PAPER CODE NO.	EXAMINER : Michael Fisher
COMP513	DEPARTMENT : Computer Science Tel. No. 4-6701

THE UNIVERSITY
of LIVERPOOL

MAY 2004 EXAMINATIONS

Bachelor of Science : Year 3
Master of Science : Year 1

Formal Methods

TIME ALLOWED : $2\frac{1}{2}$ hours

INSTRUCTIONS TO CANDIDATES

Answer **four** questions only.

If you attempt to answer more questions than the required number of questions (in any section), the marks awarded for the excess questions will be discarded (starting with your lowest mark).

1. This question concerns the basic structures used within Z specifications.

   (a) Consider the following sets: $A == \{1, 9\}$, $B == \{1, 3\}$, and $C == \{1, 5, 9\}$. What is the value of $\#((A \times B) \cap (A \times C))$? **[5]**

   (b) In Z, what is the difference between a *total* function and a *surjection*? **[4]**

   (c) Represent the sequence $\langle red, blue, red, green, yellow \rangle$ as a set of maplets. **[4]**

   (d) Represent the bag $\llbracket red, blue, red, green, yellow \rrbracket$ as a set of maplets. **[4]**

   (e) Is

   $$\mathrm{dom}(f \cap g) = (\mathrm{dom} f) \cap (\mathrm{dom}\, g)$$

   true for all sets f and g? If so, explain why; if not, give examples of f and g that show it is not the case. **[8]**

2. A library is developing a Z specification of its lending system. A systems analyst familiar with Z wrote the following as the first part of the specification (assuming that *BOOKS* and *USERS* are predefined types).

   Library
   $active : \mathbb{P} USERS$
   $defaulters : \mathbb{P} USERS$
   $lending : USERS \pfun \mathbb{P} BOOKS$
   ———
   ...

   As you can see, the analyst did not complete the specification. You are required to complete the state space schema, by adding the following invariants to it:

   (a) only active users can be lending books; **[4]**

   (b) defaulters are also active users; **[4]**

   (c) no user can be lending more than 4 books at any time. **[4]**

   (d) Now, add a schema describing the '`Borrow`' operation. This allows a user to borrow a book, as long as

   - the user is active,
   - the user is not a defaulter,
   - the user is not already borrowing 4 books, and
   - no other user is already lending the requested book.

   After the operation, the user should be recorded as lending the specified book. **[13]**

3. [About fundamentals of Temporal Logic.]

(a) We wish to say that

> "in the next moment in time, *a* will be true and, at some time after that, *b* will be true."

How might we represent this in temporal logic? **[3]**

(b) What type of structure is typically used to provide a model for propositional, discrete, linear temporal logic, with finite past, and why? **[4]**

(c) Is the following valid, i.e. true in all possible models?

$$\Box\Diamond\varphi \;\Rightarrow\; \Diamond\bigcirc\varphi$$

Justify your answer, either informally or by appealing to the formal semantics. **[6]**

(d) The formula $\Box(\varphi \Rightarrow \Diamond\psi)$ says that the formula $\varphi \Rightarrow \Diamond\psi$ is always true.

i. If we know that $\mathtt{start} \Rightarrow \varphi$, then how many times will ψ be *forced* to occur?

ii. If we know that $\Diamond\varphi$, then how many times will ψ be *forced* to occur?

iii. If we know that $\Box\varphi$ is true, how many times will ψ be *forced* to occur?

[9]

(e) We wish to say that

> "there is a moment in the future where either *c* is always true, or *d* is true in the next moment in time."

How might we represent this in temporal logic? **[3]**

4. Below is a temporal specification for a simple message-passing system consisting of two components, A and B.

$$Spec_A\colon\ \Box\left[\begin{array}{lrcl} & \mathbf{start} & \Rightarrow & p\\ \wedge & p & \Rightarrow & \bigcirc q\\ \wedge & q & \Rightarrow & \bigcirc p\\ \wedge & q & \Rightarrow & \bigcirc send\_msg \end{array}\right] \qquad Spec_B\colon\ \Box\left[\begin{array}{lrcl} & rcv\_msg & \Rightarrow & \bigcirc g\\ \wedge & f & \Rightarrow & \bigcirc g\\ \wedge & g & \Rightarrow & \bigcirc f \end{array}\right]$$

(a) What is the behaviour of $Spec_A$, i.e. how often is $send\_msg$ made true? **[7]**

(b) In

$$Spec_A \wedge Spec_B \wedge \Box[send\_msg \Rightarrow rcv\_msg]$$

what is the last formula meant to specify? **[3]**

(c) If we wish to specify that a message send will be followed, at some time in the future, by a message receipt, what formula should we modify in the specification given in (b) and what should it be changed to? **[5]**

(d) What is a *safety* property, and what general form of temporal formulae characterise such properties? **[5]**

(e) What is a *liveness* property, and what general form of temporal formulae characterise such properties? **[5]**

5. What is model checking, and why is it useful?

Write an essay on this, bringing in as many elements as appropriate, including

- the aim of model checking,
- the uses of model checking,
- the mechanisms for model checking,
- the problems with model checking and
- potential solutions to some of these problems.

[25]

6. Consider the following Promela code.

```
proctype G (chan out)
{
        int num;
        num = 0;
G1:     num = ((num*2)+1);
        out!num;
        if
        :: (num < 20) -> goto G1;
        :: (num >= 20) -> goto G2;
        fi
G2:     num = num -1;
        out!num;
        if
        :: (num > 0) -> goto G2;
        fi }

proctype S (chan in, outB, outL)
{
        int num;
S1:     in?num;
        if
        :: (num <= 10) -> outL!num; goto S1;
        :: (num > 10) -> outB!num; goto S1;
        fi }

proctype B (chan in)
{
        int num;
B1:     in?num;
        assert(num > XXX);
        goto B1 }

proctype L (chan in)
{
        int num;
L1:     in?num;
        assert(num <= XXX);
        goto L1 }

init {  chan g2s = [0] of { int };
        chan s2b = [0] of { int };
        chan s2l = [0] of { int };
        atomic { run G(g2s); run S(g2s, s2b, s2l);
                 run B(s2b); run L(s2l) }
     }
```

(a) What is the communication structure between the four processes `G`, `S`, `B`, and `L`, i.e. which channels link the processes and in which direction does information flow between them? **[6]**

(b) What is the sequence of numbers generated and sent down the `out` channel in process `G`? **[6]**

(c) In order for the assertions in processes `B` and `L` to succeed, which single integer should be used in place of '`XXX`' in both processes? Explain your answer. **[6]**

(d) If we wanted to verify that all the numbers seen by process `B` were bigger than any of the numbers seen by process `L`, how might we do this? What additional aspects might we add to the program and what temporal formula would we check? **[7]**