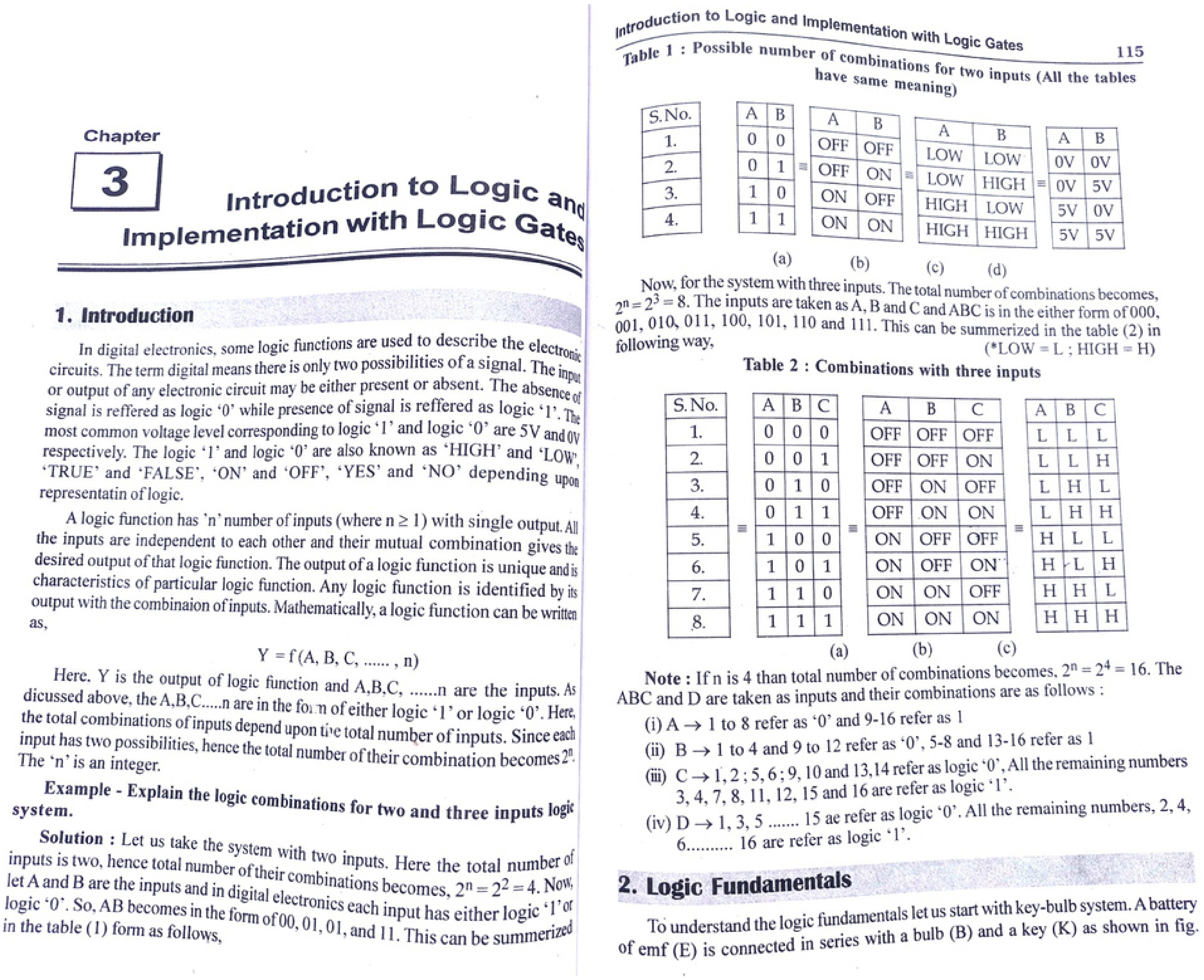
Unit-III

Introduction to Logic and implmentation with Logic Gates, functions-NOT, AND, OR, NOR, Ex-NOR, Truth tables, Boolean Algebra, De morgan's theorems; Standard forms of logical expressions, Sum of Products, Product of Sums specification of logical functions in terms of Minterms and Maxterms, Karnaugh Maps, simplification of logical functions, introduction of "don't care" states, Synthesis using only NAND or only NOR gates.

3. Introduction to Logic and Implementation with Logic Gates 114-228



Chapter 3

Introduction to Logic and Implementation with Logic Gates

1. Introduction

In digital electronics, some logic functions are used to describe the electronic circuits. The term digital means there is only two possibilities of a signal. The input or output of any electronic circuit may be either present or absent. The absence of signal is reffered as logic '0' while presence of signal is reffered as logic '1'. The most common voltage level corresponding to logic '1' and logic '0' are 5V and 0V respectively. The logic '1' and logic '0' are also known as 'HIGH' and 'LOW', 'TRUE' and 'FALSE', 'ON' and 'OFF', 'YES' and 'NO' depending upon representatin of logic.

A logic function has 'n' number of inputs (where $n \geq 1$) with single output. All the inputs are independent to each other and their mutual combination gives the desired output of that logic function. The output of a logic function is unique and is characteristics of particular logic function. Any logic function is identified by its output with the combinaion of inputs. Mathematically, a logic function can be written as,

$$Y = f(A, B, C, \ldots\ldots, n)$$

Here, Y is the output of logic function and A,B,C, ...n are the inputs. As dicussed above, the A,B,C.....n are in the form of either logic '1' or logic '0'. Here, the total combinations of inputs depend upon the total number of inputs. Since each input has two possibilities, hence the total number of their combination becomes 2^n. The 'n' is an integer.

Example - Explain the logic combinations for two and three inputs logic system.

Solution : Let us take the system with two inputs. Here the total number of inputs is two, hence total number of their combinations becomes, $2^n = 2^2 = 4$. Now, let A and B are the inputs and in digital electronics each input has either logic '1' or logic '0'. So, AB becomes in the form of 00, 01, 01, and 11. This can be summerized in the table (1) form as follows,

Table 1 : Possible number of combinations for two inputs (All the tables have same meaning)

S.No.	A	B		A	B		A	B		A	B
1.	0	0		OFF	OFF		LOW	LOW		0V	0V
2.	0	1	≡	OFF	ON	≡	LOW	HIGH	≡	0V	5V
3.	1	0		ON	OFF		HIGH	LOW		5V	0V
4.	1	1		ON	ON		HIGH	HIGH		5V	5V

(a) (b) (c) (d)

Now, for the system with three inputs. The total number of combinations becomes, $2^n = 2^3 = 8$. The inputs are taken as A, B and C and ABC is in the either form of 000, 001, 010, 011, 100, 101, 110 and 111. This can be summerized in the table (2) in following way,

(*LOW = L ; HIGH = H)

Table 2 : Combinations with three inputs

S.No.		A	B	C		A	B	C		A	B	C
1.		0	0	0		OFF	OFF	OFF		L	L	L
2.		0	0	1		OFF	OFF	ON		L	L	H
3.		0	1	0		OFF	ON	OFF		L	H	L
4.		0	1	1		OFF	ON	ON		L	H	H
5.	≡	1	0	0	≡	ON	OFF	OFF	≡	H	L	L
6.		1	0	1		ON	OFF	ON		H	L	H
7.		1	1	0		ON	ON	OFF		H	H	L
8.		1	1	1		ON	ON	ON		H	H	H

(a) (b) (c)

Note : If n is 4 than total number of combinations becomes, $2^n = 2^4 = 16$. The ABC and D are taken as inputs and their combinations are as follows :

(i) A → 1 to 8 refer as '0' and 9-16 refer as 1

(ii) B → 1 to 4 and 9 to 12 refer as '0', 5-8 and 13-16 refer as 1

(iii) C → 1, 2 ; 5, 6 ; 9, 10 and 13,14 refer as logic '0', All the remaining numbers 3, 4, 7, 8, 11, 12, 15 and 16 are refer as logic '1'.

(iv) D → 1, 3, 5 .... 15 ae refer as logic '0'. All the remaining numbers, 2, 4, 6.......... 16 are refer as logic '1'.

2. Logic Fundamentals

To understand the logic fundamentals let us start with key-bulb system. A battery of emf (E) is connected in series with a bulb (B) and a key (K) as shown in fig.

(3.1). Now let us suppose the bulb is glowing and batery works properly. Than above two statements are certain. But this only possible when key is in **ON** state and the bulb is also OK (Not fused).

If battery works properly, than **Bulb glows** only if **key is ON** and **bulb is OK.**

Here, brightnes (glow) of bulb is taken as output and it depends upon status of key and condition of bulb, so these are taken as inputs. Hence inputs are conditional they have two posibilities either key is ON or OFF and bulb is OK or fused, these may be taken as logic '1' and logic '0' respectively. Finally, the output of key bulb system is decided by the combinations of the inputs. Conclusion of above example can summerized as,

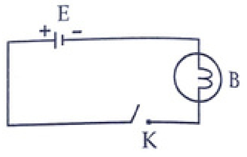


Fig. (3.1) : Key bulb system

Table 3 : Truth table for key bulb system

Is bulb OK ?	Status of key	Is bulb glow ?
YES	ON	YES
YES	OFF	NO
NO	OFF	NO
NO	ON	NO

Hence, logics are either True (1) or False (0) and any other possibilities is not acceptable in digital electronics.

3. Logic Gates

A logic gate is a device that means it has the ability to take decisions on the basis of given input combinations. Logic gates are the basics of digital electronics. A IC (integrated circuits) has large number of logic gates depending upon its scale i.e. LSI (Large Scale Integrated Circuits), MSI (Medium Scale IC) or even in VLSI (Very Large Scale IC). Different types of logical operations can be performed by interconnection of different logic gates, such a interconnection is known as logic design.

The different types of gates are known as logic circuits due to their analysis with boolean algebra.

In 1854, ***George Boole*** a mathematician invented a symbolic logic known as boolean algebra. This is very useful method to solve various logical expressions. He gave in his algebra that every input variable has either of two values : HIGH or LOW, TRUE or FALSE.

Before, 1938, there no practical use of george Boole's work. The ***Claude Shanon*** was the first person who used Boole' work in telephone switching circuits. After this, it is used to apply in various electronic circuits of computers and other devices.

Basic or fundamental gate- A circuit having one or more than one inputs with single output is called gate. All the gates are two state circuits because all the inputs and outputs are either HIGH or LOW values in digital form.

There are three types of basic gates as follows,

(1) Inverters or the 'NOT' Gate

(2) All or Nothing gate or the 'AND' Gate

(3) Any of all gate or the 'OR' Gate

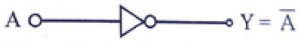
(I) Inverters or the 'NOT' Gate

The term inverter means any stage of input is converted into its compliment. A 'NOT' gate has only one input and also have only one output. A NOT gate is also called a inverter. If the input of NOT gate is in the form of '0' than output is in the form of logic '1'. the logic function can be written as,

If A is the input, then,

$$Y = \overline{A}$$
$$= A'$$
$$= \text{NOT } A \qquad .....(1)$$

and logic symbol, for NOT gate is,



Truth table for 'NOT' gate is given by,

A	Y
0	1
1	0

≡

A	Y
LOW	HIGH
HIGH	LOW

≡

A	Y
ON	OFF
OFF	ON

The electronic circuit of 'NOT' or inverter gate can be realized by using a transistor and resister as (Resister-Transistor Logic RTL) as shown in fig. (3.2).

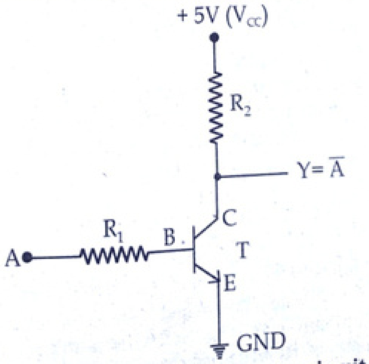


Fig. 3.2 : The transistor inverter circuit

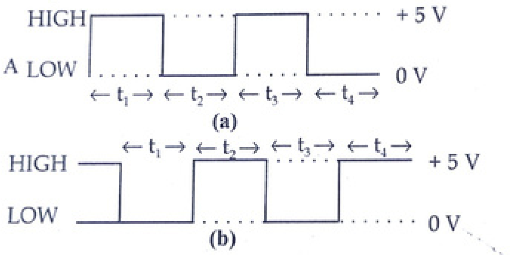
(a) Logic Operation :

The input of inverter is either 0 V or + 5V at the terminal A as shown in figure 3. In case of 0 V of input : A = 0 V, the transistor T is in cut off region and therefore it remains in OFF state. As a result no current flows through resistance R_2 and consequents no voltage drop across R_2. This combination keeps the output Y in + 5V, since + 5V is supplied already in the circuit. On the other hand the input is in the form of +5V or ON state, the transistor T goes into saturation region or T is in ON state. As result current passes through resistance R_2 and entire voltage drops across R_2 and the corresponding output Y in the case becomes 0 V. The truth table of entire operation is.

Input, A	Output, Y
0 V	5 V
5 V	0 V

(b) Time Diagram

Let a input A is applied to a NOT gate. The waveform of A is HIGH and LOW for a particular time period.


HIGH
A LOW
+ 5 V
0 V
← t_1 → ← t_2 → ← t_3 → ← t_4 →
← t_1 → ← t_2 → ← t_3 → ← t_4 →
Y
HIGH
LOW
+ 5 V
0 V

(a)

(b)

Fig. 3.3 : Time diagram of NOT gate (a) Input (b) Output

The output of NOT Gate is the inverter complement of A, so, during the time t_1 A is HIGH or + 5V, so Y is low or 0V during the same time interval t_1, Again, In the interval t_2 A is low or 0V, so Y becomes HIGH or + 5V and so on.

(2) All or Nothing/ The AND Gate

The 'AND' Gate is also a fundamental gate. The single output of 'AND' gate may obtained by two or more inputs. The minimum number of inputs to perform a 'AND' operation is two. For two inputs A and B are either in '1' state or in '0' state. The output Y is the function of their combination given by,

$$Y = f(A, B)$$
$$Y = A.B$$
$$= AB$$

The symbol *dot* (.) between input notation is used to exhibit it mathematically.

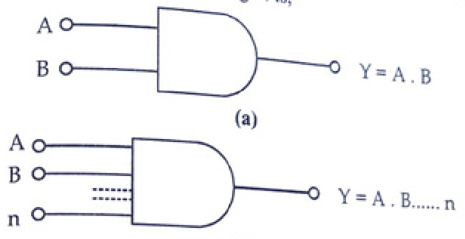
The output Y is in the '0' state if both A and B are in '0' state. The logic state of Y is '0' even one of the input is in '0' state. But, when both the A and B are in logic '1' state, then only the output of 'AND' gate becomes logic '1' state.

The 'AND' gate is also known as an "All" or "nothing" gate because the output of 'AND' gate is similar to a device whose output gives logic stage '1' if and only if all the its inputs are in logic state '1'. For many inputs the output is defined as,

$$Y = f(A, B, C, \text{......} n)$$
$$Y = A . B . C \text{.....}$$
$$Y = ABC\text{.......}$$

This reads as Y equals to A dot B dot C dot..... or A and B and C and ....

The logical symbol used for 'AND' gate is,



(a)

(b)

Fig. 3.4 (a) Logic symbol for 'AND' gate with two input and (b) 'AND' gate with n inputs

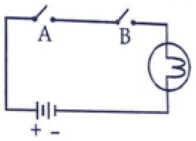


Fig. 3.5 : Two variable switching circuits

The **'Truth Table'** for 'AND' gate is given by,

Table 4 : Truth Table for 'AND' gate with two inputs

Input		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

≡

Input		Output
A	B	Y
L	L	L
L	H	L
H	L	L
H	H	H

Here L stans for LOW corresponds to logic state '0' while H stands for HIGH corresponds to logic state '1'.

Table 5 : Truth table for 'AND' gate with three inputs.

Input			Output		Input			Output
A	B	C	Y		A	B	C	Y
0	0	0	0		L	L	L	L
0	0	1	0		L	L	H	L
0	1	0	0		L	H	L	L
0	1	1	0		L	H	H	L
1	0	0	0	≡	H	L	L	L
1	0	1	0		H	L	H	L
1	1	0	0		H	H	L	L
1	1	1	1		H	H	H	H

Characteristcs of AND gate

(1) The output of AND gate is logic '1' only if all the inputs are logic '1'.

(2) If any one or more inputs are logic '0' the output will be logic '0'.

(a) Circuit Realization of 'AND' Gate

The electronic output of 'AND' gate may obtained by using one of the following circuit combinations.

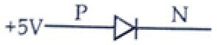
(1) Diode-Resistor Logic (DL)

(2) Resistor-Transistor Logic (RTL)

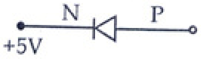
The inputs A and B to the gate is taken either 0V or + 5V in their the combinations.

(1) Diode Resistor Logic (DL) : For DL operation the switching properties of diode are used.

As we know any diode has taken in 'ON' state only when it is forward biased. (Positive terminal of diode is connected with positive end and negative terminal of diode is connected with negative end). Similarly, A diode may regarded as 'OFF' state, when it is in reverse biased. (Negative terminal of diode is connected with positive end and positive terminal of diode is connected with negative end).



'ON' State or diode conducts



'OFF' State or diode not to conduct

The above property of diode is used to perform the logic gate output.

Hence, two input diode 'AND' gate is shown in fig. (3.6).

When both the inputs A and B are + 5V than both diodes D_1 and D_2 are in reverse bias and both of them regarded s 'OFF' state. In such a case no current flows through the resistance, R and therefore no voltage drops across resistance, R. It gives the output Y as +5V, that means logically stae '1'. In remaining three cases :(i) A = 0 V ; B = +5V (ii) A = + 5V ; B = 0V (iii) A = 0V ; B = 0 V. D_1 is in 'ON' state and D_2 is 'OFF' for case (i). D_2 is 'ON' in case (iii), the entire voltage drops across resistance R and therefore the output Y becomes 0 V. Practicaly 0V means the voltage near to 0 V. Truth table will be :

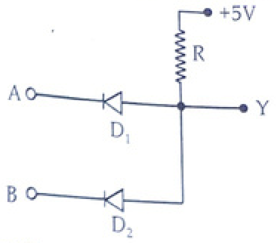
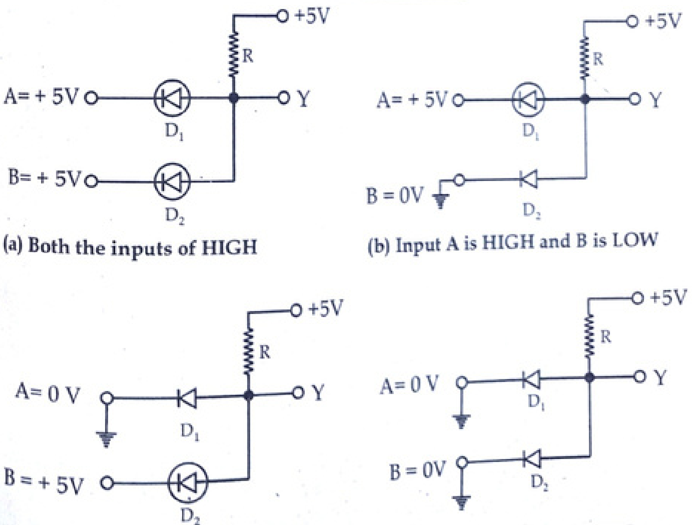


Fig. 3.6 : The two input DL 'AND' gate

Inputs		Output
A	B	Y
0 V	0 V	0 V
0 V	5 V	0 V
5 V	0 V	0 V
5V	5 V	5 V



(a) Both the inputs of HIGH

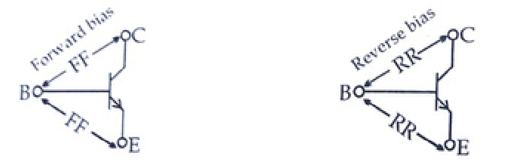
(b) Input A is HIGH and B is LOW

(c) Input A is LOW and B is HIGH

(d) Both the inputs at LOW

Fig. 3.7: A- 2-Input 'AND' gate (Encrided diode shows the reverse bias state)

(2) Resistor-Transistor Logic (RTL) : For RTL operation the switching properties of a transistor are used. As we know that any transistor in 'ON' state may regarded logically '1' state and 'OFF' stae may regarded logically '0' state. The 'ON' state of the transistor occurs when it is in saturation region (Both the junctions BC and BE of transistor are in forward biased), while the 'OFF' state occurs when it is in cut-off region (both the junctions BC and BE of transistor are ɪeverse biased) as shown in fig. (3.8).



Saturation or 'ON' state

Cut off or 'OFF' state

Fig. 3.8 : Switching properties of transistor

The above properties of transistor is used to perform the logic gate output. Here, two inputs Resistor-Trasistor Logic AND gate is shown in fig. (3.9).

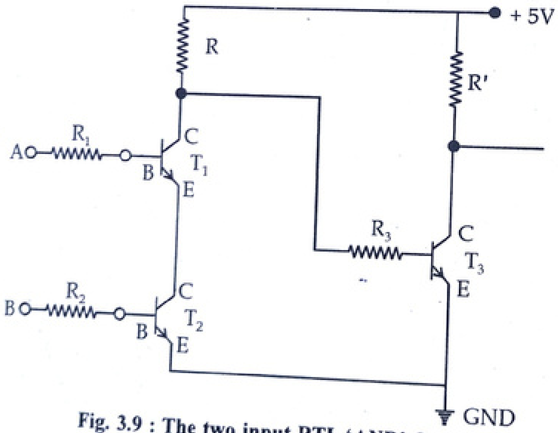


Fig. 3.9 : The two input RTL 'AND' Gate

Again, here the inputs A and B are eiether 0 V or + 5V. The logic state '0' means near 0V and logic state '1' means near to +5V.

For RTL 'AND' gate, when both A and B are at + 5V, both the transistors T_1 and T_2 are in saturation or 'ON' state, and therefore, the voltage at the collector end of transistor T_1 will drops down, so that the transistor T_3 does not get enough voltage

to drive because base voltage of T_3 is unable, therefore T_3 remains in cut off region or OFF state. As a result no current flows through resistor R' and therefore no voltage drops across R' consequently the output Y becomes ≃ 5 V (it is treated as logic state '1'). Now in the remaining three cases when (i) A = 0V ; B = +5 V (iii) A = + 5V; B = 0V and (iv) A = 0V ; B = 0V both the ransisors T_1 and T_2 are in cut-off region or 'OFF' state. Hence, the transistor T_3 in satuation or 'ON' state. As a result voltage drops across R' (Collector Resistace of T_3). So the output Y becoms Y ≅ 0V (It is treated as logic '0').

The truth table of RTL 'AND' sate is similar to the truth table of DL 'AND' gate.

(b) Time diagram

Let two inputs A and B are applied to a 'AND' gate. The waveforms of A & B are shown in figure. They are 'HIGH or 'LOW' for a particular time period.

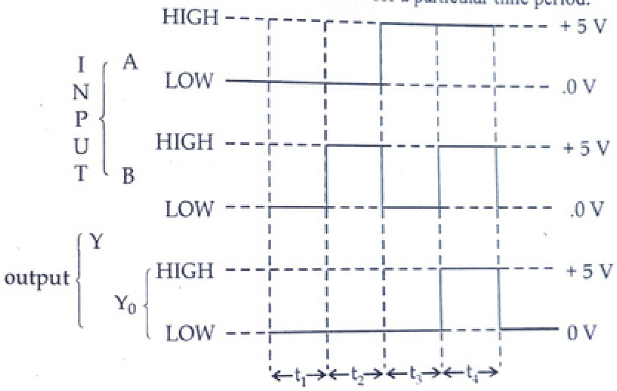


Fig. 3.10 : Time diagram of AND gate

In the time interval t_1. Both the inputs A & B are Low , so their combination to AND gate gives. Low output. Again in the time interval t_2, A is Low and B is HIGH & the AND output for this combination gives Low output. Similar results occur in time interval t_3 here A is HIGH and B is Low. But when both the inputs A & B are 'HIGH' in the time interval t_4 the output of Y of AND gate also becomes HIGH. The schematic representation of time diagram (time - voltage behaviour) for AND gate is shown in figure Fig. (3.10).

(3) Any or All gate/The OR Gate : Third and last type of fundametal gate is the OR gate. The single output of 'OR' gate may obtained by two or more inputs. Like 'AND' gate, the minimum number of inputs to perform a 'OR' operations is two.

The output Y, of 'OR' gate is the functions of the combination of inputs i.e. A &

B, and it is given by.

$$Y = f(A, B)$$
$$Y = A + B$$

Mathematicaly, the symbol plus '+' between input noation is used to exhibit an 'OR' gate.

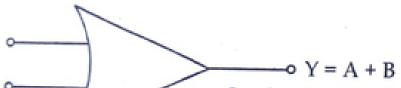
The output Y is the '0' state if both A and B are in '0' state. The logic state of Y is then '0'. When any input signal is in '1' state, then output of the 'OR' gate becomes '1', obviousely, both A & B are in '1' state give the output in '1' state.

The 'OR' galte is also known as 'Any' or 'All' gate, because the output of 'OR' gate is similar to a device whose output is '1' even if one of its inputs is I. For many inputs the output is defined as,

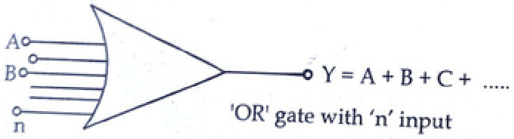
$$Y = f(A, B, C, ... n)$$
$$Y = A + B + C + ...$$

This reads as Y equals to A plus B plus C ...... or A or B or C or .....

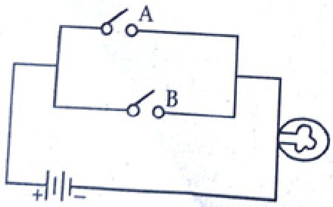
The logical symbol used for 'OR' gate is as follows :



Symbol for two input 'OR' operation



'OR' gate with 'n' input



Equivalent 2-input OR gate

Table 6 : Truth table of OR Gate with two inputs

Inputs		Output Y
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

≡

Input		Output
A	B	Y
L	L	L
L	H	H
H	L	H
H	H	H

Table 7 : Truth table with three inputs

Inputs			Output
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

≡

Inputs			Output
A	B	C	Y
L	L	L	L
L	L	H	H
L	H	L	H
L	H	H	H
H	L	L	H
H	L	H	H
H	H	L	H
H	H	H	H

Characterstic of OR gate

(1) The output of an OR gate is logic '0' only when inputs are logic '0'

(2) The output an OR gate is logic '1', when any one or more inputs are logic '1'.

(a) Circuit realization of OR gate

Y

Similar to 'AND' gate, the electronic output of 'OR' gate may obtained by using one of the following two circuits combinations.

