Digital Electronics and Computer Organization

UNIT - 1

Number System :-

The number system or the number system is the system of naming or representing numbers. A number system is defined as a system of writing to express numbers.

In a number system, these numbers are used as digits. 0 and 1 are the most common ~~number system~~ digits in the number system, that are used to represent binary numbers.

Types of Number System :-

There are various types of number systems. The four most common number system types are :-

1. Decimal number system
2. Binary number system
3. Octal number system
4. Hexadecimal number system

Binary Number System :-

The Binary number system has a base 2 because it uses two digits 0 and 1.

For example :-

$(11010\,1)_2$ is a binary number.

we can. Convert any system into binary and vice versa.

Decimal number system:-

The decimal number system has a base 10 because it uses ten digits from 0 to 9.

For example:-

$(1457)_{10}$ is a decimal number.

Octal number system:-

In Octal number system the base is 8 and it uses numbers from 0 to 7 to represent numbers. Octal numbers are commonly used in computer applications.

For example:-

$(141)_8$ is an octal number.

Hexadecimal number system:-

In hexadecimal system, numbers are written or represented with base 16. In the hex. system, the numbers are first represented just like in the decimal system. i.e. from 0 to 9. The, the numbers are represented using using the alphabet from A to F.

10 → A , 11 → B , 12 → C , 13 → D

14 → E , 15 → F .

For example.

$(14F6)_{16}$ is a Hexadecimal number.

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.

Boaleal algebra is was invented by george Boole in 1854.

The important operations performed in Boalean algebra are - conjunction(∧), disjunction (∨) and negation (¬).

→ Boolean Expression :-

Boolean expressions are the statements that use logical operators i.e. AND, OR, XOR and NOT. Thus, if we write X AND Y = True then it is a Boolean expression.

→ Truth table :-

Truth table is a table that gives all the possible values of logical variables and the combination.

The no. of row in the truth table should be equal to 2^n, where 'n' is the no. of variables in.

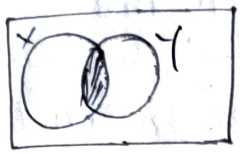
A	B	A ∧ B	A ∨ B
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

A	¬A
0	1
1	0

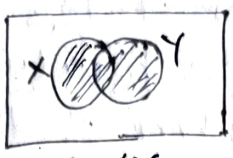
Boolean algebra operations

The basic operations of Boolean algebra are as follows.

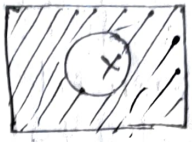
- Conjunction or AND operation
- Disjunction or OR operation
- Negation or NOT operation.



$X \wedge Y$



$X \vee Y$



$\neg X$

Laws of Boolean algebra:-

There are six type of Laws.

- Commutative law
  - $A \cdot B = B \cdot A$
  - $A + B = B + A$
- Associative law
  - $(A \cdot B) \cdot C = A \cdot (B \cdot C)$
  - $(A + B) + C = A + (B + C)$
- Distributive law
  - $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$
  - $A + (B \cdot C) = (A + B) \cdot (A + C)$
- AND Law
  - $A \cdot 0 = 0$
  - $A \cdot 1 = A$
  - $A \cdot A = A$
  - $A \cdot \bar{A} = 0$
- OR Law
  - $A + 0 = A$
  - $A + 1 = 1$
  - $A + A = A$
  - $A + \bar{A} = 1$
- Inversion Law:
  - $\bar{\bar{A}} = A$

Boolean algebra theorems:-

The two important theorems which are extremely used in Boolean algebra are De Morgan's First law and De Morgan's law.

This theorem basically helps to reduce the given Boolean expression in the simplified form.

De Morgan's first law:-

De Morgan's first law states that $\overline{(A \cdot B)} = \overline{A} + \overline{B}$.

The first law states that the complement of the product of the variable is equal to the sum of their individual complements of a variable.

Truth table :-

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

The last two columns show that

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

Hence, De Morgan's first law is Proved.

De Morgan's second law :-

De Morgan's second law states that

$$\overline{(A+B)} = \overline{A} \cdot \overline{B}$$

The second law states that the complement of the sum of variables is equal to the product of their individual complements of a variable.

