COMPUTER SCIENCE TRIPOS Part II (General)
DIPLOMA IN COMPUTER SCIENCE

Tuesday 3 June 2008 1.30 to 4.30

PAPER 11 (PAPER 2 OF DIPLOMA IN COMPUTER SCIENCE)

*Answer **five** questions.*

*Submit the answers in five **separate** bundles, each with its own cover sheet. On each cover sheet, write the numbers of **all** attempted questions, and circle the number of the question attached.*

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

STATIONERY REQUIREMENTS
Script paper
Blue cover sheets
Tags

SPECIAL REQUIREMENTS
None

1 Digital Electronics

(*a*) Briefly explain the differences between *combinational* and *sequential* logic. [2 marks]

(*b*) With the aid of appropriate diagrams, briefly explain the operation of Moore and Mealy finite state machines and highlight their differences. [6 marks]

(*c*) The state sequence for a binary counter is as follows:

A	B	C	D
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0

The counter is to be implemented using four synchronously clocked D-type flip-flops.

(*i*) Draw a state table for the counter, showing the required D inputs. [4 marks]

(*ii*) Find expressions for the D inputs, making use of unused states if appropriate. [6 marks]

(*iii*) What problem could occur when the counter circuit is powered-up? Give *two* possible general methods for overcoming the problem. [2 marks]

2 Digital Electronics

(*a*) With the aid of relevant diagrams, show the effect on the output of a combinational logic circuit of a:

(*i*) static hazard;

(*ii*) dynamic hazard. [3 marks]

(*b*) Simplify the following expressions using Boolean algebra:

(*i*) $X = (A + \overline{B} + \overline{A} \,.\, B) \,.\, (A + \overline{B}) \,.\, \overline{A} \,.\, B$

(*ii*) $Y = (A + \overline{B} + \overline{A} \,.\, B) \,.\, \overline{C}$ [4 marks]

(*c*) Given:

$$F = A \,.\, B \,.\, \overline{C} \,.\, D + A \,.\, C + B \,.\, \overline{C} \,.\, \overline{D} + \overline{B} \,.\, C + \overline{A} \,.\, \overline{C} \,.\, \overline{D} + \overline{A} \,.\, \overline{B} \,.\, \overline{C} \,.\, D$$

(*i*) Show using a Karnaugh map that F can be simplified to

$$F_1 = A \,.\, B + \overline{A} \,.\, \overline{B} + A \,.\, C + B \,.\, \overline{C} \,.\, \overline{D}$$

[2 marks]

(*ii*) Show that there are a total of four possible expressions for F. [3 marks]

(*iii*) Show how F_1 can be implemented using NAND gates and draw the circuit diagram. Assume that complemented input variables are available. [2 marks]

(*iv*) Show how the static 1 hazard in F_1 can be eliminated using a Karnaugh-map-based approach. [2 marks]

(*v*) Now implement F_1 assuming that only 2-input NAND gates are available. [4 marks]

3 Compiler Construction

(*a*) Given the following program fragment in C or Java

```
static int a = 3;
static int f(int x, int y) { int z = a+x; ...; return ... }
```

explain how the four variables (a, x, y and z) are accessed from within f at the instruction level for an architecture such as MIPS, ARM or x86. Pay particular attention as to how and when storage is allocated for these variables, and which system components of a standard compile-link-and-execute model are involved in selecting the instruction and determining the run-time address calculation. Your answer should briefly explain what occurs when f makes a recursive call to itself. [8 marks]

(*b*) Suppose we extend the language to allow nested function definitions:

```
static int a = 3;
static int f(int x) {
  static int g(int y) { int z = a+x; ...; return ... }
  return g(7);
}
```

(*i*) Complete the definition of g in such a way that, during execution of a call to f, the difference between the addresses allocated to x and y varies. [2 marks]

(*ii*) Explain a possible run-time mechanism that allows all four variables to be accessed from inside g. [6 marks]

(*c*) Suppose function-valued variables are added to the language, e.g.:

```
funvar double v(double);
v = (...) ? sin : cos;
```

and a pointer to code or to data is represented as a 32-bit value. How many bits might naturally be used to hold v? Justify your answer, emphasising the way in which it might differ according to whether the language is as given in part (*a*) or part (*b*) above. [4 marks]

4 Introduction to Functional Programming

(*a*) Specify the types of the following SML functions.