(1) Diode Resistor Logic (DL)

(2) Resistor-Transsistor Logic (RTL)

Here, also, the inputs A and B to the 'OR' gate is taken either 0V (nearly equal to 0 V) or + 5 V (Nearly equals to +5V) in their different combinations.

(1) Diode Resistor Logic (DL) : The logic operations of 'OR' gate is accomplished by using resistors and diodes. Some standard values of resistor & diode is choosed such that they are able to perform the operation. In this logic, the switching properties of diode are used as dissiused earlier. It is summetrized as

Bias	Current	State
Diode in forward bias	Current passes through it	ON
Diode in reverse bias	No current passes through it	OFF

Hence, the two-input 'OR' gate is shown in fig. (3.11).

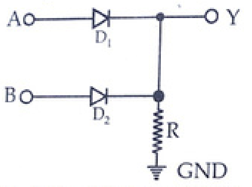


Fig. 3.11 : OR gate with DL

There are four combination of A and B i.e.

(i) A = 0V; B=0V

(ii) A = + 0 V; B = + 5 V

(iii) A = + 5V; B = 0 V

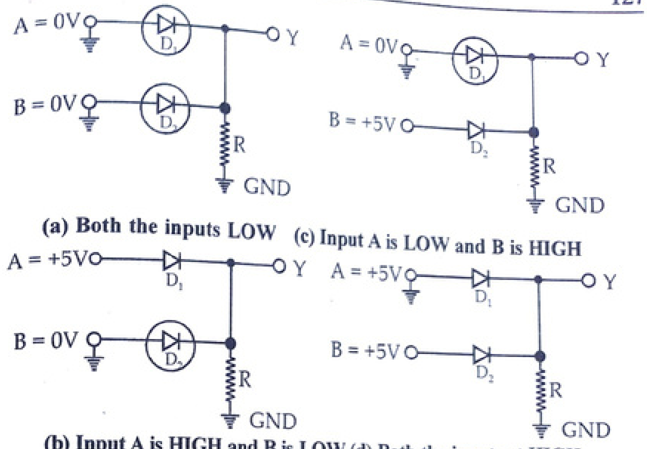
(iv) A = + 5 V; B = + 5 V

In the first combination when both A = 0 V and B = 0 V, both the diodes D_1 & D_2 are reverse biased and they do not conduct & no current flow through them, as a result no current passes through R & hence no voltage drops across R. Consequently, the ouput Y becomes 0V (Y = 0V).

In either of remaining three cases atleast one diode (D_1 or D_2) or both the diodes D_1 & D_2 are in forward biased and conducts repectively. As a result correponding diode goes into ON state or acts as short circuit and, therefore the output Y becomes + 5V (Y = + 5 V)

The truth table or 'OR' operation will be,

Inputs		Output Y
A	B	Y
0V	0V	0V
0V	5V	5V
5V	0V	5V
5V	5V	5V



(a) Both the inputs LOW **(c) Input A is LOW and B is HIGH**

(b) Input A is HIGH and B is LOW **(d) Both the inputs at HIGH**

Fig. 3.12 : A 2-Input 'OR' gate (Encircuited diodes shows Reverse bias state)

(2) Resistor-Transistor Logic (RTL)

In RTL operation, the 'OR' gate output can be understood on the basis of the switching properties of Transistor. Again, the values of resistance and transistor (transistion parameter) are choosen such that they performed as desired operation. The switching properties of transistor can be summerized as,

Bias	Base status	Status
Both the junctions CB and BE of transistor are in forward bias	Enough voltage to drive base	ON (saturation)
CB and BE are in reverse bias	Available voltage is not enough to drive base	OFF (Cut-off)

Now, the Two-input Resistance-Transistor OR gate is shown in fig. (3.13).

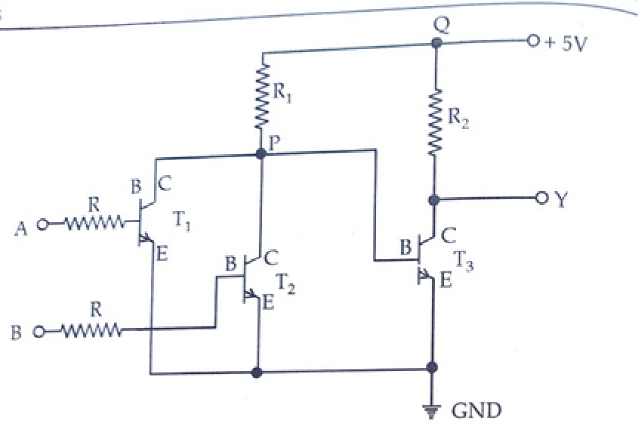


Fig. 3.13 : Two-Input RTL 'OR' gate

For RTL 'OR' gate, these are four combinations of A and B are possible.

(i) A = 0 V ; B = 0V (ii) A = 0V ; B = + 5 V

(iii) A = + 5V ; B = 0 V (iv) A = + 5V ; B = + 5 V

In the first combination, when both A and B are at 0 V, then transistor T_1 and T_2 have not enough voltage to drive or they lies in cut-off region. As a result T_1 and T_2 are 'OFF' state. In this case transistor T_3 gets enough voltage through R, to drive its base, therefore T_3 will turn into saturation region and will be in ON state. Hence, the output at Y becomes 0 V, (Y = 0V). This is because + 5V is already maintained above R_2 at point Q.

In either of remaining three cases, the corresponding transistor T_1 or T_2 or both T_1 and T_2 are in 'ON' state and therefore the voltage at the collection end of T_1 or T_2 is 0 V, this is because point P is already maintained at + 5V through R_1. As a result, these combinations does not provide sufficient voltage to drive the base of transistor T_3. Hence transistor T_3 remains in cut-off region or T_3 is said to be 'OFF' in rest of three cases. Consequently the output at Y becomes + 5V (Y = + 5V). This is because + 5V is already maintained at point Q.

The truth table of RTL 'OR' gate is similar to the truth table of DL 'OR' gate.

(b) Time diagram

Let two inputs A and B are applied to a 'OR' gate. The waveforms of A and B are given in fig. (3.14). They are HIGH or LOW for a particular time period.

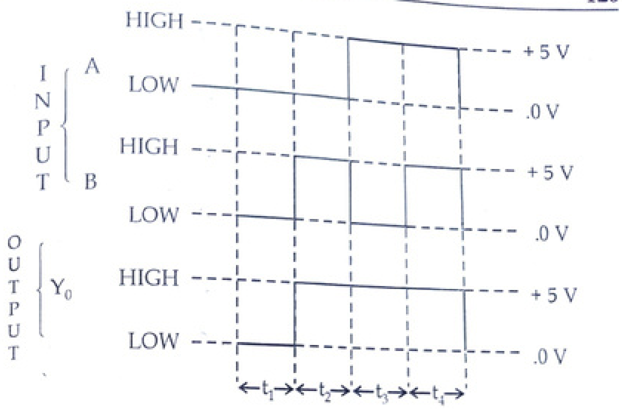


Fig. 3.14 : Time diagram for OR gate

In the time interval t_1 both the inputs A and B are LOW, so their combination to OR gate gives LOW output. Again, in time interval t_2. A is LOW and B is HIGH, the 'OR' output for this combination will be HIGH. Similar output will be shown in time interval t_3 when A is HIGH and B is LOW. For A and B both are HIGH, the output of OR gate for such combination will be HIGH as shownn time interval t_4.

*(**Note :** Here + 5V means, it represents the logic state '1' and 0 V means, it represents the logic state '0'. Any voltage above +5V is also treated as logic '1'.

4. The Universal gates

Out of above mentioned gates, the first two, NAND and NOR gates are known as universal gate. The term universal is used because any gate can be designed by using only one type of gate either NAND or NOR gate. It means each of NAND and NOR gate can realize electronic logic circuit singled-handelly. The 'NAND' and 'NOR' both the gates are able to p.rform fundamental logic functions i.e. AND, OR and NOT. Therefore 'NAND' and 'NOR' are universal building blocks of digital electronics.

NAND Gate

The term 'NAND' means NOT AND, so it is gate that can make by the used of an AND gate and a NOT gate. Since a NOT gate gives the complementry output of its input, hence the output of NAND gate gives the complementary of the output of AND gate.

Maathematically, it is defined as,

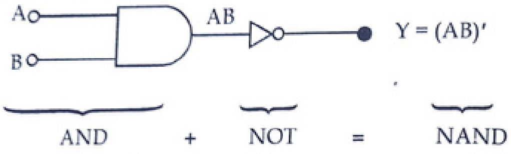
$$Y = \text{NOT AND} = f(A,B)$$
$$= \text{NOT (AB)}$$

$= (AB)'$

$= \overline{AB}$

The operation is completed AND gate followed by NOT gate.

The output of NAND is in '0' state only when all the inputs are in logic '1' state. All the remaing combinations of gives the output as logic '1'. The symbolic representation of NAND gate will be,



or it is also represented by,

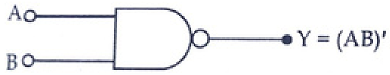


Fig. 3.15 : Two input NAND gate

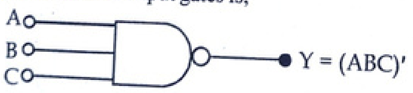
The truth table for NAND gate will be,

Inputs		Output
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

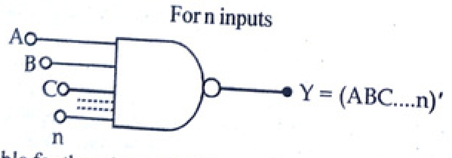
≡

Inputs		Output
A	B	Y
L	L	H
L	H	H
H	L	H
H	H	L

The logical symbol for three input gates is,



For n inputs



The truth table for three input NAND gate is,

Inputs			Output
A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	1	1	1
1	0	0	1
1	1	1	0

Bubbled OR gate/ Negative OR gate

A negative OR gate is that in which all the inputs are in inverted or complementry form. The output of a NAND gate is similar to a negative OR gate. a negative OR gate is also known as Bubbled OR gate. As we know that the output of NAND gate is in logic state '1' even if any one of its inputs is in logic state '0' hence, the NAND gate is also known by an active LOW or gate.

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

$$= A' + B'$$

Inputs		Complementry Inputs		Output
A	B	A′	B′	Y
0	0	1	1	1
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

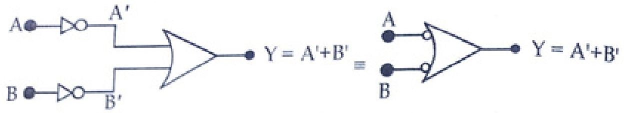
From above truth table we see, when A = 0 B = 0 than output becomes '1'. Also, the output is in logic state '1' for A = 0 ; B = 1, and A = 1 ; B = 0. But for A = 1, B = 1, the output Y becomes '0'. This is similar to a NAND gate.

Here, it is also concluded that,

$$Y = \overline{AB} = (AB)' \quad \text{(NAND gate)}$$

$$= \overline{A} + \overline{B} = A' + B' \quad \text{(Bubbeled OR/Negative OR gate)}$$

The bubbled OR gate.



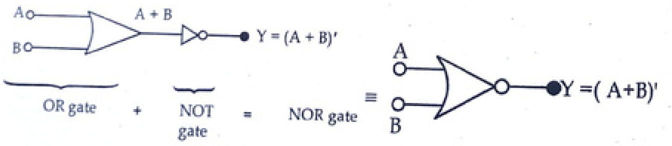
Here. The NAND gate can also be performed by first inverting the inputs and than ORing the inverted inputs.

The NOR Gate

The term NOR means NOT OR, so it is gate that can make by the use of an OR gate and a NOT gate. Since a NOT gate gives the output in complementry from of its input, hence the output of NOR gate gives the complementry of the output of OR gate. Mathematically, it is defined as,

$$\begin{aligned} Y &= \text{NOT OR} = f(A, B) \\ &= \text{NOT } (A + B) \\ &= (A + B)' \\ &= \overline{A + B} \end{aligned}$$

The operation is completed by an OR gate followed by NOT gate. The output of NOR is in logic '1' state only when all the inputs are in logic '0' state. All the remaining combinations gives the output as logic '0'. The symbolic representation of NOR gate will be



The truth table for NOR gate will be,

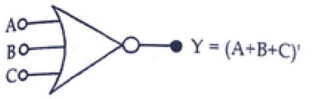
Inputs		Output
A	B	Y
0	0	1
0	0	0
1	1	0
1	1	0

≡

Inputs		Output
A	B	Y
L	L	H
L	H	L
H	L	L
H	H	L

The truth table for three input NOR gate is ,

Inputs			Output
A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0



three input NOR get

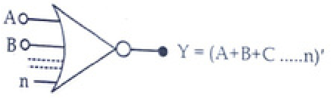


Fig. n-inputs NOR gate

Bubbled AND gate/Negative AND gate

Similar to negative OR gate, the negative AND gate is that in which all the inputs are in invered or complementry form. The output of a NOR gate is similar to a negative AND gate. A negative AND gate is also called Bubbled AND gate. As we know that the output of NOR gate is in logic state '1' only when all the its inputs are in logic state '0', hence, the NOR gate is also known by an active-low AND gate. Mathematycally,

$$\begin{aligned} Y &= \bar{A} . \bar{B} \\ &= A'.B' \end{aligned}$$

Truth table for negative AND gate

Inputs		Complementry Inputs		Output
A	B	A′	B′	Y
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

From above truth table we see that when A = 0 ; B = 0, the output becomes in logic state '1'. For all the remaining combinations of A and B e.g. A = 0 ; B = 1, A =

1 ; B = 0; A = 1; B = 1, the output is in ogic state '0'. This is similar to a NOR gate. Hence, the NOR gate can also be performed by first inverting the inputs and than ANDing the inverted inputs.

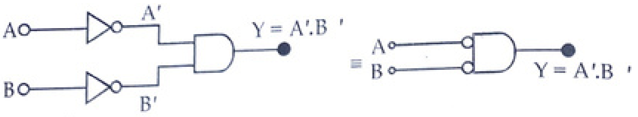


Fig. 3.16 : The bubbled AND gate

It is concluded from the above expression, that

$$Y = (A + B)' = \overline{A + B} \text{ (NOR Gate)}$$

or

$$A'.B' = \overline{A} \,.\, \overline{B} \text{ (Bubbled AND Gate)}$$

5. The Exclusive-OR gate/EX-OR Gate

The exclusive-OR gate is also known as 'EX-OR' gate or X-OR' gate. Because of its distinct characteristics, it is widely used in digital devices. It is used for parity checking, addition of binary numbers, code conversion etc.

The 'EX-OR' gate is a two input with single output. The output of 'EX-OR' gate is in logic '1' state, if one and only one of its two inputs is in logic '1' state. For remaining two combinations, when both the inputs are in logic '0' state or when both the inputs are in loic '1' state, the output becomes a logic '0' state.

Since we know that the fundamental 'OR' gate is also known as 'Inclusive OR gate. So if we compare both the 'Inclusive-OR' gate and 'Exclusive-OR gate. It can be noticed that either A or A or B both A and B are in logic state '1' the resultant OR gate output is in logic state '1'. However, for exclusive-OR gate, either A or B, but not both, must be in logic stat '1', the output of 'EX-OR' gate becomes logic '1' state. If both A and B are an logic state '1' at the same time the output of 'EX-OR' will be in logic state '0'. Due to its special output it is known by anti-coincidence gate. It is also called an inequility detector.

Truth Table : 8 : The OR gate and the 'EX-OR' gate

Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

The 'EX-OR' gate is known as Exclusive-OR gate because it excludes the condition if both the inputs are in logic '1' state. The output of 'EX-OR' gate is in logic state '1' only if exclusively one of its inputs is in logic '1' state.

Now, if we denote the input variables A and B for the 'EX-OR' gate and the output is denoted by Y, then the expression for 'EXOR' gate can be written as

Y = A EX-OR (It reads as Y is equal to a ex-or B).

$$Y = A \oplus B = A \text{ 'Ex-OR' } B$$

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

$$= AB' + A'B \qquad .....(1)$$

Truth table for logical expression given by equation (1) is

A	B	$\overline{A}$	$\overline{B}$	$A\overline{B}$	$\overline{A}B$	$Y = A\overline{B} + \overline{A}B$	$Y = A \oplus B$
0	0	1	1	0	0	0	0
0	1	1	0	0	1	1	1
1	0	0	1	1	0	1	1
1	1	0	0	0	0	0	0

The 'EX-OR' gate can also be written as,

$$Y = A \oplus B$$

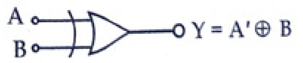
$$= (A + B)\,(\overline{AB})$$

$$= (A+B)\,(AB)' \qquad .....(2)$$

Truth table for logical expression given by equation (2) is,

A	B	(A + B)	AB	$(\overline{AB})$	$Y = (A + B)\,(\overline{AB})$	$Y = A \oplus B$
0	0	0	0	1	0	0
0	1	1	0	1	1	1
1	0	1	0	1	1	1
1	1	1	1	0	0	0

The symbolic representation of 'EX-OR' gate is given by,



The 'EX-OR' Gate

Mathematically, the summary of EX-OR gate.

$$Y = A \oplus B$$

$$0 = 0 \oplus 0$$

$$1 = 0 \oplus 1$$

$$1 = 1 \oplus 0$$

$$0 = 1 \oplus 1$$

The characteristics of 'EX-OR' gate are,

(1) No existance is possible for three or more (more than two) variable 'EX-OR' gate. If more than two variables are two be 'EX-ORed' a number of two input 'EX-OR' gates will be used.

(2) The 'EX-OR' gate is enabled with an odd number of inputs as logic state '1'.

(3) The 'EX-OR' gate is disabled by (the output is in logic state '0') by an even number of inputs as logic state '1'.

(4) Thus, the 'EX-OR' gate can be used as an odd-bits checker.

6. The Exclusive-NOR Gate/"EX-NOR" gate

The Exclusive-NOR gate is also known as "Ex-NOR' gate or 'X-NOR' gate. It has also distinct characteristics over other logic gates. In digital electronics, the 'EX-NOR' gate is used for comparing the equality of two inputs. An 'EX-NOR' is the combination of an 'EX-OR' gate and a NOT gate. So, it is regarded as an 'EX-OR' gate followed by an inverter (NOT gate).

The 'EX-NOR' gate is also a two input with single output. the output of 'EX-NOR' gate is in logic '1' state only when both the inputs are in logic '0' state or when both the inputs are in logic '1' state. The output of 'EX-NOR' gate is inlogic '0' state when one of the inputs is in logic '0' state and other is in logic '1' state. The output of 'EX-NOR' gate is in logic '1' state when its inputs coincide (ehter both inputs are in '0' state or in '1' state), hence, it is also known as coincidence gate.

Also, its output is in logic '1' state when its inputs are equal, hence it is an equality detector.

The truth table for 2-input 'EX-NOR' gate is given in Table.

Table 9 : Truth table for 2-onput 'EX-NOR' Gate

Input		Output		Input		Output
A	B	Y		A	B	Y
0	0	1	≡	L	L	H
0	1	0		L	H	L
1	0	0		H	L	L
1	1	1		H	H	H

Now, if we denote the input variables A and B for the 'EX-NOR' gate and the output is denoted by Y, then the expression for 'EXNOR' gate can be written as

Y = A⊙B (It reads as Y is equal to A ex-nor B)

$$= AB + \overline{A}\,\overline{B} \qquad .....(1)$$

Truth table for logical expression gien by equation (1) is,

A	B	$\overline{A}$	$\overline{B}$	AB	$\overline{A}\,\overline{B}$	$Y = AB + \overline{A}\,\overline{B}$	Y = A.B
0	0	1	1	0	1	1	1
0	1	1	0	0	0	0	0
1	0	0	1	0	0	0	0
1	1	0	0	1	0	1	1

The 'EX-NOR' gate expression can also be written as,

$$Y = A \odot B = A \text{ 'EX-NOR' } B$$

$$= \overline{A \oplus B}$$

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

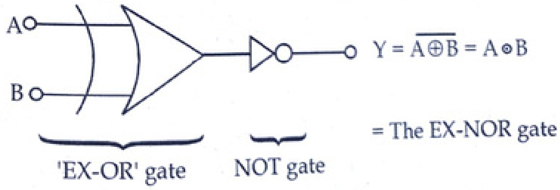
$$= (AB' + A'B)' \qquad .....(2)$$

Also, $Y = (A+B)' + AB$

and the truth table is :

A	B	A + B	(A + B)′	AB	Y = (A + B)′ + AB	Y = A . B
0	0	0	1	0	1	1
0	1	1	0	0	0	0
1	0	1	0	0	0	0
1	1	1	0	1	1	1

The symbolic representation of 'EX-NOR' gate is given by:



or,

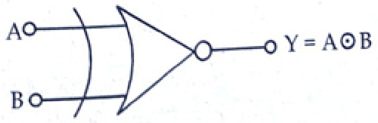


Fig. 3.17 : The 'EX-NOR' Gate

Mathematically, the summury of 'EX-NOR' Gate is,

$$Y = \overline{A \oplus B} = A \odot B$$

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

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

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

$$1 = \overline{1 \oplus 1} = 1 \odot 1$$

Hence, the output of 'EX-NOR' gate is the complement of the output of the 'EX-OR' Gate.

The characteristics of 'EX-NOR' gate are, [Parity Checker]

(1) No existance is possible for three or more (more than two) variable 'EX-NOR' gate. If more than two variables are to be EX-NORed, a number of two-input 'EX-NOR' gates will be used.

(2) The 'EX-NOR' gate is enabled with even number of inputs as logic state '1'.

(3) The 'EX-NOR' gate is disabled by an odd number of inputs as logic state '1'.

(4) The (2) and (3) characteristics holds vary good of EX-NOR gates with large number of inputs.

(5) Thus, the 'EX-NOR' gate can be usd for parity checker. If the output of 'EX-NOR' gate is in logic state '1', that is is said to be even parity, while if the output is in logic '0' state, the parity of function is odd. It is shown schematically in table with three inputs.

Table 10 : Truth table for 3-input 'EX-NOR' gate.

Inputs			Output	Parity
A	B	C	Y=A . B . C	
0	0	0	1	EVEN
0	0	1	0	ODD
0	1	0	0	ODD
0	1	1	1	EVEN
1	0	0	0	ODD
1	0	1	1	EVEN
1	1	0	1	EVEN
1	1	1	0	ODD

Note : It should be keep in mind that 'EX-NOR' of two variable is the inverted output of 'EX-OR' of those variables. But the 'EX-NOR' output of three variables in not equal to the inverted output of 'EX-OR' of those three variables. Also, the output of 'EX-NOR' of a number of variables is equal to the inverted output of the 'EX-OR' of those variables only if the number of variables is even.

For two variables $\Rightarrow A \odot B = \overline{A \oplus B}$

For three vaiables $\Rightarrow A \odot B \odot C \neq \overline{A \oplus B \oplus C}$

For four variables $\Rightarrow A \odot B \odot C \odot D = \overline{A \oplus B \oplus C \oplus D}$

7. Synthesis of logic gates by using only NAND or only NOR gates

As we know that the NAND gate and NOR gate are the universal gates. Any fundamental logic gate can be realize by using only one of the gate i.e. The NAND or the NOR gate.

In this section we will discuss that how any particular logic function can perform by using one type of universal gate.

Synthesis of logic gates by only,

(1) The NAND gate

(2) The NOR gate

(1) The NAND gate

Any fundamental of output can be obtained by using a number of NAND gates. The NOT, AND, OR and NOR ates can be performed by using only NAND gate.

(a) The NOT gate :

Since a NOT gate has single input and single output while a NAND gate base two inputs and a single output. So both the inputs of NAND gate is in the form of A.

$$Y = \overline{A.B} \qquad \because B = A$$

$$= \overline{A.A} \qquad A.A = A$$

$$= \overline{A} \qquad \because A.A = A$$

$$A \text{ AND } A = A$$

$$Y = \overline{A} \qquad .....(1)$$

Equation (1) shows the output similar to a NOT gate output. In symbolic representation,

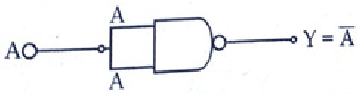


Fig. 3.18 : The NOT gate using only NAND gate

Hence, only one NAND gate is sufficient to perform a NOT gate output.

Truth Table

A	A	$A.A$	$\overline{(A.A)}$	$\overline{A}$	$Y=\overline{A}$
0	0	0	1	1	1
1	1	1	0	0	0

(b) The AND gate

Since an AND gate has two inputs with a single output. If the output of a NAND gate is the input of an other NAND gate, then the final output the AND output of the first NAND inputs.

$$Y = \overline{A.B} \text{ (Single NAND gate)}$$

$$= \overline{\overline{(A.B)}} \text{ (Double NAND gate)}$$

$$Y = A\,.\,B \text{ (AND output)} \qquad .....(2)$$

$$\because \overline{\overline{A}}\ A$$

Equation (2) Shows the output similar to a AND gate output. In symbolic representation.

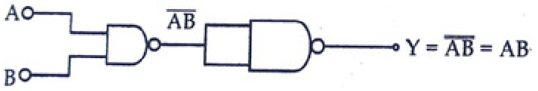


Fig. 3.19 : The AND gate using only NAND gate

Hence atleast two (Mininim two) NAND gates are required to perform a AND gate output.

A	B	AB	$\overline{AB}$	$\overline{\overline{AB}}$	$Y=A.B$
0	0	0	1	0	0
0	1	0	1	0	0
1	0	0	1	0	0
1	1	1	0	1	1

(c) The OR Gate

Similar to AND gate, an OR gate has two inputs with single output. For OR operation, first the inputs A and B are made in the inverted forms by using NAND gate and this inverted outputs are the inputs for another NAND gate. So, the output of this NAND gate gives the output similar to an OR gate. Mathematically,

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

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

$$Y = A + B \qquad .....(3)$$

Equation (3) shows the output similar to a OR gate.

In, symbolic representation,

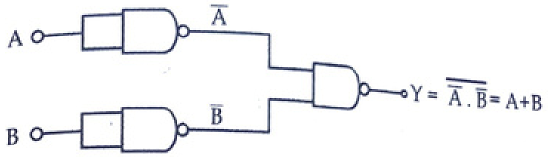


Fig. 3.20 : The OR operation by using only NAND gates

Hence, atleast three (minimum three) NAND gates are required to perform a OR gate output.

Truth table for this operation will be,

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$	$\overline{\overline{A}.\overline{B}}$	$Y=A+B$
0	0	1	1	1	0	0
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	0	1	1

(2) The NOR Gate

Similar to NAND gate, any fundamental logic output can be obtained by using a number of NOR gates. The NOT, AND and OR gate, can be performed by using only NOR gate.

(a) The NOT gate

If both the inputs of a NOR gate is in the same form i.e. A then the output of NOR gate is the complementry of the input, which is a NOT gate operation.

