Progressive Education Society's

Modern College of Arts, Science and Commerce, Shivajinagar, Pune – 5

(Autonomous College)

First Year of B. Sc. (2019 Course)

SEMESTER – I Paper -II

Course Code :19ScEleU102 **Course Name: Fundamentals of Digital Electronics**

Course Contents

Chapter 1	**Number Systems**	16 lectures
	Introduction to decimal, binary and hexadecimal number systems and their inter conversions, Unsigned and Signed binary number representations, Rules of binary addition and subtraction, Binary addition and subtraction, Subtraction using 1's and 2's complements, BCD code, Excess-3 code, Gray code, Alphanumeric representation in ASCII codes, Code conversion –binary to gray, gray to binary.	
Chapter 2	**Logic Gates**	7 lectures
	Positive and Negative Logic, OR, AND, NOT gates, NAND, NOR, EX-OR, EX-NOR gates (Symbol and truth table).	
Chapter 3	**Boolean Algebra**	12 lectures
	Boolean algebra and Boolean laws: Commutative, Associative, Distributive, AND, OR and Inversion laws, De Morgen's theorem, NAND, NOR as universal gate, K-map Basics, Min terms, Max terms, Boolean expression in SOP and POS form, Simplifications of Logic expressions using Boolean algebra rules and Karnaugh map (up to 4 variables), Implementation of Boolean expressions using basic gates.	
Chapter 4	**Experiential Learning**	1 lecture
	Group Discussion / Field Work / Mini Project.	

Text/ Reference Books:

1. Digital Electronics: Jain R.P., Tata McGraw Hill
2. Digital Principles and Applications: Malvino Leach, Tata McGraw-Hill
3. Digital Fundamentals: Floyd T.M., Pearson Education

Chapter 3	**Boolean Algebra**	12 lectures
	Boolean algebra and Boolean laws: Commutative, Associative, Distributive, AND, OR and Inversion laws, De Morgen's theorem, NAND, NOR as universal gate, K-map Basics, Min terms, Max terms, Boolean expression in SOP and POS form, Simplifications of Logic expressions using Boolean algebra rules and Karnaugh map (up to 4 variables), Implementation of Boolean expressions using basic gates.	

1. Boolean algebra and Boolean laws:

Boolean algebra is the mathematics of digital system. Infact it is a convenient and systematic way of expressing and analysing the operation of logic circuits. It is found that the knowledge of Boolean algebra is essential to study and analysis of logic circuit.

A mathematical system for formulating logical statements with symbols, so that problems can be written and solved like a ordinary algebra, was developed by the Irish logician and mathematician George Boole and known as *Boolean algebra*.

Boolean algebra was introduced in 1854. Boole is one of the persons in a long historical chain who were concerned with formalising and mechanising the process of logical thinking. Boolean algebra is a generalisation of set algebra and the algebra of propositions and is a tool for studying and applying logic.

It provides mathematical basis for expressing logic circuit functions, as well as analysing and designing of the digital system.

The Boolean axioms are the laws of Boolean algebra for addition, multiplication and for the inversion. They are known as axioms because they are the truth which can be verified for different possibilities but cannot be proved. There are different operators for Boolean algebra.

2. Boolean Operators :

The operators in Boolean algebra are slightly different than conventional algebra for the basic gates and the following operators are commonly used.

(i) Dot sign (·) : The dot sign (·) which is also expressed as (×), indicates logical product of two terms. The logical product of two terms A and B is expressed as A·B (or A × B) and is read as "A AND B". The inputs A and B are said to be ANDed.

(ii) Plus sign (+) : The (+) sign indicates the logical sum of two terms. For example, A + B represents logical sum of terms A and B and is read as "A OR B". The inputs A and B are said to be ORed.

(iii) Overbar (¯) : A sign of (¯) indicates that the terms which are overbar, are to be complemented. For example, $\overline{A}$ represents complementation of term A and is read as "NOT A".

3. Boolean Addition and Multiplication :

Addition in Boolean algebra involves variables having values of either a binary 1 or a binary 0. Binary 1 will represent a HIGH level and binary 0 will represent a LOW level in Boolean equations.

The basic rules for Boolean addition are as follows :

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 1$$

Boolean addition is the same as the OR. Notice that, it differs from binary addition in the case where two 1s are added.

Multiplication in Boolean algebra follows the same basic rules governing binary multiplication.

$$0 \cdot 0 = 0$$
$$0 \cdot 1 = 0$$
$$1 \cdot 0 = 0$$
$$1 \cdot 1 = 1$$

Boolean multiplication is the same as the AND.

Complementation in Boolean algebra is as follows :

$$\overline{0} = 1$$
$$\overline{1} = 0$$

It is same as NOT.

4. Rules and Laws of Boolean Algebra

There are certain rules and laws in Boolean algebra. We shall learn Boolean Commutative, Associative, Distributive, AND, OR and Inversion laws in this section.

The alphabets A, B, C and D can be used as variables having values 0 and 1.

Laws of Intersection :

Law 1 : $A \cdot 1 = A$

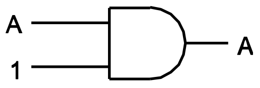


Fig. 1

If a logic 1 is applied to one of the two inputs of the AND gate and signal A to the other input, the output will be A.

Law 2 : $A \cdot 0 = 0$

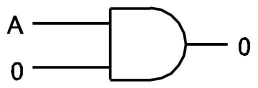


Fig. 2

If a logic 0 is applied to one of the two inputs of the AND gate and signal A to the other input, the output will be logic 0.

Laws of Union :

Law 3 : $A + 1 = 1$

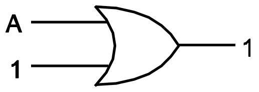


Fig. 3

If a logic 1 is applied to the two inputs of OR gate and signal A to the other input, the output will be logic 1.

Law 4 : $A + 0 = A$

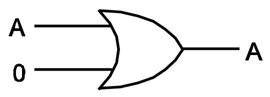


Fig. 4

If a logic 0 is applied to one of the two inputs of OR gate and A to the other input, the output will be logic A.

Laws of Tautology :

Law 5 : $A \cdot A = A$

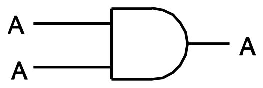


Fig. 5

If the same signal A is applied to all the inputs of AND gate, the output will be same as the input.

Law 6 : $A + A = A$

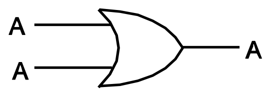


Fig. 6

If the same signal A is applied to all the inputs of OR gate, the output will be same as input A.

Laws of Complement :

Law 7 : $A \cdot \bar{A} = 0$

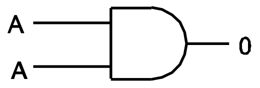


Fig. 7

If a logic signal A and its complement $\bar{A}$ is applied to an AND gate, the output will be logic 0.

Law 8 : $A + \bar{A} = 1$

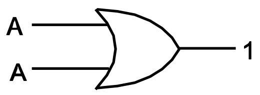


Fig. 8

If a logic signal A and its complement $\bar{A}$ is applied to an OR gate, the output will be 1.

Law of Double negation :

Law 9 : $\bar{\bar{A}} = A$

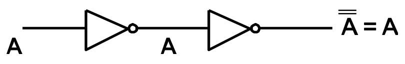


Fig. 9

The complement of the complement of A is A.

Laws of Commutation :

Law 10 : $A \cdot B = B \cdot A$

The commutative law of multiplication states that order in which variables are ANDed makes no difference at the output.

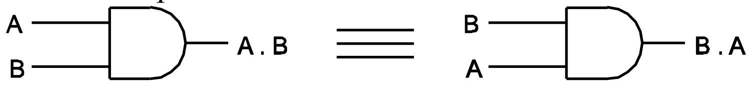


Fig. 10

Law 11 : A + B = B + A

This law states that 'the order in which the inputs are given to an OR gate makes no difference at the output'.

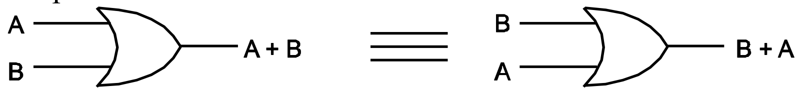


Fig. 11

Laws of Association :

Law 12 : The associative law of addition for three variables is stated as follows,

$$(A + B) + C \;=\; A + (B + C)$$

This law states that 'in the ORing of several variables, the result is same regardless of grouping of variables'.

