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

0	1	1	0	0	0	0	0	0	1

Exponent

0	0	1	0	0	0

Working 192 / 128 / 64 / 64 / 0 0110000000.1

011000001 001000

.... [3]

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

Mantissa

1	0	0	1	1	1	1	1	1	1

Exponent

0	0	1	0	0	0

Working –256. +192.5 063.5

–256 128 64 32 16 8 4 2 1 . 1/2

1 0 0 1 1 1 1 1 1 . 1

.... [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.

It can't be normalised as to normalise 01 should be the first few digits and the exp should be 8 to do that however we can't represent 8 in a 4 bit exponent bit thus we would either have to see accuracy/precision

exp will turn -8 value

[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

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

Exponent

0	1	0	1

Working 0011.00000000

3

Exp = 4+1

= 5

Denary value 3

[3]

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

Write your answer in the following boxes.

Mantissa

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

Exponent

0	0	1	0

[2]

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

precision is lost redund. leading zero
multiple repre of a single number,
.... [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

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

Exponent

-8	4	2	1
0	1	1	0

Working . 32 16 8 4 2 1
101000.00000
32+8 = 40

Epx- 4+2
=6

Denary value 40

[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

-8	4	2	1
0	1	0	1

Mantissa

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

Working Exp = 4+1 = 5   00.000110

$\frac{1}{16} + \frac{1}{32} = \frac{2+1}{32} = \frac{3}{32}$

Denary value 3/32

[3]

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

Problem 1 number calculated will change

Problem 2 same bit pattern for different numbers Software can crash

[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

-8	4	2	1
0	1	0	1

Working 00011.00000   Exp 4+1 = 5

Denary value 3

[3]

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

Write your answer in the following boxes.

Mantissa

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

Exponent

~~0~~	~~0~~	1	0

[2]

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

a number can have multiple representation.
lost in ~~acc~~ precision.

[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								Exponent							
0	1	1	1	1	1	1	1	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

0	● 0	1	1	0	0	0	0	0	0	0	0

Exponent

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]

Answer

Answer 1

1(a)	= (0)11000000.1 (conversion to binary) [1] = 0.110000001 $\times 2^8$ (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) ∞ 0.2/0.4 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 101011111111 – allow f.t. [1]
+1 to get two's complement 101100000000 [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)

= 11.1 [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 10001111 – allow f.t. [1]
+1 to get two's complement 10010000 [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]