Truth table :-

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

the last two columns shows that
$(\overline{A+B}) = \bar{A}\cdot\bar{B}$
Hence De-Morgan's second law is Proved.

Conversation of Number System :-

Number system conversation deal with the operations to change the base of the numbers. For example. To change a decimal number with base 10 to binary number with base 2.

1. Binary to Decimal conversion :-

$(1\ 1\ 1\ 1)_2$

$1\times2^0 + 1\times2^1 + 1\times2^2 + 1\times2^3$

$= 1 + 2 + 4 + 8 = (15)_{10}$

$(1\ 0\ 1)_2$

$1\times2^0 + 0\times2^1 + 1\times2^2$

$= 1 + 0 + 4 = (5)_{10}$

$(1\ 1\ 0\ 1)_2$

$1\times2^0 + 0\times2^1 + 1\times2^2 + 1\times2^3$

$= 1 + 0 + 4 + 8 = (13)_{10}$

$(0.1101)_2$

$1\times2^{-1} + 1\times2^{-2} + 0\times2^{-3} + 1\times2^{-4}$

$\frac{1}{2} + \frac{1}{4} + 0 + \frac{1}{16}$

$\frac{8+4+1}{16} = \frac{13}{16}$

$= (0.8125)_{10}$

$(101101.10101)_2$

$1\times2^0 + 0\times2^1 + 1\times2^2 + 1\times2^3 + 0\times2^4 + 1\times2^5$

$= 1 + 4 + 8 + 32 = 45$

$1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3} + 0\times2^{-4} + 1\times2^{-5}$

$\frac{1}{2} + \frac{1}{8} + \frac{1}{32}$

$= \frac{21}{32} = 0.65625$

$(101101.10101)_2 = (45.65625)_{10}$

Decimal to Binary conversation:-

$(32)_{10}$

2	32	0
2	16	0
2	8	0
2	4	0
2	2	0
2	1	1
	0	

$(100000)_2$

$(13)_{10}$

2	13	1
2	6	0
2	3	1
2	1	1
	0	

$(1101)_2$

$(0.85)_{10}$

$0.85 \times 2 = 1.7$ — 1
$0.7 \times 2 = 1.4$ — 1
$0.4 \times 2 = 0.8$ — 0
$0.8 \times 2 = 1.6$ — 1
$0.6 \times 2 = 1.2$ — 1
$0.2 \times 2 = 0.4$ — 0

$= (.110110)_2$

$(23.6)_{10}$

2	23	1
2	11	1
2	5	1
2	2	0
2	1	1
	0	

10111

$0.6 \times 2 = 1.2$ — 1
$0.2 \times 2 = 0.4$ — 0
$0.4 \times 2 = 0.8$ — 0
$0.8 \times 2 = 1.6$ — 1
$0.6 \times 2 = 1.2$ — 1
$0.2 \times 2 = 0.4$ — 0

$.100110$

$(23.6)_{10} = (10111.100110)_2$

Octal to Decimal conversion:-

In the octal number system each digit position corresponds to a power of 8 as follows.

$8^3\ 8^2\ 8^1\ 8^0 \cdot 8^{-1}\ 8^{-2}\ 8^{-3}$

↑
Octal Point

To convert from octal to decimal, multiply each octal digit by its weight and add the resulting products.

$(23)_8$

$3\times8^0 + 2\times8^1 = 3+16 = (19)_{10}$

$(257)_8$

$7\times8^0 + 5\times8^1 + 2\times8^2 = 7+40+128 = (175)_{10}$

$(0.54)_8$

$5\times8^{-1} + 4\times8^{-2}$

$\frac{5}{8} + \frac{4}{64} = \frac{44}{64} = (0.6875)_{10}$

Decimal-to-octal Conversion:-

$(175)_{10}$

8	175	7
8	21	5
8	2	2
	0	

$(257)_8$

$(0.23)_{10}$

$0.23\times8 = 1.84 \quad 1$

$0.84\times8 = 6.72 \quad 6$

$0.72\times8 = 5.76 \quad 5$

$0.76\times8 = 6.08 \quad 6$

$0.08\times8 = 0.64 \quad 0$

$(0.16560)_8$

Octal to Binary Conversion:-

You can convert from octal to binary as follows.

Change each octal digit to its binary equivalent.

