Unit-2: Model-checker NuSMV

B. Srivathsan

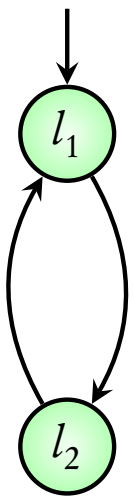
Chennai Mathematical Institute

NPTEL-course

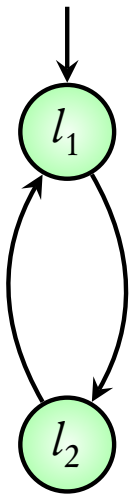
July - November 2015

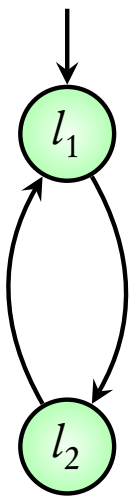
Module 2:

Simple models in NuSMV

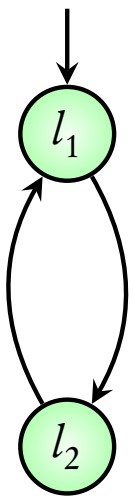

l_1
l_2

```
MODULE main
```

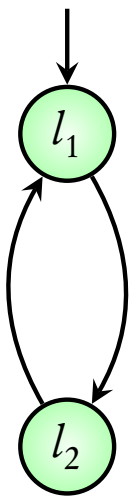




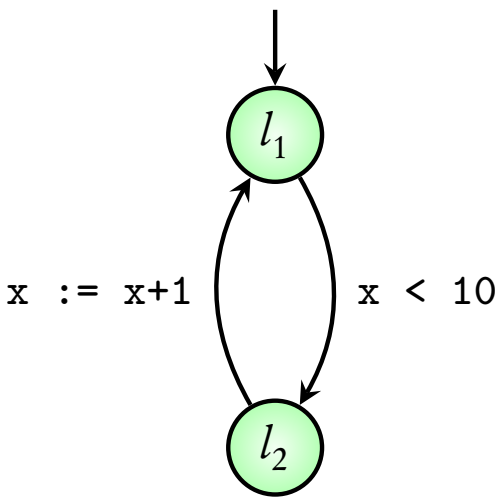

```
MODULE main

VAR

    location:  {l1,l2};
```

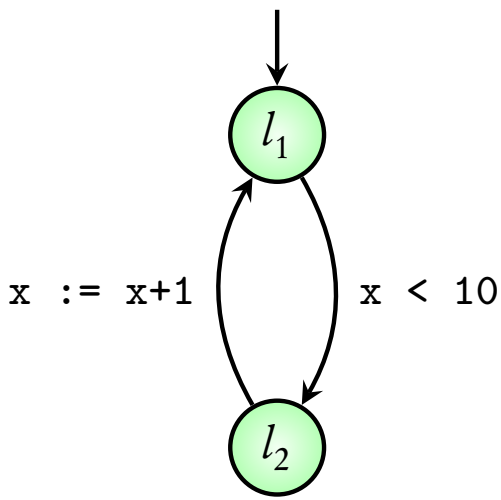



```
MODULE main
VAR
    location:  {l1,l2};
ASSIGN
    init(location) := l1;
```

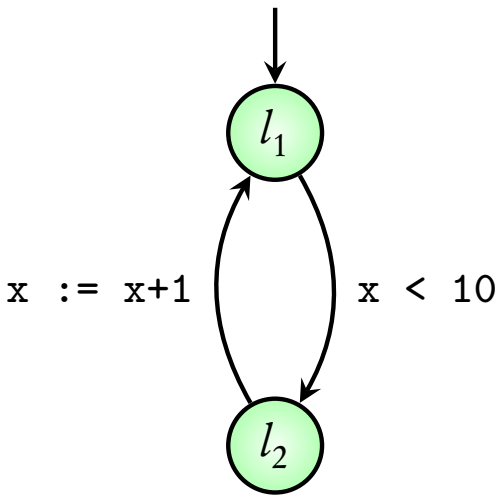



```
MODULE main
VAR
   location:  {l1,l2};
ASSIGN
   init(location) := l1;
   next(location) := case
                (location = l1) :  l2;
                (location = l2) :  l1;
                     esac;
```

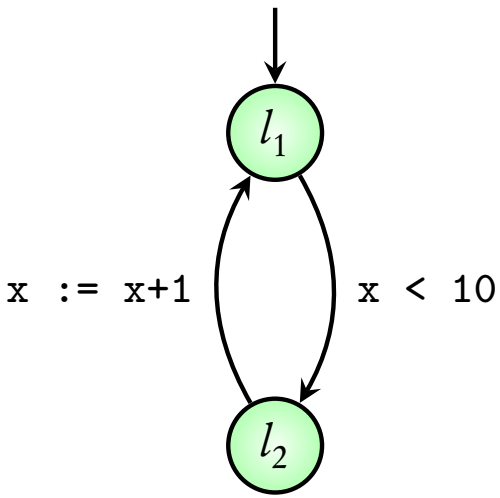

l_1
x := x+1
x < 10
l_2



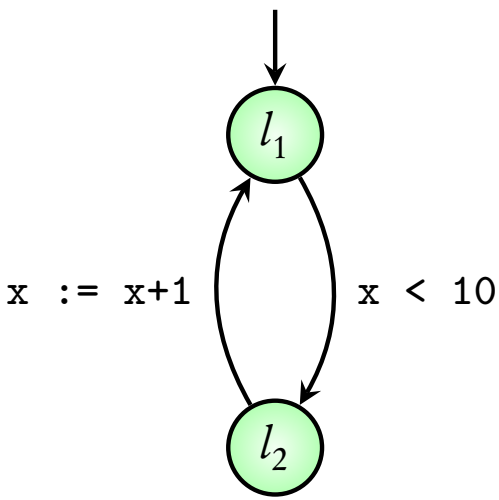

```
MODULE main
VAR
    location:  {l1,l2};

ASSIGN
    init(location) := l1;

    next(location) := case
                  (location = l1):  l2;
                  (location = l2) :  l1;

                       esac;
```



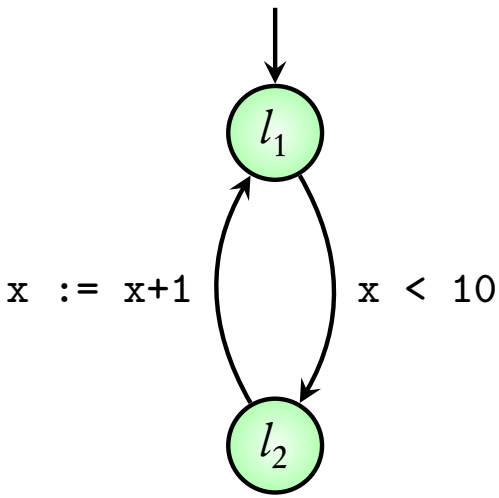

```
MODULE main
VAR
    location:  {l1,l2};
    x:  0 ..  100;
ASSIGN
    init(location) := l1;

    next(location) := case
                  (location = l1):  l2;
                  (location = l2) :  l1;

                       esac;
```



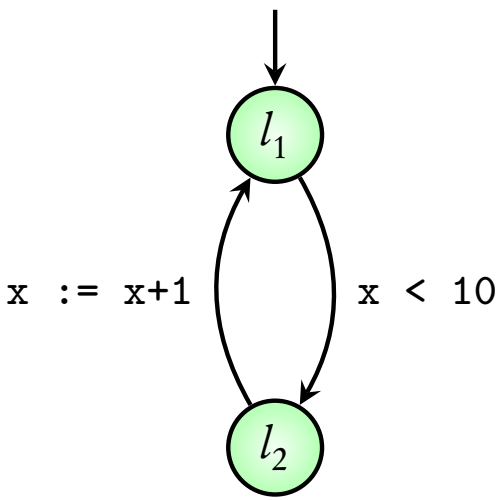

```
MODULE main
VAR
    location:  {l1,l2};
    x:  0 ..  100;
ASSIGN
    init(location) := l1;
    init(x) := 0;
    next(location) := case
                  (location = l1):  l2;
                  (location = l2) :  l1;

                      esac;
```




```
MODULE main
VAR
    location:  {l1,l2};
    x:  0 ..  100;
ASSIGN
    init(location) := l1;
    init(x) := 0;
    next(location) := case
                  (location = l1) & (x<10):  l2;
                  (location = l2) :  l1;

                       esac;
```

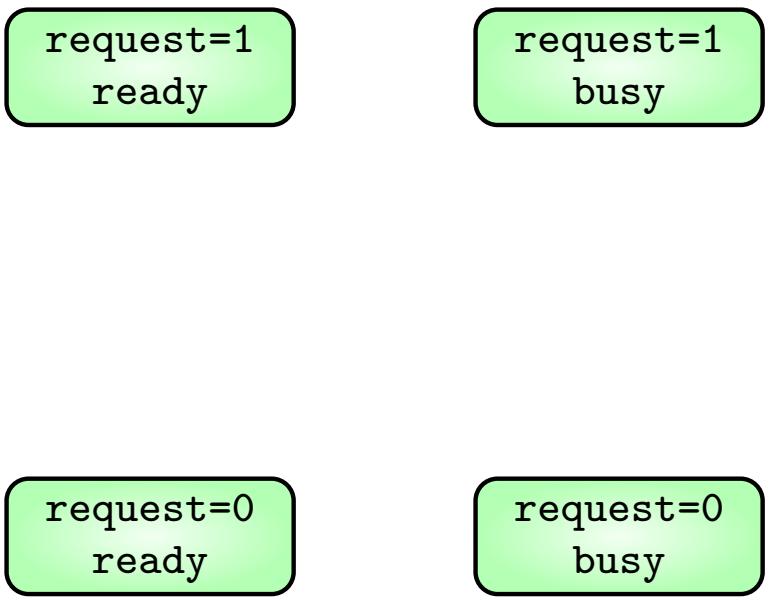



```
MODULE main
VAR
    location:  {l1,l2};
    x:  0 ..  100;
ASSIGN
    init(location) := l1;
    init(x) := 0;
    next(location) := case
                  (location = l1) & (x<10):  l2;
                  (location = l2) :  l1;
                  TRUE: location;
                       esac;
```

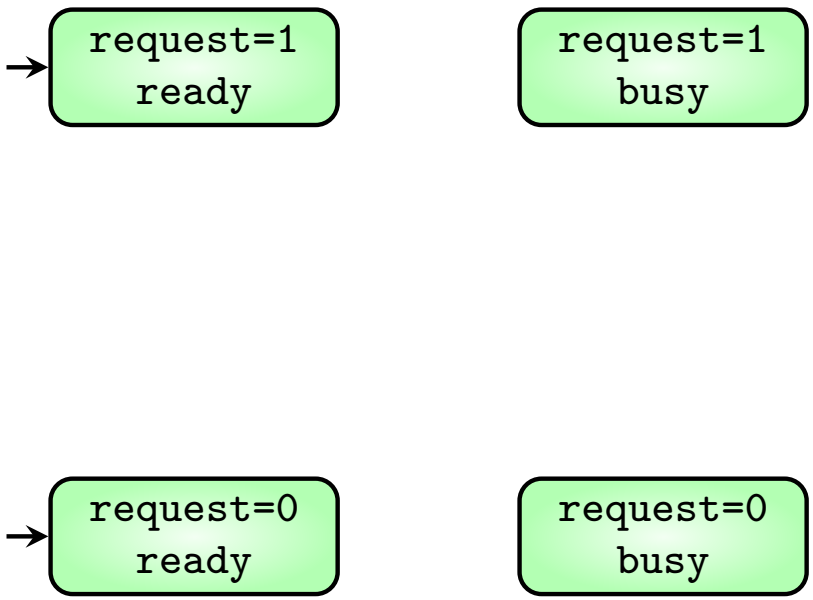



```
MODULE main
VAR
    location:  {l1,l2};
    x:  0 ..  100;
ASSIGN
    init(location) := l1;
    init(x) := 0;
    next(location) := case
                  (location = l1) & (x<10):  l2;
                  (location = l2) :  l1;
                  TRUE: location;
                       esac;
    next(x) := case
                  (location = l2) & x < 100:  x+1;
                  TRUE: x;
                esac;
```

