LAB 7 - Q2

Module 1: A simple counter

Task 1:

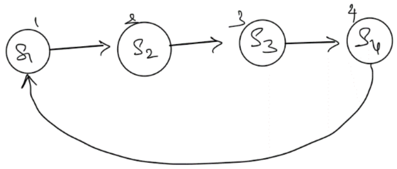
SPEC1 AG! (p.n=4) : From now on p.n state will never be 4
SPEC2 AG (p.n=1 -> AF p.n=4) : Whenever we will get state 1, in future we will definitely get state 4

Task 2:
SPEC1 AG! (p.n=4): is false, as we will eventually reach 4. Counterexample is shown in the below screenshot
SPEC2 AG (p.n=1 -> AF p.n=4): is true as it is a counter whenever we have 1 we will eventually reach 4

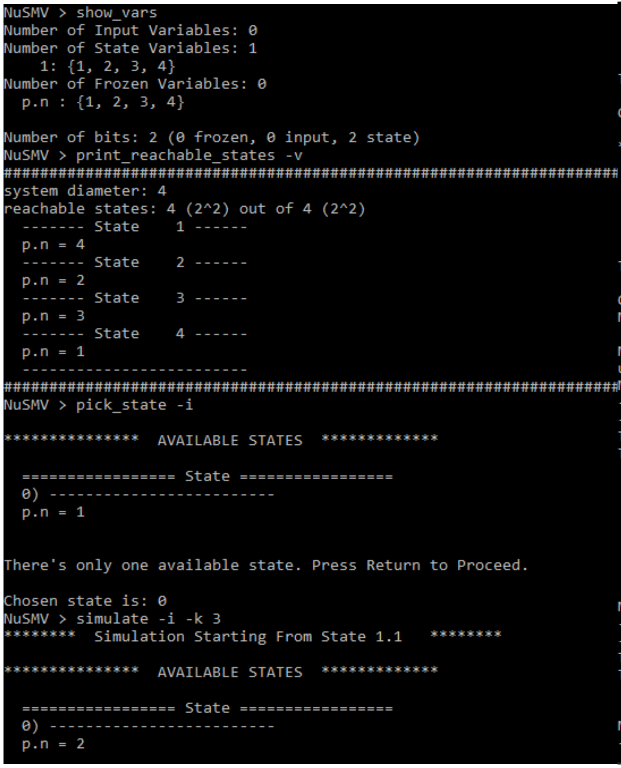
Task 3:

- Number of state variables = 1 i.e n
  p.n : {1,2,3,4}
- Reachable states:
  reachable states: 4 (2^2) out of 4 (2^2)
    p.n = 4
    p.n = 2
    p.n = 3
    p.n = 1
- There is only 1 state available to pick i.e. “p.n = 1”
- Simulation is shown in the below screenshots
- Traces have been saved

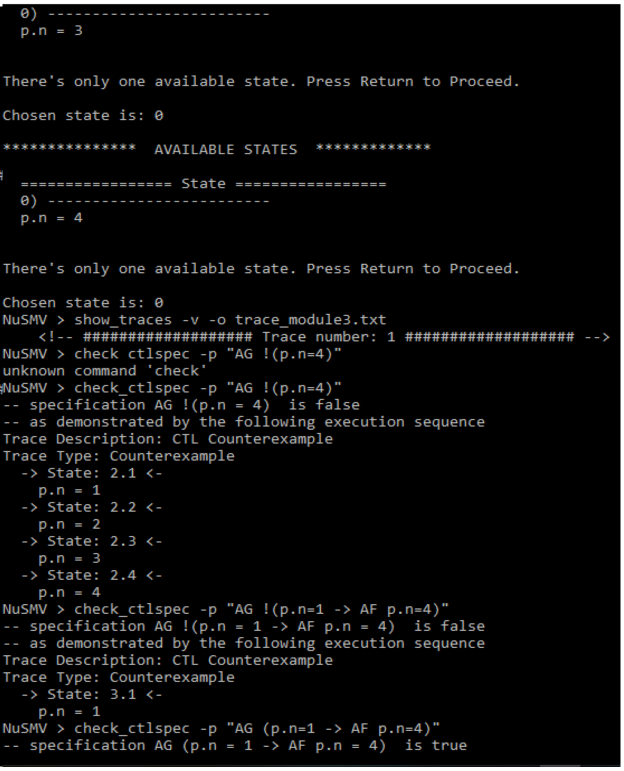
Transition Diagram:



LAB 7 - Q2



```
NuSMV > show_vars
Number of Input Variables: 0
Number of State Variables: 1
    1: {1, 2, 3, 4}
Number of Frozen Variables: 0
  p.n : {1, 2, 3, 4}

Number of bits: 2 (0 frozen, 0 input, 2 state)
NuSMV > print_reachable_states -v
####################################################################
system diameter: 4
reachable states: 4 (2^2) out of 4 (2^2)
  ------- State    1 ------
  p.n = 4
  ------- State    2 ------
  p.n = 2
  ------- State    3 ------
  p.n = 3
  ------- State    4 ------
  p.n = 1
  -------------------------
####################################################################
NuSMV > pick_state -i

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  p.n = 1


There's only one available state. Press Return to Proceed.

Chosen state is: 0
NuSMV > simulate -i -k 3
********  Simulation Starting From State 1.1   ********

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  p.n = 2
```



```
  0) -------------------------
  p.n = 3


There's only one available state. Press Return to Proceed.

Chosen state is: 0

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  p.n = 4


There's only one available state. Press Return to Proceed.

Chosen state is: 0
NuSMV > show_traces -v -o trace_module3.txt
    <!-- ################### Trace number: 1 ################### -->
NuSMV > check ctlspec -p "AG !(p.n=4)"
unknown command 'check'
NuSMV > check_ctlspec -p "AG !(p.n=4)"
-- specification AG !(p.n = 4)  is false
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
  -> State: 2.1 <-
    p.n = 1
  -> State: 2.2 <-
    p.n = 2
  -> State: 2.3 <-
    p.n = 3
  -> State: 2.4 <-
    p.n = 4
NuSMV > check_ctlspec -p "AG !(p.n=1 -> AF p.n=4)"
-- specification AG !(p.n = 1 -> AF p.n = 4)  is false
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
  -> State: 3.1 <-
    p.n = 1
NuSMV > check_ctlspec -p "AG (p.n=1 -> AF p.n=4)"
-- specification AG (p.n = 1 -> AF p.n = 4)  is true
```

Module 2: Semaphores

Task 1:

SPEC1 - "AG !(proc1.state = critical & proc2.state =critical)" : It will never happen that both the process are in critical state
SPEC2 - "AG(proc1.state = entering -> AF (proc1.state = critical))": It is possible that process 1 is entering state and may never reach critical

Task 2:

SPEC1 - "AG !(proc1.state = critical & proc2.state =critical)" : This specification is true as both the processes cannot be in critical state simultaneously
SPEC2 - "AG(proc1.state = entering -> AF (proc1.state = critical))" is false as it is not always true that if the process is in entering state it will go to critical stage . The counterexample is shown below

Task 3:

- Number of Input Variables: 1
    1: {main, proc2, proc1}
  Number of State Variables: 3
    2: {idle, entering, critical, exiting}
    1: boolean
  Number of Frozen Variables: 0
   semaphore : boolean

   proc1.state : {idle, entering, critical, exiting}

   proc2.state : {idle, entering, critical, exiting}

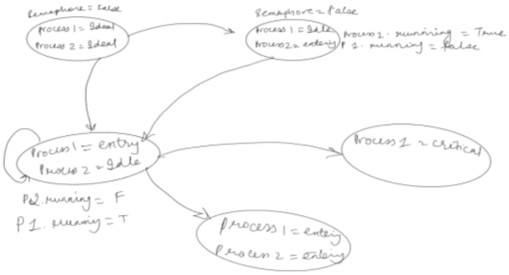
   _process_selector_ : {main, proc2, proc1}

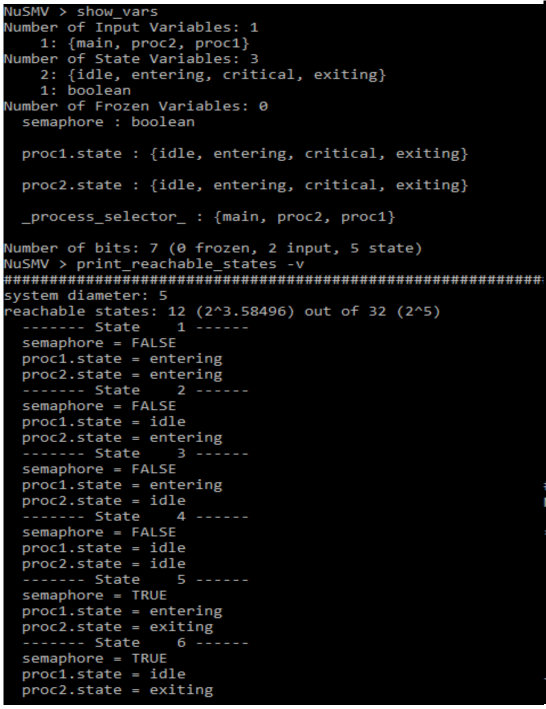
  Number of bits: 7 (0 frozen, 2 input, 5 state)

LAB 7 - Q2

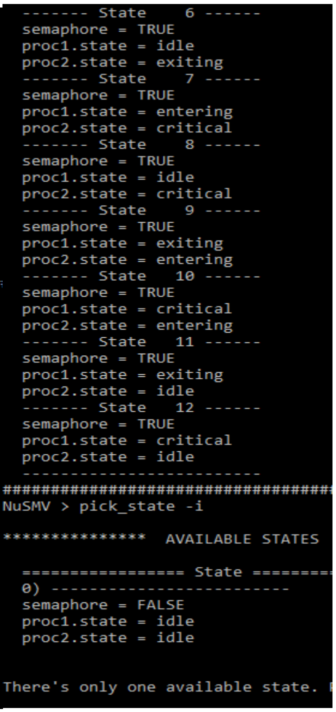
- Reachable States:
  system diameter: 5
  reachable states: 12 (2^3.58496) out of 32 (2^5)
- Pick state – Only 1 available state which is 0
- Simulation is shown in below screenshots
- Traces have been saved

