

# **ATME COLLEGE OF ENGINEERING**

**13th KM Stone, Bannur Road, Mysore - 570 028**



## **DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING**

### **NOTES**

**Course Title : Digital Logic Circuits**

**Course CODE: BEE306A**

**SEMESTER: III**

**Academic Year - 2025-26**

# **INSTITUTIONAL VISION AND MISSION**

## **VISION:**

- Development of academically excellent, culturally vibrant, socially responsible and globally competent human resources.

## **MISSION:**

- To keep pace with advancements in knowledge and make the students competitive and capable at the global level.
- To create an environment for the students to acquire the right physical, intellectual, emotional and moral foundations and shine as torchbearers of tomorrow's society.
- To strive to attain ever-higher benchmarks of educational excellence.

## **Department Vision and Mission**

### **Vision:**

To create Electrical & Electronics Engineers who excel to be technically competent and fulfill the cultural and social aspirations of the society.

### **Mission:**

- To provide knowledge to students that builds a strong foundation in the basic principles of electrical engineering, problem solving abilities, analytical skills, soft skills and communication skills for their overall development.
- To offer outcome based technical education.
- To encourage faculty in training & development and to offer consultancy through research & industry interaction.

## **Program Educational Objectives (PEOs)**

**PEO1:** To produce competent and ethical Electrical and Electronics Engineers who will exhibit the necessary technical and managerial skills to perform their duties in society.

**PEO2:** To make Graduates continuously acquire and enhance their technical and socio-economic skills.

**PEO3:** To aspire Graduates on R & D activities leading to offering solutions and excel in various career paths.

**PEO4:** To produce quality engineers who have the capability to work in teams and contribute to real time projects.

## **Program Outcomes (POs)**

**Engineering Graduates will be able to:**

**PO1: Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.

**PO2: Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

**PO3: Design / Development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

**PO4: Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

**PO5: Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

**PO6: The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

**PO7: Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO8: Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

**PO9: Individual and team work:** Function effectively as an individual and as a member or leader in diverse teams, and in multidisciplinary settings.

**PO10: Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**PO11: Project management and finance:** Demonstrate knowledge and understanding of the engineering management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO12: Life-long learning:** Recognize the need for and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

### **Program Specific Outcomes (PSOs)**

The students will develop an ability to produce the following engineering traits:

PSO1: Apply the concepts of Electrical & Electronics Engineering to evaluate the performance of power systems and also to control industrial drives using power electronics

PSO2: Demonstrate the concepts of process control for Industrial Automation, design models for environmental and social concerns and also exhibit continuous self- learning

# Digital Logic Circuit

				Academic Year: 2025-2026			
Department: Electrical and Electronics Engineering							
Course Code	Course Title	Core/Elective	Prerequisite	Contact Hours			Total Hrs/ Sessions
				L	T	P	
BEE306A	Digital Logic Circuit	Elective	Basic Electronics	3	-	-	40 theory
Topics Covered as per Syllabus							
Module-1							
Definition of combinational logic, canonical forms, Generation of switching equations from truth tables, Karnaugh maps-3,4,5 variables, Incompletely specified functions (Don't care terms) Simplifying Max term equations, Quine-McCluskey minimization technique, Quine-McCluskey using don't care terms, Reduced prime implicants Tables.							
Module-2							
General approach to combinational logic design, Decoders, BCD decoders, Encoders, digital multiplexers, Using multiplexers as Boolean function generators, Adders and subtractors, Cascading full adders, Look ahead carry, Binary comparators							
Module-3							
Basic Bistable elements, Latches, Timing considerations, The master-slave flip-flops (pulsetriggered flip-flops): SR flip-flops, JK flip-flops, Edge triggered flip-flops, Characteristic equations							
Module-4							
Registers, binary ripple counters, synchronous binary counters, Counters based on shift registers, Design of a synchronous counter, Design of a synchronous mod-n counter using clocked T, JK, D and SR flip-flops.							
Module-5							
Mealy and Moore models, State machine notation, Synchronous Sequential circuit analysis, Construction of state diagrams, counter design.							
Memories: Read only and Read/Write Memories, Programmable ROM, EPROM, Flash memory.							
List of Text Books							
“Digital Logic Applications and Design” by John M Yarbrough, 2011 edition. “HDL Programming (VHDL and Verilog)” by Nazeih M. Botros, 1 st Edition “Digital Principles and Design “, Donald D Givone, Tata McGraw Hill Edition,2002.							
List of Reference Books							
“Logic Design” by RD Sudhaker Samuel							
List of URLs, Text Books, Notes, Multimedia Content, etc: <a href="https://www.youtube.com/watch?v=VnZLRrJYa2I">https://www.youtube.com/watch?v=VnZLRrJYa2I</a>							

