BINDASS

DIGITAL LOGIC DESIGN

UNIT - 1

NUMBER SYSTEM

★ Number System :

• Number system are the technique to represent numbers in the compute system architecture, every value that you are saving or getting into/from computer memory has a defined number system.

• computer architecture supports following number system:

Binary number system
octal number system
Decimal number system
Hexadecimal (Hex) number system

★ Types of Number System :-

#. BINARY Number System :- A Binary number system has only two digits that are 0 and 1. Every number (value) represents with 0 and 1 in this number system. The Base of binary number system is 2, because it has only two digits.

OCTAL NUMBER SYSTEM :- Octal number system has only eight digits from 0 to 7. Every number (value) represents with 0,1,2,3,4,5,6,7 in this number system.

The base of octal number system is 8, because it has Only 8 digits.

DECIMAL NUMBER System :-

Decimal number system has Only ten digits from 0 to 9. Every number (value) represents with 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 in this number System. The base of decimal number system is 10, because it has only 10 digits.

HEXADECIMAL NUMBER SYSTEM :-

A HEXAdecimal number system has sixteen alphanumeric Values from 0 to 9 and A to F. Every number (value) represents with 0, 1, 2, 3 4, 5, 6, 7, 8, 9, A, B, C, D, E, F in this Number System. The base of Hexadecimal number system is 16, because it has 16 alphanumeric Values. Here A is 10, B is 11, C is 12, D, is 13, E is 14, F is 15

Representation of Number System.

Decimal	Binary	Hexadecimal
0	0	0
1	1	1
2	01	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B

Decimal	Binary	Hexadecimal
12	1100	~~A~~ C
13	1101	D
14	1110	E
15	1111	F

⇒ Many number system are in use in digital technology. The most common are :

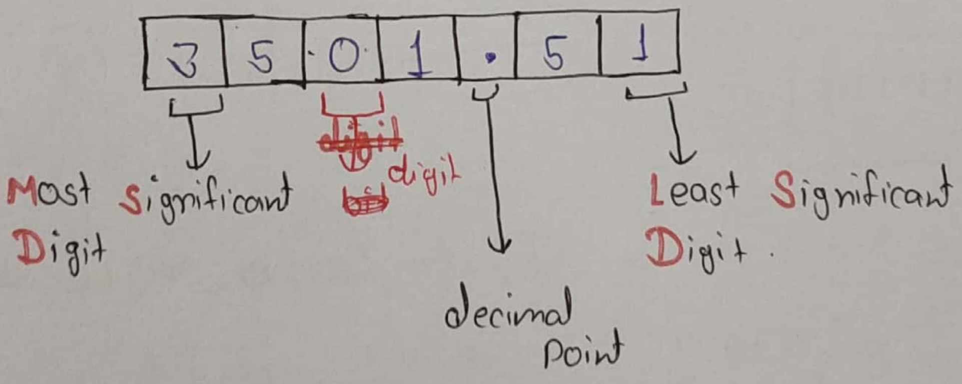
Decimal	(Base 10)
Binary	(Base 2)
octal	(Base 8)
Hexadecimal	(Base 16)

The decimal system is the number system that we use everyday.

✰ Decimal Number System :-

• The decimal system is composed of 10 numerals or symbols. These 10 symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 ; using these symbols as digits of a number, we can express any quantity.

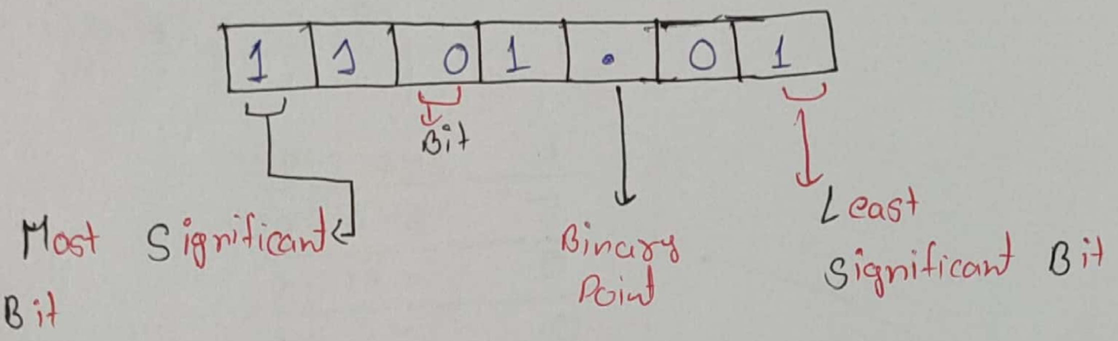
• Eg :- 3501.51



✰ Binary Number System :-

• The binary system is composed of 2 numerals or symbols 0 and 1 ; using these symbols as digits of a number, we can express any quantity.

