Number System :-

In general, any number system there is an ordered set of symbols known as digits with rules defined for performing arithmatic operations like addition, multiplication.

- The number system in which an ordered set of ten symbols - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 is known as digits are used to specify any number.

- Number system : Base or Radix of Number system.

- Number system is classified as :
  1) Binary
  2) Octal
  3) Decimal
  4) Hexadecimal

Number system	Base	Digits Used
1) Binary	2	0, 1
2) Octal	8	0, 1, 2, 3, 4, 5, 6, 7
3) Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
4) Hexadecimal	16	0 - 9, A, B, C, D, E, F [Where A = 10, B = 11, C = 12, D = 13, E = 14, F = 15.

1) Binary Number system :-

- In the binary number system, the total number of symbols are two (0, 1) ; base is 2, & the radix point is known as the binary point.

- Number of digits - 0 & 1 i.e 2
- Radix / Base - 2

e.g. $(101.01)_2$, $(1110.101)_2$

- The general representation,

$$B_2 = \sum_{i=-m}^{n} b_i \cdot 2^i$$

e.g. $(101.011)_2 = (1\times2^2) + (0\times2^1) + (1\times2^0) + (0\times2^{-1}) + (1\times2^{-2}) + (1\times2^{-3})$

2) Octal Number System :-

- The number system with base 8 is known as the octal number system.

- Number of digits - 0, 1, 2, 3, 4, 5, 6, 7 i.e 8
- Radix / Base - 8

eg : $(734)_8$, $(614.25)_8$

- The general representation,

$$O_8 = \sum_{i=-m}^{n} O_i \cdot 8^i$$

e.g: $(614.25)_8 = (6\times8^2) + (1\times8^1) + (4\times8^0) + (2\times8^{-1}) + (5\times8^{-2})$

3) Decimal Number System :-

- In the decimal number system, the total number of symbols are ten (0, 1, 2, 3, 4, 5, 6, 7, 8, 9); base is 10 & the radix point is known as the decimal point.

- The leftmost digit has the max^m value, known as "most significant digit" (MSD).

- The rightmost digit has the min^m value, known as "Least significant digit" (LSD).

- The general representation,

$$D_{10} = \sum_{i=-m}^{n} d_i \cdot 10^i$$

- e.g. $(555.55)_{10} = (5\times10^2)+(5\times10^1)+(5\times10^0)+(5\times10^{-1})+(5\times10^{-2})$

4) Hexadecimal Number System :-

- The number system with base 16 is known as the "Hexadecimal number system".
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

- The general representation,

$$H_{16} = \sum_{i=-m}^{n} h_i \cdot 16^i$$

- e.g. $(A2D.4F)_{16} = (A\times16^2) + (2\times16^1) + (D\times16^0) + (4\times16^{-1}) + (F\times16^{-2})$

* Number Conversion :-

1) Binary to Decimal :-

i) $(11010)_2 = 1\times2^4 + 1\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0$
$= 16 + 8 + 0 + 2 + 0$
$= (26)_{10}$

ii) $(1101.1101)_2 = 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0 +$
$1\times2^{-1} + 1\times2^{-2} + 0\times2^{-3} + 1\times2^{-4}$
$= 8 + 4 + 0 + 1 + 0.5 + 0.25 + 0 + 0.0625$
$= (13.8125)_{10}$

2) Binary to Octal :-

i) $(11010100)_2 = 011\ 010\ 100$
$(3\ \ 2\ \ 4)_8$

$(11010100)_2 = (324)_8$

ii) $(10001101.0111010)_2 = 010\ 001\ 101\ .\ 011\ 101\ 00\underline{0}$
$2\ \ 1\ \ 5\ \ 3\ \ 5\ \ 0$

$(10001101.0111010)_2 = (215.350)_8$

3) Binary to Hexadecimal :-

i) $(10001101.0111010)_2 = (80.74)_{16}$

ii) $(11010101.10110)_2 = (D5.B0)_{16}$

4) Octal to Binary :-

i) $(73.62)_8 = \overset{7}{111}\ \overset{3}{011}\ .\ \overset{6}{110}\ \overset{2}{010}$

$= (111011.110010)_2$

ii) $(67.357)_8 = 110\ 111\ .\ 011\ 101\ 111$

$= (110111.011101111)_2$

5) Octal to Decimal :-

i) $(164.35)_8 = 1\times8^2 + 6\times8^1 + 4\times8^0 + 3\times8^{-1} + 5\times8^{-2}$

