbols in a Boolean expression. They may take on the value '0' or '1'.
For instance, in expression (6.1), A, B and C are the three variables. In expression (6.2), P, Q, R and S are the variables:

$$\overline{A}+A.B+A.\overline{C}+\overline{A}.B.C \qquad (6.1)$$

$$(\overline{P}+Q).(R+\overline{S}).(P+\overline{Q}+R) \qquad (6.2)$$

The complement of a variable is not considered as a separate variable. Each occurrence of a variable or its complement is called a *literal*. In expressions (6.1) and (6.2) there are eight and seven literals respectively. A term is the expression formed by literals and operations at one level. Expression (6.1) has five terms including four AND terms and the OR term that combines the first-level AND terms.

6.1.2 Equivalent and Complement of Boolean Expressions

Two given Boolean expressions are said to be *equivalent* if one of them equals '1' only when the other equals '1' and also one equals '0' only when the other equals '0'. They are said to be the *complement* of each other if one expression equals '1' only when the other equals '0', and vice versa.
The complement of a given Boolean expression is obtained by complementing each literal, changing all '.' to '+' and all '+' to '.', all 0s to 1s and all 1s to 0s. The examples below give some Boolean expressions and their complements:
Given Boolean expression

$$\overline{A}.B + A.\overline{B} \tag{6.3}$$

Corresponding complement

$$(A + \overline{B}).(\overline{A} + B) \tag{6.4}$$

Given Boolean expression

$$(A + B).(\overline{A} + \overline{B}) \tag{6.5}$$

Corresponding complement

$$\overline{A}.\overline{B} + A.B \tag{6.6}$$

When ORed with its complement the Boolean expression yields a '1', and when ANDed with its complement it yields a '0'. The '.' sign is usually omitted in writing Boolean expressions and is implied merely by writing the literals in juxtaposition. For instance, A.B would normally be written as AB.

6.1.3 Dual of a Boolean Expression

The dual of a Boolean expression is obtained by replacing all '.' operations with '+' operations, all '+' operations with '.' operations, all 0s with 1s and all 1s with 0s and leaving all literals unchanged. The examples below give some Boolean expressions and the corresponding dual expressions:

Given Boolean expression

$$\overline{A}.B + A.\overline{B} \tag{6.7}$$

Corresponding dual

$$(\overline{A} + B).(A + \overline{B}) \tag{6.8}$$

Given Boolean expression

$$(A + B).(\overline{A} + \overline{B}) \tag{6.9}$$

Corresponding dual

$$A.B + \overline{A}.\overline{B} \tag{6.10}$$

Duals of Boolean expressions are mainly of interest in the study of Boolean postulates and theorems. Otherwise, there is no general relationship between the values of dual expressions. That is, both of them may equal '1' or '0'. One may even equal '1' while the other equals '0'. The fact that the dual of a given logic equation is also a valid logic equation leads to many more useful laws of Boolean algebra. The principle of duality has been put to ample use during the discussion on postulates and theorems of Boolean algebra. The postulates and theorems, to be discussed in the paragraphs to follow, have been presented in pairs, with one being the dual of the other.

Example 6.1

Find (a) the dual of $A.B + B.C + C.D$ *and (b) the complement of* $[(A.B + C).D + E].F$.

Solution

(a) The dual of $A.\overline{B} + B.\overline{C} + C.\overline{D}$ is given by $(A + \overline{B}).(B + \overline{C}).(C + \overline{D})$.

(b) The complement of $[(A.\overline{B} + \overline{C}).D + \overline{E}].F$ is given by $[(\overline{A} + B).C + \overline{D}].E + \overline{F}$.

Example 6.2

Simplify $(A.B+C.D).[(\overline{A}+\overline{B}).(\overline{C}+\overline{D})]$.

Solution

- Let $(A.B+C.D)=X$.
- Then the given expression reduces to $X.\overline{X}$.
- Therefore, $(A.B+C.D).[(\overline{A}+\overline{B}).(\overline{C}+\overline{D})]=0$.

6.2 Postulates of Boolean Algebra

The following are the important postulates of Boolean algebra:

1. $1.1=1, 0+0=0$.
2. $1.0=0.1=0, 0+1=1+0=1$.
3. $0.0=0, 1+1=1$.
4. $\overline{1}=0$ and $\overline{0}=1$.

