

# Subject

05BC2102

## Architectural Organization of Computers

Unit#1



**Marwadi**  
University

Department of  
Computer Application

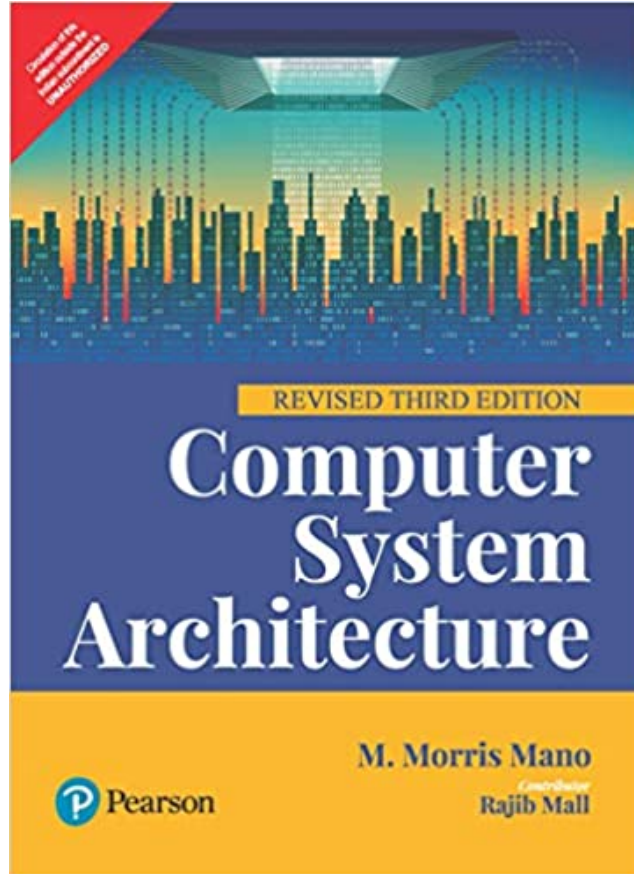
Subject:  
Architectural  
Organization of  
Computers



## Course Objectives

- 1. To understand basic organization of digital computer.
- 2. To understand various digital circuits and components.
- 3. To understand basics of CPU.
- 4. To understand basics of IO.
- 5. To understand basics of Memory.

# Books



- “Computer System Architecture”, M. Morris Mano, Pearson Publication, Third Edition.

## UNIT – 1 Digital Logic Circuits

- Digital Computers
- Logic Gates
- Boolean Algebra

# Digital Computers

- Digital computer, any of a class of devices capable of solving problems by processing information in discrete form.
- It operates on data, including magnitudes, letters, and symbols, that are expressed in binary code—i.e., using only the two digits 0 and 1.
- By counting, comparing, and manipulating these digits or their combinations according to a set of instructions held in its memory, a digital computer can perform such tasks as to control industrial processes and regulate the operations of machines

# Digital Computers

- LAPTOP
- DESKTOP
- MOBILE
- PALMTOP



# Digital Computers

- Digital computers use the binary number system, which has two digits 0 and 1.
- A binary digit is called a bit.
- Information is represented in digital computers in groups of bits.
- By using various coding techniques, groups of bits can be made to represent not only binary numbers but also other discrete symbols, such as decimal digits or letters of the alphabet.

# Digital Computers

- Decimal Number System :
- Decimal Number System comprises digits from 0-9 that are 0, 1, 2, 3, 4, 5, 6, 7, 8 & 9. The base or radix of the Decimal Number System is 10 because the total number of digits available in the Decimal Number System is 10. All the other digits can be expressed with the help of these 10 digit numbers.
- Decimal Number System is the most common and easiest number system used in our daily lives. Some of the Decimal Number System examples are:
- 341, 56, 6789, 78.



# Digital Computers

- Binary Number System.
- Digital Computers use the binary number system to manipulate and store all of their data including numbers, words, videos, graphics, and music.
- The binary number system, also called the **base-2 number system**, is a method of representing numbers that counts by using combinations of only two numerals: zero (0) and one (1).
- Some example are
  - 101, 1100101, 111000, 0001010

- Convert Decimal to binary.
- **Step 1:** Divide the given decimal number by 2 and note down the remainder.
- **Step 2:** Now, divide the obtained quotient by 2, and note the remainder again.
- **Step 3:** Repeat the above steps until you get 0 as the quotient.
- **Step 4:** Now, write the remainders in such a way that the last remainder is written first, followed by the rest in the reverse order.
- This number is the binary value of the given decimal number.

