

## UNIT - IV

### DIGITAL LOGIC

#### Digital Logic:

Boolean Algebra, Basic and Universal Logic gates, Half adder, Full adder, Simplification of logic expressions using K-maps, Multiplexer, De-multiplexer, Encoder and Decoder.

#### BOOLEAN ALGEBRA

George Boole in 1854 invented a new kind of algebra known as Boolean algebra. It is sometimes called switching algebra.

Boolean algebra is the mathematical frame work on which logic design based. It is used in synthesis & analysis of binary logical function.

#### Laws of Boolean Algebra:

1.  $0' = 1$
2.  $1' = 0$
3. If  $A=0$ ,  $A'=1$
4. IF  $A=1$ ,  $A'=0$
5.  $A'' = A$
6.  $A.0 = 0$
7.  $A.1 = A$
8.  $A.A = A$
9.  $A.A' = 0$
10.  $A+0 = A$
11.  $A+1 = 1$

**12.  $A+A=A$**

**13.  $A+A'=1$**

**14.  $A+AB=A(1+B)=A$**

**15.  $A+A'B=A+AB+A'B=A+B(A+A')=A+B$**

## **LOGIC GATES**

- It is an electronic circuit which makes logic decisions. A logic gate is a digital circuit with one or more input signal and only one output signal. All input or output signals either low voltage or high voltage. A digital circuit is referred to as logic gate for simple reason i.e. it can be analyzed on the basis of Boolean algebra.
- To make logical decisions, three gates are used. They are OR, AND and NOT gate. These logic gates are building blocks which are available in the form of IC.
- The input and output of the binary variables for each gate can be represented in a tabular column or truth table.

**1. OR Gate:** The OR gate performs logical additions commonly known as OR function. The OR gate has two or more inputs and only one output. The operation of OR gate is such that a HIGH(1) on the output is produced when any of the input is HIGH. The output is LOW(0) only when all the inputs are LOW.

- If A & B are the input variables of an OR gate and c is its output, then  $A+B$ . similarly for more than two

variables, the OR function can be expressed as

$$Y=A+B+C.$$

- Logical Symbol:



- Truth table

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

2. **AND Gate** : The AND gate performs logical multiplication, commonly known as AND function. The AND gate has two or more inputs and a single output. The output of an AND gate is HIGH only when all the inputs are HIGH. Even if any one of the input is LOW, the output will be LOW. If a & b are input variables of an AND gate and c is its output, then  $Y=A.B$

### Logic Symbol



### Truth table

Input		Output
A	B	$Y=A.B$
0	0	0
0	1	0
1	0	0
1	1	1

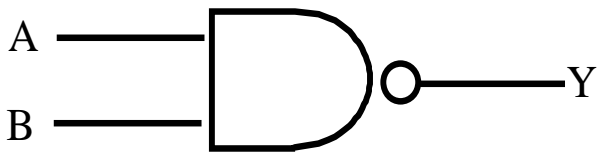
**3. Not Gate ( Inverter):** The NOT gate performs the basic logical function called inversion or complementation. The purpose of this gate is to convert one logic level into the opposite logic level. It has one input and one output. When a HIGH level is applied to an inverter, a LOW level appears at the output and vice-versa.

Truth Table:

Input	output
A	$Y = \overline{A}$
0	1
1	0

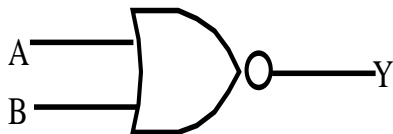


**4.NAND Gate:** The output of a NAND gate is LOW only when all inputs are HIGH and output of the NAND is HIGH if one or more inputs are LOW.



Truth Table:		
Input		Output
A	B	$Y = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

**5. NOR Gate:** The output of the NOR gate is HIGH only when all the inputs are LOW.



Truth table:

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

**6.XOR Gate or Exclusive OR gate:** In this gate output is HIGH only when any one of the input is HIGH. The circuit is also called as inequality comparator, because it produces output when two inputs are different. When both the inputs are high, then the output is low.



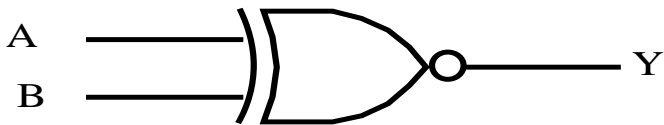
Truth Table:

Input		Output
A	B	$Y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

$$Y = A \oplus B = A \bar{B} + \bar{A} B$$



**7. XNOR Gate or Exclusive NOR Gate:** An XNOR gate is a gate with two or more inputs and one output. XNOR operation is complimentary of XOR operation. i.e. The output of XNOR gate is High, when all the inputs are identical; otherwise it is low.



Truth Table:

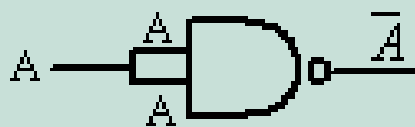
Input		Output
A	B	$Y = \bar{A} \bar{B} + AB$
0	0	1
0	1	0
1	0	0
1	1	1

## **Universal Logic Gate**

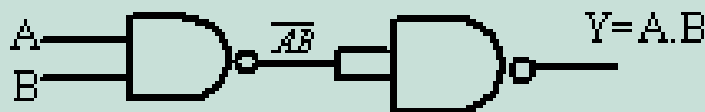
**NAND and NOR** gates are called Universal gates or Universal building blocks, because both can be used to implement any gate like AND, OR and NOT gates or any combination of these basic gates.

