Assignment 1: Promela

# Exercise 1.1: Verifying the log program

Checking the post-condition of the log program relies on calculating 2k. The verification process is listed below.

init

{

finished == 1; //wait for log calculation

int powk = 1;

int i = 0;

//calculate 2^k

do

:: i < k ->

powk = powk \* 2;

i = i + 1

:: else -> break

od;

//assert post condition

assert ( powk <= n && n <= powk\*2);

}

The commented code is attached.

# Exercise 1.2: Analysis of algorithm on slide 35

## What does the code do?

The two processes will take turns running the critical section - that is the processes perform one-bounded overtaking.

The commented code is attached in two files:

* assignment1\_2snippet.pml – The commented code.
* assignment1\_2verification.pml – The modified code with verification code.

## How does it work?

The programs uses three components:

* **flag**: Each process has a flag. The process sets the flag when ready to run the critical section.
* **turn:** Is a variable. The turn variable allows the first process hitting the guard to pass, when the other process requests a turn by setting the variable to its own \_pid. Without this variable, the system would be susceptible to a deadlock, if both processes sets their flags before passing the guard.
* **Guard:** The guard allows only one process to enter the critical section at a time. When hitting the guard a process will wait for the other process to complete the critical section, or the other to request a turn.

## How do you verify that?

The behavior of the processes is verified by using the LTL formula described in “Principles of the Spin Model Checker”, M. Ben-Ari, 2008, p. 91, henceforth known as “the book”. The processes are split into two distinct processes (user0 & user1) to allow the formula to differentiate the two. The following defines are added at the top of the model code.

#**define** u0try user0@try

#**define** u0cs user0@cs

#**define** u1cs user1@cs

The labels try and cs are added to the body of each process as shown below. The label try signifies the point in the execution where the process tries to enter the critical process. The label cs signifies the critical section of each process.

try: flag[1 - \_pid] == 0 || turn == 1 - \_pid;

cs: ncrit++;

The verification is performed by adding the following at the end of the mode code, thus applying the LTL formula using “acceptance” mode of the Spin tool.

**ltl** otk {

[](u0try -> (!u1cs U (u1cs U (!u1cs U u0cs))))

}

The LTL formula is explained in the book as follows:

*“…always, if process user0 is trying to enter its critical section( u0try is true), the computation must start with the following sequence of intervals: (a) process user1 is not in its critical section ( !u1cs ); (b) process user1 is in its critical section(u1cs); (c) again, process user1 is not in its critical section(!u1cs); and finally (d) process user0 is in its critical section (u0cs ).”*

The conclusion of the implication stated in the formula is illustrated below:

