MODULE 1

Principles of Combinational logic

Structure

1.1 Objective

1.2 Introduction

1.3 Review of Boolean Algebra.

1.4 Definition of combinational

1.5 Canonical forms

1.6 Generation of switching equations from truth tables

1.7 Karnaugh maps-3, 4 and 5 variables. Incompletely specified functions (Don't care terms).

1.8 Simplifying max - term equations.

1.9 Quine -McClusky minimization technique, Quine - McClusky using don't care terms,

1.10 Reduced Prime Implicant tables,

1.11 Map entered variables.

1.12 Outcome

1.13 Future Readings

1.1 Objevtive

- **Student at the end will be able represent a expression from TT or vice versa**
- **Different types of reducing a expression so that the Boolean expression obtained will have minimum variables which in tern help in reducing the component size.**
- **Advantages and disadvantages od reduction technique.**

1.2 Introduction

Logic Circuits are categorized into 2 types (based on whether they contain memory or not):

- Combinational Logic Circuits- Circuits without memory
- Sequential Logic Circuits- Circuits with memory

1.3 Review of Boolean Algebra.

Example 1:

xy+xy+x y = x+y

LHS x(y+y)+ x y using distributive law

x •1+ x y using y+y = 1 theorem
x(1+ y) +x y (1+y) =1 theorem
x+ x y + x y using distributive law
x + y(x+ x)
x + y• 1 = x +y RHS

Example 2:
(x+y)(x+z) = xz+xy using distributive law
x x + x z + x y + y z using P5 postulate
x z + x y +(x+x) yz using distributive law
xz+ x y + xyz+xyz using distributive law
xz(1+y) +x y(1+z) using T2 theorem
xz•1 +xy • 1 = xz + xy RHS

PRINCIPLE OF DUALITY

One can transform the given expression by interchanging the operation (+) and (•) as well as the identity elements 0 and 1 . Then the expression will be referred as dual of each other. This is known as the principle of duality.

Example x + x = 1 then the dual expression is

x • x = 0

BOOLEAN FORMULAS AND FUNCTIONS

Boolean expressions or formulas are constructed by using Boolean constants and variables with the Boolean operations like (+) , (•) and 'not'

Example: (x + y) z

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

x y z	x y	x z	x y z	f
0 0 0	0	0	0	0
0 0 1	0	0	0	0
0 1 0	0	0	0	0
0 1 1	0	0	0	0
1 0 0	0	1	0	1
1 0 1	0	0	0	0
1 1 0	1	1	1	1
1 1 1	1	0	0	1

Truth table for the above Boolean expression is

Example 2: Write a truth table for following function

f = x y z + x y + x z

x y z	x y	x z	x y z	f
0 0 0	0	0	0	0
0 0 1	0	0	0	0
0 1 0	0	0	0	0
0 1 1	0	0	0	0
1 0 0	0	1	0	1
1 0 1	0	0	0	0
1 1 0	1	1	1	1
1 1 1	1	0	0	1

1.4 Definition of combinational

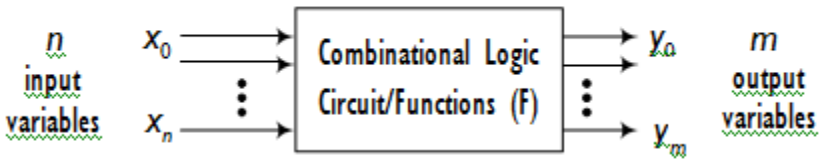
1. 4.1 COMBINATIONAL NETWORK

The inter connections 0f gates result in a gate network. If the network has the property that its outputs at any time are determined strictly by the inputs at that time then the network is said to be a combinational network. Ex. Adders. Multiplexers. etc

Let us consider an set of n signals at any time is called input state or input vector of the network. While a set of resulting signals appearing at the m output terminals is called the output state or output vector. The network can be expressed as

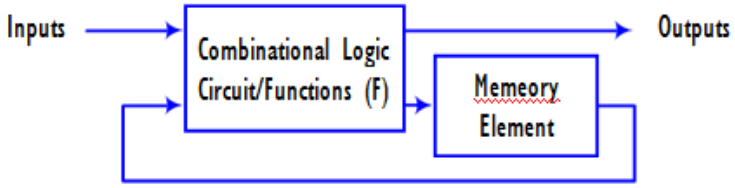
z1, z2,……zm as Boolean function then

Zi = fi(x1,x2,….xn) for i= 1,2,…m.



1.4.2. SEQUENTIAL NETWORKS

A second type of logic network is the sequential networks. Sequential network have memory property, so that the the outputs from these networks are dependent not only upon the current inputs but upon previous input as well. Feed back path are used in the sequential circuits. Ex. Counters. Shift registers etc.



1.4.3 ANALYSIS PROCEDURE

Analysis procedure for a combinational network is as follows

1. Each gate output that is only a function of the input variables is labeled.

2. Boolean algebraic expression for the outputs of each of these gates are then written.

3. Next these gates outputs that are a function of just inputs variables and previously labeled gate outputs.

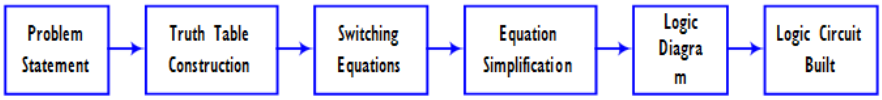
4. Then each of the previously defined labels is replaced by the already written Boolean expression and this process is continued until the output of the network is labeled and till final expression is obtained.

$f(w,x,y,z) = w•(y+z) + wxy$

$= w•G1 + G2$

$f(w,x,y,z) = G2 + G3$

•General Procedure



NORMAL FORMULAS

- **Boolean expression can be represented by following structures**

1. Sum of products (SOP or disjunctive normal form

2. Product of sum (POS or Conjunctive form

•In **SOP** normal form is a Boolean formula that is written as a single product term or as a sum (also called disjunctive) of product terms is said to be in the sum of product form or disjunctive normal form.

Example:

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

In the **POS** normal form is a Boolean formula which is written as a single sum term or as a product of sum (also called conjunctive) terms is said to be in product of sums form or conjunctive normal form.

Example:

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

1.5 Canonical forms

A procedure which will be used to write Boolean expressions form truth table is known as canonical formula. The canonical formulas are of two types

- **Minterm canonical formulas**
- **Maxterm canonical formulas**

1. 5.1 MINTERM CANONICAL FORMULAS

Minterms are product of terms which represents the functional values of the variables appear either in complemented or un complemented form.

Ex: **f(x,y,z) = x y z + x y z + x y z**

The Boolean expression which is represented above is also known as SOP or disjunctive formula

The truth table is

x y z	f
0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	0 1 0 1 1 0 0 0

m- NOTATION

To simplify the writing of a minterm in canonical formula for a function is performed using the symbol **mi.** Where **i** stands for the row number for which the function evaluates to 1.

The m-notation for 3- variable an function Boolean function

f(x,y,z) = x y z + x y z + x y z is written as

f(x,y,z) = m1+ m3 + m4 or

f(x,y,z) = ∑m(1,3,4)

A three variable m- notation truth variable

x y z	Decimal designator of row	Minterm	m-notation
0 0 0	0	x y z	m0
0 0 1	1	x y z	m1
0 1 0	2	x y z	m2
0 1 1	3	x y z	m3
1 0 0	4	x y z	m4
1 0 1	5	x y z	m5
1 1 0	6	x y z	m6
1 1 1	7	x y z	m7

1.5.2. MAXTERM CANONICAL FORM

Maxterm are sum terms where the variable appear once either in complement or un-complement
forms and these terms corresponds to a functional value representing 0.

Ex. f(x,y,z) = (x+ y+ z) (x+ y+z) (x + y + z)
= ∏M(0, 2, 5)
= M0, M2, M5

M-NOTATION

A maxterm in a canonical form can be represented as Mi. Where i stands for row number for which the the function evaluates to 0. A product of maxterms are represented as ΠM.

1.5.3 . MINTERM CANONICAL FORMULA

1. Apply DeMorgan's law a sufficient no. of times until all the NOT operations appear only with the single variables.

2. Apply distributive of AND over OR (•) over (+) i.e. x•(y+z) = xy+xz in order to manipulate the formula into disjunctive normal formula.

3. Remove duplicate literals and turns by idempotent law well as any term that are identically zero (x•x = 0)

4. If any product in the disjunctive normal formula does not contain all the variables of the Boolean function then these missing variables are introduced by ANDing the terms with logic 1 in the form of xi+xi where xi is the missing variable being introduced. This process is repeated for each missing variables in each of the product turns of the disjunctive normal form

5. Apply distributive of (•) over (+) again so that each variable appears exactly once in each term.

6. Remove duplicate term if any.

Example : (x + y) + (y + x z)(x + y)

1.5.4 . MAXTERM CANONICAL FORMULA

1. Apply DeMorgan's law until all the NOT operations appear with single variables.

2. Apply distributive law of (+) over (•) i.e. (x+yz) = (x+y)(x+z) and bring the expressions into its conjugate normal form (POS).

3. The missing variables are introduced into the sum terms by OR ing logic 0's in the form xi •xi = 0 where xi is missing variable.

4. Distributive law of (+) over (•) is again applied.

5. Duplicate literals are deleted.

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

1.5.6. COMPEMENTS OF CANONICAL FORMULAS

Even by taking complements minterm expression may result different expressions.

Ex : f(x,y,z) = ∑m(0, 2, 4, 6) its complement expression is

f(x,y,z) = ∑m(1, 3, 5,7)

Similarly

A Maxterm canonical expression may be represented in completed for as follows

Ex : f(x,y,z) = Π M (1, 2, 4, 7) its complement expression is

f(x,y,z) = Π M (0 3, 5,6)

1.6 Generation of switching equations from truth tables,

1.7 Karnaugh maps-3, 4 and 5 variables. Incompletely specified functions (Don't care terms).

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.

2 – Variable Karnaugh Map

Consider the Venn diagram for the two variables A and B.

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

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

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

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.

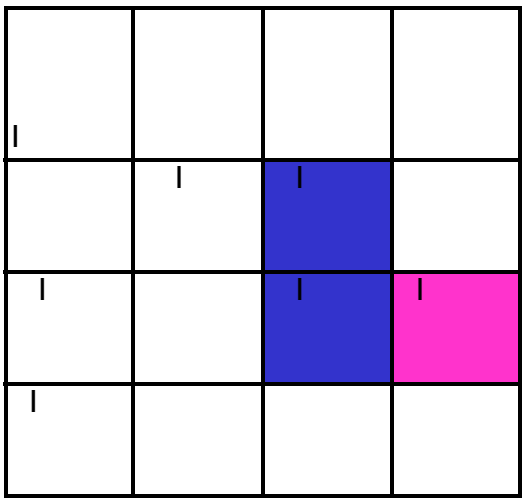
2. The construction of an n-variable map is such that any set of 1-cells or 0-cells which form a $2^a x 2^b$ rectangular grouping describing a product or sum term with n-a-b variables , where a and b are non-negative no.s

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

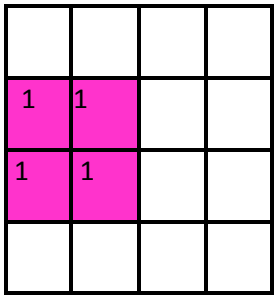
4. For three variable and four variable K-map it must be remembered that the edges are also adjacent cells or subcubes hence they will be grouped together.

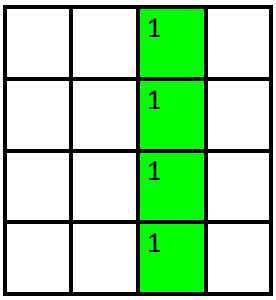
5. Given an n-variable map with a pair of adjacent 1-cells or 0-cellscan result n-1 variable.Where as if a group of four adjacent subcubes are formed than it can result n-2 variables. Finally if we have eight adjacent cells are grouped may result n-3 variable product or sum term.

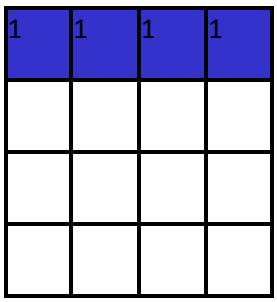
Typical pair of subcubes



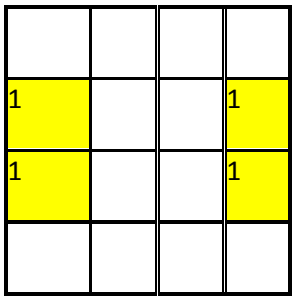
Typical group of four adjacent subcubes

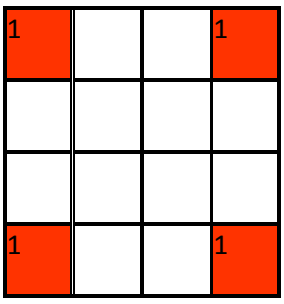




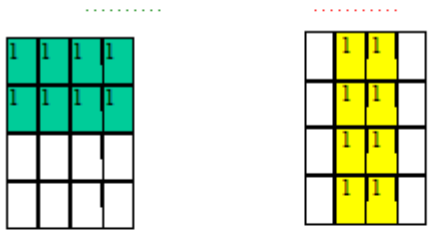


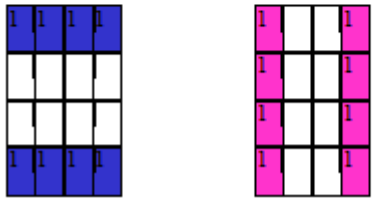
Typical group of four adjacent subcubes.



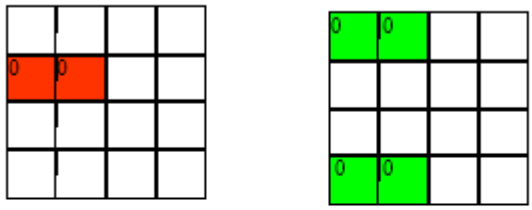


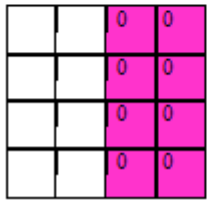
Typical group of Eight adjacent subcubes.





Typical map subcubes describing sum terms





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

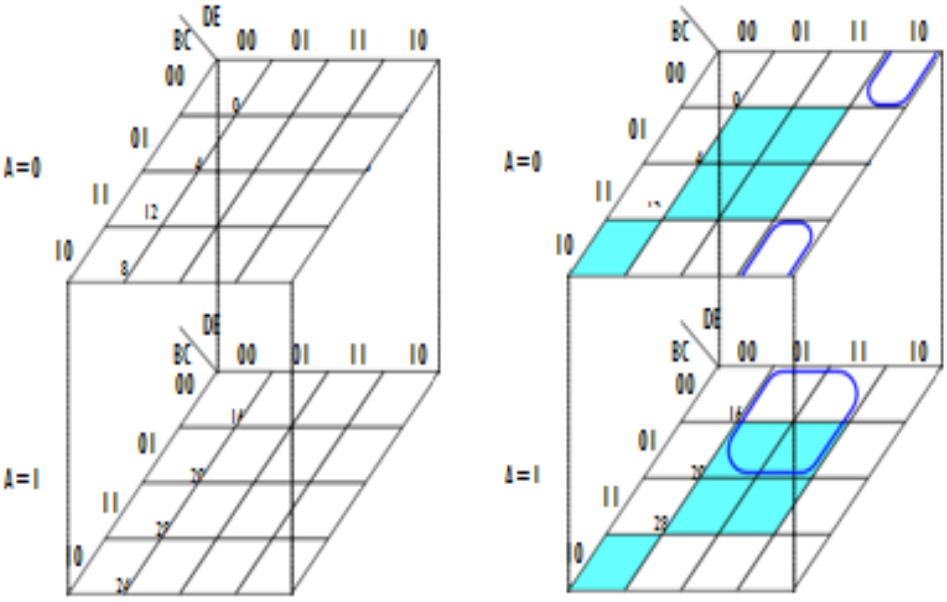
00 01 11 10

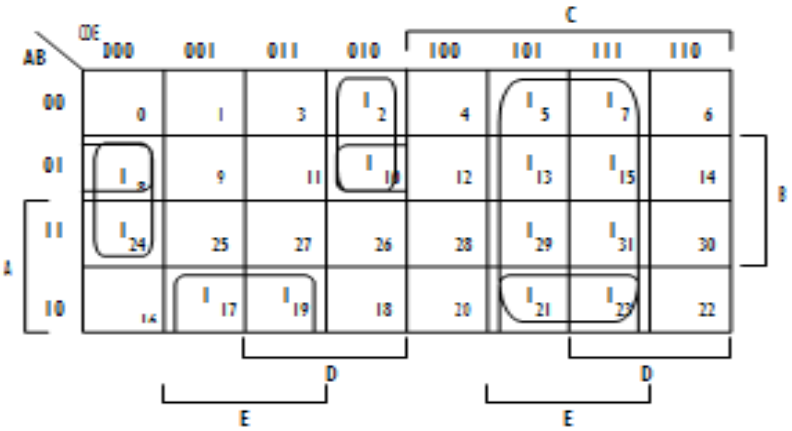
0	0	0	1
0	0	1	1

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

(I) K-Maps Revisited: 5- and 6-Variable Functions

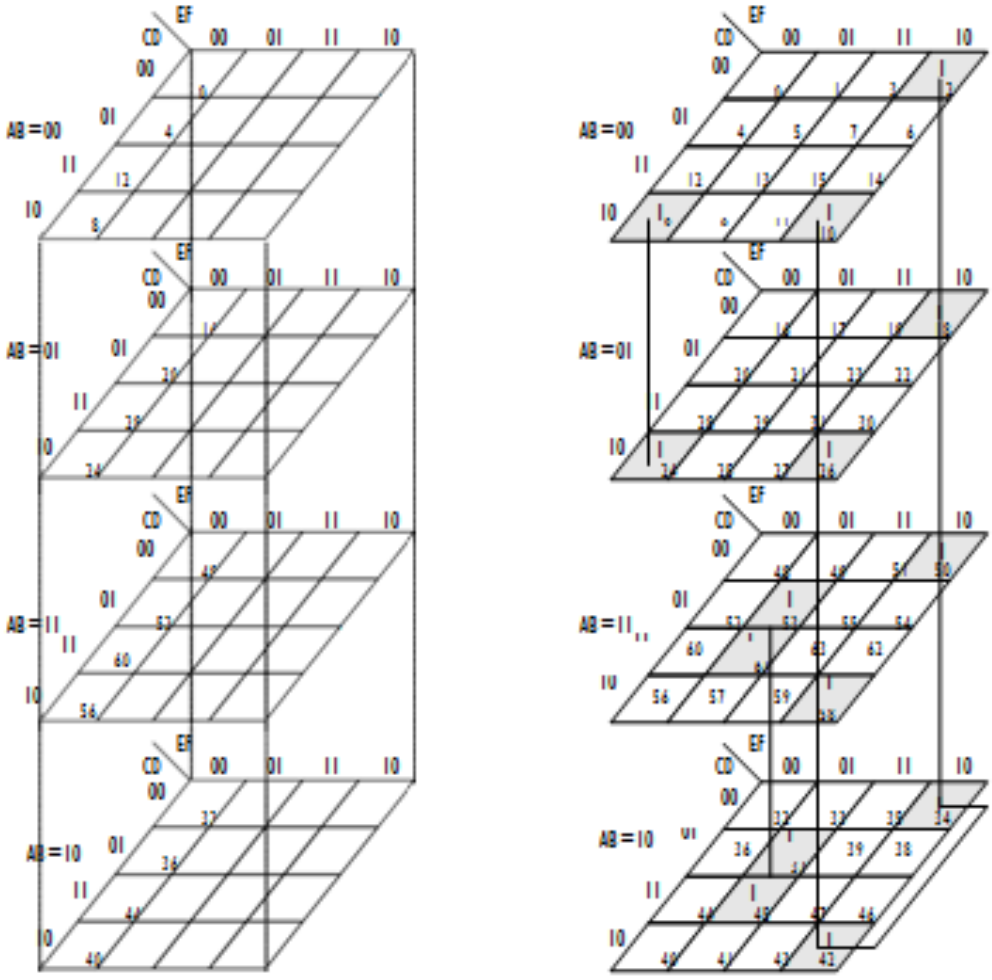
☐ Five-Variable K-Map



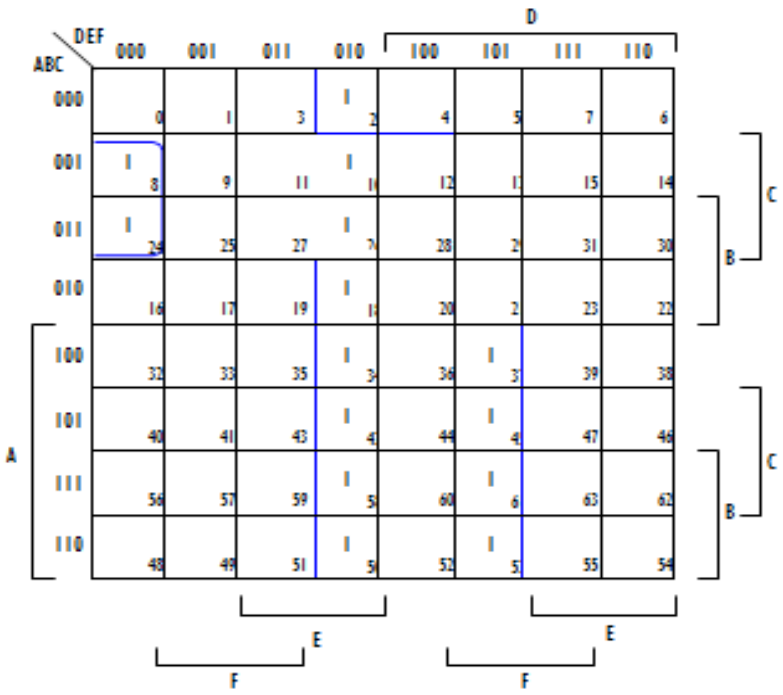


$$f(A,B,C,D,E) = \Sigma m(2,5,7,8,10,13,15,17,19,21,23,24,29,31)$$
$$= CE + BC'D'E' + AB'E + A'C'DE'$$

☐ Six-Variable K-Map



$$f(A,B,C,D,E,F) = \Sigma m(2,8,10,18,24,26,34,37,42,45,50,53,58,61)$$
$$= D'EF' + ADE'F + A'CD'F'$$



INCOMPLETE BOOLEAN FUNCTION WITH DONOT CARE CONDITIONS

In this type of Boolean functions having n-variables so that it may have 2n combinations of subsets and if all values are not specified such functions are called incompletely specified functions.

Ex. Let us consider a three variable function with following truth table We can describe the incomplete function in the SOP form as

f(x,y,z) = ∑m(0,1,7) +dc(3,5)

Also we can represent the function in the pos form as

f(x,y,z) = ΠM(2,4,6) +dc(3,5)

The odd parity generation result output expression

f(w,x,y,z) = ∑m(0,3,5,6,9) +dc(10,11,12,13,14,15)

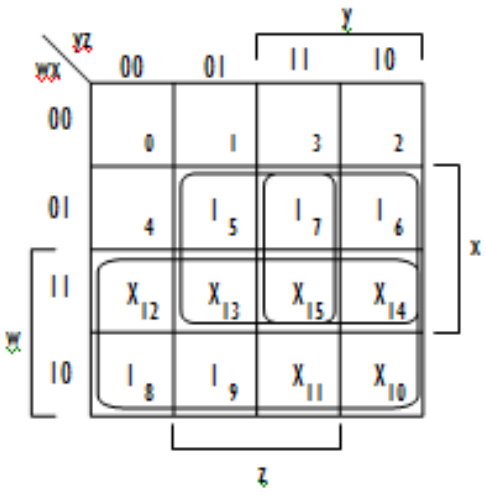
f(w,x,y,z)=w x y z + x y z + x y z + x y z + w z

(1) Incompletely Specified Functions (Don't Care Terms)

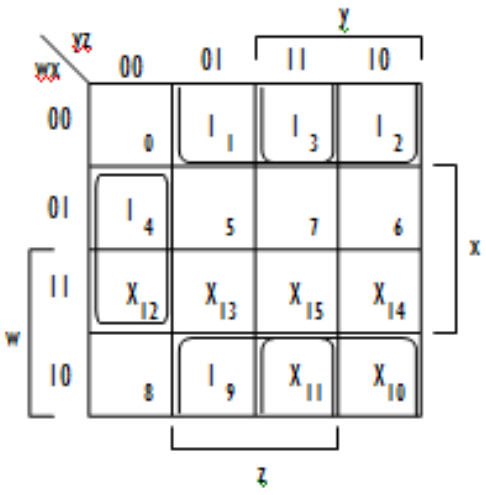
- Don't care: minterms or maxterms that are not used as part of the output
- Ex: Binary to EX-3 BCD code conversion

Binary				EX-3 BCD			
W	X	Y	Z	A	B	C	D
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

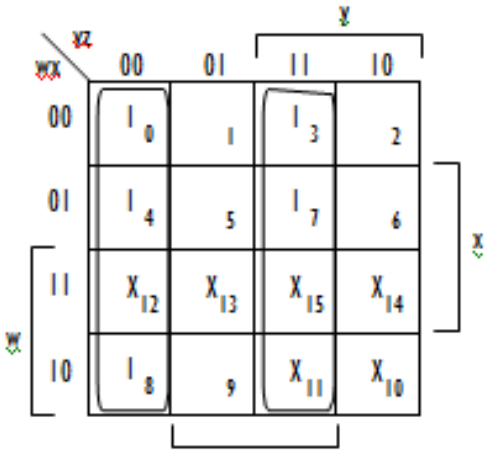
$$A = f(W,X,Y,Z) = \Sigma m(5,6,7,8,9) + \Sigma d(10,11,12,13,14,15)$$
$$B = f(W,X,Y,Z) = \Sigma m(1,2,3,4,9) + \Sigma d(10,11,12,13,14,15)$$
$$C = f(W,X,Y,Z) = \Sigma m(0,3,4,7,8) + \Sigma d(10,11,12,13,14,15)$$
$$D = f(W,X,Y,Z) = \Sigma m(0,2,4,6,8) + \Sigma d(10,11,12,13,14,15)$$



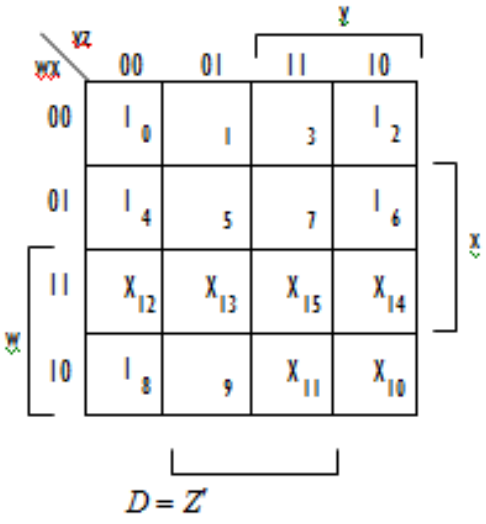
$A = W + XZ + XY$



$B = X'Y + X'Z + XY'Z'$



$C = Y'Z' + YZ'$



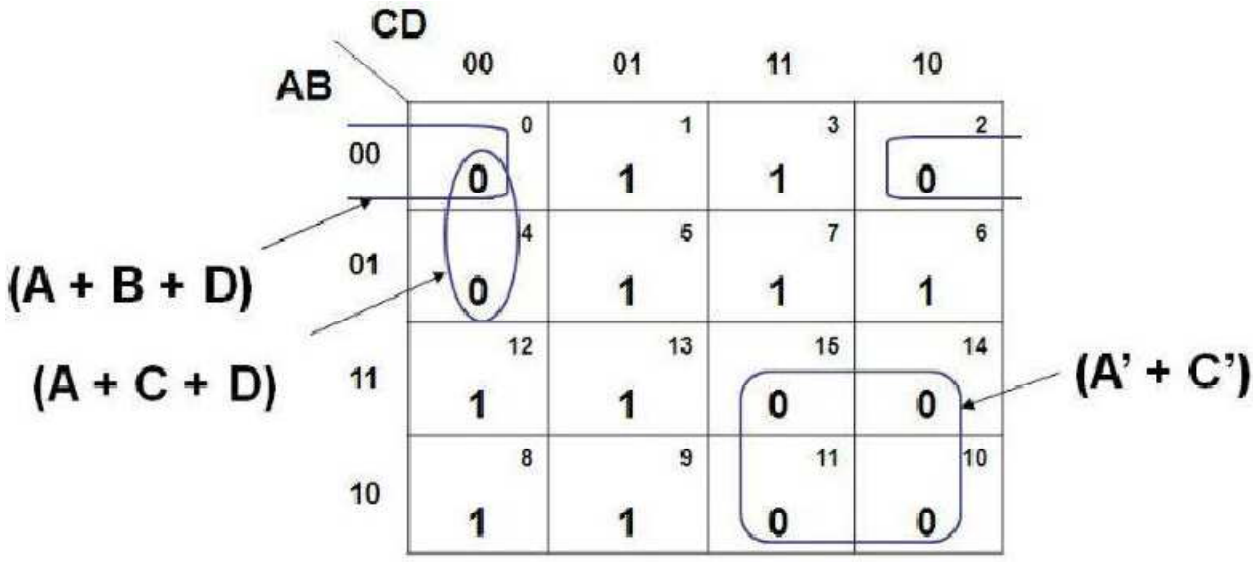
$D = Z'$

1.8 Simplifying max - term equations.

Reduce the following function using Karnaugh map technique

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

The K-map for the given logic function is drawn as shown below:



The simplified logic equation in POS form is Y = (A + B + D). (A + C + D). (A' + C').

