Module -1 **Principles of combinational logic-1** **Hrs: 10**

Definition of combinational logic,Canonical forms, Generation of switching equations from truth tables,Karnaugh maps-3, 4 and 5 variables, Incompletely specified functions (Don'tCare terms), Simplifying Max term equations.

Recommended readings:

1. John M Yarbrough, "Digital Logic Applications and Design",

Thomson Learning, 2001.

Unit-3.1, 3.2, 3.3, 3.4

combinational logic

Also known as "combinatorial logic," it refers to a digital logic function made of primitive logic gates (AND, OR, NOT, etc.) in which all outputs of the function are directly related to the current combination of values on its inputs. Any changes to the signals being applied to the inputs will immediately propagate through the gates until their effects appear at the outputs. Contrast with sequential logic.

sequential logic

A digital logic function made of primitive logic gates (AND, OR, NOT, etc.) in which the output values depend not only on the values currently being presented to its inputs, but also on previous input values. The output depends on a "sequence" of input values. Contrast with combinational logic.

Canonical Forms

There are two standard or ***canonical*** ways of expressing boolean functions:

1. ***Sum-of-products (SOP)***.

E.g.

$$X = A + \overline{B}C + \overline{A}BC$$

2. ***Product-of-sums (POS)***

E.g.

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

These representations are useful for

- direct implementation, and
- starting logic function minimization.

We will focus on SOP.

Consider

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

where

- ***product terms*** *A*, $\overline{B}C$, $\overline{A}BC$
- ***minterms*** $\overline{A}BC$

A minterm is any ANDed term containing all of the varibles (perhaps complemented).

Let's look at the truth table which corresponds to this function:

	A	*B*	*C*	*f*(*A*,*B*,*C*)
m_0	0	0	0	0
m_1	0	0	1	1
m_2	0	1	0	0
m_3	0	1	1	1
m_4	1	0	0	1
m_5	1	0	1	1
m_6	1	1	0	1
m_7	1	1	1	1

(Check this!)

Each row of the truth table corresponds to one of the $2^n = 8$ possible minterms in n=3 variables.

$$m_i = m_i(A, B, C), \quad i = 0, \ldots, 2^3 - 1$$

E.g.

$$m_3 = \overline{A}BC = 001$$

Actually, the truth table specifies the function as a ***sum of minterms***:

$$\begin{aligned} f(A, B, C) &= m_1 + m_3 + m_4 + m_5 + m_6 + m_7 \\ &= \sum m(1, 3, 4, 5, 6, 7) \end{aligned}$$

This is called the ***canonical SOP representation*** of the function *f*.

The ***minterm code*** for *n*=3 is as follows:

m_0	0	0	0	$\overline{A}$	$\overline{B}$	$\overline{C}$

m_1	0	0	1	$\overline{A}$	$\overline{B}$	C
m_2	0	1	0	$\overline{A}$	B	$\overline{C}$
m_3	0	1	1	$\overline{A}$	B	C
m_4	1	0	0	A	$\overline{B}$	$\overline{C}$
m_5	1	0	1	A	$\overline{B}$	C
m_6	1	1	0	A	B	$\overline{C}$
m_7	1	1	1	A	B	C

Complemented variables correspond to 0 and un complemented variables correspond to 1.

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

The function can be put into canonical SOP form algebraically as follows:

$$\overline{B}C = (\overline{A} + A)\overline{B}C = \overline{A}.\overline{B}C + A\overline{B}C = m_1 + m_5$$

$$A = (B + \overline{B})A = \ldots = m_4 + m_5 + m_6 + m_7$$

$$f(A,B,C) = \sum(1,3,4,5,6,7)$$

(fill in the missing steps!) and so on combining we get as before.

Any Boolean function can be expressed in canonical SOP form.

Simplification and Implementation of Boolean Functions

Boolean functions can be implemented in hardware in a number of ways. For instance, standard discrete TTL or CMOS ICs could be used, in which case it is useful to find the simplest expression for the function being implemented. Or if programmable devices are to be used, then a more direct representation of the function may be useful.

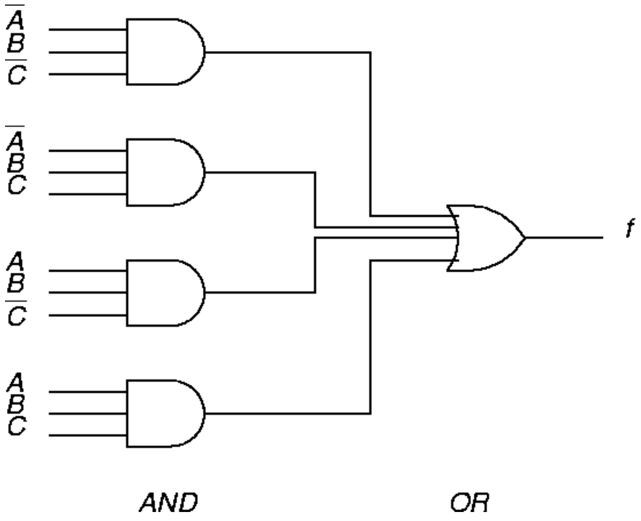
Direct Implementation

Consider the function

$$f(A,B,C) = \sum m(2,3,6,7)$$

expressed in canonical SOP form. Then assuming all variables and their complements are available we can implement this function with the AND-OR circuit of Figure as shown.

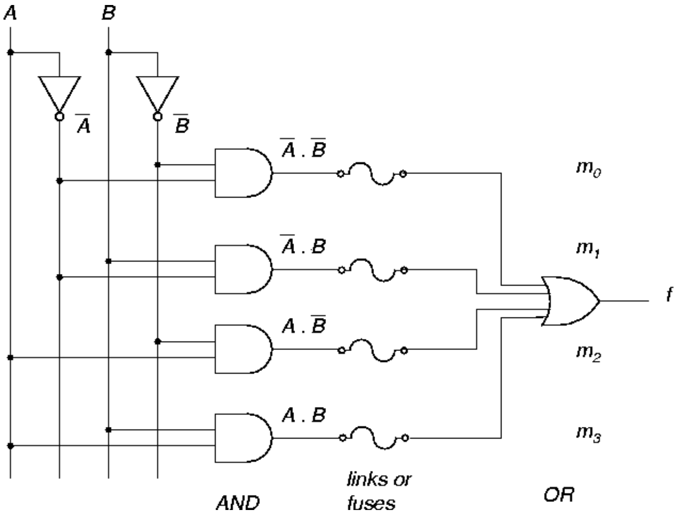
Figure : AND/OR implementation.



This implemntation is ***not minimal*** in general (i.e. can realize *f* with fewer gates).

This representation is direct and is useful when implementing with ***programmable logic devices (PLD)***. To illustrate, consider functions *f*=*f*(*A*,*B*) of two variables (n=2, 2^n=4). A PLD schematic is shown in Figure.

Figure : PLD implementation.



This PLD can program any given function *f*(*A*,*B*) by breaking appropriate links.

Karnaugh Maps (K-Maps)

Karnaugh or ***K- maps*** are useful tool fot boolean function minimization, and for visualization of the boolean function. In brief,

- K-maps provide a graphical method for minimizing boolean functions via pattern recognition forup to about *n*=6 variables.
- For larger numbers of variables, there are computer algorithms which can yield near-minimal implementations.
- K-maps are a way of expressing truth tables to make minimization easier. They are constructed from minterm codes.

Consider the boolean function

$$f(A,B) = \sum m(0,1,2)$$

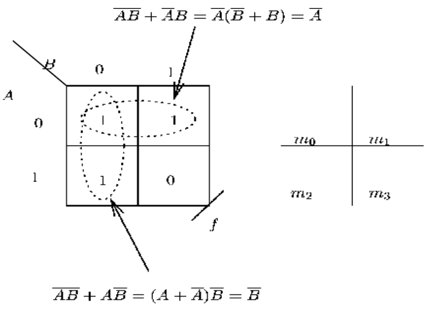
The truth table is

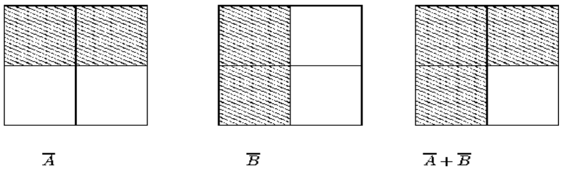
A	B	f	
0	0	1	m_0
0	1	1	m_1
1	0	1	m_2
1	1	0	m_3

The K-map is shown in Figure .The essence of the K-map is the two dimensional representation of *f*, which is equivalent to the truth table but more visual.

To minimize *f*, we loop out ***logical adjacencies***, Figure .

Figure : K-map showing looped-out terms and also corresponding minterms.


$\overline{A}\overline{B}+\overline{A}B=\overline{A}(\overline{B}+B)=\overline{A}$
B
A
0
1
0
1
1
1
1
0
f
m_0
m_1
m_2
m_3
$\overline{A}\overline{B}+A\overline{B}=(A+\overline{A})\overline{B}=\overline{B}$



Therefore

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

This is less complex than *f* in canonical SOP form.

Note. Looping out logical adjacencies is a graphical alternative to algebraic calculations.

Unit distance code (Gray code.) For two bits, the Gray code is:

00 01 11 10

Only one bit changes as you go from left to right. This code ***preserves logical adjacencies***.

The ***K-map method*** is to loop out groups of 2^n logically adjacent minterms. Each looped out group corresponds to a product term in a minimal SOP expression.

1.Loop out single 1s (*n*=0) which have no logical adjacencies.

2.Loop out all pairs of 1s (*n*=1) which cannot be included in a larger group.

3.Loop out all quads of 1s (*n*=2) which cannot be included in a larger group.

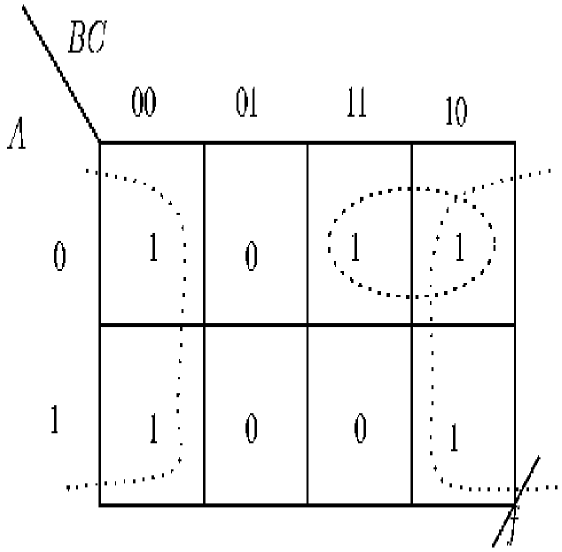
Etc.

$$f(A,B,C)=\sum m(0,2,3,4,6)$$

Example. The K-map is shown in Figure.

$$f(A,B,C)=\sum m(0,2,3,4,6)$$

Figure: K-map for.



Moving left to right or up to down in the K-map changes only one digit in the minterm code. Note the wrap-around at the ends: because of logical adjacency, the top and bottom are joined, and the left and right are joined.

n=0: none

n=1: $m_2 + m_3 = \overline{A}B$

n=2: $m_0 + m_2 + m_4 + m_6 = \overline{C}$

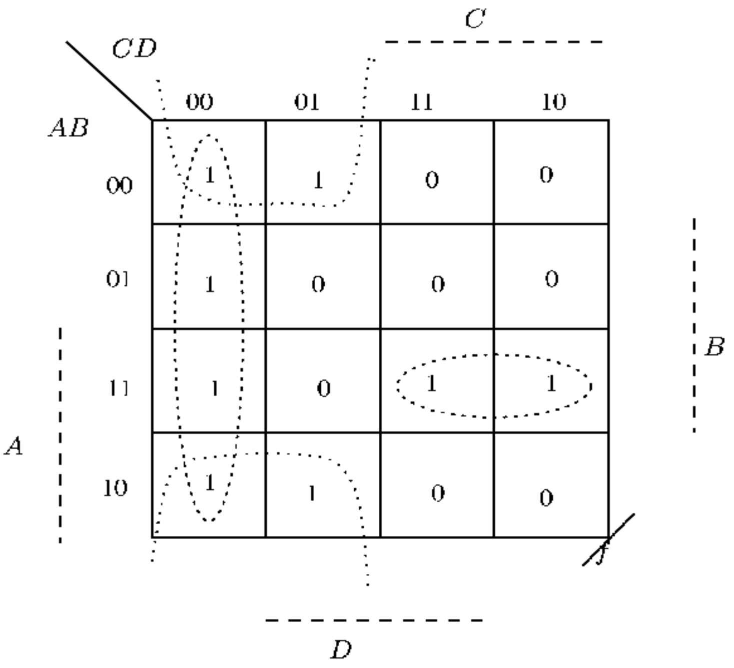
Therefore the minimal SOP representation is

$$f = \overline{A}B + \overline{C}$$
$$f(A,B,C,D) = \sum m(0,1,4,8,9,12,14,15)$$

Example. The K-map is shown in Figure.

$$f(A,B,C,D) = \sum m(0,1,4,8,9,12,14,15)$$

Figure: K-map for.



