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 the role of CPU caches in modern computer systems. (2 marks)

CPU clock frequencies are much faster than main memory access times. (1) Caches can be used to hold subsets of main memory, often sufficient to cover a program's working set, lowering the effective/average memory access time closer to the CPU clock time. (1)

b) Explain the differences between the two forms of locality exploited by caches and how they relate to key characteristics of CPU caches. (5 marks)

Caches exploit temporal and spatial locality. (1)
Temporal locality means that memory locations accessed recently are likely to be accessed again within a short time span. (1) Temporal locality is related to cache size, associativity and replacement policy, because these characteristics determine how long a cache line is likely to stay in the cache before being evicted. (1)
Spatial locality means that memory locations located close to recently accessed addresses are likely to be accessed as well. (1) Spatial locality depends on cache line/block size, which determines the amount of data close to a requested memory address that will be automatically fetched into the cache. (1)

c) Give one example for each of the two types of locality occurring in programs. (2 marks)

Temporal locality: instructions in a loop, instructions in frequently called functions, data in the stack frame (local variables) … (1 for any correct answer, max 1)
Spatial locality: sequence of instructions, data in arrays, structured data (fields in a struct or class), data in the stack … (1 for any correct answer, max 1)

d) Explain why modern computer systems frequently rely on multiple levels of cache. (4 marks)

With a single level of cache, the cache miss penalty is high enough that even at low miss rates the average memory access time is significantly higher than the CPU clock speed. (2)
Reducing the average latency can be achieved either by improving the hit rate of the cache or by reducing the miss penalty. (1)
Each additional level of cache allows to reduce the miss penalty of the previous level, ultimately reducing the effective/average memory access time for the CPU. (1)

The elements of multidimensional arrays are stored contiguously in memory, for example, a 2-dimensional array A of size N*N and containing 4-byte integers will be stored one row after another so that the address of element A[i][j] can be computed as the address of A[0][0] plus (i*N*4 + j*4).
Assume that a CPU has a 4-way set associative L1 data cache of 16KB where the cache line size is 64 bytes and the replacement policy is LRU, and that the size of the array is N = 8192.

e) How many cache lines are in each set of this CPU's L1 data cache? (1 mark)

In a 4-way set associative cache, each set contains 4 cache lines.

f) How many lines within the L1 data cache will be used by a program repeatedly traversing a single column of the array (i.e., A[0][k], A[1][k], A[2][k] … A[N-1][k] where k is constant)? (3 marks)

As each row of the array occupies 32KB, which is a multiple of the L1 data cache size, all array elements within the same column will have the same index bits and will therefore be mapped to the same set. Only one set and therefore only 4 cache lines will be used when traversing one column.

g) What type of cache misses will this program experience? (1 marks)

This program will experience conflict misses.

h) If the CPU cache is fully associative rather than 4-way set associative, how many cache lines will be used and what type of cache misses will occur? (2 marks)

In a fully associative cache, all cache entries can be used for any address in memory, so all 256 cache entries will be used. (1)
The program will experience capacity misses in this case. (1)

2. **Virtualization and Storage**

a) What is "System Virtualisation"? (1 mark)

A means of running complete operating systems, library and application stacks under the control of a layer of software, known as a Virtual Machine Monitor or Hypervisor. (1)

b) What are the major goals of System Virtualisation? (2 marks)

Isolation of the guest running operating system from the exact details of the hardware system that it is running on, (1) for example the exact CPU type, the memory configuration, and the detailed nature of its peripheral devices. (1)

c) Explain what "live migration" means and how it can be implemented using System Virtualisation. (4 marks)

Live migration is moving a VM from one hardware host system to another without significant downtime. (1) 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) Since the source VM continues to run, it will re-dirty pages that have already been copied, so these need to be detected in a second scan and re-copied. (1) When the set of "dirty" pages gets sufficiently small, the source VM is suspended, all remaining dirty pages and VM metadata is copied to the destination VMM, and the VM resumed there. (1)

d) A server disk drive is specified as having a mean "seek time" of 4 milliseconds, a "rotation speed" of 10,000 revolutions per minute and a "transfer rate" of 200 megabytes per second.

