Section A

1. **Caches**

a) What are the three categories of cache misses? Explain under which circumstances each type of cache miss will occur. For each type of cache miss, identify a cache configuration where they would be the most likely to occur.

(5 marks)

Solution: [Bookwork]
Up to 5 marks from the following. Example answers below – any valid answer accepted.

Compulsory misses: these occur when the cache is cold (e.g., when the program starts, after the cache has been flushed, etc.). (1)
These cache misses will occur in any type of cache. In particular, this is the only type of cache misses that would occur in an infinite cache. (1)

Capacity misses: this type will occur if the cache cannot hold the whole working set of a program. (1)
A fully associative cache will only have compulsory and capacity misses. (1)

Conflict misses: occur when a program accesses more memory locations mapping to a given set (I.e. with the same index bits in the cache) than the cache associativity level. (1)
This will most frequently occur in direct-mapped caches and set-associative caches. (1)

b) Describe the forms of locality exploited by caches. For each type of locality, propose a program optimization that would increase the availability of such locality.

(4 marks)

Solution: [Bookwork & Practical question]

Temporal locality: memory locations accessed recently are likely to be accessed again within a short time frame. (1)
To increase the temporal locality of a program: use stack rather than the heap; if a data structure is traversed repeatedly, divide it in blocks that fit in the cache; reduce the working set of the program; etc. (1 for any valid example; max 1).

Spatial locality: memory locations located close to recently accessed addresses are likely to be accessed as well. (1)
Spatial locality can be increased by using contiguous data structures (e.g., arrays); by traversing data structures linearly in memory (e.g., traversing row-major matrices by rows rather than by columns); etc. (1 for any valid example; max 1).

c) The Intel manual for the Skylake architecture describes the following cache organization.

First level data cache (L1D): 32 kilobytes (KB), 8-way set-associative with 64 bytes (B) cache line size and 4 cycles access latency (time required to access this cache).

Second level cache (L2): 256KB, 8-way set-associative with 64B cache line size and 12 cycles access latency.

We will assume that the RAM access time is 200 cycles.

1) Considering that you are developing an application for which you have measured a 90% cache hit rate in L1D and also 90% cache hit rate in L2, what is the average memory access time of your processor for that application? Show your reasoning. (3 marks)

Solution: [Bookwork & Problem solving]

Reason on 100 accesses: 100 accesses need to check for hit/miss in L1D, takes 400 cycles, then out of those 10% will miss and need to check hit/miss in L2, takes 120 cycles, and finally 10% will go to RAM, so 1 leftover access at 200 cycles. Total is 720 cycles, so 7.2 cycles per access on average.

Alternatively: apply the recursive equation as follows.

$$T = h_1*t_1 + (1-h_1)*(t_1 + (h_2*t_2 + (1-h_2)*(t_2+t_{RAM})))$$
$$= 0.9*4 + 0.1*(4+(0.9*12+0.1*212))$$
$$= 7.2 \text{ cycles}$$

When missing in the cache it is not only necessary to pay the memory access time, but also the cache access time to determine whether the memory location is already cached.

2) After optimizing your code to better exploit the first level of cache, you are now measuring an improved 95% L1D hit rate and still the same 90% L2 hit rate. What is the new average memory access time for your application ? Explain your reasoning.

(2 marks)

Solution: [Problem solving]

Reason on 1000 accesses: 1000 accesses need to check for hit/miss in L1D, takes 4000 cycles, then out of those 5% will miss and need to check hit/miss in L2, takes 600 cycles, and finally 10% will go to RAM, so 5 leftover access at 1000 cycles. Total is 5600 cycles, so 5.6 cycles per access on average.