Many theorems of Boolean algebra are based on these postulates, which can be used to simplify Boolean expressions. These theorems are discussed in the next section.

3.2.2 Boolean Identities

Frequently, a Boolean expression is not in its simplest form. Recall from algebra that an expression such as $2x + 6x$ is not in its simplest form; it can be reduced (represented by fewer or simpler terms) to $8x$. Boolean expressions can also be simplified, but we need new *identities*, or laws, that apply to Boolean algebra instead of regular algebra. These identities, which apply to single Boolean variables as well as Boolean expressions, are listed in Table 3.5. Note that each relationship (with the exception of the last one) has both an AND (or product) form and an OR (or sum) form. This is known as the *duality principle*. The Identity Law states that any Boolean variable ANDed with 1 or ORed with 0 simply results in the original variable. (1 is the identity element for AND; 0 is the identity element for OR.) The Null Law states that any Boolean variable ANDed with 0 is 0, and a variable ORed with 1 is always 1. The Idempotent Law states that ANDing or ORing a variable with itself produces the original variable. The Inverse Law states that ANDing or ORing a variable with its complement produces the identity for that given operation.

Identity Name	AND Form	OR Form
Identity Law	$1x = x$	$0+x = x$
Null (or Dominance) Law	$0x = 0$	$1+x = 1$
Idempotent Law	$xx = x$	$x+x = x$
Inverse Law	$x\overline{x} = 0$	$x+\overline{x} = 1$
Commutative Law	$xy = yx$	$x+y = y+x$
Associative Law	$(xy)z = x(yz)$	$(x+y)+z = x+(y+z)$
Distributive Law	$x+yz = (x+y)(x+z)$	$x(y+z) = xy+xz$
Absorption Law	$x(x+y) = x$	$x+xy = x$
DeMorgan's Law	$(\overline{xy}) = \overline{x}+\overline{y}$	$(\overline{x+y}) = \overline{x}\overline{y}$
Double Complement Law	$\overline{\overline{x}} = x$	

TABLE 3.5 Basic Identities of Boolean Algebra

You should recognize the Commutative Law and Associative Law from algebra. Boolean variables can be reordered (commuted) and regrouped (associated) without affecting the final result. The Distributive Law shows how OR distributes over AND and vice versa. The Absorption Law and DeMorgan's Law are not so obvious, but we can prove these identities by creating a truth table for the various expressions: If the right-hand side is equal to the left-hand side, the expressions represent the same function and result in identical truth tables. Table 3.6 depicts the truth table for both the left-hand side and the right-hand side of DeMorgan's Law for AND. It is left as an exercise to prove the validity of the remaining laws, in particular, the OR form of DeMorgan's Law and both forms of the Absorption Law. The Double Complement Law formalizes the idea of the double negative i.e. $x = (\bar{\bar{x}})$.

x	y	(xy)	$(\bar{x}\bar{y})$	$\bar{x}$	$\bar{y}$	$\bar{x}+\bar{y}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

TABLE 3.6 Truth Tables for the AND Form of DeMorgan's Law

3.2.3 Simplification of Boolean Expressions using Boolean identities

The algebraic identities we studied in algebra class allow us to reduce algebraic expressions (such as $10x + 2y - x + 3y$) to their simplest forms ($9x + 5y$). The Boolean identities can be used to simplify Boolean expressions in a similar fashion.

The identities are presented as pairs, with the two identities in a given pair being the dual of each other.

These identities can be very easily verified by the method of 'perfect induction'. According to this method, the validity of the expression is tested for all possible combinations of values of the variables involved. Also, since the validity of the theorem is based on its being true for all possible combinations of values of variables, there is no reason why a variable cannot be replaced with its complement, or vice versa, without disturbing the validity. Another important point is that, if a given expression is valid, its dual will also be valid.

We apply these identities in the following examples.

EXAMPLE 3.1 Suppose we have the function $F(x,y) = xy + xy$. Using the OR form of the Idempotent Law and treating the expression xy as a Boolean variable, we simplify the original expression to xy. Therefore, $F(x,y) = xy + xy = xy$.