$= 64 + 48 + 4 + 0.375 + 0.0625$

$= (116.4375)_{10}$

ii) $(314)_8 = 3\times8^2 + 1\times8^1 + 4\times8^0$

$= 3\times64 + 8 + 4$

$= 192 + 12$

$= (204)_{10}$

6) Octal to Hexadecimal :-

i) $(647.32)_8 = \overset{6}{110}\ \overset{4}{100}\ \overset{7}{111}\ .\ \overset{3}{011}\ \overset{2}{010}$

$= 110100111.011010$

$= 0001\ 1010\ 0111.\ 0110\ 1000$

$(647.32)_8 = (1A7.68)_{16}$

ii) $(13.125)_8 = 001\ 011\ .\ 001\ 010\ 101$

$= 001011.001010101$

$= 0000\ 1011.0010\ 1010\ 1000$

$(13.125)_8 = (0B.2A8)_{16}$

4) Octal to Binary :-

i) $(73.62)_8 = \overset{7}{111}\ \overset{3}{011} \cdot \overset{6}{110}\ \overset{2}{010}$
$= (111011.110010)_2$

ii) $(67.357)_8 = 110\ 111 \cdot 011\ 101\ 111$
$= (110111.011101111)_2$

5) Octal to Decimal :-

i) $(164.35)_8 = 1\times8^2 + 6\times8^1 + 4\times8^0 + 3\times8^{-1} + 5\times8^{-2}$
$= 64 + 48 + 4 + 0.375 + 0.0625$
$= (116.4375)_{10}$

ii) $(314)_8 = 3\times8^2 + 1\times8^1 + 4\times8^0$
$= 3\times64 + 8 + 4$
$= 192 + 12$
$= (204)_{10}$

6) Octal to Hexadecimal :-

i) $(647.32)_8 = \overset{6}{110}\ \overset{4}{100}\ \overset{7}{111} \cdot \overset{3}{011}\ \overset{2}{010}$
$= 110100111.011010$
$= 0001\ 1010\ 0111 \cdot 0110\ 1000$

$(647.32)_8 = (1A7.68)_{16}$

ii) $(13.125)_8 = 001\ 011 \cdot 001\ 010\ 101$
$= 001011 \cdot 001010101$
$= 0000\ 1011 \cdot 0010\ 1010\ 1000$

$(13.125)_8 = (0B.2A8)_{16}$

) Decimal to Binary :-

i) $(25)_{10}$

2	25		LSB
2	12	1	
2	6	0	
2	3	0	
2	1	1	
	0	1	MSB

$(25)_{10} = (11001)_2$

ii) $(42.125)_{10}$

convert Integer

2	42		LSB
2	21	0	
2	10	1	
2	5	0	
2	2	1	
2	1	0	
	0	1	MSB

$(42)_{10} = (101010)_2$

convert fractional

$= 0.125$

$= 0.125 \times 2 = 0.25 \rightarrow 0$

$= 0.25 \times 2 = 0.5 \rightarrow 0$

$= 0.5 \times 2 = 1 \rightarrow 1$

$(0.125)_{10} = (001)_2$

$(42.125)_{10} = (101010.001)_2$

8) Decimal to octal :-

i) $(135)_{10}$

8	135	
8	16	7
8	2	0
	0	2

LSB ↑ MSB

$(135)_{10} = (207)_8$

ii) $(245.455)_{10}$

8	245	
8	30	5
8	3	6
	0	3

LSB ↑ MS

$0.455 \times 8 = 3.64 \rightarrow 3$
$0.64 \times 8 = 5.12 \rightarrow 5$
$0.12 \times 8 = 0.96 \rightarrow 0$
$0.96 \times 8 = 7.68 \rightarrow 7$

$(245.455)_{10} = (365.3507)_8$

9) Decimal to Hexadecimal :-

i) $(936)_{10}$

16	936	
16	58	8
16	3	A
	0	3

↑

$(936)_{10} = (3A8)_{16}$

ii)

10) Hexadecimal to binary :-

i) $(5CF2)_{16} = \overset{5}{0101}\ \overset{C}{1100}\ \overset{F}{1111}\ \overset{2}{0010}$

$= (1011100\ 1111001)_2$

ii) $(17E.F6)_{16} = \overset{1}{0001}\ \overset{7}{0111}\ \overset{E}{1110} \cdot \overset{F}{1111}\ \overset{6}{0110}$

$= (0001\ 0111 \cdot 1110 . 11110110)_2$

11) Hexadecimal to decimal :-

i) $(5C.F2)_{16} = \overset{5}{0101}\ \overset{C}{1100}\ \overset{F}{1111}\ \overset{2}{0010}$

