Question 1

To check for 1 channel having no communication aborted, which means that any communication that sends a start signal will eventually send a finish signal:

$$G\ (\ channel = start \rightarrow\ F\ (channel = stop)\ )$$

There are 4 channels deployed in this system: `up[0], up[1], down[0], down[1]`. To check for any channel having no communication aborted, the above check will be checked against all channels.

G (

$$\big(up[0] = start\ \rightarrow\ F\ (up[0] = stop)\big)$$

$$\&\ (up[1] = start\ \rightarrow\ F\ (up[1] = stop))$$

$$\&\ (down[0] = start\ \rightarrow\ F\ (down[0] = stop))$$

$$\&\ (down[1] = start\ \rightarrow\ F\ (down[1] = stop))$$

)

Assuming that all messaging channels are lose-less, checking the above LTL property shows that the above property holds, therefore, there is no communication aborted in the protocol.

```
C:\Users\tanwe\Desktop\Git\CS4211_SMV2\Problem 1>nusmv -int q1.smv
*** This is NuSMV 2.6.0 (compiled on Wed Oct 14 15:37:51 2015)
*** Enabled addons are: compass
*** For more information on NuSMV see <http://nusmv.fbk.eu>
*** or email to <nusmv-users@list.fbk.eu>.
*** Please report bugs to <Please report bugs to <nusmv-users@fbk.eu>>

*** Copyright (c) 2010-2014, Fondazione Bruno Kessler

*** This version of NuSMV is linked to the CUDD library version 2.4.1
*** Copyright (c) 1995-2004, Regents of the University of Colorado

*** This version of NuSMV is linked to the MiniSat SAT solver.
*** See http://minisat.se/MiniSat.html
*** Copyright (c) 2003-2006, Niklas Een, Niklas Sorensson
*** Copyright (c) 2007-2010, Niklas Sorensson

NuSMV > go
WARNING *** Processes are still supported, but deprecated.      ***
WARNING *** In the future processes may be no longer supported. ***

WARNING *** The model contains PROCESSes or ISAs. ***
WARNING *** The HRC hierarchy will not be usable. ***
< (up[1]=stop)) & (down[0]=start -> F (down[0]=stop)) & (down[1]=start -> F (down[1]=stop)) )"
-- specification  G ((((up[0] = start ->  F up[0] = stop) & (up[1] = start ->  F up[1] = stop)) & (down[0] = start ->  F
 down[0] = stop)) & (down[1] = start ->  F down[1] = stop))  is true
NuSMV >
```

Question 2

A. To check if the solution preserves mutual exclusion, both processes should not have access to the critical section at the same time. I have created a local variable “localkey” for each process and if the “localkey” is set to true, it is currently executing the critical section, otherwise it will be set to false.

   Therefore, to check using the LTL property, I will check that both processes should not have “localkey” set to true at the same time.

$$G\,!\,(\,(process1.localkey\ =\ TRUE)\,\&\,(process2.localkey\ =\ TRUE)\,)$$

   By checking this LTL property against the SMV model, I can verify that the solution preserves mutual exclusion.

```
C:\Users\tanwe\Desktop\Git\CS4211_SMV2\Problem 2>nusmv -int q2.smv
*** This is NuSMV 2.6.0 (compiled on Wed Oct 14 15:37:51 2015)
*** Enabled addons are: compass
*** For more information on NuSMV see <http://nusmv.fbk.eu>
*** or email to <nusmv-users@list.fbk.eu>.
*** Please report bugs to <Please report bugs to <nusmv-users@fbk.eu>>

*** Copyright (c) 2010-2014, Fondazione Bruno Kessler

*** This version of NuSMV is linked to the CUDD library version 2.4.1
*** Copyright (c) 1995-2004, Regents of the University of Colorado

*** This version of NuSMV is linked to the MiniSat SAT solver.
*** See http://minisat.se/MiniSat.html
*** Copyright (c) 2003-2006, Niklas Een, Niklas Sorensson
*** Copyright (c) 2007-2010, Niklas Sorensson

NuSMV > go
WARNING *** Processes are still supported, but deprecated.      ***
WARNING *** In the future processes may be no longer supported. ***

WARNING *** The model contains PROCESSes or ISAs. ***
WARNING *** The HRC hierarchy will not be usable. ***
NuSMV > check_ltlspec -p "G !( (process1.localkey = TRUE) & (process2.localkey = TRUE) )"
-- specification  G !(process1.localkey = TRUE & process2.localkey = TRUE)  is true
NuSMV >
```

B. However, this solution is not a good solution. 1 process can keep acquiring the lock, starving the other process from the critical section. This LTL property can check if both processes eventually acquires the lock and run the critical section:

$$G\,(\,F\,(process1.localkey\ =\ TRUE)\,\&\,F\,(process2.localkey\ =\ TRUE)\,)$$

   This property is not true given the counterexample below.