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

Therefore the minimal SOP representation is

$$f = ABC + \overline{C}.\overline{D} + \overline{B}.\overline{C}$$

Don't cares. In some applications it doesn't matter what the output is for certain input values. These are called ***don't cares***.

For instance, in the ***Binary Coded Decimal*** code, not all input values occur:

0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6

0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

$$0, 1, \ldots, 9$$

The decimal numbers are those in the range, and a minimum of 4 bits is needed to encode these.

$$10, 11, \ldots, 15$$

The remaining numbers correspond to code values which are not used in BCD.

$$\phi$$

We shall use the symbols or *X* ***to denote don't cares.***

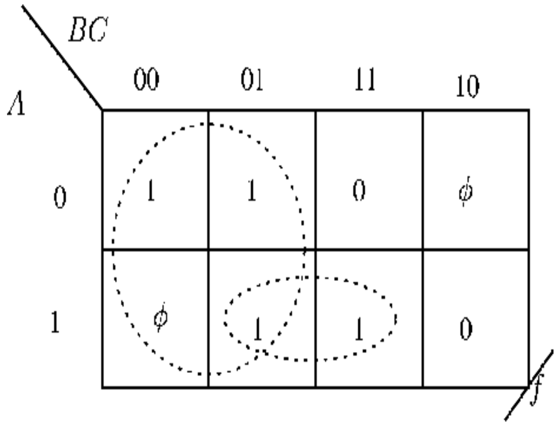
Don't cares can be exploited to help minimize boolean functions.

$$f(A,B,C) = \sum m(0,1,5,7) + \phi(2,4)$$

Example. The K-map is shown in Figure.

$$f(A,B,C) = \sum m(0,1,5,7) + \phi(2,4)$$

Figure: K-map for.



The minimal SOP representation is

$$f = \overline{B} + AC$$

KARNAUGH MAPS (K- MAP)

A method for graphically determining implicants and implicates of a Boolean function was developed by Veitch and modified by Karnaugh . The method involves a diagrammatic representation of a Boolean algebra. This graphic representation is called map.

It is seen that the truth table can be used to represent complete function of n-variables. Since each variable can have value of 0 or 1. The truth table has 2n rows. Each rows of the truth table consist of two parts (1) an n-tuple which corresponds to an assignment to the n-variables and (2) a functional value.

A Karnaugh map (K-map) is a geometrical configuration of 2n cells such that each of the n-tuples corresponds to a row of a truth table uniquely locates a cell on the map. The functional values assigned to the n-tuples are placed as entries in the cells, i.e. 0 or 1 are placed in the associated cell.

An significant about the construction of K-map is the arrangement of the cells. Two cells are physically adjacent within the configuration if and only if their respective n-tuples differ in exactly by one element. So that the Boolean law x+x=1 cab be applied to adjacent cells. Ex. Two 3- tuples (0,1,1) and (0,1,0) are physically a djacent since these tuples vary by one element.

One variable : One variable needs a map of 2^1= 2 cells map as shown below

x	f(x)
0	f(0)
1	f(1)

TWO VARIABLE : Two variable needs a map of 2^2 = 4 cells

x	y	f(x,y)
0	0	f(0,0)
0	1	f(0,1)
1	0	f(1,0)
1	1	f(1,1)

THREE VARIABLE : Three variable needs a map of 2^3 = 8 cells. The arrangement of cells are as follows

x	y	z	f(x,y,z)
0	0	0	f(0,0,0)
0	0	1	f(0,0,1)
0	1	0	f(0,1,0)
0	1	1	f(0,1,1)
1	0	0	f(1,0,0)
1	0	1	f(1,0,1)
1	1	0	f(1,1,0)
1	1	1	f(1,1,1)

FOUR VARIABLE : Four variable needs a map of 2^4 = 16 cells. The arrangement of cells are as follows

w	x	y	z	f(w,x,y,z)	w	x	y	z	f(w,x,y,z)
0	0	0	0	f(0,0,0,0)	1	0	1	0	f(1,0,1,0)
0	0	0	1	f(0,0,0,1)	1	0	1	1	f(1,0,1,1)
0	0	1	0	f(0,0,1,0)	1	1	0	0	f(1,1,0,0)
0	0	1	1	f(0,0,1,1)	1	1	0	1	f(1,1,0,1)

0 1 0 0 f(0,1,0,0)

0 1 0 1 f(0,1,0,1)

0 1 1 0 f(0,1,1,0)

0 1 1 1 f(0,1,1,1)

1 0 0 0 f(1,0,0,0)

1 0 0 1 f(1,0,0,1)

1 1 1 0 f(1,1,10)

1 1 1 1 f(1,1,1,1)

Four variable K-map.

0000	0001	0011	0010
0100	0101	0111	1010
1100	1101	1111	1110
1000	1001	1011	1010

Ex. Obtain the minterm canonical formula of the three variable problem given below

f(x, y,z) = x y z+ x y z + x y z + x y z

f(x,y,z) = ∑m(0,2,4,5)

00	01	11	11
1	0	0	1
1	1	0	0

Ex. Express the minterm canonical formula of the four variable K-map given below

00	01	11	10
1	1	0	1
1	1	0	0
0	0	0	0
1	0	0	1

f(w,x,y,z) = w x y z + w x y z + w x y z + w x y z + w x y z + w x y z

f(w,x,y,z) = ∑ m(0, 1, 2, 4, 5,

Ex. Obtain the max term canonical formula

(POS) of the three variable problem stated above

f(x,y,z) = (x + y +z)(x + y +z)(x + y +z) (x + y +z)

f(x,y,z) = ΠM(1,3,6,7)

Ex Obtain the max term canonical formula

(POS) of the four variable problem stated above

f(w,x,y,z) = (w + x + y + z) (w + x + y + z) (w + x + y + z)

(w + x + y + z) (w + x + y + z) (w + x + y + z)

(w + x + y + z) (w + x + y + z) (w + x + y + z)

f(w,x,y,z) = ΠM(3,6,7,9,11,12,13,14,15)

PRODUCT AND SUM TERM REPRESENTATION OF K –MAP

1.The importance of K-map lies in the fact that it is possible to determine the implicants and implicates of a function from the pattern of 0’s and 1’s appearing in the map. The cell of a K-map has entry of 1’s is refereed as 1-cell and that of 0,s is referred as 0-cell.

2. The construction of an n-variable map is such that any set of 1-cells or 0-cells which form a 2ax2b rectangular grouping describing a product or sum term with n-a-b variables , where a and b are non-negative no.

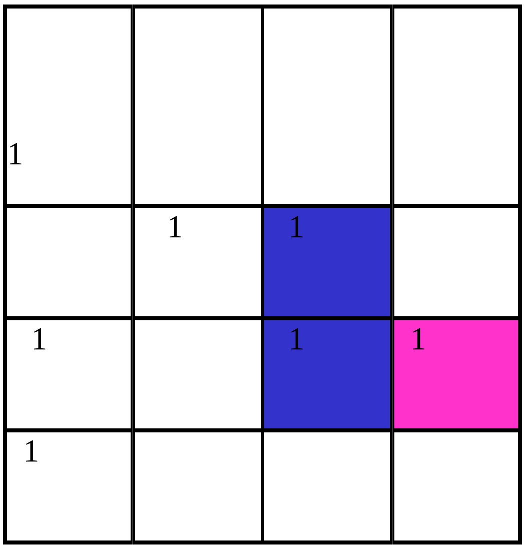
3. The rectangular grouping of these dimensions referred as Sub cubes. The sub cubes must be the power of 2 i.e. 2^{a+b} equals to 1,2,4,8 etc.

4. For three variable and four variable K-map it must be remembered that the edges are also adjacent cells or sub cubes hence they will be grouped together.

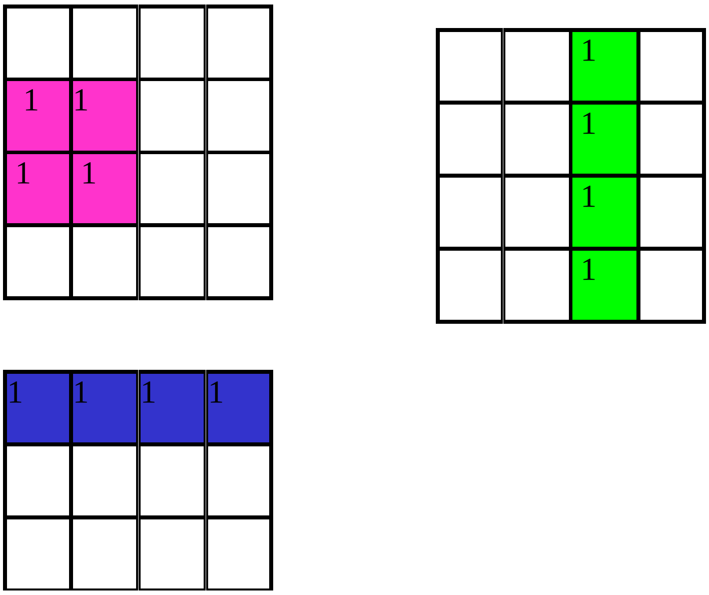
5. Given an n-variable map with a pair of adjacent 1-cells or 0-cellscan result n-1 variable. Where as if a group of four adjacent sub cubes are formed than it can result n-2 variables. Finally if we have eight adjacent cells are grouped may result n-3 variable product or sum term.

Typical pair of sub cubes

w x z



Typical group of four adjacent subcubes



Typical group of four adjacent sub cubes.

1			1
1			1

1			1
1			1

Typical group of eight adjacent sub cubes.

1	1	1	1
1	1	1	1

	1	1	
	1	1	
	1	1	
	1	1	

1	1	1	1
1	1	1	1

1			1
1			1
1			1
1			1

Typical map sub cubes describing sum terms

0	0		

0	0		
0	0		

		0	0
		0	0
		0	0
		0	0

USING K-MAP TO OBTAIN MINIMAL EXPRESSION FOR COMPLETE BOOLEAN FUNCTIONS :

How to obtain a minimal expression of SOP or POS of given function is discussed.

PRIME IMPLICANTS and K-MAPS :

CONCEPT OF ESSENTIAL PRIME IMPLICANT

00 01 11 10

0	0	0	1
0	0	1	1

f(x,y,z)= xy+ yz

ALGORITHM TO FIND ALL PRIME IMPLICANTS

A General procedure is listed below

1. For an n-variable map make 2n entries of 1’s. or 0’s.

2. Assign I = n , so that find out biggest rectangular group with dimension 2ax2b = 2 n-1.

3. If bigger rectangular group is not possible I = I-1 form the subcubes which consist of all the previously obtained subcube repeat the step till all 1-cell or 0's are covered.

Remaining is essential prime implicants

1. Essential prime implicants

2. Minimal sums

3. Minimal products

MINIMAL EXPRESSIONS OF INCOMPLETE BOOLEAN FUNCTIONS

1. Minimal sums

2. Minimal products.

EXAMPLE TO ILLUSTRATE HOW TO OBTAIN ESSENTIAL PRIMES

1. f(x,y,z) = ∑m(0,1,5,7)

Ans f(x,y,z) = xz + x y

2. f(w,x,y,z) = ∑m(1,2,3,5,6,7,8,13)

Ans. f(w,x,y,z) = w z +w y+xyz+w x y z

MINIMAL SUMS

f(w,x,y,z)=∑m(0,1,2,3,5,7,11,15)

MINIMAL PRODUCTS

F(w,x,y,z)=∑m(1,3,4,5,6,7,11,14,15)

MINIMAL EXPRESSIONS OF INCOMPLETE BOOLEAN FUNCTIONS

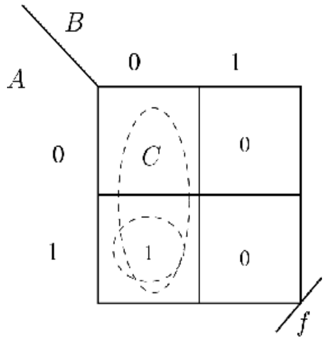
f(W,X,Y,Z)=∑m(0,1,3,7,8,12) +dc(5,10,13,14)

Entered-Variable K-Maps

A generalization of the k-map method is to introduce variables into the k-map squares. These are called ***entered variable k-maps***. This is useful for functions of large numbers of variabes, and can generally provide a clear way of representing Boolean functions.