# Decimal to binary

- For example  $100 = (?)_2$

	number	Remainder
2	100	
2	50	0
2	25	0
2	12	1
2	6	0
2	3	0
2	1	1
2	0	1

- Writing from bottom to top we get
- $(1100100)_2$

# Decimal to binary

- Convert 356 decimal to binary

# Decimal to binary

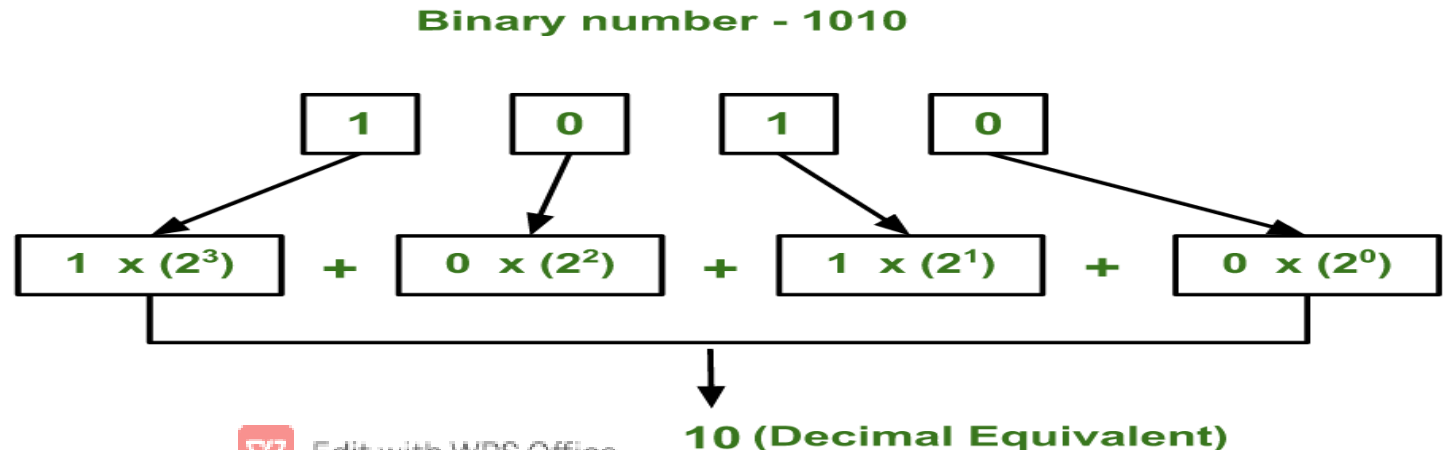
	Number	Remainder
2	356	
2	178	0
2	89	0
2	44	1
2	22	0
2	11	0
2	5	1
2	2	1
2	1	0
2	0	1

- Writing from bottom to top we get
- $(101100100)_2$

## Unit#1 Digital Logic Circuits:

# Binary to decimal

- Convert binary to decimal.
- **Step-1** Write down all the binary digit.
- **Step-2** Multiply each digit by 2.
- **Step-3** Starting from the left give the power to 2 in sequence from 0 to n-1.
- **Step-4** Sum all the numbers.



