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 <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 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

= $0.011 \text{ X } 2^3$ // 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
=$\underline{0.0101} \times 2 \uparrow 3$ [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]