1.9 Quine -McClusky minimization technique, Quine - McClusky using don't care terms,

Developed in the mid 1950s.

A systematic procedure for generating all prime implicants and extracting a minimum set of primes covering the on-set.

Ex. $F = f(A,B,C,D) = \Sigma(0,1,2,8,10,11,14,15)$

Step 1: Group binary representation of the minterms according to the number of 1's contained.

Step 2: Any two minterms which differ from each other by only one variable can be combined, and the unmatched variable removed. The minterms in one section are compared with those of the next section down only, because two terms differing by more than one bit cannot match.

Step 3: Repeat step 2.

Step 4:The unchecked terms in the table form the prime-implicants. Step 5: Prime-implicant table

- Determination of prime-implicant

Step 1		Step 2		Step 2	
	A B C D		*A B C D*		*A B C D*
0	0 0 0 0 ✓	0,1	0 0 0 -	0,2,8,10	- 0 - 0
1	0 0 0 1 ✓	0,2	0 0 - 0 ✓	0,8,2,10	- 0 - 0
2	0 0 1 0 ✓	0,8	- 0 0 0 ✓	10,11,14,15	1 - 1 -
8	1 0 0 0 ✓	2,10	- 0 1 0 ✓	10,14,11,15	1 - 1 -
10	1 0 1 0 ✓	8,10	1 0 - 0 ✓		
11	1 0 1 1 ✓	10,11	1 0 1 - ✓		
14	1 1 1 0 ✓	10,14	1 - 1 0 ✓		
15	1 1 1 1 ✓	11,15	1 - 1 1 ✓		
		14,15	1 1 1 - ✓		

- Prime-implicant table

	Minterms							
	0	1	2	8	10	11	14	15
✓0,1(000-)②	X	X						
✓0,2,8,10(-0-0)②	X		X	X	X			
✓10,11,14,15(1-1-)②					X	X	X	X
	✓③	✓①	✓①	✓①	✓③	✓①	✓①	✓①

$$F = A'B'C' + AC + B'D'$$

Ex.

$$F\left(A,B,C,D\right) = \Sigma m\left(1,3,7,11,15\right) + \Sigma d\left(0,2,5\right)$$

- Determination of prime-implicant

Step 1		Step 2	Step 2	
	A B C D			*A B C D*
0	0 0 0 0 ✓	0,1(1) ✓	0,1,2,3(1,2)	0 0 - -
1	0 0 0 1 ✓	0,2(2) ✓	0,1,2,3(1,2)	0 0 - -
2	0 0 1 0 ✓	1,3(2) ✓	1,3,5,7(2,4)	0 - - 1
3	0 0 1 1 ✓	1,5(4) ✓	1,3,5,7(2,4)	0 - - 1
5	0 1 0 1 ✓	2,3(1) ✓	3,7,11,15(4,8)	- - 1 1
7	0 1 1 1 ✓	3,7(4) ✓	3,7,11,15(4,8)	- - 1 1
11	1 0 1 1 ✓	3,11(8) ✓		
15	1 1 1 1 ✓	5,7(2) ✓		
		7,15(8) ✓		
		11,15(4) ✓		

- Prime-implicant table

	Minterms				
	1	3	7	11	15
✓ 0,1,2,3(00--) ④	X	X			
1,3,5,7(0--1)	X	X	X		
✓3,7,11,15(--11) ②		X	X	X	X
	✓⑤	✓③	✓③	✓①	✓①

$F = A'B' + CD$ or $F = A'D + C$

Ex.

$F\ \square A,B,C,D\square\ \square\ \square m\square 1,4,6,7,8,9,10,11,15\square$

- Determination of prime-implicant

Step 1		Step 2	Step 2	
	ABCD			A B C D
1	0 0 0 1 □	1,9(8)	8,9,10,11(1,2)	0 0 - -
4	0 1 0 0 □	4,6(2)	8,9,10,11(1,2)	0 0 - -
8	1 0 0 0 □	8,9(1) □		
6	0 1 1 0 □	8,10(2) □		
9	1 0 0 1 □	6,7(1)		
10	1 0 1 0 □	9,11(2) □		
7	0 1 1 1 □	10,11(1) □		
11	1 0 1 1 □	7,15(8)		
15	1 1 1 1 □	11,15(4)		

- Prime-implicant table

	Minterms								
	1	4	6	7	8	9	10	11	15
□ 1,9(-001) □	X					X			
□ 4,6(01-0) □		X	X						
6,7(011-)			X	X					
□ 7,15(-111) □				X					X

11,15(1-11)								X	X
□ 8,9,10,11(10--)□					X	X	X	X	
	□□	□□	□□	□□	□□	□□	□□	□□	□□

F □ *B*□*C*□*D* □ *A*□*BD*□ □ *BCD* □ *AB*□

1.10 Reduced Prime Implicant tables,

PRIME IMPLICATE: If the implicate does not subsumes any other implicate with fewer literals of the same function. In other words if we remove prime implicate term from the expression the remaining sum terms no longer implies the function

Ex. x and (y + z) are prime implicates

x y z	f
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	0
1 1 0	1
1 1 1	1

Ex1: Minimize f (A, B, C, D) = Σ (0, 1, 2, 8, 9, 15, 17, 21, 24, 25, 27, 31).

