Module 1: Number Systems

Binary Number System:

- Number system with the base as 2.
- Two symbols/bits 0 and 1.

Octal Number System:

- Number system with the base as 8.
- Eight symbols/bits 0,1,2,3,4,5,6,7.

Decimal Number System:

- Number system with the base as 10.
- Ten symbols/bits 0,1,2,3,4,5,6,7,8,9.

Hexadecimal Number System:

- Number system with the base as 16 or H.
- Sixteen symbols/bits 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

1 bit representation of binary nos: 2^1 = total 2 nos.

0 = decimal equivalent 0

1 = decimal equivalent 1

2 bit representation of binary nos : 2 ^2 = total 4 nos.

00 = decimal equivalent 0

01 = 1

10 = 2

11 = 3

3 bit representation of binary nos : 2 ^3 = total 8 nos = 0 to 7 (421 combination)

421

……

000 = 0

001 =1

010 = 2

011 = 3

100 =4

101 = 5

110 =6

111 =7

4 bit representation of binary nos : 2 ^4 = total 16 nos = 0 to 15 (8421 combination)

8421

……

0000 = 0

0001 =1

0010 = 2

0011 =3

0100 =4

0101 = 5

0110 = 6

0111 = 7

1000 = 8

1001 = 9

1010 = 10

1011 =11

1100 = 12

1101 = 13

1110 =14

1111=15

Representation of a decimal number:

A decimal number $(1234.567)_{10}$ can be represented as:

$$(1234.567)_{10} = 1*10^3+2*10^2+3*10^1+4*10^0+5*10^{-1}+6*10^{-2}+7*10^{-3}$$

Decimal to other base conversion:

Any decimal number can be converted into its equivalent other base (2,8,16) number.

For integer part: The conversion is obtained by continuous division by target number system base (2,8,16) and keeping track of remainders.

For fractional part: The conversion is obtained by continuous multiplication by target number system base (2,8,16) and keeping track of integers generated.

Other base to Decimal Conversion:

Any base number can be converted into its equivalent decimal number using the weights assigned to bit positions as given in example below:

Ex. 1: Convert $(1326.02)_8$ into its decimal equivalent.

$$\Rightarrow (1326.02)_8 = 1*8^3 + 3*8^2 + 2*8^1 + 6*8^0 + 0*8^{-1} + 2*8^{-2}$$

$$= 512 + 192 + 16 + 6 + 0.03125 = (726.03125)_{10}$$

Binary to other base conversion:

Binary to Decimal: Same as above mentioned topic.

Ex. 1: Convert $(1001101.011)_2$ into its decimal equivalent.

$$\Rightarrow (1001101.011)_2 = 1*2^6 + 1*2^3 + 1*2^2 + 1*2^0 + 1*2^{-2} + 1*2^{-3}$$

$$= 64 + 8 + 4 + 1 + 0.25 + 0.125 = (77.375)_{10}$$

Binary to Octal: Group binary bits into group of 3 from LSB.

Ex. 2: Convert $(1001101.011)_2$ into its octal equivalent.

$\Rightarrow (1001101.011)_2$

So, it equals $(115.3)_8$

Binary Group	1	001	101	.	011
Octal Equivalent	1	1	5	.	3

Binary to Hexadecimal: Group binary bits into group of 4 from LSB.

Ex. 3: Convert $(1001101.011)_2$ into its hexadecimal equivalent.

$\Rightarrow (1001101.011)_2$

So, it equals $(4D.6)_{16}$

Binary Group	0100	1101	.	0110
Hex Equivalent	4	D(13)	.	6

Other base to Binary Conversion:

Decimal to Binary: For **Integer part,** continuously divide decimal number by 2, and record remainder. For **fractional part,** continuously multiply by 2, keep track of integer generated.

Ex. 1: Convert $(13.375)_{10}$ into its binary equivalent.

=> Integer part:

	Quotient	Remainder	
13 / 2	6	1	↑
6 / 2	3	0	↑
3 / 2	1	1	↑
1 / 2	0	1	↑

So $(13)_{10} = (1101)_2$

Fractional part:

	Multiplication	Multiplication fraction	Integer part	
0.375 * 2	0.750	0.75	0	↓
0.75 * 2	1.50	0.50	1	↓
0.50 * 2	1.00	0	1	↓

So $(0.375)_{10} = (0.011)_2$

Finally, $(13.375)_{10} = (1101.011)_2$

Octal to Binary: Simply write 3 bit binary equivalent of each and every octal bit.

Ex. 2: Convert $(475.32)_8$ into its binary equivalent.

=> $(475.32)_8$

It equals

Octal Bit	4	7	5	.	3	2
Binary Equivalent	100	111	101	.	011	010

$(100\ 111\ 101\ .\ 011\ 010)_2$

Hexadecimal to Binary: Simply write 4 bit binary equivalent of each and every hexadecimal bit.

Ex. 3: Convert $(D9.B)_{16}$ into its binary equivalent.

=> $(D9.B)_{16}$

It equals

Octal Bit	D	9	.	B
Binary Equivalent	1101	1001	.	1011

$(1101\ 1001\ .\ 1011)_2$

Octal to other base conversion:

Octal to binary and Octal to decimal already covered above.

Octal to hexadecimal: First convert octal to its equivalent binary, and then convert that binary number to its equivalent hexadecimal taking group of 4 from LSB.

Ex. 1: Convert $(6354)_8$ to its equivalent hexadecimal number.

Octal	6	3	5	4
Binary	110	011	101	100

Binary 4 group	1100	1110	1100
Hex equivalent	C(12)	E(14)	C(12)

So, $(6354)_8 = (CEC)_{16}$

Other base to Octal Conversion:

Binary to octal and decimal to octal already discussed above.