(*i*) `fun B x y z = x (y z)` [2 marks]

(*ii*) `fun C x y z = x z y` [2 marks]

(*iii*) `fun W x y = x y y` [2 marks]

(*b*) Let `datatype` α `tree = leaf | node of` α `*` α `tree *` α `tree`
be the datatype of binary trees.

Write an SML function `DF:` α `tree` $\rightarrow$ `int tree` that given a tree outputs a tree of the same shape, but with the values at the nodes replaced by their number in depth-first order.

For example, the depth-first numbering of the tree

```
node( "a" ,
      node( "b" , node("c",leaf,leaf) , node("c",leaf,leaf) ) ,
      node( "b" , leaf , node("c",leaf,leaf) ) )
```

is the tree

```
node( 1 ,
      node( 2 , node(3,leaf,leaf) , node(4,leaf,leaf) ) ,
      node( 5 , leaf , node(6,leaf,leaf) ) )
```

[6 marks]

(*c*) Let `datatype` α `inftree = node of` α `* (unit` $\rightarrow$ α `inftree list)`
be the datatype of finite and infinite non-empty finitely-branching trees.

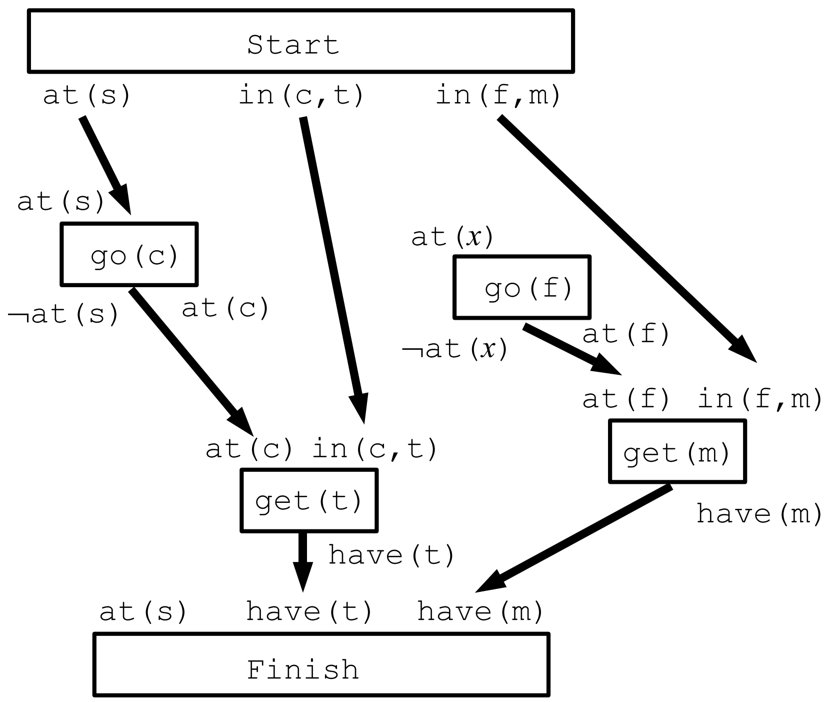
(*i*) The computation tree of a function $f : \alpha \rightarrow \alpha$ `list` starting at $s : \alpha$ is the possibly infinite tree with root s in which every node n has children $n_1, \ldots, n_k$ whenever $f(n) = [n_1, \ldots, n_k]$.