An entered variable k-map is shown in Figure.

Figure : An entered variable k-map.



Note the variable *C* in the top left square. It corresponds to

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

It can be looped out with the 1, since 1=1+*C*, and we can loop out the two terms

$$\overline{A}.\overline{B}.C \quad \text{and} \quad A.\overline{B}.C$$

to get

$$\overline{B}.C.$$

The remaining term

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

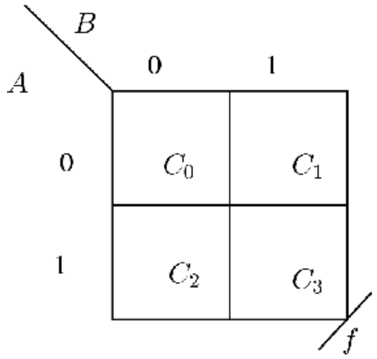
needs to be added to the cover, or more simply, just loop out the 1. The outcome is

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

Figure shows another EV k-map, with four entered variables C_0, C_1, C_2, C_3. Each of these terms are different and must be looped out individually to get

$$f = \overline{A}.\overline{B}.C_0 + \overline{A}.B.C_1 + A.\overline{B}.C_2 + A.B.C_3.$$

Figure: Another entered variable k-map.



Recommended question and answer –unit-1

Jan-2009

l a) Convert the given boolean function f(x, y, z) = [x + x Z (y + z)] into maxterm canonical formula and hence highlight the importance of canonical formul.1.

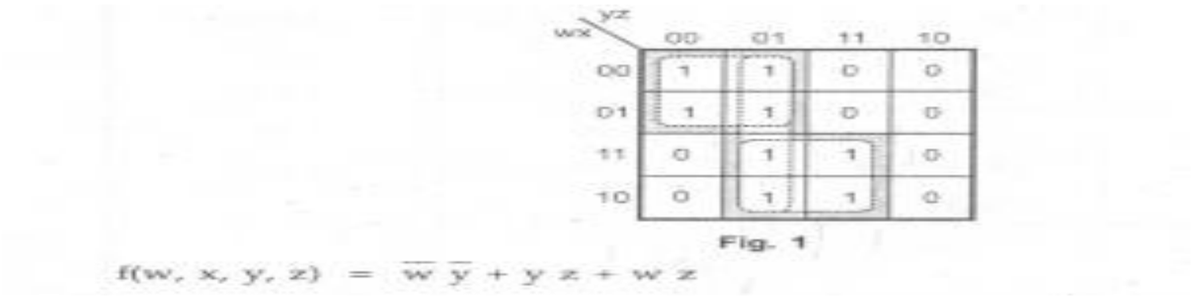
(5)

f(x,y,z)= x (y + y) (z + z) ;- x y z + x Z (y + y)
= x y z + x Y z + x yz + x Y z + x Y z + x Y z + x Y z
f(x, y, z) = x y z + x Y z + x Y z + x Y z + x Y z + x Y z

1 b) Distinguish the prime implicants and essential prime implicants. Determine the same of the function
f(w, x, y, z) = I m(O, 1, 4, 5, -9, 11, 13, 15) using K-map and hence the minimal sum expression.

(5)

Ans. : After grouping the cells, sum terms which appear in the k-map are called prime implicants groups. It is observed than some cells may appear in only one prime implicant group, while other cells may appear in more than one prime implicants group. The cells which appear in only one prime implicant group are called essential cells and corresponding prime implicants are called essential prime implicants.



wx \ yz	00	01	11	10
00	1	1	0	0
01	1	1	0	0
11	0	1	1	0
10	0	1	1	0

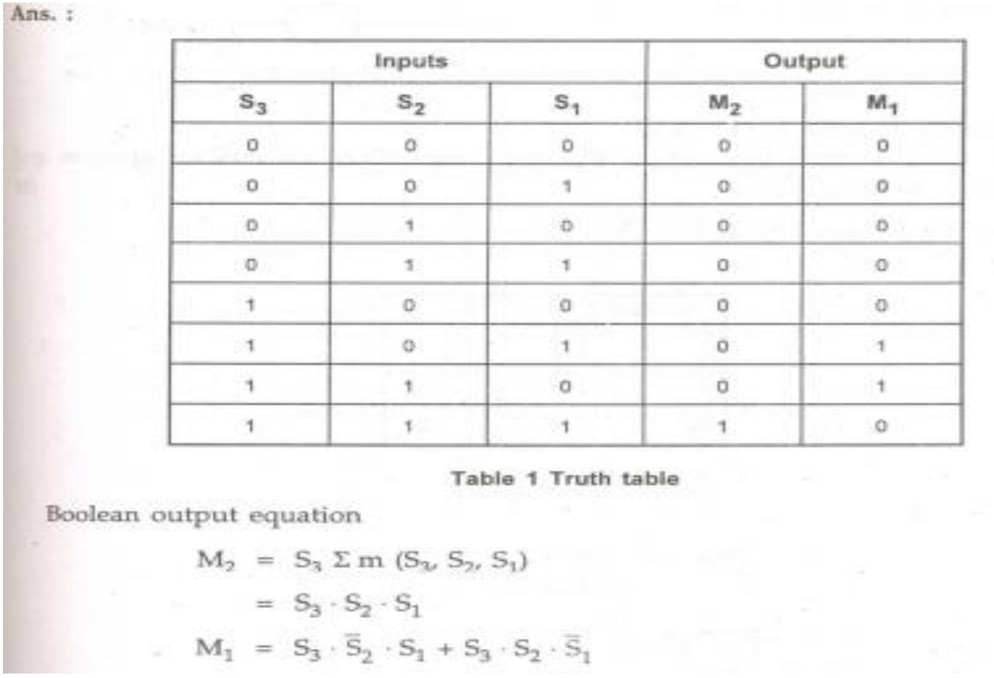
Fig. 1

$$f(w, x, y, z) = \overline{w}\,\overline{y} + y\,z + w\,z$$

Jan-2008

Q.l a) *Two motors* M2 *and* !v1; *are controlled by three sensors 531* 52 *and* 51' *One motor* M2 *is to run any time all three sensors are on. The other motor is to run whenever sensors* 52 *or* 51 *but not both are on and* 53 *is off For all sensor combinations where* M1 *is an,* M2 *is to be off except when all the three sensors are off and then both .motors must remain off Construct the truth table and write the Boolean output equation.*

(6)



Ans. :

Inputs			Output	
S_3	S_2	S_1	M_2	M_1
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

Table 1 Truth table

Boolean output equation

$$M_2 = S_3\,\Sigma\,m\,(S_3, S_2, S_1)$$

$$= S_3 \cdot S_2 \cdot S_1$$

$$M_1 = S_3 \cdot \overline{S}_2 \cdot S_1 + S_3 \cdot S_2 \cdot \overline{S}_1$$

b) *Simplify using Karnaugh map. Write the Boolean equation and realize using NAND gates*

D = *f(w, x, y,* z) = L *(0,* 2, .4, 6, 8) + L *d(lO,* 11, 12, 13, 14, 15).(6)

Ans. : D = f(w, x, y, z) = L *(0-*, 2, 4, 6, 8) + L d(10, 11, 12, 13, 14, 15).

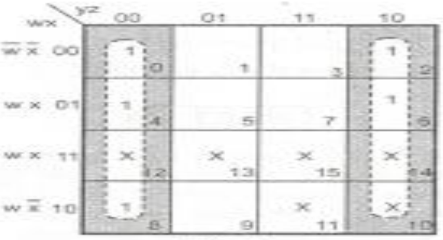


Fig. 1

$$\text{output} = \bar{y}\,\bar{z} + y\,\bar{z}$$
$$= \bar{z}\,(\bar{y} + y)$$
$$= \bar{z}$$

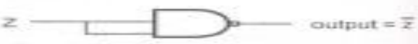


Fig. 2

c. Simplify P = f(a, b, c) = L (0,1, 4, 5, 7) using two variable Karnaugh map. Write the Boolean equation and realize using logic gates

(8)

Ans. : P = f(a, b, c) = Σ (0, 1, 4, 5, 7)

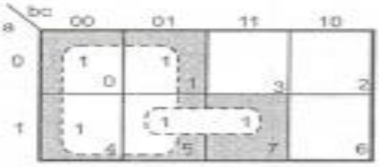
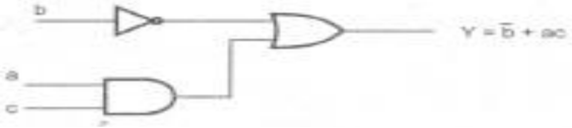


Fig. 3

$$Y = \bar{b} + ac$$



Q.2 a) *Simplify using Karnaugh map L =lea, b,* c, *d)* = 1t (2, 3, 4, 6, 7, *10,* 11, 12).

(6)

Ans. : L = f(a, b, c, d) = 1t (2, 3, 4, 6, 7, 10, 11, 12).

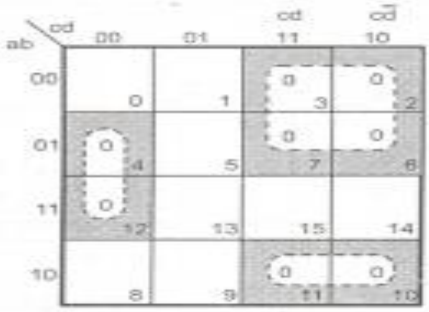


Fig. 5

$$\overline{Y} = b\bar{c}\bar{d} + \bar{a}c + ab\bar{c}$$
$$Y = \overline{b\bar{c}\bar{d} + \bar{a}c + ab\bar{c}}$$

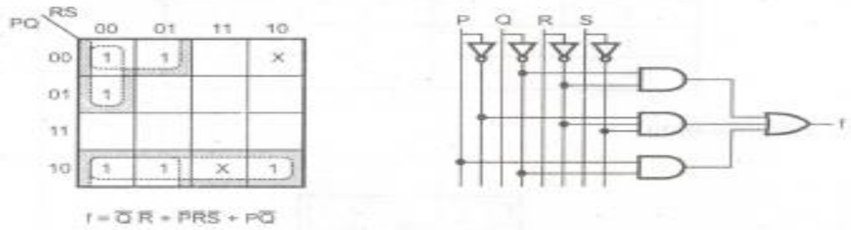
Aug 2009

Q.1 a) *Express the P.O.S. equations in a Maxterms list (decimal notations) form.*

i) T = f(A, B, C) = *(A + B +* C) *(A + B +* C) *(A + B +* C)

ii) J = f(A, B, C, D) = *(A* + B + C + D) *(A* + B + C + D) (:4 + *B* + C + D)
(A + B + C + D) *(A* + *B* + C + D) *(A* + B + C + D) (4)
i) T = f(A, B, C) = (A + B + C) (A + B + C) (A + B + C)
. . . f(A, B, C) = M2 + M3 + M6 = 1t M(2, 3, 6)
ii) J = f(A, B, C, D) = (A + B + C + D) (A + B + C + D) (A + B + C + D)
(A + B + C + D) (A + B + C + D) (A + B + C + D)
= M4 + Ms + Ms + MlO + M12 + M14
= 1t M (4, 5, 8, 10, 12, 14)
b) *Reduce the following function using K-map technique and implement using gates.*
i) f(P, Q, R, 5) = I,*m (0,* 1, 4, 8, 9, *10)* + *d* (2, 11)
ii) f(A, B, C D) = 1t *m (0,* 2, 4, *10,* 11, 14, 15) (10)
Ans.: i) (P, Q, R, S) I,ffi (0, 1,4,8,9, 10) + d (2, 11)
f(P, Q, R, S) Q R + P R S + P Q

ii) f(A, B, C, D) = π M (0 , 2, 4, 10, 11, 14, 15)

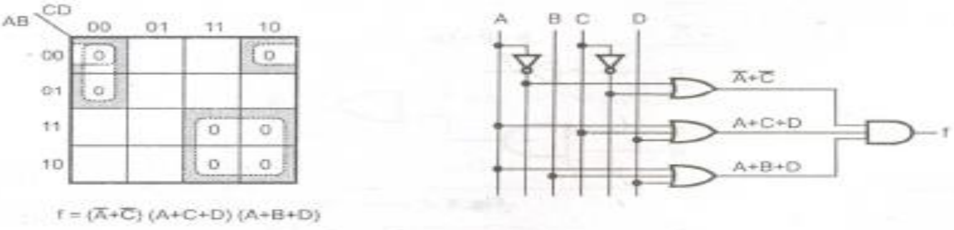


f = Q̄R + PRS + PQ̄

(a) Simplification

(b) Implementation

Fig. 1



f = (Ā+C̄) (A+C+D) (A+B+D)

