Lecture 28

Introduction of Number system and conversion among them

INTRODUCTION

- A number system defines how a number can be represented using distinct digits or symbols.
- A number can be represented differently in different systems. For example, the two numbers $(2A)_{16}$ and $(52)_8$ both refer to the same quantity, $(42)_{10}$, but their representations are different.
- Types of Number Systems:

1. Non-positional number systems
2. Positional number systems

NON-POSITIONAL NUMBER SYSTEMS

- Non-Positional Number System does not use digits for the representation instead it use symbols for the representation.
- A non-positional number system still uses a limited number of symbols in which each symbol has a value.
- The value of each symbol is fixed.
- Digit value is independent of its position.
- **DIFFICULTY**:

Difficulty to perform arithmetic with such a number system

Roman numerals are a good example of a non-positional number system. This number system has a set of symbols S = {I, V, X, L, C, D, M}. The values of each symbol are shown as:

Symbol	*I*	*V*	*X*	*L*	*C*	*D*	*M*
Value	1	5	10	50	100	500	1000

To find the value of a number, we need to add the value of symbols subject to specific rules .

POSITIONAL NUMBER SYSTEMS

GROUP OF INSTITUTIONS

In a positional number system, the position a symbol occupies in the number determines the value it represents. In this system, a number represented as:

$$\boxed{\pm\,(S_{k-1}\ \ldots\ S_2\ S_1\ S_0.\ S_{-1}\ S_{-2}\ \ldots\ S_{-l})_b}$$

has the value of:

$$n = \pm\ S_{k-1} \times b^{k-1} + \ldots + S_1 \times b^1 + S_0 \times b^0 \quad + \quad S_{-1} \times b^{-1} + S_{-2} \times b^{-2} + \ldots + S_{-l} \times b^{-l}$$

in which S is the set of symbols, b is the base (or radix).

- In a Positional Number System there are only a few symbols that represent different values, depending on the position they occupy in a number.
- The value of each digit in such a number is determined by three considerations
  - a. The digit itself
  - b. The position of the digit in the number
  - c. The base of the number system(where base is defined as the total number of digits available in the number system)

- In computer **real** numbers are referred to as floating point numbers.
- Floating point numbers are represented as

<*Integer part***>** **<***Radix Point***>** **<***Fractional part***>**

34568 . 56735

34568.56735

- Number systems include decimal, binary, octal and hexadecimal
- Each system have four number base

Number System	**Base**	**Symbol**
Binary	**Base** 2	**B**
Octal	**Base** 8	**O**
Decimal	**Base** 10	**D**
Hexadecimal	**Base** 16	**H**

DECIMAL NUMBER SYSTEM

Decimal (BASE 10)

Ten symbols 0,1,2,3,4,5,6,7,8,9 called **digits**.

Decimal numbers

Integer numbers

Integers are whole numbers

Examples 1, 2, -3, 50, 675, -560, …...

Real numbers

Numbers that has fractions like 687. 345, -49.56, …

In decimal number system the value of a digit is determined by $digit \times 10^{position}$.

In **integer numbers** the position is defined as 0,1,2,3,4,5,… starting from the rightmost position and moving one position at a time towards left.

Position	4	3	2	1	0
$10^{position}$	10^4	10^3	10^2	10^1	10^0
Position value	10000	1000	100	10	1
Digits	7	2	1	3	4
Digit Value	7×10^4	2×10^3	1×10^2	3×10^1	4×10^0

Digit Value	7×10^4	2×10^3	1×10^2	3×10^1	4×10^0
Digit Value	70000	2000	100	30	4
Integer Number	70000	+ 2000	+ 100	+ 30	+ 4

$$\boxed{72134}$$

Decimal Number System

In **floating point numbers** the position is defined as 0,1,2,3,4,5,… starting from the radix point and moving one position at a time towards left, and -1,-2,-3, … starting from the radix point and moving towards right one position at a time.

Position	2	1	0	Radix	-1	-2
Place Value	10^{2}	10^{1}	10^{0}		10^{-1}	10^{-2}
Digits	4	3	6		8	5

$$\mathbf{436.85} = 4 \times 100 + 3 \times 10 + 6 \times 1 \;.\; 8 \times 0.1 + 5 \times 0.01$$

Why Binary System?

- Computers store **numeric** (numbers) as well **non- numeric** (text, images and others) data in binary representation (binary number system).
- Each state can be represented by a number – 1 for "ON" and 0 for "OFF
- Binary number system is a two digits (0 and 1), also referred to as bits, so it is a base 2 system.
- In this system, the position definition is same as in decimal number system.
- In binary number system the value of a digit is determined by *digit* $\times \mathbf{2}^{position}$.

Binary Number System

Value = digit ×

Position	4	3	2	1	0
2^{position}	2^4	2^3	2^2	2^1	2^0
Position value	16	8	4	2	1
Binary Digits	1	1	1	0	1
Digit Value	1×2^4	1×2^3	1×2^2	0×2^1	1×2^0
	1×16	1×8	1×4	0×2	1×1
	16	8	4	0	1

$$(11101)_2 \quad + \quad = (29)_{10}$$

Binary Number System

Floating Point Number $(101.11)_2 = (5.75)_{10}$

Position	2	1	0	Radix	-1	-2
Place Value	2^{2}	2^{1}	2^{0}		2^{-1}	2^{-2}
	4	2	1		0.5	0.25
Digits	1	0	1		1	1
	1 x 4	0 x 2	1 x 1		1 x 0.5	1 x 0.25
	4	+ 0	+ 1		0.5	+ 0.25

$\underbrace{4 + 0 + 1}_{5}$ $\underbrace{0.5 + 0.25}_{75}$

OCTAL NUMBER SYSTEM

- Base 8
- Two Digits: 0, 1,2,3,4,5,6,7
- Example: 217_8
- Positional Number System

$$8^{n-1} \ \dots \ 8^3 \ 8^2 \ 8^1 \ 8^0$$

$$d^{n-1} \ \dots \ d^3 \ d^2 \ d^1 \ d^0$$

- Bit d_o is the least significant bit (LSB).
- Bit d_{n-1} is the most significant bit (MSB).

NUMBER CONVERSIONS

Binary, Octal and Hexadecimal To Decimal

- $(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
  - $= 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$
  - $= 8+4+0+1$
  - $= (13)_{10}$

Binary to decimal

- $(2057)_8 = 2 \times 8^3 + 0 \times 8^2 + 5 \times 8^1 + 7 \times 8^0$
  - $= 2 \times 512 + 0 \times 64 + 5 \times 8 + 7 \times 1$
  - $= 1024+0+40+7$
  - $= (1071)_{10}$

Octal to decimal

- $(1AF)_{16} = 1 \times 16^2 + A \times 16^1 + F \times 16^0$
  - $= 1 \times 256 + 10 \times 16 + 15 \times 1$
  - $= 256+160+15$
  - $= (431)_{10}$

Hexadecimal to decimal

More examples

Ternary (base-3) numbers
Quaternary (base-4) numbers
Quinary (base-5) numbers

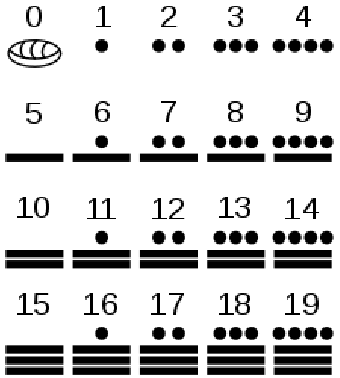
$(211)_3 = 2 \times 3^2 + 1 \times 3^1 + 1 \times 3^0$
$= 18 + 3 + 1$
$= (22)_{10}$

$(211)_4 = 2 \times 4^2 + 1 \times 4^1 + 1 \times 4^0$
$= 32 + 4 + 1$
$= (37)_{10}$

$(211)_5 = 2 \times 5^2 + 1 \times 5^1 + 1 \times 5^0$
$= 50 + 5 + 1$
$= (56)_{10}$

Senary (base-6) numbers
?? (base-7) numbers
Tridecimal or Tredecimal (base-13) numbers

Mayan number (base-20) system



Ex.

$(211)_6 = (?)_{10}$

$(211)_7 = (?)_{10}$

$(211)_{13} = (?)_{10}$

$(211)_{20} = (?)_{10}$

From Decimal to Another Base

1. Divide the decimal number by the new base.
2. Record the remainder as the right most digit.
3. Divide the quotient of the previous divide by the new base.
4. Record the remainder as the next digit.
5. Repeat step 3& 4 until the quotient becomes 0 in step 3.

Example

Convert $(25)_{10}=()_2$

Number/ Base	Quotient	Reminder
25/2	12	1
12/2	6	0
6/2	3	0
3/2	1	1
1/2	0	1

$\boxed{(25)_{10}=(11001)_2}$

From Decimal to Another Base

Convert $(42)_{10}=()_2$

2	42	Remainder
	21	0
	10	1
	5	0
	2	1
	1	0
	0	1

Convert $(42)_{10}=(101010)_2$

Convert $(952)_{10}=()_8$

8	952	Remainder
	119	0
	14	7
	1	6
	0	1

Convert $(952)_{10}=(1670)_8$

From Decimal to Another Base

Convert

$(428)_{10} = ()_{16}$

16	428	Remainder
	26	12 C
	1	10 A
	0	1

Convert

$(428)_{10} = (1AC)_{16}$

Convert

$(100)_{10} = ()_{5}$

5	100	Remainder
	20	0
	4	0
	0	4

Convert

$(100)_{10} = (400)_{5}$

From Decimal to Another Base

Convert
$(100)_{10}=()_4$

4	100	Remainder
	25	0
	6	1
	1	2
	0	1

Convert
$(100)_{10}=(1210)_4$

Convert
$(1715)_{10}=()_{12}$

12	1715	Remainder	
	142	11	B
	11	10	A
	0	11	B

Convert
$(1715)_{10}=(BAB)_{12}$

Converting from a base other than to a base other than 10

1. Convert the original number to a decimal number.
2. Convert that decimal number to the new base.

Convert $(545)_6$ to $()_4$

$$(545)_6 = 5 \times 6^2 + 4 \times 6^1 + 5 \times 6^0 = 5 \times 36 + 4 \times 6 + 5 \times 1 = 180 + 24 + 5 = (209)_{10}$$

4	209	Remainder
	52	1
	13	0
	3	1
	0	3

$$545)_6 = (209)_{10} = (3101)_4$$

Converting form a base other than to a base other than 10

Convert $(101110)_2$ to $()_8$

$(101110)_2 = 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = (46)_{10}$

8	46	Remainder
	5	6
	0	5

$(101110)_2 = (46)_{10} = (56)8$

Convert $(11010011)_2$ to $()_{16}$

$(11010011)_2 = 1 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = (211)_{10}$

16	211	Remainder	
	13	3	3
	0	13	D

$(11010011)_2 = (211)_{10} = D3)_{16}$

BINARY TO OCTAL

1. Start from the rightmost position, make groups of three binary digits.
2. Convert each group into octal digit

Convert $(101110)_2$ to $()_8$

$$\underbrace{101}_{(5}\ \underbrace{110}_{6)_8}$$

OCTAL TO BINARY

1. Convert each octal to three digit binary.

2. Combine them in a single binary number

$$(\underbrace{101}_{5}\ \underbrace{110}_{6})_2 \quad (56)_8$$

Convert $(562)_8$ to $()_2$

$$\overbrace{101}^{5}\quad\overbrace{110}^{6}\quad\overbrace{010}^{2}$$

Convert $(6751)_8$ to $()_2$

$$\overbrace{110}^{6}\quad\overbrace{111}^{7}\quad\overbrace{101}^{5}\quad\overbrace{001}^{1}$$

Binary to Hexadecimal

1.Starting from the right most position make groups of 4 binary digits

2.Convert each group its hexadecimal equivalent digits (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Convert $(10\ 1110\ 0000\ 1000)_2$ to $()_{16}$

10 1110 0000 1000

(2) (14) (0) (8)

$(2E08)_{16}$

Hexadecimal to Binary conversion

1.Convert each hexadecimal digit 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F into 4 binary digit.

Convert $(1EBA2F)_{16}$

(1)	(E)	(B)	(2)	(F)
0001	**1110**	**1011**	**0010**	**1111**

Lecture 29

Introduction of Boolean Algebra, different laws and their use in function Boolean minimization

Rules in Boolean Algebra

- Variable used can have only two values.
- Binary 1 = HIGH and Binary 0 = LOW.
- Complement of a variable is represented by an overbar (¯)/(’). Thus if B = 0 then B’= 1 and if B = 1 then B’= 0.
- Logical ORing of the variables is represented by a plus (+) sign between them. Ex- A + B
- Logical ANDing of the two or more variable is represented by a dot between them. Ex- A.B.C. or ABC.

Boolean Laws

- There are Eight types of Boolean Laws.

1) Commutative law

Any binary operation which satisfies the following expression is referred to as commutative operation.

$$\boxed{(i)\ A.B = B.A \qquad (ii)\ A + B = B + A}$$

Commutative law states that changing the sequence of the variables does not have any effect on the output of a logic circuit.

2) Associative law

This law states that the order in which the logic operations are performed is irrelevant as their effect is the same.

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

3) **Distributive law**

Distributive law states the following condition.

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

4) **AND law**

These laws use the AND operation. Therefore they are called as AND laws.

(i) $A.0 = 0$	(ii) $A.1 = A$
(iii) $A.A = A$	(iv) $A.\overline{A} = 0$

5) **OR law**

These laws use the OR operation. Therefore they are called as OR laws.

(i) $A + 0 = A$	(ii) $A + 1 = 1$
(iii) $A + A = A$	(iv) $A + \overline{A} = 1$

6) **Inversion law**

This law uses the NOT operation. The inversion law states that double inversion of a variable results in the original variable itself.

$$\boxed{\overline{\overline{A}} = A}$$

7) **Absorption law**

This law enables a reduction in a complicated expression to a simpler one by absorbing like terms.

$$A(A+B) = A$$

$$A+AB=A$$

8) De Morgan’s Law

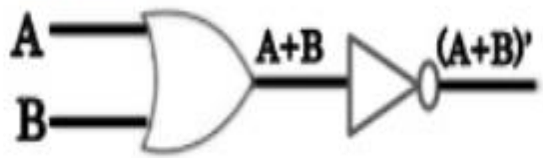
There are two “de Morgan´s” rules or theorems,

(1) Two separate terms NOR´ed together is the same as the two terms inverted (Complement) and AND´ed for example: $(A+B)' = A' . B'$

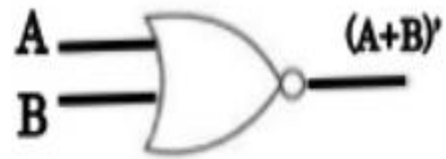
(2) Two separate terms NAND´ed together is the same as the two terms inverted (Complement) and OR´ed for example: $(A.B)' = A' + B'$

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

LHS

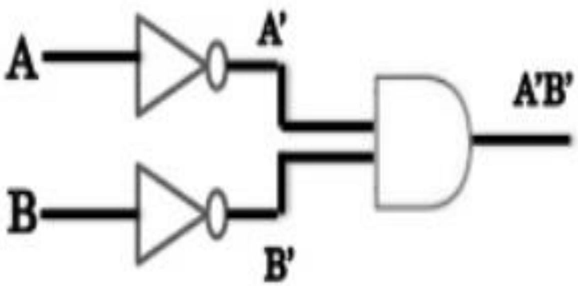


=

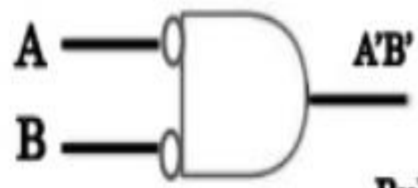


NOR gate

RHS



=



Bubbled AND gate

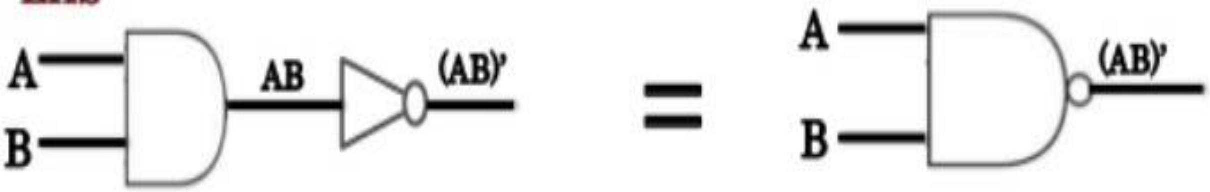
A	B	(A+B)	(A+B)'
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

=

A	B	A'	B'	A'B'
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

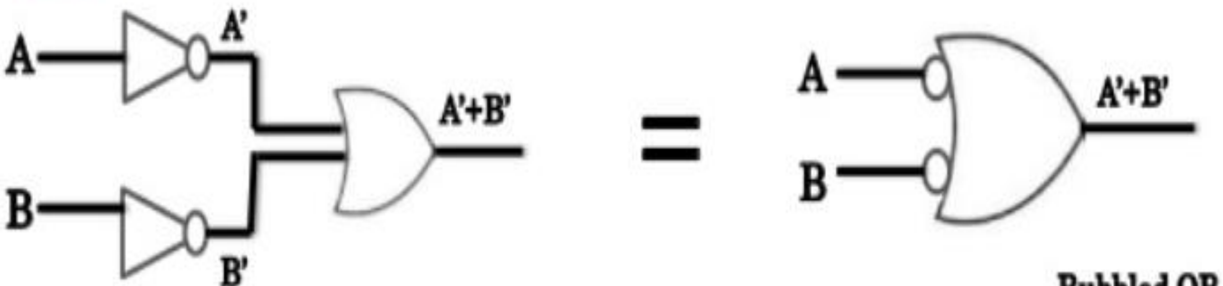
Law 2: $(AB)' = A' + B'$

LHS



NAND gate

RHS



Bubbled OR gate

A	B	AB	(AB)'
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

=

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

Examples On Boolean Laws

Example No1: Using the above laws, simplify the following expression: $(A + B)(A + C)$

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

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

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

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

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

$= A + AB + BC$

$= A(1+B) + BC$

$= A + BC$

EXAMPLE 2: C + (BC)’

- C+ (BC)’
- = C + (B’ + C’) ; De Morgan’s theorem
- = C + C’ + B’
- = 1 + B’
- = 1

EXAMPLE 3: $(AB)'(A' + B)(B' + B)$

- $(AB)'(A' + B)(B' + B)$
- $= (A' + B') (A' + B) (1)$
- $= A'A' + A'B + A'B' + B' B$
- $= A' + A'(B + B') + 0$
- $= A' + A'(1)$
- $= A' + A'$
- $= A'$

EXAMPLE 4: (A + C)(AD + AD’) + AC + C

- (A + C)(AD + AD’) + AC + C
- = (A + C) A(D + D’) + C(A + 1)
- = (A + C) A(1) + C
- = (A + C) . A + C
- = A.A+ A.C + C
- = A + AC + C
- =A(1 + C) + C
- = A + C

EXAMPLE 5: $A'(A + B) + (B + AA)(A + B')$

- $A'(A + B) + (B + AA)(A + B')$
- $=A'A + A'B + (B + A)(A + B')$
- $=0 + A'B + AB + BB' + AA + AB'$
- $= B(A' + A) + 0 + A + AB'$
- $= B(1) + A(1 + B')$
- $= B + A(1)$
- $= A + B$

EXAMPLE 6: AB + BC(B + C)

- AB + BC(B + C)
- = AB + BBC + BCC
- = AB + BC + BC
- = AB + BC ;   BC + BC = BC
- = B(A + C)

EXAMPLE 7: A + B(A + C) + AC

- A + B(A + C) + AC
- = A + AB + BC + AC
- = A(1 + B) + BC + AC
- = A + BC + AC
- = A(1 + C) + BC
- = A + BC

EXAMPLE 8: {(AB)' + C}'B

- {(AB)' + C}'B
- = {(A' + B') + C }' B
- = {A'' . B'' . C'} B
- = { A . B. C'} B
- = ABC'

Lecture 30

Introduction of Logic gates, Universal Gates, Realization of basic gates using universal gates

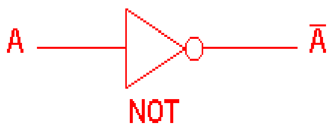
Digital Logic Gates

GROUP OF INSTITUTIONS

- Boolean functions may be practically implemented by using electronic gates.
- Electronic gates require a power supply.
- Gate **INPUTS** are driven by voltages having two nominal values, e.g. 0V and 5V representing logic 0 and logic 1 respectively.
- The **OUTPUT** of a gate provides two nominal values of voltage only, e.g. 0V and 5V representing logic 0 and logic 1 respectively. In general, there is only one output to a logic gate except in some special cases.
- There is always a time delay between an input being applied and the output responding.
- These gates are the AND, OR, NOT, NAND, NOR, EXOR and EXNOR gates.

NOT Gate

- **The NOT gate produces an inverted version of the input at its output.**
- **It is also known as an *inverter*.**
- **Symbol:**

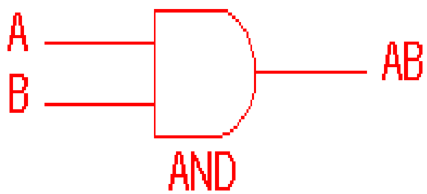


- **Truth Table:**

NOT gate	
A	$\bar{A}$
0	1
1	0

AND Gate

- The AND gate is an electronic circuit that gives a **high** output (1) only if **all** its inputs are high.
- A dot (.) is used to show the AND operation i.e. A.B. Or AB
- Symbol:



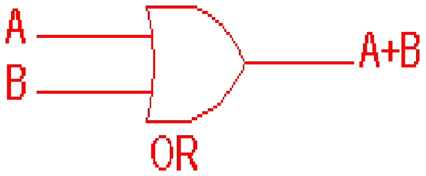
- Truth Table:

2 Input AND gate		
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate

GROUP OF INSTITUTIONS

- The OR gate is an electronic circuit that gives a high output (1) if **one or more** of its inputs are high.
- A plus (+) is used to show the OR operation.
- Symbol:



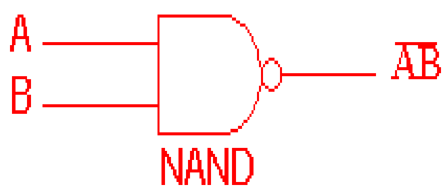
- Truth Table:

2 Input OR gate		
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

NAND Gate

GROUP OF INSTITUTIONS

- This is a NOT-AND gate
- The outputs of all NAND gates are high if **any** of the inputs are low.
- The small circle represents inversion.
- Symbol:

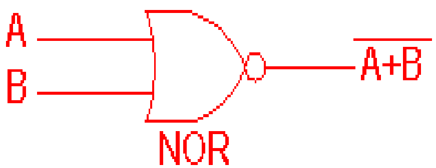


- Truth Table:

2 Input NAND gate		
A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate

- This is a NOT-OR gate
- The outputs of all NOR gates are low if **any** of the inputs are high.
- Symbol:

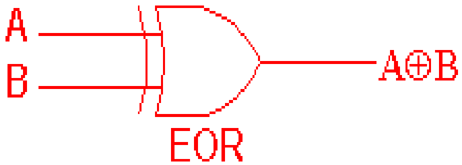


- Truth Table:

2 Input NOR gate		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

EXOR Gate

- The '**Exclusive-OR**' gate is a circuit which will give a high output if **either, but not both**, of its two inputs are high.
- An encircled plus sign ($\oplus$) is used to show the ExOR operation.
- Symbol:

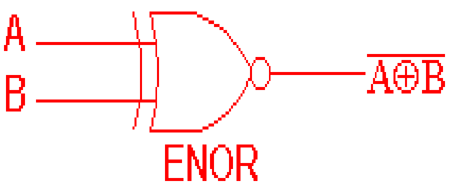


- Truth Table:

2 Input EXOR gate		
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

EX-NOR Gate

- The '**Exclusive-NOR'** gate circuit does the opposite to the EOR gate.
- It will give a low output if **either, but not both**, of its two inputs are high.
- The symbol is an EXOR gate with a small circle on the output.
- Symbol:



- Truth Table:

2 Input EXNOR gate		
A	B	$\overline{A \oplus B}$
0	0	1
0	1	0
1	0	0
1	1	1

Universal Gates

GROUP OF INSTITUTIONS

GROUP OF INSTITUTIONS

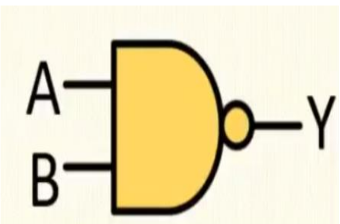
- A universal gate is a gate which can implement any Boolean function without need to use any other gate type.
- The NAND and NOR gates are universal gates.
- This is advantageous since NAND and NOR gates are economical and easier to fabricate and are the basic gates used in all IC digital logic families.

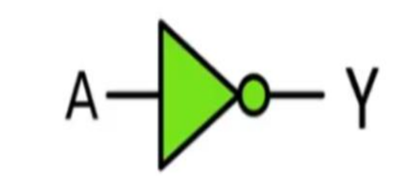
Logic Gates Using Only NAND Gates

NAND AS NOT GATE

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

$Y = \overline{A}$

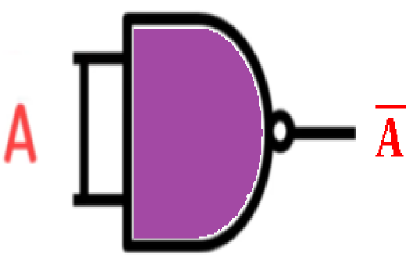




[illegible] = [illegible]

= [illegible]

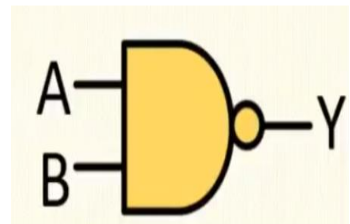
[illegible] = [illegible]

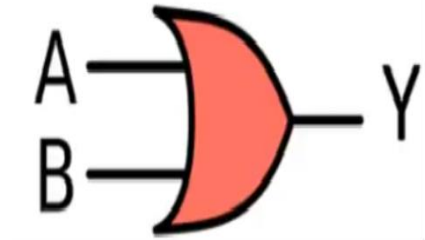


NAND AS OR GATE

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

$Y = A+B$

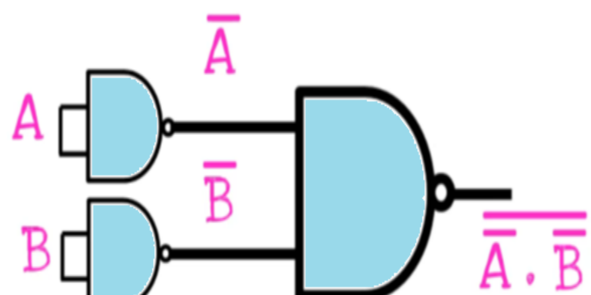




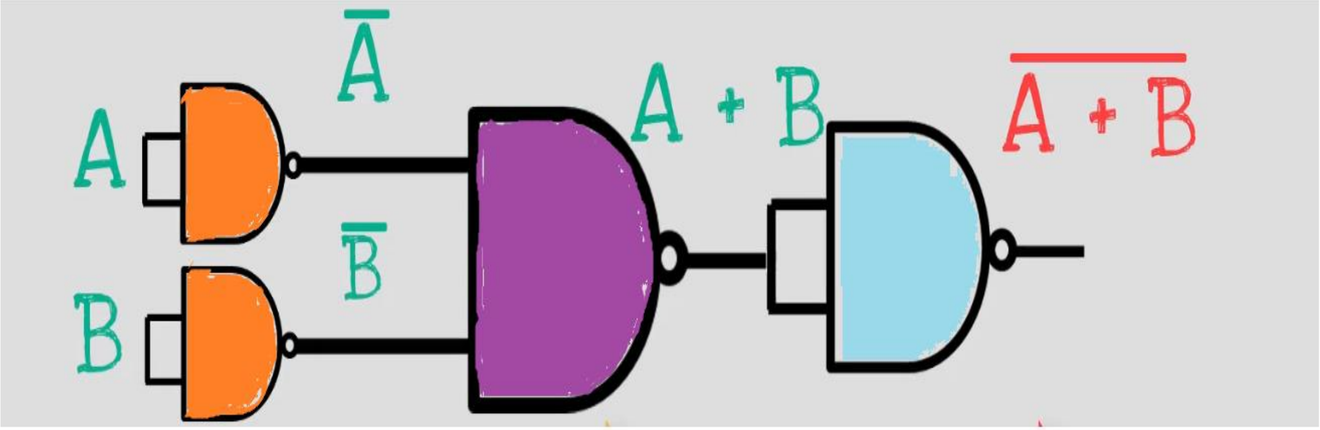
Y=A+B

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

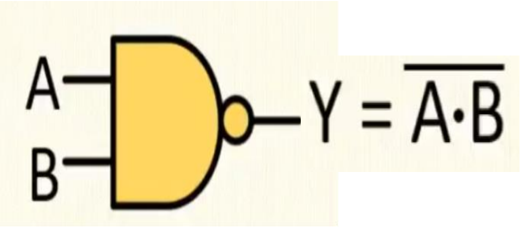
$Y = \overline{\overline{A}.\overline{B}}$

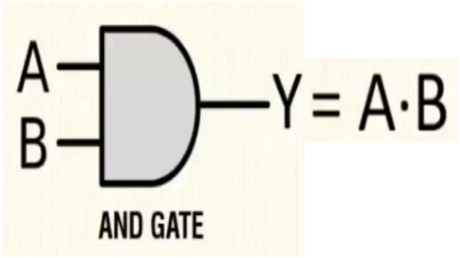


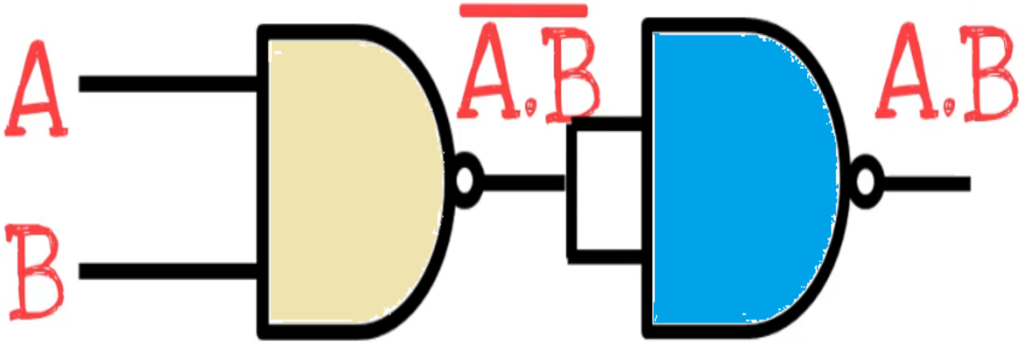
NAND AS NOR



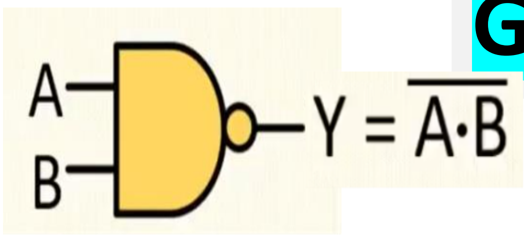
NAND AS AND GATE

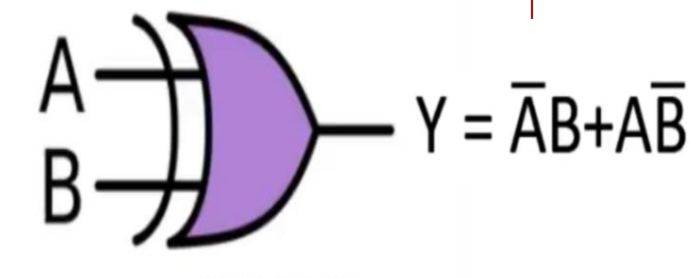






NAND AS XOR GATE





XOR GATE

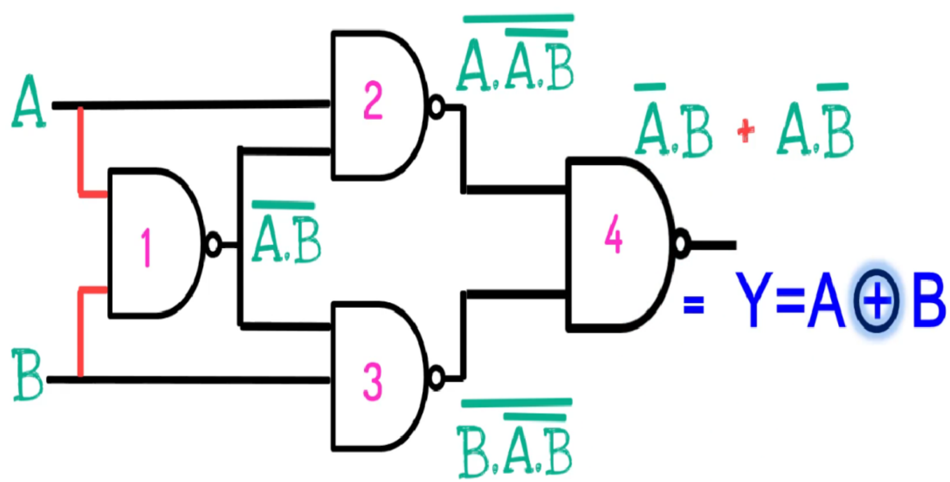
$$\bar{A}.B + A.\bar{B}$$

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

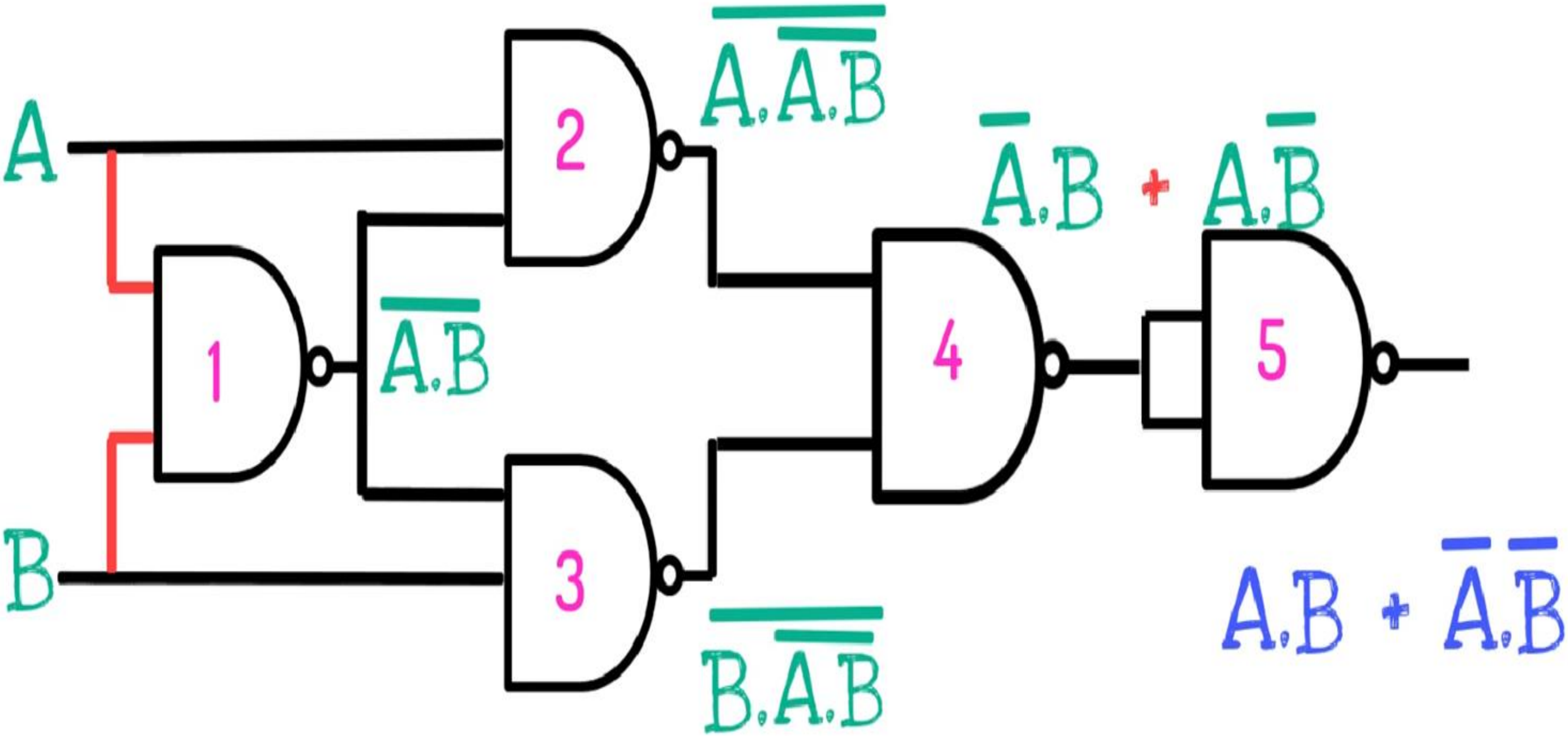
$$= A.(\bar{A}+\bar{B}) + B.(\bar{A}+\bar{B})$$

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

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



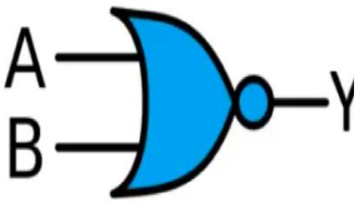
NAND AS XNOR GATE

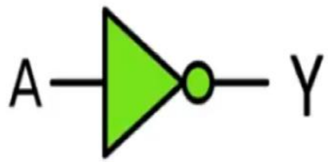


Logic Gates Using Only NOR Gates

NOR AS NOT GATE

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

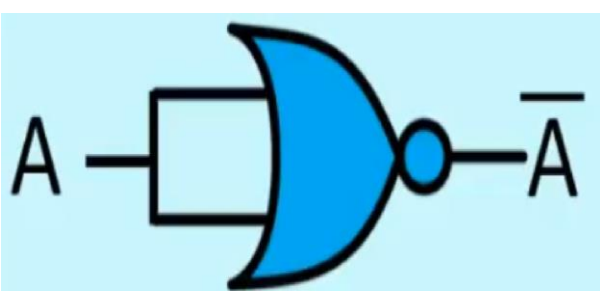




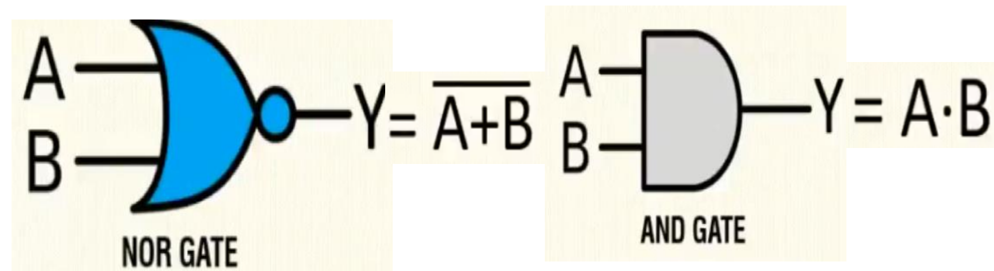
$$Y = \overline{A+B}$$

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

$$Y = \overline{A}$$

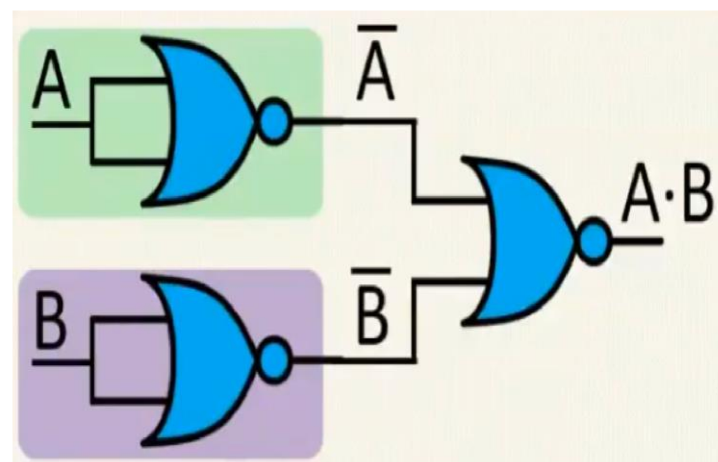


NOR AS AND GATE

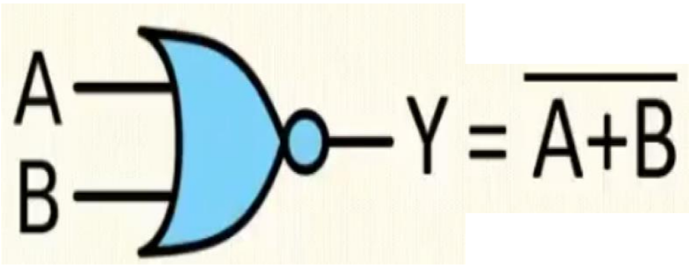


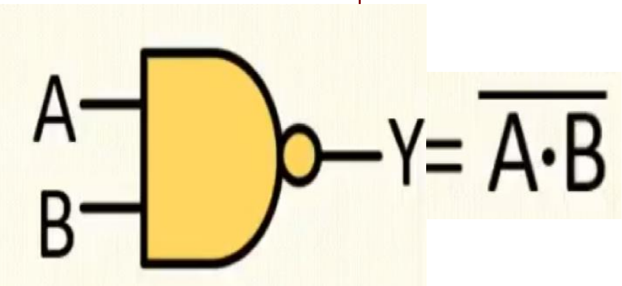
$$Y = A \cdot B = \overline{\overline{A \cdot B}}$$

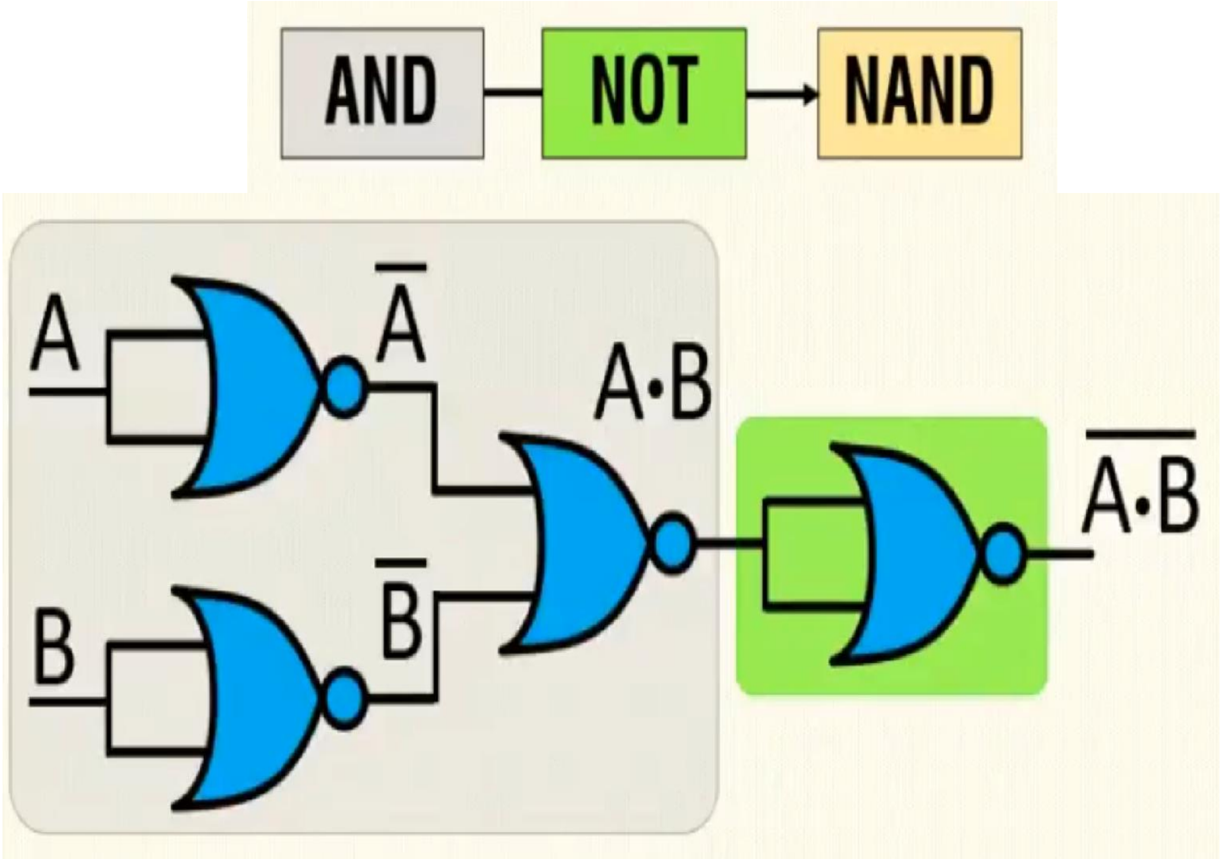
$$Y = \overline{\overline{A}+\overline{B}}$$



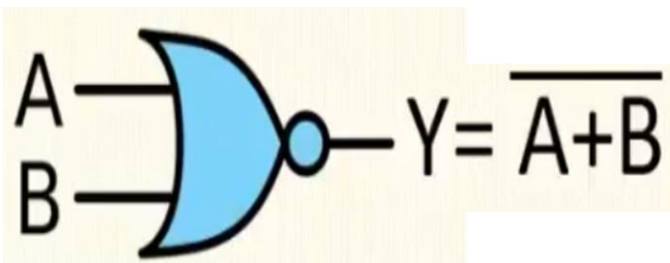
NOR AS NAND GATE

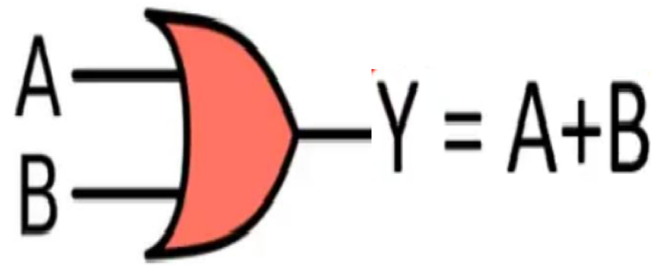


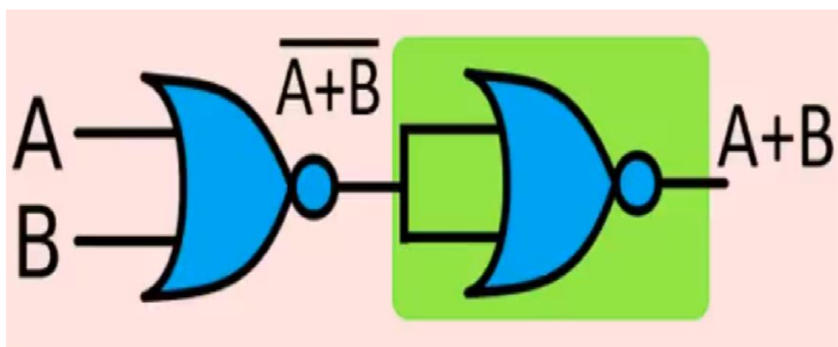




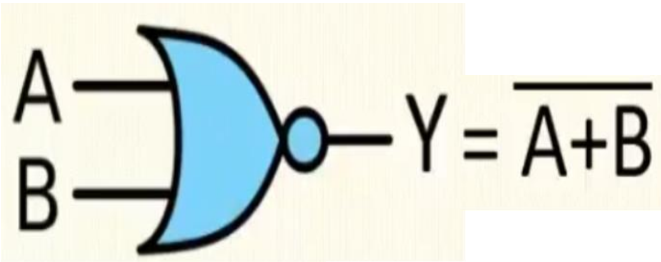
NOR AS OR GATE



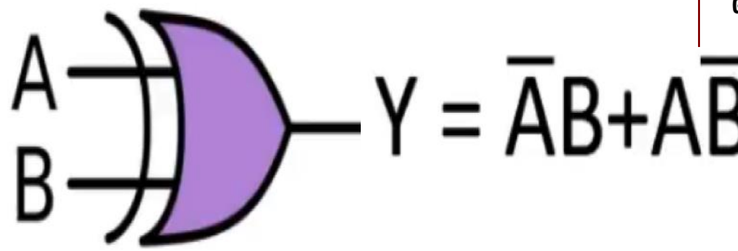




NOR AS XOR GATE



NOR GATE



XOR GATE

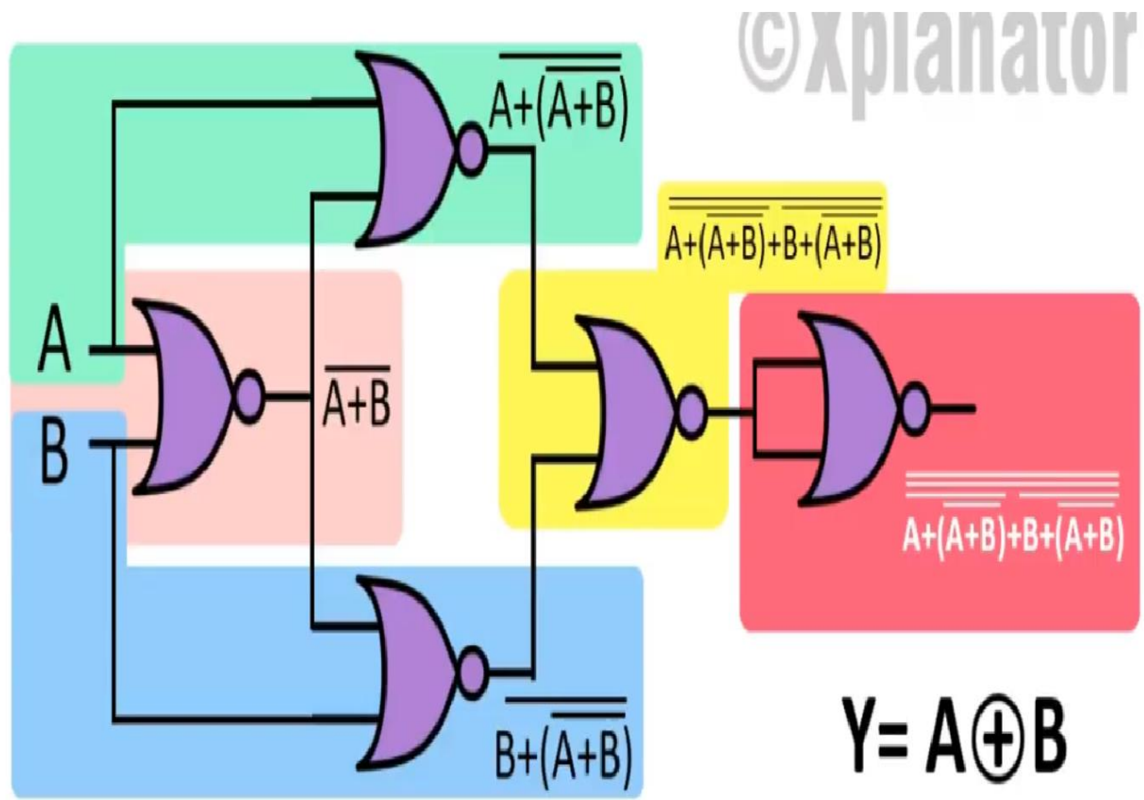
$$Y = \bar{A}B + A\bar{B}$$

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

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

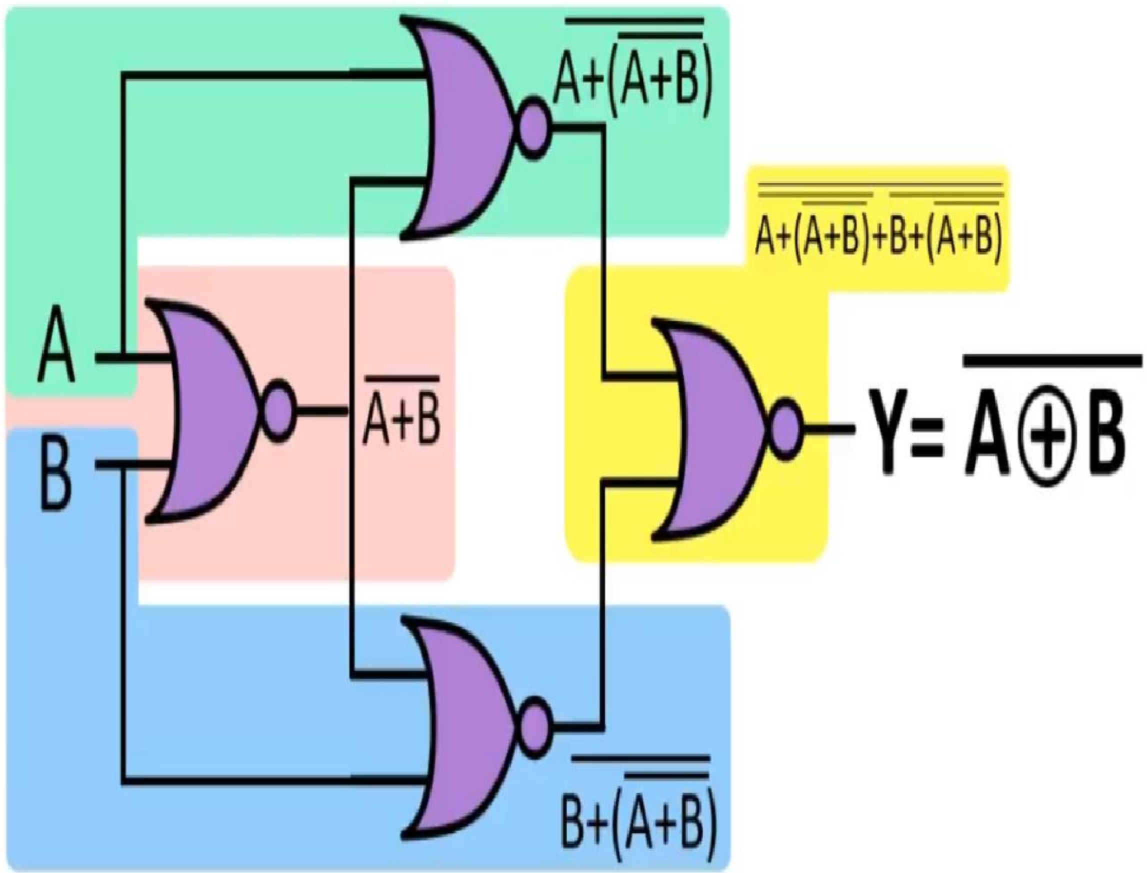
$$Y = \overline{\overline{\overline{A}} + \overline{(A+B)}} + \overline{\overline{\overline{B}} + \overline{(A+B)}}$$

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



NOR AS XNOR GATE

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



Lecture 31

SOP and POS and Canonical form representation

Boolean Function Representation

- Various way of representing a given function

1- Sum of Product Form (SOP)

2- Product of Sum Form (POS)

3- Standard or Canonical SOP Form

4- Standard or Canonical POS Form

5- Truth Table Form

	Standard or Canonical SOP Form
• The Sum of Products is abbreviated as SOP. • It is the logical expression in Boolean algebra where all the input terms are ANDed (Product) first and then ORed (summed) together. • SOP form: F(A,B,C)=A+BC'+A'BC • The variables in each term are not necessarily all the variables of the function.	• Standard SOP term must contain all the function variables either in complemented form or in uncomplemented form. • A product term which contain all the function variables either in complemented form or in uncomplemented form is called a minterm. F(A,B,C)=AB’C+A’BC'+A’BC F(A,B,C)=∑m (2,3,5)