• Eg:- 1101.01



★ CONVERSION:-

Decimal to Binary

Eg:- convert $(125)_{10} \rightarrow (?)_2$

2	125	1
2	62	0
2	31	1
2	15	1
2	7	1
2	3	1
	1	

Reminder

$\boxed{(1111101)_2}$

Eg:- convert $(125.0625)_{10} \rightarrow (?)_2$.

Eg:-

2	125	1
2	62	0
2	31	1
2	15	1
2	7	1
2	3	1
	1	

$0.0625 * 2 = 0.1250$
$0.1250 * 2 = 0.2500$
$0.2500 * 2 = 0.5000$
$0.5000 * 2 = 1.0000$

$(125.0625)_{10} = (1111101.0001)_2$

Decimal to octal :-

Eg:- $(12345)_{10} \rightarrow (?)_8$.

8	12345	1
8	1543	7
8	192	0
8	24	0
	3	

$\boxed{(30071)_8}$

Eg:- $(468.225)_{10} \rightarrow (?)_8$

8	468	4
8	58	2
	7	

$0.225 * 8 = 1.800$
$0.800 * 8 = 6.400$
$0.400 * 8 = 3.200$
$0.200 * 8 = 1.600$
$0.600 * 8 = 4.800$

$(468.225)_{10} \rightarrow (724.16314)_8$

Decimal to Hexadecimal :-

Eg:- $(2748.225)_{10} \rightarrow (?)_{16}$

16	2748	12
16	171	11
	10	

$0.225 * 16 = 3.600$
$0.600 * 16 = 9.600$

$(2748.225)_{10} \longrightarrow (ABC.39)_{16}$

Binary to Decimal.

Eg: $(11011001)_2 \rightarrow (?)_{10}$

$= 1*2^7 + 1*2^6 + 0*2^5 + 1*2^4 + 1*2^3 + 0*2^2 + 0*2^1 + 1*2^0$

$= 128 + 64 + 0 + 16 + 8 + 0 + 0 + 1$

$= 217$

ie. $(217)_{10}$.

Eg:- $(11011001.101)_2 \rightarrow (?)_{10}$

$= 1*2^7 + 1*2^6 + 0*2^5 + 1*2^4 + 1*2^3 + 0*2^2 + 0*2^1 +$
$1*2^0 + 1*2^{-1} + 0*2^{-2} + 1*2^{-3}$

$= 128 + 64 + 0 + 16 + 8 + 0 + 0 + 1 + 0.5 + 0 + 0.125$

$= 217.625$

ie. $(217.625)_{10}$

Octal to Decimal

Eg:- $(345.42)_8 \rightarrow (?)_{10}$

$= 3 * 8^2 + 4 * 8^1 + 5 * 8^0 + 4 * 8^{-1} + 2 * 8^{-2}$

$= 192 + 32 + 5 + 0.5 + 0.03125$

$= 229.53125$

ie. $(229.53125)_{10}$

Hexadecimal to Decimal.

Ex. $(1ACF)_{16} \rightarrow (?)_{10}$

$= 1 * 16^3 + 10 * 16^2 + 12 * 16^1 + 15 * 16^0$

$= 4096 + 2560 + 192 + 15$

$= 6863$

Ex. $(1ACF.BD)_{16} \rightarrow (?)_{10}$

$= 1 * 16^3 + 10 * 16^2 + 15 * 16^0 + 11 * 16^{-1} + 13 * 16^{-2}$

$= 4096 + 2560 + 192 + 15 + 0.6875 + 0.0507$

$= 6863.7382$

OCTAL to Binary:-

Eg:- $(4623)_8 \rightarrow (?)_2$.

4	6	2	3
100	110	010	011

Answer $(100110010011)_2$

Ex:- $(623.75)_8 \rightarrow (?)_2$.

6	2	3	.	7	5
110	010	011	.	111	101

Ans. $(110010011.111101)_2$

Hexadecimal to Binary.

Ex:- 1 $(4ACD)_{16} \rightarrow (?)_2$.

4	A	C	D
0100	1010	1100	1101

Ans. $(0100101011001101)_2$

Binary to octal :-

Ex. $(10110010011)_2 \rightarrow (?)_8$

010	110	010	011
2	6	2	3

Ans $(2623)_8$

Binary to Hexadecimal :-

Ex:- $(1010101100100113.1110011)_2 \rightarrow (?)_{16}$

0010	1010	1100	1001	1110	0011.
2	10	12	9	13	3

1110	0011
14	3

Ans $(2AC9D3.E3)_{16}$.

