XI Computer Science 2025

Slides Revision: 1.0

Computing Systems

S.L.O. # 1

Sub Topics: 9 Total SLO: 29

MCQ: (8) 8 Marks CRQ: (3) 9 Marks ERQ: (0) 0 Marks

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

1.1 Data Representation in a Digital Computer

SLO	Students should be able to	Cognitive Level
1.1.1	differentiate between analog and digital signals;	U
1.1.2	explain the binary data representation using binary pulses, i.e., 0/ low/ off and 1/ high/ on;	U

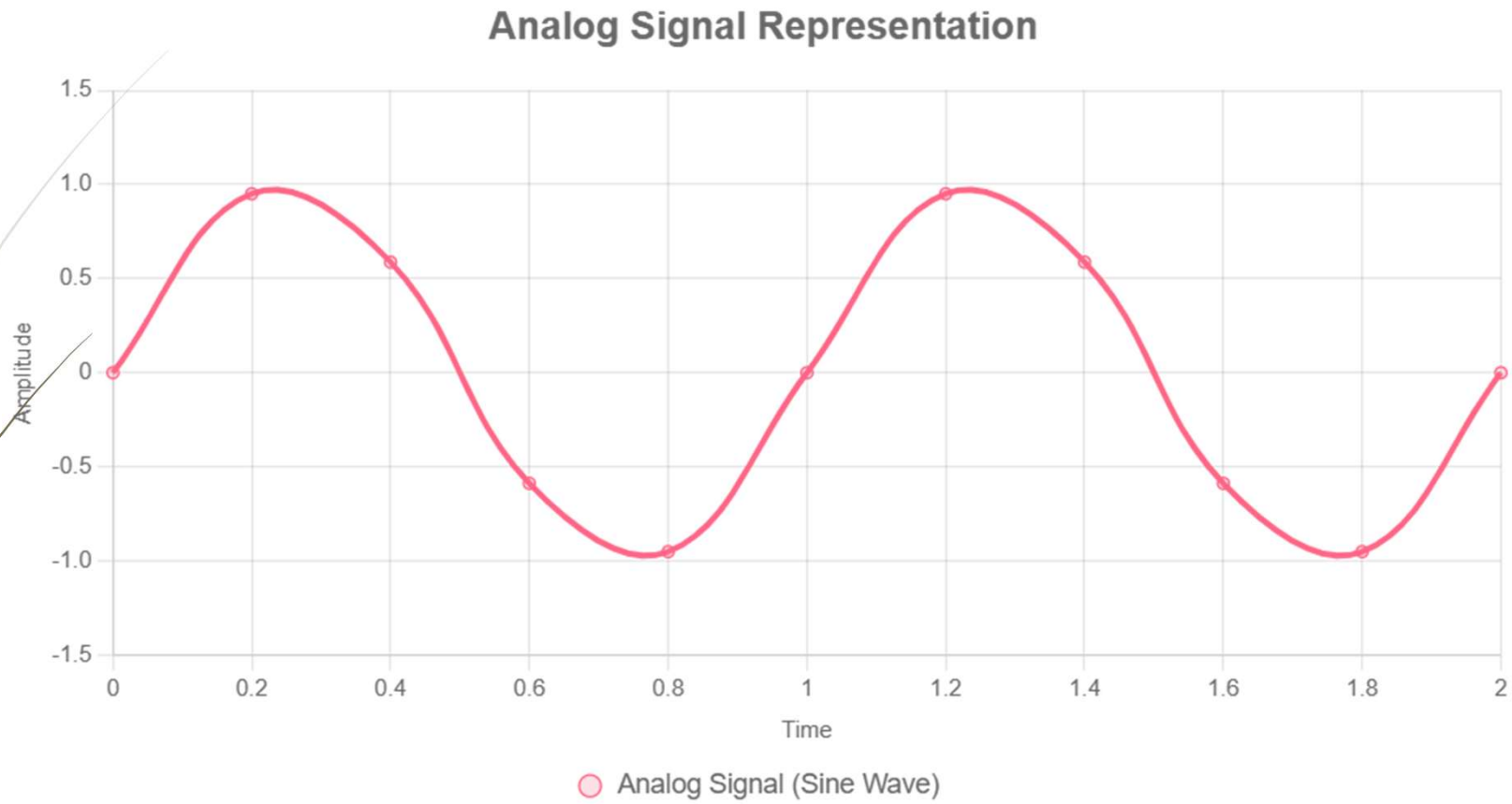
SLO 1.1.1 U

differentiate between analog and digital signals;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Analog vs Digital Signals

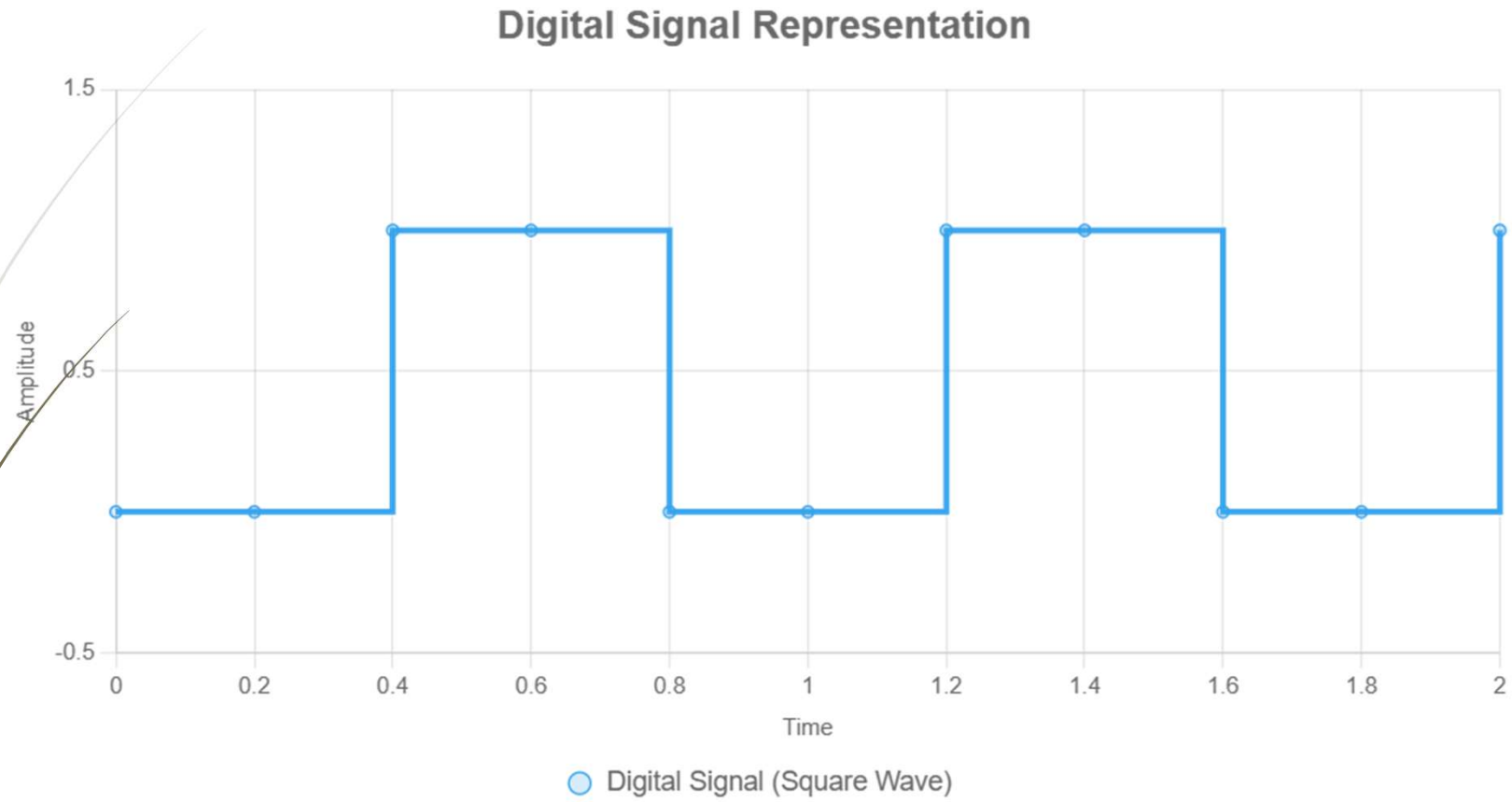
SLO 1.1.1 U



Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Analog vs Digital Signals

SLO 1.1.1 U



Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.1.1 U

Analog vs Digital Signals

Feature	Analog Signal	Digital Signal
Definition	A continuous signal that varies smoothly over time.	A discrete signal that has specific values (0 and 1).
Nature	Continuous in both time and amplitude.	Discrete in time and amplitude.
Signal Form	Sine waves or other smooth curves.	Square waves or binary values.
Representation	Represented by voltage, current, or other continuous parameters.	Represented using binary code (bits: 0s and 1s).
Accuracy	Prone to distortion and noise over distance.	Less affected by noise; more accurate over long distances.

