MODULE 2

BOOLEAN ALGEBRA

POSTULATES OF BOOLEAN ALGEBRA

❖BOOLEAN ALGEBRA

- There are 2 types of algebras existed. One is ordinary algebra and other is Boolean algebra.
- Logical operations are implemented only in Boolean algebra
- In Boolean algebra , $A+A = A$ and $A.A=A$ because, the variable A has only a logical value. It doesn’t have any numerical significance.
- In ordinary algebra, $A+A=2A$ and $A.A=A^2$, because variable A has a numerical value here.
- Boolean algebra constants are 0 and 1
- Truth table is used for verification

In Boolean algebra 1+1 = 1

In binary number system 1+1 = 10

In ordinary algebra 1+1 = 2

- There is nothing like subtraction or division in Boolean algebra
- There is no negative or fractional numbers in Boolean Algebra
- Any functional relations in Boolean algebra can be proved by the method of "Perfect Induction"

❖POSTULATES

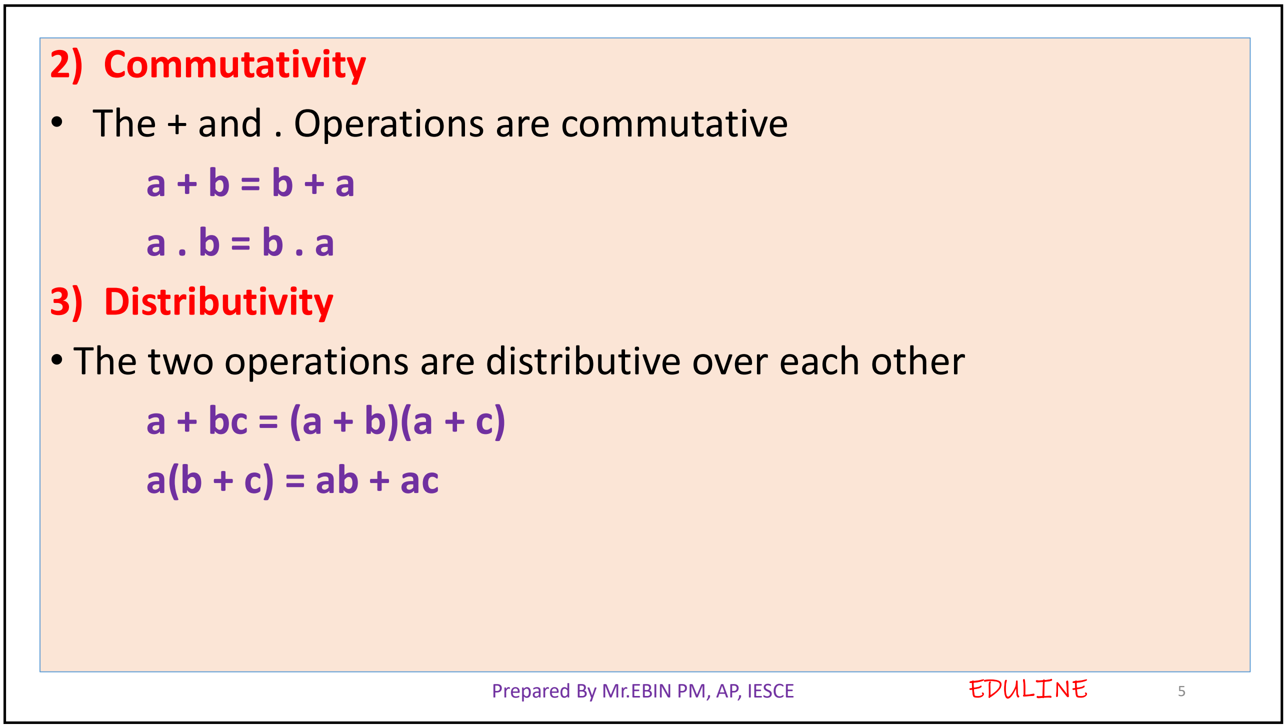
➢A set **B** of elements (a, b, c, …..) with an equivalence relation (**=**), two binary operations(**+** and **.**) and unary operation (**complement**) is a Boolean algebra if and only if the following postulates are satisfied.

1) Associativity

- The + and . operations are associative

(a + b) + c = a + (b + c)

(a . b) . c = a . (b . c)



2) Commutativity

- The + and . Operations are commutative

$a + b = b + a$

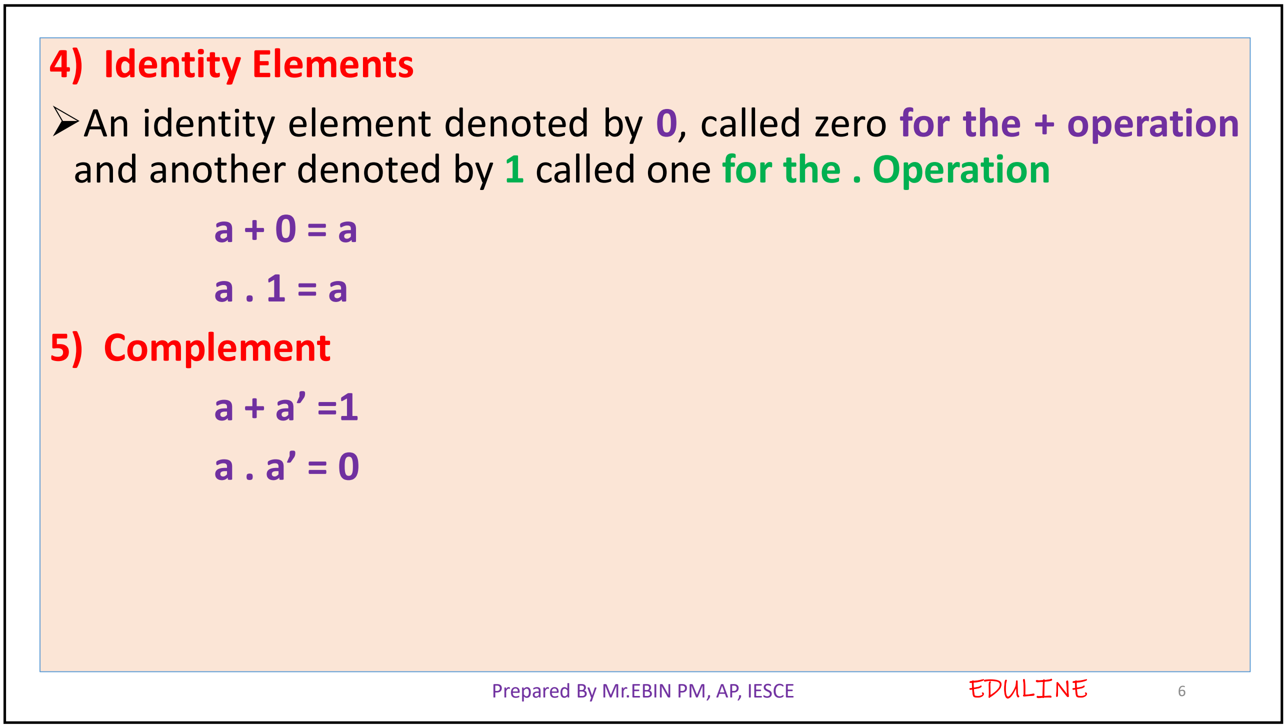
$a . b = b . a$

3) Distributivity

- The two operations are distributive over each other

$a + bc = (a + b)(a + c)$

$a(b + c) = ab + ac$



4) Identity Elements

➢ An identity element denoted by **0**, called zero **for the + operation** and another denoted by **1** called one **for the . Operation**

$a + 0 = a$

$a . 1 = a$

5) Complement

$a + a' = 1$

$a . a' = 0$

FUNDAMENTAL THEOREMS

1. Closure of identity elements - For all a Є B,

a + 1 = 1

Proof

a + 1 = (a +1) . 1 (by identity element)
= (a+1) . (a + a’) (by complement)
= a (a + a’) +1 (a + a’) (by distributive)
=a + 0 + a + a’
=a + a’ (by identity element)
=1 (by complement)

a . 0 = 0

Proof

a . 0 = a .0 + 0 (by identity element)
= a.0 + a . a’ (by complement)
= a (0 + a’) (by distributive)
= a . a’ (by identity element)
= 0 (by complement)

---

2. Equality Theorem – for all a, b, c Є B if

a + b = a + c

ab = ac

Then **b = c**

Proof

b = b. 1
= b (a + a’) [∵ a + a’ = 1]
= ab + a’b
= ac + a’b [∵ ab = ac]
= ac + a’b + 0
= ac + a’b + aa’ [∵ aa’=0]
= ac + a’(b + a)
= ac + a’(a + c) [∵ a + b = a + c]
= ac + a’a + a’c
= ac + a’c
= c (a + a’) [∵ a + a’ = 1]
= c