   1. Process 1 acquires the lock.
   2. Process 2 waits for the lock.
   3. Process 1 releases the lock.
   4. Loop back to 1.

   Therefore, I can conclude that starvation occurs in this mutual exclusion solution.

```
NuSMV > check_ltlspec -p "G ( F (process1.localkey = TRUE) & F (process2.localkey = TRUE) )"
-- specification  G ( F process1.localkey = TRUE &  F process2.localkey = TRUE)  is false
-- as demonstrated by the following execution sequence
Trace Description: LTL Counterexample
Trace Type: Counterexample
  -- Loop starts here
  -> State: 1.1 <-
    lock = FALSE
    process1.localkey = FALSE
    process2.localkey = FALSE
  -> Input: 1.2 <-
    _process_selector_ = process1
    running = FALSE
    process2.running = FALSE
    process1.running = TRUE
  -> State: 1.2 <-
    lock = TRUE
    process1.localkey = TRUE
  -> Input: 1.3 <-
    _process_selector_ = process2
    process2.running = TRUE
    process1.running = FALSE
  -> State: 1.3 <-
  -> Input: 1.4 <-
    _process_selector_ = process1
    process2.running = FALSE
    process1.running = TRUE
  -> State: 1.4 <-
    lock = FALSE
    process1.localkey = FALSE
NuSMV >
```

Question 3

In my solution, each process keeps track of its state.

States:

- c0: The process is running c0 codes.
- c1: The process is running the first if-statement of c1.
- crit: This process is running the code marked as */* *critical section* **/*.

The codes */* *non-critical section* **/* is not given any state as it has been abstracted out from the SMV solution. This will not cause any change to the mutual exclusion problem.

A. To check if the solution preserves mutual exclusion, both processes should not have access to the critical section at the same time. To check using the LTL property, I will check that both processes should not have "crit" state at the same time.