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Analog vs Digital Signals

SLO 1.1.1 U

Feature	Analog Signal	Digital Signal
Bandwidth	Generally, requires more bandwidth.	Requires less bandwidth compared to analog.
Processing	Harder to process and store.	Easier to process, compress, and store.
Examples	Human voice, analog radio, old TV broadcasts.	Computers, digital phones, CDs, digital TV.

SLO 1.1.2 U

explain the binary data representation using binary pulses, i.e., 0/ low/ off and 1/ high/ on;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Binary Data Representation

SLO 1.1.2 U

- In digital electronics and computing, binary data is represented using two distinct states 0 and 1.
- These are often called binary digits or bits, and they form the foundation of all digital communication and processing.

1. Binary Pulses:

- A **binary pulse** is a basic digital signal that switches between two levels to represent binary values:
  - **0 (Zero) → Low / Off / No voltage or current**
  - **1 (One) → High / On / Presence of voltage or current**
- These states are easily distinguishable by electronic systems, which is why they are ideal for data transmission and storage.

Binary Data Representation

SLO 1.1.2 U

2. Representation in Electrical Terms:

Binary Bit	Electrical Signal	Description
0	Low Voltage (e.g., 0V)	No signal, OFF, Low state
1	High Voltage (e.g., 5V or 3.3V)	Signal present, ON, High state

- In most digital systems (like microprocessors), 0 volts represents binary 0, and a specific positive voltage (such as +5V or +3.3V) represents binary 1.

Binary Data Representation

SLO 1.1.2 U

3. Applications:

- **Data transmission** in digital communication (Wi-Fi, USB, etc.)
- **Storage** in memory chips (RAM, SSDs)
- **Processing** in CPUs using logic gates
- Binary pulses use distinct voltage levels to represent bits: 0 as low/off, and 1 as high/on.
- This simple yet powerful method allows computers and digital devices to process and store all kinds of data—from text to images to video.

1.2 Logic Gates

SLO	Students should be able to	Cognitive Level
1.2.1	define the following terms: a. digital logic, b. logic gates and logic circuits, c. truth table;	R
1.2.2	explain the following logic gates in terms of the number of inputs and outputs using truth tables: a. AND, b. OR, c. NOT, d. NAND, e. NOR, f. Exclusive OR (XOR), g. Exclusive NOR (XNOR);	U
1.2.3	identify logic gates from the truth table;	U
1.2.4	explain the uses of logic gates in digital devices;	U
1.2.5	represent the logic circuits in the form of a truth table;	A

1.2 Logic Gates

SLO	Students should be able to	Cognitive Level
1.2.6	explain the following Boolean identities: a. Identity Law, b. Distributive Law, c. Associative Law, d. Commutative Law, e. Inverse (Complement) Law, f. De Morgan's Theorem, g. Absorption Law;	U
1.2.7	construct a logic circuit for a given real-life problem;	A

SLO 1.2.1 R

define the following terms:

a. digital logic,
b. logic gates and logic circuits,
c. truth table;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Define Terms: Digital Logic

SLO 1.2.1 R

- Digital logic is the foundation of digital systems and computing.
- It involves using binary values (0 and 1) to perform logical operations and decision-making in electronic devices.
- These operations are governed by rules of Boolean algebra and are implemented through electronic components like transistors.
- It controls how digital devices like computers, calculators, and microcontrollers process data.
- Digital logic enables operations like addition, comparison, and data storage.

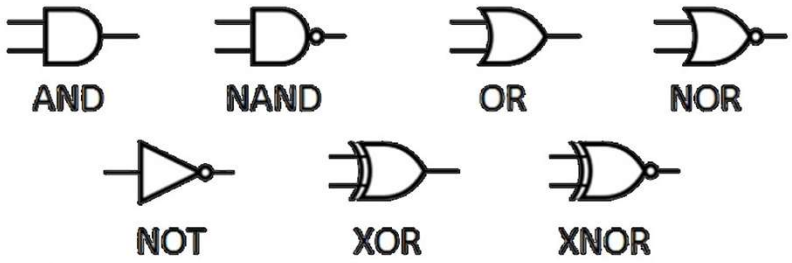
Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

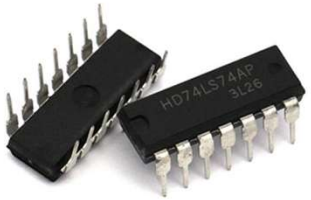
Define Terms: Logic Gates

SLO 1.2.1 R

- Logic gates are basic building blocks of digital circuits.
- They are electronic devices that perform specific logical operations on one or more binary inputs to produce a single binary output.
- Common types: AND, OR, NOT, NAND, NOR, XOR, XNOR
- Example: An AND gate outputs 1 only if all its inputs are 1.

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





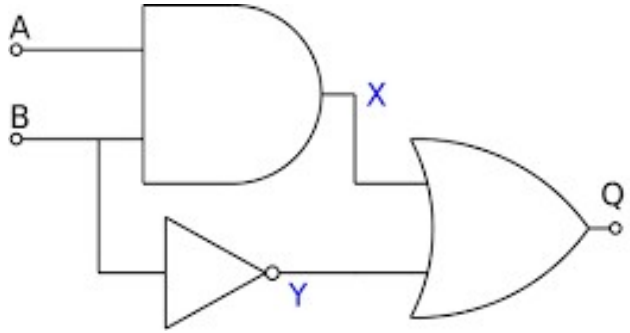
A AND B	$A \cdot B$
A OR B	$A + B$
NOT A	$\overline{A}$
A XOR B	$A \oplus B$

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Define Terms: Logic Circuits

SLO 1.2.1 R

- Logic circuits are combinations of logic gates connected together to perform complex operations.
- Combinational circuits (e.g., adders, multiplexers): Output depends only on current inputs.
- Sequential circuits (e.g., flip-flops, counters): Output depends on current inputs and previous states.



A	B	X	Y	Q
0	0	0	1	1
0	1	0	0	0
1	0	0	1	1
1	1	1	0	1

Define Terms: Truth Table

SLO 1.2.1 R

- A truth table is a chart that shows all possible input combinations to a logic gate or circuit and their corresponding outputs.
- It is used to understand and design logic gates and circuits.
- Each row of the table represents a unique set of input values.
- It helps visualize how a logic gate behaves.

Input A	Input B	Output (A AND B)
0	0	0
0	1	0
1	0	0
1	1	1

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.2 U

explain the following logic gates in terms of the number of inputs and outputs using truth tables:

a. AND,
b. OR,
c. NOT,
d. NAND,
e. NOR,
f. Exclusive OR (XOR),
g. Exclusive NOR (XNOR);

SLO 1.2.2 U

Explain the gate: (a) AND

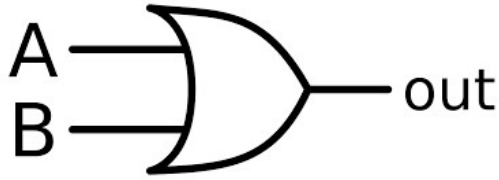
- Inputs: 2 (can be more)
- Output: 1
- Operation: Output is 1 only when all inputs are 1
- Example: A microwave oven only starts when the door is closed and the start button is pressed. Both conditions must be true for the circuit to activate.
- Expression Symbol: $A \cdot B$ or $A\&B$
- Truth Table:

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.2 U

Explain the gate: (b) OR



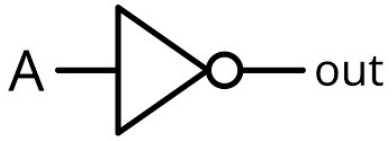
- Inputs: 2 (can be more)
- Output: 1
- Operation: Output is 1 when at least one input is 1
- Example: A car's interior light turns on if either the driver's door or the passenger door is opened.
- Expression Symbol: $A + B$
- Truth Table:

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.2 U

Explain the gate: (c) NOT



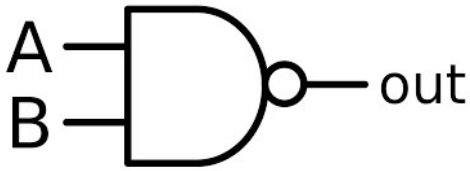
- Inputs: 1
- Output: 1
- Operation: Output is the inverse of the input
- Example: A TV's power LED is off when the system is powered on, and vice versa, using an inverter circuit.
- Expression Symbol: A' or $\overline{A}$ or $\neg A$
- Truth Table:

A	NOT A
0	1
1	0

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.2 U

Explain the gate: (d) NAND (NOT + AND)



- Inputs: 2 (can be more)
- Output: 1
- Operation: Output is 0 only when all inputs are 1; otherwise, 1
- Example: An alarm system that triggers if a window is not closed and motion is detected inside a house.
- Expression Symbol: $\overline{A \cdot B}$
- Truth Table:

A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.2 U

Explain the gate: (e) NOR (NOT + OR)

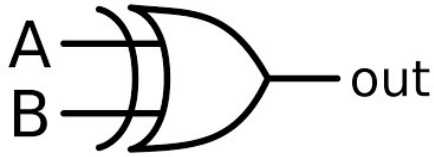
- Inputs: 2 (can be more)
- Output: 1
- Operation: Output is 1 only when all inputs are 0
- Example: A light switch circuit where flipping either switch (upstairs or downstairs) toggles the light, but flipping both does nothing.
- Expression Symbol: $\overline{A + B}$
- Truth Table:

A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.2 U

Explain the gate: (f) Exclusive OR (XOR)



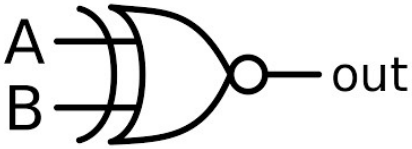
- Inputs: 2
- Output: 1
- Operation: Output is 1 only when inputs are different
- Example: A light switch circuit where flipping either switch (upstairs or downstairs) toggles the light, but flipping both does nothing.
- Expression Symbol: $A \oplus B$
- Truth Table:

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.2 U

Explain the gate: (g) Exclusive NOR (XNOR)



- Inputs: 2
- Output: 1
- Operation: Output is 1 only when inputs are the same
- Example: A keyless car entry system that unlocks only when the code entered matches the stored code exactly (same HIGH/LOW pattern).
- Expression Symbol: $\overline{A \oplus B}$ or $A \odot B$
- Truth Table:

A	B	A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.3 U

identify logic gates from the truth table;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Identify logic gates from the truth table

SLO 1.2.3 U

A	B	OUT
0	0	1
0	1	1
1	0	1
1	1	0

NAND

A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	1

OR

A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	0

XOR

A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	0

NOR

A	B	OUT
0	0	0
0	1	0
1	0	0
1	1	1

AND

A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	1

XNOR

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Identify logic gates from the truth table

SLO 1.2.3 U

A	B	C	OUT
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

OR

A	B	C	OUT
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

NAND

A	B	C	OUT
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

AND

A	B	C	OUT
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

NOR

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.4 U

explain the uses of logic gates in digital devices;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Uses of Logic Gates in Digital Devices

SLO 1.2.4 U

- Logic gates are the fundamental building blocks of digital electronics.
- They perform basic logical functions on binary inputs to produce a binary output, enabling digital devices to process data and make decisions.

1. Performing Arithmetic Operations

- Logic gates are used in the Arithmetic Logic Unit (ALU) of a computer's CPU to perform calculations like addition, subtraction, multiplication, and division.
- For example, an adder circuit (used for adding numbers) is built using XOR, AND, and OR gates to process binary digits (bits). A half-adder, for instance, uses an XOR gate to compute the sum and an AND gate to compute the carry.
- This is how calculators and computers handle math operations.

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Uses of Logic Gates in Digital Devices

SLO 1.2.4 U

2. Data Processing and Decision Making

- Logic gates process inputs to make decisions in digital devices.
- For example, they determine whether a condition is true or false based on input signals.
- In a smartphone, logic gates decide whether to unlock the screen when the correct password is entered (using AND gates to check if all conditions match).
- They also control program flow in software, like deciding which instruction to execute next in a processor.

3. Memory Storage

- Logic gates are used to create memory circuits, such as flip-flops, which store bits of data (0 or 1).
- For example, NAND or NOR gates are combined to form SR latches (Set-Reset latch) or D flip-flops (Data flip-flop), which are the basis of RAM (Random Access Memory) and registers in computers.
- This allows devices to store data temporarily, like saving a phone number in your phone's memory while making a call.

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Uses of Logic Gates in Digital Devices

SLO 1.2.4 U

4. Signal Control and Switching

- Logic gates control signals in digital devices, acting like switches to direct data flow.
- For instance, in a traffic light system, AND and OR gates process sensor inputs to decide when to change lights (e.g., green to red when a timer and sensor signal align).
- In a keyboard, logic gates detect which key is pressed and send the correct signal to the computer.

5. Encoding and Decoding Data

- Logic gates are used in encoders and decoders to convert data into formats devices can understand or display.
- For example, in a digital display (like on a calculator), a decoder circuit made of AND, OR, and NOT gates converts binary numbers into signals that light up specific segments of a 7-segment display to show digits like “5” or “9”.

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Uses of Logic Gates in Digital Devices

SLO 1.2.4 U

6. Error Detection and Correction

- Logic gates help detect and correct errors in data transmission.
- For example, XOR gates are used in parity check circuits to verify if data sent over a network has been corrupted.
- This ensures reliable communication in devices like Wi-Fi routers or USB drives.

7. Digital Communication

- By combining logic gates, complex circuits like multiplexers, demultiplexers, and counters are built. These are used in devices to select data paths, count events, or manage multiple signals.
- For example, in a digital clock, logic gates in a counter circuit keep track of seconds, minutes, and hours.

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.5 A

represent the logic circuits in the form of a truth table;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.5 A

Represent the logic circuits in the form of a truth table

- To represent logic circuits in the form of a truth table, you list all possible input combinations and show the corresponding output for the circuit.
- Each row of the truth table corresponds to one unique combination of inputs, and the output is derived based on the logic gates used in the circuit.
- **Example: Representing a Simple Logic Circuit as a Truth Table**
- Suppose we have a logic circuit with two inputs **A** and **B**, and the output **Y** is:

Y = (A AND B) OR (NOT A)

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.5 A

Represent the logic circuits in the form of a truth table

- **Step 1: List all possible inputs (A and B)**
- Since both A and B are binary, there are $2^2 = 4$ combinations:

A	B
0	0
0	1
1	0
1	1

SLO 1.2.5 A

Represent the logic circuits in the form of a truth table

- **Step 2: Calculate intermediate and final outputs**
- Compute **A AND B**
- Compute **NOT A**
- Compute **Y = (A AND B) OR (NOT A)**

A	B	A AND B	NOT A	Y = (A AND B) OR (NOT A)
0	0	0	1	1
0	1	0	1	1
1	0	0	0	0
1	1	1	0	1

SLO 1.2.6 U

explain the following Boolean identities:

a. Identity Law,
b. Distributive Law,
c. Associative Law,
d. Commutative Law,
e. Inverse (Complement) Law,
f. De Morgan's Theorem,
g. Absorption Law;

Explain Boolean Identities

SLO 1.2.6 U

a. Identity Law

- The identity law states that combining a variable with the identity element (0 or 1) using AND or OR leaves the variable unchanged. This is similar to multiplying by 1 or adding 0 in ordinary algebra.
- For OR:
  - A+0=A
  - Because OR with 0 does not change the value of A.
- For AND:
  - A·1=A
  - Because AND with 1 does not change the value of A.
- Example:
  - If A=1, then A+0=1 and A·1=1.
  - If A=0, then A+0=0 and A·1=0.