	Standard or Canonical POS Form
• POS form means that the inputs of each term are Added together using OR function then all terms are multiplied together using AND function. • The variables in each term are not necessarily all the variables of the function. • POS form: F(A,B,C)=A.(B+C').(A'+B+C')	• Standard POS term must contain all the function variables either in complemented form or in uncomplemented form. • A sum term which contain all the function variables either in complemented form or in uncomplemented form is called a maxterm. • F(A,B,C)=(A+B'+C')(A+B+C) (A'+B+C) F(A,B,C)= ΠM(0,3,4)

Minterms & Maxterms for 2 variables

- Two variable minterms and maxterms.

x	y	Index	Minterm	Maxterm
0	0	0	$m_0 = \bar{x}\,\bar{y}$	$M_0 = x + y$
0	1	1	$m_1 = \bar{x}\,y$	$M_1 = x + \bar{y}$
1	0	2	$m_2 = x\,\bar{y}$	$M_2 = \bar{x} + y$
1	1	3	$m_3 = x\,y$	$M_3 = \bar{x} + \bar{y}$

- The minterm m_i should evaluate to 1 for each combination of x and y.
- The maxterm is the complement of the minterm

Table: Minterms & Maxterms for three variables

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

Conversion of SOP to Canonical SOP

GROUP OF INSTITUTIONS

F(A,B,C)=A+BC'+A'BC

=A+BC'+A'BC

=A(B+B')(C+C')+BC'(A+A')+A'BC

=ABC+ABC'+AB'C+AB'C'+

ABC'+A'BC'+A'BC

=ABC+ABC'+AB'C+AB'C'+ A'BC'+A'BC

(A+A=A)

Conversion of POS to Canonical POS

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

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

$(A.A=A)$

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

Sum of Product Form (SOP)	Product of Sum Form (POS)
• A way of representing Boolean expressions as sum of product terms.	• A way of representing Boolean expressions as product of sum terms.
• SOP uses minterms. Minterm is product of Boolean variables either in normal form or complemented form.	• POS uses maxterms. Maxterm is sum of Boolean variables either in normal form or complemented form.
• It is sum of minterms. Minterms are represented as ‘m’	• It is product of maxterms. Maxterms are represented as ‘M’
• SOP is formed by considering all the minterms, whose output is HIGH(1)	• POS is formed by considering all the maxterms, whose output is LOW(0)

- While writing minterms for SOP, input with value 1 is considered as the variable itself and input with value 0 is considered as complement of the input.
  Example :
  If variable A is Low(0) – A’
  A is High(1) – A

- SOP form Examples:
  F(A,B,C)=A+BC'+A’BC
  F(A,B,C)=AB’C+A’BC'+A’BC
  F(A,B,C)=∑ m (2,3,5)

- While writing maxterms for POS, input with value 1 is considered as the complement and input with value 0 is considered as the variable itself.
  Example :
  If variable A is Low(0) - A
  A is High(1) - A’

- POS form Examples:
  F(A,B,C)=A.(B+C').(A'+B+C’)

F(A,B,C)=(A+B’+C’)(A+B+C)(A’+B +C)

F(A,B,C)= ΠM(0,3,4)

Example 1 – Express the Boolean function $F = A + B'C$ as standard sum of minterms.

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

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

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

$F = A + B'C = ABC + ABC' + AB'C + AB'C' + AB'C + A'B'C$

$F = A'B'C + AB'C' + AB'C + ABC' + ABC$

$= m1 + m4 + m5 + m6 + m7$

$= \sum m(1,4,5,6,7)$

Example 2 – Express the Boolean function F = (A+B')(B+C) as a product of max-terms

- $F = (A+B')(B+C)$
- I term: $(A+B') = (A+B'+CC')$

  $= (A+B'+C)\ (A+B'+C')$
- II term: $(B+C) = (AA'+B+C)$

  $= (A+B+C)\ (A'+B+C)$
- Combining both:
- $F = (A+B'+C)\ (A+B'+C')\ (A+B+C)\ (A'+B+C)$

  $= M2 * M3 * M0 * M4$

  $= \Pi M(0,2,3,4)$

Example 3 – Express the Boolean function F = xy + x’z as a product of maxterms.

- 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’)
  = M0*M2*M4*M5

  = πM(0,2,4,5)

Example 4–Convert F(A, B, C) = ∑m(1,4,5,6,7) to POS FORM

- Missing terms of minterms = terms of maxterms
- Missing terms of maxterms = terms of minterms
- $F(A, B, C) = \sum m(1,4,5,6,7) = \pi M(0,2,3)$

Example 5– Convert Boolean expression in standard form F=y'+xz'+xyz

- $F=y'+xz'+xyz$
- $F = (x+x')y'(z+z')+x(y+y')z' +xyz$
- $F = xy'z+ xy'z'+x'y'z+x'y'z'+ xyz'+xy'z'+xyz$
- F = m5, m4, m1, m0, m6, m4, m7
- $F= \sum m\ (0,1,4,5,6,7)$

Truth Table Form

Use of truth table to show all the possible combinations of input conditions that will produces an output 1 in case of SOP expression and 0 in case of POS.

Example : Generate truth table for F= xy + x'z.

where	
0	Maxterms
1	Minterms

INPUTS			OUTPUT
X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Lecture 32

Introduction of K Map: 2&3 Variable

KARNAUGH MAP (K-Map)

- Simplification of logic expression using Boolean algebra is awkward because:
  - it lacks specific rules to predict the most suitable next step in the simplification process
  - it is difficult to determine whether the simplest form has been achieved.
- A Karnaugh map is a graphical method used to obtained the most simplified form of an expression in a standard form (Sum-of-Products or Product-of-Sums).
- The simplest form of an expression is the one that has the minimum number of terms with the least number of literals (variables) in each term.
- By simplifying an expression to the one that uses the minimum number of terms, we ensure that the function will be implemented with the minimum number of gates.
- By simplifying an expression to the one that uses the least number of literals for each terms, we ensure that the function will be implemented with gates that have the minimum number of inputs.

Steps to solve expression using K-map

1. Select K-map according to the number of variables.
2. Identify minterm or maxterms as given in problem.
3. For SOP put 1's in blocks of K-map respective to the minterms.
4. For POS put 0's in blocks of K-map respective to the maxterms.
5. Make rectangular groups containing total terms in power of two like 2,4,8 ..(except 1) and try to cover as many elements as you can in one group.
6. From the groups made in step 5 find the product terms and sum them up for SOP form.

Two Variable K-Map

GROUP OF INSTITUTIONS

A two-variable function has four possible minterms. We can re-arrange these minterms into a Karnaugh map.

x	y	minterm
0	0	$x'y'$
0	1	$x'y$
1	0	xy'
1	1	xy

X \ Y	0	1
0	$x'y'$	$x'y$
1	xy'	xy

Now we can easily see which minterms contain common literals.

– Minterms on the left and right sides contain y' and y respectively.

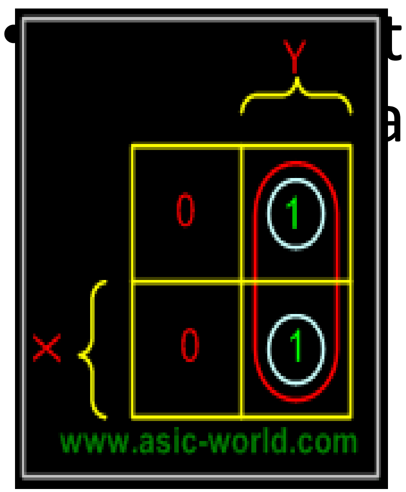
– Minterms in the top and bottom rows contain x' and x respectively.

X \ Y	0	1
0	$x'y'$	$x'y$
1	xy'	xy

	Y'	Y
X'	$x'y'$	$x'y$
X	xy'	xy

Example - X'Y+XY

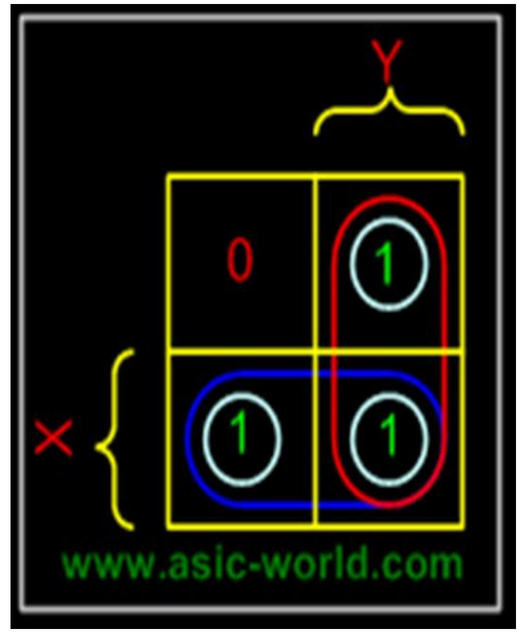
- we have the equation for two inputs X and Y
- Draw the k-map for function F with marking 1 for X'Y and XY position
- Now combine two 1's as shown in figure to form the single term
- t canceled and ains.



So F = Y

Example - X'Y+XY+XY

- Draw the k-map for function F
- mark 1 for X'Y, XY and XY position



So, F = X + Y

Example- f(A,B) = $\sum m(0,1,3)$

A \ B	0	1
0	1	1
1		1

- So, f= A’ + B

Example- f(A,B) = $\sum m(0,1,3,4)$

A \ B	0	1
0	1	1
1	1	1

- So, f= 1

Three Variable K-Map

- The number of cells in 3 variable K-map is eight, since the number of variables is three.
- The following figure shows 3 variable K-Map.

X \ YZ	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

- There is only one possibility of grouping 8 adjacent min terms.
- The possible combinations of grouping 4 adjacent min terms are {(m_0, m_1, m_3, m_2), (m_4, m_5, m_7, m_6), (m_0, m_1, m_4, m_5), (m_1, m_3, m_5, m_7), (m_3, m_2, m_7, m_6) and (m_2, m_0, m_6, m_4)}.
- The possible combinations of grouping 2 adjacent min terms are {(m_0, m_1), (m_1, m_3), (m_3, m_2), (m_2, m_0), (m_4, m_5), (m_5, m_7), (m_7, m_6), (m_6, m_4), (m_0, m_4), (m_1, m_5), (m_3, m_7) and (m_2, m_6)}.

Representation using minterms (SOP form):

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

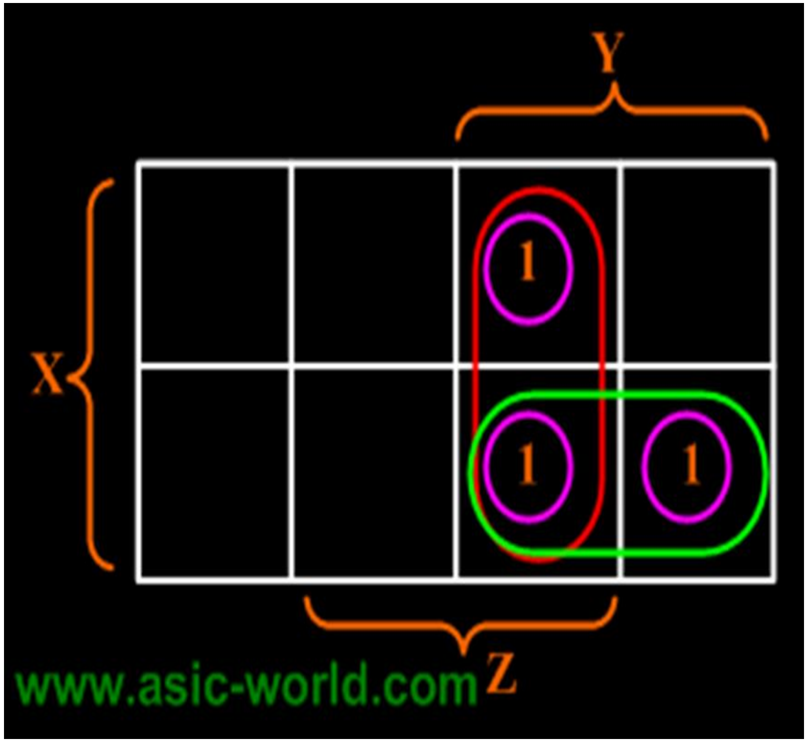
A \ BC	00	01	11	10
0	m_0 0	m_1 1	m_3 3	m_2 2
1	m_4 4	m_5 5	m_7 7	m_6 6

Representation using maxterms (POS form):

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

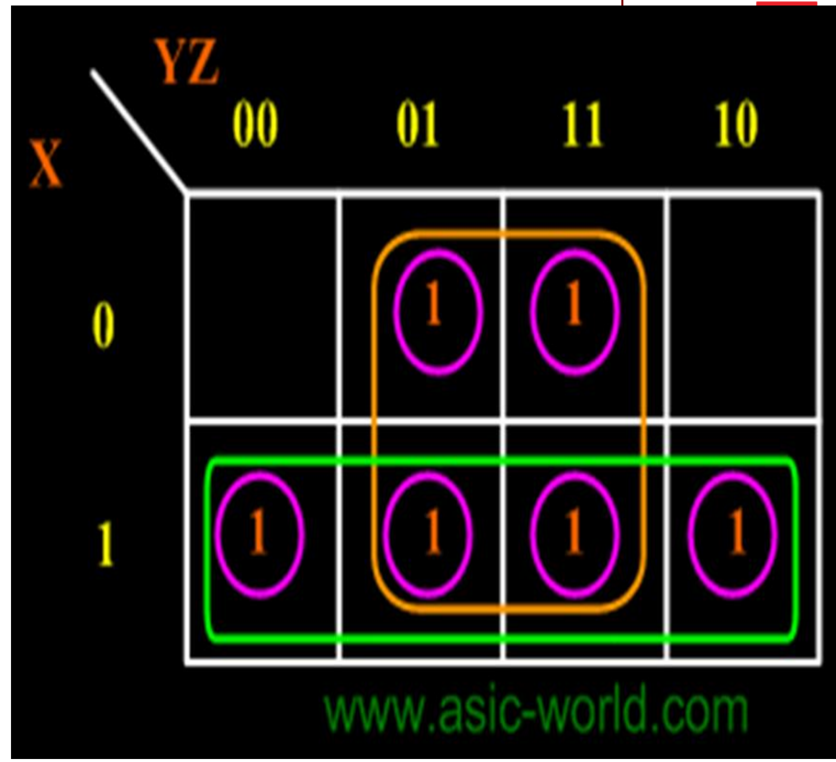
A \ BC	00	01	11	10
0	M_0 0	M_1 1	M_3 3	M_2 2
1	M_4 4	M_5 5	M_7 7	M_6 6