$$G\,!\,(\,(process0.state\ =\ crit)\ \&\ (process1.state\ =\ crit)\,)$$

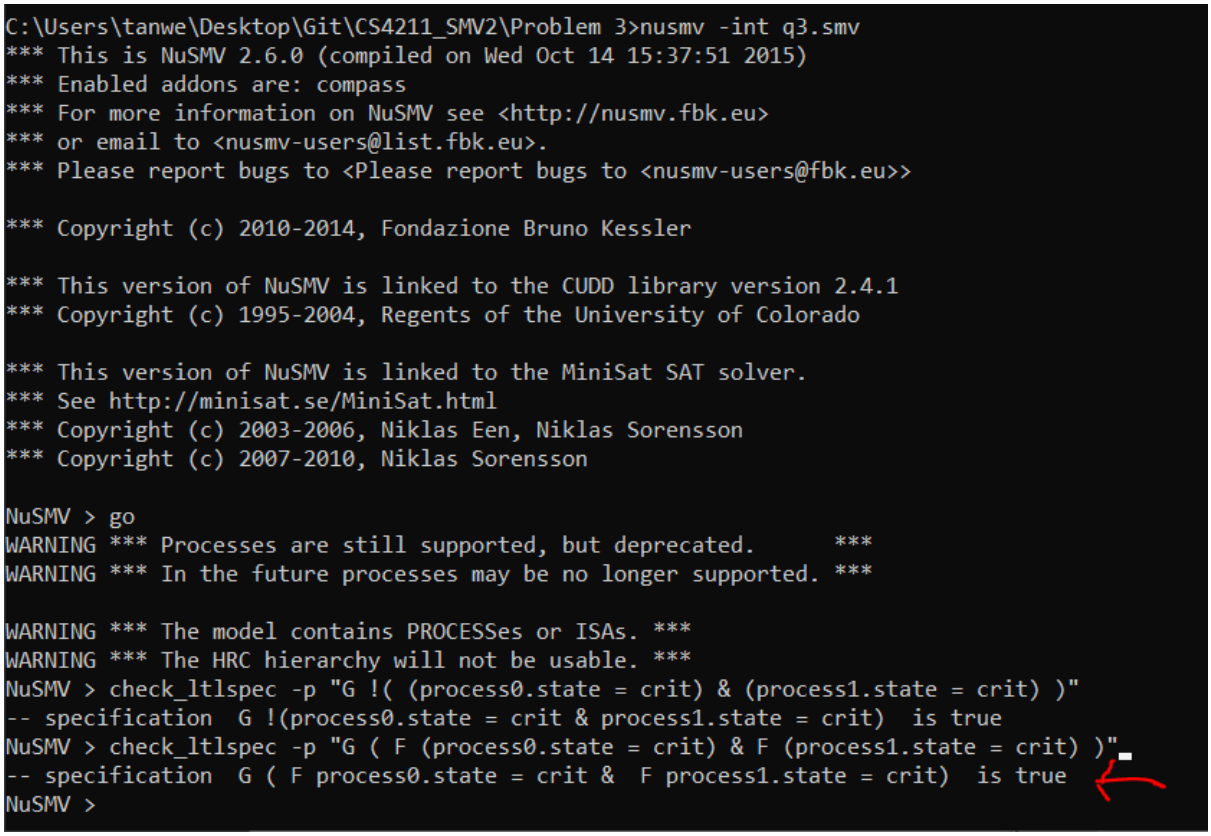
By checking this LTL property against the SMV model, I can verify that the solution preserves mutual exclusion.

```
C:\Users\tanwe\Desktop\Git\CS4211_SMV2\Problem 3>nusmv -int q3.smv
*** This is NuSMV 2.6.0 (compiled on Wed Oct 14 15:37:51 2015)
*** Enabled addons are: compass
*** For more information on NuSMV see <http://nusmv.fbk.eu>
*** or email to <nusmv-users@list.fbk.eu>.
*** Please report bugs to <Please report bugs to <nusmv-users@fbk.eu>>

*** Copyright (c) 2010-2014, Fondazione Bruno Kessler

*** This version of NuSMV is linked to the CUDD library version 2.4.1
*** Copyright (c) 1995-2004, Regents of the University of Colorado

*** This version of NuSMV is linked to the MiniSat SAT solver.
*** See http://minisat.se/MiniSat.html
*** Copyright (c) 2003-2006, Niklas Een, Niklas Sorensson
*** Copyright (c) 2007-2010, Niklas Sorensson

NuSMV > go
WARNING *** Processes are still supported, but deprecated.      ***
WARNING *** In the future processes may be no longer supported. ***

WARNING *** The model contains PROCESSes or ISAs. ***
WARNING *** The HRC hierarchy will not be usable. ***
NuSMV > check_ltlspec -p "G !( (process0.state = crit) & (process1.state = crit) )"
-- specification  G !(process0.state = crit & process1.state = crit)  is true
NuSMV >
```

B. To prove that this solution is starvation free, assuming that both processes have a fair chance of executing, I can check it using the LTL property that eventually both processes should be able to run the "crit" state.

$$G\,(\,F\,(process0.state\ =\ crit)\ \&\ F\,(process1.state\ =\ crit)\,)$$

By checking this LTL property against the SMV model, I can verify that the solution is starvation free.


C:\Users\tanwe\Desktop\Git\CS4211_SMV2\Problem 3>nusmv -int q3.smv
*** This is NuSMV 2.6.0 (compiled on Wed Oct 14 15:37:51 2015)
*** Enabled addons are: compass
*** For more information on NuSMV see <http://nusmv.fbk.eu>
*** or email to <nusmv-users@list.fbk.eu>.
*** Please report bugs to <Please report bugs to <nusmv-users@fbk.eu>>
*** Copyright (c) 2010-2014, Fondazione Bruno Kessler
*** This version of NuSMV is linked to the CUDD library version 2.4.1
*** Copyright (c) 1995-2004, Regents of the University of Colorado
*** This version of NuSMV is linked to the MiniSat SAT solver.
*** See http://minisat.se/MiniSat.html
*** Copyright (c) 2003-2006, Niklas Een, Niklas Sorensson
*** Copyright (c) 2007-2010, Niklas Sorensson
NuSMV > go
WARNING *** Processes are still supported, but deprecated. ***
WARNING *** In the future processes may be no longer supported. ***
WARNING *** The model contains PROCESSes or ISAs. ***
WARNING *** The HRC hierarchy will not be usable. ***
NuSMV > check_ltlspec -p "G !((process0.state = crit) & (process1.state = crit))"
-- specification G !(process0.state = crit & process1.state = crit) is true
NuSMV > check_ltlspec -p "G (F (process0.state = crit) & F (process1.state = crit))"
-- specification G (F process0.state = crit & F process1.state = crit) is true
NuSMV >