$(23)_8$

2 3

010 011

$\Rightarrow (10011)_2$

$(0.562)_8$

5 6 2

101 110 010

$= (0.101110010)_2$

Binary to octal Conversion :-

Simply remember to group the bits in threes, starting at the binary Point; then convert each group of three to its octal equivalent.

$(1011)_2$

001 011

1 3

$(13)_8$

$(0.01101)_2$ 011, 010

3 2

$(0.32)_8$

Hexadecimal to Binary Conversion

To convert hexadecimal number to a Binary number, convert each Hexa-decimal digit to its 4-bit equivalent Binary no.

$(9AF)_{16}$

9 A F

1001 1010 1111

$(9AF)_{16} = (100110101111)_2$

Binary - to - Hexadecimal conversion :-

to convert from binary to hexade... group the bits in fours. Starting at the binary point. then convert each group of 4 digit to its hexadecima... equivalent.

$(1110100011010110)_2$

1110 1000 1101 0110

E 8 D 6

$= (E8D6)_{16}$

Hexadecimal to decimal conversion.

In the hexadecimal number system each digit position corresponds to a power of 16. The weight of the digit positions in a hexadeciman number are as follows

$16^3\ 16^2\ 16^1\ 16^0 \bullet 16^{-1}\ 16^{-2}\ 16^{-3}$

↑

Hexadecimal Point.

to convert hexadecimal to decima... multiply each hexadecimal digit by its weight and add the resulting Pdt.

$(F8E6)_{16} = 6\times16^0 + 14\times16^1 + 8\times16^2 + 15\times16^3$

$= 6 + 224 + 2048 + 61440$

$= (63718)_{10}$

$(.39)_{16} = 3\times16^{-1} + 9\times16^{-2}$

$= \frac{3}{16} + \frac{9}{256}$

$= \frac{57}{256}$

$= (0.2227)_{10}$

Decimal to hexadecimal conversion:-

$(2479)_{10}$

16 | 2479	15 → F	↑
16 | 154	10 → A	
16 | 9	9	
0		

$(9AF)_{16}$

$(0.45)_{10}$

$0.45\times16 = 7.2$ → 7 ↓

$0.2\times16 = 3.2$ → 3

$0.2\times16 = 3.2$ → 3

$(0.733)_{16}$

Binary Arithmetic

Binary addition:-

1011 & 1100

```
  1011
+ 1100
------
 10111
------
```

01101010 & 00001010

```
  01101010
+ 00001010
----------
  01110100
----------
```

Binary subtraction:-

Foure basic cases of binary subtraction:-

0 - 0 = 0
1 - 0 = 1
1 - 1 = 0
0 - 1 = 1 Borrow 1 (10-1

1010 from 1101

```
  1101
- 1010
------
  0011
------
```

1011 from 1100

```
  1100
- 1011
------
  0001
------
```

Binary multiplication:-

The binary multiplication is very much similar to the usual multiplication method of integers. First we need to multiply each digit of one binary number to each digit of another binary number. And then add them all together to get the final result.

Multiply 110 and 100

```
  110
x 100
-----
  000
 000
110
-----
11000
-----
```

Ans:- 11000

binary Pdt of 1001 and 1011

```
  1001
x 1011
------
  1001
 1001
0000
1001
-------
1100011
-------
```

Ans:- ~~1011011~~ 1100011

Binary Division:-

The binary division operation is similar to the base 10 decimal system, except the base 2.

There are different ways to solve division Problem using binary operations. Long division is one of them

and the easiest and the most efficient way.

Binary Division Rules:-

$1 \div 1 = 1$

$1 \div 0 =$ meaningless

$0 \div 1 = 0$

$0 \div 0 =$ Meaningless

Ex:-

Solve

$01111100 \div 0010$

```
10 ) 1111100 ( 11110
   - 10
   ----
     011
     -10
     ---
      11
      -10
      ---
       11
       -10
       ---
        10
        -10
        ---
        000
        -00
        ---
          0
```

Ans:- 11110

Solve

$101101 \div 101$

```
101 ) 101101 ( 1001
    - 101
    -----
      0101
      -101
      ----
         0
```

Ans:- 1001

Sign - magnitude numbers :-

The negative decimal no. are -1, -2, -3 and so on. One way to code these as binary numbers is to convert the magnitude to its binary equivalent and Prefix the sign. With this approach, the sequence -1, -2 and -3 become -001, -010 and -011. Since everything has to be coded as strings of 0s and 1s, the + and - signs also have to be represented in binary form. For reasons given, 0 is used for the + sign and 1 for the - sign. Therefore, -001, -010, and -011 are coded as 1001, 1010 and 1011. Numbers in this form are called sign - magnitude numbers.

