**CAMBRIDGE INTERNATIONAL EXAMINATIONS**

**Cambridge International Advanced Subsidiary and Advanced Level**

# MARK SCHEME for the May/June 2015 series

# 9608 COMPUTER SCIENCE

**9608/13** Paper 1 (Written Paper), maximum raw mark 75

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

Cambridge will not enter into discussions about these mark schemes.

Cambridge is publishing the mark schemes for the May/June 2015 series for most Cambridge IGCSE®, Cambridge International A and AS Level components and some Cambridge O Level components.

**1 (a) (i)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 124 | **0** | **1** | **1** | **1** | **1** | **1** | **0** | **0** |
| –77 | **1** | **0** | **1** | **1** | **0** | **0** | **1** | **1** |

[2]

**(ii)** 124: 7 C

–77: B 3 [2]

**(b) (i)** 0 0 1 1 0 1 0 1 1 0 0 1 [1]

**(ii)**
- when denary numbers need to be electronically coded
- e.g. to operate displays on a calculator where each digit is represented
- decimal fractions can be accurately represented [2]

**2**

| Activity | First pass or second pass |
|---|---|
| any symbolic address is replaced by an absolute address | **2** |
| any directives are acted upon | **1** |
| any symbolic address is added to the symbolic address table | **1** |
| data items are converted into their binary equivalent | **1** |
| forward references are resolved | **2** |

[5]

**3 (a)** maximum of **two marks** for firewall description + maximum of **two marks** for authe… description

**Firewall**

- sits between the computer or LAN and the Internet/WAN and permits or blocks traffic to/from the network
- can be software and/or hardware
- software firewall can make precise decisions about what to allow or block as it can detect illegal attempts by specific software to connect to Internet
- can help to block hacking or viruses reaching a computer

**Authentication**

- process of determining whether somebody/something is who/what they claim to be
- frequently done through log on passwords/biometrics
- because passwords can be stolen/cracked, digital certification is used
- helps to prevent unauthorised access to data [3]

**(b)** **one** mark for security, **one** mark for integrity:

- integrity deals with validity of data/freedom from errors/data is reasonable
- security deals with protection of data
- security protects data from illegal access/loss
- integrity deals with making sure data is not corrupted after, for example, being transmitted [2]

**(c) (i)** **one mark** for each way of maintaining data security + **one mark** for an example/ enhancement

- validation (to ensure data is reasonable)
- examples include range checks, type checks, length checks, …

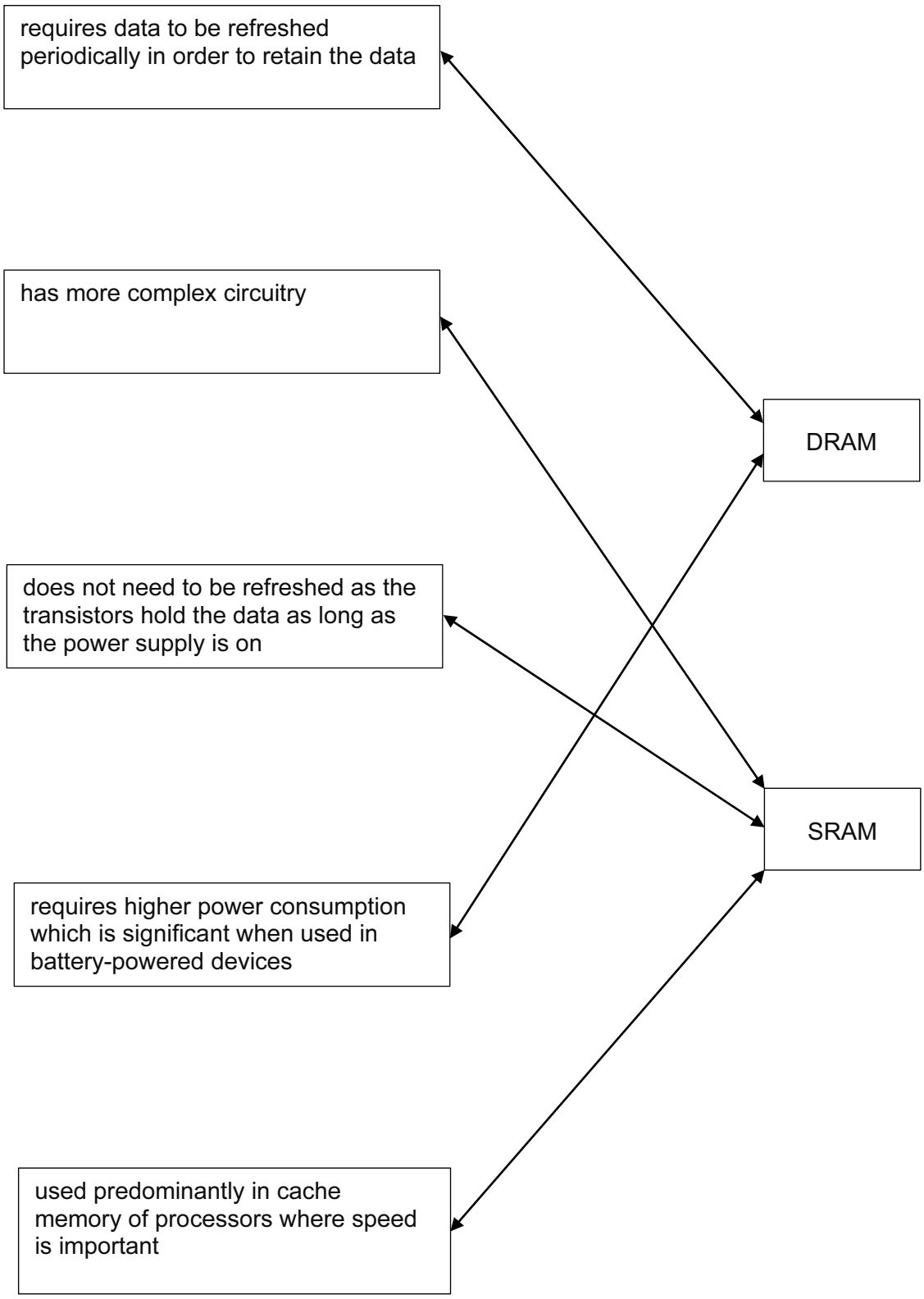
- verification (checks if data input matches original/if transmitted data matches original)
- can use double data entry or visual check/other methods such as parity checks
- doesn't check whether or not data is reasonable [3]