Example- F = XYZ'+XYZ+X'YZ



- So, F = XY + YZ

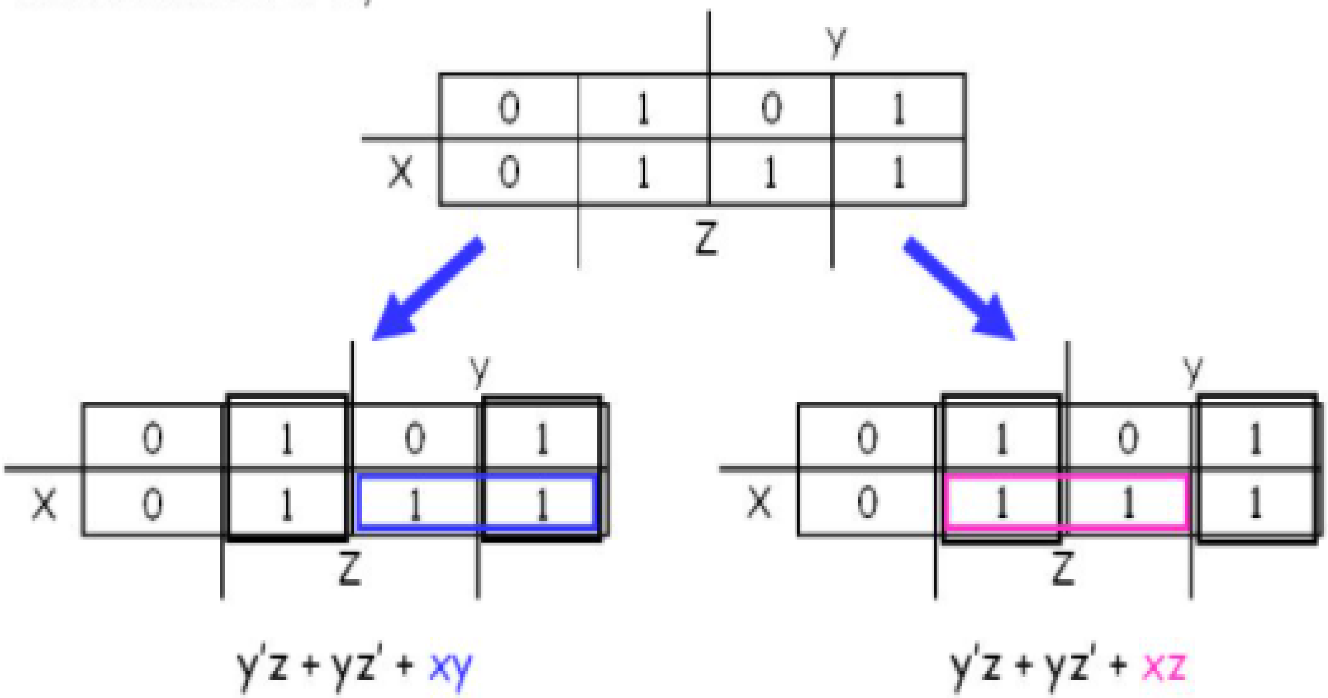
Example- F(X,Y,Z) = (1,3,4,5,6,7)



- So, F = X + Z

K-MAP CAN BE TRICKY

There may not necessarily be a *unique* MSP. The K-map below yields two valid and equivalent MSPs, because there are two possible ways to include minterm m_7



Remember that overlapping groups is possible, as shown above

Examples of K-Map

$f = \sum(0,4) = \overline{B}\,\overline{C}$

A \ BC	00	01	11	10
0	1	0	0	0
1	1	0	0	0

$f = \sum(4,5) = A\overline{B}$

A \ BC	00	01	11	10
0	0	0	0	0
1	1	1	0	0

$f = \sum(0,1,4,5) = \overline{B}$

A \ BC	00	01	11	10
0	1	1	0	0
1	1	1	0	0

$f = \sum(0,1,2,3) = \overline{A}$

A \ BC	00	01	11	10
0	1	1	1	1
1	0	0	0	0

$f = \sum(0,4) = \overline{A}C$

A \ BC	00	01	11	10
0	0	1	1	0
1	0	0	0	0

$f = \sum(4,6) = A\overline{C}$

A \ BC	00	01	11	10
0	0	0	0	0
1	1	0	0	1

$f = \sum(0,2) = \overline{A}\,\overline{C}$

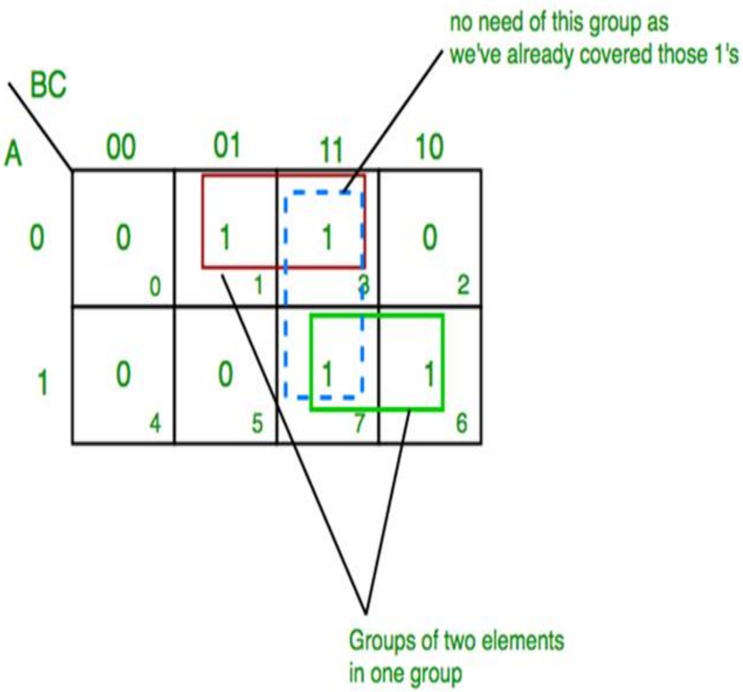
A \ BC	00	01	11	10
0	1	0	0	1
1	0	0	0	0

$f = \sum(0,2,4,6) = \overline{C}$

A \ BC	00	01	11	10
0	1	0	0	1
1	1	0	0	1

Example:1

Z= $\sum(1,3,6,7)$



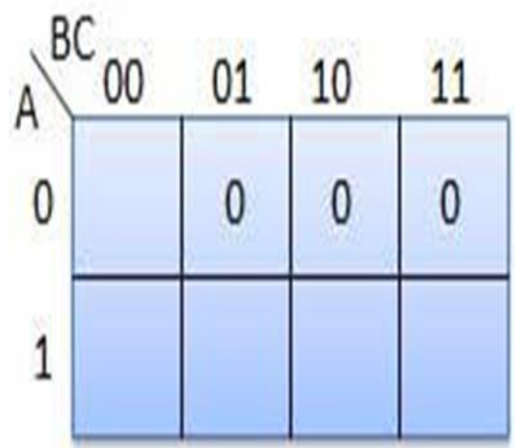
Z(A,B,C)=AB+$\bar{A}C$

Example:2

F=π(1,2,3)

In POS form $(B+\bar{C})(\bar{A}+B)(\bar{B}+\bar{C})$

$$\downarrow \ \downarrow \ \downarrow \ \downarrow \ \downarrow \ \downarrow$$
$$0 \ 1 \ 1 \ 1 \ 1 \ 0$$



$\prod m(1,3,2)$

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

Dear Students write down the answer of following questions in your subject notebook.:

Q.1 Using the Karnaugh map method, simplify the following functions, obtain their sum of the products form, and product of the sums form. Realize them with basic gates.

(*a*) $F(X, Y, Z) = \Sigma(1, 3, 4, 5, 6, 7)$

(*b*) $F(X, Y, Z) = \Sigma(1, 5, 6, 7)$

Lecture 34

K Map: 5 & 6 Variable K map and Problems on Kmap

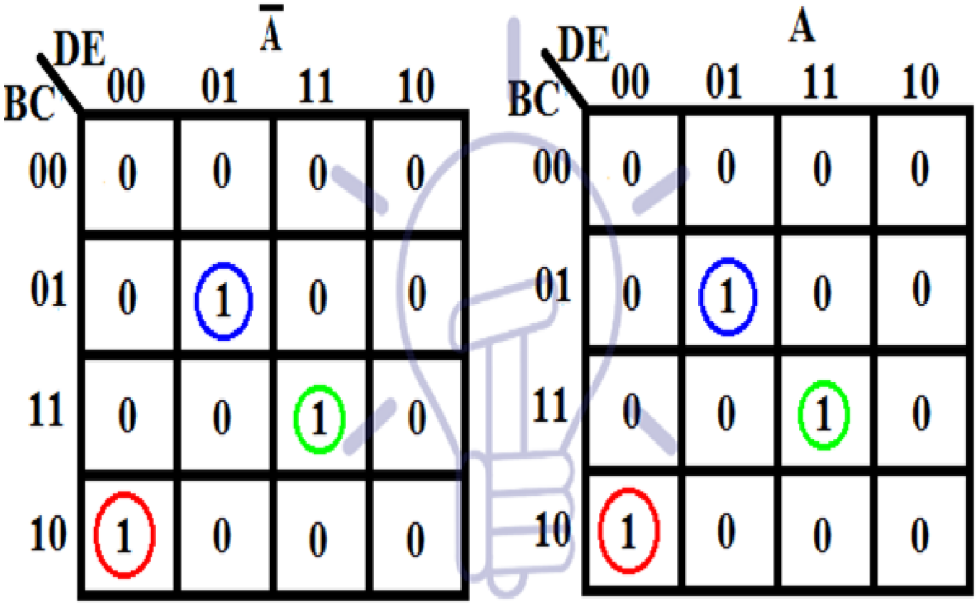
5 Variable K-Map

$\bar{A}$

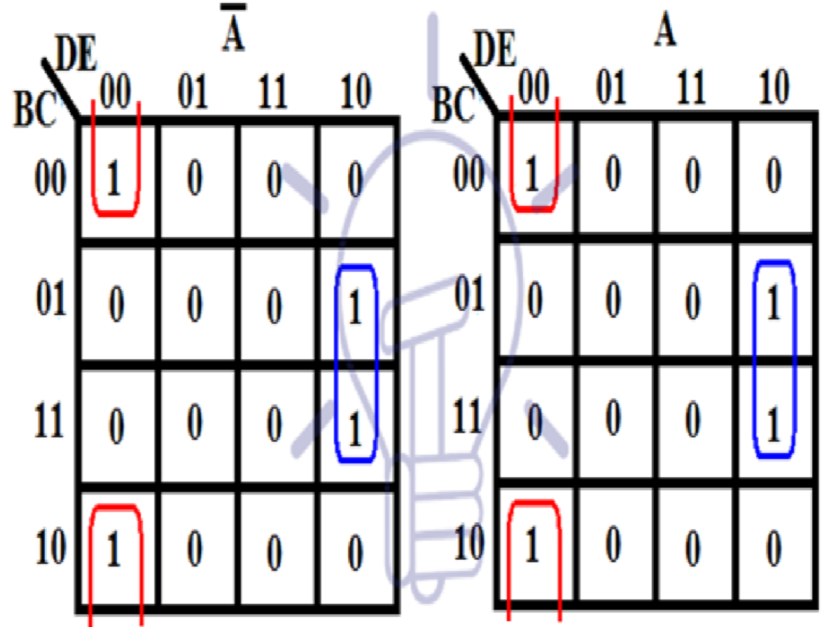
BC \ DE	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

A

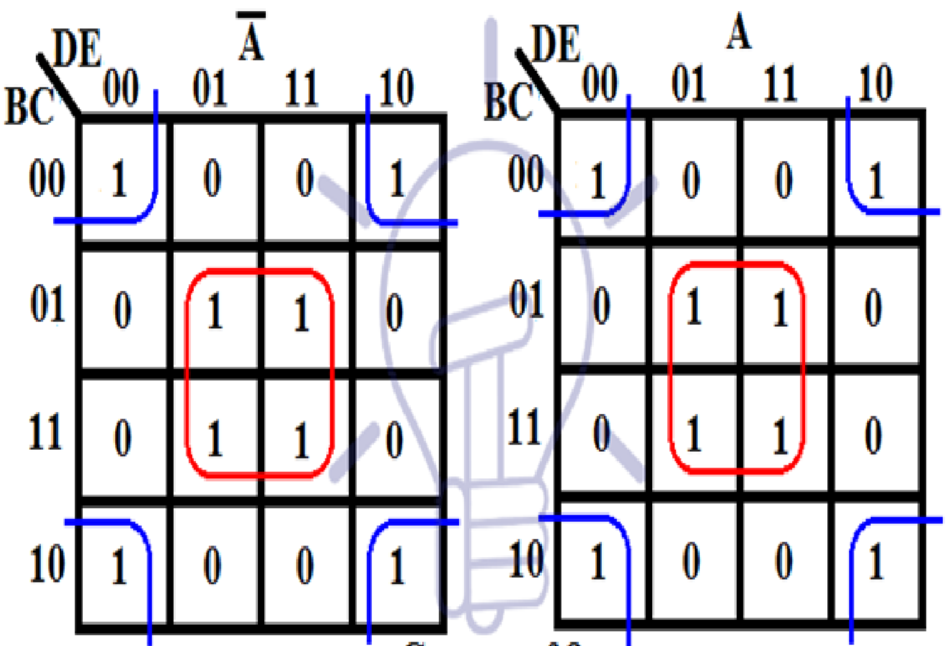
BC \ DE	00	01	11	10
00	m_{16}	m_{17}	m_{19}	m_{18}
01	m_{20}	m_{21}	m_{23}	m_{22}
11	m_{28}	m_{29}	m_{31}	m_{30}
10	m_{24}	m_{25}	m_{27}	m_{26}



