Digital Logic Circuits – BEE306A

Prepared By,

Ms. Swathi C A

Asst Professor

Dept of EEE

ATMECE, Mysuru

Course Syllabi with CO's

Faculty Name: **Mrs. Swathi C A**				Academic Year: **2025-26**				
Department: Electrical & Electronics Engineering								
Course Code	Course Title	Core/Elective	Prerequisite	Contact Hours				Total Hrs/ Sessions
				L	T	P	S	
BEE 306A	**Digital Logic Circuits**	**Elective**	**Basic Electronics**	**3**	-	-	-	**40 Hr Theory**
Objectives	• To illustrate simplification of algebraic equations using Karnaugh Maps and Quine-McClusky methods • To design decoders, encoders, digital multiplexer, adders, subtractors and binary comparators • To explain latches and flip-flops, registers and counters • To analyze Melay and Moore Models • To develop state diagrams synchronous sequential circuits • To understand the applications of sequential circuits							
Topics Covered as per Syllabus								
Module-1: **8 hours** **Principles of Combinational Logic**: 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-Mc Cluskey minimization technique, Quine-Mc Cluskey using don't care terms, Reduced prime implicants Tables.								

Module-2: **8 hours**

Analysis and Design of Combinational logic: 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: **8 hours**

Flip-Flops: Basic Bistable elements, Latches, Timing considerations, The master-slave flip-flops (pulse triggered flip-flops):SR flip-flops, JK flip-flops, Edge triggered flip- flops, Characteristic equations

Module -4: **8 hours**

Flip-Flops Applications: 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: **8 hours**

Sequential Circuit Design: 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 and Reference Books

Text Books:

(1) John M Yarbrough , Digital logic applications and design, Thomson Learning, 2001.
(2) Donald D Givone, Digital Principles and design, MC Graw Hill 2002.
(3) Charles H Roth Jr, Larry L Kinney, Fundamentals of logic design, Cengage Learning, 7th Edition.

Reference Books:

(1) D.P.Kothari and J S Dhillon, -Digital circuits and design, Pearson, 2016.
(2) Morris Mano, Digital Design, PHI, 3rd edition.
(3) K.A. Navas, Electronics Lab Manual, Vol.1, PHI 5th edition, 2015.

List of URLs, Text Books, Notes, Multimedia Content, etc	
1. https://onlinecourses.nptel.ac.in/noc20_ee32/preview 2. YouTube videos on digital electronics 3. National Instruments: https://education.ni.com/teach/resources/1104/digital-electronics	
Course Outcomes	At the end of the course the students will be able to: 1. **Explain** the concept of combinational and sequential logic circuits. [L2] 2. Analyse and **design** combinational circuits. [L3] 3. **Describe** and characterize flip flops and its applications. [L3] 4. **Design** the sequential circuits using SR, JK, D and T flip-flops and Melay and Moore applications. [L3] 5. **Design** applications of combinational and sequential circuits. [L3] 6. **Employ** the digital circuits for different applications. [L3]
Internal Assessment Marks: 50 (2 Theory Tests of 25Marks each + 2 Assignments of 10 Marks each are conducted during thesemester and marks allotted based on average all the performances).	

What is Digital Electronics?

- Branch of electronics dealing with discrete signals (0s & 1s)
- Opposite of Analog Electronics (continuous signals)
- Foundation of computers, mobiles, control systems

Analog vs Digital Signals

- Analog: Continuous, smooth variation (e.g., sound, temperature)
- Digital: Discrete steps (0 or 1, ON or OFF)
- Example: Sine wave vs Square wave

Binary Logic

- Two logic levels: 0 (Low = 0V), 1 (High = +5V or +3.3V)
- Computers use binary as circuits detect two states (ON/OFF)

Applications of Digital Electronics

- Computers and laptops
- Mobile phones
- Digital watches
- Industrial automation
- Smart home devices

Fundamentals: Logic Gates

- Logic gates are the basic building blocks of any digital system.
- Implements Boolean functions that performs a logic operation on one or more bits of input and gives a bit as an output
- The relationship between the input and the output is based on a certain logic.

