

Course Contents
Unit-05:Data Representation (6 Hrs.)
<ul style="list-style-type: none">• Introduction; Number System;• Conversion from Decimal to Binary, Octal, Hexadecimal;• Conversion of Binary, Octal, Hexadecimal to Decimal;• Conversion of Binary to Octal, Hexadecimal;• Conversion of Octal, Hexadecimal to Binary;• Binary Arithmetic;• Signed and Unsigned Numbers;• Binary Data Representation;• Binary Coding Schemes;• Logic Gates

Course Contents
Unit-05:Data Representation
<ul style="list-style-type: none">> Introduction; Number System;> Conversion from Decimal To Binary, Octal, Hexadecimal;> Conversion of Binary, Octal, Hexadecimal To Decimal;> Conversion of Binary To Octal, Hexadecimal;> Conversion of Octal, Hexadecimal To Binary;> Binary Arithmetic;> Signed and Unsigned Numbers;> Binary Data Representation;> Binary Coding Schemes;> Logic Gates

Course Contents
Introduction:
<ul style="list-style-type: none">> We use computer to process data and get desire output.> The data input can be in the form of alphabets, digits, symbols, audio, video etc but computer can only understand 0 and 1 so data must be represented in the computer in 0's and 1's.
We will study data representation in this chapter.
<ul style="list-style-type: none">> Numeric data(0,1,2..9)> Alphabetic data(A,B,C,.....Z)> Alphanumeric data – Combination of any symbols (A,BC,...Z), (0,1,2..9), or special characters (+,-, blank etc)

Course Contents
Number System:
<ul style="list-style-type: none">> There are infinite ways to represent a number.> The four commonly associated with modern computers and digital electronics are: Decimal, Binary, Octal, and Hexadecimal.
<ul style="list-style-type: none">> Decimal (base 10) is the way most human beings represent numbers. Decimal is sometimes abbreviated as dec.> Decimal counting goes:<ul style="list-style-type: none">> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, and so on.

Course Contents
Number System:
<ul style="list-style-type: none">> Binary (base 2) is the natural way most digital circuits represent and manipulate numbers.
<ul style="list-style-type: none">> Binary is sometimes abbreviated as bin.> Binary counting goes: 0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111, 10000, 10001, and so on.

Course Contents
Number System:
<ul style="list-style-type: none">> Octal (base 8) was previously a popular choice for representing digital circuit numbers in a form that is more compact than binary.
<ul style="list-style-type: none">> Octal is sometimes abbreviated as oct.> Octal counting goes: 0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17, 20, 21, and so on.

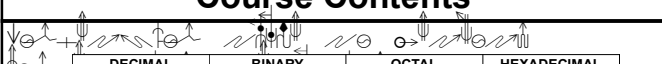
Course Contents
<p>Number System:</p> <ul style="list-style-type: none"> ➤ Hexadecimal (base 16) is currently the most popular choice for representing digital circuit numbers in a form that is more compact than binary. ➤ Hexadecimal is sometimes abbreviated as hex. • Hexadecimal counting goes: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, 10, 11, and so on.

Course Contents
<p>Number System:</p> <ul style="list-style-type: none"> ➤ Base number: The base of the number decides the valid digits that are used to make a number. Decimal number has a base 10, Octal is 8 and Hexadecimal is 16 ➤ Face Value : The face value of a digit is the digit located at that position. E.g. in decimal 52, face value at position 0 is 2 and face value at position 1 is 5 ➤ Position Value: The position value of a digit is $(\text{base}^{\text{position}})$. E.g. in decimal number 52, the position value of digit 2 is 10^0 and position value of digit 5 is 10^1.

Course Contents
<p>Number System:</p> <p>We are concerned with 4 kinds of number.</p> <ol style="list-style-type: none"> 1. Decimal Number System – Base 10 2. Binary Number System – Base 2 3. Octal Number System – Base 8 4. Hexadecimal Number System – Base 16 <ul style="list-style-type: none"> ➤ Number given as input to computer and output from computer are generally in decimal number system and are most understood by humans. ➤ But computer understand binary number system (0's and 1's). Binary data also represented internally as in Octal and Hexa due to their ease of use.

Course Contents
<p>Number System:</p> <ul style="list-style-type: none"> • All four number systems are equally capable of representing any number. • A number can be perfectly converted between the various number systems without any loss of numeric value. • Since the job of electrical and software engineers is to work with digital circuits, engineers require number systems that can best transfer information between the human world and the digital circuit world.

Course Contents
<p>Number System:</p> <ul style="list-style-type: none"> ➤ An octal number (base 8) can be up to 1/3 the length of a binary number (base 2). 8 is a whole power of 2 ($2^3=8$). That means three binary digits convert neatly into one octal digit. ➤ A hexadecimal number (base 16) can be up to 1/4 the length of a binary number. 16 is a whole power of 2 ($2^4=16$). That means four binary digits convert neatly into one hexadecimal digit. ➤ Unfortunately, decimal (base 10) is not a whole power of 2. So, it is not possible to simply chunk groups of binary digits to convert the raw state of a digital circuit into the human-centric format.

Course Contents																																																																				
 <table border="1"> <thead> <tr> <th>DECIMAL</th> <th>BINARY</th> <th>OCTAL</th> <th>HEXADECIMAL</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>2</td><td>10</td><td>2</td><td>2</td></tr> <tr><td>3</td><td>11</td><td>3</td><td>3</td></tr> <tr><td>4</td><td>100</td><td>4</td><td>4</td></tr> <tr><td>5</td><td>101</td><td>5</td><td>5</td></tr> <tr><td>6</td><td>110</td><td>6</td><td>6</td></tr> <tr><td>7</td><td>111</td><td>7</td><td>7</td></tr> <tr><td>8</td><td>1000</td><td>10</td><td>8</td></tr> <tr><td>9</td><td>1001</td><td>11</td><td>9</td></tr> <tr><td>10</td><td>1010</td><td>12</td><td>A</td></tr> <tr><td>11</td><td>1011</td><td>13</td><td>B</td></tr> <tr><td>12</td><td>1100</td><td>14</td><td>C</td></tr> <tr><td>13</td><td>1101</td><td>15</td><td>D</td></tr> <tr><td>14</td><td>1110</td><td>16</td><td>E</td></tr> <tr><td>15</td><td>1111</td><td>17</td><td>F</td></tr> </tbody> </table>	DECIMAL	BINARY	OCTAL	HEXADECIMAL	0	0	0	0	1	1	1	1	2	10	2	2	3	11	3	3	4	100	4	4	5	101	5	5	6	110	6	6	7	111	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
DECIMAL	BINARY	OCTAL	HEXADECIMAL																																																																	
0	0	0	0																																																																	
1	1	1	1																																																																	
2	10	2	2																																																																	
3	11	3	3																																																																	
4	100	4	4																																																																	
5	101	5	5																																																																	
6	110	6	6																																																																	
7	111	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																																																																	

Course Contents				
Conversion from Decimal to Binary: Example of 1792				
Decimal Number	Operation	Quotient	Remainder	Binary Result
1792	÷ 2 =	896	0	0
896	÷ 2 =	448	0	00
448	÷ 2 =	224	0	000
224	÷ 2 =	112	0	0000
112	÷ 2 =	56	0	00000
56	÷ 2 =	28	0	000000
28	÷ 2 =	14	0	0000000
14	÷ 2 =	7	0	00000000
7	÷ 2 =	3	1	100000000
3	÷ 2 =	1	1	1100000000
1	÷ 2 =	0	1	11100000000
0	done.			

