UNIT –V
REPRESENTING NUMERICAL DATA

1	**a**	Calculate the value of largest unsigned integer that can be stored as a 16-bit number.	[L3][CO4]	**[2M]**

Answer:

RESULT – 2M

Largest unsigned integer of n-bit binary number is = $2^n - 1$
Largest unsigned integer of 16-bit binary number is = $2^{16} - 1 = 65536 - 1 = 65535$

	b	Describe the unsigned binary and binary coded decimal representations with an example.	[L2][CO6]	**[5M]**

Answer:

EXPLANATION – 5M

Unsigned numbers don't have any sign, these can contain only magnitude of the number. So, representation of unsigned binary numbers are all positive numbers only. For example, representation of positive decimal numbers are positive by default. We always assume that there is a positive sign symbol in front of every number.

Representation of Unsigned Binary Numbers:

Since there is no sign bit in this unsigned binary number, so N bit binary number represent its magnitude only. Every number in unsigned number representation has only one unique binary equivalent form, so this is unambiguous representation technique. The range of unsigned binary number is from 0 to (2^n-1).

Example-: Represent decimal number 92 in unsigned binary number.

Simply convert it into Binary number, it contains only magnitude of the given number.
$= (92)_{10}$

$= (1x2^6+0x2^5+1x2^4+1x2^3+1x2^2+0x2^1+0x2^0)_{10}$

$= (1011100)_2$

It's 7 bit binary magnitude of the decimal number 92.

Binary Coded Decimal

Decimal Number	Standard BCD code
0	0000
1	0001
10	0001 0000
100	0001 0000 0000
1234	0001 0010 0011 0100

- In BCD code a decimal digit is represented by four binary bits.
- If there are two or more than two digits in a decimal number, each decimal digit is represented by four binary bits.
- Several BCD codes are there such as 2,4,2,1 BCD code, excess-3 BCD code etc.

- In the standard BCD code the weights of the binary bits are 8,4,2,1.
- The standard BCD code does not represent 1010 to 1111 i.e 10 to 15(in decimal)
- BCD codes are widely used with instruments and calculators.

	c	Convert the following decimal numbers into BCD and calculate the value by adding them: 24 and 37	[L2][CO6]	**[5M]**

Answer:

RESULT – 5M

	Decimal Number	Equivalent BCD code
1st number	**24**	**0010 0100**
2nd number	**37**	**0011 0111**
Addition	**61**	**0101 1011 (invalid BCD)** **+ 0110 (+6)** **0110 0001 (valid BCD)**

Since the BCD range is only from 0000(0) to 1001(9). To obtain valid BCD, 0110(+6) has to be added to the desired nibble.

2	**a**	Define one's complement, two's complement form and explain the relation between them.	[L3][CO6]	**[6M]**

Answer:

1's COMPLEMENT EXPLANATION – 2M
2's COMPLEMENT EXPLANATION – 2M
EXAPMPLES – 2M

1's complement Representation:
The 1's complement of a number is obtained by complementing all the bits of signed binary number. So, 1's complement of positive number gives a negative number. Similarly, 1's complement of negative number gives a positive number. That means, if you perform two times 1's complement of a binary number including sign bit, then you will get the original signed binary number.

Example Consider the negative decimal number -108. The magnitude of this number is 108. We know the signed binary representation of 108 is 01101100. It is having 8 bits. The MSB of this number is zero, which indicates positive number. Complement of zero is one and vice-versa. So, replace zeros by ones and ones by zeros in order to get the negative number. $-10810 = 100100112$

2's Complement Representation
This code is the most widely used code for integer computation. Positive numbers are represented by unsigned binary numbers. Negative numbers are formed by the following procedure.

