1995 Paper 10 Question 7

Operating System Foundations

Why are multiple buffers often used between producing and consuming processes? [4 marks]

Describe the operation of a *semaphore*. [4 marks]

What is the difference between a counting semaphore and a binary semaphore? [2 marks]

The code below is intended to synchronise buffer control between multiple producing processes and a single consuming process. `freeBuffer` and `fullBuffer` are semaphores. Describe an execution sequence which causes error.

Initialisation:

```
current_free := 0;
current_full := 0;
freeBuffer := buffers;
fullBuffer := 0;
```

Producers:

```
LOOP
     produce item;
     WAIT(freeBuffer);
     COPY (buffer[current_free], item);
     current_free := (current_free + 1) MOD buffers;
     SIGNAL(fullBuffer);
END;
```

Consumer:

```
LOOP
     WAIT(fullBuffer);
     COPY (item, buffer[current_full]);
     current_full := (current_full + 1) MOD buffers;
     SIGNAL (freeBuffer);
     consume item;
END;
```

[5 marks]

Modify the code to correct the problem. [5 marks]

1995 Paper 11 Question 7

Operating System Foundations

Consider the operation of a scheduler in a system where there are system level and user level processes. User processes may be IO bound or CPU bound and may have user controlled (negative) priority.

Describe the data structures that the scheduler might use, including parts of process descriptors that the scheduler would operate on. [10 marks]

Describe in detail the circumstances under which the scheduler would be entered and for each different circumstance outline a scheduling algorithm that might be used. [10 marks]

1996 Paper 10 Question 7

Operating System Foundations

What is meant by *memory-mapped I/O*? How might devices be protected from uncontrolled access? [3 marks]

Describe how I/O might be programmed with and without interrupts enabled. Discuss the implications of the use of these approaches within an operating system. Discuss how interrupt handling might be supported by hardware. [5 marks]

Why does a cache alleviate the possible performance penalty of using DMA? [3 marks]

Give an example involving interrupt-driven software which illustrates the danger of race conditions. [5 marks]

Outline how semaphore operations can be made atomic. [4 marks]

1996 Paper 11 Question 6

Operating System Foundations

Give a detailed criticism of the following options for supporting interactions between processes in separate address spaces:

(a) pipes

(b) asynchronous message passing

(c) synchronous message passing

[20 marks]

1997 Paper 10 Question 7

Operating System Foundations

In a given application, processes may be identified as "readers" or "writers". Multiple-reader, single-writer access to data is required to be implemented with priority for writers. Readers execute procedures `startread` and `endread` before and after reading. Writers execute `startwrite` and `endwrite`.

Give pseudocode for the `startread` procedure using

(a) semaphores

(b) conditional critical regions

(c) monitors

(d) guarded commands

Explain via comments in your pseudocode how concurrency control is achieved in each case and criticise each approach from the viewpoint of the programmer and the implementer.

[20 marks]

1997 Paper 11 Question 6

Operating System Foundations

Outline the components of interrupt handling and, where appropriate, discuss the tradeoffs of implementation by hardware or by software. [7 marks]

Give *two* examples of machine instructions on which mutual exclusion can be built. [2 marks]

Explain why temporarily suppressing interrupts can be used as a basis on which to build mutual exclusion for a uniprocessor but not for a multiprocessor. [2 marks]

For a uniprocessor describe how interrupt control, together with scheduling policy, can be used to protect writeable data shared by processes and hardware-driven routines. [3 marks]

Outline how user-level (top–down) requests for I/O can be synchronised with hardware-driven I/O software. [6 marks]

1998 Paper 10 Question 8

Operating System Foundations

You are to advise on the selection of an operating system on which network-accessible services are to be provided. Each service must be capable of supporting a specified number of simultaneous clients and multiprocessor hardware may be used.

Define and evaluate each of the following styles of support for multi-threaded processes:

(a) user threads only [10 marks]

(b) kernel threads only [10 marks]

[Hint: include selected data held in process and thread descriptors; mention calls supported at the thread-package and kernel interfaces.]

