NUMBER SYSTEM AND CODES

INTRODUCTION:-

- The term digital refers to a process that is achieved by using discrete unit.
- In number system there are different symbols and each symbol has an absolute value and also has place value.

RADIX OR BASE:-

The radix or base of a number system is defined as the number of different digits which can occur in each position in the number system.

RADIX POINT :-

The generalized form of a decimal point is known as radix point. In any positional number system the radix point divides the integer and fractional part.

N_r = [Integer part . Fractional part]

↑ Radix point

NUMBER SYSTEM:-

In general a number in a system having base or radix ' r ' can be written as

$$a_n \ a_{n-1} \ a_{n-2} \ \ldots\ldots\ldots\ldots\ldots \ a_0 \ . \ a_{-1} \ a_{-2} \ldots\ldots\ldots\ldots a_{-m}$$

This will be interpreted as

$$Y = a_n \times r^n + a_{n-1} \times r^{n-1} + a_{n-2} \times r^{n-2} + \ldots\ldots\ldots + a_0 \times r^0 + a_{-1} \times r^{-1} + a_{-2} \times r^{-2} + \ldots\ldots\ldots + a_{-m} \times r^{-m}$$

where Y = value of the entire number

a_n = the value of the n^{th} digit

r = radix

TYPES OF NUMBER SYSTEM:-

There are four types of number systems. They are

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

DECIMAL NUMBER SYSTEM:-

- The decimal number system contain ten unique symbols 0,1,2,3,4,5,6,7,8 and 9.
- In decimal system 10 symbols are involved, so the base or radix is 10.
- It is a positional weighted system.
- The value attached to the symbol depends on its location with respect to the decimal point.

In general,

$$d_n \quad d_{n-1} \quad d_{n-2} \quad \ldots\ldots\ldots\ldots \quad d_0 \;.\; d_{-1} \quad d_{-2} \ldots\ldots\ldots d_{-m}$$

is given by

$$(d_n \times 10^n) + (d_{n-1} \times 10^{n-1}) + (d_{n-2} \times 10^{n-2}) + \ldots + (d_0 \times 10^0) + (d_{-1} \times 10^{-1}) + (d_{-2} \times 10^{-2}) + \ldots + (d_{-m} \times 10^{-m})$$

For example:-

9256.26 = 9 x 1000 + 2 x 100 + 5 x 10 + 6 x 1 + 2 x (1/10) + 6 x (1/100)

= $9 \times 10^3 + 2 \times 10^2 + 5 \times 10^1 + 6 \times 10^0 + 2 \times 10^{-1} + 6 \times 10^{-2}$

BINARY NUMBER SYSTEM:-

- The binary number system is a positional weighted system.
- The base or radix of this number system is 2.
- It has two independent symbols.
- The symbols used are 0 and 1.
- A binary digit is called a bit.
- The binary point separates the integer and fraction parts.

In general,

$$d_n \quad d_{n-1} \quad d_{n-2} \quad \ldots\ldots\ldots\ldots \quad d_0 \;.\; d_{-1} \quad d_{-2} \ldots\ldots\ldots d_{-k}$$

is given by

$$(d_n \times 2^n) + (d_{n-1} \times 2^{n-1}) + (d_{n-2} \times 2^{n-2}) + \ldots + (d_0 \times 2^0) + (d_{-1} \times 2^{-1}) + (d_{-2} \times 2^{-2}) + \ldots + (d_{-k} \times 2^{-k})$$

OCTAL NUMBER SYSTEM:-

- It is also a positional weighted system.
- Its base or radix is 8.
- It has 8 independent symbols 0,1,2,3,4,5,6 and 7.
- Its base 8 = 2^3 , every 3- bit group of binary can be represented by an octal digit.

HEXADECIMAL NUMBER SYSTEM:-

- The hexadecimal number system is a positional weighted system.
- The base or radix of this number system is 16.
- The symbols used are 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E and F
- The base 16 = 24 , every 4 – bit group of binary can be represented by an hexadecimal digit.

