Examiners' commentary 2018–2019

CO1110 Introduction to computing and the internet – Zone A

General remarks

The examination was set to test the candidate's basic and deeper understanding of the material contained in the subject guide, Introduction to computing and the internet. The examination is split into two parts, A and B. Candidates had to answer two questions from each part.

Comments on specific questions

Question 1

a. Answer

i. (A) One's complement

ii. (D) When the negative exponent is too large to be expressed in the number of bits available

iii. (D) All of the above

Comments

Most candidates gave the correct answer to parts (i) and (ii). Part (iii) was most likely to be answered incorrectly, with candidates failing to understand that while the statement that they gave was true, the statements (A), (B) and (C) were all true, making (D) the correct answer.

b. Answers

i. `6` `-6`

`0110` `1010`

ii. `1100`

`1001`

`-----`

`1`**`0101`**

Answer after discarding the extra bit: `0101`

The leading zero in the result indicates both a positive number, and a change of sign from the operands, which are both negative numbers since they both have 1 as their sign bit. The change of sign bit indicates overflow.

iii. `111100`

`111001`

`------`

`1`**`110101`**

Answer after discarding the extra bit: `110101`

There is no difference in the sign of the operands and the result; hence there is no overflow.

Comments

Part (i) was normally answered correctly, with most candidates understanding that in order to find the positive representation of a two's complement number, one first has to find the unsigned binary representation of the number, then use one of the methods described in section 6.3.8 of Volume 1 of the subject guide to transform this into the negative representation of the number.

Part (ii) was also usually answered correctly, with a few candidates who did not discard the extra bit, but instead incorrectly claimed that there was overflow because adding up 4-digit numbers had resulted in a 5-digit number. As the subject guide explains in section 6.3.8, when addition in two's complement results in an extra bit, that bit is discarded to give the correct answer.

In part (iii) by far the most common mistake was with sign-extending the 4-bit two's complement numbers to 6-bits. Section 6.3.13 of Volume 1 of the guide explains that to convert a two's complement number to a two's complement number with more bits, it is first necessary to move the sign bit to the leftmost position in the new number, and then fill in the gaps with copies of the sign bit. Hence the first stage in transforming A = 1100 to a two's complement number with 6-bits would be 1 _ _ 100. The spaces are then filled with copies of the sign bit, giving 111100.

c. Answer

i. Number is positive so the sign bit is zero

63 = 111111 = 1.11111 x 2^5 = the real mantissa

The normalised mantissa is 1111 1000 0000 0000 000

The exponent is 5. Biasing it by 127 gives 127 + 5 = 132

132 = 1000 0100

Hence 63 in normalised IEEE 754 single precision format is:

0 1000 0100 1111 1000 0000 0000 0000 000

ii. 0.01101111 x 2^{-126}

Comments

Part (i) was usually answered correctly, with the most common mistake being giving the mantissa as 0.111111 x 2^6, meaning that the exponent would be given as 6 and the biased exponent as 133. Candidates should know that the IEEE 754 format includes an implicit 1 in the normalised mantissa, which means that the real mantissa must have its leading 1 immediately before the binary point, and all other digits after the binary point, which is the case with 1.11111 x 2^5.

In part (ii) the most common incorrect answer was 1.734375 x 2^{-128}. The subject guide, Volume 1 section 6.4.6 explains that the smallest possible exponent value is –127, which is reserved for special values, hence –126 is the smallest possible exponent for normalised and denormalised numbers. In their answer candidates were expected to shift the binary point, and the exponent until the exponent was –126, giving a number that could be expressed in denormalised form. See Volume 1 of the subject guide, section 6.4.8 for more on denormalised numbers.

About a half of candidates who gave the correct answer went further and transformed their answer into IEEE 754 denormalised representation. These candidates did not lose any marks, but did much unnecessary work when the above answer was all that was required. The examiners had tried to indicate this by the wording of the question, which stated that the number given should be converted 'to a form that can be expressed in IEEE 754 32-bit denormalised form'.

