CIS 721 - Real-Time Systems

Lecture 30: Times Tool

Mitch Neilsen

neilsen@ksu.edu

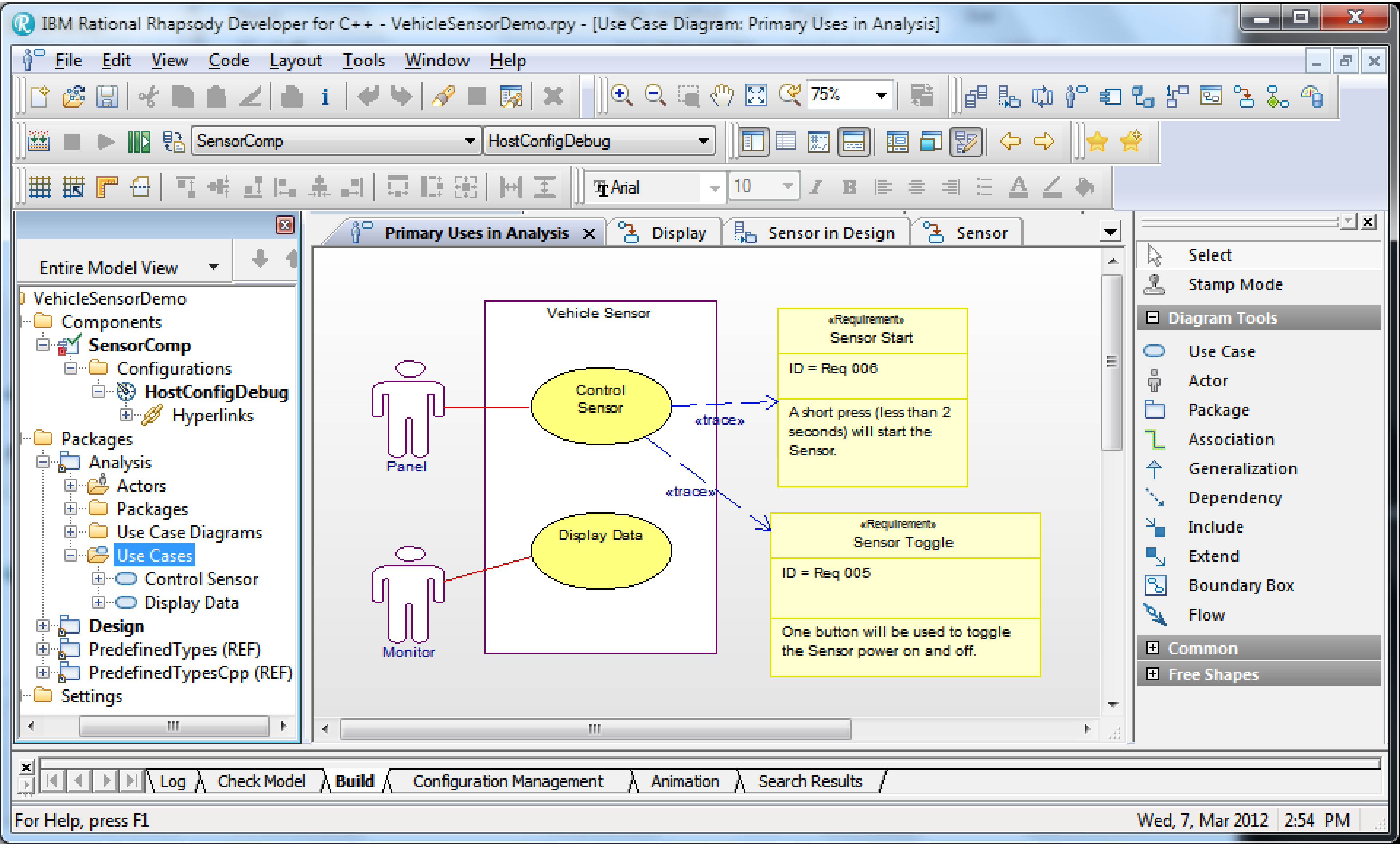
Outline

- **Development Tools**
  - **Rational Rose Real-Time (RoseRT)**
  - **IBM Rational Rhapsody**
  - **Times Tool**
    - **Lego Mindstorms RCX**

IBM Rational Rhapsody

- Successor to Rational Rose RT
- Acquired by IBM
- Targets most commercial RTOS
- Too heavy weight for PICs

Analysis Package



IBM Rational Rhapsody Developer for C++ - VehicleSensorDemo.rpy - [Use Case Diagram: Primary Uses in Analysis]

«Requirement» Sensor Start

ID = Req 006

A short press (less than 2 seconds) will start the Sensor.

«Requirement» Sensor Toggle

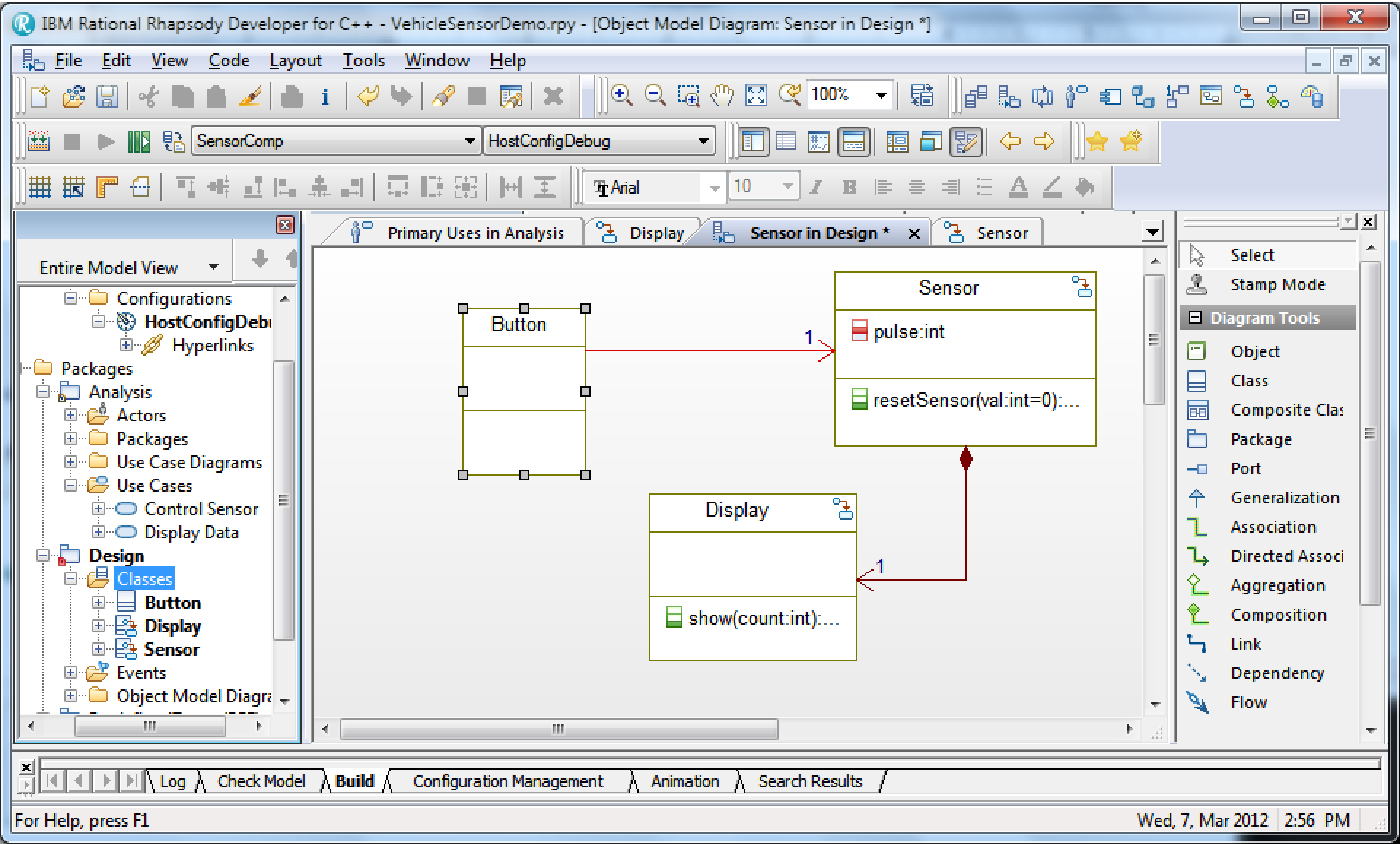
ID = Req 005

One button will be used to toggle the Sensor power on and off.

For Help, press F1

Wed, 7, Mar 2012 2:54 PM

Design Diagram - Class Diagram



IBM Rational Rhapsody Developer for C++ - VehicleSensorDemo.rpy - [Object Model Diagram: Sensor in Design *]

SensorComp

HostConfigDebug

100%

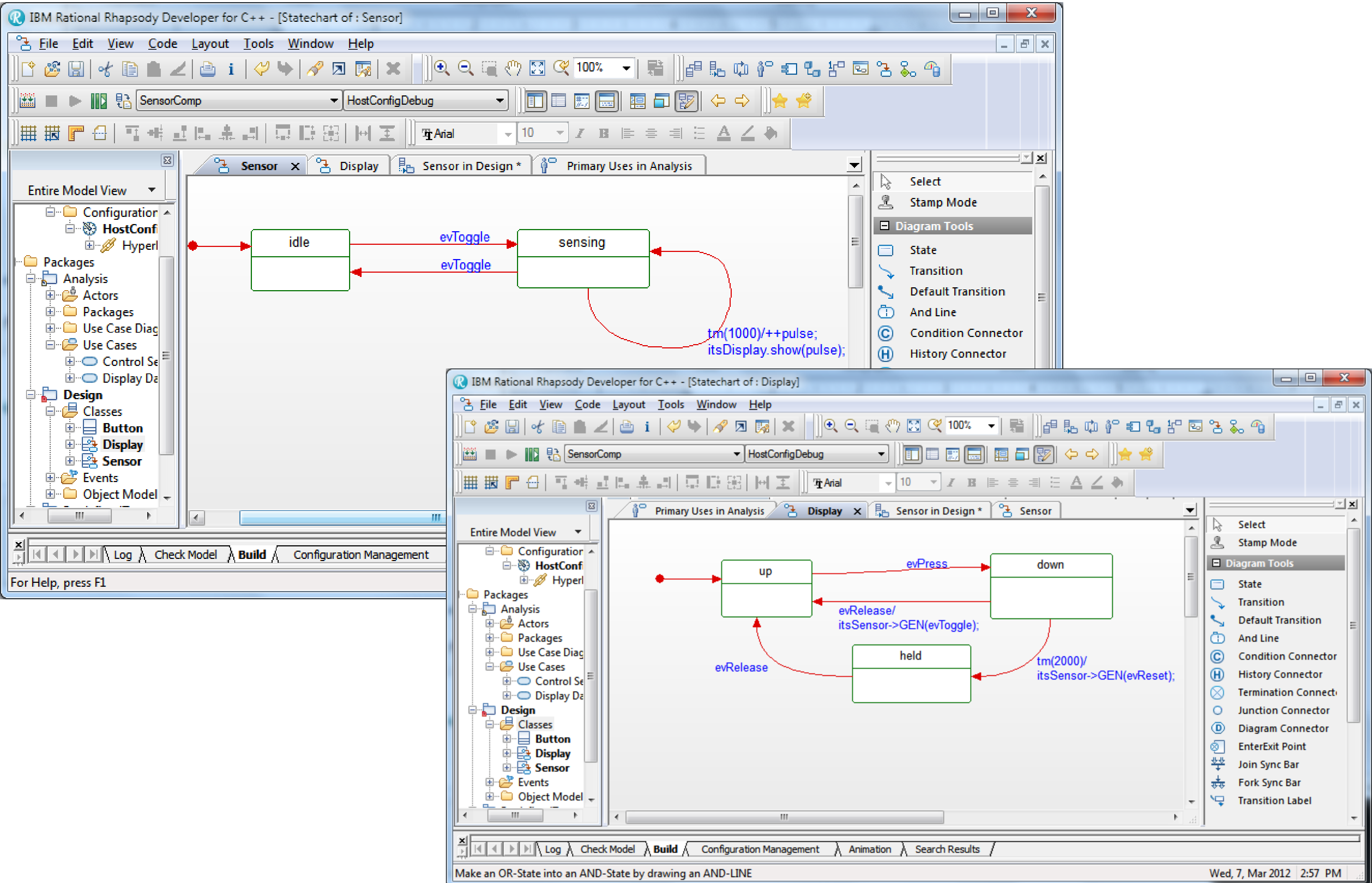
Arial

10

For Help, press F1

Wed, 7, Mar 2012 2:56 PM

Statecharts



IBM Rational Rhapsody Developer for C++ - [Statechart of : Sensor]

idle —evToggle→ sensing

sensing —evToggle→ idle

sensing: tm(1000)/++pulse; itsDisplay.show(pulse);

For Help, press F1

IBM Rational Rhapsody Developer for C++ - [Statechart of : Display]

up —evPress→ down

down —evRelease/ itsSensor->GEN(evToggle);→ up

down —tm(2000)/ itsSensor->GEN(evReset);→ held

held —evRelease→ up

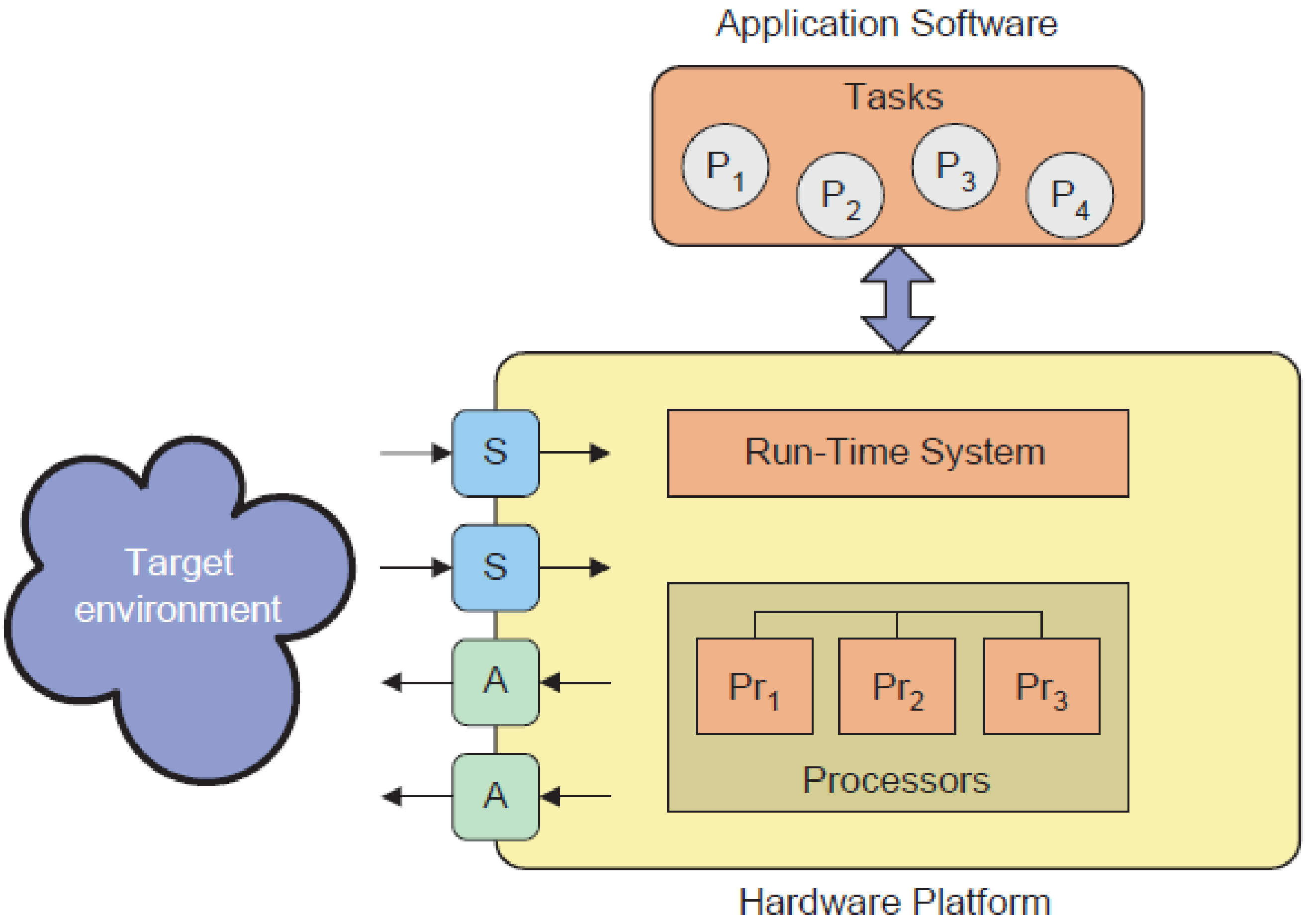
Make an OR-State into an AND-State by drawing an AND-LINE

Wed, 7, Mar 2012 2:57 PM

Times Tool

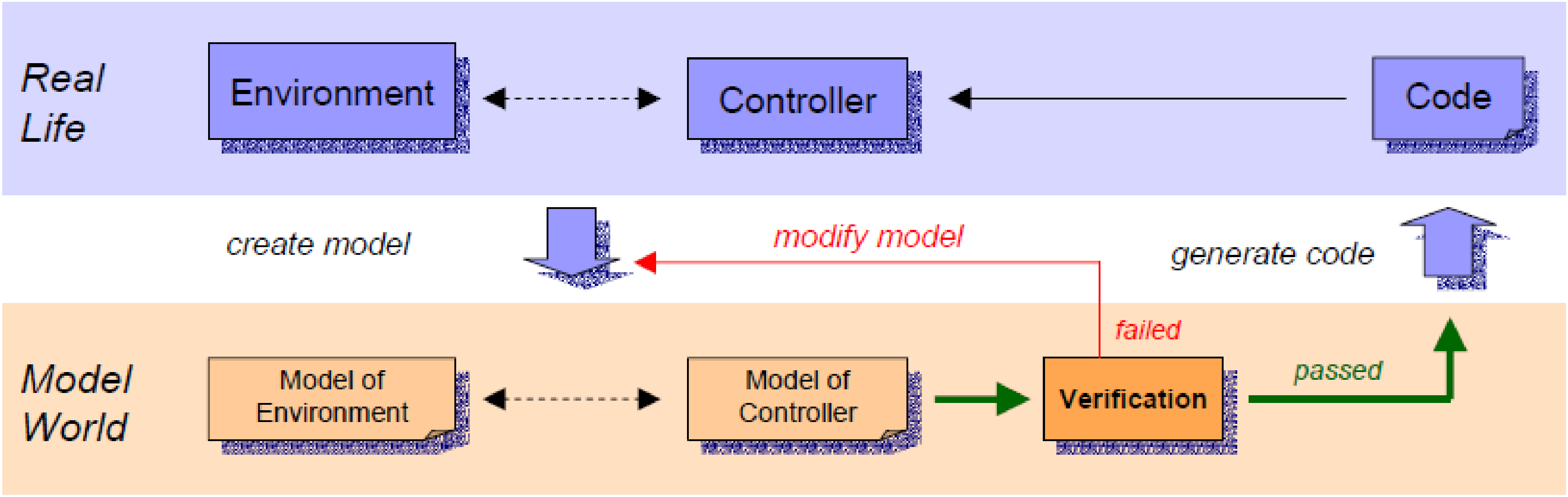
- Editor
- Simulator
- Schedulability Analysis
- Code Synthesis
- Verification (same as UPPAAL)

Real-time System Components



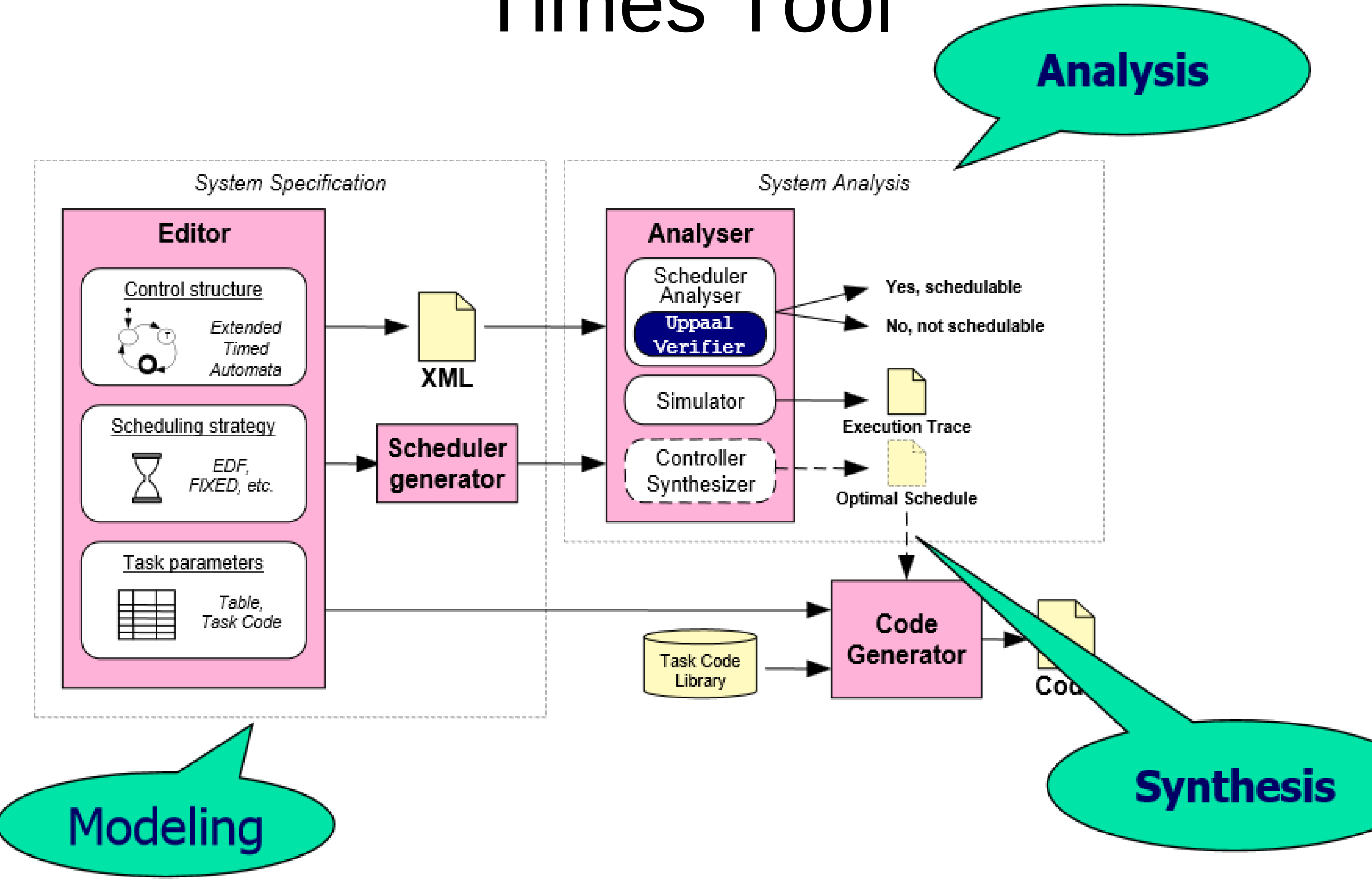
RTiS'03 (18.08.2003)

