FLOATING POINT REPRESENTATION

Papers Dock

COMPUTER SCIENCE 9618 PAPER 3

FLOATING POINT REPRESENTATION

Floating point representation is a method used in computers to represent real numbers

14.25 15.75 192.3125 -14.625

Mantissa And Exponent

In Math we use decimal numbers and if we need to write a very large or very small number we use the concept of Exponents.

For Example :

3.45 x 10 ^ 12

In the example above 3.45 is the Mantissa and 12 is the exponent with base 10 as the base for Decimal numbers is 10.

Mantissa : The Most Significant Digit Of FPN

Exponent : Power and because we are dealing with binary numbers so the base will be 2.

Note : We will be using a memorized list which will help us in solving the Questions.

(You are suppose to memorize it properly)

128	64	32	16	8	4	2	1	•	1/2	1/4	1/8	1/16	1/32

128	64	32	16	8	4	2	1	•	0.5	0.25	0.125	0.0625	0.03125

Each bit in a binary number represent a decimal value which helps in conversion of decimal number to binary numbers. Each bit represents an increasing power of 2 starting from the rightmost bit.

Question Example For Reference

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the normalised floating-point representation of +192.5 in this system. Show your working.

Mantissa **Exponent**

Working ....

....

....

....

.... [3]

There will be certain things which will be given in your Question. (No of bit for Mantissa and Exponent)

Case 1 : Positive Decimal Number To Binary Conversion

Question : You have 12 bits for Mantissa and 4 bits for Exponent. Represent 14.75 in Floating Point Representation

Step 1 : Write down the memorized list

128	64	32	16	8	4	2	1	•	0.5	0.25	0.125	0.0625	0.03125

Step 2 : Put 1 on numbers which will be added to fulfil Question Requirement

				1	1	1	0	•	1	1			
128	64	32	16	8	4	2	1	•	0.5	0.25	0.125	0.0625	0.03125

1 1 1 0 • 1 1

This representation is known as Fixed Point Representation

Step 3 : Move the decimal place all the way to the left and write the exponent value.

1 1 1 0 • 1 1

0 • 1 1 1 0 1 1

As we moved the decimal 4 places to the left the exponent value will be 4. If we move the decimal to left exponent increases and if we move exponent to the right the exponent decreases

Step 4 : Ignore decimal point and convert the exponent into binary form

0 • 1 1 1 0 1 1

0 1 1 1 0 1 1

The value of Exponent is 4 so in binary format it will be represented as 0100

				0	1	0	0
128	64	32	16	8	4	2	1

Now in this stage we have the mantissa value and the exponent value

0 • 1 1 1 0 1 1 X 2 ^ 4

As we know that the decimal will always be in between the first two binary bits and we also know that the base will always be 2 for binary numbers so we don't write the base value and the decimal number in our final answer.

0 1 1 1 0 1 1   0 1 0 0

Mantissa   Exponent

Step 5 : Store them in given spaces

Note : Always store Mantissa from Left Side and Always store exponent from Right Side and the empty boxes will be 0

0 1 1 1 0 1 1   0 1 0 0

Mantissa   Exponent

Mantissa   Exponent

0	1	1	1	0	1	1	0	0	0	0	0

0	1	0	0

Exam Style Question

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the normalised floating-point representation of +192.5 in this system. Show your working.

Mantissa   **Exponent**

Working ....

....

....

....

.... [3]

1(a)	= (0)11000000.1 (conversion to binary) = 0.110000001 × 2^8 (evidence of shifting binary point appropriately) = 0110000001 001000 (stored as mantissa and exponent)	[1] [1] [1]	3

3 In a computer system, real numbers are stored using normalised-floating point representation with:

- 8 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the normalised floating-point representation of + 21.75 in this system. Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

3(a)	**1** mark per bullet • 21.75 = 010101.11 (conversion to correct binary) • 0.1010111×2^5 (evidence of shifting binary point appropriately) • 01010111 0101 (stored as mantissa and exponent)	**3**

Case 2 : Negative Decimal Number To Binary Conversion

Question : You have 12 bits for Mantissa and 4 bits for Exponent. Represent -14.75 in Floating Point Representation

Step 1 : Ignore Negative sign and follow the steps of case 1 till step 4.

1 1 1 0 • 1 1

128	64	32	16	8	4	2	1	•	0.5	0.25	0.125	0.0625	0.03125

0 1 1 1 0 1 1 — Mantissa

0 1 0 0 — Exponent

Step 2 : Exponent would remain same and apply two's compliment method on mantissa only and store it.

Two's Compliment

Two's complement is a mathematical operation on binary numbers, and it is the most common method of representing signed integers in computers.

Steps to Find Two's Complement

1. **Invert All Bits: Change all 0s to 1s and all 1s to 0s.**
2. **Add One: Add 1 to the inverted binary number.**

0 1 1 1 0 1 1   0 1 0 0

Mantissa   Exponent

	0 1 1 1 0 1 1
One's Compliment	1 0 0 0 1 0 0
Two's Compliment	1
	1 0 0 0 1 0 1

Binary Addition

Addition	Sum	Carry
0 + 0	**0**	**0**
0 + 1	1	0
1 + 0	1	0
1 + 1	0	1
1 + 1 + 1	1	1

1 0 0 0 1 0 1 0 1 0 0

Mantissa Exponent

Mantissa Exponent

1	0	0	0	1	0	1	0	0	0	0	0

0	1	0	0

Exam Style Question

(b) Calculate the normalised floating-point representation of –192.5 in this system. Show your working.

Mantissa **Exponent**

Working ....

....

....

....

.... [3]

1(b)	1001111110 (one's complement of 10 bit mantissa) 1001111111 (two's complement of 10 bit mantissa) 1001111111 001000 (stored as mantissa and exponent)	[1] [1] [1]	**3**

1 **(a)** Real numbers are stored in a computer using floating point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and the exponent.

Write the normalised floating-point representation of –96.75 in this system.

Show your working.

Mantissa

Exponent

Working ....

....

....

....

....

....

[3]

1(a)	**One** mark per mark point (**Max 1**) • conversion of −96.75 to binary e.g., positive 96.75, flip the bits + 1 to give 10011111.01 // –128 + 16 + 8 + 4 + 2 + 1 + 0.25 / ¼ seen **One** mark per mark point (**Max 2**) • correct mantissa • correct exponent **Mantissa**: 1 0 0 1 1 1 1 1 0 1 **Exponent**: 0 0 0 1 1 1	**3**

Case 3 : Positive Binary Number to Denary

Note : You are just suppose to reverse the steps from case 1.

Calculate the exponent value and move the decimal to the original position and use the memorized list to find the denary value.

Mantissa

0	1	1	1	0	1	1	0	0	0	0	0

Exponent

0	1	0	0

Exponent

0	1	0	0
8	4	2	1

= 4

Mantissa

0 •	1	1	1	0	1	1	0	0	0	0	0

1 1 1 0 • 1 1

128	64	32	16	8	4	2	1	•	0.5	0.25	0.125	0.0625	0.03125

So after adding all the values with 1 we will get 14.75

Exam Style Question

1 In a particular computer system, real numbers are stored using floating-point representation, with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the denary value for the following floating-point number. Show your working.

Mantissa

0	1	0	1	0	0	0	0	0	0	0	0

Exponent

0	1	1	0

Working ....

....