Transition Diagram



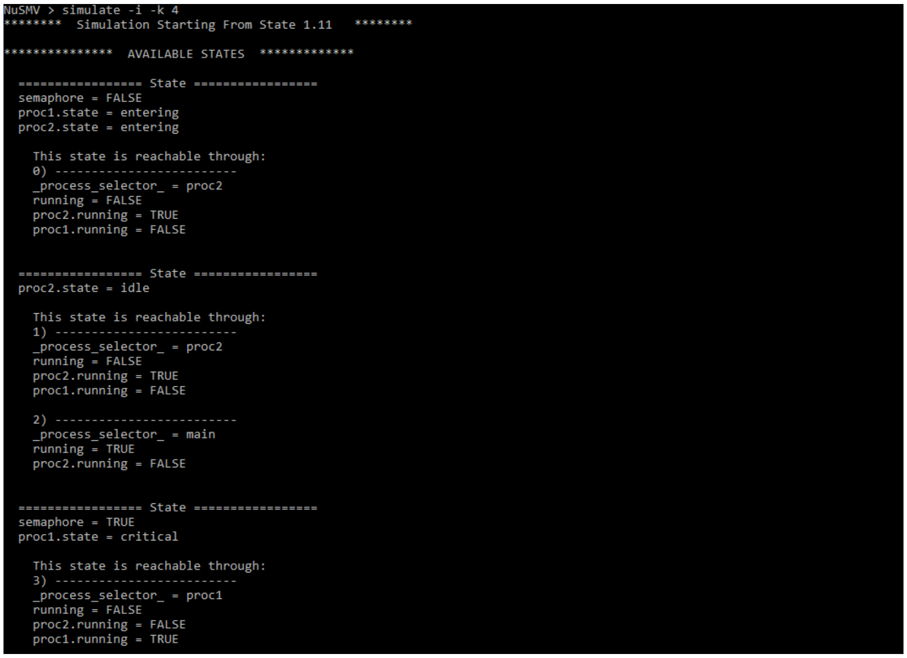


```
NuSMV > show_vars
Number of Input Variables: 1
    1: {main, proc2, proc1}
Number of State Variables: 3
    2: {idle, entering, critical, exiting}
    1: boolean
Number of Frozen Variables: 0
  semaphore : boolean

  proc1.state : {idle, entering, critical, exiting}

  proc2.state : {idle, entering, critical, exiting}

  _process_selector_ : {main, proc2, proc1}

Number of bits: 7 (0 frozen, 2 input, 5 state)
NuSMV > print_reachable_states -v
###########################################################
system diameter: 5
reachable states: 12 (2^3.58496) out of 32 (2^5)
  ------- State    1 ------
  semaphore = FALSE
  proc1.state = entering
  proc2.state = entering
  ------- State    2 ------
  semaphore = FALSE
  proc1.state = idle
  proc2.state = entering
  ------- State    3 ------
  semaphore = FALSE
  proc1.state = entering
  proc2.state = idle
  ------- State    4 ------
  semaphore = FALSE
  proc1.state = idle
  proc2.state = idle
  ------- State    5 ------
  semaphore = TRUE
  proc1.state = entering
  proc2.state = exiting
  ------- State    6 ------
  semaphore = TRUE
  proc1.state = idle
  proc2.state = exiting
```

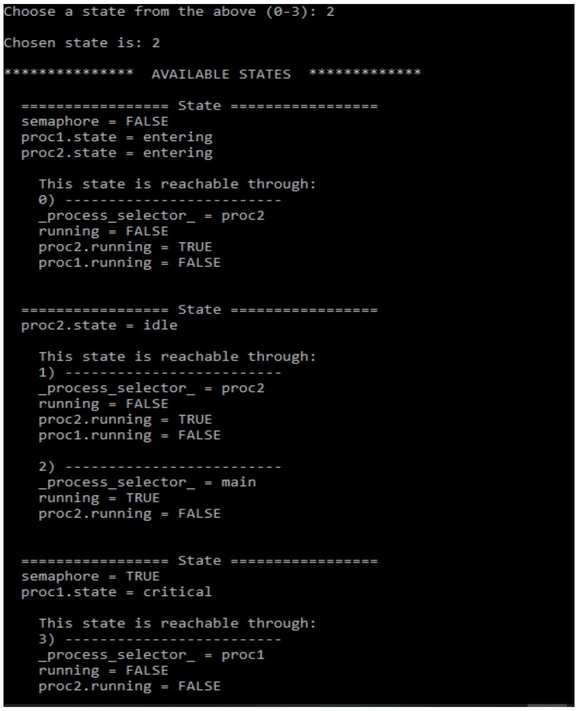


```
  ------- State    6 ------
  semaphore = TRUE
  proc1.state = idle
  proc2.state = exiting
  ------- State    7 ------
  semaphore = TRUE
  proc1.state = entering
  proc2.state = critical
  ------- State    8 ------
  semaphore = TRUE
  proc1.state = idle
  proc2.state = critical
  ------- State    9 ------
  semaphore = TRUE
  proc1.state = exiting
  proc2.state = entering
  ------- State   10 ------
  semaphore = TRUE
  proc1.state = critical
  proc2.state = entering
  ------- State   11 ------
  semaphore = TRUE
  proc1.state = exiting
  proc2.state = idle
  ------- State   12 ------
  semaphore = TRUE
  proc1.state = critical
  proc2.state = idle
  -------------------------
####################################
NuSMV > pick_state -i

***************  AVAILABLE STATES

  ================= State =========
  0) -------------------------
  semaphore = FALSE
  proc1.state = idle
  proc2.state = idle


There's only one available state. P
```