## NAND gate as Universal gate

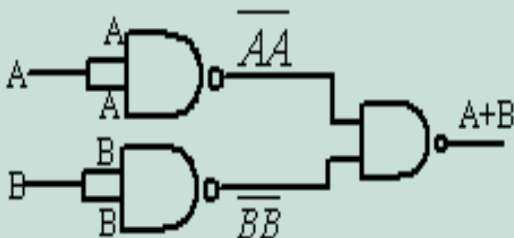
NOT operation:



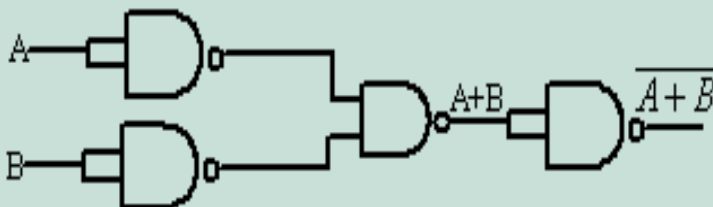
AND operation:



OR operation:

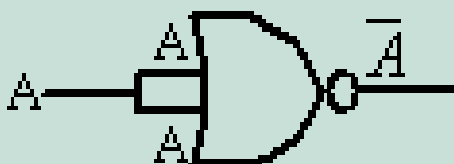


NOR operation:

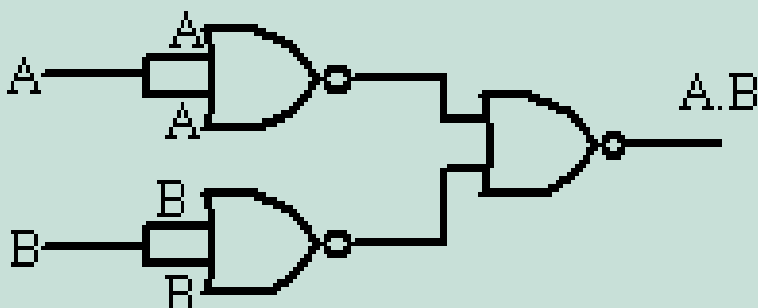


# NOR gate as Universal gate:

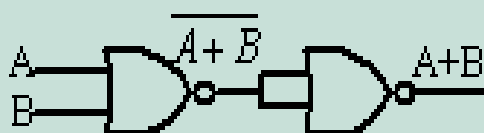
NOT operation:



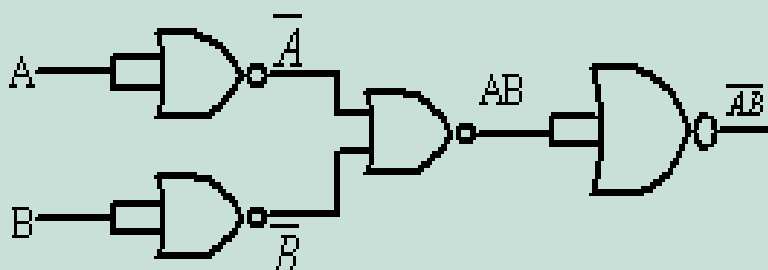
AND operation:



OR operation:

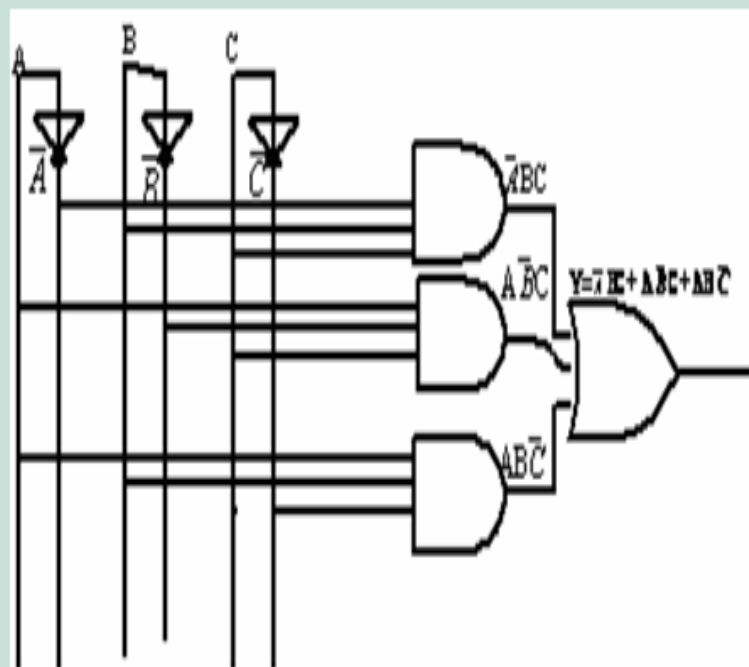


NAND operation:



Draw the logic circuit for the Boolean expression.

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



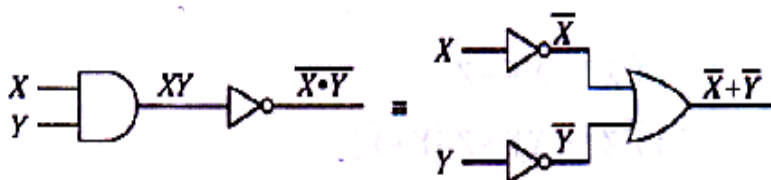
### De Morgan's Theorems:

It is one of the important properties of Boolean algebra. It is extensively useful in simplifying complex Boolean expression.