Course Contents				
Conversion from Decimal to Octal : Example to convert 1792 decimal to octal:				
Decimal Number	Operation	Quotient	Remainder	Octal Result
1792	÷ 8 =	224	0	0
224	÷ 8 =	28	0	00
28	÷ 8 =	3	4	400
3	÷ 8 =	0	3	3400
0	done.			

Course Contents				
Conversion from Decimal to Hexadecimal:				
Decimal Number	Operation	Quotient	Remainder	Hexadecimal Result
1792	÷ 16 =	112	0	0
112	÷ 16 =	7	0	00
7	÷ 16 =	0	7	700
0	done.			

Course Contents				
Conversion from Decimal to Hexadecimal:				
Decimal Number	Operation	Quotient	Remainder	Hexadecimal Result
48879	÷ 16 =	3054	15	F
3054	÷ 16 =	190	14	EF
190	÷ 16 =	11	14	EEF
11	÷ 16 =	0	11	BEEF
0	done.			

Course Contents
<p>Conversion of Binary to Decimal:</p> <p>Convert binary number $1010_{(base\ 2)}$ into decimal form</p> $1010_{(base\ 2)} = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 8 + 0 + 2 + 0 = 10$ <p>So, the required decimal number is $110101_{(base\ 2)} = 53_{(base\ 10)}$</p> <p>Alternatively, $(110101)_2 = (53)_{10}$</p> <p>Where, (base 10) means the number is in decimal number system and (base 2) means the number is in binary number system.</p>

Course Contents
<p>Conversion of Octal to Decimal:</p> <p>Convert octal number $156_{(base\ 8)}$ into decimal form</p> $156 = 1 \times 8^2 + 5 \times 8^1 + 6 \times 8^0 = 64 + 40 + 6 = 110$

Course Contents
<p>Conversion of Hexadecimal to Decimal:</p> <p>Convert hexadecimal number $A1_{(base\ 16)}$ into decimal form</p> $A1_{(base\ 16)} = Ax16^1 + 1x16^0 = 10x16^1 + 1x16^0 = 160 + 1 = 161$ <p>So, the required decimal number is</p> $A1_{(base\ 16)} = 161_{(base\ 10)}$ <p>Alternatively, $(A1)_{16} = (161)_{10}$ Where, (base 10) means the number is in decimal number system and (base 16) means the number is in hexadecimal number system.</p>

Course Contents
<p>Conversion of Binary to Octal:</p> <p>Steps:</p> <ol style="list-style-type: none"> 1. Divide the binary digits into groups of 3 digits, starting from the right. 2. Convert each group of 3 binary digits into 1 octal digit. <p>Convert Binary number 100101_2 into Octal form</p> <div style="border: 1px solid black; padding: 5px;"> <p>Step 1. Make groups of 3 digits from right 100101_2 Groups: $100_2\ 101_2$</p> <p>Step 2. Convert each 3 digits group into 1 octal digit $101_2 = 5_8\ 100_2 = 4_8$ so, $100101_2 = 45_8$</p> </div>

Course Contents
<p>Conversion of Binary to Hexadecimal:</p> <p>Steps:</p> <ol style="list-style-type: none"> 1. Divide the binary digits into groups of 4 digits, starting from the right 2. Convert each group of 4 binary digits into 1 hexadecimal digit <p>Convert Binary number 10100101_2 into Hexadecimal form</p> <div style="border: 1px solid black; padding: 5px;"> <p>Step 1. Make groups of 4 digits from right 10100101_2 Groups: $1010_2\ 0101_2$ $0101_2 = 5_{16}$ $1010_2 = 4_{16}$</p> <p>Step 2. Combine the groups so, $10100101_2 = 45_{16}$</p> </div>

Course Contents
<p>Conversion of Octal to Binary:</p> <p>Steps</p> <ol style="list-style-type: none"> 1. Convert each octal digits into 3 digits binary group 2. Combine the groups <p>Convert Octal number 45_8 into Binary form</p> <div style="border: 1px solid black; padding: 5px;"> <p>Step 1. Convert each octal digit into 3 digits binary group 45_8 Groups: $4_8\ 5_8$ $5_8 = 101_2$ $4_8 = 100_2$</p> <p>Step 2. Combine the groups so, $45_8 = 100101_2$</p> </div>

Course Contents
<p>Conversion of Hexadecimal to Binary:</p> <p>Steps</p> <ol style="list-style-type: none"> 1. Convert each hexadecimal digit into group of 4 digits binary 2. Combine the groups <p>Convert Hexadecimal number $A5_{16}$ into Binary form</p> <div style="border: 1px solid black; padding: 5px;"> <p>Step 1. Convert each hexadecimal digit into group of 4 digits binary $A5_{16}$ Groups: $A_{16}\ 5_{16}$ $5_{16} = 0101_2$ $A_{16} = 1010_2$</p> <p>Step 2. Combine the groups so, $A5_{16} = 10100101_2$</p> </div>

Course Contents
<p>Octal to Hexadecimal conversion:</p> <p>Steps</p> <ol style="list-style-type: none"> 1. Convert each octal digit into groups of 3 digits binary 2. Combine the groups from step 1 3. Divide the binary digits from step 2 into groups of 4 digits, starting from the right 4. Convert each group of 4 binary digits into 1 hexadecimal digit

Course Contents

Octal to Hexadecimal conversion:

Convert Octal number 25_8 into Hexadecimal form

Step 1. Convert each octal digit into groups of 3 digits binary 25_8
Groups: 2_8 5_8
 $5_8 = 101_2$
 $2_8 = 010_2$
Step 2. Combine the groups
so, $25_8 = 010101_2$
Step 3. Divide the binary digits from step 2 into groups of 4 digits, starting from the right
Groups: 0001_2 0101_2
Step 4. Convert each group of 4 binary digits into 1 hexadecimal digit
 $0101_2 = 5_{16}$
 $0001_2 = 1_{16}$
so, $25_8 = 15_{16}$

Course Contents

Hexadecimal to Octal conversion:

Steps:

1. Convert each hexadecimal digit into groups of 4 digits binary
2. Combine the groups from step 1
3. Divide the binary digits from step 2 into groups of 3 digits, starting from the right
4. Convert each group of 3 binary digits into 1 octal digit

Course Contents

Hexadecimal to Octal conversion:

Convert Hexadecimal number 15_{16} into Octal form

Step 1. Convert each hexadecimal digit into groups of 4 digits binary 15_{16}
Groups: 1_{16} 5_{16}
 $5_{16} = 0101_2$
 $1_{16} = 0001_2$
Step 2. Combine the groups
so, $15_{16} = 00010101_2$
Step 3. Divide the binary digits from step 2 into groups of 3 digits, starting from the right
Groups: 000_2 010_2 101_2
Step 4. Convert each group of 3 binary digits into 1 octal digit
 $101_2 = 5_8$
 $010_2 = 2_8$
 $000_2 = 0_8$
so, $15_{16} = 025_8 = 25_8$

Course Contents

Unit-05:Data Representation (6 Hrs.)

