UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 1999

BEng Honours Degree in Computing Part I
MEng Honours Degrees in Computing Part I
BSc Honours Degree in Mathematics and Computer Science Part I
MSci Honours Degree in Mathematics and Computer Science Part I
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
Associateship of the City and Guilds of London Institute

PAPER 1.4 / MC 1.4

ARCHITECTURE I

Tuesday, April 27th 1999, 2.00 – 3.30

Answer THREE questions

For admin. only:
paper contains 4 questions

1a For a 16-bit group, work out the representation for the decimal integer –437 in

i) Sign & Magnitude

ii) One's Complement

iii) Two's Complement

iv) Excess 512

Give your answers to parts a(i) to a(iv) in both binary and hexadecimal.

b For an N-bit group (N>1), what range of signed integers is representable using

i) Sign & Magnitude

ii) Two's Complement

iii) Excess M

iv) Signed BCD (you can assume that N>=8 and that N is a multiple of 4)

c State the overflow rule for Two's Complement addition and the overflow rule for Two's Complement subtraction.

d Show that if `N` is chosen be to be equal to 2^{M-1}, where `M` is the number of bits in the representation, then an `Excess N` representation will be the same as a Two's Complement representation but with the sign-bit inverted. Hint – transform the summation formula ($\sum$) for `Excess N` numbers to the summation formula ($\sum$) for Two's Complement numbers.

e Suppose you need to extend the number of bits in the Two's Complement representation of a number from M bits to N bits. Describe how this operation can be performed?

The five parts carry, respectively, 20%, 20%, 20%, 30% and 10% of the marks.

2 For this question you are required to translate the following high-level language code into **commented** Pentium assembly language code:

```
type dataT = array 0 .. 9 of
                 record
                     name: array 0..7 of char
                     age : int
                 end

var data : dataT
```

```
. . .
```

```
printAdults (data)
```

```
. . .
```

```
procedure printAdults (var list : dataT)
     for k : 0 .. 9
          if (list [k].age >= 18)
               printName (list [k].name)
          end if
     end for
end printAdults
```

```
procedure printName (var name : array 0..7 of char)
```

For your solution you should assume:

- a flat 32-bit memory model,
- 32-bit `int`'s and 8-bit `char` 's,
- register EAX need not be preserved in procedure *printAdults*
- that the assembly language version of the procedure *printName* already exists.

Your Pentium assembly language solution should include:

i) a declaration for the global variable `data`

ii) a translation of the procedure call *printAdults* (list)

iii) a translation of the procedure *printAdults*.

iv) the stack frame for your procedure, clearly labelling each item on the stack with its offset from the frame pointer.

State any additional assumptions that you make.

The four parts (i), (ii), (iii), (iv) carry, respectively, 10%, 20%, 50% and 20% of the marks

Turn over ...

3a For IEEE single-precision, define denormal floating point numbers. Give an example when they might occur.

b Suppose that the IEEE defined a new 16-bit floating point format called Tiny Precision that is the same as the IEEE Single Precision format except that the Exponent is 7 bits and held as Bias 63 and the Significand is 8 bits.

	1 bit	7 bits	8 bits
Tiny Precision Format	Sign S	Exponent E	Significand F

For this format interpret the 16-bit hexadecimal value BEA0 as a Tiny Precision number and then convert the Tiny Precision number to decimal.

c Convert the decimal number 7.75 to Tiny Precision format and convert the result into hexadecimal.

d Perform a binary addition of the Tiny Precision numbers in parts b and c and convert the result into hexadecimal. Note—conversion of the Tiny Precision result back to decimal is NOT required

e Explain why floating point addition always adjusts the number with the smaller exponent.

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

4a For the Pentium architecture (32-bit protected mode) precisely describe how the linear address of an interrupt-handler is derived from its interrupt-vector number.

b Explain why the Pentium CPU pushes the EFLAGS register onto the stack prior to calling an interrupt-handler. Couldn't the saving and restoring of this register be left to the programmer?

c Using Programmed I/O write a loop that continuously reads characters from a keyboard and outputs them to a printer. Each read character should be output immediately. Where possible reading and printing should proceed in parallel. The I/O ports for the keyboard and printer are defined as follows:

	7	6	5	4	3	2	1	0	Address Offset
Keyboard Control Port					K				200200H
Keyboard Data Port									200201H
Printer Control Port		P							300300H
Printer Data Port									300301H

For the keyboard, setting the K bit to 1 will initiate a keyboard read request. Once a character has been read into the Keyboard Data Port, the keyboard I/O controller will set the K bit to 0 to indicate completion of the transfer.

For the printer, setting the P bit to 1 will initiate a printer write request. Once the character in the Printer Data Port has been printed, the printer I/O controller will set the P bit to 0 to indicate completion of the transfer.

i) Develop a high-level language version of your programmed I/O loop.

ii) Develop a Pentium assembly language version of your programmed I/O loop.

You can assume that no keyboard transfer is in progress at the start of your loop i.e. bit K=0.

State any additional assumptions that you make.

The four parts a, b, c(i) and c(ii) carry, respectively, 25%, 25%, 20% and 30% of the marks.

End of paper