Mathematically, $Y = \overline{A+B}$ $\qquad A = B$

$$Y = \overline{A+A} \qquad \because A + A = A$$

$$Y = \overline{A} \quad .....(1) \qquad A \text{ OR } A = A$$

In symbolic representation,

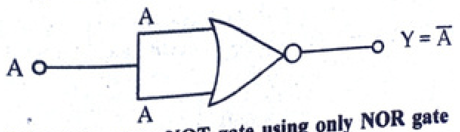


Fig. 3.21 : The NOT gate using only NOR gate

Hence, only one NOR gate is sufficient to perform a NOT gate output.

Truth table

A	A	A + A	$\overline{A+A}$	$Y = \overline{A}$
0	0	0	1	1
1	1	1	0	0

(b) The AND gate

If both the inputs of a NOR gate is taken inthis inverted form by using NOR gate. then the output of NOR gate is similar to the output of a.AND gate.

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

$$= \overline{\overline{A}} \,.\, \overline{\overline{B}}$$

$$Y = A \,.\, B \text{ (AND output)} \qquad .....(2)$$

Equation shows the output similar to a AND gate output.

In symbolic representation,

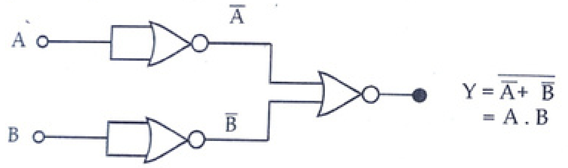


Fig. 3.22 : The AND gate using only NOR gate

Hence, atleast (mininum) three NOR gates are required to resform a AND gate output.

Truth Table

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$	$\overline{\overline{A}+\overline{B}}$	Y = A. B
0	0	1	1	1	0	0
0	1	1	0	1	0	0
1	0	0	1	1	0	0
1	1	0	0	0	1	1

(c) The OR Gate

If output of a two input NOR gate is the input of other NOR gate than the output of second NOR gate is similar to the output of OR gate, with two inputs of first NOR gate.

Mathematically,

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

$$= A + B$$

$$Y = A + B \text{ ....(3) (OR gate output)}$$

Equqtion (3) shows the output similar to the output of OR gate. In symbolic representation,

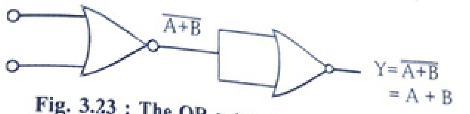


Fig. 3.23 : The OR gate using only NOR gate

Hence, atleast two NOR gates are required to perform or OR gate output.

Truth Table

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

Output of various gates with two inputs,

Input		Output of					
A	B	AND	OR	NAND	NOR	EXOR	EXNOR
0	0	0	0	1	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	1	0	0	0	1
		A . B	A + B	$\overline{A.B}$	$\overline{A+B}$	A⊕B	A⊙B

NOT

Iutput	Output
0	1
1	0
A	$\overline{A}$

8. Boolean Algebra

Boolean algebra deals with two valued system. In a logic system there are only two possibilities as logic state '1' or logic state '0'. Moreover we can say that the logic in the form of 'YES' or No. True or False ON or OFF and HIGH or LOW. However an ordinary algebra deals with multi valued system that mean any variable in such type of algebra has any value depending upon the problem. Also, it includes variety of operations i.e. Addition (+), subtraction (–), multiplication (.) and division (÷). In boolean algebra usually '+' (Plus) symbol is used for an OR operation and '.' symbol is used for AND operation. But the output of both the operation is again in the form of either logic state '1' or '0'. It matches with ordinary algebra only upto a certain extent. There is no significance of any numerical values of a variable in boolean algebra.

If $Y_1 = A$

$Y_2 = A$

Then in ordinary algebra ($y_1 + y_2$) gives the value 2A, while in boolean algebra it gives the value A due to ORed operation. In the same way ($y_1 \cdot y_2$) = A^2 in ordinary algebra, while in boolean algebra A^2 does not have any meaning the operation $y_1 \cdot y_2$ gives A. The difference on ordinary and boolean algebra is only due to the choice of A, since A rèpresent only the logical status of the variable it is either '1' or 0.

Apart from above dissimilarities, there characteristics of boolean Algebra are as follows :

(1) In boolean algebra, the variable has either '1' or '0' logic state.

(2) No existence of any numerical number, i.e. 2, 3, 4.

(3) No existence of any negative value, i.e. –2, –3, –4.

(4) No existence of any fractional value, i.e 1.5, 2.6, 3.7.

(5) No existence of substraction and division operation (–, ÷)

(6) Multiplication and Addition of variable has also logical state '0' or '1'.

(7) Addition means AND operation of variables.

(8) Multiplication means OR operation of variables.

(9) A truth table can made and varified for boolean expression.

(10) If a variable is defined by logic state '1' that means it never be in logic state '0' i.e. A = 1 or A ≠ 0.

(11) If a variable is defined by logic state '0' that means, it never be logic state '1' i.e A = 0 or A ≠ 1.

(12) A function is a perfect induction of logical expression.

Now, Is there any relationship between logic system and mathematics?

This question has no answers for the long period of several decades. It was George Boole, who tried to overcome this problem in 1854, but again since its impact has no usual application in that time of technology. It was come into existence in 1938, when shannon applied this algebra for switching circuits. Thereafter Boolean Algebra come into picture and nowadays one can easily feel how important it is. It is used to analyze and design the,

(1) Digital integrated circuits.

(2) Computer circuits

(3) Combinational circuit

(4) Multiplexer and demultiplexer

(5) Karnaugh maps

(6) Encoder and decoder

For electronic circuits, the voltage level + 5V is defined as logical state '1' while 0V is defined as logic state '0'. However, the voltage flactuate and the level 0 to 0.8 is reffers as logic state '0' while 2V to 5V is reffered as logic state '1'.

Basic operations- Since, in Boolean algebra we will consider only two types of logical state i.e. logic state '0' or logic state '1'. These logical states are similar to switching properties of a switch i.e. ON or OFF. Hence sometimes boolean algebra is also known as switching algebra. In boolean algebra, three are two types of fundamental operations. The 'AND' operation can performed by simply AND gate operation and be represented by putting the symbol '.' (dot) between two or more variables. The other one is 'OR' operation, it is performed by simply OR gate operation and be represented by putting the symbol '+' (plus) between two or more variables. The last basic operation is the 'NOT' operation, in the someway it is equivalent to a NOT gate operation. It is the complementary or inverted of a variable. **A literal is a variable or its complement.**

Some other operations that are derived from the above three basic operations are, the 'NAND' operation, the NOR operation, the 'Ex -OR' operation and the EX-NOR operation. All the basic operations can summerized in the following way :

A	B	A.B	A + B	$\overline{A.B}$	$\overline{A+B}$	$A \oplus B$	$A \odot B$
0	0	0	0	1	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	1	0	0	0	1

and the 'NOT' operation

A	$\overline{A}$
0	1
1	0

Note : The boolean alg.ora is very useful for digital systems because the electric circuits can performed by above basic operations.

Duality principle: Duality principle is very important in boolean algebra. It ensure that any theorem is proved on basic postulates of boolean algebra, than a dual theorem can be obtained by interchanging '.' (dot) with '+' (plus) sign and variable state '0' with '1'. The new theorem is also exist in boolean algebra and there is no need to prove it separately.

The interchanging of *dot* with *plus* and *variable state* with *its complement state* in known as *duality principle*.

9. Boolean Theorems :

The boolean theorems are basically some postulates that are the fundamental logic operations. So it is possible to prove every theorem by taking all possible cases. Some of the theorems are based on duality principle. Any logical expression can be minimized by using these theorems and also expressed in mathematical equation. It is easy to solve any complicated expression by using the certain rules, laws and theorems of boolean algebra.

Apart from the basic boolean operation, there are certain laws i.e.

(i) The commutative law for addition and multiplication

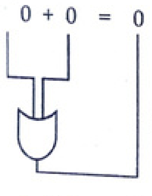
(ii) The associative law for addition and multiplication.

(iii) The distributive law.

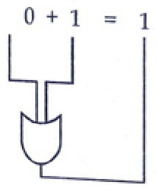
These are the same as in conventional algebra. All the theorems, laws are accept only in their original form without any proof.

(1) Boolean Addition postulates- As discussed earlier, that the boolean addition is equivalent to the 'OR' operation. The basic postulates are illustrated with their relation to the OR gate.

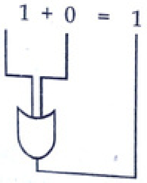
Postulate 1 :



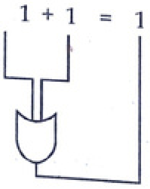
Postulate 2 :



Postulate 3 :



Postulate 4 :



A *sumterm* (addition) italic means the sum of literals (variable or its complement). A sum term is produced by an 'OR' operation without using any AND operation. i.e. A+ $\overline{B}$, $\overline{A}$ +B + $\overline{C}$, $\overline{A}$+ $\overline{B}$ +C+ $\overline{D}$ etc. A addition or sum term is equals to 'l' if one or more of the literals in the term are 'l' and it is equal to 0 only if all the literals are 0.

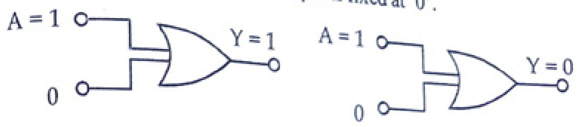
Addition rule [OR' rule]- The four OR rules are as follows,

Rule 1 : $\boxed{A+0=A}$

Any variable (or its complement) ORed with '0' is always equals to that variable (or its complement).

Also, $\boxed{\overline{A}+0=\overline{A}}$

If this input variable A is '1', then the output variable Y is '1' which is again equals to A. Now, if A is '0', the output variable Y is '0', which is also equals to A. It is illustrated in figure in which lower input is fixed at '0'.

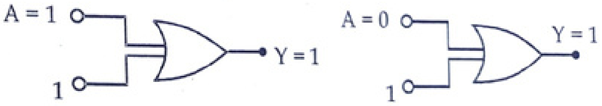


Hence, $\boxed{Y=A+0=A}$

Rule 2. $\boxed{A+1=1}$

Any variable or its complement ORed with '1' is always equals to 1. Hence the output is independent on the logical status of the variable. If the input variable is either '0' or '1' ORed with '1' always gives '1'. It is illustrated in fig. in which lower input is fixed at '1'.

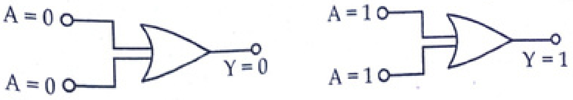
also, $\boxed{\overline{A}+1=\overline{A}}$

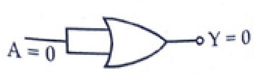


$\boxed{Y=A+1=1}$

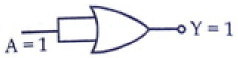
Rule 3. $\boxed{A+A=A}$

Any variable ORed with itself is always equals to that of variable. When A is '0', then 0 + 0 = 0 and when A = 1, then 1 + 1 = 1, It is illustrated in fig. in which both the inputs are the same variable.





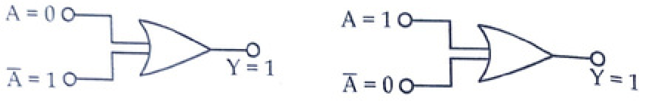
$\boxed{Y=A+A=A}$



Also, $\boxed{\overline{A}+\overline{A}=\overline{A}}$

Rule 4. $\boxed{A+\overline{A}=1}$

Any variable ORed with its complement is always equals to '1'. When A is '0', then $0+\overline{0}=0+1=1$. When A = 1, then $1+\overline{1}=1+0=1$. It is illustrated in fig. in which one input is the complement of the other.



$\boxed{Y=A+\overline{A}=1}$

Hence, the postulated of boolean addition can be summerized as :

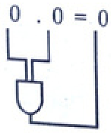
Postulate	1	:	0	+	0	=	0
Postulate	2	:	0	+	1	=	1
Postulate	3	:	1	+	0	=	1
Postulate	4	:	1	+	1	=	1

The addition rules or 'OR' rules or OR laws can be summerized as

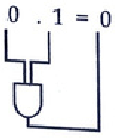
Rule 1	A	+	0	=	A
Rule 2	A	+	1	=	1
Rule 3	A	+	A	=	A
Rule 4 :	A	+	$\overline{A}$	=	1

2. Boolean multiplication postulates- As discussed earlier that the boolean multiplication is equivalent to the 'AND' operation. The basic postulates are illustrated with their relation to the AND gate.

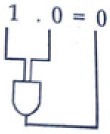
Postulate 5 :



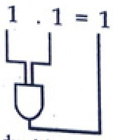
Postulate 6 :



Postulate 7 :



Postulate 8 :



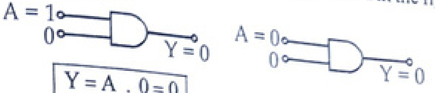
A product term means the product of literals. A product term can be produced by an AND operation without using any OR operation i.e. $A\overline{B}$, $\overline{A}B\overline{C}$, $A\overline{B}C\overline{D}$ etc. A product term is equal to '1' only when all the literals in that term is equal to '1' and it is equal to '0', when any of one literals is '0'.

Multiplication rule ['AND' rules]-

The four AND rules or AND laws as follows :

Rule 5 : $\boxed{A\ .\ 0=0}$

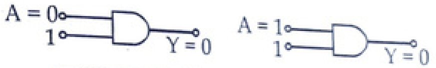
Any variable (or its complement) ANDed with '0' is always equals to '0'. If one of the input of an AND gate is '0' then the output is '0'. In such a case the output is independent on the logical status of the variable. It is illustrated in the figure in which the lower input is fixed at '0'.



$\boxed{Y=A\ .\ 0=0}$

Rule 6: $\boxed{A\ .\ 1=A}$

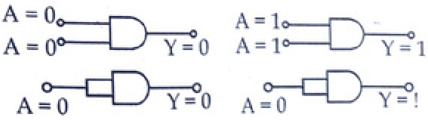
Any variable ANDed with '1' is always equals to that of variable. If A is '0', the output of AND gate is '0'. If A = 0, then the output of AND gate is '1'. In this case both the inputs are in logical state '1'. It is illustrated in figure in which lower input is fixed at '1'.



$\boxed{Y=A\ .\ 1=A}$

Rule 7 : $\boxed{A\ .\ A=A}$

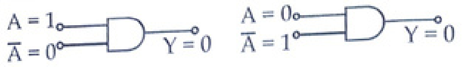
Any variable ANDed with itself is always equal to that of variable. When A is 0, then 0 . 0 = 0 and when A is 1, then 1 . 1 = 1. It is illustrated in fig in which both the inputs are same.



$\boxed{Y=A\ .\ A=A}$

Rule 8 : $\boxed{A\ .\ \overline{A}=0}$

Any variable ANDed with its complement is always equals to '0'. In this case either A or $\overline{A}$ will always be '0'. If '0' is the input of an AND gate, its output will also '0'. It is illustrated in figure in which both the inputs are complementary to each other.



$\boxed{Y = A.\overline{A} = 0}$

Hence, the postulates of boolean multiplication can be summeriged as,

Postulate 5 : 0 . 0 = 0

Postulate 6 : 0 . 1 = 0

Postulate 7 : 1 . 0 = 0

Postulate 8 : 1 . 1 = 1

The multiplication rules or AND rules can be summered as,

Rule 5 : A . 0 = 0

Rule 6 : A . 1 = A

Rule 7 : A . A = A

Rule 8 : A . $\overline{A}$ = 0

3. Complementation laws or rule- The complement of a variable means invert of that variable. It means that it changes from logical state '0' to logical state '1' and vice-versa. The following five rules are known as complementation.

Rules or laws for complementations :

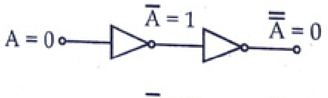
Rule 1 : $\overline{0}$ = 1

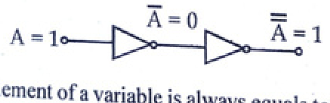
Rule 2 : $\overline{1}$ = 0

Rule 3 : $\overline{A}$ = 1 ; if A = 0

Rule 4 : $\overline{A}$ = 0 ; if A = 1

Rule 5 : $\overline{\overline{A}}$ = A





The double complement of a variable is always equals to that of variable. If we start with a variable A, then first complement gives output $\overline{A}$, now this $\overline{A}$ inverts of finally after second complement we get A, which is the original state. This is illustrated in above figure.

10. Laws of Boolean Algebra

So for, we have discussed the laws of boolean algebra. In this section we will describe the commutative laws, associative laws and distributive laws in detail.

(i) Commutative laws

(a) The commutative laws of addition for two variables can be written as,

Law 1 : $\boxed{A + B = B + A}$

This law states that the order in which the variables are ORed makes no difference. In other words we can say A OR B is the same as B OR A. Since, addition and the OR operation are the same and applicable to logical circuits.

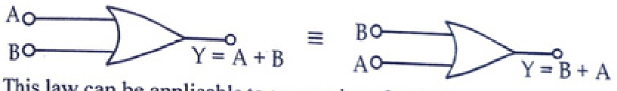
The truth table for this law is as follows,

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

≡

B	A	B+A
0	0	0
0	1	1
1	0	1
1	1	1

and the logic diagram will be,



This law can be applicable to any number of variables,

$A + B + C = B + C + A$

$= C + A + B = B + A + C$

$= A + C + B = C + B + A$

(b) The commulative law of multiplication for two variable can be written as,

Law 2 : $\boxed{A \,.\, B = B.A}$

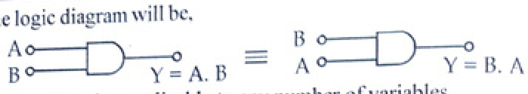
This law states that the order in which the variables are ANDed makes no difference. In order we can say that A AND B is the same as B AND A. Since, the multiplication and the AND operation are the same and also applicable to logical circuits. The truth table for this law is as follows,

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

≡

B	A	B.A
0	0	0
0	1	0
1	0	0
1	1	1

and the logic diagram will be,



This law can be also applicable to any number of variables.

A . B . C = B . C . A

= C . A . B = B . A . C

= A . C . B = C . B . A

(2) Associative laws

(a) The associative law of addition for three variables is written as follows,

Laws 1 : $\boxed{A+(B+C)=(A+B)+C}$

This law allows the grouping of variables. The law states that when ORing more than two variables, the result is the same and it is independent of the grouping of the variables. In other words, A ORed with B OR C is the same as A OR B ORed with C.

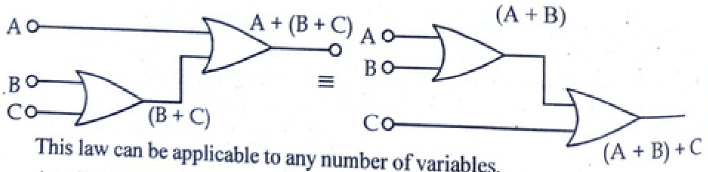
The truth table for this law is as follows :-

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

≡

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

and the logic diagram will be,



This law can be applicable to any number of variables,

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

= A + (B + C) + D

(b) The associative law of multiplication- This is written as follows, (three variables),

Law 2 : $\boxed{A.(B\ .C)=(A\ .\ B)\ .C}$

This law states that is makes no difference in what order the variables are grouped when ANDing more than two variables. In other words A ANDed with B AND C is the same as A AND B ANDed with C.

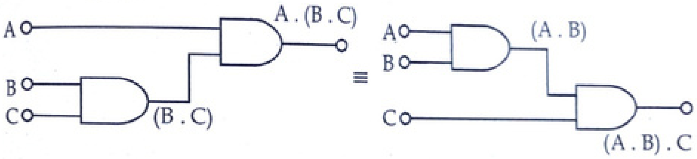
The truth table for this law is as follows :-

A	B	C	(B.C)	A.(B.C)
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

≡

A	B	C	(A . B)	(A . B) . C
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	1	0
1	1	1	1	1

and the logic diagram of the will be,



The law can also be application to any number of variables.

A. (B . C . D) = (A . B . C). D

= (A . B) . (C . D) = A . (B . C) . D

3. Distributive law- The distributive law for three variable is written as follows,

Law 1 : $\boxed{A.(B+C)=A.B+A.C}$

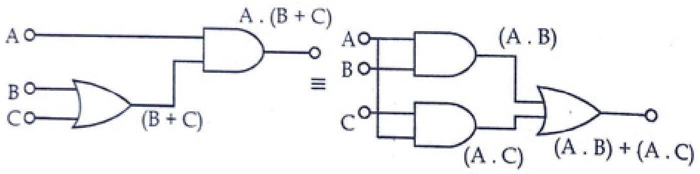
This law states that ORing of two or more variables and then ANDing the result with another single variable is equivalent to ANDing the single variable with each of the two or more variables and then ORing the products. It is the process of factoring in which the common variable A is factored out of the different product terms. It means reverse process of law 1 is also true, i.e.,

$$\boxed{A.B+A.C=A.(B+C)}$$

The truth table for law 1 is as follows,

A	B	C	(B + C)	A.(B + C)	A.B	(A.C)	(A.B) + (AC)
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

The logical diagram of law 1 will be,



Y = A . (B + C) = (A . B) + (A . C)

This law can be applicable for different combinations of variables.

A.B. (C + D) = A.B.C + A.B.D

A.B (C.D + E.F) = (A.B.C.D) + (A.B.E.F)

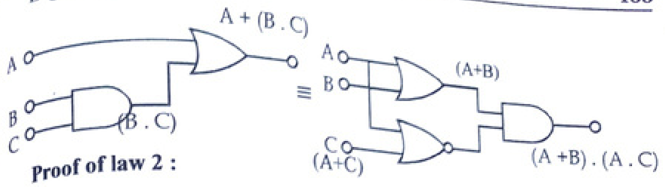
Law 2 : $\boxed{A + BC = (A + B).(A + C)}$

This law states that ANDing of two or more variables and then ORing the result with single (other) variable is equivalent to ORing that single variable with all of the variable and then ANDing the sums. Truth Table for above is as follows.

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

proved (law 2)

Logic diagram



Proof of law 2 :

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

$$\begin{aligned} R.H.S = (A + B).(A + C) &= A.A + A.C + B.A + B.C \\ &= A + A.C + A.B + B.C && \because A.A = A \\ &= A(1 + C) + A.B + B.C && \because 1 + C = 1 \\ &= A.1 + A.B + B.C && \because A.1 = A \\ &= A + A.B + B.C \\ &= A(1 + B) + B.C && \because 1 + B = 1 \\ &= A.1 + B.C && \because A.1 = 1 \\ &= A + B.C \\ &= L.H.S. \end{aligned}$$

$\boxed{RHS = LHS}$ hence proved

Law 3 : $\boxed{A + AB = A}$

This can be proved by using distribution law and rule 2.

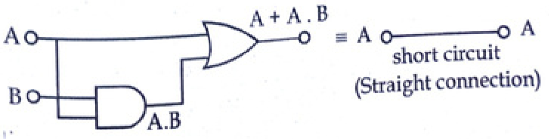
Proof :

$$\begin{aligned} A + AB &= A.(1 + B) && \because 1 + B = 1 \\ &= A.1 && A.1 = A \\ &= A \end{aligned}$$

Truth table for law 3 :

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

proved (law 3)



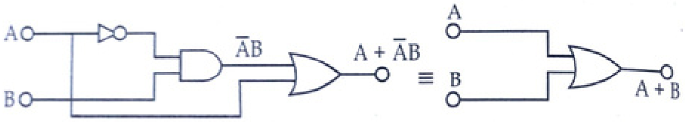
Law 4 : $\boxed{A + \overline{A}B = A + B}$

This law that ORing of a variable with the AND of its complement and other variable is equals to ORing of the two variables. The truth table for law 4 is as follows :

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

proved (law 4)

Logic diagram of law 4 will be,



Proof of law 4. $A + \overline{A}B = A + B$

$$\begin{aligned}
\text{LHS } A + \overline{A}B &= (A + AB) + \overline{A}B && \because A + AB = A \\
&= (A.A + AB) + \overline{A}B && \because A.A = A \\
&= A.A + AB + A\overline{A} + \overline{A}B && \because A\overline{A} = 0 \\
&= AA + A\overline{A} + AB + \overline{A}B && \because \text{commutative law} \\
&= A(A + \overline{A}) + B(A + \overline{A}) && A + \overline{A} = 1 \\
&= A.1 + B.1 \qquad A.1 = 1 \\
&= A + B \\
&= \text{R. H. S.}
\end{aligned}$$

$\boxed{\text{L.H.S.} = \text{R.H.S}}$ hence proved

11. Some other important boolean theorems

Apart from the above laws or theorem some other important boolean laws are also applicable to solve the logical expression and they are used directly without any proof.

(1) Absorption laws

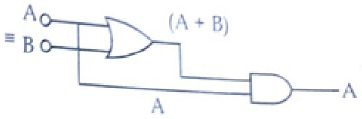
Law 1 : $A.(A + B) = A$

This law states that ANDing of a variable with the OR of that variable with another variable is equal to that of variable.

Truth table for law 1.

A	B	(A + B)	A.(A + B)
0	0	0	0
0	1	1	0
1	0	1	1
1	1	1	1

proved (law 1)



$$\begin{aligned}
\textbf{Proof. } \text{LHS } A.(A + B) &= A.A + A.B && \text{distribution law} \\
&= A + A.B && \because A.A = A \\
&= A(1 + B) && \because 1 + B = 1 \\
&= A.1 && \because A.1 = A \\
&= A \\
&= \text{RHS}
\end{aligned}$$

$\boxed{\text{L.H.S.} = \text{R.H.S}}$ hence proved

Therefore, for any variable or any term law 1, will be,

$\boxed{A.(A + X) = A}$

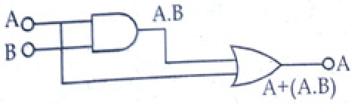
Law 2 : $\boxed{A + (A.B) = A}$

This states that ORing of a variable with the AND of that variable with another variable is equals to that variable.

Truth table for law 2.

A	B	A.B	A + (A.B)
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

proved (law 2)



$$\begin{aligned}
\textbf{Proof. } \text{L.H.S. } A + A.B &= A.(1 + B) && \because 1 + B = 1 \\
&= A.1
\end{aligned}$$

$= A \qquad \because A.1 = A$

$= RHS$

$\boxed{L.H.S. = R.H.S}$ hence proved.