EXAMPLE 3.2 Given the function $F(x,y,z) = \bar{x}yz + \bar{x}y\bar{z} + xz,$ we simplify as follows:

$$
\begin{aligned}
F(x,y,z) &= \bar{x}yz + \bar{x}y\bar{z} + xz \\
&= \bar{x}y(z + \bar{z}) + xz && \text{(Distributive)} \\
&= \bar{x}y(1) + xz && \text{(Inverse)} \\
&= \bar{x}y + xz && \text{(Identity)}
\end{aligned}
$$

At times, the simplification is reasonably straightforward, as in the preceding examples. However, using the identities can be tricky, as we see in this next example.

EXAMPLE 3.3 Given the function $F(x,y,z) = xy + \bar{x}z + yz$, we simplify as follows:

$= xy + \bar{x}z + yz(1)$ (Identity)
$= xy + \bar{x}z + yz(x + \bar{x})$ (Inverse)
$= xy + \bar{x}z + (yz)x + (yz)\bar{x}$ (Distributive)
$= xy + \bar{x}z + x(yz) + \bar{x}(zy)$ (Commutative)
$= xy + \bar{x}z + (xy)z + (\bar{x}z)y$ (Associative)
$= xy + (xy)z + \bar{x}z + (\bar{x}z)y$ (Commutative)
$= xy(1 + z) + \bar{x}z(1 + y)$ (Distributive)
$= xy(1) + \bar{x}z(1)$ (Null)
$= xy + \bar{x}z$ (Identity)

Example 3.3 illustrates what is commonly known as the *Consensus Theorem.* How did we know to insert additional terms to simplify the function? Unfortunately, there is no defined set of rules for using these identities to minimize a Boolean expression; it is simply something that comes with experience. There are other methods that can be used to simplify Boolean expressions; we mention these later in this section.

Proof	Identity Name
$(x+y)(\bar{x}+y) = x\bar{x}+xy+y\bar{x}+yy$	Distributive Law
$= 0+xy+y\bar{x}+yy$	Inverse Law
$= 0+xy+y\bar{x}+y$	Idempotent Law
$= xy+y\bar{x}+y$	Identity Law
$= y(x+\bar{x})+y$	Distributive Law (and Commutative Law)
$= y(1)+y$	Inverse Law
$= y+y$	Identity Law
$= y$	Idempotent Law

TABLE 3.7 Example Using Identities

We can also use these identities to prove Boolean equalities. Suppose we want to prove that $(x + y)(\bar{x} + y) = y$. The proof is given in Table 3.7. To prove the equality of two Boolean expressions, you can also create the truth tables for each and compare. If the truth tables are identical, the expressions are equal. This is called perfect induction method. We leave it as an exercise to find the truth tables for the equality in Table 3.7.

6.4 Representing Boolean Functions

We have seen that there are many different ways to represent a given Boolean function. For example, we can use a truth table or we can use one of many different Boolean expressions. In fact, there are an infinite number of Boolean expressions that are *logically equivalent* to one another. Two expressions that can be represented by the same truth table are considered logically equivalent. See Example 3.4.

EXAMPLE 3.4 Suppose $F(x,y,z) = x + x\bar{y}$. We can also express F as $F(x,y,z) = x + x + x\bar{y}$ because the Idempotent Law tells us these two expressions are the same. We can also express F as $F(x,y,z) = x(1 + \bar{y})$ using the Distributive Law.

To help eliminate potential confusion, logic designers specify a Boolean function using a *canonical*, or *standardized*, form. For any given Boolean function, there exists a unique standardized form. However, there are different "standards" that designers use. The two most common are the sum-of-products form and the product- of-sums form. The given Boolean expression will be in either of the two forms, and the objective will be to find a minimized expression in the same or the other form.

6.4.1 Sum-of-Products Boolean Expressions

A sum-of-products expression contains the sum of different terms, with each term being either a single literal or a product of more than one literal. It can be obtained from the truth table directly by considering those input combinations that produce a logic '1' at the output. Each such input combination produces a term. Different terms are given by the product of the corresponding literals.

The sum of all terms gives the expression. For example, the truth table in Table 6.5 can be represented by the Boolean expression

$$Y = \overline{A}\ .\overline{B}.\overline{C} + \overline{A}.B.C + A.B.\overline{C} + A.\overline{B}.C \tag{6.33}$$

Considering the first term, the output is '1' when $A = 0$, $B = 0$ and $C = 0$. This is possible only when $\overline{A}$, $\overline{B}$ and $\overline{C}$ are ANDed. Also, for the second term, the output is '1' only when B, C and $\overline{A}$ are ANDed. Other terms can be explained similarly. A sum-of-products expression is also known as a *minterm expression*.