Modeling and Implementation of Real-time Embedded Systems

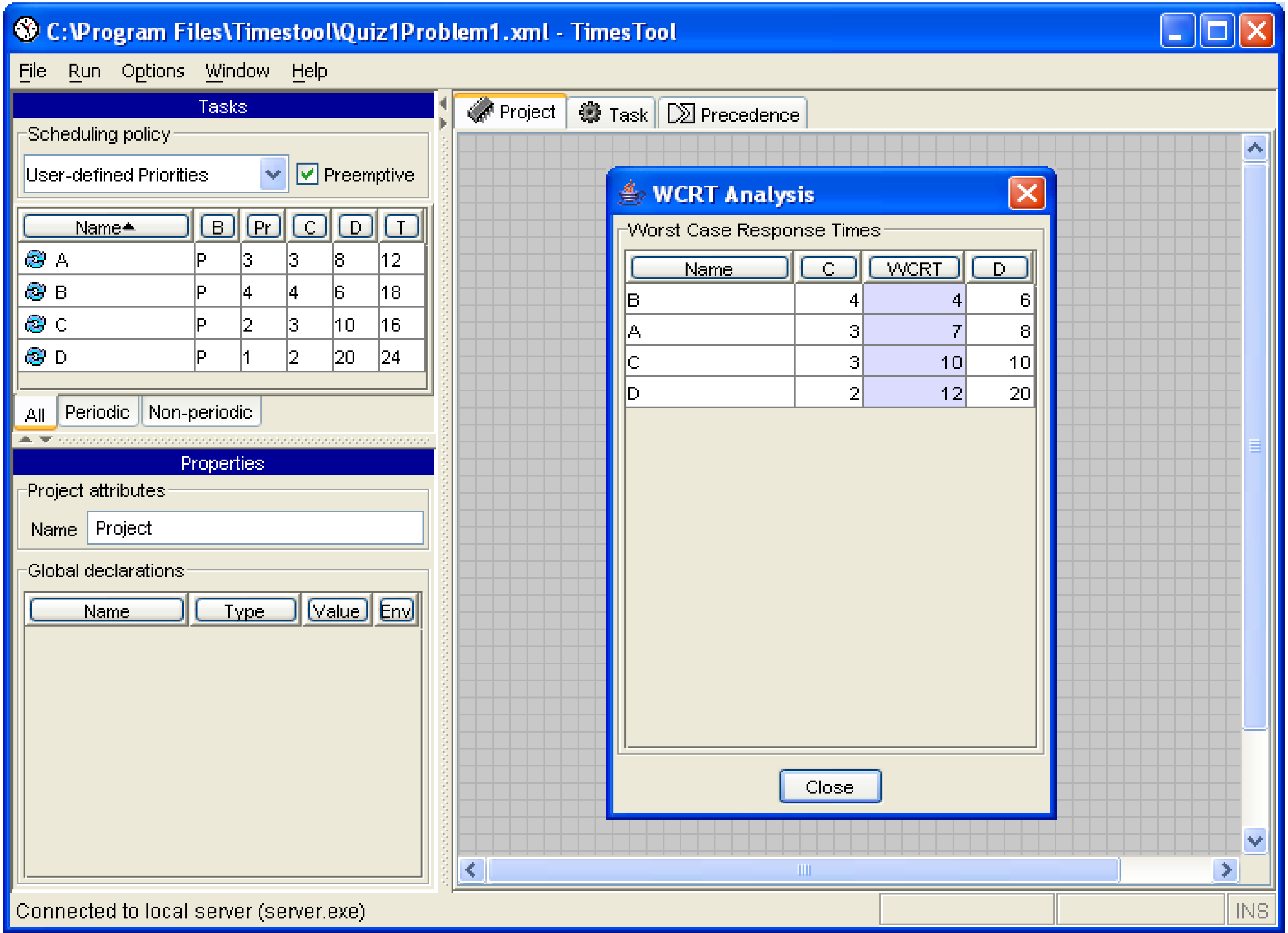


RTiS'03 (18.08.2003)

Times Tool


Analysis
System Specification
Editor
Control structure
Extended Timed Automata
Scheduling strategy
EDF, FIXED, etc.
Task parameters
Table, Task Code
XML
Scheduler generator
System Analysis
Analyser
Scheduler Analyser
Uppaal Verifier
Simulator
Controller Synthesizer
Yes, schedulable
No, not schedulable
Execution Trace
Optimal Schedule
Task Code Library
Code Generator
Co
Synthesis
Modeling

Schedulability Analysis



C:\Program Files\Timestool\Quiz1Problem1.xml - TimesTool

Scheduling policy: User-defined Priorities ☑ Preemptive

Name	B	Pr	C	D	T
A	P	3	3	8	12
B	P	4	4	6	18
C	P	2	3	10	16
D	P	1	2	20	24

Project attributes

Name: Project

Global declarations

Name	Type	Value	Env

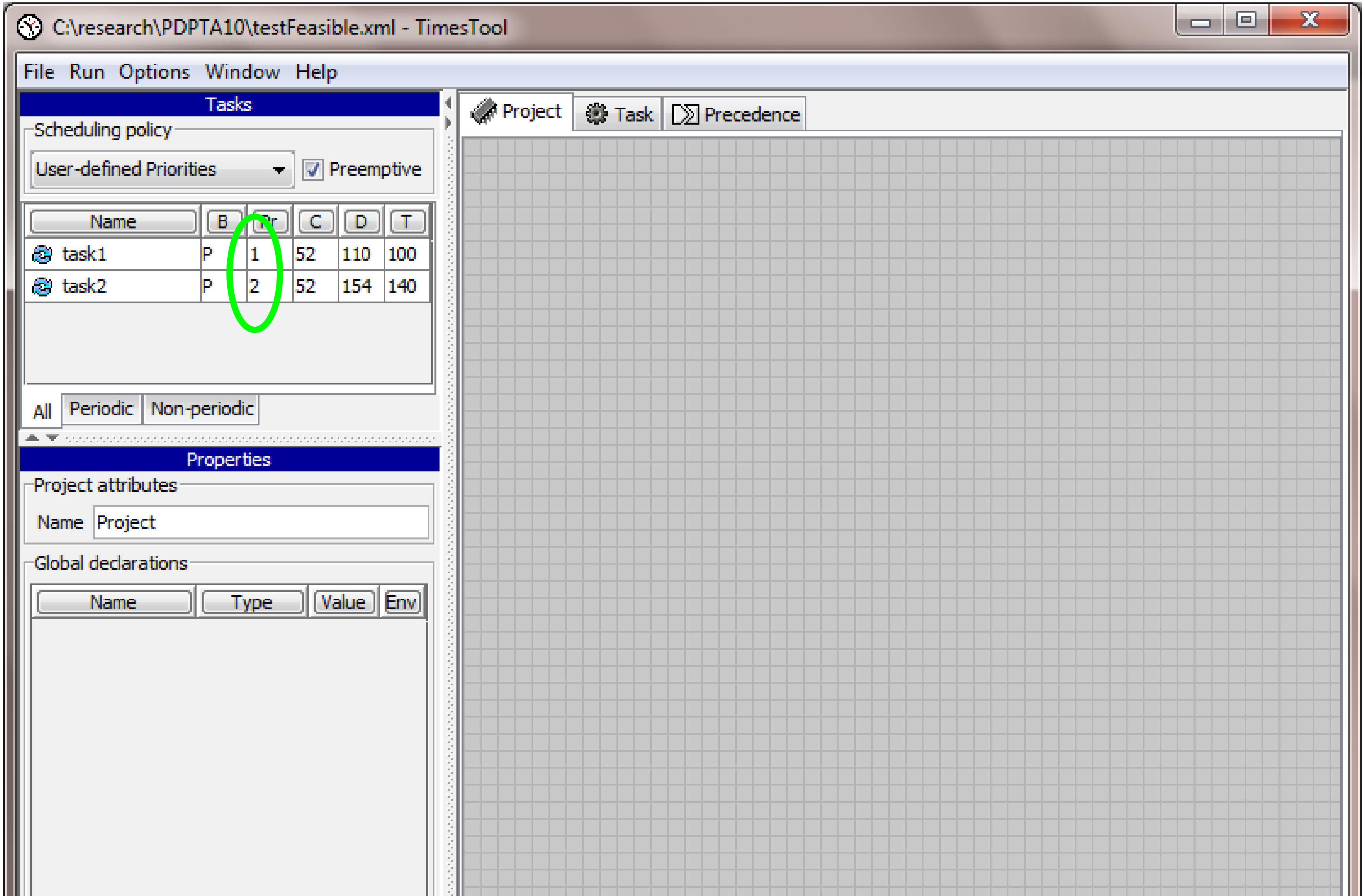
WCRT Analysis

Worst Case Response Times

Name	C	WCRT	D
B	4	4	6
A	3	7	8
C	3	10	10
D	2	12	20

Connected to local server (server.exe)

Feasible Task Set



C:\research\PDPTA10\testFeasible.xml - TimesTool

File Run Options Window Help

Tasks

Scheduling policy

User-defined Priorities ☑ Preemptive

Name	B	Pr	C	D	T
task1	P	1	52	110	100
task2	P	2	52	154	140

All | Periodic | Non-periodic

Properties

Project attributes

Name: Project

Global declarations

Name	Type	Value	Env

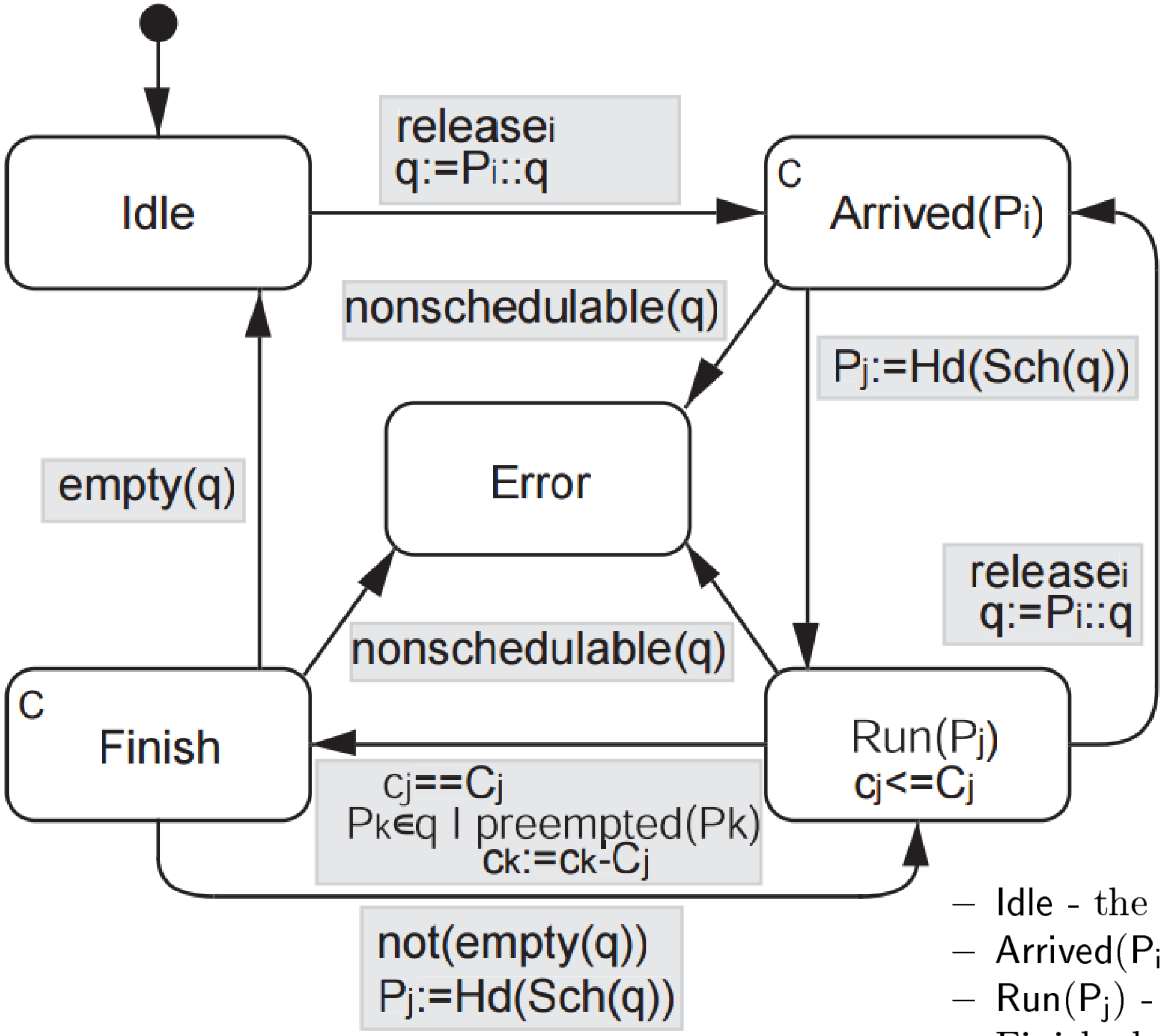
Schedulability Analysis

- Tasks can be defined to be:
  - Periodic
  - Sporadic (with minimum inter-arrival time)
  - Controlled (release controlled by user-defined automaton)
- Priorities can be defined as:
  - User-defined (high number = high priority)
  - Rate monotonic
  - Deadline monotonic
  - Earliest deadline first
  - First come, first served

Schedulability Analysis

- Performed by doing reachability testing on a SCHEDULER automaton to see if the ERROR state is reachable.
- The analysis is performed by selecting Run + Schedulability Analysis.
- The analysis works correctly most of the time, except that the default **two clocks scheduler** may fail if tasks have **arbitrary deadlines**.

General O(n)-Clocks Scheduler



- **q – ready queue**
- **empty(q) – no tasks in the queue**
- **Sch(q) – sorted according to the chosen policy**
- **Hd() - returns the first element**

– Idle - the task queue is empty,

– Arrived(P_i) - the task instance P_i has arrived,

– Run(P_j) - the task instance P_j is running,

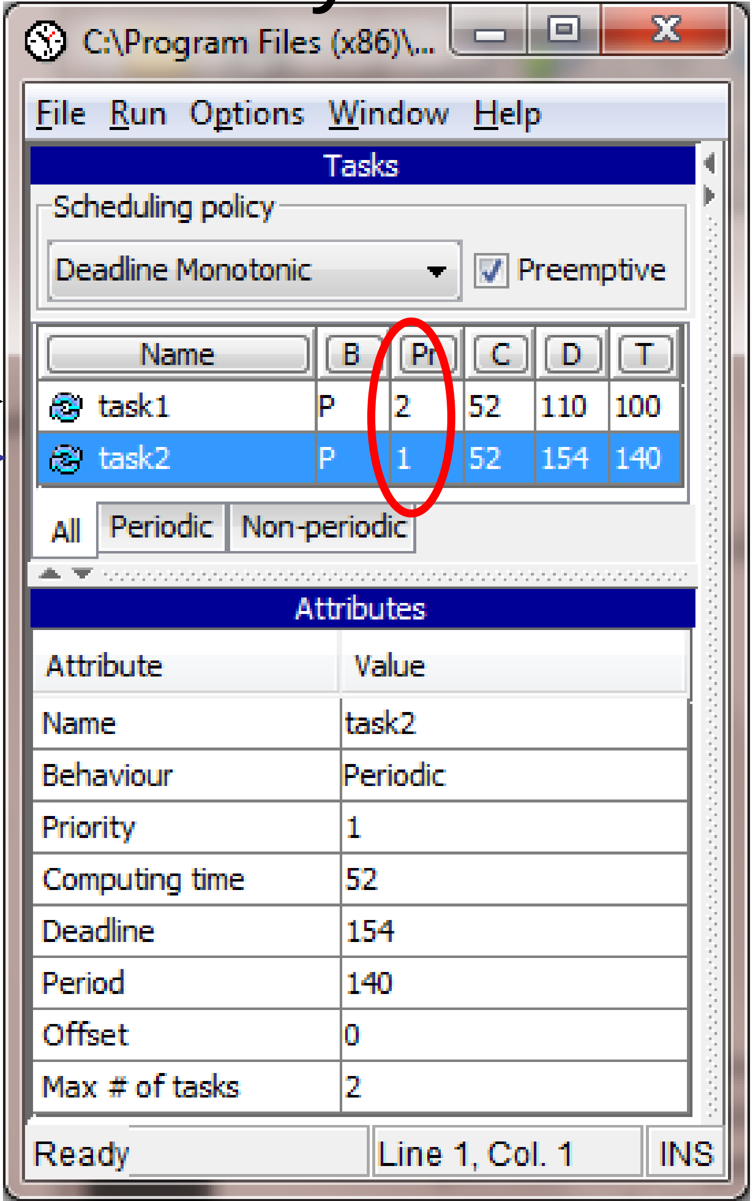
– Finished - a task instance has finished,

– Error - the task queue is non-schedulable.

Audsley's Infeasible Task Set with Arbitrary Deadlines

High Priority →

Low Priority →



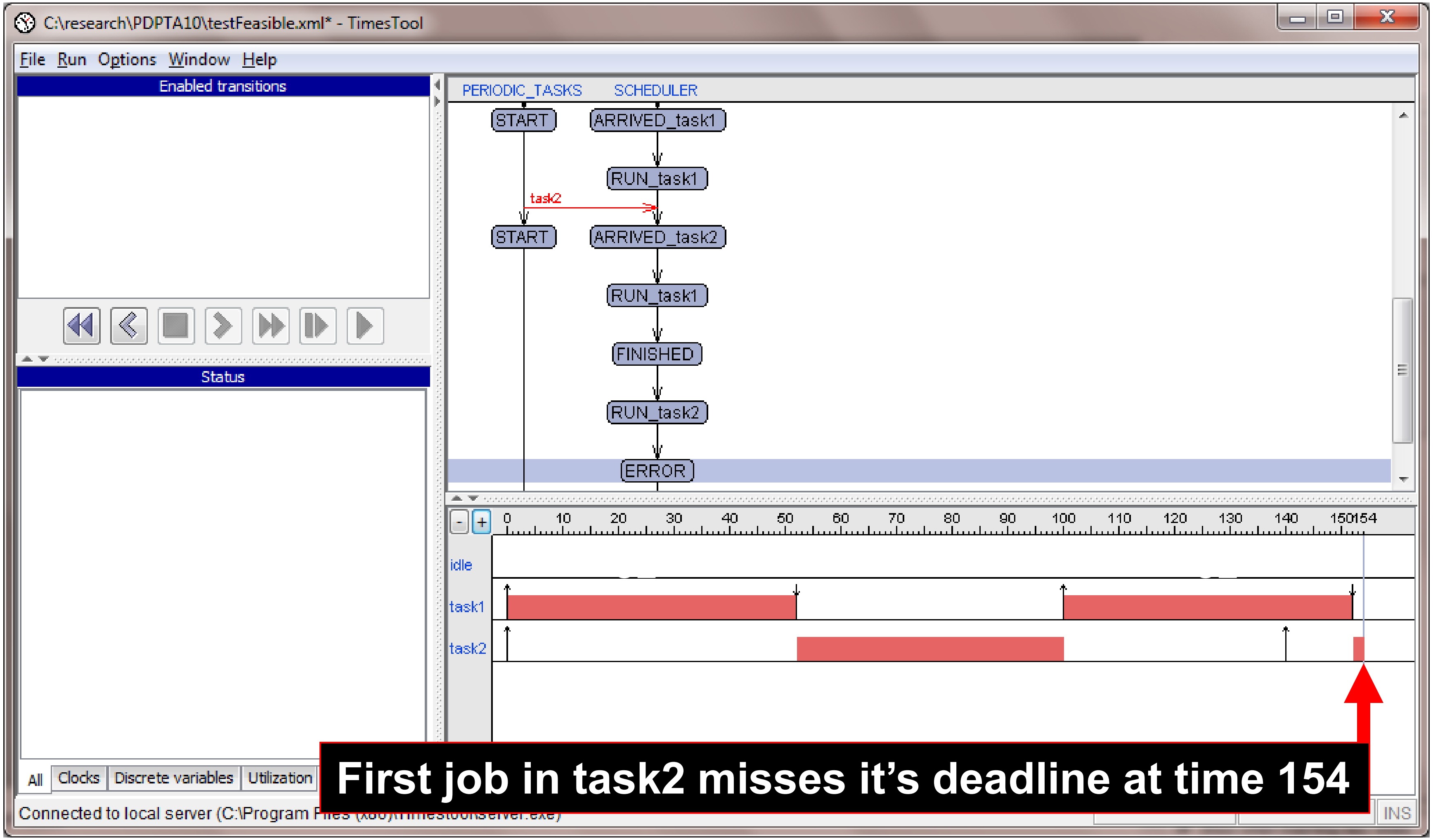
C:\Program Files (x86)\...
File Run Options Window Help
Tasks
All Periodic Non-periodic
Attributes
Ready
Line 1, Col. 1
INS

Scheduling policy: Deadline Monotonic ☑ Preemptive

Name	B	Pr	C	D	T
task1	P	2	52	110	100
task2	P	1	52	154	140

Attribute	Value
Name	task2
Behaviour	Periodic
Priority	1
Computing time	52
Deadline	154
Period	140
Offset	0
Max # of tasks	2

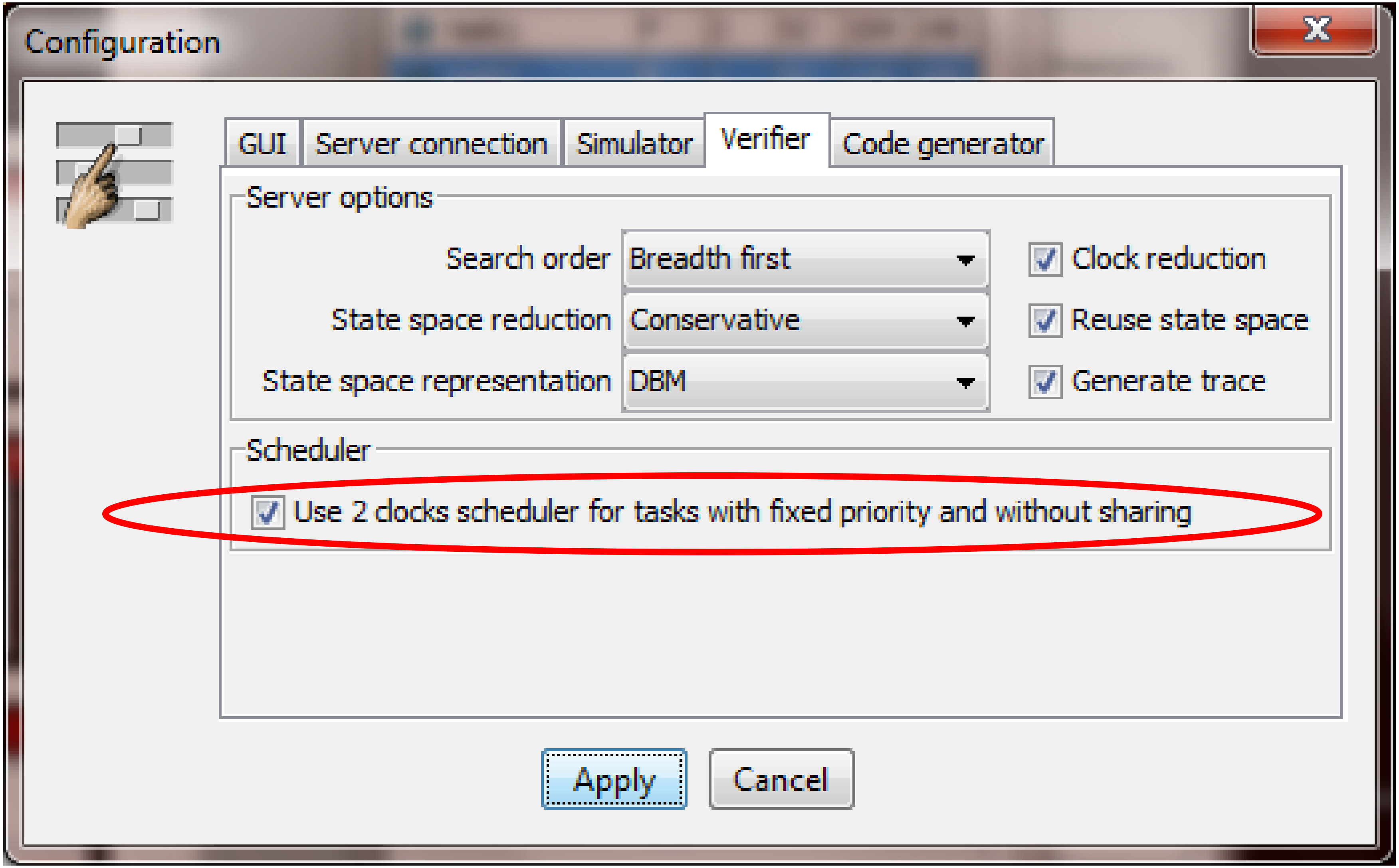
Infeasible Task Set in Simulator



Connected to local server (C:\Program Files (x86)\TimesToolServer.exe)