(2) Transposition theorem-

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

Proof. R.H.S. $= (A + C).(\overline{A} + B)$ $\qquad \because$ distribution law

$= A.\overline{A} + AB + C\overline{A} + CB \qquad \because A\overline{A} = 0$

$= 0 + \overline{A}C + AB + BC \qquad \because$ Commutation law

$= \overline{A}C + AB + BC\,(A + \overline{A}) \qquad \because A + \overline{A} = 1$

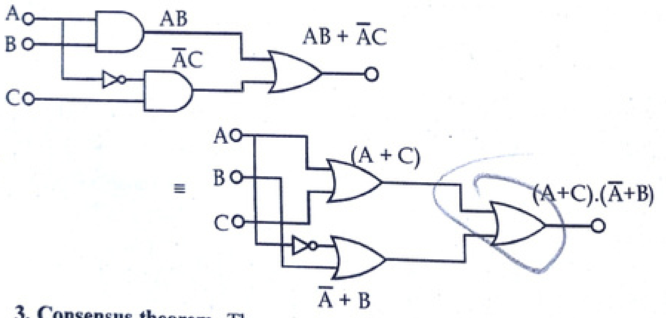
$= AB + ABC + \overline{A}BC + \overline{A}C$

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

$= AB + \overline{A}C$

$= L.H.S$

$\boxed{R.H.S. = L.H.S}$ hence proved.



3. Consensus theorem- The consensus or included factor theorems are also very useful to minimize the logical expression. The following two theorems are known as consensus theorems.

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

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

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

$= AB + \overline{A}C + BC\,(A + \overline{A}) \qquad \because A + \overline{A} = 1$

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

$= AB + ABC + \overline{A}C + \overline{A}CB \qquad \because A.1 = A$

$\because$ commutative and distributive law

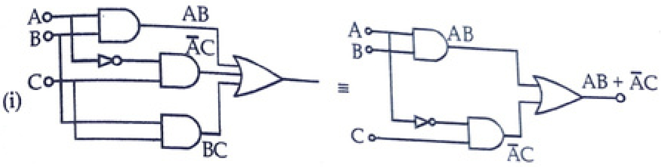
$= AB\,(1 + C) + \overline{A}C\,(1 + B) \qquad \because 1 + A = 1$

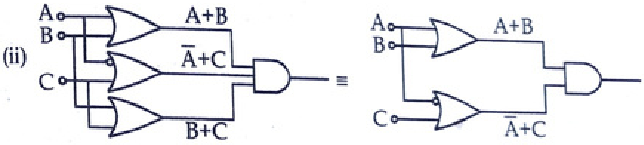
$= AB + \overline{A}C$

$= RHS$

$\boxed{L.H.S. = R.H.S}$ hence proved

Logic diagram of (i) and (ii) are





A	B	C	$\overline{A}$	AB	$\overline{A}$C	BC	(AB+$\overline{A}$C	X=(A+B)	Y=$\overline{A}$+C	Z=B+C	AB+$\overline{A}$C+BC	XY	XYZ
0	0	0	1	0	0	0	0	0	1	0	0	0	0
0	0	1	1	0	1	0	1	0	1	1	1	0	0
0	1	0	1	0	0	0	0	1	1	1	0	1	1
0	1	1	1	0	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	1	1	1	0	1	1
1	1	0	0	1	0	0	1	1	0	1	1	0	0
1	1	1	0	1	0	1	1	1	1	1	1	1	1

(Theorem 1) proved

(Theorem 2) proved

The above two theorems can be applicable to any number of variables, as follows ;

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

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

12. DeMorgan's Theorems

Demorgan was a mathematician and he proposed two important theorems for boolean algebra. Basically, these theorems provide mathematical varification of the equivalency of the NAND and bubbled OR gates, and the equivalency of the NOR and bubbled AND gates.

Hence, De-Morgans gave two very important theorems.

Theorem 1 : $\boxed{\overline{AB} = \overline{A} + \overline{B}}$

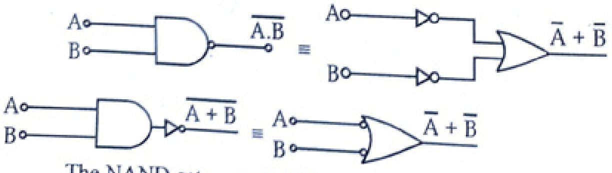
The theorem 1 status that the complement of two or more variables ANDed is equivalent to the OR of the complements of the individual variables. The other words, the complement of a product of variables is equal to the sum of the complements of the variables.

Truth table for theorem 1 will be

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

proved (De Morgan theorem)

Logical expression will be,

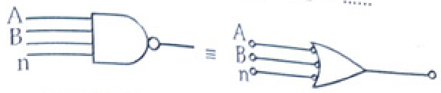


The NAND gate = Bubbled OR gate

This theorem can be applicable for any number of variables.

As $\boxed{\overline{A.B.C.D.......} = \overline{A} + \overline{B} + \overline{C} + \overline{D} + ...}$

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



Theorem 2 : $\boxed{\overline{A+B} = \overline{A}.\overline{B}}$

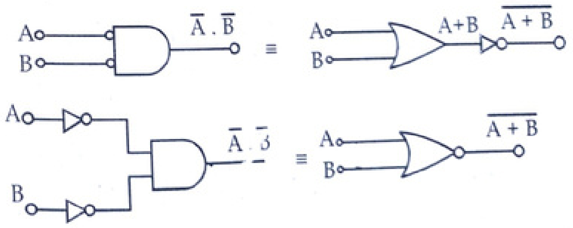
The theorem 2 states that the complement of two or more variables ORed is equivalent to the AND of the complements of the individual variables. In other words, the complement of a sum of variables is equal to the product of the complements of the variables.

Truth table for theorem 2 will be :

A	B	A+B	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

proved (De Morgan theorem 2)

Logical expression of theorem 2 will be :

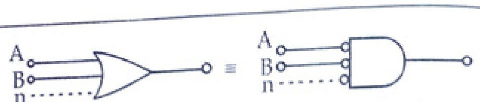


Bubbled AND gate ≡ The NOR gate

The theorem 2 can be also applicable for any number of variables.

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

Also, $\overline{AB+CDE+F+....} = \overline{AB}.\overline{CDE}.\overline{F}$



The De-morgan's theorems provide to the transformation from a POS (product of sums) to a SOP (sum of products) form. These are describe in next chapter. Such a process of transformation is knows as DEMORGANIZATION of a given function. It is also based on the duality principle. The following process or steps are taken into account for demorganigation of a function.

Step 1 : First of all identify the terms which are to be demorganized.

Step 2 : Invert or complement the whole function.

Step 3 : consider the each term as a single variable and change ANDs to ORs and vice-versa.

Step 4 : Now invert the all variables.

Example : Demorganize the function $\overline{(A+B+C).D}$

Solution :

Step : 1 Let $A+B+C=X$ and $D=Y$ so $\overline{XY}=\overline{X}+\overline{Y}$ $\because$ $\overline{A.B}=\overline{A}+\overline{B}$

Step : 2 $\overline{(A+B+C)D}$

Step : 3 $\overline{A}.\overline{B}.\overline{C}+\overline{D}$

Step : 4 $\overline{A}.\overline{B}.\overline{C}+\overline{D}$

Hence the logical expression $\overline{A+B+C).D}$ is equivalent to $\overline{A}.\overline{B}.\overline{C}+\overline{D}$.

Table 11 : Some Standard Results of Boolean Algebra :

S. No.	Standard Result	
1.	$A.0=0$	
2.	$A.1=A$	**(Raj. 2006)**
3.	$A+0=A$	**(Raj. 2006)**
4.	$A+1=1$	
5.	$AA=A$	
6.	$A+A=A$	
7.	$\overline{\overline{A}}=A$	
8.	$A\overline{A}=0$	
9.	$A+\overline{A}=1$	
10.	$A.(B+C)=AB+AC$	
11.	$A+BC=(A+B).(A+C)$	
12.	$A+AB=A$	**[Raj. 2004]**
13.	$A.(A+B)=A$	
14.	$A+\overline{A}B=A+B$	**[Raj. 2007]**
15.	$A.(\overline{A}+B)=AB$	**[Raj. 2005]**
16.	$AB+A\overline{B}=A$	**[Raj. 2005, 06]**
17.	$(A+B).(A+\overline{B})=A$	**[Raj. 2004, 07]**
18.	$AB+\overline{A}C=(A+C).(\overline{A}+B)$	
19.	$(A+B).(\overline{A}+C)=AC+\overline{A}B$	**[Raj. 2004]**
20.	$AB+\overline{A}C+BC=AB+\overline{A}C$	
21.	$(A+B).(\overline{A}+C).(B+C)=(A+B).(\overline{A}+C)$	**[Raj. 2004]**
22.	$\overline{A.B.C.D....}=\overline{A}+\overline{B}+\overline{C}+\overline{D}+....$	
23.	$\overline{A+B+C+D+....}=\overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\overline{D}$	
24.	$A+\overline{A}.B+A.\overline{B}=A+B$	
25.	$A.B+\overline{A}.B+\overline{A}.\overline{B}=\overline{A}+B$	
26.	$\overline{A}BC+A\overline{B}C+AB\overline{C}+ABC$ $=AB+BC+CA$	
27.	$(\overline{A}+B).(A+B)=B$	
28.	$\overline{A}B\overline{C}+A\overline{B}\,\overline{C}+AB\overline{C}+\overline{A}\,\overline{B}\,\overline{C}$ $=\overline{C}$	**[Raj. 2005]**
29.	$\overline{\overline{A}B+C\overline{D}}=(A+\overline{B}).(\overline{C}+D)$	
30.	$AB+\overline{AC}+A\overline{B}C\,(AB+C)=1$	**[Raj. 2004]**

13. Standard Forms for Logical Expressions

In previous section, we have discussed different types of logical expression with the combination of literal and their output. The AND, OR and NOT are the basic logical expression, while NAND and NOR are the universal building blocks to perform any type of logical expression.

The Boolean functions can be expressed in their standard form. In standard form, the function may contain any number of literals (one or more). Hence, any boolean function can be expressed in the following two forms,

(1) Sum-of-products form or SOP form.

(2) Product-of-sums form or POS form.

This type of standardization makes the evaluation, simply and implementable of boolean expression is a systematic and easier way.

Sum-of-products (SOP) form:-

The sum-of-products (SOP) form is a boolean expression that contain various AND terms of one or more literal. The each AND terms is called **product term** of SOP form. The sum express the ORing of these product terms. In other words, an SOP form can be implemented with one OR gate and two or more AND gates. The following logical expressions are the examples of SOP forms.

(1) $A + BC + \overline{A}BC$

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

(3) $ABC + CDE + \overline{B}C\overline{D}$

(4) $AB + \overline{B}C\overline{D} + AC$

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

If two or more product terms are summed in according to boolean addition, the resulting expression is a sum-of-products.

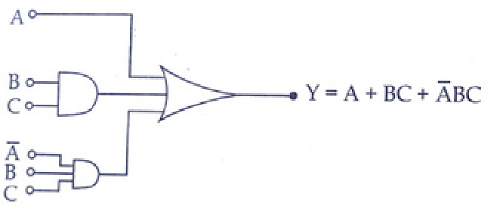


Fig. 3.24 : Implementation of the SOP expression

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

The logic diagram of a SOP expression consists of a group of AND gates that followed by a single OR gate. It is shown in fig. (3.24). For this, each product term requires an AND gate except for a term with a single literal. The logical sum is formed with an OR gate whose inputs are the outputs of the different AND gates including the single literal. Also, it is assumed that all the input variables are directly available in their complement form, so NOT gates are not included in the logic diagram. The above logic diagram is called two-level implementation. It is to be noted that for an SOP expression, a single overbar can not extend over more than one variable; however more than one variable in a term can have an overbar.

Note. An SOP expression can have the term $\overline{A}.\overline{B}.\overline{C}$, but not $\overline{ABC}$. It means $\overline{ABC}$ is not a term of SOP and it is not equal to $\overline{A}.\overline{B}.\overline{C}$ as we know from the De Morgan theorem.

Domain- In a boolean expression, the domain is the set of literals. It means that the set of variables that contained in that expression in the form of complemented or uncomplemented.

Ex. $Y = A\overline{B}\,\overline{C} + \overline{A}BC$

In above expression, the domain is the set of variable A, B and C.

Conversion into SOP form

Any boolean expression can be convert into sum-of-products (SOP) form by using boolean algebra. The basic law of boolean algebra i.e. (i) Associative law (ii) Absorption law (iii) Distributive law are used to convert the given expression into SOP form.

Product-of-sums (POS) form-

The product-of-sums (POS) form is a boolean expression that contain various OR terms of one or more literals. The each OR term is called **sum term** of POS form. The product express, the ANDing of these sum terms. In other words, an POS form can be implemented with one AND gate and two ore more OR gates. The following logical expression are the examples of POS forms. If two or more sum terms are multiplied, the resulting expression is a product-of-sums.

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

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

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

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

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

The above expression (1) has three sum terms of one, two and three literals. The implementation of this expression is as follows :

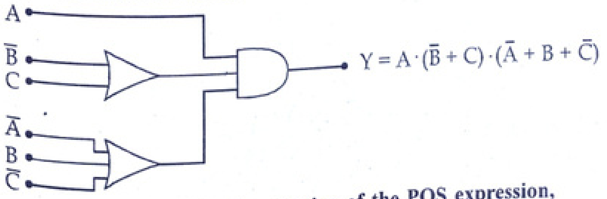

A
$\overline{B}$
C
$\overline{A}$
B
$\overline{C}$
$Y = A \cdot (\overline{B} + C) \cdot (\overline{A} + B + \overline{C})$

Fig. 3.25 : Implementation of the POS expression,

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

The logic diagram of a POS expression consists of a group of OR gates that followed by a single AND gate, as shown in fig. (3.25) For this each sum term requires an OR gate except for a term with a single literal. The logical product is formed with an AND gate, whose inputs are the outputs of the different OR gates including the single literal. Here, all the input are directly available. The description of POS form is similar to SOP form by inter changing the AND gate and OR gate with complemented varibles. In an POS expression, a single overbar cannot extend over more than one variable, however more than one variable in a term can have on overbar.

Note : An POS expression can have the term $\overline{A}+\overline{B}+\overline{C}$, *but not* $\overline{(A+B+C)}$. *It means that* $\overline{A+B+C}$ *is not a term of POS form, because it has different meaning then* $\overline{A}+\overline{B}+\overline{C}$ *as we know from De-Morgan theorem.*

Advantage of SOP and POS form-

A boolean expression may be expressed in a non-standard form neither SOP nor POS form. It requires the number of AND and OR gates to perform its logical diagram. As a result the gating level of the circuit is increased. The SOP and POS form are used to minimize the gating level of a circuit.

Ex. The expression Y = AB + C.(D + E)

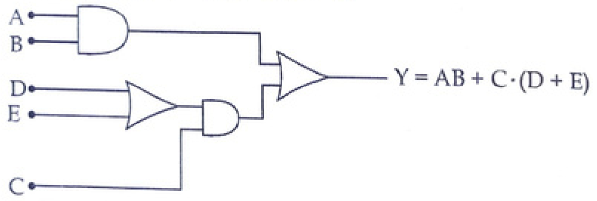


Fig. 3.26 : Circuit Realization of Y = AB + C · (D + E)

This expression requires two AND and two OR gates. It is three level implementation by boolean expression. Now,

$$Y = AB + C(D + E)$$
$$= AB + CD + CE$$

the above expression is in SOP form and it requires three AND and one OR gate. It is performed by two-level implementation.

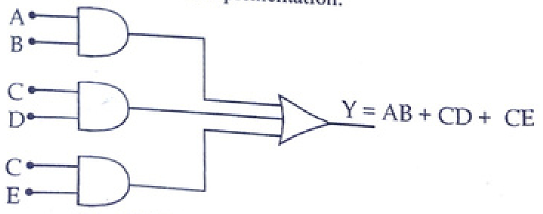


Fig.3.27: Two level implimentation of Y

14. The standard SOP and POS forms (The MINTERM and MAXTERM)

So for, we have discussed sum-of-products (SOP) expressions in which few of the **product** terms contain limited number of variables in the domain of logical expressions. Similarly, in product-of-sums (POS) expressions, the **sum** terms contain limited variables in the domain of logical expression. It means that either product term or sum term in SOP or POS from do not contain all the variables in a given expression.

A standard logical exxpression is one in which all the variables in that domain must present in each of product or sum term. The standard logical expressions are very useful to construct the truth tables and Karnaugh-map simplification. Any nonstandard logical expression can be converted to the standard form by using boolean theorems. The following two types of standard form of any logical expressions are used in K-map simplifications. The details of K-map is discussed in the next chapter.

(i) The standard SOP (SSOP) form

(ii) The Standard POS (SPOS) form

(i) The standard SOP (SSOP) form-

The standard SOP form or canonical SOP form is one in which each product term contains all the variables either in complement or uncomplement form and all the product terms summed together. In a expression $\overline{A}BC + A\overline{B}D + AB\overline{C}\,\overline{D}$ made up by the variables A, B, C and D. However, the last product term contains all the literals, but first two product terms do not contain all the literal. If we see it carefully, we see that the first product term contains three literals A, B and C. There should be forth literal D in the form of D or $\overline{D}$ for presenting the complete set of variables. Similarly in the second product form the missing variable is C, it should be there in the form of either C or $\overline{C}$, hence such type of expressions are called non-standard SOP form.

On the other hand if we see the logical expression $\overline{A}B\overline{C}D + A\overline{B}C\overline{D} + A\overline{B}CD$, each of the three product terms contain all the literals. i.e. A, B, C and D such type of expressions are called standard SOP (SSOP) or canonical SOP form.

(a) The MINTERM- In a standard or cononical sum-of-products form that contains all the variables of the function in either complemented or uncomplemented form than each product term known as a MINTERM. The minterms are also known as standard products or fundamental products. A n-variable logical expression has 2^n possible combinations of minterms. Each minterm, can be obtained by the AND operation of all the literals of that function in the domain. As we know that in a minterm, a variable is present either in complemented or uncomplemented form. For a minterm $\overline{A}.\overline{B}$ of two variable function. If Both A and B are in logical state 0 then the minterm $\bar{A}.\bar{B}$ is $\bar{0}.\bar{0}$ means 1 . 1 = 1. and of both A and B are in logical state 1 then the mineterm $\bar{A}.\bar{B}$ is $\bar{1}.\bar{1}$ means 0.0 = 0. The minterm is represented by the symbol small m. The minterms of a n-variable functions can be represented by m_0, m_1, m_2, ... $m_{(2^n-1)}$. The suffix (decimal code) indicates corresponding to the minterm combination.

The following table shows all the minterms of two and three variable logic functions.

$$\begin{aligned} Y &= \Sigma m_i \; [\Sigma = \text{logical ORing}] \\ &= \Sigma m(0,\ 1,\ 2,........) \; [m = \text{minterm}] \\ &= m_0 + m_1 + ... + m_{2^n-1} \end{aligned}$$

Table 12 : Minterms for two variables

A	B	Minerm	Minterm Notation
0	0	$\overline{A}\,\overline{B}$	m_0
0	1	$\overline{A}B$	m_1
1	0	$A\overline{B}$	m_2
1	1	AB	m_3

Table 13 : The three variable minterms table

A	B	C	Minterm	Minterm Notation
0	0	0	$\overline{A}\,\overline{B}\,\overline{C}$	m_0
0	0	1	$\overline{A}\,\overline{B}C$	m_1
0	1	0	$\overline{A}B\overline{C}$	m_2
0	1	1	$\overline{A}BC$	m_3
1	0	0	$A\overline{B}\,\overline{C}$	m_4
1	0	1	$A\overline{B}C$	m_5
1	1	0	$AB\overline{C}$	m_6
1	1	1	ABC	m_7

(b) Characteristics of Minterm :- The important characteristic of a minterm is that it has the value 1 for only one combination of n-input variables. Since there are 2^n minterms for a n-variable function, only one minterm will have the value 1, while the remaining $(2^n - 1)$ minterms will have the value 0 for any arbitrary input combination. In a two variable function Y = $A.\bar{B}$, (A = 1 and B = 0) have value 1, while the other terms, $\overline{AB}$, $\overline{A}B$ and AB have value 0. Similarly for three variable $\overline{A}BC$ (A=0, B=1, C=1) will have value 1 and all the remaining minterms will have

value 0. So, an SOP expression is equals to 1 only if one or more of the product terms in that expression is equal to 1. The sstandard SOP can be represents by Σm_i; where i is the minterm notation.

(c) Conversion of SOP form into standard SOP (SSOP) form : Each product term in an SOP form that does not contain all the variables in function can be expanded to standard form to include all variables and their complements in the function. The following procedure is taken into account to convert an SOP form into standard SOP (SSOP) form by using boolean theorem $(A + \overline{A} = 1)$.

Procedure (I)

(i) First of all write down all the product terms carefully as given in the expression.

(ii) If in any product term one or more variables are missing, expand this product term by multiplying it with the sum of each one of the missing variable and its complement $(A + \overline{A} = 1)$.

(iii) Solve the obtained function with the help of boolean theorems.

(iv) Finally, drops out the repeating term. (A + A = A)

The alternative direct method can also be applicable to obtain standard SOP form (SSOP),

Procedure (II)

(i) First of all write down all the product terms carefully as given in the expression.

(ii) Then, put Xs in product terms where variables must be inserted to form a standard product term.

(iii) Now, replace the uncomplement variable by 1s and the complemeted variable and by 0s.

(iv) Now, use all the possible combinations of Xs in product terms of 0s and 1s to get minterms.

(v) Finally, drops out the repeating terms.

(d) Four variable logical expression (Minterm):- We know that for a n-input variables, there are 2^n possible input combinations. In case of Four input variables, the sixteen $(2^4 = 16)$ possible combinations from 0000 to 1111 are available. The corresponding standard products are from $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$ to ABCD. Out of sixteen, standard products the particular is found in a quick way. It means if the input variable is 0, then the variable is in its complemented form in that standard product. It can be understood as follows,

(i) For input 0001, the first three variables are in the complemented form, while the last variable is in its uncomplement form. So the resultant product term will be $\overline{A}\,\overline{B}\,\overline{C}D$.

(ii) Similarly, for 0100, the product term will be $\overline{A}B\overline{C}\,\overline{D}$ and so on.

The minterms for 4- variable expression are numbered as m_0, m_1, m_2, ... m_{14}, m_{15}. The logical output is the ORing of different minterms. The minterms are the ANDing of all the literals present in either complemented or uncomplemented form.

$$Y(A,B,C,D) = Y = m_0 + m_1 + ...+ m_{14} + m_{15}$$

$$\Rightarrow \quad Y = \Sigma m_i\ (0, 1, 2, ... 14, 15)$$

The minterm starts form 0000 to 1111 in the different combination of literals as given in the table 14.

Table 14 : Minterms for four variables

Variable				Minterm	Minterm Notation
A	B	C	D		m_i
0	0	0	0	$\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$	m_0
0	0	0	1	$\overline{A}\,\overline{B}\,\overline{C}D$	m_1
0	0	1	0	$\overline{A}\,\overline{B}C\overline{D}$	m_2
0	0	1	1	$\overline{A}\,\overline{B}CD$	m_3
0	1	0	0	$\overline{A}B\overline{C}\,\overline{D}$	m_4
0	1	0	1	$\overline{A}B\overline{C}D$	m_5
0	1	1	0	$\overline{A}BC\overline{D}$	m_6
0	1	1	1	$\overline{A}BCD$	m_7
1	0	0	0	$A\overline{B}\,\overline{C}\,\overline{D}$	m_8
1	0	0	1	$A\overline{B}\,\overline{C}D$	m_9
1	0	1	0	$A\overline{B}C\overline{D}$	m_{10}
1	0	1	1	$A\overline{B}CD$	m_{11}
1	1	0	0	$AB\overline{C}\,\overline{D}$	m_{12}
1	1	0	1	$AB\overline{C}D$	m_{13}
1	1	1	0	$ABC\overline{D}$	m_{14}
1	1	1	1	$ABCD$	m_{15}

Each minterm is represented by m_i, where the subscript i is the decimal equivalent of the natural binary number corresponding to the minterm with uncomplemented variable taken as 1 and the complemented variable taken as 0.

(e) Logic Circuit of Minterms :- In a boolean expression, the different product terms are summed to form a standard sum-of -products (SSOP) expression. The corresponding logic circuit diagram of such minterms is accomplished by the different AND gates used for product terms and followed by a single OR gate. This is called AND-OR network.

The standard POS form (The Maxterm)

So far, we have seen POS boolean expression in which few of the sum terms do not contain all the variables. The standard POS form or cononical product-of-sums form is one in which each sum term contains all the variables either in complemented or uncomplemented form and all the sum terms multiplied together. In a four variable expression $Y = (\overline{A}+\overline{B}+\overline{C})(\overline{A}+B+\overline{D})(\overline{A}+\overline{B}+C+\overline{D})$ made up by the variables A, B, C and D. However, the last sum term contains all the variables. But first two sum terms do not contain all the variables. The first sum term should be contain the forth literal D in the form of either D or $\overline{D}$ to obtain standard POS form. Similarly the second sum term should be contain the literal C in the form of either C or $\overline{C}$. Hence the above expression is a non-standard product-of-sums (POS).

On the other hand, if we see the logical expression

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

that possess all the variables. Each sum term contain all the four variables. Such a logical expression are called standard or cononical product-of-sum (SPOS).

(a) The MAXTERM- In a domain of standard or canonical product-of-sums that contain all the variables of the function in either complemented or uncomplemented form, each sum term is known as a MAXTERM. The maxterms are also known as standard or fundamental sums. In a n-variable logical expression, there are 2^n possible combinations of maxterm hence, the number of possible combination for minterm and maxterm are same. The maxterm is represented by the symbol capital M.

$$\begin{aligned} Y &= \Pi M_i \text{ [}\Pi\text{= logical ANDing]} \\ &= M(0, 1, 2, ...) \text{ [M = Maxterm]} \\ &= M_0 \cdot M_1 \cdot ..... \cdot M_{2^n-1} \end{aligned}$$

The maxterms of a n-variable expression can be represented by M_0, M_1, M_2 .... $M_{(2^n-1)}$. The suffix indicates the decimal code corresponding to the Maxterm combination.

For a Maxterm Y = A + B, if both A and B are in logical state 0 then either A or B is in logical state 1, than the maxterm (A + B) is equal to 1. (0 + 1 = 1 + 0 = 1 + 1 = 1).

The following tables show all the possible maxterms corresponding to two and three variables logical functions.