1998 Paper 11 Question 6

Operating System Foundations

A barber provides a hair-cutting service. He has a shop with two doors: an entrance and an exit. He spends his time serving customers one at a time. When none are in the shop, the barber sleeps in the barber's chair. When a customer arrives and finds the barber sleeping, the customer awakens the barber and sits in the barber's chair to receive his haircut. After the cut is done, the barber sees the customer out through the exit door. If the barber is busy when a customer arrives, the customer waits in one of the chairs provided for the purpose. If all the chairs are full he goes away. After serving a customer the barber looks to see whether any are waiting and if so proceeds to serve one of them. Otherwise, he sleeps again in his chair.

In this question we represent the barber and his customers as synchronising processes.

A solution for the barber using only semaphores is:

```
waiting : integer      := 0;  % customers waiting to be cut
guard : semaphore      := 1;  % delimits a critical region to protect waiting
customers : semaphore  := 0;  % counting semaphore of customers
barber : semaphore     := 0;  % barber waiting for a customer (1) or not (0)?
```

The barber executes the following program:

```
WAIT (customers);              % sleeps if there are none
WAIT (guard);
  waiting := waiting −1;       % otherwise changes waiting under exclusion
  SIGNAL (barber);             % and indicates his readiness to cut hair
SIGNAL (guard);
cut hair;
```

(*a*) Give the corresponding code for a customer. [6 marks]

(*b*) Give the corresponding solution using a monitor. [10 marks]

(*c*) To what extent are the difficulties of semaphore programming alleviated by the provision of monitors? [4 marks]

1999 Paper 10 Question 8

Operating System Foundations

An operating system contains event management and process management modules. The interface of the former includes WAIT (*event*) and SIGNAL (*event*) operations to achieve hardware–software synchronisation. The interface of the latter includes BLOCK (*process*) and UNBLOCK (*process*).

(*a*) Outline the data structures you would expect to find in these modules to support event and process management. [6 marks]

(*b*) Describe the interactions between these modules as the system runs. [4 marks]

(*c*) Discuss the problems that can arise if concurrent invocation of the event management module is possible. [5 marks]

(*d*) Describe how these problems may be avoided if the operating system is guaranteed to run on a uniprocessor. [2 marks]

(*e*) Describe how these problems can be solved for a multiprocessor implementation. [3 marks]

1999 Paper 11 Question 6

Operating System Foundations

Producer and consumer processes interact via an N-slot cyclic buffer. Semaphores are defined and initialised as follows:

$$\begin{aligned} lock &: semaphore &:= 1 \\ spaces &: semaphore &:= N \\ items &: semaphore &:= 0 \end{aligned}$$

For the following programs indicate where mutual exclusion and condition synchronisation are being attempted and explain how the system may fail.

producer code	consumer code
produce item	WAIT (*lock*)
WAIT (*lock*)	WAIT (*items*)
WAIT (*spaces*)	remove item
insert item	SIGNAL (*spaces*)
SIGNAL (*items*)	SIGNAL (*lock*)
SIGNAL (*lock*)	consume item

[8 marks]

Write a monitor to manage the N-slot buffer. Discuss why the problems you pointed out in the previous part do not arise in the monitor implementation. [12 marks]

2000 Paper 10 Question 8

Operating System Foundations

(a) A software module controls a car park of known capacity. Calls to the module's procedures *enter*() and *exit*() are triggered when cars enter and leave via the barriers.

Give pseudocode for the *enter* and *exit* procedures

(i) if the module is a monitor [8 marks]

(ii) if the programming language in which the module is written provides only semaphores [4 marks]

(b) Outline the implementation of

(i) semaphores [4 marks]

(ii) monitors [4 marks]

2000 Paper 11 Question 6

Operating System Foundations

Discuss approaches to process scheduling for the following systems.