Alternatively: apply the recursive equation as follows.

$$T = h_1*t_1 + (1-h_1)*(t_1 + (h_2*t_2 + (1-h_2)*(t_2+t_{RAM})))$$
$$= 0.95*4 + 0.05*(4+(0.9*12+0.1*212))$$
$$= 5.6 \text{ cycles}$$

3) One of your colleagues suggests a different optimization that does not improve the L1D hit rate, which remains at the original 90%, but eliminates all L2

cache misses. Is that better or worse than the previous optimization? Explain why. (2 mark)

Solution: [Problem solving]

Either notice that the 1/100 access to the RAM costing 200 cycles contributes precisely 2 cycles to the average access time calculated in question (c.1), so 7.2 – 2 = 5.2 cycles. Or alternatively compute as in (c.1) with the new L2 hit rate.
The new average access time is therefore lower than in question (c.2), so this new optimization is indeed better.

4) The Intel Skylake can additionally include a large level three (L3) cache with an access time of 44 cycles. Assuming that due to its large size the L3 cache achieves 100% hit rate on your application, would this additional level of cache change which of the two optimizations in questions (c.2) and (c.3) is the most beneficial? Explain and detail your reasoning. (4 marks)

Solution: [Problem solving]

If we assume 100% hit rate in the L3 cache, this means that there will be no more accesses to the RAM. The L3 access cost will simply replace the cost of accessing the RAM. (1 mark)

As the L2 hit rate in question (c.3) is also 100%, there is no difference in that scenario. (1 mark)

*In the case of question (c.2), the same calculation can be used, replacing 5*200 cycles going to RAM by 5*44 cycles going to L3, which gives an average of 4820/1000=4.82 cycles.*
We can conclude that the addition of the L3 cache leads to a reversal: the first optimization (c.2) is now better than the second (c.3). (2 marks)

2. **Virtualization and Storage**

a) Your company is in the process of renewing the staff desktop computers. The supplier is offering a 1 terabyte (TB) hard disk drive (HDD) as standard on each desktop or an upgrade to 2TB for £40. Alternatively, you have the choice of equipping each desktop with an extra 1TB HDD (so having two 1TB HDDs) also for an extra £40. Considering that there is no room or energy constraint in the desktop enclosure and both upgrades are within budget, which option would you choose? Detail your reasoning. (3 marks)

Solution: [Bookwork & Problem solving]
Answers that deviate will be accepted if they are satisfactorily argued with well-founded arguments.

Option 1 – Choose the two 1TB drives: this provides more flexibility between either deciding to use a RAID 0 and obtain the same 2TB capacity or use RAID 1 and improve the reliability of the storage. (2 marks)
In both cases, having two drives will improve the overall bandwidth available for reading and the RAID 0 will also improve it for writing. (1 mark).

Option 2 – Choose the 2TB drive: argue that the 2TB capacity is necessary and it is detrimental to choose a RAID 0 as this would slightly reduce the overall reliability. (3 marks)

b) A high-end enterprise server is equipped with a storage system comprised of 16 identical HDDs, each with a 2TB capacity and a sustained bandwidth of 200 MB/s. The HDDs are organised as a RAID 160.

1) Considering that the storage system is symmetrically structured (i.e., each level of the system is uniform), what are the possible configurations of this RAID system (i.e., distribution of HDDs across the different RAID levels)? Explain your reasoning. (2 marks)

Solution: [Bookwork & Problem-solving]

As the RAID 160 has a RAID 1 as the bottom layer and this layer is uniform, the 16 HDDs are organized as a RAID 60 of either 8 RAID 1s of 2 disks each, or 4 RAID 1s of 4 disks each, etc. (1 mark)

However, as a RAID 6 is meaningless below 3 disks and a RAID 0 needs at least 2 disks, the only possible configuration is "2-4-2", so RAID 1 across 2 disks, RAID 6 across 4 RAID 1 pairs and finally RAID 0 across 2 RAID 6s. (1 mark)

2) What is the overall storage capacity of this storage system? Explain. (1 mark)

Solution: [Bookwork & Practical question]