$= 5 \times 16^1 + C \times 16^0 + F \times 16^{-1} + 2 \times 16^{-2}$

$= 80 + 12 + 0.9375 + 0.0078125$

$= (92.9453125)_{10}$

ii) $(D.2)_{16} = D \times 16^0 + 2 \times 16^{-1}$

$= 13.\cancel{0}.125$

$= (13.125)_{10}$

12) Hexadecimal to octal :-

i) $(1A7.68)_{16} = \overset{1}{0001}\ \overset{A}{1010}\ \overset{7}{0111} . \overset{6}{0110}\ \overset{8}{1000}$

$= \underline{110}\ \underline{100}\underline{111} . \underline{011}\underline{010}$

$= (647.32)_8$

ii)

Numbers

Decimal	Binary	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14
21	10101	25	15
22	10110	26	16
23	10111	27	17
24	11000	30	18
25	11001	31	19
26	11010	32	1A
27	11011	33	1B
28	11100	34	1C
29	11101	35	1D

○ Signed Binary Numbers :-

- The 8-bit binary no. representation is used to represent decimal no.s from -127 to $+127$

$$\underbrace{\boxed{b_7}}_{\text{Sign}}\ \underbrace{\boxed{b_6}\boxed{b_5}\boxed{b_4}\boxed{b_3}\boxed{b_2}\boxed{b_1}\boxed{b_0}}_{\text{Binary Number's}}$$

$0 \rightarrow +$ sign

$1 \rightarrow -$ sign

e.g : $(+8)_{10} = (00001000)_2$

$(-25)_{10} = (10011001)_2$

○ Binary Arithmetic :-

1 Binary Addition :-

Binary addition	Sum	Carry
0 + 0	0	0
0 + 1	1	0
1 + 0	1	0
1 + 1	0	1
1 + 1 + 1	1	1

e.g. i) $(1001)_2 + (0101)_2$

$$\begin{array}{r} \scriptstyle 1 \\ 1001 \\ +\,0101 \\ \hline 1110 \end{array}$$

$(1001)_2 + (0101)_2 = (1110)_2$

ii> $(110011)_2 + (1011101)_2$

→
```
   110011
+ 1011101
---------
 10010000
```

$(110011)_2 + (1011101)_2 = (10010000)_2$

iii> $(10101100)_2 + (0101011)_2$

→
```
  10101100
+ 01101011
----------
 100010111
```

$(10101100)_2 + (0101011)_2 = (100010111)_2$

iii> $(1111001110.1010)_2 + (1100110101.11001)_2$

→
```
   111100111.10100
+ 1100110101.11001
------------------
 10100011101.011101
```

iv> $(743)_8 + (141)_8 = (?)_2$

→ $(1111100011)_2 + (0011000001)_2$

```
  1111100011
+ 1001100001
------------
(1001000100)2
```

2. Binary Subtraction :-

Binary subtraction	Difference	Borrow
0 − 0	0	0
0 − 1	1	1
1 − 0	1	0
1 − 1	0	0

e.g i) $(1101)_2 - (1011)_2 = ?$

$\rightarrow$

```
   1          <------ Borrow
 1101
-1011
 -1
(0010)2       <------ Return
```

ii) $(1000)_2 - (110)_2 = ?$

$\rightarrow$

```
    1         <------ Borrow
 1 0 0 0
-  1 1 0
  -1
(0 0 1 0)2    <------ Return
```

iii) $(11001011.10110)_2 - (10110111.10010)_2 = ?$

$\rightarrow$

```
  1  1  1
 11001011.10110
-10110111.10010
 -1 -1  -1
(00010100.00100)2
```

iv) $(25)_{10} - (18)_{10} = ?$

$\rightarrow$

```
  11001
- 10010
  00111
```

$(25)_{10} - (18)_{10} = (7)_{10}$

3. Binary Multiplication :-

Binary multiplication i/p | o/p

i/p	o/p
0 × 0	0
0 × 1	0
1 × 0	0
1 × 1	1

e.g i) $(1101)_2 \times (101)_2 = (?)_2$

→

```
      1 1 0 1
    × 1 0 1
    ---------
      1 1 0 1
 +  0 0 0 0 x
  1 1 0 1 x x
  -----------
 (1 0 0 0 0 0 1)2
```

ii) $(1001)_2 \times (1000)_2 = (?)_2$

→

```
      1 0 0 1
    × 1 0 0 0
    ---------
      0 0 0 0
    0 0 0 0 x
 +  0 0 0 0 x x
  1 0 0 1 x x x
  -------------
 (1 0 0 1 0 0 0)2
```

iii) $(12)_{10} \times (6)_{10} = (?)_2$

```
       1 1 0 0
     × 0 1 1 0
     ---------
       0 0 0 0
     1 1 0 0 x
 + 1 1 0 0 x x
   0 0 0 0 x x x
   -------------
  (1 0 0 1 0 0 0)2
```

