An Overview on Code Synthesis and Runtime Verification

A Broad Vision of our Goals and Achievements

- **Gergely Pintér**

Budapest University of Technology and Economics
Department of Measurement and Information Systems

Key Achievements Presented in my Thesis

- A *formal semantics* for UML state machines
- A method for the *automatic implementation* of UML state machines
- Two *verification methods* for the runtime evaluation of state-based behavior

Key Achievements Presented in my Thesis

- A *formal semantics* for UML state machines
- A method for *mentation* of UML state
- Two *verificati* untime evaluation of state-based behavior

Semantics

- What does a complex statechart actually *mean?*

Key Achievements Presented in my Thesis

- A *formal semantics* for UML state machines
- A method for the *automatic implementation* of UML state machines
- Two *verifi* untime evaluation r

Code Synthesis

- How to *implement* the control structure described by a statechart?
- Demonstrated by code generation for a *μC based device.*

K
Pres s

- A *formal* chines
- A metho *tation*
  of UML st
- Two *verification methods* for the runtime evaluation of state-based behavior

Runtime Verification

- How to *check* that the application actually *behaves according* to its *specification?*
- How to add *extra timing related requirements* to a statechart and *check* them during runtime?

Key Achievements Presented in my Thesis

- A *formal semantics* for UML state machines
- A method for the *automatic implementation* of UML state machines
- Two *verification methods* for the runtime evaluation of state-based behavior

Key Achievements Presented in my Thesis

- A *formal semantics* for UML state machines
- A method for the *automatic implementation* of UML state machines
- Two *verification methods* for the runtime evaluation of state-based behavior

Research Focus:
- Unambiguous **specification**,…
- …automatic **implementation** and…
- …runtime **verification** of…
  complex control structures

Pr

- A for ... es
- A me ... on of UN
- Two ... evalua

When talking about "Control Structures"…

- We are talking about control concept of *programming and modeling languages* (e.g., do-while loops, if-else branches, functions, even processes or threads)…
- …and *not* process control concepts like PID controllers, ZOHs, etc.
- …i.e., *"how C/C++/Java/etc. statements are organized into a program"*

Research Focus:
- Unambiguous **specification**,…
- …automatic **implementation** and…
- …runtime **verification** of…

  complex control structures

Let's Focus a Bit on Automatic Code Synthesis…

- Originally aimed benefits:
  - Substitution of a *labor-intensive error-prone task* with a proven correct *automatic tool*
    - *Reduction of development costs*
      - Human effort, time, maintenance cost
    - *Increase in code quality*
      - Complex, hard to understand parts generated automatically
      - Human focus on key tasks (i.e., atomic activities), boring labor-intensive maintenance of the control structure carried out by a tool

Let’s Focus a Bit on Automatic Code Synthe .

- Originally aimed bene
  - Substitut

…derstand parts generated
…ally

  - Human focus on key tasks (i.e., atomic activities), boring labor-intensive maintenance of the control structure carried out by a tool

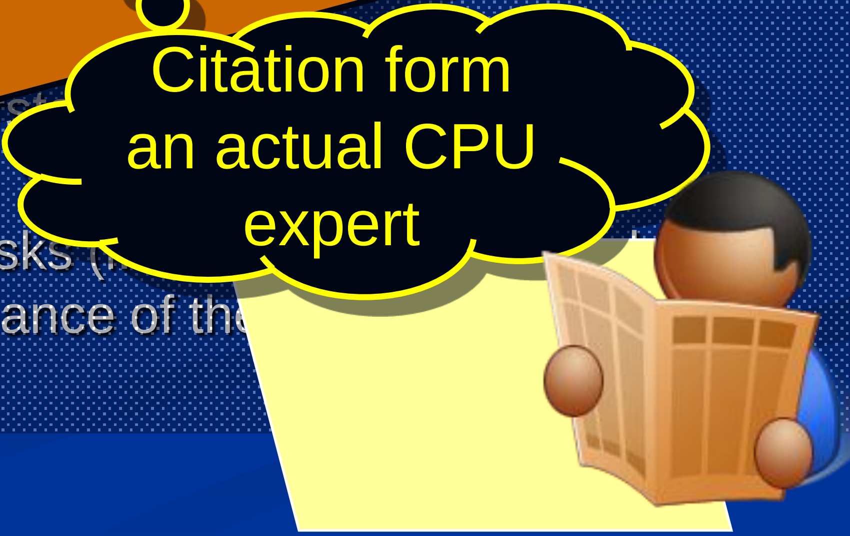
These goals are important indeed, but there is a much broader horizon ahead of us:

- We are using ever more computing cores in devices,…
- …these cores may be dedicated to various goals and…
- …are frequently idle due to “badly written programs” …
- …while consuming energy

Let’s Focus a Bit on Automatic Code Synthesi .

- Originally aimed bene
  - Substitut
  - ...derst
  - ...ally
  - Human focus on key tasks (
    labor-intensive maintenance of the
    carried out by a tool

These goals are important indeed, but there is a much broader horizon ahead of us:
- We are using ever more computing cores in devices,…
- …these cores may be dedicated to various goals and…
- …are frequently idle due to “badly written programs” …
- …while consuming energy



Let's Focus a Bit on Auto… thesis…

But *why should a programmer* understand the inner details of a multi-core CPU?
(That may have not even been manufactured yet…)

These goals … a much broader horizon…
- We are using ever more cores in devices,…
- …these cores may be dedicated to various goals and…
- …are frequently idle due to "badly written programs" …
- …while consuming energy

- Human focus on key tasks (…
labor-intensive maintenance of the…
carried out by a tool

Citation form an actual CPU expert

Let's Focus a Bit on Automatic Code Synthesi...

- Origi
  -

These ... s a much broad ... devices,... goals and... "... programs" ...
- We a
- ...these
- ...are frequ
- ...while consuming

Idea

Extend visual control models with information about the most beneficial platform and *do the mapping automatically* by the control code synthesis tool

...derstand parts generated ...ally

- Human focus on key tasks (i.e., atomic activities), boring labor-intensive maintenance of the control structure carried out by a tool

Presentation Structure

Wide Context and Future Research Goals

Achievements Until Now

Demonstration

Presentation Structure

Wide Context and Future Research Goals

Achievements Until Now

Demonstration

Presentation Structure

18

Warning: The next part of the presentation is mostly brain storming about *future research activities*. Do not expect proven, fine-tuned solutions! Our goal here is to give a broad overview on our *planned work* and *discuss* our and *your ideas* concerning the subject.

PARENTAL
ADVISORY
EXPLICIT CONTENT

Control Hierarchy

State-Transition Model

Detailed Activity Model

Process/Thread Model

Methods, Statements

Machine Instructions

Micro-Instructions

Control Hierarchy: Typical Description Forms

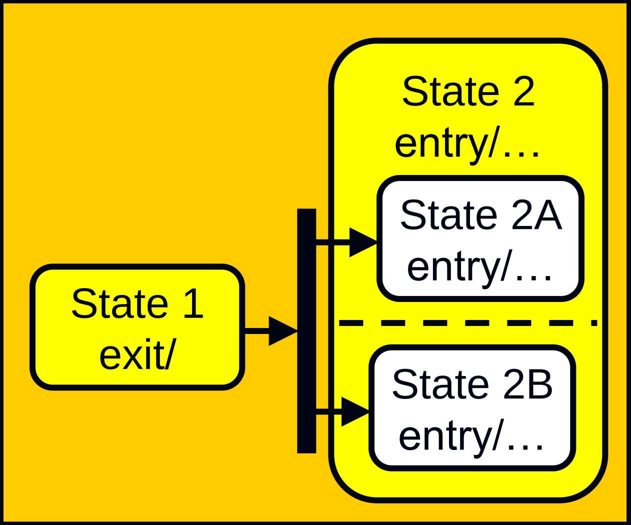
- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

Control Hierarchy: Typical Description Forms

- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

State-Transition Model

- UML statecharts
- Matlab/Stateflow diagrams
- Harel statecharts, etc.

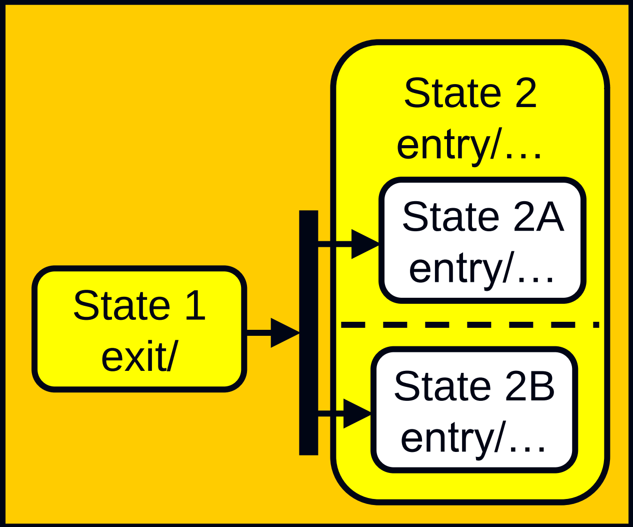


Control Hierarchy: Typical Description

Top-level organization of activities

- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

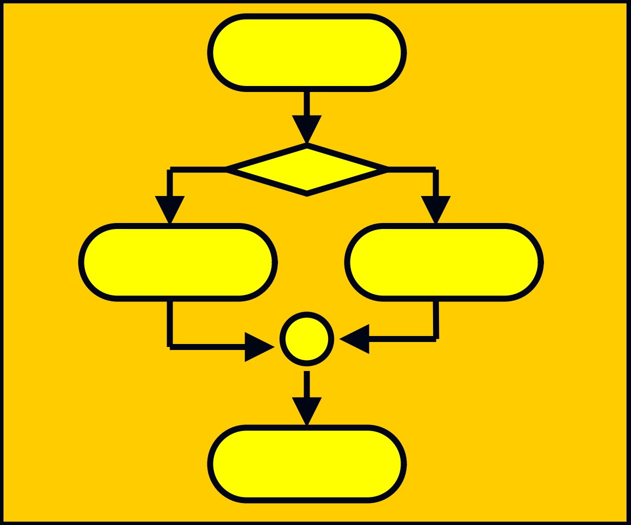
State-Transition Model
- UML statecharts
- Matlab/Stateflow diagrams
- Harel statecharts, etc.



Control Hierarchy: Typical Description Forms

- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

Detailed Activity Model
- UML activity diagrams
- Matlab/Stateflow diagrams



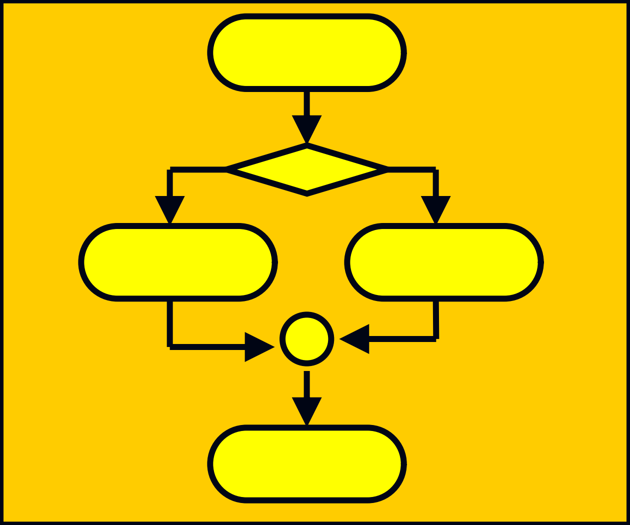
Control Hierarchy: Typical Description

Detailed specification of activities

- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

Detailed Activity Model

- UML activity diagrams
- Matlab/Stateflow diagrams



Control Hierarchy: Typical Description Forms

- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

Process/Thread Model

- Top-level structure of the source code (approximately)

```
src/
  signal_processing/
    Makefile
    some_dsp_library.c
  io/
    Makefile
    some_io_library.c
  Makefile
  main.c
```

Control Hi
Typical Descri

Assignment of activities to *program images thus selection of target architecture* (*Makefiles* are shown for a reason)

State-Transition Model

Detailed Activity Model

Process/Thread Model

Methods, Statements

Machine Instructions

Micro-Instructions

Process/Thread Model

- Top-level structure of the source code (approximately)

```
src/
  signal_processing/
    Makefile
    some_dsp_library.c
  io/
    Makefile
    some_io_library.c
  Makefile
  main.c
```

Control Hierarchy: Typical Description Forms

- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

Methods, Statements

- Source code of methods

```
#ifndef SOME_IO_LIBRARY
#define SOME_IO_LIBRARY

int
some_io_method() {
   for (…)
     if (…) {
       // …
     } else {
       // …
     }
}
```

Control Hierarchy: Typical Description

Implementation of activities in a high-level language

- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

Methods, Statements

- Source code of methods

```
#ifndef SOME_IO_LIBRARY
#define SOME_IO_LIBRARY

int
some_io_method() {
  for (…)
    if (…) {
      // …
    } else {
      // …
    }
}
```

Control Hierarchy: Typical Description Forms

- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

Machine Instructions

- Machine (assembly) language sources

```
some_io_method:
   pushq %rbp
   movq %rsp, %rbp
   subq $16, %rsp
   movl %edi, -4(%rbp)
   movl $0, -8(%rbp)
   movl -8(%rbp), %eax
   cmpl -4(%rbp), %eax
   jge .L4
```

Control Hierarchy: Typical Description Forms

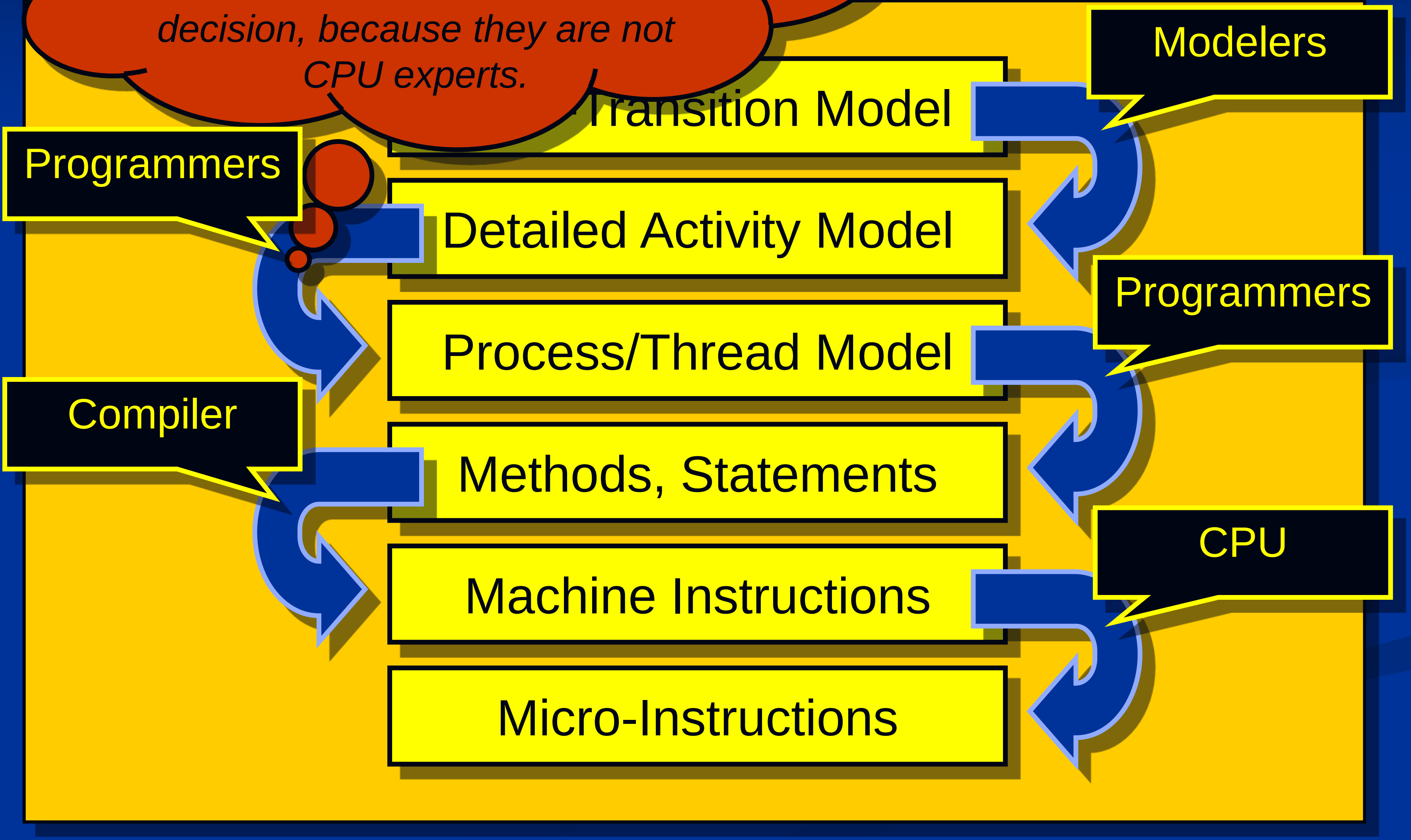
- State-Transition Model
- Detailed Activity Model
- Process/Thread Model
- Methods, Statements
- Machine Instructions
- Micro-Instructions

Micro-Instructions

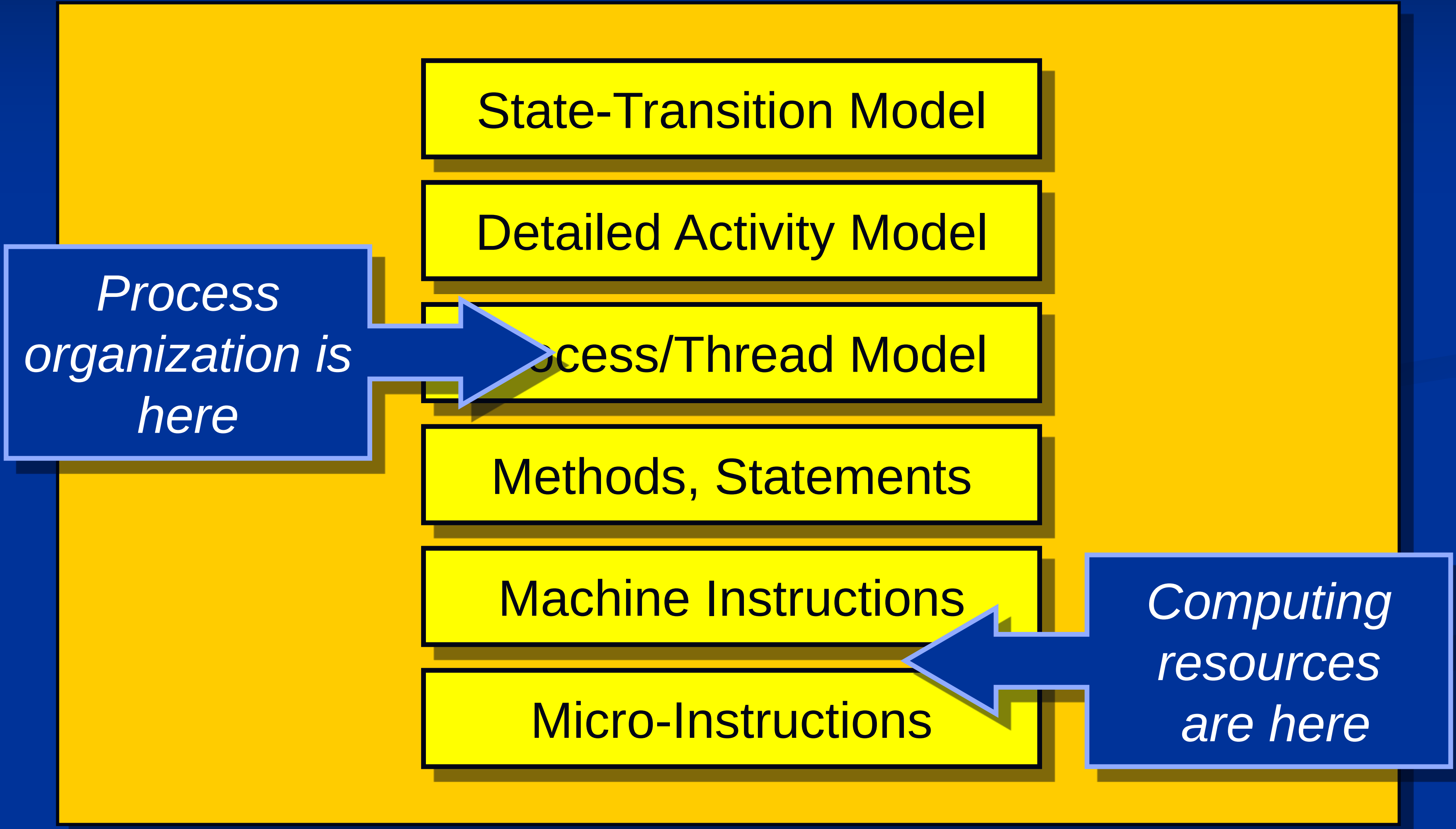
- Implementation of machine instructions in the CPU
- Multiple pipelines, ALUs, caches, etc.

archy: Jobs

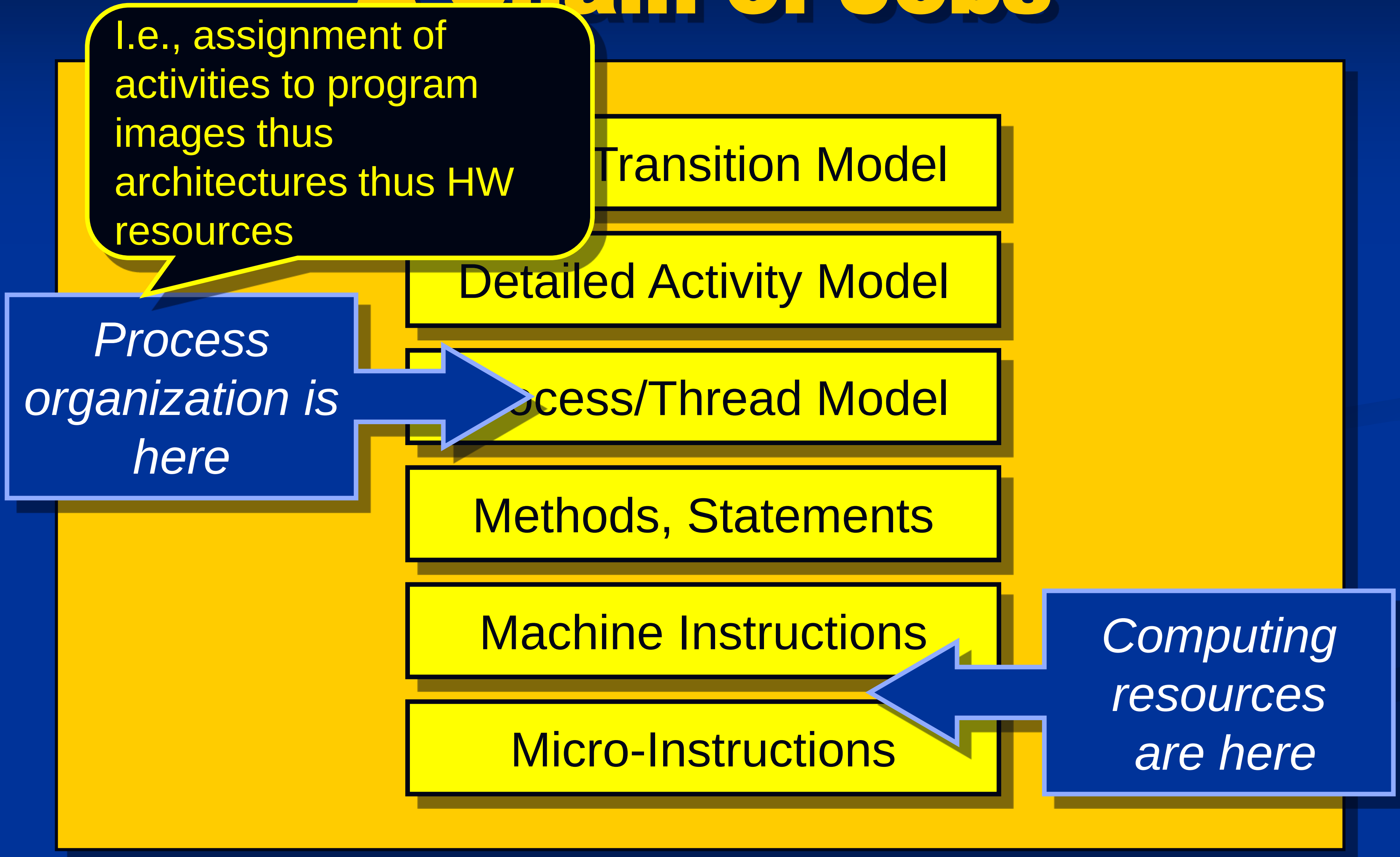
Programmers (architects or lead programmers) had to decide about process-core allocation. *This should not be their decision, because they are not CPU experts.*