....

....

Denary value ....

[3]

1(a)	Exponent = 6 (conversion of exponent to denary) 0.101 or 0.625 or 5/8 (value of mantissa) // moving of binary point 40 (answer)	**3**

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 8 bits for the mantissa, followed by
- 4 bits for the exponent

Two's complement form is used for both mantissa and exponent.

(a) **(i)** A real number is stored as the following 12-bit binary pattern:

0	1	1	0	1	0	0	0

0	0	1	1

Calculate the denary value of this number. Show your working.

....

....

....

....

....

....[3]

1 (a) (i) 01101000 0011
= <u>0.1101</u> (or <u>1/2 + 1/4 + 1/16</u>) × 2↑<u>3</u> [1+1]
= 110.1
= 6.5 [1]

Case 4 : Negative Binary Number to Denary

Note : You are just suppose to reverse the steps from case 2.

Calculate the exponent value and first apply two's compliment on mantissa and then move the decimal according to the exponent value and then use the memorized list to find the denary value.

Mantissa

1	0	0	0	1	0	1	0	0	0	0	0

Exponent

0	1	0	0

Mantissa

	1 0 0 0 1 0 1
One's Compliment	0 1 0 1 0 1 0
Two's Compliment	1
	0 1 1 1 0 1 1

0•1 1 1 0 1 1

1 1 1 0 • 1 1

128	64	32	16	8	4	2	1	•	0.5	0.25	0.125	0.0625	0.03125

So after adding all the values with 1 we will get 14.75 and as it was a negative number so the answer will be -14.75

Exam Style Question

1 In a computer system, real numbers are stored using normalised floating-point representation with:

- twelve bits for the mantissa
- four bits for the exponent.

The mantissa and exponent are both in two's complement form.

(a) Calculate the denary value for the following binary floating-point number.

Show your working.

Mantissa

1	0	0	1	0	1	1	1	0	0	1	1

Exponent

0	1	1	1

Working ....

....

....

....

....

Answer ....

[3]

1(a)	**2 marks** for working shown **1 mark** for the correct answer Working: • Correct calculation of <u>negative</u> value (any method) (= –0.11010001101) • Correctly moving the binary point 7 places (= –01101000.1101) // Exponent 7 Answer: • −104.8125 // $-104\frac{13}{16}$	**3**

(b) Calculate the denary value of the given binary floating-point number.
Show your working.

Mantissa

1	0	1	1	0	0	1	0	0	1	1

Exponent

0	1	0	0	1

Working ....

....

....

....

....

Answer ....

[3]

1(b)	**Two** marks for working • correct calculation of exponent seen • correct application of exponent to mantissa seen **One** mark for correct answer Working: = 1.0110010011 × 2^9 //exponent = 9 = 1011001001.1 (moving bp 9 places to right) // evaluate two's complement For example: –512 + 128 + 64 + 8 + 1 + 0.5 Answer: –310.5 // –310$^1/_2$

Case 5 : Negative Exponent Binary Conversion

Mantissa

1	0	1	1	0	0	0	0

Exponent

1	1	1	0

In this question see that both mantissa and exponent are negative as they both are starting with 1. So we need to apply two's compliment on both and then calculate.

Two's Compliment

Exponent

1	1	1	0

Exponent

1	1	1	0

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

So the exponent would be -2

Mantissa

1	0	1	1	0	0	0	0

Mantissa

1	0	1	1	0	0	0	0

$$
\begin{array}{cccccccc}
 & & & {}^{1} & {}^{1} & {}^{1} & {}^{1} & \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\
 & & & & & & & 1 \\
\hline
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0
\end{array}
$$

Mantissa — Exponent

0 • 1 0 1 0 0 0 0 — -2

0 • 0 0 1 0 1

128	64	32	16	8	4	2	1	•	0.5	0.25	0.125	0.0625	0.03125

So after moving the decimal two places to the left as the exponent was negative we 0.00101 and when we add the values with 1 we get 0.15625

Normalization

Normalization is a technique that is used to make your data more accurate.

0.1 Represents positive number

1.0 Represents negative number

How will we figure out that a binary representation is normalized

First and second bit should never be same

Mantissa

0	0	1	1	0	1	1	1

Exponent

0	1	0	1

0.356 x 10 ^6

If you need to make this in standard form then you would have to use one power from the exponent and the final answer would be 3.56 x 10^5. You are suppose to use the same technique to normalize Floating Point Representation.

Exam Style Question

(c) The given binary floating-point number is not normalised.

Normalise the floating-point number. Show your working.

Mantissa

0	0	0	0	0	0	0	1	1	1

Exponent

1	0	0	1	1	1

Mantissa

Exponent

Working ....

....

....

....

....

....

[3]

1(c)	**One** mark for working **One** mark for correct mantissa **One** mark for correct exponent Example answers Number of places added to exponent for normalisation −6 for number to retain its value // mantissa moved 6 places left Mantissa 0 1 1 1 0 0 0 0 0 0 Exponent 1 0 0 0 0 1	3

(c) A binary number is stored in the computer system.

Mantissa

0	0	0	1	1	1	1	0

Exponent

0	0	0	1	1	0	0	0

(i) State why the number is **not** normalised.

....

.... [1]

(ii) Write the normalised floating-point representation of the number.

Mantissa

Exponent

[2]

1(c)(i)	The mantissa does not begin with 01/10 (as its most significant bits) // the mantissa begins with 00 // first two digits are the same.
1(c)(ii)	**One** mark for each point: • Correct mantissa • Correct exponent **Mantissa**: 0 1 1 1 1 0 0 0 **Exponent**: 0 0 0 1 0 1 1 0

What problems could occur if the binary representation is not normalized ?

- Multiple representation of single number
- Precision lost
- Redundant leading zeros in mantissa

What are the problems in floating point representation ?

- 0.2 / 0.1 / 0.4 These numbers can not be exactly represented. The solution for this problem is Rounding which would cause rounding error.
- 0.2 has been represented by value greater than 0.2
- 0.4 has been represented by value greater than 0.4
- so after calculating with these rounded numbers the difference would increase and the difference will be significant.

Why Rounding Error occurs ?

- Because there is no exact representation for some binary numbers

Explain the reason why binary number are stored in normalized form ?

- Normalization minimizes the number of leading zeros
- Maximizing the precision of the number for the given number of bits
- enables very large or small number to be stored with accuracy.
- Avoids possibility of many numbers having multiple representation

Trade Off Between Mantissa And Exponent

Trade Off means Relationship

Mantissa													Exponent			
0	1	1	1	0	1	1	0	0	0	0	0		0	1	0	0

Precision

Range

12 bits for mantissa --> 8 bits for Mantissa

if we reduce bits for mantissa from 12 to 8 that means less precision

4 bits for Exponent --> 8 bits for Exponent

if we increase the bits for Exponent from 4 bits to 8 bits that means better range

Question : What is the trade-off between Mantissa and Exponent

- The trade off is between precision and range
- If more bits are used for mantissa that means better precision.
- If more bits are used for exponent that means better range.
- More no of bits for mantissa means less number of bits for exponent.

Largest Positive Number And Smallest Positive Number In A Given Scenario

You have to make sure the answer is normalized so for that we are going to have a Key which can help us in solving Question regarding largest or smallest number.