∴ b = c

3. Complementarity Theorem – for all a, b Є B,if

$a + b = 1$ and

$ab = 0$, Then

$a' = b$

$a = b'$

Proof

$a + b = 1$

$a + b = a + a'$ $[\because a + a' = 1]$

$ab = 0$

$ab = aa'$ $[\because aa' = 0]$

∴ by equality theorem

$a + b = a + a'$

$ab = aa'$

∴ $b = a'$

and , $b + a = b + b'$ $[\because b + b' = 1]$

$ba = bb'$ $[\because bb' = 0]$

∴ by equality theorem

$a = b'$

4. Identity elements 0 and 1 are complement to each other

Proof

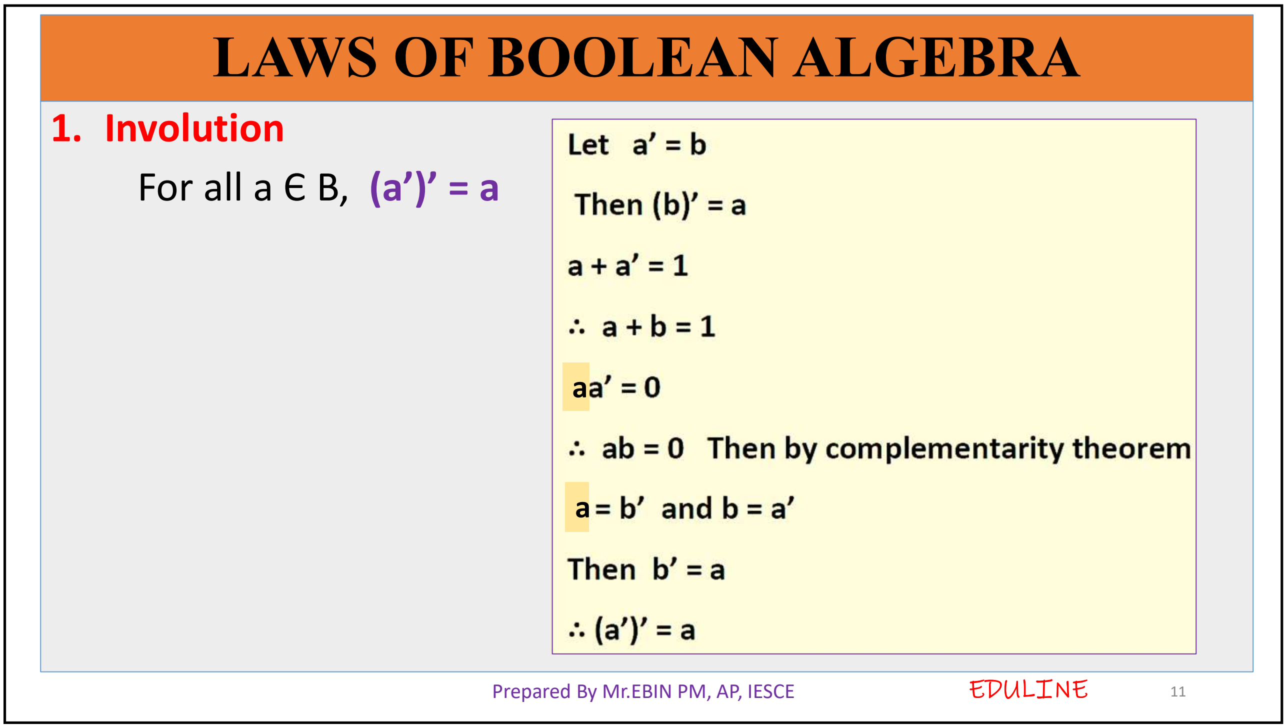
Since 0 and 1 Є B

$1 + 0 = 1$ and $1 . 0 = 0$

Then by complementarity theorem

$0' = 1$

$1' = 0$



LAWS OF BOOLEAN ALGEBRA

1. Involution

For all a ∈ B, $(a')' = a$

Let $a' = b$

Then $(b)' = a$

$a + a' = 1$

$\therefore\ a + b = 1$

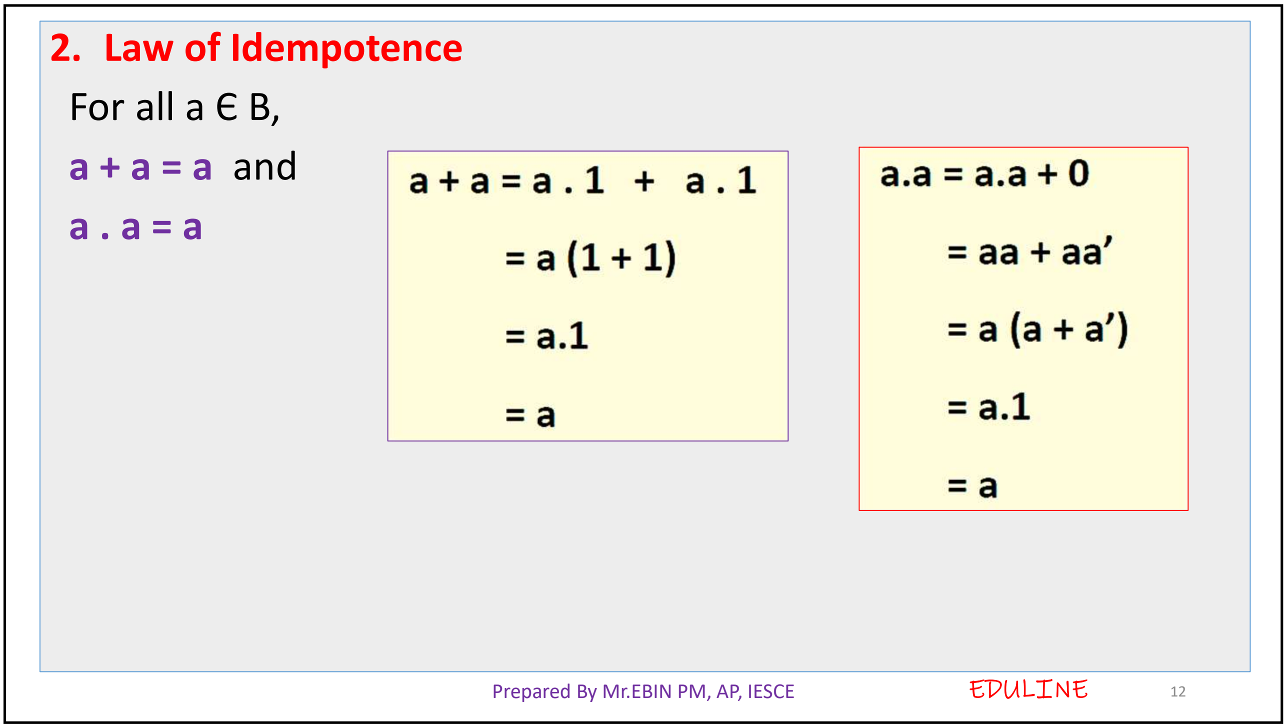
$aa' = 0$

$\therefore\ ab = 0$ Then by complementarity theorem

$a = b'$ and $b = a'$

Then $b' = a$

$\therefore (a')' = a$



2. Law of Idempotence

For all a ∈ B,

$a + a = a$ and

$a \cdot a = a$

$$\begin{aligned} a + a &= a \cdot 1 + a \cdot 1 \\ &= a\,(1 + 1) \\ &= a.1 \\ &= a \end{aligned}$$

$$\begin{aligned} a.a &= a.a + 0 \\ &= aa + aa' \\ &= a\,(a + a') \\ &= a.1 \\ &= a \end{aligned}$$

3. Law of Absorption

For all a, b Є B

a + ab = a

a(a + b) = a

Proof

a + ab = a.1 + ab
= a(1 + b)
= a.1
= a

a(a + b) = aa + ab
= a + ab
= a

DEMORGAN’S THEOREM

(a + b)’ = a’ . b’

(a . b)’ = a’ + b’

Proof

(a + b) + a’b’ = (a + a’b’) + b
= (a + a’) (a + b’) + b [Distributivity]
= (a + b’) + b
= a + (b’ + b)
= a + 1 [complement]
=1

(a + b)(a’b’) = a(a’b’) + b(a’b’)
= (aa’)b’ + a’(bb’)
= 0.b’ + a’.0
= 0

∴ (a + b) + a’b’ = 1
(a + b) (a’b’) = 0
∴ (a + b)’ = a’b’ [complementarity theorem]

Eg: Simplify the following using Boolean theorems

1. $A+A'B$

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

$=1.\ (A+B)$

$=A+B$

2. $(A+B)\ (A'+C)$

$(A+B)\ (A'+C) = AA'+AC+BA'+BC$

$= 0+AC+BA'+BC$

$= AC+BA'+BC$

BOOLEAN FUNCTIONS

❖**Literal –** A Boolean variable in the true form or in the complemented form is called a literal.

Eg: a, a’, b, b’ are literals

- The Boolean product of two or more literals is called a product term.
- The Boolean sum of two or more literals is called a sum term.

❖**Normal Form**

There are 2 types of normal forms

1. **Sum of Product (SOP)**
2. **Product of Sum (POS)**

- **Sum of Product** – SOP form is also called Disjunctive Normal Form (DNF). It is in the form

  $f(abcd) = a' + bc' + cd$

- **Product of Sum** – POS form is also called Conjunctive Normal Form (CNF). It is in the form

  $f(abcd) = (a + b)(b + c + d)$

- A Boolean function that is neither in the DNF nor in CNF

Eg1: $f1 = a'b' + b(c + d')$. Convert the function in normal form?

$f1 = a'b' + bc + bd'$

Eg2: $f2 = (a' + b)(a + cd)$. Convert the function in normal form?