**(ii)** **one mark** for each way of maintaining data integrity + **one mark** for an example/ enhancement

- parity checking
- one of the bits is reserved as parity bit
- e.g. 1 0 1 1 0 1 1 0 uses odd parity
- number of 1s must be odd
- parity is checked at receiver's end
- a change in parity indicates data corruption

- check sum
- adds up bytes in data being sent and sends check sum with the data
- calculation is re-done at receiver's end
- if not the same sum then the data has been corrupted during transmission [3]

**4 (a)**




[5]

**(b)** maximum of **two marks** for RAM and maximum of **two marks** for ROM

**RAM**

- loses contents when power turned off/volatile memory/temporary memory
- stores files/data/operating system currently in use
- data can be altered/deleted/read from and written to
- memory size is often larger than ROM

**ROM**

- doesn't lose contents when power turned off/non-volatile memory/permanent memory
- cannot be changed/altered/deleted/read only
- can be used to store BIOS/bootstrap [3]

**(c)** **one** mark for DVD-RAM, **one** mark for flash memory.

**DVD-RAM**

- data is stored/written using lasers/optical media
- DVD-RAM uses phase changing recording, in which varying laser intensities cause targeted areas in the phase change recording layer to alternate between an amorphous and a crystalline state.
- uses a rotating disk with concentric tracks
- allows read and write operation to occur simultaneously

**flash memory**

- most are NAND-based flash memory
- there are no moving parts
- uses a grid of columns and rows that has two transistors at each intersection
- one transistor is called a floating gate
- the second transistor is called the control gate
- memory cells store voltages which can represent either a 0 or a 1
- essentially the movement of electrons is controlled to read/write
- not possible to over-write existing data; it is necessary to first erase the old data then write the new data in the same location [2]

**5 (a)** **one mark** for name of bus + **one mark** for description

**address bus**

- lines used to transfer address of memory or input/output location
- unidirectional bus

**data bus**

- used to transfer data between the processor and memory/input and output devices
- bidirectional bus

**control bus**

- used to transmit control signals
- e.g. read/write/fetch/ …
- dedicated bus since all timing signals are generated according to control signal [6]