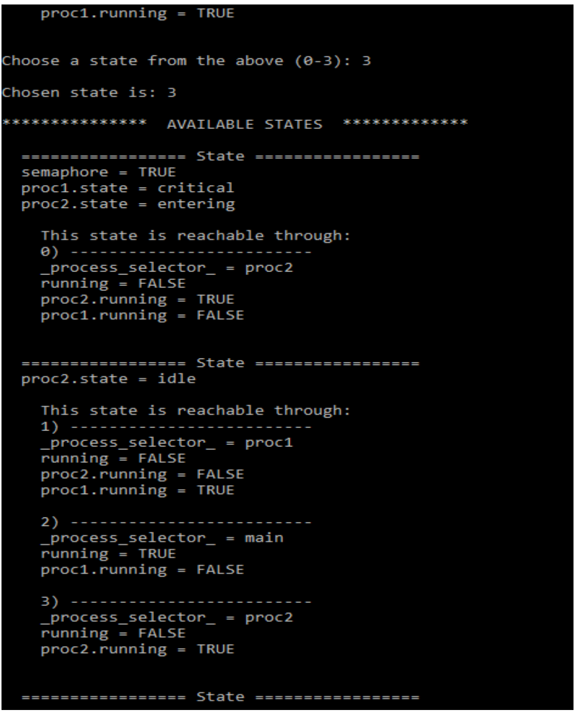
LAB 7 - Q2



```
NuSMV > simulate -i -k 4
********  Simulation Starting From State 1.11   ********

***************  AVAILABLE STATES  *************

  ================= State =================
  semaphore = FALSE
  proc1.state = entering
  proc2.state = entering

    This state is reachable through:
    0) -------------------------
    _process_selector_ = proc2
    running = FALSE
    proc2.running = TRUE
    proc1.running = FALSE


  ================= State =================
  proc2.state = idle

    This state is reachable through:
    1) -------------------------
    _process_selector_ = proc2
    running = FALSE
    proc2.running = TRUE
    proc1.running = FALSE

    2) -------------------------
    _process_selector_ = main
    running = TRUE
    proc2.running = FALSE


  ================= State =================
  semaphore = TRUE
  proc1.state = critical

    This state is reachable through:
    3) -------------------------
    _process_selector_ = proc1
    running = FALSE
    proc2.running = FALSE
    proc1.running = TRUE
```

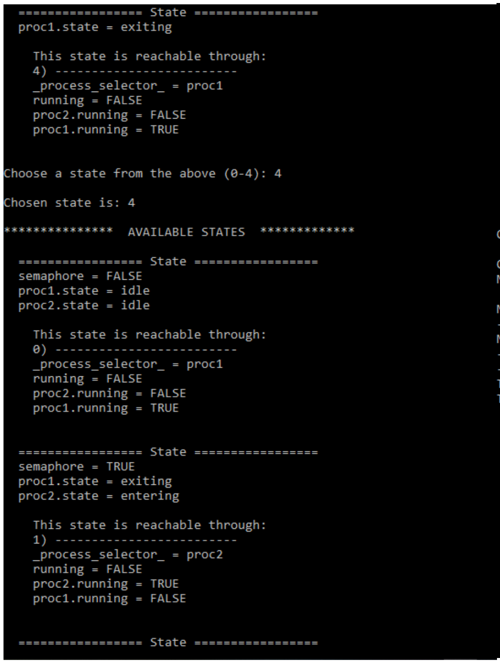


```
Choose a state from the above (0-3): 2

Chosen state is: 2

***************  AVAILABLE STATES  *************

  ================= State =================
  semaphore = FALSE
  proc1.state = entering
  proc2.state = entering

    This state is reachable through:
    0) -------------------------
    _process_selector_ = proc2
    running = FALSE
    proc2.running = TRUE
    proc1.running = FALSE


  ================= State =================
  proc2.state = idle

    This state is reachable through:
    1) -------------------------
    _process_selector_ = proc2
    running = FALSE
    proc2.running = TRUE
    proc1.running = FALSE

    2) -------------------------
    _process_selector_ = main
    running = TRUE
    proc2.running = FALSE


  ================= State =================
  semaphore = TRUE
  proc1.state = critical

    This state is reachable through:
    3) -------------------------
    _process_selector_ = proc1
    running = FALSE
    proc2.running = FALSE
```

