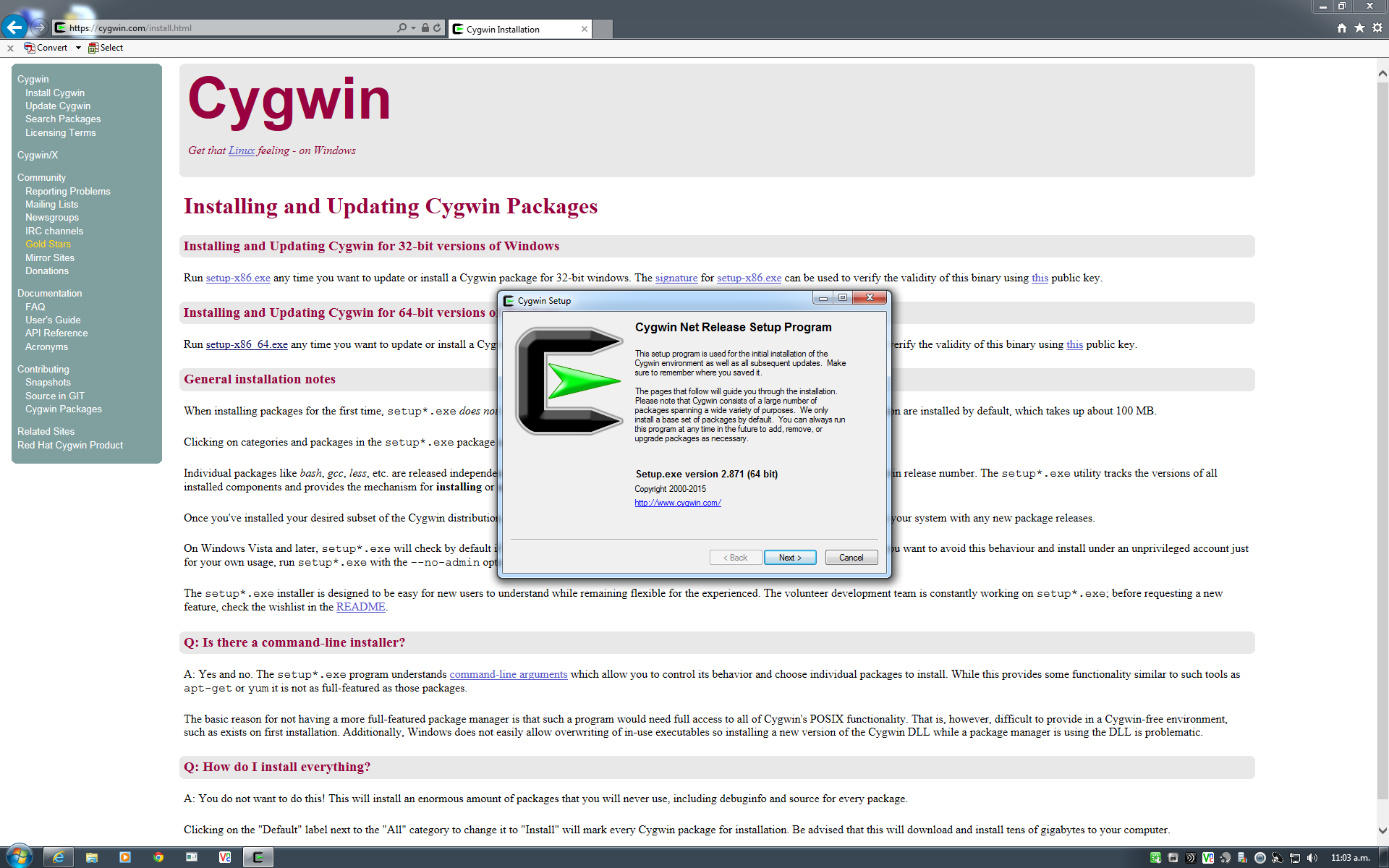
