Name: ....
Student Number ....

Software Engineering 2F04

DAY CLASS — Dr. Mark Lawford
DURATION OF EXAMINATION: 3 Hours
McMaster University Final Examination — December 2002

THIS EXAMINATION PAPER INCLUDES 6 PAGES AND 4 QUESTIONS. YOU ARE RESPONSIBLE FOR ENSURING THAT YOUR COPY OF THE PAPER IS COMPLETE. BRING ANY DISCREPANCY TO THE ATTENTION OF YOUR INVIGILATOR.

Special Instructions: The use of calculators, notes, and text books is not permitted during this exam. Answer all questions in the provided answer booklets. Fill in your name and student number and sign each booklet you use. This paper must be returned with your answers.

NOTE: Tables of proof rules appear at the back of the exam.

1. Predicate Logic I (20 marks)

In the following, assume that $f : A \to A$, $g : A \to A$ and $h : A \to A$ are all unary functions.

a) (5 marks) Write down a predicate logic formula ϕ such that $FV(\phi) = \{y\}$ (y is the only free variable in ϕ) with the property that ϕ is true if and only if y is in $Im(h)$.

b) (5 marks) Write down a predicate logic formula that is true if and only if $Im(h) \subseteq Im(g)$. (This should be a sentence - no free variables!).

c) (10 marks) Formally prove the following:

$$\vdash \forall x(h(x) = g(f(x))) \to \forall y(\exists x(h(x) = y) \to \exists x(g(x) = y))$$

2. Predicate Logic II (25 Marks Total)

a) Consider the following sequence of formulas:

$$\begin{aligned}\Gamma \;:=\; & \exists x \forall y \neg Q(x,y),\\ & \exists x(Q(x,x) \land \forall y(Q(y,y) \to x = y)),\\ & \exists x \exists y \exists z(Q(x,y) \land Q(x,z) \land y \neq z),\\ & \forall x(f(x) = x \to \exists y Q(y,x))\end{aligned}$$

The intended interpretation for these formulas is a FSM model $\mathcal{M}$ where:

$$\begin{aligned} A \;&:=\; \{a,b,c,d,e\} \text{ is the universe, the finite set of states}\\ f^{\mathcal{M}}(x) \;&:=\; a \text{ is the reset function that returns the FSM to the initial state } a,\end{aligned}$$

and the interpretation of Q, denoted $Q^{\mathcal{M}}$, is the FSM transition relation yet to be determined. In your summer job as a software developer you are given Γ as a formal specification and told to create two different versions of the transition relation $Q^{\mathcal{M}}$.

Continued on page 2

i) (3 marks) First, find a $Q^{\mathcal{M}}$ such that for this interpretation of Q we have $\mathcal{M} \not\models \Gamma$.

ii) (7 marks) Next, find a different $Q^{\mathcal{M}}$ such that for this interpretation of Q we have $\mathcal{M} \models \Gamma$.

b) Consider the new sequence of formulas:

$$\begin{aligned}\Gamma' \;:=\; & \exists x \forall y \neg Q(x,y),\\ & \exists x (Q(x,x) \wedge \forall y (Q(y,y) \to x = y)),\\ & \exists x \exists y \exists z (Q(x,y) \wedge Q(x,z) \wedge y \neq z),\\ & \forall x (f(x) = x \to \exists y Q(y,x)),\\ & \forall x \forall y \forall z (Q(x,y) \wedge Q(x,z) \to y = z)\end{aligned}$$

c) (10 marks) Determine if Γ' is inconsistent. Note: Informal arguments are not acceptable. Only a formal proof or a model will be accepted.

d) (5 marks) Based upon your answer to the previous question, does there exists a graph that provides a model $\mathcal{M}$ such that $\mathcal{M} \models \Gamma'$? Draw the graph or explain why no such graph exists.

3. Mathematical Induction (20 Total Marks)

a) (10 marks) Let $i = \sqrt{-1}$ (i.e. $i^2 = -1$). In this question you will use mathematical induction to prove DeMoivre's Theorem. Prove that for all $n \geq 1$,

$$(\cos x + i \sin x)^n = \cos(nx) + i \sin(nx)$$

Hint: The easy way to do this is use the fact that $e^{i\omega} = \cos\omega + i \sin\omega$, or alternatively you can remember your Trig identities $\cos(A+B) = \cos A \cos B - \sin A \sin B$ and $\sin(A+B) = \sin A \cos B + \cos A \sin B$.