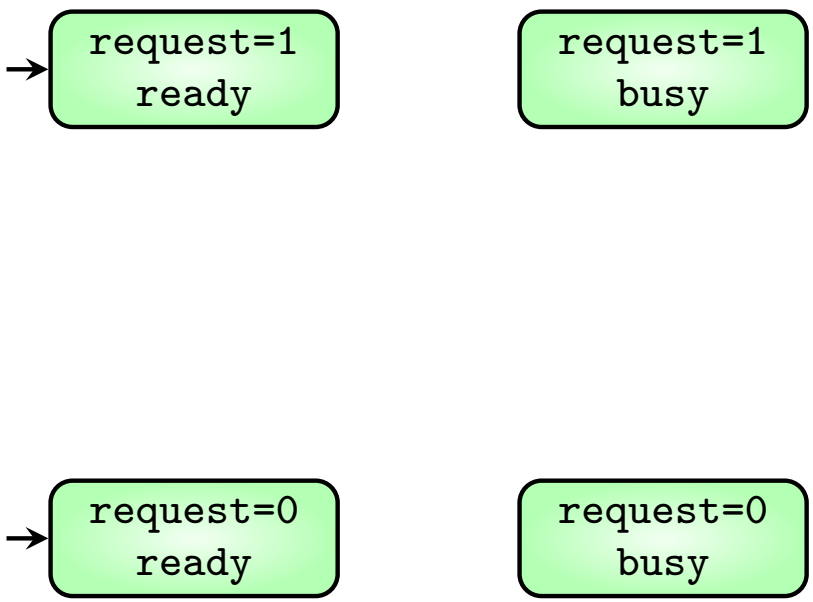
```
MODULE main
VAR
    request:  boolean;
    status:  {ready, busy}
```



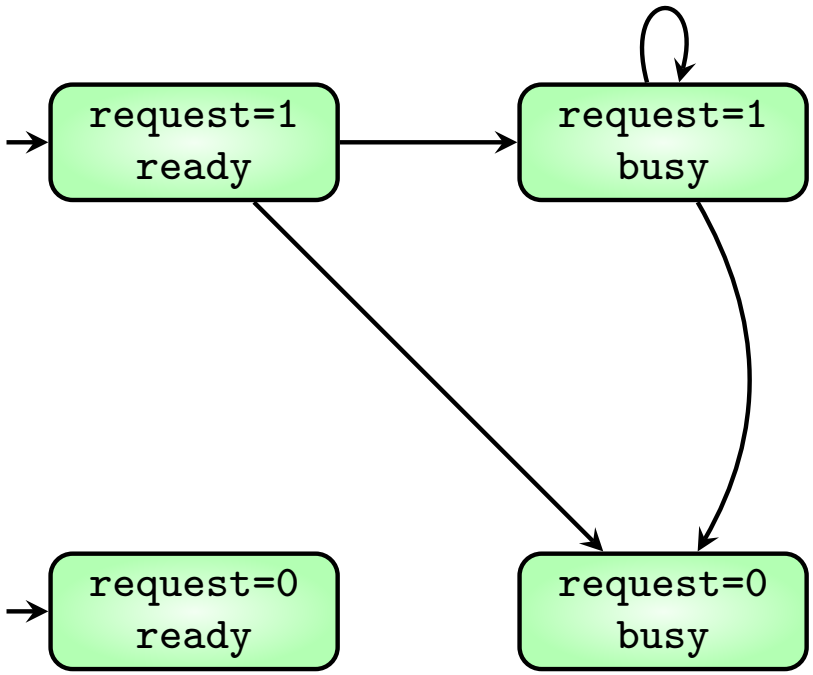

```
MODULE main
VAR
    request:  boolean;
    status:  {ready, busy}
```



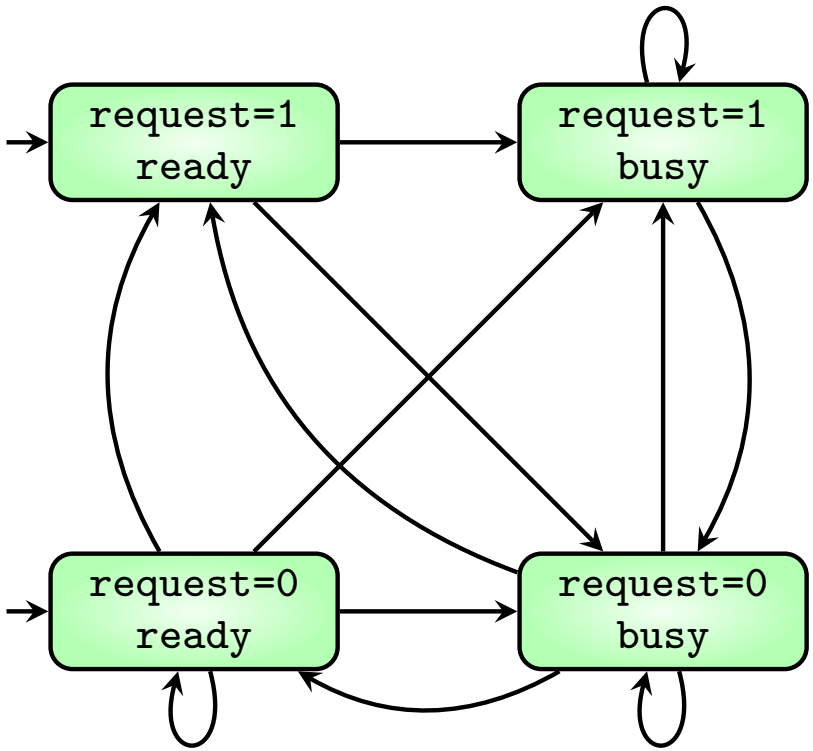

```
MODULE main
VAR
    request:  boolean;
    status:  {ready, busy}
ASSIGN
    init(status) := ready;
```




```
MODULE main
VAR
    request:  boolean;
    status:  {ready, busy}
ASSIGN
    init(status) := ready;
    next(status) := case
                    request :  busy;
                    TRUE : {ready,busy};
                          esac;
```

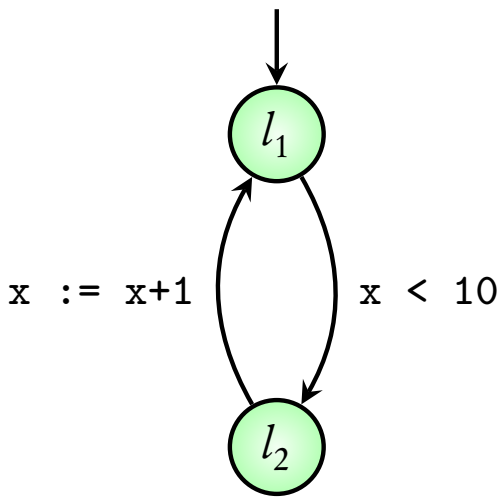



```
MODULE main
VAR
    request:  boolean;
    status:  {ready, busy}
ASSIGN
    init(status) := ready;
    next(status) := case
                    request :  busy;
                    TRUE : {ready,busy};
                          esac;
```

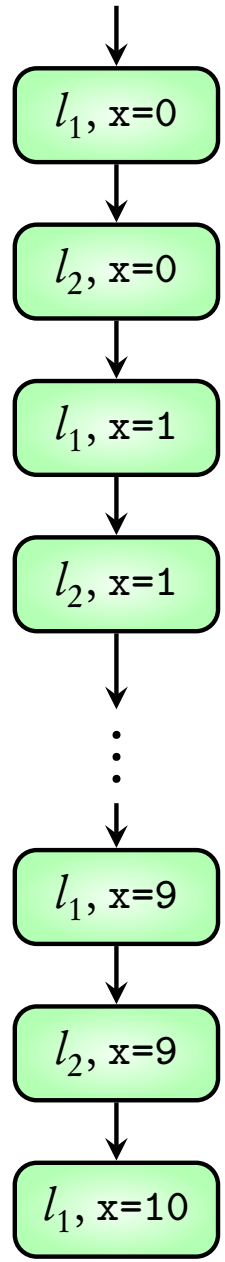



```
MODULE main
VAR
    request:  boolean;
    status:  {ready, busy}
ASSIGN
    init(status) := ready;
    next(status) := case
                    request :  busy;
                    TRUE : {ready,busy};
                          esac;
```

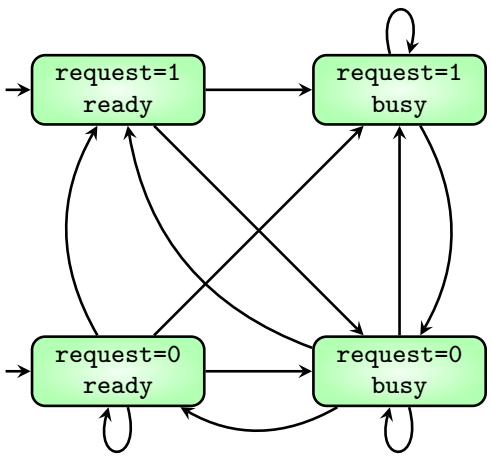
Coming next: checking **requirements** in NuSMV

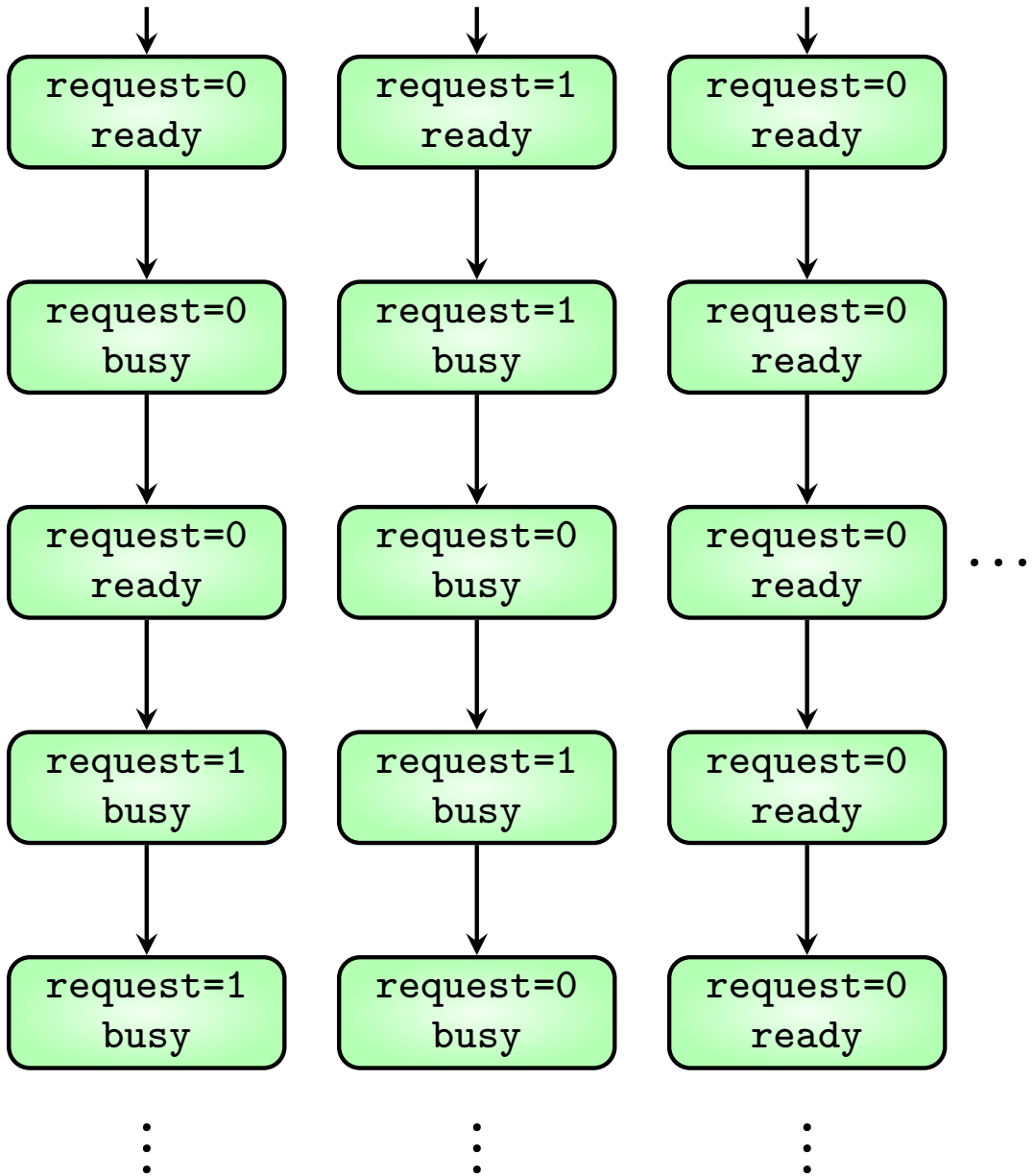


Executions



Executions


request=1
ready
request=1
busy
request=0
ready
request=0
busy


request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=0
busy
request=1
ready
request=1
busy
request=0
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready

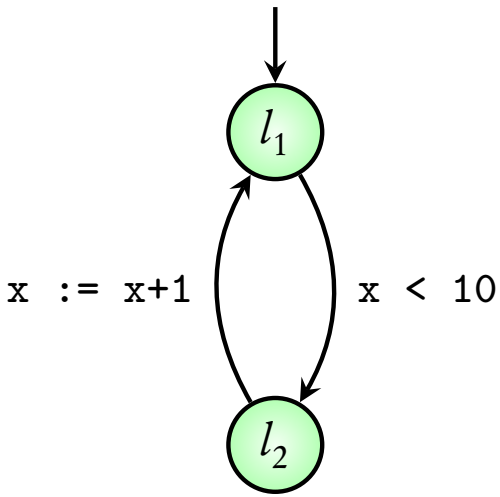
Transition system **satisfies a requirement**