**Theorem 1:** It states that “the compliments of product of two variables equal to sum of the compliments of individual variable”.

$$\overline{X \bullet Y} = \overline{X} + \overline{Y}$$

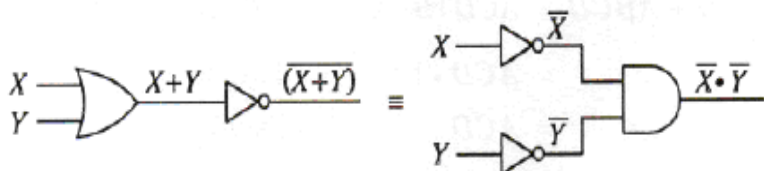
$X$	$Y$	$X \bullet Y$	$\overline{X \bullet Y}$	$X$	$Y$	$\overline{X}$	$\overline{Y}$	$\overline{X} + \overline{Y}$
0	0	0	1	0	0	1	1	1
0	1	0	1	0	1	1	0	1
1	0	0	1	1	0	0	1	1
1	1	1	0	1	1	0	0	0



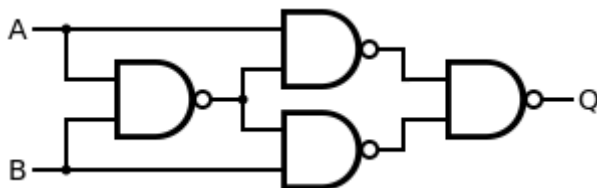
**Theorem 2:** It states that compliment of sum of two variables is equal to product of compliment of two individual variables.

$$\overline{X + Y} = \overline{X} \cdot \overline{Y}$$

X	Y	X+Y	$\overline{X+Y}$	X	Y	$\overline{X}$	$\overline{Y}$	$\overline{X} \cdot \overline{Y}$
0	0	0	1	0	0	1	1	1
0	1	1	0	0	1	1	0	0
1	0	1	0	1	0	0	1	0
1	1	1	0	1	1	0	0	0



## 1. Realize EXOR Gate using only minimum NAND Gates

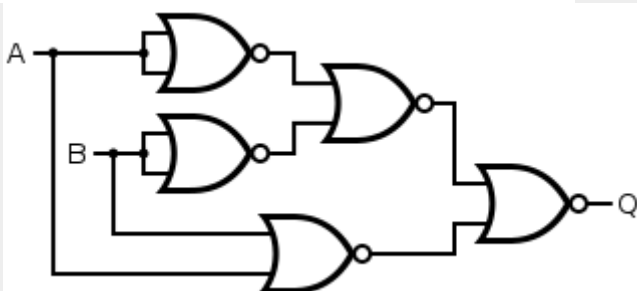




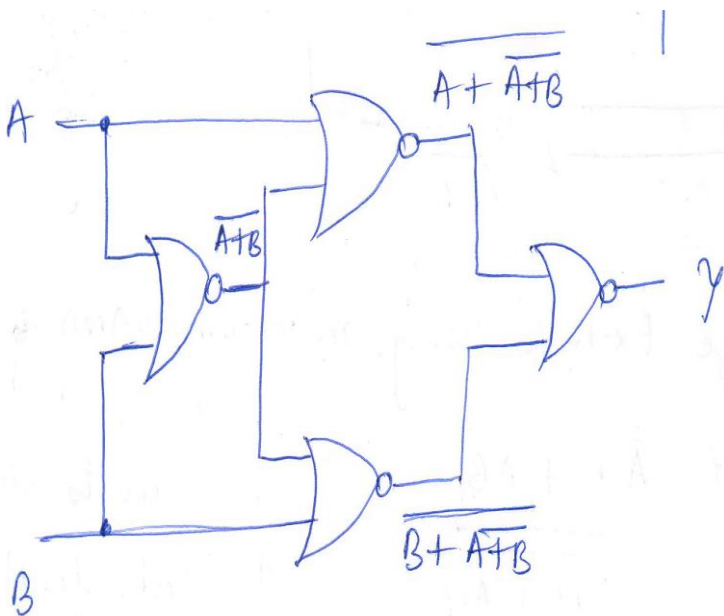
## 2. Realize EXOR Gate using only NOR Gates

$XNOR + NOR = XOR$  i.e. Add NOR gate to the output of XNOR gate. Here is the boolean algebra for XOR gate:

$$\begin{aligned} Y &= (AB + B'A')' \\ &= (AB)' \cdot (B'A')' \\ &= (A' + B') \cdot ((B')' + (A')') \\ &= (A' + B') \cdot (B + A) \\ &= A'B + A'A + BB' + B'A \\ &= A'B + AB' \text{ (Boolean expression for XOR Gate)} \end{aligned}$$



### 3. Realize EX-NOR Gate using only minimum NOR Gates

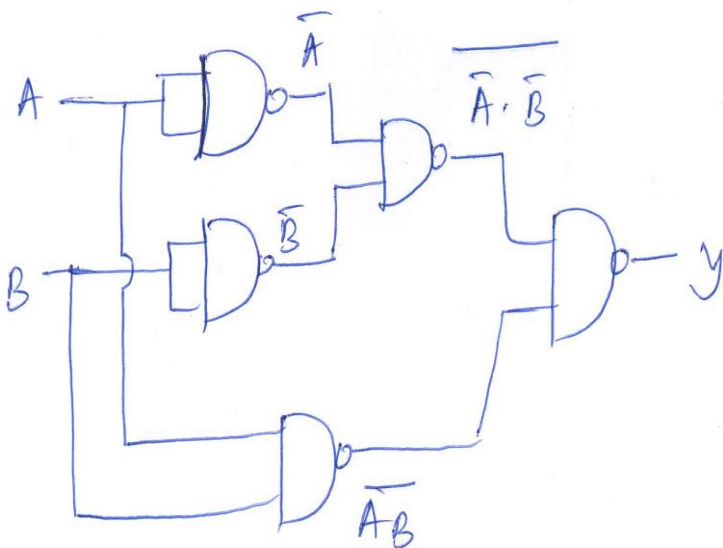


### 4. Realize EX-NOR Gate using only NAND Gates

XOR + NAND Inverter(NOT) = XNOR i.e. Add NOT Gate to the output of XOR gate as shown in the image above.

Here is the boolean algebra for XNOR gate:

$$\begin{aligned} Y &= (AB' + BA')' \\ &= (AB')' \cdot (BA')' \\ &= (A' + (B')') \cdot (B' + (A')') \\ &= (A' + B) \cdot (B' + A) \\ &= A'B' + A'A + BB' + BA \\ &= AB + A'B' \text{ (Boolean expression for XNOR Gate)} \end{aligned}$$



## Canonical and Standard Form

### Canonical Form –

In Boolean algebra, Boolean function can be expressed as Canonical Disjunctive Normal Form known as **minterm** and some are expressed as Canonical Conjunctive Normal Form known as **maxterm**.