# Binary to decimal

- Convert 11010101 to decimal

### Binary to decimal

$$1 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

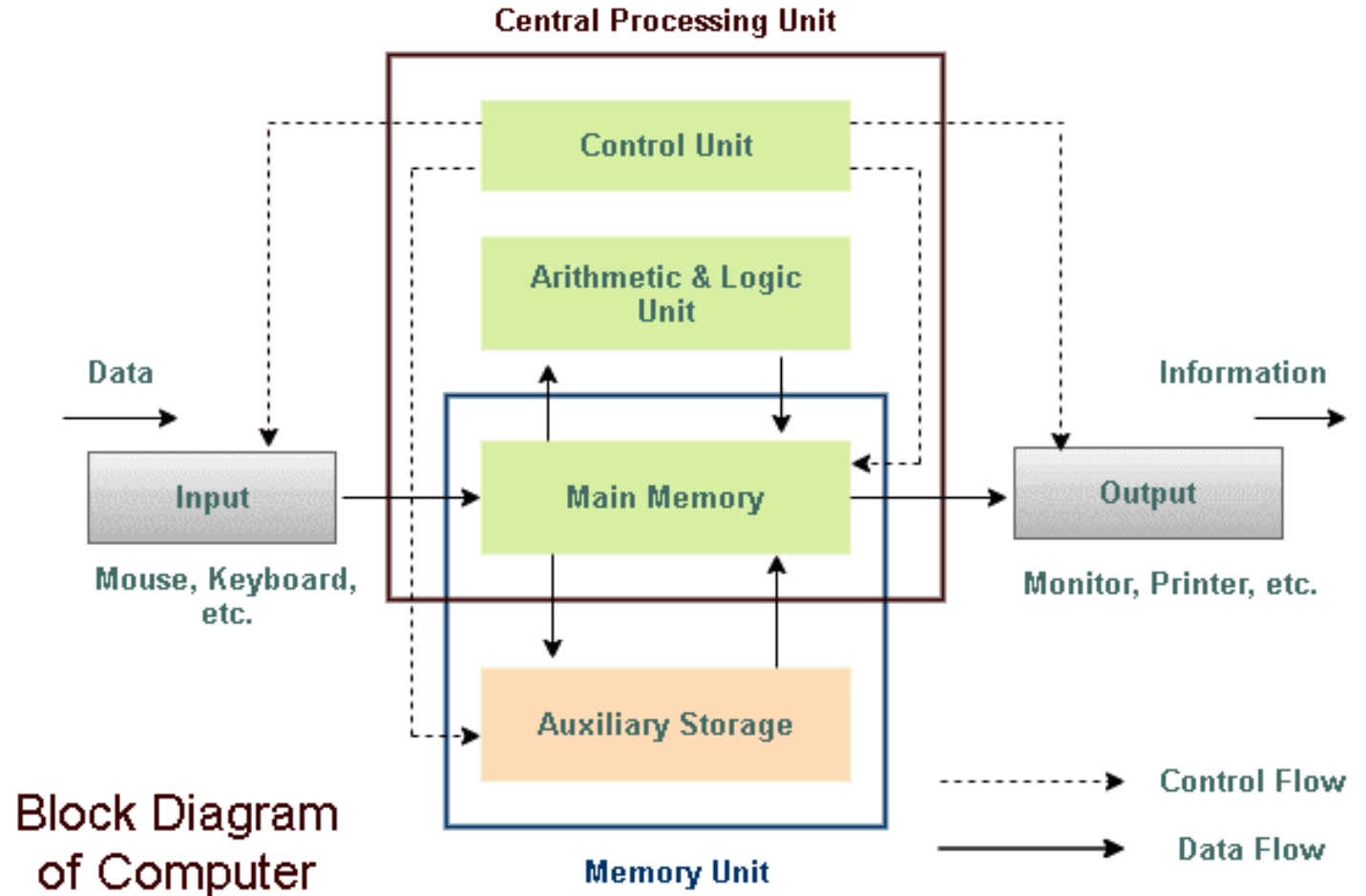
$$1 \times 128 + 1 \times 64 + 0 + 1 \times 16 + 0 + 1 \times 4 + 0 + 1 \times 1$$

$$128 + 64 + 0 + 16 + 0 + 4 + 0 + 1 = 213$$

So 11010101 binary is = 213 decimal



# Digital Computers



# Digital Computers

- Input
- All the data received by the computer goes through the input unit. The input unit comprises different devices. Like a mouse, keyboard, scanner, etc. In other words, each of these devices acts as a mediator between the users and the computer.
- The data that is to be processed is put through the input unit. The computer accepts the raw data in binary form. It then processes the data, and produces the desired output.

# Digital Computers

- The 3 major functions of the input unit are-
- Take the data to be processed by the user.
- Convert the given data into machine-readable form.
- And then, transmit the converted data into the main memory of the computer. The sole purpose is to connect the user and the computer. In addition, this creates easy communication between them.

# Digital Computers

- CPU – Central Processing Unit
- Central Processing Unit or the CPU, is the brain of the computer. It works the same way a human brain works. As the brain controls all human activities, the CPU too controls all tasks.
- Moreover, the CPU conducts all the arithmetical and logical operations in the computer.
- Now the CPU comprises of two units, namely – ALU (Arithmetic Logic Unit) and CU (Control Unit). Both of these units work in sync. The CPU processes the data as a whole.

- **ALU – Arithmetic Logic Unit**
- The Arithmetic Logic Unit is made of two terms, arithmetic and logic. There are two major functions that this unit performs.
  1. Data inserted through the input unit into the primary memory. Performs the basic arithmetical operation on it. Like addition, subtraction, multiplication, and division. It performs all sorts of calculations required on the data. Then sends back data to the storage.
  2. The unit is also responsible for performing logical operations like, AND, OR, Equal to, Less than, etc. In addition to this it conducts merging, sorting, and selection of the given data.

# Digital Computers

- **CU – Control Unit**
- The control unit as the name suggests is the controller of all the activities/tasks and operations. All this is performed inside the computer.
- The memory unit sends a set of instructions to the control unit. Then the control unit in turn converts those instructions. After that these instructions are converted to control signals.

These control signals help in prioritizing and scheduling the activities. Thus, the control unit coordinates the tasks inside the computer in sync with the input and output units.

