| | |
|---|---|
| 4(a) | **1 mark** for:<br>LSL #2 |
| 4(b) | 1 1 1 1 0 0 0 1 |
| 4(c) | 0 0 1 0 0 0 1 1 |
| 4(d) | 1 1 1 1 0 0 1 1 |

| | |
|---|---|
| 6(a) | **1 mark** for each correct description, **max 2** marks<br><br>**Register** / **Role**<br>ACC: stores the intermediate results of arithmetic and logical operations // holds the result of a calculation<br>CIR: holds the instruction currently being decoded and/or executed |
| 6(b) | **1 mark** per bullet point, **max 2** marks<br><br>e.g.<br><br>• Latency may be increased<br>• … because the cores must communicate with one another<br><br>• There is a potential for dead-lock situations<br>• … where one core may wait for information from other cores which in turn are waiting for the first one<br><br>• Not all software is designed to use multi-cores<br>• ... so some of the additional cores would be idle<br><br>• Increased heat generation<br>• ... which could cause damage to other components |
| 6(c) | **1 mark** per bullet point, **max 3** marks<br><br>• DRAM requires to be refreshed/charged SRAM does not request a refresh<br>• DRAM stores each bit as charge SRAM uses flip-flop to store each bit<br>• DRAM is less expensive to manufacture SRAM is more expensive to manufacture<br>• DRAM has slower access speeds SRAM has faster access times<br>• DRAM has higher storage/bit/data density SRAM has lower storage/bit/data density<br>• DRAM is used in main memory SRAM is used in cache |





| | |
|---|---|
| 3(a)(i) | **1 mark**<br><br>• The value of the operand is an offset value which is added to another base value to give the address from which the contents are loaded to the accumulator |
| 3(a)(ii) | **1 mark** per bullet point, **max 2** marks<br><br>• Program counter (PC)<br>• Memory Data Register (MDR)<br>• Memory Address Register (MAR)<br>• Current Instruction Register (CIR)<br>• Status Register |
| 3(b) | **1 mark** for each correct value, **max 3** marks<br><br>**Instruction** / **Value in ACC**<br>`LDM #98` / 98<br>`LDI 101` / 8<br>`LDX 100` / 32 |
| 3(c) | **1 mark** per bullet point, **max 2** marks for each<br><br>Clock Speed:<br>• Processor can perform more F-E cycles per second<br>• ... so more instructions / data can be processed each second<br><br>Cache Memory:<br>• Can store more of the most frequently used instructions<br>• … which reduces the need to access slower RAM |

**2(a)**

**3 marks** for 6 correct ticks
**2 mark** for 4 or 5 correct ticks
**1 mark** for 2 or 3 correct ticks

| Example of data | Number of bits | | | | | | |
|---|---|---|---|---|---|---|---|
| | **4** | **8** | **16** | **24** | **32** | **64** | **128** |
| the hexadecimal value F139 | | | ✓ | | | | |
| 16 000 000 unique amplitude values | | | | ✓ | | | |
| an IPv4 address | | | | | ✓ | | |
| 256 unique colours | | ✓ | | | | | |
| an IPv6 address | | | | | | | ✓ |
| the denary value 65 000 | | | ✓ | | | | |

**2(b)**

**1 mark**

1111 1001 0100

**2(c)**

**1 mark**

1111 0010

**2(d)**

**1 mark**

E33B





**11(a)**

**One** mark per mark point (**Max 6**)
MP1 `LDM #100` seen
MP2 Correct use of `STO` with labelled address (constant or answer)
MP3 Correct use of `LDD 632`
MP4 Correct use of `SUB` with labelled address (constant)

| Opcode | Operand |
|---|---|
| LDM | #100 |
| STO | Constant |
| LDD | 632 |
| SUB | Constant |
| STO | Answer |

MP5 Storing `100` at a labelled address away from the code
MP6 Labelling both addresses away from the code.

| Label | Contents |
|---|---|
| Constant: | 100 |
| Answer: | |

**11(b)**

–55