

Exemplu Model Checking

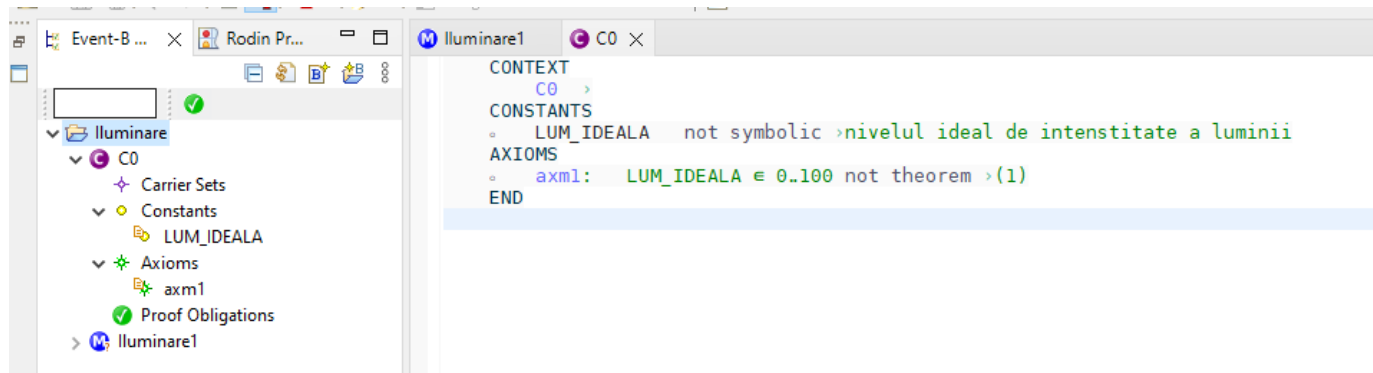
DEMO:

Se dorește modelarea sistemului de iluminare a unei camere.

Cerințele sistemului:

1. Sistemul are o luminozitate ideală (care este mereu aceeași) și se situează în intervalul de la 0 la 100.
2. Nivelul de iluminare poate fi controlat în mod controlat (maximum 10 unități) pentru a crește sau a scădea. Nivelul de iluminare nu poate fi niciodată negativ și nu poate depăși 100.
3. Iluminatul poate fi aprins sau stins. Acesta poate fi aprins doar atunci când nivelul de iluminare este mai mic decât luminozitatea ideală.
4. Dacă iluminatul este aprins, nivelul de iluminare nu poate scădea (poate doar să crească).
5. Odată ce iluminatul este aprins, acesta poate fi stins doar dacă nivelul de iluminare este mai mare sau egal cu luminozitatea ideală.

Context (C0):



Maşina (Iluminare1):

```
MACHINE
  Iluminare1 >
SEES
  ◦ C0
VARIABLES
  ◦ nivelLumina >nivelul intensitatii luminii
  ◦ statusLumina >TRUE = deschis sau FALSE = inchis
INVARIANTS
  ◦ inv1: nivelLumina ∈ 0..100 not theorem >(2)
  ◦ inv2: statusLumina ∈ BOOL not theorem >(3)
EVENTS
  ◦ INITIALISATION: not extended ordinary >
    THEN
      ◦ act1: nivelLumina = 0 >
      ◦ act2: statusLumina = FALSE >starea initiala, initial sistemul este inchis
    END
  ◦ aprindeLumina: not extended ordinary >
    WHERE
      ◦ grd1: nivelLumina < LUM_IDEALA not theorem >(5)
      ◦ grd2: statusLumina = FALSE not theorem >(5)
    THEN
      ◦ act1: statusLumina = TRUE >
    END
  ◦ inchideLumina: not extended ordinary >
    WHERE
      ◦ grd1: nivelLumina ≥ LUM_IDEALA not theorem >(3)
      ◦ grd2: statusLumina = TRUE not theorem >(3)
    THEN
      ◦ act1: statusLumina = FALSE >
    END
  ◦ cresteLum: not extended ordinary >
    ANY
      ◦ n >
    WHERE
      ◦ grd1: n ∈ 1..10 theorem >(2)
      ◦ grd2: nivelLumina + n ≤ 100 not theorem >(2)
      ◦ grd3: statusLumina = TRUE not theorem >(4)
    THEN
      ◦ act1: nivelLumina = nivelLumina + n >
    END
  ◦ scadeLum: not extended ordinary >
    ANY
      ◦ n >
    WHERE
      ◦ grd1: n ∈ 1..10 not theorem >(2)
      ◦ grd2: nivelLumina - n ≥ 0 not theorem >(2)
    THEN
      ◦ act1: nivelLumina = nivelLumina - n >
    END
END
```



Model Checking finished

OK

<< Details

=====

Coverage statistics:

Total Number of States:69

Total Number of Transitions:185

Node Statistics:

deadlocked:0

invariant_violated:0

explored_but_not_all_transitions_computed:1

live:40

open:29

invariant_not_checked:29

total:69

Operations Statistics:

SETUP_CONSTANTS:4

INITIALISATION:4

includLumina:18

crestLum:84

scadeLum:74

aprindeLumina:1

Uncovered Operations:

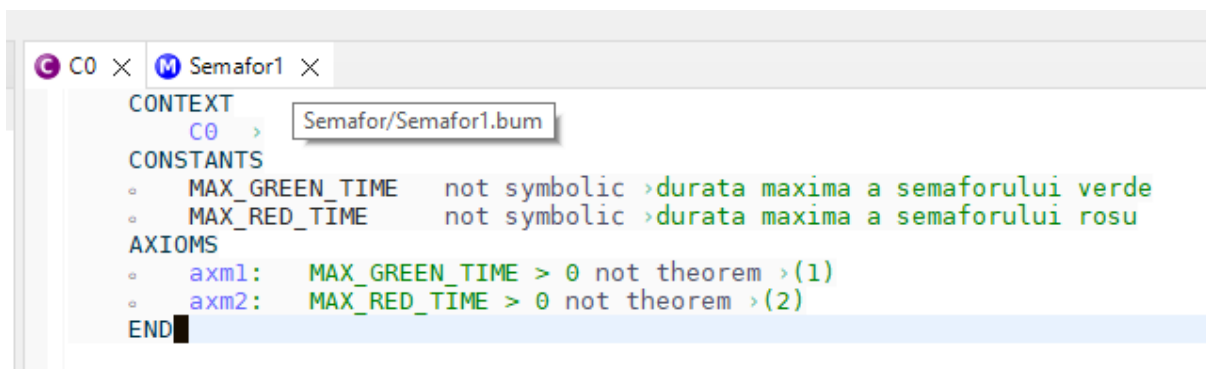
Demo Semafor:

Se dorește modelarea sistem de semaforizare al unei intersecții.

Cerințele sistemului:

1. Durata maximă a semaforului verde (MAX_GREEN_TIME) trebuie să fie mai mare decât 0.
2. Durata maximă a semaforului roșu (MAX_RED_TIME) trebuie să fie mai mare decât 0.
3. Timpul rămas pentru semaforul verde (greenTime) trebuie să fie întotdeauna pozitiv.
4. Timpul rămas pentru semaforul roșu (redTime) trebuie să fie întotdeauna pozitiv.
5. Schimbarea direcției semaforului (evenimentul "switchDirection") poate avea loc doar atunci când timpul rămas pentru ambele semafoare este 0.
6. Evenimentul "decreaseTime" trebuie să scadă timpul rămas pentru ambele semafoare (greenTime și redTime) atunci când acestea sunt mai mari decât 0.

Context:



```
CONTEXT
  C0 > Semafor/Semafor1.bum
CONSTANTS
  ◦ MAX_GREEN_TIME    not symbolic >durata maxima a semaforului verde
  ◦ MAX_RED_TIME      not symbolic >durata maxima a semaforului rosu
AXIOMS
  ◦ axm1:  MAX_GREEN_TIME > 0 not theorem >(1)
  ◦ axm2:  MAX_RED_TIME > 0 not theorem >(2)
END
```

Maşina:

```
M Semafor1 X
MACHINE
  Semafor1 >
SEES
  C0
VARIABLES
  greenTime > timpul ramas pentru semaforul verde
  redTime > timpul ramas pentru semaforul rosu
  direction > verde pentru directia nord-sud, FALSE = verde pentru directia est-vest
INVARIANTS
  inv1: greenTime ≥ 0 not theorem >(3)
  inv2: redTime ≥ 0 not theorem >(4)
EVENTS
  ◦ INITIALISATION: not extended ordinary >
    THEN
      ◦ act1: greenTime = MAX_GREEN_TIME > initializare, semaforul verde este setat la durata maxima
      ◦ act2: redTime = 0 > initializare, semaforul rosu este setat la zero
      ◦ act3: direction = TRUE > initializare, directia nord-sud primeste semnal verde
    END
  ◦ switchDirection: not extended ordinary >
    WHERE
      ◦ grd1: greenTime = 0 not theorem >(5)
      ◦ grd2: redTime = 0 not theorem >(5)
    THEN
      ◦ act1: greenTime = MAX_GREEN_TIME > schimbarea directiei, se reseteaza timpul pentru semaforul verde
      ◦ act2: redTime = MAX_RED_TIME > schimbarea directiei, se reseteaza timpul pentru semaforul rosu
      ◦ act3: direction = ¬(direction) > schimbarea directiei, se inverseaza directia verde
    END
  ◦ decreaseTime: not extended ordinary >
    WHERE
      ◦ grd1: greenTime > 0 not theorem >(6)
      ◦ grd2: redTime > 0 not theorem >(6)
    THEN
      ◦ act1: greenTime = greenTime - 1 > scaderea timpului pentru semaforul verde
      ◦ act2: redTime = redTime - 1 > scaderea timpului pentru semaforul rosu
    END
```