Logic Gates

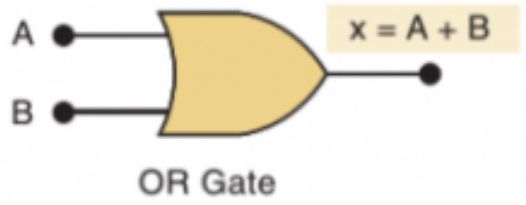
- AND Gate → Output is 1 if all inputs are 1
- OR Gate → Output is 1 if any input is 1
- NOT Gate → Inverts input
- NAND & NOR are universal gates
- Truth tables and symbols are key

Logic Gates [Contd. .]

- The basic logic gates are classified into seven types:
- AND gate, OR gate, NOT, XOR gate, NAND gate, NOR gate, XNOR gate.
- The truth table is used to show the logic gate function

1. OR Gate: Expression**: x = A + B**

Symbol:



Truth Table of OR Gate

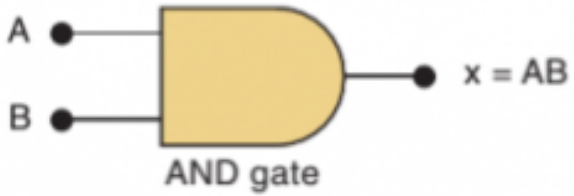
OR

A	B	x = A + B
0	0	0
0	1	1
1	0	1
1	1	1

Logic Gates [Contd. .]

2. AND Gate: Expression**: x = A . B**

Symbol:



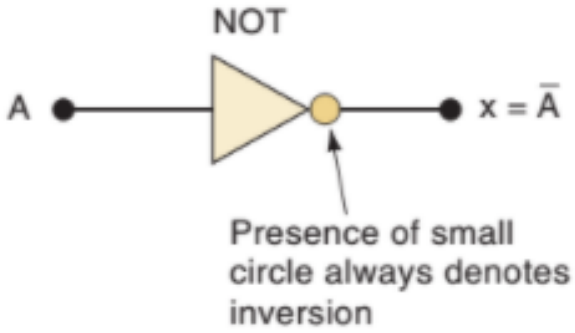
Truth Table of AND Gate

AND

A	B	x = A · B
0	0	0
0	1	0
1	0	0
1	1	1

3. NOT Gate: Expression: A = A’

Symbol:



Truth Table of NOT Gate

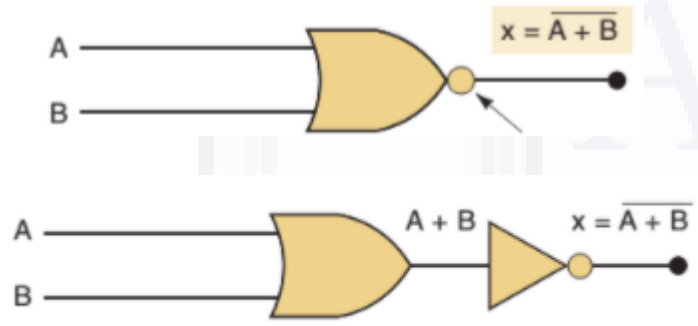
NOT

A	x = Ā
0	1
1	0

Logic Gates [Contd. .]

4. NOR Gate: Expression: $x = \overline{A + B}.$

Symbol:



Truth Table of NOR Gate

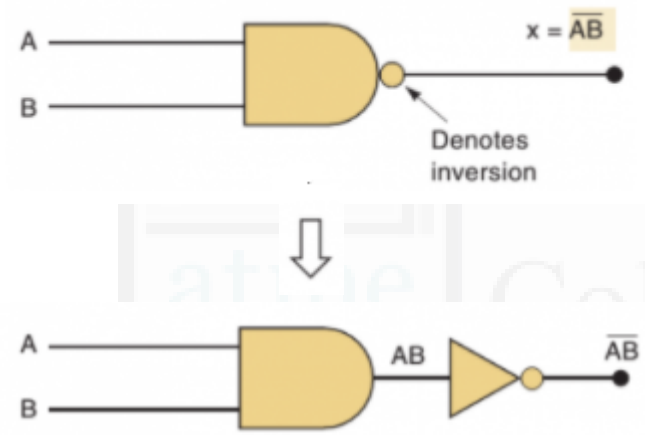
		OR	NOR
A	B	A + B	$\overline{A + B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Logic Gates [Contd. .]