## MODULE 1

### Principles of Combinational Logic

#### Structure

- Objective
- Introduction
- Review of Boolean Algebra.
- Definition of combinational
- Canonical forms
- Generation of switching equations from truth tables
- Karnaugh maps-3,4 and 5 variables. Incompletely specified functions (Don't care terms).
- Simplifying max-term equations.
- Quine-McClusky minimization technique, Quine-McClusky using don't care terms,
- Reduced Prime Implicant tables,
- Map entered variables.
- Outcome
- Future Readings

#### Objective

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

#### 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

#### Review of Boolean Algebra.

##### Example 1:

$$\begin{array}{lll} & xy + xy + xy = x + y & \\ \text{LHS} & x(y+y) + xy & \text{using distributive law} \end{array}$$

$$\begin{aligned}
 x \cdot 1 + xy & \quad \text{using } y+y=1 \text{ theorem} \\
 x(1+y) + xy & \quad (1+y)=1 \text{ theorem } x+ \\
 xy + xy & \quad \text{using distributive law } x+y(x+ \\
 x) & \\
 x+y \cdot 1 & = x+y \text{ RHS}
 \end{aligned}$$

**Example 2:**

$$\begin{aligned}
 (x+y)(x+z) & = xz + xy \quad \text{using distributive law} \\
 x x + x z + x y + y z & \quad \text{using P5 postulate} \\
 x z + x y + (x+x) y z & \quad \text{using distributive law} \\
 xz + x y + xyz + xyz & \quad \text{using distributive law} \\
 xz(1+y) + x y(1+z) & \quad \text{using T2 theorem} \\
 xz \cdot 1 + xy \cdot 1 & = xz + xy \text{ RHS}
 \end{aligned}$$

**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  $x \cdot x = 0$

**BOOLEAN FORMULAS AND FUNCTIONS**

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

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

$$f(x,y,z) = (x+y)z \text{ or } f = (x+y)z$$

xyz	xy	xz	xyz	f
000	0	0	0	0
001	0	0	0	0
010	0	0	0	0
011	0	0	0	0
100	0	1	0	1
101	0	0	0	0
110	1	1	1	1
111	1	0	0	1

Truth table for the above Boolean expression is

**Example2:** Write a truth table for the following function

$$f = xyz + xy + xz$$

xyz	xy	xz	xyz	f
000	0	0	0	0
001	0	0	0	0
010	0	0	0	0
011	0	0	0	0
100	0	1	0	1
101	0	0	0	0
110	1	1	1	1
111	1	0	0	1

### Definition of combinational

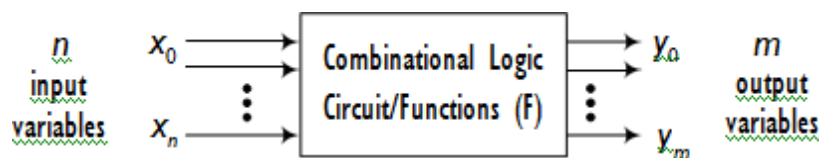
#### 1.4.1 COMBINATIONAL NETWORK

The interconnections of 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 a set of signals at any time is called input state or input vector of the network. While a set of resulting signals appearing at the output terminals is called the output state or output vector. The network can be expressed as