The MSB always represents the sign, and the remaining bits always stand for the magnitude.

Complements :-

There are two types of Complement of Binary number:

i) 1's complement
ii) 2's complement.

1's Complement :-

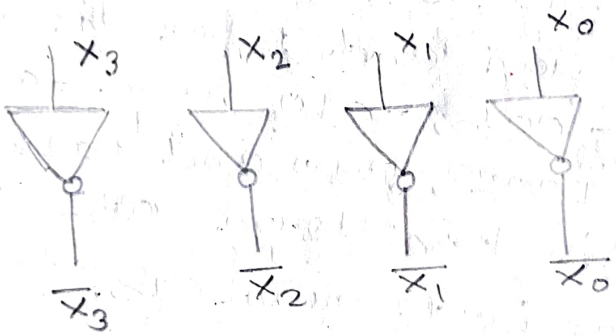
The 1's complement of a binary number is the number that results when we complement each bit.

The 4-bit input, output 1's complement of a 4-bit input. -

$x_3 x_2 x_1 x_0 = 1000$

the 1's complement is

$\overline{x}_3 \overline{x}_2 \overline{x}_1 \overline{x}_0 = 0111$



Inverters Produce the 1's complement

The same principle applies to binary numbers of any length.

Examples -

1010 → 0101

1110 1100 → 0001 0011

0011 1111 0000 0110 → 1100 0000 1111 1001

2's Complement :-

The 2's complement is the binary number that results when we add 1 to 1's complement.

As a formula:-

2's complement = 1's complement + 1

2's complement of 1011 :

1011 ⟶ 0100 (1's complement)

0100 + 1 = 0101 (2's complement)

LOGIC GATES

Logic gates are the basic building blocks of any digital system.

Three logic circuits, the inverter (NOT gate), the OR gate and the AND gate, can be used to produced any digital system

AND Gates :-

The AND gate has a high output only when all inputes are high.

The Boolean equation of AND gate is

$$Y = A \cdot B$$

Truth table of AND gate:-

A	B	Y = A.B
0	0	0
0	1	0
1	0	0
1	1	1

OR gates:-

An OR gate has two or more input signals but only one output signal. It is called an OR gate because the output voltage is high if any or all of the input voltages are high.

The Boolean expression of the OR gate is

$$Y = A + B$$

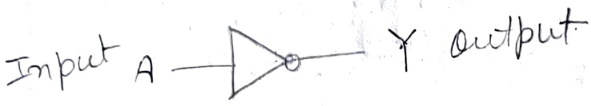
The '+' sign here represents logic operation OR and not addition operation of basic arithmetic.

Truth table of Two Input OR Gates

A	B	Y = A + B
0	0	0
0	1	1
1	0	1
1	1	1

NOT Gate (The Inverter) :-

The output of NOT gates attains state 1 if and only if the input does not attains state 1.



Symbol of NOT Gate

The boolean expression is.

$$Y = \overline{A}$$

Truth table :-

A	$Y = \overline{A}$
0	1
1	0

Universal Logic gates :-

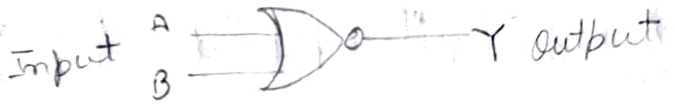
A universal gate is a logic gate which can implement any Boolean function without the need to use any other type of logic gate.

The NOR Gate and NAND gate are universal gates.

NOR Gates :-

A NOR gate, sometimes known as a "NOT-OR" gate, consists of an OR gate followed by a NOT gate.

This gate output is high only if all inputs are low.