# Digital Computers

- **Memory Unit**
- All the data that has to be processed or has been processed is stored in the memory unit. The memory unit acts as a hub of all the data. It transmits it to the required part of the computer whenever necessary.
- The memory unit works in sync with the CPU. This helps in faster accessing and processing of the data. Thus, making tasks easier and faster.
- There are two types of computer memory-
  - Primary memory
  - Secondary memory

# Digital Computers

- Output
- There is nothing to be amazed by what the output unit is used for. All the information sent to the computer once processed is received by the user through the output unit. Devices like printers, monitors, projector, etc. all come under the output unit.
- The output unit displays the data either in the form of a soft copy or hard copy. The printer is for the hard copy. The monitor is for the display. The output unit accepts the data in binary form from the computer. It then converts it into a readable form for the user.



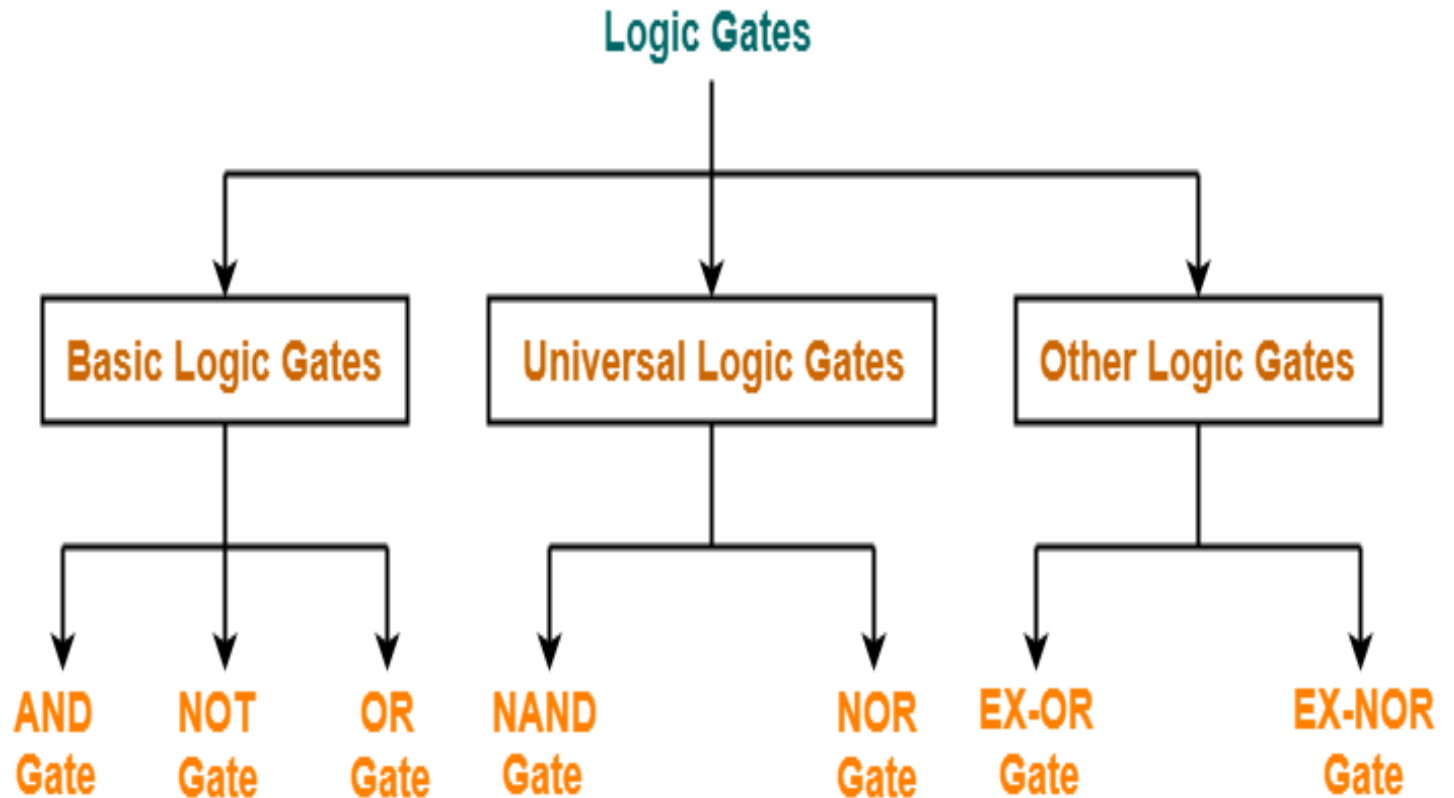
# UNIT – 1 Digital Logic Circuits

- Digital Computers
- Logic Gates
- Boolean Algebra
- Map Simplification
- Combinational Circuits
- Flip-Flops
- Sequential circuits

# Logic Gates

- Logic gates are the basic building blocks of any digital system. It is an electronic circuit having one or more than one input and only one output.
- Logic gates has two state either 1 (high) or 0 (low)
- At any one time, a digital device will be in one of these two binary situations. A light bulb can be used to demonstrate the operation of a logic gate. When logic 0 is supplied to the switch, it is turned off, and the bulb does not light up. The switch is in an ON state when logic 1 is applied, and the bulb would light up. In integrated circuits (IC), logic gates are widely employed.