Binary Addition:-

Case	A + B	Sum	Carry
1.	0 + 0	0	0
2.	0 + 1	1	0
3	1 + 0	1	0
4.	1 + 1	0	1

In fourth case, a binary ~~is~~ addition is creating a sum of (1+1 = 10) i.e. 0 is written in the given column and a carry of 1 over to the next column.

Ex:- 0011010 + 001100 = 00100110

```
      1 1            Carry
  0 0 1 1 0 1 0  = 26₁₀
+   0 0 1 1 0 0  = 12₁₀
  -------------
  0 1 0 0 1 1 0  = 38₁₀
```

Ex. 01101101 + 0011111 = ?

```
  1 1 1 1 1 1 1  →  Carry.
  0 1 1 0 1 1 0 1
+   0 0 1 1 1 1 1
-----------------
1 0 0 0 1 1 0 0
```

Binary Substraction:-

A	B	A - B	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Scanned with CamScanner

Ex.

$$\begin{array}{rccccc} & & & 0 & 10 & \\ & 1 & 1 & \cancel{1} & \cancel{0} & 1 \\ - & 0 & 1 & 0 & 1 & 0 \\ \hline & 1 & 0 & 0 & 1 & 1 \end{array} \quad \text{or.} \quad \begin{array}{ccccc} & & 0 & 2 & \\ 1 & 1 & \cancel{1} & \cancel{0} & 1 \\ 0 & 1 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 1 & 1 \end{array}$$

Binary Multiplication:

A	B	$A * B$	carry or borrow
0	0	$0 * 0 = 0$	
0	1	$0 * 1 = 0$	
1	0	$1 * 0 = 0$	
1	1	$1 * 1 = 1$	no carry/borrow bits

Ex:-

$$\begin{array}{rcccccccccl} & & & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & = (41)_{10} \\ \times & & & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & = (6)_{10} \\ \hline & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ & & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & & \\ & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & & & \\ \hline & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & = (246)_{10} \end{array}$$

Ex:-

$$\begin{array}{ccccccccc} & & & 1 & 0 & 1 & 0 & 1 & 1 \\ & & & 0 & 0 & 0 & 1 & 0 & 1 \\ \hline & & & 1 & 0 & 1 & 0 & 1 & 1 \\ & & 0 & 0 & 0 & 0 & 0 & 0 & \\ 1 & 0 & 1 & 0 & 1 & 1 & & & \\ \hline 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & \end{array}$$

Scanned with CamScanner

Binary Division :-

10000111 ÷ 00000101 = ?

```
          1 1 0 1 1
101 ) 1 0 0 0 0 1 1 1
     - 1 0 1
       1 1 0
     - 1 0 1
       0 0 1 0 0
          - 0
           1 1 1
         - 1 0 1
           0 1 0 1
             1 0 1
                 0.
```

Ex:- 101010/000110 = ?.

```
            1 1 1
110 ) 1 0 1 0 1 0
     - 1 1 0
       1 0 0 1
       - 1 1 0
         0 1 1 0
         - 1 1 0
               0.
```

Compliment of Number

1's compliment :-

The 1's complement of a ~~an~~ number is found by changing all 1's to 0's and all 0's to 1's. This is called as taking complement or 1's complement. Example of 1's complement is as follows.

Given number →	1	0	1	0	1
	↓	↓	↓	↓	↓
1's complement →	0	1	0	1	0

2's complement :-

The 2's complement of binary number is obtained by adding 1 to the least significant Bit (LSB) of 1's complement of the number.

2's complement = 1's complement + 1

Example of 2's complement is as follows.

given number →	1	0	1	0	1
	↓	↓	↓	↓	↓
1's complement →	0	1	0	1	0
Add 1 +					1
	0	1	0	1	~~0~~1

Substraction Using 1's Complement :

Ex. 1. Using 1's complement get 1010100 - 1000011

Let's suppose X = 1010100 , Y = 1000011

X =	1 0 1 0 1 0 0
1's complement of Y =	0 1 1 1 1 0 0
SUM	1 0 0 1 0 0 0 0
Carry .	+ 1.
X - Y =	0 0 1 0 0 0 1