1) Explain what each term in this specification means for the performance of disk operation. (3 marks)

The seek time measures the time it takes the head assembly to move to the track of the disk that will be accessed. (1)
The rotation speed determines the time it takes on average for a given disk sector to arrive under the head, also called "search time". (1)
The transfer rate determines the amount of data per unit of time that can be accessed sequentially once the head assembly is in position. (1)

2) How long on average would a transfer of 4 kilobytes take from a random position on the disk? And for 4 megabytes? (4 marks)

The time required to transfer data is "seek time" + "search time" + "transfer time" (1)
*At 10,000 RPM, the search time is ½ * 60/10000 = 3 milliseconds. (1 mark)*
A transfer of 4 kilobytes will take 4ms + 3ms + 4/200000s = 7.02 ms ("about 7ms" is acceptable) (1 mark)
A transfer of 4 megabytes will take 4ms + 3ms + 4/200s = 27ms (1 mark)

e) Explain the two main reasons, aside from increasing storage capacity, why more than one disk drive may be used in a single system and how these goals may be achieved. (6 marks)

The main reasons are to enhance performance and reliability. (2)
Performance is improved by striping the file system across multiple disks (RAID 0) which increases the transfer rates. (1)
Reliability can be improved either by mirroring (RAID 1) (1) or by storing parity data that allows reconstructing lost data in case of disk failure (RAID 2-6) (1).
(1 additional mark for identifying the corresponding RAID versions in each case).

Section B

3. **Pipelining**

Consider the following application implementing a simple cash till program:

```
total=0;
for (i=total_items; i>0; i--) {
  cost[i]=quantity[i]*price[i];
  total+=cost[i]*taxes[i];
}
total=total*discount;
```

And the following assembly code generated by the compiler:

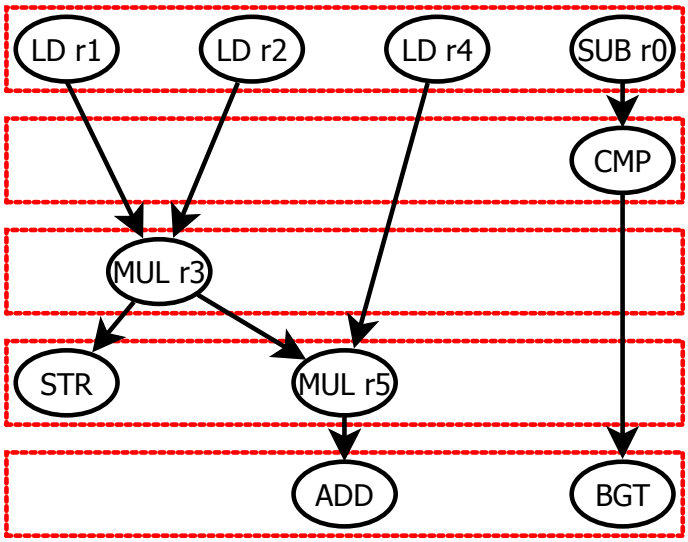
```
 1  init  adr r9, quantity     ; put addresses of arrays
 2        adr r10, price       ; in registers r9-r12
 3        adr r11, cost
 4        adr r12, taxes
 5        mov r6, #0           ; total (kept in register)
 6        ldr r0, total_items
 7  loop  ldr r1, [r9,r0]      ; quantity[i]
 8        ldr r2, [r10, r0]    ; price[i]
 9        mul r3, r1, r2       ; cost[i] (in register)
10        str r3, [r11,r0]     ; now in memory
11        ldr r4, [r12,r0]     ; taxes[i]
12        mul r5, r4, r3       ; cost[i]*taxes[i]
13        add r6, r5, r6       ; update total (in reg.)
14        sub r0, r0, #1       ; i--
15        cmp r0, #0
16        bgt loop             ; iterate while i>0
17        ldr r7, discount
18        mul r6, r6, r7       ; total * discount
19        str r6, total
```

a) Identify the possible hazards this code could suffer when run in a classic 5-stage in-order pipeline. Discuss what measures could be taken to avoid (or minimize) the penalties due to these hazards. You do not need to do the discussion in a case by case basis. (14 Marks)

Solution: [Bookwork + practical question]
*There are lots of **Data Hazards**, dependency among instructions, in the code: 6→7, 7&8→9, 9→10, 11→12, 12→13, 14→15, 15→16, 18→19 (**4 Marks**). Most of them are solved simply by using forwarding (**2 Marks**) but a few would require instructions to be reordered. 6→7 and 17→18 are solved by moving 17 after 6 outside of the loop (**2 Marks**). 8→9 and 11→12 are solved by moving 11 after 8 (**2 Marks**).*
*There is also a **Control hazard** (instruction 16) (**2 Marks**). Its effects can be mitigated by implementing a branch predictor that will potentially reduce significantly the 2-instruction penalty paid by fetching the wrong instructions. As this is a loop most of the time (especially if `total_items` is large) the branch will be taken and most implementation of predictors should be able to predict very accurately. (**2 Marks**)*

b) Draw the dependence graph of the instructions within the main body of the loop (instructions 7-16) and discuss how suitable this code would be for being executed in a 4-way superscalar processor. Assume structural hazards are never a problem. (6 Marks)

Solution: [Problem-solving]



The dependence graph of the loop body shows that there is not enough instruction level parallelism for the application to reach an ipc close to the theoretical maximum 4. In fact, the 10 instructions would take, at best, 5 cycles to be sent to execution (in a 4-way superscalar, 5-stage pipeline processor where LDs have 2 cycle dependencies) so ipc will be 2 in the best case. (4 Marks for the diagram + 2 Marks for the ILP discussion – students do not need to explain to this detail, but need to realise that dependencies limit the number of instructions that can be issued in parallel).