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Explain Boolean Identities

SLO 1.2.6 U

b. Distributive Law

- Distributive law allows us to distribute AND over OR and OR over AND, which is useful for expanding or factoring Boolean expressions.
  - AND distributes over OR: $A \cdot (B+C) = (A \cdot B) + (A \cdot C)$
  - OR distributes over AND: $A + (B \cdot C) = (A+B) \cdot (A+C)$
- Example:
  - $A \cdot (B+C)$ means A AND either B OR C. This equals to $(A \cdot B) + (A \cdot C)$ meaning either both A and B are true, or both A and C are true.
  - $A + (B \cdot C)$ means A OR both B AND C true. This is equivalent to $(A+B) \cdot (A+C)$.

Explain Boolean Identities

SLO 1.2.6 U

c. Associative Law

- Associative law states that the way variables are grouped in AND or OR operations does not affect the result.
  - For OR: (A+B)+C=A+(B+C)
  - For AND: (A·B)·C=A·(B·C)
- Example:
  - Whether you evaluate (A+B)+C or A+(B+C), the output is the same. This allows us to omit parentheses when all operators are the same.

Explain Boolean Identities

SLO 1.2.6 U

d. Commutative Law

- Commutative law states that the order of variables in AND or OR operations does not affect the result.
  - For OR: A+B=B+A
  - For AND: A·B=B·A
- Example: A+B is the same as B+A. Similarly, A·B equals B·A. This property lets us reorder terms freely.

Explain Boolean Identities

SLO 1.2.6 U

e. Inverse (Complement) Law

- Inverse law deals with a variable and its complement (NOT version):
  - OR of a variable and its complement:
    $A+A' = 1$
    Means either A is true or A is false, so the result is always true (1).
  - AND of a variable and its complement:
    $A \cdot A' = 0$
    Means A AND NOT A can never be true simultaneously, so the result is always false (0).
- Example:
  - If $A=1$, then $A+A' = 1+0=1$.
  - If $A=0$, then $A \cdot A' = 0 \cdot 1=0$.

Explain Boolean Identities

SLO 1.2.6 U

f. De Morgan's Theorem

- De Morgan's theorem provides rules to simplify the complement (NOT) of AND and OR expressions by converting them:
  - The complement of an AND is the OR of the complements: $(A \cdot B)' = A' + B'$
  - The complement of an OR is the AND of the complements: $(A+B)' = A' \cdot B'$
- This is very useful when simplifying logic expressions or designing circuits with NAND or NOR gates.
- Example:
  - If A=1 and B=0, then: $(A \cdot B)' = (1 \cdot 0)' = 0' = 1$ And $A' + B' = 0+1=1$ Both sides are equal, confirming the theorem.

Explain Boolean Identities

SLO 1.2.6 U

g. Absorption Law

- Absorption law helps to simplify expressions where a variable and a combination involving that variable appear together.
  - A+(A·B)=A Because if A is true, whole expression is true regardless of B.
  - A·(A+B)=A Because if A is false, whole expression is false regardless of B.
- Example:
  - A+(A·B): If A=1, output is 1. If A=0, then A·B=0, so output = 0. So output = A.
  - A·(A+B): If A=1, then A+B=1, so output = 1. If A=0, output = 0. So output = A.

Explain Boolean Identities

SLO 1.2.6 U

a. Identity Law

- $A+0=A$
- $A \cdot 1=A$

b. Distributive Law

- $A \cdot (B+C)=(A \cdot B)+(A \cdot C)$
- $A+(B \cdot C)=(A+B) \cdot (A+C)$

c. Associative Law

- $(A+B)+C=A+(B+C)$
- $(A \cdot B) \cdot C=A \cdot (B \cdot C)$

d. Commutative Law

- $A+B=B+A$
- $A \cdot B=B \cdot A$

e. Inverse (Complement) Law

- $A+A' =1$
- $A \cdot A' =0$

f. De Morgan's Theorem

- $(A \cdot B)' =A' +B'$
- $(A+B)' =A' \cdot B'$

g. Absorption Law

- $A+(A \cdot B)=A$
- $A \cdot (A+B)=A$

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.2.7 A

construct a logic circuit for a given real-life problem;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 1: Automatic Light Control

- A room light should turn ON if:
  - It is dark outside OR
  - The motion sensor detects movement inside the room.
- The light should be OFF otherwise.

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 1: Automatic Light Control

- Step 1: Define Inputs and Output

Input	Symbol	Meaning
Darkness	D	1 = Dark, 0 = Light
Motion detected	M	1 = Movement, 0 = No movement

Output	Symbol	Meaning
Light	L	1 = ON, 0 = OFF

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 1: Automatic Light Control

- Step 2: Write the Logical Expression
- Light turns ON if darkness OR motion detected:

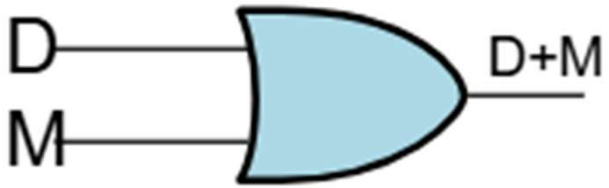
$$L=D+M$$

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 1: Automatic Light Control

- Step 3: Draw the Logic Circuit
- Use an OR gate with inputs D and M.
- The output of the OR gate is connected to the light control.



Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 1: Automatic Light Control

- Step 4: Truth Table

Input D	Input M	Output L = D + M (OR)
0	0	0
0	1	1
1	0	1
1	1	1

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 2: Security Alarm System

- The alarm should sound (activate) if:
  - The door is opened (D = 1), and
  - The security system is armed (A = 1), and
  - The window is opened (W = 1) OR the motion sensor detects movement (M = 1).

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 2: Security Alarm System

- Step 1: Define Inputs and Output

Input	Symbol	Meaning
Door opened	D	1 = Open, 0 = Closed
Security armed	A	1 = Armed, 0 = Disarmed
Window opened	W	1 = Open, 0 = Closed
Motion detected	M	1 = Movement, 0 = No movement

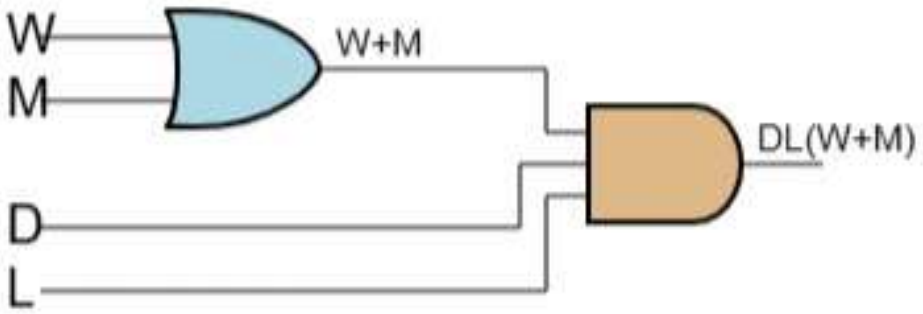
Output	Symbol	Meaning
Alarm	AL	1 = Sound ON, 0 = OFF

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 2: Security Alarm System

- Step 2: Write the Logical Expression
  - Alarm sounds if:
  - Door is opened AND security armed AND (window opened OR motion detected):
  - AL=D·A·(W+M)



Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 2: Security Alarm System

- Step 3: Draw the Logic Circuit Components
- An OR gate with inputs W and M.
- An AND gate with inputs D, A, and output from the OR gate.
- The output of the AND gate is the alarm signal AL.

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example Problem 2: Security Alarm System

- Step 4: Truth Table (Partial for clarity)

D	A	W	M	W + M	AL = D ·A ·(W + M)
0	0	0	0	0	0
1	1	0	0	0	0
1	1	1	0	1	1
1	1	0	1	1	1
1	0	1	1	1	0

Construct a Logic Circuit (Real Life)

SLO 1.2.7 A

Example 3: Vending Machine Coin Validator

- A vending machine dispenses a snack if the correct amount is inserted (e.g., $1, which is two quarters) or if a special employee key is used. This ensures only valid payments or authorized access trigger dispensing.