5. NAND Gate: Expression:

$$x = \overline{AB}.$$

Symbol:



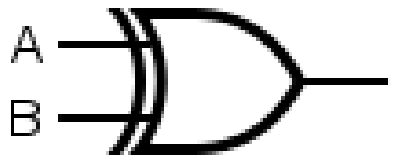
Truth Table of NOR Gate

		AND	NAND
A	B	AB	$\overline{AB}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Logic Gates [Contd. .]

6. XOR Gate: Expression**: x =** $A \oplus B$

Symbol:



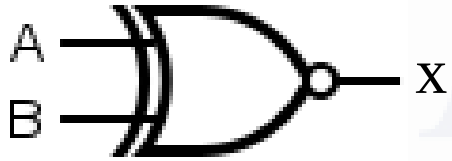
Truth Table of XOR Gate

INPUT		OUTPUT
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Logic Gates [Contd. .]

7. XNOR Gate: Expression**: x =** $: A \odot B$.

Symbol:



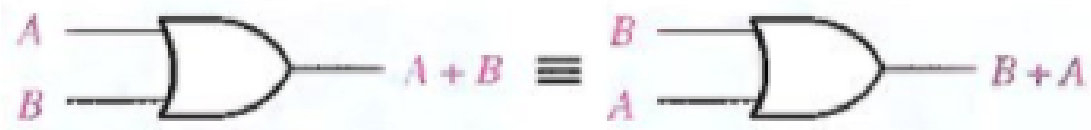
Truth Table of XOR Gate

Input		Output
A	**B**	**A XNOR B**
0	0	1
0	1	0
1	0	0
1	1	1

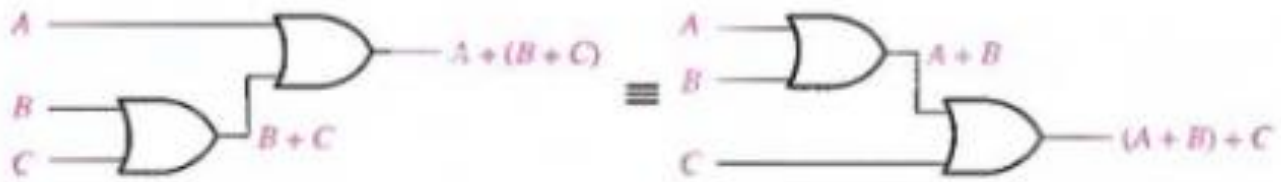
Laws and Rules of Boolean Algebra

Basic Laws:

1. **Commutative Law : A+B = B+A**



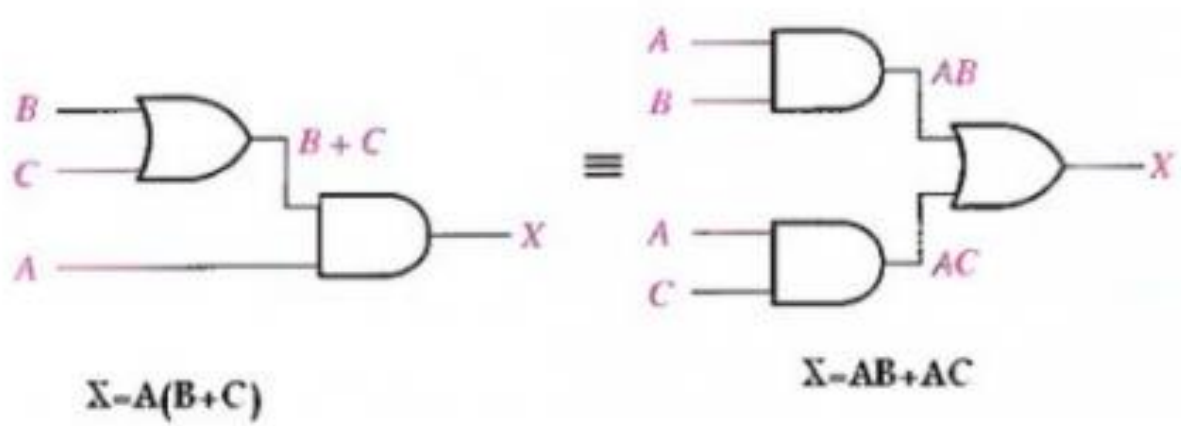
2. **Associative Law : A+ (B+C) = (A+B)+C**