*Given that the RAID 1 does not increase capacity, that a 4-way RAID 6 uses two parity disks and two data disks therefore doubling storage, and finally a 2-way RAID 0 further doubling storage, the total capacity is 2TB * 1_{RAID0} * 2_{RAID6} * 2_{RAID0} = 8TB.*

3) What is the **maximum** number of HDDs that can fail **before** any data is lost? And what is the **minimum** number of HDDs that can **always safely fail** within this RAID configuration? Explain each answer. (4 marks)

Solution: [Problem-solving]

At most one disk in each mirror (RAID 1) can fail (8 disks) and an additional 2 disks can fail from each RAID6 (4 disks), so a total 12 disks can fail before losing data in the best case. (2 marks) This can be corroborated as this would leave 4 disks, which are necessary to store 8TB of data as computed in question (b.2) if there is no duplication.

*The minimum number of disks that can always safely fail is 5 disks. The reason is that we need at least one of the sides of the RAID0 to fail to lose data, which requires three of the disks in the RAID6 to fail. However as each is replicated, this needs 3*2=6 disks to fail. As long as we lose one less disk, this RAID array is guaranteed not to lose data. (2 marks)*

4) Considering that all of the disks from the first scenario in the question (b.3) above have failed, how long will the recovery of this RAID system take once all failed HDDs have been replaced with new, empty ones? Consider that the machine is isolated and is not performing any other task. Detail your reasoning. (4 marks)

Solution: [Bookwork & Practical question]

If we consider the case where the maximum possible disks have failed without data loss, this means that there is only effectively one copy of the entire data (no replication is left). (1 mark)
The time required to reconstruct all of the parity disks will therefore depend on the time needed to read this data and write the parity, which is equal to the time needed to read or write a full 2TB disk. (1 mark)

$$T = 2TB/(200MB/s) = 10{,}000s = 2.78h$$

All of the parity disks can be written in parallel, and all of the data disks can be read in parallel, so the total is 2.78 hours (accept ~3 hours or any other rounded time format). (2 marks)

c) Define the term “checkpointing and restoring”, also called “snapshot and rollback”, explain how it can be implemented and optimized using System Virtualization. (4 marks)

Solution: [Bookwork]
(Max 4 out of the following, as covered in lectures)

When the VMM/Hypervisor pauses a Virtual Machine, enough state is saved to be able to resume the VM later. (1) Checkpointing saves this state (snapshot) in a file, alongside an image of the “physical” memory of the VM and its configuration, for later usage. (1) Restoring this checkpoint (rollback) is a matter of moving the memory image to the correct place, reloading relevant Hypervisor state from that stored earlier, and resuming execution. (1) Successive incremental checkpoints can be preserved without saving a new full memory image, instead only saving the differences in state since the previous checkpoint, e.g., using copy-on-write or marking modified memory pages as dirty. (1) Restoring the last checkpoint can then be achieved by restoring the memory pages that have been modified (marked dirty) since the last checkpoint. (1)

d) What are possible uses for checkpointing and restoring? Cite at least two.
(2 marks)

Solution: [Bookwork & Practical question]
(Any reasonable answers accepted for 1 mark up to max 2.)

Checkpointing & restoring can be used for:

- time slicing to allow other VMs to run.

- implementing VM migration.

- load balancing.

- implementing rapid provisioning by saving a fully configured and started application ready to be resumed from an image.

- implementing a reverse debugger by saving regular checkpoints, rolling back to the last checkpoint preceding a desired execution point and then advancing to the target instruction.

Section B

3. **Processor architecture**

a) Consider the following set of instructions:

```
1.    LDR R1, X
2.    MUL R1, R1, R1
3.    LDR R2, Y
4.    MUL R2, R2, R2
5.    LDR R3, Z
6.    MUL R3, R3, R3
7.    ADD R1, R1, R2
8.    ADD R1, R1, R3
```

Identify the dependencies between instructions and discuss whether they are affecting the performance of this code.