In Minterm, we look for the functions where the output results in “1” while in Maxterm we look for function where the output results in “0”.

We perform **Sum of minterm** also known as Sum of products (SOP).

We perform **Product of Maxterm** also known as Product of

sum (POS).

Boolean functions expressed as a sum of minterms or product of maxterms are said to be in canonical form.

- **Example** – Express the Boolean function  $F = xy + x'z$  as a product of maxterms

- **Solution** –

$$F = xy + x'z = (xy + x')(xy + z) = (x + x')(y + x')(x + z)(y + z) = (x' + y)(x + z)(y + z)$$

$$x' + y = x' + y + zz' = (x' + y + z)(x' + y + z')$$

$$x + z = x + z + yy' = (x + y + z)(x + y' + z)$$

$$y + z = y + z + xx' = (x + y + z)(x' + y + z)$$

$$F = (x + y + z)(x + y' + z)(x' + y + z)(x' + y + z')$$

**Standard Form** – A Boolean variable can be expressed in either true form or complemented form. In standard form Boolean function will contain all the variables in either true form or complemented form while in canonical number of variables depends on the output of SOP or POS.

A Boolean function can be expressed algebraically from a given truth table by forming a :

- minterm for each combination of the variables that produces a 1 in the function and then taking the OR of all those terms.
- maxterm for each combination of the variables that produces a 0 in the function and then taking the AND of all those terms.
- **Example** – Convert Boolean expression in standard form  $F = y' + xz' + xyz$
- **Solution** –  $F = (x+x')y'(z+z') + x(y+y')z' + xyz$   
 $F = xy'z + xy'z' + x'y'z + x'y'z' + xyz' + xy'z' + xyz$

			<i>Minterms</i>	<i>Maxterms</i>
<i>X</i>	<i>Y</i>	<i>Z</i>	<i>Product Terms</i>	<i>Sum Terms</i>
0	0	0	$m_0 = \overline{X} \cdot \overline{Y} \cdot \overline{Z} = \min(\overline{X}, \overline{Y}, \overline{Z})$	$M_0 = X + Y + Z = \max(X, Y, Z)$
0	0	1	$m_1 = \overline{X} \cdot \overline{Y} \cdot Z = \min(\overline{X}, \overline{Y}, Z)$	$M_1 = X + Y + \overline{Z} = \max(X, Y, \overline{Z})$
0	1	0	$m_2 = \overline{X} \cdot Y \cdot \overline{Z} = \min(\overline{X}, Y, \overline{Z})$	$M_2 = X + \overline{Y} + Z = \max(X, \overline{Y}, Z)$
0	1	1	$m_3 = \overline{X} \cdot Y \cdot Z = \min(\overline{X}, Y, Z)$	$M_3 = X + \overline{Y} + \overline{Z} = \max(X, \overline{Y}, \overline{Z})$
1	0	0	$m_4 = X \cdot \overline{Y} \cdot \overline{Z} = \min(X, \overline{Y}, \overline{Z})$	$M_4 = \overline{X} + Y + Z = \max(\overline{X}, Y, Z)$
1	0	1	$m_5 = X \cdot \overline{Y} \cdot Z = \min(X, \overline{Y}, Z)$	$M_5 = \overline{X} + Y + \overline{Z} = \max(\overline{X}, Y, \overline{Z})$
1	1	0	$m_6 = X \cdot Y \cdot \overline{Z} = \min(X, Y, \overline{Z})$	$M_6 = \overline{X} + \overline{Y} + Z = \max(\overline{X}, \overline{Y}, Z)$
1	1	1	$m_7 = X \cdot Y \cdot Z = \min(X, Y, Z)$	$M_7 = \overline{X} + \overline{Y} + \overline{Z} = \max(\overline{X}, \overline{Y}, \overline{Z})$

## **Half Adder:**

A combinational circuit which performs the arithmetic addition of two binary digits is called Half Adder. In the half

adder circuit, there are two inputs, one is addend and augends and two outputs are Sum and Carry.

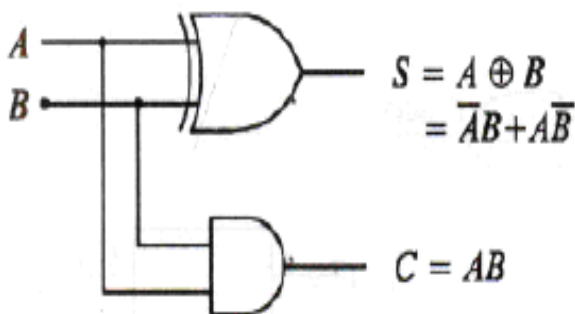
Truth Table for Half Adder

Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

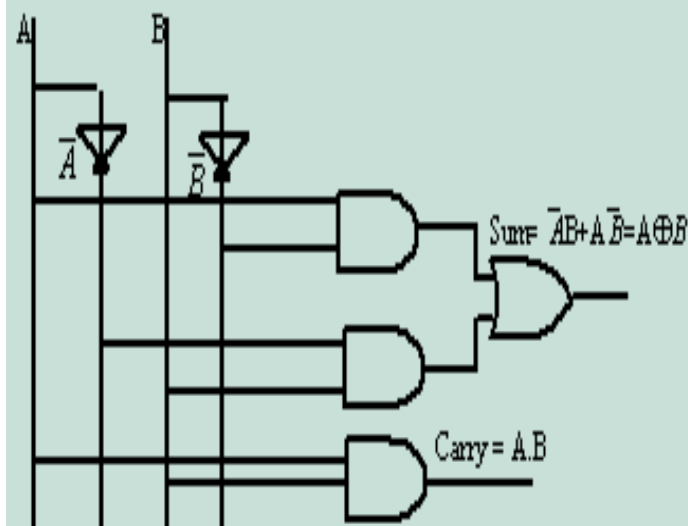
**Logic Expression :**

$$\begin{aligned}\text{Sum} &= \bar{A} B + A \bar{B} = A \oplus B \\ \text{Carry} &= A \cdot B\end{aligned}$$