Symbol of NOR Gates

The Boolean expression of NOR gate is

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

Truth table :-

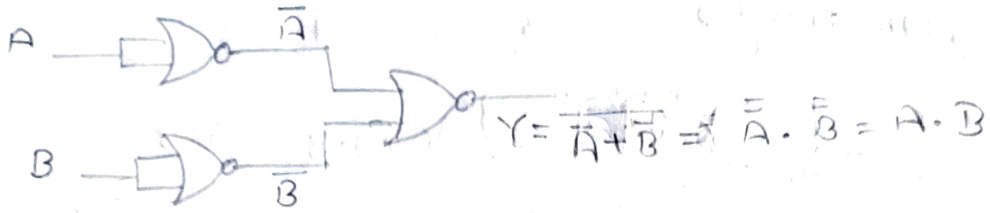
A	B	$Y = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Universality of NOR Gates :-

NOT from NOR

OR from NOR :-

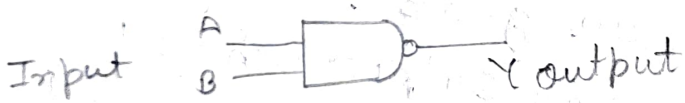
AND from NOR! -



NAND Gates! -

A NAND gate is also know as 'NOT - AND' gate, NAND gate is a AND gate followed by NOT gate.

The output of NAD gate is Low only when all inputs are high.



Symbol of NAND Gates

The boolean expression for NAND gate is.

$$Y = \overline{A \cdot B}$$

Truth table :-

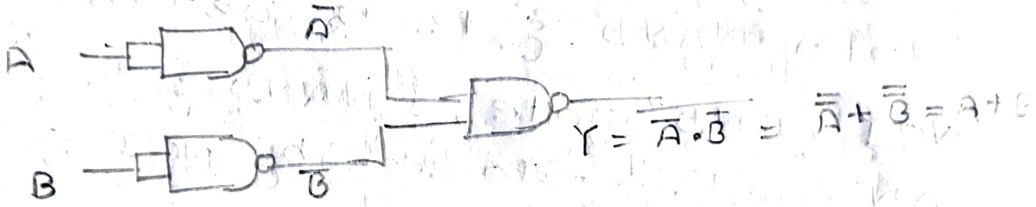
A	B	$Y = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Universality of NAND gate

NOT from NAND

AND from NAND

OR from NAND



XOR Gate :-

The XOR gate stands for the Exclusive-OR gate. it gaves a true (1 or high) output when the number of true inputs is odd.

2 Input XOR gate

The boolean expression of XOR gate is as follows -

$$Y = A \oplus B$$

$$Y = A'B + AB'$$

Truth table :-

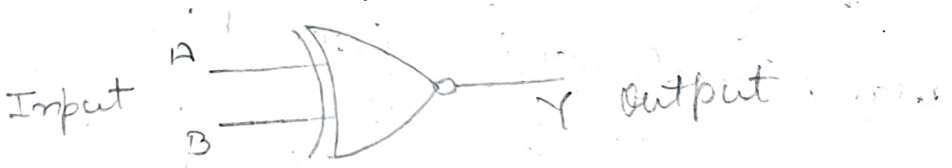
A	B	$Y = \bar{A}B + A\bar{B}$
0	0	0
0	1	1
1	0	1
1	1	0

XOR gate used to construct digital circuit that performs arithmetic operations and calculations. Especially Adders and Half - Adders.

XNOR Gate :-

The XNOR gate is the complement of the XOR gate. It is the combination of the XOR gate and NOT gate.

The output of the XNOR gate is high only when both of its input are the same.) when high input are even or zero.



The boolean expression for XNOR gate is

$$Y = \overline{A \oplus B}$$

$$Y = \bar{A} \cdot \bar{B} + AB$$

Truth table :-

A	B	$Y = \bar{A}.\bar{B} + A.B$
0	0	1
0	1	0
1	0	0
1	1	1

Logic circuit

A logic circuit is a circuit that executes a processing or controlling function in a computer.

This circuit implements logical operations on information to process it.

There are two types of logic circuit :-

1. Combinational logic circuit
2. Sequential logic circuit.

Combinational logic circuit :-

The combinational circuit is digital logic circuit that contain different types of logic gate.

The different types of logic gate like AND, OR, NOT etc. gate are combined is known as a combinational logic circuit.