Table-15 : Two variable Maxterms

A	B	Maxterm	Maxterm Notation
0	0	$A+B$	M_0
0	1	$A+\overline{B}$	M_1
1	0	$\overline{A}+B$	M_2
1	1	$\overline{A}+\overline{B}$	M_3

Table 16 : Three variable Maxterms

A	B	C	Maxterm	Maxterm Notation
0	0	0	$A+B+C$	M_0
0	0	1	$A+B+\overline{C}$	M_1
0	1	0	$A+\overline{B}+C$	M_2
0	1	1	$A+\overline{B}+\overline{C}$	M_3
1	0	0	$\overline{A}+B+C$	M_4
1	0	1	$\overline{A}+B+\overline{C}$	M_5
1	1	0	$\overline{A}+\overline{B}+C$	M_6
1	1	1	$\overline{A}+\overline{B}+\overline{C}$	M_7

(b) Characteristics of Maxterm :- The important characterstics of a maxterm is that it has the value 0 for only one combination of n-input variables. Only one maxterm will have the value 0, while the remaining (2^n-1) maxterms will have the value 1 for any arbitrary input combination. Let us take an example,

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

if, A = 0 and B = 1, then,

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

while the other maxterm are $(A+\overline{B})$, $(\overline{A}+\overline{B})\cdot(A+B)$ For above input (A = 0, B = 1) the other maxterm will be,

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

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

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

Also,

$$Y = A + B$$
$$= 0 + 1 = 1$$

Hence, for above combination, only Maxterm $(A+\overline{B})$ has value 0, while remaining combination have value 1. Hence it is concluded that, A product-of-sums expression is equal to 0 only when one or more of the sum terms in the boolean expression are equal to 0. The standard POS form can be represented by ΠM_i where, i is the maxterm notation.

(c) Conversion of POS form into standard POS form:- Each sum term in a POS form that does not contain all he variables in expression can be expended to standard form to include all variables in that function with their complements. The following procedures are taken to get cononical POS form by using boolean theorem $(A\cdot\overline{A} = 0)$.

Procedure I :-

(i) First of all write down all the sum terms carefully as given in the expression.

(ii) If in any sum term one or more variables are missing, expand this sum term by adding it with the product of each one of the missing variable and its complement. $(A\cdot\overline{A}=0)$.

(iii) Solve the obtained function with the help of boolean theorems.

(iv) Finally, drops out the repeating term. (AA=A).

The alternative direct method can also be applicable to obtain standard POS (SPOS) form or can be written in term of maxterm.

Procedure II :-

(i) First of all write down all the sum terms carefully as given in the expression.

(ii) Then, put X's in sum terms , whose variables must be inserted to form a standard sum term.

(iii) Now replace the completed variables by 1s and the uncomplemented variables by 0s.

(iv) Now apply all combinations of Xs in terms of 0s and 1s to obtain the maxterms.

(v) Finally drops out the repeating terms.

(d) Logic Circuits of Maxterms

In a boolean expression, the different sum terms are multiplied together to form a standard product-of sums expression. The corresponding logic circuit diagram of such maxterms is accomplished by the different OR gates used for sum terms and following by a single AND gate. This is called OR-AND network.

(e) Four variable logical expression (Maxterm)

As we know for four input variable, the total number of combinations are sixteen. These are from 0000 to 1111. The corresponding standard sums are from $(A + B + C + D)$ $(\overline{A}+\overline{B}+\overline{C}+\overline{D})$. In such a case, if the input variable is in the state, then the variable in its uncomplemented form is in that standard sum. The 1 state of the variable represents the complemented form of the fundamental sum.

For better understanding, the following examples are given.

(i) For input 0001, the first three variables are in the form of their uncomplemented literal, while the last variable is in its complemented form. So the resultant maxterm will be $(A+B+C+\overline{D})$.

(ii) Similarly, for 0100, the fundamental sum term will be $(A+\overline{B}+C+D)$.

The maxterm for four-variable expression is numbered from $M_0, M_1, \ldots\ldots M_{15}$. The logical output is the ANDing of different maxterms. The maxterms are the ORing of all the literals present in either complementary or uncomplementry form as shown in table 17.

$$Y(A.B.C.D) = M_0 . M_1 . M_2 \ldots\ldots\ldots M_{15}$$
$$= \Pi M_i$$
$$= \Pi M(0.\ 1.\ 2.........\ 15)$$

Table 17 : Maxterms for four variables

Variables				Minterm	Maxterm
A	B	C	D		Notation
0	0	0	0	$A+B+C+D$	M_0
0	0	0	1	$A+B+C+\overline{D}$	M_1
0	0	1	0	$A+B+\overline{C}+D$	M_2
0	0	1	1	$A+B+\overline{C}+\overline{D}$	M_3
0	1	0	0	$A+\overline{B}+C+D$	M_4
0	1	0	1	$A+\overline{B}+C+\overline{D}$	M_5
0	1	1	0	$A+\overline{B}+\overline{C}+D$	M_6
0	1	1	1	$A+\overline{B}+\overline{C}+\overline{D}$	M_7
1	0	0	0	$\overline{A}+B+C+D$	M_8
1	0	0	1	$\overline{A}+B+C+\overline{D}$	M_9
1	0	1	0	$\overline{A}+B+\overline{C}+D$	M_{10}
1	0	1	1	$\overline{A}+B+\overline{C}+\overline{D}$	M_{11}
1	1	0	0	$\overline{A}+\overline{B}+C+D$	M_{12}
1	1	0	1	$\overline{A}+\overline{B}+C+\overline{D}$	M_{13}
1	1	1	0	$\overline{A}+\overline{B}+\overline{C}+D$	M_{14}
1	1	1	1	$\overline{A}+\overline{B}+\overline{C}+\overline{D}$	M_{15}

Each minterm is represented by M_i, where the subscript i is the decimal equivalent of the natural binary number corresponding to the maxterm with complemented variable taken as 1 and the uncomplemented variable taken as 0.

15. Interrelation between Minterms end Maxterms.

Since, the total number of minterms and maxterm for a n-input function are (2^n) the same. The corresponding decimal number are 0 to (2^n-1). In a three variable function where $2^3 = 8$ be the total number of minterms and maxterms. The 0 to 7 are the corresponding decimal number. Let a expression

$$Y(A,B,C) = m_0 + m_2 + m_4 + m_6$$
$$Y = \Sigma m(0, 2, 4, 6)$$
$$Y = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C}$$

the 0, 2, 4 and 6 are the minterms. For this expression, the 1, 3, 5 and 7 are the maxterms. So its maxterm representation will be,

$$Y = M_1 . M_3 . M_5 - M_7$$
$$= \Pi M(1, 3, 5, 7)$$

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

Hence, if a logical function is expressed in terms of minterm/ maxterm, its maxterm/minterm representation can be obtained by using the following complementry characteristic. This can be well understood in the following 4-input function.

If, $Y(A, B, C, D) = m_0 + m_3 + m_6 + m_7 + m_{10} + m_9 + m_{12} + m_{14}$

$= \Sigma m\,(0, 3, 6, 7, 10, 12, 14)$

Then, $Y(A, B, C, D) = M_1 . M_2 . M_4 . M_5 . M_8 . M_9 . M_{11} . M_{13} . M_{15}$

$= \Pi M(1, 2, 4, 5, 8, 9, 11, 13, 15)$

The other examples are

(i) $Y(A, B) = m_0 + m_1 + m_3$

$= \Sigma m\,(0, 1, 3)$- minterms

then, $Y(A, B) = M_2$

$= \Pi M(2)$- maxterm

(ii) if, $Y(A, B, C) = m_0 + m_2 + m_3 + m_7$

$= \Sigma m(0, 2, 3, 7)$

then, $Y(A, B, C) = M_1 . M_4 . M_5 . M_6$

$= \Pi M(1, 4, 5, 6)$

16. Karnaugh Map of K-map

In the previous section we have discussed about the standard form of (SOP or POS) any n-variable logical expressions. Also, these standard forms are known as minterm and maxterm respectively. In this section we shall discussed about the minimization of an boolean expression. The karnaugh map or K-map is the most popular method for minimization. This is an graphical method in which each minterm or maxterm has a specific cell. Again, the K-map is basically graphical representation of the boolean function (expression) and its minimization is based on the boolean theorems as discussed in the previous section. The K-map is very quick and simple method upto the five/six variables slogical function bus of the number of variable in a logical expression is more than five/six then it become complicated. The complication arises just by the perception of human efficiency. To overcome this difficulty, another method is used to simplify. the logical expression, when the number of variable is more than five/six.

The K-map can be designed for both the standard SOP (Minterm) and standard POS (maxterm) logical expression. As we know that there are 2^n possible combinations for an n-input variable expression. Usually, the K-map has 2^n cells, corresponding to their possible combinations. Hence a specific cell location is fixed for each of minterm or maxterm. The K-map can aslo designed by using truth table. K-map for two, three and four variables have four, nine and sixteen cells respectively. The simplification or minimization is based on the basic boolean rule **AB + AB' = A** in case of minterm representation. The following processes are to be taken in account to minimize the logical (n variable) expression by using K-map.

(A) Mapping of SOP or POS expressions

(B) Minimization of SOP or POS expressions.

Now we shall discuss the K-map with different variables.

17. K-map with two, three & four variable

Four two input variable A and B, there are four possible combinations. Each of these combination is in standard SOP form an hence called a minterm or in case of standard POS form it is called maxterm. The presence of a minterm in a expression shows the output of the logic is assumed as logic 1, while absence shows the logic 0. In addition to this, the presence of a maxterm shows the output of logic as logic 0, while absence shows the logic 1. There are four, nine and sixteen cells corresponding to two, three and four variables K-map.

(1) Mapping of standard SOP and POS expressions :-

The K-map of two, three and four variable is designed in a specific way. In this variables have been marked as A, B, C, and D, and the binary number formed by these variable are taken as AB, ABC, and ABCD for two, three and four variables respectively. The decimal code corresponding to the particular combination of variables is shown in following K-maps. In the following K-maps, the variables of all possible combination is marked in which the first bit corresponds to the first variable and the second bit corresponds to second variable and so on.

B \ A	0	1
0	0	2
1	1	3

≡

B \ A	$\bar{A}$	A
$\bar{B}$	0	2
B	1	3

Fig. 3.28 : K-map for two-variables

C \ AB	00	01	11	10
0	0	2	6	4
1	1	3	7	5

≡

B \ A	$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
$\bar{C}$	0	2	6	4
C	1	3	7	5

Fig. 3.29 : K-map for three variables

CD \ AB	00	01	11	10
0	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

≡

CD \ AB	$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
$\bar{C}\bar{D}$	0	4	12	8
$\bar{C}D$	1	5	13	9
CD	3	7	15	11
$C\bar{D}$	2	6	14	10

Fig. 3.30 : K-map for four variables

For more understanding about K-map, it is required to design a K-map in term of miniterms or maxterms. The following diagram shows the K-map in terms of standard sum-of-products form or, minterms and standard product of sums form or maxterms.

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

(a)

C \ AB	00	01	11	10
0	$\bar{A}\bar{B}\bar{C}$	$\bar{A}B\bar{C}$	$AB\bar{C}$	$A\bar{B}\bar{C}$
1	$\bar{A}\bar{B}C$	$\bar{A}BC$	ABC	$A\bar{B}C$

(b)

CD \ AB	00	01	11	10
00	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}B\bar{C}\bar{D}$	$AB\bar{C}\bar{D}$	$A\bar{B}\bar{C}\bar{D}$
01	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}B\bar{C}D$	$AB\bar{C}D$	$A\bar{B}\bar{C}D$
11	$\bar{A}\bar{B}CD$	$\bar{A}BCD$	ABCD	$A\bar{B}CD$
10	$\bar{A}\bar{B}C\bar{D}$	$\bar{A}BC\bar{D}$	$ABC\bar{D}$	$A\bar{B}C\bar{D}$

(c)

Fig. 3.31 : K-map in terms of minterm for (a) Two variable (b) Three variables and (c) Four variables.

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

(a)

C \ AB	00	01	11	10
0	$A+B+C$	$A+\bar{B}+C$	$\bar{A}+\bar{B}+C$	$\bar{A}+B+C$
1	$A+B+\bar{C}$	$A+\bar{B}+\bar{C}$	$\bar{A}+\bar{B}+\bar{C}$	$\bar{A}+B+\bar{C}$

(b)

CD \ AB	00	01	11	10
00	$A+B+C+D$	$A+\bar{B}+C+D$	$\bar{A}+\bar{B}+C+D$	$\bar{A}+B+C+D$
01	$A+B+C+\bar{D}$	$A+\bar{B}+C+\bar{D}$	$\bar{A}+\bar{B}+C+\bar{D}$	$\bar{A}+B+C+\bar{D}$
11	$A+B+\bar{C}+\bar{D}$	$A+\bar{B}+\bar{C}+\bar{D}$	$\bar{A}+\bar{B}+\bar{C}+\bar{D}$	$\bar{A}+B+\bar{C}+\bar{D}$
10	$A+B+\bar{C}+D$	$A+\bar{B}+\bar{C}+D$	$\bar{A}+\bar{B}+\bar{C}+D$	$\bar{A}+B+\bar{C}+D$

Fig. 3.32 : K-map in terms of maxterms for (a) two variables (b) three variables (c) four variables

(a) Mapping of two variable SOP and POS expression

There are four squares or cell in a two variable K-map. Each of cell has a single or unique minterm. In a cell the '1' indicates that the corresponding minterm is included in the output expression, while '0' indicates that corresponding minterm does not appear in the expression for out put. Let us take an example as to map the expression $Y = A\bar{B} + \bar{A}.B$. The minterm presented in the given expression are m_2 and m_1. So it can be written as

$$Y = m_1 + m_2 = \Sigma m\,(1, 2) = 01 + 10$$

Hence, the K-map corresponds to the expression is,

B \ A	0	1
0	0	1
1	1	0

For mapping POS expression on the K-map, the 0s are placed in the cell corresponding to maxterm which are available in the expression. The 1s are placed in the cell corresponding to the maxterm which are not available in the expression. Since each term of canonical produce of sums expression is known as a maxterm. For two variable, there are four possible maxterms i.e. M_0, M_1, M_2, and M_3 and their expressions are $A + B$, $A+\bar{B}$, $\bar{A}+B$ and $\bar{A}+\bar{B}$ respectively. Let us take an example as to map the expression $Y = (A+B).\,(\bar{A}+B)\cdot(A+\bar{B})$. In this expression, the M_0, and M_1 and M_2 are present, so,

$$Y = M_0 \,.\, M_1 \,.\, M_2$$
$$= \Pi M\,(0, 1, 2)$$

Hence, the K-map corresponds to this expression is,

B \ A	0	1
0	0	0
1	0	1

(b) Mapping of three variable SOP and POS expression

A three variable K-map has eight cells and each cell represent the minterm (SOP) or maxterm (POS). The eight possible combinations corresponds to minterm, m_0, m_1, m_2, m_3, m_4, m_5, m_6 and m_7 are $\bar{A}\,\bar{B}\,\bar{C}$, $\bar{A}\,\bar{B}C$, $\bar{A}B\bar{C}$, $\bar{A}\,B\,C$, $A\bar{B}C$, $A\bar{B}\,\bar{C}$, $A\,B\bar{C}$ and $A\,B\,C$ respectively. In case of POS form, the eight possible combinations corresponds to maxterm, M_0, M_1, M_2, M_3, M_4, M_5, M_6 and M_7 are $(A + B + C)$, $(A + B +\bar{C})$, $(A + \bar{B} + C)$, $(A+\bar{B} +\bar{C})$, $(\bar{A} + B + C)$, $(\bar{A}+B+\bar{C})$, $(\bar{A}+\bar{B}+C)$ and $(\bar{A}+\bar{B}+\bar{C})$ respectively. The marking of minterm and maxterm is similar to those of two variable K-map. (0s marked for unavailable minterm and 1s marked for available minterm, while 0s marked for available maxterm and 1s marked for unavailable maxterm.) The decimal number of the cell for maxterm and minterm representation is the same, Now, let us take an example of each case.

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

The available minterms are $\bar{A}B\bar{C} = 0\,1\,0 = m_2$; $\bar{A}\,\bar{B}C = 0\,0\,1 = m_1$, $\bar{A}\,\bar{B}\,\bar{C} = 0\,0\,0 = m_0$, $A\,B\,\bar{C} = 1\,1\,0 = m_6$ and $A\,B\,C = 111 = m_7$. Hence, the K-map will be,

$$Y = m_0 + m_1 + m_6 + m_7$$
$$= \Sigma\,m\,(0, 1, 6, 7)$$

C \ AB	00	01	11	10
0	1	1	1	0
1	1	0	1	0

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

$$= M_6.\,M_5.\,M_2.\,M_0$$
$$= \Pi M\,(0, 2, 5, 6)$$

So, the K-map (POS)

C \ AB	00	01	11	10
0	0	0	0	1
1	1	1	1	0

(c) Mapping of four variable SOP and POS expression

A four variable K-map has sixteen cells and each cell of map represents either maxterm (POS) or minterm (SOP). The binary number of rows and columns are the gray code. The conditions of A and B along the column are marked by the binary number along the top of the map. Similarly, the conditions of C and D along the row are marked by the binary number along the left side of the map.

The sixteen possible combinations corresponds to minterms, m_0, m_1, m_2, ... m_{15} are $\bar{A}\,\bar{B}\,\bar{C}\,\bar{D}$, $\bar{A}\,\bar{B}\,\bar{C}\,D$, ... A B C D respectively. For POS form the maxterm M_0, M_1, ... M_{15} are (A + B + C + D), (A+B+C+ $\bar{D}$)...... ($\bar{A}+\bar{B}+\bar{C}+\bar{D}$) respectively. Now, if we want to map the following expression

$$Y = \Sigma m\,(0, 1, 3, 5, 8, 11, 14)$$

then, the K-map will be,

CD \ AB	00	01	11	10
00	0 1	4 0	12 0	8 1
01	1 1	5 1	13 0	9 0
11	3 1	7 0	15 0	11 1
10	2 0	6 0	14 1	10 0

For, Y = ΠM (2, 6, 7, 9, 11, 13, 14, 15), the K-map will be,

CD \ AB	00	01	11	10
00	0 1	4 1	12 1	8 1
01	1 1	5 1	13 0	3 0
11	3 1	7 0	15 0	11 0
10	2 0	6 0	14 0	10 1

(2) Minimization of the expression.

After mapping the K-map, it is required to minimize the expression.

(a) (i) Two variable SOP

In order to minimize a given boolean expression by using Karnaugh map, one should have to look carefully for adjacent (neighboured) cells having 1s. It means that the two or more adjacent minterms, when combine together gives the elimination of one or more variables. Two adjacent cells are said to be combined to each other when their minterms differ by single variable. Let us take an example in case of two variable K-map the minterms m_0 and m_2 i.e. $\bar{A}\ \bar{B}$ and $A\,\bar{B}$ differ by single variable A($\bar{A}$ in m_0 & A in m_2). Since $\bar{B}$ is common in both the m_0 and m_2, hence the minterms m_0 & m_2 may be combined to form a pair that eliminates the variable A as given below :

$$\begin{aligned} Y &= m_0 + m_2 \\ &= \bar{A}\,\bar{B} + A\,\bar{B} \\ &= (\bar{A} + A).\bar{B} \\ &= 1\,.\,\bar{B} = \bar{B} \qquad \therefore \text{ (Elimination of variable A)} \end{aligned}$$

B \ A	0	1
0	0 $\bar{A}\bar{B}$	2 $A\bar{B}$
1	1 $\bar{A}B$	3 AB

Further, the following groups are possible to form a pair and eliminate a variable,

(i) The minterm m_0 and m_1 can be combined together to form a pair and yield to $= m_0 + m_1 = \bar{A}\,\bar{B} + \bar{A}\,B = \bar{A} = Y_1$ [Fig. 3.33 (a)]

(ii) The minterm m_0 and m_2 can be combined together to form a pair and yield to $(m_0, m_2) = m_0 + m_2 = A\,\bar{B} + A\,\bar{B} = \bar{B} = Y_2$ [Fig. 3.33 (b)]

(iii) The minterm m_1 and m_3 can be combined together to form a pair and yield to $(m_1, m_3) = m_1 + m_3 = \bar{A}B + A\,B = B = Y_3$ [Fig. 3.33 (c)]

(iv) The minterm m_2 and m_3 can be combined together to form a pair and yield to $(m_2, m_3) = m_2 + m_3 = A\,\bar{B} + A\,B = A = Y_4$ [Fig. 3.33 (d)]

Here, it is to be noted that the minterm $m_0(\bar{A}\,\bar{B})$ and the minterm m_3 (AB) does not form a pair. It is also applicable in care of minterm m_1 ($\bar{A}$B) and minterm m_2 (A$\bar{B}$). However, all the four minterms can be combined to form a Quad and yield to the output as logic 1, as give below, [Fig. 3.33 (e)]

$$\begin{aligned} Y_5 &= \Sigma m\,(0, 1, 2, 3) \\ &= m_0 + m_1 + m_2 + m_3 \\ &= \bar{A}\,\bar{B} + \bar{A}B + A\,\bar{B} + AB \end{aligned}$$

$= \bar{A}(\bar{B} + B) + A(\bar{B} + B)$ ∵ (Elimination of B)

$= \bar{A} + A$ ∵ (Elimination of A)

$= 1$

now, it is clear that the two adjacent pair can be combined to each other and form a quad that eliminates both the variables. The combination of two or more neighboured minterm to form a pair or quad that yield to eliminate the one or both the variables is called grouping of cells in K-map. The following figure shows the possibilities of grouping in the two variable K-map.

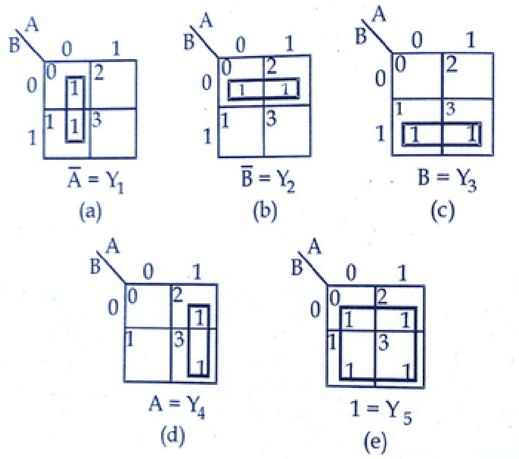


Fig. 3.33 : Grouping of minterms for two variable K-map

The Y_1, Y_2, Y_3, Y_4 and Y_5 are the reduced output depending upon the grouping of different cells.

(ii) Two variable POS :- As we see in the previous article that groups are form the pairs and quads. These are the direct evidence of minimization by using boolean theorems. Insteed to see 1s in SOP expression, here we see carefully 0s in the K-map to form a pair or quad. Here we take an example for two variable K-map, the maxterms M_0 and M_2 i.e. $(A + B)$ and $(\bar{A} + B)$ are differ by one variable A(A in M_0 and $\bar{A}$ in M_2). Since B is common in both the M_0 and M_2, hence the maxterms M_0 and M_2 may be combined to form a pair that eliminates the variable A. In the following manner,

$Y = M_0 . M_2$

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

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

$= 0 + B(A + \bar{A}) + B$

$= 0 + B$ ∴ (Elimination of A)

$= B$

Further, the following groups are possible to form a pair and eliminate a variable.

B\A	0	1
0	0 $A+B$	2 $\bar{A}+B$
1	1 $A+\bar{B}$	3 $\bar{A}+\bar{B}$

(i) In maxterm M_0 and M_1 can be combined together to form a pair and yields to $= M_0 . M_1 = (A + B) . (A + \bar{B}) = A = Y_1$ [Fig. 3.34 (a)]

(ii) In maxterm M_0 and M_2 can be combined together to form a pair and yields to $= (M_0 , M_2) = (A + B) . (\bar{A} + B) = B = Y_2$ [Fig. 3.34 (b)]

(iii) In maxterm M_1 and M_3 can be combined together to form a pair and yields to $= (M_1 . M_3) = (A + \bar{B}) . (\bar{A} + \bar{B}) = \bar{B} = Y_3$ [Fig. 3.34 (c)]

(iv) In maxterm M_2 and M_3 can be combined together to form a pair and yields to $= (M_2 , M_3) = (\bar{A} + B) . (\bar{A} + \bar{B}) = \bar{A} = Y_4$ [Fig. 3.34 (d)]

The maxterms M_0 and M_3 and M_1 and M_2 does not form a pair, However all the maxterms M_0, M_1, M_2 and M_4 can be combined to form a quad and yields to the output as logic 0 as given below. [Fig. 3.34 (e)]

$Y_5 = \Pi M(0, 1, 2, 3)$

$= M_0 . M_1 . M_2 . M_3$

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

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

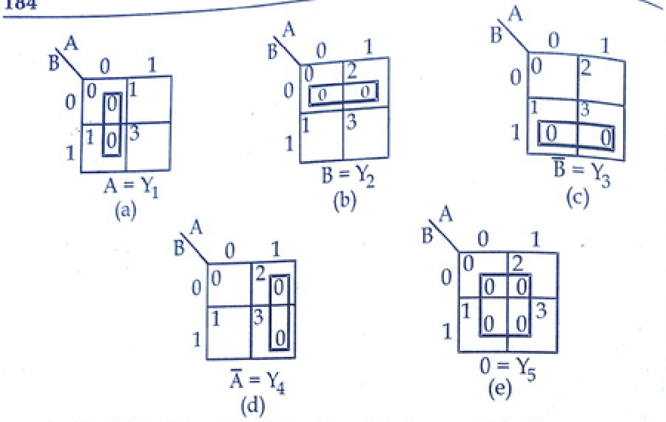
$= [A + A(B + \bar{B})] . [\bar{A} + \bar{A}(B + B)]$ ∵ (Elimination of B)

$= (A + A) . (\bar{A} + \bar{A})$ ∵ (Elimination of A)

$= A . \bar{A}$

$= 0$

Similar to SOP minimization, the pairs and quads can directly, be eliminates the variables in POS minimization. In addition to this, here one important observation is that, the combination of maxterms (in the form of pairs or quads) gives the complementary output as in case of the combination of minterms. The following figure shows the possible grouping in POS for two variable K-map.



A = Y_1 (a)

B = Y_2 (b)

$\overline{B}$ = Y_3 (c)

$\overline{A}$ = Y_4 (d)

0 = Y_5 (e)

Fig. 3.34 : Grouping of Maxterms for two variable K-map

The Y_1, Y_2, Y_3, Y_4 and Y_5 are the reduced output depending upon the grouping of different cells.

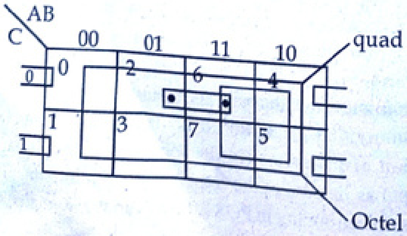
(b) Three variable SOP & POS :- In three variable K-map, the number of pairs and quads may be increased depending upon he minterm or maxterms present in a given expression. To minimize the given expression. First prepare the K-map and than see the grouping in the form of adjacent 1s (SOP) and 0s (POS). In a three variable K-map, there are eight cells and each minterm or maxterm have all the three variables. When two minterm of maxterm combine together, they will form a pair. It results to the elimination of one variable. Combination of four minterms or maxterm results the elimination of two variables. The following possibilities are in three variable K-map.