- Binary Arithmetic;
- Signed and Unsigned Numbers;
- Binary Data Representation;
- Binary Coding Schemes;
- Logic Gates

Course Contents

Binary Arithmetic:

Decimal Number System

- Uses digits 0-9
- Digits combined to form numbers like 104, 4561
- Decimal arithmetic operations
 - Addition, subtraction, multiplication, division
- For e.g., a chocolate costs Rs. 5/-. Total cost of 2 chocolates will be Rs. 10/- i.e. $(5*2)$ or $(5+5)$

Course Contents

Binary Arithmetic:

Binary Number System

- Used in computer systems
- Uses digits 0's and 1's only
- Digits combined to form numbers like 1001, 11000110
- A digit 0 or 1 is called a bit (binary digit)
 - 1001 is a 4-bit number.
 - 11000110 is an 8-bit number

Course Contents
Binary Arithmetic:
Binary Number System
<ul style="list-style-type: none">➤ All data is represented internally in a computer by a combination of bits.➤ Each symbol is represented by a combination of bits.

Course Contents
Binary Arithmetic:
Binary Arithmetic
<ul style="list-style-type: none">➤ Arithmetic operations performed on binary numbers is called <i>binary arithmetic</i>.<ul style="list-style-type: none">▪ addition, subtraction, multiplication, division.➤ Computer systems actually perform only Binary Addition and Binary Subtraction.➤ Binary Multiplication and Division is performed using some simple operations

Course Contents
Binary Arithmetic:
Binary Addition
<ul style="list-style-type: none">➤ Involves addition of two or more binary numbers.➤ Uses Binary addition rules

Course Contents																				
Binary Arithmetic:																				
Binary Addition Rules: Two Inputs																				
<table border="1"><thead><tr><th>Input 1</th><th>Input 2</th><th>Sum</th><th>Carry</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td><td>No carry</td></tr><tr><td>0</td><td>1</td><td>1</td><td>No carry</td></tr><tr><td>1</td><td>0</td><td>1</td><td>No carry</td></tr><tr><td>1</td><td>1</td><td>0</td><td>1</td></tr></tbody></table>	Input 1	Input 2	Sum	Carry	0	0	0	No carry	0	1	1	No carry	1	0	1	No carry	1	1	0	1
Input 1	Input 2	Sum	Carry																	
0	0	0	No carry																	
0	1	1	No carry																	
1	0	1	No carry																	
1	1	0	1																	

Course Contents										
Binary Arithmetic:										
Binary Addition Rules: Three Inputs										
<table border="1"><thead><tr><th>Input 1</th><th>Input 2</th><th>Input 3</th><th>Sum</th><th>Carry</th></tr></thead><tbody><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></tbody></table>	Input 1	Input 2	Input 3	Sum	Carry	1	1	1	1	1
Input 1	Input 2	Input 3	Sum	Carry						
1	1	1	1	1						

Course Contents
Binary Arithmetic:
Addition of Binary Numbers
<ol style="list-style-type: none">1. Start by adding bits in unit column (rightmost column)2. Result of adding bits of a column is a <i>sum</i> with or without a <i>carry</i>.3. Write <i>sum</i> in result of that column.4. If <i>carry</i> is present, <i>carry</i> is carried-over to addition of the adjacent left column.5. Then repeat the above steps, for each of the columns, i.e., tens column, hundreds column and so on.

Course Contents

Binary Arithmetic:

Binary Addition: Example

Example 1. Add 10 and 01. Verify answer with the help of decimal addition.

Binary Addition	Decimal Addition
$\begin{array}{r} 10 \\ + 01 \\ \hline 11 \end{array}$	$\begin{array}{r} 2 \\ + 1 \\ \hline 3 \end{array}$
$11_2 = 3_{10}$	

Course Contents

Binary Arithmetic:

Binary Addition: Example

Example 2. Add 01 and 11. Verify answer with the help of decimal addition.

Binary Addition	Decimal Addition
$\begin{array}{r} 01 \\ + 11 \\ \hline 100 \end{array}$	$\begin{array}{r} 1 \\ + 3 \\ \hline 4 \end{array}$
$100_2 = 4_{10}$	

Course Contents

Binary Arithmetic:

Binary Addition: Example

Example 3. Add 11 and 11. Verify answer with the help of decimal addition.

Binary Addition	Decimal Addition
$\begin{array}{r} 11 \\ + 11 \\ \hline 110 \end{array}$	$\begin{array}{r} 3 \\ + 3 \\ \hline 6 \end{array}$
$110_2 = 6_{10}$	

Course Contents

Binary Arithmetic:

Binary Addition: Example

Example 4. Add 1101 and 1111. Verify answer with the help of decimal addition.

Binary Addition	Decimal Addition
$\begin{array}{r} 1101 \\ + 1111 \\ \hline 11100 \end{array}$	$\begin{array}{r} 13 \\ + 15 \\ \hline 28 \end{array}$
$11100_2 = 28_{10}$	

Course Contents

Binary Arithmetic:

Binary Addition: Example

Example 5. Add 10111, 11100 and 11. Verify answer with the help of decimal addition.

Binary Addition	Decimal Addition
$\begin{array}{r} 10111 \\ 11100 \\ + 11 \\ \hline 110110 \end{array}$	$\begin{array}{r} 23 \\ 28 \\ + 3 \\ \hline 54 \end{array}$
$110110_2 = 54_{10}$	

Course Contents

Binary Arithmetic:

Binary Subtraction

> Uses *Binary subtraction rules*

Binary Subtraction Rules

Input 1	Input 2	Difference	Borrow
0	0	0	No borrow
0	1	1	1
1	0	1	No borrow
1	1	0	No borrow

Course Contents	
Binary Arithmetic:	
Subtraction of Binary Numbers	
<ol style="list-style-type: none"> 1. Start by subtracting bit in lower row from bit in upper row, in unit column. 2. If bit in upper row is less than the bit in lower row, <i>borrow</i> 1 from upper row of adjacent left column. 3. Result of subtracting two bits is the <i>difference</i>. 4. Write <i>difference</i> in result of that column. 5. Then repeat the above steps, for each of the columns, i.e., tens column, hundreds column and so on. 	