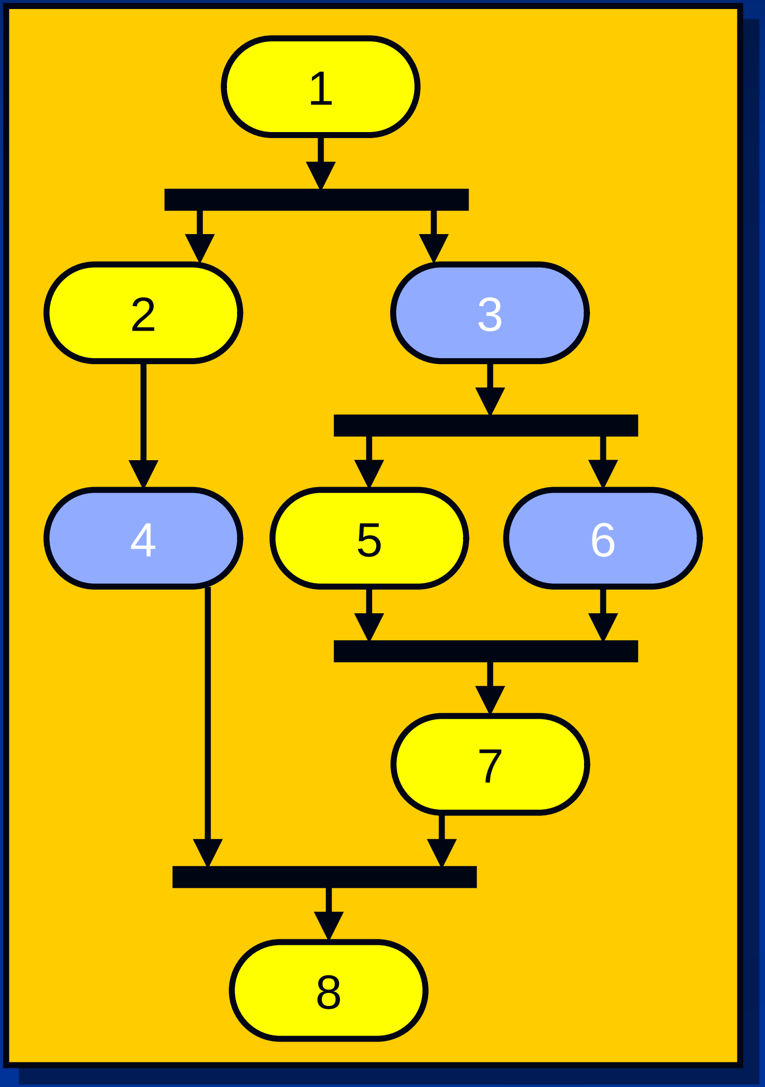
Control Hierarchy: A Chain of Jobs

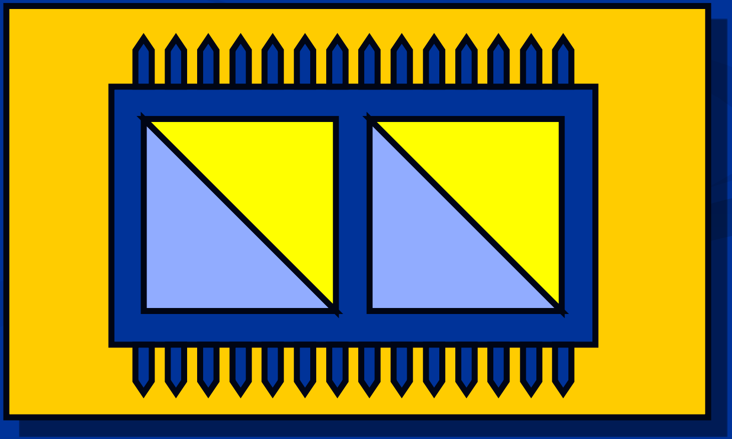


Control Hierarchy: A Chain of Jobs


I.e., assignment of activities to program images thus architectures thus HW resources
Transition Model
Detailed Activity Model
cess/Thread Model
Methods, Statements
Machine Instructions
Micro-Instructions
Process organization is here
Computing resources are here

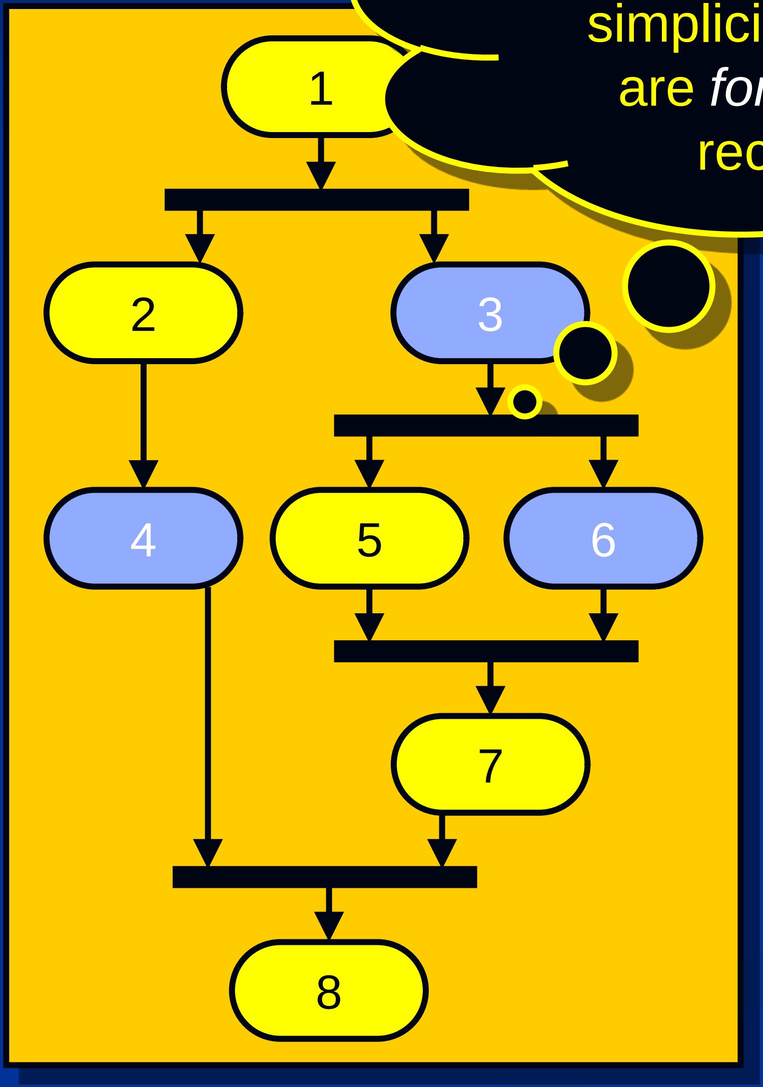
Effects of Task-Core Assignment



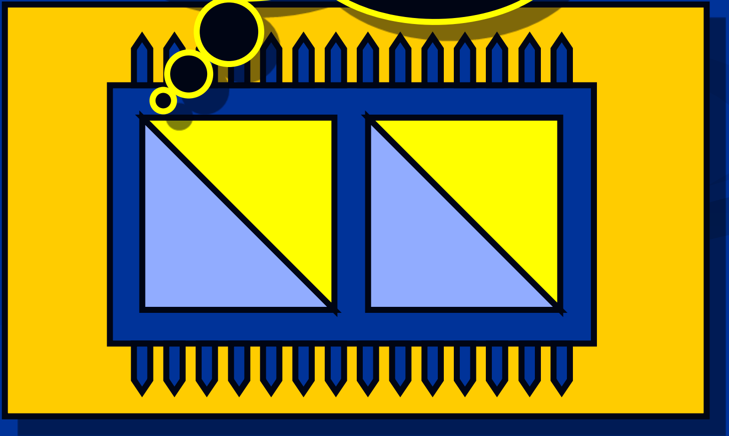


Effects of T

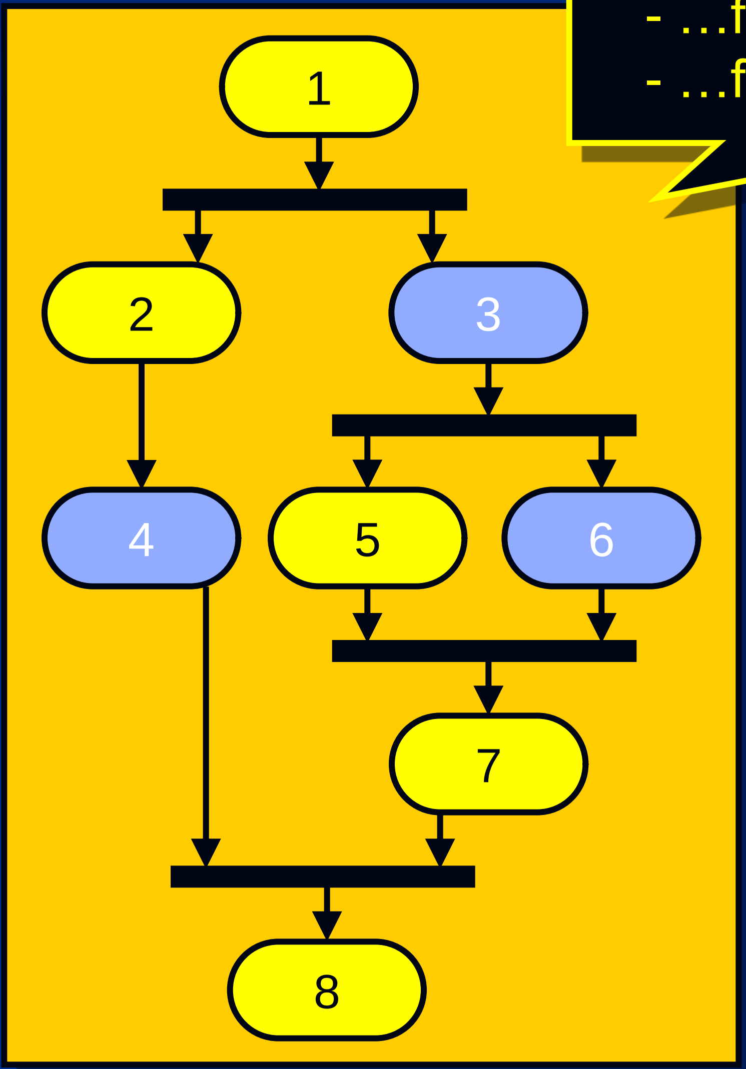
We will use this *activity model* as an example (actually a mixed form of the state and activity models for simplicity reasons). Vertical bars are *fork/join symbols*, rounded rectangles are *activities*



We will use this image for representing a *multi-core CPU*



Effects … As…

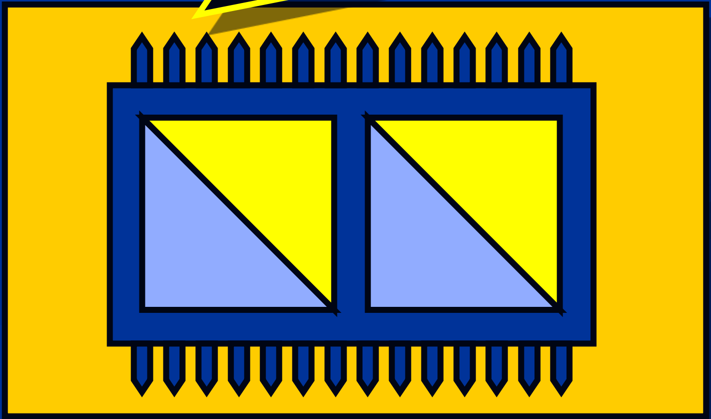


Top-Level Control Structure

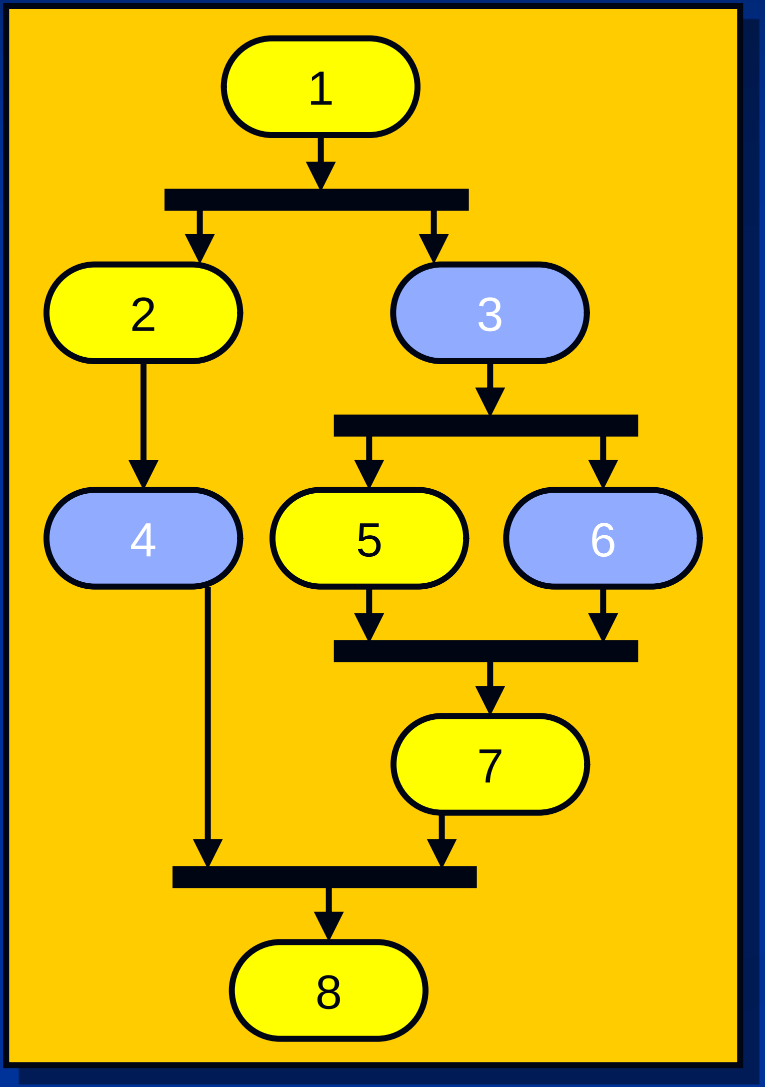
- Some mixed view of the state-transition and the activity models
- Consisting of dominantly
  - …fixed point and 🟨
  - …floating point steps 🟦

Actual CPU

- Two cores, both cores with
  - …fixed point and 🟨
  - …floating point ALUs 🟦

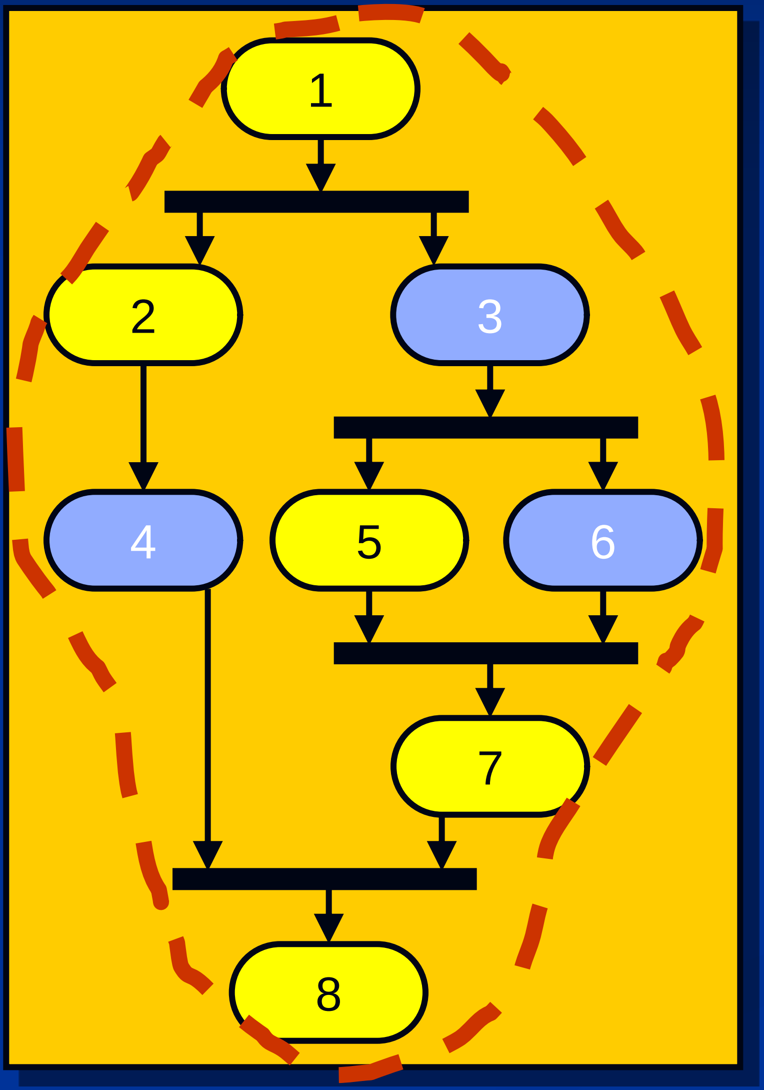


Effects of Task-Core Assignment



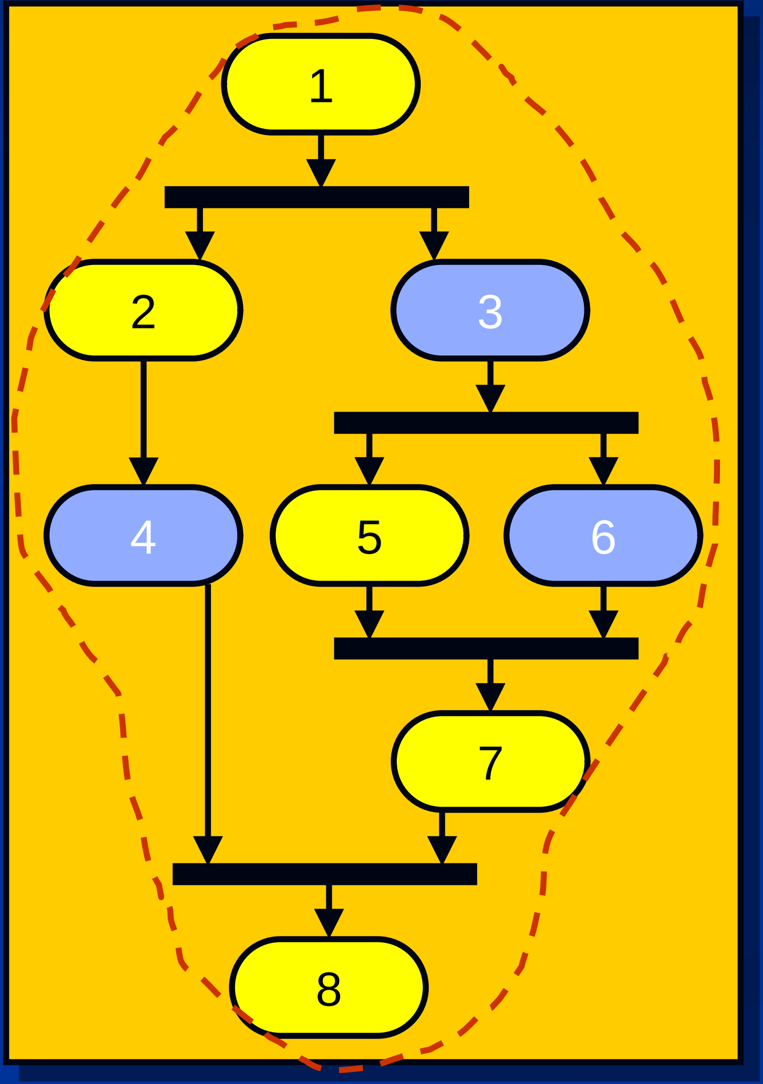
OK, so I have to organize this activity structure into a program…
(Jon, programmer)

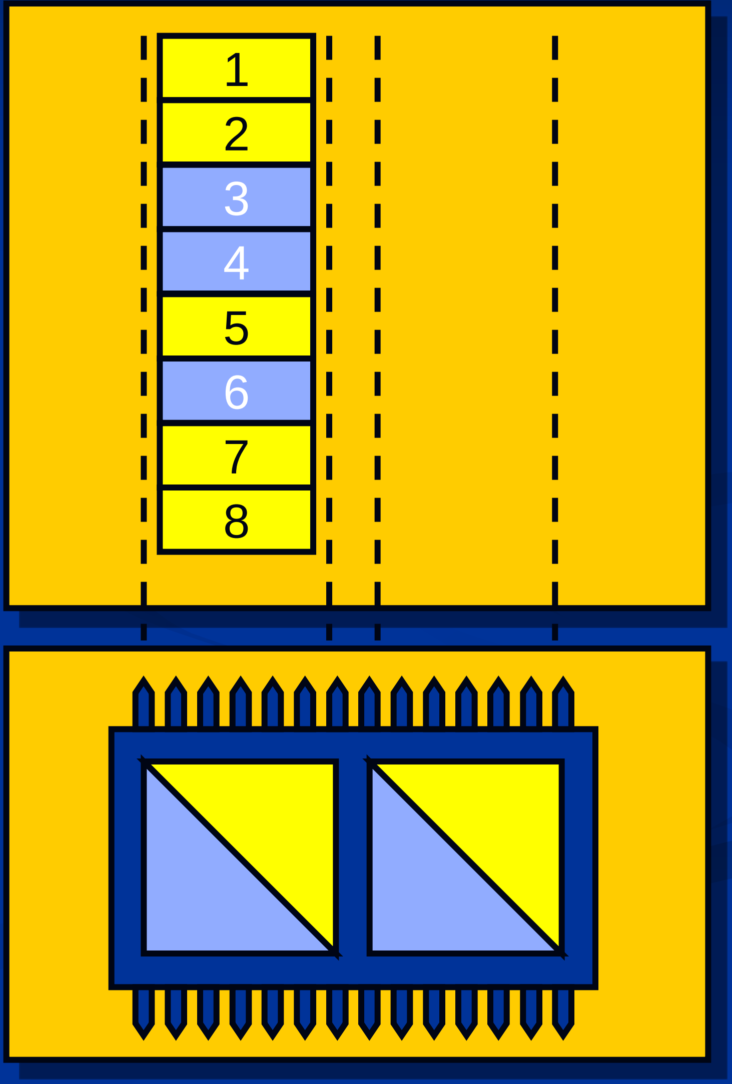
Effects of Task-Core Assignment



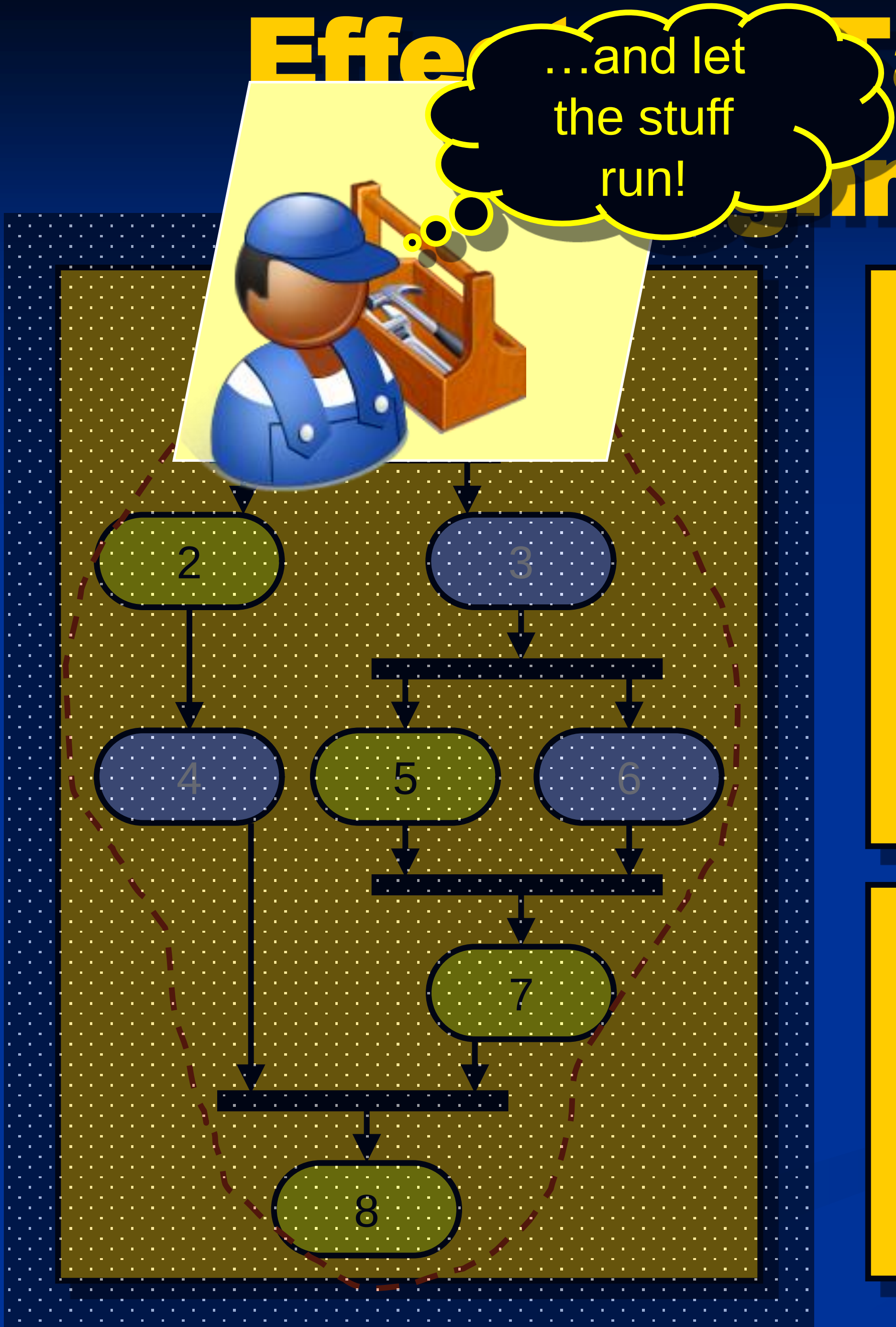
Hmmm, a single threaded executable will do it, let's see some good olde' coding…
(Jon, programmer)

