PRIFYSGOL CYMRU; UNIVERSITY OF WALES

DEGREE EXAMINATIONS JANUARY 2002

SWANSEA

Computer Science

CS 213 System Specification

Attempt 2 questions out of 3

Time allowed: 2 hours

Students are permitted to use the dictionaries provided by the University through the invigilators

CS_213
SYSTEM SPECIFICATION

(Attempt 2 questions out of 3)

Question 1

Consider the following informal description of a simple microprocessor.

- 64K 16-bit memory;
- Sixteen 16-bit user registers, R0 to R15;
- A program counter PC;
- 1-bit condition register C.

All instructions are 16 bits long, and consequently will fit in a single memory word. The first four bits of an instruction represent the opcode, and the remaining twelve bits represent three registers (four bits for each register). (In the case of the GE instruction, described below, only two of these registers will be used.)

The operations allowed are specified below in a Pascal-like notation, where R, Ra, Rb, D, Da, Db and L represent any of the registers R0 to R15, and Mem is the memory.

```
ADD D <- Ra, Rb     D := Ra+Rb; PC : = PC + 1.

NOR D <- Ra, Rb     D := not(Ra or Rb); PC := PC + 1.

BNC Da,Db,L         if C=1 then begin
                           L := PC+1 ; PC := Da + Db
                    end else PC := PC + 1.

GE Ra Rb            if Ra >= Rb then C := 1 else C := 0 ; PC := PC + 1.

LD D <- Ra, Rb      D := Mem[Ra + Rb] ; PC := PC + 1.

ST R -> Da, Db      Mem[Da + Db] := R ; PC := PC + 1.
```

(a) Formally define the *state* of the microprocessor. Your answer should include a clear, formal, description of the *types* of each component of the state. **[5 marks]**

(b) Formally specify the microprocessor. You should include in your specification all the sub-functions you use. However, it is not necessary to define basic arithmetic and logical operations, or functions for writing to memory and registers. **[15 marks]**

(c) Extend your formal specification to include a new instruction *swap*

SWP Ra Rb

The new instruction should swap the contents of registers Ra and Rb. The program counter should be incremented by 1. **[5 marks]**

Question 2

The following Maude code represents the *shift register* example from the notes: the full Maude version, with comments, was available on the web site. Note that we are using the built-in sort MACHINE-INT to represent time.

```
fmod STREAM is
    protecting MACHINE-INT
    sort A .
    sorts AStr BoolStr

    op u : -> A .

    op _[_] : AStr MachineInt -> A .
    op _[_] : BoolStr MachineInt -> Bool .

endfm

fmod SR is
    protecting STREAM .

    op SR : BoolStr AStr -> AStr .
    op delta : MachineInt BoolStr -> MachineInt .
    op NoTrue : BoolStr MachineInt -> Bool .

    var b : BoolStr .
    var x : AStr .
    var t : MachineInt .

    ceq SR(b,x)[t] = u if (t == 0) or NoTrue(b,t - 1) .
    ceq SR(b,x)[t] = x[delta(t,b)] if (t > 0) and
                                   not(NoTrue(b,t)) .

    eq delta(0,b) = 0 .
    ceq delta(t,b) = t - 1 if b[t - 1] and (t > 0) .
    ceq delta(t,b) = delta(t - 1, b) if not(b[t - 1])
                                     and (t > 0) .

    eq NoTrue(b,0) = not(b[0]) .
    ceq NoTrue(b,t) = false if b[t] .
    ceq NoTrue(b,t) = NoTrue(b, t - 1) if not(b[t]) .
endfm
```

(a) Explain the module STREAM: what is the purpose of the constant `u`? what do the two operators `_[_]` do? Why are there no defining equations? **[10 marks]**

(b) Explain the module SR. In particular, what do each of the operators `SR`, `delta` and `NoTrue` do? **[10 marks]**

(c) If we wished to run SR on sample data, what would we need to do? **[5 marks]**

Question 3

The following is a (partial) Maude representation of a stack of integers.

```
fmod STACK is
     protecting MachineInt .
     sort StackInt .

     op Empty : -> StackInt .
     op ErrorVal : -> MachineInt .
     op push : StackInt MachineInt -> StackInt .
     op top : StackInt -> MachineInt .
     op pop : StackInt -> StackInt .

     var x : MachineInt .
     var a : StackInt .

     eq top(Empty) = ErrorVal .
     eq pop(Empty) = Empty .
endfm
```

(a) The set of equations in `STACK` is not complete. Write down *two* additional equations, that define the *normal* behaviour of `top`, `pop` and `push`. In addition, in `STACK` no error is reported when the empty stack is popped. Modify the definition of `STACK` so an error is reported if an attempt is made to pop the empty stack. **[10 Marks]**

(b) Write a set of Z schemas to represent a stack of integers. (When applied to the empty stack, your pop operation can either return an error or the empty stack: the choice is left to you.) **[10 Marks]**

(c) Compare and contrast the Maude and Z representations. What are the key differences between them? Which do you prefer, and why? **[5 Marks]**