(*a*) A hard real-time system where all processes are defined statically and are periodic with known work-time per period. [4 marks]

(*b*) A shared system which runs applications controlled interactively by users. [8 marks]

(*c*) A network-attached, multimedia workstation. [8 marks]

2001 Paper 10 Question 8

Operating System Foundations

(*a*) Describe how a pool of disc I/O buffers might also be used as a disc block cache. [2 marks]

(*b*) What are the advantages and disadvantages of a buffer-cache? [3 marks]

(*c*) Give examples of where concurrency control is needed in the management of the buffer-cache. [6 marks]

(*d*) Describe how application-level read and write requests are satisfied from the buffer-cache. Include any potential delay due to concurrency control. For simplicity you may assume that a request is satisfied from a single disc block. [Hint: you may find it useful to give some detail of the data held in a disc block header in the buffer-cache.] [9 marks]

2001 Paper 11 Question 6

Operating System Foundations

An interprocess communication environment is based on *synchronous* message passing. A server is to be designed to support a moderate number of simultaneous client requests.

Clients send a request message to the server, continue in parallel with server operation, then wait for the server's reply message.

Discuss the design of the server's interaction with the clients. Include any problems you foresee and discuss alternative solutions to them. [20 marks]

2002 Paper 10 Question 7

Operating System Foundations

(*a*) Explain briefly the memory-management scheme of *paging*. [4 marks]

(*b*) Give *two* disadvantages of paging. [2 marks]

(*c*) A translation look-aside buffer (TLB) is sometimes used to optimise paging systems. Explain carefully how a TLB can be used in this way, and how it can optimise a paging system. [3 marks]

(*d*) The fictional Letni 2P chip uses (single-level) paging and has a memory access time of 8 nanoseconds and a TLB search time of 2 nanoseconds. What hit ratio (the probability that an item is in the TLB) must be achieved if we require an average (paged) memory access time of 12 nanoseconds? [4 marks]

(*e*) The management of the Letni Corporation wish you to design and evaluate a *multi-level* paging system for their new 64-bit processor, the 3P, which has 4K-sized pages.

(*i*) Give details of your proposed multi-level paging system. [5 marks]

(*ii*) State, and justify briefly, whether you think this proposal is realistic. [2 marks]

2002 Paper 11 Question 5

Operating System Foundations

(*a*) Write brief notes on the following:

(*i*) Interrupt-based I/O. [3 marks]

(*ii*) Direct Memory Access. [3 marks]

(*iii*) File access control in Unix. [3 marks]

(*iv*) Mounting file systems in Unix. [3 marks]

(*b*) You are one of the design team of the new Xinu file system. As part of this design you have proposed an inode structure similar to that in "traditional" Unix. (Blocks are assumed to be 4KB, and file pointers are 4 bytes.)

(*i*) Give details of your proposal for the inode structure. [6 marks]

(*ii*) Calculate the maximum file size, in blocks. [2 marks]

2003 Paper 10 Question 6

Operating System Foundations

(*a*) (*i*) What is meant by the *address space* of a process?

(*ii*) Give an example of how a 32-bit address space might be allocated between the various components of user and operating system (OS) code and data. You may assume that the OS occupies half the address space of every process.

(*iii*) Why does the OS region containing memory-mapped I/O interfaces need to be distinguished? What is the alternative to memory-mapped I/O?

(*iv*) How might the fact that much OS code must be permanently resident in memory be used to advantage?

(*v*) How is the OS protected from user level code at runtime?

(*vi*) How is the OS executed synchronously via calls from user level?

[13 marks]

(*b*) In what way did the provision of a protected address space per process affect the development of operating systems? Describe the hardware and software support for the development you mention in some detail. [7 marks]

2003 Paper 11 Question 6

Operating System Foundations

(*a*) By means of an example, show why concurrency control, comprising both mutual exclusion and condition synchronisation, is needed in operating systems. [4 marks]

(*b*) Explain why, in general, forbidding interrupts is not appropriate as a basis for implementing concurrency control. [2 marks]

(*c*) How can a "read-and-clear" machine instruction be used as a basis for mutual exclusion and condition synchronisation? [4 marks]

(*d*) Define *semaphores* and discuss how they can be implemented. [6 marks]

(*e*) How can semaphores be used to achieve mutual exclusion and condition synchronisation? [4 marks]

2004 Paper 10 Question 6

Operating System Foundations

(*a*) Describe a scheduling algorithm with the following properties:

- favours I/O-intensive processes
- responds dynamically when processes change their behaviour: e.g. enter a compute-bound or I/O-intensive phase
- has acceptable context switching overhead
- avoids indefinite overlook (starvation) of a process

[7 marks]

(*b*) In order to carry out its functions, a filing system holds metadata on each stored object.

