

o Number System :-

In general, any number system there is an ordered set of symbols known as digits with rules defined for performing arithmetic 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. $(101.011)_2 = (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) + (0 \times 2^{-1}) + (1 \times 2^{-2}) + (1 \times 2^{-3})$

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

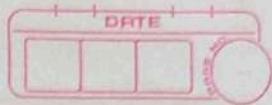
- Radix / Base - 8

e.g. $(734)_8$, $(614.25)_8$

- The general representation,

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

e.g. $(614.25)_8 = (6 \times 8^2) + (1 \times 8^1) + (4 \times 8^0) + (2 \times 8^{-1}) + (5 \times 8^{-2})$



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

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

* Number Conversion :-

1) Binary to Decimal :-

$$\begin{aligned} \text{i)} (11010)_2 &= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \\ &= 16 + 8 + 0 + 2 + 0 \\ &= (26)_{10} \end{aligned}$$

$$\begin{aligned} \text{ii)} (1101.1101)_2 &= 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + \\ &\quad 1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4} \\ &= 8 + 4 + 0 + 1 + 0.5 + 0.25 + 0 + 0.0625 \\ &= (13.8125)_{10} \end{aligned}$$

2) Binary to Octal :-

$$\begin{aligned} \text{i)} (11010100)_2 &= 011 \quad 010 \quad 100 \\ &\quad (3 \quad 2 \quad 4)_8 \end{aligned}$$

$$(11010100)_2 = (324)_8$$

$$\begin{aligned} \text{ii)} (10001101.0111010)_2 &= 010 \quad 001 \quad 101 \cdot 011 \quad 101 \quad 00 \\ &\quad 2 \quad 1 \quad 5 \quad 3 \quad 5 \quad 0 \\ (10001101.0111010)_2 &= (215.350)_8 \end{aligned}$$

3) Binary to Hexadecimal :-

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

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

4) Octal to Binary :-

$$\text{i)} (73.62)_8 = \begin{array}{r} 7 \\ 3 \\ \cdot \\ 6 \\ 2 \end{array} = 111011 \cdot 110010_2$$

$$\text{ii)} (67.357)_8 = \begin{array}{r} 6 \\ 7 \\ \cdot \\ 3 \\ 5 \\ 7 \end{array} = 110111 \cdot 011101111_2$$

5) Octal to Decimal :-

$$\text{i)} (164.35)_8 = 1 \times 8^2 + 6 \times 8^1 + 4 \times 8^0 + 3 \times 8^{-1} + 5 \times 8^{-2}$$

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

$$= (116.4375)_{10}$$

$$\text{ii)} (314)_8 = 3 \times 8^2 + 1 \times 8^1 + 4 \times 8^0$$

$$= 3 \times 64 + 8 + 4$$

$$= 192 + 12$$

$$= (204)_{10}$$

6) Octal to Hexadecimal :-

$$\text{i)} (647.32)_8 = \begin{array}{r} 6 \\ 4 \\ 7 \\ \cdot \\ 3 \\ 2 \end{array} = 110100111 \cdot 011010_2$$

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

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

$$\text{ii)} (13.125)_8 = \begin{array}{r} 1 \\ 3 \\ \cdot \\ 1 \\ 2 \\ 5 \end{array} = 001\ 011 \cdot 001\ 010\ 101_2$$

$$= 001011\cdot 001010101_2$$

$$= 0000\ 1011\cdot 0010\ 1010\ 1000$$

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

4) Octal to Binary :-

$$\text{i)} (73.62)_8 = \begin{matrix} 7 & 3 \\ 1 & 1 & 0 & 1 & 1 & \cdot & 1 & 1 & 0 & 0 & 1 & 0 \end{matrix} = (1110111.110010)_2$$

$$\text{ii)} (67.357)_8 = \begin{matrix} 6 & 7 & 3 & 5 & 7 \\ 1 & 1 & 0 & 1 & 1 & 1 & \cdot & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{matrix} = (110111.011101111)_2$$

5) Octal to Decimal :-

$$\text{i)} (164.35)_8 = 1 \times 8^2 + 6 \times 8^1 + 4 \times 8^0 + 3 \times 8^{-1} + 5 \times 8^{-2} \\ = 64 + 48 + 4 + 0.375 + 0.0625 \\ = (116.4375)_{10}$$

$$\text{ii)} (314)_8 = 3 \times 8^2 + 1 \times 8^1 + 4 \times 8^0 \\ = 3 \times 64 + 8 + 4 \\ = 192 + 12 \\ = (204)_{10}$$