Largest Positive

0.1 and the rest of the bits should be 1 as we need to make a largest number

Mantissa

0.1 and the rest should also be 1 as we need the largest positive exponent which will give us the largest number

Exponent

0	1	1	1	1	1	1	1	1	1	1	1

Mantissa

0	1	1	1

Exponent

Smallest Positive

0.1 and the rest of the bits should be 0 as we need to make a smallest number so the mantissa should the smallest

1.0 and the rest should be 0 as we need the largest negative exponent which will move our decimal to left side and we will get the smallest number

Mantissa

Exponent

0	1	0	0	0	0	0	0	0	0	0	0

Mantissa

1	0	0	0

Exponent

Overflow

Less Space More Bits To Store

Underflow

More Space Less Bits To Store

Question : State when overflow error occurs in floating point representation ?

Following an arithmetic operation a number produced exceeds the maximum value that can be stored in mantissa and exponent an overflow error occurs. This could occur when dividing by a very small number.

Question : State when underflow error occurs in floating point representation ?

Following an arithmetic operation, the result is smaller than the smallest number that can be stored in mantissa and exponent an underflow error occurs. This could occur when dividing by a very large number.

Question : 10 bits for mantissa and 6 bits for exponent. The denary number 513 cannot be stored accurately as a normalized floating-point number is this system (3)

Answer:

513 in binary is 0.1000000001 so it requires 11 bits to store accurately. Results in overflow.

Question : Describe an alteration to the way floating-point numbers are stored to enable this number to be stored accurately using the total number of bits (2)

Answer : The number of bits for mantissa must be increased. 11 bits for mantissa and 5 bits for exponent

Question : Explain why a binary representation is sometimes only an approximation to the real number it represents.

Answer :

- Real numbers can have a fractional part (such as 0.4 and 0.25)
- The fixed length of the storage means that you can't store a very large number or very small number
- There are limited decimal/fractional representation (0.5 0.25 0.125 ..)
- it isn't possible to store all fractions with the level of precision provide by the system
- the fractional part of the number is as closes as possible within the number of bits given.

Floating Point Representation

Question 1

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the normalised floating-point representation of +192.5 in this system. Show your working.

Mantissa

Exponent

Working ....

....

....

....

.... [3]

(b) Calculate the normalised floating-point representation of –192.5 in this system. Show your working.

Mantissa

Exponent

Working ....

....

....

....

.... [3]

(c) The floating-point representation has changed. There are now 12 bits for the mantissa and 4 bits for the exponent as shown.

Mantissa **Exponent**

Explain why +192.5 cannot be accurately represented in this format.

....

....

....

....

....

.... [3]

Question 2

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) The following floating-point number stored is not normalised.

Calculate the denary value for the floating-point number. Show your working.

Mantissa **Exponent**

0	0	0	0	1	1	0	0	0	0	0	0		0	1	0	1

Working ....

....

....

....

Denary value ....

[3]

(b) **(i)** Normalise the floating-point number given in **part (a)**.

Write your answer in the following boxes.

Mantissa													Exponent			

[2]

(ii) Describe **one** problem that can occur when floating-point numbers are not normalised.

....

....

.... [2]

Question 3

1 In a particular computer system, real numbers are stored using floating-point representation, with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the denary value for the following floating-point number. Show your working.

Mantissa													Exponent			
0	1	0	1	0	0	0	0	0	0	0	0		0	1	1	0

Working ....

....

....

....

Denary value ....

[3]

(b) A new operating system has been installed that has changed the way the floating-point numbers are used. The order of the exponent and the mantissa are reversed.

(i) Calculate the new denary value for the following floating-point number that has the same bit pattern as the number in **part (a)**. Show your working.

Exponent

0	1	0	1

Mantissa

0	0	0	0	0	0	0	0	0	1	1	0

Working ....

....

....

....

Denary value .... [3]

(ii) Identify **two** problems that can occur due to the change in the representation of the floating-point number.

Problem 1 ....

....

Problem 2 ....

.... [2]

Question 4

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) The following floating-point number stored is not normalised.

Calculate the denary value for the floating-point number. Show your working.

Mantissa

0	0	0	0	1	1	0	0	0	0	0	0

Exponent

0	1	0	1

Working ....

....

....

....

Denary value .... [3]

(b) **(i)** Normalise the floating-point number given in **part (a)**.

Write your answer in the following boxes.

Mantissa													**Exponent**			

[2]

(ii) Describe **one** problem that can occur when floating-point numbers are not normalised.

....

....

.... [2]

Question 5

1 In a computer system, real numbers are stored using normalised floating-point representation with:

- twelve bits for the mantissa
- four bits for the exponent.

The mantissa and exponent are both in two's complement form.

(a) Calculate the denary value for the following binary floating-point number.

Show your working.

Mantissa

1	0	0	1	0	1	1	1	0	0	1	1

Exponent

0	1	1	1

Working ....

....

....

....

....

Answer ....
[3]

(b) Calculate the normalised floating-point representation of +1.5625 in this system.

Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

(c) **(i)** Write the largest positive number that can be stored as a normalised floating-point number using this format.

Mantissa

Exponent

[2]

(ii) Write the smallest non-zero positive number that can be stored as a normalised floating-point number using this format.

Mantissa

Exponent

[2]

(d) The developer of a new programming language decides that all real numbers will now be stored using 20-bit normalised floating-point representation. She must decide how many bits to use for the mantissa and how many bits for the exponent.

Explain the trade-off between using either a large number of bits for the mantissa, or a large number of bits for the exponent.

....

....

....

....

....

.... [3]

Question 6

1 **(a)** A computer stores real numbers using floating-point representation. The floating-point numbers have:

- eight bits for the mantissa
- four bits for the exponent.

The mantissa and exponent are both stored in two's complement format.

(i) Calculate the denary value of the following floating-point number.

Show your working.

Mantissa

0	0	1	1	0	1	1	1

Exponent

0	1	0	1

Working ....

....

....

....

....

Answer ....

[3]

(ii) State why the floating-point number in **part (a)(i)** is **not** normalised.

....

.... [1]

(iii) Give the floating-point number in **part (a)(i)** in normalised two's complement format.

Mantissa

Exponent

[2]

(b) (i) Convert the denary number +11.625 into a normalised floating-point number.

Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

(ii) Convert the denary number −11.625 into a normalised floating-point number.

Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

(c) A student enters the following into an interpreter:

```
OUTPUT(0.2 * 0.4)
```

The student is surprised to see that the interpreter outputs the following:

```
0.08000000000000002
```

Explain why the interpreter outputs this value.

....

....

....

....

....

.... [3]

Question 7

1 In a computer system, real numbers are stored using normalised floating-point representation with:

- twelve bits for the mantissa
- four bits for the exponent.

The mantissa and exponent are both in two's complement form.

(a) Calculate the denary value for the following binary floating-point number.

Show your working.

Mantissa

1	0	0	1	0	1	1	1	0	0	1	1

Exponent

0	1	1	1

Working ....

....

....

....

....

Answer ....

[3]

(b) Calculate the normalised floating-point representation of +1.5625 in this system.

Show your working.

Working ....

....

....

....

....

....

Mantissa **Exponent**

[3]

(c) (i) Write the largest positive number that can be stored as a normalised floating-point number using this format.

Mantissa **Exponent**

[2]

(ii) Write the smallest non-zero positive number that can be stored as a normalised floating-point number using this format.