$f1 = (a' + b)(a + d)(a + c)$

CANONICAL FORM

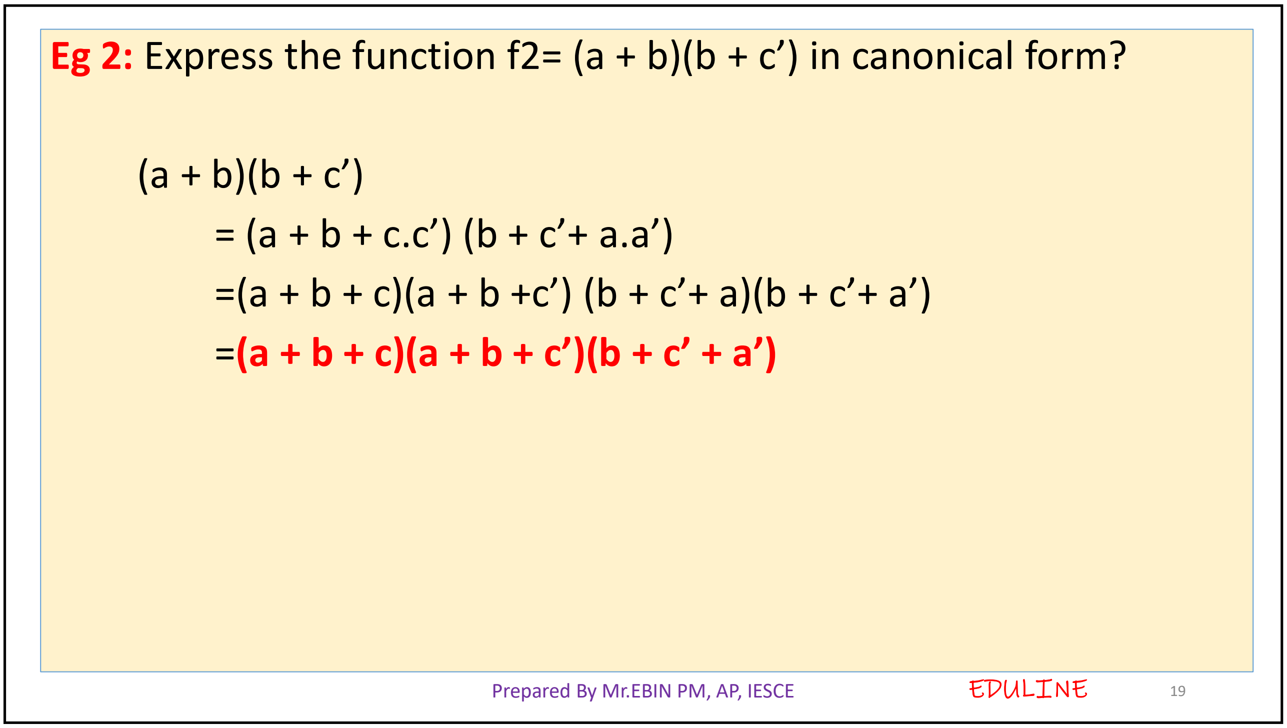
- When each of the terms of a Boolean function expressed either in SOP or POS form has all the variables in it, it is said to be expressed in canonical form.
- Canonical form cannot have the same term more than once.
- Canonical SOP is called Disjunctive Canonical Form (DCF)
- Canonical POS is called Conjunctive Canonical Form (CCF)

Eg 1: Express the function $f1 = ab'c + bc' + ac$ in canonical form?

$$f1 = ab'c + bc'.1 + ac.1$$

$$= ab'c + bc'(a + a') + ac(b + b')$$

$$= \mathbf{ab'c + bc'a + bc'a' + acb}$$



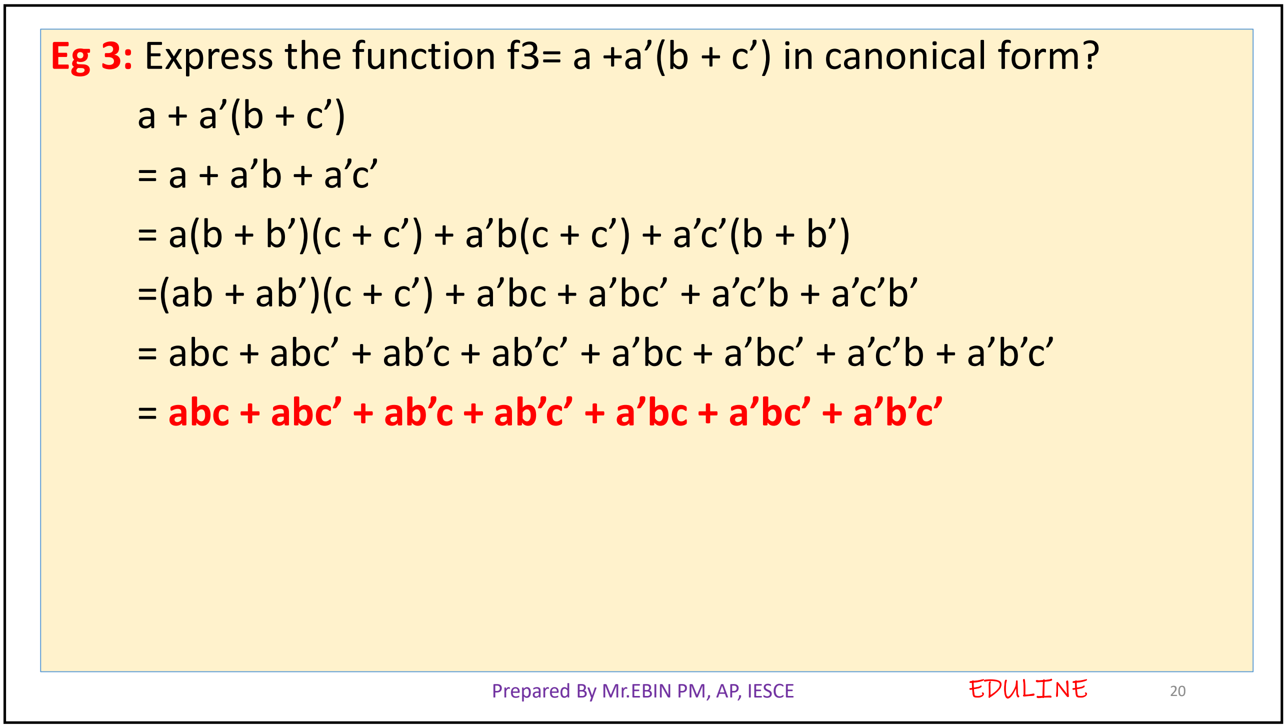
Eg 2: Express the function $f2= (a + b)(b + c')$ in canonical form?

$(a + b)(b + c')$

$= (a + b + c.c') (b + c'+ a.a')$

$=(a + b + c)(a + b +c') (b + c'+ a)(b + c'+ a')$

$=$**$(a + b + c)(a + b + c')(b + c' + a')$**



Eg 3: Express the function $f3= a +a'(b + c')$ in canonical form?

$a + a'(b + c')$

$= a + a'b + a'c'$

$= a(b + b')(c + c') + a'b(c + c') + a'c'(b + b')$

$=(ab + ab')(c + c') + a'bc + a'bc' + a'c'b + a'c'b'$

$= abc + abc' + ab'c + ab'c' + a'bc + a'bc' + a'c'b + a'b'c'$

$=$ **$abc + abc' + ab'c + ab'c' + a'bc + a'bc' + a'b'c'$**

FUNDAMENTAL PRODUCT (MINTERM)

- The number of all possible minterms of **n** variables is $\mathbf{2^n}$
- 0 is written for a complemented variable (a’ = 0)
- 1 is written for the uncomplemented variable (a = 1)
- Each minterm is designated as $\mathbf{m_i}$, the subscript **i** is the decimal value of the binary number.

Minterm	Binary number	Symbolic Representation
a’b’c’	000	m_0
a’b’c	001	m_1
a’bc’	010	m_2
a’bc	011	m_3
ab’c’	100	m_4

Minterm	Binary number	Symbolic Representation
ab’c	101	m_5
abc’	110	m_6
abc	111	m_7

FUNDAMENTAL SUM (MAXTERM)

- The number of all possible maxterms of **n** variables is $\mathbf{2^n}$
- 0 is written for a uncomplemented variable (a = 0)
- 1 is written for the complemented variable (a’ = 1)
- Each maxterm is designated as $\mathbf{M_i}$, the subscript **i** is the decimal value of the binary number.

Maxterm	Binary number	Symbolic Representation
a+b+c	000	M_0
a+b+c’	001	M_1
a+b’+c	010	M_2
a+b’+c’	011	M_3
a’+b+c	100	M_4

Maxterm	Binary number	Symbolic Representation
a’+b+c’	101	M_5
a’+b’+c	110	M_6
a’+b’+c’	111	M_7

- $\sum$ indicate that the terms are minterms and the function is a summation.
- $\prod$ indicate that the terms are maxterms and the function is a product.
- The complement of **m_i** is **M_i** and vice versa. That is,

$m_i' = M_i$

$M_i' = m_i$

$f1 = a'b'c' + a'b'c + abc$
$= m_0+m_1+m_7$
$= \boldsymbol{\sum(0, 1, 7)}$

$f2 = (a+b+c)(a'+b'+c)(a'+b'+c')$
$= M_0M_6M_7$
$= \boldsymbol{\prod(0, 6, 7)}$

❖Examples

1. Reduce $x(x'+yz)$

$x(x' + yz)$
$= xx' + xyz$
$= \mathbf{xyz}$

2. Reduce $x(x'y + x'z)$

$x(x'y + x'z)$
$= xx'y + xx'z$
$= \mathbf{0}$

3. prove that $ab'(c + bd) + a'b'= b'c + a'b'$

$ab'(c + bd) + a'b'$
$= ab'c + ab'bd + a'b'$
$= ab'c + a'b'$
$= b'(ac + a')$
$= b'(a' + c)(a' + a)$
$= \mathbf{a'b' + b'c}$

❖Disjunctive & Conjunctive Canonical Forms (DCF & CCF)

- DCF is same as sum of minterms (canonical SOP)
- CCF is same as product of maxterms (canonical POS)
- If a Boolean function is expressed in the DCF, it can also be expressed in the CCF
- **Eg1:** Express the function f1=a'bc+ab'c'+abc in the other type of canonical form?

a'bc + ab'c'+abc

= 011 100 111 So, $f_i = m_3+m_4+m_7$ (minterms)

Then Maxterms are $f_j= M_0.M_1.M_2.M_5.M_6$

Eg 2: Express the function f2=a'b'c'd'+a'bc'd+a'bcd' in the other type of canonical form?

a'b'c'd' + a'bc'd + a'bcd'

= 0000 0101 0110 So, $f_i = m_0+m_5+m_6$ (minterms)

Then Maxterms are

$f_j= M_1.M_2.M_3.M_4.M_7.M_8.M_9.M_{10}.M_{11}.M_{12}.M_{13}.M_{14}.M_{15}$

➢A Boolean function expressed as a sum of minterms or as a product of maxterms can be converted in to the other form as given by

$$\sum m_i = \prod M_j \quad \& \quad \prod M_i = \sum m_j$$

- Where, the subset i and j are two partitions of the entire set of 2^n subscripts of either m's or M's

Q: Expand A’+B’ to minterms and maxterms?

