Module 1-
Computer Fundamentals

-Neha Surti

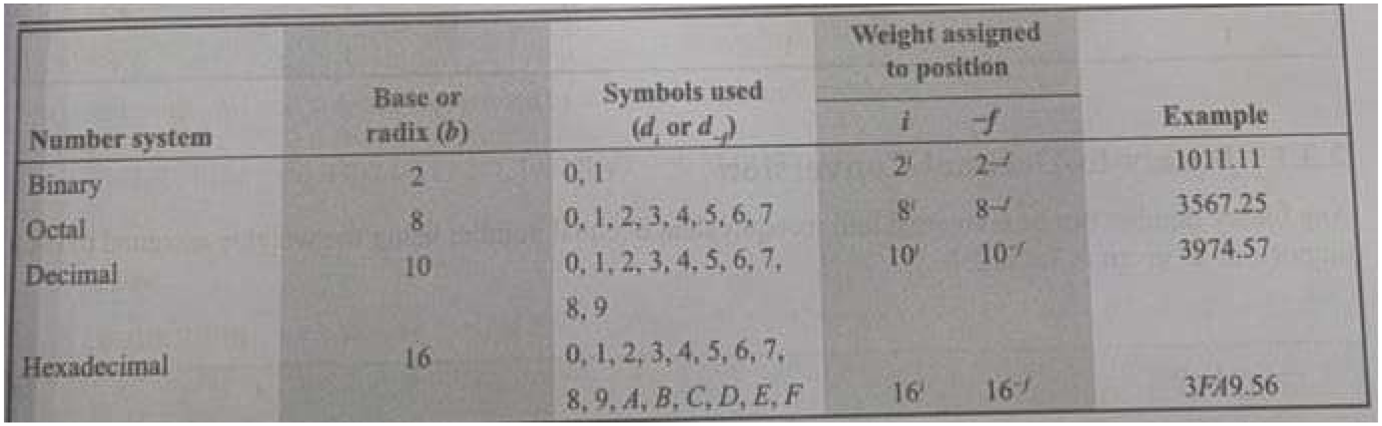
- Introduction to Number System and Codes
- Number Systems: Binary, Octal, Decimal, Hexadecimal
- Codes: Grey, BCD, Excess-3, ASCII, Boolean Algebra
- Logic Gates: AND, OR, NOT, NAND, NOR, EX-OR
- Overview of computer organization and architecture
- Basic Organization of Computer and Block Level functional Units, Von Neumann Model

Number System

A number N in base or radix b can be written as:

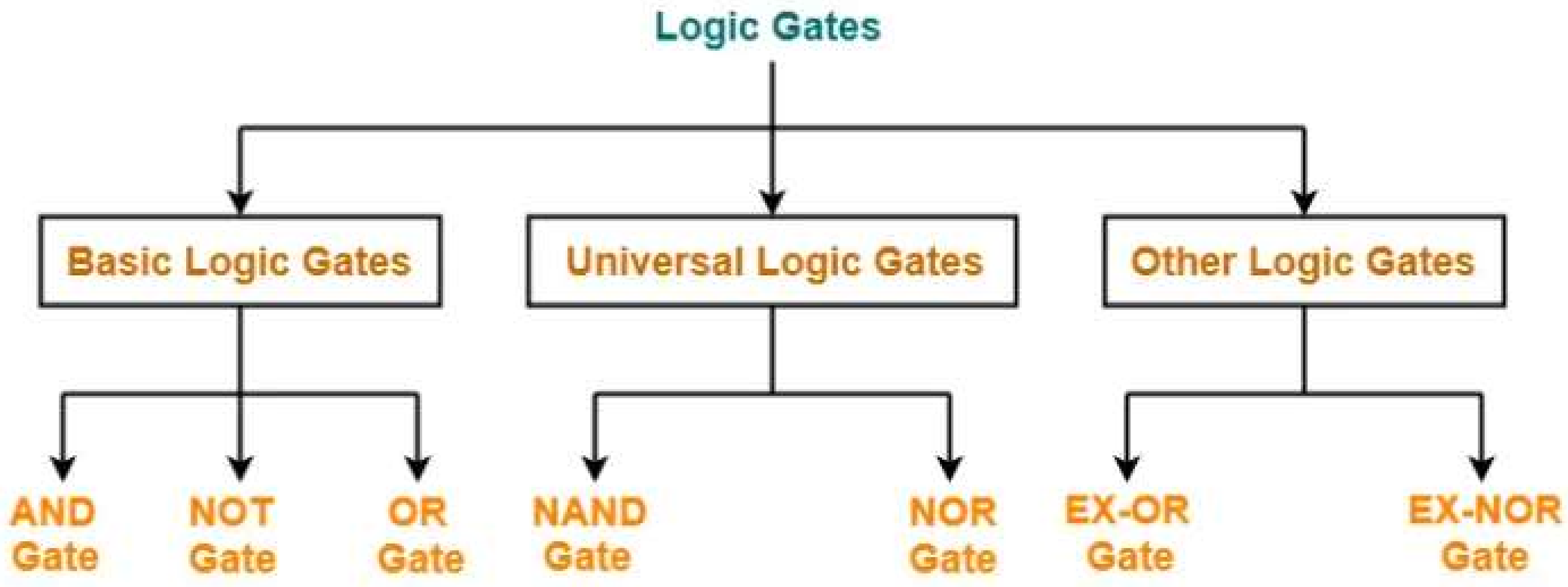
$(N)_b = d_{n-1}\ d_{n-2}$ -- -- -- -- $d_1\ d_0\ .\ d_{-1}\ d_{-2}$ -- -- -- -- d_{-m}

dn-1 = Most significant bit (MSB)
d-m = Least significant bit (LSB)



Number system	Base or radix (b)	Symbols used (d_i or d_{-f})	Weight assigned to position		Example
			i	$-f$	
Binary	2	0, 1	2^i	2^{-f}	1011.11
Octal	8	0, 1, 2, 3, 4, 5, 6, 7	8^i	8^{-f}	3567.25
Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	10^i	10^{-f}	3974.57
Hexadecimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, *A*, *B*, *C*, *D*, *E*, *F*	16^i	16^{-f}	3*F*49.56

Logic Gates



Logic Gates

Basic Logic Gates

Universal Logic Gates

Other Logic Gates

AND Gate

NOT Gate

OR Gate

NAND Gate

NOR Gate

EX-OR Gate

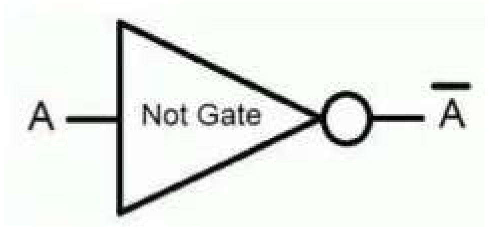
EX-NOR Gate

Types of Logic Gates

NOT Gate

- The NOT gate produces high output when the input is low and vice versa.
- Logical Expression: X=A′

Logical Symbol.

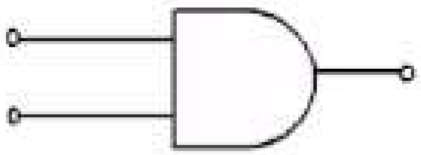


Input	Output
0	1
1	0

AND Gate

- The AND gate produces high output only when all the inputs are high.
- When any of the inputs is low the output is low.
- Logical Expression: X=A.B

Logical Symbol



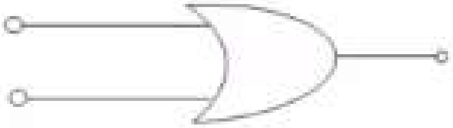
Truth Table

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

OR Gate

- The OR gate produces high output only when any of the inputs is high.
- When all the inputs are low, the output is low.
- Logical Expression: X=A+B