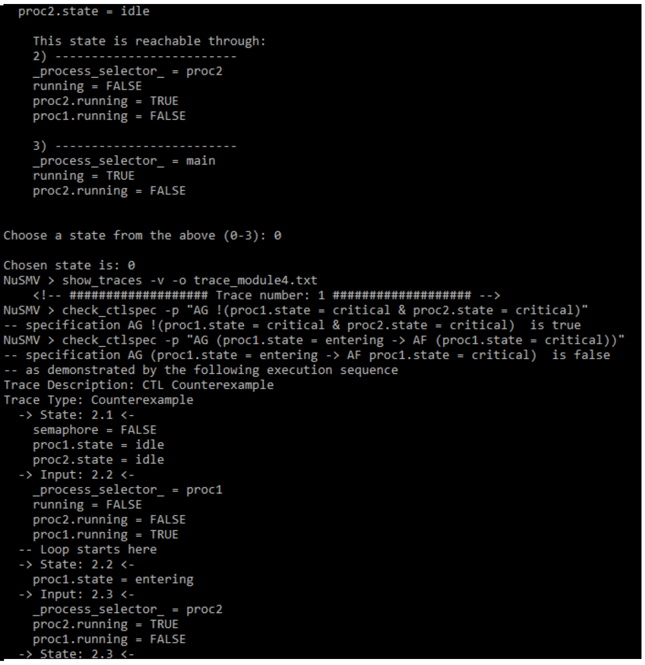


```
    proc1.running = TRUE


Choose a state from the above (0-3): 3

Chosen state is: 3

***************  AVAILABLE STATES  *************

  ================= State =================
  semaphore = TRUE
  proc1.state = critical
  proc2.state = entering

    This state is reachable through:
    0) -------------------------
    _process_selector_ = proc2
    running = FALSE
    proc2.running = TRUE
    proc1.running = FALSE


  ================= State =================
  proc2.state = idle

    This state is reachable through:
    1) -------------------------
    _process_selector_ = proc1
    running = FALSE
    proc2.running = FALSE
    proc1.running = TRUE

    2) -------------------------
    _process_selector_ = main
    running = TRUE
    proc1.running = FALSE

    3) -------------------------
    _process_selector_ = proc2
    running = FALSE
    proc2.running = TRUE


  ================= State =================
```

LAB 7 - Q2



```
  ================= State =================
  proc1.state = exiting

    This state is reachable through:
    4) -------------------------
    _process_selector_ = proc1
    running = FALSE
    proc2.running = FALSE
    proc1.running = TRUE


Choose a state from the above (0-4): 4

Chosen state is: 4

***************  AVAILABLE STATES  *************

  ================= State =================
  semaphore = FALSE
  proc1.state = idle
  proc2.state = idle

    This state is reachable through:
    0) -------------------------
    _process_selector_ = proc1
    running = FALSE
    proc2.running = FALSE
    proc1.running = TRUE


  ================= State =================
  semaphore = TRUE
  proc1.state = exiting
  proc2.state = entering

    This state is reachable through:
    1) -------------------------
    _process_selector_ = proc2
    running = FALSE
    proc2.running = TRUE
    proc1.running = FALSE


  ================= State =================
```



```
  proc2.state = idle

    This state is reachable through:
    2) -------------------------
    _process_selector_ = proc2
    running = FALSE
    proc2.running = TRUE
    proc1.running = FALSE

    3) -------------------------
    _process_selector_ = main
    running = TRUE
    proc2.running = FALSE


Choose a state from the above (0-3): 0

Chosen state is: 0
NuSMV > show_traces -v -o trace_module4.txt
    <!-- ################### Trace number: 1 ################### -->
NuSMV > check_ctlspec -p "AG !(proc1.state = critical & proc2.state = critical)"
-- specification AG !(proc1.state = critical & proc2.state = critical)  is true
NuSMV > check_ctlspec -p "AG (proc1.state = entering -> AF (proc1.state = critical))"
-- specification AG (proc1.state = entering -> AF proc1.state = critical)  is false
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
  -> State: 2.1 <-
    semaphore = FALSE
    proc1.state = idle
    proc2.state = idle
  -> Input: 2.2 <-
    _process_selector_ = proc1
    running = FALSE
    proc2.running = FALSE
    proc1.running = TRUE
  -- Loop starts here
  -> State: 2.2 <-
    proc1.state = entering
  -> Input: 2.3 <-
    _process_selector_ = proc2
    proc2.running = TRUE
    proc1.running = FALSE
  -> State: 2.3 <-
```

Module 3: Spin

Task 1:

check_ctlspec - p "AG (request -> AF state = busy)": Whenever we have a request we will at some point of time get busy

check_ltlspec - p "G (request -> AF state = busy)": This is not a valid LTL formula as AF is a CTL property

check_ctlspec - p "G (request -> F state = busy)": This is not a valid CTL property. However, it is a valid LTL property which states that whenever we have a request we will get busy

check_ctlspec - p "EG (request -> AG state = busy)": states that sometime when we have request we will always be busy

Task 2:

"AG (request -> AF state = busy)": is true
"G (request -> AF state = busy)": Not a valid formula
"G (request -> F state = busy)": is true
"EG (request -> AG state = busy)": I s not valid as we will eventually come out of busy state. Counterexample is shown below:

Task 3:

- Number of Input Variables: 0
  Number of State Variables: 2
      1: boolean
      1: {ready, busy}
  Number of Frozen Variables: 0
    request : boolean

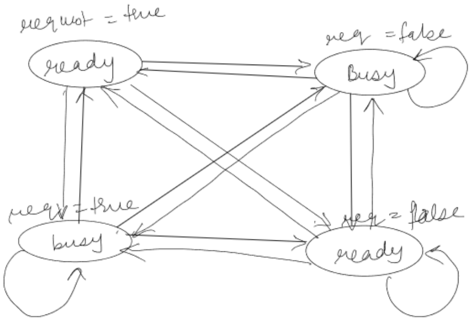
    state : {ready, busy}
- Reachable states:
  reachable states: 4 (2^2) out of 4 (2^2)
    ------- State   1 ------
    request = TRUE

