TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI

VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS – 2019

TRIMESTER 1

SWEN 326

SAFETY-CRITICAL SYSTEMS

Time Allowed: TWO HOURS

CLOSED BOOK

Permitted materials: No calculators permitted.
Non-electronic Foreign language to English dictionaries are allowed.

Instructions: Answer all questions

Question	Topic	Marks
1.	Risk, Hazards and Failure	30
2.	Testing	30
3.	Static Analysis	30
4.	Design Validation	30
	Total	120

1. Risk, Hazards and Failure **(30 marks)**

(a) Consider the following description of a system for controlling a *water boiler*.

"The water boiler is operated by a *software controller*. If the *sensor* reports that the water temperature is too low, the heater should be *turned on*. Likewise if the temperature is too hot, the heater should be *turned off*. Under no circumstance should the water temperature be allowed to reach boiling point."

i. **(2 marks)** Following the terminology of IEC61508, identify the *Equipment Under Control* for the water boiler system.

ii. **(2 marks)** Identify an important *hazard* for the water boiler system.

iii. **(4 marks)** Briefly, discuss how the *risk* of the above hazard occurring might be estimated.

iv. **(2 marks)** Briefly, discuss how the above hazard is *mitigated* in the system.

(Question 1 continued on next page)

(Question 1 continued)

v. **(2 marks)** When calculating risk, one will often consider only a *single component failure*. Briefly, discuss what such an analysis might conclude for the water boiler system.

vi. **(2 marks)** Briefly, outline one simple approach for mitigating against a single component failure.

(b) Software bugs pose considerable risk to safety-critical systems.

i. **(4 marks)** Briefly, discuss the following statement:

"Catching errors at runtime may be too late."

ii. **(4 marks)** Briefly, discuss the following statement:

"Catching errors at compile time reduces overall risk."

(Question 1 continued on next page)

(Question 1 continued)

(c) The *"Power-of-Ten"* rules provide a simple set of coding guidelines for developing safety-critical software.

i. **(4 marks)** Rule 1 prohibits the use of *"direct or indirect recursion"*. Briefly, discuss the motivation behind this rule.

ii. **(4 marks)** Rule 2 requires that *"all loops must have a fixed upper-bound"*. Briefly, discuss the motivation behind this rule.

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

2. Testing **(30 marks)**

(a) Consider the following Java class which compiles without error:

```java
public static final int RED = 0;
public static final int GREEN = 1;
public static final int BLUE = 2;

public static int[] countRGB(int[] items) {
   int[] cs = new int[3]; // counts
   //
   for(int i=0;i!=items.length;++i) {
      switch(items[i]) {
      case RED:
         cs[RED] = cs[RED] + 1;
         break;
      case GREEN:
         cs[GREEN] = cs[GREEN] + 1;
         break;
      case BLUE:
         cs[BLUE] = cs[BLUE] + 1;
         break;
      default:
         // Do nothing
      }
   }
   return cs;
}
```

i. **(8 marks)** Draw the *control-flow graph* for the `countRGB(int[])` method.

(Question 2 continued on next page)

(Question 2 continued)

ii. **(3 marks)** Give test inputs which achieve 100% *branch coverage* of `countRGB()`.

iii. **(4 marks)** Briefly, state the *prime path coverage* criterion.

iv. **(5 marks)** Identify five *prime paths* for `countRGB()`:

(1)

(2)

(3)

(4)

(5)

(Question 2 continued on next page)

(Question 2 continued)

(b) Consider testing the following method using the Modified Condition / Decision Coverage (MC/DC) criterion.

```
public static int max3(int x, int y, int z) {
   if(x <= z && y <= z) {
      return z;
   } else if(x <= y) {
      return y;
   } else {
      return x;
   }
}
```

i. **(3 marks)** MC/DC requires all *decisions* and *conditions* take every possible outcome. Briefly, discuss what this means using the above example to illustrate.

ii. **(3 marks)** Give test inputs which achieve 100% for the MC/DC criterion.

iii. **(4 marks)** Fuzz testing is a simple form of *random testing*. Briefly, discuss the challenges faced in fuzz testing the `max3` method.

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

3. Static Analysis **(30 marks)**

An important form of static analysis for embedded systems is *Control-Flow Analysis*.

```
0x0000:  rjmp 7
0x0001:  rjmp -1
0x0002:  cp r24, r22
0x0003:  cpc r25, r23
0x0004:  brge 1
0x0005:  movw r16, r12
0x0006:  ret
0x0007:  rjmp -1
0x0008:  push r28
0x0009:  push r29
0x000A:  movw r24, r8
0x000B:  rcall -10
0x000C:  mov r22, r28
0x000D:  pop r29
0x000E:  pop r28
0x000F:  rjmp -1
```

(a) **(5 marks)** Using the above to illustrate, briefly discuss what a Control-Flow Analysis does.