(a) Simplification

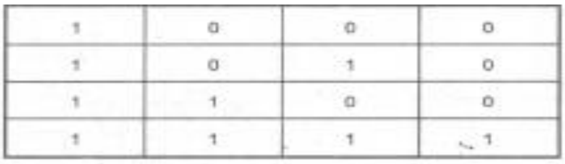
(b) Implementation

Fig. 2

c) *Design a logic circuit with inputs P, Q, R so that output S is high whenever P is zero or whenever Q = R = 1.* (6)

ns. :

P	Q	R	S
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1



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

K-map simplification :

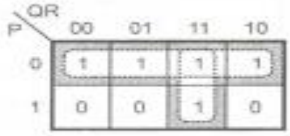


Fig. 3

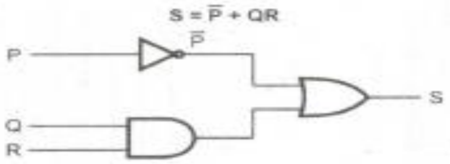


Fig. 4

Aug 2008

Q.l a) *Simplify the following expression using Kamaugh map. Implement the simpl~l. circuit using the gates as indicated.*
i) !(ABCD) = *L* m (2, 3, 4, 5, 13, 15) + *Lex* (8, 9, 10, 11) *use only NAND gat~*
ii) !*(ABCD)* = 1t (2, 3, 4, 6, 7, *10,* 11, 12) *use only NOR gates.*
to implement these circuits.

i) f(ABCD) = *L* m (2, 3, 4, 5, 13, 15) + *Lx* (8, 9, la, 11)

SOP = (Sum of product)

$F(ABCD) = \bar{A}\bar{C}D + CB + A\bar{D}$

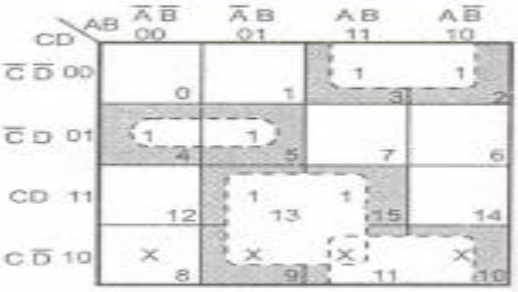


Fig. 1 (a)

Implementation using NAND gates :

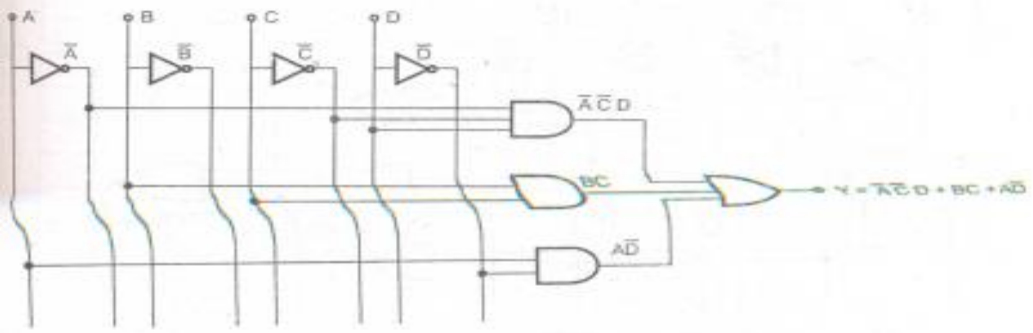


Fig. 1 (b)

ii) $f(ABCD) = \Sigma\pi\,(2,\ 3,\ 4,\ 6,\ 7,\ 10,\ 11,\ 12)$

POS = (Product of sum)

K-map simplification :

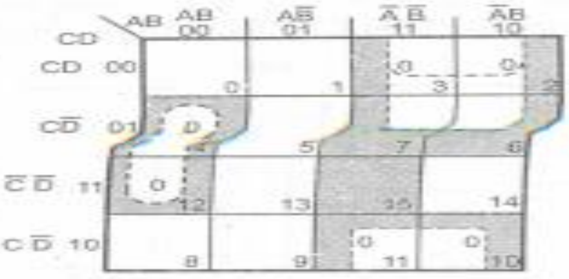


Fig. 1 (c)

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

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

Implementation using NOR gates :

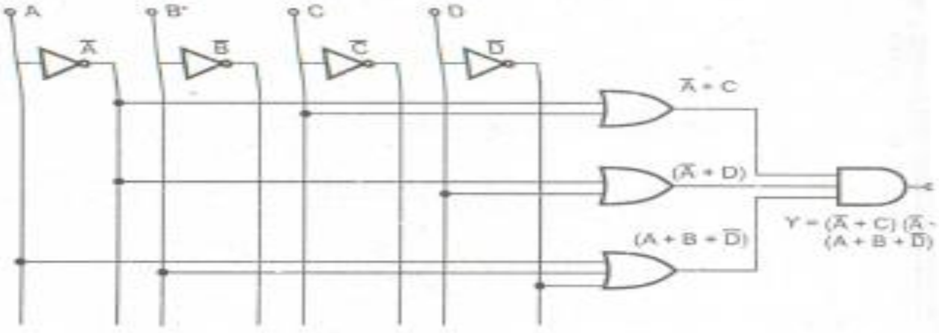


Fig. 1 (d)

Aug-2007

Q.1 a) *Using the theorems of Boolean algebra, simplify the following.*

i) $y_1 = D(\overline{A}+B)+\overline{B}(C+AD)$

ii) $y_2 = \overline{A\overline{B}+ABC}+A(B+A\overline{B})$

iii) $y_3 = \overline{ABC}+\overline{A}\overline{BC}+A\overline{BC}+AB\overline{C}$ **[9]**

Sol. : This topic is not included in the new syllabus.

b) *Express the following expressions in canonical form :*

i) $y_1 = (A+B)(A+C)(B+\overline{C})$

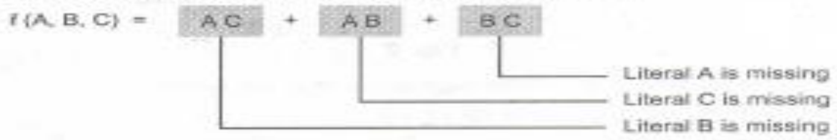
ii) $y_2 = AC+AB+BC$ **[6]**

Sol. : i)

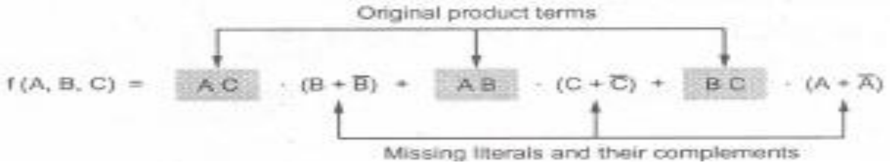
$$\begin{aligned} y_1 &= (A+B)(A+C)(B+\overline{C}) \\ &= ((A+B)+(C\cdot\overline{C}))((A+C)+(B\cdot\overline{B}))((B+\overline{C})+(A\cdot\overline{A})) \\ &= (A+B+C)(A+B+\overline{C})(A+C+B)(A+C+\overline{B}) \\ &\quad (B+\overline{C}+A)(B+\overline{C}+\overline{A}) \\ &= (A+B+C)(A+B+\overline{C})(A+\overline{B}+C)(\overline{A}+B+\overline{C}) \end{aligned}$$

ii) $y_2 = AC + AB + BC$

Step 1 : Find the missing literal/s in each product term



Step 2 : AND product term with (missing literal + its complement)


Original product terms
f (A, B, C) = AC · (B + B̄) + AB · (C + C̄) + BC · (A + Ā)
Missing literals and their complements

Step 3 : Expand the terms and reorder literals.

Expand : $f(A, B, C) = ACB + AC\overline{B} + ABC + AB\overline{C} + BCA + BC\overline{A}$

Recorder : $f(A, B, C) = ABC + A\overline{B}C + ABC + AB\overline{C} + ABC + \overline{A}BC$

Step 4 : Omit repeated product terms

$$f(A, B, C) = ABC + A\overline{B}C + \boxed{ABC} + AB\overline{C} + \boxed{ABC} + \overline{A}BC$$

$$\therefore \quad f(A, B, C) = ABC + A\overline{B}C + AB\overline{C} + \overline{A}BC$$

Sol. : The Boolean expression for EX-OR gate is : $Y = A\overline{B}+\overline{A}B$

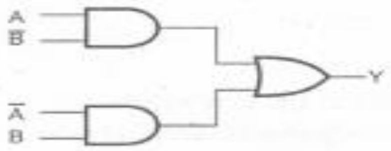


Fig. 1

We can implement AND-OR logic by using NAND-NAND logic as shown in Fig. 2.

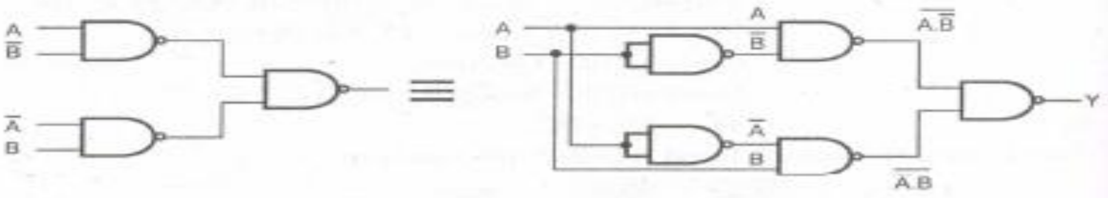
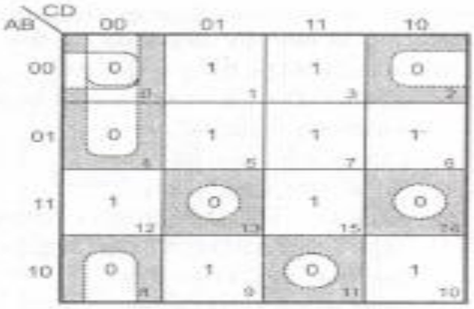


Fig. 2

Q.2 a) *i) Express the following min term expression in POS form :*
$y(A,B,C,D) = \Sigma m(1,3,5,6,7,9,10,12,15)$
ii) Express the following max term expression in SOP form :
$y(A,B,C) = \pi M(0,3,5,6)$

Sol. : $y(A, B, C, D) = \Sigma m(1, 3, 5, 6, 7, 9, 10, 12, 15)$

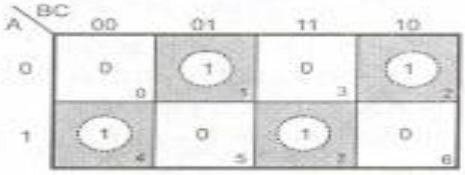


AB \ CD	00	01	11	10
00	0 0	1 1	1 3	0 2
01	0 4	1 5	1 7	1 6
11	1 12	0 13	1 15	0 14
10	0 8	1 9	0 11	1 10

Fig. 3

$$\begin{aligned}\overline{Y} &= \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{C}\,\overline{D} + \overline{B}\,\overline{C}\,\overline{D} + AB\overline{C}D + ABC\overline{D} + A\overline{B}CD \\ &= (A+B+D)(A+C+D)(B+C+D)(\overline{A}+\overline{B}+C+\overline{D}) \\ &\quad (\overline{A}+\overline{B}+\overline{C}+D)(\overline{A}+B+\overline{C}+\overline{D})\end{aligned}$$

ii) $Y(A, B, C) = \pi M(0, 3, 5, 6)$



A \ BC	00	01	11	10
0	0 0	1 1	0 3	1 2
1	1 4	0 5	1 7	0 6

Fig. 4

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

b) i) *What are the advantage, disadvantages of K map?*
ii) Simplify the following function in SOP form using K Map:

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

Sol. : i) **Advantages of K-map method:**

1. It provides a systematic approach for simplifying a Boolean expression.
2. It is very convenient method for simplifying a Boolean expression upto six variables.

Disadvantages of K-map method:

1. As the number of variables increases it is difficult to make judgements about which combinations form the **minimum** expression. In case of complex problem with 7, 8, or even 10 variables it is almost an impossible task to simplify expression by the mapping method.