# Logic Gates

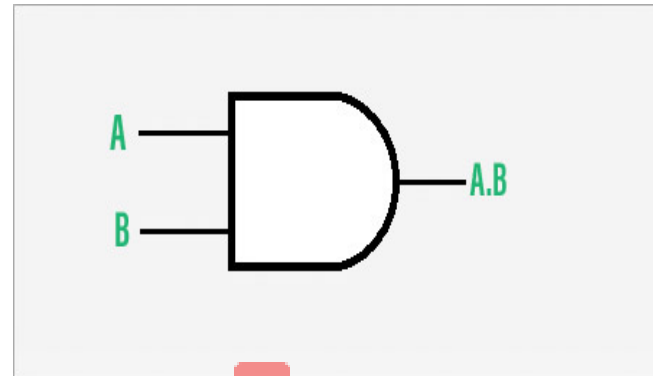


Types of Logic Gates

# Logic Gates AND gate

- An AND gate has a single output and two or more inputs.
- 1. When all of the inputs are 1, the output of this gate is 1.
- 2. The AND gate's Boolean logic is  $Y=A.B$  if there are two inputs A and B.
- An AND gate's symbol and truth table are as follows:

BLOCK DIAGRAM / SYMBOL



TRUTH TABLE

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

## Unit#1 Digital Logic Circuits:

### Logic Gates AND gate

Number of possible combination depend upon the variable so if n input are there then  $2^n$  possible combination possible

So if 3 variable are there then  $2^3 = 8$  possibility

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

Logic Gates  
AND gate

3 Input AND Gate Truth Table

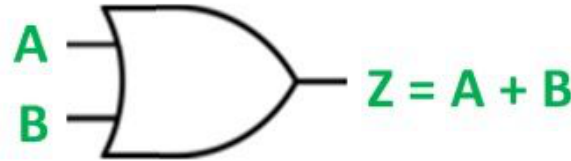
Inputs			Outputs
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



# Logic Gates OR gate

- Two or more inputs and one output can be used in an OR gate. An or gate has single output and two or more inputs
- The logic of this gate is that if at least one of the inputs is 1, the output will be 1.
- The OR gate's output will be given by the following mathematical procedure if there are two inputs A and B:  $Y=A+B$

BLOCK DIAGRAM / SYMBOL



TRUTH TABLE

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

# Logic Gates OR gate

TRUTH TABLE

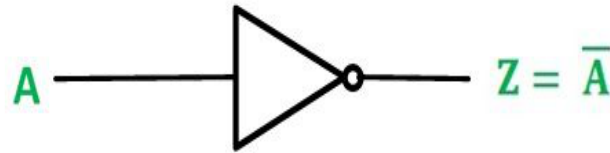
INPUTS			OUTPUT
W	X	Y	Z
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1



# Logic Gates NOT gate

- The NOT gate is a basic one-input, one-output gate.
1. When the input is 1, the output is 0 and vice versa. A NOT gate is sometimes called as an inverter because of its feature.
  2. If there is only one input A, the output may be calculated using the Boolean equation  $Y=A'$ .

BLOCK DIAGRAM / SYMBOL



TRUTH TABLE

A	A'
0	1
1	0

# Logic Gates

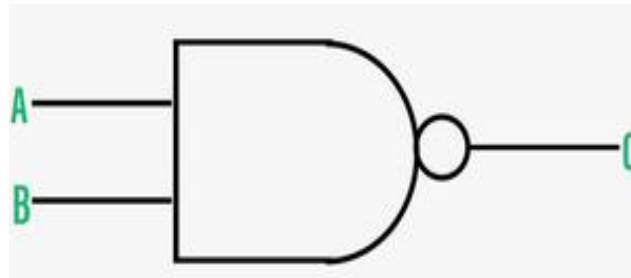
- Universal logic gates are the logic gates that are capable of implementing any Boolean function without requiring any other type of gate.
- They are called as “**Universal Gates**” because-
  - They can realize all the binary operations.
  - All the basic logic gates can be derived from them.
- There are following two universal logic gates
  1. NAND Gate
  2. NOR Gate
- Explain in next slide.

this gate output is 0 when the all inputs are 1 otherwise output of this gate is 1.