Step 1: Say we have obtained the following prime implicants:

P_1: BC'D'	P_4: ABDE	P_7: ABC'D'	P_{10}: A'B'C'E'
P_2: C'D'E	P_5: BCDE	P_8: A'B'D'E	P_{11}: A'B'C'D'
P_3: A'C'D'	P_6: ABC'E	P_9: B'C'D'E	

Step 2: Prime Implicant Chart

	0	1	2	8	9	15	17	21	24	25	27	31
P1				X	X				X	X		
P2		X			X		X			X		
P3	X	X		X	X							
P4											X	X
* P5						(X)						X
P6										X	X	
P7									X	X		
* P8							X	(X)				
P9		X					X					
* P10	X		(X)									
P11	X	X										

The prime implicants P5, P7, and P10 are essential. They are included in the solution. They do not cover all the minterms. So secondary essential prime implicants have to be found by using the reduced prime implicant chart.

Reduced Prime Implicant Chart (Essential prime implicants removed)

We are not able to find columns with single 'X'. Now we find the dominance relations.

Column Dominance:

9 > 8

25 > 24

Row Dominance:

P1 > P7

P6 > P4

P2 > P9

P2 > P11

	1	8	9	24	25	27
P1		X	X	X	X	
P2	X		X		X	
P3	X	X	X			
P4						X
P6					X	X
P7				X	X	
P9	X					
P11	X					

Prime Implicant Chart Reduction Steps:

 All the dominating columns and dominated rows of a prime-implicant chart can be removed without affecting the table for obtaining a minimal solution.

 Dominating column is guaranteed to be covered by the row that covers its dominated column.

 The columns of the dominated row are guaranteed to be covered by its dominating row.

Finding Secondary Essential PIs:

PI Chart after the dominating columns and the dominated rows are deleted:

MTs / PIs	1	√ 8	√ 24	√ 27
** P1		X	(X)	
P2	X			
P3	X	X		
** P6				(X)

Final Solution

Min term / PIs	√ 1
P2	X
P3	X

Minterm 1 can be covered by P2 or P3. If we select P2, we have the solution:

Y = P1 + P2 + P5 + P6 + P8 + P10

1.11 Map entered variables.

In entered variable map one of the input variables is placed inside Karnaugh map. This is done separately noting how the input variable is related with output. This reduces the Karnaugh map size by one degree. This technique is particularly useful for mapping problems with more than four input variables.

Example:

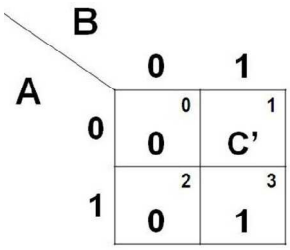
Consider the 3-variable truth table as shown below. The output Y is rewritten in terms of variable C.

A	B	C	Y	Y
0	0	0	0	0
0	0	1	0	
0	1	0	1	C'
0	1	1	0	
1	0	0	0	0
1	0	1	0	
1	1	0	1	1
1	1	1	1	

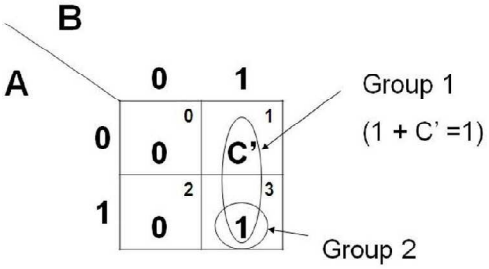
The 3 – variable truth table reduces to 2 – variable truth table as shown below:

A	B	Y
0	0	0
0	1	C'
1	0	0
1	1	1

The 2 – variable Karnaugh map is drawn as shown below:



The Karnaugh map is now called an entered variable map. The simplification of entered variable map is as illustrated next:



The product term representing each group is obtained by including map entered variable in the group as an additional ANDed term. Group 1 gives B.(C') and group 2 gives AB.1. Therefore, the simplified expression is obtained as Y = BC' + AB.

1.12 Outcome

- **Representation of Boolean expression in canonical forms.**
- **Reduce gates with minimum number of variable using different techniques.**

1.13 Future Readings

http://nptel.ac.in/courses/117105080/

https://www.youtube.com/watch?v=VnZLRrJYa2I
"Logic Design" by RD Sudhaker Samuel

"Digital Logic Applications and Design" by John M Yarbrough, 2011 edition.