CONVERSION FROM ONE NUMBER SYSTEM TO ANOTHER :-

1. **BINARY NUMBER SYSTEM:-**

(a) Binary to decimal conversion:-

In this method, each binary digit of the number is multiplied by its positional weight and the product terms are added to obtain decimal number.

For example:

(i) Convert $(10101)_2$ to decimal.

Solution :

(Positional weight)	$2^4\ 2^3\ 2^2\ 2^1\ 2^0$
Binary number	10101
	$= (1 \times 2^4) + (0 \times 2^3) + (1\times 2^2) + (0 \times 2^1) + (1 \times 2^0)$
	$= 16 + 0 + 4 + 0 + 1$
	$= (21)_{10}$

(ii) Convert $(111.101)_2$ to decimal.

Solution:

$(111.101)_2 = (1 \times 2^2) + (1 \times 2^1) + (1\times 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3})$

$= 4 + 2 + 1 + 0.5 + 0 + 0.125$

$= (7.625)_{10}$

(b) Binary to Octal conversion:-

For conversion binary to octal the binary numbers are divided into groups of 3 bits each, starting at the binary point and proceeding towards left and right.

Octal	Binary	Octal	Binary
0	000	4	100
1	001	5	101
2	010	6	110
3	011	7	111

For example:

(i) Convert $(101111010110.110110011)_2$ into octal.

Solution :

Group of 3 bits are	101	111	010	110	.	110	110	011
Convert each group into octal =	5	7	2	6	.	6	6	3

The result is **$(5726.663)_8$**

(ii) Convert $(10101111001.0111)_2$ into octal.

Solution :

Binary number	10	101	111	001	.	011	1
Group of 3 bits are =	010	101	111	001	.	011	100
Convert each group into octal =	2	5	7	1	.	3	4

The result is **$(2571.34)_8$**

(c) Binary to Hexadecimal conversion:-

For conversion binary to hexadecimal number the binary numbers starting from the binary point, groups are made of 4 bits each, on either side of the binary point.

Hexadecimal	Binary	Hexadecimal	Binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

For example:
(i) Convert $(1011011011)_2$ into hexadecimal.

Solution:

Given Binary number	10	1101	1011
Group of 4 bits are	0010	1101	1011
Convert each group into hex =	2	D	B

The result is $(\textbf{2DB})_{16}$

(ii) Convert $(01011111011.011111)_2$ into hexadecimal.

Solution:

Given Binary number	010	1111	1011	.	0111	11
Group of 3 bits are =	0010	1111	1011	.	0111	1100
Convert each group into octal =	2	F	B	.	7	C

The result is $(\textbf{2FB.7C})_{16}$

2. DECIMAL NUMBER SYSTEM:-

(a)Decimal to binary conversion:-

In the conversion the integer number are converted to the desired base using successive division by the base or radix.

For example:
(i) Convert $(52)_{10}$ into binary.

Solution:

Divide the given decimal number successively by 2 read the integer part remainder upwards to get equivalent binary number. Multiply the fraction part by 2. Keep the integer in the product as it is and multiply the new fraction in the product by 2. The process is continued and the integer are read in the products from top to bottom.

2 | 52	
2 | 26	— 0
2 | 13	— 0
2 | 6	— 1
2 | 3	— 0
2 | 1	— 1
0	— 1

Result of $(52)_{10}$ is $\mathbf{(110100)_2}$

(ii) Convert $(105.15)_{10}$ into binary.