Example: To produce the number -5:

1. Write the binary number for 5 0101
2. Take the complement of it 1010
3. Add 1 1011

This code has the advantage that arithmetic operations are straightforward. Subtraction is accomplished by binary addition of the 2's complement.

	b	Calculate the 16-bit 1's and 2's complements of the following binary numbers. (i). 10000 (ii). 100111100001001 (iii). 0100111000100100	[L3][CO6]	**[6M]**

Answer:

EACH RESULT – 2M

	Given Binary Number	1's complement	2's complement
i	10000	01111	01111 + 1 10000

	Given Binary Number	1's complement	2's complement
ii	100111100001001	011000011110110	011000011110110 + 1 011000011110111

	Given Binary Number	1's complement	2's complement
iii	0100111000100100	1011000111011011	1011000111011011 + 1 1011000111011100

3	**a**	Define nine's complement, ten's complement and explain the relation between them.	[L2][CO4]	**[6M]**

Answer:

9's Complement: To obtain the 9's Complement of a decimal number each digit of the number is subtracted from 9.
For example ,
The 9's Complement of 45 is (99 – 45) = 54 and
The 9's Complement of 523 is (999 – 523) = 476.

10's Complement: The 10's Complement of a decimal number can be obtained by adding the 1 to its 9's Complement result.

The relation between **9's complement & 10's complement** is

The 10's Complement of a decimal number = its 9's Complement + 1

	b	Determine the result for the following decimal numbers operation by performing addition and convert each result to five-digit 10's complementary form, (i) 24379 5098 (ii) 24379 −5098 (iii) −24379 5098	[L3][CO4]	**[6M]**

Answer:

EACH RESULT – 2M

*For a **Positive number**, the 10's complement is the same as the number.*
*For a **Negative number**, the 10's complement is its 9's Complement + 1*

Addition in 10's complement is done using the following method:

- Add the two numbers
- Carry beyond the specified number of digits is dropped/discarded/ignored.

i) Given decimal number	9's complementary form	10's complementary form
24379	24379	24379
5098	5098	05098
	Addition	29477
5 digit 10's complement form of result **29477**		**70523**

ii) Given decimal number	9's complementary form	10's complementary form
24379	24379	24379
-5098	99999-05098 94901	94902(94901 + 1)
	Addition	119281 **19281**(discard carry)
5 digit 10's complement form of result **19281**		**80719**

iii) Given decimal number	9's complementary form	10's complementary form
-24379	99999-24379 75620	75621(75620 +1)
5098	5098	05098
	Addition	**80719**
5 digit 10's complement form of result **80719**		**19281**

4	**a**	Explain the procedure for adding two numbers in 2's complement form. As an example, convert +38 and −24 to 8-bit 2's complement form and add them.	[L2][CO6]	**[9M]**

Answer:

PROCEDURE – 4M
RESULT – 5M

Procedure for adding two numbers in 2's complement:

If decimal is *positive/negative*:

STEP 1: Convert the magnitudes of two decimal numbers to binary.

STEP 2: Pad 0's to the binary number to obtain desired bit size(8-bits).

STEP 3: Invert bits to achieve 1's-complement (for both the binary numbers).

STEP 4: Add 1 to the result of one's complement to achieve 2's-complement number. (for both the binary numbers).

STEP 5: Thus, now adding two numbers in 2's complement.

Given Decimal number		Its equivalent binary number	Its equivalent 1's complementary form	Its equivalent 2's complementary form
STEP 1	+ 38	0 100110		
	- 24	1 11000		
STEP 2	+ 38	0 **0100110 (8-bits)**		
	- 24	1 **0011000 (8-bits)**		
STEP 3	+ 38		0 1011001	
	- 24		1 1100111	
STEP 4	+ 38			0 1011010
	- 24			1 1101000
STEP 5	+ 38	Adding two numbers in 2's complement		**0 1000010**
	- 24			

	b	Determine the 9's complementary representation for the three-digit number −467.	[L3][CO4]	**[3M]**

Answer:

RESULT-3M

To obtain the 9's Complement of a decimal number, each digit of the number is subtracted from 9.

The number is negative so complement the number = 999 – 467
=532

5	**a**	Calculate the result by performing addition of the following two floating point numbers and round the result to five places of precision. i) 05199520 ii) 625.2035 iii) 1024.775E2 +04967850 +25.7585 +512.225E0	[L3][CO4]	**[6M]**

Answer:

EACH RESULT – 2M

STEP 1: write the given numbers in scientific form or exponential form.