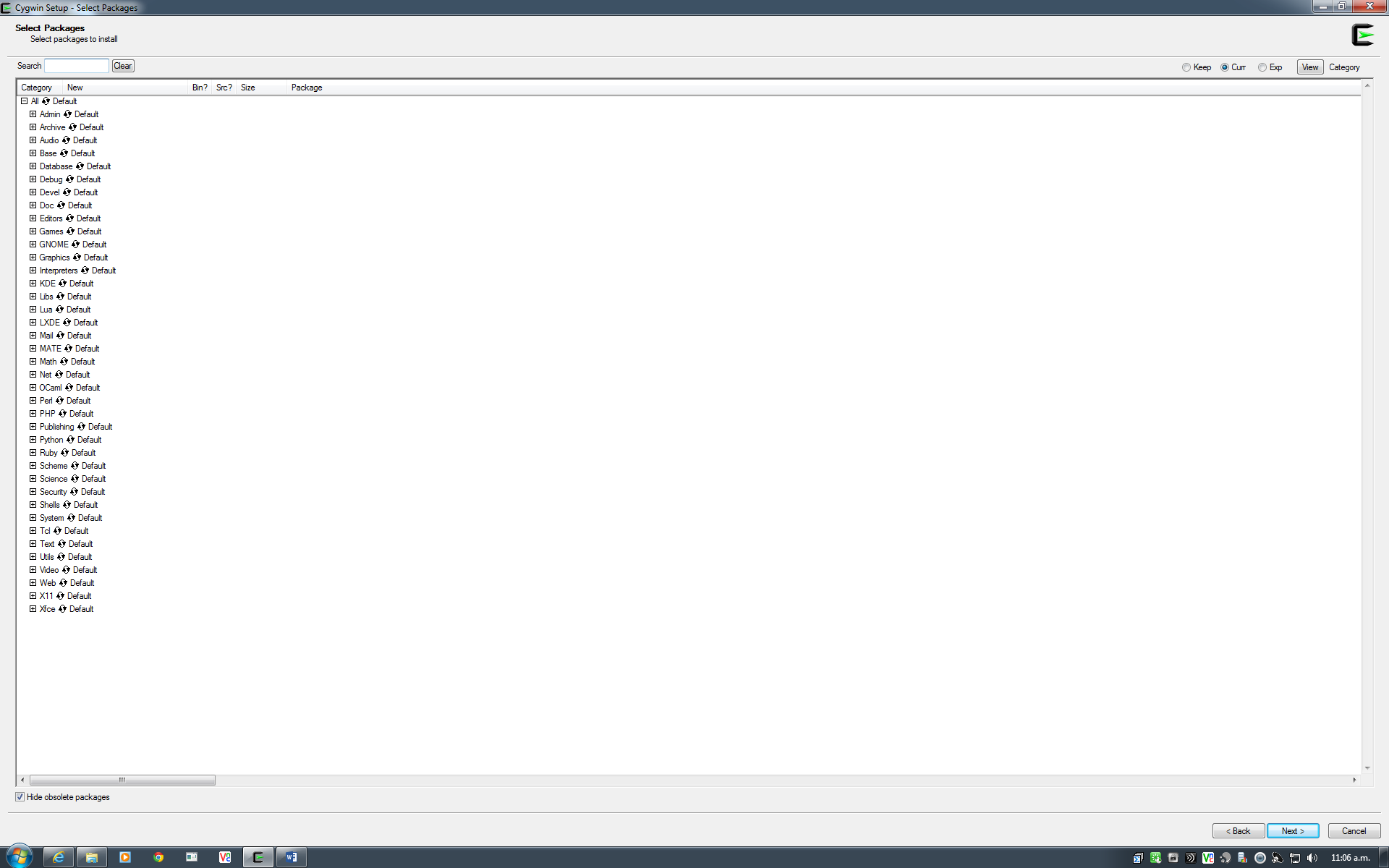