Course Contents							
Binary Arithmetic:							
Binary Subtraction: Example							
Example 1. Subtract 01 from 11. Verify answer with the help of decimal subtraction.							
<table border="1"> <thead> <tr> <th>Binary Subtraction</th> <th>Decimal Subtraction</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 11 \\ - 01 \\ \hline 10 \end{array}$ </td> <td> $\begin{array}{r} 3 \\ - 1 \\ \hline 2 \end{array}$ </td> </tr> <tr> <td colspan="2" style="text-align: center;">$10_2 = 2_{10}$</td> </tr> </tbody> </table>	Binary Subtraction	Decimal Subtraction	$\begin{array}{r} 11 \\ - 01 \\ \hline 10 \end{array}$	$\begin{array}{r} 3 \\ - 1 \\ \hline 2 \end{array}$	$10_2 = 2_{10}$		
Binary Subtraction	Decimal Subtraction						
$\begin{array}{r} 11 \\ - 01 \\ \hline 10 \end{array}$	$\begin{array}{r} 3 \\ - 1 \\ \hline 2 \end{array}$						
$10_2 = 2_{10}$							

Course Contents							
Binary Arithmetic:							
Binary Subtraction: Example							
Example 2. Subtract 01 from 10. Verify answer with the help of decimal subtraction.							
<table border="1"> <thead> <tr> <th>Binary Subtraction</th> <th>Decimal Subtraction</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 010 \\ \cancel{1}0 \\ - 01 \\ \hline 01 \end{array}$ </td> <td> $\begin{array}{r} 2 \\ - 1 \\ \hline 1 \end{array}$ </td> </tr> <tr> <td colspan="2" style="text-align: center;">$01_2 = 1_{10}$</td> </tr> </tbody> </table>	Binary Subtraction	Decimal Subtraction	$\begin{array}{r} 010 \\ \cancel{1}0 \\ - 01 \\ \hline 01 \end{array}$	$\begin{array}{r} 2 \\ - 1 \\ \hline 1 \end{array}$	$01_2 = 1_{10}$		
Binary Subtraction	Decimal Subtraction						
$\begin{array}{r} 010 \\ \cancel{1}0 \\ - 01 \\ \hline 01 \end{array}$	$\begin{array}{r} 2 \\ - 1 \\ \hline 1 \end{array}$						
$01_2 = 1_{10}$							

Course Contents							
Binary Arithmetic:							
Binary Subtraction: Example							
Example 3. Subtract 0111 from 1110. Verify answer with the help of decimal subtraction.							
<table border="1"> <thead> <tr> <th>Binary Subtraction</th> <th>Decimal Subtraction</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 1010 \\ 00010 \\ \cancel{1}\cancel{1}\cancel{1}0 \\ - 0111 \\ \hline 0111 \end{array}$ </td> <td> $\begin{array}{r} 14 \\ - 7 \\ \hline 7 \end{array}$ </td> </tr> <tr> <td colspan="2" style="text-align: center;">$0111_2 = 7_{10}$</td> </tr> </tbody> </table>	Binary Subtraction	Decimal Subtraction	$\begin{array}{r} 1010 \\ 00010 \\ \cancel{1}\cancel{1}\cancel{1}0 \\ - 0111 \\ \hline 0111 \end{array}$	$\begin{array}{r} 14 \\ - 7 \\ \hline 7 \end{array}$	$0111_2 = 7_{10}$		
Binary Subtraction	Decimal Subtraction						
$\begin{array}{r} 1010 \\ 00010 \\ \cancel{1}\cancel{1}\cancel{1}0 \\ - 0111 \\ \hline 0111 \end{array}$	$\begin{array}{r} 14 \\ - 7 \\ \hline 7 \end{array}$						
$0111_2 = 7_{10}$							

Course Contents							
Binary Arithmetic:							
Binary Subtraction: Example							
Example 4. Subtract 10010 from 10101. Verify answer with the help of decimal subtraction.							
<table border="1"> <thead> <tr> <th>Binary Subtraction</th> <th>Decimal Subtraction</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 010 \\ 10\cancel{1}01 \\ - 10010 \\ \hline 00011 \end{array}$ </td> <td> $\begin{array}{r} 21 \\ - 18 \\ \hline 3 \end{array}$ </td> </tr> <tr> <td colspan="2" style="text-align: center;">$00011_2 = 3_{10}$</td> </tr> </tbody> </table>	Binary Subtraction	Decimal Subtraction	$\begin{array}{r} 010 \\ 10\cancel{1}01 \\ - 10010 \\ \hline 00011 \end{array}$	$\begin{array}{r} 21 \\ - 18 \\ \hline 3 \end{array}$	$00011_2 = 3_{10}$		
Binary Subtraction	Decimal Subtraction						
$\begin{array}{r} 010 \\ 10\cancel{1}01 \\ - 10010 \\ \hline 00011 \end{array}$	$\begin{array}{r} 21 \\ - 18 \\ \hline 3 \end{array}$						
$00011_2 = 3_{10}$							

Course Contents							
Binary Arithmetic:							
Binary Subtraction: Example							
Example 5. Subtract 101111 from 110001. Verify answer with the help of decimal							
<table border="1"> <thead> <tr> <th>Binary Subtraction</th> <th>Decimal Subtraction</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 11 \\ 0101010 \\ 1\cancel{1}0001 \\ - 101111 \\ \hline 000010 \end{array}$ </td> <td> $\begin{array}{r} 49 \\ - 47 \\ \hline 2 \end{array}$ </td> </tr> <tr> <td colspan="2" style="text-align: center;">$000010_2 = 2_{10}$</td> </tr> </tbody> </table>	Binary Subtraction	Decimal Subtraction	$\begin{array}{r} 11 \\ 0101010 \\ 1\cancel{1}0001 \\ - 101111 \\ \hline 000010 \end{array}$	$\begin{array}{r} 49 \\ - 47 \\ \hline 2 \end{array}$	$000010_2 = 2_{10}$		
Binary Subtraction	Decimal Subtraction						
$\begin{array}{r} 11 \\ 0101010 \\ 1\cancel{1}0001 \\ - 101111 \\ \hline 000010 \end{array}$	$\begin{array}{r} 49 \\ - 47 \\ \hline 2 \end{array}$						
$000010_2 = 2_{10}$							

Course Contents

Binary Arithmetic:

Signed and Unsigned numbers

- > A binary number may be positive or negative.
- > Generally, symbols “+” and “-” represent positive and negative numbers, respectively.
- > In computer, sign of a binary number has to be represented using 0 and 1.

Course Contents

Binary Arithmetic:

Signed and Unsigned numbers

- > *n*-bit signed binary number consists of two parts
 - Sign bit, and Magnitude.
- > Left most bit is called Most Significant Bit (MSB).
- > MSB is the sign bit.
- > Remaining *n*-1 bits denote magnitude of number.

Course Contents

Binary Arithmetic:

Signed and Unsigned numbers

- > Sign bit is 0 for a positive number and 1 for a negative number.
 - 0 1100011 is a positive number. Sign bit is 0, and,
 - 1 1001011 is a negative number. Sign bit is 1.

Positive Number	Negative Number
<div style="border: 1px solid black; padding: 2px; display: inline-block;"> 0 1100011 <small>MSB</small> </div>	<div style="border: 1px solid black; padding: 2px; display: inline-block;"> 1 1001011 <small>MSB</small> </div>

Course Contents

Binary Arithmetic:

Signed and Unsigned numbers

- > Data range for 8-bit signed number is:
 - -128 to +127 (-2⁷ to +2⁷-1).
 - Leftmost bit is sign bit.
- > In *n*-bit unsigned binary number, magnitude of number *n* is stored in *n* bits.
- > Data range for 8-bit unsigned number is:
 - 0 to 255 (2⁸ = 256).