Substraction Using 2's complement

Ex. 1. Using 2's complement get 10110 - 11010

Let suppose X = 10110 , Y = 11010

X =	1 0 1 1 0
2's complement of Y =	0 0 1 1 0
Sum	1 1 1 0 0

Now we observed that result is negative so we have to take 2's complement of the result (11100)

ie = 00100

Ex. 2. Using 2's complement get 11010 - 10110

Let suppose $x = 11010$, $y = 10110$

$$
\begin{array}{rr}
x = & 1\,1\,0\,1\,0 \\
\text{2's complement of } y = & 0\,1\,0\,1\,0 \\
\hline
\text{Sum} & 1\,0\,0\,1\,0\,0
\end{array}
$$

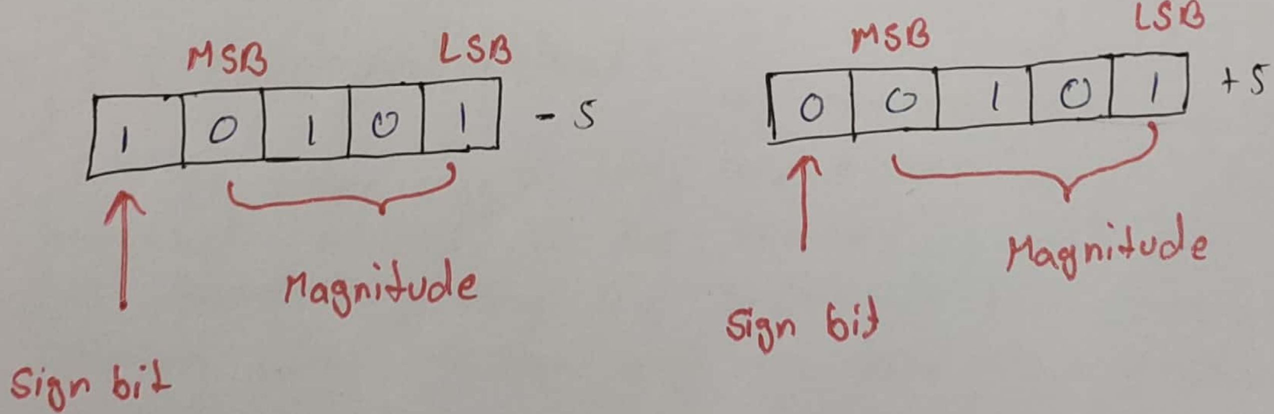
Drop the Carry

00100 Ans

☆ Negative Number Representation

1. Signed Megnitude Method: In this method, number is divided into two parts: sign bit and magnitude. If the number is positive then sign bit will be 0 and if number is negative then sign bit will be 1. Magnitude is represented with the binary form of the number to be represented.

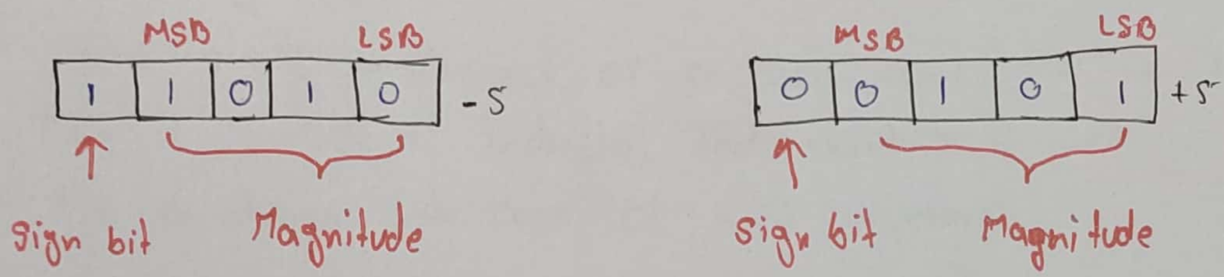
Ex. Let we are using 5 bits register. The representation of -5 to +5 will be.



2. 1's Complement Method:

+ve numbers are represented in the same way as they are represented in sign magnitude method. If the number is -ve then it's represented using 1's complement. first represent the number with +ve sign and then take 1's complement of that number.

Ex:- let we are using 5 bits register. The representation of -5 and +5 will be as follows:-



+5 is represented as it is represented in sign magnitude.
-5 is represented using the following steps:

- +5 = 0 0 1 0 1
- Take 1's complement of 0 0101 and that is 1 1010.