Logical Symbol



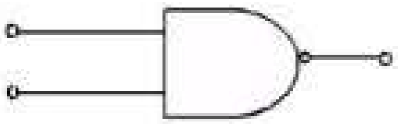
Truth Table

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

NAND Gate

- NAND gate can be used in combination to perform the AND, OR and inverter operations.
- It is constructed by attaching NOT gate at the output of AND gate, hence it is called NOT-AND
- A NAND gate produces a low output only when all the inputs are high, when any of the inputs is the output will be high.
- Logical Expression: X=(A.B)′

Logical Symbol



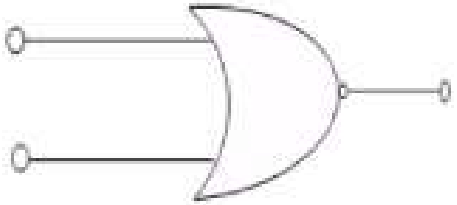
Truth Table

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

NOR Gate

- Like NAND gate, the NOR gate can also be used as a universal gate.
- It can be used in combination to perform the AND, OR and inverter operations.
- It is constructed by attaching NOT gate at the output of OR gate, hence it is called NOT-OR Gate
- A NOR gate produces a low output when any of its inputs is high. The output is high only when a input are low.
- Logical Expression: X=(A+B)′

Logical Symbol



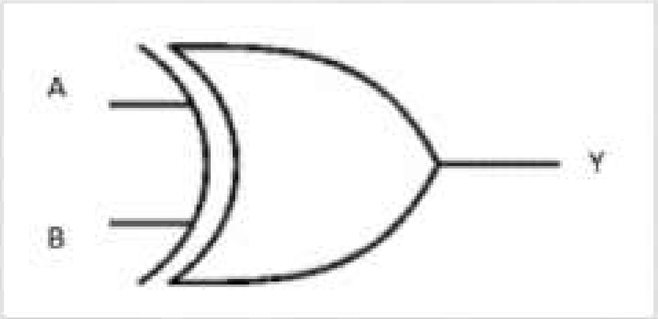
Truth Table

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

The Exclusive-OR Gate

- If both inputs are low or both inputs are high, then it produces the low output otherwise it produc output.
- Logical Expression: X=AB′+A′B = A⊕B

- Logical Symbol

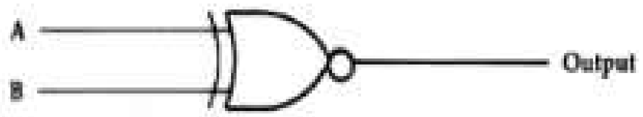


Inputs		Output
A	B	AB'+A'B=X
0	0	0.0'+0'.0 =0+0=0
0	1	0.1'+0'+1=0+1=1
1	0	1.0'+1'.0=1+0=1
1	1	1.1'+1'.1=0+0=0

The Exclusive-NOR Gate

- The Exclusive-NOR Gate is the compliment of the Exclusive-OR gate.
- If both the inputs are low or both are high, then it produces high output otherwise it produces the output
- Logical Expression: X=AB+A′B′ = A⊙B

- Logical symbol



Inputs		Output
A	B	AB'+A'B=X
0	0	0.0+0'.0'=0+1=0
0	1	0.1+0'+1'=0+0=0
1	0	1.0+1'.0'=0+0=0
1	1	1.1+1'.1'=1+0=1

Boolean Algebra

- Developed by English Mathematician George Boole in the 19th Century.
- It includes rules for manipulation of binary variables.
- It is the basis of all digital systems like computers, calculators, etc.
- It contains basic operators like AND, OR, and NOT, etc.
- Binary variables are represented using capital letters e.g. ‘A’, ‘B’, etc.