Laws and Rules of Boolean Algebra

Basic Laws:

3. Distributive Law: A (B+C) = AB+ AC



Laws and Rules of Boolean Algebra

Basic Laws:

4. Duality Property

- Duals are just the opposites of original operators

- Switching Logic depends upon Duality property b/w AND and OR operators

Eg: Operator	Dual
AND	OR
OR	AND
0	1
1	0

Laws and Rules of Boolean Algebra

Basic Laws:

5. Idempotency Property

- Idempotency refers to "sameness"

Hence, $x + x = x$

$x.x = x$

Laws and Rules of Boolean Algebra

12 Basic Rules of Boolean Algebra

1. $A+0=A$	7. $A \cdot A = A$
2. $A+1=1$	8. $A \cdot \overline{A} = 0$
3. $A \cdot 0 = 0$	9. $\overline{\overline{A}} = A$
4. $A \cdot 1 = A$	10. $A+AB=A$
5. $A+A=A$	11. $A+\overline{A}B=A+B$
6. $A+\overline{A}=1$	12. $(A+B)(A+C)=A+BC$

DeMorgan's Theorems

- DeMorgan's Theorems uses the principle of Duality

There are two statements under DeMorgan's Theorem

1. The complement of an AND function is the OR of the complemented input variables

Eg: Y = $\overline{A.B}$ then Y = $\bar{A} + \bar{B}$

2. The complement of an OR function is the AND of the complemented input variables

Y = $\overline{A + B}$ then z = $\bar{A} \,.\, \bar{B}$

Reduction of Switching Equations using Boolean Algebra

Ex 1.

$$F = BC + B\bar{C} + BA$$

$$F = B(C + \bar{C}) + BA$$

$$F = B \cdot 1 + BA$$

$$F = B(1 + A)$$

$$F = B$$

Realization of Switching Functions

Eg: AB + A(B + C) + B(B + C)

AB + AB + AC + BB + BC

Since, (BB = B) to the fourth term

AB + AB + AC + B + BC

Since (AB + AB = AB) to the first two terms.

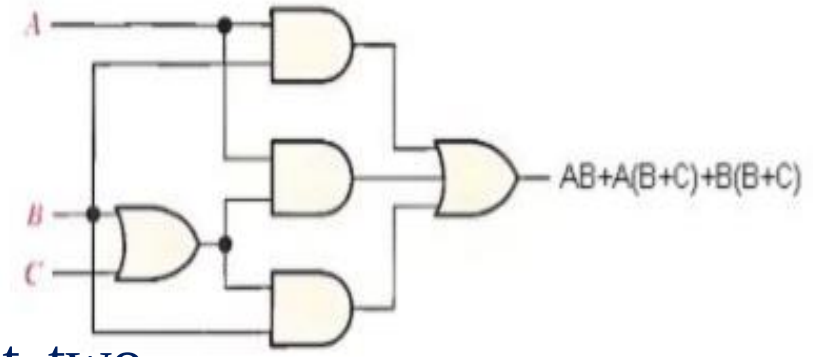
AB + AC + B + BC

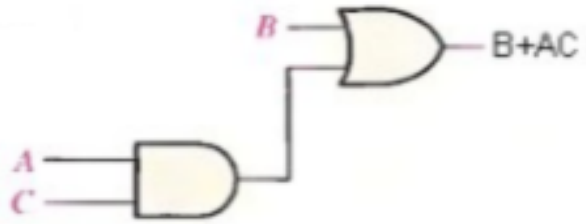
Since (B + BC = B) to the last two terms