(b) **(3 marks)** Identify the two *unreachable* instructions a Control-Flow Analysis would uncover for the above.

(Question 3 continued on next page)

(Question 3 continued)

(c) **(5 marks)** A Control-Flow Analysis is said to be *conservative*. Using an example to illustrate, briefly discuss what this means.

(d) Indirect jumps pose a significant challenge for Control-Flow Analysis.

i. **(3 marks)** Briefly, discuss what the problem is with indirect jumps.

ii. **(4 marks)** Jump tables are often implemented with indirect jumps. Briefly, discuss what a jump table is and why they are relatively easy to reason about.

(Question 3 continued on next page)

(Question 3 continued)

(e) Control-Flow Analysis can be used to determine the *worst-case stack usage* of an embedded program.

i. **(4 marks)** Briefly, discuss what this means.

ii. **(3 marks)** Briefly, discuss why loops and recursive methods present a challenge for determining the worst-case stack usage.

The following function uses a *stack allocated array*:

```
void rotate(int data[]) {
  int tmp[16];
  for(int i=0;i<16;++i) {
     tmp[i] = data[i];
  }
  for(int i=0;i<4;++i) {
   for(int j=0;j<4;++j) {
     data[(i*4)+j] = tmp[(j*4)+i];
} } }
```

iii. **(3 marks)** Briefly, discuss why stack allocated arrays present another challenge for determining the worst-case stack usage.

Student ID: ....

SPARE PAGE FOR EXTRA ANSWERS

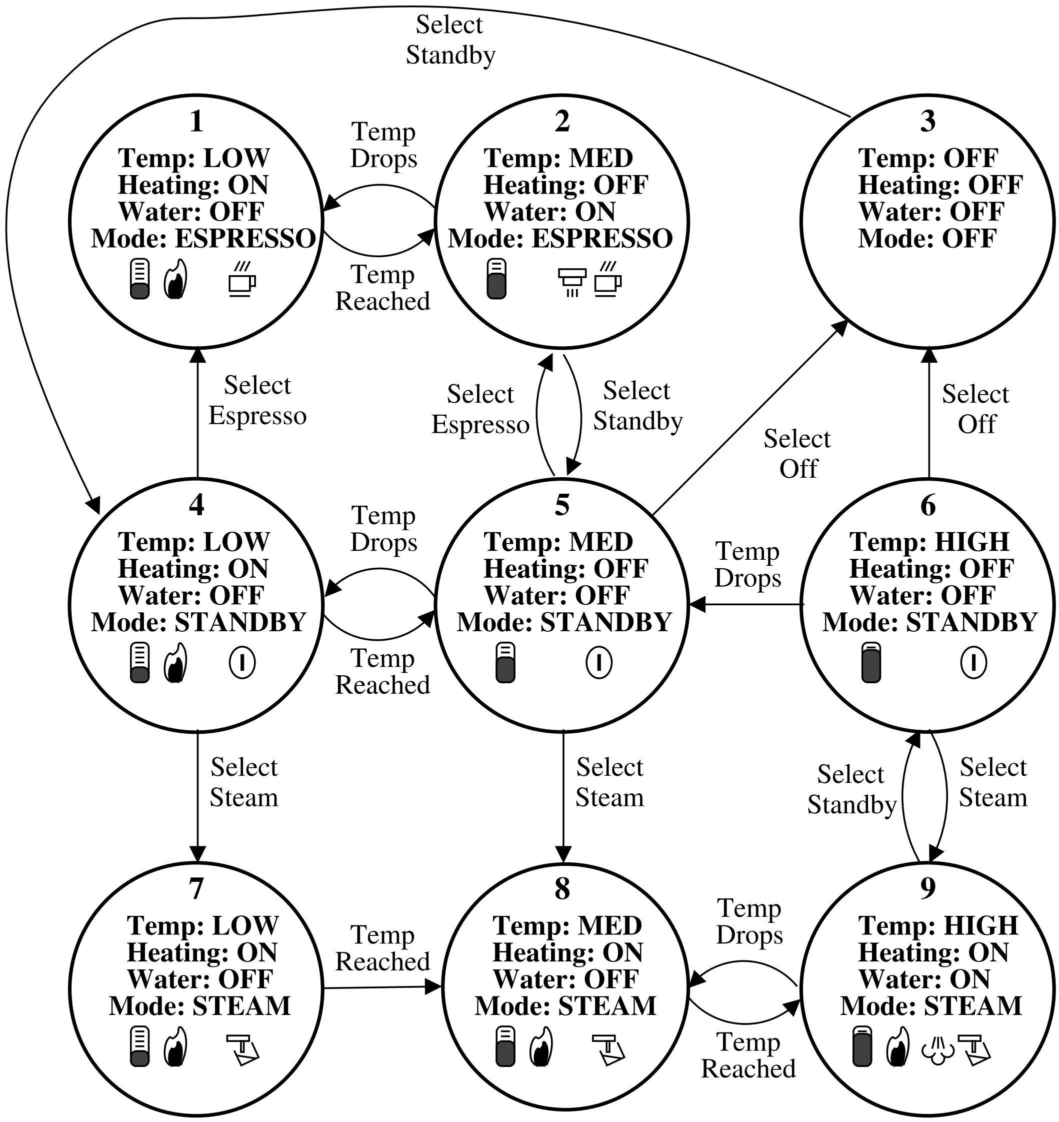
Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