6) Octal to Hexadecimal :-

$$\text{i)} (647.32)_8 = \begin{matrix} 6 & 4 & 7 & \cdot & 3 & 2 \\ 1 & 1 & 0 & 1 & 0 & 0 \end{matrix} = 110100111.011010 \\ = 0001\ 1010\ 0111\cdot 0110\ 1000$$

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

$$\text{ii)} (13.125)_8 = \begin{matrix} 1 & 3 & \cdot & 1 & 2 & 5 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{matrix} = 001011\cdot 00101010 \\ = 0000\ 1011\cdot 0010\ 1010\ 1000$$

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

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

Convert fractional

2	42	LSB	$= 0.125$
2	21	0	$- 0.125 \times 2 = 0.25 \rightarrow 0$
2	10	1	$- 0.25 \times 2 = 0.5 \rightarrow 0$
2	5	0	$- 0.5 \times 2 = 1 \rightarrow 1$
2	2	1	
2	1	0	
0	1	MSB	

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

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

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



8) Decimal to octal :-

i) $(135)_{10}$

8	135	
8	16	7
8	2	0
0	2	

MSB

$(135)_{10} = (207)8$

ii) $(245.455)_{10}$

8	245	
8	30	5
8	3	6
0	3	MSB

LSB

$$0.455 \times 8 = 3.64 \rightarrow 3$$

$$0.64 \times 8 = 5.12 \rightarrow 5$$

$$0.12 \times 8 = 0.96 \rightarrow 0$$

$$0.96 \times 8 = 7.68 \rightarrow 7$$

$(245.455)_{10} = (365.3507)8$

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 :-

$$\text{i)} (5CF2)_{16} = \begin{array}{r} 5 \\ | \\ 0101 \end{array} \begin{array}{r} C \\ | \\ 1100 \end{array} \begin{array}{r} F \\ | \\ 1111 \end{array} \begin{array}{r} 2 \\ | \\ 0010 \end{array}$$
$$= (10111001111001)_2$$

$$\text{ii)} (17E.F6)_{16} = \begin{array}{r} 1 \\ | \\ 0001 \end{array} \begin{array}{r} 7 \\ | \\ 0111 \end{array} \begin{array}{r} E \\ | \\ 1110 \end{array} \begin{array}{r} F \\ | \\ 1111 \end{array} \begin{array}{r} 6 \\ | \\ 0110 \end{array}$$
$$= (00010111\cdot1110\cdot11110110)_2$$

11) Hexadecimal to decimal :-

$$\text{i)} (5CF2)_{16} = \begin{array}{r} 5 \\ | \\ 0101 \end{array} \begin{array}{r} C \\ | \\ 1100 \end{array} \begin{array}{r} F \\ | \\ 1111 \end{array} \begin{array}{r} 2 \\ | \\ 0010 \end{array}$$
$$= 5 \times 16^4 + C \times 16^3 + F \times 16^2 + 2 \times 16^1$$
$$= 80 + 12 + 0.9375 + 0.0078125$$
$$= (92.9453125)_{10}$$

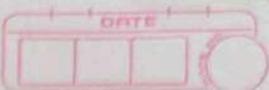
$$\text{ii)} (D.2)_{16} = \begin{array}{r} D \\ | \\ 0101 \end{array} \begin{array}{r} 2 \\ | \\ 1100 \end{array}$$
$$= D \times 16^0 + 2 \times 16^{-1}$$
$$= 13 + 0.125$$
$$= (13.125)_{10}$$

12) Hexadecimal to octal :-

$$\text{i)} (1A7.68)_{16} = \begin{array}{r} 1 \\ | \\ 0001 \end{array} \begin{array}{r} A \\ | \\ 1010 \end{array} \begin{array}{r} 7 \\ | \\ 0111 \end{array} \begin{array}{r} : \\ | \\ 6 \end{array} \begin{array}{r} 8 \\ | \\ 0110 \end{array} \begin{array}{r} : \\ | \\ 1000 \end{array}$$
$$= \underline{\underline{110}} \underline{\underline{100111}} \cdot \underline{\underline{011010}}$$
$$= (647.32)_8$$

ii)

