COMP25212

Two hours

No special instructions.

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

System Architecture

Date

Time

Please answer any THREE Questions from the FOUR questions provided

Use a SEPARATE answerbook for each SECTION

For full marks your answers should be concise as well as accurate.
Marks will be awarded for reasoning and method as well as being correct

---

The use of electronic calculators is permitted provided they
are not programmable and do not store text.

[PTO]

Section A

1. **Caches**

a) Explain why CPU caches are important in the design of modern computer architectures, highlighting what would happen in case a computer does not include caches in its design. (3 marks)

CPU clock frequencies are much faster than main memory access times. (1 mark)
Caches can be used to hold subsets of main memory, but the majority of program working sets, to lower effective memory access times to close to the CPU clock time. (1 mark)
CPU would be unable to run at full speed without the addition of the caches. (1 mark)

b) Explain the terms "Write Back" and "Write Through" in the context of a write operation to cached data. (2 marks)

When updating the content of a cache it is important to consider also the correspondent data in main memory.
"Write through" means that the memory is updated every time cache is written to. (1 mark)
"Write back" means that the memory is updated only when the data is displaced from the cache.(1 mark)

c) With regards to the types of cache describe:

i) The characteristic of a fully associative cache and the access details of such memory (2 marks)

In a fully associative cache, any memory location can appear in any cache location. (1 mark)
Every cache location needs to be searched on every access to see if a hit occurs. (1 mark)

ii) The characteristic of a direct mapped cache and the access details of such memory (2 marks)

In a direct mapped cache, each memory location can occur at only one location in the cache (1 mark), since the cache is indexed by the low-order bits of the memory address. This means that only one cache location needs to be searched on every memory access (1 mark)

iii) The characteristic of a set associative cache and a comparison to the fully associative and/or direct mapped cache (2 marks)

In a set-associative cache, each memory location can occur in only a small subset of *cache locations. (1 mark) A set-associative cache can be described as multiple direct-mapped caches operating in parallel. (1 mark)*

d) Why does cache memory give rise to functional problems in computer systems that implement Direct Memory Access facilities (DMA) for peripheral devices? Give an example of two problems and two possible solutions. (6 marks)

Caches hold a copy of sections of memory close to the CPU. (1 mark)
The simplest cache strategy assumes that memory is only accessed by CPU and is always consistent. (1 mark)
With a cache "Write back" policy, DMA may read from main memory a superceded data. (1 mark)
DMA may write to main memory a new value, but the caches hold an outdated version of the value. (1 mark)
The cache may be flushed before every DMA operation. (1 mark)
A snooping device may be placed on the memory bus to check the DMA operations and update the memory and the cache as needed. (1 mark)

e) Explain how cache misses can be categorized as Compulsory, Capacity and Conflict misses. (3 marks)

Compulsory miss: a miss on a memory location that could not possibly have been in the cache. (1 mark)
Capacity miss: a miss generated because the cache is full of other irems. (1 mark)
Conflict miss: a miss that occurs when two or more locations in main memory compete for the same locations in a cache. (1 mark)

2. **Storage and Virtualization**

a) Explain what "Seek time", "Search time" and "Transfer time" mean when used to describe hard disk operations. (3 x 2 marks)

Seek time: the time required to move the read/write heads until they are positioned over the correct cylinder; (2 marks)
Search time: the time required between ending the seek and the required data becoming available to read/write under the heads – on average half the time of one rotation; (2 marks)
Transfer time: the time to transfer the required data between the computer (usually memory) and the disk – the reciprocal of the transfer rate; (2 marks)

b) A modern desktop drive is specified as having a capacity of 6 terabytes, a transfer rate of 150 megabytes per second, a rotation rate of 7200 revolutions per minute and a mean seek time of 8 milliseconds. Use binary orders of magnitude (as specified by the Joint Electron Device Engineering Council)

i) How long on average would a transfer of 8 kilobytes take from a random position on the disk? (2 marks)

Starting from a random position:
*search time = 0.5 * (60 / 7200) (seconds) = 4.166 ms*
transfer time = 8 / 153600 (seconds) = 52 us

total time = seek time + search time + transfer time =
total time = 8 + 4.166 + 0.052 = 12.218 ms

ii) How long would it take to read the entire disk sequentially? (1 mark)

Seek and search time can be considered insignificant, as the transfer time dominates:
transfer time = 6291456 (MB) / 150 (MB/s) = 41943 (seconds)
about 11 hours and a half (1 mark)

c) What is the difference between process virtualization and system virtualization? (4 marks)

Process virtualization is the execution of one or more processes under the control of a virtualization layer, which, in turn, runs as an application with the support of an underlying operating system. (2 marks)
System virtualization is the execution of one or more complete operating systems under the control of a virtualization layer (VMM or hypervisor) which may run stand-alone or as an operating system application. (2 marks)

d) What are the major goals of system virtualization? (3 marks)

Up to 3 Marks of the following:

Workload consolidation, giving energy and management savings; (1 mark)
Rapid provisioning, by copying pre-configured virtual machines; (1 mark)
Transparent load balancing, by migrating active VMs from heavily-loaded systems to lightly-loaded systems; (1 mark)
High availability, by keeping a hot-standby VM continuously live-migrated from the live VM; (1 mark)

e) What is “live migration” and how this can be implemented using system virtualization? (4 marks)

Live migration is moving a VM from one hardware host system to another without significan downtime. (1 mark)
It is achieved by first copying all memory pages from the source VM to the destination, clearing the VMM's idea of a "dirty" for each page as it is copied. (1 mark)
Since the source VM continues to run, it will re-dirty pages that have been copied, so these pages need to be detected and updated at the destination. (1 mark)
When the set of "dirty" pages gets sufficiently small, the source VM is suspended, all remaining dirty pages and VM metadata are transferred to the destination, and the VM is resumed there (1 mark).

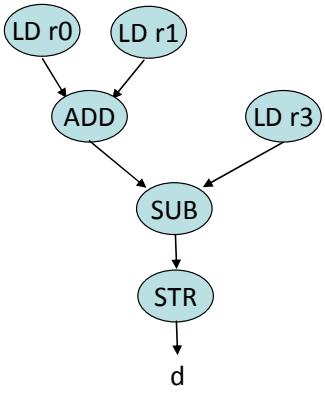
Section B

3. **Pipelining**

a) Draw the data-dependency graph for the following application. (2 marks)

```
ldr r1, x
ldr r2, y
add r4, r1, r2
ldr r3, z
sub r5, r4, r3
str r5, d
```

Solution: [NOTE: this is a problem-solving question]



b) Draw a space-time diagram simulating the execution of the application above in an in-order 2-way superscalar 5-stage pipeline. Assume all memory accesses hit the cache and that all types of forwarding are implemented. (4 marks)

Solution: [NOTE: this is a problem-solving question]

	t=0	t=1	t=2	t=3	t=4	t=5	t=6	t=7	t=8	t=9	t=10	t=11
ldr r1 x	IF	ID	EX	MEM	WB							
ldr r2 y	IF	ID	EX	MEM	WB							
add r4 r1 r2		IF	ID	stall	EX	MEM	WB					
ldr r3 z		IF	ID	stall	EX	MEM	WB					
sub r5 r4 r3			IF	stall	ID	EX	MEM	WB				
str r5 a			IF	stall	ID	EX	MEM	WB				

Applied penalties:
Instructions overpassing others (-3 marks)
Incorrect forwarding (-1 mark)
Data dependencies not observed (-4 marks)

c) Explain what extra hardware is needed to transform a classic 5-stage pipeline into an out-of-order pipeline. (4 marks)

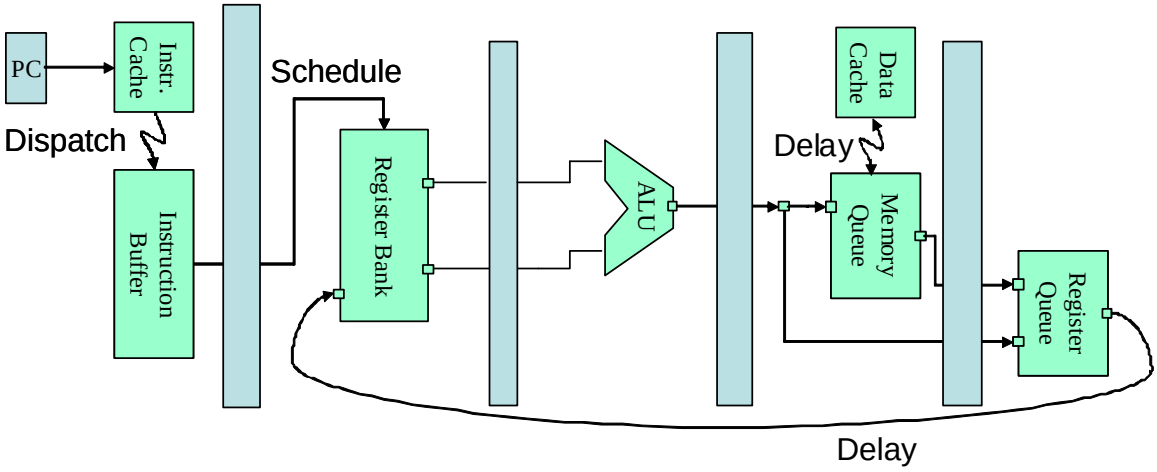
[Bookwork]

Instruction Issue: ***(2 marks)***

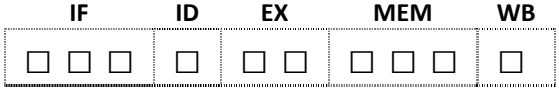
*Add a **dynamic scheduling unit** able to detect dependencies and dispatch independent instructions; e.g. Instruction buffer+schedule (below), scoreboard, reservation stations.*

Results management: ***(2 marks)***

Add some logic to the writing of results to avoid output conflicts (WAW, WAR); e.g. Memory and Register queues (below), stalling units, or register renaming.



d) Consider a 10-stage fully pipelined processor as the one below.



