COMP25111

Two hours

Question A1 and Question B1 are COMPULSORY

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Operating Systems

Date: Monday 18th January 2016

Time: 14:00 - 16:00

Please answer Questions A1 and B1
and
TWO other Questions from A2, B2 or B3

Use a SEPARATE answerbook for each SECTION

This is a CLOSED book examination

The use of electronic calculators is NOT permitted

[PTO]

Section A

Please note that Question A1 is worth half the marks of the other question in this Section

A1. **Compulsory**

a) A web browser in the UK accesses a database in New Zealand. How long does it take for the data to arrive back in the UK after the user presses the 'access' key on the web page? For simplicity, assume that: (1) the times to process the browser key, to retrieve the data from the database, and to send and receive messages are all negligible; and (2) the round trip data transfer takes the same time as light to travel round the circumference of the Earth (a distance of 40,000 kilometres at a speed of 3×10^8 metres/second). Assume the computer running the browser has a clock rate of 3GHz and can execute one instruction per clock cycle. How many instructions could that computer execute while waiting for the response from the database? (2 marks)

b) Explain briefly the actions that must occur when a *context switch* happens in an operating system. (2 marks)

c) What does the term *deadlock* mean? How may deadlock occur? (2 marks)

d) What is the key difference between a *system call* and a call to a *library routine*? Briefly explain why this is important. (2 marks)

e) In Linux, how does a shell implement a *pipe* between commands? (2 marks)

A2. a) In the context of *non-preemptive* process scheduling for a single CPU, what are the three main states that a process may be in, and what are the three transitions amongst these states caused by? (3 marks)

b) What additional transition between states is possible in *preemptive* process scheduling for a single CPU? Why is this new transition useful? (2 marks)

c) Three processes, A, B and C, have the following computational needs: Process A requires a 7 time-unit CPU burst followed by a 4 time-unit I/O burst then a 7 time-unit CPU burst before it terminates; Process B requires a 5 time-unit CPU burst followed by a 4 time-unit I/O burst then a 5 time-unit CPU burst before it terminates; Process C requires a 3 time-unit CPU burst followed by a 6 time-unit I/O burst then a 3 time-unit CPU burst before it terminates. Draw a diagram showing the states of these processes as they are run by a non-preemptive First Come First Served (FCFS) scheduler until they all terminate, assuming that they all start ready at time-unit 0 and are queued in the order A (first), B, C (last), and the time for a context switch is negligible. What is the *average wait time* (the time between entering the ready queue for the first time and starting to run on the CPU) for the three processes? What is the *average turnround time* (the time between entering the ready queue for the first time and terminating) for the three processes? (3 marks)

d) The three processes in part c) are now subjected to a preemptive Round Robin (RR) scheduler with a time-slice of 3 time-units. Draw a diagram showing the states of the processes from start to finish. What is the average wait time for the three processes? What is the average turnround time for the three processes? (4 marks)

e) In a hierarchical file system, a *pathname* defines how a file is to be located in the filestore. Describe in detail the algorithm used to locate a file when given its full (absolute) pathname. (3 marks)

f) In a file system using *index nodes* (*i-nodes*), the value obtained by the algorithm in part e) is the identifier of the i-node that describes the target file. Explain how the i-node is found on disk. (2 marks)

g) Assume that each i-node contains up to 8 pointers to the first 8 blocks of the file on disk, then up to 4 pointers to *single indirect blocks* on disk that each contain up to 1024 pointers to the next blocks of the file on disk, and then up to one pointer to a *double indirect block* on disk that contains up to 1024 pointers to single indirect blocks on disk that each contain up to 1024 pointers to the next blocks of the file on disk. Any unused pointers contain a special 'null' value. What is the size (in blocks) of the largest possible file in this system? (3 marks)

[PTO]

Section B

Please note that Question B1 is worth half the marks of the other questions in this Section

B1. **Compulsory**

a) What is a *page replacement algorithm*? (2 marks)

b) In the context of operating systems, what is meant by the term *programmed I/O*? Give a brief answer with respect to a keyboard I/O device – flowcharts are not required. (2 marks)

c) An input/output (I/O) module in a computer system has an interface using *Direct Memory Access* (DMA). Explain what the DMA interface does. (2 marks)

d) One method for implementing virtual memory is *paged virtual memory*. The virtual memory translation procedure can be viewed as a series of steps. Explain the paged virtual memory translation procedure by outlining the sequence of steps for translating a virtual address to a physical address. (2 marks)

e) Contrast what is meant by the terms *multiprogramming* and *fixed partitions*. (2 marks)

B2. a) In the context of a *segmented memory*, name and briefly describe two alternative algorithms that can be used to determine where to place a new segment in a memory that has external fragmentation. (4 marks)

b) In the case of a computer system with real memory, explain why a multiprogramming operating system would need to *relocate* code. (4 marks)

c) On a paged machine with three page frames available, a particular process makes accesses to the following virtual pages in the order shown below.