Hexadecimal to octal: First convert hexadecimal to its equivalent binary, and then convert that binary number to its octal equivalent by taking group of 3 from LSB.

Ex. 2: Convert $(CEC)_{16}$ to its equivalent octal number.

=>

Hexadecimal	C	E	C
Binary	1100	1110	1100

Binary 3 group	110	011	101	100
Octal equivalent	6	3	5	4

So, $(CEC)_{16} = (6354)_8$

Hexadecumal to other base conversion:

Hexadecimal to octal and hexadecimal to binary already discussed above.

Hexadecimal to Decimal: Multiply each bit of hexadecimal number by weights assigned by bit positions.

Ex.1: Convert $(18F.C)_{16}$ to its decimal equivalent.

=> F means 15 and C means 12.

So, $(18F.C)_{16} = 1*16^2 + 8*16^1 + F*16^0 + C*16^{-1} = 256 + 128 + 15 + 0.75 = (399.75)_{10}$

Other base to Hexadecimal Conversion:

Binary to hexadecimal and octal to hexadecimal already discussed above.

Decimal to hexadecimal:

For integer part: The conversion is obtained by continuous division by 16 and keeping track of remainders.

For fractional part: The conversion is obtained by continuous multiplication by 16 and keeping track of integers generated.

Ex.2: Convert $(726.03125)_{10}$ to its equivalent hexadecimal number.

=> Integer part:

	Quotient	**Remainder**	
726 / 16	45	6	↑
45 / 16	2	D(13)	↑
2 / 16	0	2	↑

So $(726)_{10} = (2D6)_8$

Fractional part:

	Multiplication	**Multiplication fraction**	**Integer part**	
0.03125 * 16	0.5	0.5	0	↓
0.5 * 16	8.0	0	8	↓

So $(0.03125)_{10} = (0.08)_{16}$

Finally, $(726.03125)_{10} = (2D6.08)_{16}$

1’s Complement of a number:

It is obtained by inverting each and every bit in given number i.e. invert 0 to 1 and vice a versa.

Eg. 1’s complement of (1010) is (0101)

2’s Complement of a number:

It is obtained by adding 1 to 1’s complement of a number.

Eg. 2’s complement of (0101) is 1010+1=1011

1’s Complement subtraction Procedure

Given A – B to perform, first find 1’s complement of B. Then add A with 1’s complement of B. Observe result. If carry is generated result is positive. Add this carry to result to get final answer. If carry is not generated result is negative and in 1’s complement form. We have to get 1’s complement of result to get final answer.

2’s Complement addition Procedure

Given A – B to perform, first find 2’s complement of B. Then add A with 2’s complement of B. Observe result. If carry is generated result is positive. Discard carry, result obtained is final answer. If carry is not generated result is negative and in 2’s complement form. We have to get 2’s complement of result to get final answer.

7's complement of a number:

7's complement of a number is obtained by subtracting each and every bit of that number by 7.

Eg. 7's complement of (344) is 777 – 344 = 433

8's complement of a number:

7's complement of a number is obtained by adding 1 to 7's complement of given number.

Eg. 8's complement of (344) is 433 + 1 = 434

7's Complement subtraction Procedure

Given A – B to perform, first find 7's complement of B. Then add A with 7's complement of B. Observe result. If carry is generated result is positive. Add this carry to result to get final answer. If carry is not generated result is negative and in 7's complement form. We have to get 7's complement of result to get final answer.

8's Complement subtraction Procedure

Given A – B to perform, first find 8's complement of B. Then add A with 8's complement of B. Observe result. If carry is generated result is positive. Discard carry, result obtained is final answer. If carry is not generated result is negative and in 8's complement form. We have to get 8's complement of result to get final answer.

15's complement of a number:

15's complement of a number is obtained by subtracting each and every bit of that number by 15.

Eg. 15's complement of (879) is 151515 – 879 = 786

16's complement of a number:

16's complement of a number is obtained by adding 1 to 16's complement of given number.

Eg. 16's complement of (879) is 786 + 1 = 787

15's Complement subtraction Procedure

Given A – B to perform, first find 15's complement of B. Then add A with 15's complement of B. Observe result. If carry is generated result is positive. Add this carry to result to get final answer. If carry is not generated result is negative and in 15's complement form. We have to get 15's complement of result to get final answer.

16's Complement subtraction Procedure

Given A – B to perform, first find 16's complement of B. Then add A with 16's complement of B. Observe result. If carry is generated result is positive. Discard carry, result obtained is final answer. If carry is not generated result is negative and in 16's complement form. We have to get 16's complement of result to get final answer.

Codes:

Binary Code: A code having any two values 0 and 1. Very useful for computer processing.

BCD Code: BCD means binary coded decimal. This code converts each bit of given decimal number into binary. Eg. If you want to encode number 37 into BCD it will be: 0011 0111

Excess 3 Code: It is an extension of BCD code. In excess 3 we add binary 3 (11) to each and every BCD digit. If given BCD is (0011 0111) its excess 3 code will be (0110 1010).

Gray Code: Its an important code. It's a reflected code in which each number differs its previous and next bit by 1 bit only. It is used in K-maps.

ASCII Code: Its American Standard Code for Information Interchange. It is code for representing 128 characters as number from 0 to 127.

Hamming Code: It is used for error detection as well as correction. 7 bit Hamming code can encode 4 bit data with 3 bits for parity. Parity can be even or odd. Parity bit P1 check bit numbers 1, 3, 5, 7, P2 check bit numbers 2, 3, 6, 7, while P4 checks bit numbers 4, 5, 6, 7.

Generally parity bits are placed at powers of 2. That is $2^0(1)$, $2^1(2)$, $2^2(4)$, $2^3(8)$ …

Hamming code can detect at most 1 bit in error.