(i) Pair :- A group of two adjacent 1s (SOP) or 0s (POS)

(ii) Quad :- A group of four adjacent 1s (SOP) or 0s (POS)

(iii) Octal : - A group of eight adjacent 1s (SOP) or 0s (POS). Some authors write **octel** or **octet**.

It is to be noted that grouping of variables is either horizontal or vertical but never be diagonal.



The pair formed by the combination of minterm or maxterm (0, 4), (1, 5) (Leftmost & right most) are known as fold back pairs.

Simplification with pair, quads and octels, (SOP)

(a) Pair : Let in a expression m_2 and m_6 minterms are combine to each other than, Y = Σm(2, 6)

$$Y = m_2 + m_6$$

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

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

$$Y = B\overline{C} \qquad \because \text{(Elimination of A)}$$

The other pairs can be simplify in the same way.

(b) Quad : Let in a expression the minterms m_0, m_1, m_4 and m_5 are combine together. Since it is a open folded back quad. So,

$$Y = \Sigma m\,(0, 1, 4, 5)$$

$$Y = m_0 + m_1 + m_4 + m_5$$

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

$$= \overline{A}\,\overline{B}(\overline{C} + C) + A\overline{B}\,(\overline{C} + C) \qquad \because \text{(Elimination of A)}$$

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

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

$$= \overline{B}$$

Hence, it is clear that a quad always gives a single variable. The other quads can be simplify in the same way.

(c) Octal : In which all the minterms combine together and gives the output of expression as logic 1. It gives the elimination of all the three variables.

$$Y = \Sigma m\,(0, 1, 2, 3, 4, 5, 6, 7)$$

$$= m_0 + m_1 + m_2 + m_3 + m_4 + m_5 + m_5 + m_7$$

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

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

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

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

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

$$= 1$$

Simplification of logical expression with pairs, quads and octels (POS)

(a) Pair : Let in a expression M_2 and M_6 maxterms combine together then one variable will be omitted from the expression as follows,

$$\begin{aligned}
Y &= \Pi M\,(2.\,6)\\
&= M_2.\,M_6\\
&= (A + \overline{B} + C).\,(\overline{A} + \overline{B} + C)\\
&= A\,\overline{A} + A\overline{B} + AC + \overline{B}\,\overline{A} + \overline{B}\,\overline{B} + \overline{B}C + C\overline{A} + C\overline{B} + CC\\
&= 0 + A\overline{B} + AC + \overline{A}\,\overline{B} + \overline{B}C + \overline{A}C + \overline{B}C + C + \overline{B}\\
&= A\overline{B} + AC + \overline{A}\,\overline{B} + \overline{B}(1 + C) + \overline{A}C + C(\overline{B} + 1)\\
&= A\overline{B} + AC + (\overline{A} + 1)\overline{B} + (\overline{A} + 1)C\\
&= \overline{B}\,(1 + A) + (1 + A)\,C\\
&= \overline{B} + C
\end{aligned}$$

(b) Quad : In the same way the quad formed by the maxterms, M_0, M_1, M_4, M_5 gives the expression in one variable (Elimination of two variable)

$$\begin{aligned}
Y &= \Pi M\,(0,\,1,\,4,\,5)\\
&= (A + B + C).\,(A + B + \overline{C}).\,(\overline{A} + B + C).\,(\overline{A} + B + \overline{C})\\
&= B
\end{aligned}$$

The other pairs and quads can be solve in the same way.

(c) Octal : To form a octal, all the either maxterms combine together and give the result as logic 0, here, all the three variables elimintes.

$$\begin{aligned}
Y &= \Pi M(0,\,1,\,2,\,3,\,4,\,5,\,6,\,7)\\
&= M_0.\,M_1.\,M_2.M_3.\,M_4.\,M_5.M_6.M_7\\
&= 0
\end{aligned}$$

The output of the expression with the combination of minterms (SOP) is complimentary to the combination of maxterms (POS).

(c) Four variable SOP & POS

There are sixteen cells in a four variable K-map. The possibility of pairs, quads and octals is more. But again it depends upon the minterms or maxterms available in a given expression. The grouping of adjacent cells to form a pair, quad or octel is similar as discussed for two or three variable K-map. Since, in a four variable K-map there are four variables A, B, C, D and the minterms or maxterm have all the variables in either complemented or uncomplemented form. The pair reduced the expression by one variable, the quad reduced the expression by two variable and octel reduced the expression by three variable. In a special care, while all the minterms are present in the expression that leads to the output as logic 1 and when all the maxterm are present the output of the expression goes to logic 0.

Here, the number of folebd back pais, quads and octels is also more.

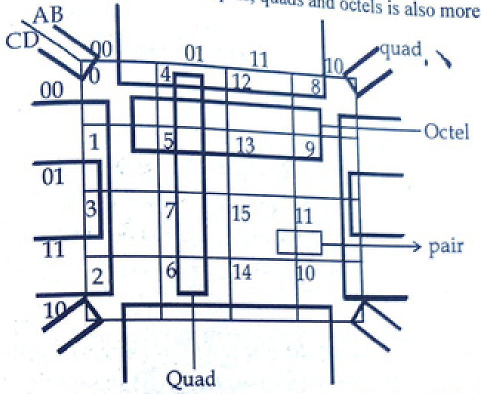


Fig. 3.35 : Various possibilities of pairs, quads of octal in a four variable K-map.

In a three variable K-map, there is only one possibility of octel (all the minterm or maxterms combine together). Instead of this, in a four variable K-map. There are eight possibilities to form a octel. They are as follows.

Table - 18 : Grouping of octals in four variable K-map

Possibility No.	Combination of Cell No.
1	(0,1,3,2,4,5,7,6)
2	(4,5,7,6,12,13,15,14)
3	(12,13,15,14,8,9,11,10)
4	(0,4,12,8,1,5,13,9)
5	(1,5,13,9,3,7,15,11)
6	(3,7,15,11,2,6,14,10)
7	(0,4,12,8,2,6,14,10)
8	(0,1,3,2,8,9,11,10)

For SOP expression we are dealing with adjacent ones, while in POS we are dealing with adjacent zeros. The possibilities of grouping is same in both the SOP and POS expressions. One can also find the number of groups for qunds or pairs in two, three or four variables K-map by keeping in mind the table 18. Also two adjacent ones (Zeros) makes a pair and two adjacent pairs make a quad and two adjacent quads make a octal.

Simplifications of four variable K-map (SOP & POS)

(a) Pair :-A large number of possibilities to form a pair in four variable K-map. One pair of minterm or maxterm reduced the expression by one variable.

Let (SOP) $Y = \Sigma m\,(0, 1)$

$= m_0 + m_1$

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

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

$= \bar{A}\,\bar{B}\,\bar{C}$ $\quad\therefore$ (Elimination of D)

For POS $Y = \pi M(0, 1)$

$= M_0 \,.\, M_1$

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

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

(b) Quad : A quad reduced the expression by two variables.

Let (SOP) $Y = \Sigma m\,(0, 1, 2, 3)$

$= m_0 + m_1 + m_2 + m_3$

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

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

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

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

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

for POS $Y = \Pi M\,(0, 1, 2, 3)$

$= M_0.\,M_1.\,M_2.M_3 = A + B$

(c) Octal : A octel reduced the expression by three variables,

Let (SOP) $Y = \Sigma m\,(0, 1, 3, 2, 4, 5\,, 7, 6)$

$= m_0 + m_1 + m_3 + m_2 + m_4 + m_5 + m_7 + m_6$

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

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

$= \bar{A}$

In POS, $Y = \Pi M\,(0, 1, 3, 2, 4, 5, 7, 6)$

$= M_0.\,M_3.\,M_2.\,M_4.\,M_5\,M_7.\,M_6$

$= A$

For all minterms (SOP)

$Y = \Sigma m\,(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,)$

$= m_0 + m_1 + \ldots.\, m_{15}$

$\boxed{Y = 1}$

For all maxterms (POS)

$Y = \Pi M\,(0, 1, 2, \ldots\ldots\ldots\ldots\, 15)$

$= M_0.\,M_1 \ldots\ldots\, M_{15}$

$\boxed{Y = 0}$

5.2.3 Mapping of K-map from Truth table

For a given truth table, the output of logical state 1 corresponds to various combination of variables is available. These combinations are also available in K-map. So for mapping a K-map by using given truth table we have to mark 1 in the corresponding cell number in case of SOP expression. The remaining cells are leave blank or marks with 0 (zero). The pairs, quad and octals are then marked by using adjacent ones in SOP K-map.

For POS K-map, the available maxterm is marked with 0s in the K-map.

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

The cell number 0 and 3 are marked with 0 and cell number 1 and 2 marked with 1.

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

A	*B*	*y*
0	0	0
0	1	1
1	0	1
1	1	0

≡

B \ A	0	1
0	^{0} 0	^{2} 1
1	^{1} 1	^{3} 0

Truth table **K-map**

$Y = (A + \bar{B}).\,(\bar{A} + B)$ (POS from)

As we know that the minterm is marked with 1s and maxterm is marked with 0s.

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

$Y(A, B, C) = m_1 + m_2 + m_4 + m_7$

Table 19 : Truth table

Cell No.	A	B	C	Y
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	0	1	1	1

C \ AB	00	01	11	10
0	^{0} 0	^{2} 1	^{6} 0	^{4} 1
1	^{1} 1	^{3} 0	^{7} 1	^{5} 0

POS form of the above expressing will be, $Y = (A+B+C).\,(A+\bar{B}+\bar{C}).(\bar{A}+B+\bar{C}).\,(\bar{A}+\bar{B}+C)$

These maxterms will be marked with zeros. For the following truth table we need to enter either zeros or ones. In a K-map, it is understood that if only ones are entered, then remaining cells are empty and they all are treated as zeros. If only zeros are entered, then remaining cells are empty and they all are treated as ones.

Table 20 : Truth table and K-map (four variable)

Cell No	Inputs				Output
	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	0
14	1	1	1	0	1
15	1	1	1	1	1

≡

CD \ AB	00	01	11	10
00	0 1	4	12 1	8
01	1	5 1	13	9 1
11	3	7 1	15 1	11
10	2	6	14 1	10

K-map

The value of output Y (either 0 or 1) for each cell number or row of truth table must be entered in the corresponding cells of the K-map.

Simplification by K-map

As we discussed earlier when one or more variables appear in both the complemented and uncomplemented form within a group, than these variables can be omitted. However following steps are to be taken in account to simplify the expression using K-map.

STEP 1 : First of all construct a K-map for a given number of variables.

STEP2 : Now, enter the ones in those cells for which the combination is available in the expression.

STEP 3 : Check the map and marked these ones which cannot combined to form a group. These are singles.

STEP 4 : Now, look upon the adjacent ones that grouped in the form of pair, quad or octel.

STEP 5 : Use only minimum number of groups, for the first see about octel, then quad and finally pair. It is possible of over lapping of groups for common ones.

STEP 6: Now omit or eliminate the repeating groups.

STEP 7 : At last write down the expression of logical sums for each group.

18. Don't Care Combinations

(Don't Care Conditions)

Certain logical expression, in which the combinations of variables are not specified and hence these unspecified combinations of variables are called don't care combinations. Apart from the completely or precisely minterms or maxterms, there can be some minterms or maxterm for which the output is of no important consequence. Therefore the don't care combinations are stands for incomplemetly specified combinations of variables. The don't care conditions or combinations are generally denoted either by d(small D) or × (cross) or ϕ (Phi). For simplification using K-map, the don't care combinations are treated as logic 1, if it is useful to reductions of the expression or otherwise it may be treated as logic 0 and left behind in a sum-of-products (SOP) K-map. Further these dont' care may be treated as logic 0, if it is useful to reduction of the expression or otherwise may be treated as logic 1 in a product of sums (POS) K-map.

Now, it is clear that the don't care combination may be taken either 1 or 0 depending upon the simplicity of the process. An SOP expression with don't care can be converted into a POS expression and vice-versa. The following example can help you about better understanding don't care combinations.

NUMERICALS

■ **Example-1. Show the EX-OR gate with three inputs.**

Sol. : The output of tw oinputs EX OR gate is,

$$Y(A, B) = A \oplus B$$

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

For three inputs the output will be,

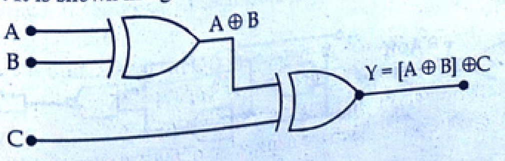
$$Y(A, B, C) = [A \oplus B] \oplus C$$

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

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

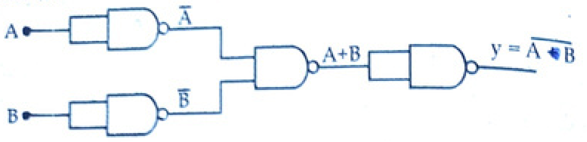
$$= A\overline{B}\overline{C} + \overline{A}B\overline{C} + ABC$$

So, two EX-OR gates are used to perform the logical diagram of a three inpout EX OR gate. It is shown in fig.

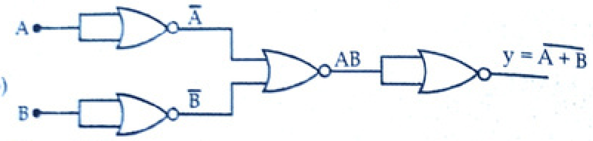


■ **Example-2. Rellise the circuit diagram of (a) NOR gate by using only NAND (b) NaND** gate by using only NOR gate.

Sol. : (a)



(b)



■ **Example-3. Determine the logical equivalent of the following expression.**

(a) $A \oplus 1$ (b) $A \oplus 0$ (c) $A \odot 0$ (d) $A \odot 0$

Sol. : (a)

$$
\begin{aligned}
A \oplus 1 &= A.\,\overline{1} + \overline{A}.1 \\
&= A.0 + \overline{A}.1 \\
&= 0 + \overline{A} \\
&= \overline{A}
\end{aligned}
$$

(b)

$$
\begin{aligned}
A \oplus 0 &= A.\,\overline{0} + \overline{A}.0 \\
&= A.1 + \overline{A}.0 \\
&= A
\end{aligned}
$$

(c)

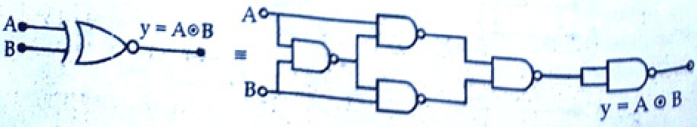
$$
\begin{aligned}
A \odot 1 &= \overline{A}.\,\overline{1} + A.1 \\
&= \overline{A}.0 + A.1 \\
&= A
\end{aligned}
$$

(d)

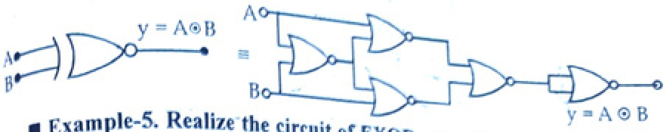
$$
\begin{aligned}
A \odot 0 &= \overline{A}.\overline{0} + A.0 \\
&= \overline{A}.1 + A.0 \\
&= \overline{A}
\end{aligned}
$$

■ **Example-4. Realize the EXNOR gate with (a) only NAND (b) only NOR**

Sol. : (a)

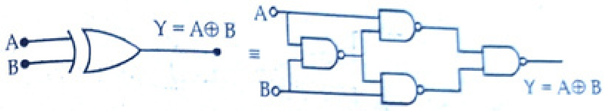


(b)

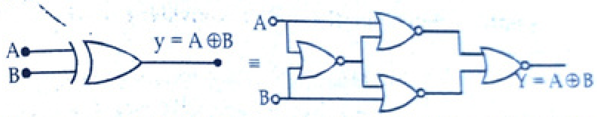


■ **Example-5. Realize the circuit of EXOR gate with (a) only NAND (b) only NOR**

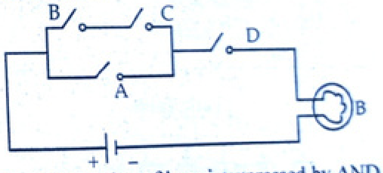
Sol. : (a)



(b)



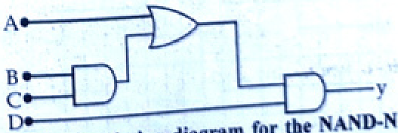
■ **Example-6. Write the expression of the following switching diagram, and realize its logic diagram.**



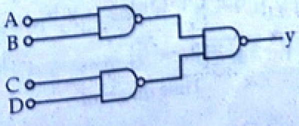
Sol. : The series combination of keys is expressed by AND gate and parallel combination by OR gate.

so

$$
\begin{aligned}
Y &= [(B.C) + A].\,D \\
&= [A + (B.C)]\,.\,D
\end{aligned}
$$



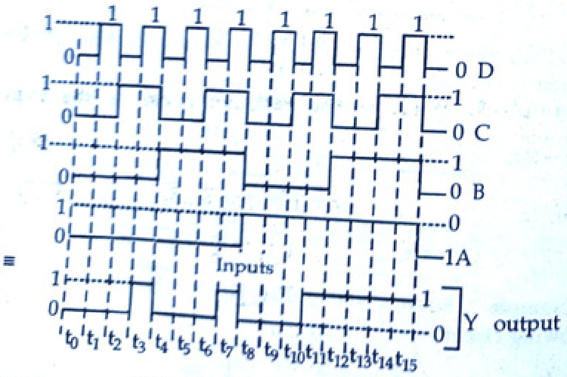
■ **Example-7. Show a timing diagram for the NAND-NAND circuit of the following circuit** **[Raj. 2006]**



Sol. : In the above NAND-NAND circuit, there are four inputs, hence, The total number of possible combinations will be $2^4 = 16$. The truth table of the circuit will be :

Inputs				Output	
A	B	C	D	Y	Time Interval
0	0	0	0	0	t_0
0	0	0	1	0	t_1
0	0	1	0	0	t_2
0	0	1	1	1	t_3
0	1	0	0	0	t_4
0	1	0	1	0	t_5
0	1	1	0	0	t_6
0	1	1	1	1	t_7
1	0	0	0	0	t_8
1	0	0	1	0	t_9
1	0	1	0	0	t_{10}
1	0	1	1	1	t_{11}
1	1	0	0	1	t_{12}
1	1	0	1	1	t_{13}
1	1	1	0	1	t_{14}
1	1	1	1	1	t_{15}

Truth Table


1
0
1
1
1
1
1
1
1
1
0 D
1
0 C
1
0 B
0
1A
Inputs
1
0
1
0
Y output
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15}

$Y = \overline{\overline{AB}.\ \overline{CD}}$

Time diagram

■ **Example-8. Reduce the following using boolean theorems.**

(i) PQQR (ii) P + Q + Q + R (iii) $P\overline{P}$ + QQ (iv) PQ + PQ$\overline{R}$ (v) PQ$\overline{Q}$R (vi) P + Q + $\overline{Q}$ + R

Sol. : (i) PQQR

$= P.Q.Q.R$

$= P.Q.R$ **Ans.** $\qquad \because A.A = A$

(ii) $P + Q + Q + R$

$= P + Q + R$ **Ans.** $\qquad \because A + A = A$

(iii) $P\overline{P} + QQ$

$= 0 + Q.Q \qquad \because A\overline{A} = 0$

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

$= Q$ **Ans.** $\qquad \because 0 + A = A$

(iv) $PQ + PQ\overline{R}$

$= PQ(1 + \overline{R}) \qquad \because 1 + A = 1$

$= PQ$ **Ans.**

(v) $PQ\overline{Q}R$

$= P.Q.\overline{Q}.R \qquad \because A.\overline{A} = 0$

$= 0$ **Ans.**

(iv) $P + Q + \overline{Q} + R$

$= P + 1 + R \qquad \because A + \overline{A} = 1$

$= P + R + 1$

$= 1 + (P + R) \qquad \because 1 + A = 1$

$= 1$ **Ans**

■ **Example-9. What is the reduced form of the expression $\overline{(A+B)}.\overline{(\overline{A}+\overline{B})}$.** **[Raj. 2004]**

Sol. : Since,

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

$= \overline{A}.\overline{B}.\overline{\overline{A}}.\overline{\overline{B}} \qquad \overline{\overline{A}} = A$

$= \overline{A}A.\overline{B}.B \qquad \overline{A}A = 0$

$= 0.0$

$= 0$

■ **Example-10. Solve expression, $Y = AB + \overline{AC} + A\overline{B}C\,(AB+C)$ using boolean theorems.** **[Raj. 2004]**

Sol. :

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

$$= AB + \overline{AC} + A\overline{B}C.\,AB + A\overline{B}C.C \quad \because B\overline{B} = 0$$

$$= AB + \overline{AC} + A\overline{B}BC + A\overline{B}C \quad \because C.C = C$$

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

$$= AB + \overline{A} + \overline{C} + A\overline{B}C \quad \because \overline{AC} = \overline{A} + \overline{C}$$

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

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

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

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

$= 1$ **Ans.**

■ **Example-11. Prove that**

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

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

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

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

Sol. : (i) $A + A\overline{B} + \overline{A}.B$

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

$$= A.\,1 + \overline{A}.B$$

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

$= A + B$ **Proved**

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

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

$$= B,1 + \overline{A}.\overline{B}$$

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

$= \overline{A} + B$ **Proved**

(iii) $AB + BC + \overline{B}C$

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

$$= AB + C.1$$

$= AB + C$ **Proved**

(iv) $(\overline{A} + B).\,(A + B)$

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

$$= 0 + \overline{A}B + AB + B \quad \because A.A = A$$

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

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

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

$$= B + 0 \quad \because A + A = A$$

$= B$ **Proved**

■ **Example-12. Realize the circuit of 'EX-OR' gate with the help of boolean theorems by using only.**

(a) NOR gate (b) NAND gate

Sol. : (a) We know that the output of 'EXOR' gate is,

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

$$= A\overline{A} + A\overline{B} + \overline{A}B + B\overline{B} \quad \because A\overline{A} = 0$$

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

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

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

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

This is the required expression and the logic circuit will be,

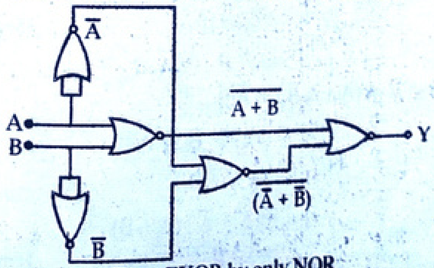


Fig. : EXOR by only NOR

NOTE : Reader can verity it by using truth table. (both a & b)

(b) Again,

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

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

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

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

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

$$=\overline{\overline{(A.\overline{AB})}\;.\;\overline{(B.\overline{AB})}}$$

Which is required expression and for this the logic circuit will be

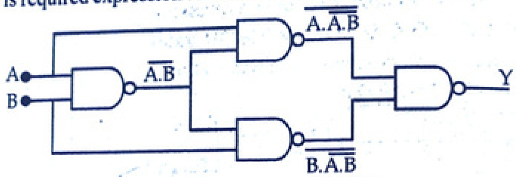


Fig. EX.OR by only NAND

■ **Example-13. Prove the following using boolean theorems,**

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

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

(iii) $\overline{A}BC+A\overline{B}C+AB\overline{C}+ABC=AB+BC+CA$

Sol. : (i) $AB+\overline{A}B+\overline{A}\,\overline{B}=(A+\overline{A})B+\overline{A}\,\overline{B}$

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

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

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

Hence Proved

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

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

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

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

$$=A+B$$

Hence Proved

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

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

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

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

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

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

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

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

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

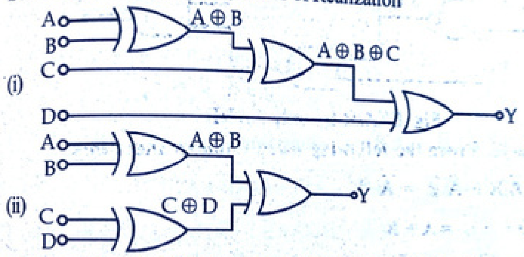
$$=AC+BC+AB$$

$$=AB+BC+CA$$

Hence Proved

■ **Example-14. Realize the logic expression $Y=A\oplus B\oplus C\oplus D$ using EX-OR gates.**

Sol. : There are two possibilities of Realization



■ **Example-15. Make a 4-input NAND gate using 2-input NAND gates.**

Sol. : Let A, B, C and D are the four inputs and Y be the output. Therefore,

$$Y=\overline{A.B.C.D}$$

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

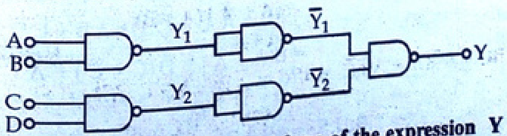
$$=(A.B).(C.D)$$

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

$$=\overline{Y_1}.\overline{Y_2} \qquad Y_1=\overline{(A.B)}\ \&\ Y_2=\overline{(C.D)}$$

$$Y=\overline{Y_1.Y_2}$$

The realization will be,



■ **Example-16. What is the reduced form of the expression $Y=(A+B).(A+\overline{B})$.** **[Raj. 2007]**

Sol. :

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

$$=AA+A\overline{B}+BA+B\overline{B}$$

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

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

$= A + A$

$= A$ **Ans.**

■ **Example-17. Find the compliment of the expression of $Y = A + BC\overline{D}$.** **[Raj. 2006]**

Sol. :

$$Y = A + BC\overline{D}$$

$$\overline{Y} = \overline{A + BC\overline{D}}$$

$$\overline{Y} = \overline{A}\,.\,\overline{(BC\overline{D})}$$

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

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

■ **Example-18. If $A\overline{B} + \overline{A}B = C$, show that $A\overline{C} + \overline{A}C = B$.**

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

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

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

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

$= AB + \overline{A}B$

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

$= B$ **Hence Proved**

■ **Example-19. Prove that $AB + \overline{C}(\overline{A} + \overline{D}) = AB + BD + \overline{B}\,\overline{D} + \overline{A}\,\overline{C}D$, If $\overline{A}B + C\overline{D} = 0$.**

Sol. :

L.H.S. $= AB + \overline{C}(\overline{A} + \overline{D}) + 0$

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

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

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

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

R.H.S. $= AB + BD + \overline{B}\,\overline{D} + \overline{A}\,\overline{C}D + 0$

$= AB + BD + \overline{B}\,\overline{D} + \overline{A}\,\overline{C}D + \overline{A}B + C\overline{D}$