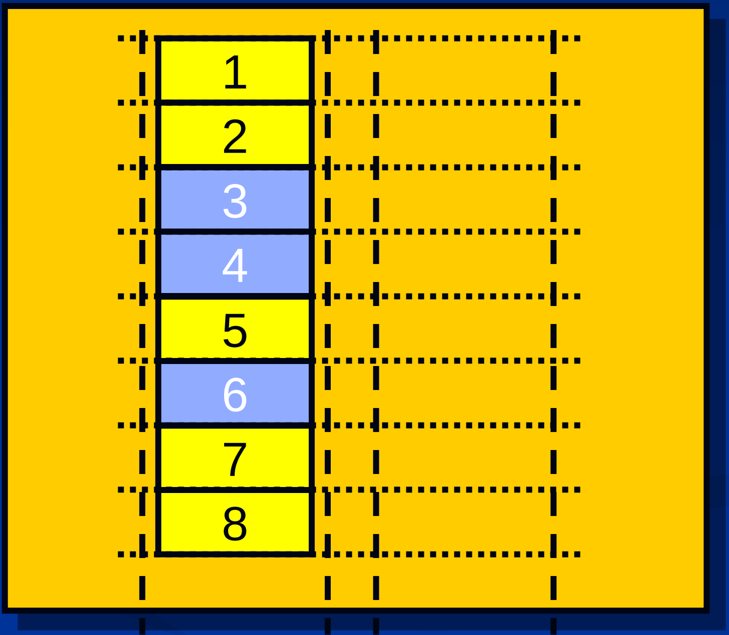
Effects of Task-Core Assignment

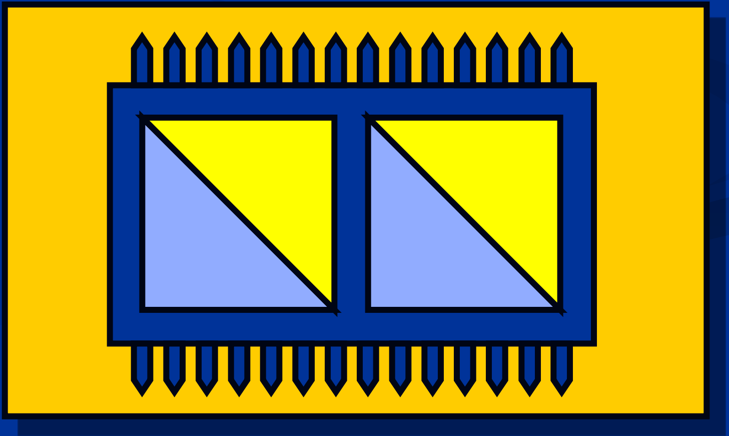




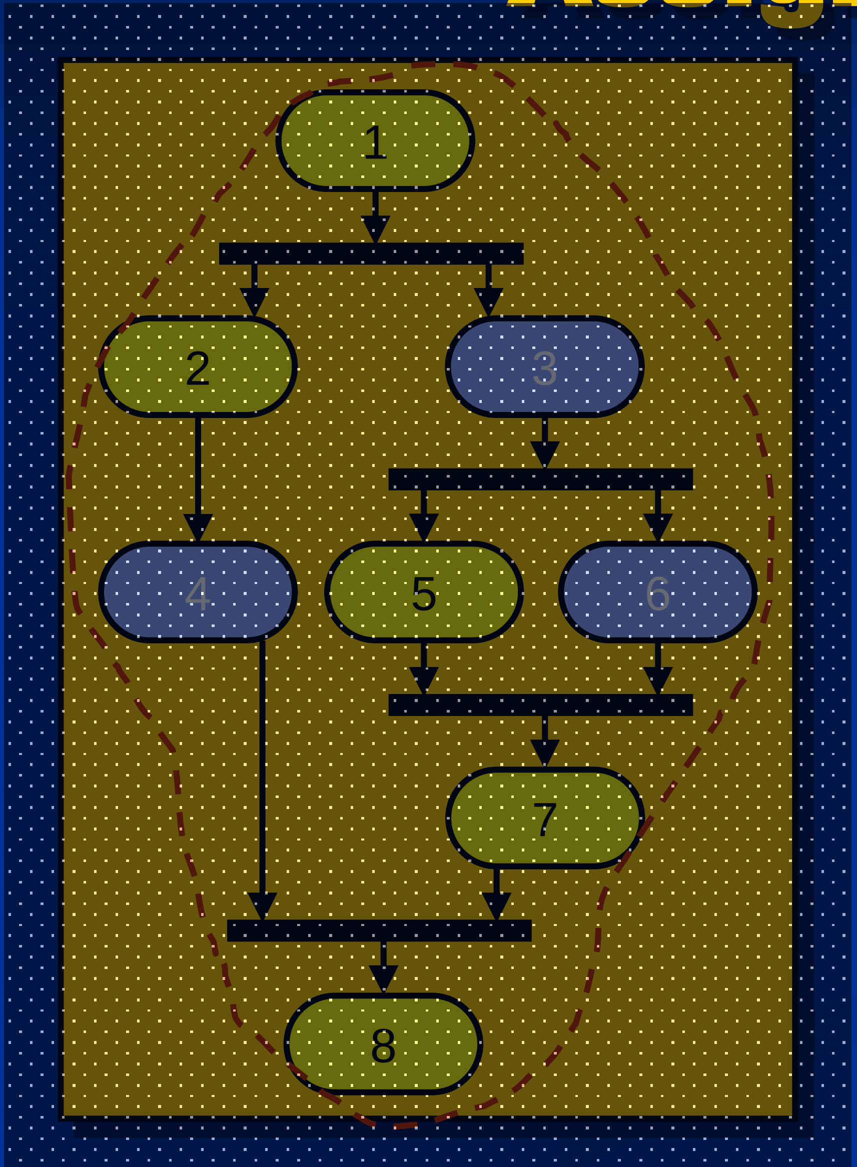
Effe Task-Core
ment

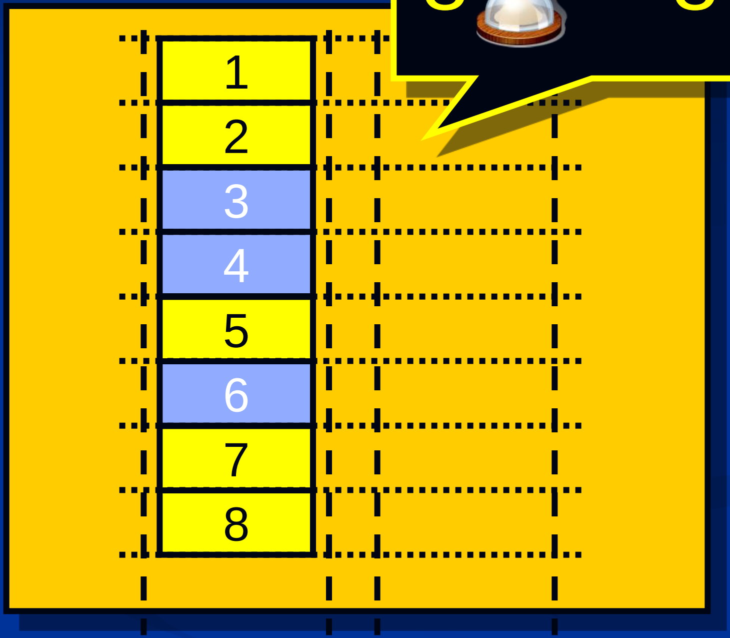






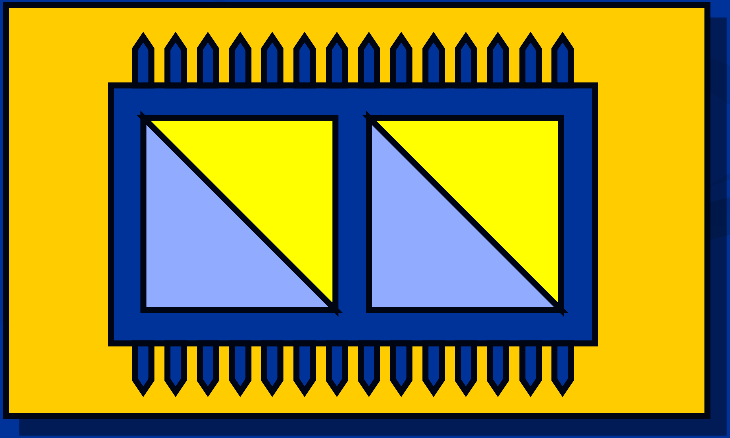
Effects of Task-Co Assignment





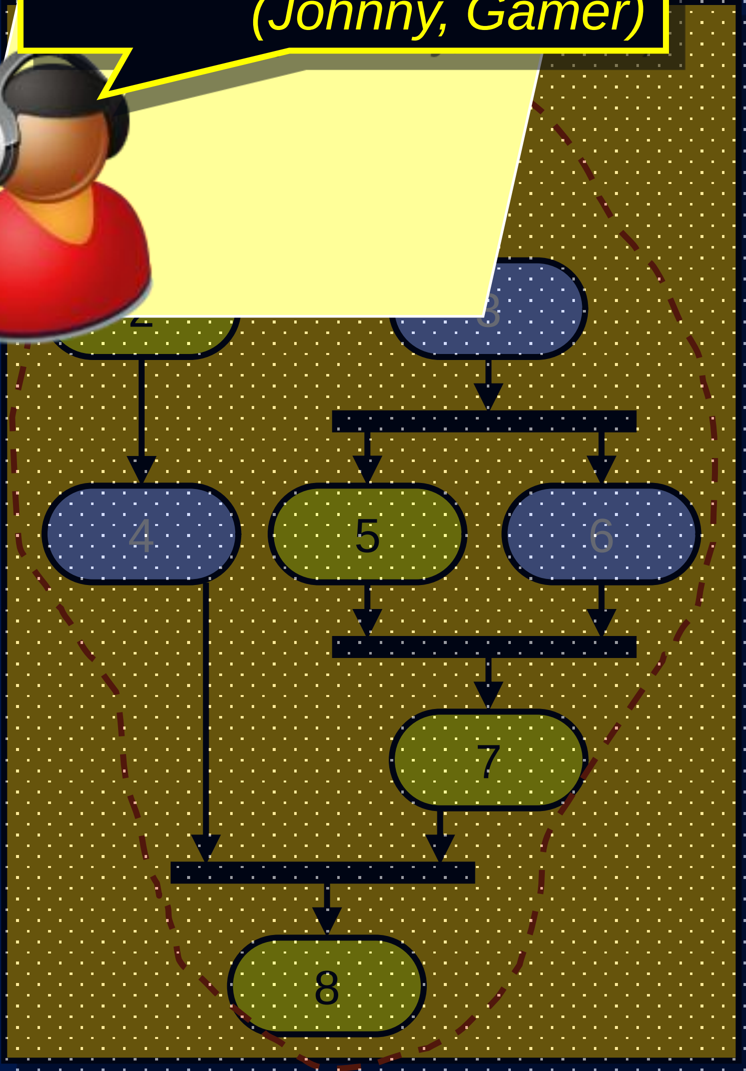
Time/Power Consumption

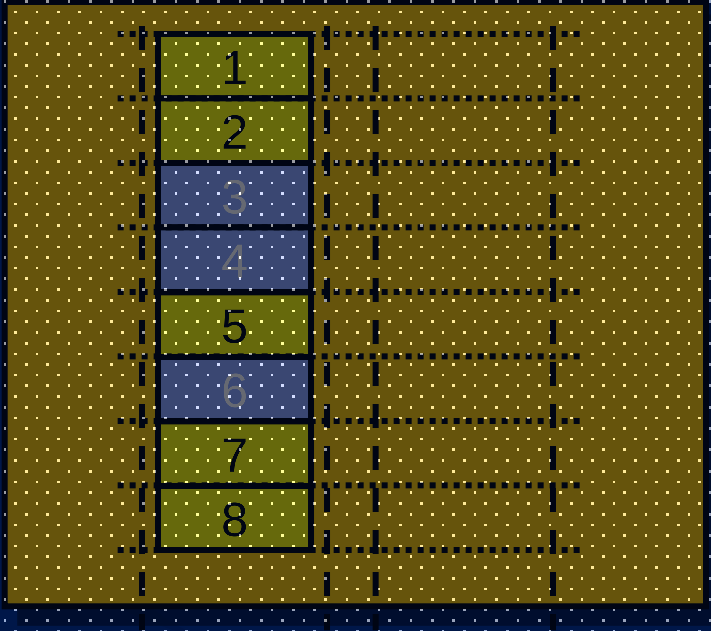
8 8

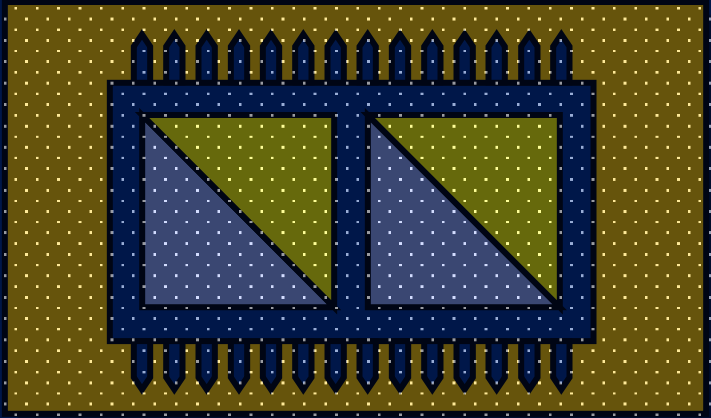


Effects of Task-Core Alignment

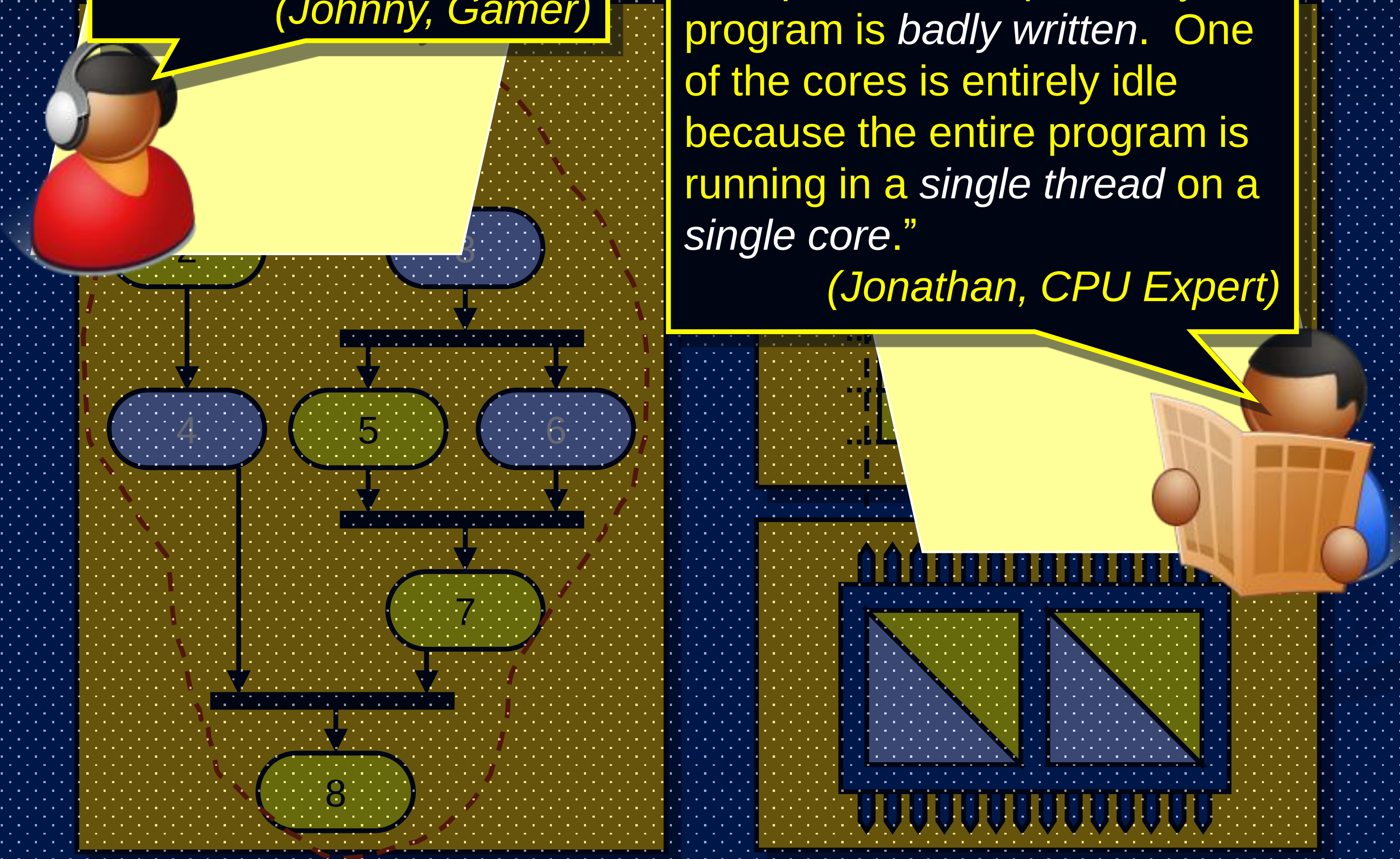
“I paid 2000 bucks for this computer and my game is still *crawling*.”
(Johnny, Gamer)







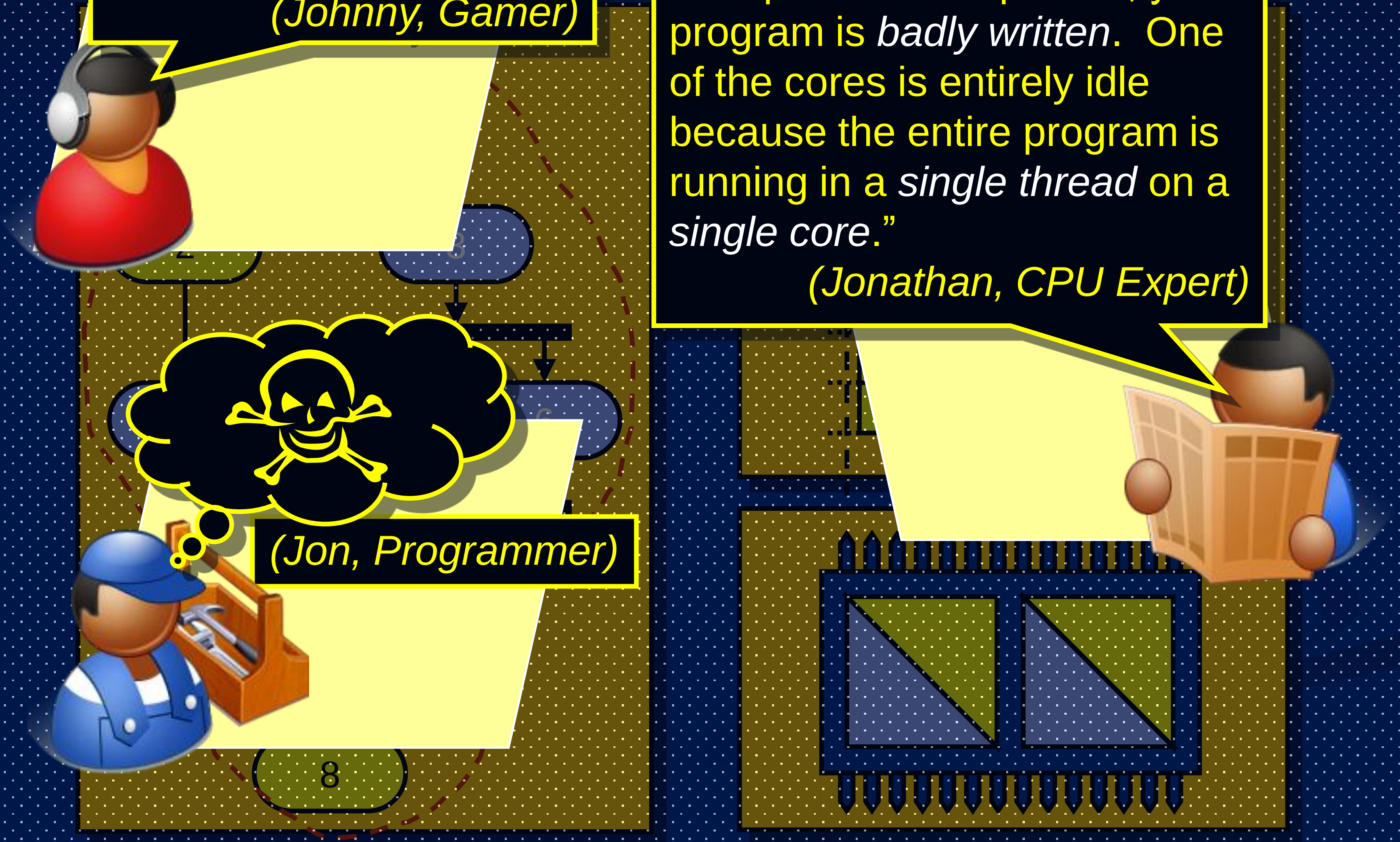
Effects of Task-Core Assignment



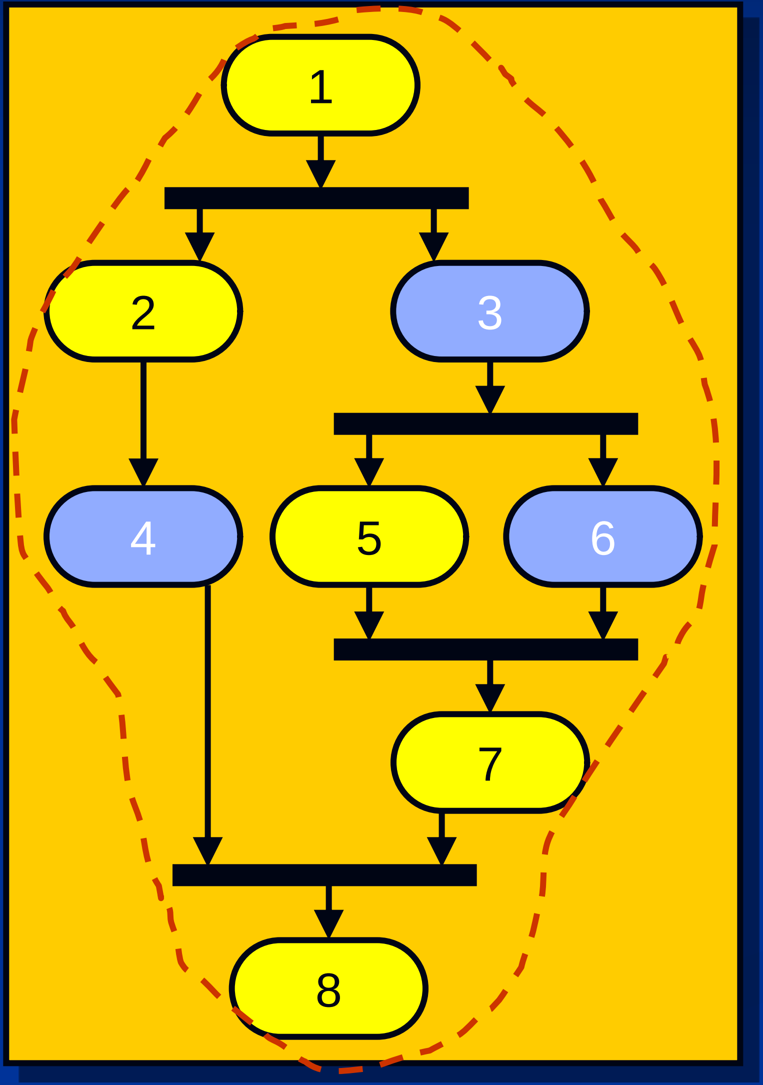
"I paid 2000 bucks for this computer and my game is still *crawling*."
(Johnny, Gamer)

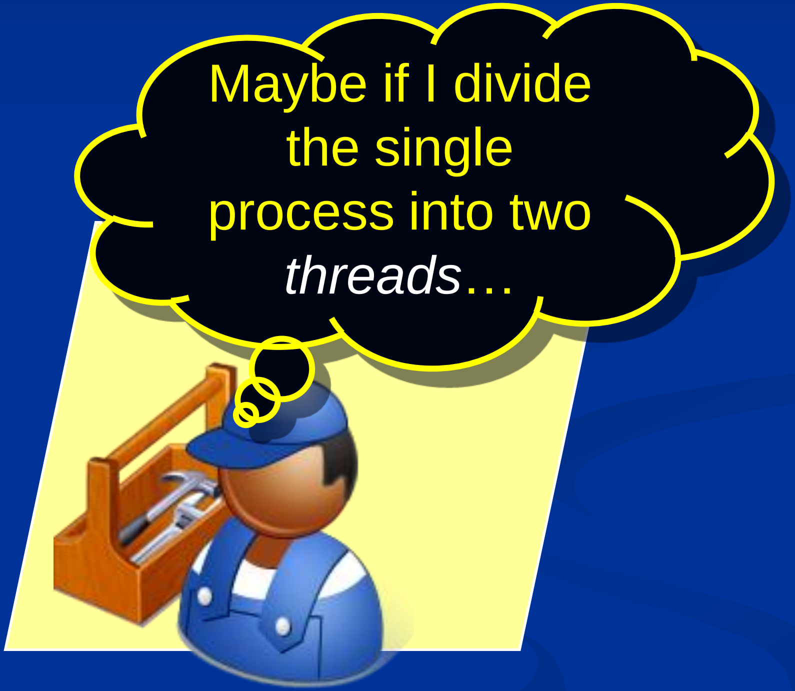
"Our processor is perfect, your program is *badly written*. One of the cores is entirely idle because the entire program is running in a *single thread* on a *single core*."
(Jonathan, CPU Expert)

Effects of Task-Core Assignment


"I paid 2000 bucks for this computer and my game is still *crawling*."
(Johnny, Gamer)
"Our processor is perfect, your program is *badly written*. One of the cores is entirely idle because the entire program is running in a *single thread* on a *single core*."
(Jonathan, CPU Expert)
(Jon, Programmer)
2
3
6
8