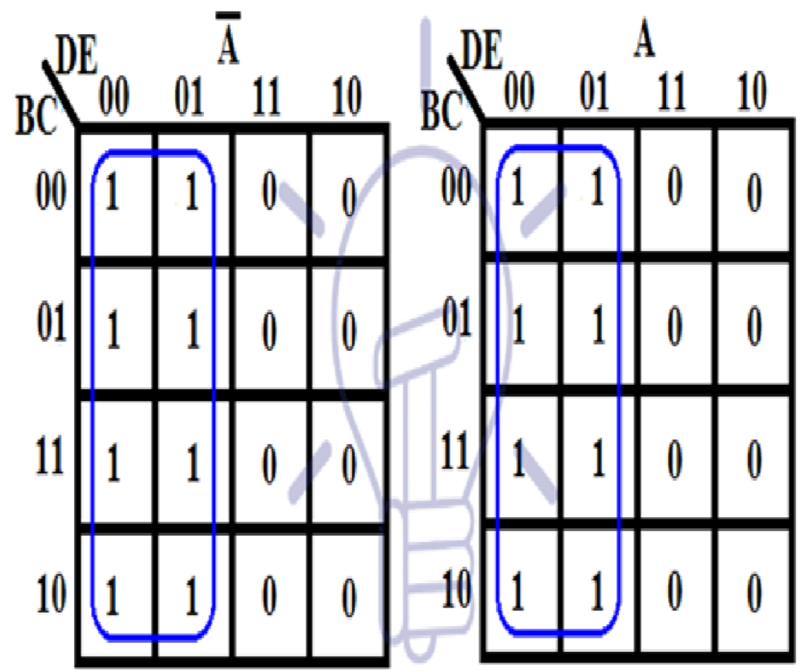
Groups of 2



Groups of 4



Groups of 8



Group of 16

Example -1

$F(A,B,C,D,E) = \sum (m_0, m_2, m_5, m_7, m_8, m_{10}, m_{16}, m_{21}, m_{23}, m_{24}, m_{27}, m_{31})$

$\overline{A}$

BC \ DE	00	01	11	10
00	1	0	0	1
01	0	1	1	0
11	0	0	0	0
10	1	0	0	1

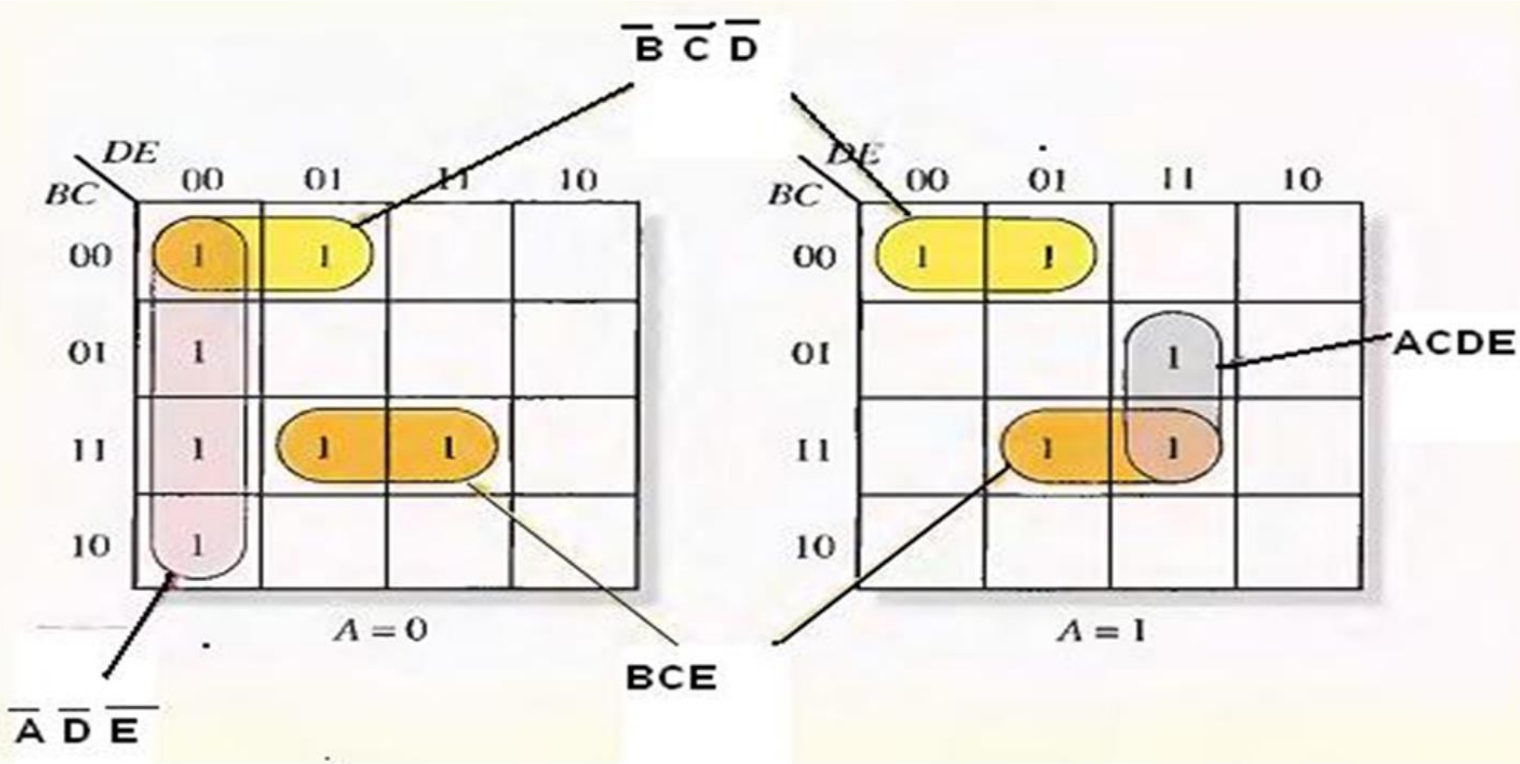
A

BC \ DE	00	01	11	10
00	1	0	0	0
01	0	1	1	0
11	0	0	1	0
10	1	0	1	0

$$F = \overline{A}\,\overline{C}\,\overline{E} + \overline{C}\,\overline{D}\,\overline{E} + \overline{B}CE + ABDE$$

Example -2

$$X = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\,\overline{E} + \overline{A}\,\overline{B}C\overline{D}\,\overline{E} + \overline{A}BC\overline{D}\,\overline{E} + \overline{A}B\overline{C}\,\overline{D}\,\overline{E} + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}E + \overline{A}BC\overline{D}E$$
$$+ \overline{A}BCDE + A\overline{B}\,\overline{C}\,\overline{D}\,\overline{E} + A\overline{B}\,\overline{C}\,\overline{D}E + ABC\overline{D}E + ABCDE + A\overline{B}CDE$$



$$X = \overline{A}\,\overline{D}\,\overline{E} + \overline{B}\,\overline{C}\,\overline{D} + BCE + ACDE$$

6 Variable K-Map

GROUP OF INSTITUTIONS

- 6-variable k-map is a complex k-map which can be drawn. Visualizing 6-variable k-map is a little bit tricky.
- 6 variables make 64 min terms, this means that the k-map of 6 variables will have 64 cells. Its geometry becomes difficult to draw as these cells are adjacent to each other in all direction in 3-dimensions i.e. a cell is adjacent to upper, lower, left, right, front and back cells at the same time. It will be draw like 5 variable k-map.

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

CD \ EF	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

$\bar{A}$, B

CD \ EF	00	01	11	10
00	m_{16}	m_{17}	m_{19}	m_{18}
01	m_{20}	m_{21}	m_{23}	m_{22}
11	m_{28}	m_{29}	m_{31}	m_{30}
10	m_{24}	m_{25}	m_{27}	m_{26}

A, $\bar{B}$

CD \ EF	00	01	11	10
00	m_{32}	m_{33}	m_{35}	m_{34}
01	m_{36}	m_{37}	m_{39}	m_{38}
11	m_{44}	m_{45}	m_{47}	m_{46}
10	m_{40}	m_{41}	m_{43}	m_{42}

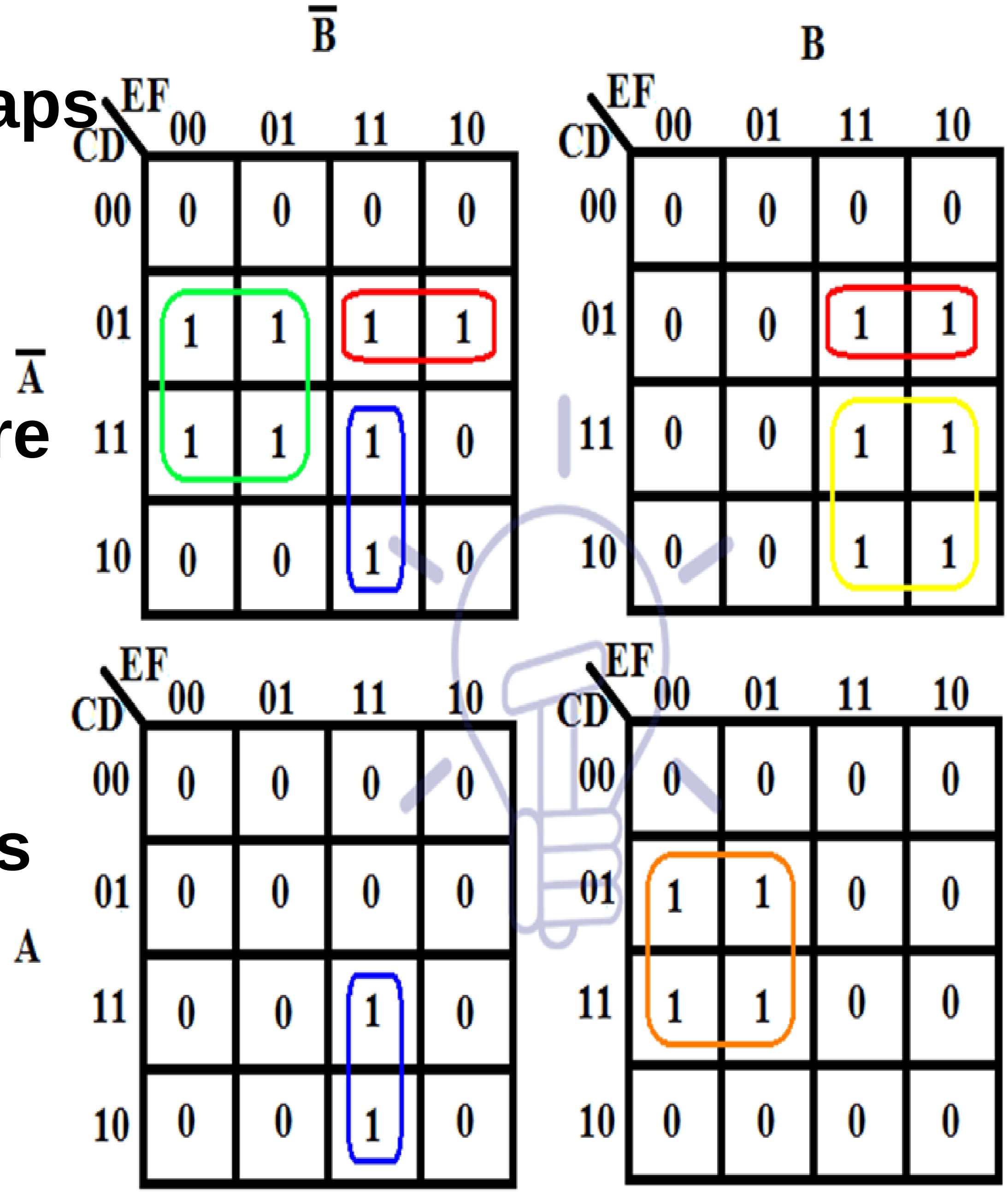
A, B

CD \ EF	00	01	11	10
00	m_{48}	m_{49}	m_{51}	m_{50}
01	m_{52}	m_{53}	m_{55}	m_{54}
11	m_{60}	m_{61}	m_{63}	m_{62}
10	m_{56}	m_{57}	m_{59}	m_{58}

- Visualize these k-maps on top of each other.

- In this example, there are 5 groups of 4 min-terms.

- Notice the min-terms in the diagonal K-maps, they make a separate group because these



Example -1

$F = \sum (m_0, m_2, m_8, m_9, m_{10}, m_{12}, m_{13}, m_{16}, m_{18}, m_{24}, m_{25}, m_{26}, m_{29}, m_{31}, m_{32}, m_{34}, m_{35}, m_{39}, m_{40}, m_{42}, m_{43}, m_{47}, m_{48}, m_{50}, m_{56}, m_{58}, m_{61}, m_{63})$

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

CD \ EF	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	1	1	0	0
10	1	1	0	1

$\bar{A}$, B:

CD \ EF	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	0	1	1	0
10	1	1	0	1

A, $\bar{B}$:

CD \ EF	00	01	11	10
00	1	0	1	1
01	0	0	1	0
11	0	0	1	0
10	1	0	1	1

A, B:

CD \ EF	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	0	1	1	0
10	1	0	0	1

$F = = \overline{DF} + \bar{A}C\bar{E}F + \overline{AB}C\bar{E} + BCDF + A\bar{B}EF$

- The 6-variable k-map is made from 4-variable 4 k-maps. As you can see variable A on the left side select 2 k-maps row-wise between these 4 k-maps. A = 0 for the upper two K-maps and A = 1 for the lower two K-maps. Variable B on top of these K-maps select 2 k-maps column-wise. B = 0 for left 2 K-maps and B = 1 for right 2 K-maps.
- Imagine these 4-variable K-maps as a single square, these k-maps are adjacent to each other horizontally and vertically but not diagonally because these cells have 1-bit difference. The groups between these k-maps should be made as done in 5-variable K-map but you cannot make groups between diagonal k-maps.