Course Contents

Binary Arithmetic:

Complement of Binary Numbers

- > Used in computer for simplification of subtraction operation.
- > Two types of complements
 - 1's complement, and
 - 2's complement.

Course Contents

Binary Arithmetic:

1's complement of Binary number

- > Change bits 1 to 0 and bits 0 to 1.
- > Some examples

Binary Numbers	1 1 0	1 0 1 1	1 1 0 1 1 1 1
	↓	↓	↓
1's Complement	0 0 1	0 1 0 0	0 0 1 0 0 0 0

Course Contents

Binary Arithmetic:

2's complement of Binary number

- Add 1 to the 1's complement of the binary number.
- Some Examples:

Binary Numbers	1 1 0	1 0 1 1	1 1 0 1 1 1 1
	↓	↓	↓
1's Complement	0 0 1	0 1 0 0	0 0 1 0 0 0 0
	↓	↓	↓
2's Complement	0 1 0	0 1 0 1	0 0 1 0 0 0 1

Course Contents

Binary Arithmetic:

Binary Data Representation

- Binary number can also have a binary point, in addition to sign.
- *Binary point* used for representing fractions, integers and integer-fraction numbers.
- *Registers* are high-speed storage areas in CPU of computer. All data is brought into a register before it gets processed.

Course Contents

Binary Arithmetic:

Binary Data Representation

- Two ways of representing position of binary point in a register
 - Fixed Point Number Representation, and
 - Floating Point Number Representation.

Course Contents

Binary Arithmetic:

Fixed Point Number Representation

- Assumes binary point is fixed at one position.
- Represents +ve integer binary signed number as-
 - Sign bit is 0. Magnitude is a positive binary number.
- Represents -ve integer binary signed number as-
 - Sign bit is 1
 - Magnitude is represented in any one of three ways-
 - Signed Magnitude representation
 - Signed 1's complement representation
 - Signed 2's complement representation

Course Contents

Binary Arithmetic:

Fixed Point Number Representation..

- Signed Magnitude representation
 - Magnitude is positive binary number itself.
- Signed 1's complement representation
 - Magnitude is 1's complement of positive binary number.
- Signed 2's complement representation
 - Magnitude is 2's complement of positive binary number.

Course Contents

Binary Arithmetic:

Fixed Point Representation Signed Number 18

+18	0 0 0 1 0 0 1 0	Sign bit is 0	Binary equivalent of +18
	1 0 0 1 0 0 1 0	Sign bit is 1	Binary equivalent of +18
	1 1 0 1 1 0 1	Sign bit is 1	1's complement of +18
-18	1 1 0 1 1 1 0	Sign bit is 1	2's complement of +18

Course Contents
<p>Binary Arithmetic:</p> <p>Fixed Point Number Representation</p> <ul style="list-style-type: none"> > Signed magnitude and signed 1's complement representation are rarely used in computer arithmetic. > Signed 2's complement representation is used to represent negative numbers.

Course Contents
<p>Binary Arithmetic:</p> <p>Signed Binary Number: Addition</p> <ul style="list-style-type: none"> > Represent positive number in binary form. <ul style="list-style-type: none"> ▪ For e.g., +5 is 0000 0101, +10 is 0000 1010 > Represent negative number in 2's complement form. <ul style="list-style-type: none"> ▪ For e.g., -5 is 111 1 1011, -10 is 1111 0110 > Add bits of the two signed binary numbers. > Ignore any carry out from sign bit position.

Course Contents
<p>Binary Arithmetic:</p> <p>Signed Binary Number: Addition</p> <ul style="list-style-type: none"> > Negative output is automatically in 2's complement form. > Get decimal equivalent of negative output number <ul style="list-style-type: none"> ▪ Find its 2's complement, and ▪ Attach a negative sign to the obtained result.

Course Contents						
<p>Binary Arithmetic:</p> <p>Signed Binary Number: Addition</p> <p>Example 1. Add binary +5 and +10. Verify answer with the help of decimal addition.</p> <table border="1"> <thead> <tr> <th>Binary Addition</th> <th>Decimal Addition</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 00000101 \\ + 00001010 \\ \hline 00001111 \end{array}$ </td> <td> $\begin{array}{r} + 5 \\ + 10 \\ \hline + 15 \end{array}$ </td> </tr> <tr> <td colspan="2" style="text-align: center;"> $00001111_2 = +15_{10}$ </td> </tr> </tbody> </table>	Binary Addition	Decimal Addition	$\begin{array}{r} 00000101 \\ + 00001010 \\ \hline 00001111 \end{array}$	$\begin{array}{r} + 5 \\ + 10 \\ \hline + 15 \end{array}$	$00001111_2 = +15_{10}$	
Binary Addition	Decimal Addition					
$\begin{array}{r} 00000101 \\ + 00001010 \\ \hline 00001111 \end{array}$	$\begin{array}{r} + 5 \\ + 10 \\ \hline + 15 \end{array}$					
$00001111_2 = +15_{10}$						

Course Contents						
<p>Binary Arithmetic:</p> <p>Signed Binary Number: Addition</p> <p>Example 2. Add binary -5 and +10. Verify answer with the help of decimal addition.</p> <table border="1"> <thead> <tr> <th>Binary Addition</th> <th>Decimal Addition</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 111111 \\ 11111011 \\ + 00001010 \\ \hline 00000101 \end{array}$ </td> <td> $\begin{array}{r} - 5 \\ + 10 \\ \hline + 5 \end{array}$ </td> </tr> <tr> <td colspan="2" style="text-align: center;"> $00000101_2 = +5_{10}$ </td> </tr> </tbody> </table>	Binary Addition	Decimal Addition	$\begin{array}{r} 111111 \\ 11111011 \\ + 00001010 \\ \hline 00000101 \end{array}$	$\begin{array}{r} - 5 \\ + 10 \\ \hline + 5 \end{array}$	$00000101_2 = +5_{10}$	
Binary Addition	Decimal Addition					
$\begin{array}{r} 111111 \\ 11111011 \\ + 00001010 \\ \hline 00000101 \end{array}$	$\begin{array}{r} - 5 \\ + 10 \\ \hline + 5 \end{array}$					
$00000101_2 = +5_{10}$						

Course Contents						
<p>Binary Arithmetic:</p> <p>Signed Binary Number: Addition</p> <p>Example 3. Add binary +5 and -10. Verify answer with the help of decimal addition.</p> <table border="1"> <thead> <tr> <th>Binary Addition</th> <th>Decimal Addition</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 1 \\ 00000101 \\ + 11110110 \\ \hline 11111011 \end{array}$ </td> <td> $\begin{array}{r} + 5 \\ - 10 \\ \hline - 5 \end{array}$ </td> </tr> <tr> <td colspan="2" style="text-align: center;"> $11111011_2 = -5_{10}$ </td> </tr> </tbody> </table>	Binary Addition	Decimal Addition	$\begin{array}{r} 1 \\ 00000101 \\ + 11110110 \\ \hline 11111011 \end{array}$	$\begin{array}{r} + 5 \\ - 10 \\ \hline - 5 \end{array}$	$11111011_2 = -5_{10}$	
Binary Addition	Decimal Addition					
$\begin{array}{r} 1 \\ 00000101 \\ + 11110110 \\ \hline 11111011 \end{array}$	$\begin{array}{r} + 5 \\ - 10 \\ \hline - 5 \end{array}$					
$11111011_2 = -5_{10}$						