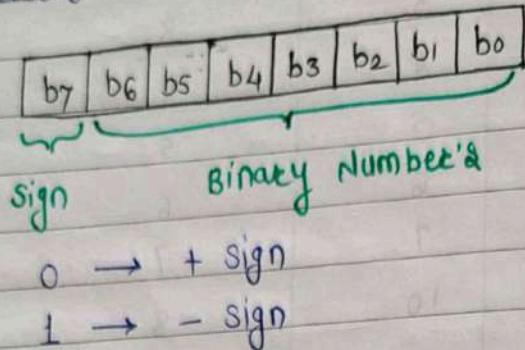
o Numbers



Decimal	Binary	Octal	Hexadecimal
0	0 0 0 0	0	0
1	0 0 0 1	1	1
2	0 0 1 0	2	2
3	0 0 1 1	3	3
4	0 1 0 0	4	4
5	0 1 0 1	5	5
6	0 1 1 0	6	6
7	0 1 1 1	7	7
8	1 0 0 0	10	8
9	1 0 0 1	11	9
10	1 0 1 0	12	A
11	1 0 1 1	13	B
12	1 1 0 0	14	C
13	1 1 0 1	15	D
14	1 1 1 0	16	E
15	1 1 1 1	17	F
16	1 0 0 0 0	20	10
17	1 0 0 0 1	21	11
18	1 0 0 1 0	22	12
19	1 0 0 1 1	23	13
20	1 0 1 0 0	24	14
21	1 0 1 0 1	25	15
22	1 0 1 1 0	26	16
23	1 0 1 1 1	27	17
24	1 1 0 0 0	30	18
25	1 1 0 0 1	31	19
26	1 1 0 1 0	32	1A
27	1 1 0 1 1	33	1B
28	1 1 1 0 0	34	1C
29	1 1 1 0 1	35	1D

Signed Binary Numbers :-

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



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) $(1001)_2 + (0101)_2$

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

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



$$\text{ii)} (110011)_2 + (1011101)_2$$

$$\rightarrow \begin{array}{r} 11111 \\ 1110011 \\ + 11011101 \\ \hline 10010000 \end{array}$$

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

$$\text{iii)} (10101100)_2 + (0101011)_2$$

$$\rightarrow \begin{array}{r} 111 \\ 10101100 \\ + 10110101 \\ \hline 100010111 \end{array}$$

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

$$\text{iv)} (111100111 \cdot 1010)_2 + (1100110101 \cdot 11001)_2$$

$$\rightarrow \begin{array}{r} 111100111 \cdot 1010 \\ + 1100110101 \cdot 11001 \\ \hline 10100011101 \cdot 01101 \end{array}$$

$$\text{iv)} (743)_8 + (141)_8 = (?)_2$$

$$\rightarrow (111100011)_2 + (001100001)_2$$

$$\begin{array}{r} 111100011 \\ + 001100001 \\ \hline 1001000100 \end{array}$$

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) $(1101)_2 - (1011)_2 = ?$

$$\begin{array}{r}
 \xrightarrow{\quad} & \begin{array}{r} 1 \\ 1101 \end{array} & \leftarrow \text{Borrow} \\
 - & \underline{1011} & \\
 \hline
 & (0010)_2 & \leftarrow \text{Return}
 \end{array}$$

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

$$\begin{array}{r}
 \xrightarrow{\quad} & \begin{array}{r} 1 \\ 1000 \end{array} & \leftarrow \text{Borrow} \\
 - & \underline{110} & \\
 \hline
 & (0010)_2 & \leftarrow \text{Return}
 \end{array}$$

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

$$\begin{array}{r}
 \xrightarrow{\quad} & \begin{array}{r} 1 \\ 11001011.10110 \end{array} \\
 - & \underline{10110111.10010} \\
 \hline
 & (00010100.00100)_2
 \end{array}$$

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

$$\begin{array}{r}
 \xrightarrow{\quad} & \begin{array}{r} 11001 \\ - 10010 \\ \hline 00111 \end{array} \\
 & (25)_{10} - (18)_{10} = (7)_{10}
 \end{array}$$

3. Binary Multiplication :-

Binary multiplication i/p

$$0 \times 0$$

$$010$$

$$0 \times 1$$

$$0$$

$$1 \times 0$$

$$0$$

$$1 \times 1$$

$$1$$

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

$$\begin{array}{r} 1101 \\ \times 101 \\ \hline 1101 \\ + 0000x \\ \hline 1101xx \\ (1000001)_2 \end{array}$$

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

$$\begin{array}{r} 1001 \\ \times 1000 \\ \hline 0000 \\ 0000x \\ + 0000xx \\ \hline 1001xxxx \\ (1001000)_2 \end{array}$$

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

$$\begin{array}{r} 1100 \\ \times 0110 \\ \hline 0000 \\ 1100x \\ + 11100xx \\ \hline 00000xxx \\ (1001000)_2 \end{array}$$