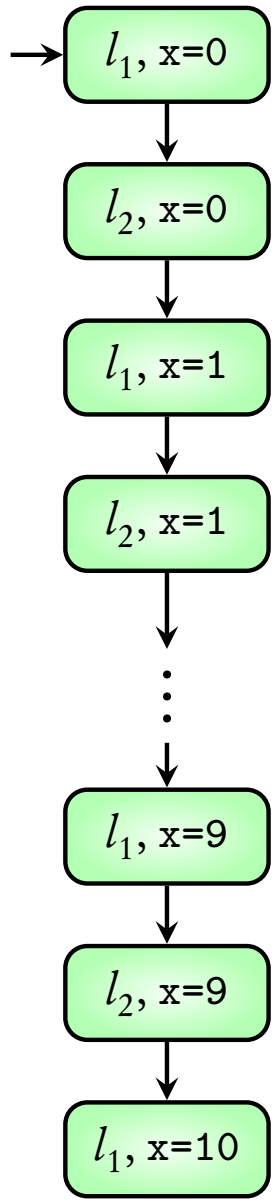
means

all its executions satisfy the requirement

Requirement type 1: G

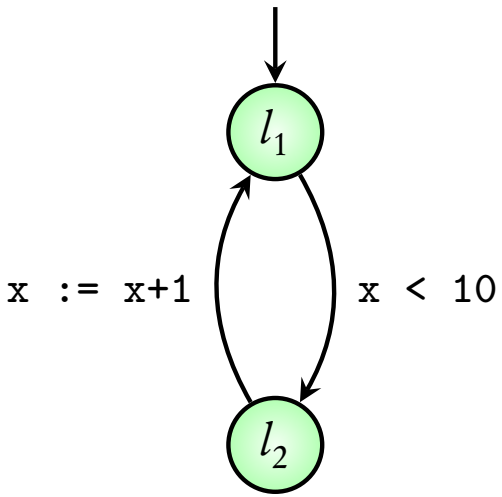
Requirement type 1: G





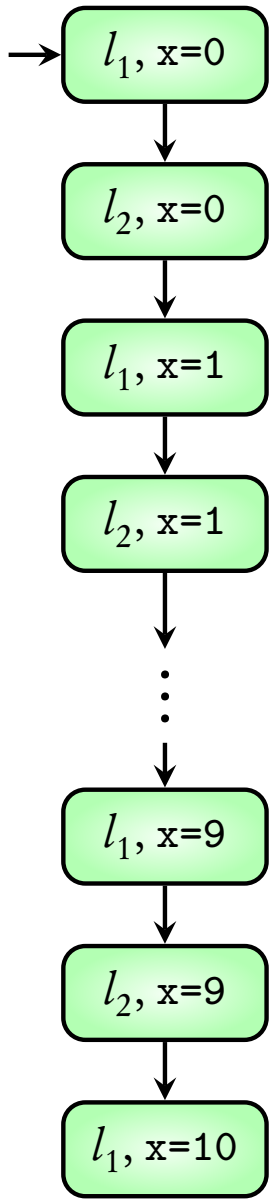
G (x >= 0)

Requirement type 1: G



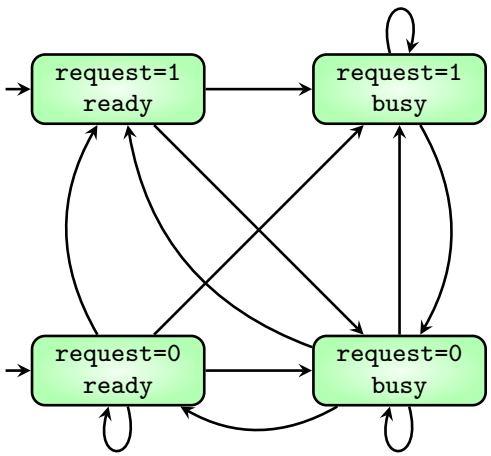
TS of above PG with initial value x=0

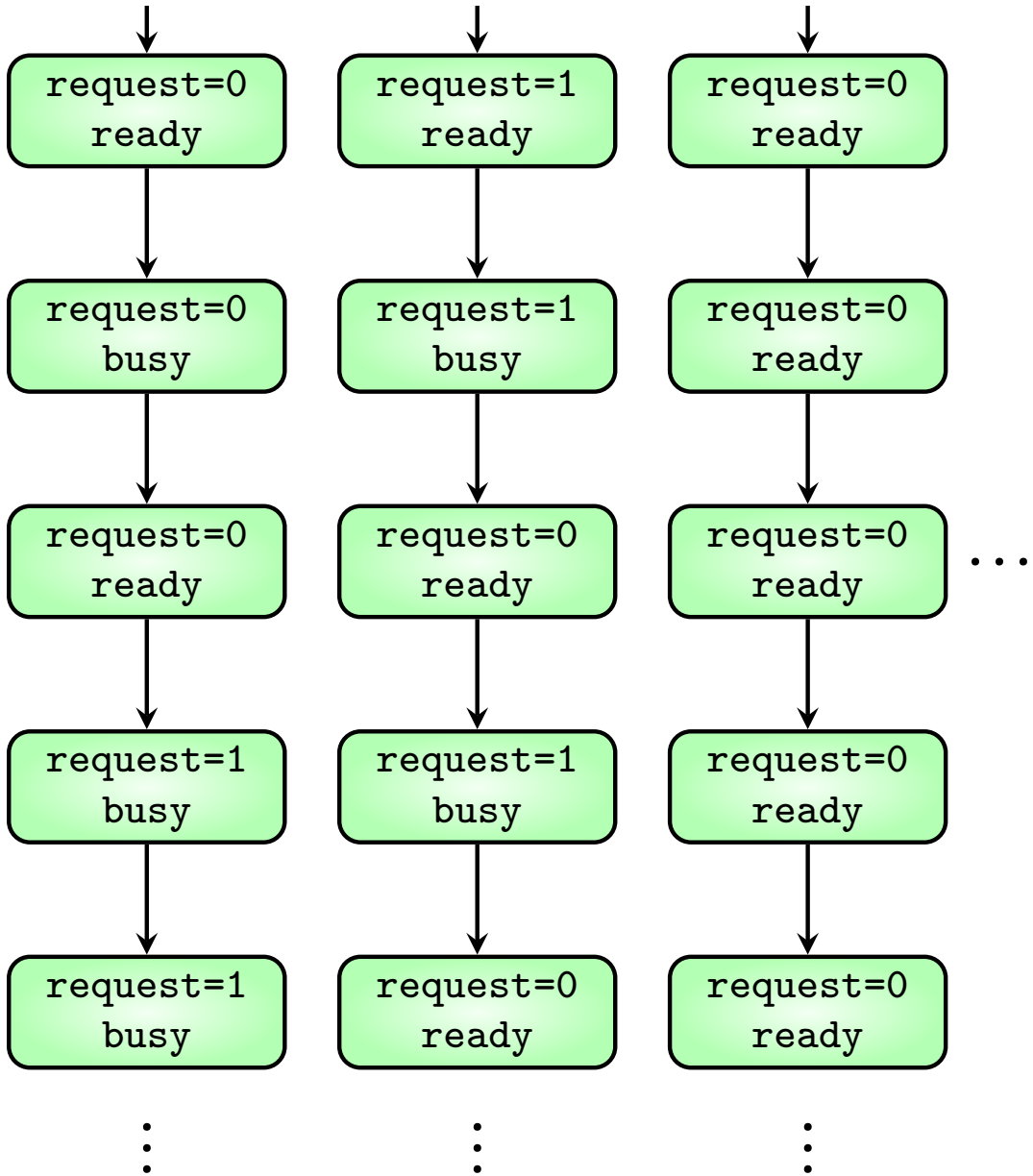
satisfies G (x >= 0)



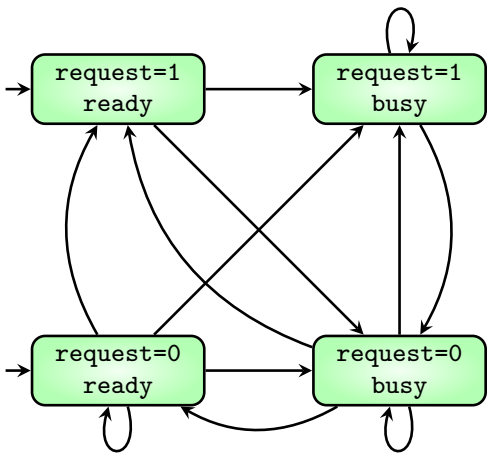
G (x >= 0)

G (request=0)

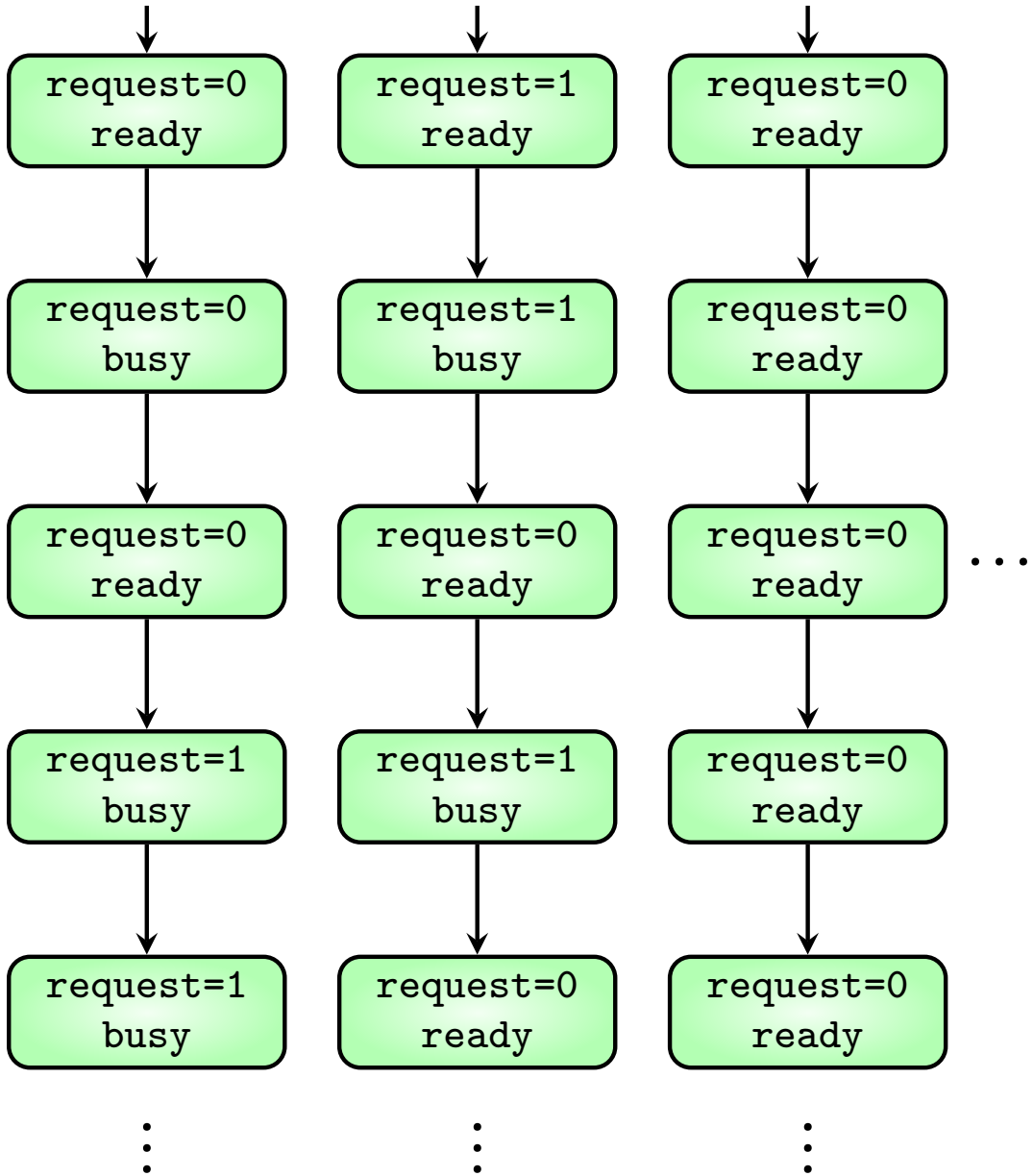

request=1
ready
request=1
busy
request=0
ready
request=0
busy


request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=1
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
...