Question 2

a. Answer

i. (C) Binary code that specifies the operation to be performed by the processor

ii. (A) Instruction register

iii. (B) Millions of floating point operations per second

Comments

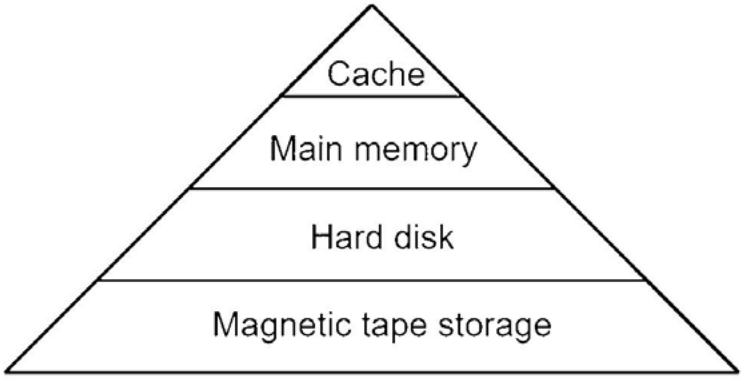
The majority of candidates gave correct answers to part (a).

b. Answer

i. *Any 2 of the following:*

- **Speed of memory** decreases with increasing distance from the processor.
- **Cost of memory** decreases with increasing distance from the processor.
- **Frequency of access** decreases with increasing distance from the processor.

ii.



iii. With pipelining, M instructions can take **N + M – 1** time units.

Comments

In parts (i) and (ii) most candidates gave correct answers, however part (iii) was answered incorrectly about half the time. Incorrect answers included (M x N)/M, which, after cancelling the M from the top and bottom, reduces to N, and also (M x N)/N, which reduces mathematically to M.

For a pipeline that works perfectly N + M –1 is the fastest possible time to execute M instructions with N stages. This is because instructions can be overlapped in a pipeline, with one instruction started in the first cycle, taking N stages to complete, and each subsequent instruction adding 1 to the total time units, or cycles, taken because the final stage only of each subsequent instruction is pushed into a new cycle, hence N + M –1.

For example, if an instruction has five stages, then three instructions could take 5 + 3 –1 = 7 cycles (or time units) to complete. This can be seen in the table below, where the instruction cycle consists of five stages IF; ID; EX; MEM; and WB:

Cycle	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Instruction 1	IF	ID	EX	MEM	WB		
Instruction 2		IF	ID	EX	MEM	WB	
Instruction 3			IF	ID	EX	MEM	WB

c. Answer

i. Spatial locality states that memory locations accessed in any time period are likely to be clustered in one place, because instructions are normally accessed sequentially.

To exploit this tendency, when the cache admits a new memory unit, it also admits the surrounding memory units.

It does this by dividing memory into blocks, each block fits into a line in the cache. When a new memory unit is required, the cache admits the entire block containing that memory unit into a line in the cache.

ii.

Main memory block numbers		Cache lines
{0, 4, 8, 12, 16}	➔	Line 0
{1, 5, 9, 13, 17}	➔	Line 1
{2, 6, 10, 14, 18}	➔	Line 2
{3, 7, 11, 15, 19}	➔	Line 3

iii. (A) Direct mapped cache address

(B) Associative mapped cache address

(C) Set associative mapped cache address

Comments

All candidates gained some credit for part (i) but often missed some marks for vague answers; for example, not explaining why the principle of spatial locality considers that memory locations accessed in any time period are likely to be clustered (because programs are usually executed sequentially). Many candidates stated that when the cache admits a new memory unit, it also admits surrounding, or nearby, memory units, but could not, or did not, state how this is implemented by dividing memory into blocks that fit in a cache line.

Part (ii) was usually answered wrongly; quite often candidates assigned block numbers 1–5 to line 0, 6–10 to line 1, and so on. In fact, block numbers are assigned to cache lines by taking the modulo of the block number with respect to the number of cache lines. If the modulus is 0, then the block number is assigned to cache line zero, and if it is 1 then the block number is assigned to cache line 1, and so on. The organisation of direct mapped caches is discussed in Volume 1 of the subject guide, section 4.7.4.

In answering part (iii) many candidates confused direct mapped with associative mapped, so it was only (C) that was usually given correctly.

Question 3

a. Answer

i. (A) CPU, memory and I/O devices

ii. (A) Transistors

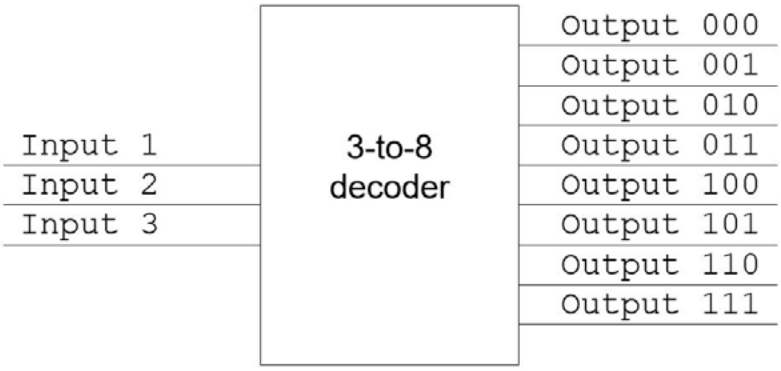
iii. (D) All of the above

Comments

In parts (i) and (ii) some incorrect answers were seen with incorrect answers distributed fairly evenly suggesting candidates were guessing. Part (iii) was also often answered incorrectly, since many candidates understood that one or other of the statements was true, but did not realise that all of the statements (A), (B) and (C) were correct, making (D) the right answer.