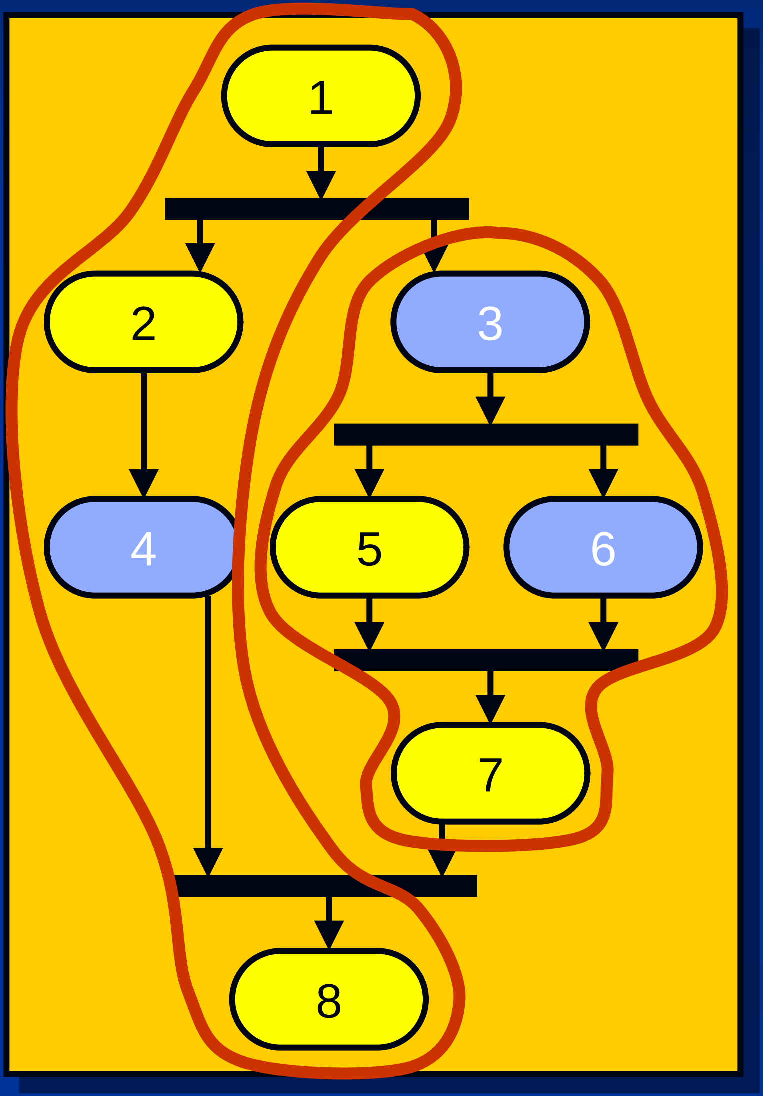
Effects of Task-Core Assignment

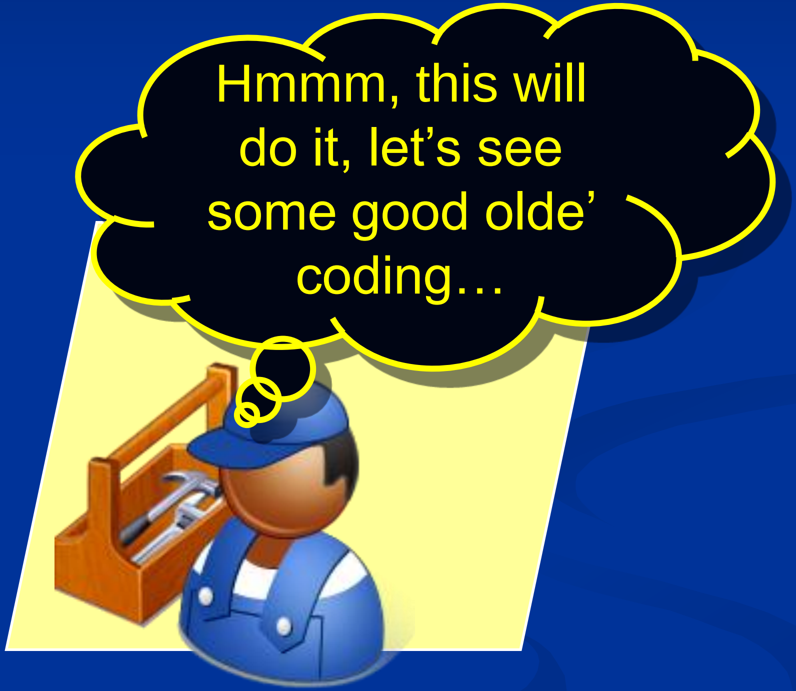




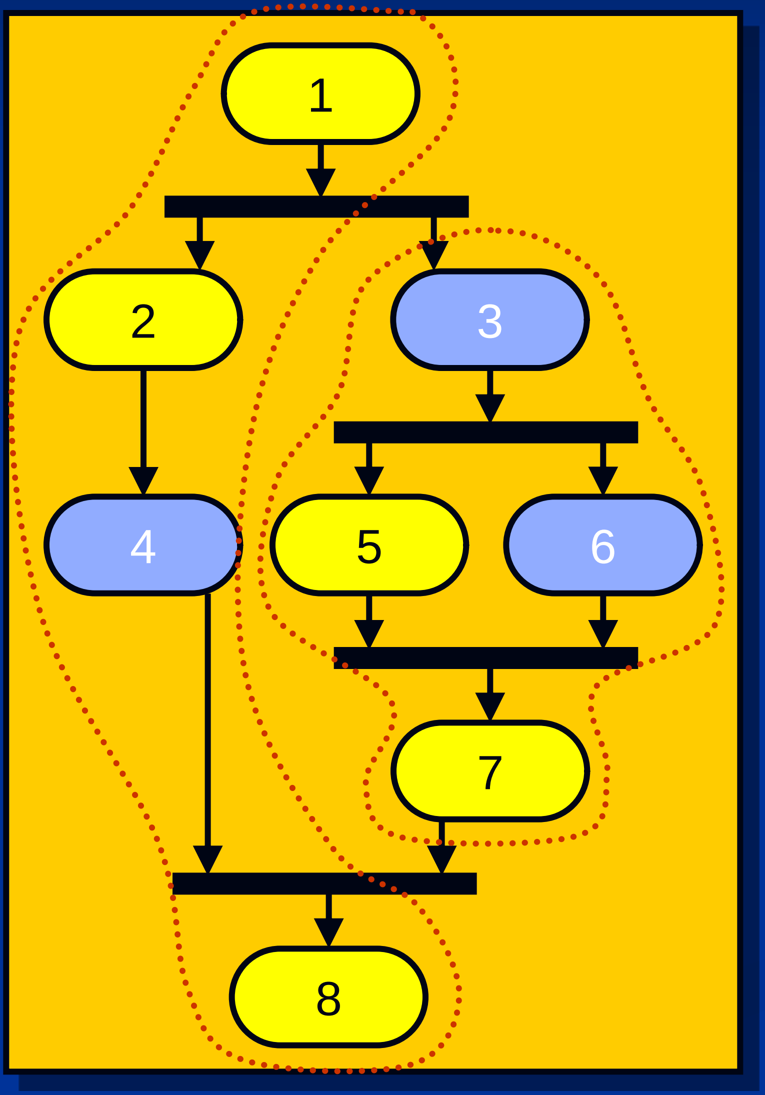
Maybe if I divide the single process into two *threads*…

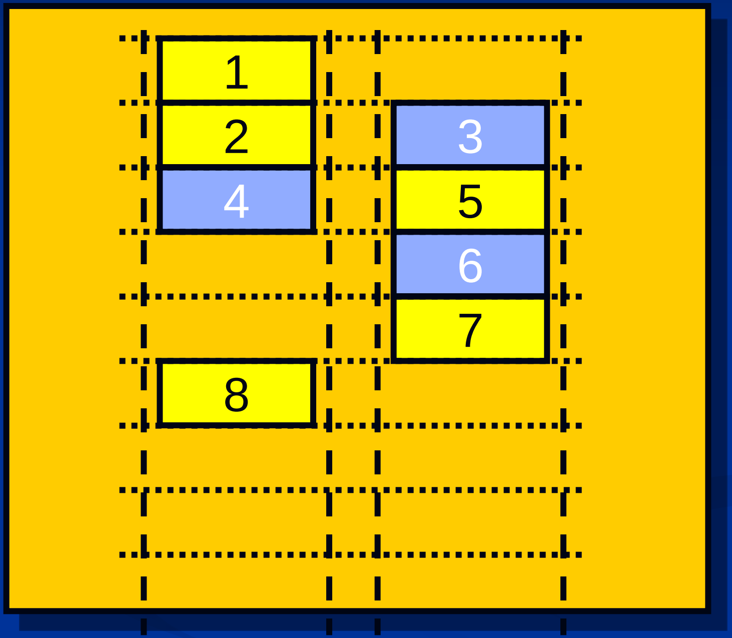
Effects of Task-Core Assignment

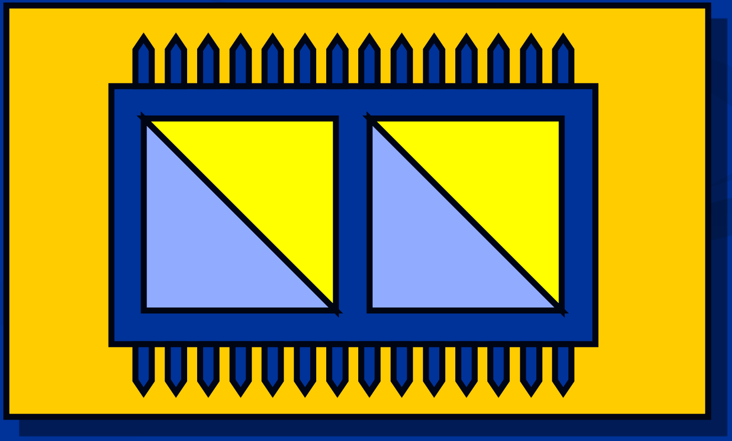




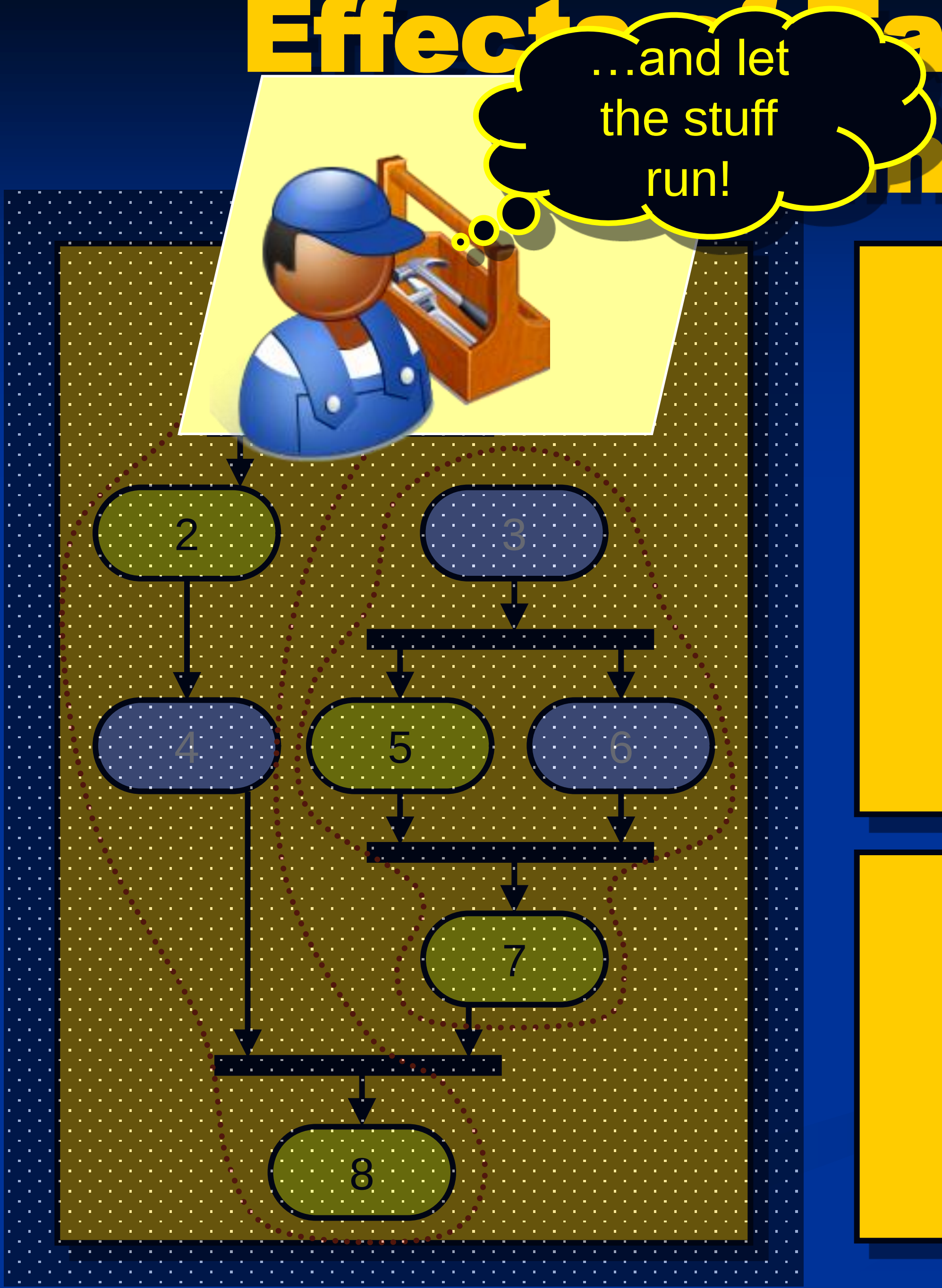
Effects of Task-Core Assignment

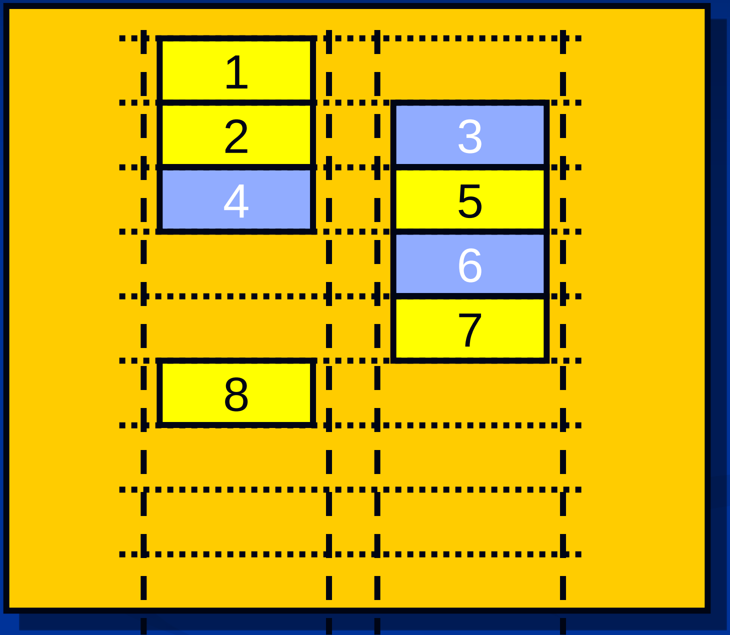


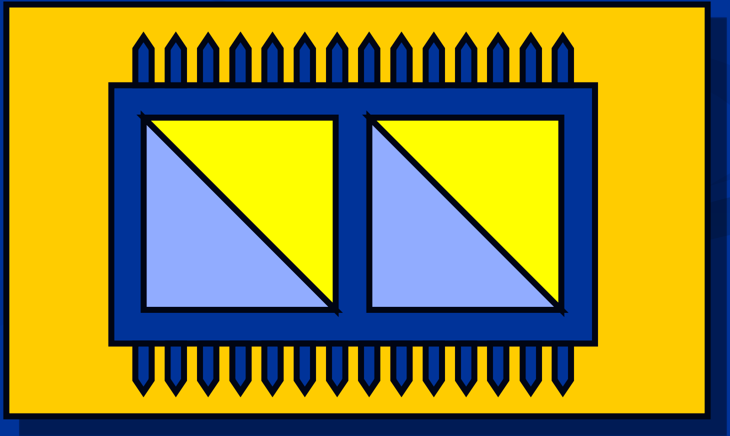




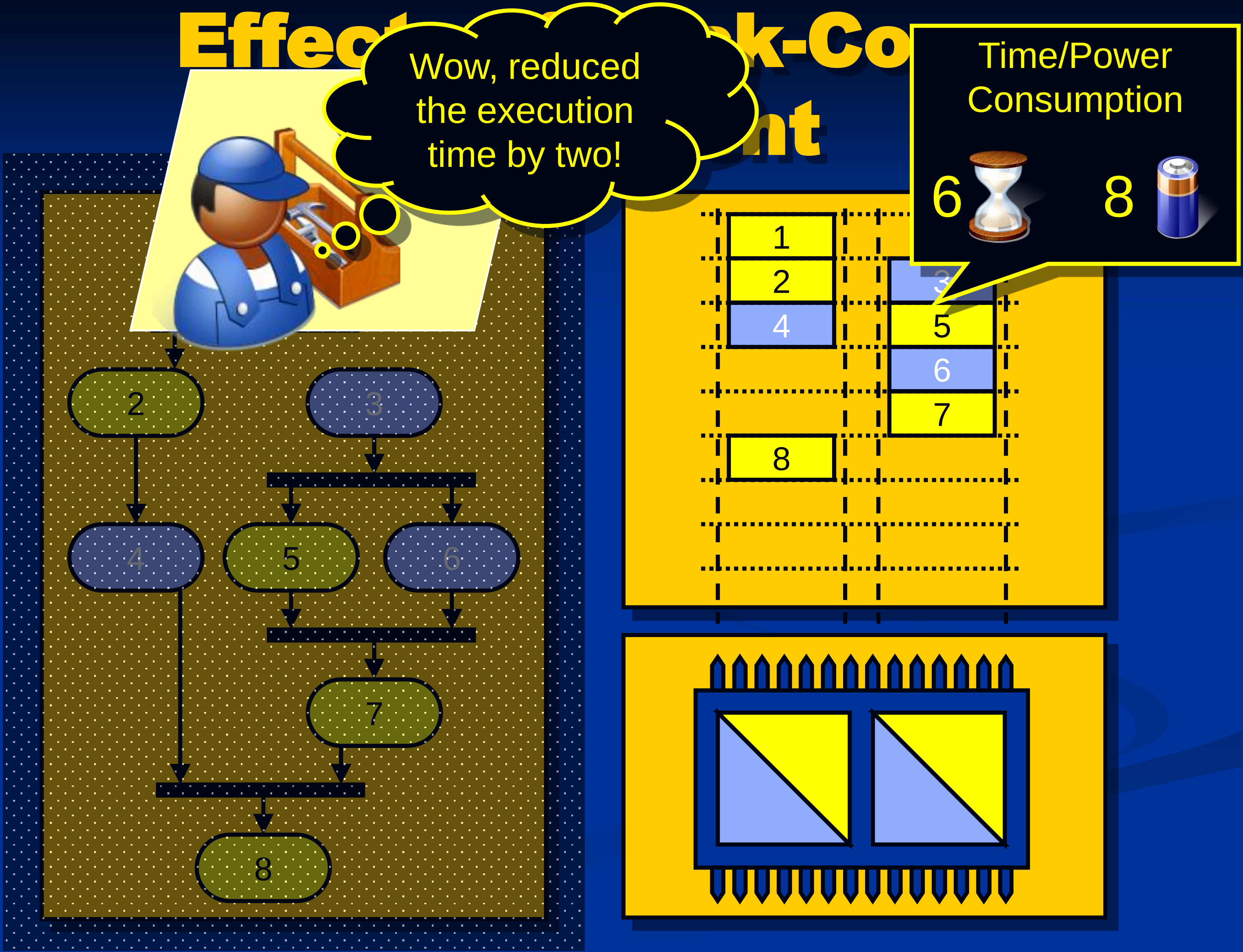
Effect ask-Core ment







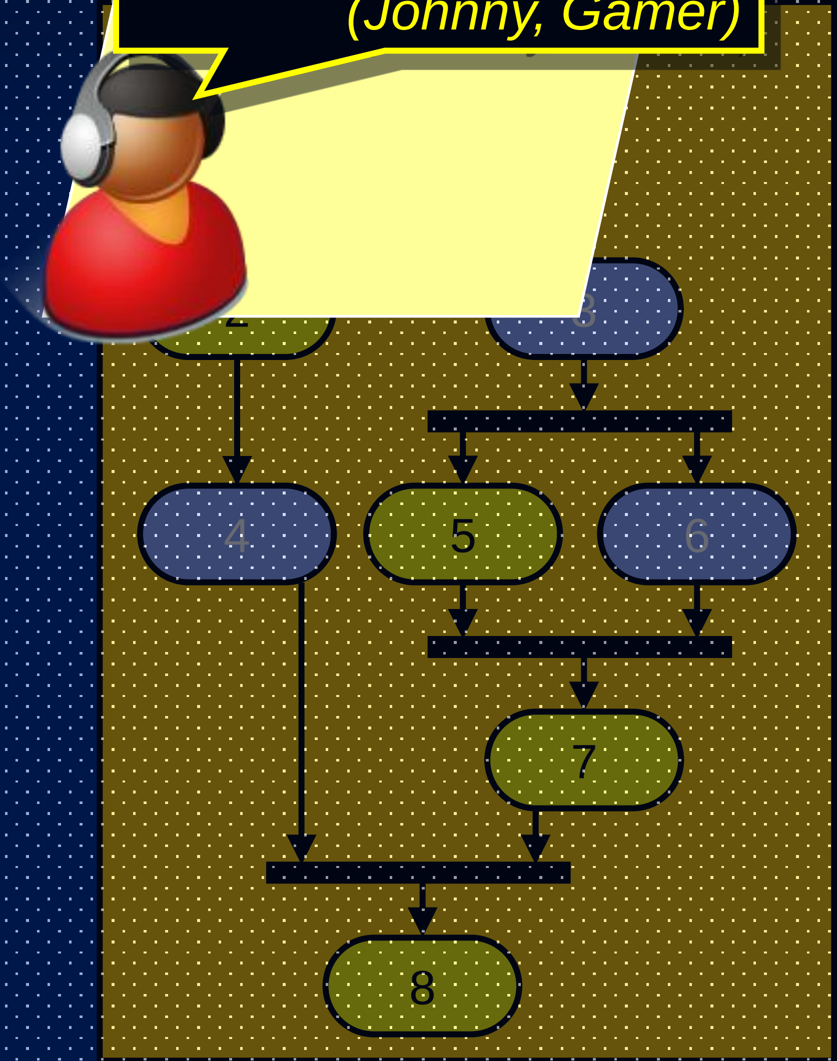
Effec k-Co nt

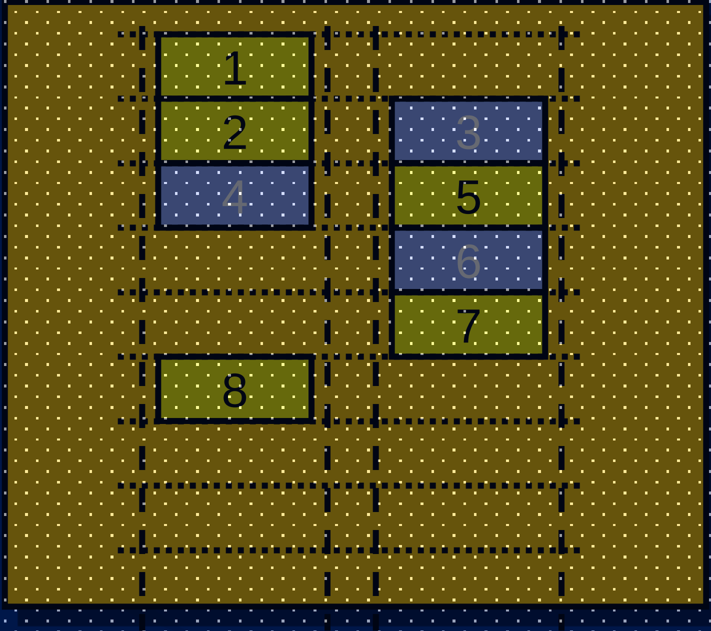


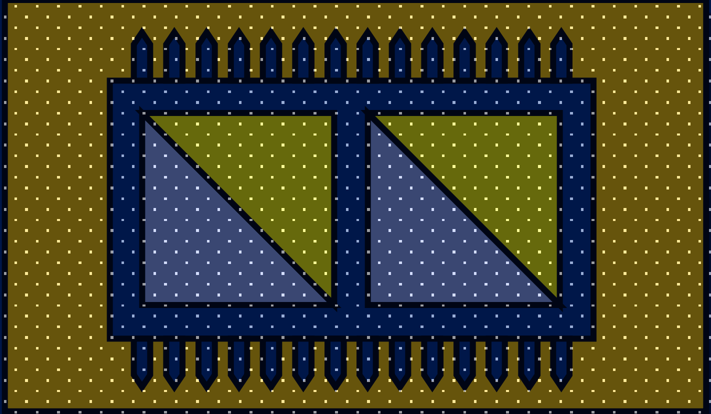
Wow, reduced the execution time by two!
Time/Power Consumption
6
8
1
2
3
4
5
6
7
8
2
3
4
5
6
7
8

Effects of Task-Core ignment

“I paid 2000$ for this laptop and it is *burning* a hole in my pants.”
(Johnny, Gamer)