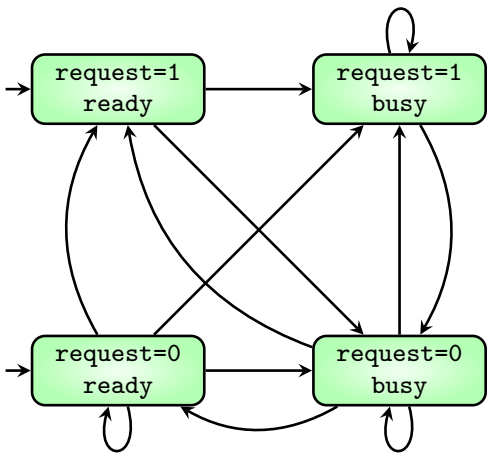
G (request=0)

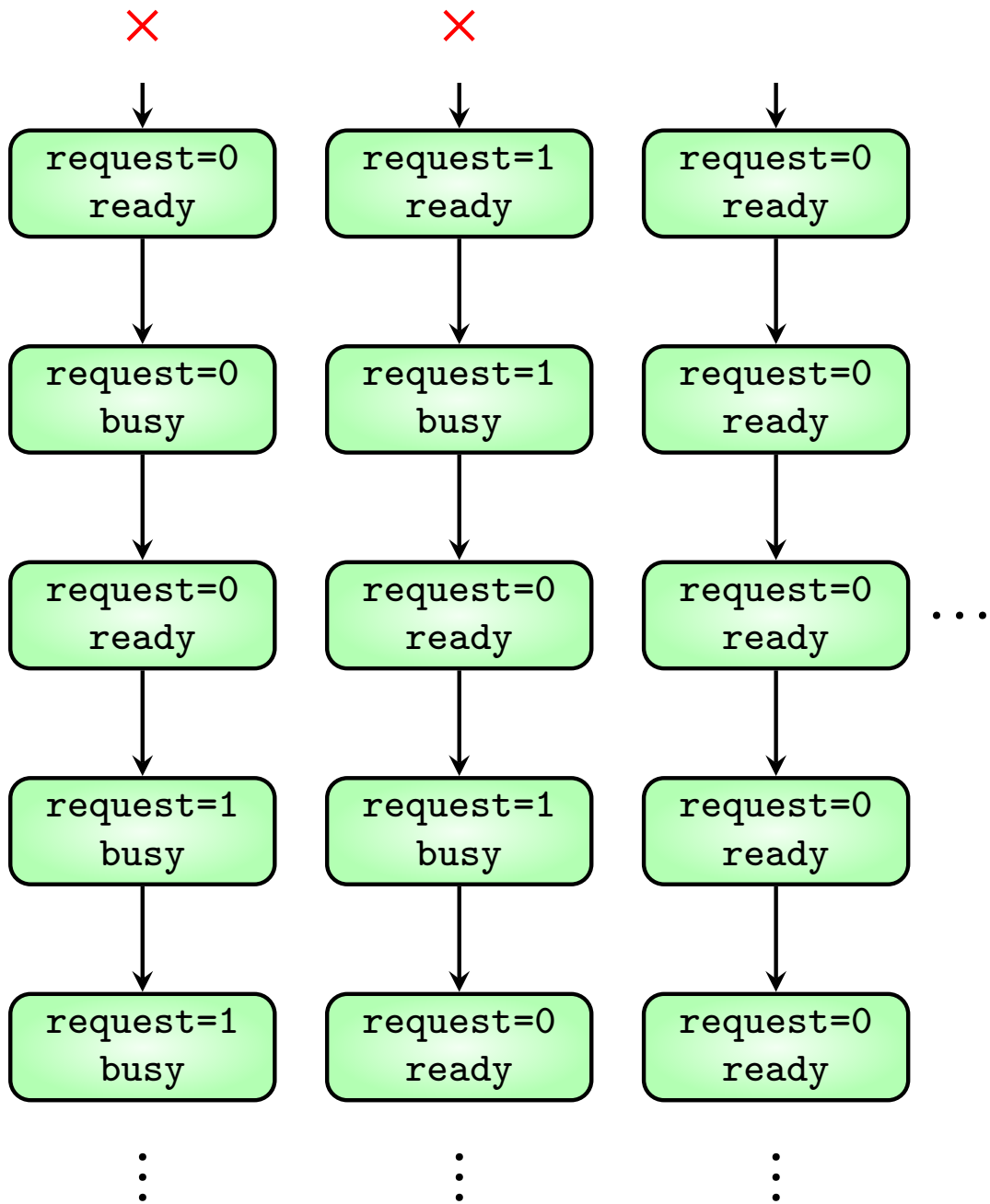

request=1
ready
request=1
busy
request=0
ready
request=0
busy

×

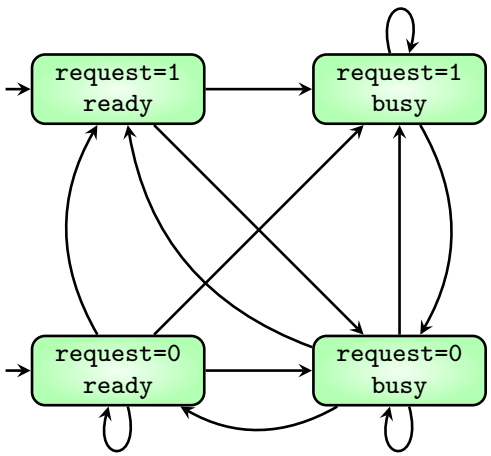

request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=1
ready
request=1
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready

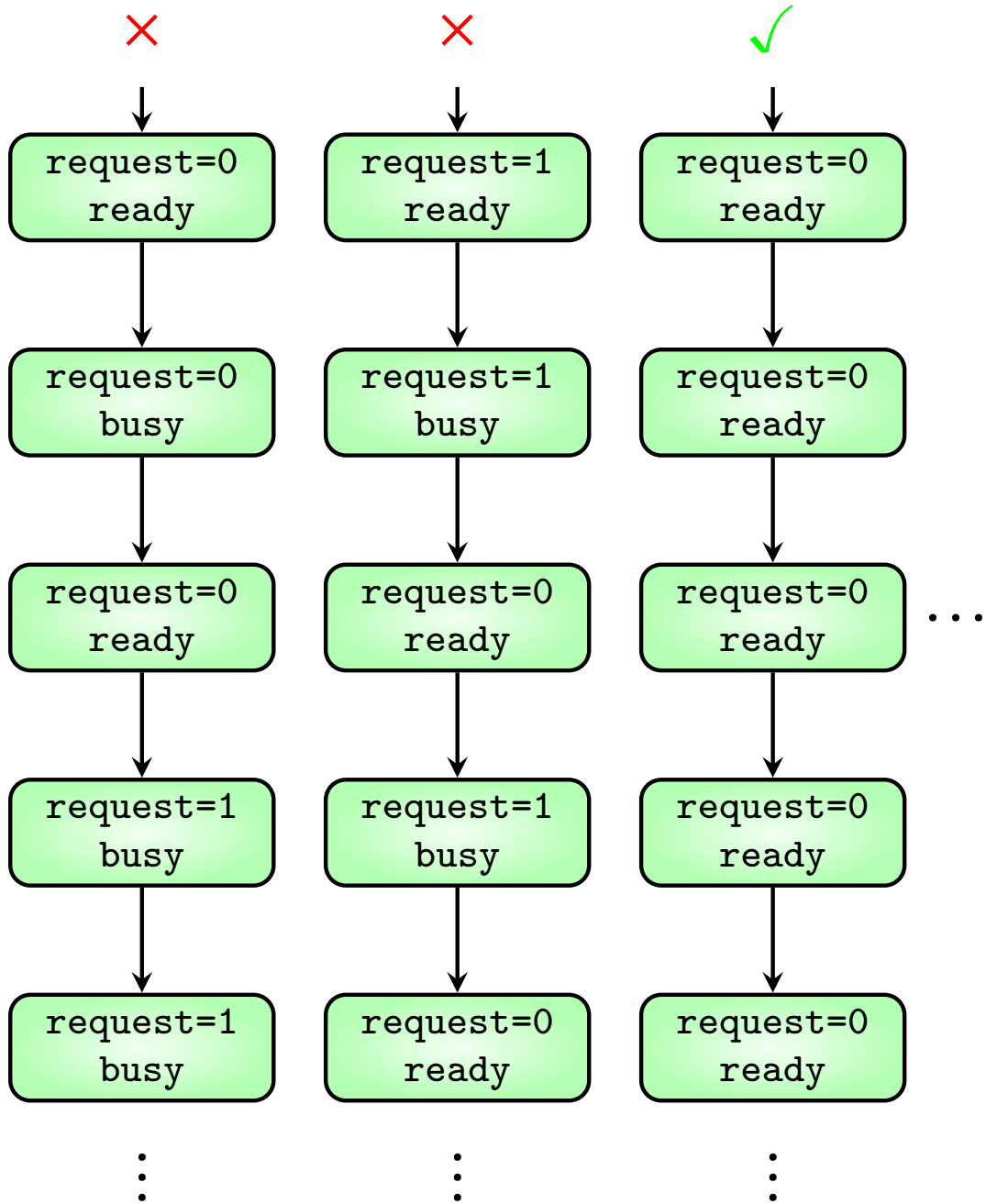
G (request=0)


request=1
ready
request=1
busy
request=0
ready
request=0
busy

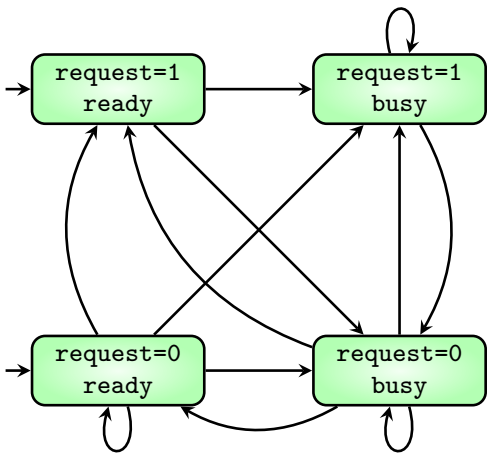

request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=1
ready
request=1
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready

G (request=0)


request=1
ready
request=1
busy
request=0
ready
request=0
busy

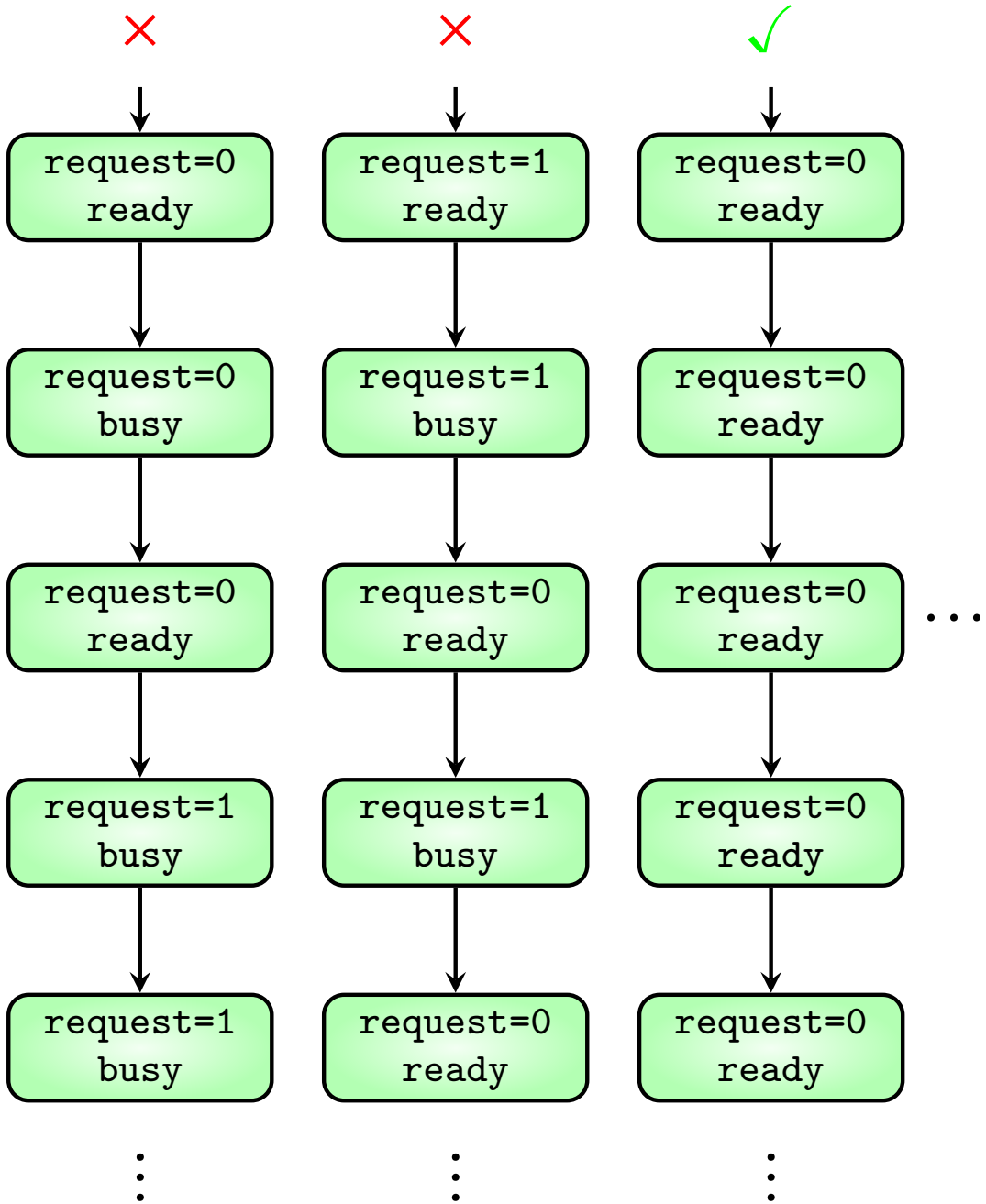

request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=1
busy
request=1
ready
request=1
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready

G (request=0)



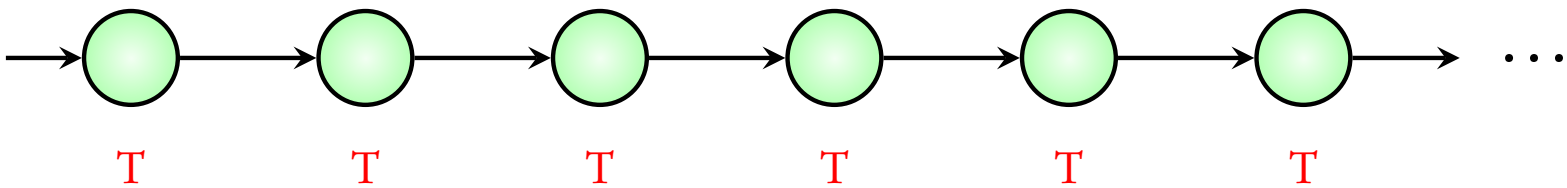
TS does not satisfy

G (request=0)



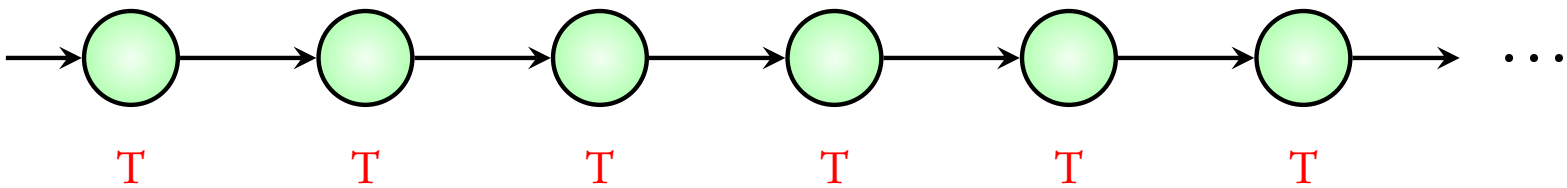
Execution **satisfies** G `(expr)` if

`expr` evaluates to T in **all its states**



Execution **satisfies** G `(expr)` if

`expr` evaluates to T in **all its states**



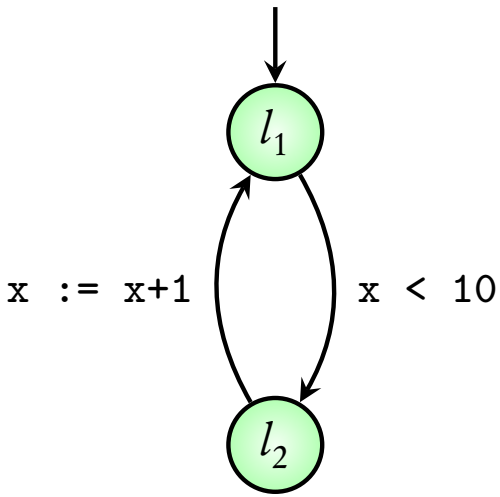
Transition system **satisfies** G `(expr)` if

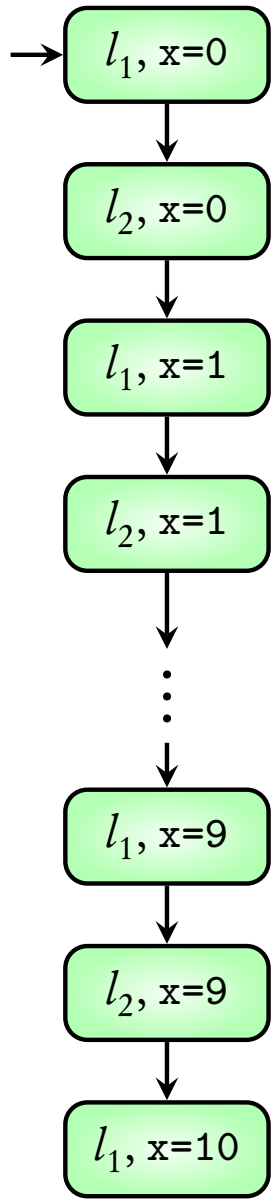
all its executions satisfy G `(expr)`

Checking the G requirement: NuSMV demo

Requirement type 2: F

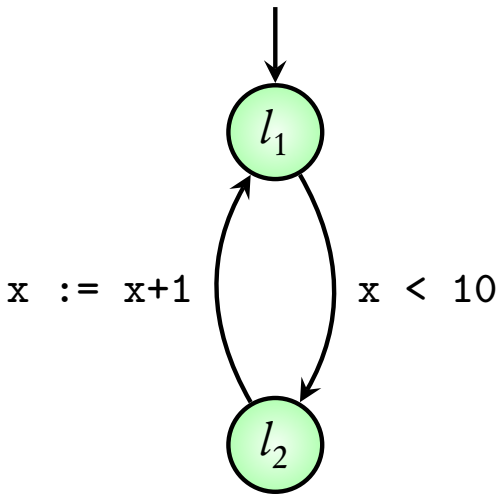
Requirement type 2: F





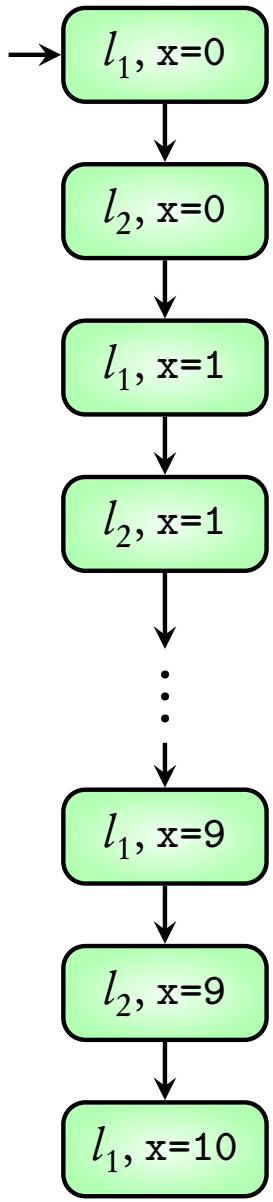
F (x >= 5)

Requirement type 2: F



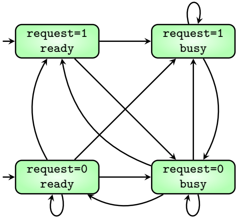
TS of above PG with initial value x=0

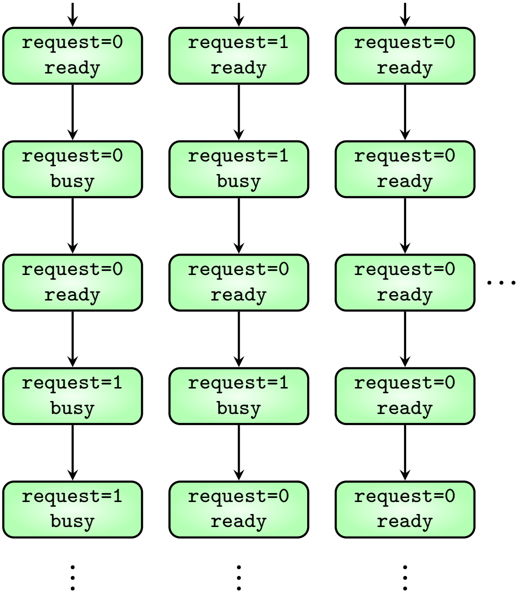
satisfies F (x >= 5)



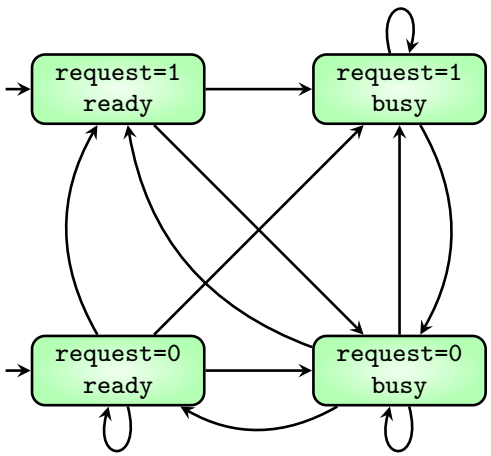
F (x >= 5)

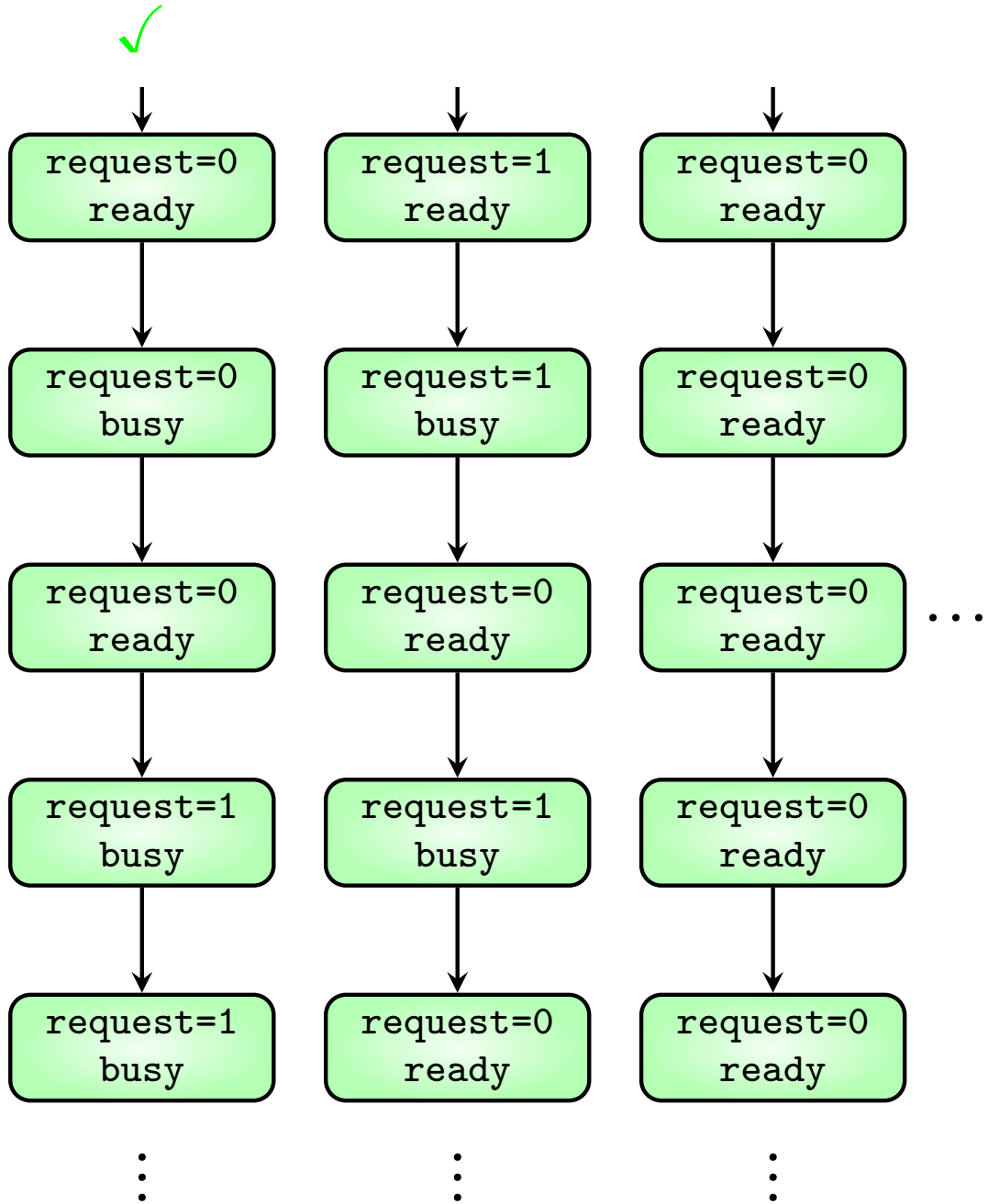
F (request=1)


request=1
ready
request=1
busy
request=0
ready
request=0
busy

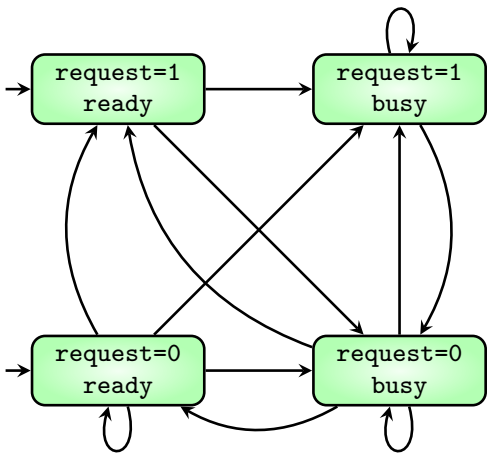

request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=1
ready
request=1
busy
request=0
ready
request=1
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
...