2. Another important point is that the K-map simplification is manual techniqu and simplification process is heavily depends on the human abilities.

ii) $f(A, B, C, D) = \overline{A}\,\overline{B}C + AD + B\overline{D} + C\overline{D} + A\overline{C}$

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

$= \overline{A}\,\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + (ABD + A\overline{B}D)$
$(C+\overline{C}) + (AB\overline{D} + \overline{A}B\overline{D})(C+\overline{C}) +$
$(AC\overline{D} + \overline{A}C\overline{D})(B+\overline{B}) +$
$(AB\overline{C} + A\overline{B}\,\overline{C})(D+\overline{D})$

$= \overline{A}\,\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + ABCD +$
$A\overline{B}CD + AB\overline{C}D + A\overline{B}\,\overline{C}D +$
$ABC\overline{D} + \overline{A}BC\overline{D} + AB\overline{C}\,\overline{D} + \overline{A}B\overline{C}D$
$+ ABC\overline{D} + \overline{A}BC\overline{D} + AC\overline{B}\,\overline{D} +$
$\overline{A}\,\overline{B}C\overline{D} + AB\overline{C}D + A\overline{B}\,\overline{C}D +$
$AB\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}\,\overline{D}$

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

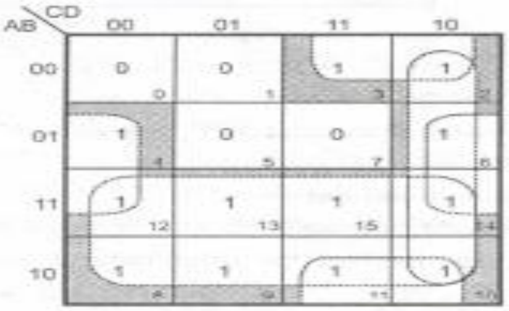


Fig. 5

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

c) *Simplify the following function in POS form using K map:*
$f(A, B, C, D) = \pi M(0, 1, 2, 5, 8, 9, 10)$.

: $f(A, B, C, D) = \pi M(0, 1, 2, 5, 8, 9, 10)$

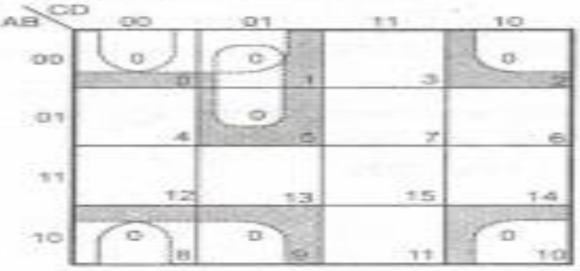


Fig. 6

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

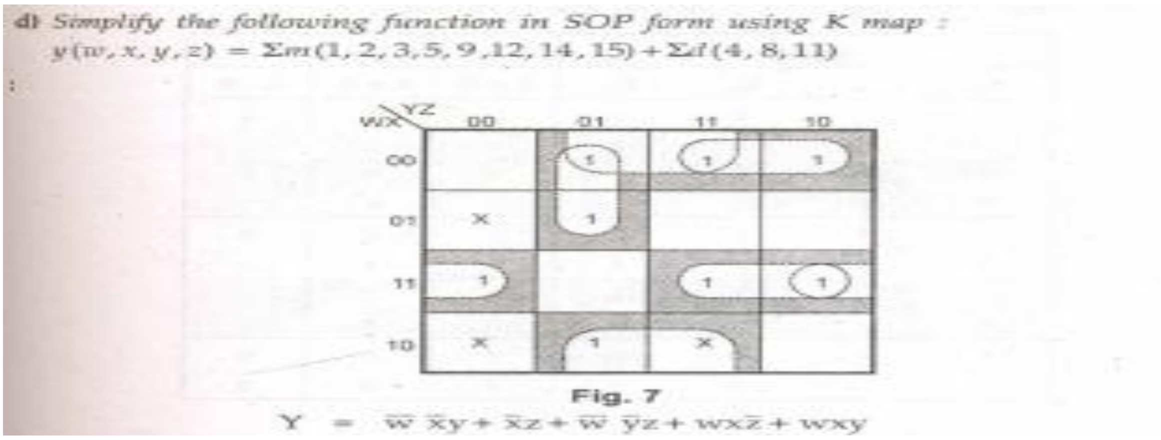
$f = \overline{\overline{B}\,\overline{C} + \overline{B}\,\overline{D} + \overline{A}\,\overline{C}D}$

$= \left(\overline{\overline{B}\,\overline{C}}\right)\left(\overline{\overline{B}\,\overline{D}}\right)\left(\overline{\overline{A}\,\overline{C}D}\right)$

$= \left(\overline{\overline{B}} + \overline{\overline{C}}\right)\left(\overline{\overline{B}} + \overline{\overline{D}}\right)\left(\overline{\overline{A}\,\overline{C}} + \overline{D}\right)$

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

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



d) *Simplify the following function in SOP form using K map :*
$y(w,x,y,z) = \Sigma m(1,2,3,5,9,12,14,15) + \Sigma d(4,8,11)$

:

WX \ YZ	00	01	11	10
00		1	1	1
01	X	1		
11	1		1	1
10	X	1	X	

Fig. 7

$$Y = \bar{w}\,\bar{x}y + \bar{x}z + \bar{w}\,\bar{y}z + wx\bar{z} + wxy$$

Principles of combinational Logic-2

Quine-McCluskey minimization technique- Quine-McCluskey using don’t care terms, educed Prime Implicant Tables, Map entered variables

Recommended readings:

1. John M Yarbrough, “Digital Logic Applications and Design”,
Thomson Learning, 2001.

Unit- 3.5, 3.6

QUINE – McCLUSKEY METHOD

Using K-maps for simplification of Boolean expressions with more than six variables becomes a tedious and difficult task. Therefore a tabular method illustrate below can be used for the purpose.

ALGORITHM FOR GENERATING PRIME IMPLICANTS

The algorithm procedure is listed below

1.Express each minterm of the function in its binary representation.

2. List the minterms by increasing index.

3. Separate the sets of minterms of equal index with lines.

4. Let i = 0.

5. Compare each term of index I with each term of index I+1. For each pair of terms that can combine which has only one bit position difference.

6. Increase I by 1 and repeat step 5 . The increase of I continued until all terms are compared. The new list containing all implicants of the function that have one less variable than those implicants in the generating list.

7. Each section of the new list formed has terms of equal index. Steps 4,5, and 6 are repeated on this list to form another list. Recall that two terms combine only if they have their dashes in the same relative positions and if they differ in exactly one bit position.

8. The process terminates when no new list is formed .

9. All terms without check marks are prime implicants.

Example: Find all the prime implicants of the function

$f(w,x,y,z) = \sum m(0,2,3,4,8,10,12,13,14)$

Step 1: Represent each minter in its 1-0 notation

no.	minterm	1-0 notation	index
0	w x y z	0 0 0 0	0
2	w x y z	0 0 1 0	1
3	w x y z	0 0 1 1	2
4	w x y z	0 1 0 0	1
8	w x y z	1 0 0 0	1
10	w x y z	1 0 1 0	2
12	w x y z	1 1 0 0	2
13	w x y z	1 1 0 1	3
14	w x y z	1 1 1 0	3

Step 2: List the minterm in increasing order of their index.

No.	w x y z	index
0	0 0 0 0	Index 0
2	0 0 1 0	
4	0 1 0 0	Index 1
8	1 0 0 0	
3	0 0 1 1	
10	1 0 1 0	Index 2
12	1 1 0 0	
13	1 1 0 1	
14	1 1 1 0	Index 3

	W x y z	index
0.2	0 0 – 0	
0,4	0 – 0 0	
0,8	- 0 0 0	
2,3	0 0 1 –	
2,10	- 0 1 0	
4,12	- 1 0 0	
8,10	1 0 – 0	
8,12	1 – 0 0	
10,14	1 – 1 0	
12,13	1 1 0 –	
12,14	1 1 - 0	

	w x y z
(0, 2, 8, 10) (0, 4, 8,12)	__ 0 __ 0 __ __ 0 0(index 0)
(8,10,12,14)	1__ __ 0 (index 1)

F(w,x,y,z)=x z + y z +w z+w x y +w x z

PETRICK’S METHOD OF DETERMINING IRREDUNDANT EXPRESSIONS

FIND THE PRIME IMPLICANTS AND IRREDUNDANT EXPRESSION

F(W,X,Y,Z)= ∑M(0,1,2,5,7,8,9,10,13,15)

A=X Y , B= X Z C= Y Z D= X Z

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

P = (A +C)(BD) = ABD +BCD

F1(W,X,Y,Z)= ABD =X Y +X Z +X Z

F2(W,X,Y,Z) = BCD = X Z + Y Z +X Z

DECIMAL METHOD FOR OBTAINING PRIME IMPLICANTS

The prime implicants can be obtained for decimal number represented minterms.In this procedure binary number are not used to find out prime implicants

f(w, x,y,z) =∑m(0,5,6,7,9,10,13,14,15)

fsop= xy +xz+xyz+wyz+w x y z

MAP ENTERED VARIABLE(MEV)

It is graphical approach using k-map to have a variable of order n. Where in we are using a K-map of n-1 variable while map is entered with ouput function and variable.

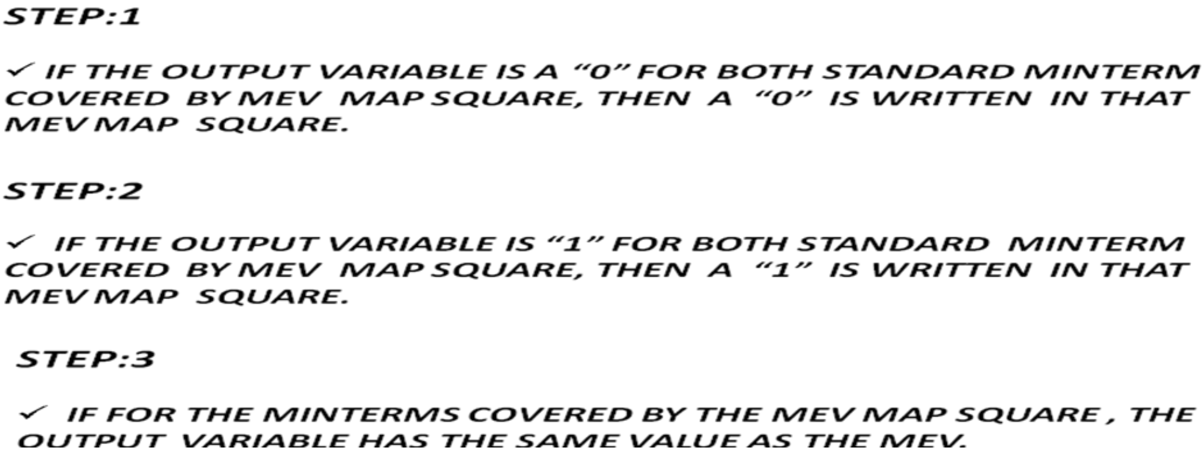
f(w,x,y.z) = ∑m(2,3,4,5,13,15) +dc(8,9,10,11)

Ans.fsop= w z +x y + w x y

- karnaugh mapping is the best manual technique for boolean equation simplification, yet when the map sizes exceed five or six variable unwidely.

- the technique called "map entered variables " (mevs) increases the effective size of a karnaugh maps, allowing a smaller map to handle a greater no. of variables

- the map dimension and the no. of problem variables are related by 2n = m, where n = no.of problem variable, m = no. of squares in k-maps. mev k-maps permit a cell to contain a single (x) or a complete switching expression, in addition to the 1s, 0s and don't care terms.



STEP:1

✓ ***IF THE OUTPUT VARIABLE IS A "0" FOR BOTH STANDARD MINTERM COVERED BY MEV MAP SQUARE, THEN A "0" IS WRITTEN IN THAT MEV MAP SQUARE.***

STEP:2

✓ ***IF THE OUTPUT VARIABLE IS "1" FOR BOTH STANDARD MINTERM COVERED BY MEV MAP SQUARE, THEN A "1" IS WRITTEN IN THAT MEV MAP SQUARE.***

STEP:3

✓ ***IF FOR THE MINTERMS COVERED BY THE MEV MAP SQUARE , THE OUTPUT VARIABLE HAS THE SAME VALUE AS THE MEV.***

STEP:4

✓ IF FOR THE STANDARD MINTERM COVERED BY A MEV MAP SQUARE, THE OUTPUT & MEV VARIABLES ARE COMPLIMENTS WRITE THE MEV COMPLEMENT INTO MEV MAP SQUARE.

STEP:5

✓ IF FOR STANDARD MINTERMS COVERED BY A MEV MAP SQUARE , THE OUTPUT VARIABLE IS A DON'T CARE TERM, WRITE A DON'T CARE SYMBOL " x" INTO THE MEV MAP SQUARE.

