UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 1998

MSc Degree in Computing Science
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Diploma of Membership of Imperial College

PAPER COMP II

ARCHITECTURE AND OPERATING SYSTEMS

Friday, May 15th 1998, 2.00 - 4.00

Answer THREE questions
Answer at least ONE question from Section A
Answer at least ONE question from Section B

For admin. only: paper contains 4 questions

Section A *(Use a separate answer book for this Section)*

1 a Registers, Main store and Magnetic disks form a memory hierarchy. Some of these levels allow data of different sizes to be stored and accessed. For each of these levels of storage outline the addressing mechanisms used to access the different sizes supported.

b Instructions can occupy anything from 2 to 6 bytes of main store depending on the type of instruction and the addressing modes used to specify the operands. The internal instruction register however is of fixed size (say 2 bytes).

i Provide a micro-code level description of the fetch phase of the basic machine cycle.

ii How does the Basic Machine cycle cope with instructions that are longer than the internal instruction register?

iii What would be the effect on your micro code description of increasing the width of the external system data bus? What would be the effect on other parts of the basic machine cycle?

c What is the minimum that the processor hardware must do to correctly respond to an external interrupt?

The three parts carry, respectively, 30%, 50%, 20% of the marks.

2 a The following hexadecimal digits represent values stored at successive byte locations in the memory of an 8086-based microcomputer.

```
 5678                                   567F
••••••••••••••••••••••••••••••••••••••••••••••
• A7 • 39 • F8 • FC • 62 • FC • 64 • FC •
••••••••••••••••••••••••••••••••••••••••••••••
```

i) The 16-bit value at location `567A` is a two's complement integer. What is its representation in base 10?

ii) The 16-bit value at location `567C` is an address. What is its representation in base 10?

b Describe the operation of the following 8086 instructions.

i) **PUSH** and **POP**

ii) **CALL** and **RET**

Include what operands each takes and any registers affected by its use. Give an example of when each might be used.

c The following C function returns the nth number in the Fibonacci series

```
int fib(int n)
{
    if (n == 1 || n == 2) /* n is 1 or n is 2 */
        return 1;
    else return fib(n-1) + fib(n-2);
}
```

Write an equivalent subroutine in 8086 assembler. Your solution should use a stack frame, preserve the contents of any registers used, and include **EQU**ate statements and informative comments.

The three parts carry, respectively, 30%, 30%, 40% of the marks

Turn Over...

Section B *(Use a separate answer book for this Section)*

3a Object code files include *relocation* information and *external symbol tables.* Explain why these are needed.

b Two object code files Main.o and Sub.o contain the object file information presented as follows:

Sub.o

address	opcode	operand	reloc?
0	3	2	Yes
1	6	1	No
2	4	0	Yes
3	8	0	No

sub	code	0
A	undefined	0

Main.o

address	opcode	operand	reloc?
0	0	0	No
1	5	2	Yes
2	5	0	Yes
3	9	0	No

main	code	1
A	data	0
sub	undefined	1

The format used is as described in lectures, including the "linked list" technique for undefined symbols. Note that you do *not* need to know what instructions the opcode values represent.

Write down, using the same format, the object file that results from linking these together, with Sub.o first in memory (starting at address 0) and Main.o following after. Include relocation information and an external symbol table with just one entry, for main.

c Suppose a program is to run using paged virtual memory. Is relocation needed (i) at link time, (ii) at load time, (iii) at run time? Explain your answers.

d You have a computer with 64Mbytes of main memory. Unfortunately you do not have any sophisticated paging, segmentation or swapping facilities on it. You do, however, have some hardware registers allowing you to load and protect several programs at the same time. Your operating system has left 50 Mbytes of memory available to run jobs/processes in it and it has the following queue of work to be performed:

Job Number	Memory required (Mbytes)	Time to complete (Seconds)
1	15	5
2	10	30
3	20	5
4	5	35
5	10	20
6	15	20
7	10	10

Using both the Best Fit and the Worst Fit algorithms, show how the operating system would process this queue of jobs. At each change of state, i.e. when jobs are removed and loaded into memory, draw the contents of the memory. How long does it take to finish all the jobs in the queue for each strategy? Assume that:

- all finished jobs are removed from memory before attempting to load the next one in the queue.
- the queue is a strict queue and jobs may not be selected from the middle of the queue.
- the time given in the table means the number of seconds that the job will occupy memory.

The four parts carry, respectively, 20%, 30%, 20%, 30% of the marks.

4a i) Explain the terms *critical region* and *mutual exclusion* in relation to concurrent processes.

ii) Give an example to show how, when two processes share variables, errors can arise if mutual exclusion is not enforced. Say what the critical regions are in your example.

iii) Describe how semaphore operations can be included in a program to enforce mutual exclusion. To what value must the semaphore be initialized?

b Suppose you are writing a procedure Fred to be called by numerous concurrent processes, and you wish to include code to count its calls and output a '*' character every hundred calls. Show how this can be achieved using semaphores. (You may use any standard high-level language, or pseudocode. The only way you can manipulate semaphores is by using the operations InitSema, P and V.) Remember to state how the semaphores must be initialized. Explain the role of the semaphore in your answer.

c Why in general should interrupt handlers not call the P operation? What might this do to the Simple Kernel?

d A multi-level queue implementation has three levels of process priority. Queue A holds processes that block within 100ms, Queue B holds processes that block within 500ms and Queue C holds processes that block within 2 seconds. Any process in Queue A will pre-empt any processes in Queues B or C. Similarly, any process in Queue B will pre-empt any process in Queue C.

Describe how an operating system might dynamically assign priorities for processes that have varying blocking times by placing them in different queues. Explain why processes with a lot of user interaction would get a good response.

The four parts carry, respectively, 40%, 30%, 10%, 20% of the marks.

End of paper