STEP 2: For addition shift the decimal point either left or right to make equal exponent value for both the numbers.

STEP 3: Now add the two floating point numbers. Normalize the result if required.

STEP 4: Normalize the result in the format **$0.mmmm \times 10^{E}$,**

Where m is mantissa and no leading zero/s should be there and E is exponent. Moreover, the integer part cannot be a non-zero.

STEP 5: Rounding the result to five places of precision.

i) Given decimal number	Exponential form	Normalized/floating-point form
05199520	5199520×10^0	0.5199520×10^7
04967850	4967850×10^0	0.4967850×10^7
	Addition	**1.016737×10^7**
	Normalized result	**0.1016737×10^8**
Rounding the result to five places of precision		**0.10167×10^8**

ii) Given decimal number	Exponential form	Normalized /floating-point form
625.2035	625.2035×10^0	0.6252035×10^3
+25.7585	25.7585×10^0	0.257585×10^2 (E is not same as above) 0.02575850×10^3
	Addition	**0.650962×10^3**
Rounding the result to five places of precision		**0.65096×10^3**

iii) Given decimal number	Exponential form	Normalized /floating-point form
1024.775E2	1024.775×10^2	0.1024775×10^6
+512.225E0	512.225×10^0	0.512225×10^3 (E is not same as above) 0.000512225×10^6
	Addition	**0.102989725×10^6**
Rounding the result to five places of precision		**0.10299×10^6**

b	Calculate the result by performing subtraction of the following two floating point numbers and round the result to five places of precision. i) 05199520 -03967850 ii) 625.2035 -25.7585 iii) 7024.775E2 -512.225E0	[L2][CO4]	**[6M]**

Answer:

EACH RESULT – 2M

STEP 1: write the given numbers in scientific form or exponential form.

STEP 2: For subtraction shift the decimal point either left or right to make equal exponent value for both the numbers.

STEP 3: Now subtract the two floating point numbers. Normalize the result if required.

STEP 4: Normalize the result in the format **$0.mmmm \times 10^E$,** Where m is mantissa and no leading zero/s should be there and E is exponent. Moreover, the integer part cannot be a non-zero.

STEP 5: Rounding the result to five places of precision.

i) Given decimal number	Exponential form	Normalized/ floating-point form
05199520	5199520×10^0	0.5199520×10^7
-03967850	3967850×10^0	0.3967850×10^7
	Subtraction	**0.123167×10^7**
Rounding the result to five places of precision		**0.12317×10^7**

ii) Given decimal number	Exponential form	Normalized /floating-point form
625.2035	625.2035 X 10^0	0.6252035 X 10^3
-25.7585	25.7585 X 10^0	0.257585 X 10^2 (E is not same as above) 0.02575850 X 10^3
	Subtraction	**0.599445 X 10^3**
Rounding the result to five places of precision		**0.59944 X 10^3**

iii) Given decimal number	Exponential form	Normalized /floating-point form
7024.775E2	7024.775 X 10^2	0.7024775 X 10^6
-512.225E0	**512.225** X 10^0	0.512225 X 10^3(E is not same as above) 0.000512225 X 10^6
	Subtraction	**0.701965275 X 10^6**
Rounding the result to five places of precision		**0.70196 X 10^6**

6

a Determine the 16-bit 2's complementary binary representation for the decimal numbers 2021 and -2021 [L3][CO6] **[4M]**

Answer:

EACH RESULT – 2M

The range of numbers in 2's complement is **– (2^{n-1}) to 2^{n-1} - 1**

S	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
sign	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Fig:16-bit binary number

For the number 2021

Given decimal number	Binary form	16-bit 2's complementary form
2001	111 1101 0001	1's complementary form + 1 **0**111 1000 0010 1110 + 1
		0111 1000 0010 1111