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

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

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

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

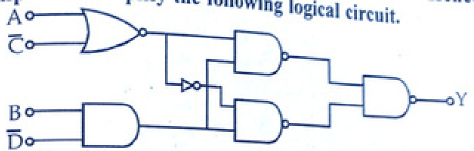
$= B + \overline{D}(1 + C) + \overline{A}\,\overline{C}D$

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

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

$=$ L.H.S. **Hence Proved**

■ **Example-20. Simplify the following logical circuit.**



Sol. : From the above circuit, the expression will be,

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

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

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

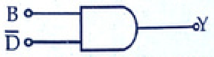
$$= \overline{\overline{B\overline{D}} + (A + \overline{C}).(\overline{A + \overline{C}})}$$

$\because (A + B).(A + C) = A + BC$

$$= \overline{\overline{B\overline{D}}}$$

$$= B\overline{D}$$

hence,



■ **Example-21. Find the reduced form of $[A\overline{B}(C + BD) + \overline{A}\,\overline{B}]C$** **[Raj. 2005]**

Sol. :

$= [A\overline{B}C + A\overline{B}BD + \overline{A}\,\overline{B}]C$

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

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

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

$= \overline{B}C$

■ **Example-22. Find the reduced form of AB + A(B + C) + B(B+C)** **[Raj. 2005]**

Sol. :

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

$= AB + AC + B + BC$

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

$= AB + AC + B$

$= AC + B(1 + A)$

$= B + AC$ **Ans.**

■ **Example 23. Expand the function $Y = \overline{A} + \overline{B}$ to minterms or standard sum-of-products and draw its logical diagram.**

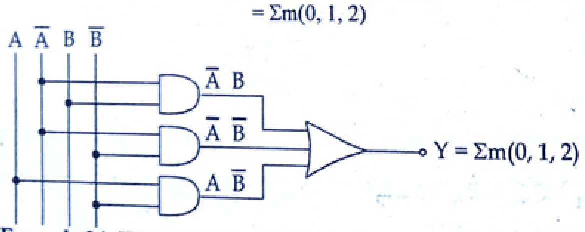
Solution. As, the given expression is a two-variable function. In the first term, $\overline{A}$ is the variable and the variable B is not present, so it is multiplied by $(B + \overline{B})$. In the second term $\overline{B}$ is present and the variable A is missing, so it is multiplying by $(A + \overline{A})$. Hence,

$$\begin{aligned} Y = \overline{A} + \overline{B} &= \overline{A}.1 + \overline{B}.1 \\ &= \overline{A}.(B + \overline{B}) + \overline{B}.(A + \overline{A}) \\ &= \overline{A}B + \overline{A}\,\overline{B} + \overline{B}A + \overline{B}\,\overline{A} \\ &= \overline{A}B + \overline{A}\,\overline{B} + A\overline{B} \\ &= 01 + 00 + 10 \\ &= m_1 + m_0 + m_2 \\ &= \Sigma m(0, 1, 2) \end{aligned}$$

Alternative method

Solution

$$\begin{aligned} Y = \overline{A} + \overline{B} &= \overline{A}.X + X.\overline{B} \\ &= 0.X + X.0 \\ &= 0.0 + 0.1 + 1.0 + 0.0 \\ &= 0.0 + 0.1 + 1.0 \\ &= \overline{A}\,\overline{B} + \overline{A}B + A\overline{B} \\ &= \Sigma m(0, 1, 2) \end{aligned}$$



■ **Example 24. Write the truth tale of the boolean expression, $Y(A,B,C,D) = ABCD + \overline{A}BCD + A\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$ and draw its logical circuit diagram.**

Solution : The above expression has 4-variables. So $2^4 = 16$ possible combinations are available. Out of sixteen only four are present in the expression, hence these four combinations have HIGH output, while remaining combinations have LOW output. The truth table for above expression becomes as follows :

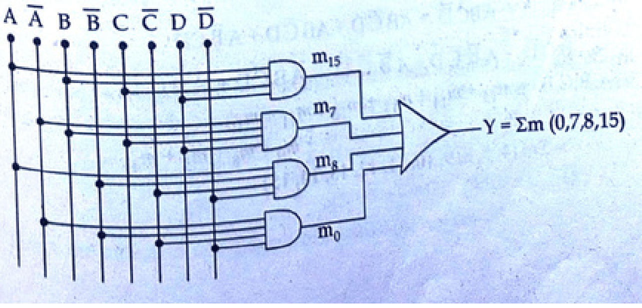
Trutable of Y

Variables				m_i	Minterm	Y (output)
A	B	C	D			
0	0	0	0	m_0	$\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$	1 ←
0	0	0	1	m_1	$\overline{A}\,\overline{B}\,\overline{C}D$	0
0	0	1	0	m_2	$\overline{A}\,\overline{B}C\overline{D}$	0
0	0	1	1	m_3	$\overline{A}\,\overline{B}CD$	0
0	1	0	0	m_4	$\overline{A}B\overline{C}\,\overline{D}$	0
0	1	0	1	m_5	$\overline{A}B\overline{C}D$	0
0	1	1	0	m_6	$\overline{A}BC\overline{D}$	0
0	1	1	1	m_7	$\overline{A}BCD$	1 ←
1	0	0	0	m_8	$A\overline{B}\,\overline{C}\,\overline{D}$	1 ←
1	0	0	1	m_9	$A\overline{B}\,\overline{C}D$	0
1	0	1	0	m_{10}	$A\overline{B}C\overline{D}$	0
1	0	1	1	m_{11}	$A\overline{B}CD$	0
1	1	0	0	m_{12}	$AB\overline{C}\,\overline{D}$	0
1	1	0	1	m_{13}	$AB\overline{C}D$	0
1	1	1	0	m_{14}	$ABC\overline{D}$	0
1	1	1	1	m_{15}	$ABCD$	1 ←

In term of minterms, $Y = m_0 + m_7 + m_8 + m_{15}$

$Y = \Sigma m(0, 7, 8, 15)$

Logic circuit ≡



■ Example 25 : Expand the following logical expression to minterms and maxterms, then draw their logical circuit diagram.

$$\mathbf{Y(A,B,C,D) = A + B\overline{C} + AB\overline{D} + ABCD}$$

Solution. Since, the above boolean expression is a four-variable SOP logical function. In the first term, the variable B, C, and D are not present. So we have to convert the expression in standard sum-of-products or minterms. For this, we have to multiply it by $(B+\overline{B}).(C+\overline{C}).(D+\overline{D})$

Hence, term I will be,

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

$$= ABCD + ABC\overline{D} + AB\overline{C}D + AB\overline{C}\,\overline{D} + A\overline{B}CD$$

$$+ A\overline{B}C\overline{D} + A\overline{B}\,\overline{C}D + A\overline{B}\,\overline{C}\,\overline{D} \qquad ....(1)$$

Similarly, in the second term. $B\overline{C}$, the variables A and D are not present, so it is multiplied by $(A+\overline{A})\cdot(D+\overline{D})$.

Hence term II. will be,

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

$$= AB\overline{C}D + AB\overline{C}\,\overline{D} + \overline{A}B\overline{C}D + \overline{A}B\overline{C}\,\overline{D} \qquad ....(2)$$

In the third term $AB\overline{D}$, the variable C is not present so it is mulitplied by $(C+\overline{C})$. Hence term III will be,

$$AB\overline{D} = AB\overline{D}\,(C+\overline{C})$$

$$= ABC\overline{D} + AB\overline{C}\,\overline{D} \qquad ....(3)$$

The forth and last term ABCD contains all the variable A, B, C and D. Hence it is itself a minterm, so

$$ABCD = ABCD \qquad ....(4)$$

Now, combined all the terms using equation (1), (2), (3) & (4) Y(A, B, C, D) = $A + B\overline{C} + AB\overline{D} + ABCD$

$$= ABCD + ABC\overline{D} + AB\overline{C}D + AB\overline{C}D + A\overline{B}CD$$

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

$$Y(A,B,C,D) = m_{15} + m_{14} + m_{13} + m_{12} + m_{11} + m_{10}$$

$$+ m_9 + m_8 + m_5 + m_4$$

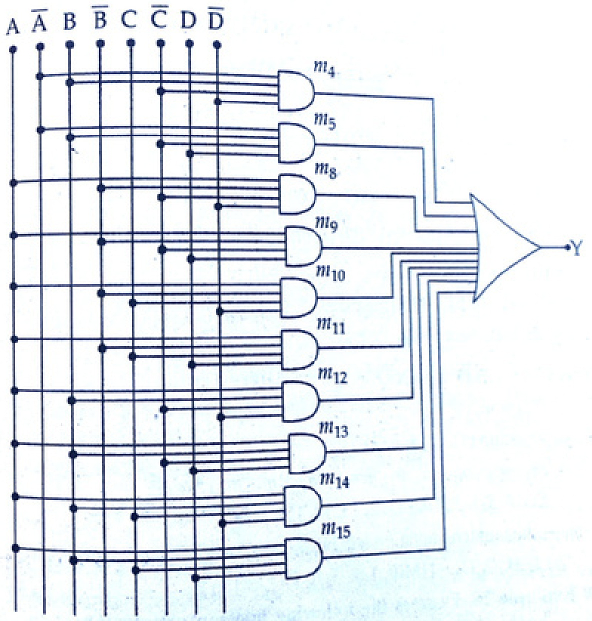
$$= \Sigma m\,(4, 5, 8, 9, 10, 11, 12, 13, 14, 15)$$

For above 4-variable standard sum-of-products expression, there are $2^4 = 16$ products term. Out of them, 4, 5, 8, 9, 10, 11, 12, 13, 14, 15 are the minterms. Therefore the corresponding maxterms will be 0, 1, 2, 3, 6, 7 and hence their representation will be,

$$Y(A, B, C, D) = M_0.\,M_1.\,M_2.\,M_3.\,M_6.\,M_7$$

$$= \Pi M(0, 1, 2, 3, 6, 7) \qquad .....(6)$$

Logical circuit diagram of equation (5) will be,



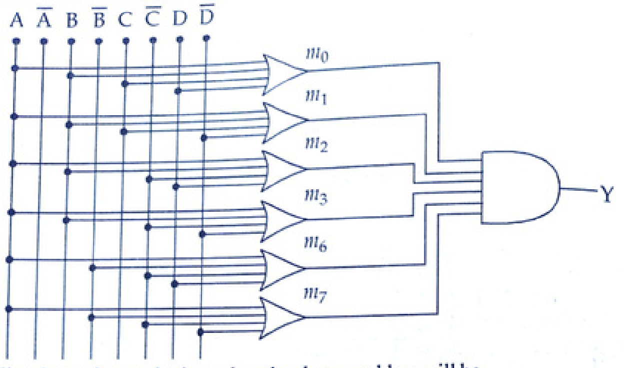
Equation (6) respresents the maxterms of the above expression, then,

$$Y(A, B, C, D) = M_0\cdot M_1\cdot M_2\cdot M_3\cdot M_6\cdot M_7$$

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

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

hence the logical circuit diagram of equation (6) will be,



The alternative method to solve, the above problem will be,

Term I :- A = A XXX = 1 XXX

$= 1000 + 1001 + 1010 + 1011 + 1100 + 1101 + 1110 + 1111$

$= m_8 + m_9 + m_{10} + m_{11} + m_{12} + m_{13} + m_{14} + m_{15}$ ....(1)

Term II :- $XB\overline{C}X = X10X$

$= 0100 + 0101 + 1100 + 1101$

$= m_4 + m_5 + m_{12} + m_{13}$ .....(2)

Term III:- $AB\overline{D} = ABX\overline{D} = 11 \times 0 = 1100 + 1110$

$= m_{13} + m_{15}$ .....(3)

From equation (1), (2) and (3)

$= m_4 + m_5 + m_8 + m_9\ m_{10} + m_{11} + m_{12} + m_{13}\ m_{14} + m_{15}$

$= \Sigma m(4, 5, 8, 9, 10, 11, 12, 13, 14, 15)$

Hence, the maxterm notation will be,

$Y(A, B, C, D) = \Pi M(0, 1, 2, 3, 6, 7)$

■ **Example-26. Express the following boolean expression,**

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

in (i) standard SOP and (ii) standard POS form. Also, show their minterm and maxterm notations.

Solution : Standard SOP form

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

Above expresion is a three variable expression and in the first term two variables B and C are missing, while in second term the variable A is missing, So, by introducing these missing variable using the boolean property, $A + \overline{A} = 1$.

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

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

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

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

$\because A + A = A$

The above equation is the standard SOP form of the given expression. Now for minterm representation,

$$Y(A,B,C) = m_7 + m_6 + m_5 + m_4 + m_1$$

$$= m_1 + m_4 + m_5 + m_6 + m_7$$

Therefore, $Y(A, B, C) = \Sigma m(1, 4, 5, 6, 7)$ **Ans.**

(ii) Standard POS form

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

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

Now the first term contains two literals, and variable C is missing, while in second term variable B is missing, so by introducing the missing variable using the property $A\overline{\boldsymbol{A}} = 0$.

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

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

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

The above equation is the standard POS form of the given expression Now, for maxterm representation,

$$Y(A,B,C) = M_2 \cdot M_3 \cdot M_0$$

$$= M_0 \cdot M_2 \cdot M_3$$

Therefore, $= Y(A,B,C) = \Pi M(0,2,3)$ **Ans.**

■ **Example-27. Express the following expression in its standard POS form :**

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

Solution : In the above three variable expression, there are three terms each term required a additional literal to form its standard POS form. These are C, A and B pectively, So ($\therefore A\overline{A} = 0$)

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

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

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

$[\because A+BC = (A+B)\cdot(A+C)]$

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

$[\because A+A = A]$

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

This is the standard POS form.

$Y(A.B.C) = M_2{\cdot}M_3{\cdot}M_0{\cdot}M_4{\cdot}M_6$
$= M_0{\cdot}M_2{\cdot}M_3{\cdot}M_4{\cdot}M_6$

$Y = \Pi M\,(0,2,3,4,6)$ **Ans.**

■ **Example-28. Given the logical expression.**

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

(i) Convert this equation in Sum-of-Products (SOP) form and in Product-of-Sums (POS) form. [Raj. 2004, 2007]

(ii) Design the circuits with only one type of gates corresponding to SOP and POS equations obtained earlier. [Raj. 2007]

(iii) Convert the above SOP and POS equations in standard SOP and standard POS forms. [Raj. 2004]

(iv) Write the above standard SOP and POS equations in the form of minterm and maxterm notations. [Raj. 2004]

(v) Simplify the above SOP and POS equation in standard SOP and standard POS forms. [Raj. 2007]

Solution : (i) (a) Sum-of-products form :

$Y(A,B,C) = (A+BC)\ .\ (B+\overline{C}\,A)$
$= A.\,(B+\overline{C}\,A) + BC(B+\overline{C}\,A)$ ← A+BC]
$= AB + A\overline{C}A + BCB + BC\overline{C}A$ (A+B).(A+C)
$= AB + A.A\overline{C} + B.BC + BC\overline{C}A$

$(\therefore A\cdot A = A;\ B\cdot B = B;\ C\overline{C} = 0)$

$Y = AB + A\overline{C} + BC + 0$

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

Answer

(b) Product-of-sums form

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

(Distribution law : $A+BC = (A+B)\,(A+C)$)

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

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

Answer

(ii) Circuit Realization using one type of gates,

(a) SOP form realization using NAND gates:

We know that.

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

Therefore, $\overline{Y} = \overline{AB + A\overline{C} + BC}$

$= \overline{AB}\cdot\overline{A\overline{C}}\cdot\overline{BC}$ (De Morgan theorem) $\therefore \overline{A+B} = \overline{A}\cdot\overline{B}$

$\boxed{Y = \overline{\overline{AB}.\ \overline{A\overline{C}}\ .\ \overline{BC}}}$ $\therefore \overline{\overline{A}} = A$

Therefore, the logical circuit will be only one type of gates, (NAND).

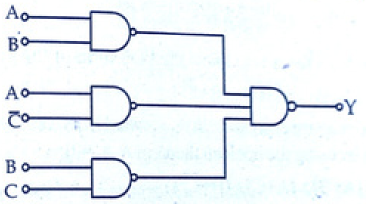


Figure . Circuit Relization of Y using only NAND gates

(b) POS form realization using NOR gates,

We know that,

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

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

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

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

Therefore, the logical circuit wi'. be only one type of gate, (NOR).

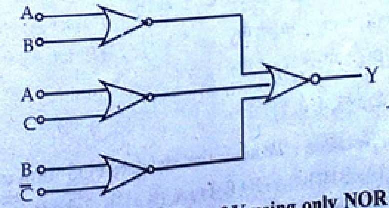


Figure : Circuit realization of Y using only NOR gates

(iii) (a) Standard SOP term : we know that the SOP form of given equation is :

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

Since, the expression is a three variable expression and above three product terms does not contain all the three literals so, to convert in standard form we have to introduce the missing literal by using the boolean theorem $A + \overline{A} = 1$.

$$Y = AB + A\overline{C} + BC$$
$$= AB.1 + A\overline{C}.1 + BC.1$$
$$= AB\,(C + \overline{C}) + A(B + \overline{B})\overline{C} + (A + \overline{A})\,BC$$
$$= ABC + AB\overline{C} + AB\overline{C} + A\overline{B}\,\overline{C} + ABC + \overline{A}\,BC$$

$$Y = ABC + AB\overline{C} + A\overline{B}\,\overline{C} + \overline{A}\,BC$$ **Answer**

(b) Standard POS form : we know the POS form of the given expression is,

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

Again, to convert above expression in its standard POS form we have to introduce the missing literal by using the boolean theorem $A\overline{A} = 0$.

$$Y = (A+B)\cdot(A+C)\cdot(B+\overline{C})$$
$$= (A+B+0)\cdot(A+C+0)\cdot(B+\overline{C}+0)$$
$$= (A+B+C\overline{C})\cdot(A+C+B\overline{B})\cdot(B+\overline{C}+A\,\overline{A})$$
$$= (A+B+C)\cdot(A+B+\overline{C})\cdot(A+C+B)\cdot(A+C+\overline{B})$$
$$(B+\overline{C}+A)\cdot(B+\overline{C}+\overline{A})$$

$\because A+BC = (A+B)\,(A+C)$

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

$\therefore$ Commutative law $\therefore$ A.A = A

$$Y = (A+B+C)\,.\,(A+B+\overline{C})\,.\,(A+\overline{B}+C)\,.\,(\overline{A}+B+\overline{C})$$ **Answer**

(iv) Minterm Notation : We know that the standard SOP form,

$$Y = ABC + AB\overline{C} + A\overline{B}\,\overline{C} + \overline{A}\,BC$$
$$= m_7 + m_6 + m_4 + m_3$$
$$= m_3 + m_4 + m_6 + m_7$$

$$Y = \Sigma m\,(3, 4, 6, 7)$$ **Answer**

(b) Maxterm Notation : We know that standard POS form of the expression,

$$Y = (A+B+C)\cdot(A+B+\overline{C})\cdot(A+\overline{B}+C)\,.\,(\overline{A}+B+\overline{C})$$
$$= M_0\cdot M_1\cdot M_2\cdot M_5$$

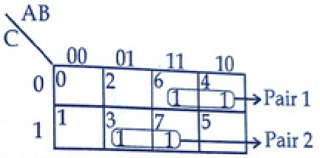
$$Y = \Pi M\,(0, 1, 2, 5)$$ **Answer**

(v) (a) Simplification of standard SOP form:

We know that,

$$Y = (A,B,C) = m_3 + m_4 + m_6 + m_7$$

So, using K-map.



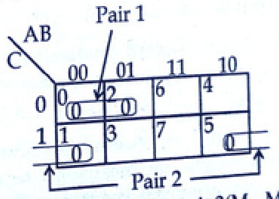
There are two pairs. Pair (m_4, m_6) eliminate the literal B, while pair 2 (m_3, m_7) eliminate the literal A. So, the simplified form will be,

$$Y = (m_3 + m_7) + (m_4 + m_6)$$
$$= (\overline{A}\,BC + ABC) + (A\overline{B}\,\overline{C} + AB\overline{C})$$
$$= (\overline{A} + A)BC + A\overline{C}\,(\overline{B} + B)$$
$$Y = BC + A\overline{C}$$ **Answer**

(b) Simplification of standard POS form : We know that,

$$Y\,(A, B, C) = M_0\cdot M_1\cdot M_2\cdot M_5$$

So, the K-map mill be,



Here, also two pairs. Pair $1(M_0, M_2)$ and pair $2(M_1, M_5)$ gives the result.

$$Y = M_0\cdot M_2\cdot M_1\cdot M_5$$
$$= (A+B+C)\cdot(A+\overline{B}+C)\cdot(A+B+\overline{C})\cdot(\overline{A}+B+\overline{C})$$
$$= (A+C+B\overline{B})\cdot(B+\overline{C}+A\,\overline{A})$$

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

$$= (A+C+0)\cdot(B+\overline{C}+0)$$ **Answer**

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

■ **Example-29. Find the reduced form of the following expression,** **[Raj. 2005]**

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

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

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

$= \overline{A}\,\overline{C}.1 + A\overline{C}.1$ $\quad\therefore A+\overline{A} = 1$

$= \overline{A}\,\overline{C} + A\overline{C}$

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

$= \overline{C}$ **Ans.**

■ **Example-30. Find the sum-of-products (SOP) form of the following expression,**

$Y(A,B,C) = (A+B).(\overline{A}+C).(B+C)$ **[Raj. 2004]**

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

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

$\because A\overline{A} = 0$

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

$\because AB = BA$

$= ACB+ACC+\overline{A}BB+\overline{A}BC+BCB+BCC$

$= ABC+AC+\overline{A}B+\overline{A}BC+BC+BC$ $\quad\because A+A=A$

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

$= BC+AC+\overline{A}B$

$Y = AC+\overline{A}B+BC$ **Ans.**

■ **Example-31. Find the standard product-of-sums form of a boolean function F=Σm(0, 1, 5, 7, 8, 9, 12) given in standard Sum of product form.** **[Raj. 2004]**

Solution : As we know that any standard SOP form can be converted into standard POS form by converting the minterms into maxterms by their complimentary decimal numbers. The given expression is a three variable function So, decimal numbers will be from 0, 1, 2 ...., 15.

$F = \Sigma m(0, 1, 5, 7, 8, 9, 12)$

Available minterms in the SSOP form are,

$F = m_0 + m_1 + m_5 + m_7 + m_8 + m_9 + m_{12}$

So the SPOS is the complementary decimal numbers, they are; 2, 3, 4, 6, 10, 11, 13, 14 and 15. Hence SPOS representation will be,

$F = M_2 \cdot M_3 \cdot M_4 \cdot M_6 \cdot M_{10} \cdot M_{11} \cdot M_{13} \cdot M_{14} \cdot M_{15}$

$F = \Pi M(2, 3, 4, 6, 10, 11, 13, 14, 15)$ **Answer.**

■ **Example-32. Find the canonical Sum-of-products (SOP) form of the following function,**

$Y(A,B,C,D) = AB+ACD$

Solution : To convert the given SOP into canonical SOP we have to introduce the missing literals. Since, the given function is a four variable expression, hence,

$Y(A,B,C,D) = AB+ACD$

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

$= (ABC+AB\overline{C})(D+\overline{D}) + ABCD + A\overline{B}CD$

$= ABCD+ABC\overline{D}+AB\overline{C}D+AB\overline{C}\,\overline{D}+ABCD+A\overline{B}CD$

$= ABCD+ABC\overline{D}+AB\overline{C}D+AB\overline{C}\,\overline{D}+A\overline{B}CD$ **Ans.**

■ **Example-33. Express the following function into minterm and Maxterm notation.**

$Y = (A,B,C,D) = D$

Solution : Since, the above function is a four variable function

Here, $\quad Y = D \Rightarrow Y = 1.1.1.D$

(a) Minterm : $= (A+\overline{A}).(B+\overline{B}).(C+\overline{C}).D$

$= (AB+A\overline{B}+\overline{A}B+\overline{A}\,\overline{B})(C+\overline{C}).D$ $\quad\because A+\overline{A}=1$

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

$= ABCD+AB\overline{C}D+A\overline{B}CD+A\overline{B}\,\overline{C}D+\overline{A}BCD+\overline{A}B\overline{C}D$
$+\overline{A}\,\overline{B}CD+\overline{A}\,\overline{B}\,\overline{C}D$

$= m_{15}+m_{13}+m_{11}+m_9+m_7+m_5+m_3+m_1$

$= \Sigma m(1, 3, 5, 7, 9, 11, 13, 15)$

(b) Maxterm Notation :

$Y = \Pi M(0, 2, 4, 6, 8, 10, 12, 14)$ **Ans.**

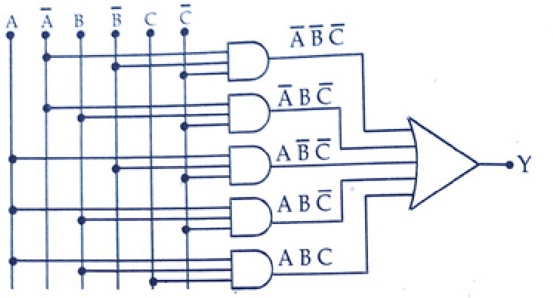
■ **Example-34. Draw a logic circuit for the following Truth table given in SSOP form.**

Inputs			Output
A	**B**	**C**	**Y**
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Solution : Since the output is in the form of SSOP. So, the boolean expression will be,

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

and the logic circuit will be.



■ **Example-35. Evaluate the following SOP expression and find its equivalent POS expression.**

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

Sol. : Since, there are three variable, So the total 2^3=8 combinations are possible. Out of them five are available is SOP expression. Hence the POS expression must contain the remaining three.

$Y = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,B\overline{C} + \overline{A}\,BC + A\overline{B}\,C + ABC$

$= 000+010+011+101+111$ (SOP form)

The equivalent POS will be,

$= 001+100+110$

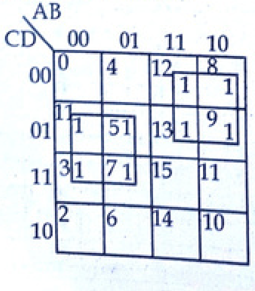
$= (A+B+\overline{C}).(\overline{A}+B+C).(\overline{A}+\overline{B}+C)$ (POS form)

■ **Example 36. Simplify the following expression by using K-map.**

$Y (A, B, C, D) = \Sigma m (1, 3, 5, 7, 8, 9, 12, 13)$

Solution : $Y (A, B, C,D) = m_1 + m_3 + m_5 + m_7 + m_8 + m_9 + m_{12} + m_{13}$

K-map for the above expression will be,



CD \ AB	00	01	11	10
00	0	4	12 1	8 1
01	1 1	5 1	13 1	9 1
11	3 1	7 1	15	11
10	2	6	14	10

For this problem, there is no octal and pairs. There are two quads formed by the cells (1, 3, 5, 7) and (12, 8, 13, 9). The first quad (1, 3, 5, 7) gives,

$= m_1 + m_3 + m_5 + m_7$

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

$= \overline{A}\,\overline{B}\,\overline{D} + \overline{A}BD$

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

$= \overline{A}D$ ....(1)

The second quad (12, 8, 13, 9) gives,

$= m_{12} + m_8 + m_{13} + m_9$

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

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

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

$= A\overline{C}$

Hence, the output will be (simplified)

$\boxed{Y = \overline{A}D + A\overline{C}}$

Since there is no further grouping is possible and no more ones are ungrouped. However quad (1, 5, 13, 9) is there but the ones of this quad are already grouped in quad first and second.