Solution:

Integer part		Fraction part
2 | 105		0.15 x 2 = 0.30
2 | 52	— 1	0.30 x 2 = 0.60
2 | 26	— 0	0.60 x 2 = 1.20
2 | 13	— 0	0.20 x 2 = 0.40
2 | 6	— 1	0.40 x 2 = 0.80
2 | 3	— 0	0.80 x 2 = 1.60
2 | 1	— 1	
0	— 1	

Result of $(105.15)_{10}$ is $\mathbf{(1101001.001001)_2}$

(b)Decimal to octal conversion:-

To convert the given decimal integer number to octal, successively divide the given number by 8 till the quotient is 0. To convert the given decimal fractions to octal successively multiply the decimal fraction and the subsequent decimal fractions by 8 till the product is 0 or till the required accuracy is obtained.

For example:

(i) Convert $(378.93)_{10}$ into octal.

Solution:

8 | 378		0.93 x 8 = 7.44
8 | 47	— 2	0.44 x 8 = 3.52
8 | 5	— 7	0.52 x 8 = 4.16
0	— 5	0.16 x 8 = 1.28

Result of $(378.93)_{10}$ is $\mathbf{(572.7341)_8}$

(c)Decimal to hexadecimal conversion:-

The decimal to hexadecimal conversion is same as octal.

For example:

(i) Convert $(2598.675)_{10}$ into hexadecimal.

Solution:

	Remainder Decimal	Remainder Hex		Hex
16 | 2598			0.675 x 16 = 10.8	A
16 | 162	— 6	6	0.800 x 16 = 12.8	C
16 | 10	— 2	2	0.800 x 16 = 12.8	C
0	— 10	A	0.800 x 16 = 12.8	C

Result of $(2598.675)_{10}$ is $\mathbf{(A26.ACCC)_{16}}$

3. OCTAL NUMBER SYSTEM:-

(a)Octal to binary conversion:-

To convert a given a octal number to binary, replace each octal digit by its 3- bit binary equivalent.

For example:

Convert $(367.52)_8$ into binary.

Solution:

Given Octal number is	3	6	7	.	5	2
Convert each group octal to binary	= 011	110	111	.	101	010

Result of $(367.52)_8$ is **$(011110111.101010)_2$**

(b) Octal to decimal conversion:-

For conversion octal to decimal number, multiply each digit in the octal number by the weight of its position and add all the product terms

For example: -

Convert $(4057.06)_8$ to decimal

Solution:

$$(4057.06)_8 = 4 \times 8^3 + 0 \times 8^2 + 5 \times 8^1 + 7 \times 8^0 + 0 \times 8^{-1} + 6 \times 8^{-2}$$
$$= 2048 + 0 + 40 + 7 + 0 + 0.0937$$
$$= (2095.0937)_{10}$$

Result is **$(2095.0937)_{10}$**

(c) Octal to hexadecimal conversion:-

For conversion of octal to Hexadecimal, first convert the given octal number to binary and then binary number to hexadecimal.

For example :-

Convert $(756.603)_8$ to hexadecimal.

Solution :-

Given octal no.		7	5	6	.	6	0	3
Convert each octal digit to binary	=	111	101	110	.	110	000	011
Group of 4bits are	=	0001	1110	1110	.	1100	0001	1000
Convert 4 bits group to hex.	=	1	E	E	.	C	1	8

Result is **$(1EE.C18)_{16}$**

(4) HEXADECIMAL NUMBER SYSTEM :-

(a) Hexadecimal to binary conversion:-

For conversion of hexadecimal to binary, replace hexadecimal digit by its 4 bit binary group.

For example:

Convert $(3A9E.B0D)_{16}$ into binary.

Solution:

Given Hexadecimal number is	3	A	9	E	.	B	0	D
Convert each hexadecimal digit to 4 bit binary	= 0011	1010	1001	1110	.	1011	0000	1101

Result of $(3A9E.B0D)_8$ is **$(0011101010011110.101100001101)_2$**

(b) Hexadecimal to decimal conversion:-

For conversion of hexadecimal to decimal, multiply each digit in the hexadecimal number by its position weight and add all those product terms.