AB + AC + B

(AB + B = B) to the first and third terms

→ B+AC





Definition of combinational Logic

There are two types of logic networks

Combinational network

Sequential networks

Combinational Logic Network

- It is a technique of combining the basic gates into circuits that perform some desired function.
- These circuits will not exhibit “Memory”
- Output of the system depends on the present input

Eg: Adders, Subtractors, Decoders, Encoders, Multipliers

Combinational Logic Diagram:

Sequential Networks

- These networks exhibit a “Memory”
- i.e. The Output is characterized by the combination of present input and also depends upon on the past history of the input

Eg: Flip flops

Sequential Logic Diagram:

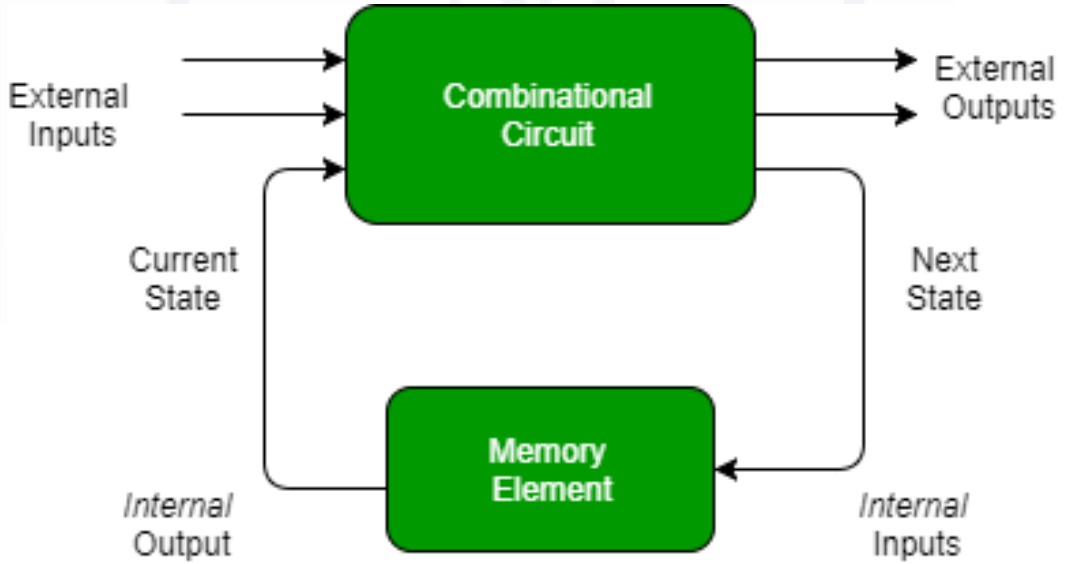


Figure: Sequential Circuit

Important Terminologies

1. Literal: A Boolean variable or its complement is known as Literal. Ex: A, $\bar{A}$, X, $\bar{B}$ etc
2. Product Term: Logical AND (product) of multiple literals

Ex: AB, A$\bar{B}$, XY$\bar{Y}$ etc

3. Sum Term: Logical OR (Sum) of multiple literals

Ex: A+B, A+$\bar{B}$, X+Y+$\bar{Y}$ etc

4. Sum of Product (SOP):

When two or more product terms are summed by Boolean addition, the resulting expression is a sum-of-products (SOP). Ex: AB + ABC , ABC + CDE + BCD , AB + BCD + AC

Important Terminologies (Contd..)

5. Product-of-sums (POS) When two or more sum terms are multiplied, the resulting expression is a product-of-sums

Ex: $(\overline{A} + B)(A + \overline{B} + C)$

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

6. Minterms: It is a product of 'n' variables in which each appears exactly once in true or complemented form.

7. Maxterms: It is a sum of the 'n' variables in which each appears exactly once in true or complemented form.

Important Terminologies (Contd..)