b. Answer

i.



ii. Each chip has 2^8 bytes on it. There are 64, or 2^6 chips. Since memory is byte addressable, the number of addressable memory locations is the number of bytes. There are 2^{14} bytes, so the number of addressable memory locations is 2^{14}.

This means that 14 bits are needed for memory addresses.

iii. There are 2^6 chips, so 6 address lines for chip select.

Byte selection can be calculated by the number of bits for memory addresses minus the bits of the memory address that are needed for chip selection.

Hence for byte selection, 14 – 6 = 8 address lines are needed.

Comments

Most candidates answered part (i) correctly, although for some reason a small number gave the output numbers in decimal, which gained no credit. Outputs should also have been given with 3 bits only: using 4, or as one candidate did, 8 bits (e.g. the first output 00000000, second 00000001, etc.), is incorrect since the 3 inputs mean that the output can only have 3 bits.

In part (ii) some correct answers were seen, with many errors. One common error was to give the answer as the total number of bytes or bits, since there are 2^{14} bytes some candidates gave 16,384 as their answer, while others gave 131,072 which is the number of bits (2^{17}). These candidates clearly did not read Chapter 3 of Volume 1 of the subject guide, in particular section 3.2.7 which notes that if the memory is composed of 2^A addressable memory units, then the number of bits needed for each memory unit to have a unique address is *A*. Since the memory given in (ii) is byte addressable, and since there are 2^{14} bytes, then *A* in this case is 14.

Few candidates received full credit for part (iii) of the question. Many candidates gave the number of lines for chip selection correctly, but could not give the correct answer for byte selection, with various incorrect answers seen. Other candidates managed to give the correct number of lines for byte selection, since these candidates correctly understood that with $256 = 2^8$ bytes on a chip, 8 address lines were needed for byte selection; however these candidates often gave various different, and wrong, answers for chip selection.

One incorrect answer seen several times, was to give the number of lines needed for byte selection as 14. These candidates perhaps were thinking that every single byte needed its own address, instead of understanding that with the most significant bits of the memory address used to select the chip, memory addresses are unique even when every chip has its bytes numbered the same.

Candidates were given some credit if their methods for calculating the answer were correct, but the answer itself was not. For example, candidates who answered 17 to part (ii), assuming that the memory was bit addressable, might give the number of lines for byte selection as 17 minus the number of lines for chip select.

c. Answer

i. **Programmed I/O**: The CPU issues a read/write command to the device's I/O module and sits idle. The CPU wastes time in checking if the state for the I/O module of the device is ready.

Interrupt-Driven I/O: The CPU issues a read/write command to the device's I/O module then goes to do other things. When the I/O module is ready it uses an interrupt signal to tell the CPU it is ready to read/write from/to the device.

Unlike Programmed I/O, the CPU does not waste time in waiting and checking if the I/O module of the device is ready.

ii. In both programmed and interrupt-driven I/O the CPU spends a lot of time transferring data as it has to handle the transmission of each word of data.

Comments

This question was not answered well, with many candidates demonstrating in their answers that they had no idea even that programmed I/O and interrupt-driven I/O were protocols used to manage I/O requests by the processor. Section 5.8 of Volume 1 of the subject guide discusses programmed I/O and interrupt-driven I/O further.

Where candidates did understand what programmed I/O and interrupt-driven I/O were, they often lost marks by being vague, for example, many wrote that with interrupt-driven I/O the processor issues a command to an I/O module and then waits for an interrupt signal from the module, but did not state that while it is waiting the CPU does other things. Quite often the comparison between the two I/O protocols was also not clearly and explicitly made.

Even candidates who could describe programmed I/O and interrupt-driven I/O often gave answers to (ii) that were clearly guesses. Correct answers were rare. Again, this is discussed in Volume 1 of the subject guide, section 5.8.3, and is also a question in the sample examination question given in Chapter 5.