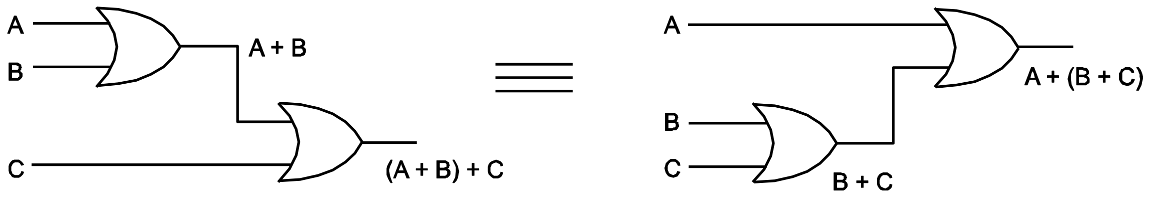


Fig. 12

Law 13 : The associative law of multiplication is stated as follows for three variables.

$$(A \cdot B) \cdot C \;=\; A \cdot (B \cdot C)$$

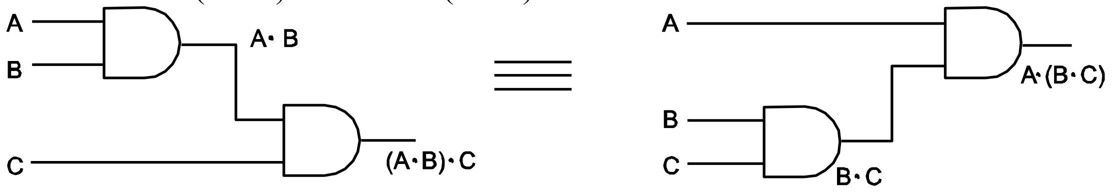


Fig. 13

Laws of Distribution :

Law 14 : The distributive law for three variables is written as follows :

$$A (B + C) \;=\; AB + AC$$

This law states that 'ORing several variables and ANDing the result with a single variable is equivalent to ANDing the single variable with each of several variables and then ORing the products'.

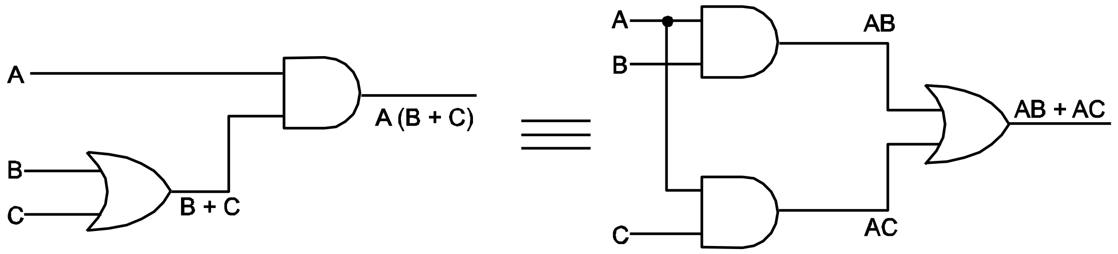


Fig. 14

Law 15 : $A + (B \cdot C) \;=\; (A + B) \cdot (A + C)$ **[Comp. Sci. : M-14, 15]**

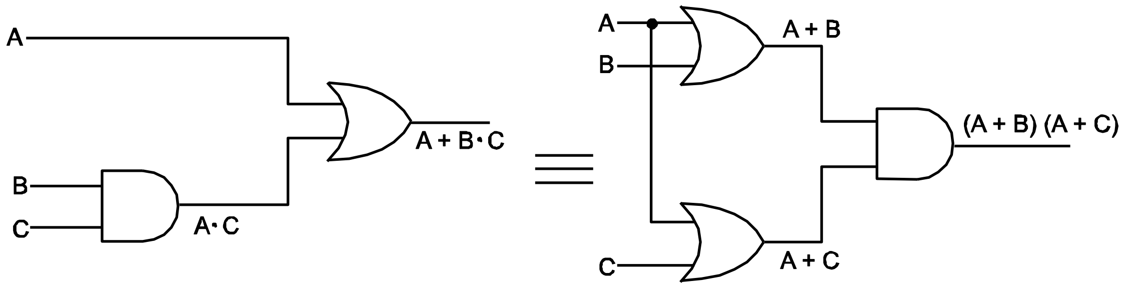


Fig. 15

Proof : $(A + B) \cdot (A + C) = AA + AB + AC + BC$

$= A(1 + B + C) + BC \quad \ldots (1)$

Though B and C have any value either 0 or 1,

$1 + B + C = 1$

Therefore equation (1) becomes $= A \cdot 1 + BC$

From law (1), $A \cdot 1 = A$

$\therefore \quad (A + B)(A + C) = A + BC$

Laws of Absorption :

Law 16 : $A + A \cdot B = A$

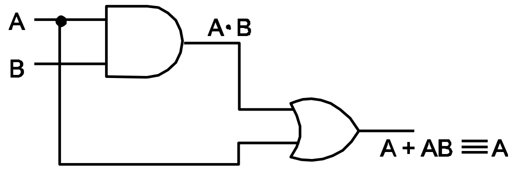


Fig. 16

Proof : Consider $A + A \cdot B = A \cdot (1 + B)$

$= A \cdot 1 \quad \because 1 + B = 1$

$= A \quad \because A \cdot 1 = A$

Law 17 : $A \cdot (A + B) = A$

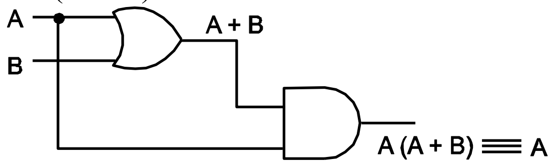


Fig. 17

Proof : $A \cdot (A + B) = A \cdot A + A \cdot B$

$= A + AB \quad \because A \cdot A = A$

$= A(1 + B)$

$= A \cdot 1 \quad \because 1 + B = 1$

$= A \quad \because A \cdot 1 = A$

Law 18 : $A \cdot B + \overline{B} = A + \overline{B}$

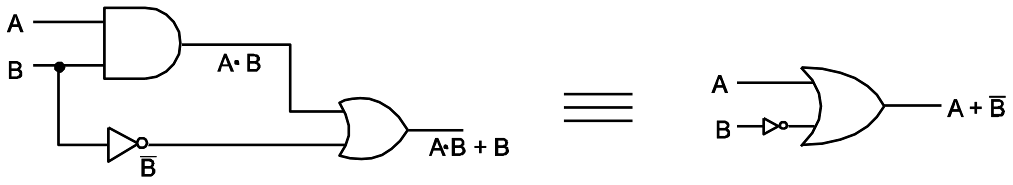


Fig. 18

Proof : $A \cdot B + \overline{B} = A \cdot B + \overline{B} \cdot 1 \quad \because \overline{B} \cdot 1 = \overline{B}$

$= A \cdot B + \overline{B} \cdot (A + 1) \quad \because A + 1 = 1$

$= A \cdot B + \overline{B}A + \overline{B} \cdot 1$

$= A \cdot (B + \overline{B}) + \overline{B} \quad \because \overline{B} \cdot 1 = \overline{B}$

$= A + \overline{B} \quad \because B + \overline{B} = 1$

Law 19 : $A \cdot (\overline{A} + B) = A \cdot B$

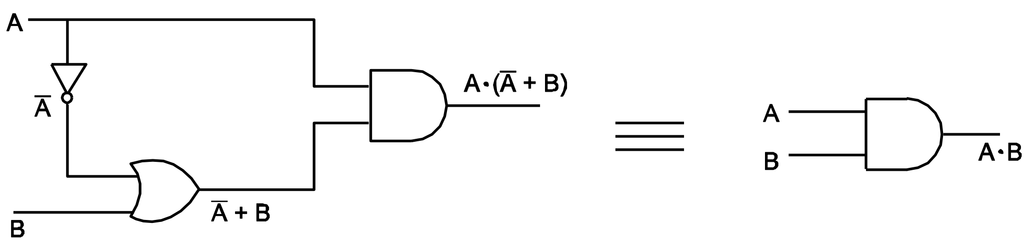


Fig. 19

Proof :
$$A \cdot (\overline{A} + B) = A\overline{A} + AB$$
$$= 0 + AB \qquad \because A \cdot \overline{A} = 0$$
$$= A \cdot B$$

Law 20 : $A \cdot \overline{B} + B = A + B$

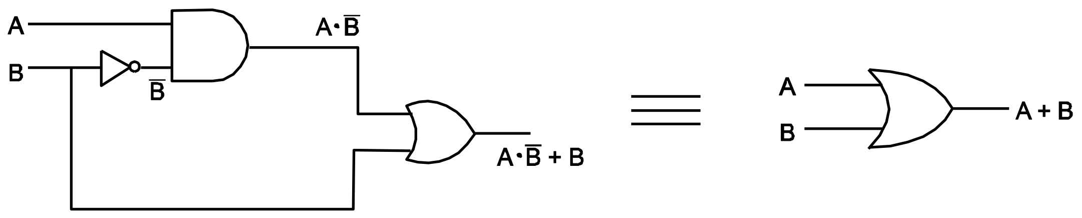


Fig. 20

Proof : Consider
$$A \cdot \overline{B} + B = A \cdot \overline{B} + B \cdot 1 \qquad \because B \cdot 1 = B$$
$$= A \cdot \overline{B} + B \cdot (A + 1) \qquad \because A + 1 = 1$$
$$= A \cdot \overline{B} + B \cdot A + B \cdot 1$$
$$= A(\overline{B} + B) + B$$
$$= A \cdot 1 + B \qquad \because \overline{B} + B = 1$$
$$= A + B \qquad \because A \cdot 1 = A$$

Table 1 : Boolean algebraic theorems

Law	Classification	Algebraic definition
1 2	Laws of intersection	$A \cdot 1 = A$ $A \cdot 0 = 0$
3 4	Laws of union	$A + 1 = 1$ $A + 0 = A$
5 6	Laws of tautology	$A \cdot A = A$ $A + A = A$
7 8	Laws of complement	$A \cdot \overline{A} = 0$ $A + \overline{A} = 1$
9	Law of double negation	$\overline{\overline{A}} = A$
10 11	Laws of commutation	$A \cdot B = B \cdot A$ $A + B = B + A$

12 13	Laws of association	$(A + B) + C = A + (B + C)$ $(A \cdot B) \cdot C = A \cdot (B \cdot C)$
14 15	Laws of distribution	$A (B + C) = A \cdot B + A \cdot C$ $A + (B \cdot C) = (A + B) \cdot (A + C)$
16 17 18 19 20	Laws of absorption	$A + A \cdot B = A$ $A \cdot (A + B) = A$ $A \cdot B + \overline{B} = A + \overline{B}$ $A \cdot (\overline{A} + B) = AB$ $A \cdot \overline{B} + B = A + B$

5. Simplification of Logic Equations using Laws of Boolean Algebra

Many times it is essential to reduce number of gates required for designing a digital circuit, reduce a particular expression to its simplest form, ultimately which reduces size and price (cost) of circuit. By applying basic laws, rules and theorems of Boolean algebra, it is possible to implement practically.