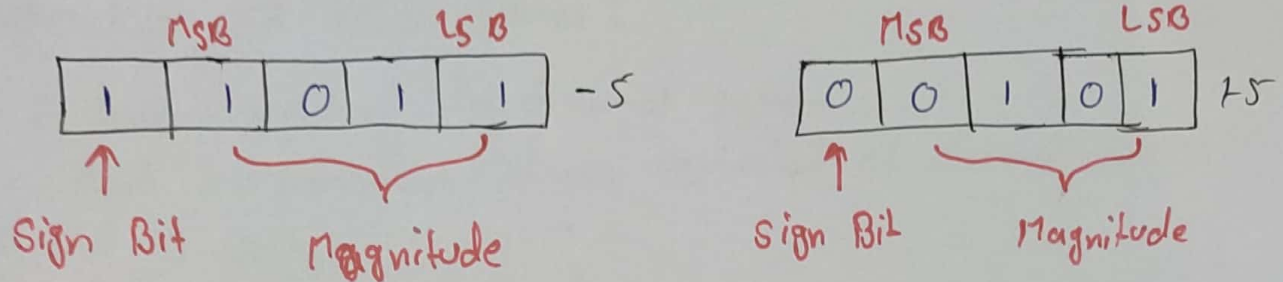
MSB is 1 which indicates that number is -ve.

MSB is always 1 in case of -ve numbers.

3. 2's Complement Method:

+ve numbers are represented in the same way as they are represented in sign magnitude method. If the number is -ve then it is represented using 2's complement. First represent the number with +ve sign and then take 2's complement of that number.

Ex:- Let we are using 5 bits register. The representation of -5 & +5 will be as follows:-



+5 is represented as it is represented in sign magnitude method. -5 is represented using the fallowing steps:

- +5 = 0 0101
- Take 2's complement of 0 0101 and that is 1 1011

MSB is 1 which indicates that number is -ve

MSB is always 1 in case of -ve numbers.

- CODES :

★ Binary Coded Decimal Number (BCD): In this code each decimal digit is represented by a 4-bit binary number. BCD is a way to express each of the decimal digits with a binary ~~number~~ code. In BCD, with 4-bits we can represed 16 numbers i.e. (0000 to 1111). But in BCD code only 1st ten of these are used i.e. (0000 to 1001). Remaining six code combination ie (1010 to 1111) are invalid in BCD.

Decimal	0	1	2	3	4	5	6	7
BCD	0000	0001	0010	0011	0100	0101	0110	0111

8	9
1000	1001

Advantage of BCD Codes:

- It is very similar to decimal system.
- We need to remember binary equivalent of decimal numbers 0 to 9 only.

Disadvantage of BCD codes:

- Addition & subtraction of BCD have different rules.
- BCD arithmetic is little more complicated.
- BCD needs more number of bits than binary to represent the decimal number. So BCD is less efficient than binary.

Ex:- $(834)_{10} \longrightarrow (?)_{BCD}$

$$\begin{array}{ccc} (8 & 3 & 4)_{10} \\ \downarrow & \downarrow & \downarrow \\ (1000 & 0011 & 0100)_{BCD} \end{array}$$

Answer $= (100000110100)_{BCD}$

☆ Gray Code :-

It is non-weighted code and it is non arithmatic codes. That means there are no specific weights assigned to the bit position. It is very special feature that, only one bit will change each time the decimal number is incremented.

Application of Gray code :-

- Gray code is popularly used in the shaft position encoders.
- A shaft position encoders produces a code word which represents the angular position of the shaft.

Ex:- $(27)_{10} \longrightarrow (?)_G$ or Gray

$(27)_{10} \longrightarrow (11011)_2$

xor XOR XOR XOR

1 1 0 1 1

$(1\ 0\ 1\ 1\ 0)_{Gray}$

Ans $(10110)_{Gray}$.

A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0.

Ex:- $(110010)_G \longrightarrow (?)_2$.

1 1 0 0 1 0

as it is XOR XOR XOR XOR XOR

1 0 0 0 1 1

Ex:- $(111000101)_G \longrightarrow (?)_2$

1 1 1 0 0 0 1 0 1

as it is XOR XOR XOR XOR XOR XOR XOR XOR

1 0 1 1 1 1 0 0 1

Excess-3 Code :-

It is also called as XS-3 Code.

It is also called as XS-3 code. It is non-weighted code used to express decimal numbers. The Excess-3 code word are derived from the 8421 BCD code word $(0011)_2$ or $(3)_{10}$ to each number in 8421. The XS-3 Codes are Obtained as follows:

Decimal Number $\longrightarrow$ 8421 BCD $\xrightarrow[0011]{Add}$ Excess-3

Example

Decimal	BCD 8421	XS-3 BCD + 0011
0	0000	0011
1	0001	0100
2	0012	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

Ex:- $(1532)_{10} \rightarrow (?)_{XS-3}$

$(1 \quad 5 \quad 3 \quad 2)_{10}$

	1	5	3	2
	0001	0101	0011	0010
+	0011	0011	0011	0011
	0100	1000	0110	0101

$(0100100001100101)_{XS-3}$ Ans.