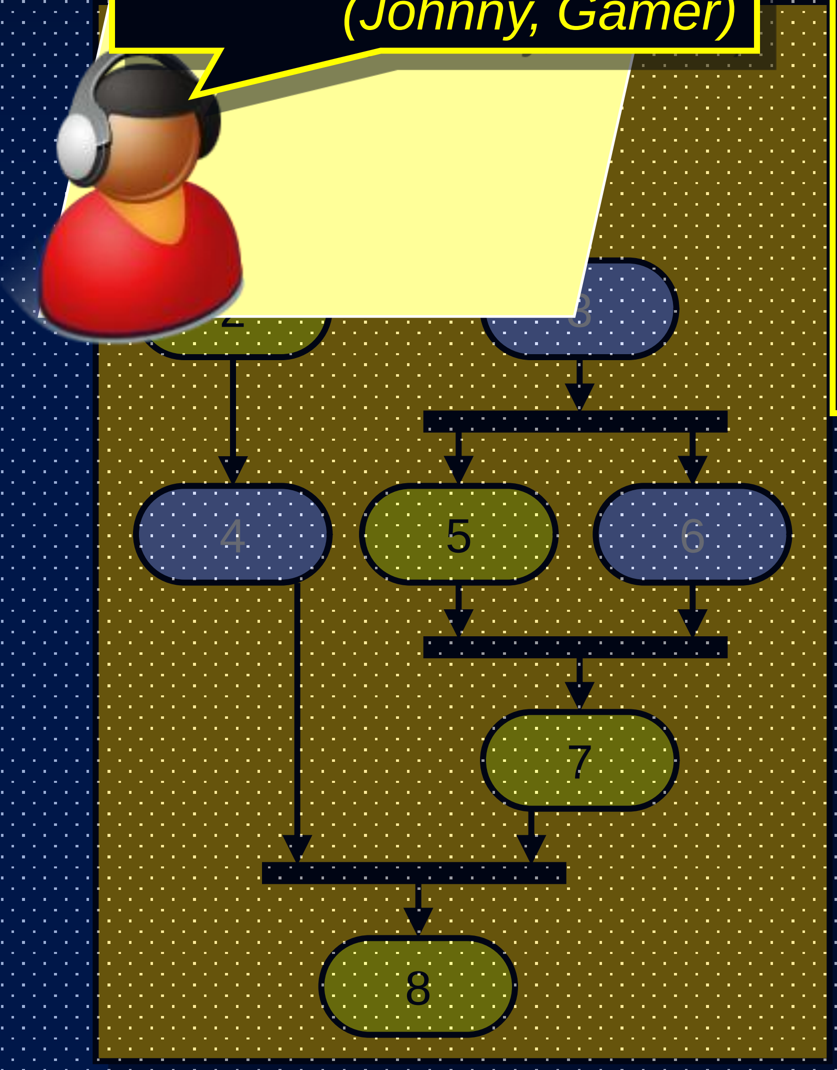
Effects of Task-Core

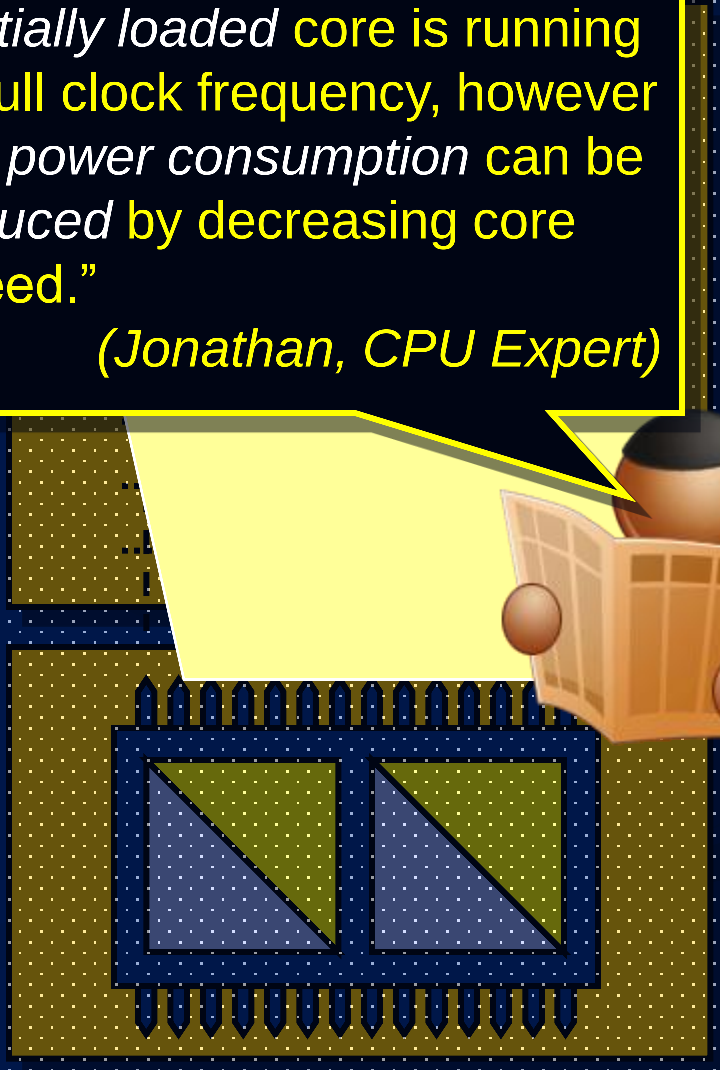
“I paid 2000$ for this laptop and it is *burning* a hole in my pants.”

(Johnny, Gamer)

“Our processor is perfect, your program is *badly written*. A *partially loaded* core is running at full clock frequency, however the *power consumption* can be *reduced* by decreasing core speed.”

(Jonathan, CPU Expert)





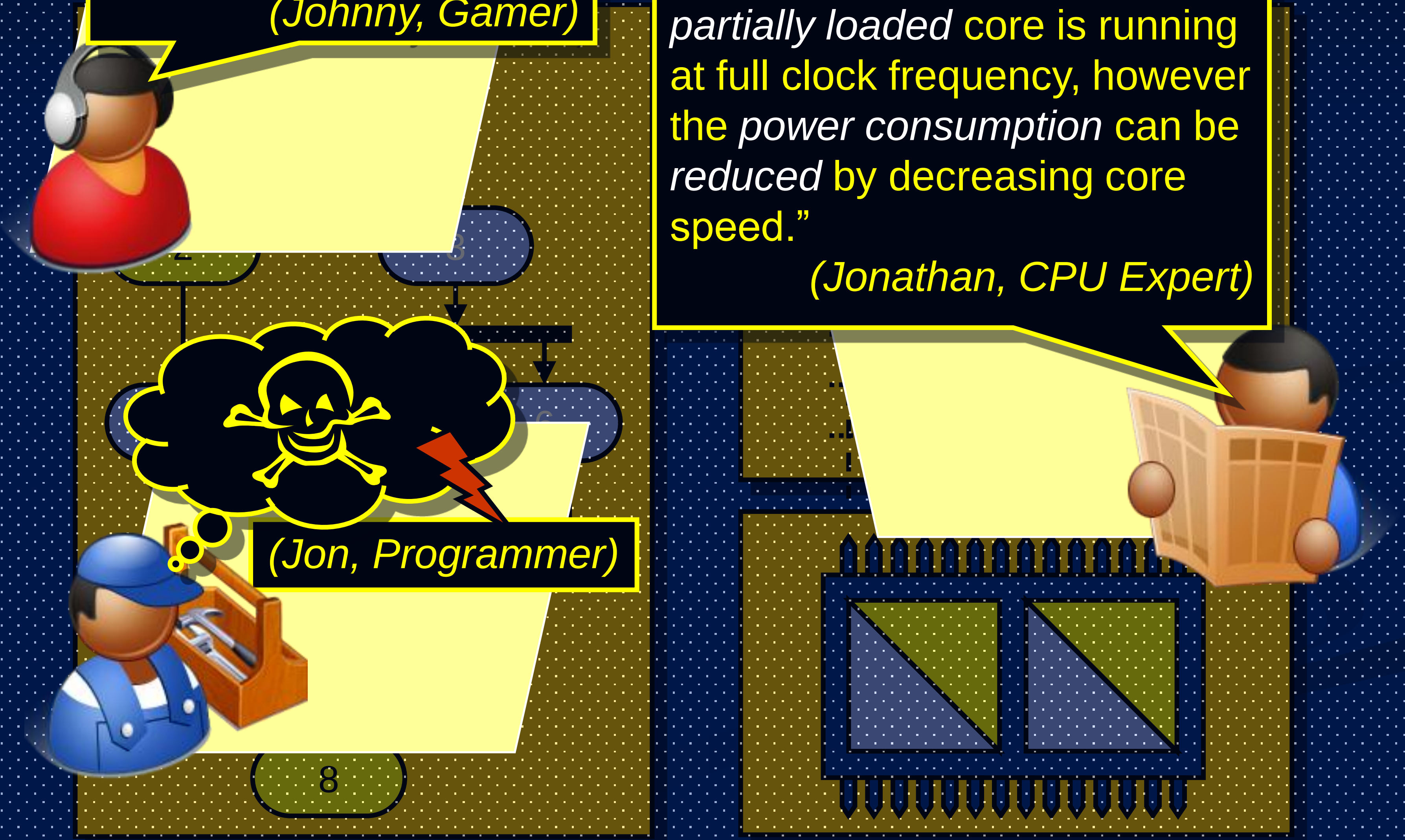
Effects of Task-Core

“I paid 2000$ for this laptop and it is *burning* a hole in my pants.”

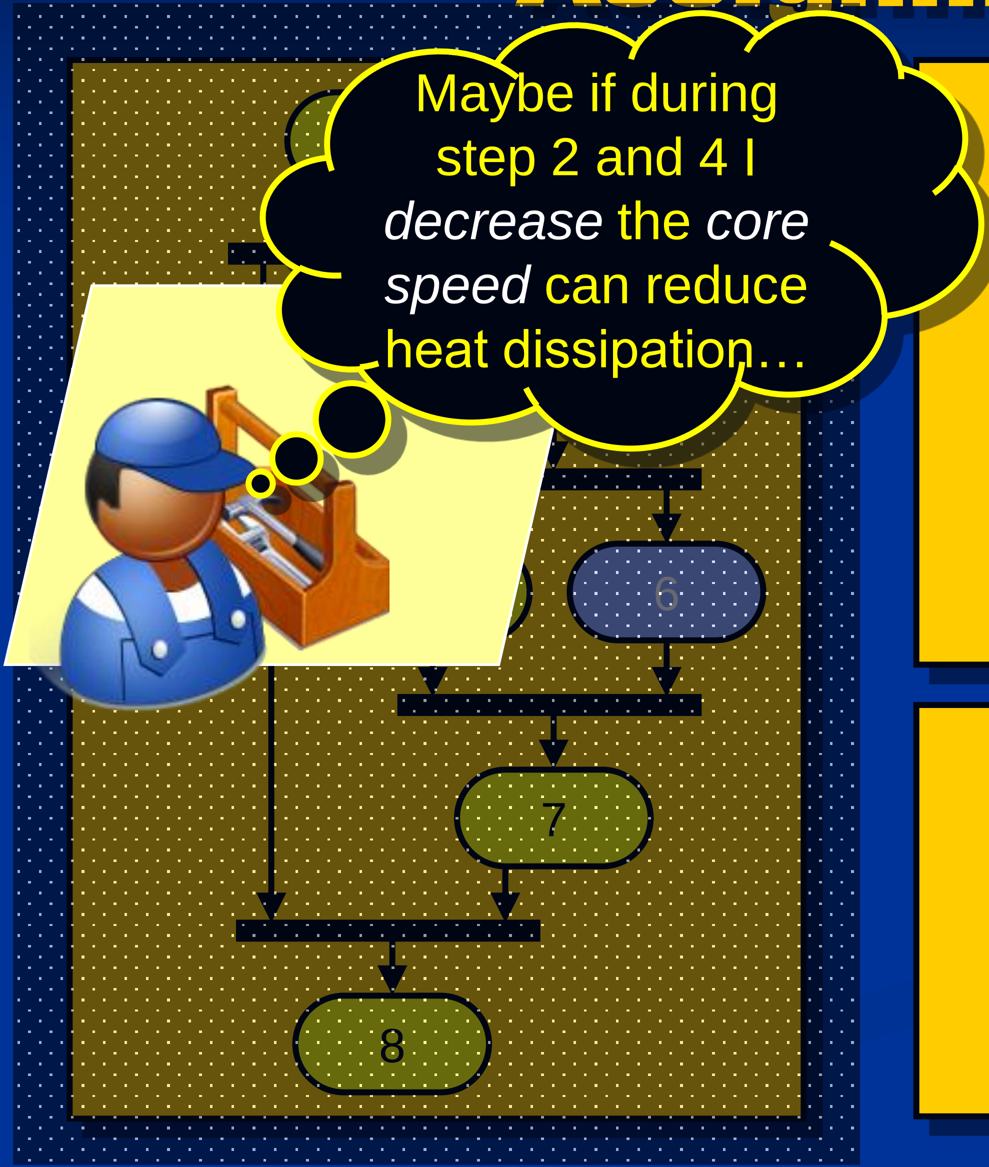
(Johnny, Gamer)

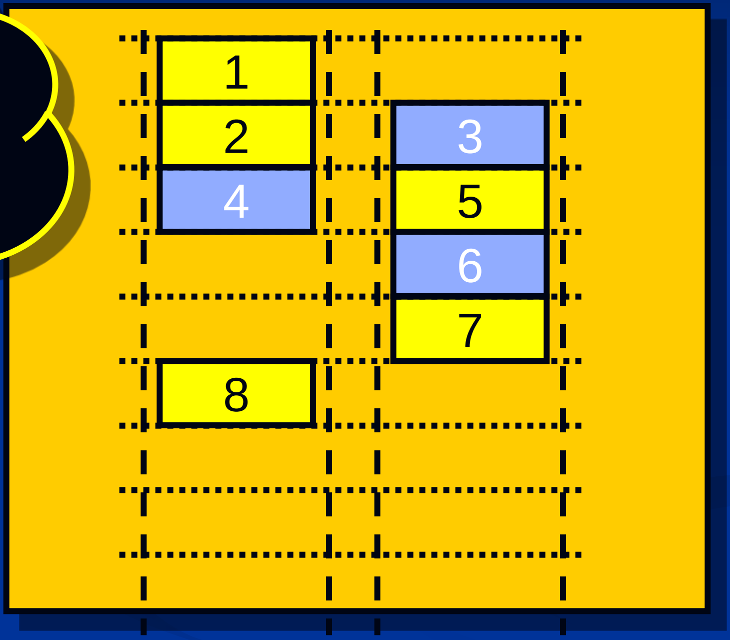
“Our processor is perfect, your program is *badly written*. A *partially loaded* core is running at full clock frequency, however the *power consumption* can be *reduced* by decreasing core speed.”

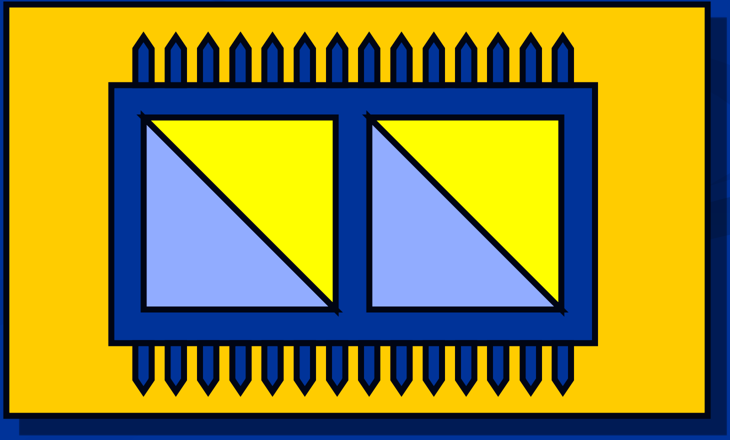
(Jonathan, CPU Expert)



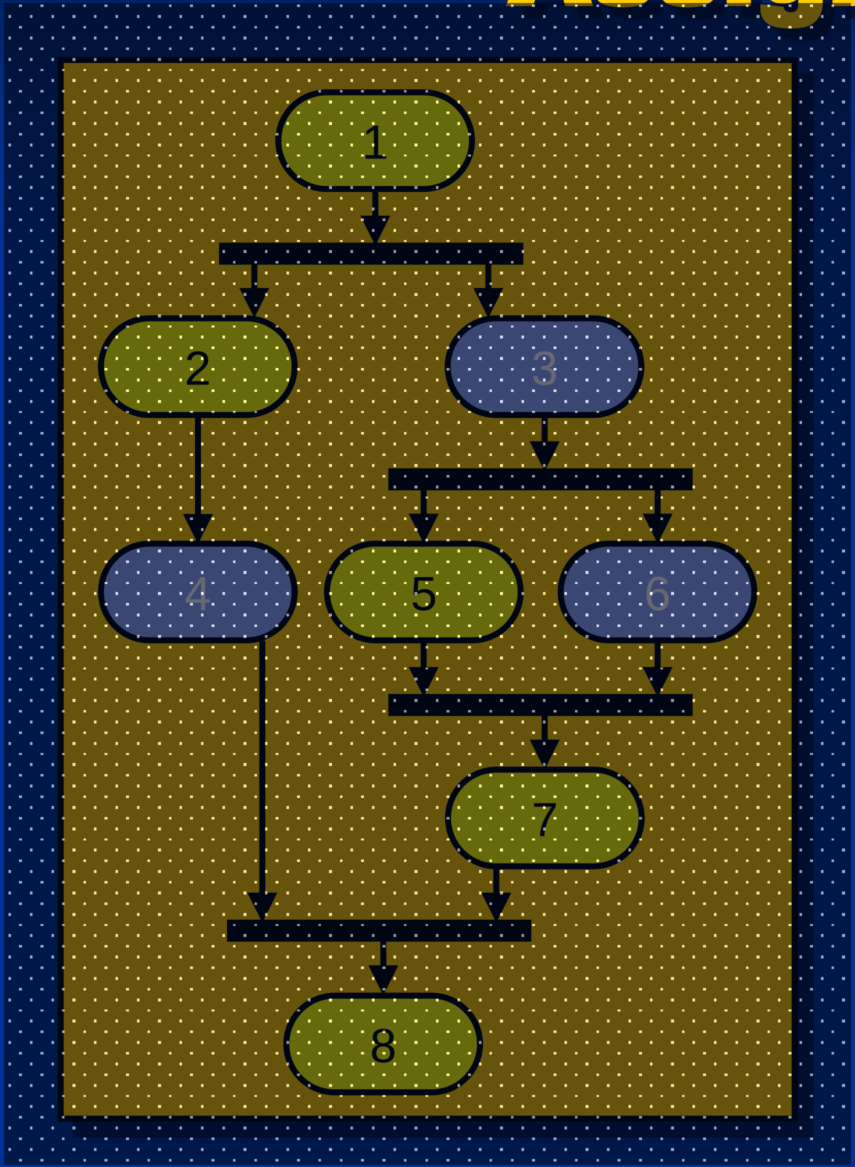
Effects of Task-Core Assignment

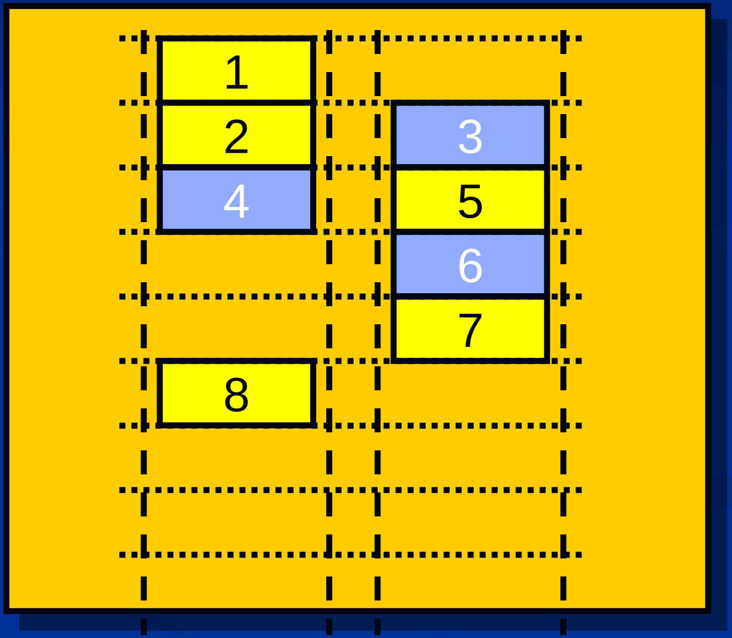

Maybe if during step 2 and 4 I *decrease the core speed* can reduce heat dissipation…
6
7
8