**Circuit Symbol of Half adder:**



The circuit for Half Adder using Basic Gates is as follows:

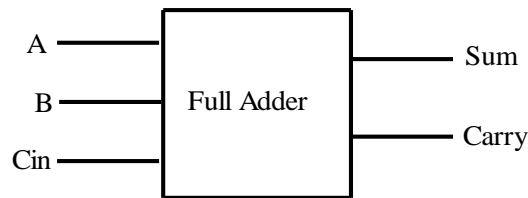


**Full Adder:** The full adder is a combinational circuit that performs the arithmetic sum of three input bits.

- It consists of three inputs and two outputs. Two of the inputs are variables, denoted by A and B, represent the two significant bit to be added. The third input  $C_{in}$

represents carry form the previous lower significant position.

**Truth Table for Full Addder**



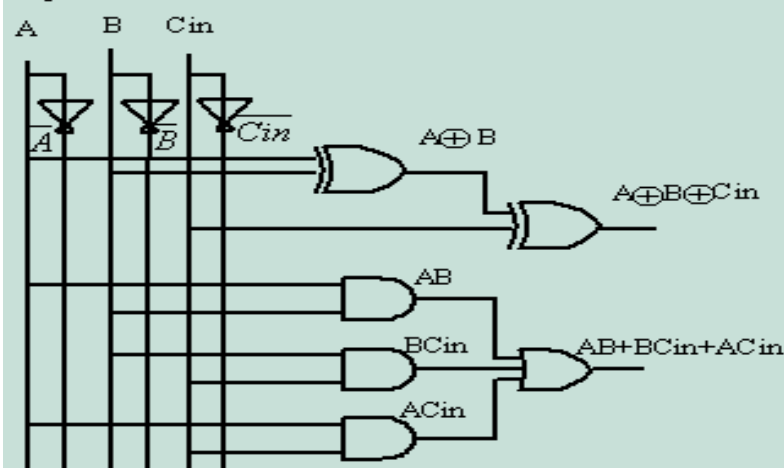
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



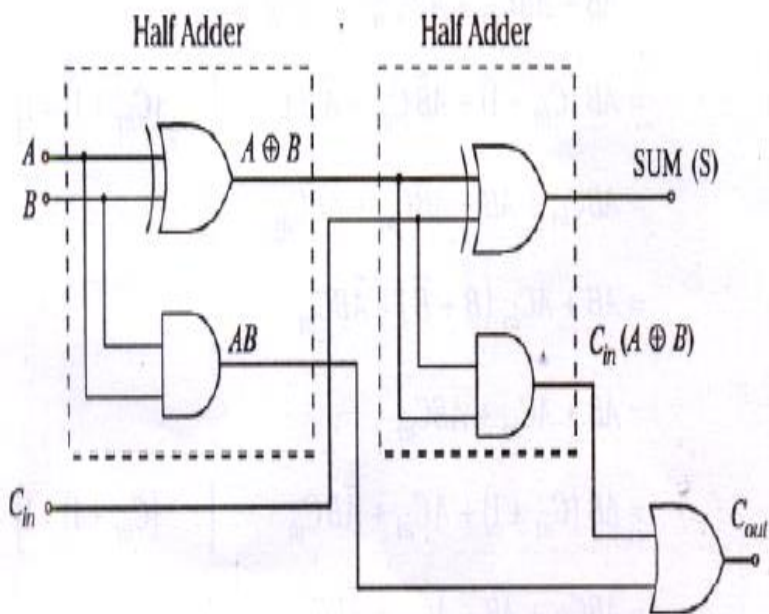
$$\begin{aligned}
 \text{Sum} &= \bar{A} \bar{B} \text{Cin} + \bar{A} B \overline{\text{Cin}} + A \bar{B} \overline{\text{Cin}} + A B \text{Cin} \\
 &= \bar{A} [ \bar{B} \text{Cin} + B \overline{\text{Cin}} ] + A [ \bar{B} \overline{\text{Cin}} + B \text{Cin} ] \\
 &= \bar{A} [ B \oplus \text{Cin} ] + A [ \overline{B \oplus \text{Cin}} ] \\
 &= A \oplus B \oplus \text{Cin}
 \end{aligned}$$

$$\begin{aligned}
 \text{Carry} &= \bar{A} B \text{Cin} + A \bar{B} \text{Cin} + A B \overline{\text{Cin}} + A B \text{Cin} \\
 &= \bar{A} B \text{Cin} + A \bar{B} \text{Cin} + A B (\overline{\text{Cin}} + \text{Cin}) \\
 &= \bar{A} B \text{Cin} + A \bar{B} \text{Cin} + A B \\
 &= \bar{A} B \text{Cin} + A (\bar{B} \text{Cin} + B) \\
 &= \bar{A} B \text{Cin} + A B + A \text{Cin} \\
 &= B (\bar{A} \text{Cin} + A) + A \text{Cin} \\
 &= B (A + \text{Cin}) + A \text{Cin} \\
 &= A B + B \text{Cin} + A \text{Cin}
 \end{aligned}$$

Implementation of Full Adder:



**Full adder Circuit using two Half adders:**



**Fig 5.4:Full adder Circuit using two Half adders**

## SIMPLIFICATION USING BOOLEAN ALGEBRA

A simplified Boolean expression uses the fewest gates possible to implement a given expression.