4. Binary Division :-

Dividend Divisor quotient remainder

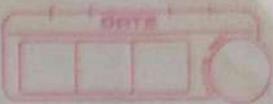
0	1	0	1
1	1	1	0

$$\text{e.g.) } (1101101)_2 \div (101)_2 = ?$$

$$\begin{array}{r} 1010 \\ \overline{)1101101} \\ -101 \\ \hline 0011 \\ -00 \\ \hline 111 \\ -101 \\ \hline 0100 \\ -000 \\ \hline 1001 \\ -101 \\ \hline 000 \\ \text{Remainder } (0100)_2 \end{array}$$

$$\text{i)} (11001)_2 \div (101)_2 = (?)_2$$

$$\begin{array}{r} 101 \longrightarrow Q \\ \overline{)11001} \\ -101 \\ \hline 001 \\ -000 \\ \hline 00101 \\ -101 \\ \hline 000 \\ \text{--- } (000)_2 \longrightarrow R \end{array}$$



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

o One's (1's) Complement :-

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

$$\text{Ex: } (110110110)_2 \quad \leftarrow \text{Binary}$$

$$001001001 \quad \leftarrow 1\text{'s complement}$$

o 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.

$$\text{Ex: } \begin{array}{r} 110110110 \\ \text{MSB} \qquad \qquad \qquad \text{LSB} \end{array} \quad \leftarrow \text{Binary}$$

$$001001001 \quad \leftarrow 1\text{'s complement}$$

$$\underline{+ \ 1}$$

$$001001010 \quad \leftarrow 2\text{'s complement}$$

$$\text{Ex: } \begin{array}{r} 0011101 \\ \text{MSB} \qquad \qquad \qquad \text{LSB} \end{array} \quad \leftarrow \text{Binary}$$

$$\begin{array}{r} 0/0/1/1/0/1 \\ \text{---} \qquad \qquad \qquad + \ 1 \\ 0011110 \end{array} \quad \begin{array}{r} 1100010 \\ \text{---} \qquad \qquad \qquad + \ 1 \\ 1100011 \end{array} \quad \leftarrow 1\text{'s complement}$$

$$\begin{array}{r} 0011110 \\ \text{---} \qquad \qquad \qquad + \ 1 \\ 1100011 \end{array} \quad \leftarrow 2\text{'s complement}$$

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 $(1101)_2 \rightarrow 0010$

Step 2) $\begin{array}{r} 1001 \\ 0010 \\ \hline 1011 \end{array}$ → 1st Number
→ 1's complement of 2nd number

There is no carry so carry = 0

Step 3) IF carry = 0, Result is -ve
1's complement of $(1011)_2 \rightarrow (0100)_2$

∴ $(1001)_2 - (1101)_2 = (0100)_2$

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

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

Step 2) : 1000

$$\begin{array}{r} 1000 \\ 1000 \\ \hline 0000 \end{array}$$

As there is carry, the result is +ve

Step 3) : 0000

$$\begin{array}{r} + 1 \\ \hline 0001 \end{array}$$

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 numbers.
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) $(1001)_2 - (1101)_2$

→ Step 1 : 2's complement of $(1101)_2 \rightarrow (0011)_2$

Step 2 : Add

$$\begin{array}{r} 1\ 0\ 0\ 1 \\ + 1 \\ \hline 0\ 0\ 1\ 1 \\ \hline 1\ 1\ 0\ 0 \end{array}$$

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

Step 3 :- 2's complement of $(1100)_2 \rightarrow (0100)_2$

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

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

→ Step 1 : 2's complement of $(0111)_2 \rightarrow (1001)_2$

Step 2 :- Add

$$\begin{array}{r} 1\ 0\ 0\ 1 \\ + 1 \\ \hline 1\ 0\ 0\ 1 \\ \hline 0\ 0\ 1\ 0 \end{array}$$

Carry is 1, result = +ve

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

o 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 numbers, 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.

o Types of Weighted code :-

- i> Decimal Number
- ii> BCD or 8421 code.

* Non - Weighted code :-

- There is not a positional weighted.
- Each position within a binary number is not assigned a fixed value.

o Types of Non- weighted code:-

- i> Excess - 3 codes
- ii> Gray code.