$z_1, z_2, \dots, z_m$  as Boolean function then  $Z_i$

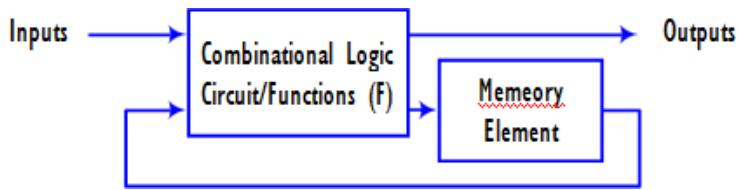
$$= f_i(x_1, x_2, \dots, x_n) \text{ for } i = 1, 2, \dots, m.$$



#### 1.4.2. SEQUENTIAL NETWORKS

A second type of logic network is the sequential networks. Sequential networks have memory property, so that the outputs from these networks are dependent not only upon the current inputs but upon previous inputs as well. Feedback paths 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

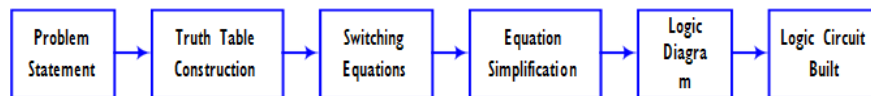
1. Each gate output that is only a function of the input variables is labeled.
2. Boolean algebraic expression for the outputs of each of these gates are then written.
3. Next these gates outputs that are a function of just inputs variables and previously labeled gate outputs.
4. Then each of the previously 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.

$$f(w,x,y,z) = w \cdot (y+z) + wxy$$

$$= w \cdot G1 + G2$$

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

#### • 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. It is said to be in the sum of product form or disjunctive normal form.

Example:

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

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

Example:

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

### Canonical forms

A procedure which will be used to write Boolean expressions from 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 represent the functional values of the variables appear either in complemented or un complemented form.

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

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

#### m- NOTATION

To simplify the writing of a minterm canonical formula for a function performed using the symbol ***m***. Where ***i*** stands for the row number for function evaluate to 1.

function is  
which the

The m-notation for 3-variable function Boolean function

$$f(x,y,z)=xyz+xyz+xyz \text{ is written as}$$

$$f(x,y,z)=m1+m3+m4 \text{ or}$$

$$f(x,y,z)=\sum m(1,3,4)$$

### A three variable- notation truth variable

xyz	Decimal designator of row	Minterm	m-notation
0 0 0	0	$\bar{x} \bar{y} \bar{z}$	m0
0 0 1	1	$\bar{x} \bar{y} z$	m1
0 1 0	2	$\bar{x} y \bar{z}$	m2
0 1 1	3	$\bar{x} y z$	m3
1 0 0	4	$x \bar{y} \bar{z}$	m4
1 0 1	5	$x \bar{y} z$	m5
1 1 0	6	$x y \bar{z}$	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 correspond to a functional value representing 0.

$$\begin{aligned} \text{Ex. } f(x,y,z) &= (x+y+z)(x+y+z)(x+y+z) \\ &= \prod M(0,2,5) \\ &= M0, M2, M5 \end{aligned}$$

### M-NOTATION

A maxterm in a canonical form can be represented as  $M_i$ . Where  $i$  stands for row number for which the function evaluates to 0. A product of maxterms are represented as  $\prod M$ .

### MINTERM CANONICAL FORMULA

1. Apply DeMorgan's law as sufficient no. of times until all the NOT operations appear only with the single variables.
2. Apply distributive of AND over OR ( $\bullet$ ) over (+) i.e.  $x \bullet (y+z) = xy + xz$  in order to manipulate the formula into disjunctive normal formula.
3. Remove duplicate literals and turn by idempotent law as any term that are identically zero ( $x \bullet x = x$ ).

4. If any product in the disjunctive normal form 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  $x_i + x_i$  where  $x_i$  is the missing variable being introduced. This process is repeated for each missing variable in each of the product terms of the disjunctive normal form
5. Apply distributive law of  $(\bullet)$  over  $(+)$  again so that each variable appears exactly once in each term.
6. Remove duplicate terms if any.