Mantissa **Exponent**

[2]

(d) The developer of a new programming language decides that all real numbers will now be stored using 20-bit normalised floating-point representation. She must decide how many bits to use for the mantissa and how many bits for the exponent.

Explain the trade-off between using either a large number of bits for the mantissa, or a large number of bits for the exponent.

....

....

....

....

....

.... [3]

Question 8

1 Real numbers are stored using floating-point representation in a computer system.

This representation uses:

- 8 bits for the mantissa, followed by
- 4 bits for the exponent.

Two's complement form is used for both the mantissa and the exponent.

(a) **(i)** A real number is stored as a 12-bit normalised binary number as follows:

Mantissa

0	1	0	1	0	0	1	0

Exponent

0	0	1	0

Calculate the denary value for this binary number. Show your working.

Working ....

....

....

Denary value ....
[3]

(ii) Calculate the normalised binary number for –3.75. Show your working.

Mantissa

Exponent

Working ....

....

....

....
[3]

(b) The number of bits available to represent a real number is increased to 16.

State the effect of increasing the size of the exponent by 4 bits.

....

.... [1]

(c) State why some binary representations can lead to rounding errors.

....

.... [1]

(d) Complete the following descriptions by inserting the **two** missing terms.

.... can occur in the exponent of a floating-point number, when the exponent has become too large to be represented using the number of bits available.

A calculation results in a number so small that it cannot be represented by the number of bits available. This is called ....

[2]

Question 9

8 **(a)** The following 16-bit binary pattern represents a floating-point number stored in two's complement form. The twelve most significant bits are used for the mantissa and the four least significant bits are used for the exponent.

Most significant bit ↓ | **Least significant bit** ↓

0	1	1	1	0	0	0	0	0	0	0	0	1	1	0	1

(i) Identify the binary value of the exponent.

.... [1]

(ii) Identify the binary value of the mantissa.

.... [1]

(iii) State whether the number stored is positive or negative. Justify your choice.

Positive or negative ....

Justification ....

....

....

[2]

(iv) Convert the binary floating-point number in **part (a)** into denary. Show your working.

Working ....

....

....

....

....

Denary value ....

[3]

(b) The number of bits used for the exponent is increased to eight, and the number of bits used for the mantissa is decreased to eight.

State the effects of this change.

....

....

....

.... [2]

Question 10

1 In a computer system, real numbers are stored using normalised floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- Two's complement form for both mantissa and exponent.

(a) Find the denary value for the following binary floating-point number.

Mantissa

1	0	1	1	1	0	0	1	1	0	1	0

Exponent

0	1	0	1

Show your working.

Working ....

....

....

....

....

Answer ....

[3]

(b) Calculate the normalised floating-point representation of 5.25 in this system. Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

(c) The size of the mantissa is decreased and the size of the exponent is increased.

State how this affects the range and precision of the numbers that the computer system can represent.

....

....

....

....[2]

Question 11

3 In a computer system, real numbers are stored using normalised-floating point representation with:

- 8 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the normalised floating-point representation of + 21.75 in this system. Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

(b) Find the denary value for the following binary floating-point number.

Mantissa

1	0	1	1	0	0	0	0

Exponent

1	1	1	0

Show your working.

Working ....

....

....

....

....

Answer ....

[3]

Question 12

2 **(a)** A computer system stores real numbers using floating-point representation. The floating-point numbers have:

- eight bits for the mantissa
- four bits for the exponent.

The mantissa and exponent are both in two's complement form.

(i) Calculate the denary value of the following floating-point number.

Mantissa

0	0	1	1	1	0	0	0

Exponent

0	1	1	1

Show your working.

Working ....

....

....

....

....

Answer ....

[3]

(ii) State how you know the floating-point number in **part (a)(i)** is not normalised.

....

....[1]

(iii) Normalise the floating-point number in **part (a)(i)**.

Mantissa

Exponent

[2]

(b) (i) Write the largest positive number that this system can represent as a normalised floating-point number in this format.

Mantissa

Exponent

[2]

(ii) Write the smallest positive number that can be stored as a normalised floating-point number in this format.

Mantissa

Exponent

[2]

(c) The number of bits available to represent a real number is increased to 16.

State the effect this has on the numbers that can be represented, if the additional four bits are used in the:

(i) mantissa ....

....[1]

(ii) exponent ....

....[1]

(d) A student enters the following code into an interpreter.

```
X = 0.1
Y = 0.2
Z = 0.3
OUTPUT (X + Y + Z)
```

The student is surprised to see the output:

```
0.6000000000000001
```

Explain why this is output.

....

....

....

....

....

....[3]

Question 13

1 **(a)** A computer system uses floating-point representation to store real numbers. The floating-point numbers have:

- 8 bits for the mantissa
- 8 bits for the exponent

The mantissa and exponent are both in two's complement form.

(i) Calculate the denary value of the following floating-point number. It is **not** in normalised form.

Mantissa

0	0	1	0	1	0	1	0

Exponent

0	0	0	0	0	1	0	1

Show your working.

Working ....

....

....

....

....

Answer .... [3]

(ii) Convert the denary number +7.5 into a normalised floating-point number.

Show your working.

Mantissa

Exponent

Working ....

....

....

....

.... [3]

(iii) Convert the denary number –7.5 into a normalised floating-point number.

Show your working.

Mantissa | **Exponent**

Working ....

....

....

....

....

[3]

(b) A normalised floating-point number is shown.

Mantissa

0	1	1	1	1	1	1	1

Exponent

0	1	1	1	1	1	1	1

(i) State the significance of this binary number.

....

....[1]

(ii) State what will happen if a positive number is added to this number.

....

....[1]

Question 14

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent

(a) Calculate the floating-point representation of +2.5 in this system. Show your working.

Mantissa													Exponent			

....

....

....

....

....

.... [3]

(b) Calculate the floating-point representation of −2.5 in this system. Show your working.

Mantissa													Exponent			

....

....

....

....

....

.... [3]

(c) Find the denary value for the following binary floating-point number. Show your working.

Mantissa													Exponent			
0	● 0	1	1	0	0	0	0	0	0	0	0		0	0	1	1

....

....

....

....

....

.... [3]

(d) **(i)** State whether the floating-point number given in **part (c)** is normalised or not normalised.

.... [1]

(ii) Justify your answer given in **part (d)(i)**.

....

.... [1]

(e) The system changes so that it now allocates 8 bits to both the mantissa and the exponent.

State **two** effects this has on the numbers that can be represented.

1 ....

....

2 ....

.... [2]

Question 15

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 8 bits for the mantissa
- 8 bits for the exponent
- two's complement form for both mantissa and exponent

(a) Calculate the floating point representation of +3.5 in this system. Show your working.

Mantissa

Exponent

....

....

....

....

....

.... [3]

(b) Calculate the floating-point representation of –3.5 in this system. Show your working.

Mantissa

Exponent

....

....

....

....

....

.... [3]

(c) Find the denary value for the following binary floating-point number. Show your working.

Mantissa							
0 ●	1	1	1	0	0	0	0

Exponent							
0	0	0	0	0	1	0	0

....

....

....

....

....

.... [3]

(d) (i) State whether the floating-point number given in **part (c)** is normalised or not normalised.

.... [1]

(ii) Justify your answer given in **part (d)(i)**.

....

.... [1]

(e) Give the binary two's complement pattern for the negative number with the largest magnitude.