## Logic Gates NAND gate

- A NAND gate, sometimes known as a 'NOT-AND' gate, is essentially a Not gate followed by an AND gate.
1. This gate's output is 1 only if none of the inputs is 1. Alternatively, when all of the inputs are not high and at least one is low, the output is high.
  2. If there are two inputs A and B, the Boolean expression for the NAND gate is  $Y=(A.B)'$

BLOCK DIAGRAM / SYMBOL



TRUTH TABLE

A	B	$(A.B)'$
0	0	1
0	1	1
1	0	1
1	1	0

# Logic Gates NAND gate

A	B	C	O
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

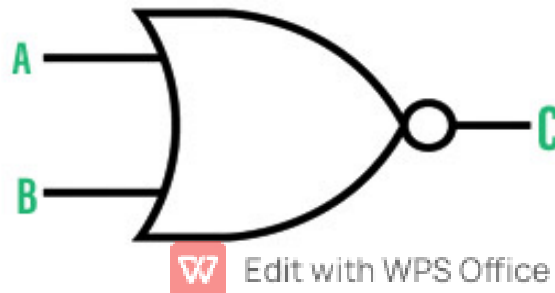
# Logic Gates NOR gate

- A NOR gate, sometimes known as a “NOT-OR” gate, consists of an OR gate followed by a NOT gate.
1. This gate's output is 1 only when all of its inputs are 0. Alternatively, when all of the inputs are low, the output is high.
  2. The Boolean statement for the NOR gate is  $Y = (A+B)'$  if there are two inputs A and B.

TRUTH TABLE

A	B	$(A+B)'$
0	0	1
0	1	0
1	0	0
1	1	0

BLOCK DIAGRAM / SYMBOL



### Logic Gates NOR gate

Input			Output
A	B	C	Y
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

# Logic Gates Ex-OR gate

- The Exclusive-OR or 'Ex-OR' gate is a digital logic gate that accepts more than two inputs but only outputs one value.
1. If number of the 'High' inputs are odd then the output of the XOR Gate is 'High', If number of the 'High' inputs are even the output is 'Low'.
  2. The Boolean equation for the XOR gate is  $Y = A'.B + A.B'$  if there are two inputs A and B.

BLOCK DIAGRAM / SYMBOL



TRUTH TABLE

A	B	(A xor B)
0	0	0
0	1	1
1	0	1
1	1	0



### Logic Gates Ex-OR gate

Inputs			Output
A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



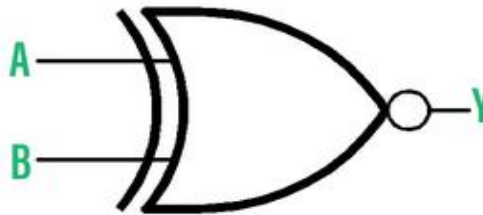
# Logic Gates

## Ex-NOR gate

- The Exclusive-NOR or 'EX-NOR' gate is a digital logic gate that accepts more than two inputs but only outputs one.
- If number of the 'High' inputs are even then the output of the XOR Gate is 'High', If number of the 'High' inputs are odd the output is 'Low'.
  - If there are two inputs A and B, then the XNOR gate's Boolean equation is:  $Y = A.B + A'B'$ .

TRUTH TABLE

BLOCK DIAGRAM / SYMBOL



A	B	(A XNOR B)
0	0	1
0	1	0
1	0	0
1	1	1







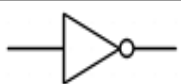
### Logic Gates Ex-NOR gate

Inputs			outputs
W	X	Y	$Q = A \oplus B \oplus C$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

# Unit#1 Digital Logic Circuits:

- Summary

## Logic Gates

Logic Gate	Symbol	Description	Boolean
AND		Output is at logic 1 when, and only when all its inputs are at logic 1, otherwise the output is at logic 0.	$X = A \cdot B$
OR		Output is at logic 1 when one or more are at logic 1. If all inputs are at logic 0, output is at logic 0.	$X = A + B$
NAND		Output is at logic 0 when, and only when all its inputs are at logic 1, otherwise the output is at logic 1	$X = \overline{A \cdot B}$
NOR		Output is at logic 0 when one or more of its inputs are at logic 1. If all the inputs are at logic 0, the output is at logic 1.	$X = \overline{A + B}$
XOR		Output is at logic 1 when one and Only one of its inputs is at logic 1. Otherwise is it logic 0.	$X = A \oplus B$
XNOR		Output is at logic 0 when one and only one of its inputs is at logic 1. Otherwise it is logic 1. Similar to XOR but inverted.	$X = \overline{A \oplus B}$
NOT		Output is at logic 0 when its only input is at logic 1, and at logic 1 when its only input is at logic 0. That's why it is called and INVERTER	$X = \overline{A}$