Boolean Algebra Theorems

Properties of 0 and 1

1. $A + 0 = A$
2. $A \,.\, 1 = A$
3. $A + 1 = 1$
4. $A \,.\, 0 = 0$

Indempotence (Identity) Law- a variable remains unchanged when it is ORed or ANDed with itself

1. $A + A = A$
2. $A \,.\, A = A$

Complementary Law- if a complement is added to a variable, it gives 1 and if multiplied, it gives 0

1. $A + A' = 1$
2. $A \,.\, A' = 0$

Distributive Law- opening of brack

1. $A \,.\, (B + C) = AB + AC$
2. $A + BC = (A + B)\,(A + C)$

Absorption (Redundance) Law

1. $A + AB = A$
2. $A\,(A + B) = A$

Associative law- the order of opera
not matter if the priority of variable
same

1. $A + (B + C) = (A + B) + C$
2. $A \,.\, (B \,.\, C) = (A \,.\, B) \,.\, C$

De Morgan’s Theorems

It states that the operation of an AND or OR logic circuit is unchanged if all inputs are inverted, the is changed from AND to OR, and the output is inverted.

1. $(A . B)' = A' + B'$
2. $(A + B)' = A' . B'$

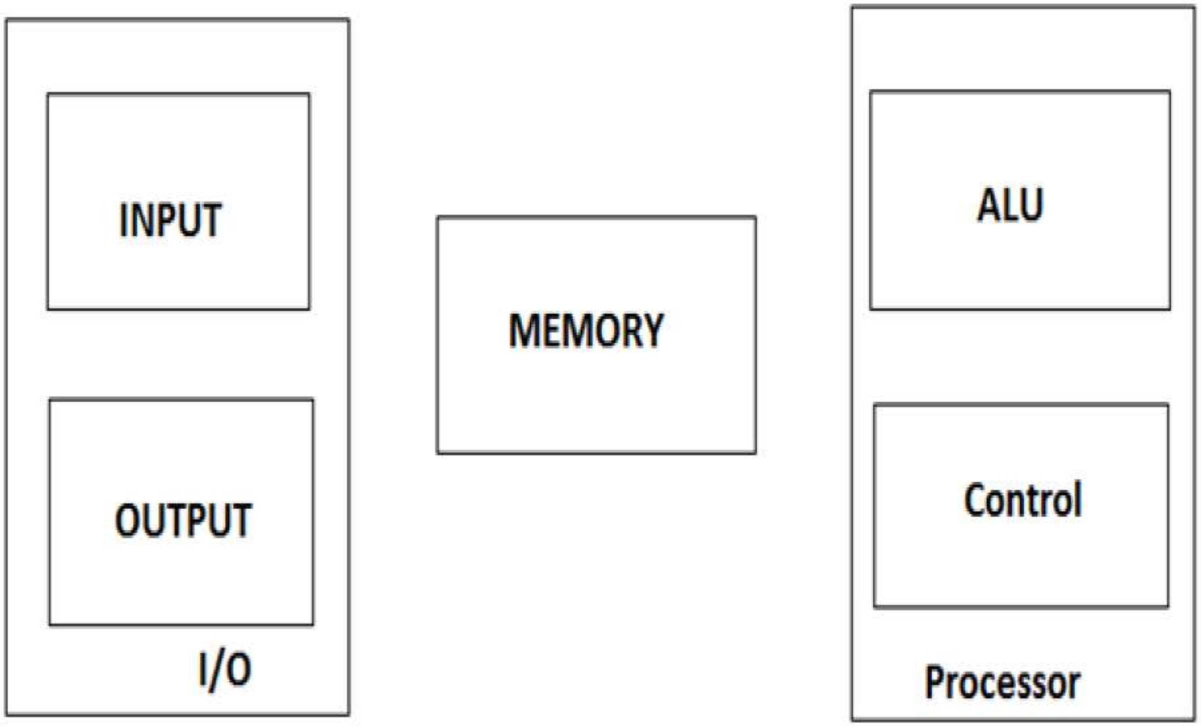
Computer Architecture

- Computer Architecture refers to those attributes of a system visible to a programmer O
  attributes of a system that have a direct impact on the logical execution of a program.
- Examples:
  - the instruction set
  - the number of bits used to represent various data types
  - I/O mechanisms
  - memory addressing techniques