First job in task2 misses it’s deadline at time 154

Times Options – 2-clocks scheduler



Server options

Search order: Breadth first

State space reduction: Conservative

State space representation: DBM

☑ Clock reduction

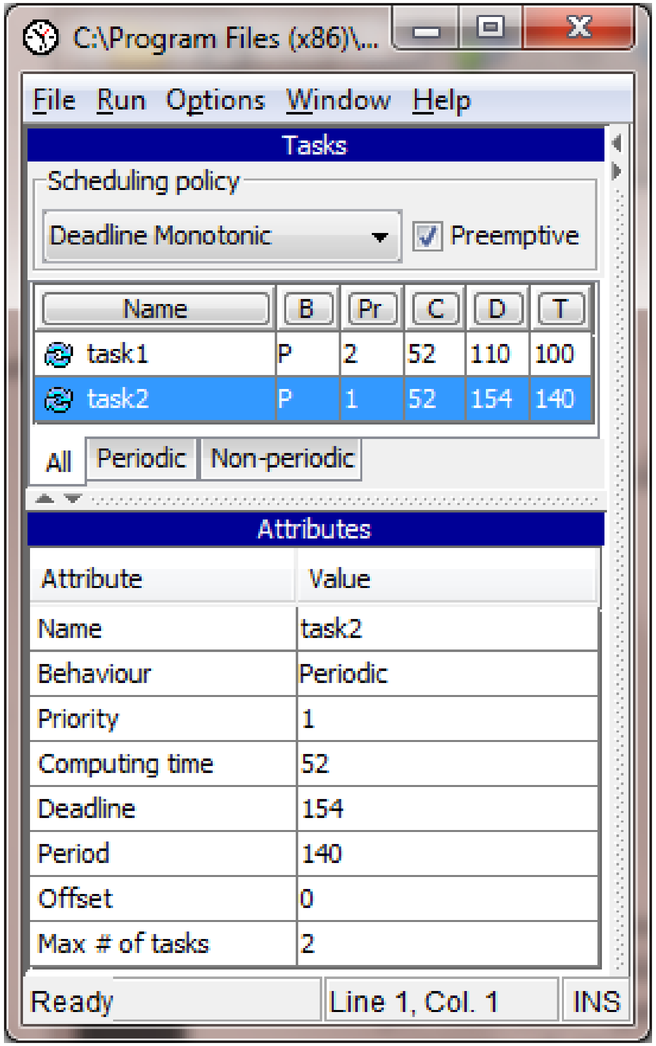
☑ Reuse state space

☑ Generate trace

Scheduler

☑ Use 2 clocks scheduler for tasks with fixed priority and without sharing

Times Tool Reports that the Task Set is Feasible

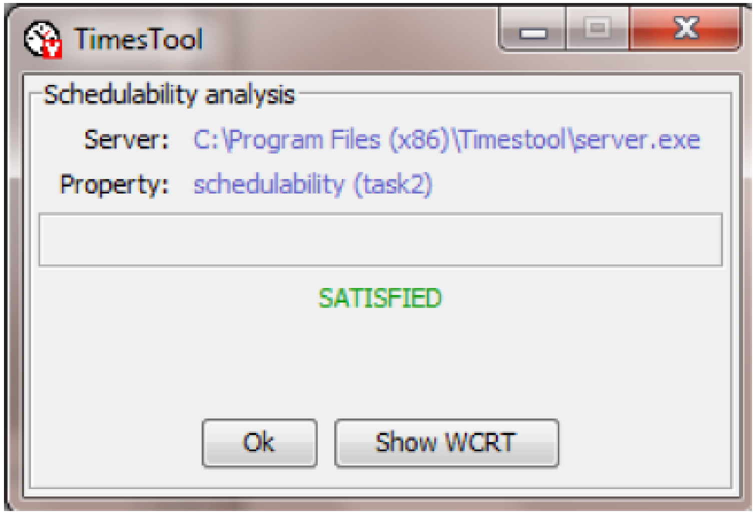


Scheduling policy: Deadline Monotonic ☑ Preemptive

Name	B	Pr	C	D	T
task1	P	2	52	110	100
task2	P	1	52	154	140

Attribute	Value
Name	task2
Behaviour	Periodic
Priority	1
Computing time	52
Deadline	154
Period	140
Offset	0
Max # of tasks	2

Ready | Line 1, Col. 1 | INS

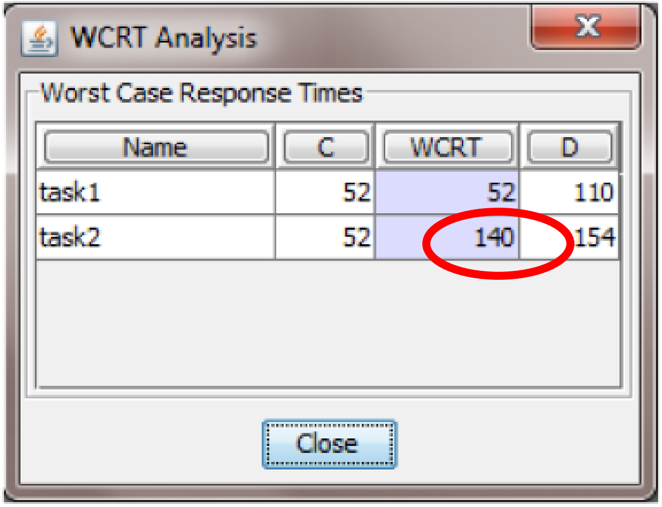


Schedulability analysis

Server: C:\Program Files (x86)\Timestool\server.exe

Property: schedulability (task2)

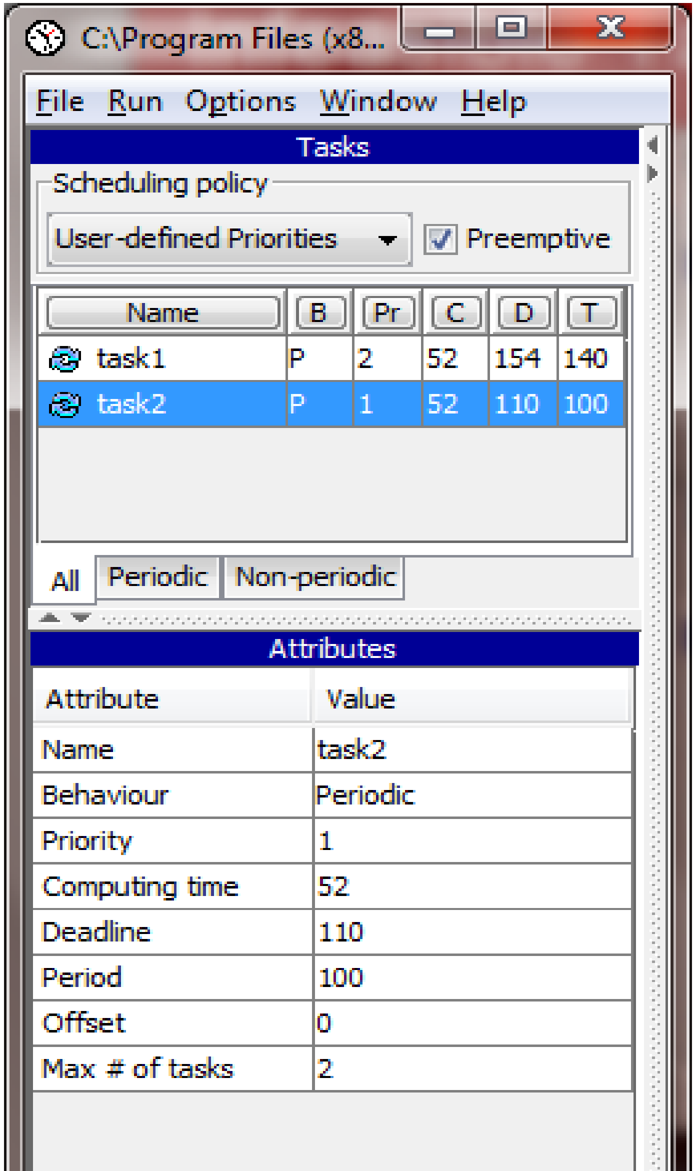
SATISFIED



Worst Case Response Times

Name	C	WCRT	D
task1	52	52	110
task2	52	140	154

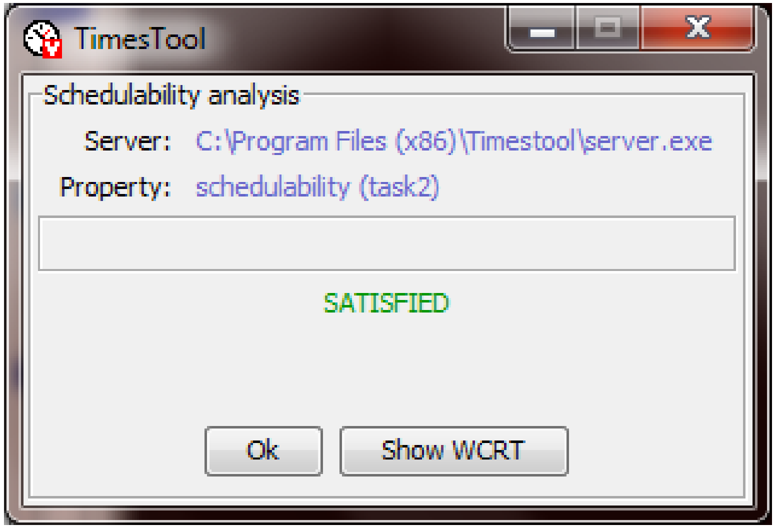
Switching Task Priorities, Times Tool Reports that the Task Set is Feasible



Scheduling policy: User-defined Priorities ☑ Preemptive

Name	B	Pr	C	D	T
task1	P	2	52	154	140
task2	P	1	52	110	100

Attribute	Value
Name	task2
Behaviour	Periodic
Priority	1
Computing time	52
Deadline	110
Period	100
Offset	0
Max # of tasks	2

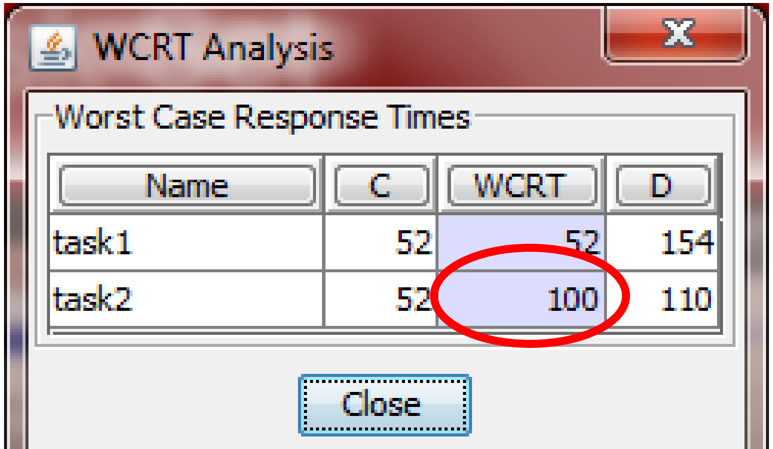


Schedulability analysis

Server: C:\Program Files (x86)\Timestool\server.exe

Property: schedulability (task2)

SATISFIED



Worst Case Response Times

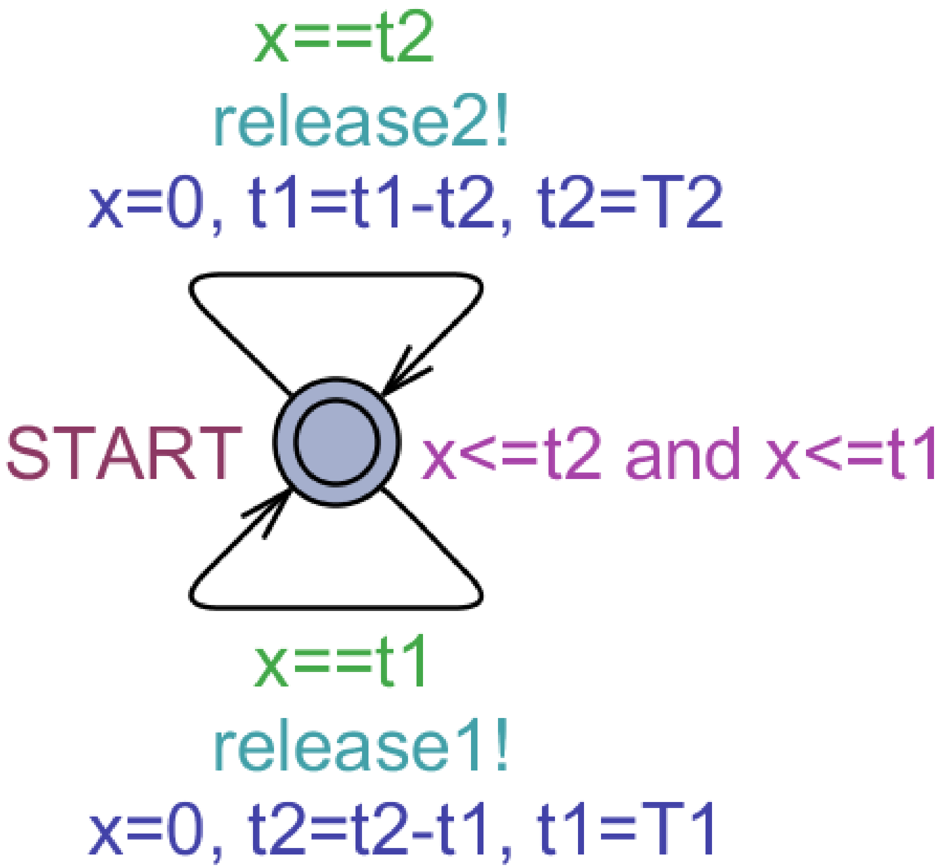
Name	C	WCRT	D
task1	52	52	154
task2	52	100	110

But, the wrong WCRT!

New 2-Clocks Scheduler

- Correctly detects when an arbitrary task set is feasible.
- Only requires two clocks for n tasks:
  - One clock to check if the currently scheduled task has missed its deadline, and
  - One clock to check that the running task has completed it’s execution.
- For simplicity, we consider Audsley’s examples with just two tasks, but it is trivial to generalize to n tasks.

PERIODIC_TASKS Automaton



x==t2

release2!

x=0, t1=t1-t2, t2=T2

START

x<=t2 and x<=t1

x==t1

release1!

x=0, t2=t2-t1, t1=T1

T1 = Period of Task 1

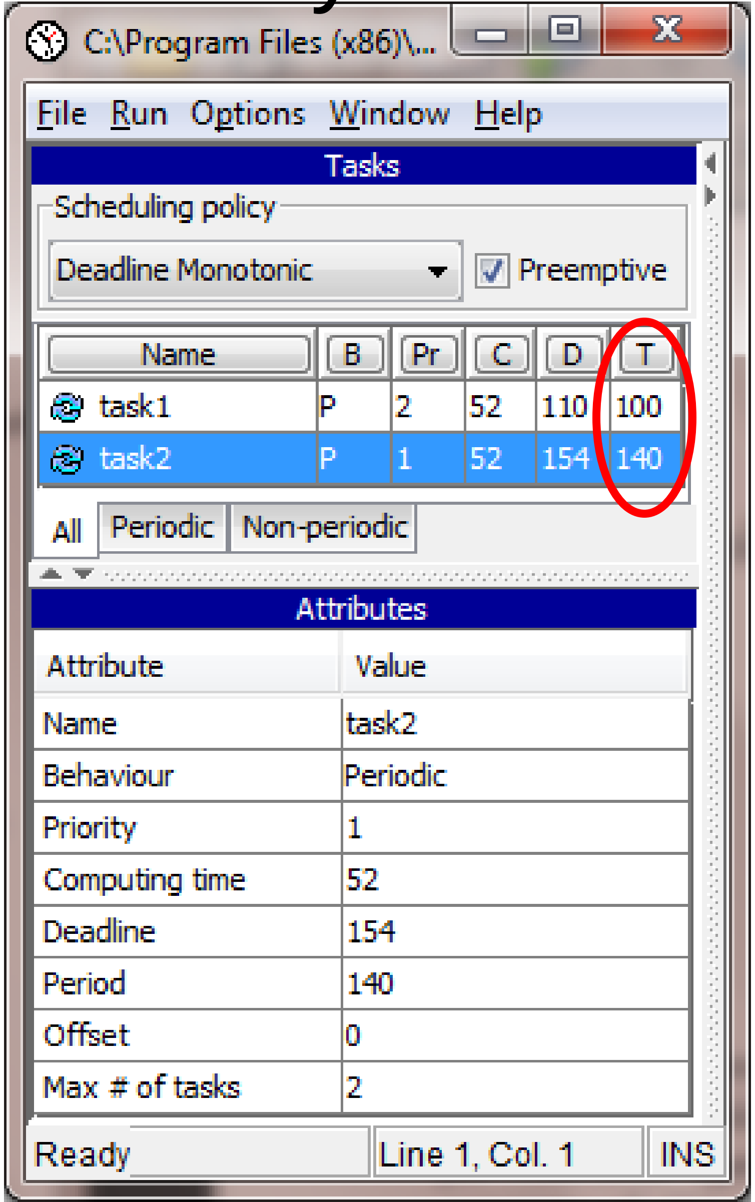
T2 = Period of Task 2

t1 = time before releasing a job in Task 1

t2 = time before releasing a job in Task 2

Note that the job generating automaton requires one clock x.

Audsley's Infeasible Task Set* with Arbitrary Deadlines


C:\Program Files (x86)\...
File Run Options Window Help
Tasks
Attributes
All Periodic Non-periodic
Ready
Line 1, Col. 1
INS

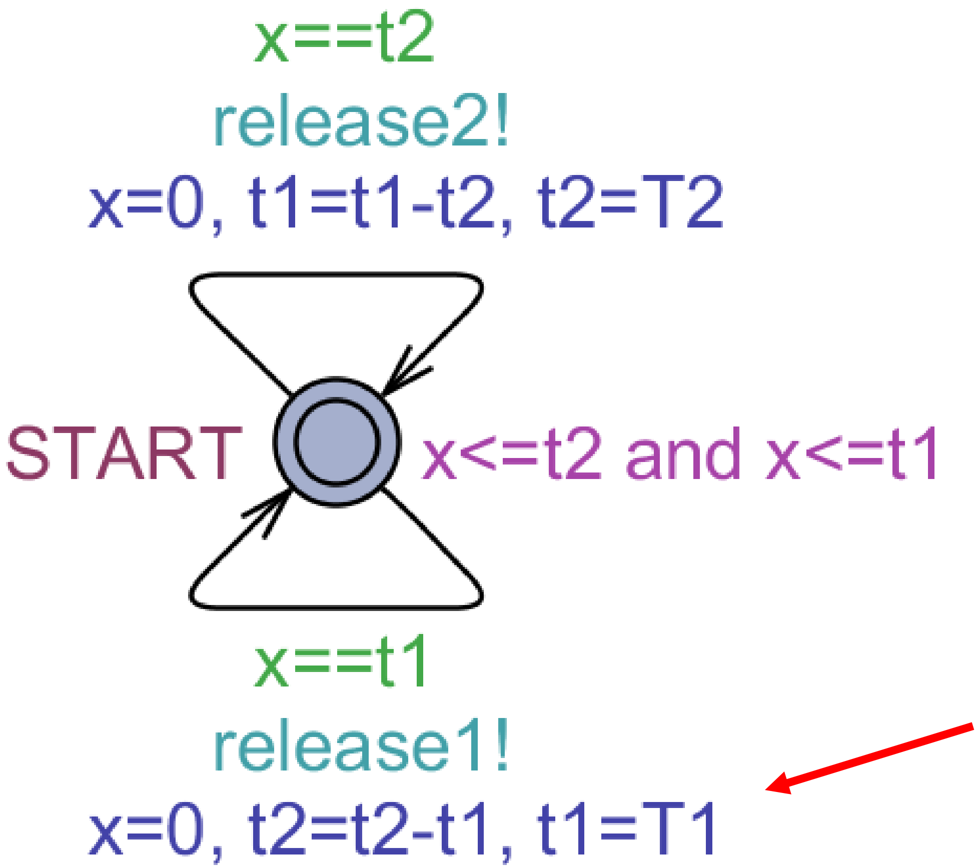
Scheduling policy: Deadline Monotonic ☑ Preemptive

Name	B	Pr	C	D	T
task1	P	2	52	110	100
task2	P	1	52	154	140

Attribute	Value
Name	task2
Behaviour	Periodic
Priority	1
Computing time	52
Deadline	154
Period	140
Offset	0
Max # of tasks	2

* using a Deadline Monotonic priority assignment and a priority-based preemptive scheduler

PERIODIC_TASKS Automaton



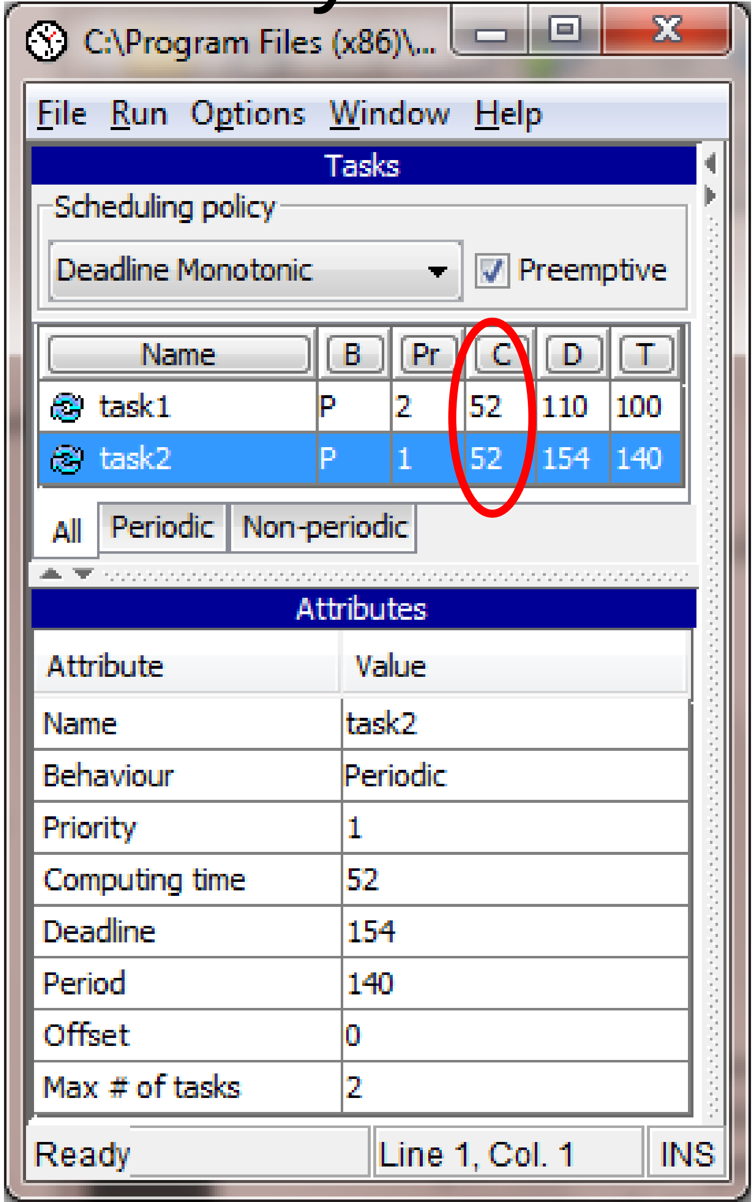
T1 = 100
T2 = 140

t1 = 0, initially
t2 = 0, initially

Then, t1 = 100,
and t2 = 140,

Then, t2 = 40,
and t1 = 100,
…

Audsley's Infeasible Task Set with Arbitrary Deadlines



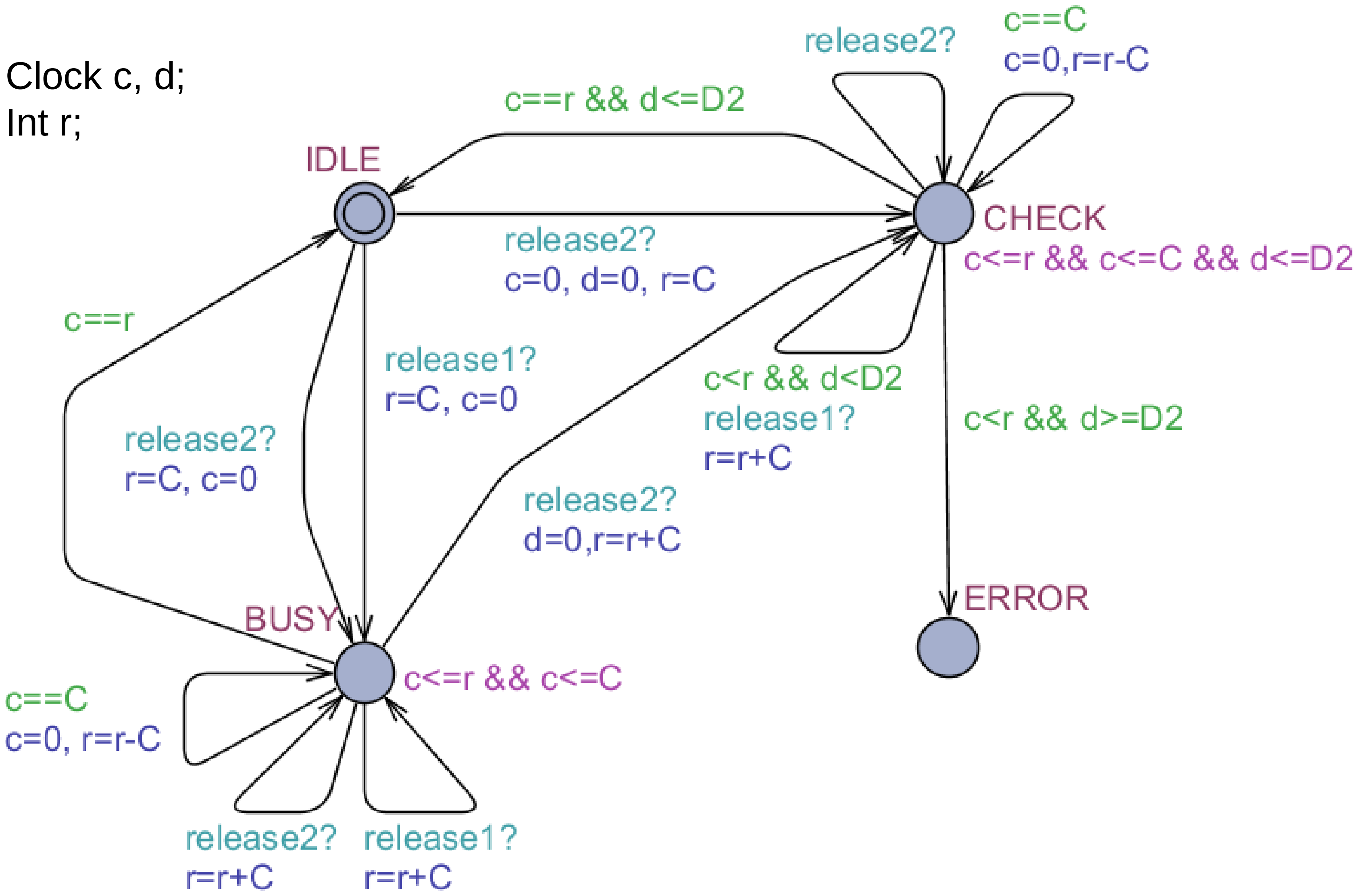
Scheduling policy: Deadline Monotonic ☑ Preemptive