A’+B’
= A’.1 + B’.1
= A’(B + B’) + B’(A + A’)
= A’B + A’B’ + B’A + B’A’
= A’B + A’B’ + AB’
= 01 + 00 + 10
= $m_1 + m_0 + m_2$
= **∑(0, 1, 2)**

Corresponding maxterms is M_3
= **∏(3)**

DUAL & COMPLEMENT

❖DUAL

To obtain the dual of an expression

1. Change the ORs to ANDs , ANDs to ORs
2. Change the 0s to 1s, 1s to 0s
3. **Do not complement the variables**

Eg: A’B + A’BC’ + A’BCD + A’BC’D’E

DUAL = (A’+B).(A’+B+C’).(A’+B+C+D).(A’+B+C’+D’+E)

❖COMPLEMENT

To obtain the complement of an expression

1. Change the ANDs to ORs , ORs to ANDs
2. Change the 0s to 1s , 1s to 0s
3. **Complement each variable**

Eg: AB + A(B+C) + B’(B+D)

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

IMPORTANT QUESTIONS

1. Find the SOP form of (A+C) (AB’+AC) (A’C’+B’) ?

(A+C) (AB’+AC) (A’C’+B’) simply multiply

= (AAB’+AAC+CAB’+CAC) (A’C’+B’)

= (AAB’A’C’+AACA’C’+CAB’A’C’+CACA’C’+AAB’B’+AACB’+CAB’B’+CACB’)

= AB’ + ACB’ + ACB’ + ACB’

= **AB’ + AB’C**

2. For the function $f=AB'D+A'BC+BC'D'$. Obtain this f in SOP form?

$$f = AB'D + A'BC + BC'D'$$
$$= AB'D(C+C') + A'BC(D+D') + BC'D'(A+A')$$
$$= AB'DC + AB'DC' + A'BCD + A'BCD' + BC'D'A + BC'D'A'$$

3. Express the Boolean function $f=xy+x'z$ in a product of maxterms form?

$$f = xy + x'z$$
$$= xy(z+z') + x'z(y+y')$$
$$= xyz + xyz' + x'zy + x'zy'$$
$$= m_7+m_6+m_3+m_2$$
$$= M_0.M_1.M_4.M_5 \quad = \prod(0, 1, 4, 5)$$

4. Find the minterms of $A+B$?

$$A + B$$
$$= A(B+B') + B(A+A')$$
$$= AB + AB' + BA + BA'$$
$$= AB + AB' + BA'$$
$$= m_3+m_2+m_1$$
$$= \sum(1, 2, 3)$$

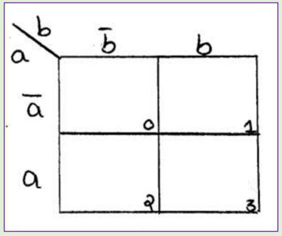
5. Convert f=A’B’C’+A’B’C+A’BC’+AB’C’+ABC’+ABC into POS form?

f = A’B’C’+A’B’C+A’BC’+AB’C’+ABC’+ABC
= ∑(0, 1, 2, 4, 6, 7)
= ∏(3, 5)
= (A+B’+C’) (A’+B+C’)
= 0 1 1 1 0 1
3 5

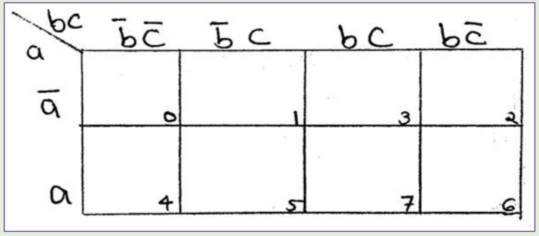
THE KARNAUGH MAP (K-MAP)

- The K-map is a very convenient way of representing a switching function.
- A two variable K-map will have 2^2=4 cells
- A three variable K-map will have 2^3=8 cells
- A four variable K-map will have 2^4=16 cells

2 variable K-map



3 variable K-map

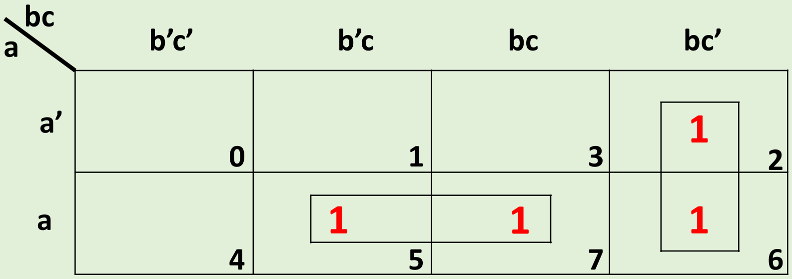


4 variable K-map

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

- We can group the elements in to pairs, quads and octect
- In a pair, one variable is reduced.
- In a quad, two variables can be eliminated
- In an octect, three variables can be eliminated

Eg: Reduce the expression AB+AB'C+A'BC'+BC' using K-map?



a \ bc	b'c'	b'c	bc	bc'
a'	0	1	3	1 2
a	4	1 5	1 7	1 6

Ans: **ac + bc'**

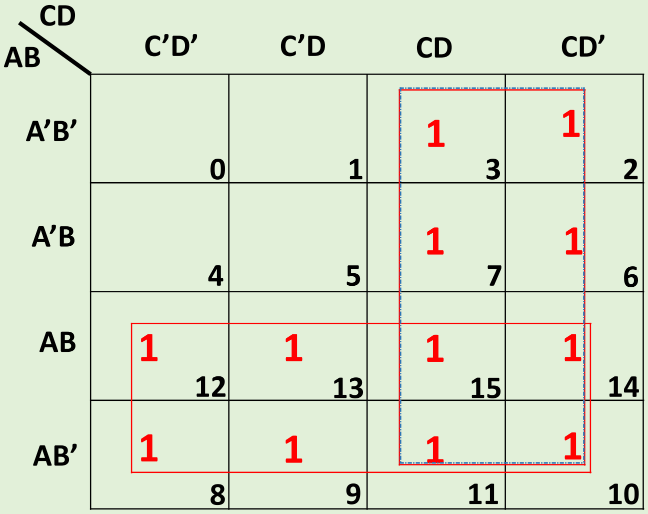
Eg: Convert A+B’C’ to minterms using K-map?

a \ bc	b’c’	b’c	bc	bc’
a’	1 0	1	3	2
a	1 4	1 5	1 7	1 6

minterms = ∑(0, 4, 5, 6, 7)

= A’B’C’ + AB’C’ + AB’C + ABC’ + ABC

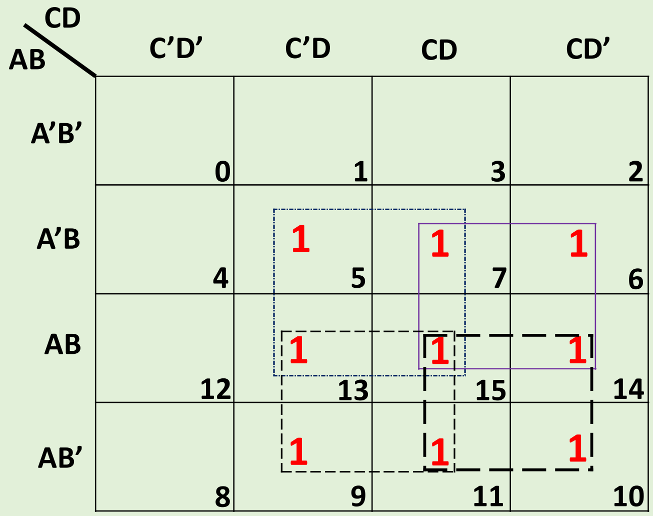
Eg: Reduce AB+AC+C+AD+AB’C+ABC using K-map?



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

Ans: **A+C**

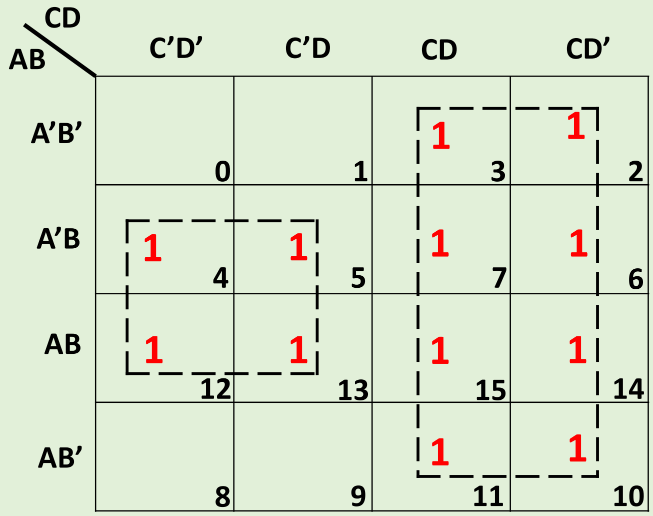
Eg: Reduce ∑(5, 6, 7, 9, 10, 11, 13, 14, 15) using K-map?