**(b) (i)** the program counter is incremented [1]

**(ii)** the data stored at the address held in MAR is copied into the MDR [1]

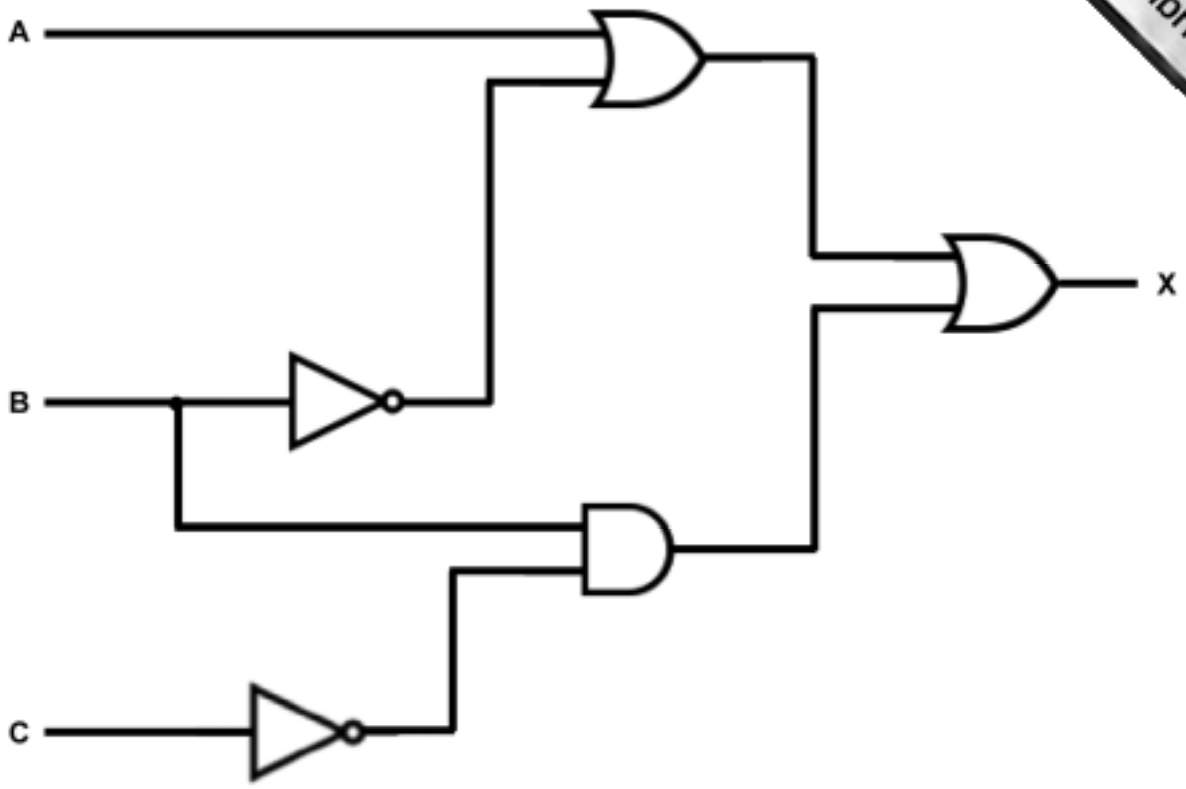
**(iii)** the contents of the Memory Data Register is copied into the Current Instruction Register [1]

**(c)**
- the MAR is loaded with the operand of the instruction // loaded with 35
- the Accumulator is loaded with the contents of the address held in MAR // the Accumulator is loaded with the contents of the address 35 [2]

**(d) (i)**
- a signal
- from a device/program that it requires attention from the processor [2]

**(ii)**
- at a point during the fetch-execute cycle …
- check for interrupt
- if an interrupt flag is set/ bit set in interrupt register
- all contents of registers are saved
- PC loaded with address of interrupt service routine [4]

**6 (a)**




[5]

**(b)**

| A | B | C | working | X | |
| --- | --- | --- | --- | --- | --- |
| **0** | **0** | **0** | | **1** | } 1 mark |
| **0** | **0** | **1** | | **1** | |
| **0** | **1** | **0** | | **1** | } 1 mark |
| **0** | **1** | **1** | | **0** | |
| **1** | **0** | **0** | | **1** | } 1 mark |
| **1** | **0** | **1** | | **1** | |
| **1** | **1** | **0** | | **1** | } 1 mark |
| **1** | **1** | **1** | | **1** | |

[4]

**(c)** ((A is NOT 1 AND B is 1) OR (B is NOT 1 OR C is 1)) AND C is NOT 1
< – – – – – 1 mark – – – – – > < – – – – – 1 mark – – – – – > < – – – – –1 mark – –

NOTE: all brackets may not be shown – but check answer still correct

Alternatives include:

((NOT A AND B) OR (NOT B OR C)) AND NOT C

$(\overline{A} . B + (\overline{B} + C)) . \overline{C}$

NOTE: expressions may be reversed but still OK

(e.g. NOT C AND ((NOT A AND B) OR (NOT B OR C))

NOT C AND ((NOT B OR C) OR (NOT A AND B)) and so on) [3]

**7 (a) (i)**

| Accumulator: | **0** | **1** | **1** | **1** | **0** | **1** | **0** | **1** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

[1]

**(ii)**

| Accumulator: | **0** | **1** | **1** | **0** | **1** | **0** | **0** | **1** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

[1]

**explanation**

- content of 124 is 0 1 1 1 1 1 1 1
- this is equivalent to 127
- contents of 127 are 0 1 1 0 1 0 0 1 [2]

**(iii)**

| Accumulator: | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **1** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

[1]

**explanation**

- index register value = 6
- 120 + 6 = 126
- contents of 126 placed in the accumulator [2]

**(b)** 1 mark for each correct value in the table.

| Accumulator | Memory address | | | |
| --- | --- | --- | --- | --- |
| | 320 | 321 | 322 | 323 |
| | 49 | 36 | 0 | 0 |
| 36 | | | | |
| 37 | | | | |
| | | | | 37 |
| 49 | | | | |
| 50 | | | | |
| | | | 50 | |

[6]