**Example :**  $(x + y) + (y + xz)(x + y)$

### MAXTERM CANONICAL FORMULA

1. Apply De Morgan's law until all the NOT operations appear with single variables.
2. Apply distributive law of  $(+)$  over  $(\bullet)$  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 in logic 0's in the form  $x_i \bullet x_i = 0$  where  $x_i$  is missing variable.
4. Distributive law of  $(+)$  over  $(\bullet)$  is again applied.
5. Duplicate literals are deleted.

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

### 1.5.6. COMPLEMENTS OF CANONICAL FORMULAS

Even by taking complements minterm expression may result different expressions.

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

$$\sum m(1, 3, 5, 7)$$

Similarly

A Maxterm canonical expression may be represented in completed form as follows Ex :

$f(x, y, z) = \prod M(1, 2, 4, 7)$  its complement expression is

$$f(x, y, z) = \prod M(0, 3, 5, 6)$$

**Generation of switching equations from truth tables,**

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

A method for graphically determining implicants and implicants 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  $2^n$  rows. Each row of the truth table consists 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  $2^n$  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 for  $f(x)$

0  $f(0)$

1  $f(1)$

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

$x \quad y \quad f(x,y)$

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 is as follows

$xyz \quad f(x,y,z)$

000  $f(0,0,0)$

001  $f(0,0,1)$

010	$f(0,1,0)$
011	$f(0,1,1)$
100	$f(1,0,0)$
101	$f(1,0,1)$
110	$f(1,1,0)$
111	$f(1,1,1)$

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

wxyz	$f(w,x,y,z)$	wxyz	$f(w,x,y,z)$
0 0 00	$f(0,0,0,0)$	1010	$f(1,0,1,0)$
0 0 01	$f(0,0,0,1)$	1011	$f(1,0,1,1)$
0 0 10	$f(0,0,1,0)$	1100	$f(1,1,0,0)$
0 0 11	$f(0,0,1,1)$	1101	$f(1,1,0,1)$
0 1 00	$f(0,1,0,0)$	1110	$f(1,1,1,0)$
0 1 01	$f(0,1,0,1)$	1111	$f(1,1,1,1)$
0 1 10	$f(0,1,1,0)$		
0 1 11	$f(0,1,1,1)$		
1 0 00	$f(1,0,0,0)$		
1 0 01	$f(1,0,0,1)$		

0000	0001	0011	0010
0100	0101	0111	0110
1100	1101	1111	1110
1000	1001	1011	1010

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

$$f(x,y,z)=xy z+xyz+xyz+xyz \quad f(x,y,z) =$$

$$\Sigma m(0,2,4,5)$$

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 implicants 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 referred 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  $2^a \times 2^b$  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 is 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-cells can result in n-1 variable. Where as if a group of four adjacent subcubes are formed then it can result in n-2 variables. Finally if we have eight adjacent cells are grouped may result n-3 variable product or sum term.

Typical pairs of subcubes

1			
	1	1	
1		1	1
1			

Typical group of four adjacent subcubes

1	1		
1	1		

		1	
		1	
		1	
		1	

1	1	1	1

Typical group of four adjacent subcubes.

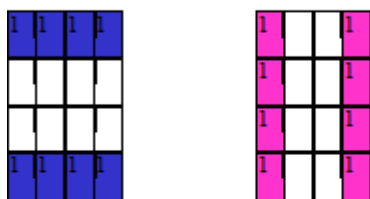
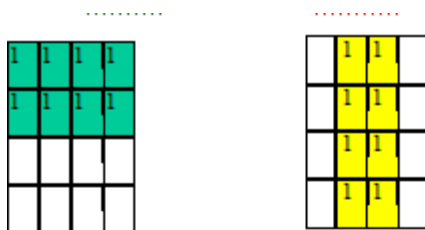
1			1
1			1