### Example

Using Boolean algebra techniques,

simplify this

expression:

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

### Solution

Step 1: Apply the distributive law to the second and third terms in the expression, as follows:

$$AB + AB + AC + BB + BC$$

Step 2: Apply rule 7 ( $BB = B$ ) to the fourth term.

$$AB + AB + AC + B + BC$$

Step 3: Apply rule 5 ( $AB + AB = AB$ ) to the first two terms.

$$AB + AC + B + BC$$

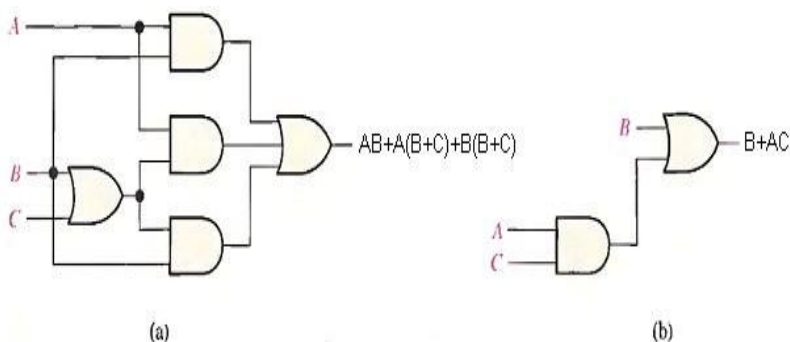
Step 4: Apply rule 10 ( $B + BC = B$ ) to the last two terms.

$$AB + AC + B$$

Step 5: Apply rule 10 ( $AB + B = B$ ) to the first and third terms.

$$B + AC$$

At this point the expression is simplified as much as possible.



## Boolean Algebra Practice Problems

- |                                                |                                                    |
|------------------------------------------------|----------------------------------------------------|
| 1) $a + 0 =$ _____                             | 14) $y + \overline{xy} =$ _____                    |
| 2) $\overline{a} \cdot 0 =$ _____              | 15) $\overline{xy} + x \cdot y =$ _____            |
| 3) $a + \overline{a} =$ _____                  | 16) $\overline{x} + \overline{yx} =$ _____         |
| 4) $a + a =$ _____                             | 17) $\overline{(w + x + y + z)} \cdot y =$ _____   |
| _____                                          | 5) $\overline{a} + ab =$ _____                     |
| _____                                          | 18) $(x + y)(x + y) =$ _____                       |
| _____                                          | 6) $\overline{a} + ab =$ _____                     |
| _____                                          | 19) $w + [w + (\overline{wx})] =$ _____            |
| 7) $\overline{a(a + b)} =$ _____               | 20) $x[x + (\overline{xy})] =$ _____               |
| 8) $ab + \overline{ab} =$ _____                | 21) $\overline{(x + x)} =$ _____                   |
| 9) $\overline{(a + b)(a + b)} =$ _____         | 22) $\overline{(x + x)} =$ _____                   |
| 10) $\overline{a(a + b + c + \dots)} =$ _____  | 23) $\overline{w + (\overline{wxyz})} =$ _____     |
| For (11), (12), (13), $f(a, b, c) = a + b + c$ | 24) $\overline{w \cdot (\overline{wxyz})} =$ _____ |

$$11) f(a, b, ab) = \underline{\hspace{2cm}} \quad 25) \overline{xz} + \overline{xy} + \overline{zy} = \underline{\hspace{2cm}}$$

$$12) f(a, b, a \cdot b) = \underline{\hspace{2cm}} \quad 26) (x + z)(\overline{x} + y)(z + y) = \underline{\hspace{2cm}}$$

$$13) f[a, b, (ab)] = \underline{\hspace{2cm}} \quad 27) \overline{x} + y + xy\overline{z} = \underline{\hspace{2cm}}$$

## Solutions to the Boolean Algebra Practice Problems

$$1) a + 0 = a$$

$$2) a \cdot 0 = 0$$

$$3) a + \overline{a} = 1$$

$$1) a + a = a$$

$$5) a + ab = \underline{a(1 + b)} = a$$

$$6) a + ab = (a + \underline{a})(\overline{a} + b) = a + b$$

$$7) \underline{a}(\overline{a} + b) = \overline{a}a + ab = ab$$

$$8) ab + \overline{ab} = \underline{b(a + \overline{a})} = b$$

$$9) (\overline{a} + \underline{\overline{b}})(a + b) = \overline{a}a + ab + \overline{b}a + \overline{b}b = \overline{a}a + ab + \overline{a}b + \overline{b}b = \overline{a}(1 + \overline{b} + b) = a$$

$$10) \underline{a}(a + b + c + \dots) = aa + ab + ac + \dots = a + ab + ac + \dots = a$$

$$11) f(a, b, ab) = a + b + ab = a + b$$

$$12) f(a, b, a \cdot \overline{b}) = a + b + ab = \overline{a} + b + a = 1$$

$$13) f[a, b, (ab)] = a + b + (ab) = \overline{a} + b + a + b = 1$$

$$14) y + \overline{y}y = y$$

$$15) \overline{xy} + x \cdot \overline{y} = \underline{\overline{x}(\overline{y} + y)} = \overline{x}$$

$$16) x + \overline{y}x = \underline{\overline{x}(1 + y)} = \overline{x}$$

$$17) (w + x + y + z) \cdot \overline{y} = \underline{\overline{y}}$$

$$18) (x + \underline{\overline{y}})(x + y) = x$$

$$19) w + [w + (\overline{wx})] = w$$

$$20) \underline{\overline{x}}[x + (\overline{xy})] = x$$

$$21) \overline{(x + \overline{x})} = x$$

$$22) (x + \overline{x}) = 0$$

$$23) \overline{w + (\overline{wxy})} = \overline{w(1 + \overline{xyz})} = \overline{w}$$

$$24) \overline{w \cdot (\overline{wxyz})} = \overline{w(w + x + y + z)} = \overline{w}$$

$$25) \overline{xz} + \overline{xy} + \overline{zy} = \overline{xz} + \overline{xy}$$