Name	B	Pr	C	D	T
task1	P	2	52	110	100
task2	P	1	52	154	140

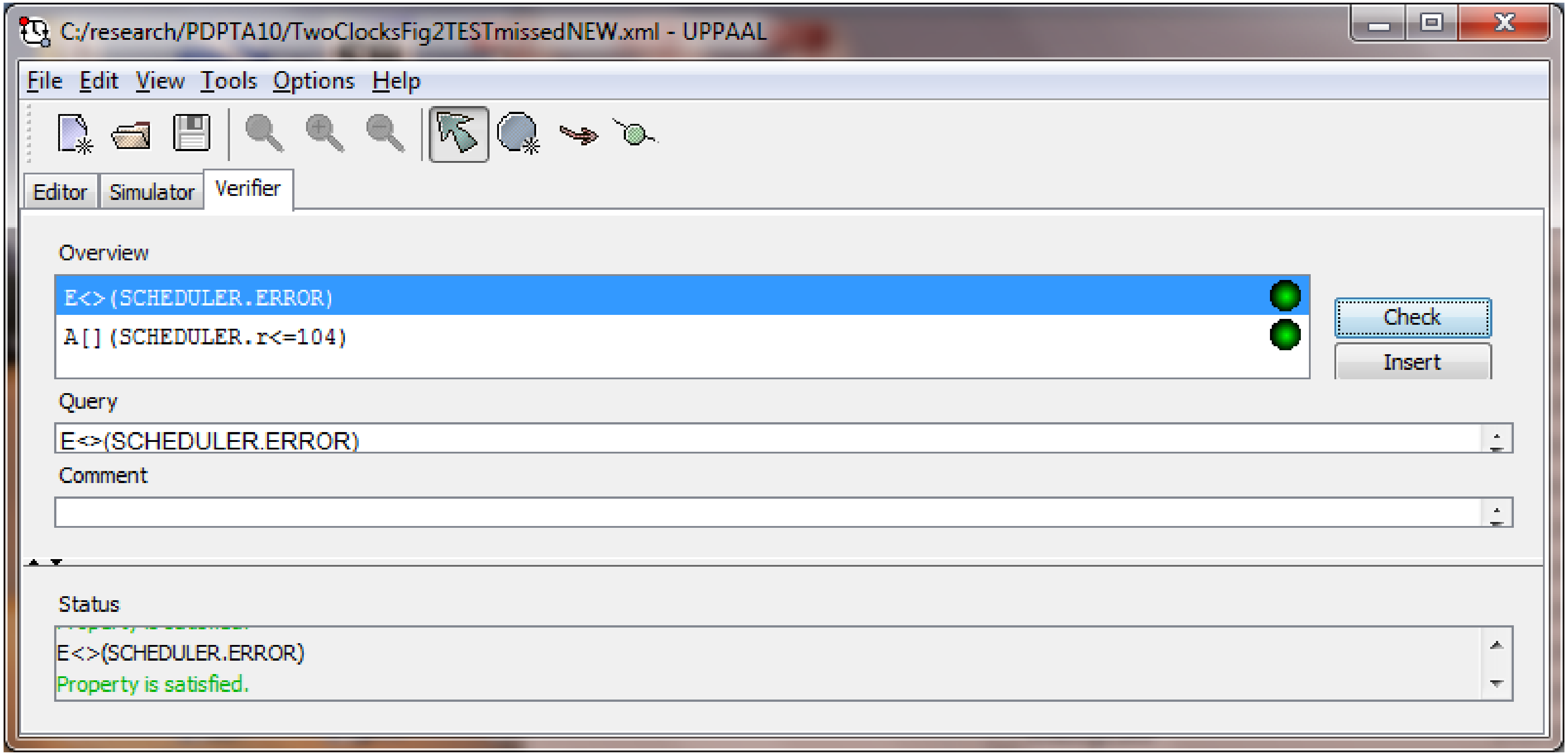
Attribute	Value
Name	task2
Behaviour	Periodic
Priority	1
Computing time	52
Deadline	154
Period	140
Offset	0
Max # of tasks	2

SCHEDULER Automaton, C=52

Clock c, d;
Int r;



Task Set is not Feasible


C:/research/PDPTA10/TwoClocksFig2TESTmissedNEW.xml - UPPAAL
File Edit View Tools Options Help
Editor
Simulator
Verifier
Overview
Check
Insert
Query
Comment
Status

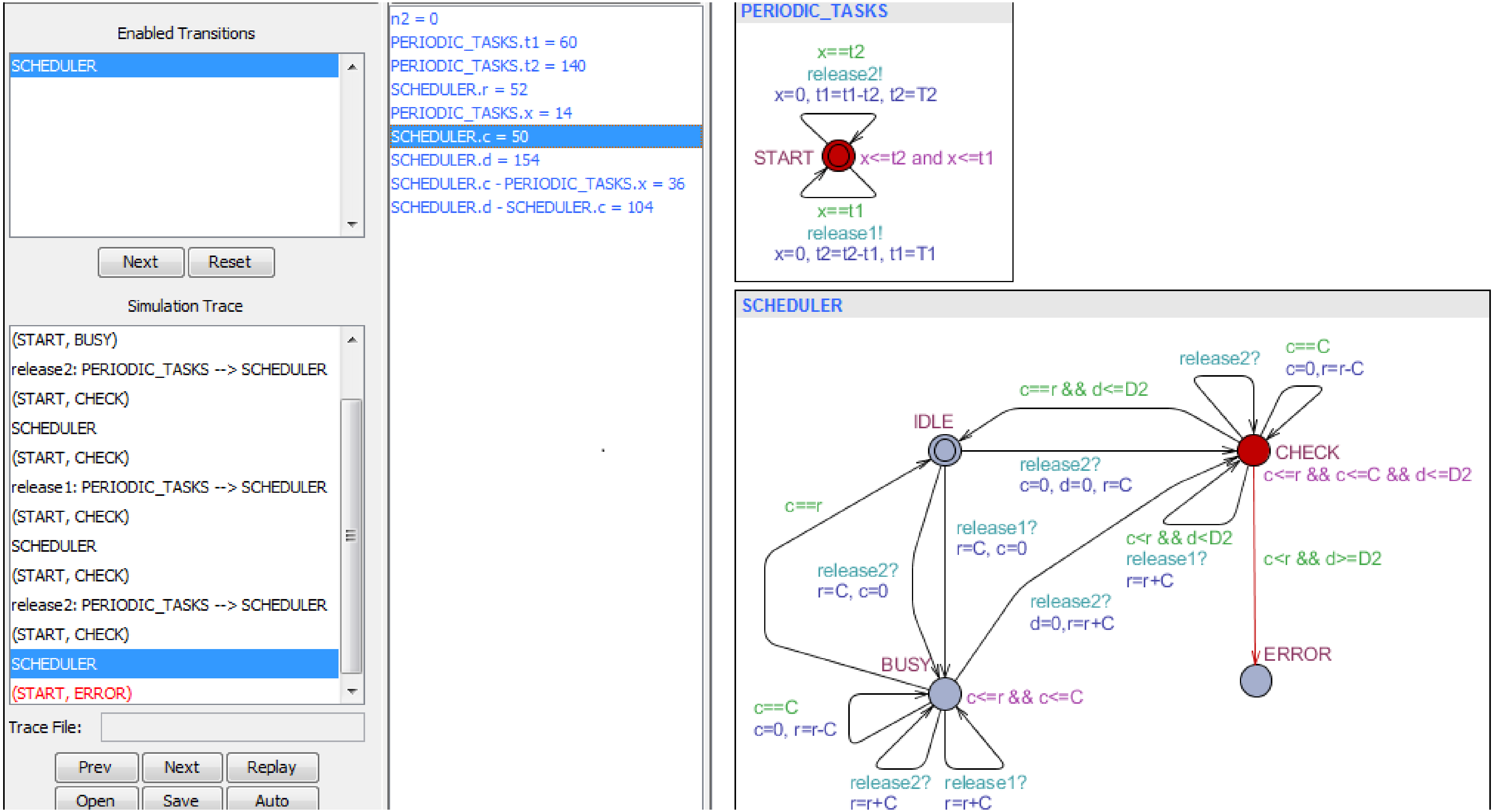
```
E<>(SCHEDULER.ERROR) ●
A[](SCHEDULER.r<=104) ●
```

Query: E<>(SCHEDULER.ERROR)

Comment:

```
E<>(SCHEDULER.ERROR)
Property is satisfied.
```

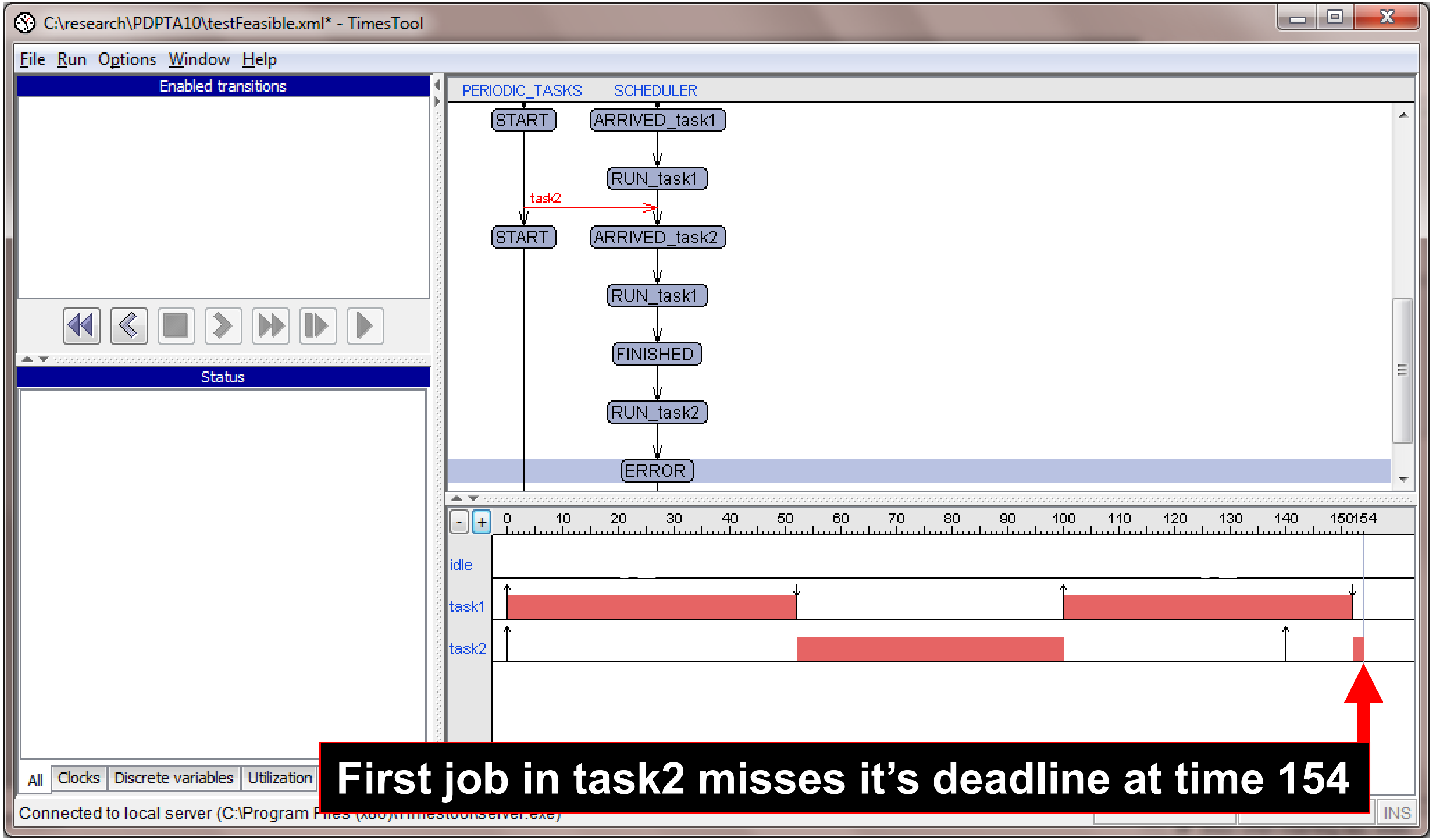
UPPAAL Diagnostic Trace



```
(START, BUSY)
release2: PERIODIC_TASKS --> SCHEDULER
(START, CHECK)
SCHEDULER
(START, CHECK)
release1: PERIODIC_TASKS --> SCHEDULER
(START, CHECK)
SCHEDULER
(START, CHECK)
release2: PERIODIC_TASKS --> SCHEDULER
(START, CHECK)
SCHEDULER
(START, ERROR)
```

```
n2 = 0
PERIODIC_TASKS.t1 = 60
PERIODIC_TASKS.t2 = 140
SCHEDULER.r = 52
PERIODIC_TASKS.x = 14
SCHEDULER.c = 50
SCHEDULER.d = 154
SCHEDULER.c - PERIODIC_TASKS.x = 36
SCHEDULER.d - SCHEDULER.c = 104
```

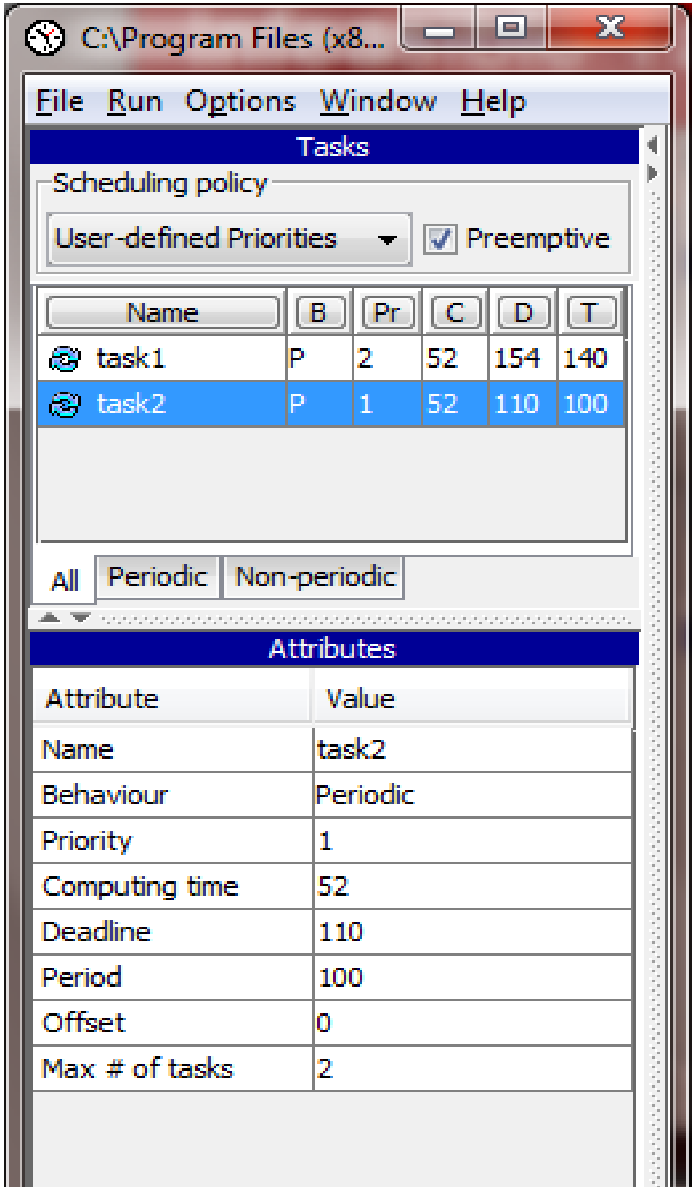
Infeasible Task Set in Simulator



Connected to local server (C:\Program F

First job in task2 misses it’s deadline at time 154

Audsley's Feasible Task Set with Arbitrary Deadlines (Priorities Switched)



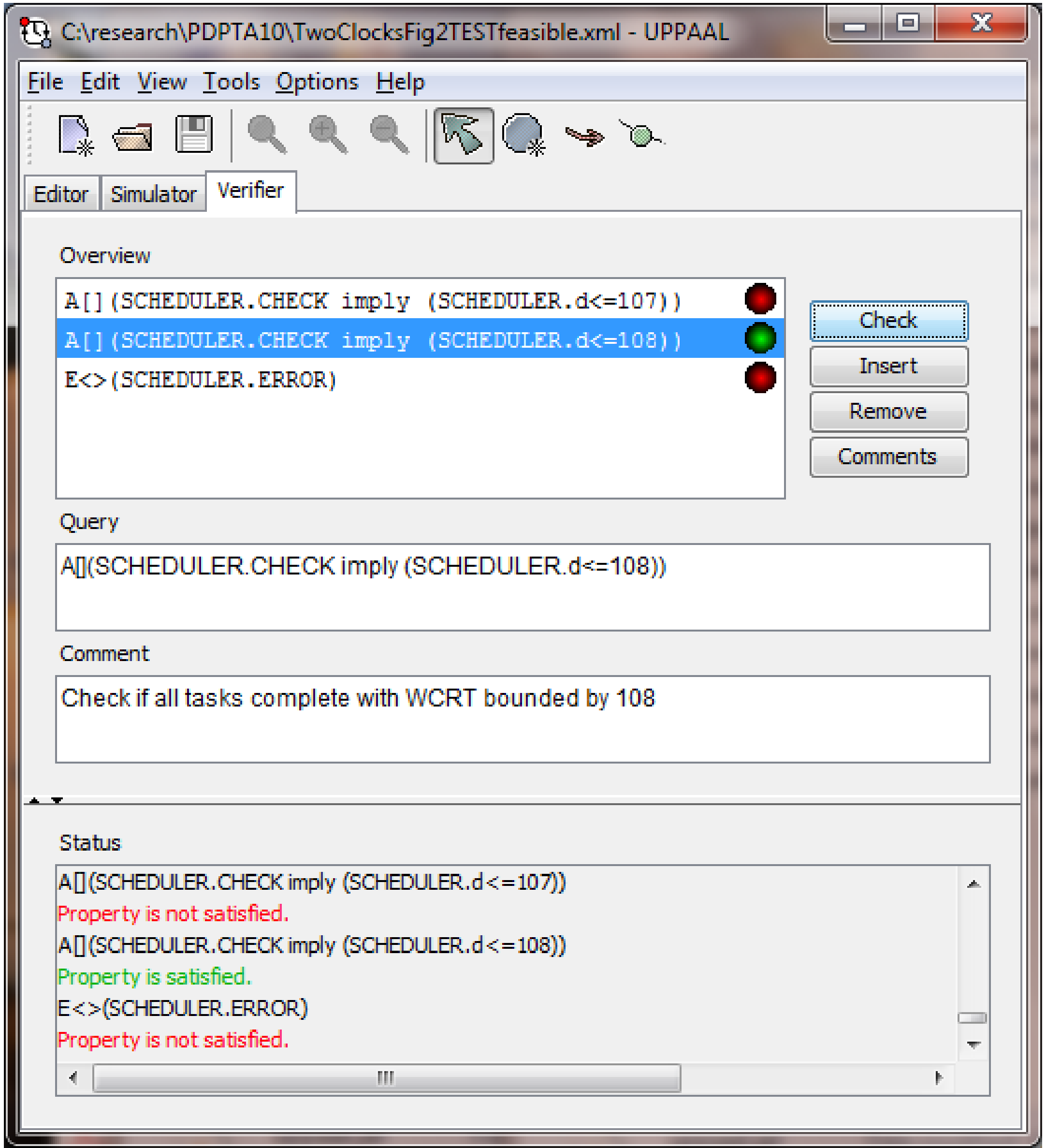
Scheduling policy

User-defined Priorities ☑ Preemptive

Name	B	Pr	C	D	T
task1	P	2	52	154	140
task2	P	1	52	110	100

Attribute	Value
Name	task2
Behaviour	Periodic
Priority	1
Computing time	52
Deadline	110
Period	100
Offset	0
Max # of tasks	2

Task Set is Feasible, WCRT = 108



C:\research\PDPTA10\TwoClocksFig2TESTfeasible.xml - UPPAAL
File Edit View Tools Options Help
Editor
Simulator
Verifier
Check
Insert
Remove
Comments

Overview

```
A[](SCHEDULER.CHECK imply (SCHEDULER.d<=107)) 🔴
A[](SCHEDULER.CHECK imply (SCHEDULER.d<=108)) 🟢
E<>(SCHEDULER.ERROR) 🔴
```

Query

A[](SCHEDULER.CHECK imply (SCHEDULER.d<=108))