4. Design Validation **(30 marks)**

Consider the following description of a simple *coffee machine*:

> "The Espresso Machine makes coffee by pushing hot water through ground coffee beans. A steam function is provided for steaming milk. A *control dial* is used to operate the machine. It has four positions: *Standby*, *Espresso*, *Steam* and *Off*. In *Standby mode*, the water is heated, and a *heat sensor* turns the heating off when the temperature is reached, and on again when it drops. In *Espresso mode* the machine passes hot water out into the ground coffee, whilst in *Steam mode* it shoots very hot steam out into the milk."

A *state machine diagram* for the coffee machine has been provided:



(Question 4 continued on next page)

(Question 4 continued)

(a) For each of the following statements, indicate whether you think it is a true or false statement based on the state machine diagram.

i. **(2 marks)** The machine continuously heats the water when “Steam” is selected.

ii. **(2 marks)** The water achieves the highest temperature when “Espresso” is selected.

iii. **(2 marks)** When “Espresso” is selected, the machine always pushes water out immediately.

iv. **(2 marks)** After steaming milk, the machine must cool down before it can make espresso again.

v. **(2 marks)** When the machine overheats, it automatically shuts down.

(b) Provide a suitable *execution trace* for the following scenarios. Your execution trace may start from whichever state you chose.

i. **(2 marks)**
“John selected steam, but the machine was cold and it took a long time before he got steam.”

ii. **(2 marks)**
“Jane turned the machine on in the morning. When she selected Espresso later on, water came out immediately.”

(Question 4 continued on next page)

(Question 4 continued)

(c) Consider the following (Whiley) model of the espresso machine state:

```
type AbstractState is {bool heating, bool water, int mode}
where mode >= 0 && mode <= 3

type State is (AbstractState s)
where (s.mode == OFF) ==> !(s.water || s.heating)
where (s.mode == STANDBY) ==> !s.water
where (s.mode == ESPRESSO) ==> (s.heating != s.water)
where (s.mode == STEAM) ==> s.heating

function tempChange(State s1, int sensor) -> (State s2)
// Event doesn't change operating mode
ensures (s1.mode == s2.mode):
   //
   AbstractState s = s1
   //
   if s1.mode == STANDBY:
      s.heating = (sensor < MED)
   else if s1.mode == ESPRESSO:
      s.heating = (sensor < MED)
      s.water = (sensor >= MED) && (sensor < HIGH)
   else if s1.mode == STEAM:
      s.water = (sensor >= HIGH)
      s.heating = true
   //
   return s

function selectStandby(State s1) -> (State s2)
// Resulting mode must be standby
ensures s2.mode == STANDBY:
   //
   State s = {mode: STANDBY, heating: false, water: false}
   //
   if s1.mode == OFF:
      s.heating = true
   else if s1.mode == ESPRESSO:
      s.heating = s1.heating
   //
   return s
```

i. **(2 marks)** Give a valid instance of `State` which corresponds to state five from the diagram on page 14.

(Question 4 continued on next page)

(Question 4 continued)

ii. **(2 marks)** Give a valid instance of `AbstractState` which does not correspond to *any* state from the diagram on page 14.

iii. **(4 marks)** Complete the following function to model the "Select Espresso" transition from the diagram on page 14.

```
function selectEspresso(State s1) -> (State s2)
requires s1.mode == STANDBY:
ensures s2.mode == ESPRESSO:
```

iv. **(4 marks)** Briefly, discuss why attempting to verify `tempChange()` produces the following error:

```
error:25: type invariant may not be satisfied
    return s
           ^
```

v. **(4 marks)** Currently, `selectStandby()` does not handle `s1.mode==STEAM`. Briefly, discuss why this is difficult to do.

* * * * * * * * * * * * * * *

Student ID: ....

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.