Birkbeck

(University of London)

BSc/FD EXAMINATION

Department of Computer Science and Information Systems

COMPUTER ORGANIZATION AND SYSTEM SOFTWARE (BUCI055H5)

CREDIT VALUE: 15 credits

Date of examination: 25/05/2017
Duration of paper: 14:30–16:30

There are **five** questions on this paper.

Answer **all** questions.

Each question carries **20** marks in total.

The use of supplementary material (notes, textbooks) and electronic calculators is **not** permitted.

1. Consider the following assembly code.

```
1. LOAD R, #1         // R <- 1
2. LOAD S, #1         // S <- 1
3. LOAD T, #(k-2)     // T <- k-2
4. ADD  AC, R, S      // AC <- R+S
5. T--                // T <- T-1
6. LOAD R, S          // R <- S
7. LOAD S, AC         // S <- AC
8. BRP  4, T
9. STOR M, AC         // M <- AC
```

where `R, S, T, AC` are registers, `M` is a memory location, `#` denotes numeric value and `BRP X, Y` stands for "branch to `X` if `Y` is positive".

(a) Explain what this code computes (assuming that `k` is a positive whole number greater than two).

(8 marks)

(b) Identify the various addressing modes in the code.

(6 marks)

(c) Identify the dependencies in the code.

(6 marks)

2. Consider the code in the previous question.

(a) Explain the idea of pipelining and the effect of (conditional) branch instructions on the pipeline.

(6 marks)

(b) Show the pipeline activity when the code is executed on a pipelined computer with input value `k=4`. There are five pipeline stages: fetch, decode, register read, execute and write back, and there is an instruction cache on board (initially empty) that can store ten decoded instructions. You can assume that certain instructions skip some of the stages, but make these assumptions explicit.

(14 marks)

3. (a) Explain the main idea of paging and its benefit over traditional (non-virtual) memory managements.

(6 marks)

(b) Explain what the page table (PT) and the translation lookaside buffer (TLB) are and how they are used in a paging system to compute physical addresses.

(6 marks)

(c) Assume that

- accessing and searching the TLB takes 5 ns,
- transferring the data from the TLB into a register takes 10 ns,
- updating a record in the TLB takes 15 ns,
- accessing and searching the PT takes 50 ns,
- transferring the data from the PT into a register takes 70 ns,
- the TLB hit ratio is 0.6.

How many nanoseconds are needed on average to perform all the TLB and PT operations needed to compute the physical address for a given virtual address (assuming that all referenced pages are in main memory).

(8 marks)

4. (a) Explain the importance of input/output (I/O) management in the context of operating systems.

(4 marks)

(b) Explain how RAID schemas help improving disk I/O.

(6 marks)

(c) A hard disk spins at 6000 rpm (revolution per minute), and it takes 100 μs (on average) for the head to traverse one track. Consider the following sequence of disk track requests: 27, 129, 110, 186, 147, 41, 10, 64, 120. Assume that initially the head is at track 30 and is moving in the direction of decreasing track number. Compute the time it takes to serve the requests using FIFO, SSF (shortest seek first) and SCAN (elevator) algorithms.

(10 marks)

5. Consider the following attempt to solve the dining philosophers problem for five

philosophers.

```
semaphore fork[5] = 1
semaphore s = 1
int i

void philosopher(int i)
{
 while(true){
  think();
  wait(s);
  wait(fork[i]);
  wait(fork[i + 1]  mod 5);
  signal(s);
  eat();
  wait(s);
  signal(fork[i]);
  signal(fork[i + 1]  mod 5);
  signal(s);
 }
}
```

(a) Explain what semaphores are and the effect of the wait and signal operations. Describe how semaphores can be used for concurrent programming.

(8 marks)

(b) Explain whether this code avoids deadlock. [Hint: Consider the case when a philosopher is eating and one of his neighbours is trying to pick up a fork.]

(8 marks)

(c) Explain whether this code avoids starvation. [Hint: Can a philosopher be permanently stuck?]

(4 marks)