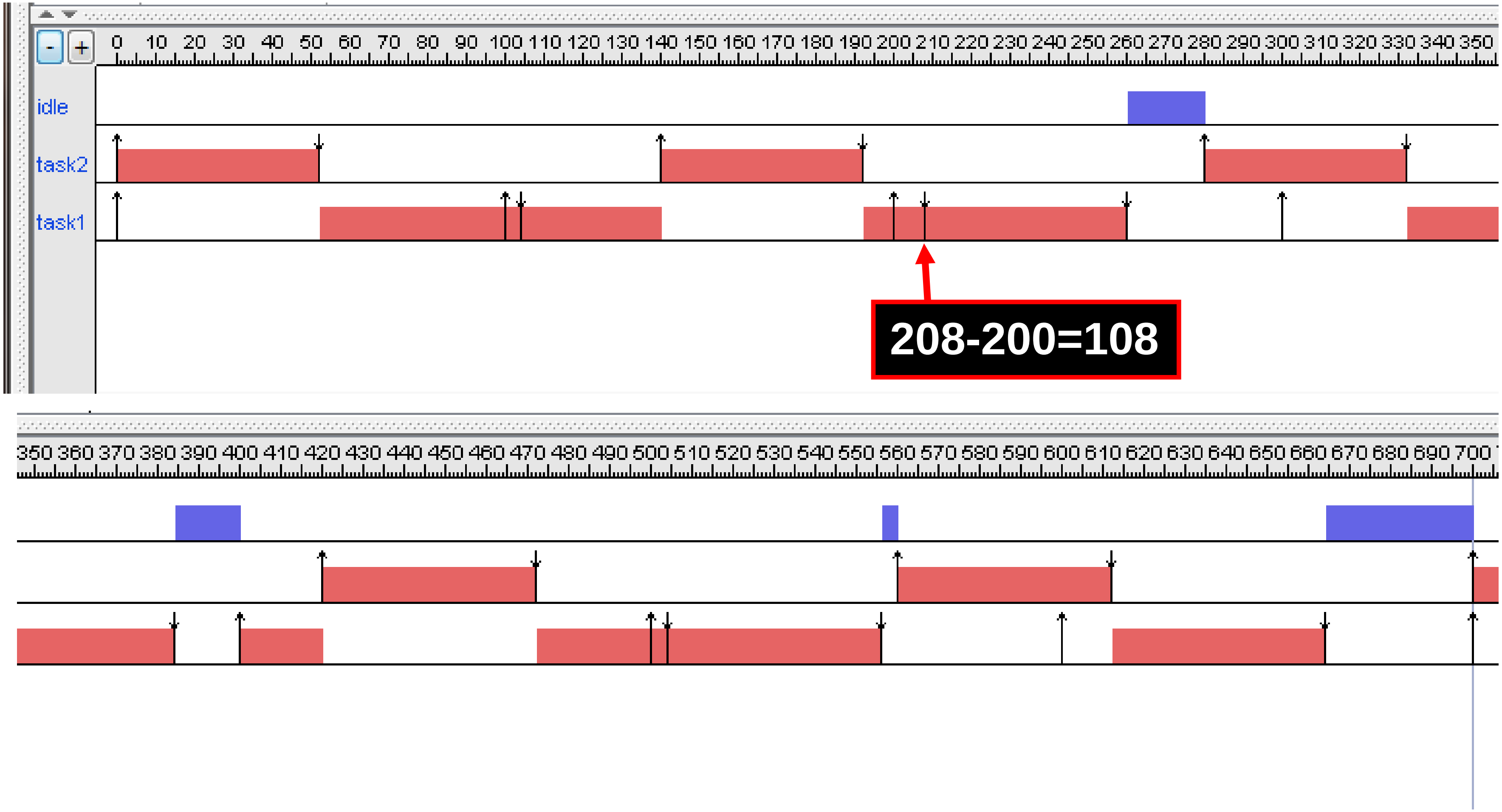
Comment

Check if all tasks complete with WCRT bounded by 108

Status

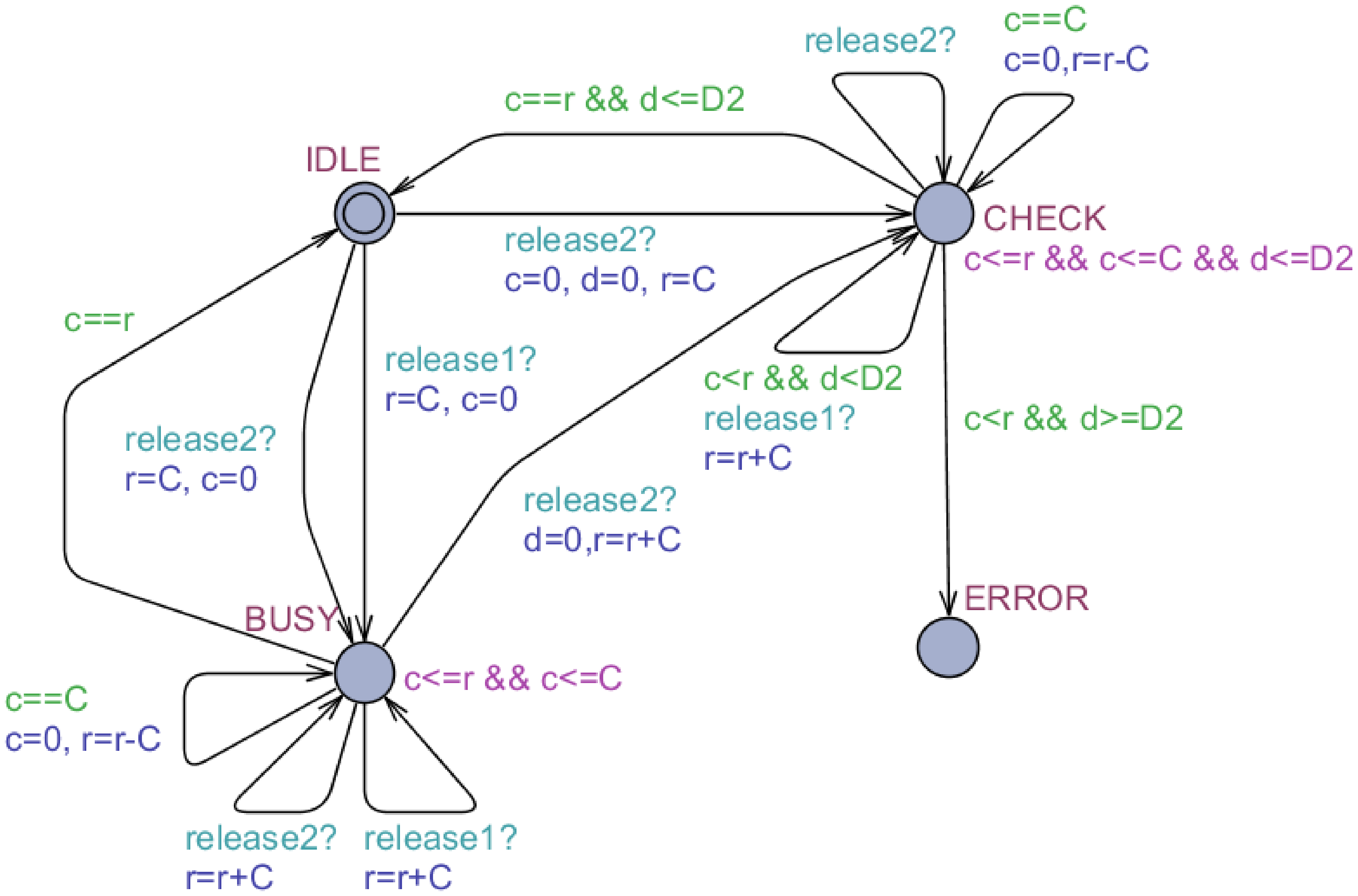
```
A[](SCHEDULER.CHECK imply (SCHEDULER.d<=107))
Property is not satisfied.
A[](SCHEDULER.CHECK imply (SCHEDULER.d<=108))
Property is satisfied.
E<>(SCHEDULER.ERROR)
Property is not satisfied.
```

Gannt Chart – over one hyperperiod

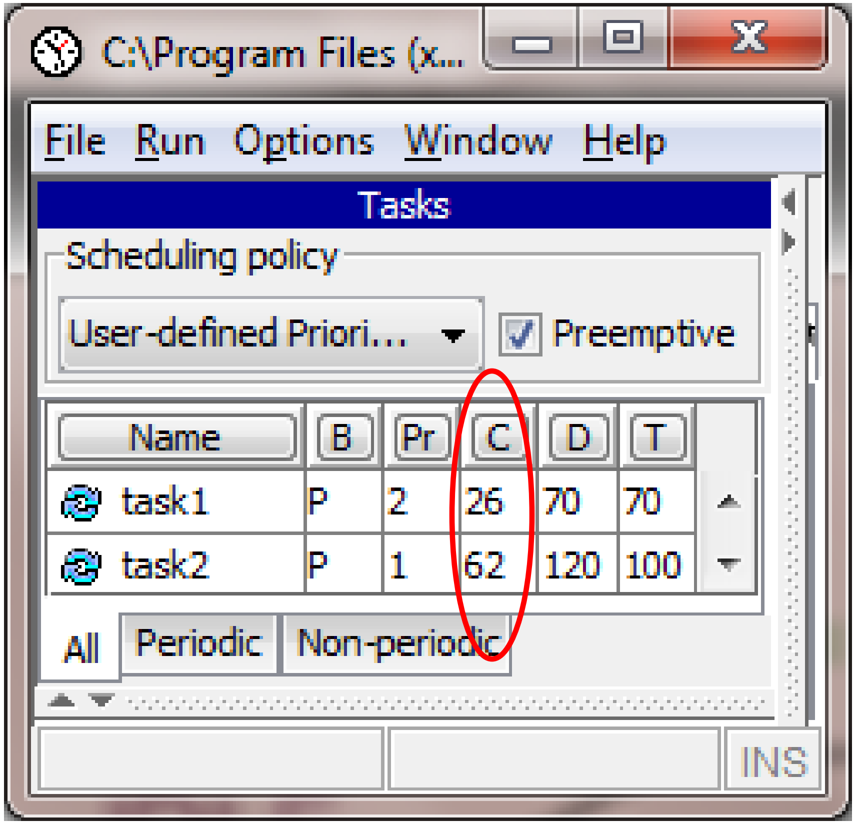


208-200=108

SCHEDULER Automaton

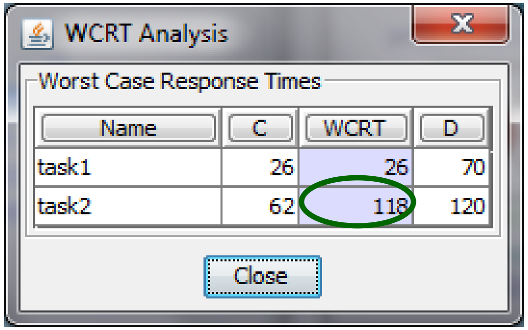


Another Example



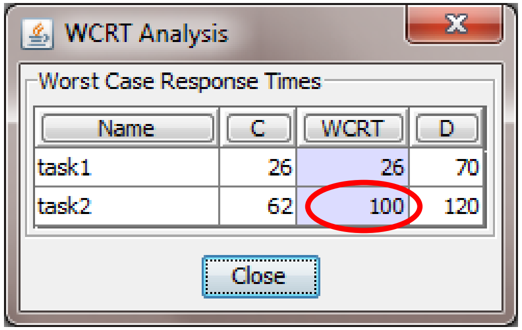
Scheduling policy: User-defined Priori... ☑ Preemptive

Name	B	Pr	C	D	T
task1	P	2	26	70	70
task2	P	1	62	120	100



Worst Case Response Times

Name	C	WCRT	D
task1	26	26	70
task2	62	118	120



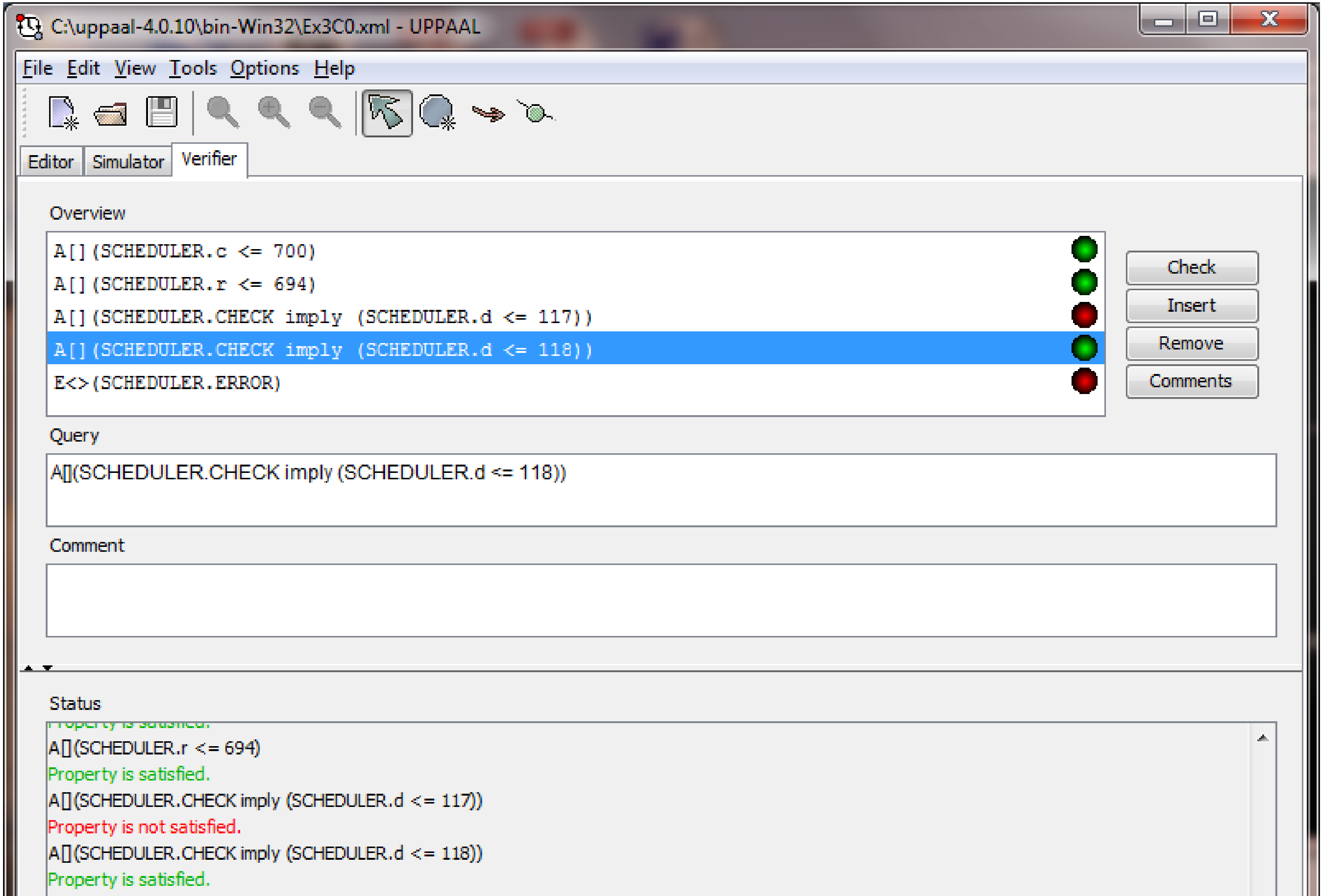
Worst Case Response Times

Name	C	WCRT	D
task1	26	26	70
task2	62	100	120

with Times Tool’s 2-clock scheduler

C1 = 26 and C2 = 62, but what about C?

Verifier Output using new 2-clocks model



C:\uppaal-4.0.10\bin-Win32\Ex3C0.xml - UPPAAL
File Edit View Tools Options Help
Editor
Simulator
Verifier
Overview
Check
Insert
Remove
Comments
Query
Comment
Status

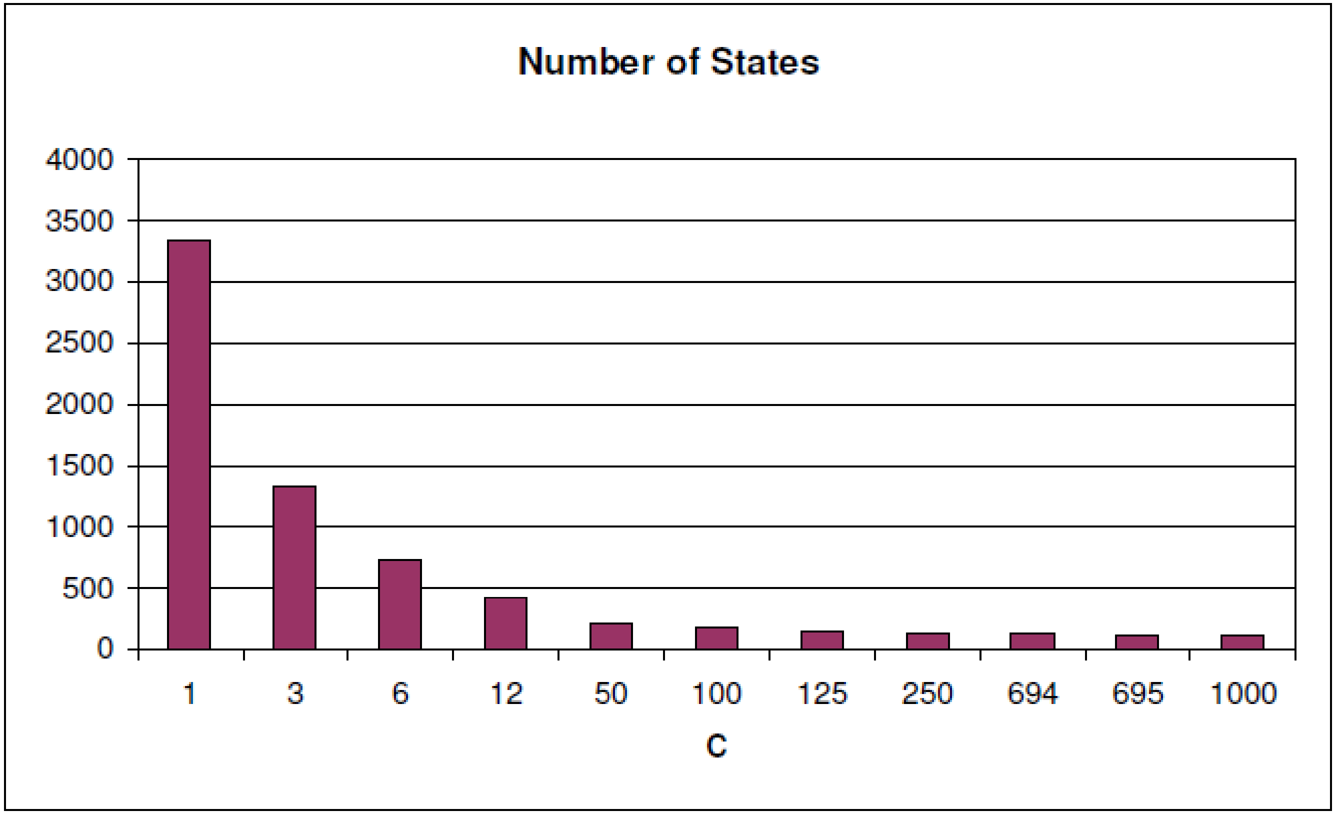
```
A[] (SCHEDULER.c <= 700)                              ●
A[] (SCHEDULER.r <= 694)                              ●
A[] (SCHEDULER.CHECK imply (SCHEDULER.d <= 117))      ●
A[] (SCHEDULER.CHECK imply (SCHEDULER.d <= 118))      ●
E<> (SCHEDULER.ERROR)                                 ●
```

Query: A[](SCHEDULER.CHECK imply (SCHEDULER.d <= 118))

```
Property is satisfied.
A[](SCHEDULER.r <= 694)
Property is satisfied.
A[](SCHEDULER.CHECK imply (SCHEDULER.d <= 117))
Property is not satisfied.
A[](SCHEDULER.CHECK imply (SCHEDULER.d <= 118))
Property is satisfied.
```

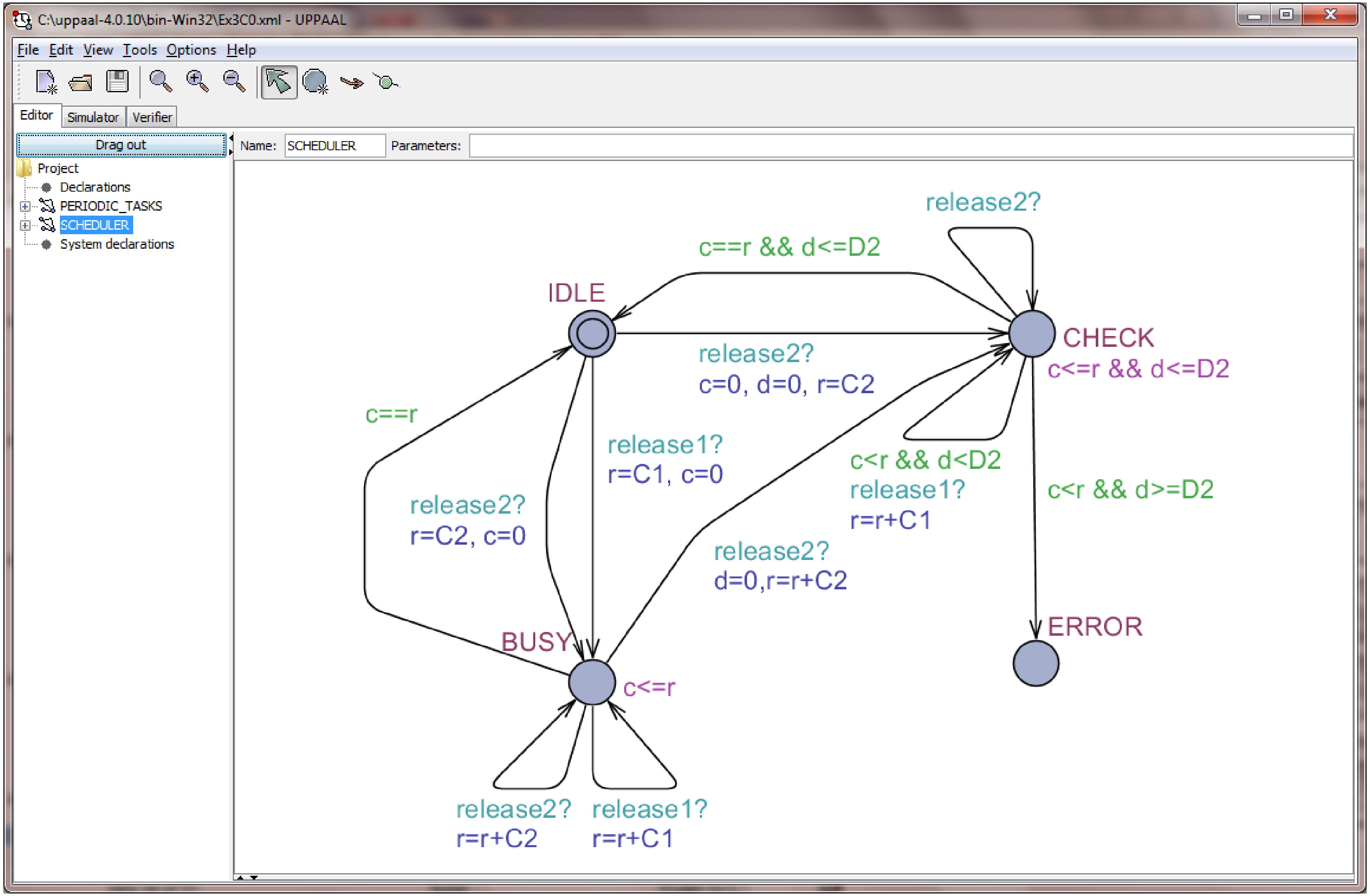
Effect of Setting C

C	States
1	3337
3	1322
6	729
12	419
50	212
100	174
125	141
250	129
694	123
695	120
1000	120



Effect of Removing C

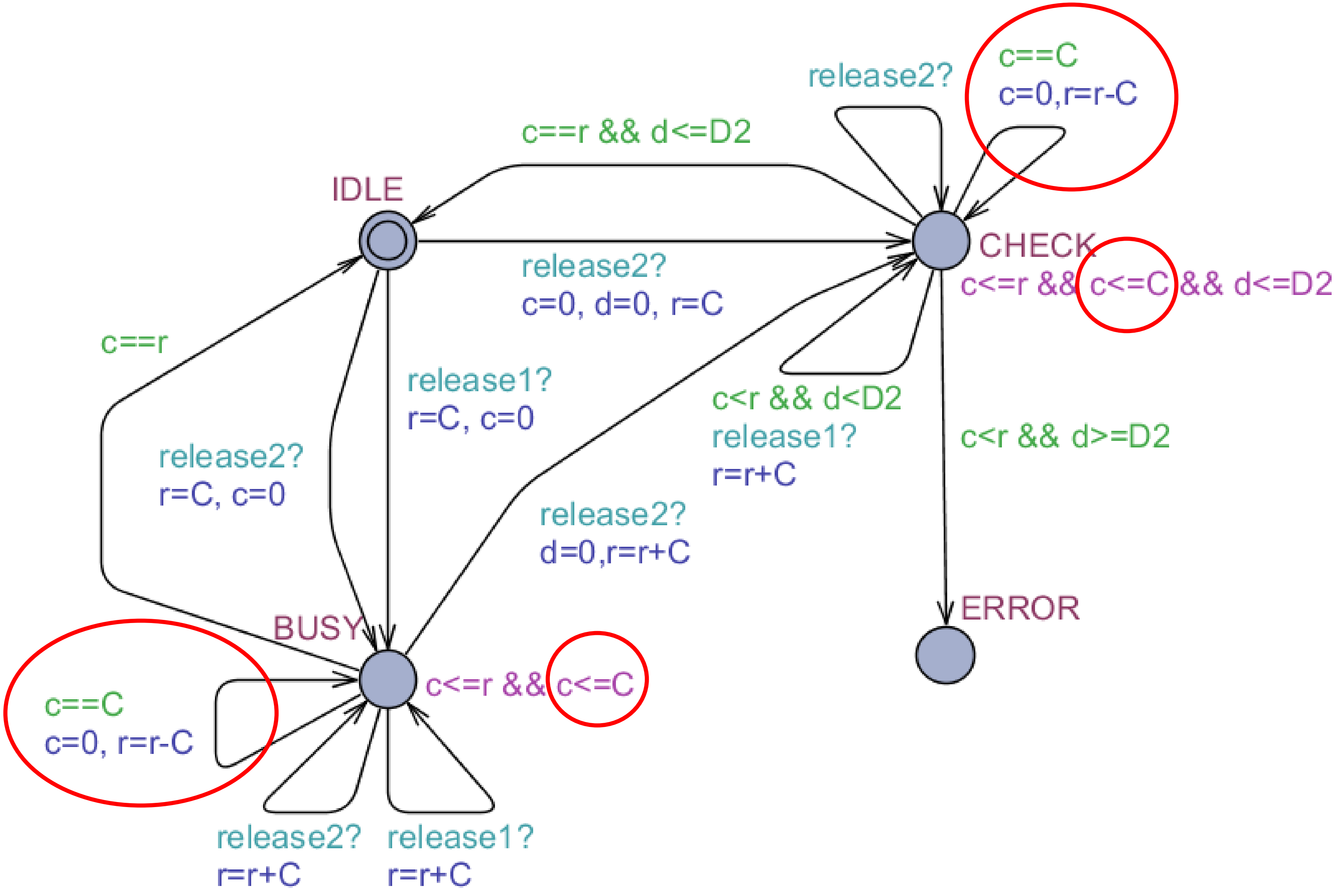
- Number of States generated drops to 120