Mantissa							
●							

Exponent							

[2]

Question 16

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 8 bits for the mantissa, followed by
- 8 bits for the exponent

Two's complement form is used for both mantissa and exponent.

(a) **(i)** A real number is stored as the following two bytes:

Mantissa

0	0	1	0	1	0	0	0

Exponent

0	0	0	0	0	0	1	1

Calculate the denary value of this number. Show your working.

....

....

....

....

....

....[3]

(ii) Explain why the floating-point number in **part (a)(i)** is not normalised.

....

....[2]

(iii) Normalise the floating-point number in **part (a)(i)**.

Mantissa

Exponent

[2]

(b) **(i)** Write the largest positive number that can be written as a normalised floating-point number in this format.

Mantissa

Exponent

[2]

(ii) Write the smallest positive number that can be written as a normalised floating-point number in this format.

Mantissa

Exponent

[2]

(iii) If a positive number is added to the number in **part (b)(i)** explain what will happen.

....

....

....

....[2]

(c) A student writes a program to output numbers using the following code:

```
X ← 0.0
FOR i ← 0 TO 1000
   X ← X + 0.1
   OUTPUT X
ENDFOR
```

The student is surprised to see that the program outputs the following sequence:

0.0 0.1 0.2 0.2999999 0.3999999 ……

Explain why this output has occurred.

....

....

....

....

....

....[3]

Question 17

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 8 bits for the mantissa, followed by
- 4 bits for the exponent

Two's complement form is used for both mantissa and exponent.

(a) (i) A real number is stored as the following 12-bit binary pattern:

0	1	1	0	1	0	0	0		0	0	1	1

Calculate the denary value of this number. Show your working.

....

....

....

....

....

....[3]

(ii) Give the normalised binary pattern for +3.5. Show your working.

....

....

....

....

....

....[3]

(iii) Give the normalised binary pattern for –3.5. Show your working.

....

....

....

....

....

....[3]

The number of bits available to represent a real number is increased to 16.

(b) **(i)** If the system were to use the extra 4 bits for the mantissa, state what the effect would be on the numbers that can be represented.

....

....[1]

(ii) If the system were to use the extra 4 bits for the exponent instead, state what the effect would be on the numbers that can be represented.

....

....[1]

(c) A student enters the following expression into an interpreter:

```
OUTPUT (0.1 + 0.2)
```

The student is surprised to see the following output:

0.30000000000000001

Explain why this output has occurred.

....

....

....

....

....

....[3]

Question 18

1 Real numbers are stored in a computer system using floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- Two's complement form for both the mantissa and the exponent.

(a) Calculate the normalised floating-point representation of –7.25 in this system.
Show your working.

Mantissa

Exponent

Working ....

....

....

....

....

....

....

.... [3]

(b) Calculate the denary value of the given binary floating-point number.
Show your working.

Mantissa

1	0	1	1	0	0	0	1	1	1

Exponent

0	0	0	1	1	1

Working ....

....

....

....

....

Answer .... [3]

(c) The given binary floating-point number is not normalised.

Normalise the floating-point number. Show your working.

Mantissa / **Exponent**

0	0	0	0	0	0	0	1	1	1	1	0	0	1	1	1

Mantissa / **Exponent**

Working ....

....

....

....

....

....

[3]

(d) The denary number 513 cannot be stored accurately as a normalised floating-point number in this computer system.

(i) Explain the reason for this.

....

....

....

....

....

.... [3]

(ii) Describe an alteration to the way floating-point numbers are stored to enable this number to be stored accurately using the same total number of bits.

....

....

....

.... [2]

Question 19

1 **(a)** Numbers are stored in a computer using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and exponent.

(i) Write the normalised floating-point representation of the following unsigned binary number using this system.

1011100.011001

Working ....

....

....

....

....

Mantissa **Exponent**

[2]

(ii) State the consequence of storing the binary number in **part (a)(i)** as a floating-point number in this system. Justify your answer.

Consequence ....

....

Justification ....

....

....

[2]

(b) Explain the reason why binary numbers are stored in normalised form.

....

....

....

....

....

....

.... [3]

Question 20

1 Real numbers are stored in a computer system using floating-point representation with:

- 8 bits for the mantissa
- 8 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Write the normalised floating-point representation of +202 in this system.
Show your working.

Mantissa									**Exponent**							

Working ....

....

....

....

....

....

[3]

(b) Write the normalised floating-point representation of –202 in this system.
Show your working.

Mantissa **Exponent**

Working ....

....

....

....

....

.... [3]

(c) A binary number is stored in the computer system.

Mantissa **Exponent**

0	0	0	1	1	1	1	0		0	0	0	1	1	0	0	0

(i) State why the number is **not** normalised.

....

.... [1]

(ii) Write the normalised floating-point representation of the number.

Mantissa **Exponent**

[2]

Question 21

1 Normalised floating-point numbers are stored in a computer system using two's complement for both the mantissa and the exponent with:

- 11 bits for the mantissa
- 5 bits for the exponent.

(a) Write the largest positive two's complement binary number that can be stored in this system.

Mantissa

Exponent

[1]

(b) Calculate the denary value of the given binary floating-point number.
Show your working.

Mantissa

1	0	1	1	0	0	1	0	0	1	1

Exponent

0	1	0	0	1

Working ....

....

....

....

....

Answer ....

[3]

(c) State when underflow occurs in a binary floating-point system.

....

....

....

.... [2]

Question 22

1 Numbers are stored in two different computer systems by using floating-point representation.

System 1 uses:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and the exponent.

System 2 uses:

- 8 bits for the mantissa
- 8 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(a) Calculate the normalised floating-point representation of 113.75 **and** show how it would be represented in each of these two systems.

Show your working.

System 1

Mantissa

Exponent

System 2

Mantissa

Exponent

Working ....

....

....

....

....

....

[4]

(b) Explain the problem that occurred in **part (a)** when representing the number in **system 2**.

....

....

....

.... [2]

Question 23

1 Numbers are stored in a computer using floating point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and exponent.

(a) Write the normalised floating-point representation of the following binary number using this system:

0101010.111

Show your working.

Working ....

....

....

....

....

Mantissa **Exponent**

[2]

(b) Describe the reason why the normalised form of the following binary number cannot be represented accurately using this system.

0101011.111001

....

....

....

....

....

.... [3]

Question 24

1 Real numbers are stored in a computer using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and exponent.

(a) Write the normalised floating-point representation of +65.25 in this system.

Show your working.

Mantissa **Exponent**

Working ....

....

....

....

....

.... [3]

(b) Explain the problem that will occur in storing the normalised floating-point representation of +65.20 in this system.

....

....

....

.... [2]

Question 25

1 **(a)** Real numbers are stored in a computer using floating point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and the exponent.

Write the normalised floating-point representation of –96.75 in this system.

Show your working.

Mantissa											**Exponent**					

Working ....

....

....

....

....

....

[3]

(b) Explain why a binary representation is sometimes only an approximation to the real number it represents.

....

....

....

....

....