CD AB	C'D'	C'D	CD	CD'
A'B'	0	1	3	2
A'B	4	1 5	1 7	1 6
AB	12	1 13	1 15	1 14
AB'	8	1 9	1 11	1 10

Ans: **BD+DA+CB+CA**

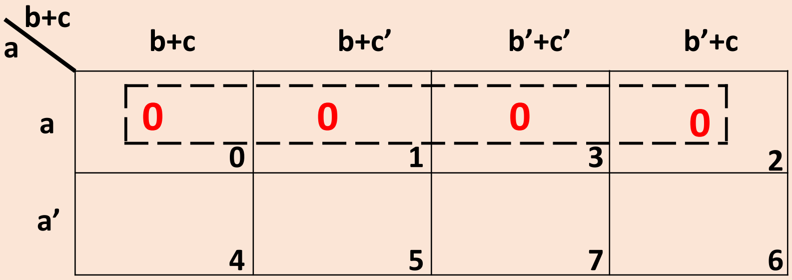
Eg: Reduce AB'C+B+BD'+ABD'+A'C using K-map?



CD AB	C'D'	C'D	CD	CD'
A'B'	0	1	1 3	1 2
A'B	1 4	1 5	1 7	1 6
AB	1 12	1 13	1 15	1 14
AB'	8	9	1 11	1 10

Ans: **C + BC'**

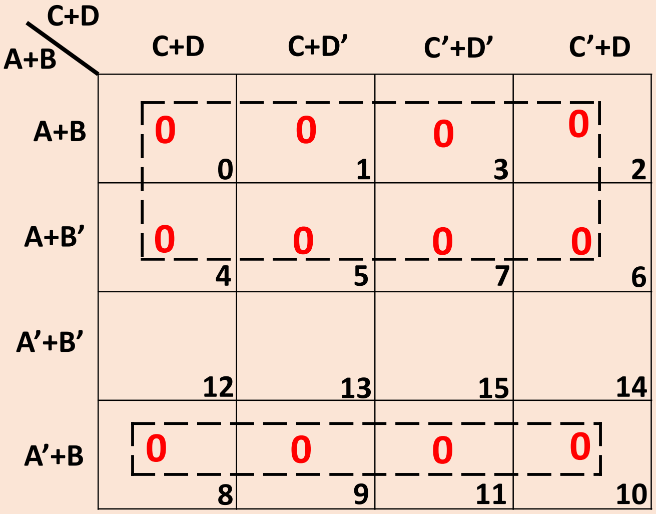
Eg: Reduce the expression (a+b)(a+b’+c)(a+c’) using K-map?



a \ b+c	b+c	b+c’	b’+c’	b’+c
a	0 0	0 1	0 3	0 2
a’	4	5	7	6

Ans: **a**

Eg: Reduce A(B+C’)(A+B’)(B+C+D’) using K-map?

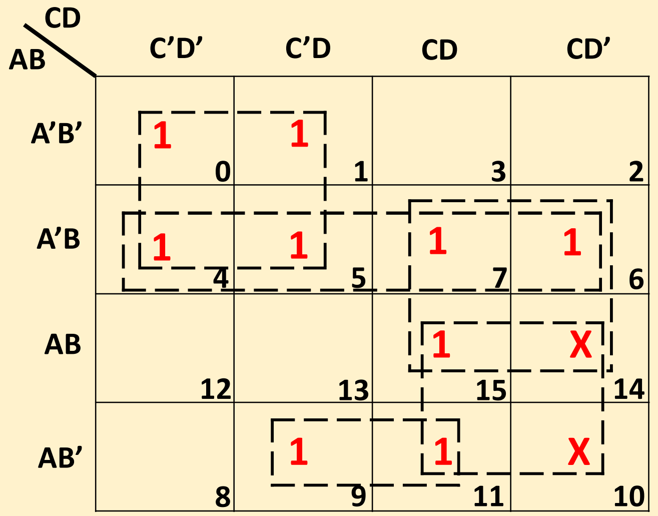


A+B \ C+D	C+D	C+D’	C’+D’	C’+D
A+B	0 0	0 1	0 3	0 2
A+B’	0 4	0 5	0 7	0 6
A’+B’	12	13	15	14
A’+B	0 8	0 9	0 11	0 10

Ans: **A(A’+B)**

DON’T CARE CONDITIONS

Eg: Reduce ∑(0,1,4,5,6,7,9,11,15) + d(10,14) using K-map?

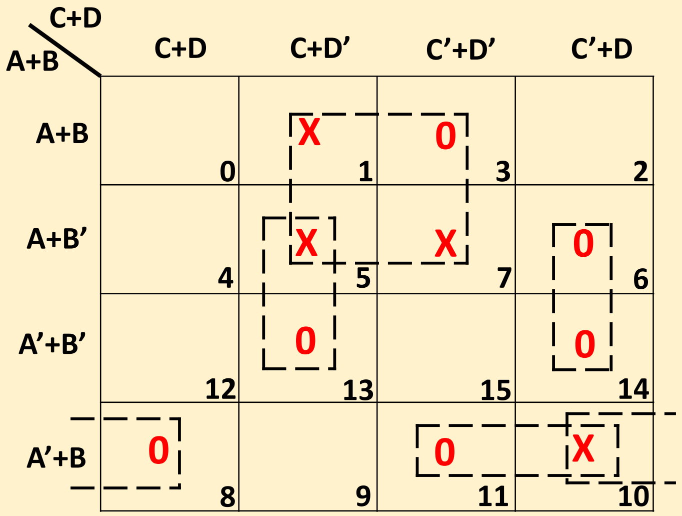


AB \ CD	C’D’	C’D	CD	CD’
A’B’	1 0	1 1	3	2
A’B	1 4	1 5	1 7	1 6
AB	12	13	1 15	X 14
AB’	8	1 9	1 11	X 10

Ans: **A’C’+A’B+BC+AC+AB’D**

DON’T CARE CONDITIONS

Eg: Reduce ∏(3,6,8,11,13,14).d (1,5,7,10) using K-map?

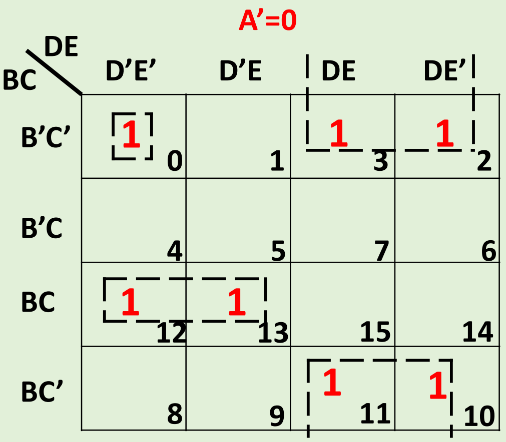


A+B \ C+D	C+D	C+D’	C’+D’	C’+D
A+B	0	X 1	0 3	2
A+B’	4	X 5	X 7	0 6
A’+B’	12	0 13	15	0 14
A’+B	0 8	9	0 11	X 10

Ans: **(A+D’). (B’+C’+D) .**
(A’+B+C’).(B’+C+D’).
(A’+B+D)

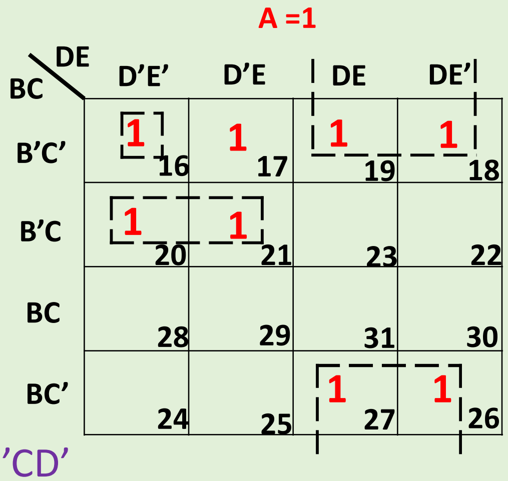
5 variable K-map - Here , number of minterms=2^5 =32

Eg: f(A,B,C,D,E) = ∑(0,2,3,10,11,12,13,16,17,18,19,20,21,26,27)



A'=0

BC \ DE	D'E'	D'E	DE	DE'
B'C'	1 0	1	1 3	1 2
B'C	4	5	7	6
BC	1 12	1 13	15	14
BC'	8	9	1 11	1 10



A =1

BC \ DE	D'E'	D'E	DE	DE'
B'C'	1 16	1 17	1 19	1 18
B'C	1 20	1 21	23	22
BC	28	29	31	30
BC'	24	25	1 27	1 26

Ans: C'D+AB'C'D'E'+AB'C'D'+A'BCD'+AB'CD'

LOGIC GATES

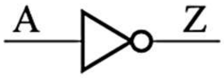
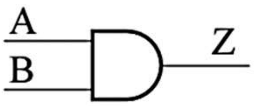
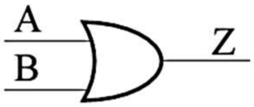
- Logic Gates are the basic building blocks of any digital system.
- It is an electronic circuit having one or more than one input and only one out put.
- The relationship between the input and the output is based on a certain logic.
- Based on this, logic gates are named as