Course Contents					
Binary Arithmetic:					
Signed Binary Number: Addition					
Example 4. Add binary -5 and -10. Verify answer with the help of decimal addition.					
<table border="1"> <thead> <tr> <th>Binary Addition</th> <th>Decimal Addition</th> </tr> </thead> <tbody> <tr> <td> $\begin{array}{r} 1111111 \\ 11111011 \\ + 11110110 \\ \hline 111110001 \end{array}$ </td> <td> $\begin{array}{r} -5 \\ -10 \\ \hline -15 \end{array}$ </td> </tr> </tbody> </table>	Binary Addition	Decimal Addition	$\begin{array}{r} 1111111 \\ 11111011 \\ + 11110110 \\ \hline 111110001 \end{array}$	$\begin{array}{r} -5 \\ -10 \\ \hline -15 \end{array}$	
Binary Addition	Decimal Addition				
$\begin{array}{r} 1111111 \\ 11111011 \\ + 11110110 \\ \hline 111110001 \end{array}$	$\begin{array}{r} -5 \\ -10 \\ \hline -15 \end{array}$				
$1111\ 0001_2 = -15_{10}$					

Course Contents
Binary Arithmetic:
Signed Binary Number: Subtraction
<ul style="list-style-type: none"> > Changed to addition of two signed numbers. > Sign of second number is changed before performing the addition operation.

Course Contents
Binary Arithmetic:
Signed Binary Number: Subtraction
$\begin{array}{l} (+A) - (+B) = (+A) + (-B) \\ (+A) - (-B) = (+A) + (+B) \\ (-A) - (-B) = (-A) + (+B) \\ (-A) - (+B) = (-A) + (-B) \end{array}$

Course Contents
Binary Coding Schemes:
Binary Coding Schemes
<ul style="list-style-type: none"> > Data - alphabetic, numeric, alphanumeric, sound, video. > Data represented as combination of bits in computer. > Bits are grouped in a fixed size. > Code is made by combining bits of definite size.

Course Contents
Binary Coding Schemes:
Binary Coding Schemes...
<ul style="list-style-type: none"> > Represents symbols in a standard code. > Combination of bits represents a unique symbol. > Standard code enables programmers to use same combination of bits to represent a symbol in data.

Course Contents
Binary Coding Schemes:
Binary Coding Schemes...
<ul style="list-style-type: none"> > Commonly used binary coding schemes: <ul style="list-style-type: none"> ▪ ASCII, ▪ EBCDIC, and ▪ Unicode

Course Contents

Binary Coding Schemes:

EBCDIC

- EBCDIC stands for Extended Binary Coded Decimal Interchange Code
- 8-bit code. 4 bits for zone; 4 bits for digit.
- Allows $2^8 = 256$ combinations.
- Represents 256 unique symbols.
- Used mainly in mainframe computers.

Course Contents

Binary Coding Schemes:

ASCII

- ASCII stands for American Standard Code for Information Interchange
- Two types of ASCII codes
 - ASCII-7 and
 - ASCII-8.

Course Contents

Binary Coding Schemes:

ASCII-7

- Standard ASCII code.
- 7-bit code. 3 bits for zone; 4 bits for digits.
- Allows $2^7 = 128$ combinations.
- Represents 128 unique symbols.
- ASCII-7 modified by IBM to ASCII-8.

The **zone** used by **ASCII** for alphabets is 0100. For e.g. A is represented as 0100(**zone**)0001(**digit**). The hex equivalent is 41 for A.
Character A = 0100 0001; Character B = 0100 0010

Course Contents

Binary Coding Schemes:

ASCII-8

- Extended version of ASCII-7.
- 8-bit code. 4 bits for zone; 4 bits for digit.
- Allows $2^8 = 256$ combinations.
- Represents 256 unique symbols.
- Widely used to represent data in computer.

Course Contents

Binary Coding Schemes:

ASCII-8

- ASCII-8 code represents 256 symbols.
 - 0-31 for control characters.
 - Non-printable. Carriage Return (CR), Bell (BEL).
 - 48-57 for numeric 0-9.
 - 65-90 for uppercase letters A-Z.
 - 97-122 for lowercase letters a-z.
 - 128-255 are extended ASCII codes.

Course Contents

Binary Coding Schemes:

Unicode

- Universal character encoding standard
 - Represents text, symbols, characters in multi-lingual environments.
 - Uniquely represent a symbol in languages like Chinese, Japanese etc.
 - Represents mathematical and scientific symbols.
- 32 bit code.
- Allows $2^{32} = \text{approx. 4 billion combinations}$.

Course Contents

Binary Coding Schemes:

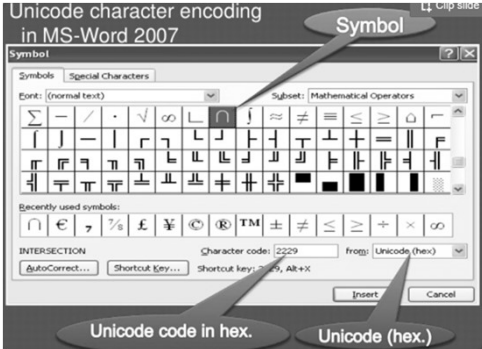
Unicode...

- Compatible with ASCII-8 codes.
- Unicode's first 256 codes identical to ASCII-8 codes.
- Implemented by *character encodings*.
- UTF-8 : A character encoding
 - Most commonly used encoding scheme.
 - Uses 8-32 bits per code.

UTF: Unicode Transformation Format

Course Contents

Binary Coding Schemes:



Unicode code in hex. Unicode (hex.)

Course Contents

Binary Coding Schemes:

UTF encodings include:

- UTF-1, a retired predecessor of UTF-8, maximizes compatibility with ISO 2022, no longer part of The Unicode Standard;
- UTF-7, a 7-bit encoding sometimes used in e-mail, often considered obsolete (not part of The Unicode Standard, but only documented as an informational RFC, i.e., not on the Internet Standards Track);
- UTF-8, an 8-bit variable-width encoding which maximizes compatibility with ASCII;
- UTF-EBCDIC, an 8-bit variable-width encoding similar to UTF-8, but designed for compatibility with EBCDIC (not part of The Unicode Standard);
- UTF-16, a 16-bit, variable-width encoding;
- UTF-32, a 32-bit, fixed-width encoding.

Course Contents

Unit-05:Data Representation (6 Hrs.)

- **Signed and Unsigned Numbers;**
- **Binary Data Representation;**
- **Binary Coding Schemes;**
- **Logic Gates**

Course Contents

Converting Decimal Fraction to Binary, Octal and Hexa:

Example 5: Convert 0.2345 from Base 10 to Base 2.