.... [3]

Answer

Answer 1

1(a)	= (0)11000000.1 (conversion to binary) [1] = 0.110000001 × 2^8 (evidence of shifting binary point appropriately) [1] = 0110000001 001000 (stored as mantissa and exponent) [1]	3
1(b)	1001111110 (one's complement of 10 bit mantissa) [1] 1001111111 (two's complement of 10 bit mantissa) [1] 1001111111 001000 (stored as mantissa and exponent) [1]	3
1(c)	Any **three** from: • Exponent too large to fit in 4 bits as a two's complement number • Exponent will turn negative/–8 • … therefore, point moves the wrong way • Value will be approx. +0.0029(296875)	3

Answer 2

1(a)	Exponent = 5 (conversion of exponent to denary) 0.00011 or 0.09375 or 3/32 (value of mantissa) //moving of binary point 3 (answer)	3
1(b)(i)	Mantissa = 011000000000 Exponent = 0010	2
1(b)(ii)	Any **two** from Precision lost Redundant leading zeros in the mantissa Bits lost off right hand end / least significant end Multiple representations of a single number	2

Answer 3

1(a)	Exponent = 6 (conversion of exponent to denary) 0.101 or 0.625 or 5/8 (value of mantissa) // moving of binary point 40 (answer)	3
1(b)(i)	Exponent = 5 (conversion of exponent to denary) 0.0000000110 or 3/1024 (value of mantissa) // moving of binary point 0.09375 or 3/32 (answer)	3
1(b)(ii)	Any **two** from The number calculated will change The same bit pattern is for a different number Software may crash (if not updated)	2

Answer 4

1(a)	Exponent = 5 (conversion of exponent to denary) 0.00011 or 0.09375 or 3/32 (value of mantissa) //moving of binary point 3 (answer)	**3**
1(b)(i)	Mantissa = 011000000000 Exponent = 0010	**2**
1(b)(ii)	Any **two** from Precision lost Redundant leading zeros in the mantissa Bits lost off right hand end / least significant end Multiple representations of a single number	**2**

Answer 5

1(a)	**2 marks** for working shown **1 mark** for the correct answer Working: ∞ Correct calculation of <u>negative</u> value (any method) (= –0.11010001101) ∞ Correctly moving the binary point 7 places (= –01101000.1101) // Exponent 7 Answer: ∞ -104.8125 // $-104\frac{13}{16}$	**3**
1(b)	**2 marks** for working shown **1 mark** for the correct answer Working: ∞ Correct conversion to binary (01.1001) ∞ Correct calculation of exponent (1) Answer: ∞ (Mantissa) 0110 0100 0000 (Exponent) 0001	**3**
1(c)(i)	**1 mark** per bullet point ∞ Mantissa = 0111 1111 1111 ∞ Exponent = 0111	**2**
1(c)(ii)	**1 mark** per bullet point ∞ Mantissa = 0100 0000 0000 ∞ Exponent = 1000	**2**
1(d)	**1 mark** per bullet point to **max 3** ∞ The trade-off is between range and precision ∞ Any increase in the number of bits for the mantissa, means fewer bits available for the exponent // Any decrease in the number of bits for the mantissa, means more bits available for the exponent ∞ More bits used for the mantissa will result in better precision ∞ More bits used for the exponent will result in a larger range of numbers ∞ Fewer bits used for the mantissa will result in worse precision ∞ Fewer bits used for the exponent will result in a smaller range of numbers	**3**

Answer 6

1(a)(i)	**2 marks** for working **1 mark** for correct answer Working: ∞ = 0. 0110111 x 2^5 places // exponent = 5 ∞ = 1101.11 (moving bp 5) Answer: ∞ = 13.75 // 13 ¾	**3**
1(a)(ii)	The first two bits of the mantissa are 0 / the same / not different / are not 01	**1**
1(a)(iii)	**1 mark** per bullet point ∞ Mantissa = 01101110 ∞ Exponent = 0100	**2**
1(b)(i)	**2 marks** for working **1 mark** for correct answer Working: ∞ 01011.101 ∞ 0.1011101 × 2^4 // showing calculation of exponent = 4 Answer: ∞ 01011101 0100	**3**
1(b)(ii)	**2 marks** for working **1 mark** for correct answer Working: ∞ 10100.011 // 10100011 correct use of two's complement or other method ∞ Exponent = 4 Answer: ∞ 10100011 0100	**3**
1(c)	**1 mark** per bullet point (max 3) ∞ <u>0.2/0.4</u> cannot be represented exactly in binary / rounding error ∞ 0.2 has been represented by a value just greater than 0.2 // 0.4 has been represented by a value just greater than 0.4 ∞ Therefore multiplying these two representations together increases the difference ∞ difference after the calculation is significant enough to be seen (given the number of positions after the decimal place)	**3**

Answer 7

1(a)	**2 marks** for working shown **1 mark** for the correct answer Working: ∞ Correct calculation of negative value (any method) (= –0.11010001101) ∞ Correctly moving the binary point 7 places (= –01101000.1101) // Exponent 7 Answer: ∞ -104.8125 // $-104\frac{13}{16}$	**3**
1(b)	**2 marks** for working shown **1 mark** for the correct answer Working: ∞ Correct conversion to binary (01.1001) ∞ Correct calculation of exponent (1) Answer: ∞ (Mantissa) 0110 0100 0000 (Exponent) 0001	**3**
1(c)(i)	**1 mark** per bullet point ∞ Mantissa = 0111 1111 1111 ∞ Exponent = 0111	**2**
1(c)(ii)	**1 mark** per bullet point ∞ Mantissa = 0100 0000 0000 ∞ Exponent = 1000	**2**
1(d)	**1 mark** per bullet point to **max 3** ∞ The trade-off is between range and precision ∞ Any increase in the number of bits for the mantissa, means fewer bits available for the exponent // Any decrease in the number of bits for the mantissa, means more bits available for the exponent ∞ More bits used for the mantissa will result in better precision ∞ More bits used for the exponent will result in a larger range of numbers ∞ Fewer bits used for the mantissa will result in worse precision ∞ Fewer bits used for the exponent will result in a smaller range of numbers	**3**

Answer 8

1(a)(i)	**1 mark** per bullet point • Exponent 0010 = 2 • Mantissa 0.1010010 becomes 010.10010 // $\frac{41}{64}$ // $2 + \frac{1}{2} + \frac{1}{16}$ • Answer $2\frac{9}{16}$ // 2.5625	**3**
1(a)(ii)	**1 mark** per bullet point • -3.75 = 100.01000 // $-4 + \frac{1}{4}$ / 0.25 • 100.01000 becomes 1.0001000 Exponent = +2 • Answer: Mantissa = 10001000 Exponent = 0010	**3**
1(b)	Only the range is increased (no effect on precision)	**1**
1(c)	**1 mark** per bullet point to **max 1** • There is no **exact** binary conversion for some numbers • More bits are needed to store the number than are available	**1**
1(d)	First term: Overflow Second term: Underflow	**2**

Answer 9

8(a)(i)	1101	**1**
8(a)(ii)	011100000000	**1**
8(a)(iii)	1 mark for positive, 1 for justification • Positive … • … the most significant / first bit in the mantissa is 0	**2**
8(a)(iv)	1 mark per bullet point • Exponent = 1011 = -3 // binary point moved 3 places left • Mantissa 0.111 becomes 0.000111 // ⅞ // ½ + ¼ + ⅛ // $2^{-1} + 2^{-2} + 2^{-3}$ • Answer: 7 / 64 // 0.109375	**3**
8(b)	1 mark per bullet point • Increases the range • Decreases the precision	**2**