For example: -
Convert $(A0F9.0EB)_{16}$ to decimal

Solution:

$(A0F9.0EB)_{16}$ = $(10 \times 16^3)+(0 \times 16^2)+(15 \times 16^1)+(9 \times 16^0)+(0 \times 16^{-1})+(14 \times 16^{-2})+(11 \times 16^{-3})$
= $40960 + 0 + 240 + 9 + 0 + 0.0546 + 0.0026$
= $(41209.0572)_{10}$

Result is $\mathbf{(41209.0572)_{10}}$

(c) Hexadecimal to Octal conversion:-

For conversion of hexadecimal to octal, first convert the given hexadecimal number to binary and then binary number to octal.

For example :-
Convert $(B9F.AE)_{16}$ to octal.

Solution :-

Given hexadecimal no.is		B	9	F	.	A	E		
Convert each hex. digit to binary	=	1011	1001	1111	.	1010	1110		
Group of 3 bits are	=	101	110	011	111	.	101	011	100
Convert 3 bits group to octal.	=	5	6	3	7	.	5	3	4

Result is $\mathbf{(5637.534)_8}$

BINARY ARITHEMATIC OPERATION :-

1. BINARY ADDITION:-

The binary addition rules are as follows

0 + 0 = 0 ; 0 + 1 = 1 ; 1 + 0 = 1 ; 1 + 1 = 10 , i.e 0 with a carry of 1

For example :-

Add $(100101)_2$ and $(1101111)_2$.
Solution :-

```
       1 0 0 1 0 1
  +  1 1 0 1 1 1 1
  ----------------
   1 0 0 1 0 1 0 0
```

Result is $\mathbf{(10010100)_2}$

2. BINARY SUBTRACTION:-

The binary subtraction rules are as follows

0 - 0 = 0 ; 1 - 1 = 0 ; 1 - 0 = 1 ; 0 - 1 = 1 , with a borrow of 1

For example :-

Substract $(111.111)_2$ from $(1010.01)_2$.

Solution :-

```
    1010.010
-    111.111
    0010.011
```

Result is **$(0010.011)_2$**

3. BINARY MULTIPLICATION:-

The binary multiplication rules are as follows

0 x 0 = 0 ; 1 x 1 = 1 ; 1 x 0 = 0 ; 0 x 1 = 0

For example :-

Multiply $(1101)_2$ by $(110)_2$.

Solution :-

```
        1101
x        110
        0000
       1101
+     1101
     1001110
```

Result is **$(1001110)_2$**

4. BINARY DIVISION:-

The binary division is very simple and similar to decimal number system. The division by '0' is meaningless. So we have only 2 rules

0 ÷ 1 = 0

1 ÷ 1 = 1

For example :-

Divide $(10110)_2$ by $(110)_2$.

Solution :-

```
110 ) 101101 ( 111.1
    -  110
        1010
         110
         1001
          110
          110
          110
          000
```

Result is **$(111.1)_2$**

1's COMPLEMENT REPRESENTATION :-

The 1's complement of a binary number is obtained by changing each 0 to 1 and each 1 to 0.

For example :-

Find $(1100)_2$ 1's complement.

Solution :-

Given	1	1	0	0
1's complement is	0	0	1	1

Result is $\mathbf{(0011)_2}$

2's COMPLEMENT REPRESENTATION :-

The 2's complement of a binary number is a binary number which is obtained by adding 1 to the 1's complement of a number i.e.

2's complement = 1's complement + 1

For example :-

Find $(1010)_2$ 2's complement.

Solution :-

Given		1	0	1	0
1's complement is		0	1	0	1
	+				1
2's complement		0	1	1	0

Result is $\mathbf{(0110)_2}$

SIGNED NUMBER :-

In sign – magnitude form, additional bit called the sign bit is placed in front of the number. If the sign bit is 0, the number is positive. If it is a 1, the number is negative.

For example:-

0 1 0 1 0 0 1 = +41

↑

Sign bit

1 1 0 1 0 0 1 = -41

↑

Sign bit

SUBSTRACTION USING COMPLEMENT METHOD :-

1's COMPLEMENT:-

In 1's complement subtraction, add the 1's complement of subtrahend to the minuend. If there is a carry out, then the carry is added to the LSB. This is called end around carry. If the MSB is 0, the result is positive. If the MSB is 1, the result is negative and is in its 1's complement form. Then take its 1's complement to get the magnitude in binary.

For example:-

Subtract $(10000)_2$ from $(11010)_2$ using 1's complement.

Solution:-

```
  1 1 0 1 0                   1 1 0 1 0                        =   26
- 1 0 0 0 0      =>        +  0 1 1 1 1  (1's complement)     = - 16
                Carry  →    1 0 1 0 0 1                          + 10
                           +          1
                              0 1 0 1 0   = +10
```

Result is **+10**

2's COMPLEMENT:-

In 2's complement subtraction, add the 2's complement of subtrahend to the minuend. If there is a carry out, ignore it. If the MSB is 0, the result is positive. If the MSB is 1, the result is negative and is in its 2's complement form. Then take its 2's complement to get the magnitude in binary.

For example:-

Subtract $(1010100)_2$ from $(1010100)_2$ using 2's complement.

Solution:-

```
  1 0 1 0 1 0 0                 1 0 1 0 1 0 0                           =    84
- 1 0 1 0 1 0 0      =>      +  0 1 0 1 1 0 0  (2's complement)        = -  84
                              1  0 0 0 0 0 0 0 ( Ignore the carry)          0
                       =                  0 (result = 0)
```

Hence MSB is 0. The answer is positive. So it is +0000000 = **0**

DIGITAL CODES:-

In practice the digital electronics requires to handle data which may be numeric, alphabets and special characters. This requires the conversion of the incoming data into binary format before it can be processed. There is various possible ways of doing this and this process is called encoding. To achieve the reverse of it, we use decoders.

WEIGHTED AND NON-WEIGHTED CODES:-

There are two types of binary codes

1) Weighted binary codes
2) Non- weighted binary codes

In weighted codes, for each position (or bit) ,there is specific weight attached.

For example, in binary number, each bit is assigned particular weight 2n where 'n' is the bit number for n = 0,1,2,3,4 the weights are 1,2,4,8,16 respectively.

Example :- BCD

Non-weighted codes are codes which are not assigned with any weight to each digit position, i.e., each digit position within the number is not assigned fixed value.

Example:- Excess – 3 (XS -3) code and Gray codes

BINARY CODED DECIMAL (BCD):-

BCD is a weighted code. In weighted codes, each successive digit from right to left represents weights equal to some specified value and to get the equivalent decimal number add the products of the weights by the corresponding binary digit. 8421 is the most common because 8421 BCD is the most natural amongst the other possible codes.

For example:-

$(567)_{10}$ is encoded in various 4 bit codes.

Solution:-

Decimal	→	5	6	7
8421 code	→	0101	0110	0111
6311 code	→	0111	1000	1001
5421 code	→	1000	0100	1010

BCD ADDITION:-

Addition of BCD (8421) is performed by adding two digits of binary, starting from least significant digit. In case if the result is an illegal code (greater than 9) or if there is a carry out of one then add 0110(6) and add the resulting carry to the next most significant.

For example:-

Add 679.6 from 536.8 using BCD addition.

Solution:-

6 7 9 . 6			0110	0111	1001	.	0110	(679.6 in BCD)
+ 5 3 6 . 8	=>+		0101	0011	0110	.	1000	(536.8 in BCD)
1 2 1 6 . 4			1011	1010	1111	.	1110	(All are illegal codes)
			+ 0110	+0110	+0110	.	+0110	(Add 0110 to each)
		0001	0010	0001	0110	.	0100	
		1	2	1	6	.	4	(corrected sum = 1216.4)

Result is **1216.4**

BCD SUBTRACTION:-

The BCD subtraction is performed by subtracting the digits of each 4 – bit group of the subtrahend from corresponding 4 – bit group of the minuend in the binary starting from the LSD. If there is no borrow from the next higher group[then no correction is required. If there is a borrow from the next group, then 6_{10} (0110) is subtracted from the difference term of this group.