For the number - 2021

Given decimal number	Binary form	16-bit 2's complementary form
-2001	111 1101 0001	1's complementary form + 1 **1**111 1000 0010 1110 + 1
		1111 1000 0010 1111

b Describe the exponential notation with an example. [L2][CO4] **[8M]**

Answer:

EXPLANATION-4M
EXAMPLE-2M

- ✓ A real number consists of two parts, an integer part and a fraction.
- ✓ The real decimal number can also be represented in a **scientific form** or **exponential form** as **M X R^e**.
- ✓ Where M represents Mantissa which may be either a fraction or an integer.
- ✓ R is the radix of the number system used. **e** represents exponent.
- ✓ A real or floating point number is represented in a computer in two parts; the first part represents a signed fixed-point number called **Mantissa**.
- ✓ The second part indicates the position of the decimal (binary) point, called **exponent.**

✓ **For example i)** a decimal number **5834.876** is represented in floating-point representation as given below

0.5834876×10^4

Sign 0	.5834876 **Mantissa**	**Sign** 0	04 **Exponent**

For example ii) a decimal number **0.0058346** is 0.58346×10^{-2}

Sign 0	.5834600 **Mantissa**	**Sign** 1	02 **Exponent**

✓ M and e are physically present in the registers. The radix R and the radix point (decimal or binary point) are not present in the register. These are assumed things.

7	**a**	Compute the floating-point representation for 0.0000019557.	[L3][CO4]	**[3M]**

Answer:

RESULT – 3M

A Decimal number **0.0000019557** is represented in floating-point representation as given below
0.0000019557 can also be represented as **0.19557×10^{-5}**

Sign 0	.19557 **Mantissa**	**Sign** 1	05 **Exponent**

	b	Compute division of the following two numbers, normalize the result obtained and round it to 3-bit. i) 04220000 / 02712500 ii) 625.2035 /25.7585 iii) 7024.775E2/512.225E0	[L3][CO5]	**[9M]**

Answer:

EACH RESULT – 3M

STEP 1: write the given numbers in scientific form or exponential form.

STEP 2: Now Divide the two floating point numbers. Normalize the result if required.

STEP 3: Normalize the result in the format **$0.mmmm \times 10^E$,**

Where m is mantissa and no leading zero/s should be there and E is exponent. Moreover, the integer part cannot be a non-zero.

STEP 4: Rounding the result to three places of precision.

i) Given decimal number	Exponential form	Normalized /floating-point form
04220000 (Dividend)	4220000 X 10^0	0.4220000 X 10^7
02712500 (Divisor)	2712500 X 10^0	0.2712500 X 10^3
	Division result	**0.55576037 X 10^4**
Rounding the result to three places of precision		**0.556 X 10^4**

ii) Given decimal number	Exponential form	Normalized /floating-point form
625.2035 (Dividend)	**625.2035** X 10^0	0.6252035 X 10^3
25.7585 (Divisor)	**25.7585** X 10^0	0.257585 X 10^2
	Division result	**2.42717355 X 10^1**
	Normalized result	**0.242717355 X 10^2**
Rounding the result to three places of precision		**0.243 X 10^2**

ii) Given decimal number	Exponential form	Normalized /floating-point form
7024.775E2 (Dividend)	**7024.775** X 10^2	0.7024775 X 10^6
512.225E0 (Divisor)	**512.225** X 10^0	0.512225 X 10^3
	Division result	**1.37142369 X 10^3**
	Normalized result	**0.137142369 X 10^4**
Rounding the result to three places of precision		**0.137 X 10^4**

e

a Represent the decimal number 171.625 in IEEE 754 format. [L2][CO4] **[3M]**

Answer:

REPRESENTATION – 3M

Given decimal number = 171.625

STEP 1: Separate Integer part and fractional part of the given decimal number

Integer part = 171

Fractional part = 0.625

STEP 2: Convert the Integer part into binary, using division method

division = quotient + **remainder**;
171 ÷ 2 = 85 + **1**;
85 ÷ 2 = 42 + **1**;
42 ÷ 2 = 21 + **0**;
21 ÷ 2 = 10 + **1**;
10 ÷ 2 = 5 + **0**;
5 ÷ 2 = 2 + **1**;
2 ÷ 2 = 1 + **0**;
1 ÷ 2 = 0 + **1**;

$(171)_{10} = (10101011)_2$

STEP 3: Convert the fractional part into binary, using multiplication method

Multiply it repeatedly by 2.
Keep track of each integer part of the results.
Stop when we get a fractional part that is equal to zero.