Typical group of eight adjacent

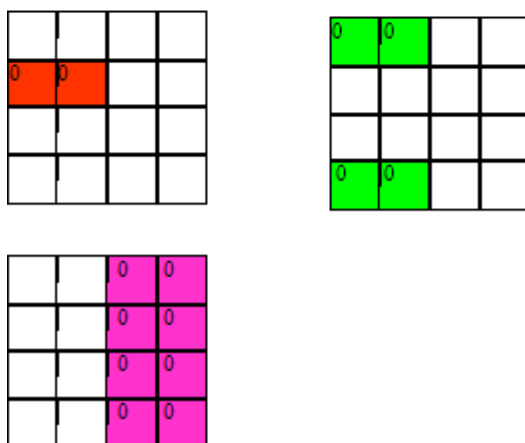
1			1
1			1

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 a given function is discussed. **PRIME**

**IMPLICANTS and K-MAPS :**

CONCEPT OF ESSENTIAL PRIME IMPLICANT

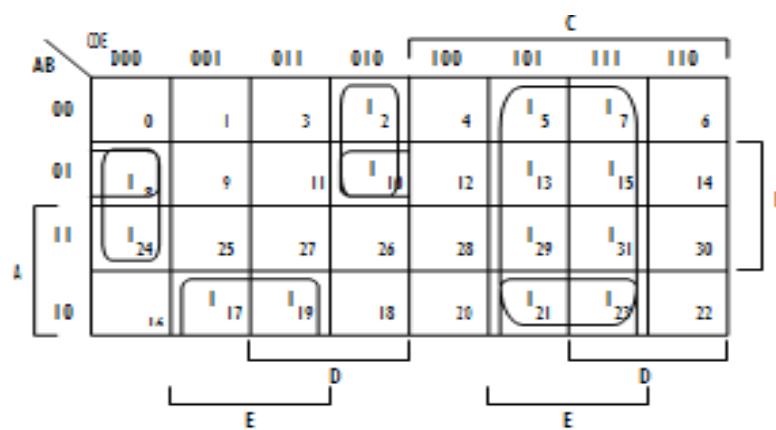
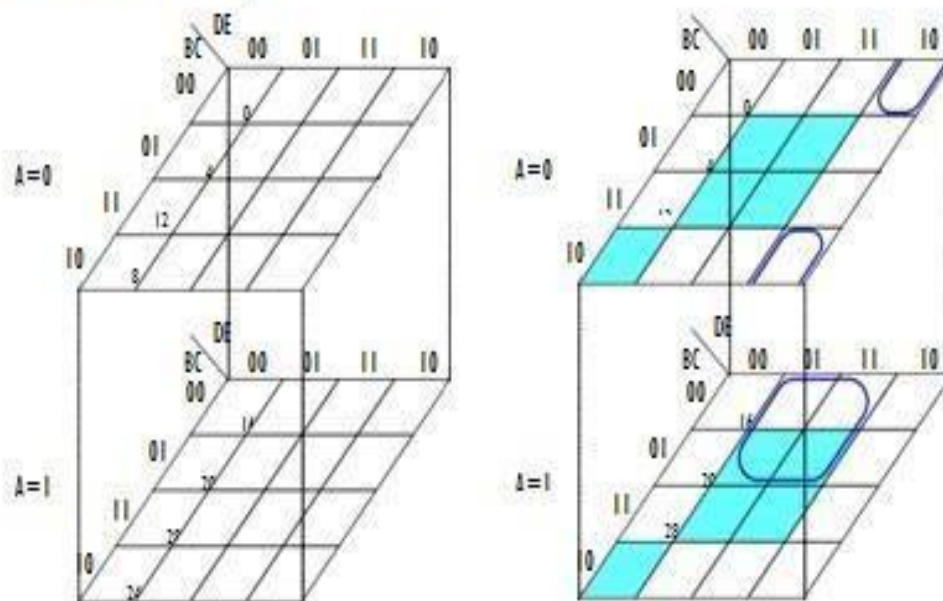
**00    01    11    10**