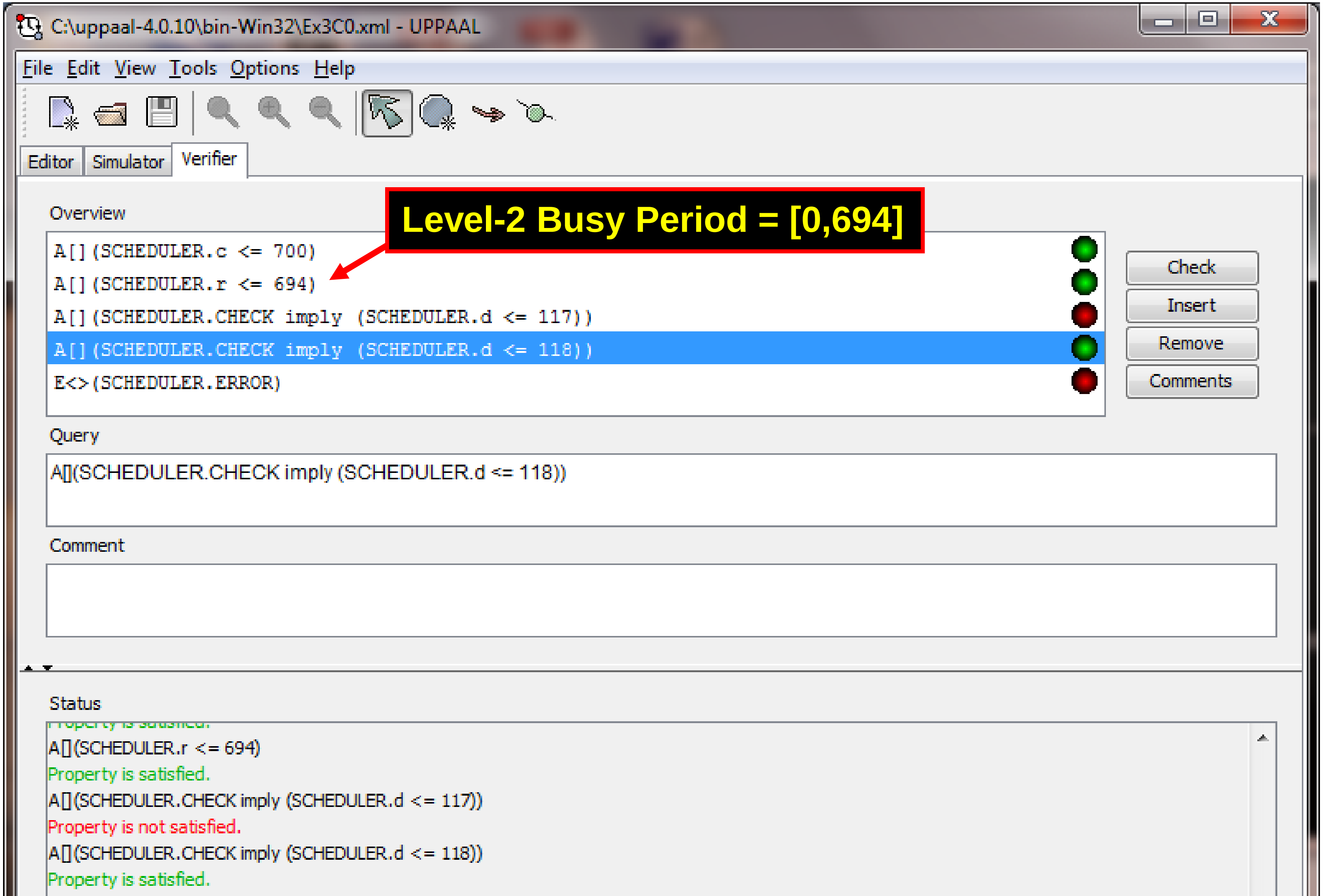
C:\uppaal-4.0.10\bin-Win32\Ex3C0.xml - UPPAAL

Name: SCHEDULER Parameters:

Old SCHEDULER Automaton



Verifier Output using new 2-clocks model


C:\uppaal-4.0.10\bin-Win32\Ex3C0.xml - UPPAAL
File Edit View Tools Options Help
Editor
Simulator
Verifier
Overview
Check
Insert
Remove
Comments
Query
Comment
Status

Level-2 Busy Period = [0,694]

Overview

```
A[](SCHEDULER.c <= 700)                                  🟢
A[](SCHEDULER.r <= 694)                                  🟢
A[](SCHEDULER.CHECK imply (SCHEDULER.d <= 117))          🔴
A[](SCHEDULER.CHECK imply (SCHEDULER.d <= 118))          🟢
E<>(SCHEDULER.ERROR)                                     🔴
```

Query

```
A[](SCHEDULER.CHECK imply (SCHEDULER.d <= 118))
```

Status

```
Property is satisfied.
A[](SCHEDULER.r <= 694)
Property is satisfied.
A[](SCHEDULER.CHECK imply (SCHEDULER.d <= 117))
Property is not satisfied.
A[](SCHEDULER.CHECK imply (SCHEDULER.d <= 118))
Property is satisfied.
```

Summary

- The 2-Clocks Scheduler presented in this paper can be used to correctly test the feasibility of periodic task sets with arbitrary deadlines.
- All models and source code are available on-line.

Code Synthesis

- Code Synthesis is used to construct BrickOS source code.
- The resulting Makefile and source code is somewhat buggy, but can be modified to build executable BrickOS applications.

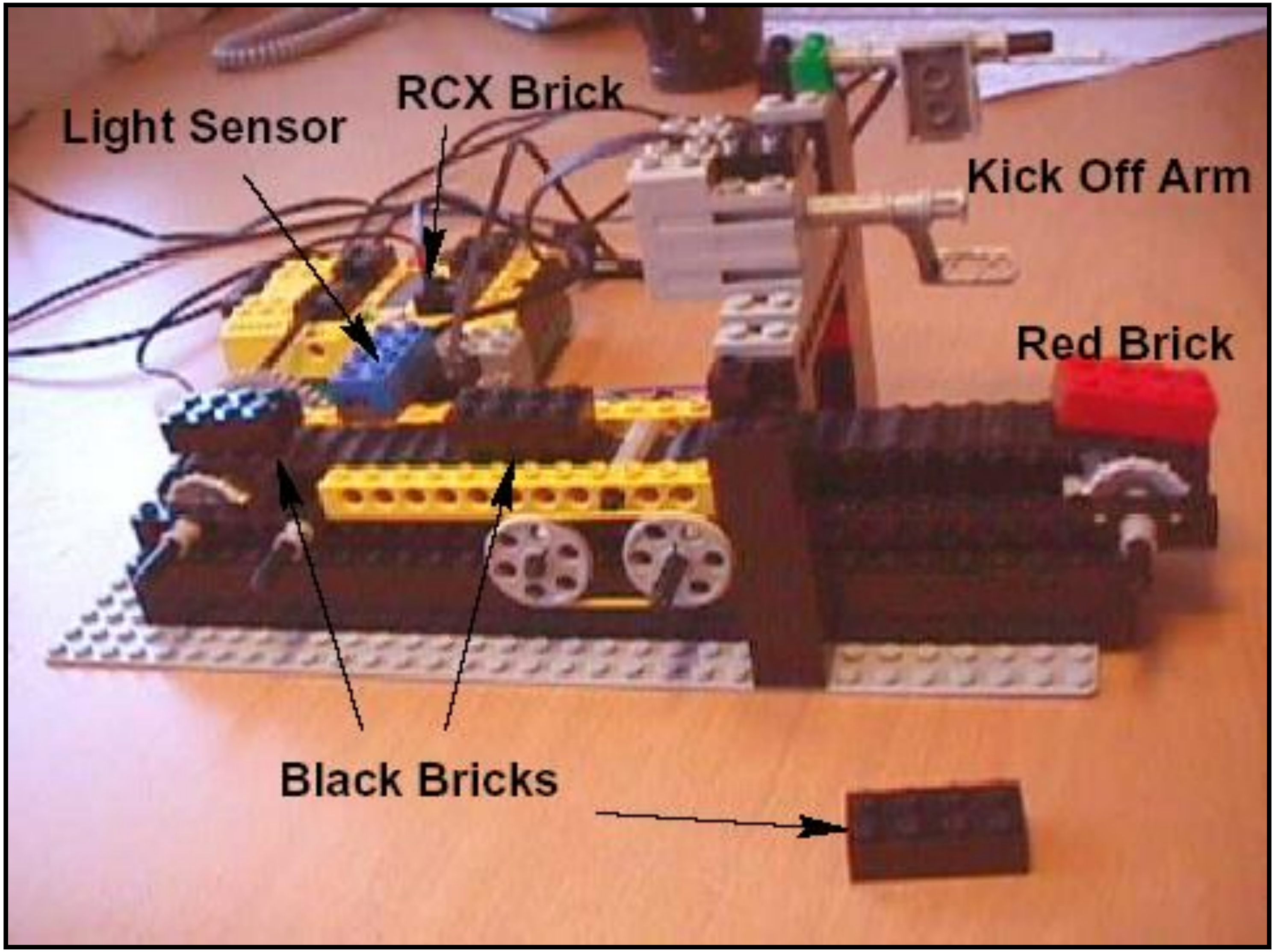
What files are generated?

- **Makefile**
- **brickos_kernel.c -** kernel code interpreting an automata structure.
- **brickos_system.h** - type and macros definitions.
- **brickos_interface.h** – BrickOS kernel API definition.
- **brickos_hooks.h** - definition of hooks executed at events in the kernel (used by the logging module).
- ***basename*_init.c** - a stub for user hardware initialization code.
- ***basename*_init.h** - API definition of the initialization code.
- ***basename*_global.h** - an empty file where the shared global variables are be defined, if they are not defined in the model.
- ***basename*.h** - generated definitions of the constants used in the code (e.g. number of transitions).
- ***basename*.c** - the main code generated including the tasks and the automata structure.

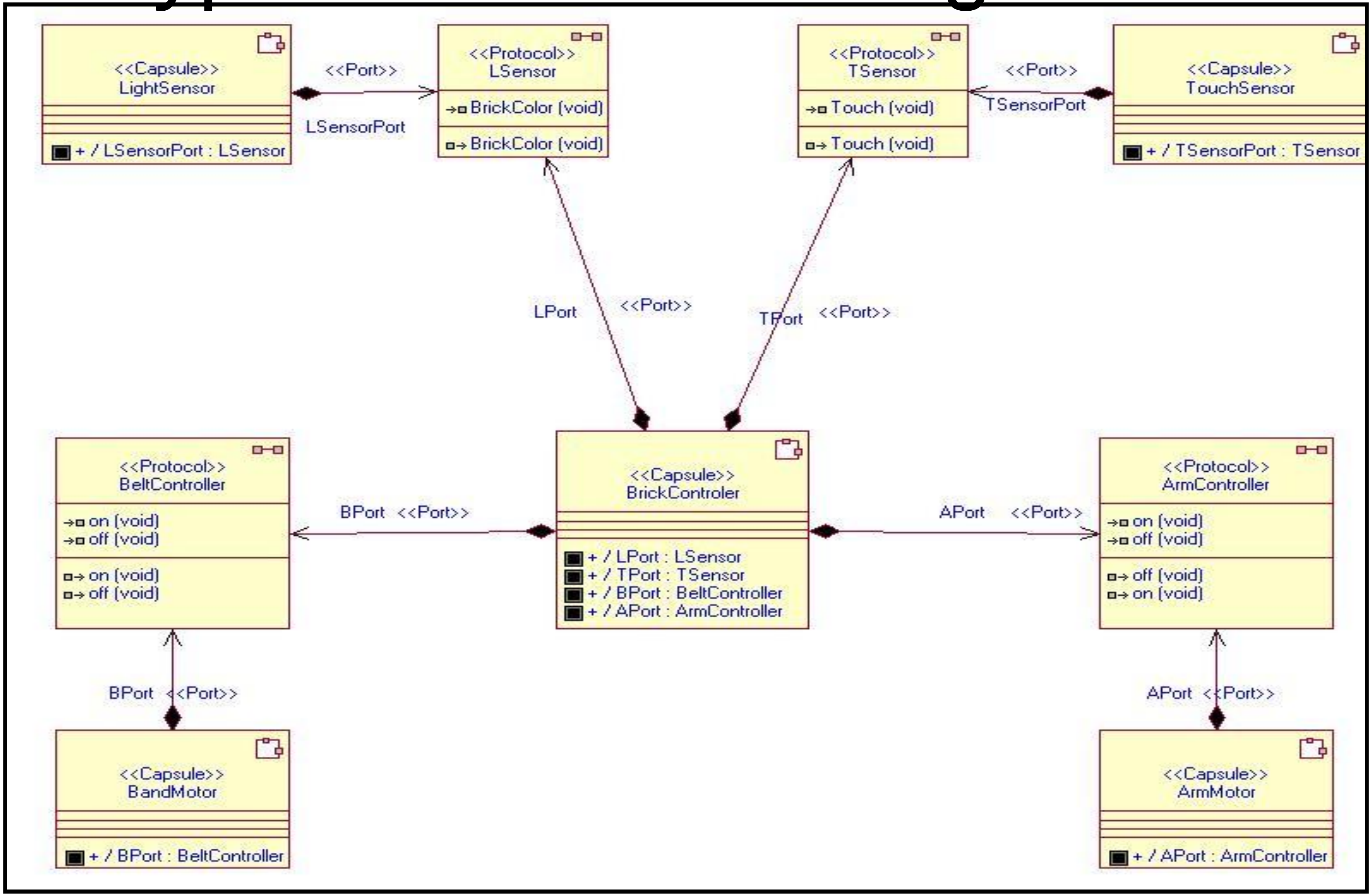
Brick Sorting/Parting Problem

- Sort (part) different colored 2x2 Lego bricks into bins containing bricks of the same color (the same number of bricks of a given color).
- Fix the maximum number of colors to be sorted.
- Limitations:
  - Hardware: Color sensor in RIS 2.0
  - Software: Limited memory available for program and data
- Example: 6-colored brick sorter (requires 3 engines): http://www.philohome.com/bricksorters/sorter3.htm
- Movie: bs3fast.mov

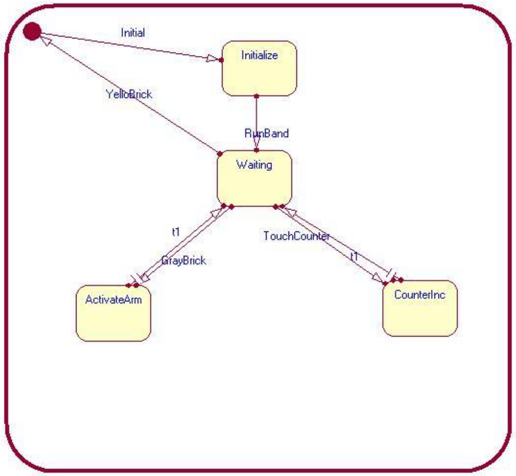
Brick Sorter - Typical Design



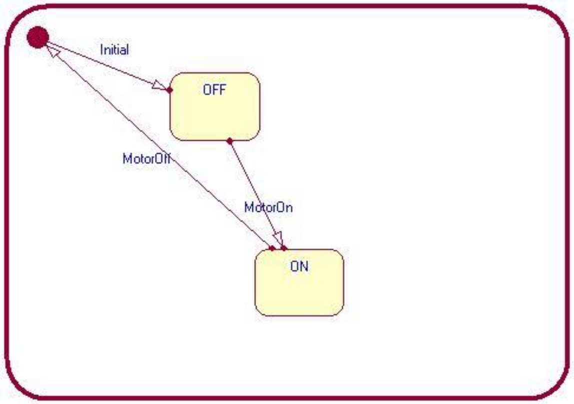
Typical RoseRT Design Model



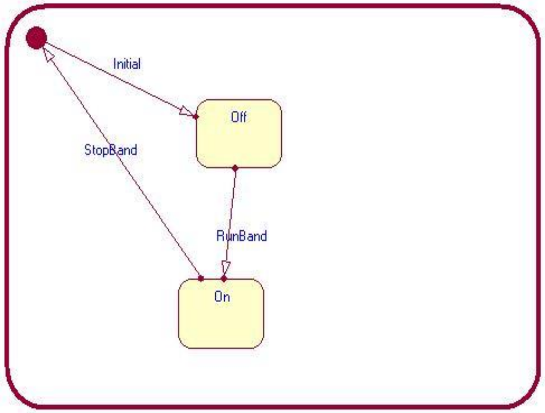
Design (State Diagrams)



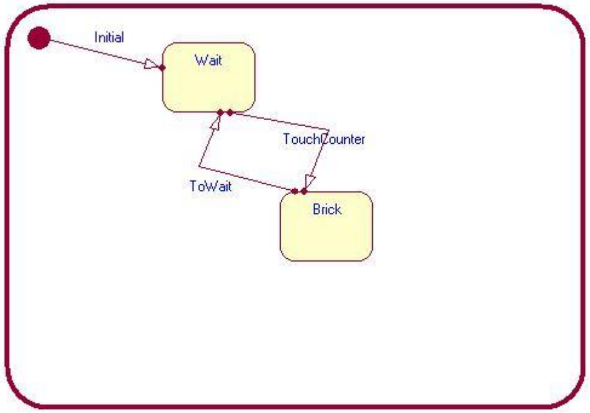
Controller



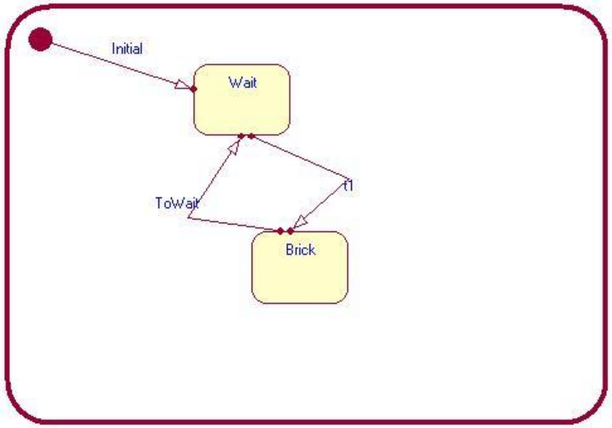
Arm Motor



Band Motor

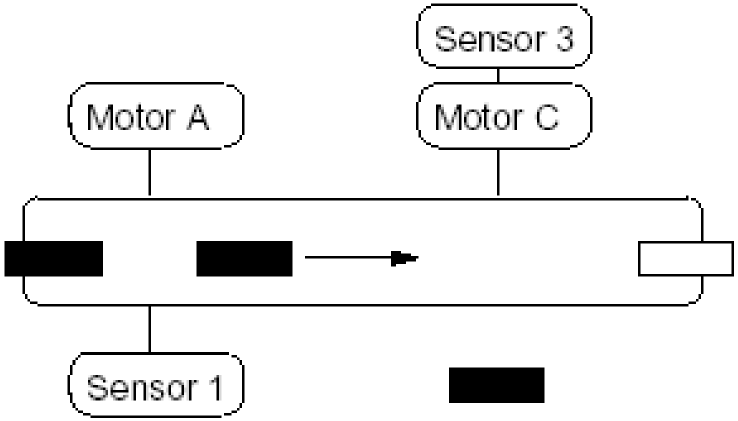


Touch Sensor



Light Sensor

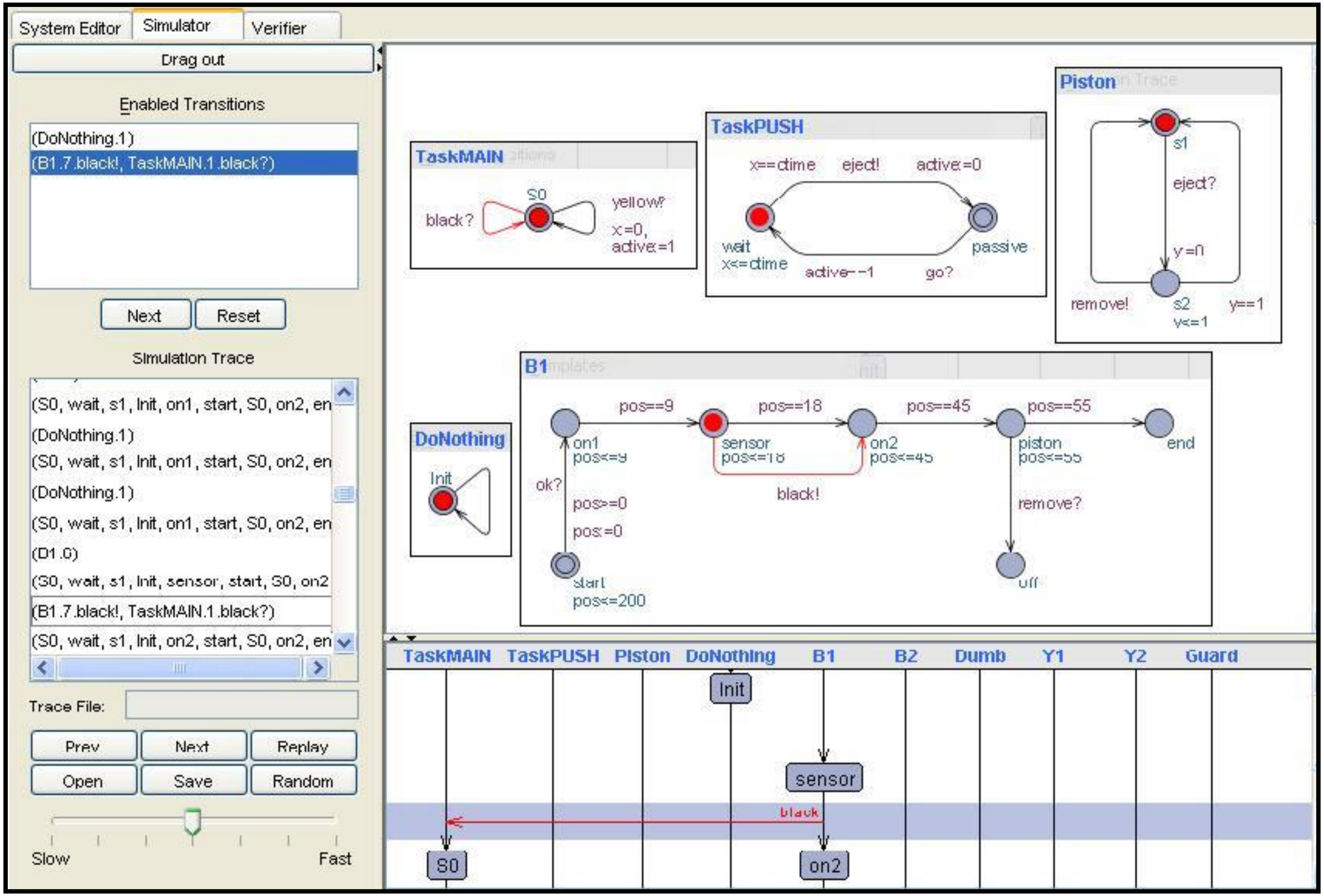
Defective NQC Code



```
int b=0, active1=0, active2=0;
int DELAY=25;
int LIGHT_LEVEL=42;
task main{
        Sensor(IN_1, IN_LIGHT);
        Sensor(IN_3, IN_SWITCH);
        Fwd(OUT_A,1);
        start kick_off;
        while(true){
                wait(IN_1<=LIGHT_LEVEL);
                if(b==0){
                        ClearTimer(1);
                        active1=1;
                }
                if(b==1){
                        ClearTimer(2);
                        active2=1;
                }
                b=-b+1;
                wait(IN_1>LIGHT_LEVEL);
        }
}
```

```
task kick_off{
        while(true){
                wait(Timer(1)>DELAY && active1==1);
                active1=0;
                Fwd(OUT_C,1);
                Sleep(6);
                Rev(OUT_C,1);
                wait(IN_3==1);
                Off(OUT_C);
                wait(Timer(2)>DELAY && active2==1);
                active2=0;
                Fwd(OUT_C,1);
                Sleep(6);
                Rev(OUT_C,1);
                wait(IN_3==1);
                Off(OUT_C);
        }
}
```

Verification Model (UPPAAL)



Enabled Transitions

(DoNothing.1)
(B1.7.black!, TaskMAIN.1.black?)