How long will it take to run them in a classic 5-stage pipeline? Provide an explanation for your answer.

Calculate the Cycles-per-Instruction (CPI) ratio.

Assume all forms of forwarding are implemented and all Instructions and Data are available in the L1 cache. (8 Marks)

Solution: [Problem-solving]
There are many real data dependencies (RAW) in this code. Many of them are between the EX stages of the instructions and do not affect the performance of the code thanks to the forwarding paths (2 → 7), (4 → 7), (6 → 8) and (7 → 8). However, these between the MEM and EX stages of the following consecutive pairs of instructions: (1 → 2), (3 → 4) and (5 → 6) require 1 cycle penalty each. ***[4 Marks]***

This means that the overall execution time will be 8 cycles to execute 8 instructions, plus 4 cycles to go down the pipeline, plus 3 cycle penalties due to Data hazards; i.e. 15 cycles. ***[3 Marks]***

Hence, 8 instructions executed in 15 cycles gives us a CPI of 15/8 = 1.875 ***[1 Mark]***

b) Reorder the instructions so that you minimise the execution time. What would be the execution time now? And the CPI? (6 Marks)

Solution: [Problem-solving]
In order to avoid the aforementioned data dependencies, we need to separate the MULs from their respective LDRs by, at least, one instruction. There are many ways we could reorder the code, but the easiest would be to first LDR the 3 registers and then execute the MULs .
[3 Marks]

This means that, with No penalties, the execution time will be now 8 cycles to execute the 8 instructions, plus 4 cycles to go down the pipeline for a total of 12 cycles. ***[2 Marks]***

Similarly, now the CPI is 12/8= 1.5 ***[1 Mark]***

c) Discuss the differences between the Scoreboard and the Tomasulo Out-of-order processor architectures. (6 Marks)

Solution: [Bookwork]
Scoreboard follows a centralized design whereas Tomasulo is distributed based on reservation stations. ***[1 Mark]***

Tomasulo's decentralized structure allows for better scalability, i.e., a larger instruction window. ***[1 Mark]***

Tomasulo Reservation Stations perform register renaming which avoids WAR and WAW dependencies. Scoreboard stalls the pipeline upon WAW and stalls completion upon WAR. ***[2 Marks]***

Scoreboard units write results directly into the register bank which allows parallel writes in the same cycle. Tomasulo relies on the Common Data Bus and so, result writing is serialized. However, the CDB provides forwarding seamlessly as data will be delivered to the Register Bank at the same time as to any dependent Reservation Station. In Scoreboard, a cycle is wasted to write and then read from the Register Bank. ***[2 Marks]***

4. **Multicore architectures**

a) Explain the concept of cache coherence in the context of multicore processors and its importance in shared memory systems. (2 Marks)

Solution: [Bookwork] - ***1 Mark each***
In multicore environments the main memory is shared by the different cores. However, if a cache hierarchy is implemented, each core will have copies of main memory's data in their local caches. Keeping all these copies up-to-date is what is known as cache coherence.

Cache coherency is important because if no precaution is taken, two cores could see different values when accessing the same variable which is against the semantics/principles of shared memory systems which may lead to unstable or erroneous execution.

b) Discuss the similarities and differences between Snoopy and Directory cache coherence protocols in terms of the variety protocols they support, where and how status information is stored, their communication infrastructure and their scalability. (8 Marks)

Solution: [Bookwork] - ***1 Mark each up to 8 Marks***
Similarities:
Both can implement a long list of protocols (MSI, MESI, MOESI, ...) ***[1 Mark]***
Both can use invalidate or update protocols. ***[1 Mark]***

Invalidate is usually preferred in order to minimise messages. ***[1 extra Mark]***