Page number: 3 7 1 3 2 1 3 7 0

Show the contents of the three page frames and the cumulative total number of page faults (PFs) after each memory access assuming that a *least recently used* (LRU) page replacement algorithm is in use and that the page frames are initially empty. The kind of diagram you should produce is depicted in Figure 1, below. (4 marks)

Page access:	3	7	1	3	2	1	3	7	0
Most recent	X	X	X	X	X	X	X	X	X
Second most	X	X	X	X	X	X	X	X	X
Third most	X	X	X	X	X	X	X	X	X
PFs	X	X	X	X	X	X	X	X	X

Figure 1. Typical diagram showing three page frames and the cumulative total number of page faults (PFs).

d) Given that the overall actions that a CPU takes to handle an interrupt can be summarised in six basic steps, list and briefly describe the six basic steps. (4 marks)

e) Given a 4GB virtual address space and an associated 512KB page size, calculate the *number of pages* in this virtual address space. **NOTE**: To gain full marks you must show full working. (2 marks)

f) Given a physical address size of 1GB and an associated 256KB page size, calculate the *number of page frames* in the physical address space. **NOTE**: To gain full marks you must show full working. (2 marks)

[PTO]

B3. a) Given the simple page table diagram in Figure 2 (below) and the 32-bit virtual address 0x00040003: Determine the physical address given the data in the simple page table. (1 mark)

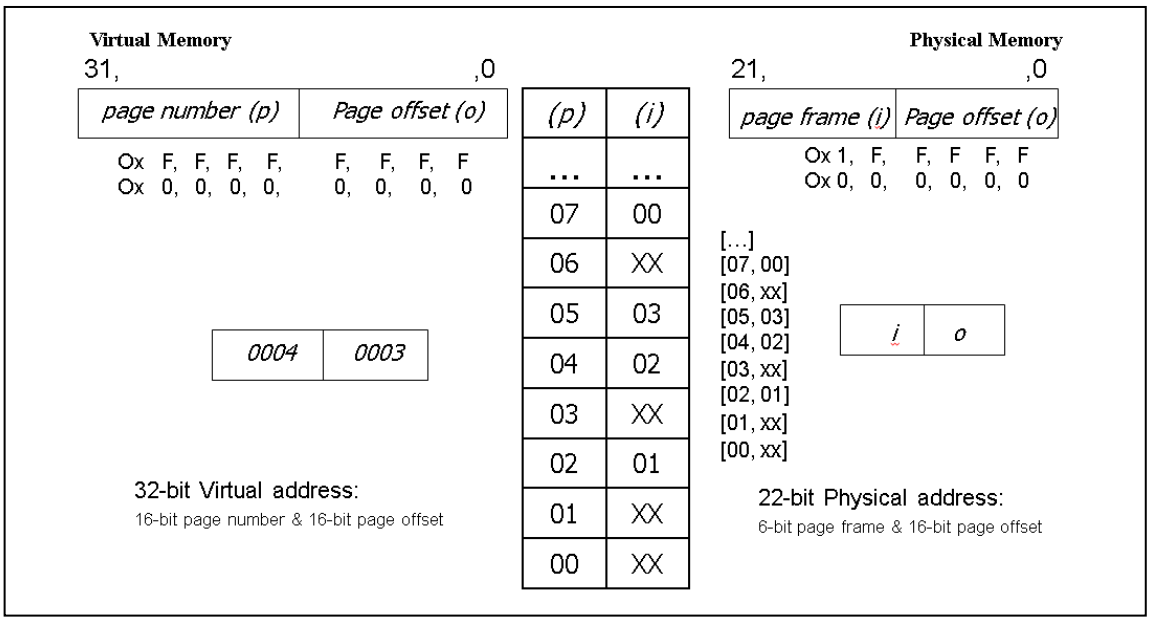

Figure 2. Simple page table diagram.

b) Demand paging can be viewed as a sequence of steps. Explain the demand paging procedure by outlining the sequence of steps. (4 marks)

c) State and briefly describe the steps an operating system takes to load a single user program into memory and then execute it, given this is *uniprogramming*. (5 marks)

d) Explain what is meant by a *critical region* of a multi-threaded program. Explain what is meant by *mutual exclusion* for a critical region. Explain what a *semaphore* is and describe the P (procure) and V (vacate) operations that can be performed on it. Explain how a semaphore can be used to enforce mutual exclusion for a critical region of a multi-threaded program. (4 marks)

(Question B3 continues on the following page)

(Question B3 continues from the previous page)

e) In a certain system, the execution of three threads, A, B and C, is synchronised using two semaphores, S1 and S2, as shown below. Both semaphores are initialised to zero and are used only in the sections of code shown below. The threads share the variables x and y.

Thread A	Thread B	Thread C
…	…	…
P(S1)	P(S2)	x = 0
x = x + 1	y = 2	V(S1)
x = x – 2	y = y – 1	V(S2)
…	…	y = 3
…	…	…

i) What happens to the variable x, and why? (1 mark)

ii) There can be more than one outcome for variable y. What can happen, and why? (2 marks)

iii) Rewrite the code above, keeping the assignments to x and y unchanged, but changing how and/or where S1 and S2 are used so that, when all three threads have finished, x has the same final value as before, and y has the value 1. Explain why your code achieves this. (1 mark)

iv) Rewrite the code above, keeping the assignments to x and y unchanged, but using any number of semaphores you need in any way so that, when all three threads have finished, x has the same final value as before, and y has the value 3. Use the minimum number of extra semaphores needed to achieve this. Explain why your code works. (2 marks)

END OF EXAMINATION