Following examples illustrate the technique.

Example 1 : Simplify the expression AB + A (B + C) + B (B + C) using Boolean algebra techniques.

Solution : Consider

$$\begin{aligned} AB + A(B + C) + B(B + C) &= AB + AB + AC + BB + BC \\ &= A(B + B) + AC + B(B + C) && \because B \cdot B = 1 \\ &= AB + AC + B && \because B + B = B,\ B(B + C) = B \\ &= AB + B + AC \\ &= B(A + 1) + AC \\ &= B \cdot 1 + AC && \because A + 1 = 1 \\ &= B + AC \end{aligned}$$

$$\therefore \quad AB + A(B + C) = B + AC$$

The logic circuit will be

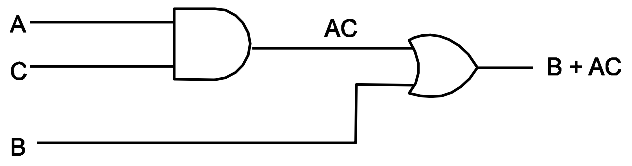


Fig. 21

Example 2 : Simplify the expression $[A\overline{B}(C + BD) + \overline{A}\,\overline{B}]\,C$ using Boolean algebra techniques.

Solution : Consider

$$\begin{aligned} [A\overline{B}(C + BD) + \overline{A}\,\overline{B}]\,C &= (A\overline{B}C + A\overline{B}BD + \overline{A}\,\overline{B})\,C \\ &= (A\overline{B}C + A \cdot 0 \cdot D + \overline{A}\,\overline{B})\,C && \because \overline{B}\,B = 0 \\ &= (A\overline{B}C + 0 + \overline{A}\,\overline{B})\,C \\ &= (A\overline{B}C + \overline{A}\,\overline{B})\,C \\ &= A\overline{B}CC + \overline{A}\overline{B}C \\ &= A\overline{B}C + \overline{A}\overline{B}C && \because C \cdot C = C \end{aligned}$$

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

$$= \bar{B}C \cdot (1) \qquad \because A + \bar{A} = 1$$

$$= \bar{B}C$$

$$\therefore \quad [A\bar{B}\,(C + BD) + \bar{A}\bar{B}]\,C = \bar{B}C$$

The logic circuit will be

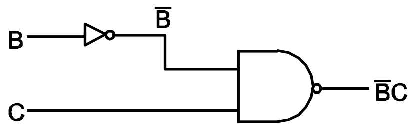


Fig. 22

Example 3 : Using Boolean algebra techniques, simplify the following expressions as much as possible : (i) A(A + B), (ii) A($\bar{A}$ + AB), (iii) BC + $\bar{B}$C, (iv) A(A + $\bar{A}$B).

Solution : (i)

$$A\,(A + B) = A{\cdot}A + A{\cdot}B$$

$$= A + A{\cdot}B \qquad \because A{\cdot}A = A$$

$$= A\,(1 + B)$$

$$= A \cdot (1) \qquad \because 1 + B = 1$$

$$= A$$

$$\therefore \quad A\,(A + B) = A$$

(ii)

$$A\,(\bar{A} + AB) = A\bar{A} + A{\cdot}AB$$

$$= 0 + A{\cdot}AB \qquad \because A\bar{A} = 0$$

$$= AB \qquad \because A{\cdot}A = A$$

$$\therefore \quad A\,(\bar{A} + AB) = AB$$

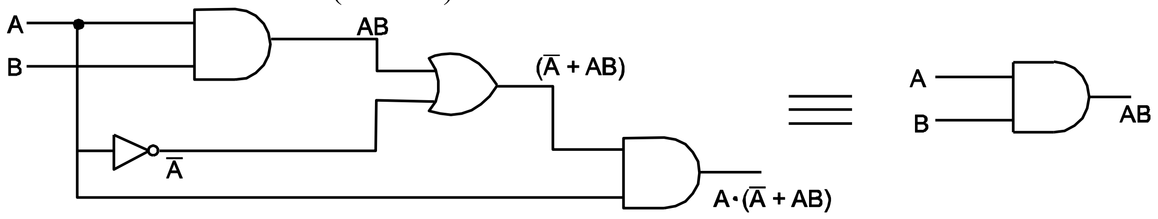


Fig. 23

(iii)

$$BC + \bar{B}C = C\,(B + \bar{B})$$

$$= C \cdot (1) \qquad \because B + \bar{B} = 1$$

$$= C$$

(iv)

$$A\,(A + \bar{A}B) = A{\cdot}A + A{\cdot}\bar{A}B$$

$$= A + A{\cdot}\bar{A}B \qquad \because A{\cdot}A = A$$

$$= A + 0 \qquad \because A{\cdot}\bar{A} = 0$$

$$= A \qquad \because A + 0 = A$$

Example 4 : Reduce the following Boolean expression and draw the logic diagram :

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

Solution : Consider

$$\bar{A}BC\bar{D} + BC\bar{D} + B\bar{C}\bar{D} + B\bar{C}D = BC\bar{D}(\bar{A}+1) + B\bar{C}(\bar{D}+D) \qquad (\because \bar{A}+1 = 1, D+\bar{D} = 1)$$

$$= BC\bar{D}\cdot(1) + B\bar{C}\cdot(1) = BC\bar{D} + B\bar{C} = B(C\bar{D} + \bar{C})$$

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

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

The logic circuit will be

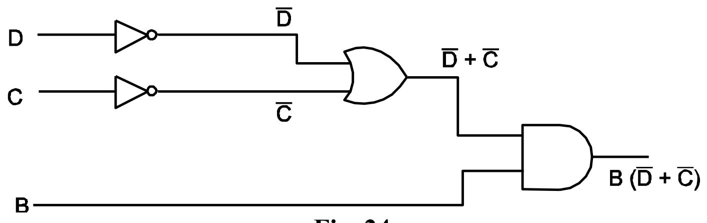


Fig. 24

Example 5 : Simplify the following equations using laws of Boolean algebra :

(i) $Y = ABCD + ABC + AB + A\bar{B}$

Solution : Consider

$$ABCD + ABC + AB + A\bar{B} = ABC(D+1) + A(B+\bar{B})$$

$$= ABC\cdot(1) + A\cdot(1) \qquad (\because D+1 = 1, B+\bar{B} = 1)$$

$$= ABC + A$$

Circuit for $Y = ABCD + ABC + AB + A\bar{B}$ can be constructed as follows :

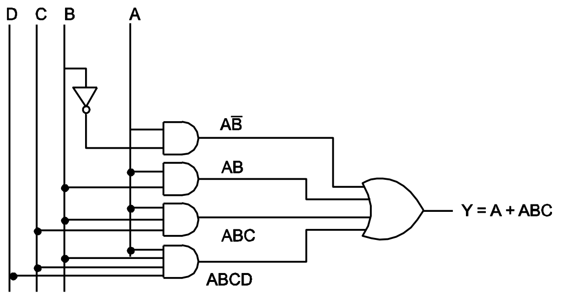


Fig. 25

Here four input AND and four input OR gate are used.

(ii) $Y = \bar{A} + AB + A\bar{B}$

Solution : Consider

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

$$= \bar{A} + A(1) = \bar{A} + A = 1 \qquad (\because B+\bar{B} = 1)$$

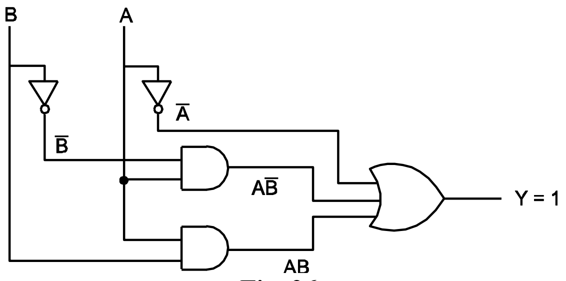


Fig. 26

Here three input OR gate is used.