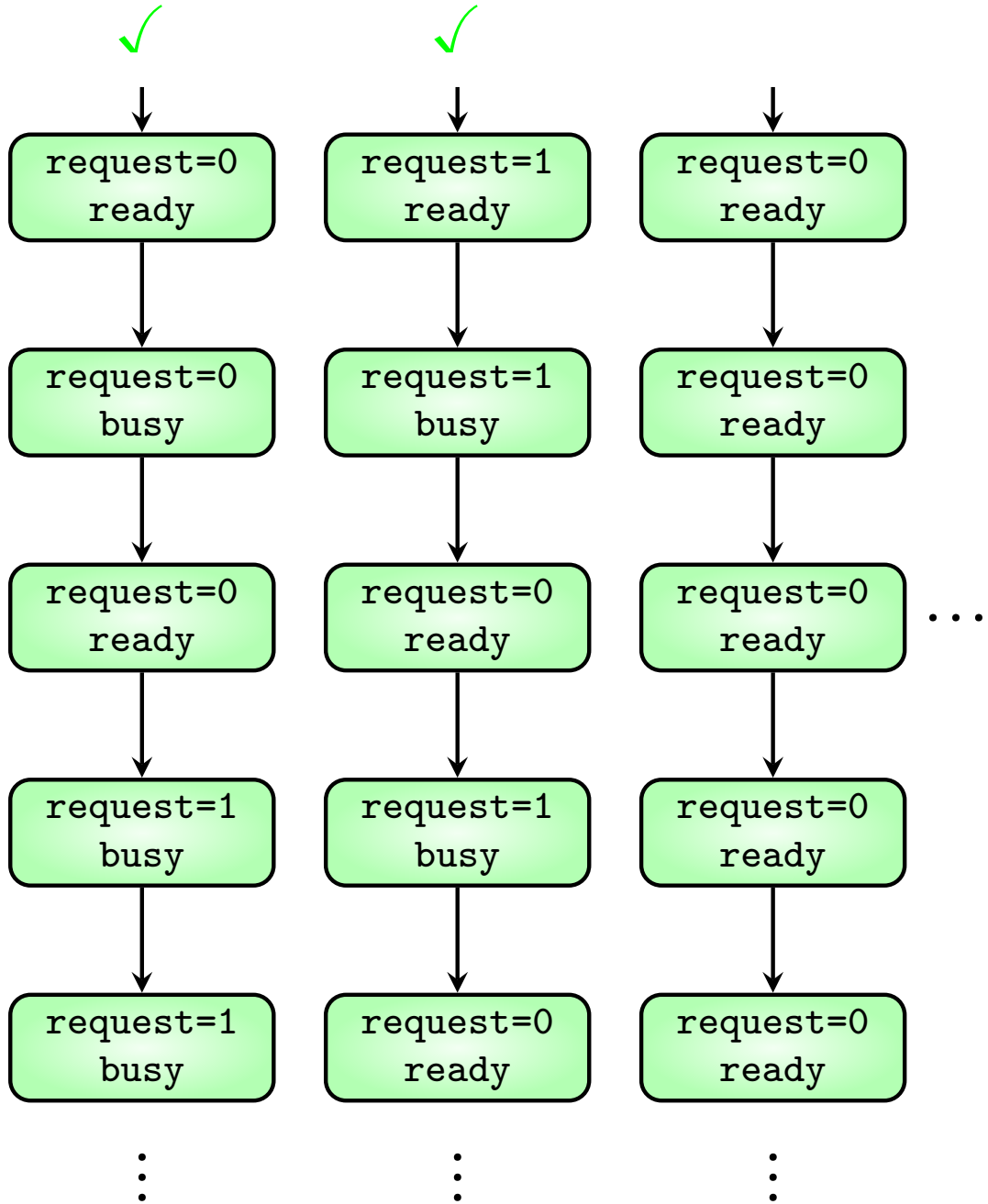
F (request=1)


request=1
ready
request=1
busy
request=0
ready
request=0
busy

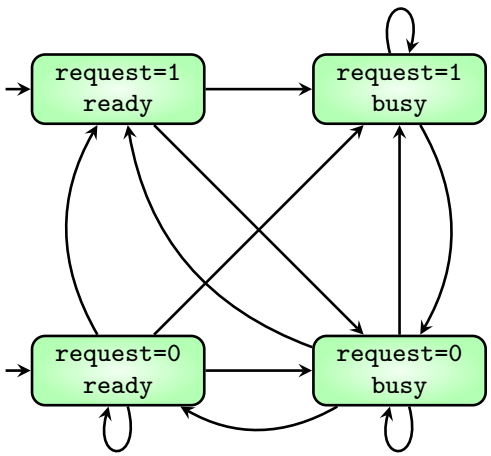

request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=1
busy
request=1
ready
request=1
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready

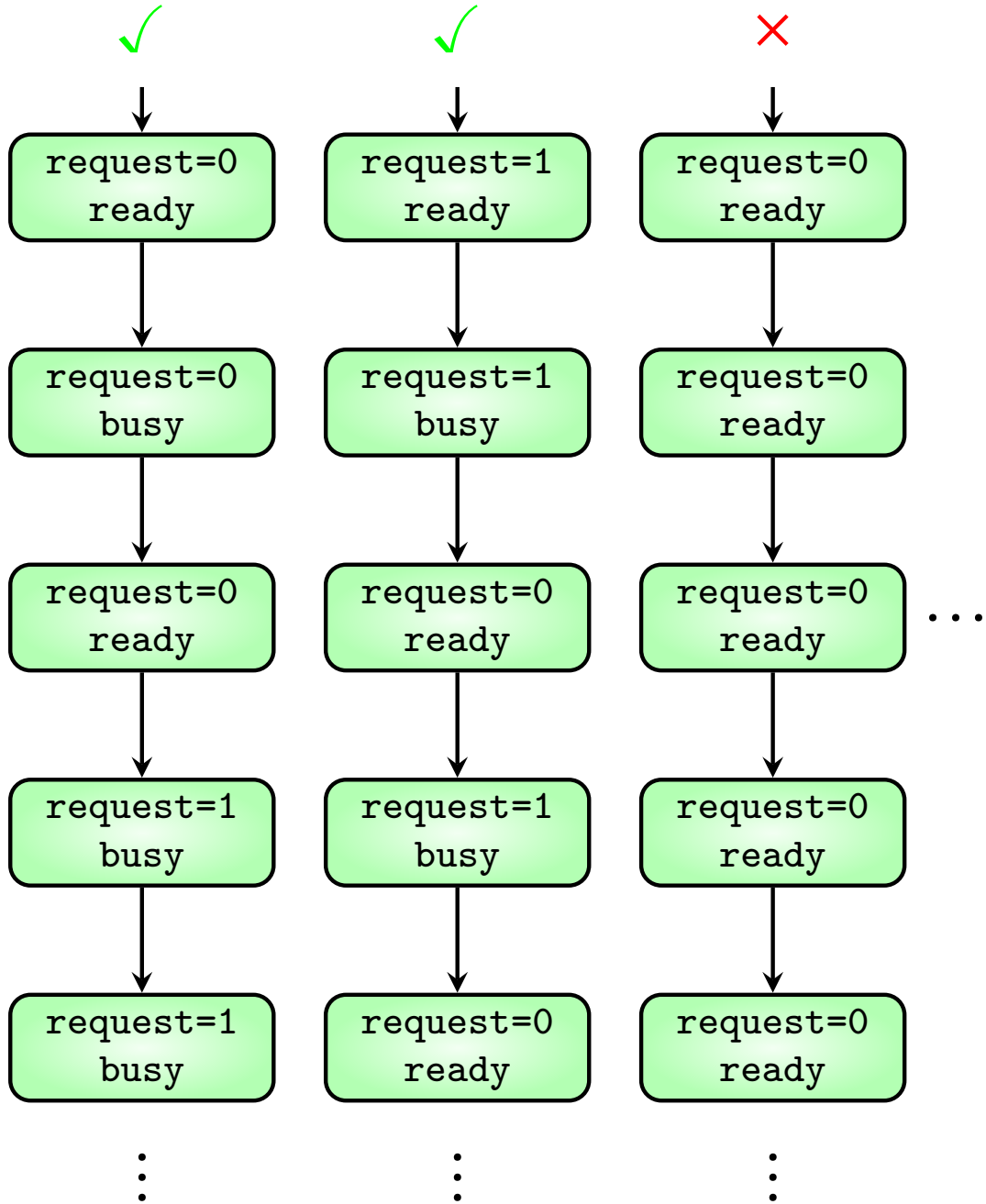
F (request=1)


request=1
ready
request=1
busy
request=0
ready
request=0
busy

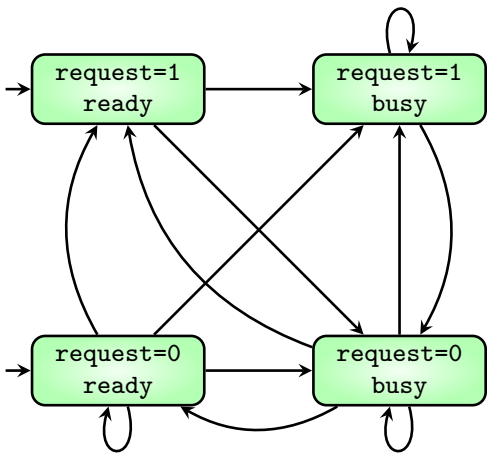

request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=1
busy
request=1
ready
request=1
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready

F (request=1)


request=1
ready
request=1
busy
request=0
ready
request=0
busy

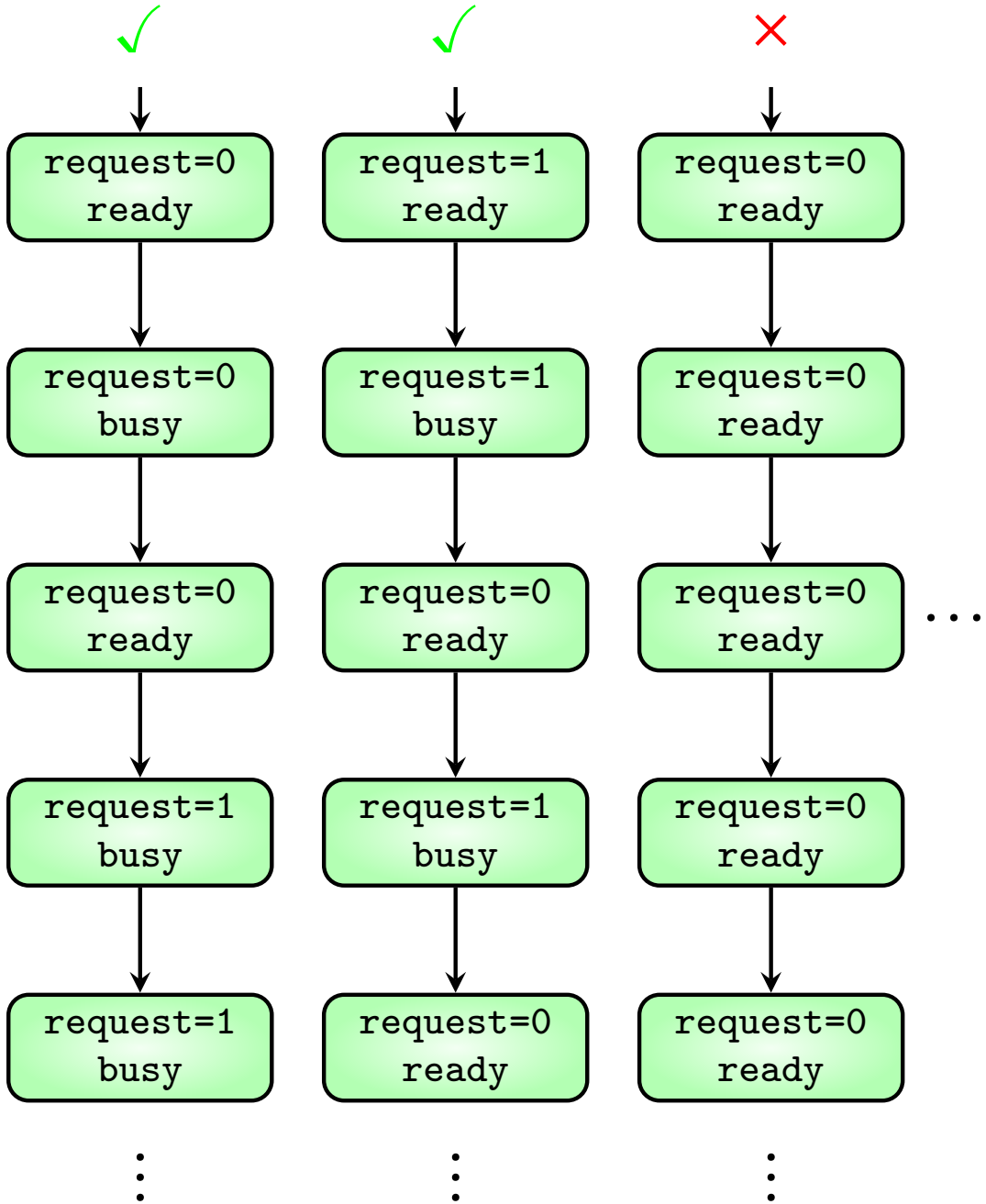

request=0
ready
request=0
busy
request=0
ready
request=1
busy
request=1
busy
request=1
busy
request=1
ready
request=1
busy
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready
request=0
ready