0.2345	
—x2	
0.4690	
—x2	
0.9380	
—x2	
1.8760	
—x2	
3.7520	
—x2	
7.5040	
—x2	
15.0080	
—x2	
30.0160	
—x2	
60.0320	
—x2	
120.0640	
—x2	
240.1280	
—x2	
480.2560	
—x2	
960.5120	
—x2	
1920.10240	
—x2	
3840.20480	
—x2	
7680.40960	
—x2	
15360.81920	
—x2	
30720.163840	
—x2	
61440.327680	
—x2	
122880.655360	
—x2	
245760.1310720	
—x2	
491520.2621440	
—x2	
983040.5242880	
—x2	
1966080.10485760	
—x2	
3932160.20971520	
—x2	
7864320.41943040	
—x2	
15728640.83886080	
—x2	
31457280.167772160	
—x2	
62914560.335544320	
—x2	
125829120.671088640	
—x2	
251658240.134217280	
—x2	
503316480.268434560	
—x2	
1006632960.536869120	
—x2	
2013265920.1073738240	
—x2	
4026531840.2147476480	
—x2	
8053063680.4294952960	
—x2	
16106127360.8589905920	
—x2	
32212254720.1717981120	
—x2	
64424509440.3435962240	
—x2	
128849018880.6871924480	
—x2	
257698037760.1374368960	
—x2	
515396075520.2748737920	
—x2	
1030792151040.5497475840	
—x2	
2061584302080.10994951680	
—x2	
4123168604160.21989903360	
—x2	
8246337208320.43979806720	
—x2	
16492674416640.87959613440	
—x2	
32985348833280.175919226880	
—x2	
65970697666560.351838453760	
—x2	
131941395333120.703676907520	
—x2	
263882790666240.1407353815040	
—x2	
527765581332480.2814707630080	
—x2	
1055531162664960.5629415260160	
—x2	
2111062325329920.11258830520320	
—x2	
4222124650659840.22517661040640	
—x2	
8444249301319680.45035322081280	
—x2	
16888498602639360.90070644162560	
—x2	
33776997205278720.180141282245120	
—x2	
67553994410557440.360282564490240	
—x2	
135107988821114880.720565128980480	
—x2	
270215977642229760.1441130257960960	
—x2	
540431955284459520.2882260515921920	
—x2	
1080863910568919040.5764521031843840	
—x2	
2161727821137838080.11529042076886880	
—x2	
4323455642275676160.23058084153773760	
—x2	
8646911284551352320.46116168307547520	
—x2	
17293822569102704640.92232336615095040	
—x2	
34587645138205409280.184464672301900160	
—x2	
69175290276410818560.368929344603800320	
—x2	
138350580552821637120.737858689207600640	
—x2	
276701161105643274240.1475717378415012800	
—x2	
553402322211286548480.2951434756830025600	
—x2	
1106804644222573097600.5902869513660051200	
—x2	
2213609288445146195200.11805739027320102400	
—x2	
4427218576890292390400.23611478054640204800	
—x2	
8854437153780584780800.47222956109280409600	
—x2	
17708874307561169561600.94445912218560819200	
—x2	
35417748615122339123200.188891824371217384000	
—x2	
70835497230244678246400.377783648742434768000	
—x2	
141670994460489356492800.755567297484869536000	
—x2	
283341988920978712985600.1511134949697391040000	
—x2	
566683977841957425971200.3022269899394782080000	
—x2	
1133367955683914851942400.6044539798789564160000	
—x2	
2266735911367829703884800.120890795955791283200000	
—x2	
4533471822735659407769600.241781591911582566400000	
—x2	
9066943645471318815539200.483563183823165132800000	
—x2	
18133887290942637631078400.967126367646330265600000	
—x2	
36267774581885275262156800.1934252752932605312000000	
—x2	
72535549163770550524313600.3868505505865210624000000	
—x2	
145071098327541101048627200.7737011011730421248000000	
—x2	
290142196655082202097254400.15474022023460844960000000	
—x2	
580284393310164404194508800.30948044046921689920000000	
—x2	
1160568786620328808389017600.61896088093843379840000000	
—x2	
2321137573240657616778035200.1237921761876867596800000000	
—x2	
4642275146481315233556070400.2475843523753735193600000000	
—x2	
9284550292962630467112140800.49516870475074703872000000000	
—x2	
18569100585925260934224281600.99033740950149407744000000000	
—x2	
37138201171850521868448563200.1980674819029888154880000000000	
—x2	
74276402343701043736897126400.3961349638059776309760000000000	
—x2	
148552804687402087473794252800.7922699276119552619520000000000	
—x2	
297105609374804174947588505600.15845398542339105391040000000000	
—x2	
594211218749608349895177011200.31690797084678210782080000000000	
—x2	
1188422437499216699790354022400.633815941693564215641600000000000	
—x2	
2376844874998433399580708044800.12676318833871284312832000000000000	
—x2	
4753689749996866799161416089600.25352637667742568625664000000000000	
—x2	
9507379499993733598322832179200.507052753354851372513280000000000000	
—x2	
19014758999987467196645664358400.1014105507099702745026560000000000000	
—x2	
38029517999974934393291328716800.20282110141994054900531200000000000000	
—x2	
76059035999949868786582657433600.40564220283988109801062400000000000000	
—x2	
152118071999899377573165314867200.811284405679762196021248000000000000000	
—x2	
304236143999798755146330629734400.1622568913595244392042560000000000000000	
—x2	
608472287999597510292661259468800.3245137827190488784085120000000000000000	
—x2	
1216944575999195020585322518937600.6490275654380977568170240000000000000000	
—x2	
2433889151998390041170645037875200.12980553107619551336340480000000000000000	
—x2	
4867778303996780082341290075750400.259611062152391026726809600000000000000000	
—x2	
9735556607993560164682580151500800.5192221243047820534536192000000000000000000	
—x2	
19471113215987120329365160303001600.103844424860956410690723840000000000000000000	
—x2	
38942226431974240658730320606003200.2076888497219128213814476800000000000000000000	
—x2	
77884452863948481317460641212006400.41537769944382564276289536000000000000000000000	
—x2	
15576890572789696263492128244012800.830755398887651285525790720000000000000000000000	
—x2	
31153781145579392526984256488025600.1661510797755302571051581440000000000000000000000	
—x2	
62307562291158785053968512976051200.33230215955106051421031628800000000000000000000000	
—x2	
124615124582317570107937025952102400.664604319102121028420632576000000000000000000000000	
—x2	
249230249164635140215874051904204800.1329208638044242056841251532800000000000000000000000	
—x2	
498460498329270280431748103808409600.26584172760884841136825030656000000000000000000000000	
—x2	
996920996658540560863496207616819200.531683455217696822736500613120000000000000000000000000	
—x2	
1993841993317081121726992415233638400.10633669103533936454730012262400000000000000000000000000	
—x2	
3987683986634162243453984830467276800.212673382070678729094600245248000000000000000000000000000	
—x2	
7975367973268324486907969660934553600.4253467641413574581892004904960000000000000000000000000000	
—x2	
15950735946536648973815939321869107200.8506935282827149163784009809920000000000000000000000000000	
—x2	
31901471893073297947631878643738214400.170138705656542983275680196198400000000000000000000000000000	
—x2	
63802943786146595895263757287476428800.3402774113130859665513603923968000000000000000000000000000000	
—x2	
127605887572293191790527514574952857600.68055482262617193310272078479360000000000000000000000000000000	
—x2	
255211775144586	