4. Binary Division :-

Dividend	Divisor	Quotient	Remainder
0	1	0	1
1	1	1	0

e.g i) $(1101101)_2 \div (101)_2 = ?$

→

```
       1010
101 ⟌ 1101101
     -101↓
      0011
        00
       111
      -101
       0100
      -000
       1001
      -101
```

Remainder $(0100)_2$

ii) $(11001)_2 \div (101)_2 = (?)_2$

→

```
       101  ⟶ Q
101 ⟌ 11001
     -101
      001
     -000
      00101
      -101
     (000)₂ ⟶ R
```

One's and two's Complement's Arithmetic :-

One's (1's) Complement :-

To obtained 1's complement of binary number change all '1's' with '0's' & all '0s' with '1's'.

Ex : $(110110110)_2$ ⟵ Binary

001001001 ⟵ 1's complement

Two's (2's) Complement :-

To obtained 2's complement of a binary number first take 1's complement of number & then add '1' to LSB of 1's complement.

Ex :- 110110110 ⟵ Binary
MSB LSB

001001001 ⟵ 1's complement
+ 1
001001010 ⟵ 2's Complement

Ex :- 0011101 ⟵ Binary
MSB LSB

~~0011101~~
~~+ 1~~
~~0011110~~

1100010 ⟵ 1's complement
+ 1
1100011 ⟵ 2's Complement

1) 1's complement Subtraction :-

Steps :-

- Find 1's complement of given Binary number.
- Add the 1's complemented binary numbers.
- In the addition if carry is 0, determine the 1's complement of addition outcome to get result of subtraction & the result is negative.
- If in addition, the carry is 1, the resultant number is +ve & no need of taking complement. Add a carry bit with LSB of result.

Ex. i) $(1001)_2 - (1101)_2$

→ Step 1) 1's complement of $(1101)_2$ → 0010

Step 2)

```
  1001  ——→ 1st Numer
  0010  ——→ 1's complement of 2nd number
  ----
  1011
```

There is no carry so carry = 0

Step 3) If carry = 0, Result is -ve

1's complement of $(1011)_2 \rightarrow (0100)_2$

$\therefore (1001)_2 - (1101)_2 = (0100)_2$

ii) $(1000)_2 - (0111)_2$

→ step 1) : 1's complement of $(0111)_2 \rightarrow (1000)_2$

step 2) :

$$\begin{array}{r} 1000 \\ 1000 \\ \hline 10000 \end{array}$$

As there is carry, the result is +ve

step 3) :

$$\begin{array}{r} 0000 \\ + \quad 1 \text{ carry} \\ \hline 0001 \end{array}$$

step 4) : If carry = 1, result is +ve

$\therefore (1000)_2 - (0111)_2 = +(0001)_2$

2) 2's Complement Subtraction :-

Steps :-

1. Find 2's complement of -ve number.
2. Add 2's complemented binary number's.
3. If carry = 0, determine 2's complement for addition to get the result & the result is negative.
4. If carry = 1, the resultant no. is +ve then discard carry.

Ex. i) $(1001)_2 - (1101)_2$

→ <u>step 1</u> : 2's complement of $(1101)_2 \rightarrow (0011)_2$

$$\begin{array}{r} 0010 \\ +\ \ 1 \\ \hline 0011 \end{array}$$

<u>step 2</u> : Add

$$\begin{array}{r} 1001 \\ 0011 \\ \hline 1100 \end{array}$$

There is no carry, carry = 0, result = -ve

<u>step 3</u> :- 2's complement of $(1100)_2 \rightarrow (0100)_2$

$\therefore (1001)_2 - (1101)_2 = -(0100)_2$

ii) $(1001)_2 - (0111)_2$

→ <u>step 1</u> : 2's complement of $(0111)_2 \rightarrow (1001)_2$

$$\begin{array}{r} 1000 \\ +\ \ 1 \\ \hline (1001)_2 \end{array}$$

<u>step 2</u> :- Add

$$\begin{array}{r} 1001 \\ +1001 \\ \hline 10010 \end{array}$$

carry is 1, result = +ve

$\therefore (1001)_2 - (0111)_2 = +(0010)_2$

Codes :-

- A group of binary bits that used to represent the character, numbers and symbols is defined as binary codes.

- Code is a symbolic representation of discrete information which may be present in the form of number's, letters or physical quantities.

- The group of symbol is called "code".

* Weighted Code :-
- There is a positional weighted.