b) (10 marks) Let us consider why rule MI is a valid rule of inference. In this problem you will show how the antecedents of rule MI can be used to write down a proof of $\phi[2|n]$ and $\phi[a|n]$ in general. Let

$$\Gamma := \phi[0/n], \forall m(\phi[m/n] \to \phi[m+1/n]), 0+1=1, 1+1=2$$

.

i) Formally show that $\Gamma \vdash \phi[2/n]$.

ii) What lines would you add to the proof in (i) to show $\Gamma, 2+1=3 \vdash \phi[3/n]$?

iii) Given some $a \in \mathbb{N}$, how many lines would it take to show:
$\Gamma, 2+1=3, 3+1=4, 4+1=5, \ldots, (a-1)+1=a \vdash \phi[a/n]$?

4. Tabular Software Specifications & Dependent Types (35 Total Marks)

A digital control system makes use of multiple sensors. The sensor values of all the analog inputs to the system are stored in a 64 element array of reals called `AI`. In the array of sensor values there are 4 Power, 2 Pressure, and 3 Temperature sensor values at the following locations:

Sensor Type	Number of Sensors (N)	Array Index of Sensors Values
Power	4	6, 7, 8, 9
Pressure	2	62, 63
Temperature	3	1, 2, 3

Continued on page 3

A general sensor interface program, `GetSensor(Sensor, N)`, has been written that takes as input the type of sensor and the number of sensors of that type and returns an array of N real values containing the sensors values for specified type of sensor. E.g. `GetSensor(Pressure, 2)` returns a two element array of type real containing `AI(62)` and `AI(63)` in locations 1 and 2 respectively.

Theory `GetSensors` contains the function `GetSensor` which models the access program described above. Note the use of dependent types in the range to model the variable size of the returned array. Here `LAMBDA(i:subrange(1,N)):AI(i+5)` is the PVS notation for the λ (function) abstraction operator you have seen in SFWR ENG 2A04. Thus this expression gives the function mapping $\texttt{i} \in \{1, 2, \ldots, \texttt{N}\}$ to `AI(i+5)`.

Trying to prove the Type Correctness Conditions (TCCs) for this theory results in several unproven TCCs for the `GetSensor` function. Figure 2 is one of the unproven TTCs `GetSensor_TCC2`. Beneath it is the unprovable sequent that results when attempting to prove it.

a) (5 marks) Write down the characteristic formula for the unprovable sequent in Figure 2.

b) (5 marks) Find a counter example that makes the characteristic formula you obtained in (a) false (i.e. find values of `i!1` and `N!1` that make the formula false) and confirm that it makes `GetSensor_TCC2` false.

c) (5 marks) The unprovable TCCs indicate that the use of `GetSensor` could result in undefined terms. What is the unprovable TCC `GetSensor_TCC2` indicating about potential problems regarding the use of the access program `GetSensor`?

d) (10 marks) In the theory `GetSensor`, a horizontal condition table is used to define the function $\texttt{f:Sensor\_t} \rightarrow \mathbb{N}$ that maps a sensor type to the number of sensors (e.g. `f(Pressure)=2`). Write down the completeness and disjointness TCCs that PVS would generate for this table to prove that it defines a total function.

e) (5 marks) You will now give a new version of the function `GetSensor` called `GetSensor2`. Unlike most sequels (e.g Harry Potter), this one will be better (e.g. Star Wars Episode 2). Eliminate the possibility of undefined terms occurring in `GetSensor2` through the use of dependent types in the arguments and in the body of the `COND` statement (Hint: Use `f`).

Using this definition of `GetSensor` when performing software verification would force you to prove that every call to the access program is made with the correct value of `N`.

f) (5 marks) A different implementation of `GetSensor` in a newer programming language makes use of the languages ability to return arrays of different sizes without having to specify the size of the return array in the function's inputs. Define a function `GetSensor3`(Sensor) that now only takes the sensor type as input and returns an array containing the appropriate sensor values. Take car to insure that this new version of `GetSensor` does not have any undefined terms. Explain why `GetSensor3` as implemented in the newer programming language might be "safer".