STEP:6

✓ IF FOR STANDARD MINTERMS COVERED BY A MEV MAP SQUARE , THE OUTPUT VARIABLE IS A DON'T CARE TERM IN 1 CASE & "0" IN THE OTHER, WRITE "0"

✓ IF THE OUTPUT VARIABLE IS A DON'T CARE TERM IN 1 CASE & "1" IN THE OTHER, WRITE "1"

TO READ THE SIMPLIFIED FUNCTION FROM A MEV K-MAP, FOLLOW THESE STEPS:

STEP:1

✓ DETERMINE THE EPIs CONSISTING OF ONLY 1s ALONG WITH ANY DON'T CARE TERMS THAT MAY EXIST.{ i.e. COVER THE 1s IN K-MAP.}

STEP:2

✓ CONSIDER THE 1s AS DON'T CARE TERMS ONCE STEP 1 IS COMPLETED, BECAUSE ALL OF THE 1s HAVE BEEN PREVIOUSLY COVERED.

STEP:3

✓ GROUP ALL THE IDENTICAL MEV TERMS WITH 1s OR DON'T CARE TERMS TO MAXIMIZE THE MEV EPI SIZE.

✓ ANY MINTERM THAT ARE NOT CONTAINED IN THE MEV EPI ARE CONSIDERED TO BE 0s.{ i.e. COVER ALL THE MEVs IN THE K-MAP }

STEP:4

✓ DETERMINE THE MEV EPIs BY READING THE K-MAP IN THE NORMAL FASHION.

✓ THEN " AND " THE MEV VARIABLE OR EXPRESSION WITH THE REMAINING MAP VARIABLES.

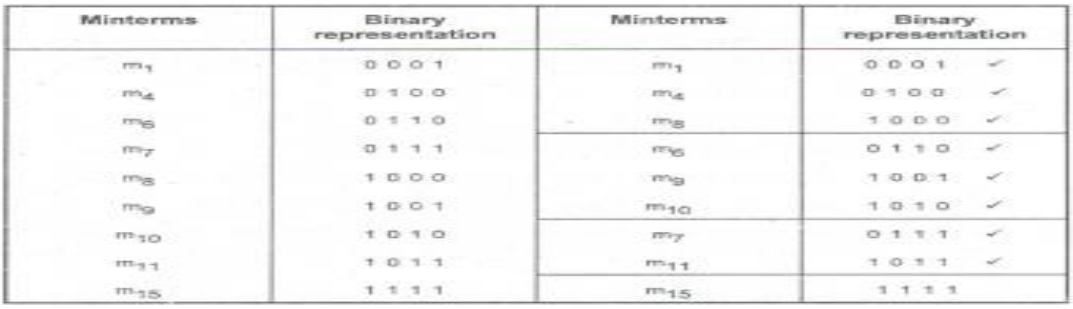
Recommended question and answer –unit-2

Jan-2009

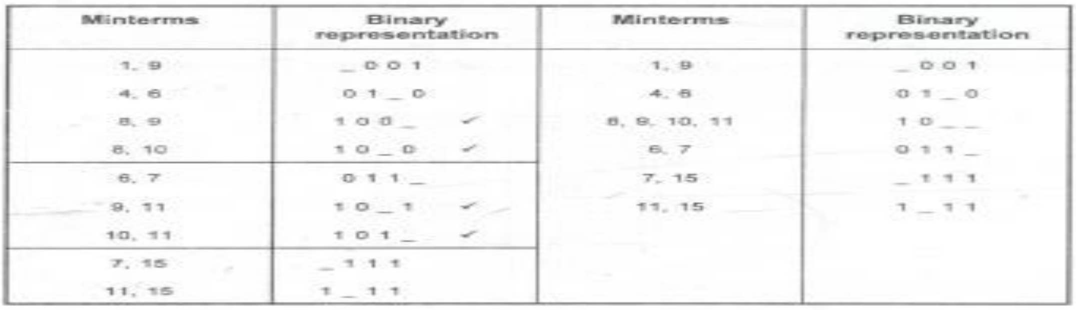
Q.2 a) *Using Quine-Mcluskey method and prime implicant reduction table, obtain the minimal sum expression for the Boolean function*

F(w, x, y, z) = L *mO,* 4, 6, 7, 8,*9, 10,* 11, 15). .
(12)

f(w, X, y, z) = *L* m (I, 4, 6, 7, 8, 9, 10, 11, 15)



Minterms	Binary representation	Minterms	Binary representation
m_1	0 0 0 1	m_1	0 0 0 1 ✓
m_4	0 1 0 0	m_4	0 1 0 0 ✓
m_6	0 1 1 0	m_8	1 0 0 0 ✓
m_7	0 1 1 1	m_6	0 1 1 0 ✓
m_8	1 0 0 0	m_9	1 0 0 1 ✓
m_9	1 0 0 1	m_{10}	1 0 1 0 ✓
m_{10}	1 0 1 0	m_7	0 1 1 1 ✓
m_{11}	1 0 1 1	m_{11}	1 0 1 1 ✓
m_{15}	1 1 1 1	m_{15}	1 1 1 1



Minterms	Binary representation	Minterms	Binary representation
1, 9	_ 0 0 1	1, 9	_ 0 0 1
4, 6	0 1 _ 0	4, 6	0 1 _ 0
8, 9	1 0 0 _ ✓	8, 9, 10, 11	1 0 _ _
8, 10	1 0 _ 0 ✓	6, 7	0 1 1 _
6, 7	0 1 1 _	7, 15	_ 1 1 1
9, 11	1 0 _ 1 ✓	11, 15	1 _ 1 1
10, 11	1 0 1 _ ✓		
7, 15	_ 1 1 1		
11, 15	1 _ 1 1		

:. f(w, x, y, z) = x y z + W x Z + W x + W x Y + x Y z + W Y z

b) *Obtain the minimal product of the following Boolean functions using VEM technique:*

f(w, x, y, z) = *L m* (1, 5, 7, 10, 11) + *dc* (2, 3, 6, 13) **.** **(8)**

Ans. : f(w, x, y, z) = *L* m (1, 5, 7, 10, 11) + dc (2, 3, ,6, 13)

Writing these rninterrns in SOP forills we get,

$$f(w, x, y, z) = \bar{w}\,\bar{x}\,\bar{y}\,z + \bar{w}\,\bar{x}\,y\,\bar{z} + \bar{w}\,\bar{x}\,y\,z + \bar{w}\,x\,\bar{y}\,z + \bar{w}\,x\,y\,\bar{z} + \bar{w}\,x\,y\,z$$
$$+\, w\,\bar{x}\,y\,\bar{z} + w\,\bar{x}\,y\,z + w\,x\,\bar{y}\,z$$

Now converting 4-variable terill into 3-variable terill we get,

f(w, x, y, z) = illa z + ill1 Z + illl Z + ill2 Z + ill3 Z + ill3 Z + illS Z + illS Z + ill6 Z

= illa Z + illl (z + z) + ill2 Z + ill3 (z + z) + illS (z + z) + ill6'Z

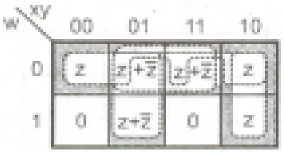


Fig. 2

Applying grouping technique we get,

f(w, x, y, z) = w Z + X Y + w Y + x y Z

Jan-2008

c. Simplify P = f(a, b, c) = L (0,1, 4, 5, 7) using two variable Karnaugh map. Write the Boolean equation and realize using logic gates (8)

Ans. : P = f(a, b, c) = Σ (0, 1, 4, 5, 7)

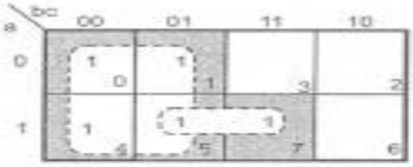
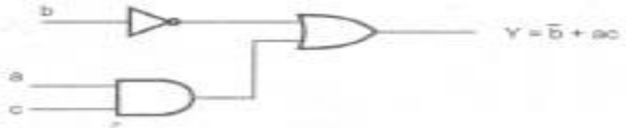


Fig. 3

Y = $\bar{b}$ + ac

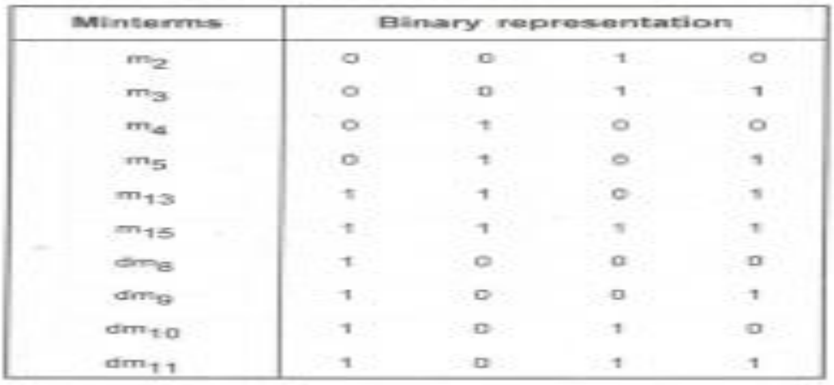


b) *Simplify using Quine* Mc *Cluskey tabulation algorithm -*

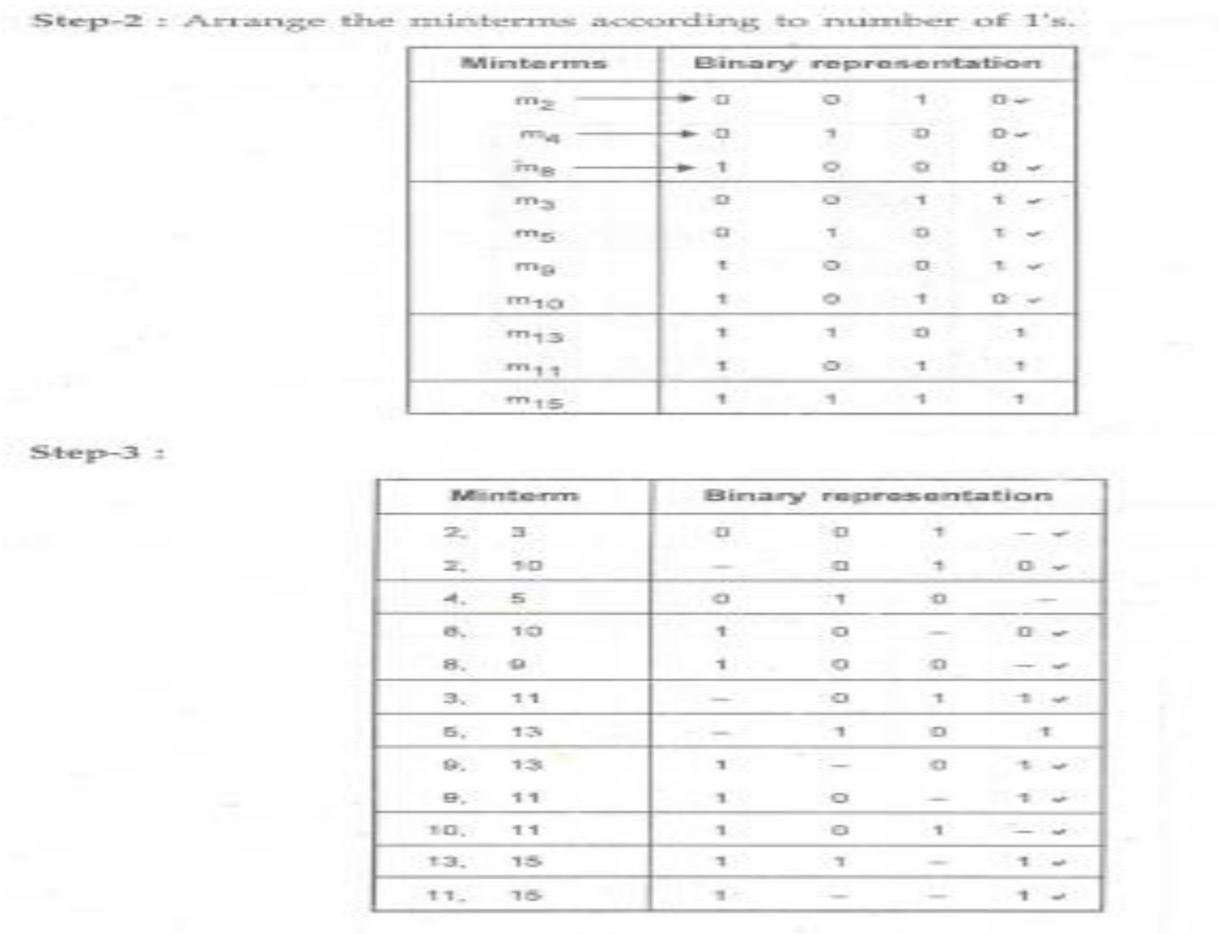
v = lea, b, c, *d) = L* (2, 3, 4, 5, 13, 15) + *L d* (8, 9, *10, 11)*
(14)

Ans:

Step-1 : List all minterms in binary form.