F (request=1)



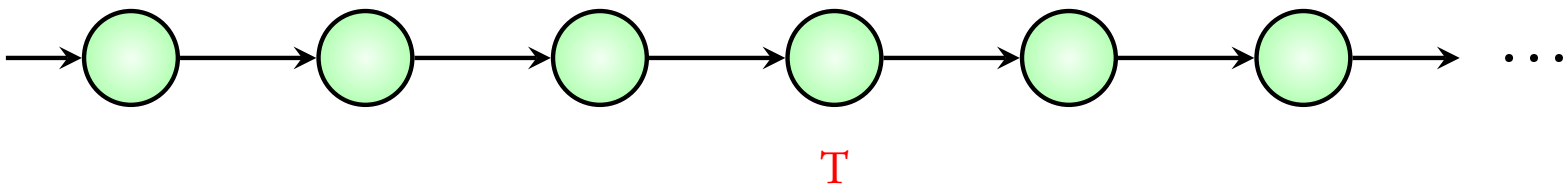
TS does not satisfy

F (request=1)



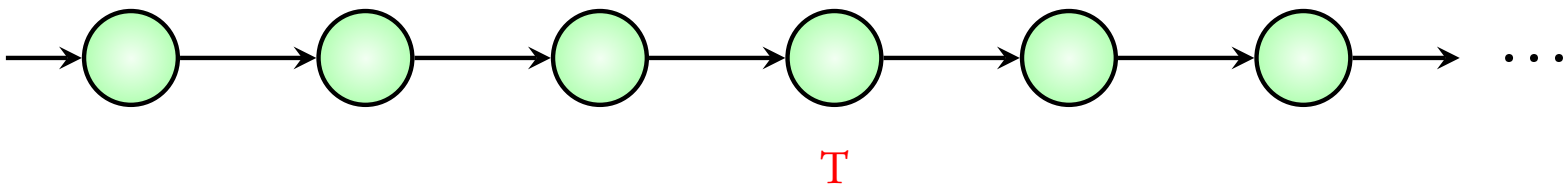
Execution **satisfies** F `(expr)` if

`expr` evaluates to T in **one of its states**



Execution **satisfies** `F (expr)` if

`expr` evaluates to T in **one of its states**



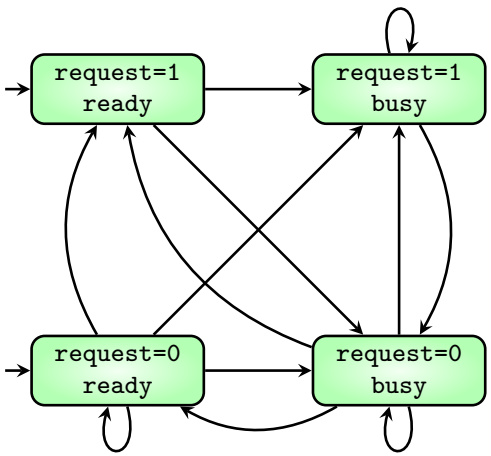
Transition system **satisfies** `F (expr)` if

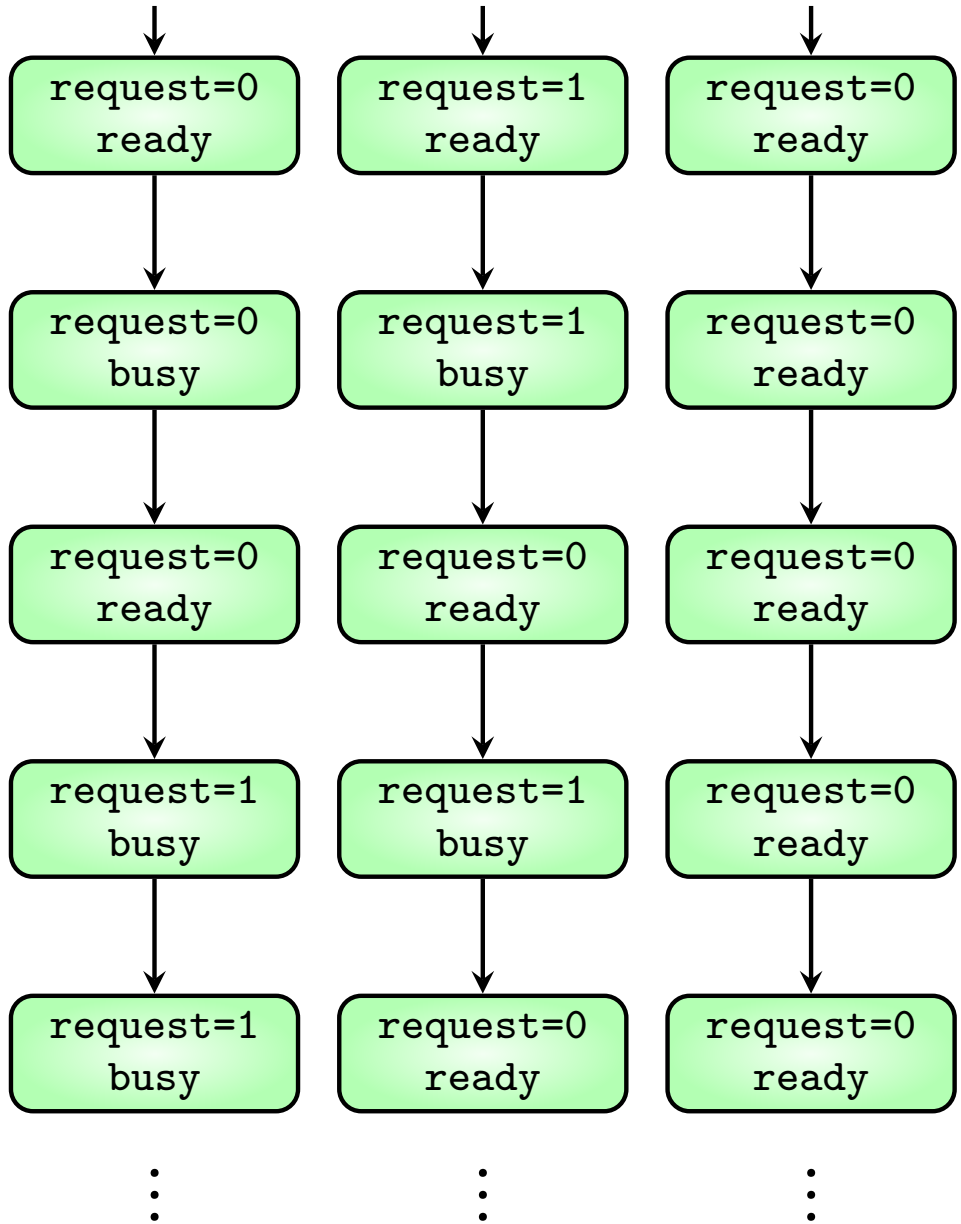
all its executions satisfy `F (expr)`

Checking the **F** requirement: **NuSMV demo**

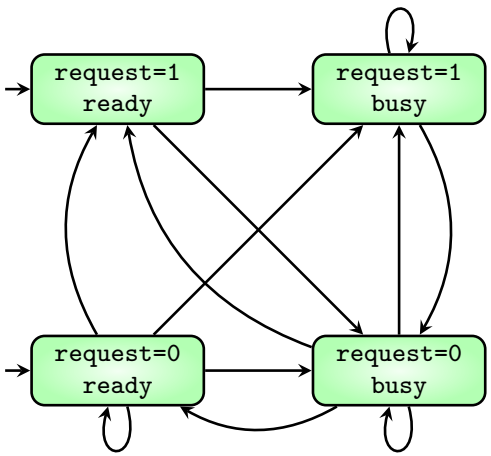
Coming next: Combining G and F

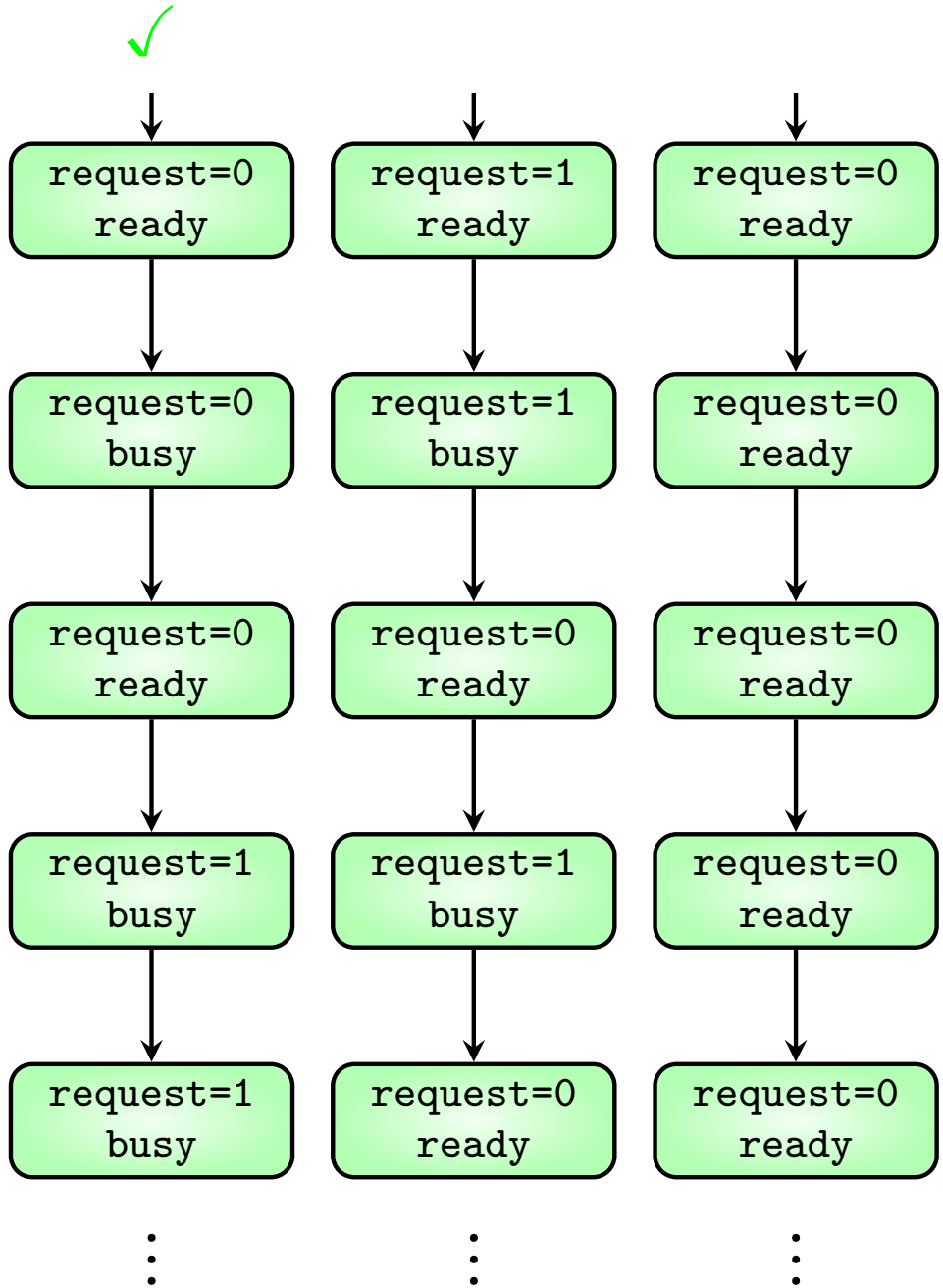
G (request=1 => F status=busy)