Differences:
In Snoopy all cores have a global view, whereas in directory-based the directory stores the global state. ***[1 Mark]***
Snoopy stores very simple info: cache state. Directory requires more complex info: cache state + directory state + sharing vector. ***[1 Mark]***
Snoopy relies on a shared medium (e.g. bus), directory uses point-to-point (network on chip). ***[1 Mark]***
Snoopy minimises the number of messages whereas directory requires many extra messages to access the directory and due to multicast serialisation ***[1 Mark]***
BUT Bus can transmit a single message at once whereas the NoC supports parallel message transmission. ***[1 Mark]***
Snoopy has limited scalability, whereas Directory scales better ***[1 Mark]***
Snoopy scalability can be improved by a hierarchy of buses. Directory can be further scaled up by using a distributed directory ***[1 extra Mark]***

c) We are planning to replace the school server supporting our teaching infrastructure. We contacted two separate providers and each of them provided the best configuration within our budget. The specifications of the servers are as follows:

	Provider #1	**Provider #2**
# of cores	8 out-of-order cores	36 in-order cores
Multithreading	2 threads - SMT	No Multithreading
Superscalar	4-way	3-way
Clock freq.	2 GHz	1 GHz
RAM	128GB	192GB
Cache Hierarchy	L1: 64KB dedicated L2: 256KB dedicated L3: 32MB Shared	L1: 64KB dedicated L2: 256KB dedicated L3: 24MB Shared

Compute the following metrics for each of the systems: (8 Marks)

1) single-thread peak performance (instructions per second)
2) peak IPC (instructions per cycle) of the system
3) peak computing throughput of the system (instructions per second)
4) the total number of hardware threads supported by the system

Solution: [Problem-solving] - ***1 Marks per metric/config***

For single-thread performance we need to multiply the number of superscalar lanes by the clock frequency
For peak IPC, we need to multiply the number of cores times the maximum number of instructions that can be executed per cycle.
For the computing throughput we need to multiply the peak IPC and the clock frequency.
For the number of threads, we multiply the number of cores times the threads per core

Results are as follow

Provider # 1
Single-thread: 4 x 2 = 8 Ginstr/s
Peak IPC: 4 x 8 = 32 instr/cycle
Peak Throughput: 4 x 8 x 2 = 64 Ginstr/s
Number of threads: 8 x 2 = 16 threads

Provider # 2
Single-thread: 3 x 1 = 3Ginstr/s
Peak IPC: 3 x 36 = 108 instr/cycle
Peak Throughput: 3 x 36 x 1 = 108 Ginstr/s
Number of threads: 36 x 1 = 36 threads

d) In order to decide between these two alternatives, we have analysed the workloads that run on top of the current system. We found out that the run time is divided roughly evenly into 2 different phases: i) a strictly sequential one and ii) a fully parallel one. Given the disparity of the two alternative systems we modelled these workloads in a full system simulator and obtained per-thread IPCs of 2.7 and 1.5 for phase i) and ii), respectively for Provider #1 solution. Similarly, for Provider #2 system, we got and IPC of 1.4 in both phases. Based on these estimations, discuss which system would be able to provide a higher performance, justifying your decision numerically. (2 Marks)

Solution: [Problem-solving + Critique]

To compute the achievable performance (computing throughput) for each phase and system we need to multiply the per-thread IPC by the number of threads involved and the core frequency. Hence, we get the following: [1 Mark]

Provider #1
Phase i) 2.7 x 1 x 2 = 5.4 Ginstr/s
Phase ii) 1.5 x 16 x 2 = 48 Ginstr/s

Provider #2
Phase i) 1.4 x 1 x 1 = 1.4 Ginstr/s
Phase ii) 1.4 x 36 x 1 = 50.4 Ginstr/s

Based on these, Provider #1 system seems a much better option as its performance during phase i) will be substantially better (~4x), while the performance during phase ii) will be reduced very slightly (~5%). [1 Mark]

Discussions using the results in c) or in the table provided above may get 1 Mark if justified properly.