## Logic Gates

NOT



INPUT		OUTPUT
A		
0		1
1		0

AND



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

OR



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

XOR



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

NAND



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

NOR



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

XNOR

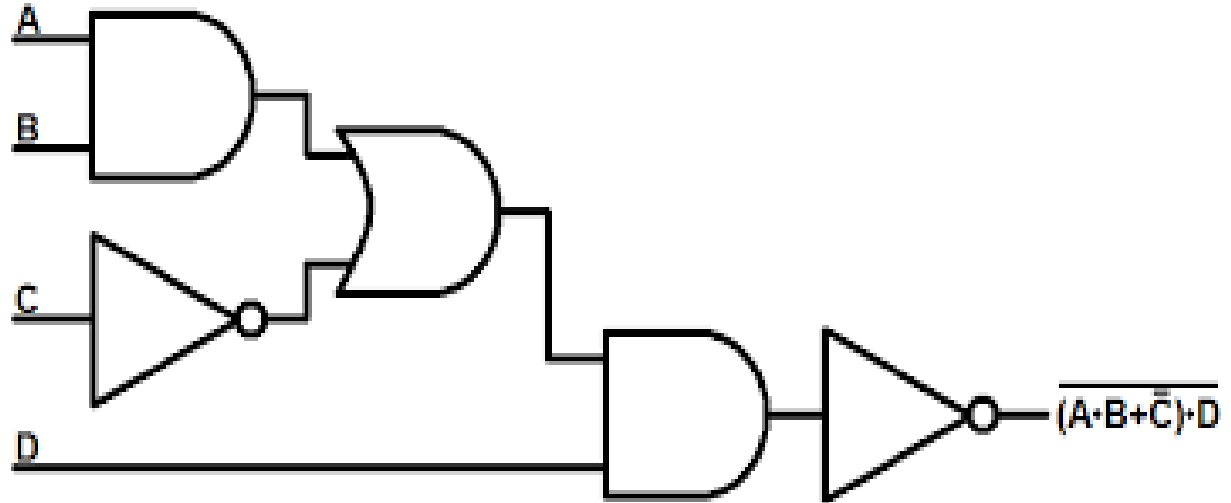


INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	1

## Unit#1 Digital Logic Circuits:

### Logic Diagram

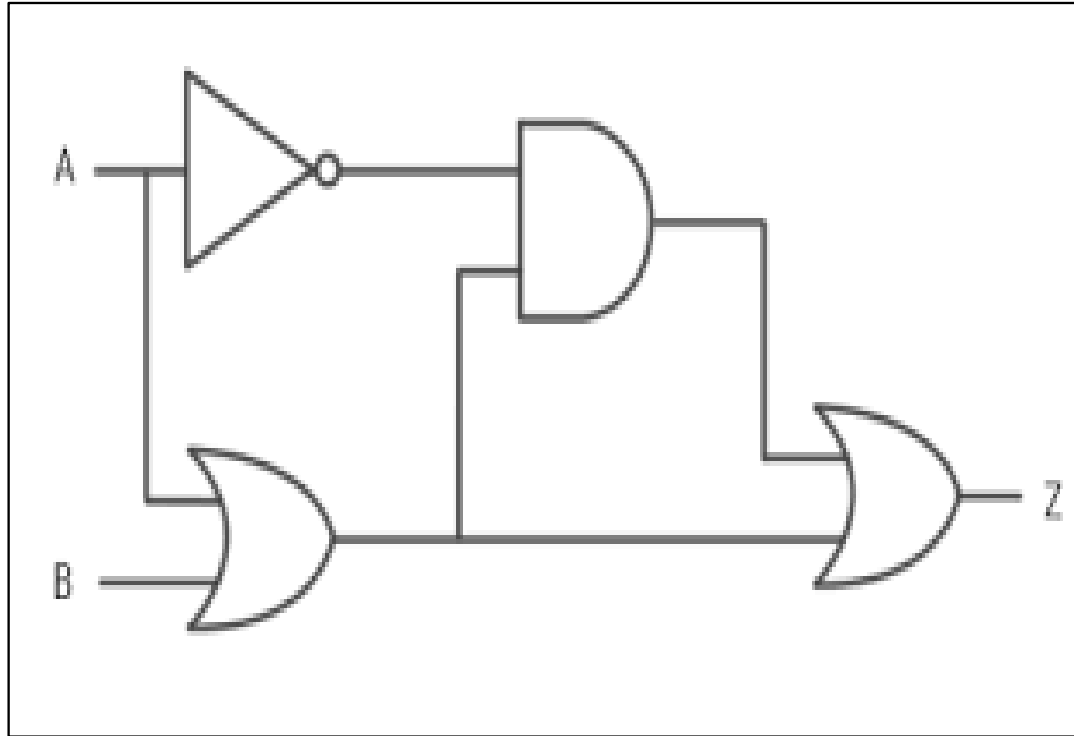
- $((A \cdot B + C') \cdot D)'$



## Unit#1 Digital Logic Circuits:

### Logic Diagram

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



# Logic Diagram

- Draw by your self
- $A.B + A'.C + B'.(A+C')$

# UNIT – 1 Digital Logic Circuits

- Digital Computers
- Logic Gates
- Boolean Algebra
- Map Simplification
- Combinational Circuits
- Flip-Flops
- Sequential circuits