Simulation Trace

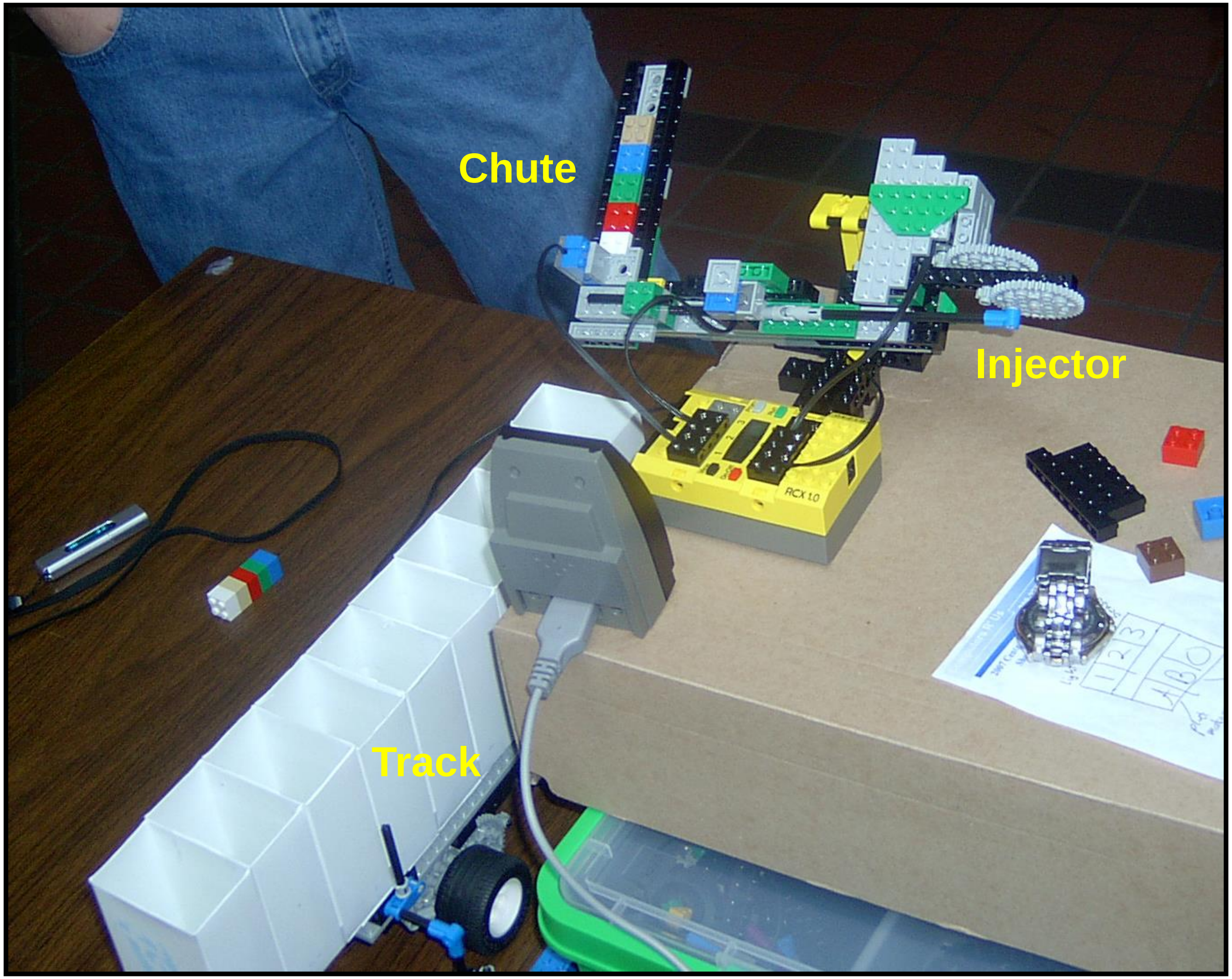
(S0, wait, s1, Init, on1, start, S0, on2, en
(DoNothing.1)
(S0, wait, s1, Init, on1, start, S0, on2, en
(DoNothing.1)
(S0, wait, s1, Init, on1, start, S0, on2, en
(D1.0)
(S0, wait, s1, Init, sensor, start, S0, on2
(B1.7.black!, TaskMAIN.1.black?)
(S0, wait, s1, Init, on2, start, S0, on2, en

Trace File:

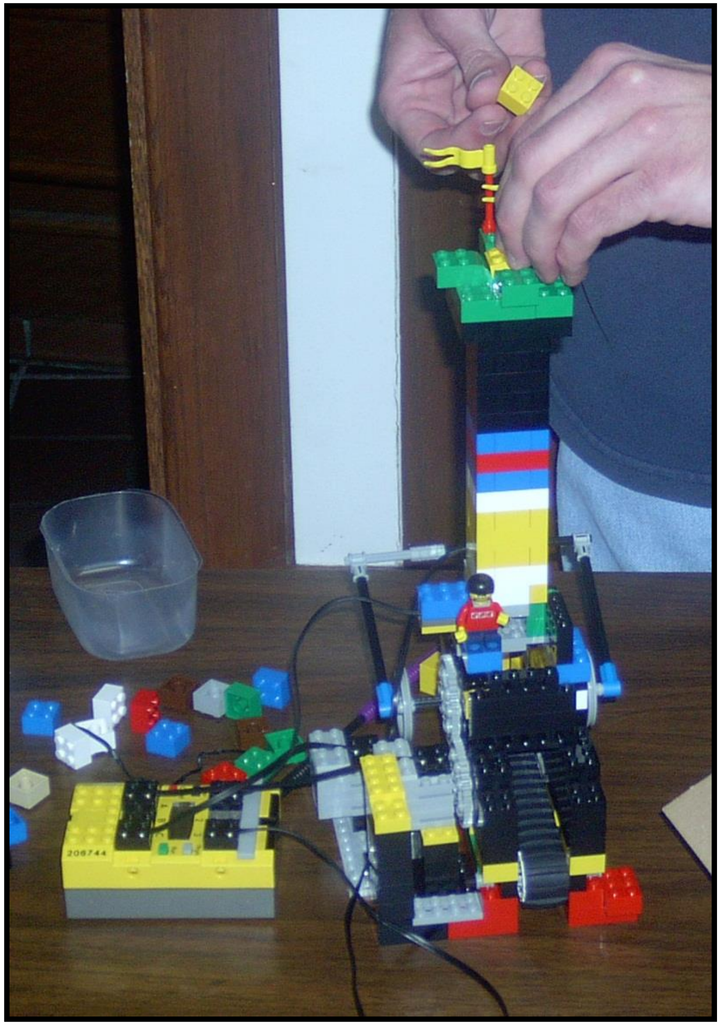
Properties to Verify

- **Safety** properties; e.g., bricks are sorted correctly:
  - A[ ] (not B1.end)
  - A[ ] (not R1.off)
- **Liveness** properties; e.g., bricks eventually leave the system:
  - R1.on1 --> R1.end
  - B1.on1 --> B1.off

Precise Brick Sorter



Optional (Friendly) Competition



Development Tools

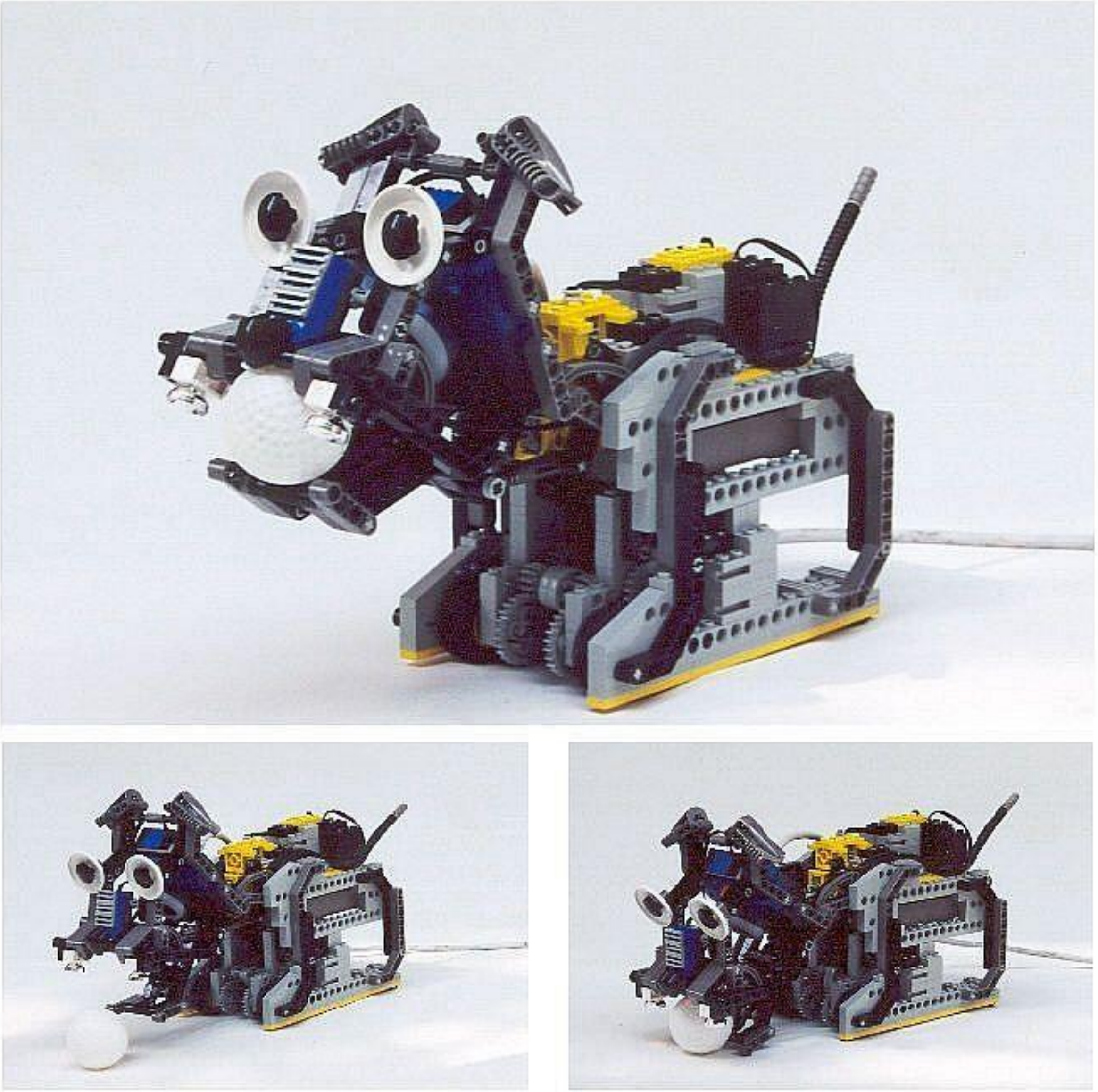
- Development Environment/Host
  - Cywin/Windows: http://www.cygwin.com/
  - Linux
- IR Tower Driver (from LegoMindstormsSDK)
  - https://online.ksu.edu/COMS/player/content/CIS_721/content/modules/Programs/Tower.zip
- Cross-Compiler H8/300
  - Build OS firmware - brickOS.srec, download to brick using util/firmdl3 boot/brickOS.srec.
  - Build Application executable - <filename>.lx, download to brick using util/dll demo/<filename>.lx.
  - More details: http://brickos.sourceforge.net/docs/INSTALL-cygwin.html
  - Patch to add support for current versions of brickos-0.9.0 and gcc 4.0.2,etc. http://csd.informatik.uni-oldenburg.de/~hoenicke/rcx/brickOS.html
- BrickOS Emulator (BrickEmu)
  - http://csd.informatik.uni-oldenburg.de/~hoenicke/rcx/

Design/Analysis Tools

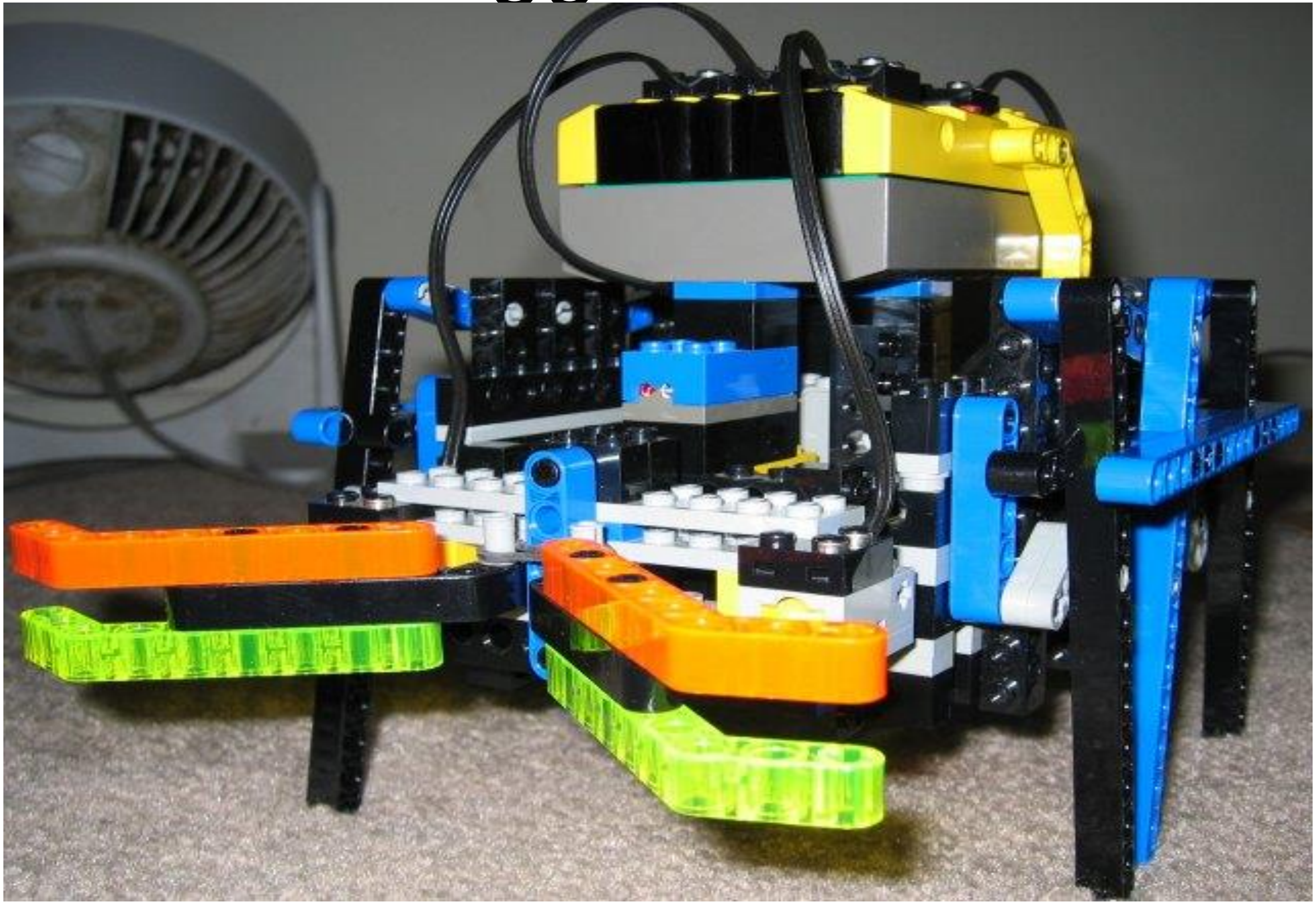
- Rational Rose Real-Time
- Times Tool
  - See www.timestool.com
  - Automatically generates brickOS source, but code is still a bit buggy :-(.
  - Includes simulator, schedulability tester and verification tool - UPPAAL.