(*i*) What is this metadata likely to comprise? [6 marks]

(*ii*) Describe the directory service functions of a filing system, including how the metadata is used. [7 marks]

2004 Paper 11 Question 6

Operating System Foundations

Two operating systems OS-A and OS-B offer only synchronous system calls, for example, for I/O. In addition, OS-A supports only one process per user-level address-space whereas OS-B supports multi-threaded applications.

(a) (i) Explain how an application-level runtime system or library running on OS-A can provide the user threads needed by concurrent programs. [8 marks]

(ii) Discuss any disadvantages of supporting a concurrent programming language in this way. [3 marks]

(iii) Are there any advantages of having only user threads? [1 mark]

(b) (i) Explain the *differences* from the runtime described for OS-A of a runtime for OS-B which maps user threads to kernel threads. [5 marks]

(ii) Are the disadvantages you discussed in part (a)(ii) overcome? Explain. [2 marks]

(iii) Have any problems been introduced by the use of kernel threads? [1 mark]

2005 Paper 10 Question 6

Operating System Foundations

(*a*) A device driver process carries out character I/O via a Universal Asynchronous Receiver/Transmitter (UART).

(*i*) Why is hardware–software synchronisation needed? [1 mark]

(*ii*) Describe polled operation. [2 marks]

(*iii*) Describe interrupt-driven operation. [2 marks]

(*iv*) Draw a state transition diagram for the device-driver process. Indicate the events that cause each transition and in each case explain the effect on the device driver's process descriptor and the operating system's scheduling queues. Assume interrupt-driven software. [7 marks]

(*b*) The device driver process fills/empties a buffer of fixed size in an I/O buffer area. A process carrying out application requests reads and writes data in variable-sized amounts from the buffer.

(*i*) Why must mutually exclusive access to the buffer be enforced? [2 marks]

(*ii*) Why is condition synchronisation needed? [2 marks]

(*iii*) What is wrong with the following pseudocode fragment from the device-driver's specification, where:

- `buffer-lock` is a semaphore initialised to `1`,
- `space` is a semaphore initialised to the buffer size in bytes,
- `data` is a semaphore initialised to `0`?

on input:	on output:
`WAIT(buffer-lock);`	`WAIT(buffer-lock);`
`if` buffer is full `then WAIT(space);`	`if` buffer is empty `then WAIT(data);`
write a character into the buffer;	read a character from the buffer;
`SIGNAL(buffer-lock);`	`SIGNAL(buffer-lock);`

[4 marks]

2005 Paper 11 Question 6

Operating System Foundations

(a) A system has paging hardware but no segmentation hardware. Discuss the likely structure of a process page table. What information would you expect to be held in a page table entry? [5 marks]

(b) Describe the operation of the following hardware support options for paging:

(i) A pair of processor registers.
PTBR holds the address of the base of the page-table of the current process.
PTLR holds the size of the page table in bytes. [5 marks]

(ii) A TLB (Translation Lookaside Buffer). [10 marks]

2006 Paper 10 Question 7

Operating System Foundations

(*a*) Describe the steps involved in resolving any component of a file pathname. [6 marks]

(*b*) (*i*) Describe the steps involved in

- creating a file;
- deleting a file. [8 marks]

(*ii*) Discuss the possible effects of a crash, causing loss of main memory, at various points during file creation and deletion. [6 marks]

2006 Paper 11 Question 6

Operating System Foundations

(*a*) (*i*) Define a scheduling algorithm based on multilevel feedback queues. [6 marks]

(*ii*) Discuss the properties of your algorithm. Where appropriate, outline different design decisions that could have been taken. [6 marks]

(*b*) Discuss the implications of an operating system supporting

(*i*) single-threaded processes only;

(*ii*) multi-threaded processes. [8 marks]

2007 Paper 10 Question 12

Operating System Foundations

Operating systems are integrated closely with the hardware on which they run.