Write an SML function `CT: (`$\alpha \rightarrow \alpha$ `list)` $\rightarrow \alpha \rightarrow \alpha$ `inftree` such that `CT` f s is the computation tree of f starting at s. [4 marks]

(*ii*) Define the datatype α `seq` of finite and infinite lists of type α and write an SML function `BF:` α `inftree` $\rightarrow$ α `seq` that lists the nodes of a tree according to a breadth-first traversal. [4 marks]

5 Artificial Intelligence I

A brilliant student has finished his exams and is making a well-deserved cup of tea. He is confused, however, and is trying to use the *partial order planning* algorithm to solve part of the problem. Using the abbreviations `f` for "fridge", `c` for "cupboard", `s` for "sink", `m` for "milk" and `t` for "tea", his start state is $\{\texttt{at}(\texttt{s}), \texttt{in}(\texttt{c}, \texttt{t}), \texttt{in}(\texttt{f}, \texttt{m})\}$. Using x and y to denote variables, he has two actions. The first action is $\texttt{get}(y)$ having preconditions $\texttt{at}(x)$ and $\texttt{in}(x, y)$, and effect $\texttt{have}(y)$. The second action is $\texttt{go}(y)$ having precondition $\texttt{at}(x)$ and effects $\neg\texttt{at}(x)$ and $\texttt{at}(y)$. His goal is $\{\texttt{at}(\texttt{s}), \texttt{have}(\texttt{t}), \texttt{have}(\texttt{m})\}$. So far he has made the following attempt at finding a plan:



In this diagram, arrows denote causal links.

(*a*) Can the $\texttt{at}(x)$ precondition on $\texttt{go}(\texttt{f})$ be achieved by adding an ordering constraint and causal link from `Start` to $\texttt{go}(\texttt{f})$, and perhaps one or more further ordering constraints, in such a way that the plan remains valid? Explain your answer. [4 marks]

(*b*) Describe a method, different from any suggested in part (*a*), by which the $\texttt{at}(x)$ precondition on $\texttt{go}(\texttt{f})$ can be achieved in such a way that the plan remains valid. [8 marks]

(*c*) Describe a way in which the plan can be completed after making the addition you have described in part (*b*). [8 marks]

6 Introduction to Security

(*a*) The following files are shown by an `ls -l` command on a typical Unix system:

```
-r-xr-sr-x  1 charlie acct    70483 2008-01-04 22:53 accounting
-r--rw----  1 alice   acct   139008 2008-05-13 14:53 accounts
-rwxr-xr-x  1 system  system 230482 1997-04-27 22:53 editor
-rw-r--r--  1 alice   users    7072 2008-06-01 22:53 cv.txt
-r--r-----  1 bob     gurus   19341 2008-06-03 13:29 exam
-r--r-----  1 alice   gurus    6316 2008-06-03 16:25 solutions
```

Unix users `alice` and `bob` are both members of only the group `users`, while `charlie` is a member of only the group `gurus`. Application `editor` allows users to read and write files of arbitrary name and change their permissions, whereas application `accounting` only allows users to append data records to the file `accounts`. Draw up an access control matrix with subjects {`alice`, `bob`, `charlie`} and objects {`accounts`, `cv.txt`, `exam`, `solutions`} that shows for each combination of subject and object whether the subject will, in principle, be able to read (R), (over)write (W), or at least append records (A) to the respective object. [9 marks]

(*b*) A C program uses the line

```
buf = (char *) malloc((n+7) >> 3);
```

in order to allocate an $\lceil \frac{\mathtt{n}}{8} \rceil$-bytes long memory buffer, large enough to receive `n` bits of data, where `n` is an unsigned integer type.

(*i*) How could this line represent a security vulnerability? [2 marks]

(*ii*) Modify the expression that forms the argument of the `malloc()` call to avoid this vulnerability without changing its normal behaviour. [3 marks]

(*c*) Name *three* types of covert channels that could be used to circumvent a mandatory access control mechanism in an operating system that labels files with confidentiality levels and give a brief example for each. [6 marks]

7 Data Structures and Algorithms

Quicksort can be described as a recursive in-place sorting algorithm that performs a `partition()` operation on the given array and then invokes itself twice on two distinct subranges of the array.