Table 6.5 truth table of boolean expression of equation 6.33.

A	*B*	*C*	*Y*
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

6.4.2 Product-of-Sums Expressions

A product-of-sums expression contains the product of different terms, with each term being either a single literal or a sum of more than one literal. It can be obtained from the truth table by considering those input combinations that produce a logic '0' at the output. Each such input combination gives a term, and the product of all such terms gives the expression. Different terms are obtained by taking the sum of the corresponding literals. Here, '0' and '1' respectively mean the uncomplemented and complemented variables, unlike sum-of-products expressions where '0' and '1' respectively mean complemented and uncomplemented variables.

To illustrate this further, consider once again the truth table in Table 6.5. Since each term in the case of the product-of-sums expression is going to be the sum of literals, this implies that it is going to be implemented using an OR operation. Now, an OR gate produces a logic '0' only when all its inputs are in the logic '0' state, which means that the first term corresponding to the second row of the truth table will be $A+B+\overline{C}$. The product-of-sums Boolean expression for this truth table is given by $(A+B+\overline{C}).(A+\overline{B}+C).(\overline{A}+B+C).(\overline{A}+\overline{B}+\overline{C})$.

Transforming the given product-of-sums expression into an equivalent sum-of-products expression is a straightforward process. Multiplying out the given expression and carrying out the obvious simplification provides the equivalent sum-of-products expression:

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

A given sum-of-products expression can be transformed into an equivalent product-of-sums expression by (a) taking the dual of the given expression, (b) multiplying out different terms to get the sum-of-products form, (c) removing redundancy and (d) taking a dual to get the equivalent product-of-sums expression. As an illustration, let us find the equivalent product-of-sums expression of the sum-of-products expression

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

The dual of the given expression $=(A+B).(\overline{A}+\overline{B})$:

$$(A+B).(\overline{A}+\overline{B})=A.\overline{A}+A.\overline{B}+B.\overline{A}+B.\overline{B}=0+A.\overline{B}+B.\overline{A}+0=A.\overline{B}+\overline{A}.B$$

The dual of $(A.\overline{B}+\overline{A}.B)=(A+\overline{B}).(\overline{A}+B)$. Therefore

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

6.4.3 Expanded Forms of Boolean Expressions

Expanded sum-of-products and product-of-sums forms of Boolean expressions are useful not only in analysing these expressions but also in the application of minimization techniques such as the Quine–McCluskey tabular method and the Karnaugh mapping method for simplifying given Boolean expressions. The expanded form, sum-of-products or product-of-sums, is obtained by including all possible combinations of missing variables.

As an illustration, consider the following sum-of-products expression:

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

It is a three-variable expression. Expanded versions of different minterms can be written as follows:

- $A.\overline{B}=A.\overline{B}.(C+\overline{C})=A.\overline{B}.C+A.\overline{B}.\overline{C}$.
- $B.\overline{C}=B.\overline{C}.(A+\overline{A})=B.\overline{C}.A+B.\overline{C}.\overline{A}$.
- $A.B.\overline{C}$ is a complete term and has no missing variable.
- $\overline{A}.C=\overline{A}.C.(B+\overline{B})=\overline{A}.C.B+\overline{A}.C.\overline{B}$.

The expanded sum-of-products expression is therefore given by

$$\begin{aligned}&A.\overline{B}.C+A.\overline{B}.\overline{C}+A.B.\overline{C}+\overline{A}.B.\overline{C}+A.B.\overline{C}+\overline{A}.B.C+\overline{A}.\overline{B}.C=A.\overline{B}.C+A.\overline{B}.\overline{C}\\&\quad+A.B.\overline{C}+\overline{A}.B.\overline{C}+\overline{A}.B.C+\overline{A}.\overline{B}.C\end{aligned}$$

As another illustration, consider the product-of-sums expression

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

It is four-variable expression with A, B, C and D being the four variables. $\overline{A}+B$ in this case expands to $(\overline{A}+B+C+D).(\overline{A}+B+C+\overline{D}).(\overline{A}+B+\overline{C}+D).(\overline{A}+B+\overline{C}+\overline{D})$.

The expanded product-of-sums expression is therefore given by

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

6.4.4 Canonical Form of Boolean Expressions

An expanded form of Boolean expression, where each term contains all Boolean variables in their true or complemented form, is also known as the *canonical form* of the expression.