Question 4

a. Answer

i. (B) Application → Transport → Internet → Network Access

ii. (A) UDP

iii. (D) All of the above

Comments

Most candidates gave completely correct answers to part (a) with only a very small number of wrong answers seen.

b. Answer

i. When the time-to-live field reaches zero, the datagram is discarded and an ICMP message sent to the source host.

ii. *Any of the following:*

- 4 and 6 (or 0100 and 0110) could be in the version field.
- **Or** 4 only (or 0100), as this is an IPv4 header.
- **Or** the version field records if the datagram is from the IPv4 or IPv6 protocol.

iii. The total length field records the length of the datagram including header and data (in bytes).

Comments

Part (b) was answered well overall, although some lost credit for not including in their answer to part (i) that an ICMP message would be sent to the message source. Some answers to (iii) were unclear, with candidates stating that the field gave the total length of the data. It may be that these candidates were using 'data' to mean both the header and the data, but this needed to be made clear for full credit.

c. Answer

i. For full credit an answer should have included the following two points:

Data packets are expected to be received in order, and only data received in order will be acknowledged.

An acknowledgement confirms receipt of all unacknowledged data received with a smaller sequence number.

ii. Assume a situation where the receiver has received packet x–1, followed by packets x+1, x+2, followed by many other in-order packets. The receiver cannot signal to the sender that while it has not received packet x, it has received packet x+1, followed by a great many other in-order packets.

- If the sender follows the accepted standard and retransmits only the first unacknowledged segment, it must wait for the acknowledgement before it can decide what and how much to resend. Thus, retransmission reverts to a send-and-wait paradigm, slowing overall transmission time.
- Alternatively, the sender can send again all packets starting with packet x, even though many of them have already been successfully received. This is a potentially large overhead.

Comments

In answering part (i) a surprisingly large number of candidates stated that the receiver notifies the sender of packets that have not been received. In fact, the TCP protocol practises positive acknowledgement, which means that only packets correctly received are acknowledged, there is no notification of missing packets allowed for in the original protocol. There is a scheme called SACK, the Selective ACKnowledgement scheme, which allows for a receiver to give details of gaps in received bytes, but this is not mandatory, and the scheme does not replace cumulative acknowledgement.

For full credit candidates needed to give an answer that contained the two numbered points given above, although most candidates gave an answer that only covered one of the two points. Some candidates needed to note that a description of the 3-way handshake, used to establish a TCP connection, is not a description of cumulative acknowledgement.

In answer to part (ii) most candidates only made the point that where packets were received out of order, the missing packet may be sent again, together with all other packets sent after that packet, a potentially large overhead. These candidates received most of the credit for part (ii), but for full credit should also have covered the first bullet point above.

Question 5

a. Answer

i. (A) SMTP

ii. (B) Inline; Document level; External style sheet

OR

(C) External style sheet; Document level; Inline

iii. (B) 201.168.67.0/26

Comments

Mostly correct answers to part (a) were seen. In part (ii) most candidates answered (B), with a minority answering (C). Since the question implied but did not state clearly that the order of precedence should be from highest to lowest, the examiners decided to accept both (B) and (C) as correct.

Most candidates answered (iii) correctly, although a number of wrong answers were given. CIDR notation consists of the network address, followed by a decimal number indicating the number of leading 1s in the subnet mask when given in binary octets. The subnet mask is 255.255.255.192, which is 11111111 11111111 11111111 11000000 in binary octets, hence there are 26 1s. The number 26 indicates the number of bits that are to be considered as part of the network address, with the rest of the bits in the network address borrowed for use in assigning subnets.

b. Answer

i. The subnet mask is 255.255.255.240

= 11111111 11111111 11111111 11110000.

The number of bits borrowed from the network address is 4.

Hence the number of possible subnets is $2^4 = 16$ (accept $2^4 - 2 = 14$)

ii. The number of bits representing the host is 4, hence the number of possible hosts in each subnet is $2^4 - 2 = 14$

iii. The address of the first subnet is 220.108.192.0

host addresses 220.108.192.1 – 220.108.192.14

Also accepted: first usable subnet is 220.108.192.16

host addresses 220.108.192.17 – 220.108.192.30

Comments

Many correct answers to part (b) were given, but at the same time a large minority of candidates had no idea how to approach this question.

In considering answers to part (b) the examiners were aware that the current subject guide states that out of every set of possible subnets, 2, comprising the all 1s and all zero addresses, will be unusable for technical reasons. Since the guide was published the situation has changed, hardware has been designed to cope with all subnet addresses. Because of this the examiners decided to accept either 2^4 or $2^4–2$ as the correct answer to (i), and to accept the address of the first subnet in part (iii) as 220.108.192.0 or 220.108.192.16. Similarly, there are two possible ranges of host addresses; both were accepted as correct.