(iii) $Y = AB + \bar{A}B + ABC$

Solution : $Y = AB + ABC + \bar{A}B$

$= AB(1 + C) + \bar{A}B$

$= AB(1) + \bar{A}B = AB + \bar{A}B$ $(\because 1 + C = 1)$

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

$= B$ $(\because A + \bar{A} = 1)$

Let us consider the following circuit to solve the given equation.

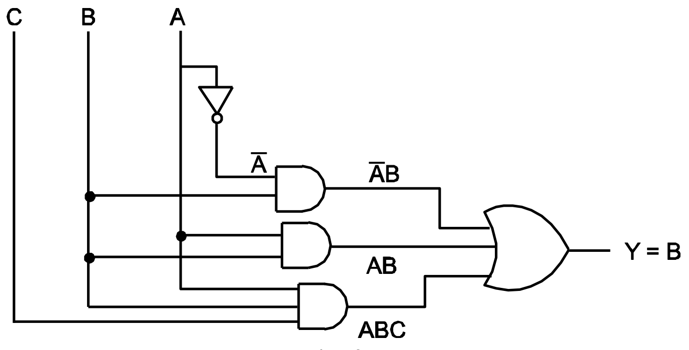


Fig. 27

(iv) $A = AB + BC + \bar{B}A + \bar{A}B$

Solution : $Y = AB + \bar{A}B + BC + \bar{B}A = B(A + \bar{A}) + BC + \bar{B}A$

$= B + BC + \bar{B}A$ $(\because A + \bar{A} = 1)$

$= B(1 + C) + \bar{B}A$

$= B + \bar{B}A$ $(\because 1 + C = 1)$

Let us consider the following circuit to solve the given equation.

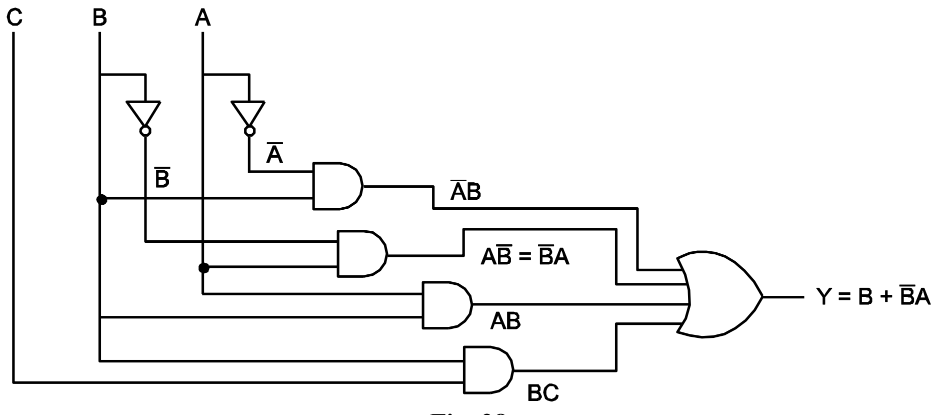


Fig. 28

6. De Morgan's theorem:

De Morgan, a logician and mathematician proposed two theorems which are important parts of Boolean algebra.

De Morgan's first theorem

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

The complement of product is equal to the sum of the complements. The complement of two or more variables ANDed is the same as the OR of the complement of each individual variables.

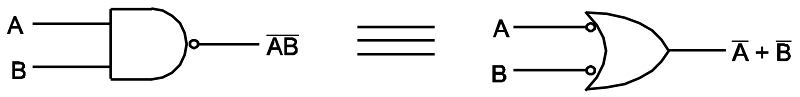


Fig. 29

It can also be constructed with AND, OR and NOT gate as follows :

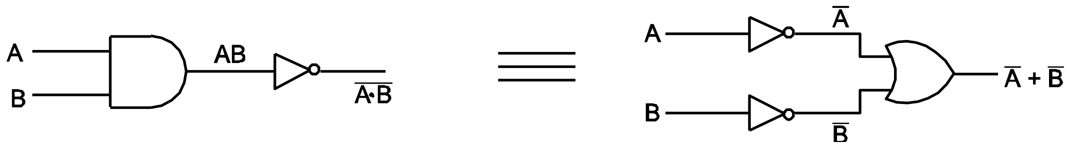


Fig. 30

Table 2 : De Morgan's first theorem

Input		Intermediate value			Output	
A	**B**	**AB**	$\overline{A}$	$\overline{B}$	$\overline{A \cdot B}$	$\overline{A} + \overline{B}$
0	0	0	1	1	1	1
0	1	0	1	0	1	1
1	0	0	0	1	1	1
1	1	1	0	0	0	0

De Morgan's Second Theorem:

The complement of a sum is equal to the product of the complements. It can be expressed as

$$\overline{A + B} \quad = \quad \overline{A} \cdot \overline{B}$$

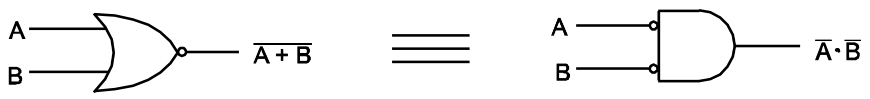


Fig. 31

Using AND, OR and NOT gate, the circuit can be drawn as

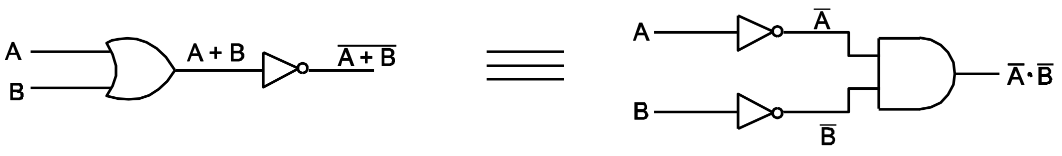


Fig. 32

Table 3 : De Morgan's second theorem

Input		**Intermediate value**			**Output**	
A	**B**	**$A + B$**	**$\overline{A}$**	**$\overline{B}$**	**$\overline{A + B}$**	**$\overline{A} \cdot \overline{B}$**
0	0	0	1	1	1	1
0	1	1	1	0	0	0
1	0	1	0	1	0	0
1	1	1	0	0	0	0

Now consider the NAND operation of three variables.

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

$$\text{And} \qquad \overline{A + B + C} = \overline{A} \cdot \overline{B} \cdot \overline{C}$$

The above results can be easily extended to any number of variables.

Applications of De Morgan's Theorems

De Morgan's theorems can be used for simplifying logic function.

Example 1 : Simplify the following expressions.

Solution : (i) $\overline{\overline{A + B} + \overline{C}} = \overline{\overline{A + B}} \cdot \overline{\overline{C}}$ $\quad [\because \overline{\overline{A + B}} = A + B \text{ and } \overline{\overline{C}} = C]$

$= A + B \cdot C$

$= (A + B) \cdot C$

(ii) $\overline{\overline{A} + B + \overline{CD}} = \overline{\overline{\overline{A} + B}} \cdot \overline{\overline{CD}} = (\overline{A} + B)\, CD$

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

$= \overline{\overline{A} + \overline{B} + AB}$ $\qquad \because \overline{A} + \overline{A} = \overline{A}$

$= \overline{\overline{A} + A + \overline{B}}$ $\qquad \because AB + \overline{B} = A + \overline{B}$

$= \overline{1 + \overline{B}} = \overline{1} = 0$

$\overline{\overline{A}\overline{B} + \overline{A} + AB} = 0$

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

$= A + \overline{B} + \overline{B} + \overline{C} + \overline{C} + D$ $\qquad \because \overline{\overline{A}} = A \text{ and } \overline{\overline{D}} = D$

$= A + \overline{B} + \overline{C} + D$ $\qquad \because \overline{B} + \overline{B} = \overline{B} \text{ and } \overline{C} + \overline{C} = \overline{C}$

$\therefore \quad \overline{(\overline{A} \cdot B)\,(BC)\,(C\overline{D})} = A + \overline{B} + \overline{C} + D$

7. NAND, NOR as Universal Gate:

The NAND and NOR gates have universal property and can be used for performing AND, OR and NOT functions, so an AND/OR/NOT logic circuit can be converted to NAND/NOR logic.

NAND as NOT Gate

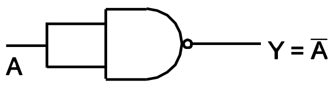


Fig. 33

$$Y=\overline{A\cdot A}=\overline{A}$$

A NOT gate can be obtained from NAND gate by connecting all the inputs together.

NAND as AND Gate

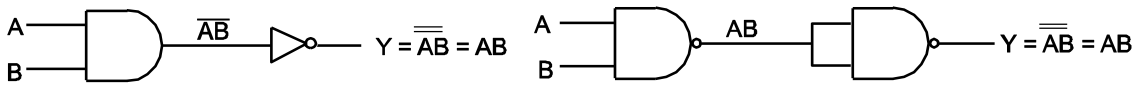

A
B
$\overline{AB}$
Y = $\overline{\overline{AB}}$ = AB
A
B
AB
Y = $\overline{\overline{AB}}$ = AB

Fig. 34

The AND gate can be obtained by simply inverting output of NAND gate.