#) multiplying = integer + fractional part;
1) 0.625 × 2 **= 1** + 0.25;
2) 0.25 × 2 = **0** + 0.5;
3) 0.5 × 2 **= 1** + 0;

$(0.625)_{10} = (0.101)_2$

STEP 4: Write the representation of given decimal number in binary form

$(171.625)_{10} = (10101011.101)_2$

STEP 5: Normalize the binary representation of the number.

$(171.625)_{10} = (10101011.101)_2$
$= 1.0101011101 \times 2^7$

The above is in the form of IEEE 32-bit format

$\pm 1.M \times 2^{E-127} = 1.0101011101 \times 2^7$

M = 0101011101

Mantissa has to be normalized to 23 bits

Hence, **M** = 010 1011 1010 0000 0000 0000

SIGN OF MANTISSA = 0 (positive)

E-127 = 7 => E = 127+7 => E = $(134)_{10} = (10000110)_2$

STEP 6: The three elements that make up the number's 32 bit single precision IEEE 754 binary floating point representation:

Sign (1 bit) = 0 (a positive number)

Exponent (8 bits) = 1000 0110

Mantissa (23 bits) = 010 1011 1010 0000 0000 0000

Number 171.625 converted from decimal system (base 10) to 32 bit single precision IEEE 754 binary floating point:
0 - 1000 0110 - 010 1011 1010 0000 0000 0000

	b	Convert the decimal number 253.75 to binary floating point form.	[L2][CO6]	**[3M]**

Answer:

RESULT – 3M

Given decimal number = 253.75

STEP 1: Separate Integer part and fractional part of the given decimal number

Integer part = 253

Fractional part = 0.75

STEP 2: Convert the Integer part into binary, using division method

division = quotient + **remainder**;
253 ÷ 2 = 126 + **1**;
126 ÷ 2 = 63 + **0**;
63 ÷ 2 = 31 + **1**;
31 ÷ 2 = 15 + **1**;
15 ÷ 2 = 7 + **1**;
7 ÷ 2 = 3 + **1**;
3 ÷ 2 = 1 + **1**;
1 ÷ 2 = 0 + **1**;

$(253)_{10} = (11111101)_2$

STEP 3: Convert the fractional part into binary, using multiplication method

Multiply it repeatedly by 2.
Keep track of each integer part of the results.
Stop when we get a fractional part that is equal to zero.

#) multiplying = integer + fractional part;
1) 0.75 × 2 **= 1** + 0.50;
2) 0.50 × 2 = **1** + 0.0;

$(0.75)_{10} = (0.11)_2$

STEP 4: Write the representation of given decimal number in binary form

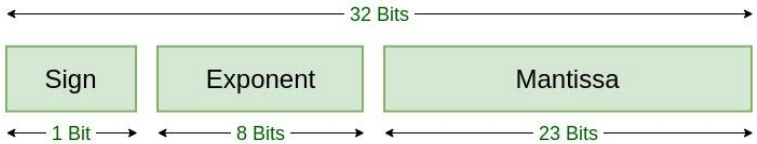
$(253.75)_{10} = (11111101.11)_2$

	c	The IEEE provides a standard 32-bit format for floating point numbers. The format for a number is specified as ±1.M × 2E − 127. Explain each part of this format.	[L2][CO4]	**[6M]**

Answer:

EXPLANATION – 3M
DIAGRAM -3M

IEEE floating point representation for binary real numbers consists of three parts. For a 32-bit (called single precision) number, they are:



Single Precision
IEEE 754 Floating-Point Standard

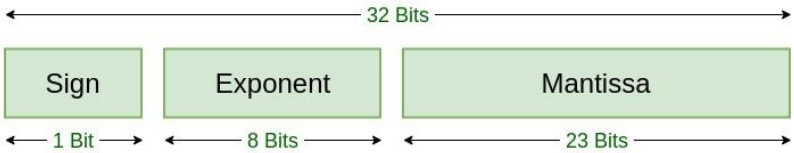
1. Sign, for which 1 bit is allocated.
2. Mantissa (called significand in the standard) is allocated 23 bits.
3. Exponent is allocated 8 bits. As both positive and negative numbers are required for the exponent, instead of using a separate sign bit for the exponent, the standard uses a biased representation. The value of the bias is 127. Thus an exponent 0 means that –127 is stored in the exponent field. A stored value 198 means that the exponent value is (198 – 127) = 71.

9

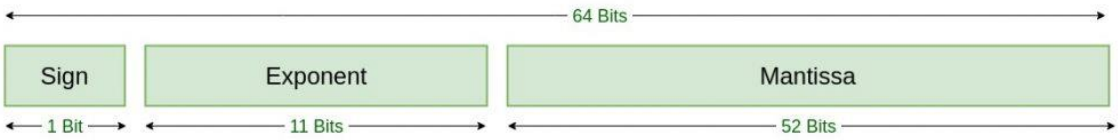
a	Illustrate the structure of Typical 32-bit & 64-bit Floating Point Format.	[L2][CO4]	**[3M]**

Answer:

EACH STRUCTURE – 1.5 M



Single Precision
IEEE 754 Floating-Point Standard



Double Precision
IEEE 754 Floating-Point Standard

b	Briefly explain about IEEE 754 Standard.	[L2][CO4]	**[6M]**

Answer:

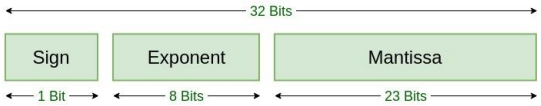
EXPLANATION – 3M
DIAGRAM - 3M

The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is a technical standard for floating-point computation which was established in 1985 by the Institute of Electrical and Electronics Engineers (IEEE). The standard addressed many problems found in the diverse floating point implementations that made them difficult to use reliably and reduced their portability. IEEE Standard 754 floating point is the most common representation today for real numbers on computers, including Intel-based PC's, Macs, and most Unix platforms.

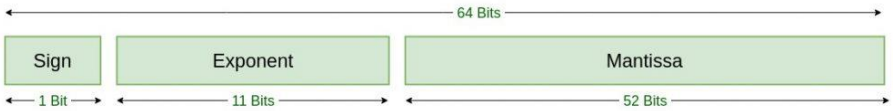
There are several ways to represent floating point number but IEEE 754 is the most efficient in most cases. IEEE 754 has 3 basic components:

1. The Sign of Mantissa –
   This is as simple as the name. 0 represents a positive number while 1 represents a negative number.
2. The Biased exponent –
   The exponent field needs to represent both positive and negative exponents. A bias is added to the actual exponent in order to get the stored exponent.
3. The Normalised Mantissa –
   The mantissa is part of a number in scientific notation or a floating-point number, consisting of its significant digits. Here we have only 2 digits, i.e. O and 1. So a normalised mantissa is one with only one 1 to the left of the decimal.

IEEE 754 numbers are divided into two based on the above three components: single precision and double precision.