Most candidates answered part (ii) correctly, while a few lost credit by answering $2^4 = 16$. While it is the case that the all zeros and the all 1s subnet addresses can now be used, it is not the case that the all zeros and all 1s host addresses can be used, hence 2 must be subtracted from the potential host addresses to give all possible host addresses.

c. Answer

i. (A) External style sheet

(B) Document level

(C) Inline

ii. For full credit an answer should have included the following three points:

- A CSS file, with the house style, can be linked to by all persons and departments developing and maintaining web pages for the organisation. Hence the work of specifying the style only needs to be done once, rather than separately for each web page.
- CSS separates style and content or equivalent, e.g. one department can specialise in design, while other departments focus on their content.
- If changes to the house style are needed, then instead of changing individually possibly hundreds of web pages, only the content of the CSS file needs to be changed.

Comments

Most answers to (i) were correct, and all answers to (ii) gained some credit. Many candidates lost credit in part (ii) by giving answers that were too short and not detailed enough, and hence did not cover the three bullet points above. For example, some candidates stated that changing the house style need only be done in one place, the CSS file, without ever explaining how the CSS file defines and enforces the house style, because all web pages link to it.

Question 6

a. Answer

i. (C) A malicious self-replicating program that attaches itself to legitimate programs

ii. (D) All of the above

iii. (B) Copyright is automatically granted to an author

Comments

Mostly correct answers to part (a) were given, although a large minority of candidates gave an incorrect answer to part (ii) since they did not understand that all of the statements (A), (B) and (C) were correct, making (D) the correct answer.

b. Answer

i. A RAT is a Remote Access Trojan, a particular kind of Trojan payload that, once installed, gives a hacker complete control of a computer. A RAT uses a client/server system to gain illicit control of computers: the server is somehow installed on the victim's computer and attempts to contact the hacker's system (the client).

ii. A number of computers with the same RAT installed may be networked, so that the hacker can send one command that all machines will carry out. This can mean a DDoS attack from potentially millions of devices can be launched with one command.

iii. A DoS or DDoS attack is hard to defend against because it exploits features of legitimate protocols. Since the network requests are legitimate, it is very hard to distinguish between requests from genuine users, and those with a malicious intent.

Comments

Part (i) was often answered by candidates who had no idea what RAT stood for, and made various incorrect guesses. These candidates also struggled to give a sensible answer to part (ii). Those candidates who did understand what a RAT was, often answered part (ii) by stating that a DDoS attack could be used as a distraction while a RAT is installed. To receive any credit candidates needed to at least explain that many machines infected with the same RAT could be used to launch a DDoS attack.

In part (iii) many candidates considered that the reason that DoS attacks are hard to defend against is that it is impossible to predict when an attack may happen, or that it is because it is hard to know where the attack originates from. In fact, the examiners were looking for candidates to answer that a DoS or DDoS attack is hard to defend against because the network requests are normally legitimate, hence it is hard to tell which requests are from legitimate users, and which are malicious. Candidates were given some credit for discussing protection and mitigation of DoS attacks; for example, some candidates made the points that: (1) anti-virus software and firewalls may not be effective against DoS attacks; (2) that one defence can be to block or divert requests, but this also affects legitimate users; and (3) another defence is to increase the bandwidth of the server.

c. Answer

There is no right or wrong answer to this question. A good answer could argue for or against software patenting, providing that the arguments made were sound. The examiners were looking for answers that made sensible, relevant points, deployed coherent argumentation, showed depth of knowledge and understanding, were well-structured and written with clarity.

Comments

Candidates lost marks for this question by putting forward arguments that were undeveloped and lacking in consideration of counter arguments. For example, many candidates argued that patenting is necessary, as it means that developers can profit from their software, without seeming to be aware that many companies sell and profit from software that is not patented, and cannot be patented in most jurisdictions.

Many candidates wrote about the practice of patent trolling, but were not clear that this is mainly a problem in the USA. These candidates also did not explore whether allowing the patenting of software would inevitably encourage patent trolling, or whether patent trolling was a peculiarly American phenomenon. Many candidates made the point that patents guarantee the right to exploit an invention, which in turn safeguarded the income of inventors. Other candidates explored more nuanced arguments to do with the expense of getting and enforcing a patent, meaning that often patent protection is only available to larger companies, and that smaller businesses may be at a disadvantage, or may even be pushed out of the market altogether.