0	0	0	1
0	0	1	1

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

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

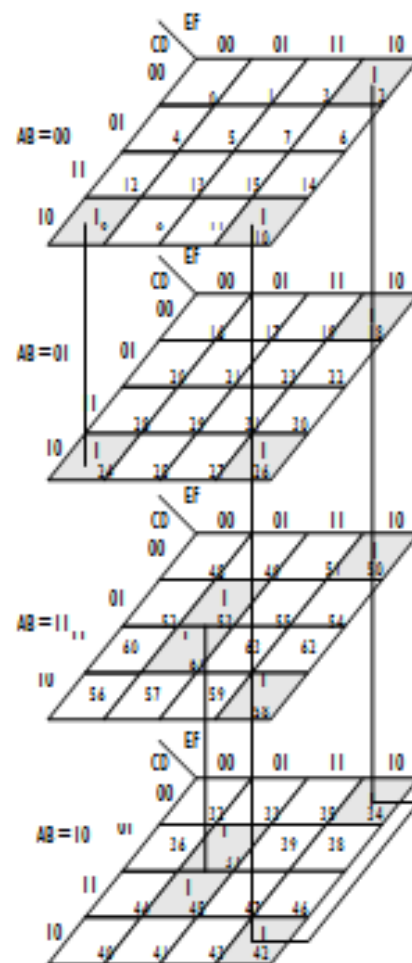
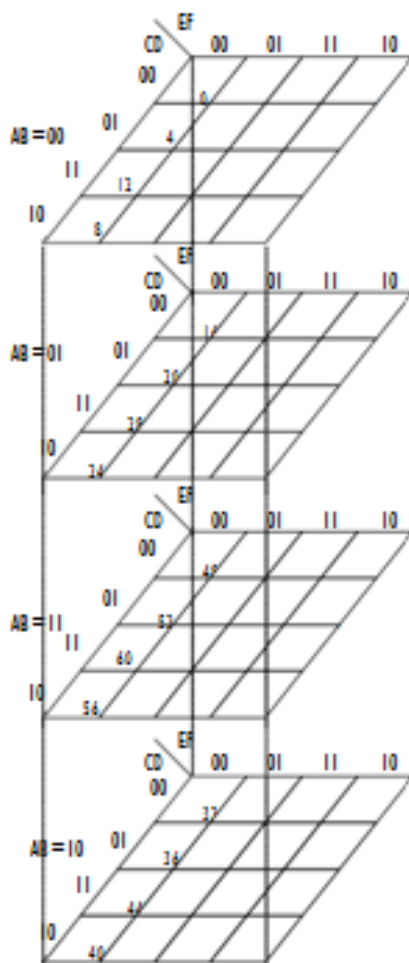
## □ Five-Variable K-Map



$$f(A,B,C,D,E) = \sum m(2,5,7,8,10,13,15,17,19,21,23,24,29,31)$$

$$= CE + BC'D'E' + AB'E + A'CDE'$$

□ Six-Variable K-Map



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

$$= D'EF' + ADE'F + A'CD'F'$$

DEF		D							
		000	001	011	010	100	101	111	110
ABC	000				1				
	001	1							
	011				1				
	010								
	100				1				
	101								
	111				1				
	110								

Groupings: A (rows 000-110), B (columns 000-010), C (columns 100-110), E (columns 000-010 and 100-110), F (columns 000-001 and 100-101).

### INCOMPLETE BOOLEAN FUNCTION WITH DONOT CARE CONDITIONS

In this type of Boolean functions having  $n$ -variables so that it may have  $2^n$  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

$$f(x,y,z) = \sum m(0,1,7) + dc(3,5)$$

Also we can represent the function in the POS form as  $f(x,y,z) =$

$$\prod M(2,4,6) + dc(3,5)$$

The odd parity generation result output expression

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

$$f(w,x,y,z) = wx'yz + x'yz + x'y'z + xyz + w'z$$

**(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-3BCD			
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

$$A = f(W, X, Y, Z) = \Sigma m(5, 6, 7, 8, 9) + \Sigma d(10, 11, 12, 13, 14, 15)$$

$$B = f(W, X, Y, Z) = \Sigma m(1, 2, 3, 4, 9) + \Sigma d(10, 11, 12, 13, 14, 15)$$

$$C = f(W, X, Y, Z) = \Sigma m(0, 3, 4, 7, 8) + \Sigma d(10, 11, 12, 13, 14, 15)$$

$$D = f(W, X, Y, Z) = \Sigma m(0, 2, 4, 6, 8) + \Sigma d(10, 11, 12, 13, 14, 15)$$