NAND as OR Gate

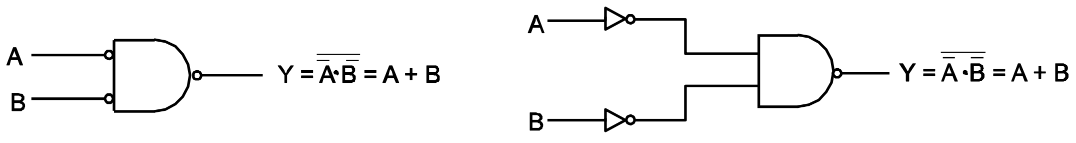
The OR gate can be designed by using NAND gate.

$$Y=A+B$$

$$Y=\overline{\overline{A}}+\overline{\overline{B}}$$

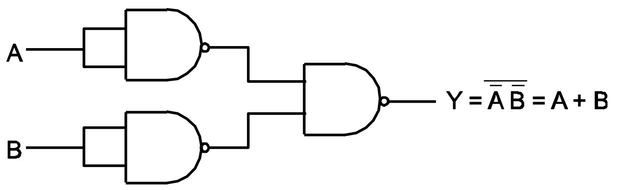
$$Y=\overline{\overline{A}\cdot\overline{B}}$$

Therefore an OR gate operation can be obtained by NANDing $\overline{A}$ and $\overline{B}$.


A
B
Y = $\overline{\overline{A}\cdot\overline{B}}$ = A + B
A
B
Y = $\overline{\overline{A}\cdot\overline{B}}$ = A + B

$$Y=\overline{\overline{A}\cdot\overline{B}}=A+B$$

NOT NAND



OR gate using NAND gate
Fig. 35

NOR as NOT Gate

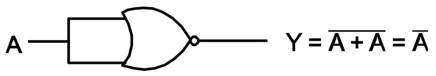


Fig. 36

Two inputs of OR gate are connected together, it gives inverter of input signal i.e. NOT.

NOR as AND Gate

The AND gate can be designed by using NOR gate.

$$Y = A \cdot B$$

$$Y = \overline{\overline{A}} \cdot \overline{\overline{B}}$$

$$Y = \overline{\overline{A} + \overline{B}}$$

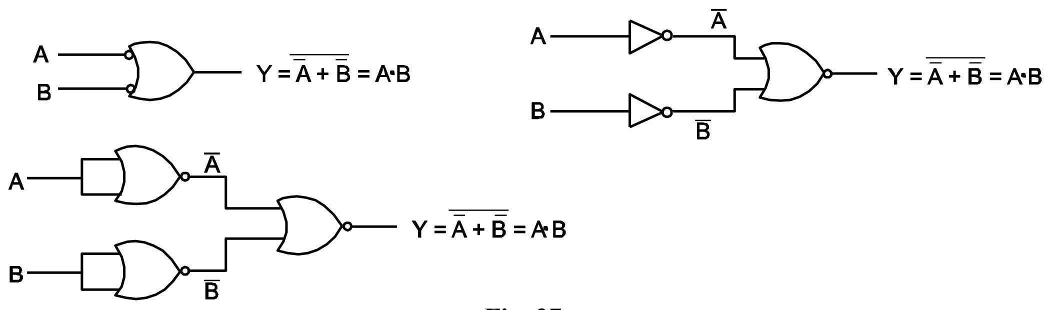
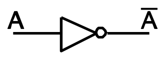
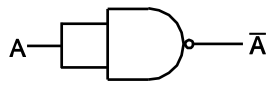
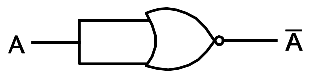
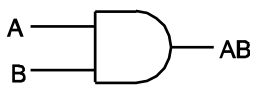
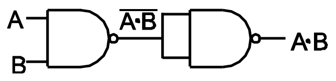
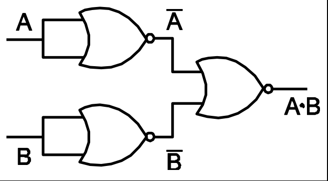
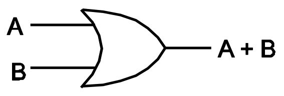
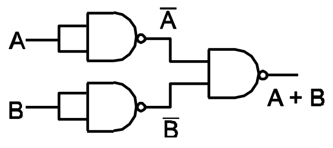
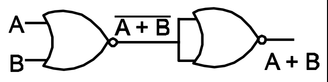

A
B
Y = $\overline{\overline{A} + \overline{B}}$ = A·B
A
$\overline{A}$
B
$\overline{B}$
Y = $\overline{\overline{A} + \overline{B}}$ = A·B
A
$\overline{A}$
B
$\overline{B}$
Y = $\overline{\overline{A} + \overline{B}}$ = A·B

Fig. 37

As a summary NAND , NOR as universal gate

Function	Symbol	NAND	NOR
1. NOT			
2. AND			
3. OR			

8. K-map Basics, Min terms, Max terms

The Karnaugh map or K-map provides a systematic method for simplifying a Boolean expression and can be used as visual display of fundamental products needed for a sum of products solution.

The K-map is composed of an arrangement of adjacent 'cells' each representing one particular combination of variables in product form. The K-map consists of 2^n cells, where n is number of variables.

For example, there are four combinations of the products of two variables A and B and their complements $\overline{A}\overline{B}$, $\overline{A}$ B, A$\overline{B}$ and AB. Therefore, the K-map must have four cells, with each cell representing one of the variable combination.

Advantages of K-map

(i) As we have seen the laws of Boolean algebra, but it is difficult to apply these laws when number of variables are large. In such a case, it makes easy by using K-map.

(ii) Use of large number of laws of Boolean algebra increases the chances of error because one have to remember all these laws and has to apply them at correct place. But by using K-map it is not necessary to remember all the laws, according to circuit situation K-map may apply. It provides easiest way to produce simplest Boolean expression with minimum chances of error.

Format of a two variable K-map can be represented as follows :

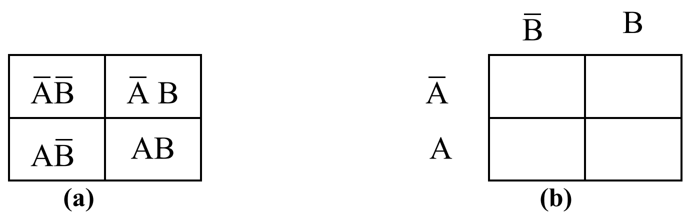
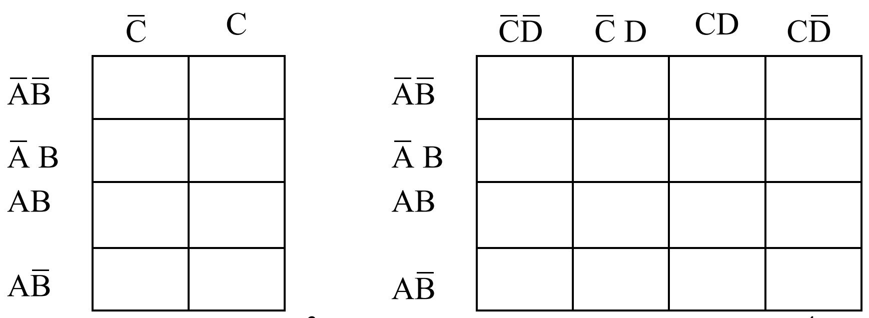


Fig. 38

Variable combination is shown in Fig. 38(a) and actually how K-map can be arranged with variables outside the cell is shown in Fig. 38(b).

Extensions of the K-map to three and four variables are shown in Fig. 39.



(a) Three variable map ($2^3 = 8$ cells) **(b) Four variable map ($2^4 = 16$ cells)**

Fig. 39

Karnaugh maps can be used for five, six or more variables.

We see how K-map can be drawn from the following Table 4.

Table 4

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

Output Y is taken by random selection of 1. First we draw the blank map.

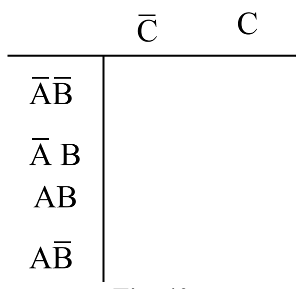


Fig. 40

The vertical column is labelled as $\bar{A}\bar{B}$, $\bar{A}B$, AB and $A\bar{B}$.

Output Y = 1 appears for 010, 110 and 111. The fundamental products for these input conditions are $\bar{A}B\bar{C}$, $AB\bar{C}$, ABC (bar on variables where it is 0).

Now, enter 1s for these products in K-map.

	$\bar{C}$	C
$\bar{A}\bar{B}$		
$\bar{A}B$	1	
AB	1	1
$A\bar{B}$		

Fig. 41

Enter 0s in the remaining spaces. Then final K-map becomes as in Fig. 42

	$\bar{C}$	C
$\bar{A}\bar{B}$	0	0
$\bar{A}B$	1	0
AB	1	1
$A\bar{B}$	0	0

Fig. 42

Pairs , Quads and Octets:

(a) Pair : The K map shown in Fig. 43 contains a pair of 1s that are horizontally adjacent (next to each other).

