Spec (Warm-up Assignment)

Start working on this assignment after the Week 1 lectures. *You will need some Week 2 material to finish it*.

This assignment is worth 10% and due before **Monday June 20th, 6pm** local time Sydney.

Late submissions are accepted up to five days after the deadline, but at a penalty: 5% off your total mark per day.

The purpose of this assignment is to ensure that every student in this course is familiar with the tools we will be using.

This assignment has to be done individually.

The task involving Spin is supposed to be relatively easy. Finding the assertion network however can be hard. This component serves to (a) re-inforce understanding of the Owicki/Gries method and (b) provide a modest challenge.

Task

Consider the following algorithm presented in Ben-Ari's notation.

Algorithm Y	
bit b[2] ← {0, 0}	
p	**q**
loop forever	loop forever
p1: non-critical section	q1: non-critical section
p2: b[0] ← 1	q2: b[1] ← 1
p3: while b[1] = 1	q3: while b[0] = 1
p4: b[0] ← 1	q4: b[1] ← 0
p5: await (b[0] = 1)	q5: await (b[0] = 0)
p6: b[0] ← 1	q6: b[1] ← 1
p7: critical section	q7: critical section
p8: b[0] ← 0	q8: b[1] ← 0

1. *(40 marks)* Use Spin to check whether Algorithm Y is a solution to the critical section problem. Address all four desiderata from the lectures (mutual exclusion, eventual entry, absence of deadlock, absence of unnecessary delay).
2. *(40 marks)* Encode Algorithm Y as a parallel composition of two transition diagrams. Define an assertion network Q such that the assertions at the locations

representing the critical sections express mutual exclusion. Prove that Q is inductive. (It is ok to focus on the processes after the initialisation of b. It is not ok to make the assertions at the entry locations unreasonably strong.)

3. *(20 marks)* Identify any superfluous statements in the algorithm. That is, can any statements be replaced by `skip` without changing the behaviour of Algorithm Y? Justify your answers, preferrably using your transition diagram and assertion network.

Prepare a report that explains your findings. Make it concise and convincing.

Deliverables

algY.pml
: faithful implementations of Algorithm Y in Promela. Feel free to assume that this Promela header file is present in the same directory as your .pml file, if you want to use the same critical section boilerplate as in the lectures.

algY.pdf
: A PDF of your report. Diagrams may be hand-drawn (but must be readable). Prose must be typeset, preferably with LaTeX (although not mandatory). List your student ID near the top of the document. The document should describe your efforts, incorporate the previous deliverables, quote some output of Spin as evidence if that is helpful, and contain the Owicki/Gries-style proof. Make sure your Spin results are reproducible by specifying which (if any) non-default options you used for checking each property.

Testing

Submissions will be tested on CSE servers, where Spin's command line interface is available. A compatible version of the ispin GUI is accessible for vlab users with this command:

```
% ~cs3151/bin/ispin
```

...and can be used remotely over SSH as follows:

```
% ssh -Y z<my_digits>@login.cse.unsw.edu.au ~cs3151/bin/ispin
```

You may also wish to install Spin locally. Instructions here: http://spinroot.com

Submission Instructions

The `give` command to be run is:

```
% give cs3151 assn0 algY.pml algY.pdf
```

You may also use the online `give` interface to submit. Beware of rather tight submission size limits.