Course Contents

Converting Decimal Integer, Fraction to Binary, Octal and Hexa:

Example 6: Convert 34.4674 from Base 10 to Base 2.

to Base	Number	Remainder	
			0.4674
			<u> </u> x 2
2	34		0.9348
			<u> </u> x 2
2	17	0	1.8696
			<u> </u> x 2
2	8	1	1.7392
			<u> </u> x 2
2	4	0	1.4784
			<u> </u> x 2
2	2	0	0.9568
			<u> </u> x 2
2	1	0	1.8136
			<u> </u> x 2
0	0	1	

The binary equivalent of (34)₁₀ is (100010)₂

The binary equivalent of (0.4674)₁₀ is (.011101)₂

The binary equivalent of (34.4674)₁₀ is (100010.011101)₂

Course Contents

Converting Decimal Integer, Fraction to Binary, Octal and Hexa:

Example 7: Convert 34.4674 from Base 10 to Base 8.

to Base	Number	Remainder	
			0.4674
			<u> </u> x 8
8	34		3.7392
			<u> </u> x 8
8	4	2	5.9136
			<u> </u> x 8
	0	4	7.3088
			<u> </u> x 8
			2.4704

The octal equivalent of (34)₁₀ is (42)₈

The octal equivalent of (0.4674)₁₀ is (.3572)₈

The octal equivalent of (34.4674)₁₀ is (42.3572)₈

Course Contents

Converting Decimal Integer, Fraction to Binary, Octal and Hexa:

Example 8: Convert 34.4674 from Base 10 to Base 16.

to Base	Number	Remainder	
			0.4674
			<u> </u> x 16
16	34		28044
			<u> </u> x 16
16	4	2	4674x
			<u> </u> x 16
	0	2	94784
			<u> </u> x 16
			28704
			<u> </u> x 16
			4784x
			<u> </u> x 16
			76544
			<u> </u> x 16
			39264
			<u> </u> x 16
			6544x
			<u> </u> x 16
			104904
			<u> </u> x 16
			29424
			<u> </u> x 16
			4904x
			<u> </u> x 16
			78464

The hexadecimal equivalent of (34)₁₀ is (22)₁₆

The hexadecimal equivalent of (0.4674)₁₀ is (.97A7)₁₆

The hexadecimal equivalent of (34.4674)₁₀ is (22.97A7)₁₆

Course Contents

Converting Fraction Binary, Octal and Hexa to Decimal

<p>1011 from Base 2 to Base 10</p> $1011 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$ $= 1 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 1 \cdot 1$ $= 8 + 0 + 2 + 1$ $= 11$ <p>The decimal equivalent of (1011)₂ is 11.</p>	<p>62 from Base 8 to Base 10</p> $62 = 6 \cdot 8^1 + 2 \cdot 8^0$ $= 6 \cdot 8 + 2 \cdot 1$ $= 48 + 2$ $= 50$ <p>The decimal equivalent of (62)₈ is 50.</p>	<p>C15 from Base 16 to Base 10</p> $C15 = C \cdot 16^1 + 1 \cdot 16^0 + 5 \cdot 16^{-1}$ $= 12 \cdot 256 + 1 \cdot 16 + 5 \cdot 1$ $= 3072 + 16 + 5$ $= 3093$ <p>The decimal equivalent of (C15)₁₆ is 3093</p>
---	--	---

Example 10: Convert .1101 from Base 2 to Base 10.
Convert .345 from Base 8 to Base 10.
Convert .15 from Base 16 to Base 10.

<p>.1101 from Base 2 to Base 10</p> $.1101 = 1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4}$ $= 1/2 + 1/4 + 0 + 1/16$ $= 13/16$ $= .8125$ <p>The decimal equivalent of (.1101)₂ is .8125</p>	<p>.345 from Base 8 to Base 10</p> $.345 = 3 \cdot 8^{-1} + 4 \cdot 8^{-2} + 5 \cdot 8^{-3}$ $= 3/8 + 4/64 + 5/512$ $= 229/512$ $= .447$ <p>The decimal equivalent of (.345)₈ is .447</p>	<p>.15 from Base 16 to Base 10</p> $.15 = 1 \cdot 16^{-1} + 5 \cdot 16^{-2}$ $= 1/16 + 5/256$ $= 21/256$ $= .082$ <p>The decimal equivalent of (.15)₁₆ is .082</p>
--	--	---

Course Contents

Converting Fraction Binary, Octal and Hexa to Decimal

Example 11: Convert 1011.1001 from Base 2 to Base 10.
Convert 24.36 from Base 8 to Base 10.
Convert 4D.21 from Base 16 to Base 10.

<p>1011.1001 from Base 2 to Base 10</p> $1011.1001 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4}$ $= 8 + 0 + 2 + 1 + 1/2 + 0 + 0 + 1/16$ $= 11 + 9/16$ $= 11.5625$ <p>The decimal equivalent of (1011.1001)₂ is 11.5625</p>	<p>24.36 from Base 8 to Base 10</p> $24.36 = 2 \cdot 8^1 + 4 \cdot 8^0 + 3 \cdot 8^{-1} + 6 \cdot 8^{-2}$ $= 16 + 4 + 3/8 + 6/64$ $= 20 + 30/64$ $= 20.46875$ <p>The decimal equivalent of (24.36)₈ is 20.46875</p>	<p>4D.21 from Base 16 to Base 10</p> $4D.21 = 4 \cdot 16^1 + D \cdot 16^0 + 2 \cdot 16^{-1} + 1 \cdot 16^{-2}$ $= 64 + 13 + 2/16 + 1/256$ $= 77 + 33/256$ $= 77.1289$ <p>The decimal equivalent of (4D.21)₁₆ is 77.1289</p>
--	--	--

Course Contents

• Binary Data Representation;

A binary number may also have a binary point, in addition to the sign. The binary point is used for representing fractions, integers and integer-fraction numbers. *Registers* are high-speed storage areas within the Central Processing Unit (CPU) of the computer. All data are brought into a register before it can be processed. For example, if two numbers are to be added, both the numbers are brought in registers, added, and the result is also placed in a register. There are two ways of representing the position of the binary point in the register—fixed point number representation and floating point number representation.

The *fixed point number representation* assumes that the binary point is fixed at one position either at the extreme left to make the number a fraction, or at the extreme right to make the number an integer. In both cases, the binary point is not stored in the register, but the number is treated as a fraction or integer. For example, if the binary point is assumed to be at extreme left, the number 1100 is actually treated as 0.1100.

The *floating point number representation* uses two registers. The first register stores the number without the binary point. The second register stores a number that indicates the position of the binary point in the first register.

We shall now discuss representation of data in the fixed point number representation and floating point number representation.