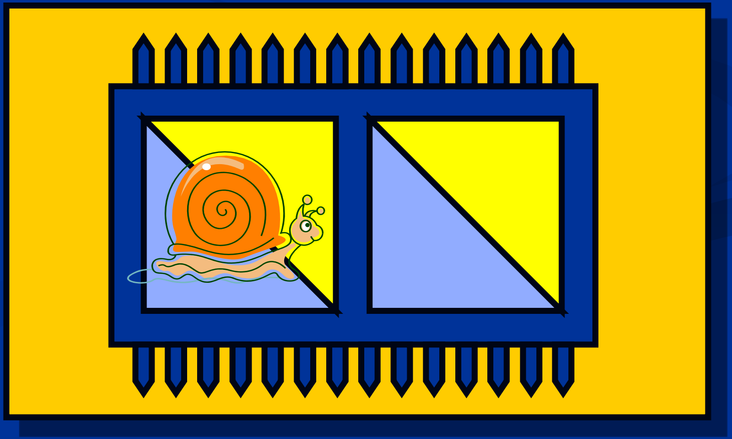




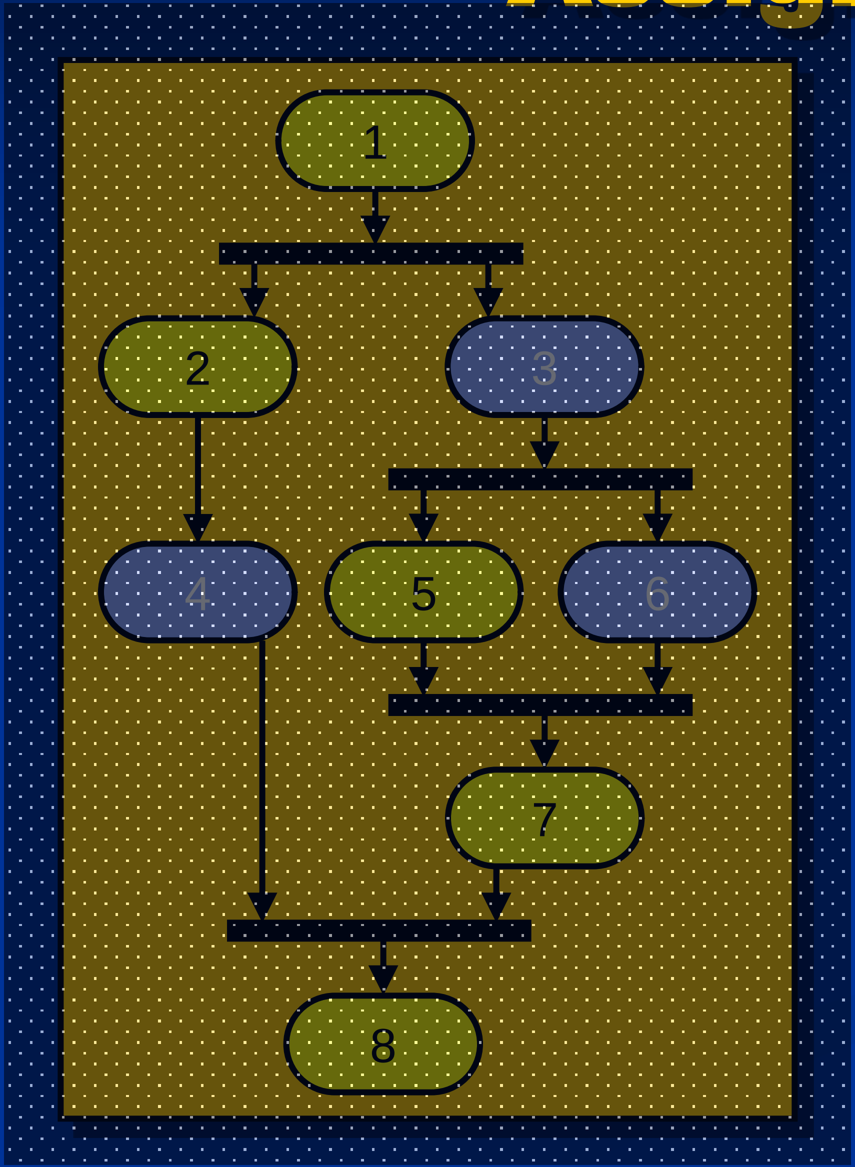
Effects of Task-Core Assignment

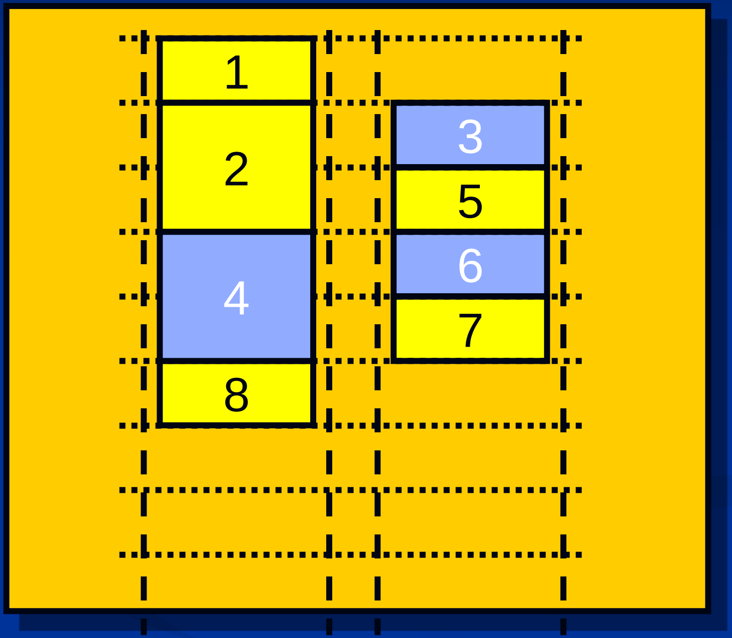


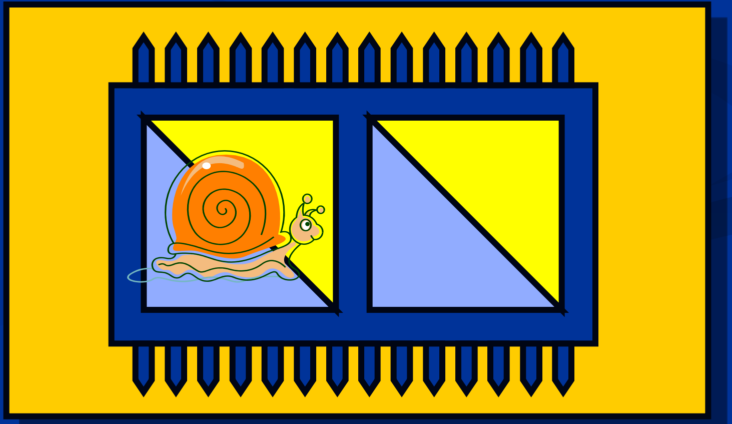




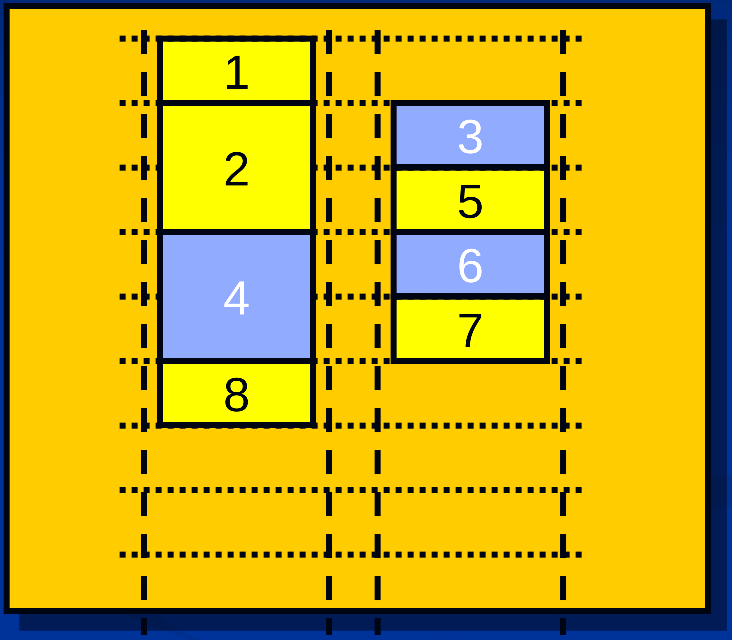
Effects of Task-Core Assignment

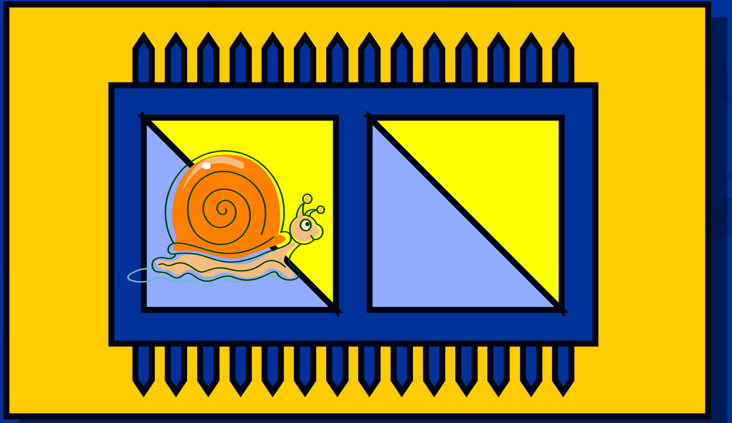




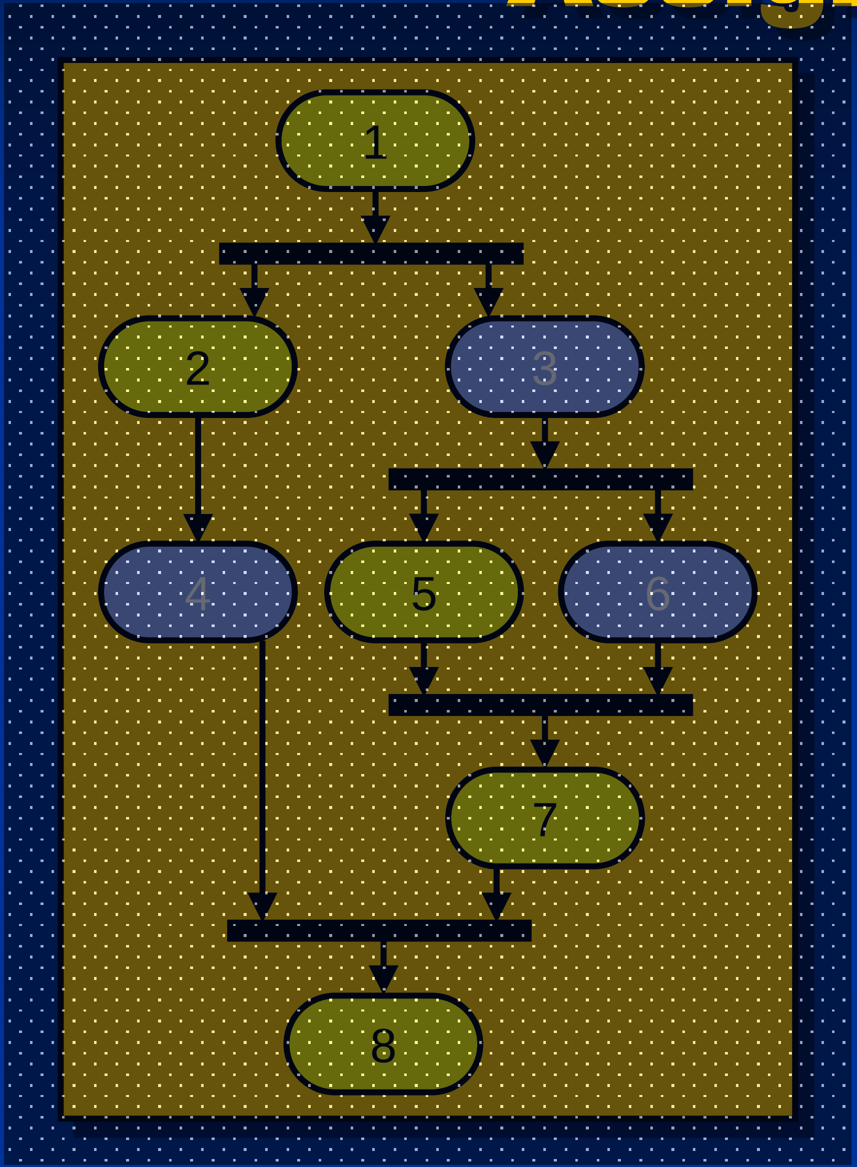


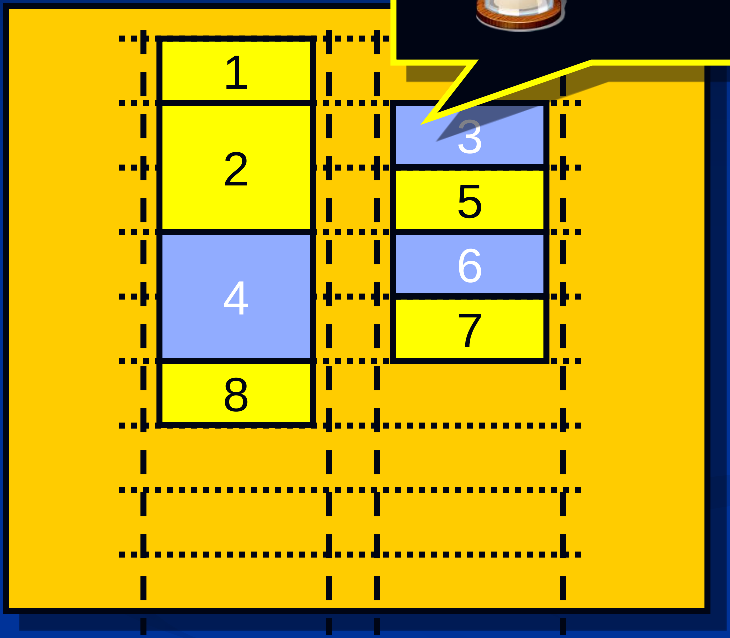
Effects of Task-Core Assignment





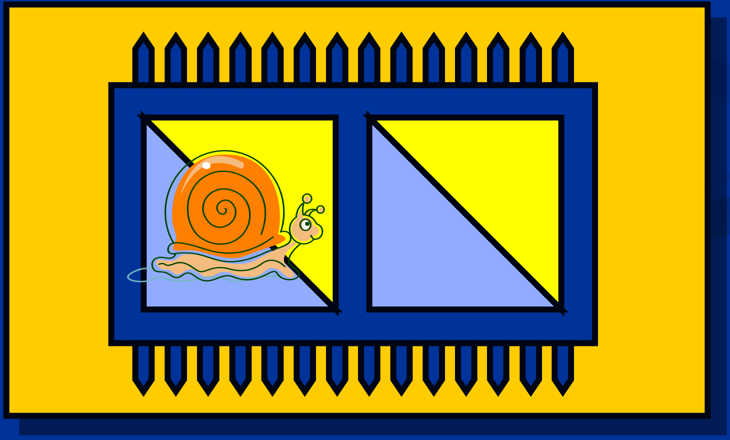
Effects of Task-Co
Assignment



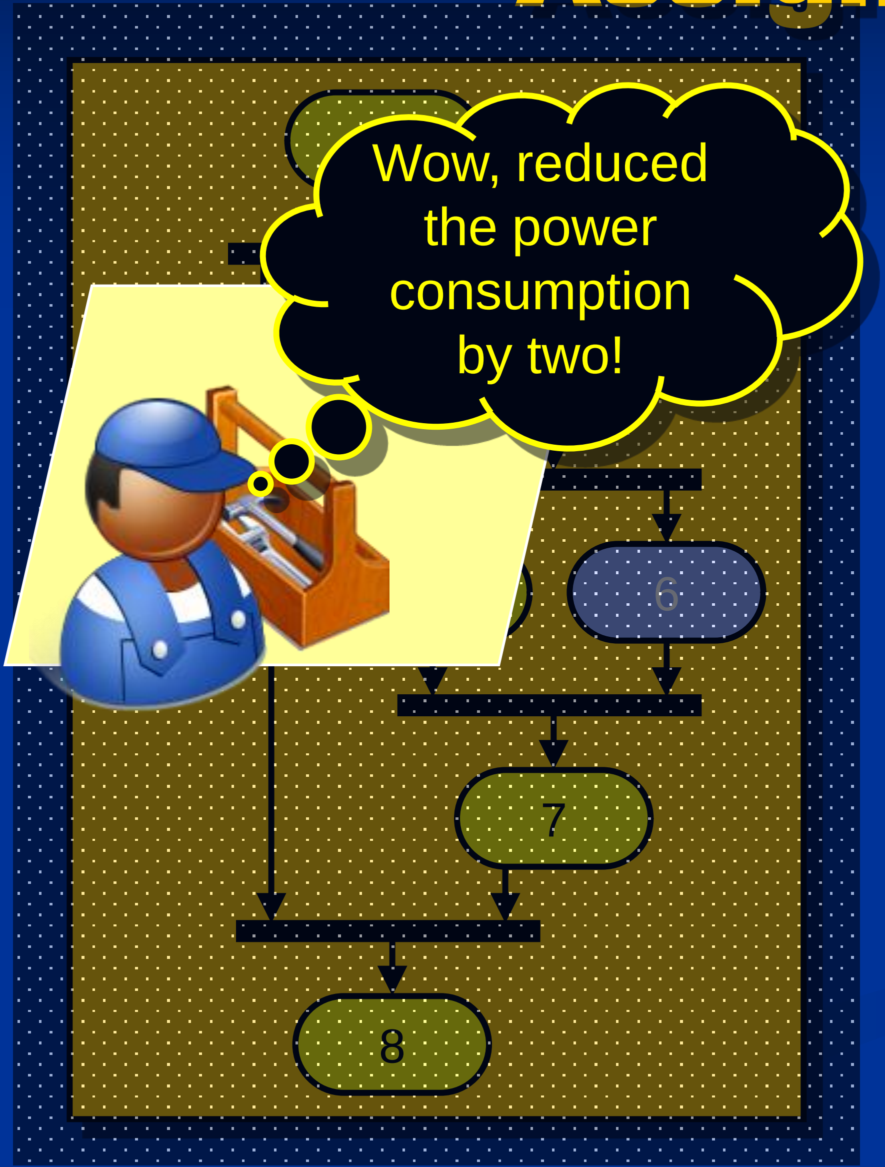


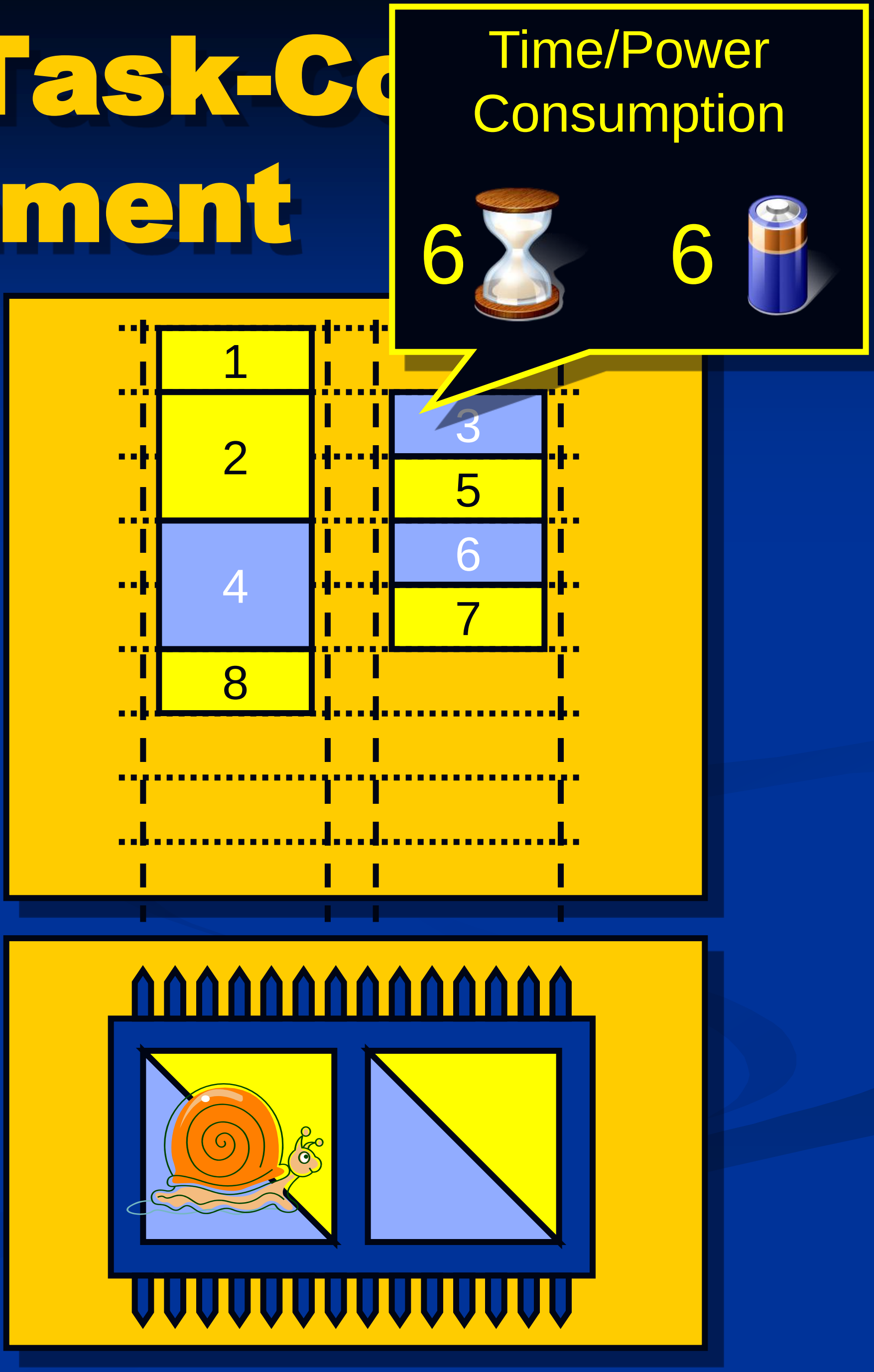
Time/Power Consumption

6 6



Effects of Task-Co Assignment





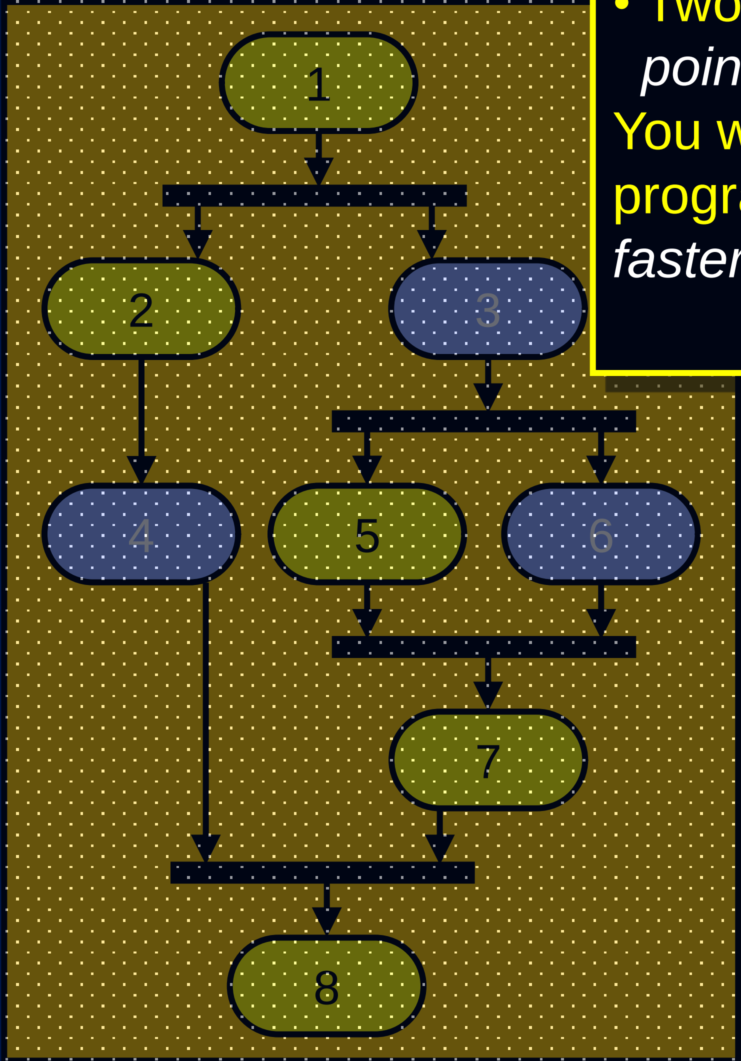
Effects
As

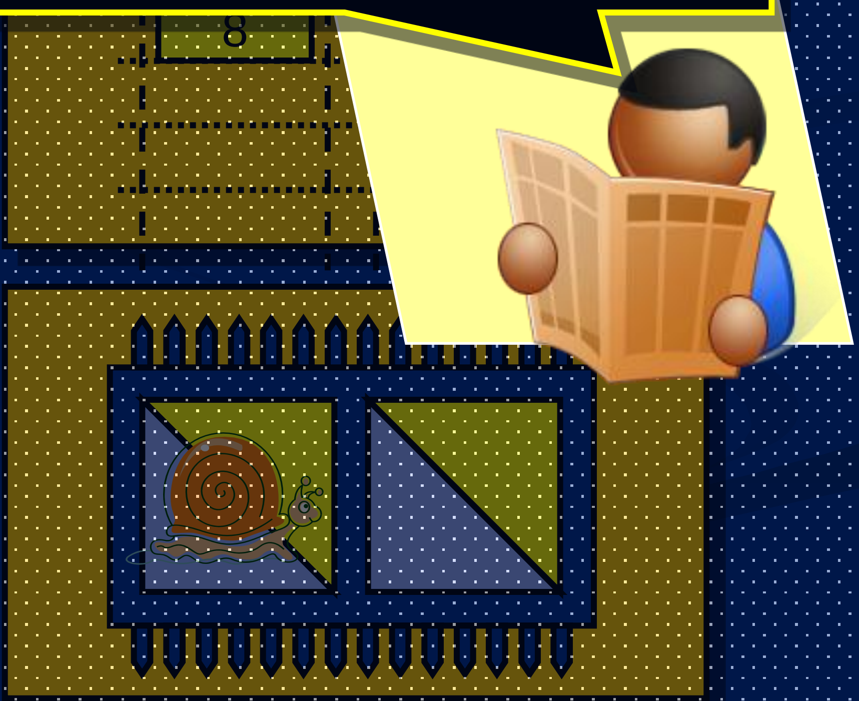
“Let me draw your attention to our *brand new embedded CPU* with…

- One *general-purpose fixed-point core* and…
- Two *high speed* RISC *floating point cores*.

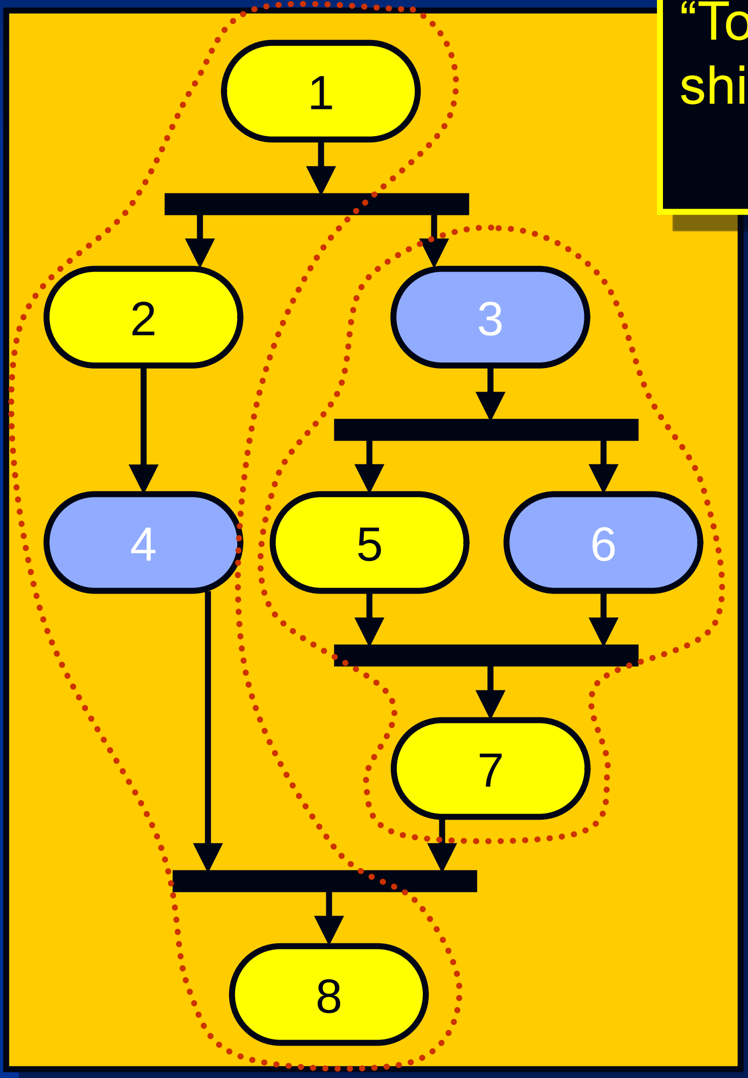
You will be able to re-organize your program such way that it will run even *faster* with *lower power consumption*…”

(Jonathan, CPU Expert)