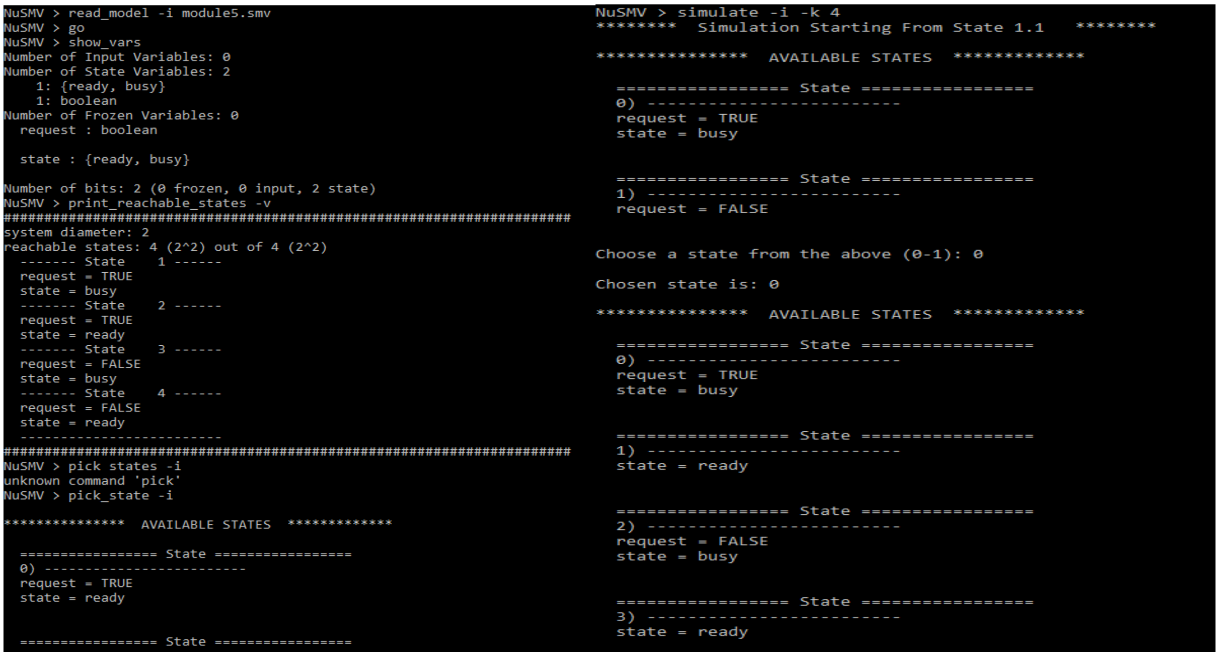
LAB 7 - Q2

state = busy
------- State   2 ------
request = TRUE
state = ready
------- State   3 ------
request = FALSE
state = busy
------- State   4 ------
request = FALSE
state = ready

- There are 2 available states:
  0) -------------------------
  request = TRUE
  state = ready
  1) -------------------------
  request = FALSE
- Simulation is shown in screenshot
- Traces have been saved

Transition Diagram

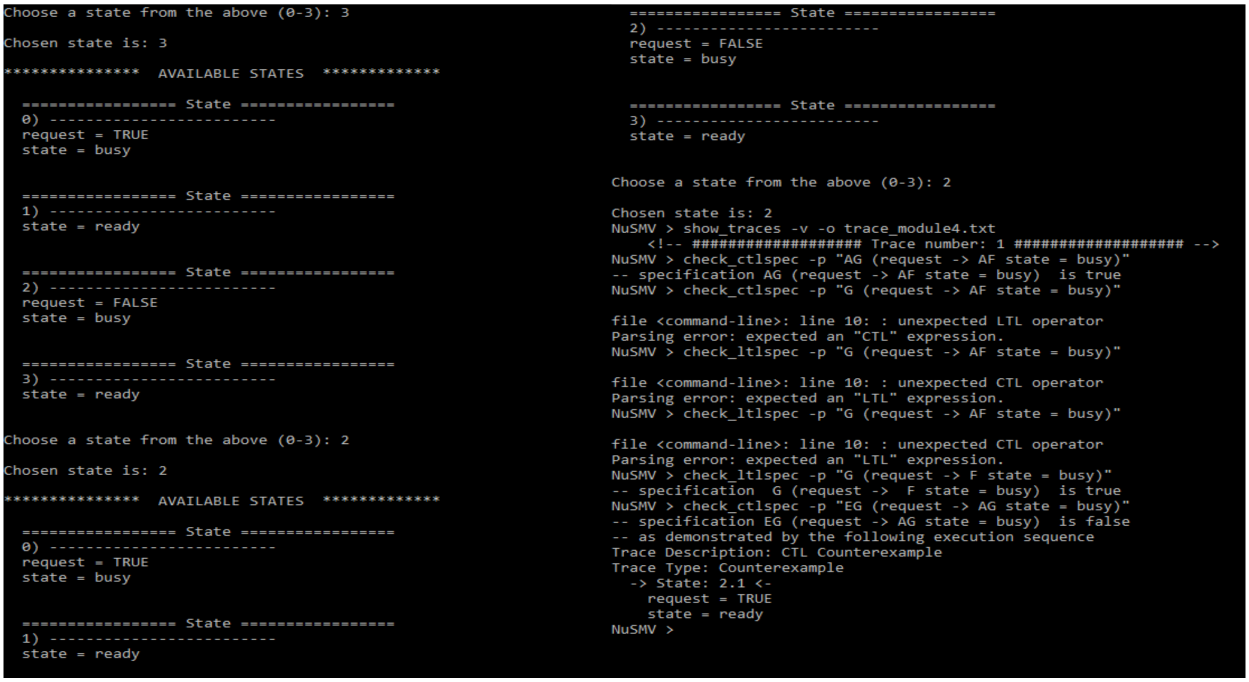




```
NuSMV > read_model -i module5.smv
NuSMV > go
NuSMV > show_vars
Number of Input Variables: 0
Number of State Variables: 2
    1: {ready, busy}
    1: boolean
Number of Frozen Variables: 0
  request : boolean

  state : {ready, busy}

Number of bits: 2 (0 frozen, 0 input, 2 state)
NuSMV > print_reachable_states -v
######################################################################
system diameter: 2
reachable states: 4 (2^2) out of 4 (2^2)
  ------- State    1 ------
  request = TRUE
  state = busy
  ------- State    2 ------
  request = TRUE
  state = ready
  ------- State    3 ------
  request = FALSE
  state = busy
  ------- State    4 ------
  request = FALSE
  state = ready
  -------------------------
######################################################################
NuSMV > pick states -i
unknown command 'pick'
NuSMV > pick_state -i

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  request = TRUE
  state = ready


  ================= State =================
```

```
NuSMV > simulate -i -k 4
********  Simulation Starting From State 1.1   ********

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  request = TRUE
  state = busy


  ================= State =================
  1) -------------------------
  request = FALSE


Choose a state from the above (0-1): 0

Chosen state is: 0

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  request = TRUE
  state = busy


  ================= State =================
  1) -------------------------
  state = ready


  ================= State =================
  2) -------------------------
  request = FALSE
  state = busy


  ================= State =================
  3) -------------------------
  state = ready
```