1 BCD or 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

168421

iii) 27 — Binary - 10111

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 (X_{S-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)

$$\begin{array}{r}
 (1 \ 5 \ 3 \ 2)_{10} \\
 \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
 \begin{array}{cccc}
 0001 & 0101 & 0011 & 0010 \\
 + 0011 & 0011 & 0011 & 0011 \\
 \hline
 0100 & 1000 & 0110 & 0101
 \end{array}
 \xrightarrow{\text{BCD}}
 \xrightarrow{\text{Add 3}}
 \end{array}$$

$(0100 \ 1000 \ 0110 \ 0101)_{X_{S-3}}$

ii)

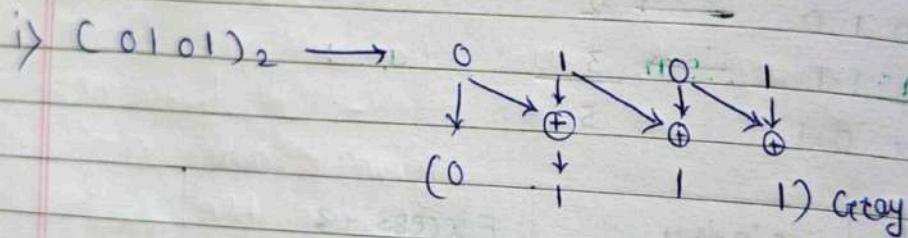
$$\begin{array}{r}
 0010 \rightarrow 2 \\
 + 0011 \rightarrow 3 \\
 \hline
 0101
 \end{array}$$

Decimal Number	Binary Number	Excess - 3 code
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1001
6	0110	1000
7	0111	1011
8	1000	1010
9	1001	1101

3. Gray code :-

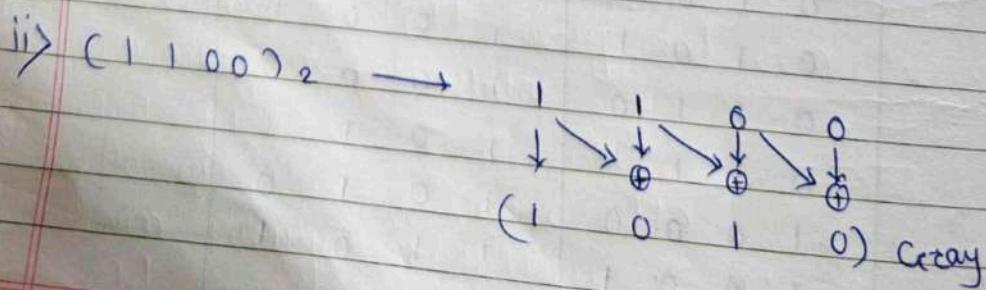
- It is non weighted code.
- It is introduced by Frank Gray.
- It is also known as 'Reflective code', 'Unit distance', 'Minimum error code'.
- There is a change of single bit in two successive codes.
- Reduces process of switching.

Ex. i) Convert binary to gray code :-



Ex - OR addition

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



iii) $(27)_{10} \rightarrow (?) \text{ array}$

16 8 4 2 1

$(27)_{10} \rightarrow \begin{array}{ccccc} 1 & 1 & 0 & 1 & 1 \\ \downarrow & \searrow & \downarrow & \searrow & \downarrow \\ 0 & 1 & 1 & 0 & 1 \end{array}$

(1 0 1 1 0) array.

o convert array to binary :-

i) (1 0 1 1) array $\rightarrow \begin{array}{cccc} 1 & 0 & 1 & 1 \\ \downarrow & \searrow & \downarrow & \searrow \\ 0 & 1 & 0 & 1 \end{array} \rightarrow \text{array}$

$(1 \ 1 \ 0 \ 1)_2 \rightarrow \text{Binary}$

ii) (1 0 1 1 0) array $\rightarrow \begin{array}{ccccc} 1 & 0 & 1 & 1 & 0 \\ \downarrow & \searrow & \downarrow & \searrow & \downarrow \\ 0 & 1 & 0 & 1 & 0 \end{array} \rightarrow \text{array}$

$(1 \ 1 \ 0 \ 1 \ 0)_2 \rightarrow \text{Binary}$

Decimal No.	Binary				array			
	B ₃	B ₂	B ₁	B ₀	C ₃	C ₂	C ₁	C 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

o 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 is 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.

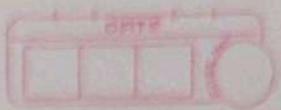
eg i) Determine the odd parity and even parity bits for following 7-bit data.

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 \Rightarrow '0'
Odd parity \Rightarrow '1'

iii) 1101101

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

∴ Even parity \Rightarrow '1'
Odd parity \Rightarrow '0'

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

Data bits	Even parity	odd parity	Data bits	Even parity	odd parity
0 0 0 0	0	1	1 0 0 0	1	0
0 0 0 1	1	0	1 0 0 1	0	1
0 0 1 0	1	0	1 0 1 0	0	1
0 0 1 1	0	1	1 0 1 1	1	0
0 1 0 0	1	0	1 1 0 0	0	1
0 1 0 1	0	1	1 1 0 1	1	0
0 1 1 0	0	1	1 1 1 0	1	0
0 1 1 1	1	0	1 1 1 1	0	1