Minterms	Maxterms
Input variable is complemented when it has a value “0” in the Truth Table	Input variable is complemented when it has a value “1” in the Truth Table
Denoted by ‘m’	Denoted by ‘M’
Reprsented as SOP form	Represented as POS form

Canonical Form

- Canonical expressions are not simplified form
- These contain redundancies

Canonical sum of Products: It is a complete set of Minterms that define when an output variable is logical 1.

Ex: a'bc+ab'c+abc'

Canonical Product-of-sums: : It is a complete set of Maxterms that define when an output variable is logical 0.

Ex: (a+b+c) (a'+b+c) (a+b'+c)

Canonical Form

Input Variable			Minterm		Maxterm	
a	b	c	Term	Designation	Term	Designation
0	0	0	a’b’c’	m_0	a+b+c	M_0
0	0	1	a’b’c	m_1	a+b+c’	M_1
0	1	0	a’bc’	m_2	a+b’+c	M_2
0	1	1	ab’c’	m_3	a’+b+c	M_3
1	0	0	a’bc	m_4	a+b’+c’	M_4
1	0	1	ab’c	m_5	a’+b+c’	M_5
1	1	0	abc’	m_6	a’+b’+c	M_6
1	1	1	abc	m_7	a’+b’+c’	M_7

Steps to convert SOP to Canonical Form

1. Identifying the missing variable(s) in each AND term
2. AND the missing term and its complement with the original AND term, xy(z+z') since (z+z') = 1
3. Apply the distribution property and expand the term

   i.e. xyz + xyz'

Steps to convert POS to Canonical Form

1. Identifying the missing variable(s) in each OR term
2. OR the missing term and its complement with the original OR term, x+y'+zz' since (zz' = 0)
3. Apply the distribution property and expand the term

   i.e. (x+y'+z) (x+y'+z')

Numerical on Canonical Form

Place the following equations into the proper canonical form

1. P = f(a,b,c) = ab'+ac'+bc

Soln: First term, AND (c+c') with ab': ab'(c+c') = ab'c+ab'c'

Second Term, AND (b+b') with ac' : ac'(b+b') = abc'+ab'c'

Third Term, AND (a+a') with bc : abc + a'bc

→ P = ab'c+ab'c' + abc'+ab'c' + abc + a'bc

P = ab'c+ ab'c' + abc'+ abc + a'bc

2. G = f(w,x,y,z) = w'x+yz' ----- Try it !!

Numerical on Canonical Form

3. T = f(a,b,c) = (a+b')(b'+c) ----- (POS Equn)

Soln:

First term, OR cc' with (a+b') = (a+b'+c) (a+b'+c')

Second Term, OR aa' with (b'+c) = (a+b'+c) (a'+b'+c)

→ T = f(a,b,c) = (a+b'+c) (a+b'+c') (a'+b'+c)

4. J = f(A,B,C,D) = (A+B'+D)(A+B'+C) ---- Try it !!

Generation of Switching Equations from Truth Table

Input Variable			Output Variable
a	b	c	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