***Note:** Some students may realise that ILP can be improved by filling the gaps with instructions from other iterations or even mention more sophisticated methods such as loop unrolling. Such answers will be considered and marks will be granted on the grounds of how well the discussion is articulated.*

4. **Multithreading / Multicore**

a) Superscalar, multithreading and multicore are three different techniques used to improve the performance of processors. All of them exploit different forms of parallelism. Discuss the main differences between them and whether they are compatible with each other. (8 Marks)

Solution: [Bookwork]

A superscalar processor replicates execution logic to allow issuing (and executing) several instructions per cycle from a single instruction flow (thread). It exploits Instruction Level Parallelism. (2 Marks)

With multithreading the control and status logic (including register bank and TLBs) are replicated so that instructions from several, typically 2, instruction flows (threads) can be issued but the execution logic is shared among the different threads. It exploits Thread Level Parallelism by sharing the resources of a single processing core. (2 Marks)

Finally, with multicore all the processing logic (control + execution) is replicated so that the different instruction flows are executed independently. It exploits Thread Level Parallelism by duplicating all the core logic. (2 Marks)

All of them can be implemented in the same processor and indeed most modern processors implement all of them together. (e.g. Intel i7 Extreme 'Haswell-E' architecture has up to 8 cores with 2-way hyperthreading and 4-way superscalar: 8 cores, 16 threads, 32 peak IPC). (2 Marks)

b) Explain why instruction reordering can be useful to improve the performance of processors and discuss the benefits and limitations of doing it in the compiler or in hardware. (6 Marks)

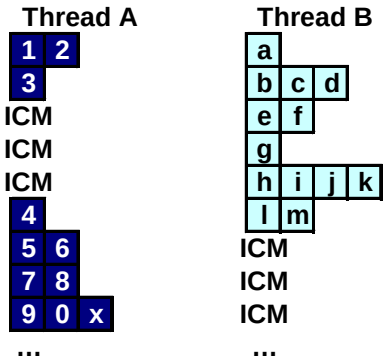
Solution: [Bookwork]

Instruction reordering can help to reduce the impact of data dependencies between instructions by placing independent instructions between two dependent instructions so that the processor can do useful work rather than stalling while the dependency lasts. (2 Marks)

In the compiler: (.5 marks per identified pro/con – up to 2 Marks)
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), some dependencies need to be calculated dynamically so static analysis may not be able to detect them

In hardware: (.5 marks per identified pro/con – up to 2 Marks)
Pros: Dynamic scheduling, Legacy compatibility, Simple compiler, application independent
Cons: Hardware complexity affects freq. power, etc, small window of instructions, precise interruptions are not possible.

c) Consider a 4-way superscalar processor with 2-way multithreading and 2 simple programs, A and B, with the instructions issued as per the diagrams below. ICM stands for instruction cache miss and means no instruction can be issued that cycle.



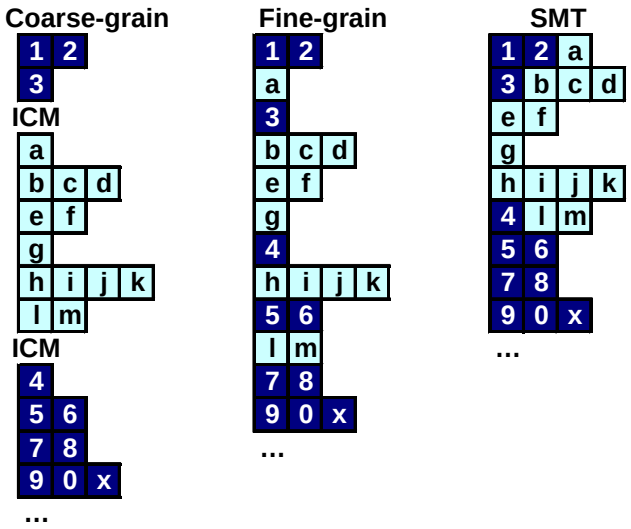
Draw a diagram showing the execution flow (issuing only) of the two threads if the processor used:

i) Coarse-grain multithreading
ii) Fine-grain multithreading
iii) Simultaneous multithreading

Assume the processor starts issuing instructions from A. (6 Marks)

Solution [Problem solving]

2 marks each.



END OF EXAMINATION