LAB 7 - Q2



```
Choose a state from the above (0-3): 3

Chosen state is: 3

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  request = TRUE
  state = busy


  ================= State =================
  1) -------------------------
  state = ready


  ================= State =================
  2) -------------------------
  request = FALSE
  state = busy


  ================= State =================
  3) -------------------------
  state = ready

Choose a state from the above (0-3): 2

Chosen state is: 2

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  request = TRUE
  state = busy


  ================= State =================
  1) -------------------------
  state = ready
```

```
  ================= State =================
  2) -------------------------
  request = FALSE
  state = busy


  ================= State =================
  3) -------------------------
  state = ready


Choose a state from the above (0-3): 2

Chosen state is: 2
NuSMV > show_traces -v -o trace_module4.txt
    <!-- ################### Trace number: 1 ################### -->
NuSMV > check_ctlspec -p "AG (request -> AF state = busy)"
-- specification AG (request -> AF state = busy)  is true
NuSMV > check_ctlspec -p "G (request -> AF state = busy)"

file <command-line>: line 10: : unexpected LTL operator
Parsing error: expected an "CTL" expression.
NuSMV > check_ltlspec -p "G (request -> AF state = busy)"

file <command-line>: line 10: : unexpected CTL operator
Parsing error: expected an "LTL" expression.
NuSMV > check_ltlspec -p "G (request -> AF state = busy)"

file <command-line>: line 10: : unexpected CTL operator
Parsing error: expected an "LTL" expression.
NuSMV > check_ltlspec -p "G (request -> F state = busy)"
-- specification  G (request ->  F state = busy)  is true
NuSMV > check_ctlspec -p "EG (request -> AG state = busy)"
-- specification EG (request -> AG state = busy)  is false
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
  -> State: 2.1 <-
    request = TRUE
    state = ready
NuSMV >
```

<u>Module 4:</u> Traffic lights

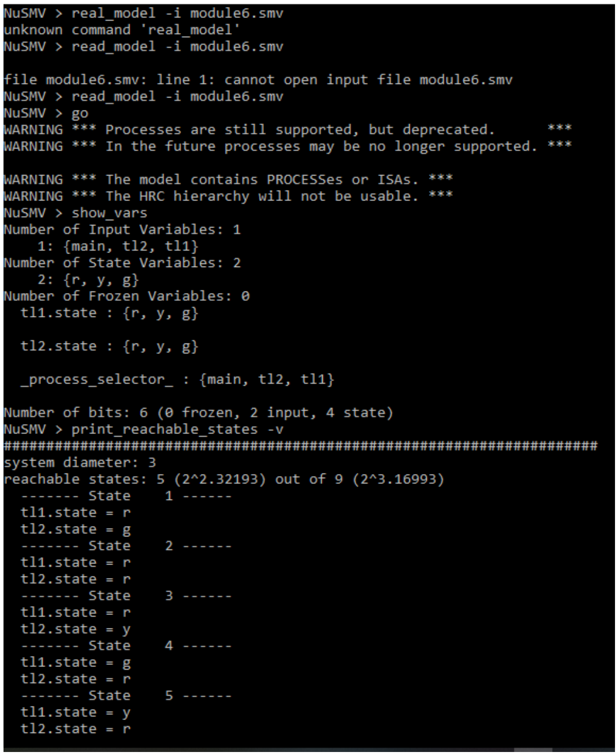
Task 1: ! F (tl1.state = g & tl2.state = g) : Both the greens lights of 2 states cannot be green at same time