Other Robots

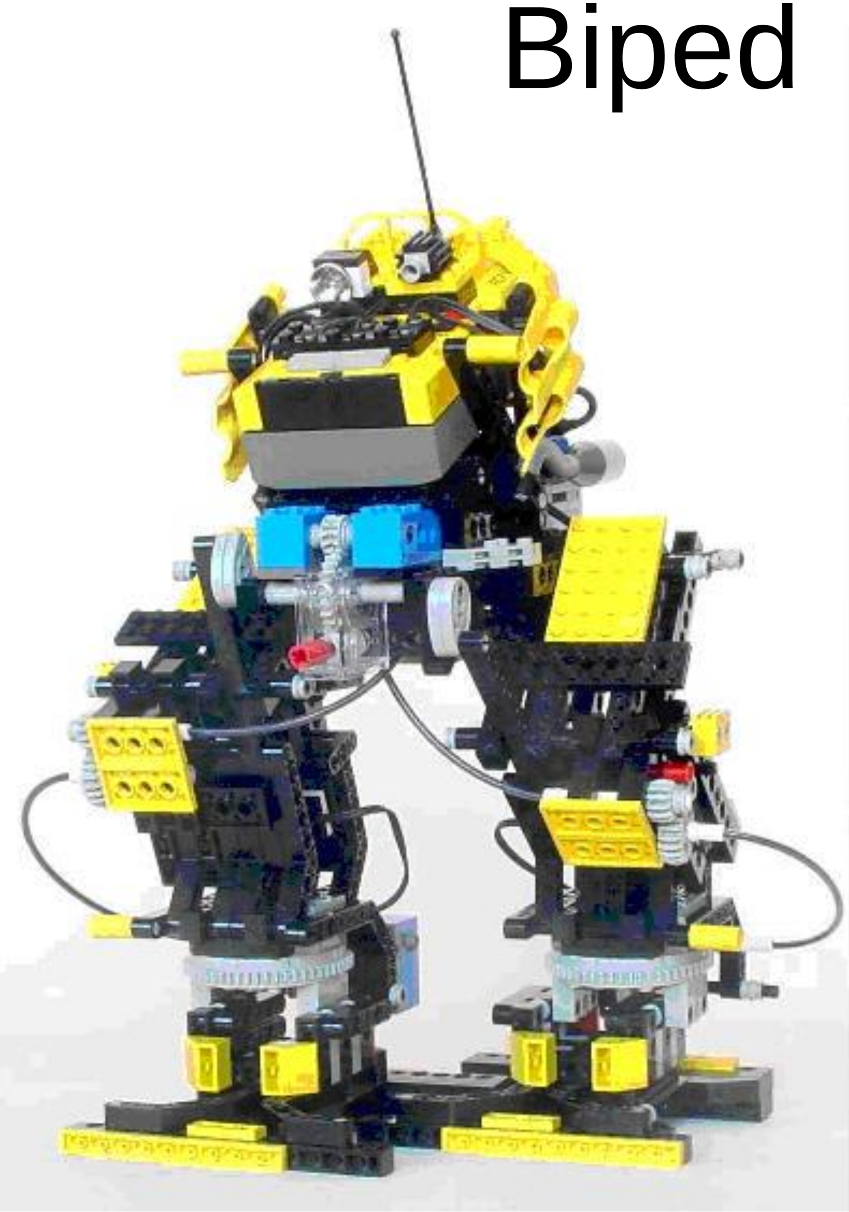
MIBO

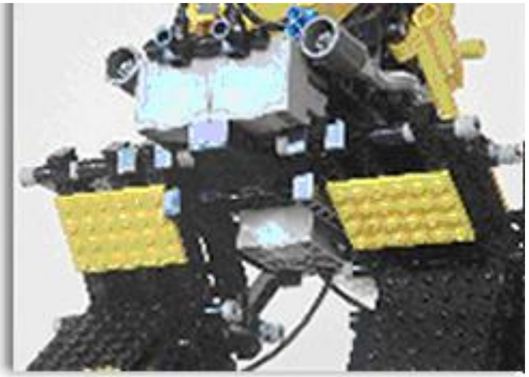


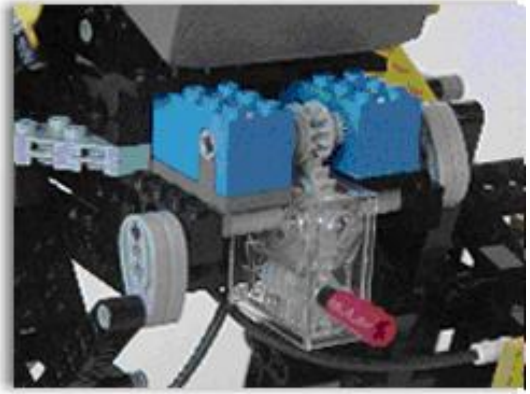
Six-Legged Walker

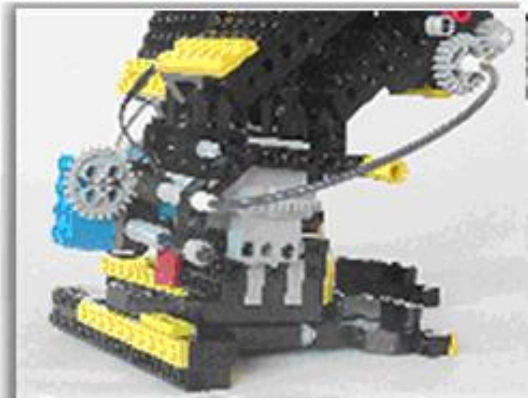


Biped

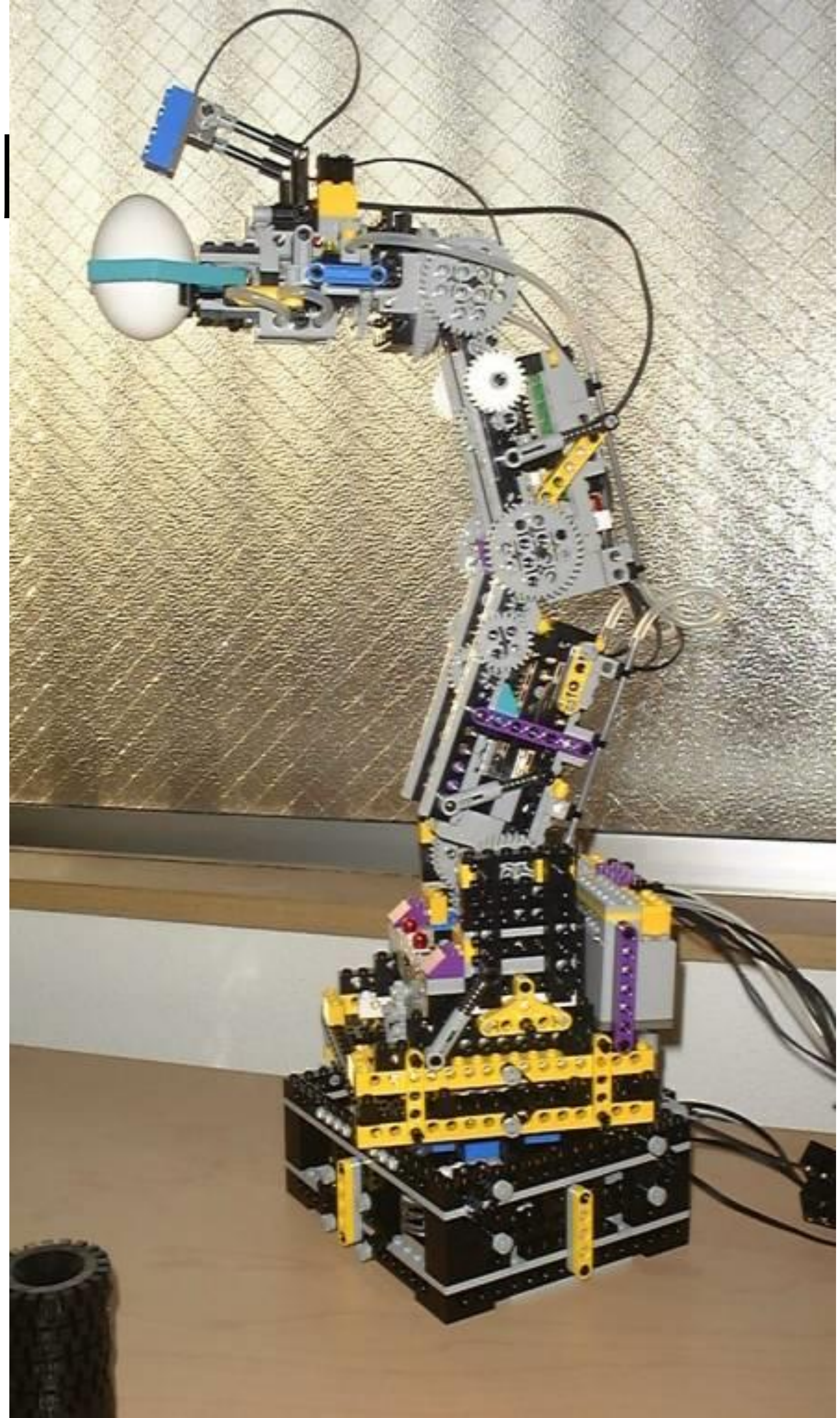




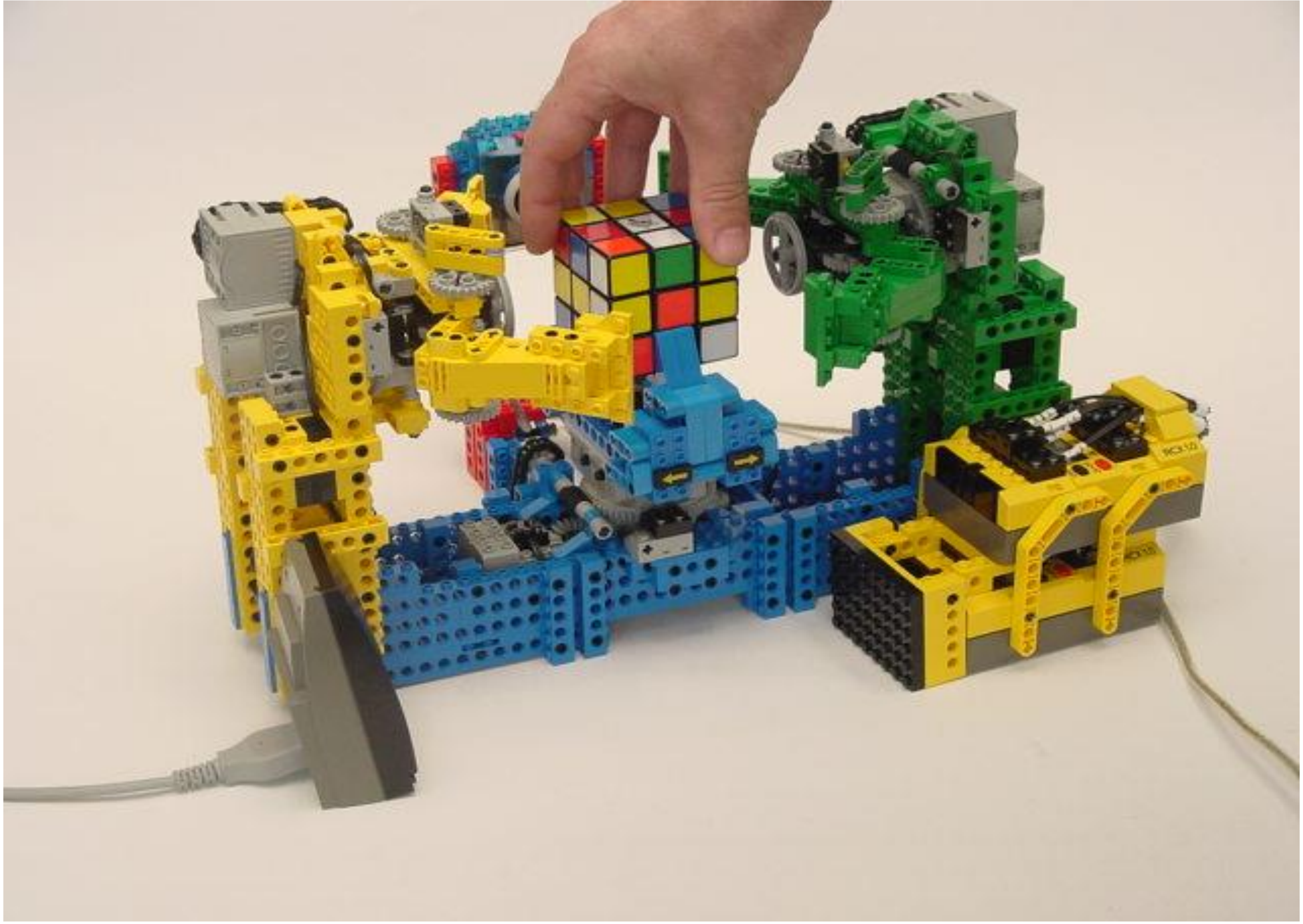




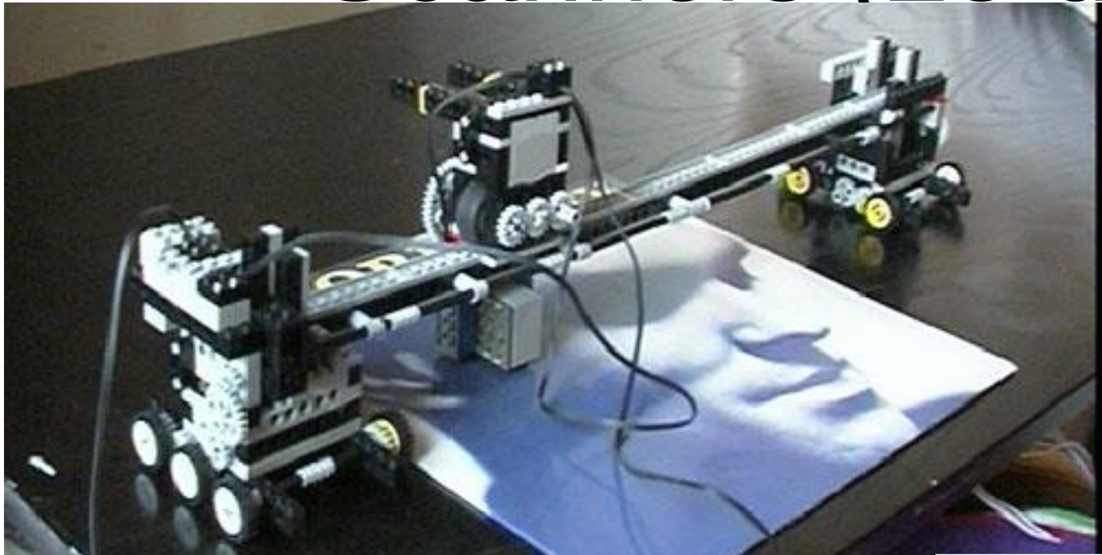
Rob

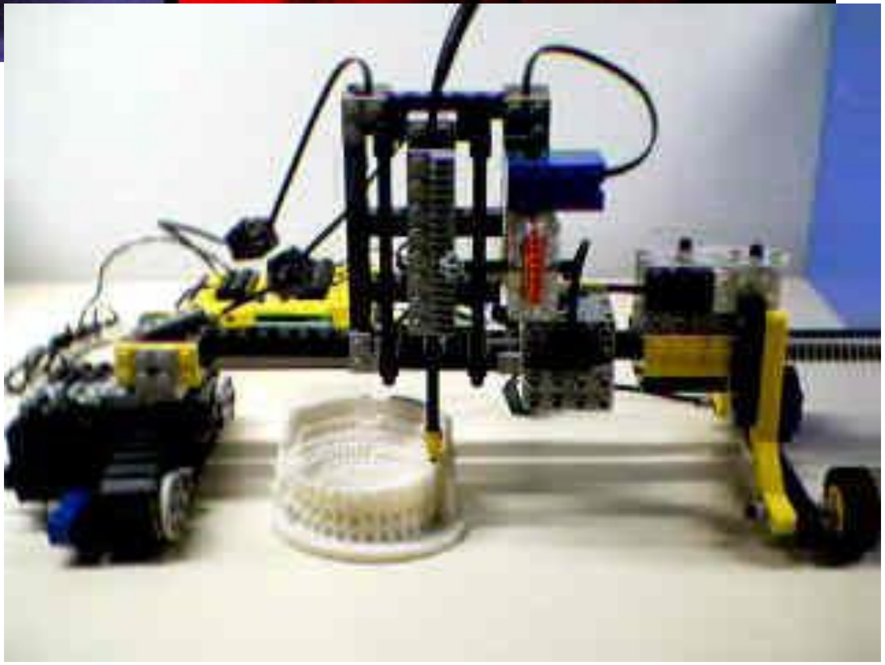


Cube Solver

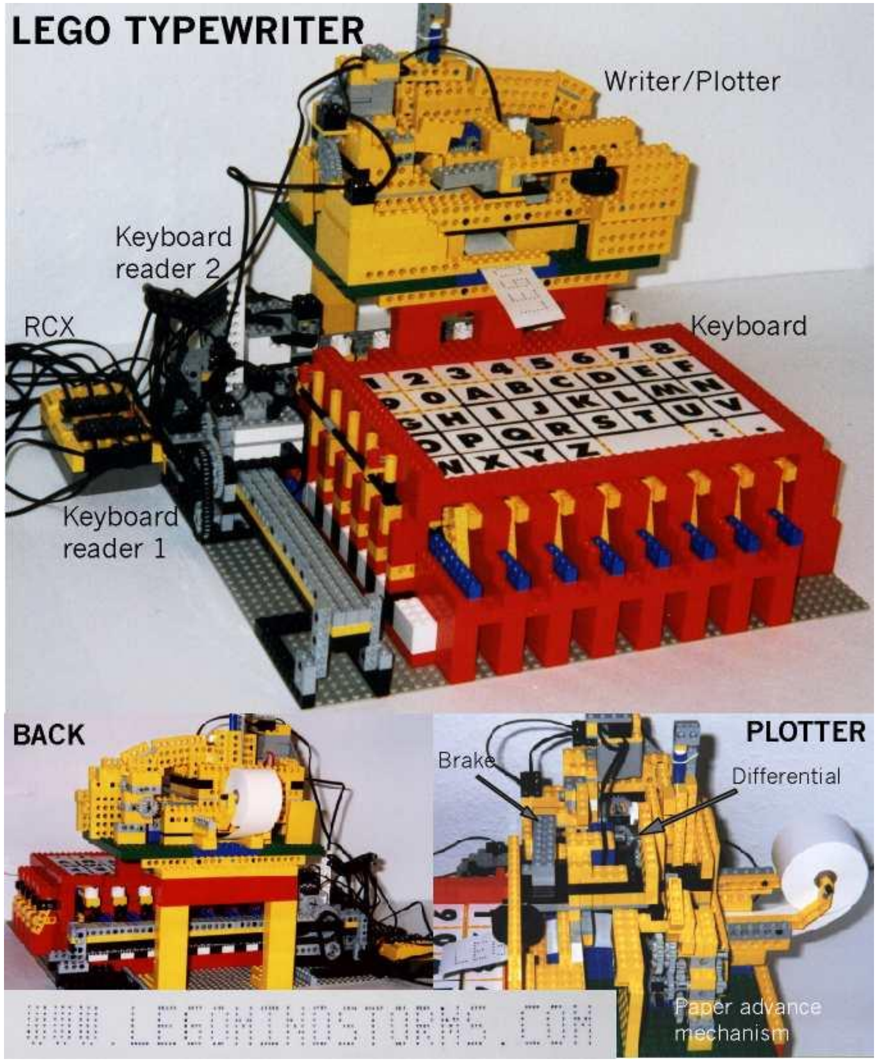


Scanners (25 dpi / 3D)

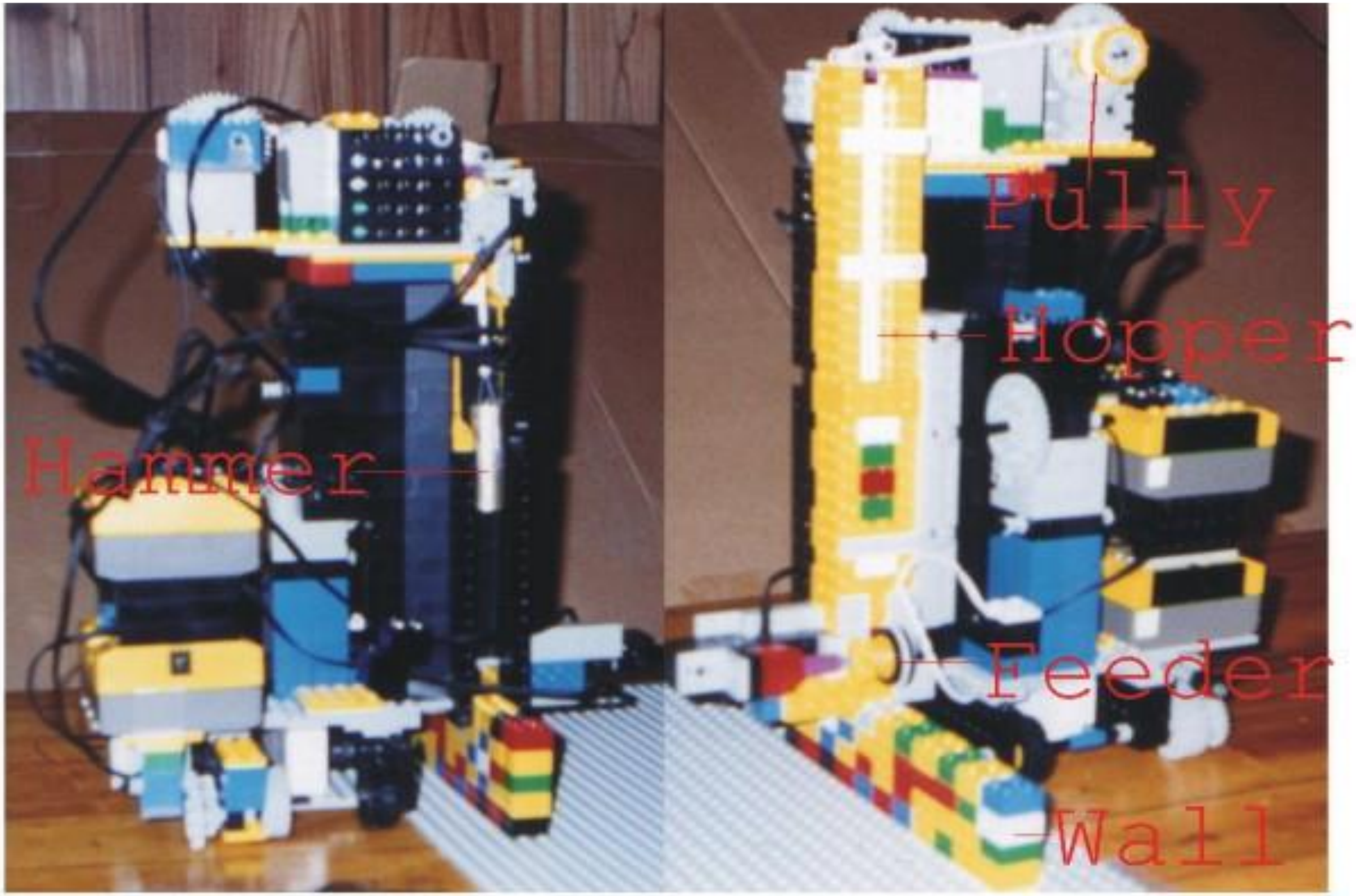




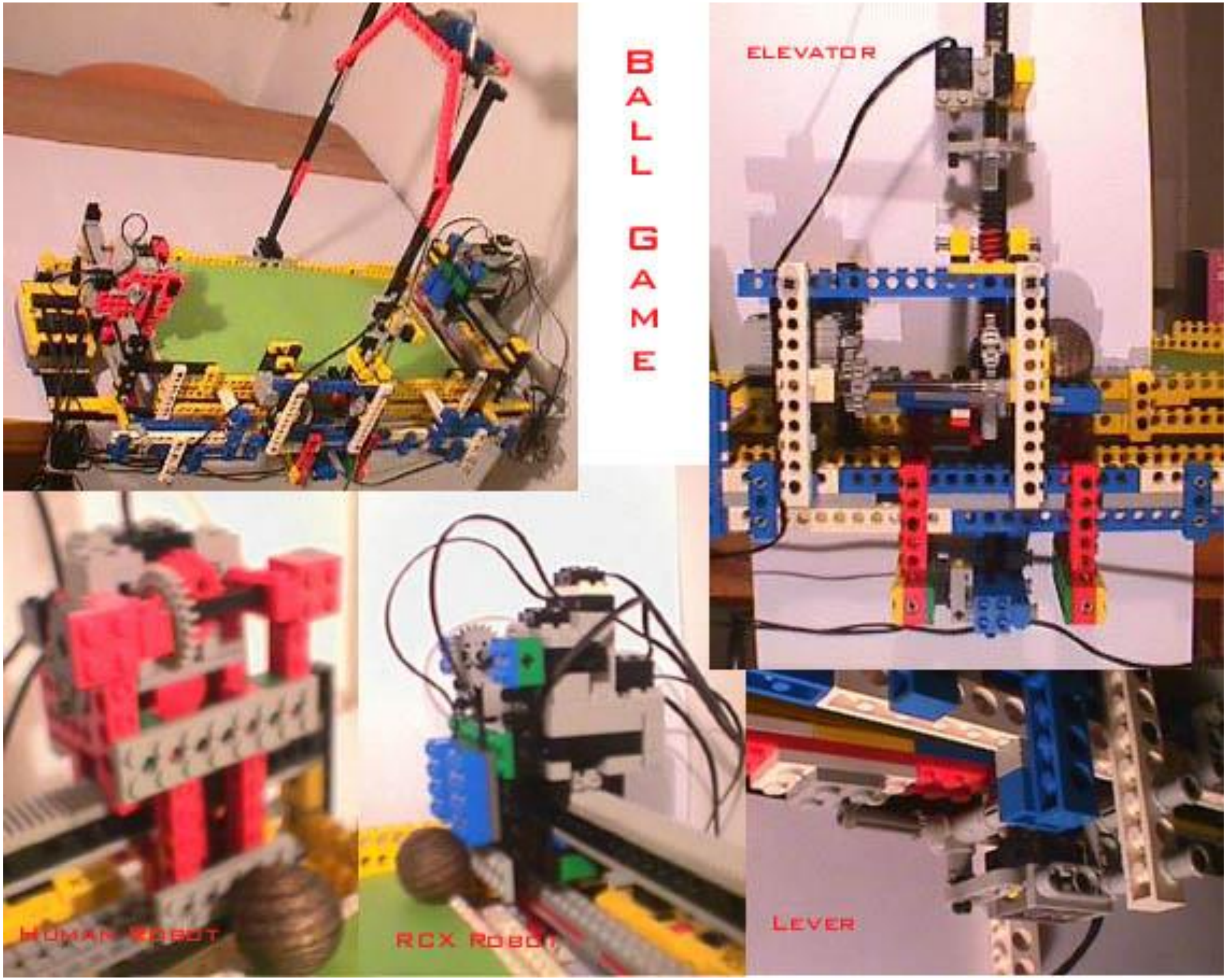
Typewriter


LEGO TYPEWRITER
Writer/Plotter
Keyboard reader 2
RCX
Keyboard
Keyboard reader 1
BACK
PLOTTER
Brake
Differential
Paper advance mechanism
WWW.LEGOMINDSTORMS.COM

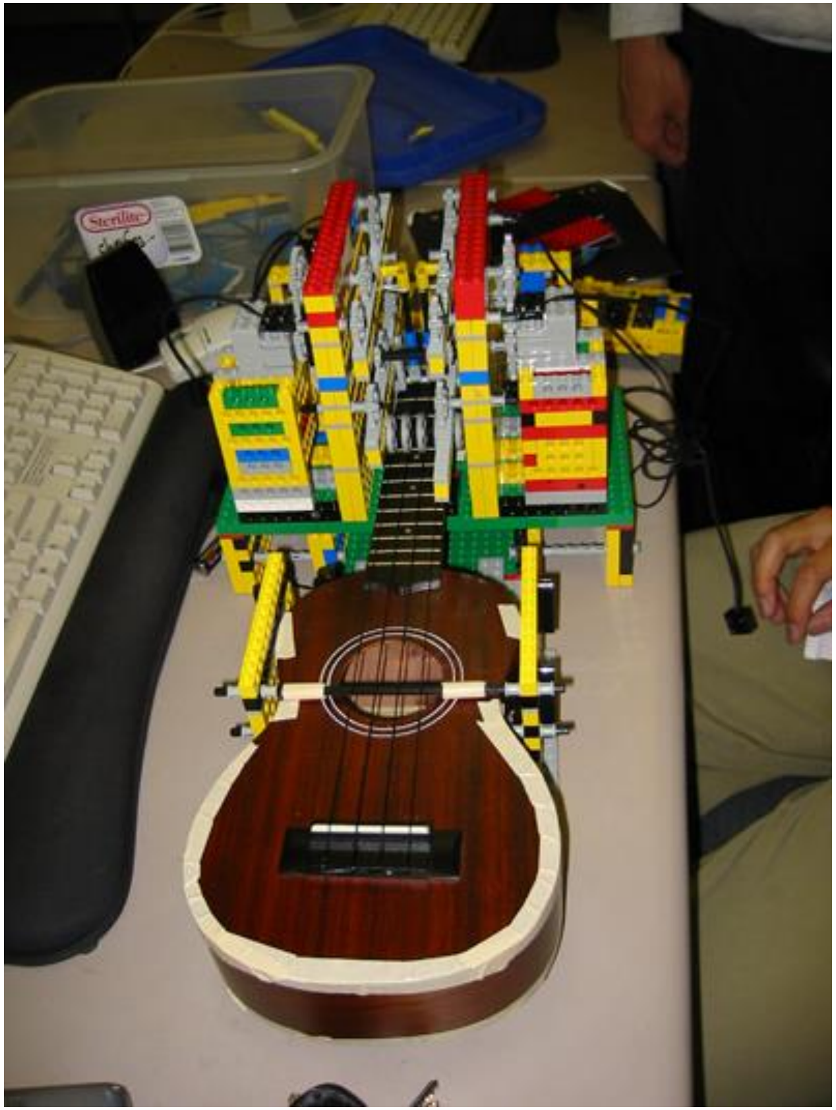
Brick Layer



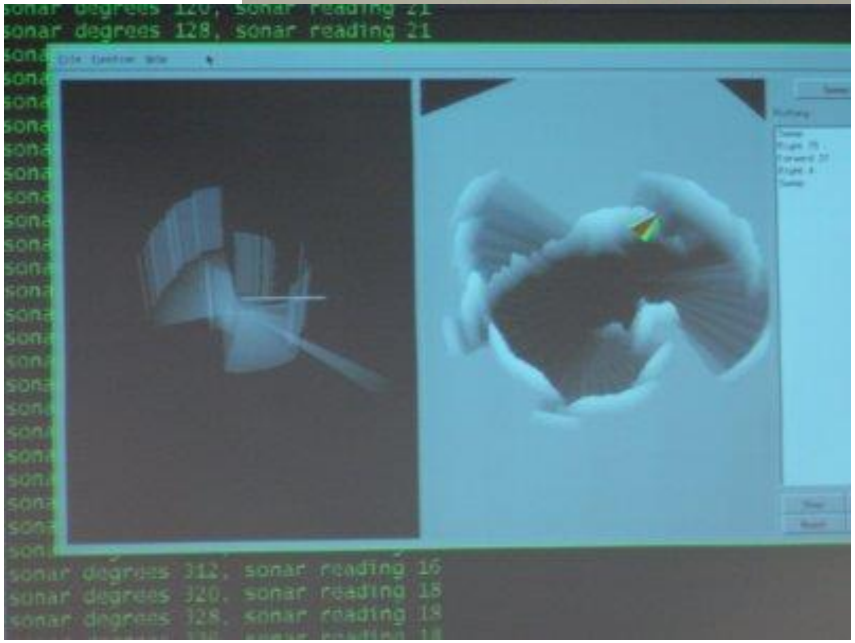
Ball Game



Ukulele Player



Sonar



```
sonar degrees 120, sonar reading 21
sonar degrees 128, sonar reading 21
sonar degrees 312, sonar reading 16
sonar degrees 320, sonar reading 18
sonar degrees 328, sonar reading 18
```

Jitter - First Mindstorm Robot In Space