Answer 10

1(a)	**1** mark per bullet max **2** ∞ 0101 = 5 (conversion of exponent to denary) ∞ 1.01110011010 = –0.10001100110 (conversion of mantissa to negative binary number) ∞ –10001.100110 (binary value)// –0.54980469 (denary value of mantissa) // –563/1024 Or ∞ Use exponent to denormalise mantissa **1** mark for correct answer ∞ = –17 19/32 // –17.59375	**3**
1(b)	**1** mark per bullet ∞ 5.25 = 101.01 (conversion to binary) ∞ = 0.10101 × 2^3 (evidence of shifting binary point appropriately) ∞ 010101000000 0011 (stored as mantissa and exponent)	**3**
1(c)	**1** mark per bullet ∞ (Size of mantissa decreased means that) precision is reduced ∞ (Size of exponent is increased means that) range is increased	**2**

Answer 11

3(a)	**1** mark per bullet • 21.75 = 010101.11 (conversion to correct binary) • 0.1010111 × 2^5 (evidence of shifting binary point appropriately) • 01010111 0101 (stored as mantissa and exponent)	**3**

3(b)	**1** mark per bullet, max **2**
	
	• 1110 = –2 (conversion of exponent to denary)
	• 1.011000 = –0.101 (conversion of mantissa to negative binary number)// –0.625 (denary value of mantissa)// –5/8
	• –0.00101 (binary value) //
	
	Or
	• Use exponent to denormalise mantissa
	
	1 mark for correct answer
	• –5/32 // –0.15625
		3

Answer 12

2(a)(i)	1 mark per bullet point: ∞ Correct value for exponent identified e.g. (0.0111 × 2^)7 ∞ Used to give correct value e.g. 111 000 (1/4 + 1/8 +1/16) × 128, 0.4375 ∞ Correct answer i.e. 56	**3**
2(a)(ii)	The two most significant bits are 0 in the mantissa // In mantissa, 2nd bit is not the inverse of 1st bit	**1**
2(a)(iii)	1 mark per bullet point: ∞ Mantissa = 01110000 ∞ Exponent = 0110	**2**
2(b)(i)	1 mark per bullet point: ∞ Mantissa = 01111111 ∞ Exponent = 0111	**2**
2(b)(ii)	1 mark per bullet point: ∞ Mantissa = 01000000 ∞ Exponent = 1000	**2**
2(c)(i)	Precision of numbers represented will increase	**1**
2(c)(ii)	Range of numbers represented will increase	**1**
2(d)	1 mark per bullet point to max 3: ∞ 0.1/0.2/0.3 cannot be represented exactly in binary / rounding errors ∞ adding two or more inaccurate representations together <u>increases</u> the probability of <u>inaccuracy</u> ∞ giving an answer where the difference is significant enough to be seen	**3**

Answer 13

1(a)(i)	1 mark per bullet point: • Correct value for exponent identified e.g. (0.010101 × 2^)5 • Used to give correct value e.g. 1010.1 or 21/64 x 32 • Correct answer i.e. 10.5 // 10½	**3**
1(a)(ii)	1 mark per bullet point: • Correct binary value i.e. 111.1 • Value for exponent identified e.g. (0.1111 × 2^)3 • Correct answer i.e. 01111000 00000011	**3**
1(a)(iii)	1 mark per bullet point: • Any working method for conversion • Applied accurately • Correct answer i.e. 10001000 00000011	**3**
1(b)(i)	<u>Largest</u> (positive) number (in this format)	**1**
1(b)(ii)	Overflow // too large to represent // would become negative	**1**

Answer 14

1 **(a)** +2.5
= 010100000000 0010 [3]
Give full marks for correct answer (normalised or not normalised)

= 10.1 [1]
= 0.101×2^2 // evidence of shifting binary point appropriately [1]

[Max 3]

(b) –2.5
101100000000 0010
Give full marks for correct answer

One's complement of 12-bit mantissa of +2.5 <u>101011111111</u> – allow f.t. [1]
+1 to get two's complement <u>101100000000</u> [1]

[Max 3]

(c) 3 [3]
Give full marks for correct answer

= 0.011 X 2^3 // exponent is 3 [1]
= 11.0 // (1/4+1/8) * 8 [1]

[Max 3]

(d) **(i)** Not normalised [1]

(ii) First two bits should be different for normalised number
// because the number starts with 00 [1]

(e) reduced accuracy [1]
increased range [1]

Answer 15

1 **(a)** +3.5
01110000 00000010 [3]
Give full marks for correct answer (normalised or unnormalised)

= <u>11.1</u> [1]
= 0.111×2^2 // evidence of shifting binary point appropriately [1]

[Max 3]

(b) –3.5
10010000 00000010 [3]
3 marks for correct answer

One's complement of 8-bit mantissa for +3.5 <u>10001111</u> – allow f.t. [1]
+1 to get two's complement <u>10010000</u> [1]

[Max 3]

(c) 14 [3]
3 marks for correct answer

=0.111 X 2^4 // exponent is 4 [1]
=1110.0 / (1/2 + 1/4 + 1/8) * 16 [1]

[Max 3]

(d) (i) Normalised [1]

(ii) Leftmost two bits are different for normalised representation
// because the pattern starts with 01 [1]

(e)

1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

[1]
[1]

Answer 16

1 (a) (i) 00101000 00000011
=<u>0.0101</u> × 2 ↑3 [1]
=10.1 [1]
=2.5 [1]

(ii) For a positive number (mantissa starts with a zero) [1]
bit after binary point (second bit from left) should be a one [1]

(iii) 00101000 00000011
= 01010000 00000010 [1+1]

(b) (i) 01111111 0111111 [1+1]

(ii) 01000000 1000000 [1+1]

(iii) number will become too large to represent [1]
which will result in overflow [1]

(c) Any point 1 mark

0.1 cannot be represented exactly in binary
0.1 represented here by a value just less than 0.1
the loop keeps adding this approximate value to counter
until all accumulated small differences become significant enough to be seen

[max 3]

Answer 17

1 (a) (i) 01101000 0011
= <u>0.1101</u> (or <u>1/2 + 1/4 + 1/16</u>) × 2↑<u>3</u> [1+1]
= 110.1
= 6.5 [1]

(ii) +3.5
= 11.1 [1]
= 0.111 × 2↑2 (or indication of moving binary point correctly) [1]
= 01110000 0010 [1]

(iii) 01110000 Allow f.t. from **(ii)**
10001111 One's complement on mantissa [1]
10001111 +1 Two's complement [1]

= 10010000 0010 [1]

(b) **(i)** Precision/accuracy of numbers represented will increase [1]

(ii) Range of numbers represented will increase [1]

(c) Any point, 1 mark (max. 3)

0.1/0.2 cannot be represented exactly in binary // rounding error [1]
0.1 represented by a value just greater than 0.1 // 0.2 represented by a value just greater than 0.2 [1]
adding two representations together adds the two differences [1]
summed difference significant enough to be seen [1]
[max. 3]

[Total: 14]