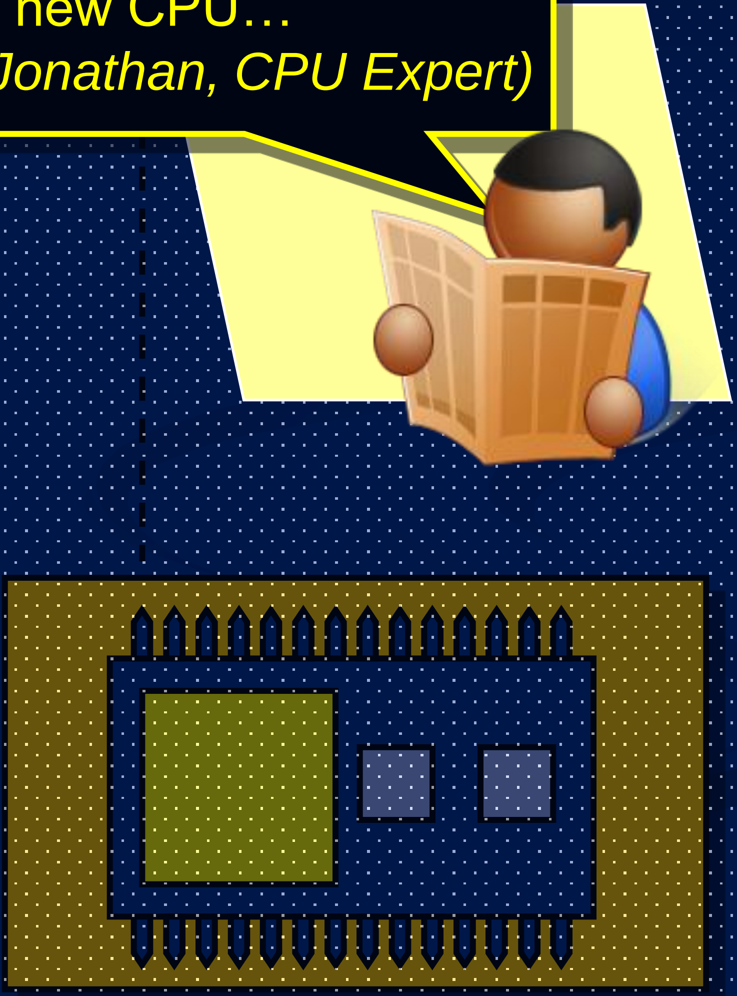




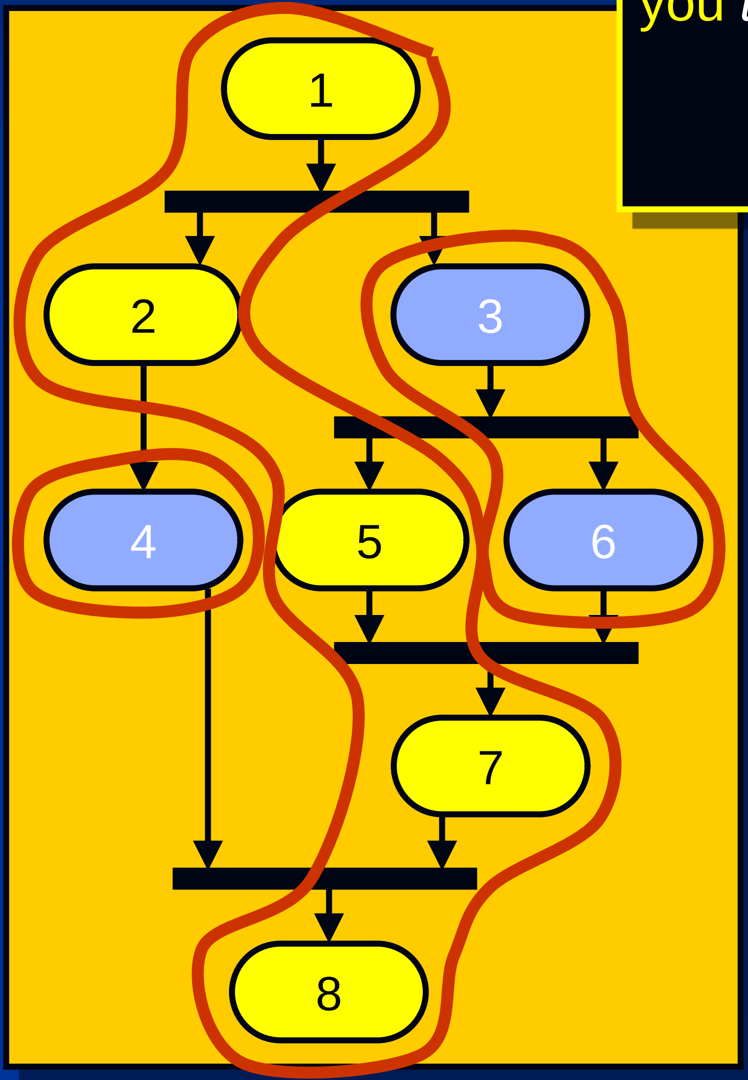
Effects of Task-Core Assignment



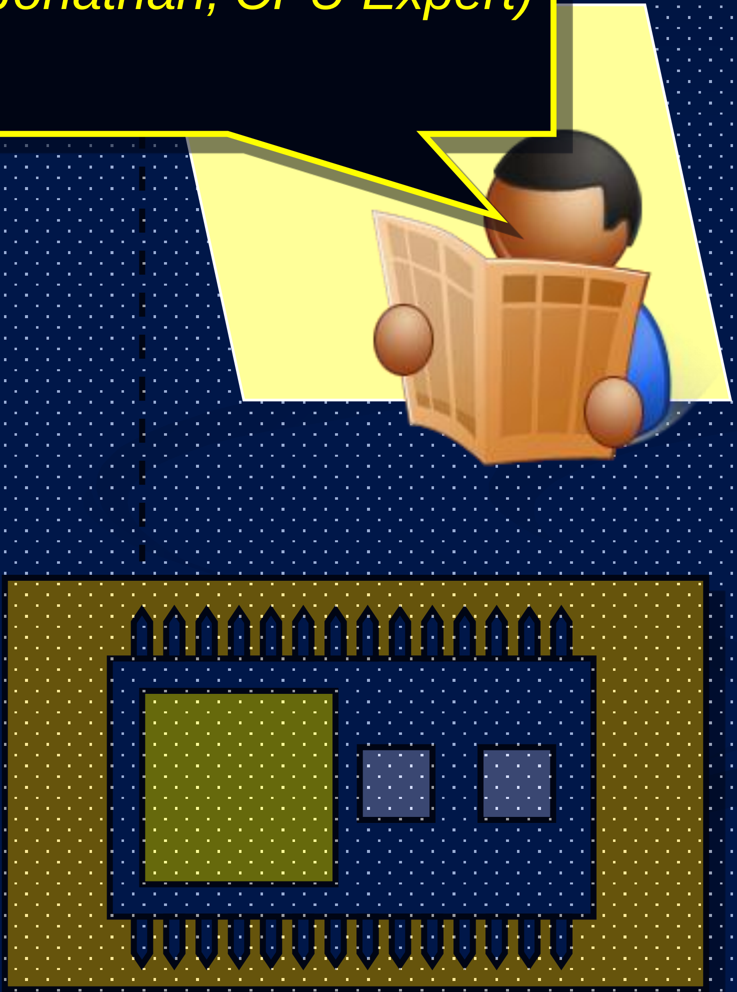
“To *exploit the benefits* of the shining new CPU…”
(Jonathan, CPU Expert)



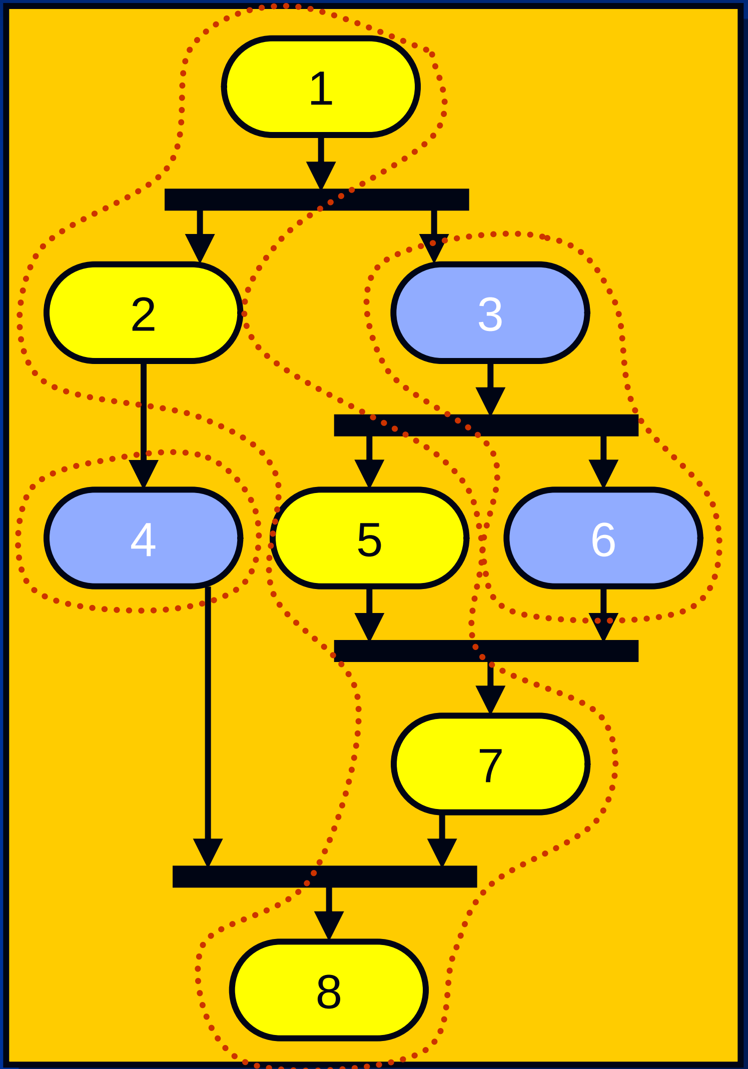
Effects of Task-Core Ass

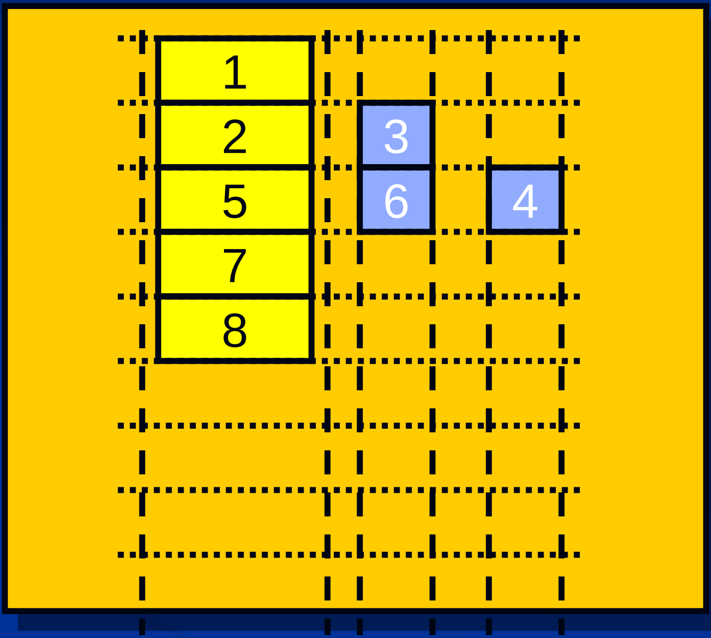


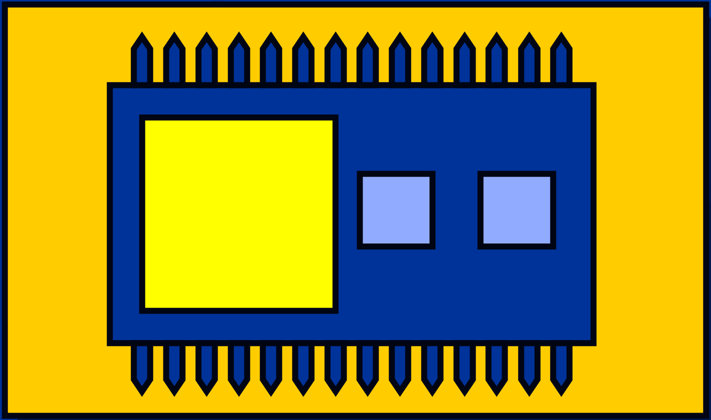
"…you only have to *re-organize* you *threads* this way…"
(Jonathan, CPU Expert)



Effects of Task-Core Assignment

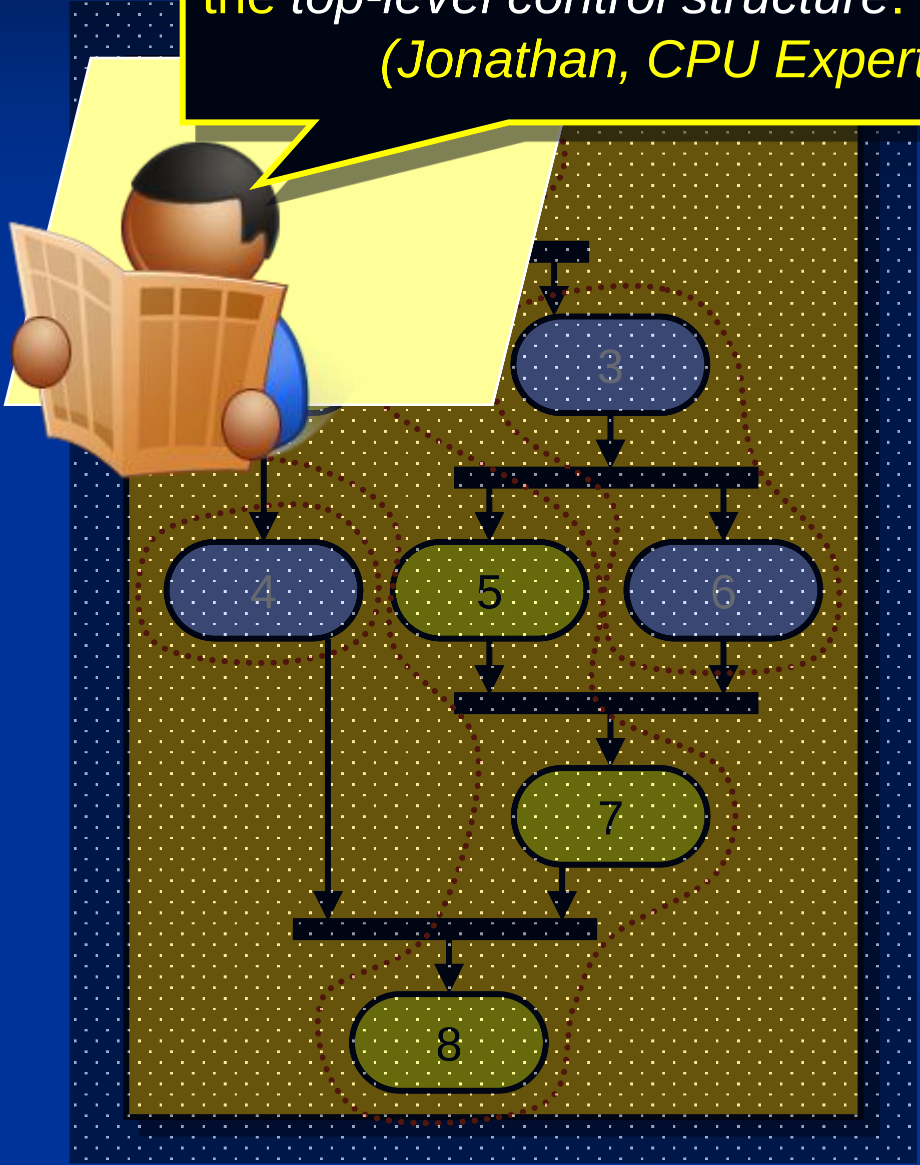


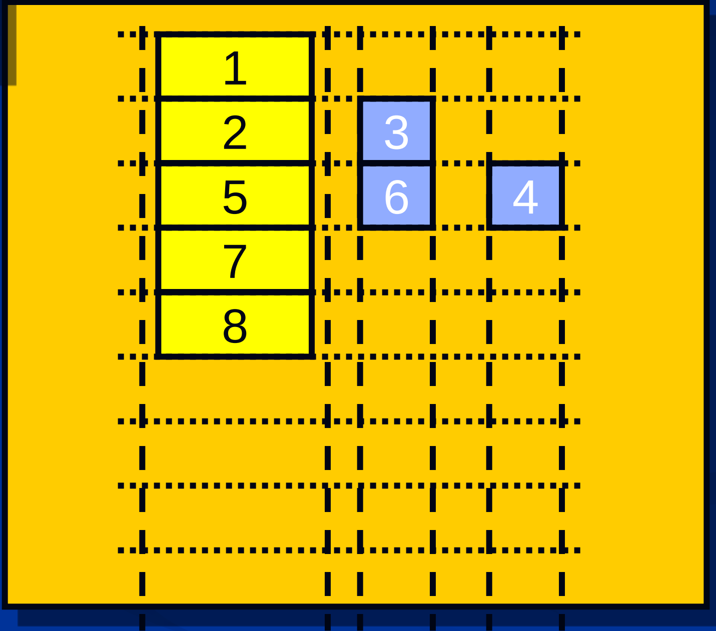


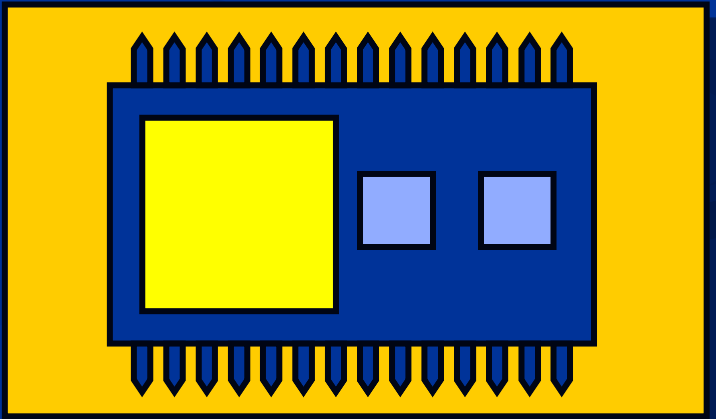


Task-Core ment

"…and *look*: spared some more time. Just had to *re-organize* the implementation of the *top-level control structure*."
(Jonathan, CPU Expert)



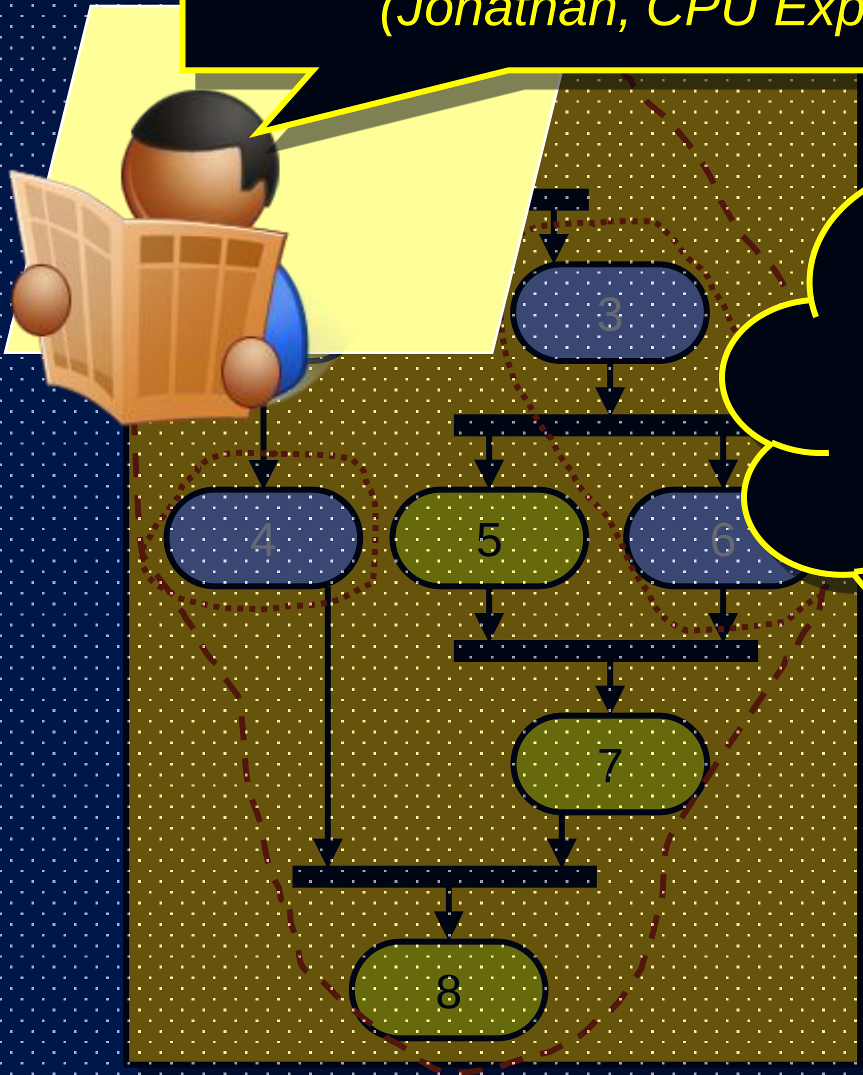


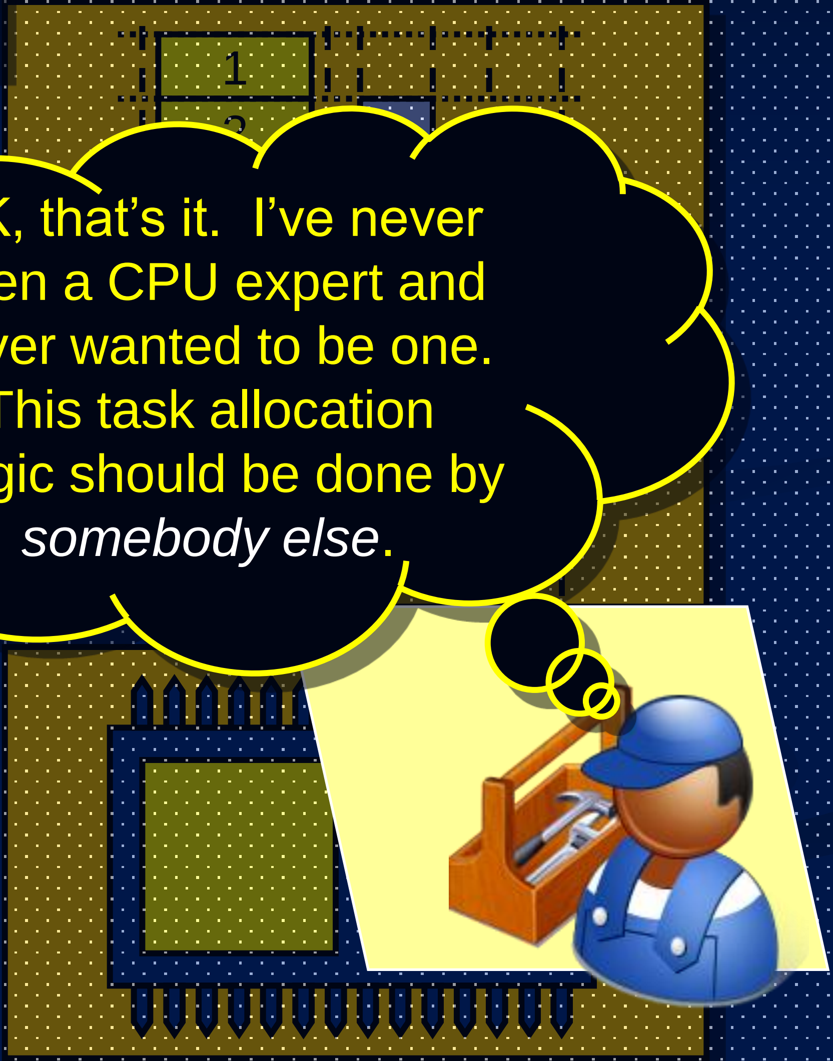


Task-Core …ment

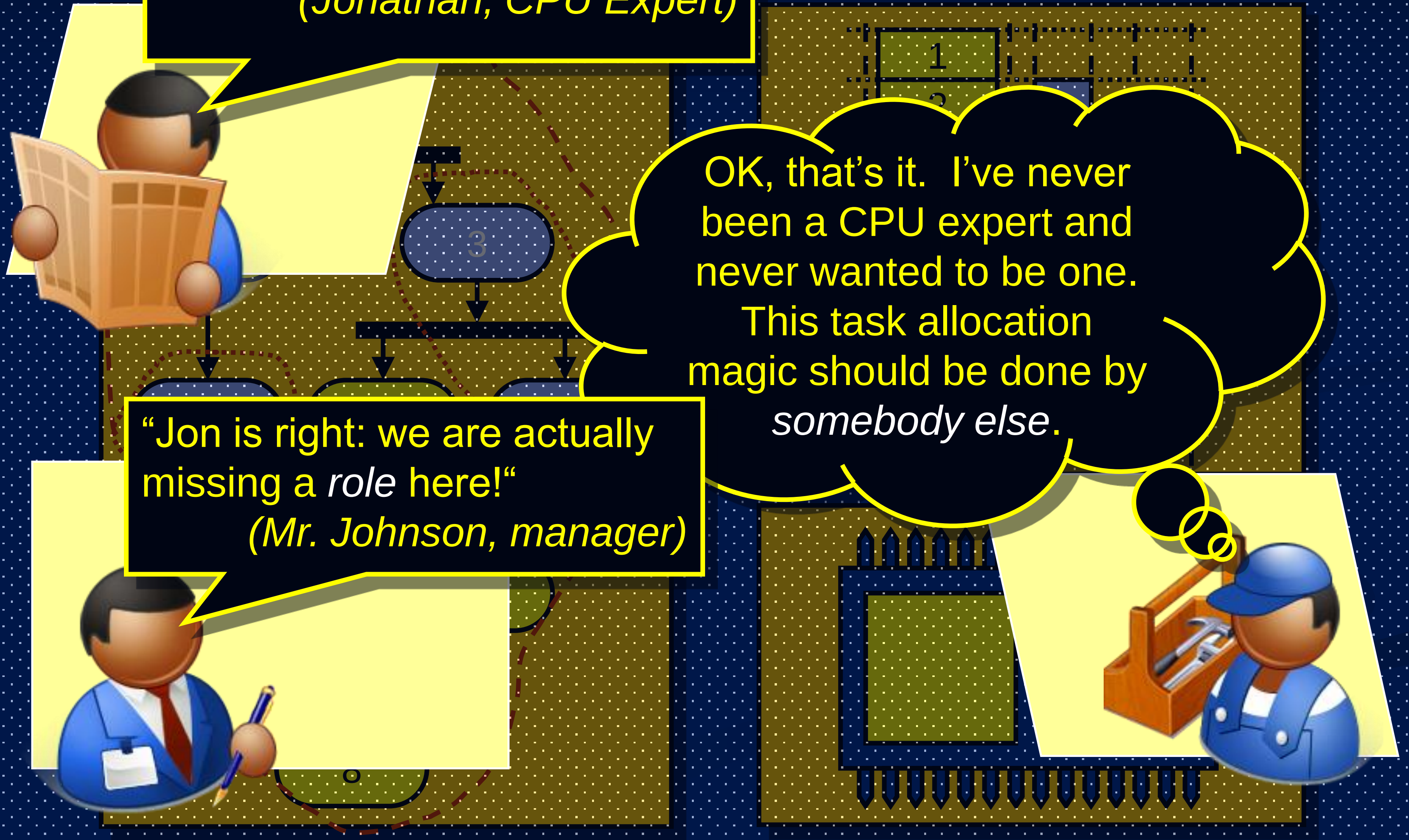
"…and *look*: spared some more time. Just had to *re-organize* the implementation of the *top-level control structure*."
(Jonathan, CPU Expert)

OK, that's it. I've never been a CPU expert and never wanted to be one. This task allocation magic should be done by *somebody else*.