P (a'bc +ab'c'+ ab'c+ abc'+abc) => P = f(a,b,c) = $\sum(5,4,6,7,3)$

Numerical examples on Switching Equations from Truth Table

1. The Minterm expression for the output variable M is

$M = f(a,b,m,s) = a'bms+ab'ms+abms$

$\Rightarrow M = f(a,b,m,s) = \sum(7,11,15)$

2. The Maxterm expression for the output variable M is

$M = f(a,b,m,s) = (a'+b+m+s)\ (a+b'+m+s)\ (a+b+m+s)\ (a+b'+m'+s)$

$M = f(a,b,m,s) = \Pi\ (8,11,0,6)$

Numerical examples on Switching Equations from Truth Table

3. H = f(A,B,C) = A'BC+A'B'C+ABC Express the SOP Equation in a minterm list (shorthand decimal notation)

Soln: 1. $A'BC = 011_2 = 3$

2. $A'B'C = 001_2 = 1$

3. $ABC = 111_2 = 7$

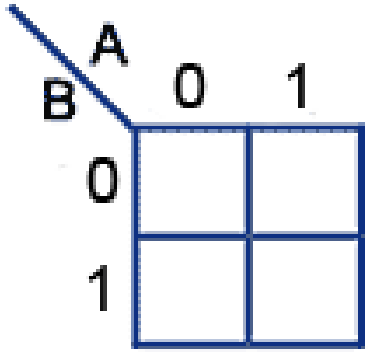
→ $H = f(A,B,C) = \sum(1,3,7)$

Introduction to Karnaugh-Maps

- Karnaugh Maps offer a graphical method of reducing a digital circuit to its minimum number of gates
- It is a Matrix of Squares
- Each square represents a "cell" containing Minterms or Maxterm from a Boolean equation
- K-Maps allows Identification of input variables and helps to reduce the output equation
- The Karnaugh map can be populated with data from either a truth table or a Boolean equation.

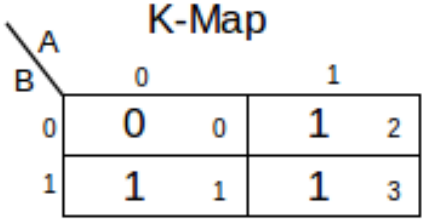
Types of K-Maps

- Circuits with 2 inputs A and B require maps with $2^2 = 4$ cells
- Circuits with 3 inputs A B and C require maps with $2^3 = 8$ cells
- Circuits with 4 inputs A B C and D require maps with $2^4 = 16$ cells
  - 2 Variable K-Map



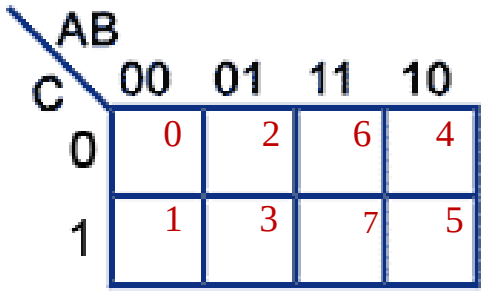
2 Variable Truth Table

	A	B	F
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1



Types of K-Maps

- 3 Variable K-Map



Input Variable binary weighting

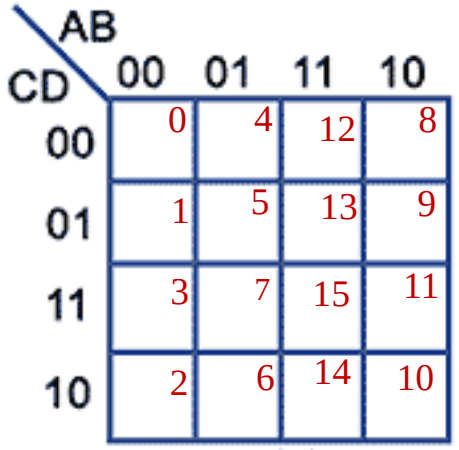
MSB LSB

A B C

- Note:
- Edge numbering does not follow the normal binary counting sequence.
- It uses a Gray Code sequence where only one bit changes from one cell to the next.

Types of K-Maps

- 4 Variable K-Map



Input Variable binary weighting

MSB LSB

A B C D

- Note:
- Edge numbering does not follow the normal binary counting sequence.
- It uses a Gray Code sequence where only one bit changes from one cell to the next.

Rules of Cell Grouping in K-Maps

1. Groups should contain as many ‘1’ cells (i.e. cells containing a logic 1) as possible and no blank cells.
2. Groups can only contain 1, 2, 4, 8, 16 or 32... etc. cells (powers of 2).
3. A ‘1’ cell can only be grouped with adjacent ‘1’ cells that are immediately above, below, left or right of that cell; no diagonal grouping.
4. Groups of ‘1’ cells can overlap. This helps make smaller groups as large as possible, which is an advantage in finding the simplest solution.
5. The top/bottom and left/right edges of the map are considered to be continuous, so larger groups can be made by grouping cells across the top and bottom or left and right edges of the map
6. There should be as few groups as possible.

PI and EPI

1. Prime Implicants (PI): Group of Minterms that can be combined with any other Minterms or groups
2. Essential Prime Implicants (EPI): It is a PI in which one or more Minterms are unique.

ATME College of Engineering

Department of EEE
Emitting Elite Energy

THANK YOU