Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

M.Sc. in Mathematics & Computational Science

Fundamentals of High Performance Execution

Date: Monday 25th January 2010

Time: 09.45 – 11.45

Please answer TWO Questions from the FOUR questions provided

This is an OPEN book examination

The use of electronic calculators is permitted provided they are not programmable and do not store text
Please answer TWO Questions from the FOUR questions provided

1. a) Describe the general format and operation of the three main classes of instruction that are encountered in a typical computer with a register-based instruction set architecture (ISA).  

b) The following C program is compiled for a register-based ISA, with optimisations disabled. Identify any parts of the code fragment between the marked start point and end point that you would expect to lead to poor performance, explaining in each case why.  

NOTES: The function double random () returns a random value of type double. Arrays in C are laid out in memory in row-major order.

```c
#define n 100000

/* declare three large matrices */
double a[n][n]
double b[n][n]
double c[n][n]

/* declare a function add () */
double add (double x, double y)
    {return x + y;}

/* main program */
int main ()
    {
        int i, j;
        for (i = 0; i < n; i++)
            {for (j = 0; j < n; j++)
                {
                a [i][j] = random ()
                b [i][j] = random ();
            }
        }
        /* start point */
        for (i = 0; i < n ; i++ )
            {for (j = 0; j < n ; j++)
                {c[i][j] = add (a [j][i],b [j][i]);
                 }
            }
        /* end point */
        for (i = 0 ; i < n; i++)
            {for (j = 0 ; j < n ; j++)
                { printf ("%f ", c[i][j]);
                 }
            }
        /* end of program */
```
(Question 1 continues from the previous page)

c) Describe three optimisation techniques that you would expect to improve the performance of the code fragment described in part b). In each case, explain what the optimisation technique would do to the compiled code and how this would improve the performance. Quantify the expected improvement in performance in each case, making reasonable assumptions. (9 marks)

2. a) Describe the memory hierarchy associated with a modern microprocessor, from the registers in the processor to the swap space on the hard disk. Explain the characteristics and operation of each layer of the hierarchy, making clear how high performance can be obtained and who (or what) is responsible for obtaining it at each level. (12 marks)

b) A 1GHz microprocessor system, with a paged virtual memory, has the following memory hardware characteristics:

Two levels of cache:
L1: 32 kbytes, 1ns access time
L2: 4 Mbytes, 8ns access time

Main memory:
1 Gbyte, 60ns access time, 64 kbyte page size

Hard disk subsystem:
30 ms maximum head movement (inner to outer track) time
10 ms rotation period
100 Mbytes/sec data transfer rate

The expected local miss rates for a particular long-running application program are as follows:

L1 cache, 3 misses per 100 accesses
L2 cache, 6 misses per 100 accesses
Main memory, 1 miss per 4,000,000 accesses

The operating system takes an average of 10 microseconds to process a page swap before starting the disk transfer that reads in the requested page.

Calculate the average access time from the processor to the L1 cache. (8 marks)
3. a) Give an overview of the main purposes of an operating system for a typical desktop computer. (3 marks)

b) Among other things, the operating system is responsible for managing *processes*, that is, executable images of programs in execution. The intention is to allow multiple processes to be *active* at the same time, while ensuring that only one of the active processes is actually *running* on the processor at any one time. Explain how an operating system can achieve this when the user is interactively starting and possibly stopping many processes over time. (4 marks)

c) The operating system is also responsible for managing and maintaining a disk-based *filestore*, which may be shared among distinct user and/or application programs. Explain from a high level viewpoint how an operating system can achieve this. (3 marks)

d) At a lower level, the operating system is responsible for managing interactions between the computer and its input/output (I/O) devices.

i) Explain how a character typed on the keyboard can be detected by the operating system and passed to the active process for which it is intended. (3 marks)

ii) When I/O data rates become large (e.g. for transfers to/from the hard disk or the communications network), devices are managed using *block transfers* of data. Explain how this affects the actions of the operating system when a running process reads a single character from a *file* in the disk-based filestore. (5 marks)

iii) How does the operating system make the two actions, described in i) and ii) above, appear identical to the processes involved? (2 marks)
4. a) Consider the following very simple grammar that describes the construction of a string composed of the characters $a, b, c, d, e$.

   \[
   S \rightarrow aABe \\
   A \rightarrow Abc | b \\
   B \rightarrow d
   \]

   Using both top-down and bottom-up parsing, show (by deriving the parse tree) that the input string

   \[abccbcde\]

   is a valid string in the given grammar. (8 marks)

b) The following grammars and associated semantic rules describe arithmetic expressions involving identifiers, id, integer constants, num, and the operators $+, -, \times$:

   **Grammar 1**

<table>
<thead>
<tr>
<th>Production</th>
<th>Semantic Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>[expr \rightarrow expr + term]</td>
<td>[expr.t := expr.t \parallel term.t \mid ^{+}]</td>
</tr>
<tr>
<td>[expr \rightarrow expr - term]</td>
<td>[expr.t := expr.t \parallel term.t \mid ^{-}]</td>
</tr>
<tr>
<td>[expr \rightarrow term]</td>
<td>[expr.t := term.t]</td>
</tr>
<tr>
<td>[term \rightarrow term \times factor]</td>
<td>[term.t := term.t \parallel factor.t \mid \times]</td>
</tr>
<tr>
<td>[factor \rightarrow id]</td>
<td>[factor.t := \text{`id'}]</td>
</tr>
<tr>
<td>[factor \rightarrow num]</td>
<td>[factor.t := \text{`num'}]</td>
</tr>
</tbody>
</table>

   **Grammar 2**

<table>
<thead>
<tr>
<th>Production</th>
<th>Semantic Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>[expr \rightarrow term + expr]</td>
<td>[expr.t := term.t \parallel expr.t \mid ^{+}]</td>
</tr>
<tr>
<td>[expr \rightarrow term - expr]</td>
<td>[expr.t := term.t \parallel expr.t \mid ^{-}]</td>
</tr>
<tr>
<td>[expr \rightarrow term]</td>
<td>[expr.t := term.t]</td>
</tr>
<tr>
<td>[term \rightarrow term \times factor]</td>
<td>[term.t := term.t \parallel factor.t \mid \times]</td>
</tr>
<tr>
<td>[factor \rightarrow id]</td>
<td>[factor.t := \text{`id'}]</td>
</tr>
<tr>
<td>[factor \rightarrow num]</td>
<td>[factor.t := \text{`num'}]</td>
</tr>
</tbody>
</table>

   By constructing the annotated syntax trees, show that

   \[a \times b + c - 2 \times d - a\]

   where $a, b, c, d$ are identifiers, and 2 is an integer constant, is a valid arithmetic expression for each grammar; obtain the two postfix notations for the expression and comment on the difference. (12 marks)