As an illustration, $f(A.B, C) = \overline{A}.\overline{B}.\overline{C}+\overline{A}.\overline{B}.C+A.B.C$ is a Boolean function of three variables expressed in canonical form. This function after simplification reduces to $\overline{A}.\overline{B}+A.B.C$ and loses its canonical form.

6.4.5 Σ and Π Nomenclature

Σ and Π notations are respectively used to represent sum-of-products and product-of-sums Boolean expressions. We will illustrate these notations with the help of examples. Let us consider the following Boolean function:

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

We will represent this function using Σ notation. The first step is to write the expanded sum-of-products given by

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

Different terms are then arranged in ascending order of the binary numbers represented by various terms, with true variables representing a '1' and a complemented variable representing a '0'. The expression becomes

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

The different terms represent 0001, 0101, 1000, 1001 and 1111. The decimal equivalent of these terms enclosed in the Σ then gives the Σ notation for the given Boolean function. That is, $f(A, B, C, D) = \sum 1, 5, 8, 9, 15$.

The complement of $f(A, B, C, D)$, that is, $f'(A, B, C, D)$, can be directly determined from Σ notation by including the left-out entries from the list of all possible numbers for a four-variable function. That is,

$$f'(A, B, C, D) = \sum 0, 2, 3, 4, 6, 7, 10, 11, 12, 13, 14$$

Let us now take the case of a product-of-sums Boolean function and its representation in Π nomenclature. Let us consider the Boolean function

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

The expanded product-of-sums form is given by

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

The binary numbers represented by the different sum terms are 0011, 1011, 1100 and 0111 (true and complemented variables here represent 0 and 1 respectively). When arranged in ascending order, these numbers are 0011, 0111, 1011 and 1100. Therefore,

$$f(A, B, C, D) = \prod 3, 7, 11, 12 \quad \text{and} \quad f'(A, B, C, D) = \prod 0, 1, 2, 4, 5, 6, 8, 9, 10, 13, 14, 15$$

An interesting corollary of what we have discussed above is that, if a given Boolean function $f(A,B,C)$ is given by $f(A, B, C) = \sum 0, 1, 4, 7$, then

$$f(A, B, C) = \prod 2, 3, 5, 6 \quad \text{and} \quad f'(A, B, C) = \sum 2, 3, 5, 6 = \prod 0, 1, 4, 7$$

Optional combinations can also be incorporated into Σ and Π nomenclature using suitable identifiers; ϕ or d are used as identifiers. For example, if $f(A, B, C) = \overline{A}.\overline{B}.\overline{C} + A.\overline{B}.\overline{C} + A.\overline{B}.C$ and $\overline{A}.B.C$, $A.B.C$ are optional combinations, then

$$f(A, B, C) = \sum 0, 4, 5 + \sum_{\phi} 3, 7 = \sum 0, 4, 5 + \sum_{d} 3, 7$$

$$f(A, B, C) = \prod 1, 2, 6 + \prod_{\phi} 3, 7 = \prod 1, 2, 6 + \prod_{d} 3, 7$$

Example 6.8

For a Boolean function $f(A, B) = \sum 0, 2$, prove that $f(A, B) = \prod 1, 3$ and $f'(A, B) = \sum 1, 3 = \prod 0, 2$.

Solution

- $f(A, B) = \sum 0, 2 = \overline{A}.\overline{B} + A.\overline{B} = \overline{B}.(A + \overline{A}) = \overline{B}$.
- Now, $\prod 1, 3 = (A + \overline{B}).(\overline{A} + \overline{B}) = A.\overline{A} + A.\overline{B} + \overline{B}.\overline{A} + \overline{B}.\overline{B} = A.\overline{B} + \overline{A}.\overline{B} + \overline{B} = \overline{B}$.
- Now, $\sum 1, 3 = \overline{A}.B + A.B = B.(\overline{A} + A) = B$.
  and $\prod 0, 2 = (A + B).(\overline{A} + B) = A.\overline{A} + A.B + B.\overline{A} + B.B = A.B + \overline{A}.B + B = B$.
- Therefore, $\sum 1, 3 = \prod 0, 2$.
- Also, $f(A, B) = \overline{B}$.
- Therefore, $f'(A, B) = B$ or $f'(A, B) = \sum 1, 3 = \prod 0, 2$.