1.3 Karnaugh Map (K-Map)

SLO	Students should be able to	Cognitive Level
1.3.1	simplify two-variable and three-variable Boolean functions using Karnaugh map;	A
1.3.2	convert the given algebraic expression into its Sum of Products (SOP) and Product of Sums (POS) forms;	A

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.3.1 A

simplify two-variable and three-variable Boolean functions using Karnaugh map;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Karnaugh Map (K-Map)

SLO 1.3.1 A

- Maurice Karnaugh introduced the technique in 1953.
- In many digital circuits and practical problems, we need to find expressions with minimum variables.
- We can minimize Boolean expressions of 2, 3, 4 variables very easily using K-map without using any Boolean algebra theorems.
- It helps to simplify logic into simpler form by organizing grid from truth table values.
- It needs expression are in canonical form (A canonical form is when every term of the expression contains all the variables of the function, either complemented or uncomplemented.)
- K-map can take two forms:
  - Sum of product (SOP)
  - Product of Sum (POS)

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Karnaugh Map (K-Map)

SLO 1.3.1 A

1. Two-variable example (Free Shipping)

- An online shop offers Free Shipping (F)
  - if Price ≥ $50 (A) or
  - the buyer is a Loyalty Member (B).
- Variables:
  - A = 1 if price ≥ $50, else 0
  - B = 1 if loyalty member, else 0
- Boolean Equation
  F = A + B

A	B	F=A+B
0	0	0
0	1	1
1	0	1
1	1	1

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Karnaugh Map (K-Map)

SLO 1.3.1 A

- Sum of Product (SOP)
  - Minterms (where F = 1): rows 1, 2, 3
  - A=0 => A', A=1 => A
  - B=0 => B', B=1 => B
  - Canonical SOP (sum of minterms):
    - F (A, B) = A'B + AB' + AB
    - F (A, B) = $m_0 + m_2 + m_3$
    - F (A, B) = $\Sigma m(1, 2, 3)$
    - F (A, B) = $\Sigma(1, 2, 3)$

A	B	F=A+B
0	0	0
0	1	1
1	0	1
1	1	1

Decimal	A	B	F=A+B
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1

Decimal	A	B	Minterm	F=A+B
0	0	0	A' . B'	0
1	0	1	A' . B	1
2	1	0	A . B'	1
3	1	1	A . B	1

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Karnaugh Map (K-Map)

SLO 1.3.1 A

- Product of Sum (POS)
  - Maxterms (where F = 0): rows 0
  - A=0 => A, A=1 => A'
  - B=0 => B, B=1 => B'
  - Canonical POS (product of maxterms):
    - F(A, B)=(A+B) (already minimal)
    - $F(A, B) = M_0$
    - $F(A, B) = \Pi M(0)$
    - $F(A, B) = \Pi(0)$

A	B	F=A+B
0	0	0
0	1	1
1	0	1
1	1	1

Decimal	A	B	F=A+B
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1

Decimal	A	B	Maxterm	F=A+B
0	0	0	A + B	0
1	0	1	A + B'	1
2	1	0	A' + B	1
3	1	1	A' + B'	1

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.3.1 A

Karnaugh Map (K-Map)

2) Three-variable example (Security System)

- A home Alarm (F) activates if at least two of these are 1: Door (A), Window (B), Motion (C).
  - A = door sensor
  - B = window sensor
  - C = motion sensor
- Boolean equation
  - F=AB+AC+BC

A	B	C	# of 1s	F
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	2	1
1	0	0	1	0
1	0	1	2	1
1	1	0	2	1
1	1	1	3	1

Karnaugh Map (K-Map)

SLO 1.3.1 A

- Sum of Product (SOP)

Decimal	A	B	C	Minterms	F
0	0	0	0	A' B' C'	0
1	0	0	1	A' B' C	0
2	0	1	0	A' B C'	0
3	0	1	1	A' B C	1
4	1	0	0	A B' C'	0
5	1	0	1	A B' C	1
6	1	1	0	A B C'	1
7	1	1	1	A B C	1

- Canonical SOP:
  - F(A, B, C) =A'BC+AB'C+ABC'+ABC
  - $F(A, B, C) = m_3 + m_5 + m_6 + m_7$
  - $F(A, B, C) = \Sigma m(3, 5, 6, 7)$
  - $F(A, B, C) = \Sigma(3, 5, 6, 7)$

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

Karnaugh Map (K-Map)

SLO 1.3.1 A

- Product of Sum (POS)

Decimal	A	B	C	Maxterms	F
0	0	0	0	A + B + C	0
1	0	0	1	A + B + C'	0
2	0	1	0	A + B' + C	0
3	0	1	1	A + B' + C'	1
4	1	0	0	A' + B + C	0
5	1	0	1	A' + B + C'	1
6	1	1	0	A' + B' + C	1
7	1	1	1	A' + B' + C'	1

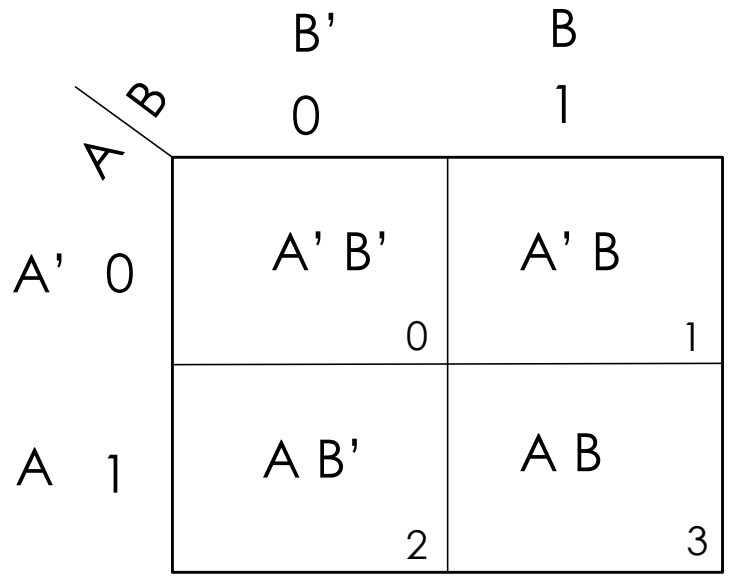
- Canonical POS:
  - F(A, B, C) =(A+B+C)(A+B+C')(A+B'+C)(A'+B+C)
  - $F(A, B, C) = M_0 . M_1 . M_2 . M_4$
  - $F(A, B, C) = \Pi M(0,1,2,4)$
  - $F(A, B, C) = \Pi(0,1,2,4)$

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

2 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Structure of 2 Variable K-Map
  - Number of cells $2^2=4$.



Binary	Decimal
00	0
01	1
10	2
11	3

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

2 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Rules

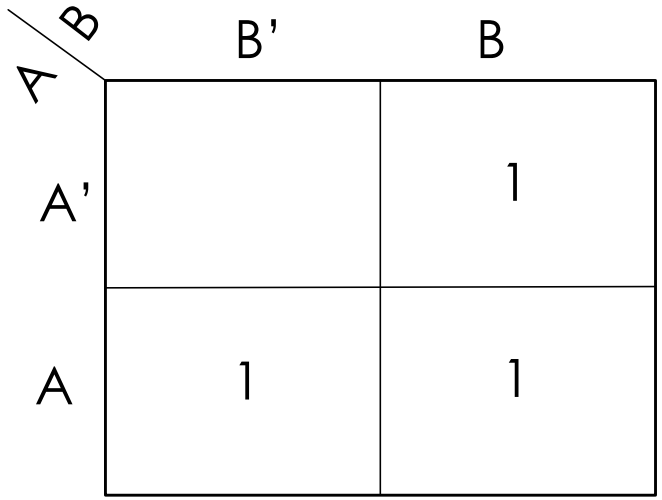
1. Group 1's in powers of 2 → allowed group sizes are 1, 2, or 4.
   - Group of 1 → keep both variables.
   - Group of 2 → one variable eliminated.
   - Group of 4 → all variables eliminated (function = 1).
2. Groups must be rectangular (1×2, 2×1, or 2×2).
3. Wrap-around adjacency allowed
   - Left edge and right edge are adjacent.
   - Top edge and bottom edge are adjacent.
4. Overlap is allowed if it helps form bigger groups.
5. Always make the largest groups possible (maximize simplification).