(*a*) Describe the purpose, I/O parameters and effect of the `partition()` procedure and explain what the *pivot* is. Pseudocode is not required. [3 marks]

(*b*) Give pseudocode for the `quicksort()` procedure that would call the `partition()` procedure you described in (*a*). Prove that your `quicksort()` will always terminate. [3 marks]

(*c*) Analyse the worst-case behaviour of Quicksort and discuss possible ways of improving it. [4 marks]

(*d*) Some researchers have suggested choosing the pivot from a randomly chosen location in the input array. Discuss the advantages and disadvantages of such a solution. How does it affect the worst-case and average-case behaviour? [5 marks]

(*e*) Define the median of an array of n numbers. Then explain clearly how to implement a `median()` procedure that would use the `partition()` procedure you described in part (*a*). (You may, if you wish, illustrate your answer with pseudocode.) Briefly analyse the complexity of this procedure. [5 marks]

8 Algorithms II

(*a*) Briefly describe the Dijkstra algorithm for finding shortest paths from a single source and explain why it cannot be used on graphs with negative edge weights. [Pseudocode is not required.] [4 marks]

(*b*) Describe and explain in detail the Johnson algorithm that finds all-pairs shortest-paths by repeatedly applying Dijkstra to each vertex, even in graphs with negative edge weights. [Pseudocode is not required but all phases of the algorithm must be clearly explained.] [7 marks]

(*c*) Some people wonder why it would not be simpler to reweight edges by adding a sufficiently large constant K to each edge weight so as to make all the weights positive. Prove that this cannot work. [2 marks]

(*d*) In Johnson's algorithm, why do we introduce a new vertex s? Could we not use, instead of a new vertex, one of the vertices of the original graph? Either prove that we can or provide a counterexample. [7 marks]

9 Operating System Foundations

(*a*) Why is I/O buffering used by operating systems? [2 marks]

(*b*) Explain why both mutual exclusion and condition synchronisation are needed for controlling access by concurrent processes to a shared buffer. [3 marks]

(*c*) Why is forbidding interrupts not a general solution to implementing concurrency control? [1 mark]

(*d*) Explain how a "read and clear" instruction can be used as a basis for building concurrency control. [2 marks]

(*e*) Define semaphores, including how they differ from a simple free/busy flag. [2 marks]

(*f*) The following pseudocode fragments represent access by a single producer and a single consumer to a shared, N-slot, cyclic buffer.

`items` is a semaphore initialised to 0
`spaces` is a semaphore initialised to the buffer size, N

```
producer repeats:              consumer repeats:
   produce data                   WAIT (items)
   WAIT (spaces)                  remove data from buffer
   insert data in buffer          SIGNAL (spaces)
   SIGNAL (items)                 consume data
```

(*i*) Explain in detail how the semaphores are being used to enforce concurrency control. [6 marks]

(*ii*) Extend the code fragments for multiple producers and multiple consumers, explaining how your solution implements concurrency control. [4 marks]

10 Business Studies

(*a*) Describe *five* criteria that an investor might use to evaluate a business. Which is the most important, and why? [5 marks]

(*b*) Distinguish between *marketing* and *selling*. [5 marks]

(*c*) Distinguish between *quantitative* and *qualitative* market research. [5 marks]

(*d*) A PC manufacturer obtains the following results from test marketing PC systems:

System including	Price ex VAT	Number sold
19 inch monitor and software bundle	£299	1000
19 inch monitor, software bundle and a printer	£399	750
22 inch monitor and software bundle	£499	400
22 inch monitor, printer but no additional software	£599	500

If the test market area represents 1% of the target population, what price point and how many sales should be expected for a system with a 22 inch monitor but with neither software nor printer? [5 marks]

END OF PAPER