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	1	1	0	0
AB	0	0	0	0
$A\bar{B}$	0	0	0	0

(a)

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	$\boxed{1 \quad 1}$		0	0
AB	0	0	0	0
$A\bar{B}$	0	0	0	0

(b)

Fig. 43 : Horizontally adjacent ones

The first 1 represents the product of $\bar{A}\,B\bar{C}\bar{D}$ and second 1 the product of $\bar{A}\,B\bar{C}\,D$.
As we move from the first 1 to second 1, only one variable goes from complemented to uncomplemented ($\bar{D}$ to D), the other variables do not change the form ($\bar{A}\,B\bar{C}$ remains unchanged). In such case, we can eliminate the variable that changes the form.

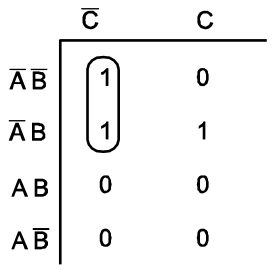
The sum of product (SOP) in which each individual term called as minterm represents

$$
\begin{aligned}
Y &= \bar{A}\,B\bar{C}\bar{D} + \bar{A}\,B\bar{C}\,D \\
&= \bar{A}\,B\bar{C}\,(\bar{D} + D) && \because \bar{D} + D = D + \bar{D} \\
&= \bar{A}\,B\bar{C}\,(D + \bar{D}) \\
&= \bar{A}\,B\bar{C} && \because D + \bar{D} = 1 \\
\therefore \quad Y &= \bar{A}\,B\bar{C}
\end{aligned}
$$

Adjacent 1s as shown in Fig.44 (a) complements are dropped out. For easy identification we will encircle a pair of adjacent 1s as shown in Fig. 44 (b).

For the pair of horizontally or vertically adjacent 1s, we can eliminate the variable that appears in both complemented and uncomplemented form.

Examples of pairs for 3 variables



	$\bar{C}$	C
$\bar{A}\bar{B}$	1	0
$\bar{A}B$	1	1
AB	0	0
$A\bar{B}$	0	0

Fig. 44 (c)

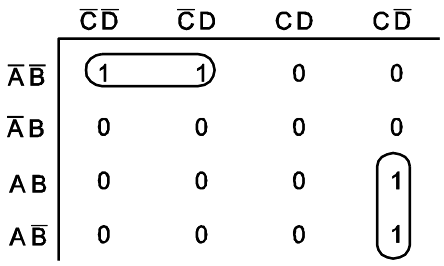
B goes from complemented to uncomplemented form ($\bar{B}$ to B).

$$Y = \bar{A}\bar{B}\bar{C} + \bar{A}\,B\bar{C}$$

$$= \bar{A}\bar{C}$$

If more than one pair exist on a Karnaugh map, we can OR the simplified products to get the Boolean equation.

For example,



	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	1	1	0	0
$\bar{A}B$	0	0	0	0
AB	0	0	0	1
$A\bar{B}$	0	0	0	1

Fig. 45

For upper horizontal pair,

$$Y = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}\,D$$

$$\therefore \quad Y = \bar{A}\bar{B}\bar{C}$$

and for the lower vertical pair.

$$Y = ABC\bar{D} + A\bar{B}\,C\bar{D}$$

$$\therefore \quad Y = AC\bar{D}$$

The corresponding Boolean equation for this map is

$$Y = \bar{A}\bar{B}\bar{C} + AC\bar{D}$$

(b) Quad : A quad is a group of four 1s that are horizontally or vertically adjacent. The 1s may be end to end or in the form of square.

This group can be formed by combining top row, bottom row, left column, right column, just like in pairing. Infact, quad eliminates two variables and their complements.

Consider K-map for three variables.

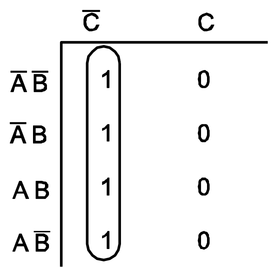


Fig. 46

Looking in K-map from Fig. 46, we find except for $\bar{C}$, other variables AB are changed from complement to uncomplement form and/or vice versa. Therefore, output Y becomes from Boolean algebra,

$$Y = \bar{A}\bar{B}\bar{C} + \bar{A}\,B\bar{C} + AB\bar{C} + A\bar{B}\bar{C}$$

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

$$Y = \bar{A}\bar{C}\,(B + \bar{B}) + A\bar{C}\,(B + \bar{B}) \qquad (\because B + \bar{B} = 1)$$

$$Y = \bar{A}\bar{C} + A\bar{C}$$

$$Y = \bar{C}\,(\bar{A} + A)$$

$$Y = \bar{C} \qquad (\because \bar{A} + A = 1)$$

The other combinations of quads are

(a)

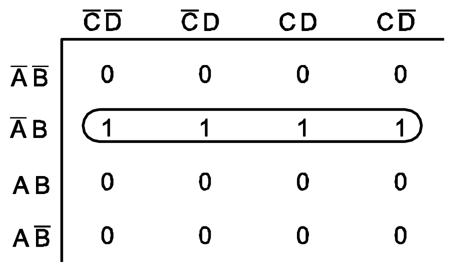


Fig. 47

$$Y = \bar{A}\,B\bar{C}\bar{D} + \bar{A}\,B\bar{C}\,D + \bar{A}\,BCD + \bar{A}\,BC\bar{D}$$

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

$$= \bar{A}\,B\bar{C} + \bar{A}\,BC \qquad (\because D + \bar{D} = 1)$$

$$= \bar{A}\,B\,(\bar{C} + C) \qquad (\because \bar{C} + C = 1)$$

$$= \bar{A}\,B$$

(b)

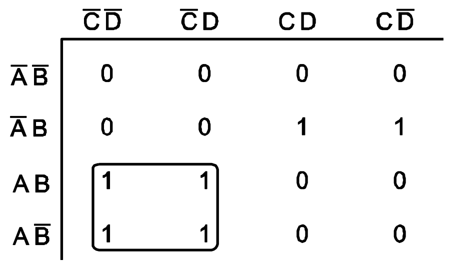
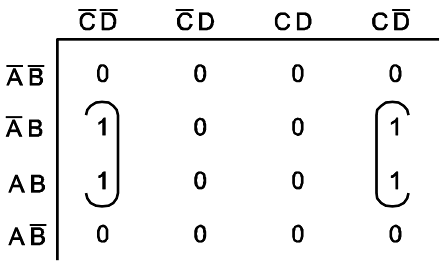


Fig. 48

$$\begin{aligned} Y &= AB\bar{C}\bar{D} + AB\bar{C}\,D + A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}\,D \\ &= AB\bar{C}\,(\bar{D} + D) + A\bar{B}\bar{C}\,(\bar{D} + D) \\ &= AB\bar{C} + A\bar{B}\bar{C} \\ &= A\bar{C}\,(B + \bar{B}) \qquad (\because B + \bar{B} = 1) \\ &= A\bar{C} \end{aligned}$$

(c)



	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	1	0	0	1
AB	1	0	0	1
$A\bar{B}$	0	0	0	0

Fig.49

$$Y = \bar{A}\,B\bar{C}\bar{D} + \bar{A}\,BC\bar{D} + AB\bar{C}\bar{D} + ABC\bar{D}$$

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

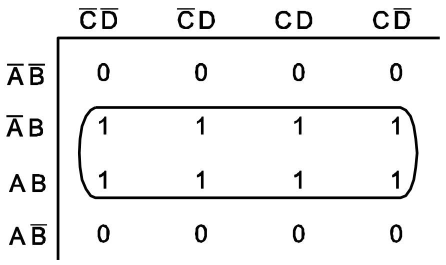
$$Y = \bar{A}\,B\bar{D} + AB\bar{D} \qquad (\because \bar{C} + C = 1)$$

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

$$Y = B\bar{D}$$

(c) Octet : This is a group of eight adjacent 1s. An octet like this eliminates three variables and their complements. Octet can be considered as a pair of quads.

(i) Consider the K-map shown in Fig. 50



	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	1	1	1	1
AB	1	1	1	1
$A\bar{B}$	0	0	0	0

Fig. 50

The Boolean expression will be

$$Y = \bar{A}\,B\bar{C}\bar{D} + \bar{A}\,B\bar{C}\,D + \bar{A}\,BCD + \bar{A}\,BC\bar{D}$$
$$+ AB\bar{C}\bar{D} + AB\bar{C}\,D + ABCD + ABC\bar{D}$$

$$\therefore \quad Y = \bar{A}B\bar{C}\,(\bar{D} + D) + \bar{A}BC\,(D + \bar{D}) + AB\bar{C}\,(\bar{D} + D) + ABC\,(D + \bar{D})$$

$$Y = \bar{A}B\bar{C} + \bar{A}BC + AB\bar{C} + ABC$$

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

$$= \bar{A}B + AB = B\,(\bar{A} + A) = B$$

(ii) Consider the K-map shown in Fig. 51.

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	0	0
AB	1	1	1	1
$A\bar{B}$	1	1	1	1

Fig. 51

$$Y=AB\bar{C}\bar{D} + AB\bar{C}D + ABCD + ABC\bar{D}$$

$$+ A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D + A\bar{B}CD + A\bar{B}C\bar{D}$$

First eliminate D,

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

$$=AB\bar{C} + ABC + A\bar{B}\bar{C} + A\bar{B}C$$

Now eliminate C,

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

$$= AB + A\bar{B}$$

Now eliminate B,

$$Y = A(B + \bar{B})$$

$$Y = A$$

In this way, three variables B, C, D and their complements dropout from the corresponding product.