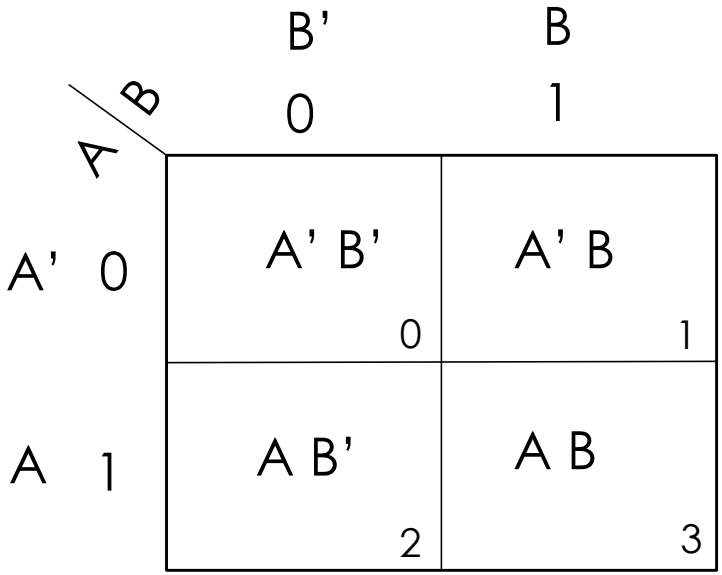
Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

2 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Fill Up
  - F(A, B)=A′B+AB′+AB

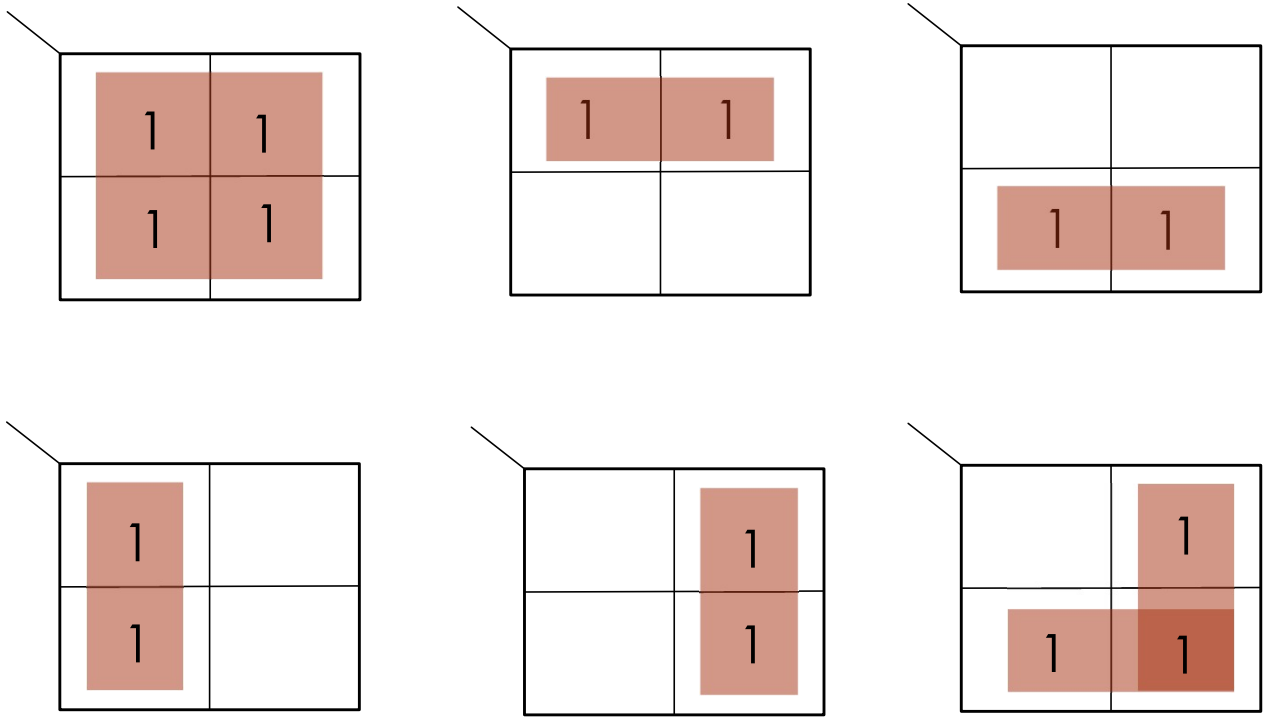




2 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Grouping

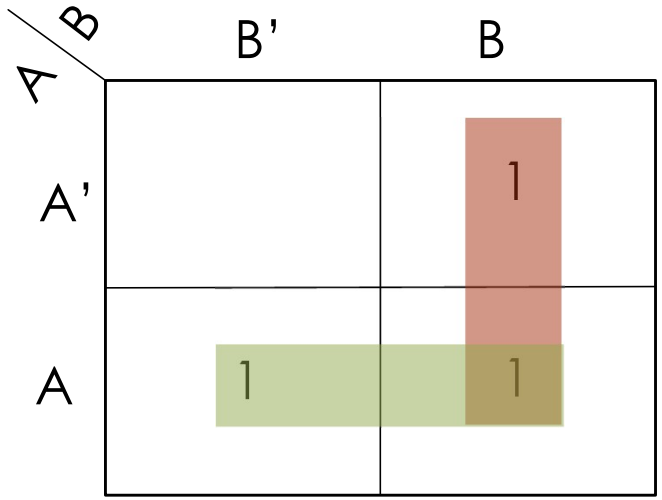


Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

2 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Group Up
  - F(A, B)=A′B+AB′+AB



2 Variable Karnaugh Map (K-Map) SOP

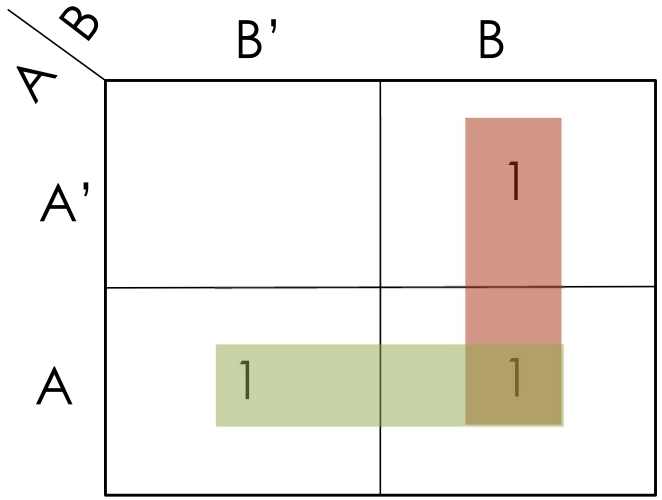
SLO 1.3.1 A

- Simplification outcomes
  - Single 1 (no grouping possible): SOP term = full minterms (e.g., A'B).
  - Pair of 1's in a row/column: eliminates one variable.
    - Example: cells m2 (A=1,B=0) + m3 (A=1,B=1) → group → expression = A.
    - All 4 cells = 1: expression simplifies to 1.
    - No 1's at all: expression simplifies to 0.

2 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Solve
  - F(A, B)=A′B+AB′+AB



- Solution (Method 1)
  - Common in Green Group: A
  - Common in Red Group: B
  - Result is F(A, B) = A + B

2 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Practice Equations
  - F(A,B)=AB′+A′B
  - F(A,B)=Σm(0,1,2)
  - F(A,B)=AB+A′B
  - F(A,B)=A′B′+AB

3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- 3 Variable K-Map Structure

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

Binary	Gray Code
00	00
01	01
10	11
11	10

Binary	Decimal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Rules

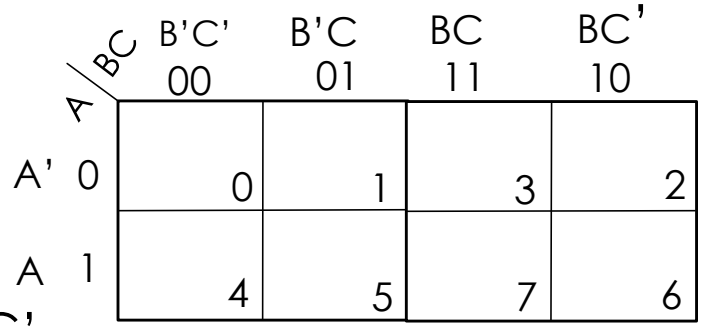
1. Group 1's in powers of 2 → sizes can be 1, 2, 4, or 8.
   - 1 → no simplification (all three variables remain).
   - 2 → one variable eliminated.
   - 4 → two variables eliminated.
   - 8 → all variables eliminated (function = 1).
2. Groups must be rectangular → (1×2, 2×1, 2×2, 4×1).
3. Wrap-around adjacency allowed
   - Leftmost & rightmost columns are adjacent.
   - Top row & bottom row are adjacent.
4. Overlap is allowed if it helps form larger groups.
5. Always make the largest possible groups first.
6. Each group must contain only 1's (no 0's in between).