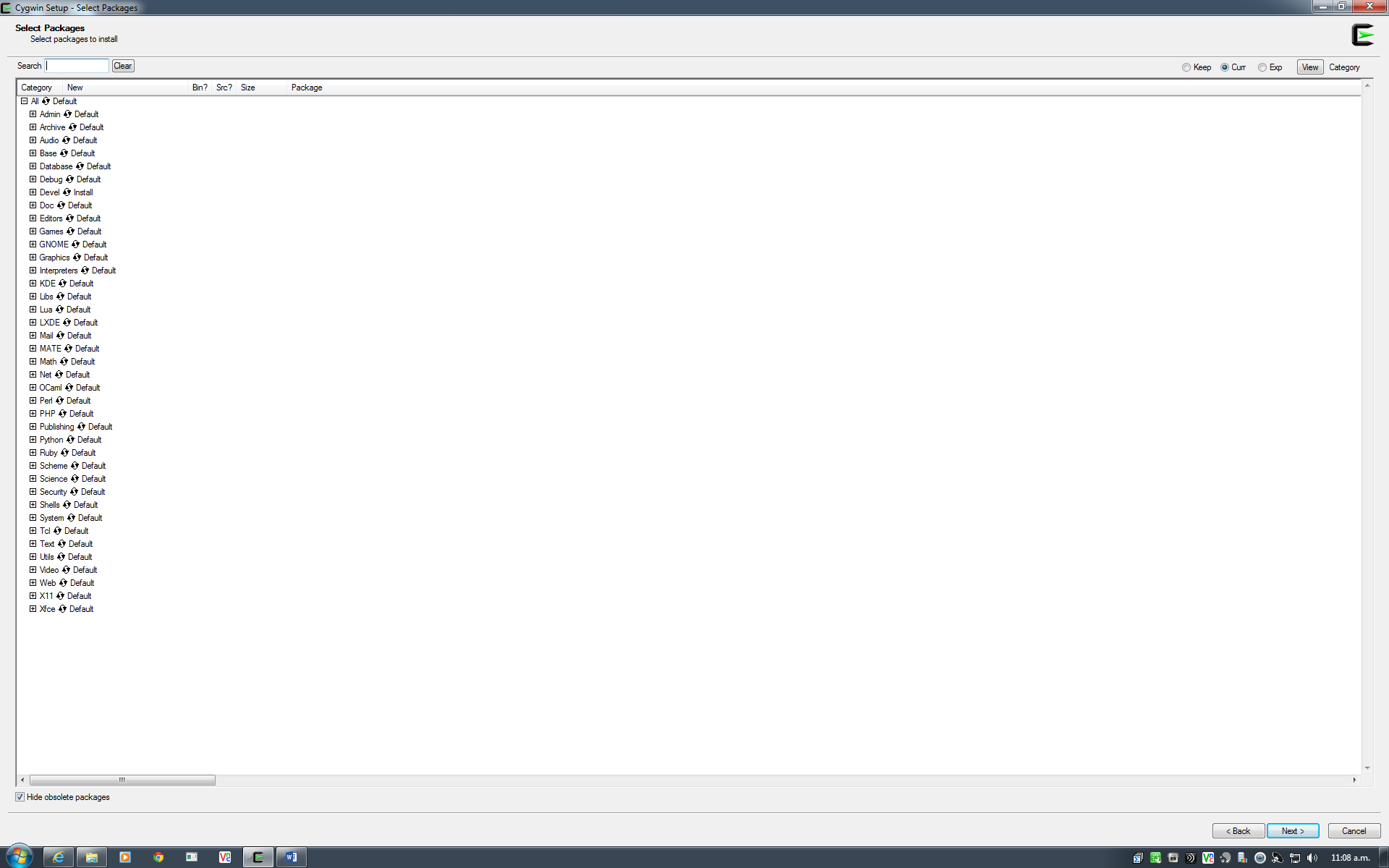
**COMPSYS-705**