For example:-

Subtract 147.8 from 206.7 using 8421 BCD code.

Solution:-

2 0 6 . 7		0010	0000	0110	.	0111	(206.7 in BCD)
- 1 4 7 . 8	=>-	0001	0100	0111	.	1000	(147.8 in BCD)
5 8 . 9		0000	1011	1110	.	1111	(Borrows are present)
			- 0110	-0110	.	- 0110	
			0101	1000	.	1001	
			5	8	.	9	(corrected difference = 58.9)

Result is **$(58.9)_{10}$**

EXCESS THREE(XS-3) CODE:-

The Excess-3 code, also called XS-3, is a non- weighted BCD code. This derives it name from the fact that each binary code word is the corresponding 8421 code word plus 0011(3). It is a sequential code. It is a self complementing code.

XS-3 ADDITION:-

In XS-3 addition, add the XS-3 numbers by adding the 4 bit groups in each column starting from the LSD. If there is no carry out from the addition of any of the 4 bit groups, subtract 0011 from the sum term of those groups. If there is a carry out, add 0011 to the sum term of those groups

For example:-

Add 37 and 28 using XS-3 code.

Solution:-

3 7   0110 1010   (37 in XS-3)

\+ 2 8   =>   + 0101 1011   (28 in XS-3)

6 5   1011 **1**1010   (Carry is generated)

\+ 1   (Propagate carry)

1100 0101   (Add 0110 to correct 0101 and

\- 0011 +0011   subtract 0011 to correct 1100)

1001 1000   (Corrected sum in XS-3 = 65_{10})

XS-3 SUBTRACTION:-

To subtract in XS-3 number by subtracting each 4-bit group of the subtrahend from the corresponding 4-bit group of the minuend starting from the LSD. If there is no borrow from the next 4-bit group. add 0011 to the difference term of such groups. If there is a borrow, subtract 0011 from the difference term.

For example :-

. **Subtract 175 from 267 using XS-3 code**.

Solution :-`

267   0101 1010 1010   (267 in XS-3)

-175   =>   - 0100 1010 1000   (175 in XS-3)

092   0000 1111 0010   (Correct 0010 and 0000 by adding 0011 and

+0011 -0011 +0011   correct 1111 by subtracting 0011)

0011 1100 0101   (Corrected difference in XS-3 = 92_{10})

ASCII CODE:-

The American Standard Code for Information Interchange (ASCII) pronounced as 'ASKEE' is widely used alphanumeric code. This is basically a 7 bit code. The number of different bit patterns that can be created with 7 bits is 27 = 128 , the ASCII can be used to encode both the uppercase and lowercase characters of the alphabet (52 symbols) and some special symbols in addition to the 10 decimal digits. It is used extensively for printers and terminals that interface with small computer systems. The table shown below shows the ASCII groups.

The ASCII code

LSBs	**MSBs**							
	000	001	010	011	100	101	110	111
0000	NUL	DEL	Space	0	@	P	P	
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s