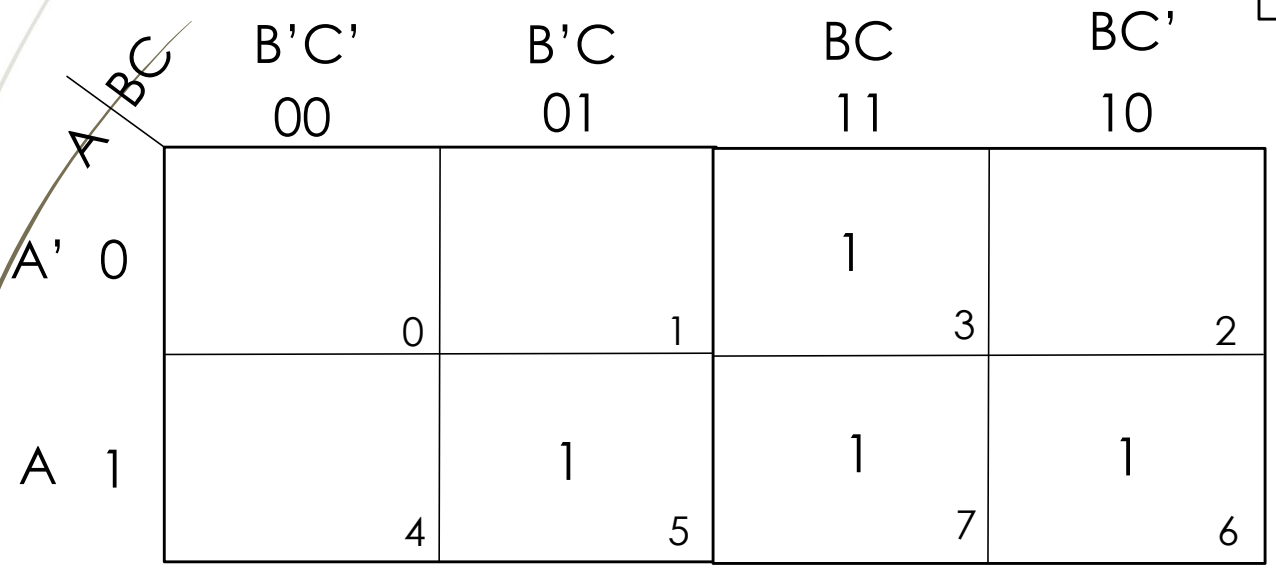
Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Fill Up
  - F(A, B, C) = Σm(3, 5, 6, 7)