		Y			
		00	01	11	10
W\X	00	0	1	3	2
	01	4	5	7	6
	11	X <sub>12</sub>	X <sub>13</sub>	X <sub>15</sub>	X <sub>14</sub>
	10	8	9	X <sub>11</sub>	X <sub>10</sub>

$$A = W + XZ + XY$$

		Y			
		00	01	11	10
W\X	00	0	1	3	2
	01	4	5	7	6
	11	X <sub>12</sub>	X <sub>13</sub>	X <sub>15</sub>	X <sub>14</sub>
	10	8	9	X <sub>11</sub>	X <sub>10</sub>

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

		Y			
		00	01	11	10
W\X	00	0	1	3	2
	01	4	5	7	6
	11	X <sub>12</sub>	X <sub>13</sub>	X <sub>15</sub>	X <sub>14</sub>
	10	8	9	X <sub>11</sub>	X <sub>10</sub>

$$C = Y'Z' + YZ'$$

		Y			
		00	01	11	10
W\X	00	0	1	3	2
	01	4	5	7	6
	11	X <sub>12</sub>	X <sub>13</sub>	X <sub>15</sub>	X <sub>14</sub>
	10	8	9	X <sub>11</sub>	X <sub>10</sub>

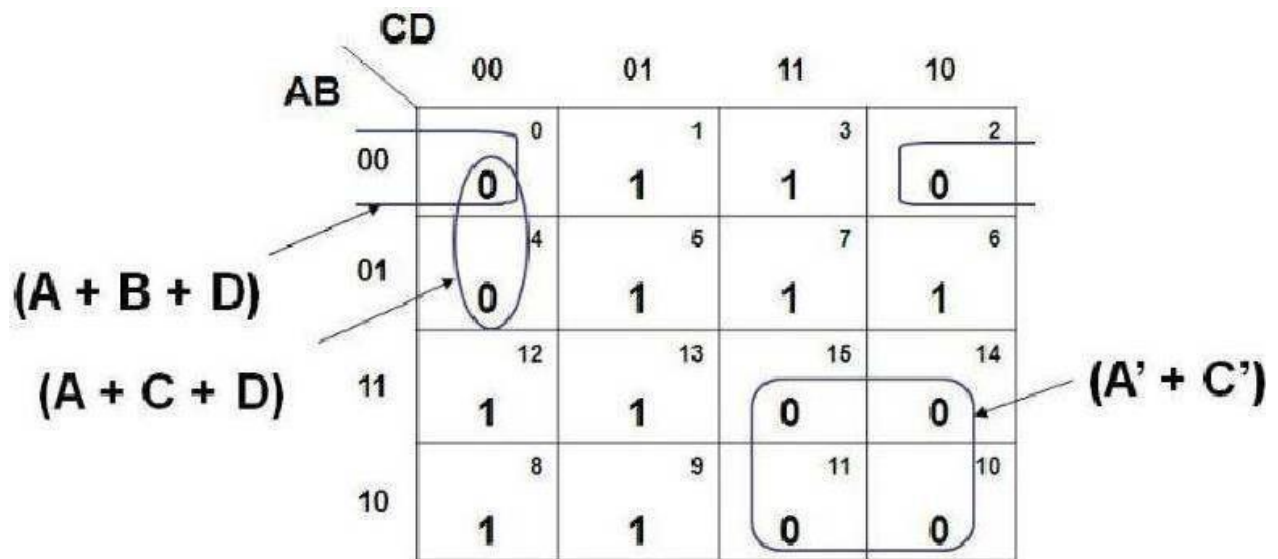
$$D = Z'$$

### Simplifying max-termequations.

ReducethefollowingfunctionusingKarnaughmaptechnique f

$$(A, B, C, D) = \pi M(0, 2, 4, 10, 11, 14, 15)$$

TheK-mapforthegivenlogicfunctionisdrawnasshown below:



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

**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	
<i>ABCD</i>		<i>ABCD</i>		<i>ABCD</i>	
0	0000✓	0,1	000-	0,2,8,10	-0-0
1	0001✓	0,2	00-0✓	0,8,2,10	-0-0
2	0010✓	0,8	-000✓		
8	1000✓	2,10	-010✓	10,11,14,15	1-1-
10	1010✓	8,10	10-0✓	10,14,11,15	1-1-
11	1011✓	10,11	101-✓		
14	1110✓	10,14	1-10✓		
15	1111✓	11,15	1-11✓		
		14,15	111-✓		

- Prime-implicant table

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

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

Ex.