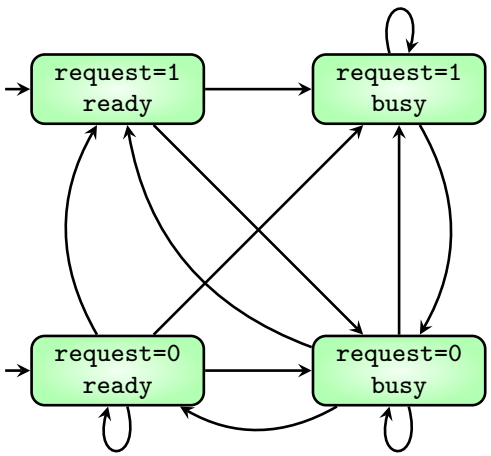


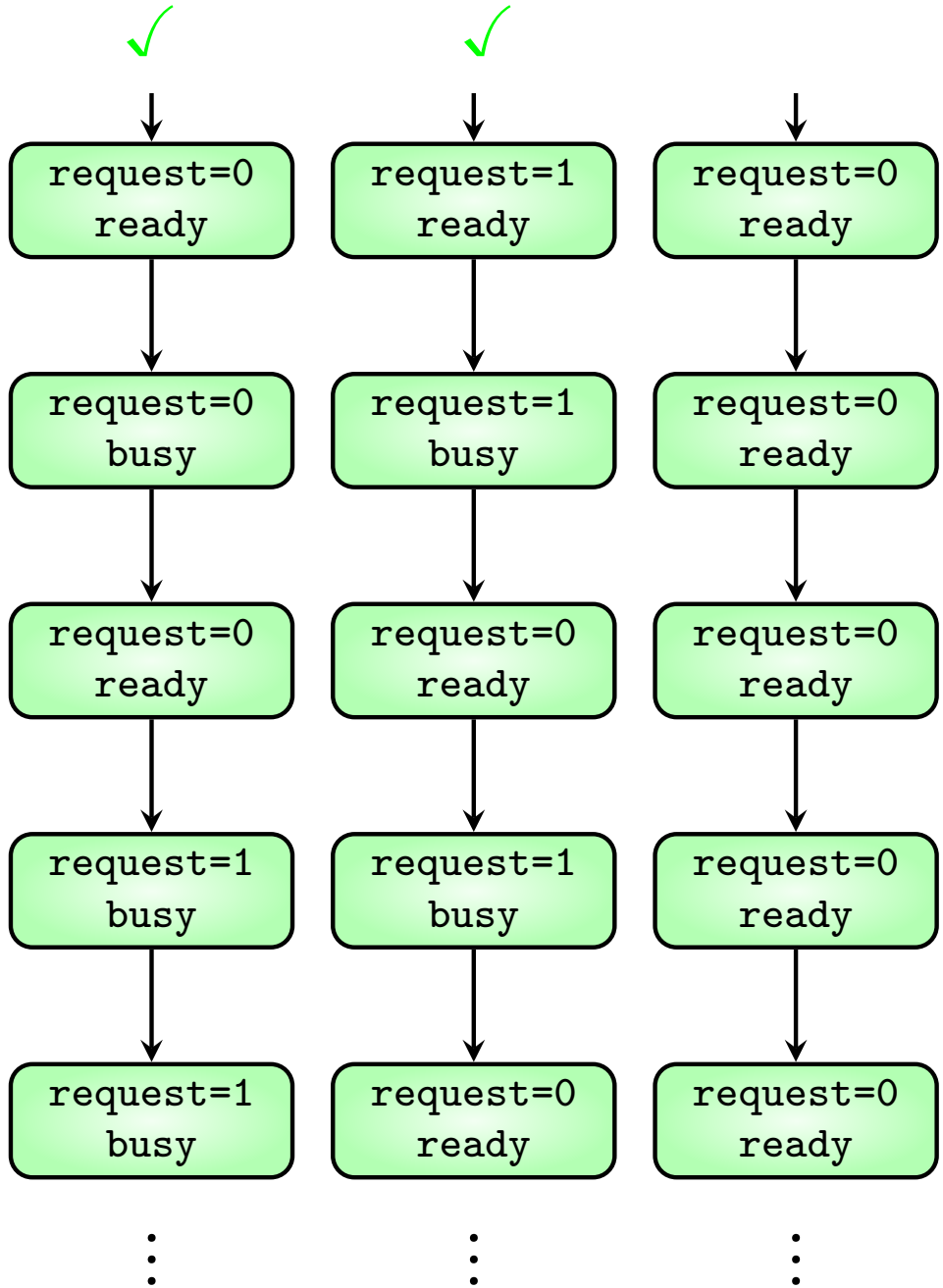
G (request=1 => F status=busy)



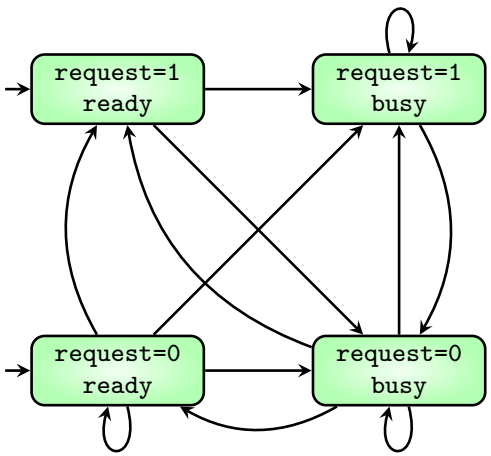


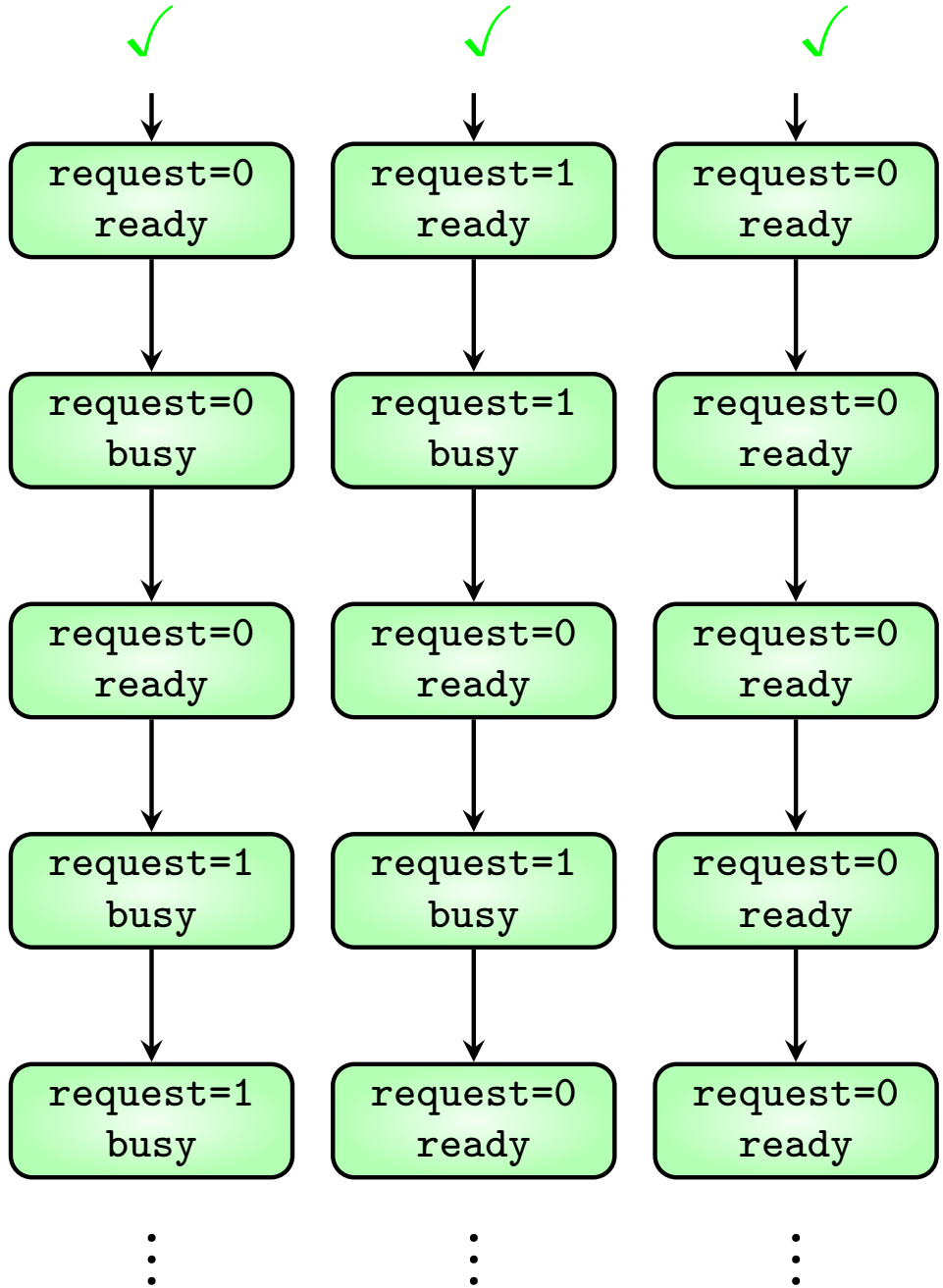
G (request=1 => F status=busy)



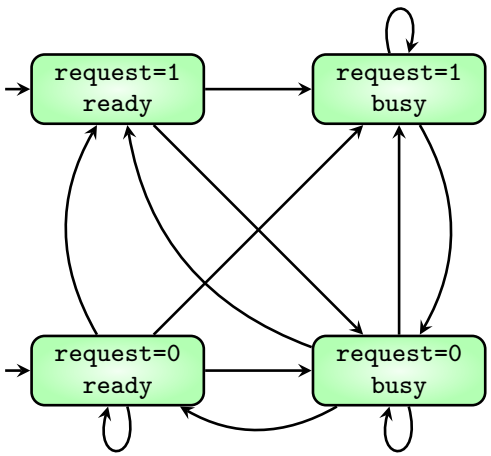


G (request=1 => F status=busy)



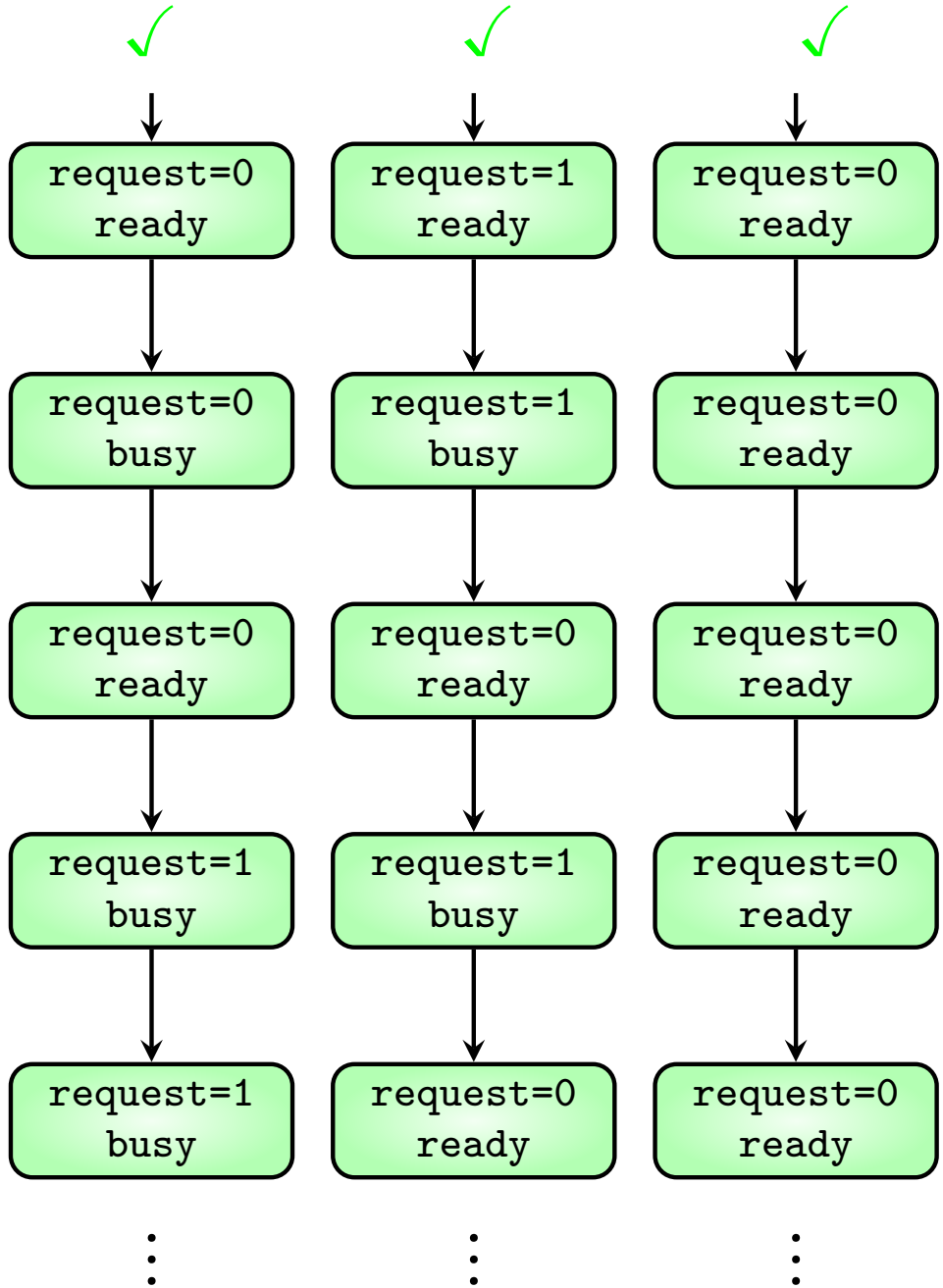


G (request=1 => F status=busy)



TS satisfies

G (request => F (status=busy))



Summary

Using NuSMV

Format for writing models

G and F requirements