**Spin Lab Exercise**

***Part I : Cygwin installation (Working on your own PC)***

1. Go to <https://www.cygwin.com/>, download the  [setup-x86.exe](https://www.cygwin.com/setup-x86.exe) (32-bit installation) or [setup-x86\_64.exe](https://www.cygwin.com/setup-x86_64.exe) (64-bit installation)
2. Launch the installer. Be sure to specify a temp folder to hold the local package directory. 
3. When you reach the screen that allows you to select which packages to install, please toggle the options from Default to Install for the packages called **Devel**. By default, *Cygwin* will NOT install it, however gcc package is necessary for *Spin*.

1. Continue with the installation until Setup is complete.
2. You will run *Spin* inside the *Cygwin* environment, please refer to Part II.

***Part II: Spin installation (Working in MDLS labs)***

1. Download the *Spin* binary from Canvas (might be pre-installed on the lab machines).
2. Copy the *Spin* binary to C:\cygwin64\usr\local\bin.
3. Open the *Cygwin* bash shell – can be found by searching for cygwin in the Start Search.
4. In *Cygwin*, type: $ spin -V # make sure *Spin* is there, you will see:

Spin Version 6.4.3 -- 16 December 2014

spin: error, no filename specified

1. cd to the directory containing the Promela file that you would like to run/verify with *Spin*. Recommend: place your model file \*. pml in the C:\cygwin64\home\username\.
2. Now, you are set to go with *Spin.*

***Part III: Spin Exercise***

1. Download the mutex\_ltl.pml and mutex\_assert.pml file from Canvas
2. Open the *Cygwin* bash shell, and you will use command lines to run *Spin*;

Given a model system specified in *Promela*, *Spin* can either perform random simulations of the system's execution or it can generate a C program that performs a fast exhaustive verification of the system state space.

1. *Simulation*

Syntax : spin [options] file

1. Try a simulation run:

$ spin -p -l mutex\_assert.pml # print out local vars at every step

1. You can see all options that *Spin* supports by executing:

$ spin --

1. Please refer to <http://spinroot.com/spin/Man/Spin.html> for more options
2. *Verification*
3. Generate source codes for a model-specific verifier

Use spin –a to generate a verifier (model checker) for the specification. The output is written into a set of C files, named pan.[ cbhmt ] that can be compiled to produce an executable verifier.

$ spin -a mutex\_assert.pml

$ ls # list all the files under the current directory, you will see pan.\* files.

1. Compile and get an executable verifier

$ gcc -o pan pan.c

The executable program pan can now be executed to perform the verification.

1. Run the verifier and get the verification result.

$ ./pan

The output will be:

hint: this search is more efficient if pan.c is compiled -DSAFETY

pan:1: assertion violated (mutex!=2) (at depth 11)

pan: wrote mutex\_assert.pml.trail

(Spin Version 6.4.3 -- 16 December 2014)

Warning: Search not completed

+ Partial Order Reduction

Full statespace search for:

never claim - (none specified)

assertion violations +

acceptance cycles - (not selected)

invalid end states +

State-vector 44 byte, depth reached 20, errors: 1

121 states, stored

47 states, matched

168 transitions (= stored+matched)

2 atomic steps

hash conflicts: 0 (resolved)

Stats on memory usage (in Megabytes):

0.008 equivalent memory usage for states (stored\*(State-vector + overhead))

0.291 actual memory usage for states

128.000 memory used for hash table (-w24)

0.534 memory used for DFS stack (-m10000)

128.730 total actual memory usage

pan: elapsed time 0.001 seconds

You can also use a zero as an argument to the first option forces the state space search to continue, even if errors are found.

$ ./pan -c0

1. Get the counterexample

$ spin -t -p mutex\_assert.pml

1. Please refer to <http://spinroot.com/spin/Man/Spin.html> for more options
2. *More exercises*
3. Use Spin to simulate or verify a model system specified in Promela (e.g. the “software-only” solution to the mutex problem in the lecture notes) and understand the results.
4. You are encouraged to practice on your own model and verify some properties learned from the course.

***Part IV: On-line references:***

[***http://spinroot.com/spin/Man/Manual.html***](http://spinroot.com/spin/Man/Manual.html)

[***http://spinroot.com/spin/Man/index.html***](http://spinroot.com/spin/Man/index.html)