9. Boolean expression in SOP and POS form:

Boolean expressions can be used to build the logic circuit. If we have the expression $Y = A + B + C$ and asked to build a circuit that perform this logic function, we can very easily see that there must be an OR gate having three inputs A, B and C. The circuit can be realised using three input OR gate as shown in Fig. 52.

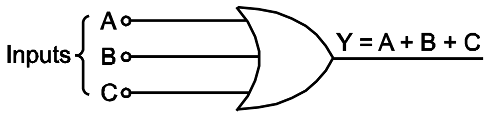


Fig. 52: Logic diagram for Boolean expression Y = A + B + C

It is found that the Boolean expression come in two forms. All Boolean expressions can be converted into either of two standard forms; the sum of product form or the product of sum forms. It is found that this standardisation makes the evaluation, simplification and implementation of Boolean expressions much more systematic and easier.

Sum of Product (SOP) Form :

A product term consist of the product of multiplication of variables or their complements (known as *literal*). When two or more product terms are summed by Boolean addition, the resulting expression is known as sum of product (SOP) form.

The examples of SOP are,

$$Y=AC+BC$$
$$Y=AB+ABC$$
$$Y=ABC+BCD+AB\overline{D} \text{ etc.}$$

The sum of product form is called *minterm form* in engineering texts and the product of sum form is called the *maximum term form* by engineers, technicians and scientist.

An SOP expression can contain a single variable term. 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. Thus, $\overline{A}\overline{B}C$ is not valid but $\overline{A}\ \overline{B}\ \overline{C}$ is valid in SOP.

The set of variables contained in the expression in either complemented or uncomplemented form is known as domain of a Boolean expression. For example, the domain of the expression $A\overline{B}+ABC$ is the set of variable A, B and C.

Implementation of an SOP Expression :

The sum of product (SOP) expression contains product of sum terms. A product term is produced by an AND operation and the sum or addition of two or more product terms is produced by an OR operation. Therefore, an SOP expression is implemented by an AND-OR logic. Consider the SOP expression y = AB + BC + CD, it can be implemented as shown in Fig. 53.

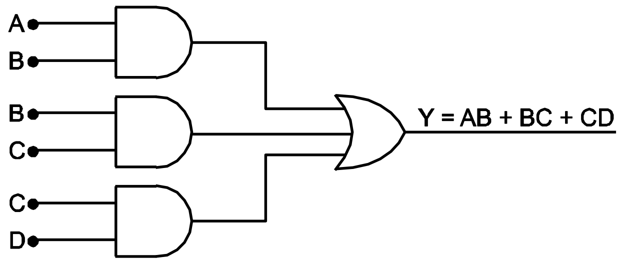


Fig. 53 : SOP of Boolean expression Y = AB + BC + CD

The Product of Sum (POS) Form :

A sum term consist of the sum i.e. Boolean addition of literals (variables or their complements). When two or more sum terms are multiplied, the resulting expression is known as product of sum (POS) form, for example,

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

POS expression can contain a single variable term. 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, POS expression can have the term $\overline{A}+\overline{B}+\overline{C}$ but will not have $\overline{A+B+C}$.

Implementation of a POS Expression :

Given Boolean expression, of the POS form can be implemented by 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.

The expression Y = (A + B) (B + C) (C + D) can be implemented as shown in Fig. 54 POS form.

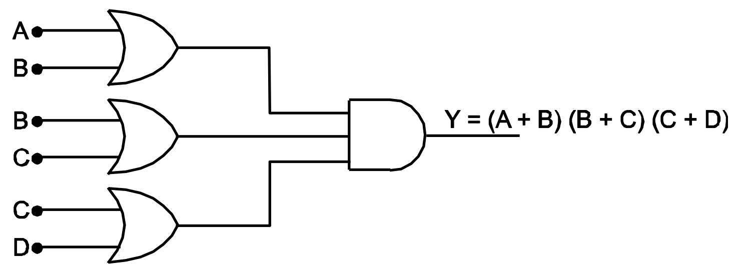


Fig. 54 : POS form of Y = (A + B) (B + C) (C + D)

Standard or Canonical SOP and POS forms : A logic expression is said to be in the standard or canonical form. If each SOP term consists of all the literals/variables in their complemented or uncomplemented form, the standard SOP form is

$$Y = \underbrace{ABC} + \underbrace{A\overline{B}C} + \underbrace{AB\overline{C}}$$

Each product term
contain all the three variables

The Boolean expression is said to be in standard POS form, if all the terms in POS consist of all the literals in their complemented or uncomplemented form. For example,

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

Each sub term
contain all the literals

Each individual term in the standard SOP form is called *minterm*.

e.g. Consider,

$$Y = \underbrace{ABC} + \underbrace{AB\overline{C}} + \underbrace{\overline{A}B\overline{C}}$$

Each individual terms
called minterm

Each individual term in the standard POS form is called *maxterm*.

e.g. Consider,

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

Each individual terms
called maxterm

Conversion of SOP/POS Expression to its standard SOP/POS:

The given Boolean expression can be converted to their corresponding SOP and POS forms. As seen in the standard SOP form each product term consist of all the literals.

e.g. $Y=AB + A\overline{B} + \overline{A}\overline{B}$ is standard SOP

but $Y=AB + ABC + AB\overline{C}$ is not standard SOP

The conversion of expression into standard SOP form is a three step process :

(1) For each term find the missing literal,

(2) Then AND the term with the term formed by ORing the missing literal and its complement,

(3) Simplify the obtained equation.

Conversion to Standard POS Form

In the standard POS form each sum term consists of all the literals in the complemented or uncomplemented form e.g.

$Y = (\bar{A} + B) \cdot (\bar{A} + \bar{B}) \cdot (A + \bar{B})$ is in the standard POS

whereas, $Y = (\bar{A} + \bar{B}) \cdot (A + B + C)$ is in the non-standard POS form.

The given POS expression can be converted into standard POS form using three steps.

Step 1 : For each term find the missing literal.

Step 2 : Then OR each term with the term formed by ANDing the missing literal in that term with its complement.

Step 3 : Simplify the expression to obtain standard POS form.

10. Simplifications of Logic expressions using Boolean algebra rules and Karnaugh map:

Example : 1 Convert the expression

$Y = AB + A\bar{C} + BC$ in the standard SOP form.

Solution : Given expression is

$$Y = AB + A\bar{C} + BC$$

Step 1 : Find the missing literal in each term.

$$Y = \underbrace{AB}_{\substack{\uparrow \\ \text{Missing} \\ \text{literal} \\ \text{is C}}} + \underbrace{A\bar{C}}_{\substack{\uparrow \\ \text{Missing} \\ \text{literal} \\ \text{is B}}} + \underbrace{BC}_{\substack{\uparrow \\ \text{Missing} \\ \text{literal} \\ \text{is A}}}$$

Step 2 : AND each term with its (Missing literal + Its complement).

$$Y = AB\,(C + \bar{C}) + A\bar{C}\,(B + \bar{B}) + BC\,(A + \bar{A})$$

Note that $C + \bar{C} = 1$ then it does not changes the value of expression.

Step 3 : Simplification of the expression.

$$Y = AB\,(C + \bar{C}) + A\bar{C}\,(B + \bar{B}) + BC\,(A + \bar{A})$$

$$Y = ABC + AB\bar{C} + AB\bar{C} + A\bar{B}\bar{C} + ABC + \bar{A}BC$$

$$= (ABC + ABC) + (AB\bar{C} + AB\bar{C}) + A\bar{B}\bar{C} + \bar{A}BC$$

Since $A + A = A$

$$Y = ABC + AB\bar{C} + A\bar{B}\bar{C} + \bar{A}BC$$

Since in the above expression each term contain all the literals it is in the standard SOP form.

Example 2 : Convert the expression

$Y = (A + B) \cdot (A + \bar{B}) \cdot (B + \bar{C})$ into the standard POS forms.

Solution : Find the missing literal in each term

$$Y = \underbrace{(A+B)}_{\substack{\uparrow\\ \text{Missing}\\ \text{literal}\\ \text{is C}}} \cdot \underbrace{(A+\bar{B})}_{\substack{\uparrow\\ \text{Missing}\\ \text{literal}\\ \text{is C}}} \cdot \underbrace{(B+\bar{C})}_{\substack{\uparrow\\ \text{Missing}\\ \text{literal}\\ \text{is A}}}$$

Step 1 : OR each term with missing literal and its complement.

$$Y = (A + B + C\,\bar{C}) \cdot (A + \bar{B} + C\,\bar{C}) \cdot (B + \bar{C} + A\bar{A})$$

Step 2 : Simplify the expression

$$Y = (A + B + C\,\bar{C}) \cdot (A + \bar{B} + C\,\bar{C}) \cdot (B + \bar{C} + A\bar{A}) \qquad \ldots (1)$$

Let $p = A + B, \quad q = A + \bar{B}, \quad r = B + \bar{C},$

$\therefore$ $$Y = (p + C\bar{C}) \cdot (q + C\bar{C}) \cdot (r + A\bar{A}) \qquad \ldots (2)$$

Since $A + BC = (A + B)(A + C)$

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

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

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

Since each term of above expression contains all the literals so the equation is standard POS form.