$$26) (x + \underline{\overline{z}})(\overline{x} + y)(z + y) = (x + z)(\overline{x} + y) = \overline{xy} + \overline{xz}$$

$$27) \overline{x} + y + xy\overline{z} = \overline{x} + y + z$$

### Simplify the Boolean expression

1.  $\overline{X} \overline{Y} Z + \overline{X} Y Z$

$$= \overline{X} Z [\overline{Y} + Y]$$

$$= \overline{X} Z [1] = \overline{X} Z$$

2.  $f = X(\overline{X} + Y)$

$$= X \overline{X} + XY = 0 + XY = XY$$

3.  $f = B(A+C)+C$

$$= BA + BC + C$$

$$= BA + C(1+B)$$

$$= BA + C$$

4.  $XY + XYZ + XY \overline{Z} + \overline{X} YZ$

$$= XY(Z + \overline{Z}) + XYZ + XY \overline{Z} + \overline{X} YZ$$

$$XYZ + XY \overline{Z} + XY \overline{Z} + \overline{X} YZ + XYZ$$

$$XYZ(1+1) + XY \overline{Z}(1+1) + \overline{X} YZ$$

$$XYZ + XY \overline{Z} + \overline{X} YZ$$

$$XY(Z + \overline{Z}) + \overline{X} YZ$$

$$XY + \overline{X} YZ$$

$$Y(X + \overline{X} Z)$$

$$Y(X + Z)$$

5.  $XYZ + \overline{X} Y + XY \overline{Z}$

$$= Y(\overline{X} + X \overline{Z}) + XY \overline{Z}$$

$$= Y(\overline{X} + \overline{Z}) + XY \overline{Z}$$

$$= Y \overline{X} + Y \overline{Z} + XY \overline{Z}$$

$$= Y \overline{X} + Y(Z + X \overline{Z})$$

$$= Y \overline{X} + Y(Z + X)$$

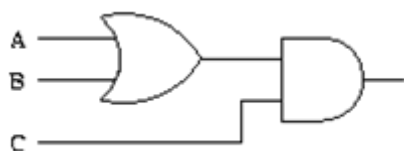
$$= Y(Z + \overline{X} + X)$$

$$= Y(Z + 1)$$

$$= Y$$

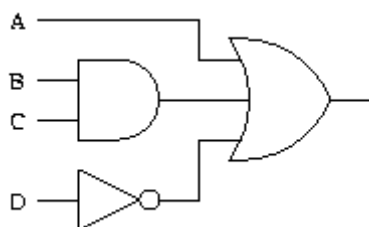
- Draw a logic circuit for  $(A + B)C$ .

Soln:



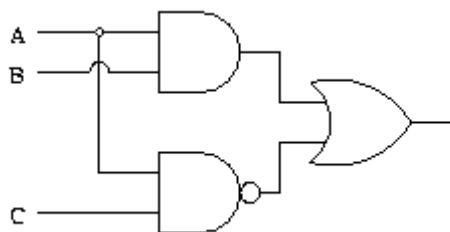
- Draw a logic circuit for  $A + BC + D$ .

Soln:



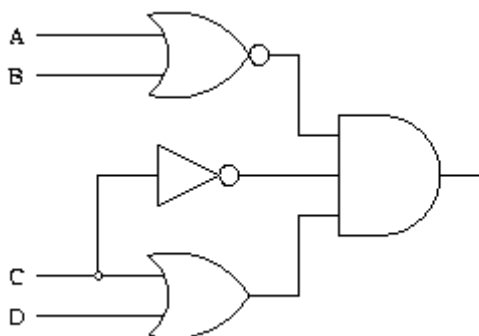
- Draw a logic circuit for  $AB + AC$ .

Soln:



- Draw a logic circuit for  $(A + B)(C + D)C$ .

Soln:



## 2-variable k-map

There are 4 cells (2<sup>2</sup>) in the 2-variable k-map. It will look like (see below image)

	X	X'
Y		
Y'		

The possible min terms with 2 variables (A and B) are  $A.B$ ,  $A.B'$ ,  $A'.B$  and  $A'.B'$ . The conjunctions of the variables (A, B) and (A', B) are represented in the cells of the top row and (A, B') and (A', B') in cells of the bottom row. The following table shows the positions of all the possible outputs of 2-variable Boolean function on a K-map.



A	B	Possible Outputs	Location on K-map
0	0	$A'B'$	0
0	1	$A'B$	1
1	0	$AB'$	2
1	1	$AB$	3



A general representation of a 2 variable K-map plot is shown below.

		B	
		0	1
A	0	$A'B'$ 0	$A'B$ 1
	1	$AB'$ 2	$AB$ 3



When we are simplifying a Boolean equation using Karnaugh map, we represent the each cell of K-map containing the conjunction term with 1. After that, we group the adjacent cells with possible sizes as 2 or 4. In case of larger k-maps, we can group the variables in larger sizes like 8 or 16.

The groups of variables should be in rectangular shape, that means the groups must be formed by combining adjacent cells either vertically or horizontally. Diagonal shaped or L-shaped groups are not allowed. The following example demonstrates a K-map simplification of a 2-variable Boolean equation.

Example

Simplify the given 2-variable Boolean equation by using K-map.

$$F = XY' + X'Y + X'Y'$$

First, let's construct the truth table for the given equation,

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	1

We put 1 at the output terms given in equation.

	X	X'
Y		1
Y'	1	1

In this K-map, we can create 2 groups by following the rules for grouping, one is by combining ( $X'$ , Y) and ( $X'$ , Y') terms and the other is by combining ( $X$ , Y') and ( $X'$ , Y') terms. Here the lower right cell is used in both groups.

After grouping the variables, the next step is determining the minimized expression.

By reducing each group, we obtain a conjunction of the minimized expression such as by taking out the common terms from two groups, i.e.  $X'$  and  $Y'$ . So the reduced equation will be  $X' + Y'$ .