Computer Organization

- Computer Organization refers to the operational units and their interconnections that rea the architectural specifications.
- Examples of organizational attributes includes those hardware details that are transparen the programmer:
  - control signals
  - interfaces between computer and peripherals
  - the memory technology being used

Basic Organization Of Computer

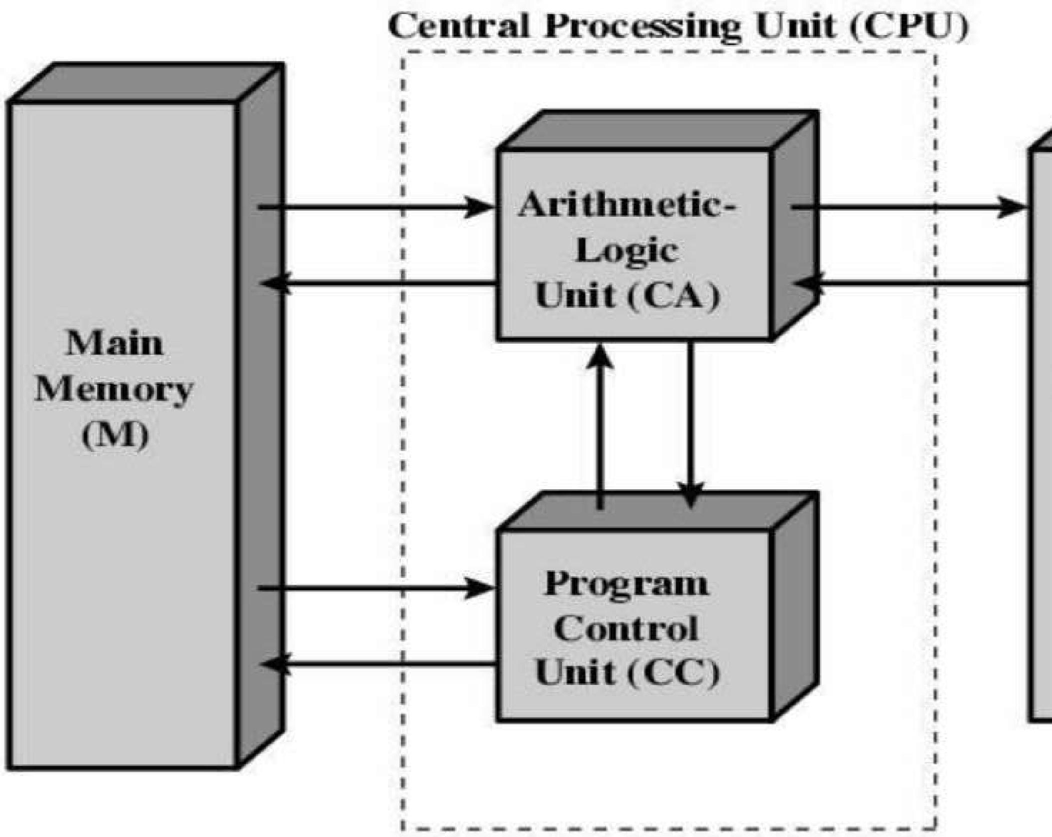


Von Neumann Model

Stored Program Concept

It consists of-

- A main memory, which stores both data and instructions
- An ALU capable of operating on binary data
- A control unit, which interprets the instructions in memory and causes them to be executed
- I/O equipment operated by the control unit


Central Processing Unit (CPU)
Main Memory (M)
Arithmetic-Logic Unit (CA)
Program Control Unit (CC)

Thank You