(a) Distinguish the hardware and software involved in interrupt-driven I/O. [4 marks]

(b) Describe *three* uses of the interrupt mechanism in addition to device I/O. [3 marks]

(c) Give examples of how the privilege state bit in a CPU's status register is used. [2 marks]

(d) (i) What could go wrong in a system that does not make use of a timing device in process scheduling? [1 mark]

(ii) What class of scheduling algorithms would be impossible to implement without a timing device? [2 marks]

(e) What is the main advantage of using half the virtual address space of a process for the operating system and half for applications? [1 mark]

(f) Explain memory-mapped I/O. [2 marks]

(g) In a paging system, would you expect every page of the operating system address space to be:

(i) mappable in the Translation Lookaside Buffer (TLB)? [2 marks]

(ii) capable of being cached? [1 mark]

Explain your answers.

(h) How do DMA (Direct Memory Access) devices operate? [2 marks]

2007 Paper 11 Question 12

Operating System Foundations

(*a*) Each object named in a filing system has associated with it a fixed-length metadata record (file control block). Describe *three* ways in which the disk blocks allocated to a file have been recorded, as part of its metadata, in various filing systems. State the good and bad properties of these approaches. [9 marks]

(*b*) (*i*) What is meant by a *hard link*? [2 marks]

(*ii*) Can a file or directory have no hard links? Explain. [1 mark]

(*c*) Filing systems make use of an in-memory cache of disk blocks. This is a shared data structure, accessed by device drivers and processes carrying out application-level requests for I/O.

(*i*) What are the advantages and disadvantages of not writing synchronously to disk? [2 marks]

(*ii*) Discuss why both mutual exclusion and condition synchronisation are needed for this data structure. [4 marks]

(*d*) One filing system names the blocks in its disk-block cache as filing-system-ID, block-number. In another, the blocks are offsets within a file. What are the implications of these approaches? [2 marks]

2008 Paper 10 Question 10

Operating System Foundations

(*a*) Assume a 32-bit architecture with hardware support for paging, in the form of a translation lookaside buffer (TLB), but no hardware support for segmentation. Assume that the TLB is shared rather than flushed on process switching and that the operating system designers are supporting "soft" segments.

(*i*) In addition to page number and page base, what fields would you expect to find in each TLB register?
How would each of these be used? [4 marks]

(*ii*) What fields would you expect to find in a process page table?
How would each of these fields be used? [6 marks]

(*b*) (*i*) Outline the function of a timing device. [2 marks]

(*ii*) Why are timers essential in multiprogramming operating systems? [2 marks]

(*iii*) Explain the operation of a multi-level feedback queue in process scheduling. [6 marks]

2008 Paper 11 Question 9

Operating System Foundations

(*a*) Why is I/O buffering used by operating systems? [2 marks]

(*b*) Explain why both mutual exclusion and condition synchronisation are needed for controlling access by concurrent processes to a shared buffer. [3 marks]

(*c*) Why is forbidding interrupts not a general solution to implementing concurrency control? [1 mark]

(*d*) Explain how a "read and clear" instruction can be used as a basis for building concurrency control. [2 marks]

(*e*) Define semaphores, including how they differ from a simple free/busy flag. [2 marks]

(*f*) The following pseudocode fragments represent access by a single producer and a single consumer to a shared, N-slot, cyclic buffer.

`items` is a semaphore initialised to 0
`spaces` is a semaphore initialised to the buffer size, N

producer repeats:	consumer repeats:
produce data	`WAIT (items)`
`WAIT (spaces)`	remove data from buffer
insert data in buffer	`SIGNAL (spaces)`
`SIGNAL (items)`	consume data

(*i*) Explain in detail how the semaphores are being used to enforce concurrency control. [6 marks]

(*ii*) Extend the code fragments for multiple producers and multiple consumers, explaining how your solution implements concurrency control. [4 marks]