Task 2: ! F (tl1.state = g & tl2.state = g) the property holds true

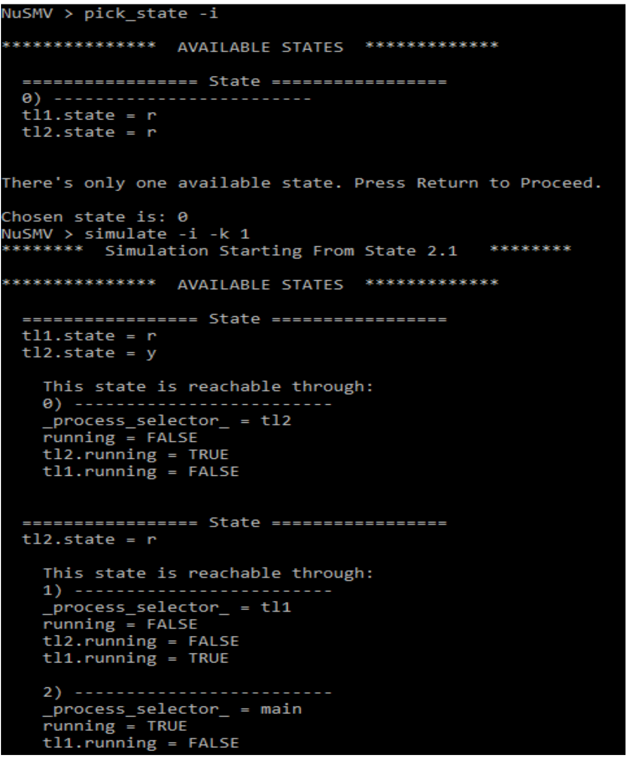
Task 3:

- Number of variables
  Number of Input Variables: 1
  1: {main, tl2, tl1}
  Number of State Variables: 2
  2: {r, y, g}
  Number of Frozen Variables: 0
  tl1.state : {r, y, g}
  tl2.state : {r, y, g}
  _process_selector_ : {main, tl2, tl1}
  Number of bits: 6 (0 frozen, 2 input, 4 state)
- 5 out of 9 are reachable states
- Only 1 state available to pick
- Simulation is shown below
- Traces were saved

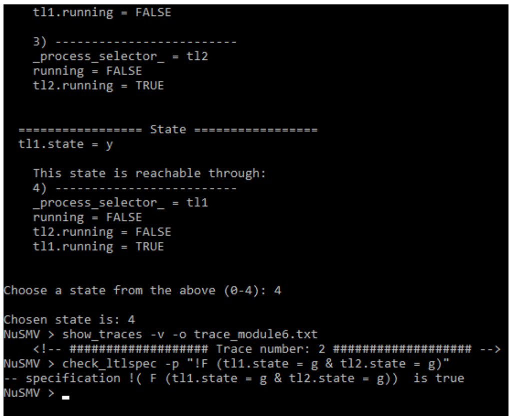
LAB 7 - Q2



```
NuSMV > real_model -i module6.smv
unknown command 'real_model'
NuSMV > read_model -i module6.smv

file module6.smv: line 1: cannot open input file module6.smv
NuSMV > read_model -i module6.smv
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 > show_vars
Number of Input Variables: 1
    1: {main, tl2, tl1}
Number of State Variables: 2
    2: {r, y, g}
Number of Frozen Variables: 0
  tl1.state : {r, y, g}

  tl2.state : {r, y, g}

  _process_selector_ : {main, tl2, tl1}

Number of bits: 6 (0 frozen, 2 input, 4 state)
NuSMV > print_reachable_states -v
######################################################################
system diameter: 3
reachable states: 5 (2^2.32193) out of 9 (2^3.16993)
  ------- State    1 ------
  tl1.state = r
  tl2.state = g
  ------- State    2 ------
  tl1.state = r
  tl2.state = r
  ------- State    3 ------
  tl1.state = r
  tl2.state = y
  ------- State    4 ------
  tl1.state = g
  tl2.state = r
  ------- State    5 ------
  tl1.state = y
  tl2.state = r
```



```
NuSMV > pick_state -i

***************  AVAILABLE STATES  *************

  ================= State =================
  0) -------------------------
  tl1.state = r
  tl2.state = r


There's only one available state. Press Return to Proceed.

Chosen state is: 0
NuSMV > simulate -i -k 1
********  Simulation Starting From State 2.1   ********

***************  AVAILABLE STATES  *************

  ================= State =================
  tl1.state = r
  tl2.state = y

    This state is reachable through:
    0) -------------------------
    _process_selector_ = tl2
    running = FALSE
    tl2.running = TRUE
    tl1.running = FALSE


  ================= State =================
  tl2.state = r

    This state is reachable through:
    1) -------------------------
    _process_selector_ = tl1
    running = FALSE
    tl2.running = FALSE
    tl1.running = TRUE

    2) -------------------------
    _process_selector_ = main
    running = TRUE
    tl1.running = FALSE
```



```
    tl1.running = FALSE

    3) -------------------------
    _process_selector_ = tl2
    running = FALSE
    tl2.running = TRUE


  ================= State =================
  tl1.state = y

    This state is reachable through:
    4) -------------------------
    _process_selector_ = tl1
    running = FALSE
    tl2.running = FALSE
    tl1.running = TRUE


Choose a state from the above (0-4): 4

Chosen state is: 4
NuSMV > show_traces -v -o trace_module6.txt
    <!-- ################### Trace number: 2 ################### -->
NuSMV > check_ltlspec -p "!F (tl1.state = g & tl2.state = g)"
-- specification !( F (tl1.state = g & tl2.state = g))  is true
NuSMV >
```