- Each position of number represents a specific weight.

- Types of Weighted code :-
  - i) Decimal Number
  - ii) BCD or 8421 code.

* Non - Weighted code :-
- There is not a positional weighted.

- Each position with in a binary number is not assigned a fixed value.

- Types of Non- weighted code :-
  - i) Excess - 3 codes
  - ii) Gray code.

1. BCD (Binary Coded Decimal) code :-

- This is also known as 8-4-2-1 or weighted code

- In this code each decimal digit (0-9) is represented by 4-bit binary equivalent.

Ex :- i) $(396)_{10}$ ⟶ Binary (110001100)

↓↓↓

(0011 1001 0110) ⟶ (BCD)

ii) $(83)_{10}$

↓ ↓

(1000) (0011) (1000 0011) ⟶ BCD

16 8 4 2 1

iii) 27 — Binary - 1 0 1 1 1

27 — BCD - 0010 0111

Decimal	BCD
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

2. Excess - 3 (XS-3) Code :-

- It is non weighted code.
- In this first we find BCD of each decimal digit then add 3 to each decimal value.

Ex:- i) $(1\ 5\ 3\ 2)_{10}$

	0001	0101	0011	0010	⟶ BCD
+	0011	0011	0011	0011	⟶ Add 3
	(0100	1000	0110	0101 $)_{XS-3}$	

ii) 0010 ⟶ 2
+ 0011 ⟶ 3
0101 ⟶ 5

Decimal Number	Binary Number	Excess - 3 code
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	1 0 0 1
6	0 1 1 0	1 0 0 0
7	0 1 1 1	1 0 1 1
8	1 0 0 0	1 0 1 0
9	1 0 0 1	1 1 0 1

3. Gray Code :-

- It is non weighted code.

- It is introduce by frank Gray.

- It is also known as 'Reflective Code', 'Unit distance', 'Minimum error code'.

- There is a change of single bit in two successive codes

- Reduces process of switching.

Ex. Convert Binary to gray code :-

i) $(0101)_2 \longrightarrow$ 0 1 0 1

$\oplus$ $\oplus$ $\oplus$

(0 1 1 1) Gray

Ex - OR addition

A	B	A $\oplus$ B
0	0	0
0	1	1
1	0	1
1	1	0

ii) $(1100)_2 \longrightarrow$ 1 1 0 0

$\oplus$ $\oplus$ $\oplus$

(1 0 1 0) Gray

iii) $(27)_{10} \longrightarrow (?)$ Gray

16 8 4 2 1

$(27)_{10} \longrightarrow$ 1 1 0 1 1

(1 0 1 1 0) Gray.

o Convert Gray to Binary :-

i) (1 0 1 1) Gray $\longrightarrow$ 1 0 1 1 $\longrightarrow$ Gray

$(1\ 1\ 0\ 1)_2 \longrightarrow$ Binary

ii) (1 0 1 1 1 0) Gray $\longrightarrow$ 1 0 1 1 1 0 $\longrightarrow$ Gray

$(1\ 1\ 0\ 1\ 0\ 0)_2 \longrightarrow$ Binary

Decimal No.	Binary				Gray			
	B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

Error detecting and correcting Codes :-

- The codes that are used to detect the error are called " Error detection codes".

- The commonly used error detection and correction methods are :-
    1) Parity code
    2) Hamming code

Parity Code :-

- A parity code is an extra bit that to be added with data bits to detect the errors appear in digital transmission.

- Based on the bit values the parity code is classified as :-
    1) Even parity
    2) Odd parity

- In even parity, Parity bit is selected as '0' if no. of 1's present in the data is even.
  While the parity bit is selected as '1' if no. of 1's present in the data is odd.

e.g 1) Determine the odd parity and even parity bits for following 7-bit data.

i) 1011011

No. of 1's - 5 (i.e odd)

∴ Even parity for 1011011 is '1'.

∴ odd parity for 1011011 is '0'.

ii) 1010110
No. of 1's = 4 (i.e even)
∴ Even parity ⇒ '0'
odd parity ⇒ '1'

iii) 1101101
No. of 1's = 5 (i.e odd)
∴ Even parity ⇒ '1'
odd parity ⇒ '0'

○ Even and odd parity bit for 4-bit data :-

Data bits	Even parity	odd parity	Data bits	Even parity	odd parity
0000	0	1	1000	1	0
0001	1	0	1001	0	1
0010	1	0	1010	0	1
0011	0	1	1011	1	0
0100	1	0	1100	0	1
0101	0	1	1101	1	0
0110	0	1	1110	1	0
0111	1	0	1111	0	1