UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2001

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

PAPER M2

ARCHITECTURE AND OPERATING SYSTEMS

Wednesday 9 May 2001, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

1a The following are the data declarations for an 8086 assembler program

```
.data
hi      db 'hot$'           ; whew
lo      db 'cold$'          ; brrr
h       dw  100             ; boiling point
l       dw -273             ; absolute zero
```

List the contents of the corresponding byte locations in memory in hexadecimal. (ASCII 'a' = 97 decimal, '$' = 36 decimal)

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 i) The following C function returns the minimum of two integer parameters

```
int min2(int a, int b)
{
    if (a < b)
        return a;
    else return b;
}
```

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

ii) The following C function returns the minimum of three integer parameters

```
int min3(int a, int b, int c)
{
    return min2(a, min2(b, c));
}
```

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

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

2 This question concerns interrupts and interrupt handlers in a simple system, such as the NARC machine described in the lectures, or, if you prefer, the Simple Kernel.

a Explain briefly the sequence of events by which an external event leads to execution of an interrupt handler.

b Suppose two interrupts occur at exactly the same time:

i What determines which of the interrupts is serviced first?

ii When is the second interrupt serviced?

c Suppose we have a simple machine such as the NARC from the lecture course, and we set up a clock device to interrupt the processor once every second. We install the following interrupt handler to keep track of elapsed minutes and seconds (which are held in global integer variables):

```
void InterruptHandler() {
        saveProcessorState();

        seconds = (seconds + 1) % 60;
        if (seconds == 0)
          minutes = minutes + 1;

        restoreProcessorState();
        rti;    // restore PC and re-enable interrupts
}
```

(recall that "`n % m`" computes the remainder of `n/m`).

Now, consider an application program which needs to print out the time:

```
int main() {
        while (1) {
          do_some_work();
          cout << "Time: " << minutes << "mins\n";
          cout << "      " << seconds << "secs\n";
        }
}
```

i Explain what anomalous behaviour could occasionally occur, and why.

ii Write down a modified version of the `main` function above which avoids this problem. Explain briefly why your solution works.

iii Check that your solution works properly even if execution of `main` proceeds extremely slowly for some reason. Explain your reasoning, or explain any change you need to make.

(The six parts/subparts carry, respectively, 30%, 15%, 10%, 15%, 15% and 15% of the marks).

3 a The main components of a micro-programmed Control Unit are Control Store; Micro Program Counter; Micro Instruction Register; and the Sequencing Logic. Give a brief description of the function of each of these four components. You do not need to give an overall view of the control unit at this stage

b Compare and contrast your descriptions in part a) with a similar description of the function of the equivalent components in the Basic Machine Model which are Main Store; Program Counter (also known as the Instruction Pointer); Basic Machine Instruction Register; and the Control Unit (Be careful to describe the Control unit as a whole rather than repeat your descriptions of part a)

c By using as an example the simple 16-bit basic machine instruction `mov ax,bx` (which will move the contents of the 16 bit `bx` register into the 16 bit `ax` register) show how all the components described in parts a & b work together to fetch and execute the instruction. You should start with a high level description of the basic machine cycle and then expand on individual phases. Be careful not to repeat yourself, it is sufficient to refer to a single, more comprehensive description of a particular operation if that operation is not significantly different.

The three parts carry, respectively, 25%, 25%, 50% of the marks.

4a State whether each of the following statements is True or False. In each case, give a *brief* (one sentence) justification for your answer.

i) Two processes that wish to share data must use explicit interprocess communication mechanisms, but threads associate with the same process can access shared data without using explicit interthread communication mechanisms.

ii) Preemptive CPU scheduling policies incur a higher overhead than non-preemptive policies, but give better response time for interactive jobs.

iii) Introducing priority preemption prevents starvation in a static multi-level priority queue scheduling scheme.

iv) Under a FIFO page replacement algorithm, an increase in the capacity of the FIFO queue will never increase the number of page faults for a given stream of page accesses.

b With reference to the Simple Kernel, describe two situations in which it is necessary to perform a context switch.

c

i) Describe the purpose of a Process Address Register (PAR) in the context of a virtual memory system.

ii) Consider a paged virtual memory system that must support a maximum virtual address space of 4GB per process and up to 256MB of physical memory. What is the page size if a PAR is 36 bits long? You may assume that 1MB = 2^{20} bytes and that 1GB = 2^{30} bytes.

iii) Now write down a general expression for the page size s (in bytes) given a maximum virtual address space of 2^v bytes, a maximum physical memory size of 2^p bytes and a PAR size of b bits.

The three parts carry, respectively, 40%, 20% and 40% of the marks.