  AND gate

  OR gate

  NOT gate

Elementary (Basic) Logic Gates

Name	Inverter (NOT Gate)	AND Gate	OR Gate
Symbol			

Truth Table

A	Z
0	1
1	0

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

Logic Equation

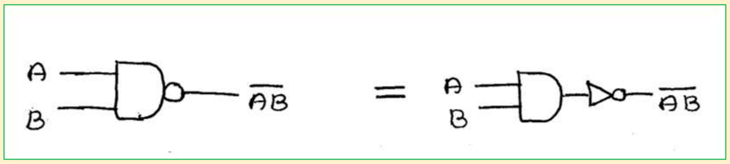
$Z = A' = \overline{A}$

$Z = A \bullet B = AB$

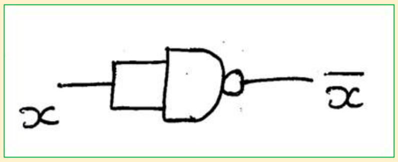
$Z = A + B$

NAND GATE

- NAND means Not AND. Ie, AND gate is NOTed
- NAND gate is equivalent to bubbled OR gate
- Bubbled OR gate is also called negative OR gate
- Bubbled NAND gate is equivalent to OR gate



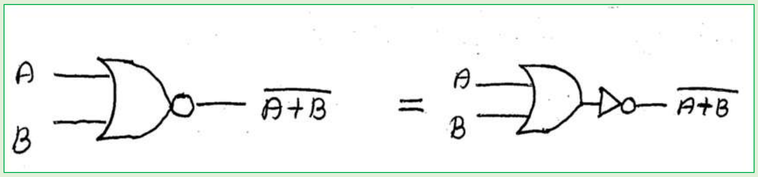
(NAND gate)



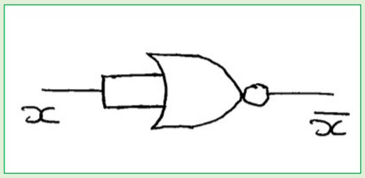
(NAND as an inverter)

NOR GATE

- NOR means Not OR. Ie, OR output is NOTed
- NOR gate is equivalent to bubbled AND gate
- Bubbled AND gate is also called negative AND gate
- Bubbled NOR gate is equivalent to AND gate



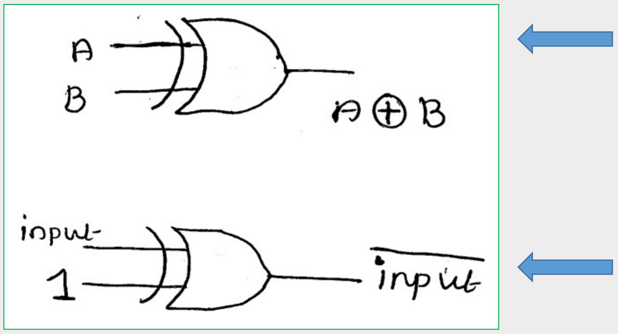
(NOR gate)



(NOR as an inverter)

X-OR GATE

- It is called anti-coincidence gate or inequality detector
- It is an odd function
- Fan –in of XOR is 2
- A $\oplus$ B = A'B + AB'

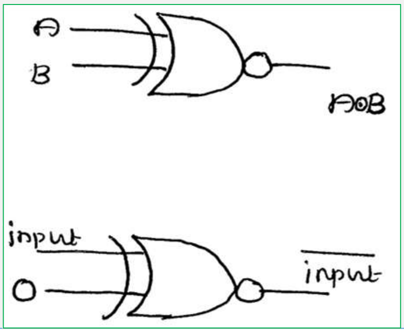


Logic Symbol

XOR as an inverter

X-NOR GATE

- Combination of X-OR and NOT gate
- It is also called Coincidence gate
- It can be used as an equality detector
- Its output is 1 only when its input are equal
- $A \odot B = AB + A'B'$



Logic Symbol

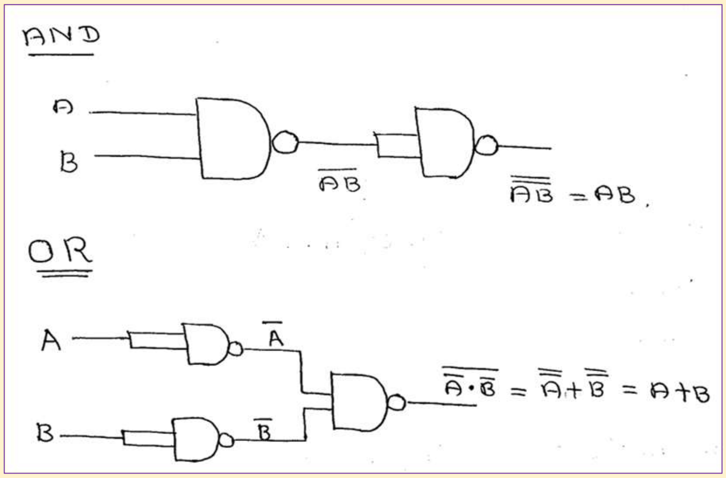
X-NOR as an inverter

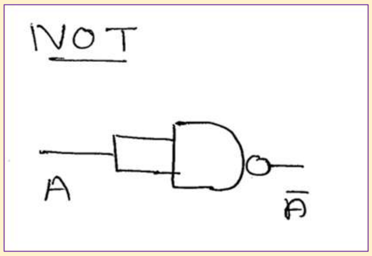
FUNCTIONALLY COMPLETE SETS

- A set of operations is called a functionally complete set if and only if any Boolean function can be expressed by operations belonging to the set only
- It can be seen that the NAND and NOR operations alone form a functionally complete set
- It is there fore possible to implement any switching function using only one type of gate , either NAND or NOR.
- Hence these two operations are known as **universal operations**

NAND REALIZATION

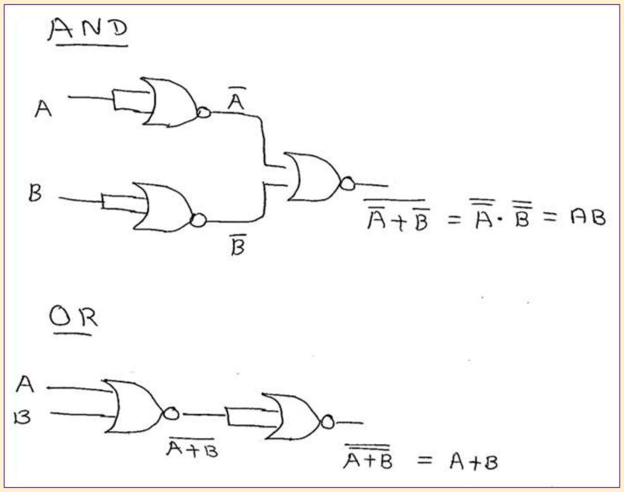
❖Implementation of Basic Gates using **NAND only**

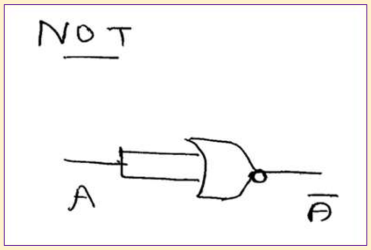




NOR REALIZATION

❖Implementation of Basic Gates using **NOR only**





BINARY NUMBERS

❖Simple way to write Binary numbers:

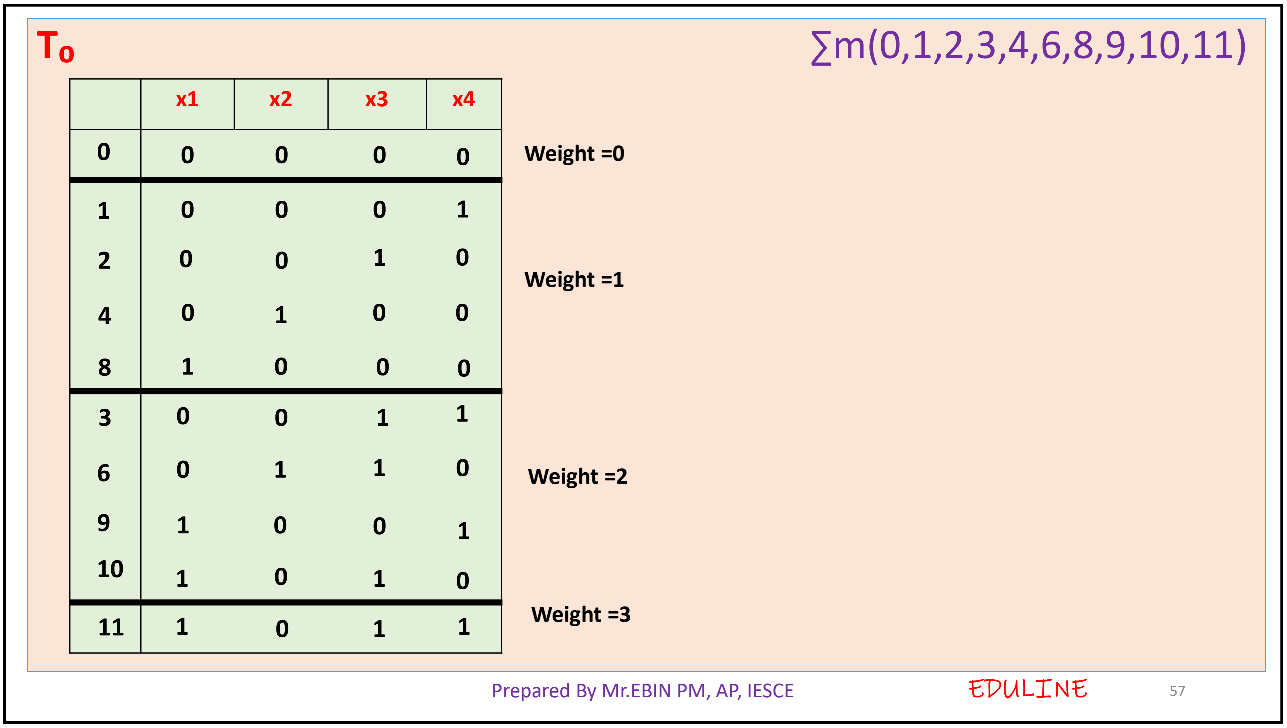
Decimal Number	Binary Number
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