# Boolean Algebra

- Boolean Algebra is used to analyze and simplify the digital (logic) circuits. It uses only the binary numbers i.e. 0 and 1. It is also called as **Binary Algebra** or **logical Algebra**. Boolean algebra was invented by **George Boole** in 1854.
- Variable used can have only two values. Binary 1 for HIGH and Binary 0 for LOW.
- Complement of a variable is represented by an overbar (-). Thus, complement of variable B is represented as  $\overline{B}$ . Thus if  $B = 0$  then  $\overline{B} = 1$  and  $B = 1$  then  $\overline{B} = 0$ .

# Boolean Algebra

- Boolean Laws
- There are six types of Boolean Laws.

### 1) Commutative law

- Any binary operation which satisfies the following expression is referred to as commutative operation.

$$(i) A.B = B.A$$

$$(ii) A + B = B + A$$

- Commutative law states that changing the sequence of the variables does not have any effect on the output of a logic circuit.

# Boolean Algebra

### 2) Associative law

- This law states that the order in which the logic operations are performed is irrelevant as their effect is the same.

$$(i) (A.B).C = A.(B.C)$$

$$(ii) (A + B) + C = A + (B + C)$$

### 3) Distributive law

- Distributive law states the following condition.

$$A.(B + C) = A.B + A.C$$

# Boolean Algebra

### 4) AND law

- These laws use the AND operation. Therefore they are called as AND laws.

$$(i) A \cdot 0 = 0$$

$$(ii) A \cdot 1 = A$$

$$(iii) A \cdot A = A$$

$$(iv) A \cdot \overline{A} = 0$$

### 5) OR law

- These laws use the OR operation. Therefore they are called as OR laws.

$$(i) A + 0 = A$$

$$(ii) A + 1 = 1$$

$$(iii) A + A = A$$

$$(iv) A + \overline{A} = 1$$

# Boolean Algebra

### 6) INVERSION law

- This law uses the NOT operation. The inversion law states that double inversion of a variable results in the original variable itself.

$$\overline{\overline{A}} = A$$

# Boolean Algebra

- DE MORGAN'S LAW
- De Morgan has suggested two theorems which are extremely useful in Boolean Algebra. The two theorems are discussed below.

- Theorem 1

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

- Theorem 2

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

# Boolean Algebra

- Table showing verification of the De Morgan's first theorem

A	B	$\overline{A.B}$	$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
0	0	1	1	1	1
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	0	0	0

- Column 3 and 6 are same so  $\overline{A.B} = \overline{A} + \overline{B}$

# Boolean Algebra

- Table showing verification of the De Morgan's second theorem

A	B	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A} \cdot \overline{B}$
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	0

- Column 3 and 6 are same so  $\overline{A+B} = \overline{A} \cdot \overline{B}$



# Boolean Algebra

- Example of simplify the Boolean expression.

$$\begin{aligned}(A + B)(A + C) &= AA + AC + AB + BC \\&= A + AC + AB + BC \\&= A(1 + C + B) + BC \\&= A \cdot 1 + BC \\&= A + BC\end{aligned}$$

# Boolean Algebra

- Example of simplify the Boolean expression.

$$\begin{aligned} & (A' + B') (A + C') + B' (B + C) \\ &= A'(A + C') + B' (A + C') + B'B + B'C \\ &= A'A + A'C' + B'A + B'C' + B'B + B'C \\ &= 0 + A'C' + B'A + B'C' + 0 + B'C \quad [\because A'A = 0] \\ &= A'C' + B'A + B' (C' + C) \\ &= A'C' + B'A + B' \quad [\because A + A' = 1] \\ &= A'C' + B' (A + 1) \\ &= A'C' + B' \quad [\because A + 1 = 1] \end{aligned}$$



# Boolean Algebra

- Simplify by our self.
- $BC + BC' + BA = B$
- $(A + C)(AD + AD') + AC + C = A + C$
- $A'(A + B) + (B + AA)(A + B') = A + B$
- $A + A'B + A'B'C + A'B'C'D + A'B'C'D'E = A + B + C + D + E$

# Boolean Algebra

- $BC + BC' + BA$
- $B(C + C' + A)$
- $B(1 + A) \quad (C + C' = 1)$
- $B(1) \quad (1 + A = 1)$
- $B$