(IF and MEM: 3 stages each; EX: 2 stages; ID and WB: 1 stage each)

i) How many cycle penalties will be incurred by the different kinds of Hazards in such processor? (6 marks)

[Problem-solving]

Data Hazards:

Up to 5 cycle penalty (1EX+3MEM+1WB) for 2 consecutive instructions with data dependencies if forwarding is not implemented. 1st instruction needs to WB before the 2nd advances to EX (see below). (2 marks)

	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*
add r1,r0,r3	*IF*	*IF*	*IF*	*ID*	*EX*	***EX***	***MEM***	***MEM***	***MEM***	***WB***						
sub r1,r0,r3		*IF*	*IF*	*IF*	*ID*	*stall*	*stall*	*stall*	*stall*	*stall*	*EX*	*EX*	*MEM*	*MEM*	*MEM*	*WB*

Control Hazards:

Unconditional branches: 3 cycles (2IF + 1ID) (2 marks)

	1	*2*	*3*	*4*	*5*	*6*	*7*
b n	*IF*	***IF***	***IF***	***ID***	*EX*	*EX*	*MEM*
inst 1		*IF*	*IF*	*IF*	*ID*	*EX*	*EX*
inst 2			*IF*	*IF*	*IF*	*ID*	*EX*
inst 3				*IF*	*IF*	*IF*	*ID*
n					*IF*	*IF*	*IF*

Conditional branches: 5 cycles (2 IF + 1 ID + 2 EX) (2 marks)

	1	*2*	*3*	*4*	*5*	*6*	*7*
Beq n	*IF*	***IF***	***IF***	***ID***	***EX***	***EX***	*MEM*
inst 1		*IF*	*IF*	*IF*	*ID*	*EX*	*EX*
inst 2			*IF*	*IF*	*IF*	*ID*	*EX*
inst 3				*IF*	*IF*	*IF*	*ID*
inst 4					*IF*	*IF*	*IF*
inst 5						*IF*	*IF*
n							*IF*

Other well-founded answers might get accepted as well.

ii) Discuss the techniques to mitigate/solve these penalties. (4 marks)

[Bookwork]
Data Hazard: To reduce the impact of data hazards, forwarding/bypassing can be used. It consists in adding connections from each stage after EX (i.e. 7-10) to the input of the EX (5) and MEM (7) stages. (1 mark)
Reordering instructions during compilation can help to reduce the number of pipeline stalls by separating enough cycles those instructions that have data dependencies. (1 mark)

Control Hazard:
The solution is either to stall the pipeline once a branch instruction is decoded, or to instrument the compiler to add NOP instructions after each branch instruction. (1 mark)
A more advanced solution is to use branch prediction and perform speculative execution, but a roll-back mechanism needs to be implemented to undo any changes made if prediction fails. (1 mark)

Incorrect hazards or solutions penalise the total mark

4. **Multithreading / Multicore**

Consider a modern 6-core processor with 2-way multithreading and 4-way superscalar pipelines.

a) What is this processor's peak IPC? (2 marks)

[Problem-solving]
6 cores, each of them with a 4-way superscalar can issue/commit up to 24 instructions in a single cycle.

b) How many concurrent threads are supported by this processor? (2 marks)

[Problem-solving]
6 cores running 2 threads each: 12 in total

c) Assume the processor has a MESI cache coherency protocol with copy back. For the following sequence of accesses to variable 'x', show the bus transactions, the cache states and the actions in main memory. Assume all the corresponding cache lines start in the 'I' state. (10 marks)

core0: LDR r0, x

[Bookwork]
Read miss: Cache0 sends a MEM_READ. No other cache has it, so main memory sends the value. Cache0 changes to 'E' (2 marks)

core1: LDR r1, x

Read miss: Cache1 sends a MEM_READ. Cache0 has the cache line, so sends the value. Cache0 and Cache1 change to 'S' (2 marks)

core2: STR r0, x

Write miss: Cache2 sends a RWITW. Cache0 and Cache1 change to 'I', main memory sends the value and Cache2 changes to 'M' (2 marks)

core3: LDR r3, x

Read miss: Cache3 sends a MEM_READ. Cache2 has the cache line, so sends the value to Cache3 and updates main memory. Cache3 and Cache2 change to 'S' (2 marks)

core3: STR r5, x

Write hit: Cache3 sends an INVALIDATE. Cache2 changes to 'I', Cache3 changes to 'M' (2 marks)

d) Explain the benefits and limitations of reordering instructions in the compiler or in hardware. (6 marks)

[Bookwork]
1 mark for each correctly identified pro/con e.g.

Compiler
Pros: simpler hardware design (i.e. less area and power), static analysis can be more detailed (i.e. larger window of instruction)
Cons: binary tied to a particular hardware (lack of portability), extra instructions (NOPS)

Hardware
Pros: Legacy compatibility, Simple compiler, application independent
Cons: Hardware complexity affects freq. power, etc, small window of instructions, precise interruptions are not possible.

END OF EXAMINATION