Example:5

Obtain minimal product of the sums for the function

$$F\,(W,X,Y,Z) = \pi(0, 1, 4, 5, 6, 8, 9, 12, 13, 14).$$

	$y+z$	$y+\bar{z}$	$\bar{y}+\bar{z}$	$\bar{y}+z$
$w+x$	0	0	1	1
$w+\bar{x}$	0	0	1	0
$\bar{w}+\bar{x}$	0	0	1	0
$\bar{w}+x$	0	0	1	1

$$F = y(\bar{x}+z)$$

Example:6

Obtain minimal product of the sums for the function

$$F\,(W, X, Y, Z) = \sum(0,\ 1,\ 2,\ 5,\ 8,\ 9,\ 10).$$

	Y′Z′	Y′Z	YZ	YZ′
W′X′	1	1	0	1
W′X	0	1	0	0
WX	0	0	0	0
WX′	1	1	0	1

$$F = (\bar{X} + Z)\,(\bar{W} + \bar{X})(\bar{Y} + \bar{Z}).$$

Example:7

$$F=WX'Y' + WY + W'YZ'$$

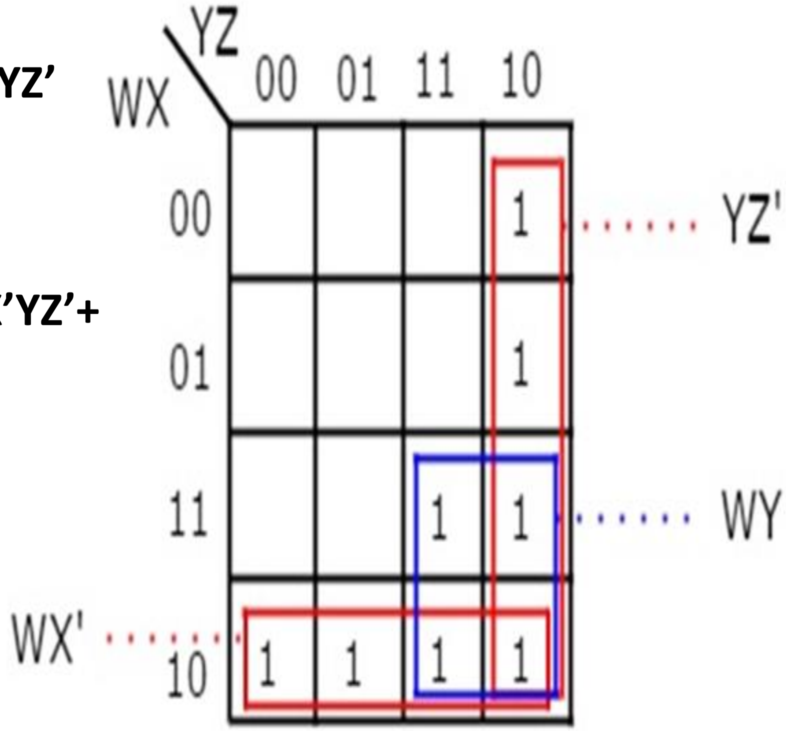
$F=WX'Y' + WY + W'YZ'$

$=WX'Y'(Z+Z')+WY(X+X')+W'YZ'(X+X')$

$= WX'Y'Z+ WX'Y'Z'+ \color{red}{WXY+ WX'Y}\color{black}{+ W'XYZ' +W'X'YZ'}$

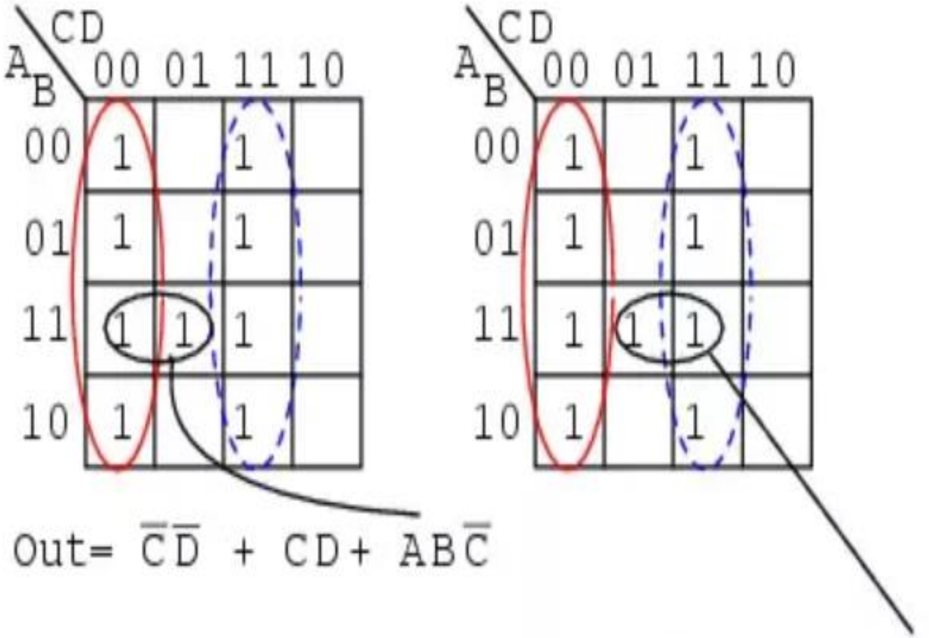
$= WX'Y'Z+ WX'Y'Z'+ WXY(Z+Z') + WX'Y(Z+Z') + W'XYZ' +W'X'YZ'$

$= WX'Y'Z+ WX'Y'Z'+ WXYZ+ WXYZ'+ WX'YZ+ WX'YZ'+ W'XYZ' +W'X'YZ'$



Example 8:

$$Out = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + \bar{A}BCD + AB\bar{C}\bar{D} + AB\bar{C}D + ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD$$



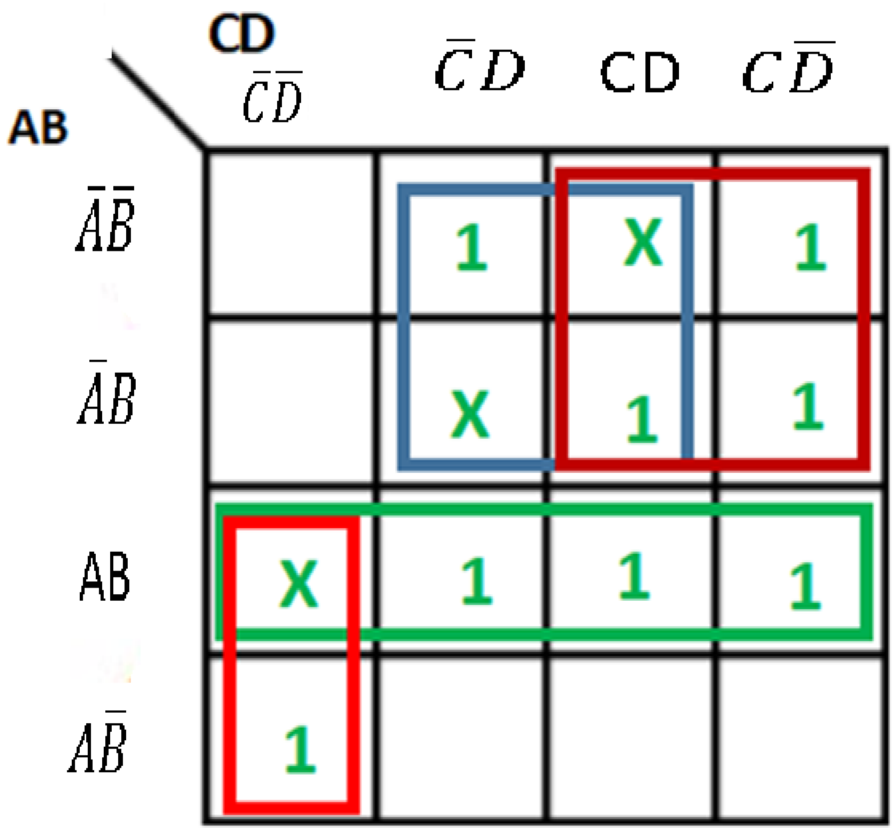
$$Out = \bar{C}\bar{D} + CD + AB\bar{C}$$

$$Out = \bar{C}\bar{D} + CD + ABD$$

Here there are two possible solutions. Both are correct. But any one answer with minimal cost is considered. It is your choice.

Example-9

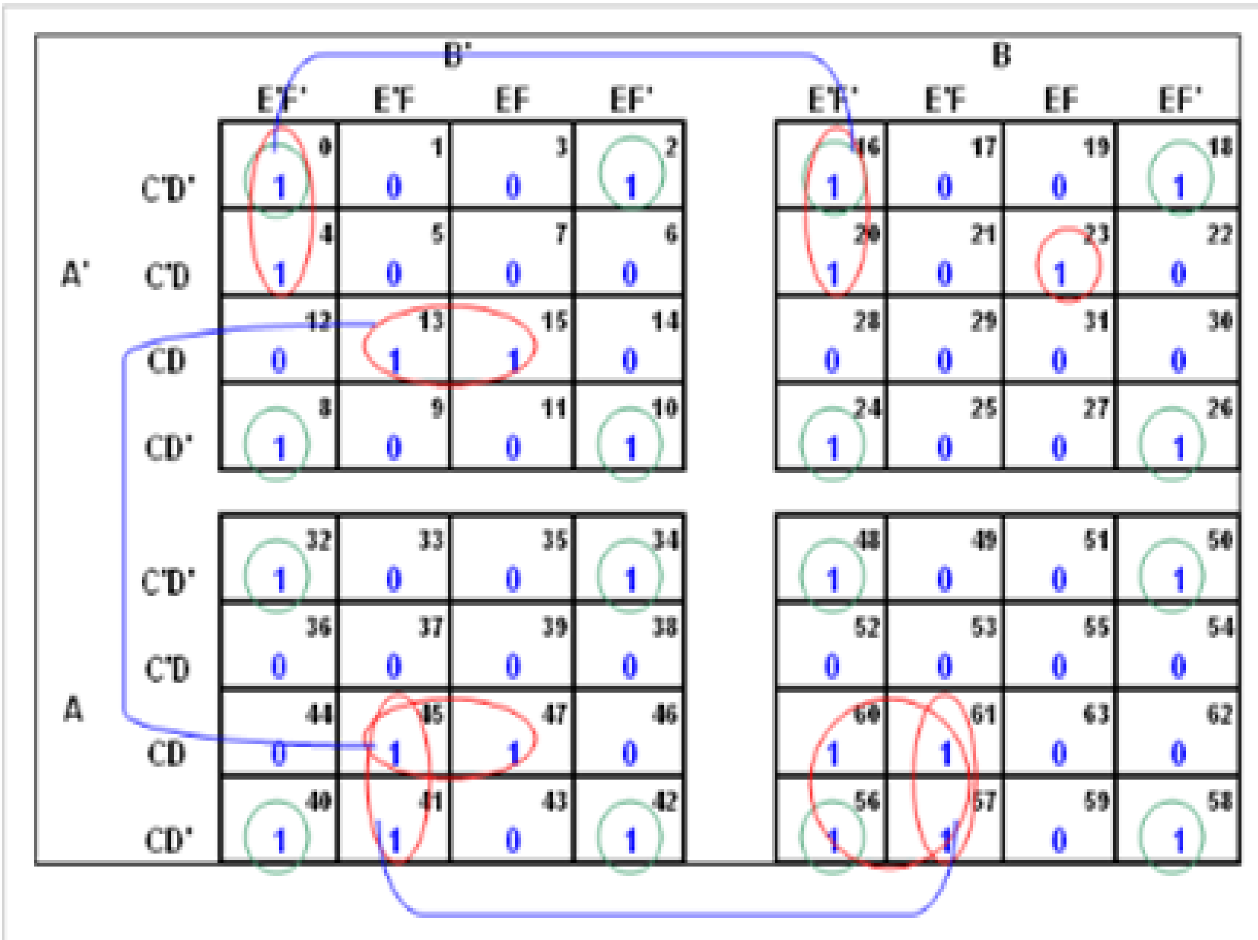
Minimise the following function in SOP minimal form using K-Maps: F(A, B, C, D) = m(1, 2, 6, 7, 8, 13, 14, 15) + d(3, 5, 12)



F=AC'D' + A'D + A'C + AB

Example -10

$F = \Sigma(0, 2, 4, 8, 10, 13, 15, 16, 18, 20, 23, 24, 26, 32, 34, 40, 41, 42, 45, 47, 48, 50, 56, 57, 58, 60, 61)$



$F = D'F' + ACE'F + B'CDF + A'C'E'F' + ABCE' + A'BC'DEF$

Example -11

$F = \Sigma(0, 1, 2, 3, 4, 5, 8, 9, 12, 13, 16, 17, 18, 19, 24, 25, 36, 37, 38, 39, 52, 53, 60, 61)$

		B'				B			
		E'F'	E'F	EF	EF'	E'F'	E'F	EF	EF'
A'	C'D'	0: 1	1: 1	3: 1	2: 1	16: 1	17: 1	19: 1	18: 1
	C'D	4: 1	5: 1	7: 0	6: 0	20: 0	21: 0	23: 0	22: 0
	CD	12: 1	13: 1	15: 0	14: 0	28: 0	29: 0	31: 0	30: 0
	CD'	8: 1	9: 1	11: 0	10: 0	24: 1	25: 1	27: 0	26: 0
A	C'D'	32: 0	33: 0	35: 0	34: 0	48: 0	49: 0	51: 0	50: 0
	C'D	36: 1	37: 1	39: 1	38: 1	52: 1	53: 1	55: 0	54: 0
	CD	44: 0	45: 0	47: 0	46: 0	60: 1	61: 1	63: 0	62: 0
	CD'	40: 0	41: 0	43: 0	42: 0	56: 0	57: 0	59: 0	58: 0