Minterms	Binary representation			
m_2	0	0	1	0
m_3	0	0	1	1
m_4	0	1	0	0
m_5	0	1	0	1
m_{13}	1	1	0	1
m_{15}	1	1	1	1
dm_8	1	0	0	0
dm_9	1	0	0	1
dm_{10}	1	0	1	0
dm_{11}	1	0	1	1

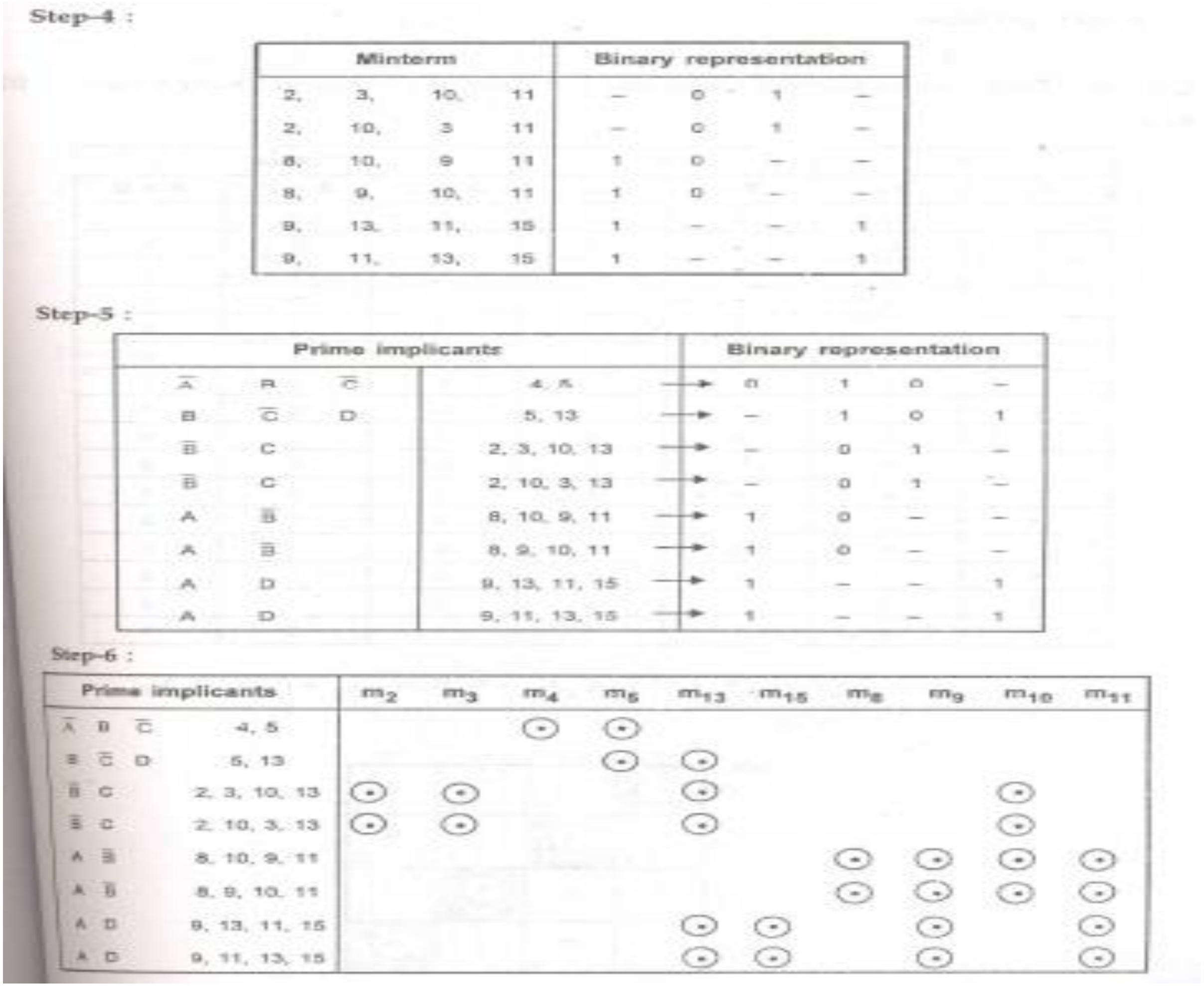


Step-2 : Arrange the minterms according to number of 1's.

Minterms	Binary representation			
m_2 →	0	0	1	0 ✓
m_4 →	0	1	0	0 ✓
m_8 →	1	0	0	0 ✓
m_3	0	0	1	1 ✓
m_5	0	1	0	1 ✓
m_9	1	0	0	1 ✓
m_{10}	1	0	1	0 ✓
m_{13}	1	1	0	1
m_{11}	1	0	1	1
m_{15}	1	1	1	1

Step-3 :

Minterm	Binary representation			
2, 3	0	0	1	— ✓
2, 10	—	0	1	0 ✓
4, 5	0	1	0	—
8, 10	1	0	—	0 ✓
8, 9	1	0	0	— ✓
3, 11	—	0	1	1 ✓
5, 13	—	1	0	1
9, 13	1	—	0	1 ✓
9, 11	1	0	—	1 ✓
10, 11	1	0	1	— ✓
13, 15	1	1	—	1 ✓
11, 15	1	—	—	1 ✓



Step-4 :

Minterm				Binary representation			
2,	3,	10,	11	–	0	1	–
2,	10,	3	11	–	0	1	–
8,	10,	9	11	1	0	–	–
8,	9,	10,	11	1	0	–	–
9,	13,	11,	15	1	–	–	1
9,	11,	13,	15	1	–	–	1

Step-5 :

Prime implicants					Binary representation			
$\overline{A}$	B	$\overline{C}$	4, 5	→	0	1	0	–
B	$\overline{C}$	D	5, 13	→	–	1	0	1
$\overline{B}$	C		2, 3, 10, 13	→	–	0	1	–
$\overline{B}$	C		2, 10, 3, 13	→	–	0	1	–
A	$\overline{B}$		8, 10, 9, 11	→	1	0	–	–
A	$\overline{B}$		8, 9, 10, 11	→	1	0	–	–
A	D		9, 13, 11, 15	→	1	–	–	1
A	D		9, 11, 13, 15	→	1	–	–	1

Step-6 :

Prime implicants		m_2	m_3	m_4	m_5	m_{13}	m_{15}	m_8	m_9	m_{10}	m_{11}
$\overline{A}$ B $\overline{C}$	4, 5			⊙	⊙						
B $\overline{C}$ D	5, 13				⊙	⊙					
$\overline{B}$ C	2, 3, 10, 13	⊙	⊙			⊙				⊙	
$\overline{B}$ C	2, 10, 3, 13	⊙	⊙			⊙				⊙	
A $\overline{B}$	8, 10, 9, 11							⊙	⊙	⊙	⊙
A $\overline{B}$	8, 9, 10, 11							⊙	⊙	⊙	⊙
A D	9, 13, 11, 15					⊙	⊙		⊙		⊙
A D	9, 11, 13, 15					⊙	⊙		⊙		⊙

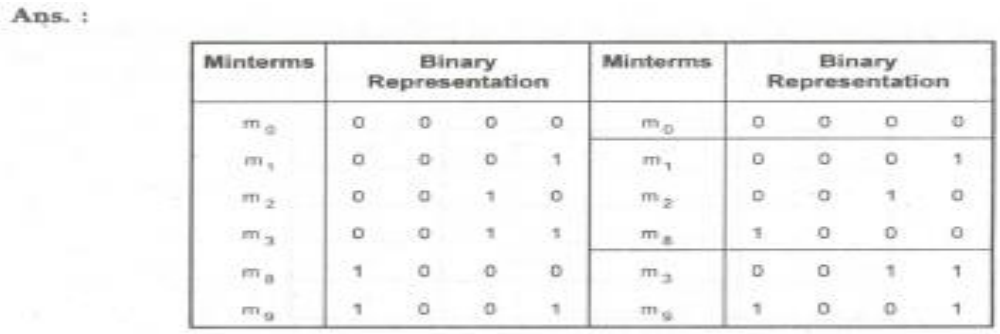
Final expression

Y= AbC + bCd+ Bc +a B+ aD

<u>Aug-2009</u>

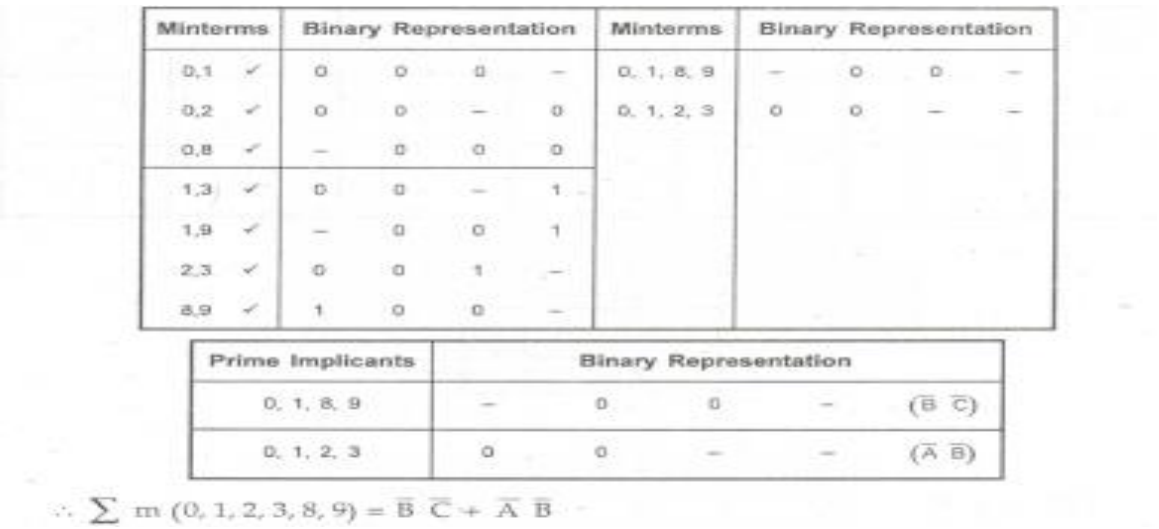
Q.2 a) *Using Quine McCluskey method simplify the following function.*

f (a, b, c, *d) = Lm (0*, 1, 2, 3, 8, 9)



Ans. :

Minterms	Binary Representation				Minterms	Binary Representation			
m_0	0	0	0	0	m_0	0	0	0	0
m_1	0	0	0	1	m_1	0	0	0	1
m_2	0	0	1	0	m_2	0	0	1	0
m_3	0	0	1	1	m_8	1	0	0	0
m_8	1	0	0	0	m_3	0	0	1	1
m_9	1	0	0	1	m_9	1	0	0	1



Minterms	Binary Representation				Minterms	Binary Representation			
0,1 ✓	0	0	0	–	0, 1, 8, 9	–	0	0	–
0,2 ✓	0	0	–	0	0, 1, 2, 3	0	0	–	–
0,8 ✓	–	0	0	0					
1,3 ✓	0	0	–	1					
1,9 ✓	–	0	0	1					
2,3 ✓	0	0	1	–					
8,9 ✓	1	0	0	–					