Single Precision
IEEE 754 Floating-Point Standard



Double Precision
IEEE 754 Floating-Point Standard

c	Convert the decimal number 253.75 to 32-bit IEEE 754 floating-point form.	[L3][CO4]	**[3M]**

Answer:

RESULT – 3M

Given decimal number = 253.75

STEP 1: Separate Integer part and fractional part of the given decimal number

Integer part = 253 and Fractional part = 0.75

STEP 2: Convert the Integer part into binary, using division method

division = quotient + **remainder**;
253 ÷ 2 = 126 + **1**;
126 ÷ 2 = 63 + **0**;
63 ÷ 2 = 31 + **1**;
31 ÷ 2 = 15 + **1**;
15 ÷ 2 = 7 + **1**;
7 ÷ 2 = 3 + **1**;
3 ÷ 2 = 1 + **1**;
1 ÷ 2 = 0 + **1**;

$(253)_{10} = (11111101)_2$

STEP 3: Convert the fractional part into binary, using multiplication method

Multiply it repeatedly by 2.
Keep track of each integer part of the results.
Stop when we get a fractional part that is equal to zero.

#) multiplying = integer + fractional part;
1) 0.75 × 2 **= 1** + 0.50;
2) 0.50 × 2 = **1** + 0.0;

$(0.75)_{10} = (0.11)_2$

STEP 4: Write the representation of given decimal number in binary form

$(253.75)_{10} = (11111101.11)_2$

STEP 5: Normalize the binary representation of the number.

$(253.75)_{10} = (11111101.11)_2$
$= 1.\ 111110111 \times 2^7$

The above is in the form of IEEE 32-bit format

$\pm 1.M \times 2^{E-127} = 1.\ 111110111 \times 2^7$

M = 111110111

Mantissa has to be normalized to 23 bits

Hence, **M** = 1 1111 0111 0000 0000 0000

SIGN OF MANTISSA = 0 (positive)

E-127 = 7 => E = 127+7 => E = $(134)_{10} = (10000110)_2$

STEP 6: The three elements that make up the number's 32 bit single precision IEEE 754 binary floating point representation:

Sign (1 bit) = 0 (a positive number)

Exponent (8 bits) = 1000 0110

Mantissa (23 bits) = 1 1111 0111 0000 0000 0000

Number 253.75 converted from decimal system (base 10) to 32 bit single precision IEEE 754 binary floating point:
0 - 1000 0110 - 1 1111 0111 0000 0000 0000

10	Determine the result of multiplying two floating point numbers, normalize and round the result to 3-digit. **i)** 05220000 ×04712500 ii) 625.2035 x25.7585 iii) 7024.775E2 x512.225E0	[L3][CO4]	**[12M]**

Answer:

EACH RESULT – 4M

STEP 1: write the given numbers in scientific form or exponential form.

STEP 2: For Multiplication shift the decimal point either left or right to make equal exponent value for both the numbers.

STEP 3: Now multiply the two floating point numbers. Normalize the result if required.

STEP 4: Normalize the result in the format **0.mmmm X 10^E,**

Where m is mantissa, and no leading zero/s should be there and E is exponent. Moreover, the integer part cannot be a non-zero.

STEP 5: Rounding the result to three places of precision.

i) Given decimal number	**Exponential form**	**Normalized/floating-point form**
05220000	522 X 10^4	0.522 X 10^7
04712500	47125 X 10^2	0.47125 X 10^7
Multiplication		**0.2459925 X 10^{14}**
Rounding the result to five places of precision		**0.245 X 10^{14}**

ii) Given decimal number	**Exponential form**	**Normalized /floating-point form**
625.2035	625.2035 X 10^0	0.6252035 X 10^3
25.7585	25.7585 X 10^0	0.257585 X 10^2 (E is not same as above) 0.02575850 X 10^3
Multiplication		**0.0161043043547 X 10^6**
Normalized result		**0.161043043547 X 10^5**
Rounding the result to five places of precision		**0.161 X 10^5**

iii) Given decimal number	**Exponential form**	**Normalized /floating-point form**
7024.775E2	7024.775 X 10^2	0.7024775 X 10^6
512.225E0	512.225 X 10^0	0.512225 X 10^3 (E is not same as above) 0.000512225 X 10^6
Addition		**0.0003598265374 X 10^{12}**
Normalized result		**0. 3598265374 X 10^9**
Rounding the result to three places of precision		**0.359 X 10^9**

ALL THE BEST FOR YOUR EXAMINATIONS