TABULAR MINIMIZATION PROCEDURE

- Tabular minimization method is used to solve more than 6 variable functions
- Commonly used tabular method is Quine-Mc Cluskey (QM)method

Eg1: simplify using QM method f(a b c d)=∑m(0,1,2,3,4,6,8,9,10,11)

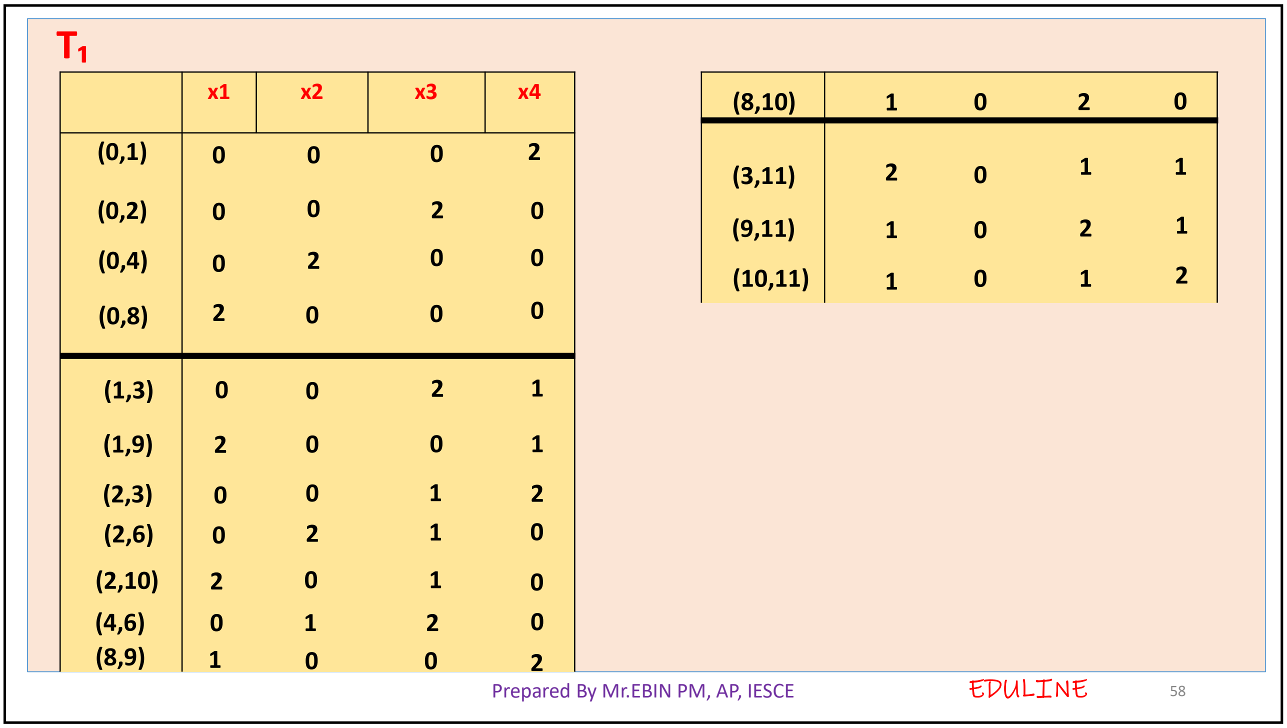
- Weight of a cube is said to be in terms of one's present in the minterm.
- For example, 1100 has weight 2.
- First we create a table T_0, which contains only zero cubes. Then we construct T_1 which contains only one cubes and so on.



T_0

$\sum m(0,1,2,3,4,6,8,9,10,11)$

	x1	x2	x3	x4	
0	0	0	0	0	Weight =0
1	0	0	0	1	Weight =1
2	0	0	1	0	
4	0	1	0	0	
8	1	0	0	0	
3	0	0	1	1	Weight =2
6	0	1	1	0	
9	1	0	0	1	
10	1	0	1	0	
11	1	0	1	1	Weight =3



T_1

	x1	x2	x3	x4
(0,1)	0	0	0	2
(0,2)	0	0	2	0
(0,4)	0	2	0	0
(0,8)	2	0	0	0
(1,3)	0	0	2	1
(1,9)	2	0	0	1
(2,3)	0	0	1	2
(2,6)	0	2	1	0
(2,10)	2	0	1	0
(4,6)	0	1	2	0
(8,9)	1	0	0	2

(8,10)	1	0	2	0
(3,11)	2	0	1	1
(9,11)	1	0	2	1
(10,11)	1	0	1	2

T_2

	x1	x2	x3	x4	
(0,1,2,3)	0	0	2	2	
(0,1,8,9)	2	0	0	2	
(0,2,4,6)	0	2	2	0	A
(0,2,8,10)	2	0	2	0	
(1,3,9,11)	2	0	2	1	
(2,3,10,11)	2	0	1	2	
(8,10,9,11)	1	0	2	2	

T_3

	x1	x2	x3	x4	
(0,1,2,3,8,9,10,11)	2	0	2	2	B

Prime cubes = A , B

Prime cube table

Prime Cubes	0	1	2	3	4	6	8	9	10	11
A(0,2,4,6)	x		x		x	x				
B(0,1,2,3,8,9,10,11)	x	x	x	x			x	x	x	x

Essential Prime cubes = A, B

= 0220 + 2022

= **A'D' + B'**

T_0

Eg 2: f=∑(6,7,8,9)+d(10,11,12, 13,14,15)

	x1	**x2**	**x3**	**x4**	
8	**1**	**0**	**0**	**0**	**Weight =1**
6	**0**	**1**	**1**	**0**	**Weight =2**
9	**1**	**0**	**0**	**1**	
10	**1**	**0**	**1**	**0**	
12	**1**	**1**	**0**	**0**	
7	**0**	**1**	**1**	**1**	**Weight =3**
11	**1**	**0**	**1**	**1**	
13	**1**	**1**	**0**	**1**	
14	**1**	**1**	**1**	**0**	
15	**1**	**1**	**1**	**1**	**Weight =4**

T_1

	x1	**x2**	**x3**	**x4**
(8,9)	**1**	**0**	**0**	**2**
(8,10)	**1**	**0**	**2**	**0**
(8,12)	**1**	**2**	**0**	**0**
(6,7)	**0**	**1**	**1**	**2**
(6,14)	**2**	**1**	**1**	**0**
(9,11)	**1**	**0**	**2**	**1**
(9,13)	**1**	**2**	**0**	**1**
(10,11)	**1**	**0**	**1**	**2**
(10,14)	**1**	**2**	**1**	**0**
(12,13)	**1**	**1**	**0**	**2**
(12,14)	**1**	**1**	**2**	**0**

(7,15)	**2**	**1**	**1**	**1**
(11,15)	**1**	**2**	**1**	**1**
(13,15)	**1**	**1**	**2**	**1**
(14,15)	**1**	**1**	**1**	**2**

T_2

	x1	x2	x3	x4	
(8,9,10,11)	1	0	2	2	
(8,9,12,13)	1	2	0	2	
(8,10,12,14)	1	2	2	0	
(6,7,14,15)	2	1	1	2	A
(9,11,13,15)	1	2	2	1	
(10,11,14,15)	1	2	1	2	
(12,13,14,15)	1	1	2	2	

T_3

	x1	x2	x3	x4	
(8,9,10,11,12 ,13,14,15)	1	2	2	2	B

Prime cubes = A , B

Prime cube table

Prime Cubes	6	7	8	9
A(6,7,14,15)	x	x		
B(8,9,10,11,12,13,14,15)			x	x

- For creating prime cube table, we will not consider Don't care terms

Essential Prime cubes = A, B

= 2112 + 1222

= **BC + A**

T_0

Eg 3: f=∑(2,6,13,14,15,18,26,30)

	x1	x2	x3	x4	x5	
2	0	0	0	1	0	Weight =1
6	0	0	1	1	0	Weight =2
18	1	0	0	1	0	
13	0	1	1	0	1	Weight =3
14	0	1	1	1	0	
26	1	1	0	1	0	
15	0	1	1	1	1	Weight =4
30	1	1	1	1	0	

T_1

	x1	x2	x3	x4	x5	
(2,6)	0	0	2	1	0	A
(2,18)	2	0	0	1	0	B
(6,14)	0	2	1	1	0	C
(18,26)	1	2	0	1	0	D
(13,15)	0	1	1	2	1	E
(14,15)	0	1	1	1	2	F
(14,30)	2	1	1	1	0	G
(26,30)	1	1	2	1	0	H

Prime cube table

Prime Cube	2	6	13	14	15	18	26	30
A(2,6)	X	X						
B(2,18)	X					X		
C(6,14)		X		X				
D(18,26)						X	X	
E(13,15)			X		X			
F(14,15)				X	X			
G(14,30)				X				X
H(26,30)							X	X

Essential prime cube= E

Selective Prime cube table

Prime Cube	2	6	14	18	26	30
A(2,6)	X	X				
B(2,18)	X			X		
C(6,14)		X	X			
D(18,26)				X	X	
E(13,15)						
F(14,15)			X			
G(14,30)			X			X
H(26,30)					X	X

- Here, we cannot apply the dominance relation. This type of function is called Cyclic function. For solving cyclic functions, we use branching method.

- In the above table, F(14,15) gets deleted, being dominated by both G and C
- In branching method, we select say A, so that we delete A(2,6) and (2,6) minterms from the above table.