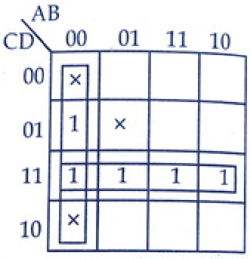
■ **Example 37. Write the logical expression for the following truth table and (a) simplify it using K-map (b) Draws the logic diagram of the result.**

Cell No.	Variables				Output
	A	B	C	D	Y
0	0	0	0	0	×
1	0	0	0	1	1
2	0	0	1	0	×
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	×
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	1

Solution : The logical expression of the above truth table will be,

$$Y(A, B, C, D) = \Sigma m\,(1, 3, 7, 11, 15) + \Sigma d\,(0, 2, 5)$$

Because the minterm m_1, m_3, m_7, m_{11} and m_{15} are specified completely, while minterms m_0, m_2 & m_{15} are an specified, so they are don't care combinations. So the K-map of the above expression will be constructed by marking in the specified minterms and the corresponding cell numbers are 1, 3, 7, 11 and 15. Also the cell number corresponding to dont care combinations as 0, 2, 5 are marked with × (cross). By assuming these crosses as 1, there are two quads. First is formed by cells 3, 7, 11 and 15 second is formed by cells 0, 1, 2 and 3. Now all the ones have been covered and hence, the reduced expression will be,



CD \ AB	00	01	11	10
00	×			
01	1	×		
11	1	1	1	1
10	×			

Quad 1

$$Y\,(A.B.C.D) = m_0 + m_1 + m_2 + m_3$$

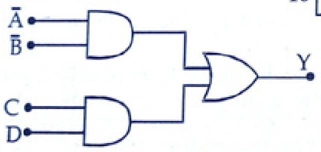
$$= \bar{A}\ \bar{B}$$

and Quad 2

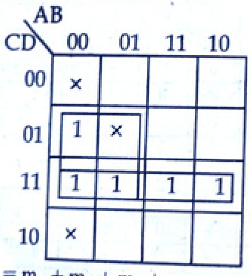
$$Y(A, B, C, D) = m_3 + m_7 + m_{11} + m_{15}$$

$$= C\,D$$

Hence, $\boxed{Y = \bar{A}\bar{B} + CD}$ ....(1)



The above problem can also be solved as,



CD \ AB	00	01	11	10
00	×			
01	1	×		
11	1	1	1	1
10	×			

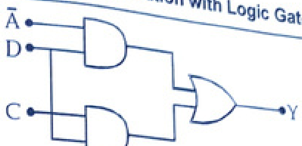
Quad 1 $Y(A, B, C, D) = m_1 + m_3 + m_5 + m_7$

$$= \bar{A}D$$

Quad 2 $Y(A, B, C, D) = m_3 + m_7 + m_{11} + m_{15}$

$$= CD$$

so, $\boxed{Y = \bar{A}\,D + CD}$ .....(2)



Both the results (1) and (2) are possible, it depends upon the selection of quads.

■ **Example 38. Plot the following logical expression.**

$$\mathbf{Y(A, B, C, D) = ABCD + A\bar{B}\bar{C}\,\bar{D} + A\bar{B}C + AB} \qquad \textbf{[Raj. 2004]}$$

on a four variable Karnaugh map and obtain its simplified expression.

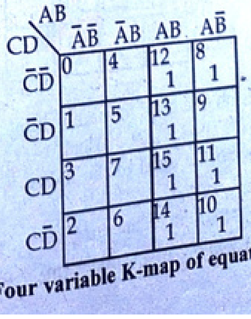
Solution : To represent a given logical expression on K-map, it is essential to represent that in either standard SOP or POS form. The above four variable expression is not in its standard form, so first of all we find its standard SOP form as follows,

$$Y(A, B, C, D) = \overset{\text{I}}{ABCD} + \overset{\text{II}}{A\bar{B}\bar{C}\,\bar{D}} + \overset{\text{III}}{A\bar{B}C} + \overset{\text{IV}}{AB} \qquad \text{....(i)}$$

The I & II product terms have all the literals, but III & IV do not have all the literals, hence,

$$\begin{aligned} Y &= ABCD + A\bar{B}\bar{C}\,\bar{D} + A\bar{B}C(D+\bar{D}) + AB(C+\bar{C})\,(D+\bar{D}) \\ &= ABCD + A\bar{B}\bar{C}\,\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D} + AB \\ &\qquad (CD + C\bar{D} + \bar{C}D + \bar{C}\,\bar{D}) \\ &= ABCD + A\bar{B}\bar{C}\,\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D} + ABCD + \\ &\qquad ABC\bar{D} + AB\bar{C}D + AB\bar{C}\,\bar{D} \\ &= m_{15} + m_8 + m_{11} + m_{10} + m_{14} + m_{13} + m_{12} \\ &= m_8 + m_{10} + m_{11} + m_{12} + m_{13} + m_{14} + m_5 \\ &= \Sigma m\,(8, 10, 11, 12, 13, 14, 15) \end{aligned} \qquad \text{....(ii)}$$

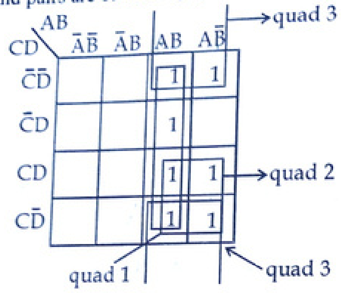
Equation (ii) is in the standard SOP from of the above expression so, its K-map representation will be,



CD \ AB	$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
$\bar{C}\bar{D}$	0	4	12 1	8 1
$\bar{C}D$	1	5	13 1	9
CD	3	7	15 1	11 1
$C\bar{D}$	2	6	14 1	10 1

Fig. : Four variable K-map of equation (ii)

To simplify the expression we have to find octels, quads pairs. Since there is no octal, three quads and pairs are covered by quads.

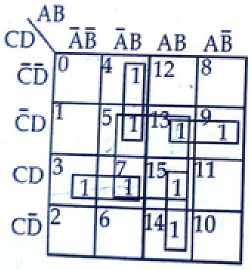


$$Y = m_8 + m_{10} + m_{11} + m_{12} + m_{13} + m_{14} + m_{15}$$
$$= (m_{12} + m_{13} + m_{14} + m_{15}) + (m_{10} + m_{11} + m_{14} + m_{15}) + (m_8 + m_{12} + m_{10} + m_{14})$$

$\boxed{Y = AB + AC + A\bar{D}}$ **Answer.**

■ **Example 39. Plot and simplify the following logical expression,**
Y (A, B, C, D) = Σm (3, 4, 5, 7, 9, 13, 14, 15)
by using K-map method.

Solution : Since, above logical expression is a four variable (SOP) expression, hence, its K-map representation will be,



For the above expression one quad formed by mintems (5, 7, 13, 15) and four pairs formed by minterms (3, 7); (4, 5), (9, 13) and (14, 15). Also all the ones of quad are covered by the pairs. So the quad of the above K-map can be left and the simplified expression will be,

$$Y = m_3 + m_4 + m_5 + m_7 + m_9 + m_{13} + m_{14} + m_{15}$$
$$Y = (m_3 + m_7) + (m_4 + m_5) + (m_9 + m_{13}) + (m_{14} + m_{15})$$

$\boxed{Y = \bar{A}CD + \bar{A}B\bar{C} + A\bar{C}D + ABC}$ **Answer.**

■ **Example 40. Plot the following logical expression on K-map,**
(i) Y(A, B, C, D) = Σm(0 , 2, 5, 7, 10, 13, 15)
(ii) Y(A, B, C, D) = πM (0, 1, 4, 7, 11, 13, 14)
(iii) Y(A, B) = Σm (0, 3)
(iv) Y(A, B) = πM (0, 2)

Solution : (i) Since the logical expression is a four variable expression and it is in standard SOP form. So all the minterms or corresponding cell are marked with 1s. Hence the K-map representation will be,

CD \ AB	$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
$\bar{C}\bar{D}$	0 1	4	12	8
$\bar{C}D$	1	5 1	13 1	9
CD	3	7 1	15 1	11
$C\bar{D}$	2 1	6	14	10 1

Y = Σm (0, 2, 5, 7, 10, 13, 15)

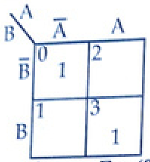
(ii) The above logical expression is also a four variable expression, but it is in the standard POS form. So all the cells of maxterms will be marked with 0s. Here the K-map representation will be,

CD \ AB	00	01	11	10
00	0 0	4 0	12	8
01	1 0	5	13 0	9
11	3	7 0	15	11 0
10	2	6	14 0	10

Y = ΠM (0, 1, 4, 7, 11, 13, 14)

(iii) The logical expression, Y(A, B) = Σm (0, 3) is a two variables expression. Also, it is in standard SOP form and the corresponding minterms will be marked with 1s.

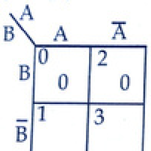
$$Y = \Sigma m\ (0, 3)$$
$$= m_0 + m_3 = \bar{A}\ \bar{B} + AB$$



The K-map for y = Σm (0, 3)

(iv) The logical expression Y(A, B) = ΠM(0, 2) is a two variable expression, and it is in standard POS form. So the corresponding maxterms will be marked with 0s.

$$Y = \Pi M(0,2) = M_0 . M_2$$
$$= (A + B).(\overline{A} + B)$$



The K-Map for y = ΠM(0, 2)

■ Example 41. Simplify the following four variable logic function using K-map.

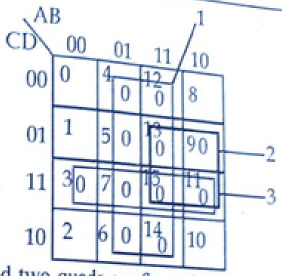
$$Y = (A,B,C,D) = (A+B+\overline{C}+\overline{D}).(\overline{A}+C+\overline{D}).(\overline{A}+B+\overline{C}+\overline{D}).$$
$$.(\overline{B}+C).(\overline{B}+\overline{C}).(A+\overline{B}).(\overline{B}+\overline{D})$$

Solution : The above expression is a four variable function, it is in POS form, but not in standard POS form. To represent this expression on a four variable K-map we draw a table first that indicates the 0s (zeros) entries of the corresponding cell. For this it is to be noted that even it a cell is involved in more than one terms a 0 (Zero) is to be entered only once.

Secondary, we shall see the possibility or grouping of zeros, to form a octel, quad and pairs.

S.No.	Sum term	Cellsmarked with 0s
1.	$A+B+\overline{C}+\overline{D}$	A = 0, B = 0, C = 1, D = 1
2.	$\overline{A}+C+\overline{D}$	A = 1, C = 0, D = 1
3.	$\overline{A}+B+\overline{C}+\overline{D}$	A = 1, B = 0, C = 1, D = 1
4.	$\overline{B}+C$	B = 1, C = 0
5.	$\overline{B}+\overline{C}$	B = 1, C = 1
6.	$A+\overline{B}$	A = 0, B = 1
7.	$\overline{B}+\overline{D}$	B = 1, D = 1

The K-map of above expression will be.

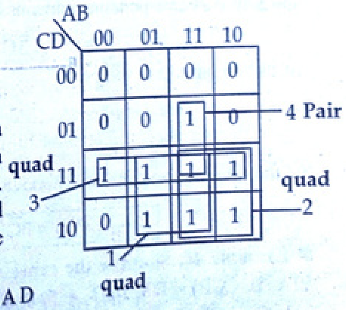


Here, one octel and two quads are formed that covered all the zeros, so the octel gives the result as $\overline{B}$ while the quad 2 gives the $(\overline{A}+\overline{D})$ and quad 3 gives the $(\overline{C}+\overline{D})$, hence. The simplified expression will be,

$$Y(A,B,C,D) = \overline{B}.(\overline{A}+\overline{D}).(\overline{C}+\overline{D})$$ **Answer**

■ Example 42. What is the simplified boolean equation for the karnaugh map of the following figure. **[Raj. 2005]**

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



Solution : The above K-map is a four variable K-map. For simplification we have to see octel, quads and pairs. Since three is no octel, three quads and one pair that covered all the ones of the K-map.

(i) Quad 1 gives the expression = A D

(ii) Quad 2 gives the expression = A C

(iii) Quad 3 gives the expression = A B

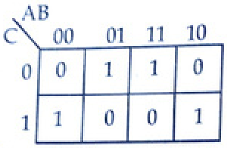
(iv) Pair 4 gives the expression = B C D

Hence, the simplified expression will be,

$$Y(A, B, C, D) = A D + A C + A B + B C D$$

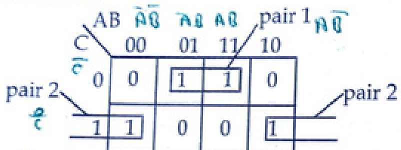
$$\boxed{Y = A(B + C + D) + BCD}$$ **Answer.**

■ **Example 43. What is the simplified boolean expression in SOP form from K-map given below.** **[Raj. 2004, 2007]**



C \ AB	00	01	11	10
0	0	1	1	0
1	1	0	0	1

Solution : The above K-map is a three variable K-map. In this expression two pairs are formed as follows :



C \ AB	00	01	11	10
0	0	1	1	0
1	1	0	0	1

The corresponding minterms for pair 1 are,

$$Y_1 = \bar{A}B\bar{C} + AB\bar{C}$$

and its minimized form will be

$$Y_1 = (\bar{A} + A)B\bar{C}$$

$$Y_1 = B\bar{C}$$

Similarly, the corresponding minterms for pair 2 are,

$$Y_2 = \bar{A}\,\bar{B}C + A\bar{B}C$$

and its minimize form will be,

$$Y_2 = (\bar{A} + A).\bar{B}C$$

$$Y_2 = \bar{B}C$$

So, the simplified expression for above K-map is,

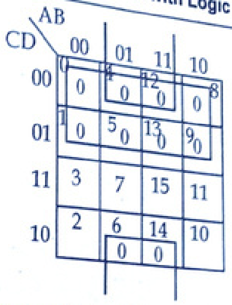
$$\boxed{Y = Y_1 + Y_2 = B\bar{C} + \bar{B}C}$$ **Answer.**

■ **Example 44. Simplify the expression Y(A, B, C, D) = ΠM (0, 1, 4, 5, 6, 8, 9, 12, 13,14) using K-map method.**

Solution : The above expression is in the standard POS form, so the maxterms and their corresponding cells numbers. are marked with 0s.

$$Y(A, B, C, D) = \pi M(0, 1, 4, 5, 6, 8, 9, 12, 13, 14)$$

The K-map will be,



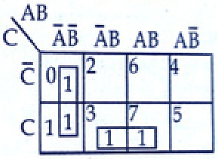
CD \ AB	00	01	11	10
00	0 0	4 0	12 0	8 0
01	1 0	5 0	13 0	9 0
11	3	7	15	11
10	2	6 0	14 0	10

There is one octal. That gives the result C and one quad that gives the result ($\bar{B}$ +D). So, the simplified expression will be,

$$\boxed{Y = C.(\bar{B} + D)}$$ **Answer**

■ **Example 45. Simplify Y(A, B, C) = Σm (0, 1, 3, 7)**

Solution : The three variable K-map will be,



C \ AB	$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
$\bar{C}$	0 1	2	6	4
C	1 1	3 1	7 1	5

The 1s are marked in the corresponding cell number since the function is in SOP form. Here two pairs are formed.

Pair 1 gives = $\bar{A}\,\bar{B}\,\bar{C} + \bar{A}\,\bar{B}C$

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

$$Y_1 = \bar{A}\,\bar{B}$$

Pair 2 gives = $\bar{A}BC + ABC$

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

$$Y_2 = BC$$

Hence, $Y = Y_1 + Y_2 = \bar{A}\,\bar{B} + BC$ **Answer**

Questions

Very Short Answer Type Questions **[Raj. 2010]**

1. Define logic gate.
2. Define fundamental gate.
3. What do you mean by AND gate. ?

4. What do you mean by OR gate ?
5. What do you mean by NOT gate ?
6. What is NAND gate ?
7. What is NOR gate ? **[Raj. 2006]**
8. Define Ex-OR gate. **[Raj. 2006]**
9. What is the significance of Ex-OR gate.
10. Define universal gate.
11. Write the name of the universal gate.
12. State Demorgan's Law.
13. Define Boolean algebra.
14. Define the SOP form.
15. What is the POS form.
16. What is minterm ? **[Raj. 2010, 2009]**
17. Define Maxterm. **[Raj. 2010, 2009]**
18. Define the standard SOP form.
19. What is K-map.
20. Write the uses of K-map.
21. What do you understand by "Don't Care" state. **[Raj. 2004]**
22. Define the standard POS form.
23. What do you mean by domain of a expression.

Long Answer Type Questions

1. Write a short note on logic functions.
2. What is Ex-NOR gate, give its significance. **[Raj. 2006]**
3. Show that NAND and NOR gates are universal gates.
4. Realize the circuit by using RTL logic for, NOT gate.
5. What is diode logic (DL), explain, it for AND and OR gate.
6. What are the TRUE & FALSE statements.
7. For the logic expression $Y = A\overline{B} + \overline{A}B$,
   (a) Which logic operation gives the output Y
   (b) Make its truth table
   (c) Realize it by using fundamental gates
   (d) Realize it by using only NAND gates.
8. Realize the logic operation for $Y = A \oplus B \oplus C$ using EX OR gate
9. Make a truth table for three input
   (a) AND gate (b) OR gate (c) NAND gate (d) NOR gate

10. Explain how, the basic or fundamental gates can be realize by using only.
    (a) NAND gate (b) NOR gate **[Raj. 2010]**
11. Is it possible to make a NOT gate with two inputs, how ?
12. Which of the following are equivalent to $A \oplus B$ and which to $A \odot B$,
    (a) $\overline{A} \oplus B$ (b) $\overline{A} \odot B$ (c) $\overline{A} \odot \overline{B}$ (d) $A \oplus \overline{B}$ (e) $A \odot \overline{B}$
13. How does OR addition differ from the ordinary addition method.
14. How does AND multiplication differ from the ordinary multiplication.
15. Write a short note on duality principle.
16. State and prove DeMorgan theorem. **[Raj. 2010]**
17. State and prove absorption theorem.
18. Give the boolean expression for the following statement a
    Y is a 1 only if A is a 1 and B is a 1 or if A is a 0 and B is a 0.
    Ans. $(AB + \overline{A}\,\overline{B})$
19. Prove the statement : A positive logic NAND operation is equivalent to a negative logic NOR operation and vice-versa.
20. Discuss the various theorems used in boolean algebra.
21. Give the logical states of variables for the following terms,
    (i) m_0 (ii) M_0 (iii) m_6 (d) M_7
    (v) m_{15} . (vi) M_{14}
22. What do your understand by literal, explain it using examples.
23. Minimize the following functions and realize using minimum number of gates.
    (i) $Y_1 = \Sigma m\,(0, 3, 5, 6, 9, 10, 12, 15)$
    (ii) $Y_2 = \Sigma m\,(0, 1, 2, 3, 11, 12, 14, 15)$
24. Express the following functions in standard form (SOP/POS).
    (i) $Y_1 = AB + A\overline{C} + C + AD + A\overline{B}C + ABC$
    (ii) $Y_2 = \overline{A}B + \overline{B}CD + AC + B\overline{C}D$
    (iii) $Y_3 = (A+B) . (B+CA)$
    (iv) $Y_4 = (B+\overline{C}) . (A+\overline{B}+C) . (\overline{A}+C)$
25. Simplify the logic expression, $Y(A,B,C,D) = \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + A\overline{B}C\overline{D}$ and make a circuit using NAND-NAND. Also, show the simplified NOR-NOR circuit.
26. Given the following Boolean function,
    $Y = B\overline{C}D + \overline{B}\,\overline{C}D + \overline{A}BC + A\overline{B}C + ABC,$
    (i) Obtain the minterms
    (ii) Draw logic circuit

(iii) Simplify the function.

27. For Given combinations of three variable 001, 010, 100 and 111, the output is HIGH, then express the output in maxterms minterms and obtain its simplified SOP/POS. forms.

28. What is the reduced expression of $Y=AB+\overline{AC}+A\overline{B}C\,(AB+C)$

[Raj. 2004]

29. Identefy each of the following expression as SOP, SSOP, POS and SPOS;

(i) $AB+\overline{A}BC+AC\overline{D}$ (ii) $(\overline{A}+B+\overline{C})(A+\overline{B}+C)$

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

(v) $AB+A\overline{B}$ (vi) $(A+B)(\overline{A}+\overline{B})$

(vii) $A+BC$

30. What do you mean by domain of a expression ?

31. Given the logical expression,

$Y = ABC + B\overline{C}D + \overline{A}BC$

(a) Make a truth table

(b) Simplify using K-map

(c) Realize Y using NAND gates only.

32. Draw the logical expression :

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

on a 4 variable K-map. Make a truth table and obtain the simplified expression from the K-map. Design a logic circuit for the realization of simplified expression.

[Raj. 2004]

33. What do you understand by "don't care" state. **[Raj. 2004]**

34. A truth table has a high output, for an input of 0 0 0 0, low output for 0 0 0 1 to 1 0 0 1, and don't care for 1 0 1 0 to 1 1 1 1.

35. Make a Karnaugh map for the following functions.

(i) $Y_1 = ABC + \overline{A}BC + \overline{B}\,\overline{C}$

(ii) $Y_2 = A + B + \overline{C}$ **[Raj. 2005]**

(iii) $Y_3 = AB + \overline{B}CD$

36. Explain the following terms,

(i) Pair (ii) Quad (iii) Octal

37. Discuss the grouping of maxterms/minterms in K-map.

Numerical Questions

1. Realize the following expression.

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

2. Prove that

$A \oplus B = \overline{A} \oplus \overline{B}$

3. Apply De Morgan theorem to each of the given expression :

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

(c) $\overline{AB+CD}$ (d) $\overline{(A+\overline{B}).(\overline{C}+D)}$

[**Ans :** (a) $\overline{A}+\overline{B}\overline{C}$ (b) $\overline{A}+\overline{B}+\overline{C}+\overline{D}$, (c) $(\overline{A}+\overline{B}).(\overline{C}+\overline{D})$, (d) $\overline{A}B+C\overline{D}$]

4. Reduce the following expression :

(i) $ABC\,(ABC + 1)$ (ii) $A+\overline{A}+B+C$

(iii) $AAB + ABB + BCC$ (iv) $AB + B + A + C$

(v) $AB + BB + C + \overline{B}$ (iv) $A(\overline{A}+B)$

[**Ans.** : ABC, (ii) 1, (iii) AB + BC, (iv) A + B + C (v) AB]

5. Using boolean techniques, simplify the following expression,

(i) $AB + A\,(B + C) + B\,(B + C)$ (ii) $AB\,(C + B\overline{D}) + \overline{AB}$

(iii) $A + AB + ABC$ (iv) $A\overline{B}C\,(BD + CDE) + \overline{AC}$

[**Ans.** : (i) $B + AC$, (ii) $A\overline{B\overline{C}}D$, (iii) A (iv) $A(C+\overline{B}DE)$]

6. Prove the following using boolean algebra :

(i) $\underline{\underline{AB + CD = (}}A + C)(B + C)(A + D)(B + D)$

(ii) $\overline{\overline{AB}} + \overline{A} + AB = 1$

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

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

7. If AB = 1, Prove that

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

8. Find the equivalent of the following boolean expressions:

(i) $(A + B)(\overline{A} + C)(B + C)$ **[Raj. 2004]**

(ii) $(\overline{A}B + \overline{A}C)(BC + B\overline{C})(ABC)$ **[Raj. 2004]**

(iii) $\overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C}$ **[Raj. 2005]**

[Raj. 2005]

(iv) $[A\overline{B}\,(C + BD) + \overline{A}\,\overline{B}\,]C$ **[Raj. 2007]**

(v) $A + BC$ **[Raj. 2006]**

(vi) $A + BC + \overline{D}$ **[Raj. 2007]**

(vii) $(A + B).(A + \overline{B})$ **[Raj. 2007]**

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

9. Obtain the minimal Sum-of-Products (SOP) expression for the following function

and implement the same using universal gates.

$Y(A,B,C,D) = \Sigma m\,(0, 2, 3, 5, 7, 8, 13, m) + \Sigma d\,(1, 6, 12)$

10. Using the K-map method. Simplify the following boolean function and obtain (a) Minimum SOP (b) Minimal POS expressions.

$Y = (A, B, C, D) = \Sigma m\,(0, 2, 3, 6, 7) + \Sigma d\,(8, 10, 11, 15)$

[**Ans.** (i) $\bar{A}C + B\bar{D}$ (ii) $\bar{A}(C+\bar{D})(\bar{B}+C)$]

11. Simplify the expressin $Y(A, B, C, D) = \Sigma m\,(1, 5, 10, 11, 1, 13, 15)]$ using K-map method.

[**Ans.** $Y = \bar{A}\bar{C}D + AB\bar{C} + ACD + A\bar{B}C$]

12. Represent the following expression on K-map

(i) $Y\,(A, B, C) = \Sigma m\,\,(0, 2, 3, 6, 7)$

(ii) $Y\,(A, B, C) = \Pi M\,(0, 1, 4, 6, 7)$

(iii) $Y\,(A, B, C, D) = \Sigma m\,\,(1, 3, 4, 5, 7, 9, 11, 13)$

(iv) $Y\,(A, B, C, D) = \Pi M\,(0, 1, 2, 6, 7, 8, 10, 12, 14, 15)$

13. Express the following function in standard SOP from and draw a K-map.

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

14. Minimize the following logic functions using K-maps simplificaiton.

(a) $Y, (A, B, C, D) = \Sigma m\,(1, 3, 5, 8, 9, 11, 15) + \Sigma d\,(2, 13)$

(b) $Y_2\,(A, B, C, D) = \Pi M\,(1, 2, 3, 8, 9, 10, 11, 14) - \Sigma d\,(7, 15)$

15. Write the minimal expression for the following K-map.

CD\AB

1	0	1	1
1	1	1	1
0	1	1	0
1	1	1	0

[**Ans.** $BD + AB + \bar{B}\,\bar{C} + \bar{A}B\bar{D}$]

■■■