### 3 variable K-maps

For a 3-variable Boolean function, there is a possibility of 8 output min terms. The general representation of all the min terms using 3-

variables is shown below.

A	B	C	Output Function	Location on K-map
0	0	0	$A'B'C'$	0
0	0	1	$A'B'C$	1
0	1	0	$A'BC'$	2
0	1	1	$A'BC$	3
1	0	0	$AB'C'$	4
1	0	1	$AB'C$	5
1	1	0	$ABC'$	6
1	1	1	$ABC$	7

A typical plot of a 3-variable K-map is shown below. It can be observed that the positions of columns 10 and 11 are interchanged so that there is only change in one variable across adjacent cells. This modification will allow in minimizing the logic.

		BC			
		00	01	11	10
A	0	<sup>0</sup> $A'B'C'$	<sup>1</sup> $A'B'C$	<sup>3</sup> $A'BC$	<sup>2</sup> $A'BC'$
	1	<sup>4</sup> $AB'C'$	<sup>5</sup> $AB'C$	<sup>7</sup> $ABC$	<sup>6</sup> $ABC'$

Up to 8 cells can be grouped in case of a 3-variable K-map with other possibilities being 1,2 and 4.

### Example

Simplify the given 3-variable Boolean equation by using k-map.

$$F = X' Y Z + X' Y' Z + X Y Z' + X' Y' Z' + X Y Z + X Y' Z'$$

First, let's construct the truth table for the given equation,

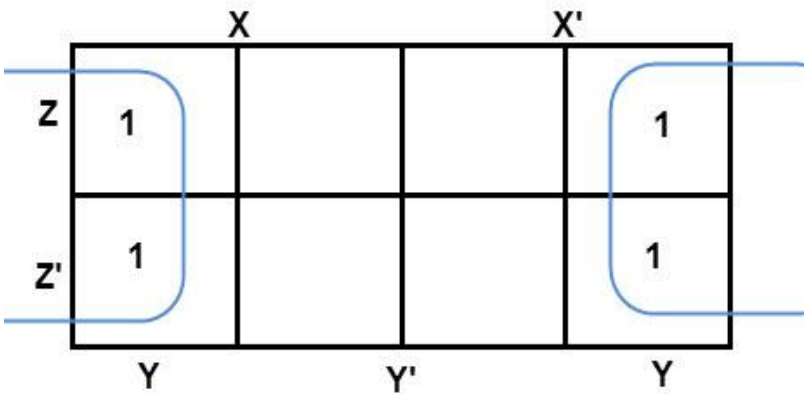
x	y	z	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

?

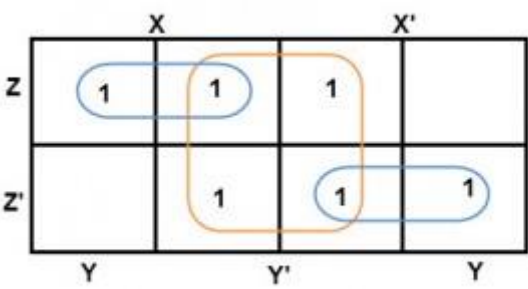
We put 1 at the output terms given in equation.

There are 8 cells (23) in the 3-variable k-map. It will look like (see below image).

The largest group size will be 8 but we can also form the groups of size 4 and size 2, by possibility. In the 3 variable Karnaugh map, we consider the left most column of the k-map as the adjacent column of rightmost column. So the size 4 group is formed as shown below.



And in both the terms, we have 'Y' in common. So the group of size 4 is reduced as the conjunction Y. To consume every cell which has 1 in it, we group the rest of cells to form size 2 group, as shown below.



2

The 2 size group has no common variables, so they are written with their variables and its conjugates. So the reduced equation will be  $X Z' + Y' + X' Z$ . In this equation, no further minimization is possible.

## 4 variable K-maps

There are 16 possible min terms in case of a 4-variable Boolean function. The general representation of minterms using 4 variables is shown below.

A	B	C	D	Output function	K-map location
0	0	0	0	$A'B'C'D'$	0
0	0	0	1	$A'B'C'D$	1
0	0	1	0	$A'B'CD'$	2
0	0	1	1	$A'B'CD$	3
0	1	0	0	$A'BC'D'$	4
0	1	0	1	$A'BC'D$	5
0	1	1	0	$A'BCD'$	6
0	1	1	1	$A'BCD$	7
1	0	0	0	$AB'C'D'$	8
1	0	0	1	$AB'C'D$	9
1	0	1	0	$AB'CD'$	10
1	0	1	1	$AB'CD$	11
1	1	0	0	$ABC'D'$	12
1	1	0	1	$ABC'D$	13
1	1	1	0	$ABCD'$	14
1	1	1	1	$ABCD$	15

A typical 4-variable K-map plot is shown below. It can be observed that both the columns and rows of 10 and 11 are interchanged.

CD		00	01	11	10
AB	00	0 $A'B'C'D'$	1 $A'B'C'D$	3 $A'B'CD$	2 $A'B'CD'$
	01	4 $A'BC'D'$	5 $A'BC'D$	7 $A'BCD$	6 $A'BCD'$
	11	12 $ABC'D'$	13 $ABC'D$	15 $ABCD$	14 $ABCD'$
	10	8 $AB'C'D'$	9 $AB'C'D$	11 $AB'CD$	10 $AB'CD'$

The possible number of cells that can be grouped together are 1, 2, 4, 8 and 16.

### Example

Simplify the given 4-variable Boolean equation by using k-map.  $F(W, X, Y, Z) = (1, 5, 12, 13)$

Sol:  $F(W, X, Y, Z) = (1, 5, 12, 13)$

YZ		00	01	11	10
WX	00		1		
	01		1		
	11	1	1		
	10				

?

By preparing k-map, we can minimize the given Boolean equation as  $F = WY'Z + W'Y'Z$