Prime Cube	14	18	26	30
B(2,18)		X		
C(6,14)	X			
D(18,26)		X	X	
G(14,30)	X			X
H(26,30)			X	X

D and G are selected

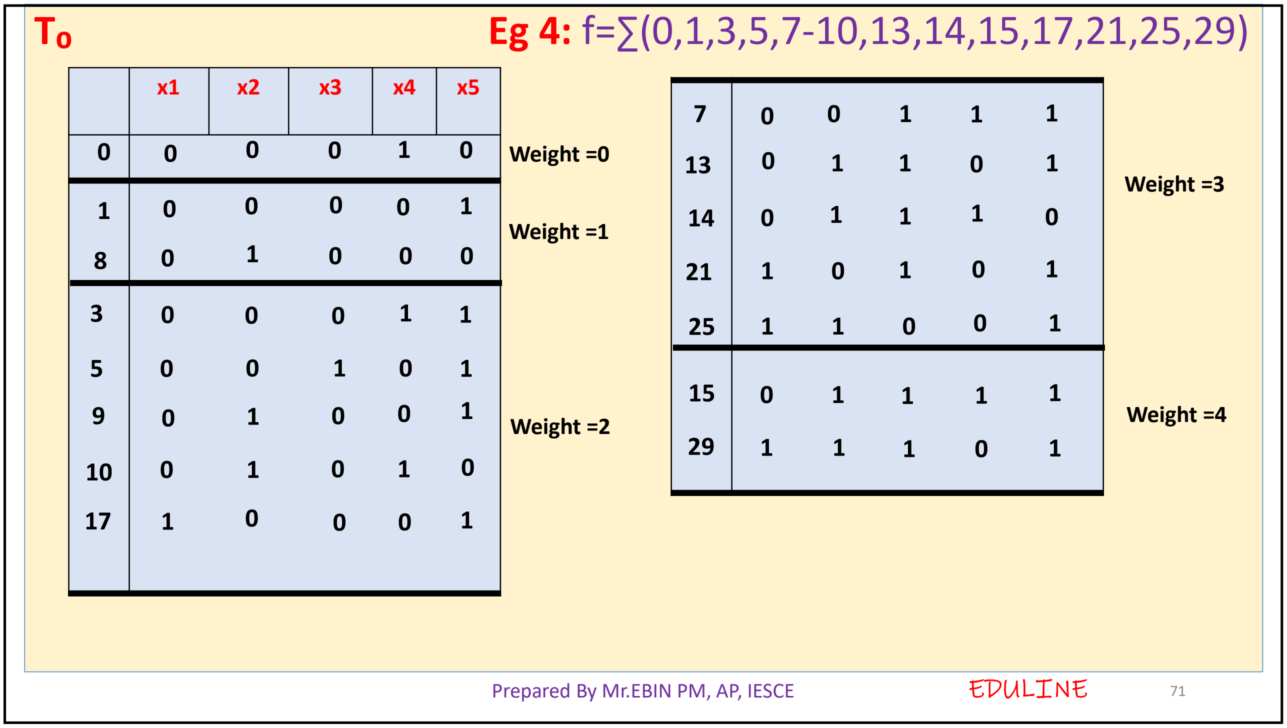
= **E + {A + D + G}**

- Another solution, when we select B, we delete B(2,18) and (2,18) minterms from the table

Prime Cube	6	14	26	30
A(2,6)	X			
C(6,14)	X	X		
D(18,26)			X	
G(14,30)		X		X
H(26,30)			X	X

C and H are selected

= **E + {B + C + H}**



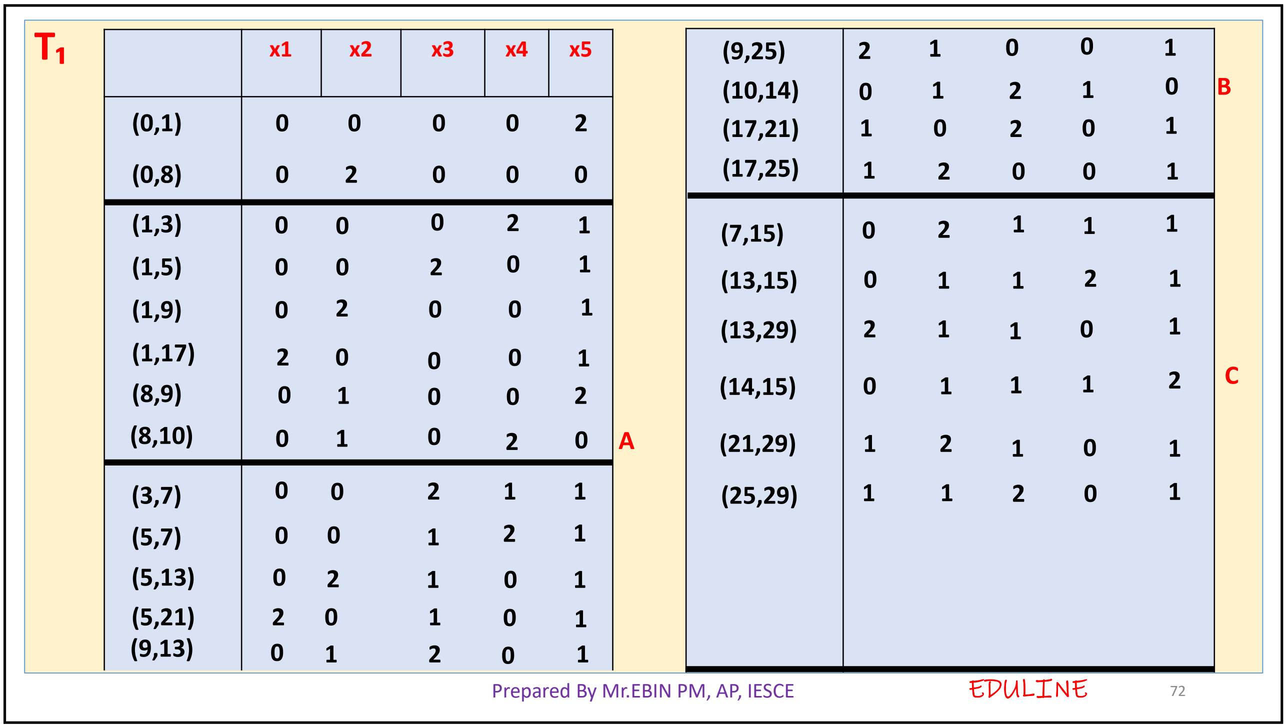
T₀

Eg 4: f=∑(0,1,3,5,7-10,13,14,15,17,21,25,29)

	x1	x2	x3	x4	x5	
0	0	0	0	1	0	Weight =0
1	0	0	0	0	1	Weight =1
8	0	1	0	0	0	
3	0	0	0	1	1	Weight =2
5	0	0	1	0	1	
9	0	1	0	0	1	
10	0	1	0	1	0	
17	1	0	0	0	1	

7	0	0	1	1	1	Weight =3
13	0	1	1	0	1	
14	0	1	1	1	0	
21	1	0	1	0	1	
25	1	1	0	0	1	
15	0	1	1	1	1	Weight =4
29	1	1	1	0	1	

Prepared By Mr.EBIN PM, AP, IESCE EDULINE 71



T₁

	x1	x2	x3	x4	x5	
(0,1)	0	0	0	0	2	
(0,8)	0	2	0	0	0	
(1,3)	0	0	0	2	1	
(1,5)	0	0	2	0	1	
(1,9)	0	2	0	0	1	
(1,17)	2	0	0	0	1	
(8,9)	0	1	0	0	2	
(8,10)	0	1	0	2	0	A
(3,7)	0	0	2	1	1	
(5,7)	0	0	1	2	1	
(5,13)	0	2	1	0	1	
(5,21)	2	0	1	0	1	
(9,13)	0	1	2	0	1	

(9,25)	2	1	0	0	1	
(10,14)	0	1	2	1	0	B
(17,21)	1	0	2	0	1	
(17,25)	1	2	0	0	1	
(7,15)	0	2	1	1	1	
(13,15)	0	1	1	2	1	
(13,29)	2	1	1	0	1	
(14,15)	0	1	1	1	2	C
(21,29)	1	2	1	0	1	
(25,29)	1	1	2	0	1	

Prepared By Mr.EBIN PM, AP, IESCE EDULINE 72

T_2

	x1	x2	x3	x4	x5	
(0,1,8,9)	0	2	0	0	2	D
(1,3,5,7)	0	0	2	2	1	E
(1,5,9,13)	0	2	2	0	1	
(1,5,17,21)	2	0	2	0	1	
(1,9,17,25)	2	2	0	0	1	
(5,7,13,15)	0	2	1	2	1	F
(5,13,21,29)	2	2	1	0	1	
(9,13,25,29)	2	1	2	0	1	
(17,21,25,29)	1	2	2	0	1	

T_3

	x1	x2	x3	x4	x5	
(1,5,9,13,17,21,25,29)	2	2	2	0	1	G

Prime cubes = A, B, C, D, E, F, G

- Next we create prime cube table.

Prime cube table

Prime Cube	0	1	3	5	7	8	9	10	13	14	15	17	21	25	29
A(8,10)						x		x							
B(10,14)								x		x					
C(14,15)										x	x				
D(0,1,8,9)	x	x				x	x								
E(1,3,5,7)		x	x	x	x										
F(5,7,13,15)				x	x				x		x				
G(1,5,9,13,17,21,25,29)		x		x			x		x			x	x	x	x

Essential prime cube= D,E,G

Selective Prime cube table

Prime Cube	10	14	15
A(8,10)	x		
B(10,14)	x	x	
C(14,15)		x	x
F(5,7,13,15)			x

Now we apply row domination

Selective prime cube= B, F

Final solution = **D+ E+ G + {B + F}**

= 02002+00221+22201+01210+02121

= **A’C’D’ + A’B’E + D’E + A’BDE’ +A’CE**