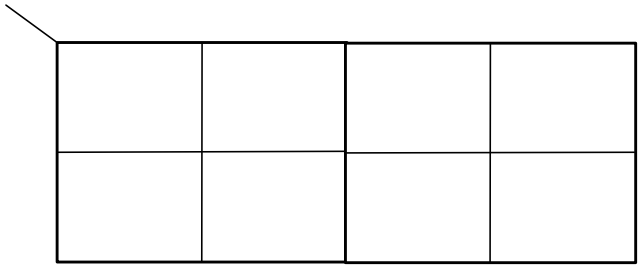


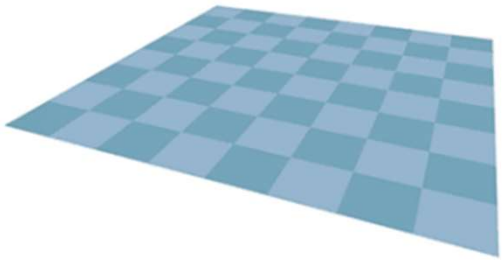


3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Grouping



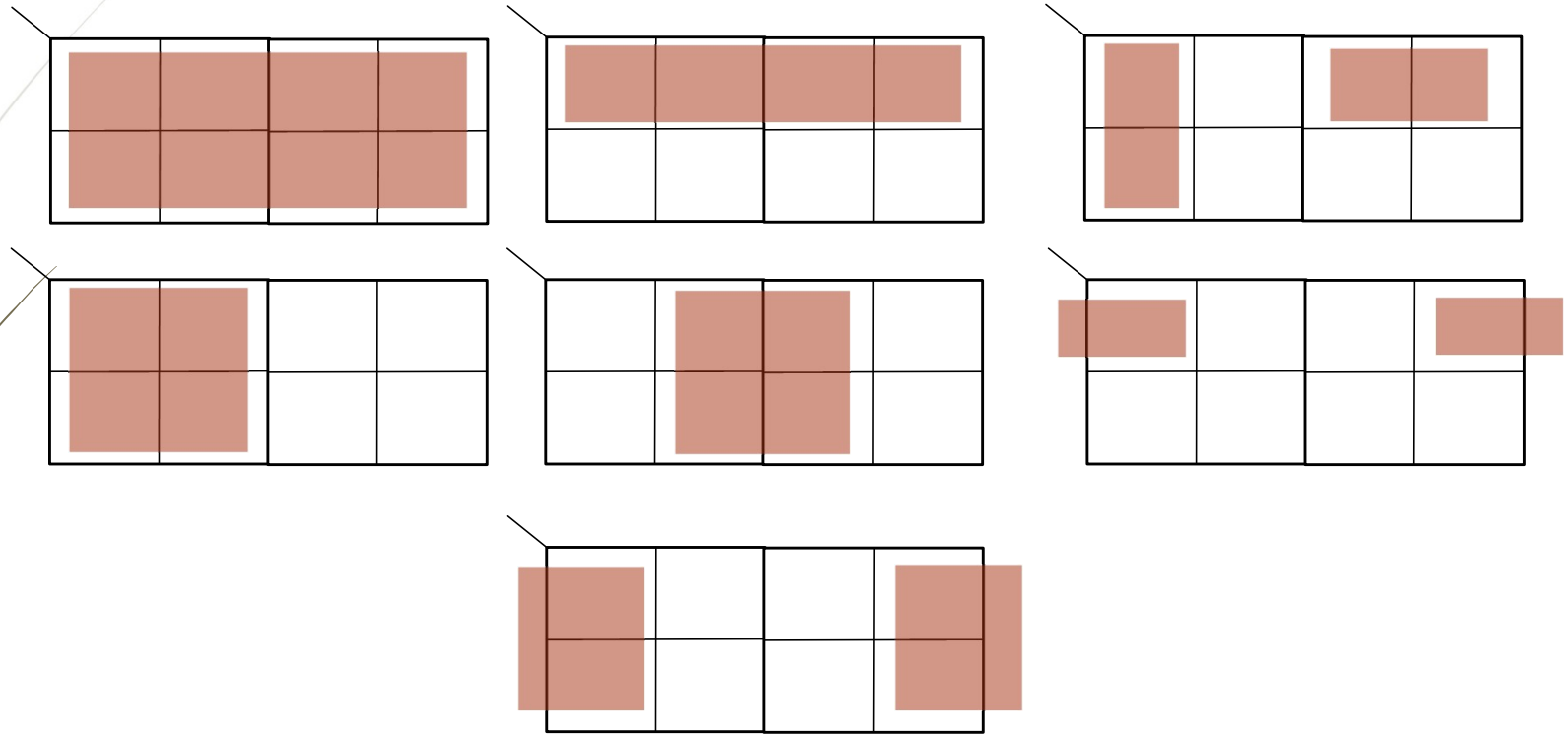


Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Grouping

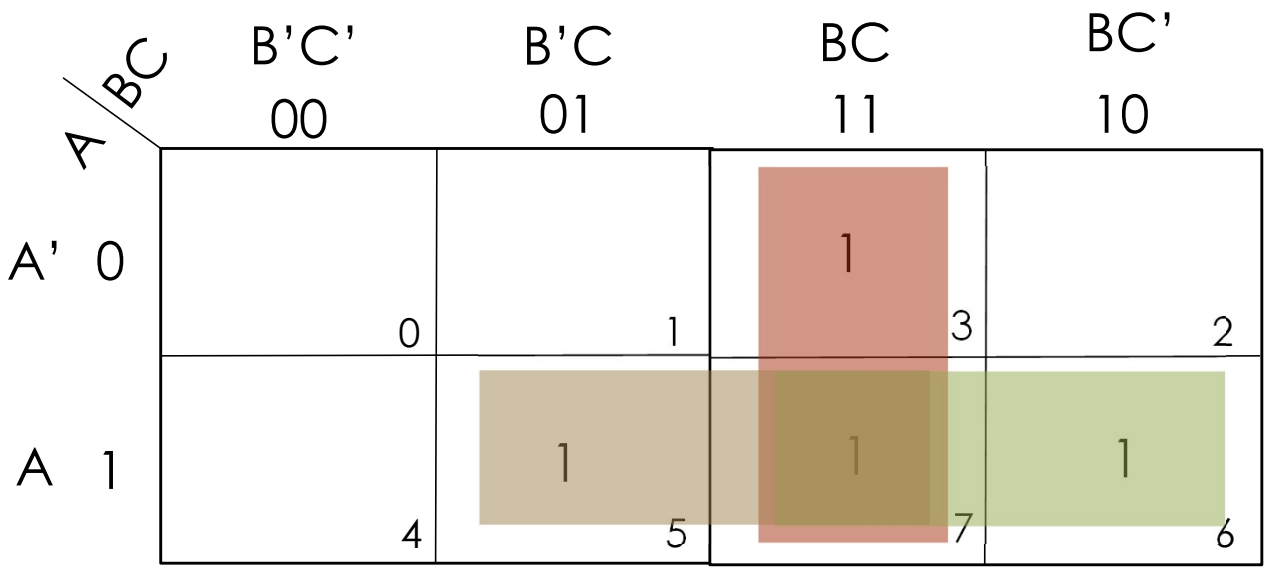


Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Group Up
  - F(A, B, C) = Σm(3, 5, 6, 7)



3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

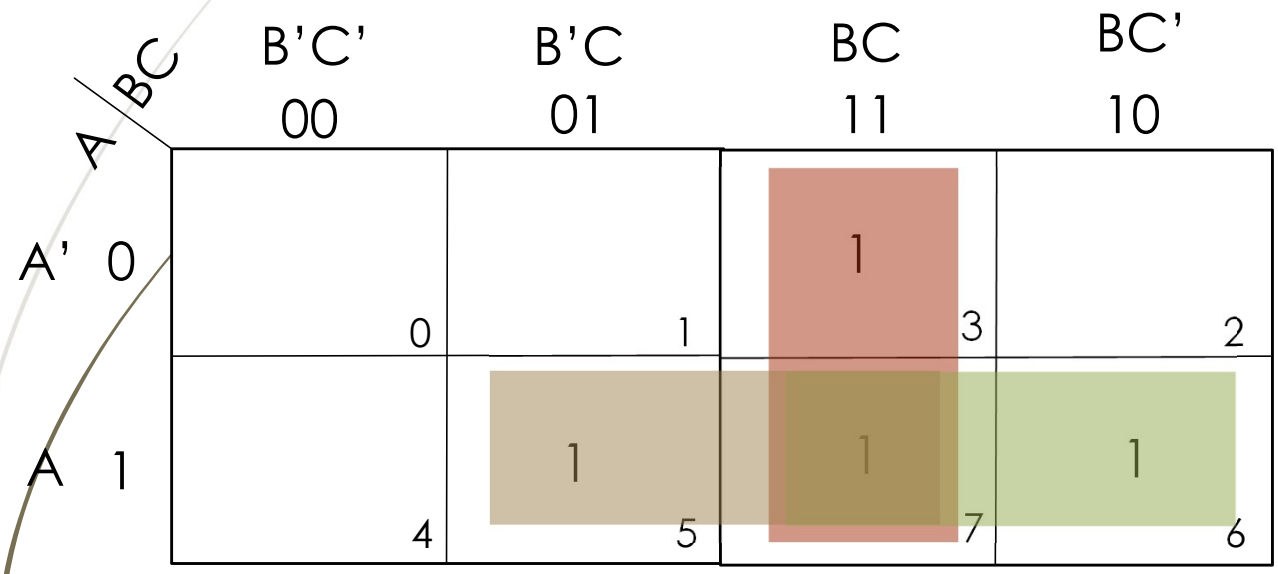
- Simplification outcomes
  - Group of 1 (single cell)→ No simplification (all 3 variables remain).
    - Example: Cell m2=A′BC′.
  - Group of 2 (two adjacent cells)→ Eliminates 1 variable (because it changes within the group).
    - Example: Group m2 (A'BC') + m3 (A'BC) → expression = A′B.
  - Group of 4 (rectangle of 4 adjacent cells)→ Eliminates 2 variables.
    - Example: Group m4,m5,m6,m7 (entire row A=1) → expression = A.
  - Group of 8 (all cells are 1)→ Eliminates all 3 variables. Expression = 1 (always true).

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Solve
  - F(A, B, C) = Σm(3, 5, 6, 7)



- Solution (Method 1)
  - G1: BC
  - G2: AC
  - G3: AB
- F = BC + AC + AB

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

3 Variable Karnaugh Map (K-Map) SOP

SLO 1.3.1 A

- Practice Equations
  - F(A,B,C)=A′B′C′+A′BC+AB′C
  - F(A,B,C)=Σm(1,3,5,7)
  - F(A,B,C)=Σm(0,2,4,6)
  - F(A,B,C)=Σm(1,2,3,7)

SLO 1.3.2 A

convert the given algebraic expression into its Sum of Products (SOP) and Product of Sums (POS) forms;

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.3.2 A

Convert the given algebraic expression into its SOP and POS forms

SOP for F(A,B,C)=AB′+BC

We must make sure every term has all three variables (A, B, C).

- Step A — Expand each product term
  - For AB′ : Add missing variable C:
    AB′ = AB′(C+C′) = AB′C+AB′C′ → Distributive Law
  - For BC: Add missing variable A:
    BC=BC(A+A′)=ABC+A′BC
- Step B — Collect terms
  F=AB′C+AB′C′+ABC+A′BC

SLO 1.3.2 A

Convert the given algebraic expression into its SOP and POS forms

SOP for F(A,B,C)=AB′+BC

- Step B — Collect terms
  F=AB′C+AB′C′+ABC+A′BC
- Step C — Index minterms
  - Write as minterms Σm().
    - AB′C → A=1, B=0, C=1 → binary 101 → decimal 5
    - AB′C′ → 100 → decimal 4
    - ABC → 111 → decimal 7
    - A′BC → 011 → decimal 3
    - F=Σm(3,4,5,7)

Binary	Decimal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.3.2 A

Convert the given algebraic expression into its SOP and POS forms

POS for F(A,B,C)=AB′+BC

- Step A — Build truth table quickly

A	B	C	AB'+BC	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	1+0=1	1
1	0	1	1+0=1	1
1	1	0	0+0=0	0
1	1	1	0+1=1	1

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903

SLO 1.3.2 A

Convert the given algebraic expression into its SOP and POS forms

POS for F(A,B,C)=AB′+BC

- Step B — Write maxterms for zeros
  - Row 0: (A=0,B=0,C=0) → (A+B+C)
  - Row 1: (0,0,1) → (A+B+C′)
  - Row 2: (0,1,0) → (A+B′+C)
  - Row 6: (1,1,0) → (A′+B′+C)
- Step C — Product of all maxterms
  F=(A+B+C)(A+B+C′)(A+B′+C)(A′+B′+C)
- Step D — Product of all maxterms
  F=ΠM(0,1,2,6)

No	A	B	C	AB'+BC	F	Maxterm
0	0	0	0	0+0=0	0	A+B+C
1	0	0	1	0+0=0	0	A+B+C′
2	0	1	0	0+0=0	0	A+B′+C
3	0	1	1	0+0=0	0	
4	1	0	0	1+0=1	1	
5	1	0	1	1+0=1	1	
6	1	1	0	0+0=0	0	A′+B′+C
7	1	1	1	0+1=1	1	

Prepared By: Naseer Ahmad (Nusrat Jahan Boys College) 0333-9790903