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



- Determination of prime-implicant

Step1		Step2	Step2	
ABCD			ABC D	
0	0000 ✓	0,1(1) ✓ 0,2(2) ✓	0,1,2,3(1,2)	00- -
1	0001 ✓		0,1,2,3(1,2)	00- -
2	0010 ✓			
3	0011 ✓	1,3(2) ✓ 1,5(4) ✓	1,3,5,7(2,4)	0-- 1
5	0101 ✓		1,3,5,7(2,4)	0-- 1
7	0111 ✓	2,3(1) ✓ 3,7(4) ✓ 3,11(8) ✓ 5,7(2) ✓ 7,15(8) ✓ 11,15(4) ✓	3,7,11,15(4,8)	--1 1
11	1011 ✓		3,7,11,15(4,8)	--1 1
15	1111 ✓			

- Prime-implicant table

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

$$F = A'B' + CD \text{ or } F = A'D + C$$

Ex.

$$F(A, B, C, D) = \sum m(1, 4, 6, 7, 8, 9, 10, 11, 15)$$

- Determination of prime-implicant

Step1		Step2	Step2	
ABCD			ABCD	
1	0 0 0 1 □	1,9(8) 4,6(2) 8,9(1) □	8,9,10,11(1,2)	0 0 --
4	0 1 0 0 □		8,9,10,11(1,2)	0 0 --
8	1 0 0 0 □			
6	0 1 1 0 □	8,10(2) □ 6,7(1)		
9	1 0 0 1 □			
10	1 0 1 0 □	9,11(2) 10,11(1)		
7	0 1 1 1			
11	1 0 1 1	7,15(8) 11,15(4)		
15	1 1 1 1 □			

- Prime-implicanttable

	Minterms								
	1	4	6	7	8	9	10	11	15
<input type="checkbox"/> 1,9(-001)	X					X			
<input type="checkbox"/> 4,6(01-0)		X	X						
<input type="checkbox"/> 6,7(011-)			X	X					
<input type="checkbox"/> 7,15(-111)				X					X

11,15(1-11)								X	X
8,9,10,11(10--)					X	X	X	X	
	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>
$F = B \cdot CD + \bar{A} \cdot BD + \bar{B}CD + AB$									

### Reduced Prime Implicant tables,

**PRIME IMPLICATE:** If the implicate does not subsume 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  $f(A, B, C, D) = \Sigma (0, 1, 2, 8, 9, 15, 17, 21, 24, 25, 27, 31)$ .

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

$P_1: BCD'$	$P_4: ABDE$	$P_7: ABC'D'$	$P_{10}: A'B'CE'$
$P_2: CDE$	$P_5: BCDE$	$P_8: AB'DE$	$P_{11}: ABC'D'$
$P_3: A'CD'$	$P_6: ABCE$	$P_9: B'C'DE$	

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

**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:

		B	
		0	1
A	0	0 <sup>0</sup> 0	1 <sup>1</sup> C'
	1	0 <sup>2</sup> 0	1 <sup>3</sup> 1

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

		B	
		0	1
A	0	0 <sup>0</sup> 0	1 <sup>1</sup> C'
	1	0 <sup>2</sup> 0	1 <sup>3</sup> 1

Group 1 (1 + C' = 1)

Group 2

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  $Y = BC' + AB$ .

**Outcome**

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

**Future Readings**

**<http://nptel.ac.in/courses/117105080/>**

**<https://www.youtube.com/watch?v=VnZLRrJYa2I>**

**“Logic Design” by RDSudhaker Samuel**

**“Digital Logic Applications and Design” by John M Yarbrough, 2011 edition.**





