UNIVERSITY OF LONDON

IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 1997

BSc Honours Degree in Mathematics and Computer Science Part II
MSci Honours Degree in Mathematics and Computer Science Part II
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science

PAPER MC2.5

OPERATING SYSTEMS

Thursday, April 24th 1997, 10.00 - 11.30

Answer TWO questions

For admin. only: paper contains 3 questions

1a i) What is meant by *batch processing?*

ii) When batch processing was introduced, it required new features in machine architecture and instructions. Explain why they were needed and what they were.

b How is an *associative store* used to improve performance in address decoding for paged virtual memory?

What would be the consequences of having an associative store that is too small, or too large?

c The Simple Kernel described in lectures is for many processes running on a single processor. Its mechanisms for achieving mutual exclusion for memory access would fail in a multi-processor system. Explain why, and suggest what hardware features would be needed to get round this.

2a i) What are the datatypes of the *count* and the *queue* of a semaphore? What do they represent if the semaphore is being used to control some resource?

ii) Describe (using code, pseudocode or otherwise) the actions of the three semaphore procedures InitSema, P and V, as implemented in the Simple Kernel.

b Describe how the following restrictions could be enforced using semaphores.

i) A bridge carries traffic with a maximum of 10 vehicles on it at any time.

ii) A process A must wait until processes B, C and D have reached certain points in their execution.

c Show that for semaphores as implemented in a, it is always true (except within the semaphore operations themselves) that

i) count ≥ 0, and

ii) either count = 0 or queue = empty.

(Hint: Show that InitSema establishes these two conditions, and that if they hold before a call of P or V then they will still hold afterwards. Also, prove the two conditions together.)

3a What is a *context switch* in a multiprocessing environment? What are the *source* and *target* of a context switch?

Explain (without going into machine-dependent detail) how the Switch procedure in the Simple Kernel performs a context switch.

b An application program, written to run using the Simple Kernel presented in lectures, contains the following (and no other) calls of Create and Install:

```
Create(Stir, 500, high);
FOR i := 1 TO 5 DO
    Create(Cook, 500, normal);
    Create(Eat, 500, normal)
END;
Install(Mouse, 32);
Install(Cat, 40)
```

i) What processes will be in existence under the kernel and what code will they normally be executing? What distinguishes between them?

The following sequence of events takes place in these processes:

- A Cook process C is running, executing code in Cook.
- A Mouse interrupt occurs.
- Mouse makes a system call, the semaphore operation V, as its final command.
- A Cook process C' (different from C) is scheduled.

ii) At which point or points is there a context switch? For each one, identify the source and target processes, and describe approximately the values of the program counter and stack pointer that will be saved.

iii) Later on, C is rescheduled and there is a context switch back to it. Describe the sequence of return instructions between the end of that Switch and the resumption of execution of Cook.

iv) In the whole of the above sequence, between interruption of C and its resumption, at what points would interrupts get disabled or enabled?

The two parts carry, respectively, 20%, 80% of the marks.

End of paper