Answer 18

1(a)	Working: **one mark** for calculation of the mantissa and **one mark** for calculation or use of the exponent **Exponent: one** from: = 0.11101×2^3 // 0.11101×2^{11} // 0.11101×10^3 // 0.11101×10^{11} = 1.00011×2^3 // 1.00011×2^{11} // 1.00011×10^3 // 1.00011×10^{11} = appropriate shifting of binary point for +7.25 **Mantissa: one** from: = 111.01 (conversion to binary +7.25 – 10 bits) = 0111010000 (mantissa 10 bits for +7.25 = 1000101111(one's complement mantissa for –7.25) = 1000110000 (two's complement mantissa for –7.25) Correct Answer **(Max 1)** Mantissa: 1 0 0 0 1 1 0 0 0 0 Exponent: 0 0 0 0 1 1	**3**

1(b)	**One** mark for working out the exponent **One** mark for working out the mantissa **One** mark for the correct answer Example answers • $=1.011000111 \times 2^7$ (exponent is 7) • =10110001.11 // –128 + 32 + 16 + 1 + 0.5 + 0.25 // convert to positive 01001110.01 (and add a minus sign to the answer) • –78.25	**3**
1(c)	**One** mark for working **One** mark for correct mantissa **One** mark for correct exponent Example answers Number of places added to exponent for normalisation −6 for number to retain its value // mantissa moved 6 places left Mantissa 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 Exponent 1 | 0 | 0 | 0 | 0 | 1	**3**

1(d)(i)	**One** mark for each correct marking point (**Max 3**) • Requires 11 bits / more than 10 bits to store (accurately) / reference to maximum (positive) number that can be stored = 511 • Denary 513 in binary is 1000000001 // Normalised: 0.1000000001 • Results in overflow	**3**

1(d)(ii)	**One** mark for each correct marking point (**Max 2)** • The number of bits for the mantissa must be increased • 11/12 bits mantissa **and** 5/4 bits exponent	**2**

Answer 19

1(a)(i)	**One** mark for each correct marking point (**Max 2)** • 010111000110 (correct mantissa) • 0111 (correct exponent)	**2**
1(a)(ii)	**One** mark for each correct consequence **One** mark for each correct justification **Consequence** • The precision/accuracy of the number would be reduced **Justification** • … because the least significant bits of the original number have been truncated/lost // the original number had 13 bits / 14 bits with sign but the mantissa can only store 12 bits	**2**
1(b)	**One** mark for each correct marking point (**Max 3)** • To store the maximum range of numbers in the minimum number of bytes / bits • Normalisation minimises the number of leading zeros/ones represented • Maximising the number of significant bits // maximising the (potential) precision / accuracy of the number for the given number of bits • … enables very large / small numbers to be stored with accuracy. • Avoids the possibility of many numbers having multiple representations.	**3**

Answer 20

1(a)	**Two** marks for working **One** mark for correct answer Working: Conversion to binary + 202 = 11001010 // repeated division by 2 // 128 + 64 + 8 + 2 Appropriate shifting of binary point for + 202 = 0.1100101 × 2^8 // exponent = 8 Answer: = 01100101 00001000 (stored as mantissa and exponent)
1(b)	**Two** marks for working **One** mark for correct answer Working: • Appropriate method of conversion e.g. = 10011010 (one's complement of 8-bit mantissa) = 10011011 (two's complement of 8-bit mantissa) –256 + 32 + 16 + 4 +2 • Realisation that the exponent doesn't change // value of exponent = 8 // appropriate shifting of binary point Answer: = 10011011 00001000 (stored as mantissa and exponent)
1(c)(i)	The mantissa does not begin with 01/10 (as its most significant bits) // the mantissa begins with 00 // first two digits are the same.

1(c)(ii)	**One** mark for each point: • Correct mantissa • Correct exponent **Mantissa**: 0 1 1 1 1 0 0 0 **Exponent**: 0 0 0 1 0 1 1 0

Answer 21

1(a)	**Mantissa**: 0 1 1 1 1 1 1 1 1 1 1 **Exponent**: 0 1 1 1 1
1(b)	**Two** marks for working • correct calculation of exponent seen • correct application of exponent to mantissa seen **One** mark for correct answer Working: = 1.0110010011 × 2^9 //exponent = 9 = 1011001001.1 (moving bp 9 places to right) // evaluate two's complement For example: –512 + 128 + 64 + 8 + 1 + 0.5 Answer: –310.5 // –310 $^1/_2$
1(c)	**One** mark per point • Following an arithmetic/logical operation • … the result is too small to be precisely represented in the available system // When the number of bits is not enough / too small for the computer's allocated word size / to represent the binary number

Answer 22

1(a)	**One** mark per mark point (**Max 4**) • conversion of 113.75 to binary seen 1110001.11 • exponent for normalisation 7 converted to binary 111 // evidence of binary point moved 7 places // evidence of finding exponent = 7 • system 1 answer • system 2 answer showing correct version from system 1 **System 1** **Mantissa**: 0 1 1 1 0 0 0 1 1 1 **Exponent**: 0 0 0 1 1 1 **System 2** **Mantissa**: 0 1 1 1 0 0 0 1 **Exponent**: 0 0 0 0 0 1 1 1	**4**
1(b)	**One** mark per mark point (**Max 2**) • the mantissa in system 2 does not have enough bits to store the whole binary number // 10 bits required and only 8 bits available • so precision is lost / the number is truncated	**2**

Answer 23

1(a)	**One** mark per mark point • correct mantissa • correct exponent **with** associated working Answer **Mantissa**: 0 1 0 1 0 1 0 1 1 1 **Exponent**: 0 0 0 1 1 0 Working exponent = 6 (movement of 6 bicimal places seen to find what exponent should be) calculation of denary 6 to binary (000)110	**2**
1(b)	**One** mark per mark point (**Max 3**) MP1 the mantissa of the number would need to be 0.101011111001 / 13 bits / digits MP2 … it can only store 10 bits / digits MP3 The **3** least significant digits would be truncated MP4 …causing a loss of precision	**3**

Answer 24

1(a)	**One** mark for working (**Max 1**) • conversion of 65.25 to binary seen e.g. 1000001.01 = 65.25 // 64 + 1 + 0.25 / ¼ **One** mark per mark point (**Max 2**) • correct mantissa • correct exponent **Mantissa**: 0 1 0 0 0 0 0 1 0 1 0 0 **Exponent**: 0 1 1 1	**3**
1(b)	**One** mark per mark point (**Max 2**) MP1 the decimal fraction 0.20 cannot be represented **exactly** (the closest is 0.25 / 0.1875) MP2 therefore, there will be a loss of precision due to a rounding error/truncation	**2**

Answer 25

1(a)	**One** mark per mark point (**Max 1**) • conversion of −96.75 to binary e.g., positive 96.75, flip the bits + 1 to give 10011111.01 // –128 + 16 + 8 + 4 + 2 + 1 + 0.25 / ¼ seen **One** mark per mark point (**Max 2**) • correct mantissa • correct exponent **Mantissa**: 1 0 0 1 1 1 1 1 0 1 **Exponent**: 0 0 0 1 1 1	**3**
1(b)	**One** mark per mark point (**Max 3**) MP1 Real numbers (can) have a fractional part (such as 1/3 and ½) / (such as 0.4 and 0.25) MP2 The fixed length of the storage means that you can't store very large / very small numbers MP3 Binary numbers represent numbers based on powers of 2, with limited fractional representations such as 1/2, 1/4, 1/8, 1/16, etc. MP4 It isn't possible to store all fractions with the level of precision provided by this system MP5 …the fractional part of the number is as close as possible within these constraints.	**3**