2.5 Data Representation: Number Systems: Test Pack

Total Marks

OCR J277 · Calculators not allowed · about 20 minutes

Name: _____ Date: ____ / ____ / _____

Answer ALL questions. Show all your working.

1 Computers represent all data internally using binary digits (bits), where each digit is either 0 or 1. Which of these 4-bit binary numbers represents the denary value 9?

- ☐ A) 1001
- ☐ B) 1010
- ☐ C) 0110
- ☐ D) 1100

(Total for Question 1 is 1 mark)

---

2 One byte of data is made up of two nibbles.

(a) State how many hexadecimal digits are needed to represent one byte of data. **(1)**

(b) Explain why two hexadecimal digits can represent exactly one byte, with no bits left over. **(2)**

(Total for Question 2 is 3 marks)

© Revision Library | revisionlibrary.co.uk

3 Binary addition follows the same column method as denary addition, but any column that totals 2 or more requires a carry into the next column.

(a) Add the two 8-bit binary numbers 01001101 and 00110110. Show your carries and give your answer as an 8-bit binary number. **(2)**

.....

(b) State the denary value of the result to check your answer to part (a) is correct. **(1)**

(Total for Question 3 is 3 marks)

---

4 A binary shift moves every bit in a binary number a fixed number of places to the left or right, filling any empty places with zeros.

(a) State the effect on the denary value of a binary number when a single left binary shift is performed. **(1)**

(b) The 8-bit register value 00001101 (denary 13) is shifted left by 2 places. State the new 8-bit binary value and its denary equivalent. **(2)**

(c) The 8-bit register value 11000000 (denary 192) is shifted left by 2 places. State the resulting 8-bit binary value, and explain why this shift no longer represents multiplying the denary value by 4. **(2)**

(Total for Question 4 is 5 marks)

© Revision Library | revisionlibrary.co.uk

5 A programmer writes the following algorithm, using the OCR Exam Reference Language, to convert a positive denary number entered by the user into its binary representation, using the divide-by-2 remainder method:

```
number = input("Enter a positive whole number: ")
number = int(number)
binary = ""
while number > 0
remainder = number MOD 2
binary = str(remainder) + binary
number = number DIV 2
endwhile
```

```
print(binary)
```

(a) Complete a trace table for this algorithm when the user enters the value 22, showing the value of number at the start of each pass of the loop, the remainder calculated, the value of binary after it is updated, and the new value of number after the DIV 2 step. **(5)**

(b) State the value that would be output by print(binary) once the algorithm has finished running. **(1)**

(Total for Question 5 is 6 marks)

© Revision Library | revisionlibrary.co.uk

Mark scheme · 2.5 Data Representation: Number Systems: Test Pack

Question 1

A cao **B1**

Answer: A) 1001

Question 2

(a) 2 cao **B1**

(a) Answer: 2

(b) one byte is made up of 8 bits, and each hexadecimal digit represents exactly 4 bits (one nibble) oe **B1**

(b) 2 hex digits x 4 bits = 8 bits, which exactly matches the size of a byte, so no bits are left over oe **B1**

(b) Answer: One byte is 8 bits, and each hexadecimal digit represents exactly 4 bits (a nibble). Since 2 x 4 = 8, two hexadecimal digits map exactly onto one byte with no bits left over.

Question 3

(a) correct column addition attempted with carries shown/propagated correctly **M1**

(a) 10000011 cao **A1**

(a) Answer: 10000011

(b) 131 cao (ft from their answer to (a)) **B1**

(b) Answer: 131

Question 4

(a) it multiplies the denary value by 2 oe **B1**

(a) Answer: It multiplies the value by 2

(b) 00110100 stated **M1**

(b) 52 stated (ft from their binary answer) **A1**

(b) Answer: 00110100 = 52

(c) 00000000 stated as the resulting 8-bit value **B1**

(c) the two leftmost 1 bits are shifted off the end of the 8-bit register and lost (overflow), because the true result (192x4=768) needs more than 8 bits to store, so the register cannot hold it correctly oe **B1**

(c) Answer: 00000000; the two leading 1 bits are shifted off the left-hand end of the register and lost, because the true result of 192x4=768 needs more than 8 bits, so an overflow occurs and the stored value is incorrect.

Question 5

(a) pass 1: number=22, remainder=0, binary="0", new number=11, all correct **B1**

(a) pass 2: number=11, remainder=1, binary="10", new number=5, all correct **B1**

(a) pass 3: number=5, remainder=1, binary="110", new number=2, all correct **B1**

(a) pass 4: number=2, remainder=0, binary="0110", new number=1, all correct **B1**

(a) pass 5: number=1, remainder=1, binary="10110", new number=0, all correct (loop then ends as number is no longer > 0) **B1**

(a) Answer: Pass 1: 22,0,"0",11 | Pass 2: 11,1,"10",5 | Pass 3: 5,1,"110",2 | Pass 4: 2,0,"0110",1 | Pass 5: 1,1,"10110",0

(b) 10110 cao, ft from their trace table **B1**

(b) Answer: 10110

© Revision Library | revisionlibrary.co.uk