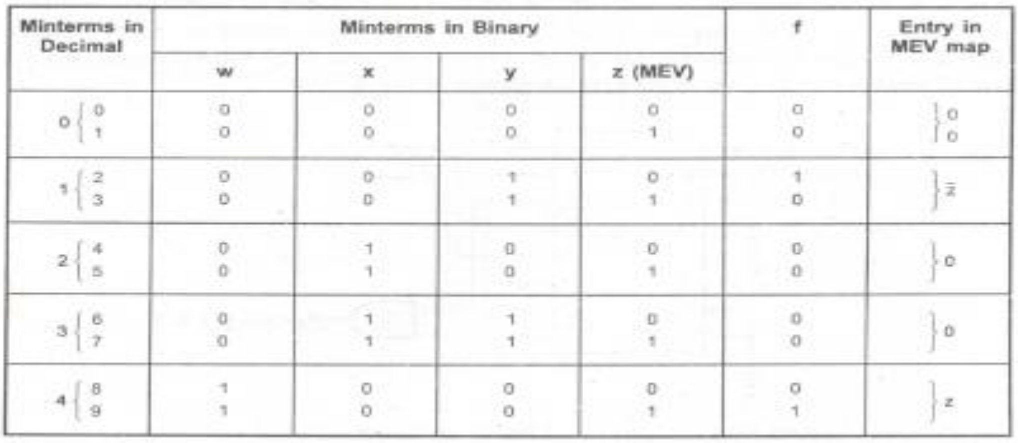
Prime Implicants	Binary Representation				
0, 1, 8, 9	–	0	0	–	$(\bar{B}\ \bar{C})$
0, 1, 2, 3	0	0	–	–	$(\bar{A}\ \bar{B})$

$$\therefore \sum m\,(0, 1, 2, 3, 8, 9) = \bar{B}\ \bar{C} + \bar{A}\ \bar{B}$$

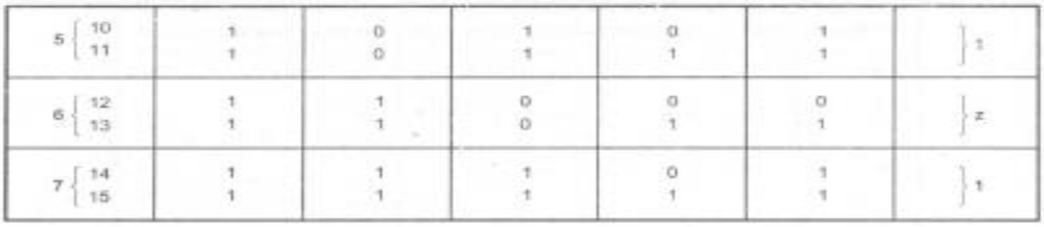
b) *Write the map entered variable K-map for the Boolean function.*

f(w~ X, y, z) = *Lm(2*, 9, *10*, 11, 13, 14, 15)

f(w, X, y, z) = *Lm(2*, 9, la, 11, 13, 14, 15)



Minterms in Decimal	Minterms in Binary				f	Entry in MEV map
	w	x	y	z (MEV)		
0 {0 1	0 0	0 0	0 0	0 1	0 0	}0 0
1 {2 3	0 0	0 0	1 1	0 1	1 0	} $\bar{z}$
2 {4 5	0 0	1 1	0 0	0 1	0 0	} 0
3 {6 7	0 0	1 1	1 1	0 1	0 0	} 0
4 {8 9	1 1	0 0	0 0	0 1	0 1	} z



5 { 10 11	1 1	0 0	1 1	0 1	1 1	} 1
6 { 12 13	1 1	1 1	0 0	0 1	0 1	} z
7 { 14 15	1 1	1 1	1 1	0 1	1 1	} 1

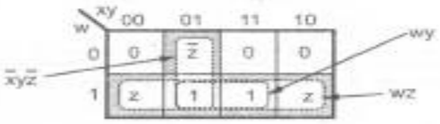


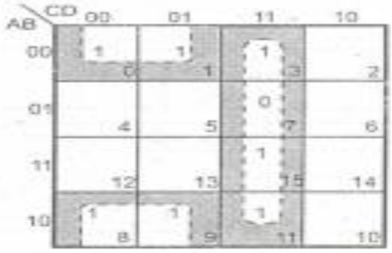
Fig. 5

$\therefore$ f (w, x, y, z) = $\bar{x}\,y\,\bar{z}$ + wy + wz

Aug-2008

Q.2a) *Simplify the logic function given below, using Quine-McCluskey technique. Y(ABCD) = L m (0,* 1, 3, 7, 8, 9, 11, 15). *Realize the expression using universal gates.*

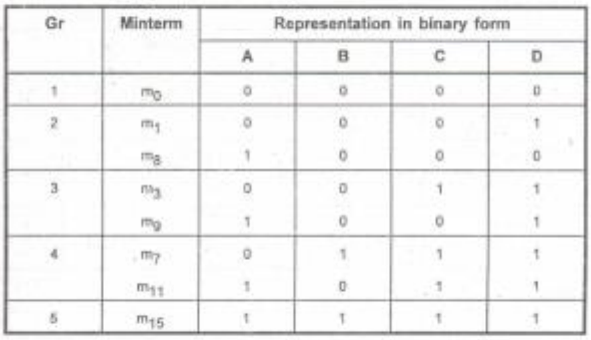
Y(ABCD) = Σ m (0, 1, 3, 7, 8, 9, 11, 15)



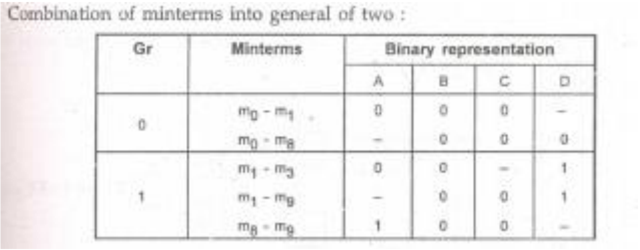
Y = CD + $\bar{B}\,\bar{C}$

Fig. 3 (a)

Simplification using Quine-McCluskey method :

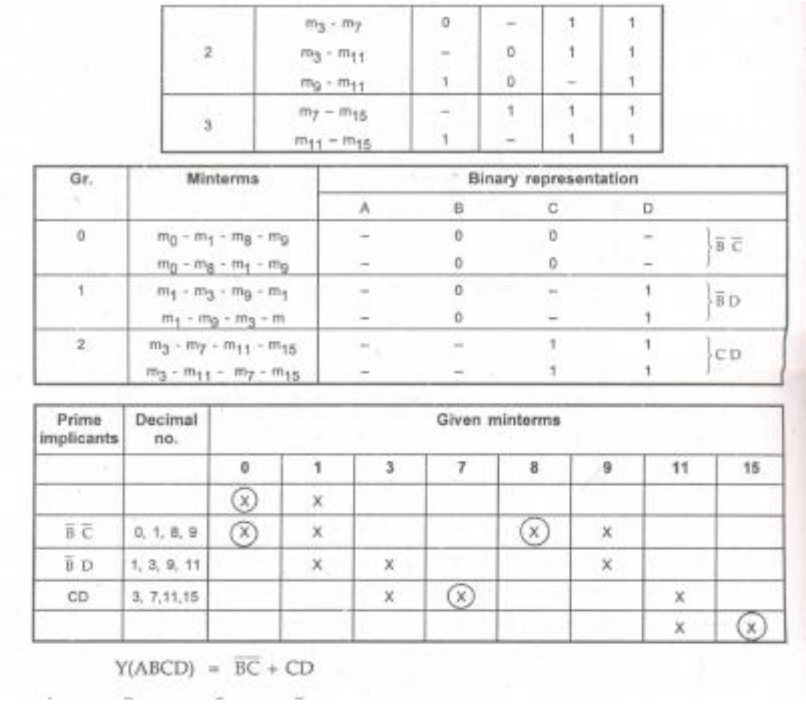


Gr	Minterm	Representation in binary form			
		A	B	C	D
1	m_0	0	0	0	0
2	m_1	0	0	0	1
	m_8	1	0	0	0
3	m_3	0	0	1	1
	m_9	1	0	0	1
4	m_7	0	1	1	1
	m_{11}	1	0	1	1
5	m_{15}	1	1	1	1



Combination of minterms into general of two :

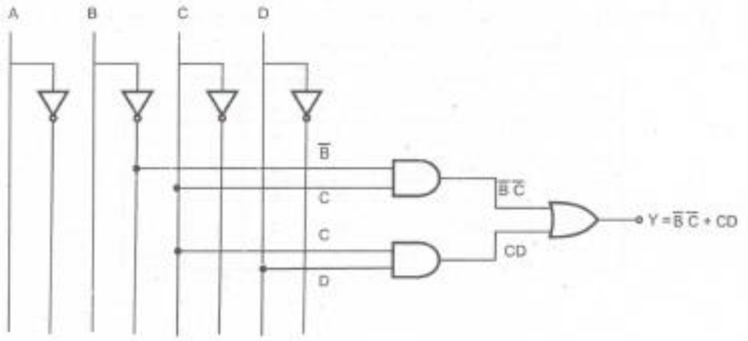
Gr	Minterms	Binary representation			
		A	B	C	D
0	$m_0 - m_1$	0	0	0	–
	$m_0 - m_8$	–	0	0	0
1	$m_1 - m_3$	0	0	–	1
	$m_1 - m_9$	–	0	0	1
	$m_8 - m_9$	1	0	0	–
2	$m_3 - m_7$	0	–	1	1
	$m_3 - m_{11}$	–	0	1	1
	$m_9 - m_{11}$	1	0	–	1
3	$m_7 - m_{15}$	–	1	1	1
	$m_{11} - m_{15}$	1	–	1	1



Gr.	Minterms	Binary representation				
		A	B	C	D	
0	$m_0 - m_1 - m_8 - m_9$	–	0	0	–	$\bar{B}\,\bar{C}$
	$m_0 - m_8 - m_1 - m_9$	–	0	0	–	
1	$m_1 - m_3 - m_9 - m_1$	–	0	–	1	$\bar{B}\,D$
	$m_1 - m_9 - m_3 - m$	–	0	–	1	
2	$m_3 - m_7 - m_{11} - m_{15}$	–	–	1	1	$C\,D$
	$m_3 - m_{11} - m_7 - m_{15}$	–	–	1	1	

Prime implicants	Decimal no.	Given minterms							
		0	1	3	7	8	9	11	15
		⊗	X						
$\bar{B}\,\bar{C}$	0, 1, 8, 9	⊗	X			⊗	X		
$\bar{B}\,D$	1, 3, 9, 11		X	X			X		
CD	3, 7,11,15			X	⊗			X	
								X	⊗

$$Y(ABCD) = \overline{BC} + CD$$

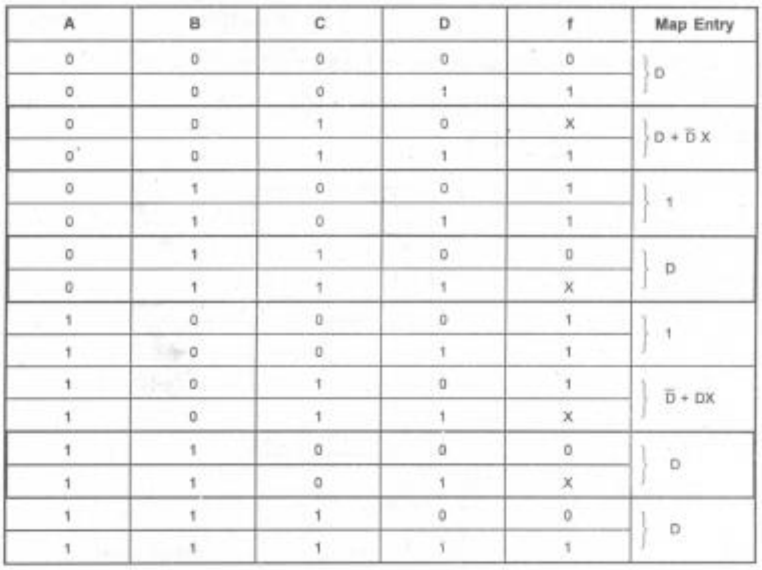


b) *Simplify the logic function given below using variable - entered mapping (VEM) technique.* Y *(ABeD) = L m* (1, 3, 4, 5, 8, 9, *10,* 15) + *L d* (2, 7, 11, 12, 13). **(8)**

Ans. :

1. Use A, B, C as ordinary K-map variable

2. Make D the map-entered variable



A	B	C	D	f	Map Entry
0	0	0	0	0	D
0	0	0	1	1	
0	0	1	0	X	D + D̄ X
0	0	1	1	1	
0	1	0	0	1	1
0	1	0	1	1	
0	1	1	0	0	D
0	1	1	1	X	
1	0	0	0	1	1
1	0	0	1	1	
1	0	1	0	1	D̄ + DX
1	0	1	1	X	
1	1	0	0	0	D
1	1	0	1	X	
1	1	1	0	0	D
1	1	1	1	1	

3. Now using VEM, make the maps :

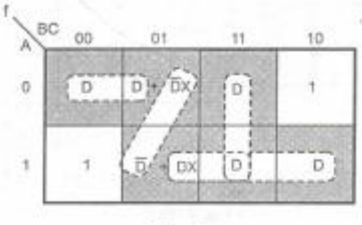


Fig. 4

$$f = \bar{A}\,\bar{B} + BC + AB + AC + \bar{B}C$$
$$= \bar{A}\bar{B} + AB + BC + \bar{B}C + AC$$
$$f = \bar{A}\bar{B} + AB + C + (B + \bar{B}) + AC$$
$$f = \bar{A}\bar{B} + AB + C + AC$$
$$= \bar{A}\bar{B} + AB + C\,(1 + A)$$
$$f = \bar{A}\bar{B} + AB + C$$