11. Implementation of Boolean expressions using basic gates:

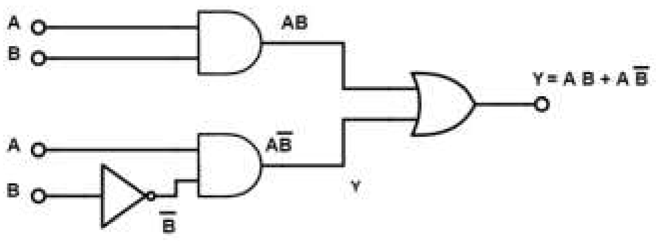
The sum of product or SOP form is represented by using basic logic gates like AND gate and OR gate. The SOP form implementation will have the AND gate at its input side and as the output of the function is the sum of all product terms, it has an OR gate at its output side. This is important to remember that we use NOT gate to represent the inverse or complement of the variables.

Logic gate implementation

Input side	AND gate
Output side	OR gate

Now to study implementation of Boolean expression in two variables consider equation $Y = AB + A\bar{B}$.

In the given SOP function, we have one compliment term, $A\overline{B}$. So to represent the compliment input, we are using the NOT gates at the input side. And to represent the product term, we use AND gates.



Hence as shown in logic diagram equation $Y = AB + A\overline{B}$ is represented by using logic gates.

---oOo---

Solved Examples

Example 1 : Simplify the equation and then draw logic diagram.

$$Y = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\;B\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C}$$

Solution : Consider $Y = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C}$

$$\begin{aligned} \therefore \quad Y &= \overline{A}\,\overline{C}\,(\overline{B} + B) + A\overline{C}\,(\overline{B} + B) \\ &= \overline{A}\,\overline{C}\,(1) + A\overline{C}\,(1) && (\because \overline{B} + B = 1) \\ &= \overline{A}\,\overline{C} + A\overline{C} \\ &= \overline{C}\,(\overline{A} + A) && (\because \overline{A} + A = 1) \\ &= \overline{C} \end{aligned}$$

The logic circuit to solve the above equation is

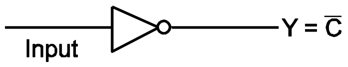


Fig. 55

We can verify the result by considering the input conditions : A = 0, B = 0 and C = 1.

The expected result is $Y = \overline{C}$ i.e. Y = 0

By applying the input, we get,

$$\begin{aligned} Y &= \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\;B\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C} \\ &= \quad + \overline{0}\;0\;\overline{1} + 0\;\overline{0}\;\overline{1} + 0\;0\;\overline{1} \\ &= 0 + 0 + 0 + 0 \\ &= 0 \\ \text{i.e. } Y &= \overline{C} \end{aligned}$$

Example 2 : Simplify the following equation and then draw logic diagram and truth table.

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

Solution : Consider $Y = A\overline{B}C + A\overline{B}\overline{C} + B$

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

$= A\overline{B}(1) + B$ $(\because C + \overline{C} = 1)$

$= A\overline{B} + B$

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

The logic circuit is,

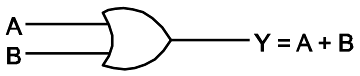


Fig. 56

Table 2.6 : Truth table

Inputs		**Output**
A	**B**	**Y = A + B**
0	0	0
0	1	1
1	0	1
1	1	1

Example 3 : Simplify the following Boolean equation and then draw logic diagram and truth table :

$$Y = AB\overline{C} + ABC + BC$$

Solution : $Y = AB\overline{C} + ABC + BC$

$Y = AB(\overline{C} + C) + BC$ $(\because C + \overline{C} = 1)$

$Y = AB(1) + BC$

$Y = AB + BC$

$Y = B(A + C)$

The logic circuit is,

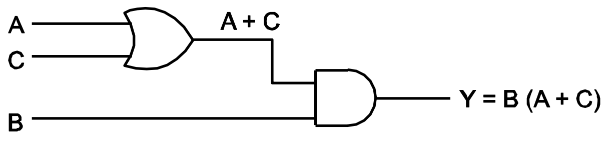


Fig. 57

Table 2.7 : Truth table

Inputs			**Output**
A	**B**	**C**	**Y = B (A + C)**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Example 4 : Minimise the equation y = $\bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + ABC$ using Boolean Algebra or K-maps.

Solution : (i) Using Boolean algebra : Rearranging the above

$$
\begin{aligned}
y &= \bar{A}\bar{B}C + A\bar{B}C + \bar{A}BC + ABC \\
&= (\bar{A} + A)\,\bar{B}C + (\bar{A} + A)\,BC \qquad (\because A + \bar{A} = 1) \\
&= \bar{B}C + BC = C\,(\bar{B} + B) \\
y &= C
\end{aligned}
$$

$$\therefore \quad y = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + ABC$$

(ii) Using K-map : Since equation contains three variables, the K-map will have $2^3 = 8$ cells.

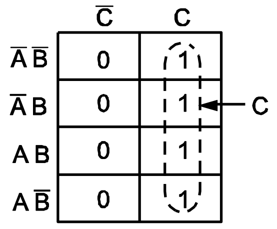


Fig. 58

$$y = C$$

**Example 5 : ** Convert the following SOP expression into standard SOP form :

$$Y = AB + AC + B\bar{C}$$

Solution : Given expression is

$$Y = AB + AC + B\bar{C}$$

Step 1 : Find the missing literal in each term

$$
\begin{array}{cccccc}
Y = & AB & + & AC & + & B\bar{C} \\
 & \uparrow & & \uparrow & & \uparrow \\
 & \text{missing} & & \text{missing} & & \text{missing} \\
 & \text{literal is} & & \text{literal is} & & \text{literal is} \\
 & C & & B & & A
\end{array}
$$

Step 2 : And each term with its (missing literal + its complement)

$$Y = AB\,(C + \bar{C}) + AC\,(B + \bar{B}) + B\bar{C}\,(A + \bar{A})$$

Since $C + \bar{C} = 1$, the value of expression does not change as

$$
\begin{aligned}
Y &= ABC + AB\bar{C} + ABC + A\bar{B}C + AB\bar{C} + \bar{A}B\bar{C} \\
&= ABC + ABC + AB\bar{C} + AB\bar{C} + A\bar{B}C + \bar{A}B\bar{C} \quad (\because A + A = A) \\
&= ABC + AB\bar{C} + A\bar{B}C + \bar{A}B\bar{C}
\end{aligned}
$$

Since in the above expression each term contains all the literals it is in the standard SOP form.

Example 6 : Simply the following using Boolean algebra.

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

Solution :

$$
\begin{aligned}
Y &= \bar{A}\bar{B}C + \bar{A}\cdot\bar{B}\cdot\bar{\bar{C}} + \bar{A}\bar{B}\bar{C}D && \text{Using } \overline{A + BC} = \bar{A}\cdot\bar{B}\cdot\bar{C} \\
&= \bar{A}\bar{B}C + \bar{A}\cdot\bar{B}\cdot C + \bar{A}\bar{B}\bar{C}D && \text{Using } \bar{\bar{A}} = A \\
&= \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C}D && (\because A + A = A)
\end{aligned}
$$

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

QUESTIONS

1. State various laws of Boolean algebra.
2. State and verify De Morgan's 1st and 2nd theorems.
3. Explain procedure for converting a logic circuit into NAND logic.
4. Explain how logic circuit can be converted to NOR logic circuit.
5. What is K-map ? Where is it used ? What are its advantages ? Explain 3 variable K-map with suitable example.
6. Draw logic symbols and truth tables for NAND, NOR, EX-OR and NOT gate.
7. Simplify following equations using laws of Boolean algebra :

   (i)$Y = AB + BC + \overline{B}A + \overline{A}B$, (ii) $Y = ABCD + ABC + AB + A\overline{B}$

   (iii)$Y = \overline{A} + AB + A\overline{B}$ (iv) $Y = AB + \overline{A}B + ABC$

8. Write a note on logic families.
9. Why NAND gate is called as universal building block ? Explain it with suitable example.

10. Use only NOR gate to build NAND, OR and EX-OR gates.
11. What are the logic families ? Give their different characteristics.
12. Design all basic gates using NOR gate.
13. Reduce the following Boolean expressions :

    (i)$\overline{A \cdot (A + C)}$, (ii) $(\overline{C} + B)(C + B)$, (iii) $AC\overline{D} + \overline{A}C\overline{D}$

    (iv)$A\overline{B} + ABC + A(B + A\overline{B})$
14. Minimise the following expressions by using K-map :

    (i)$ABC + \overline{A}B\overline{C} + B$, (ii) $\overline{A}B\overline{C}D + AB\overline{C}D + ABC\overline{D} + A\overline{B}C\overline{D}$
15. Draw a logic circuit and obtain truth table for following expression :

    $$Y = AC\overline{D} + \overline{A}BC$$
16. Convert the following expressions into their standard SOP and POS forms :

    (a)$Y=AB + AC + BC$

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

    (c)$Y=A + B + C + ABC$

---oOo---