"Can we use deMorgan's? " - almost every 2F04 student this year

Continued on page 4

```
GetSensors : THEORY
  BEGIN

  AI: ARRAY[subrange(1,64)->real]

  Sensor_t: TYPE = {Power, Pressure, Temperature}

  GetSensor(Sensor:Sensor_t,N:int):ARRAY[subrange(1,N)->real]=
      COND
        Sensor=Power -> LAMBDA(i:subrange(1,N)):AI(i+5),
        Sensor=Pressure -> LAMBDA(i:subrange(1,N)):AI(i+61),
        Sensor=Temperature -> LAMBDA(i:subrange(1,N)):AI(i)
      ENDCOND

  f(Sensor:Sensor_t):nat = TABLE
        %------------------------%
        |Sensor=Power       | 4||
        %------------------------%
        |Sensor=Pressure    | 2||
        %------------------------%
        |Sensor=Temperature | 3||
        %------------------------%
        ENDTABLE

  END GetSensors
```

Figure 1: PVS code for Question 4

```
% Subtype TCC generated (at line 12, column 54) for  i + 61
    % expected type  subrange(1, 64)
  % unfinished
GetSensor_TCC2: OBLIGATION
  FORALL (N: int, Sensor: Sensor_t):
    Sensor = Pressure IMPLIES
     (FORALL (i: subrange(1, N)): 1 <= i + 61 AND i + 61 <= 64);


GetSensor_TCC2 :

[-1]  1 <= i!1
[-2]  i!1 <= N!1
  |-------
[1]   61 + i!1 <= 64

Rule?
```

Figure 2: Unproven TCC for Question 4

Continued on page 5

Propositional Logic Proof Rules

	introduction	elimination
$\wedge$	$\dfrac{\phi \quad \psi}{\phi \wedge \psi}\wedge i$	$\dfrac{\phi \wedge \psi}{\phi}\wedge e_1 \quad \dfrac{\phi \wedge \psi}{\psi}\wedge e_2$
$\vee$	$\dfrac{\phi}{\phi \vee \psi}\vee i_1 \quad \dfrac{\psi}{\phi \vee \psi}\vee i_2$	$\dfrac{\phi \vee \psi \quad \boxed{\begin{array}{c}\phi\\ \vdots\\ \chi\end{array}} \quad \boxed{\begin{array}{c}\psi\\ \vdots\\ \chi\end{array}}}{\chi}\vee e$
$\rightarrow$	$\dfrac{\phi \quad \phi \rightarrow \psi}{\psi}\rightarrow e$	$\dfrac{\boxed{\begin{array}{c}\phi\\ \vdots\\ \psi\end{array}}}{\phi \rightarrow \psi}\rightarrow i$
$\leftrightarrow$	$\dfrac{\phi \rightarrow \psi \quad \psi \rightarrow \phi}{\phi \leftrightarrow \psi}\leftrightarrow i$	$\dfrac{\phi \leftrightarrow \psi}{\phi \rightarrow \psi}\leftrightarrow e_1 \quad \dfrac{\phi \leftrightarrow \psi}{\psi \rightarrow \phi}\leftrightarrow e_2$
$\neg$	$\dfrac{\boxed{\begin{array}{c}\phi\\ \vdots\\ \bot\end{array}}}{\neg\phi}\neg i$	$\dfrac{\phi \quad \neg\phi}{\bot}\neg e$
$\neg\neg$	$\dfrac{\phi}{\neg\neg\phi}\neg\neg i$	$\dfrac{\neg\neg\phi}{\phi}\neg\neg e$
$\bot$	see $\neg e$	$\dfrac{\bot}{\phi}\bot e$

Continued on page 6

Additional Propositional Logic Proof Rules

$$\frac{\phi \to \psi}{\neg\phi \lor \psi} \to 2\lor \qquad \frac{\neg\phi \lor \psi}{\phi \to \psi} \lor 2 \to$$

$$\frac{\phi \to \psi \quad \neg\psi}{\neg\phi}\text{MT} \qquad \frac{\boxed{\begin{array}{c}\neg\phi \\ \vdots \\ \bot\end{array}}}{\phi}\ \text{RAA}$$

$$\frac{}{\phi \lor \neg\phi}\text{LEM}$$

Additional Predicate Logic Proof Rules

	introduction	elimination
$\forall$	$\dfrac{\boxed{\begin{array}{l}x_0 \\ \quad\vdots \\ \quad\phi[x_0/x]\end{array}}}{\forall x\phi}\ \forall i$	$\dfrac{\forall x\phi}{\phi[t/x]}\forall e$
$\exists$	$\dfrac{\phi[t/x]}{\exists x\phi}\exists i$	$\dfrac{\exists x\phi \quad \boxed{\begin{array}{ll}x_0 & \phi[x_0/x] \\ & \vdots \\ & \chi\end{array}}}{\chi}\ \exists e$
$=$	$\dfrac{}{t = t} = i$	$\dfrac{t_1 = t_2 \quad \phi[t_1/x]}{\phi[t_2/x]}\ = e$

The End