<table>
<tr><td>0100</td><td>EOT</td><td>DC4</td><td>$</td><td>4</td><td>D</td><td>T</td><td>d</td><td>t</td></tr>
<tr><td>0101</td><td>ENQ</td><td>NAK</td><td>%</td><td>5</td><td>E</td><td>U</td><td>e</td><td>u</td></tr>
<tr><td>0110</td><td>ACK</td><td>SYN</td><td>&</td><td>6</td><td>F</td><td>V</td><td>f</td><td>v</td></tr>
<tr><td>0111</td><td>BEL</td><td>ETB</td><td>‘</td><td>7</td><td>G</td><td>W</td><td>g</td><td>w</td></tr>
<tr><td>1000</td><td>BS</td><td>CAN</td><td>(</td><td>8</td><td>H</td><td>X</td><td>h</td><td>x</td></tr>
<tr><td>1001</td><td>HT</td><td>EM</td><td>)</td><td>9</td><td>I</td><td>Y</td><td>i</td><td>y</td></tr>
<tr><td>1010</td><td>LF</td><td>SUB</td><td>*</td><td>:</td><td>J</td><td>Z</td><td>j</td><td>z</td></tr>
<tr><td>1011</td><td>VT</td><td>ESC</td><td>+</td><td>;</td><td>K</td><td>[</td><td>k</td><td>{</td></tr>
<tr><td>1100</td><td>FF</td><td>FS</td><td>,</td><td><</td><td>L</td><td>\</td><td>l</td><td>|</td></tr>
<tr><td>1101</td><td>CR</td><td>GS</td><td>-</td><td>=</td><td>M</td><td>]</td><td>m</td><td>}</td></tr>
<tr><td>1110</td><td>SO</td><td>RS</td><td>.</td><td>></td><td>N</td><td>^</td><td>n</td><td>~</td></tr>
<tr><td>1111</td><td>SI</td><td>US</td><td>/</td><td>?</td><td>O</td><td>_</td><td>o</td><td>DLE</td></tr>
</table>

EBCDIC CODE:-

The Extended Binary Coded Decimal Interchange Code (EBCDIC) pronounced as ‘eb – si- dik’ is an 8 bit alphanumeric code. Since 28 = 256 bit patterns can be formed with 8 bits. It is used by most large computers to communicate in alphanumeric data. The table shown below shows the EBCDIC code.

The EBCDIC code

<table>
<tr><th>LSD (Hex)</th><th colspan="16">MSD(Hex)</th></tr>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td></tr>
<tr><td>0</td><td>NUL</td><td>DLE</td><td>DS</td><td></td><td>SP</td><td>&</td><td></td><td></td><td></td><td></td><td></td><td></td><td>[</td><td>]</td><td>\</td><td>0</td></tr>
<tr><td>1</td><td>SOH</td><td>DC1</td><td>SOS</td><td></td><td></td><td></td><td>/</td><td></td><td>a</td><td>j</td><td>~</td><td></td><td>A</td><td>J</td><td></td><td>1</td></tr>
<tr><td>2</td><td>STX</td><td>DC2</td><td>FS</td><td>SYN</td><td></td><td></td><td></td><td></td><td>b</td><td>k</td><td>s</td><td></td><td>B</td><td>K</td><td>S</td><td>2</td></tr>
<tr><td>3</td><td>ETX</td><td>DC3</td><td></td><td></td><td></td><td></td><td></td><td></td><td>c</td><td>l</td><td>t</td><td></td><td>C</td><td>L</td><td>T</td><td>3</td></tr>
<tr><td>4</td><td>PF</td><td>RES</td><td>BYP</td><td>PN</td><td></td><td></td><td></td><td></td><td>d</td><td>m</td><td>u</td><td></td><td>D</td><td>M</td><td>U</td><td>4</td></tr>
<tr><td>5</td><td>HT</td><td>NL</td><td>LF</td><td>RS</td><td></td><td></td><td></td><td></td><td>e</td><td>n</td><td>v</td><td></td><td>E</td><td>N</td><td>V</td><td>5</td></tr>
<tr><td>6</td><td>LC</td><td>BS</td><td>EOB</td><td>YC</td><td></td><td></td><td></td><td></td><td>f</td><td>o</td><td>w</td><td></td><td>F</td><td>O</td><td>W</td><td>6</td></tr>
<tr><td>7</td><td>DEL</td><td>IL</td><td>PRE</td><td>EOT</td><td></td><td></td><td></td><td></td><td>g</td><td>p</td><td>x</td><td></td><td>G</td><td>P</td><td>X</td><td>7</td></tr>
<tr><td>8</td><td></td><td>CAN</td><td></td><td></td><td></td><td></td><td></td><td></td><td>h</td><td>q</td><td>y</td><td></td><td>H</td><td>Q</td><td>Y</td><td>8</td></tr>
<tr><td>9</td><td></td><td>EM</td><td></td><td></td><td></td><td></td><td></td><td></td><td>i</td><td>r</td><td>z</td><td></td><td>I</td><td>R</td><td>Z</td><td>9</td></tr>
<tr><td>A</td><td>SMM</td><td>CC</td><td>SM</td><td></td><td>Ø</td><td>!</td><td>|</td><td>:</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>B</td><td>VT</td><td></td><td></td><td></td><td>.</td><td>$</td><td>,</td><td>#</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>C</td><td>FF</td><td>IFS</td><td></td><td>DC4</td><td><</td><td>*</td><td>%</td><td>@</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>D</td><td>CR</td><td>IGS</td><td>ENQ</td><td>NAK</td><td>(</td><td>)</td><td>_</td><td>‘</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>E</td><td>SO</td><td>IRS</td><td>ACK</td><td></td><td>+</td><td>;</td><td>></td><td>=</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>F</td><td>SI</td><td>IUS</td><td>BEL</td><td>SUB</td><td>|</td><td>‘</td><td>?</td><td>‘</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

GRAY CODE:-

The gray code is a non-weighted code. It is not a BCD code. It is cyclic code because successive words in this differ in one bit position only i.e it is a unit distance code.

Gray code is used in instrumentation and data acquisition systems where linear or angular displacement is measured. They are also used in shaft encoders, I/O devices, A/D converters and other peripheral equipment.

BINARY- TO – GRAY CONVERSION:-

If an n-bit binary number is represented by B_n B_{n-1} - - - - - B_1 and its gray code equivalent by G_n G_{n-1} - - - - - G_1, where B_n and G_n are the MSBs , then gray code bits are obtained from the binary code as follows

$G_n = B_n$

$G_{n-1} = B_n \oplus B_{n-1}$

.

.

.

.

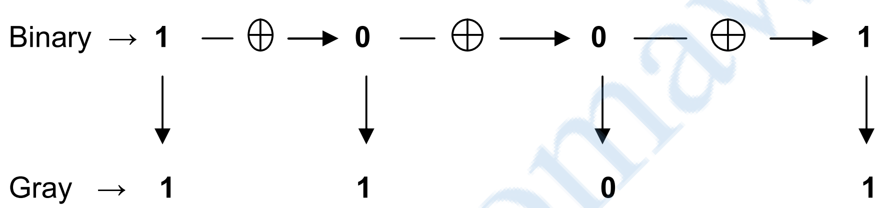
$G_1 = B_2 \oplus B_1$

Where the symbol $\oplus$ stands for Exclusive OR (X-OR)

For example :-

Convert the binary 1001 to the Gray code.

Solution :-`



The gray code is **1101**

GRAY- TO - BINARY CONVERSION:-

If an n-bit gray number is represented by G_n G_{n-1} ------- G_1 and its binary equivalent by B_n B_{n-1} - - - - - B_1, then binary bits are obtained from Gray bits as follows :

$B_n = G_n$

$B_{n-1} = B_n \oplus G_{n-1}$

.

.

.

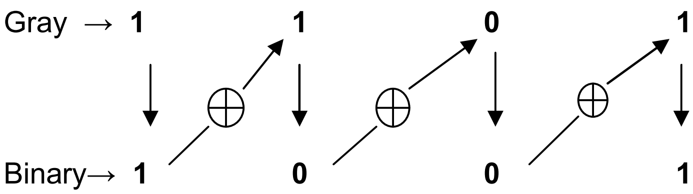
.

$B_1 = B_2 \oplus G_1$

For example :-

Convert the Gray code 1101 to the binary.

Solution :-



The binary code is **1001**