Task-Core ment



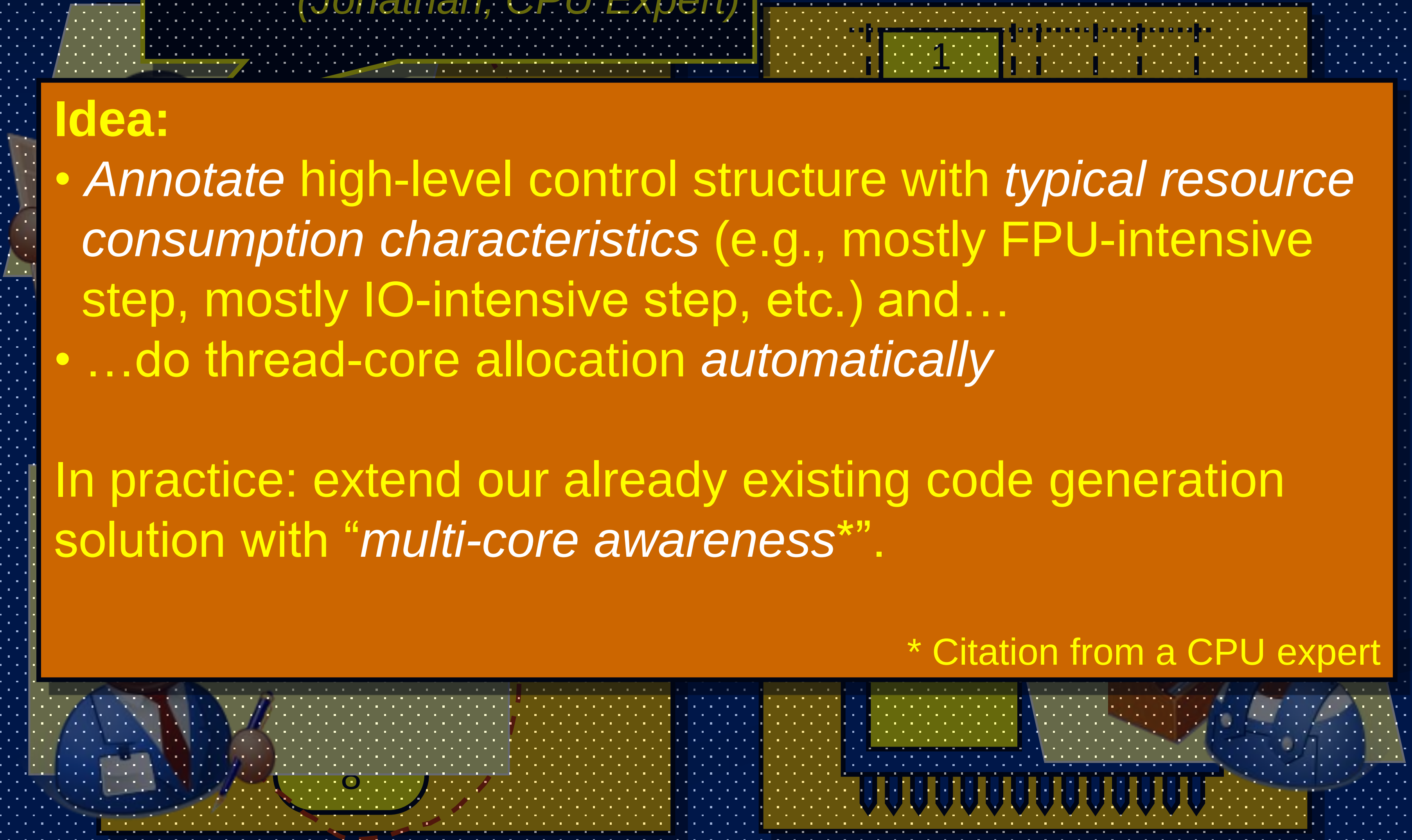
"…and *look*: spared some more time. Just had to re-write the *top-level control structure*."
(*Jonathan, CPU Expert*)

"Jon is right: we are actually missing a *role* here!"
(*Mr. Johnson, manager*)

OK, that's it. I've never been a CPU expert and never wanted to be one. This task allocation magic should be done by *somebody else*.

ask-Core ment

"…and *look*: spared some more time. Just had to re-write the *top-level control structure*."
(Jonathan, CPU Expert)



Idea:

- *Annotate* high-level control structure with *typical resource consumption characteristics* (e.g., mostly FPU-intensive step, mostly IO-intensive step, etc.) and…
- …do thread-core allocation *automatically*

In practice: extend our already existing code generation solution with "*multi-core awareness**".

* Citation from a CPU expert

Presentation Structure

Wide Context and Future Research Goals

Achievements Until Now

Demonstration

Presentation Structure

Wide Context and Future Research Goals

Achievements Until Now

Demonstration

Achievements Until Now…

- Understanding complex control structures
  - Unambiguous formal semantics for UML statecharts
    - Mapped to Kripke transition systems
  - Relations of activities expressed by *PERT-graphs*
    - Support for arbitrary complexity
    - This is the entry point for multi-core awareness!
- Automatic implementation of control structures
  - Automatic code synthesis for ANSI-C and Java
    - Demonstrated even on a *Mitmot* device
  - …actually schedules the precisely calculated activity PERT graphs to a single thread…

Achievements Until Now…

- Runtime verification
  - Reference specification:
    - UML Statecharts
    - Temporal correctness criteria (PLTL)
  - Not even mentioned here but at least as important as code synthesis

Achievements Until Now…

Solving Jon’s Problem…

- [✓] Formal Semantics for Statecharts
- [✓] Control Code Synthesis (Multi-Core Unaware)
- [ ] Resource/Multi-Core Awareness…

Still working…

Achievements Until Now...

Solving Jon's Problem...

- ✓ Formal Semantics for Statecharts
- ✓ Control Code Synthesis (Multi-Core Unaware)
- ☐ wareness...

By the way, this was my PhD thesis…

Still working...

Achievements Unti

Solving Jon’s Problem...

- ✓ Formal Semantics for Statecharts
- ✓ Control Code Synthesis (Multi-Core Unaware)
- ☐ wareness...

By the way, this was my PhD thesis…

Still working...

Presentation Structure

Wide Context and Future Research Goals

Achievements Until Now

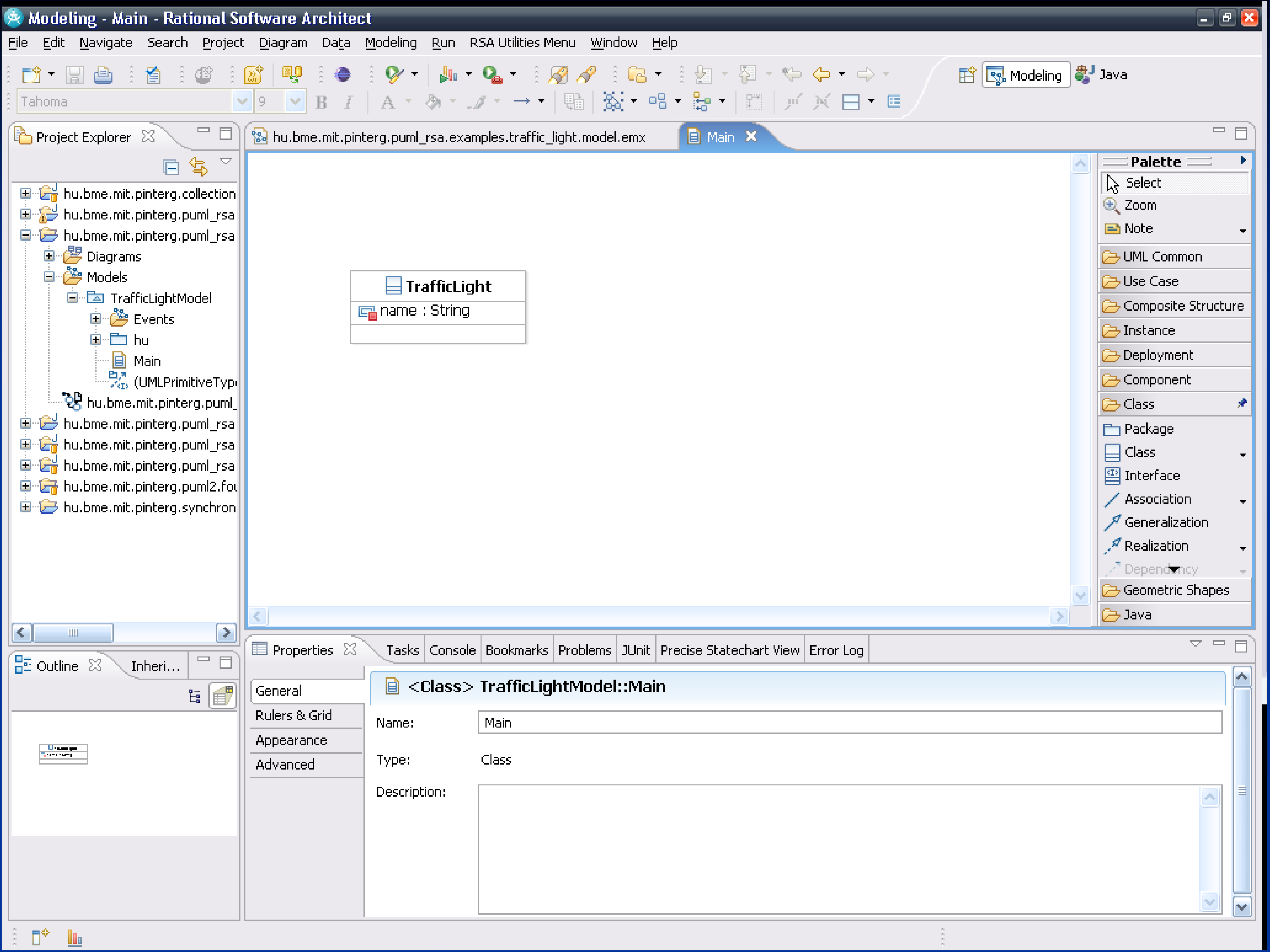
Demonstration

Presentation Structure

Wide Context and Future Research Goals

Achievements Until Now

Demonstration

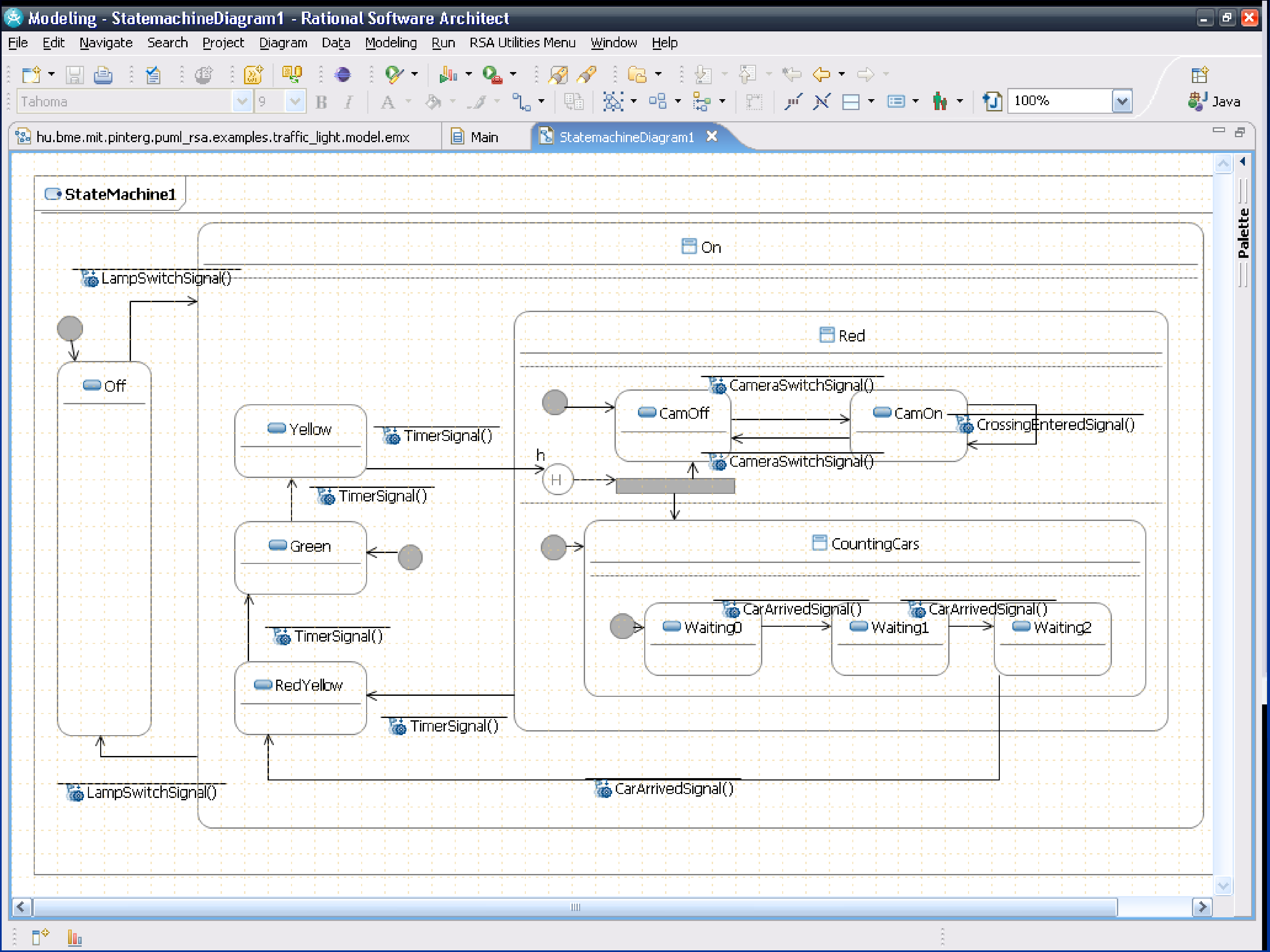


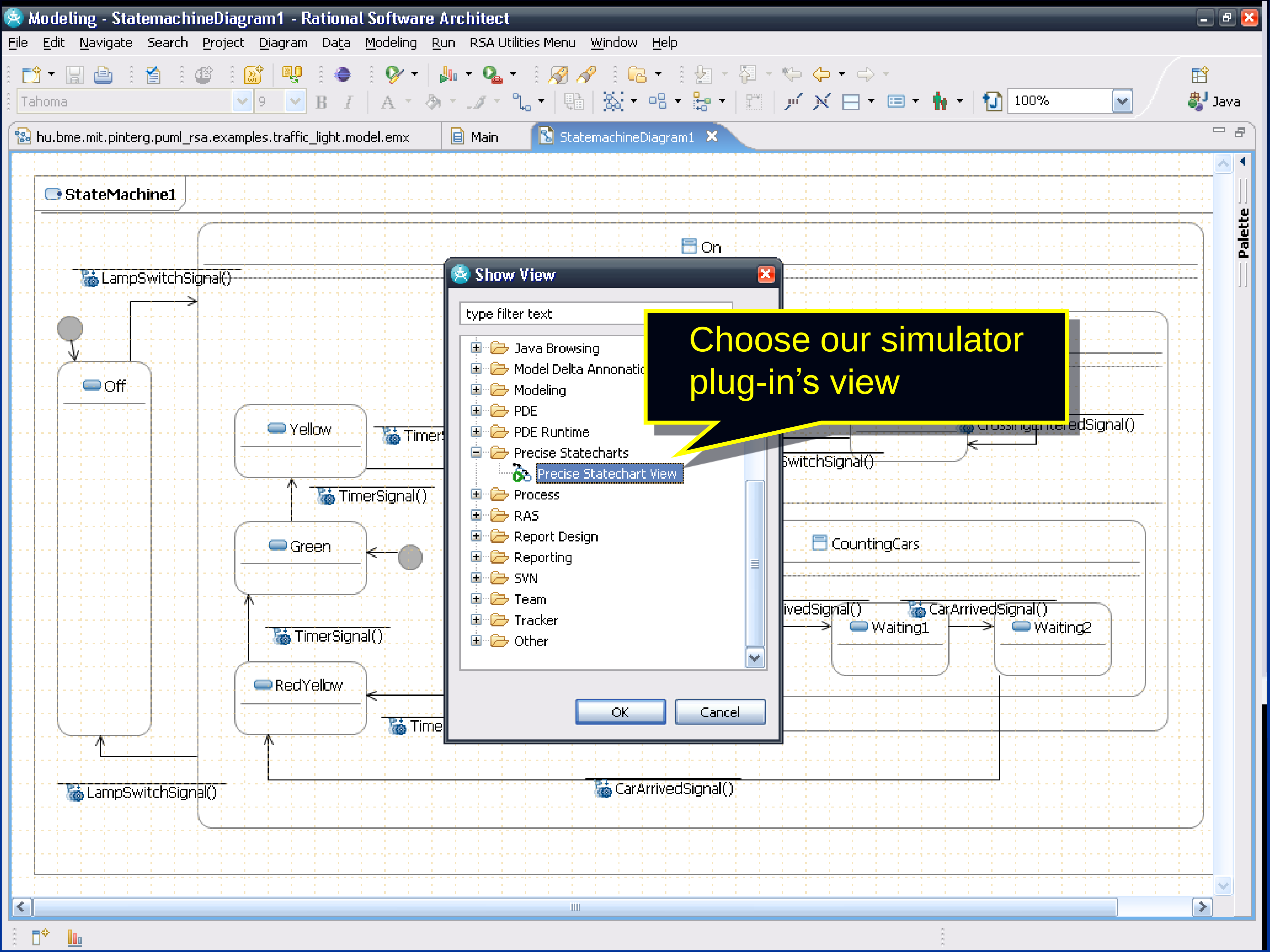
<Class> TrafficLightModel::Main

Name: Main

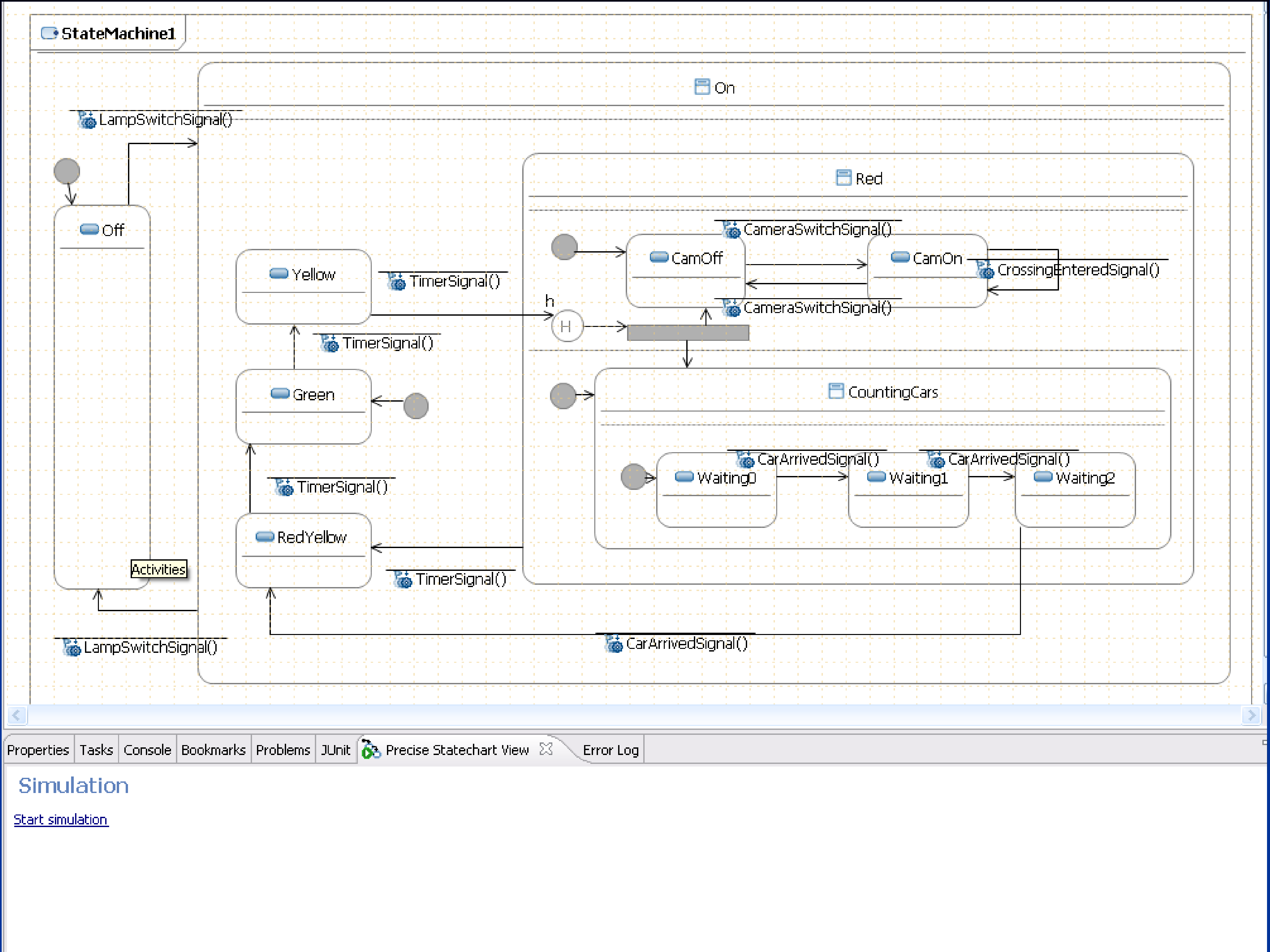
Type: Class

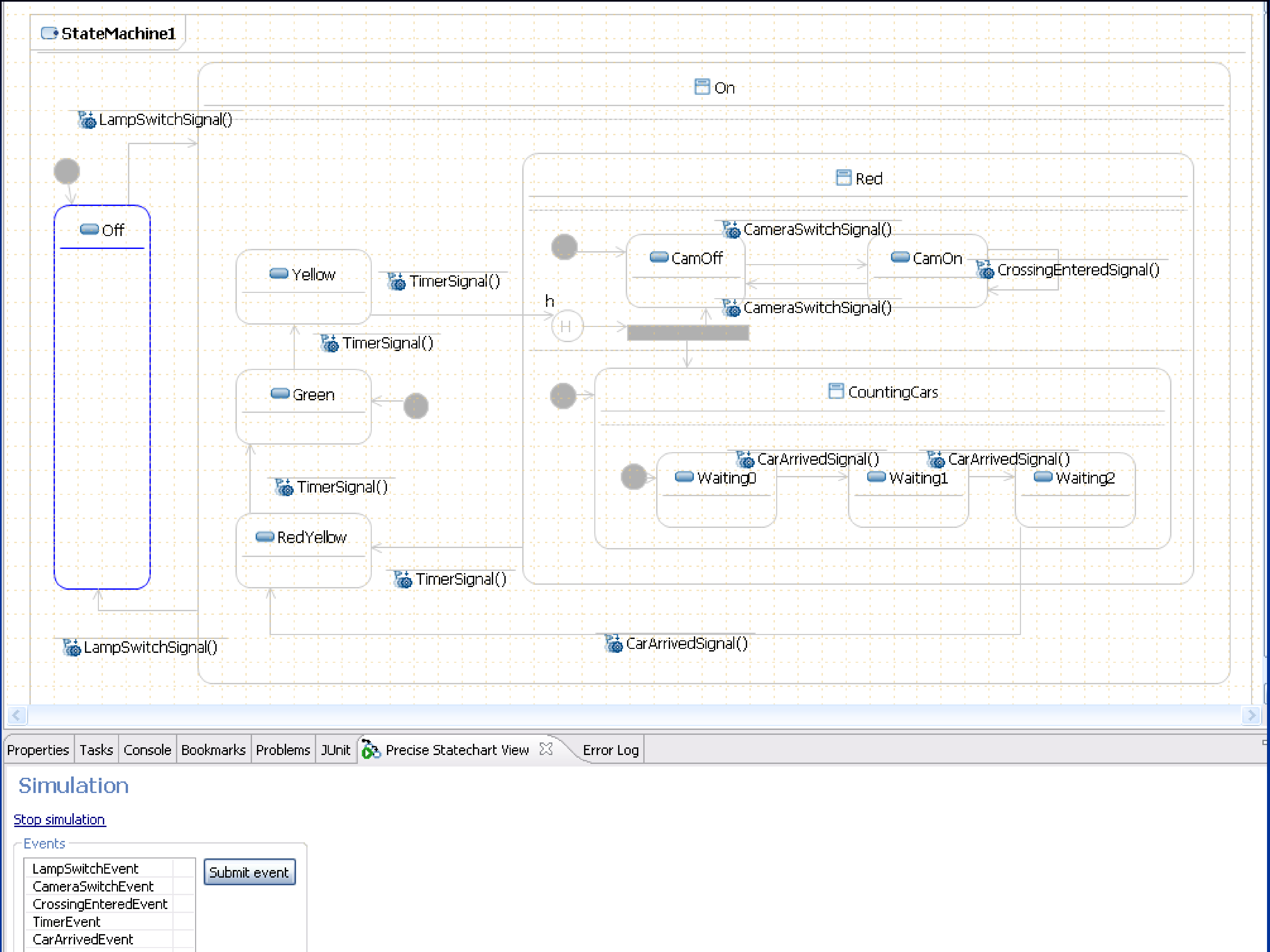
Description:


Modeling - StatemachineDiagram1 - Rational Software Architect
File Edit Navigate Search Project Diagram Data Modeling Run RSA Utilities Menu Window Help
Tahoma
9
100%
Java
hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.model.emx
Main
StatemachineDiagram1
StateMachine1
On
LampSwitchSignal()
Off
Red
CameraSwitchSignal()
CamOff
CamOn
CrossingEnteredSignal()
CameraSwitchSignal()
Yellow
TimerSignal()
h
H
TimerSignal()
Green
CountingCars
CarArrivedSignal()
CarArrivedSignal()
Waiting0
Waiting1
Waiting2
TimerSignal()
RedYellow
TimerSignal()
LampSwitchSignal()
CarArrivedSignal()
Palette



Choose our simulator plug-in’s view


StateMachine1
On
LampSwitchSignal()
Off
Red
CameraSwitchSignal()
CamOff
CamOn
CrossingEnteredSignal()
CameraSwitchSignal()
Yellow
TimerSignal()
h
H
TimerSignal()
Green
CountingCars
TimerSignal()
CarArrivedSignal()
CarArrivedSignal()
Waiting0
Waiting1
Waiting2
RedYellow
Activities
TimerSignal()
LampSwitchSignal()
CarArrivedSignal()
Properties
Tasks
Console
Bookmarks
Problems
JUnit
Precise Statechart View
Error Log
Simulation
Start simulation

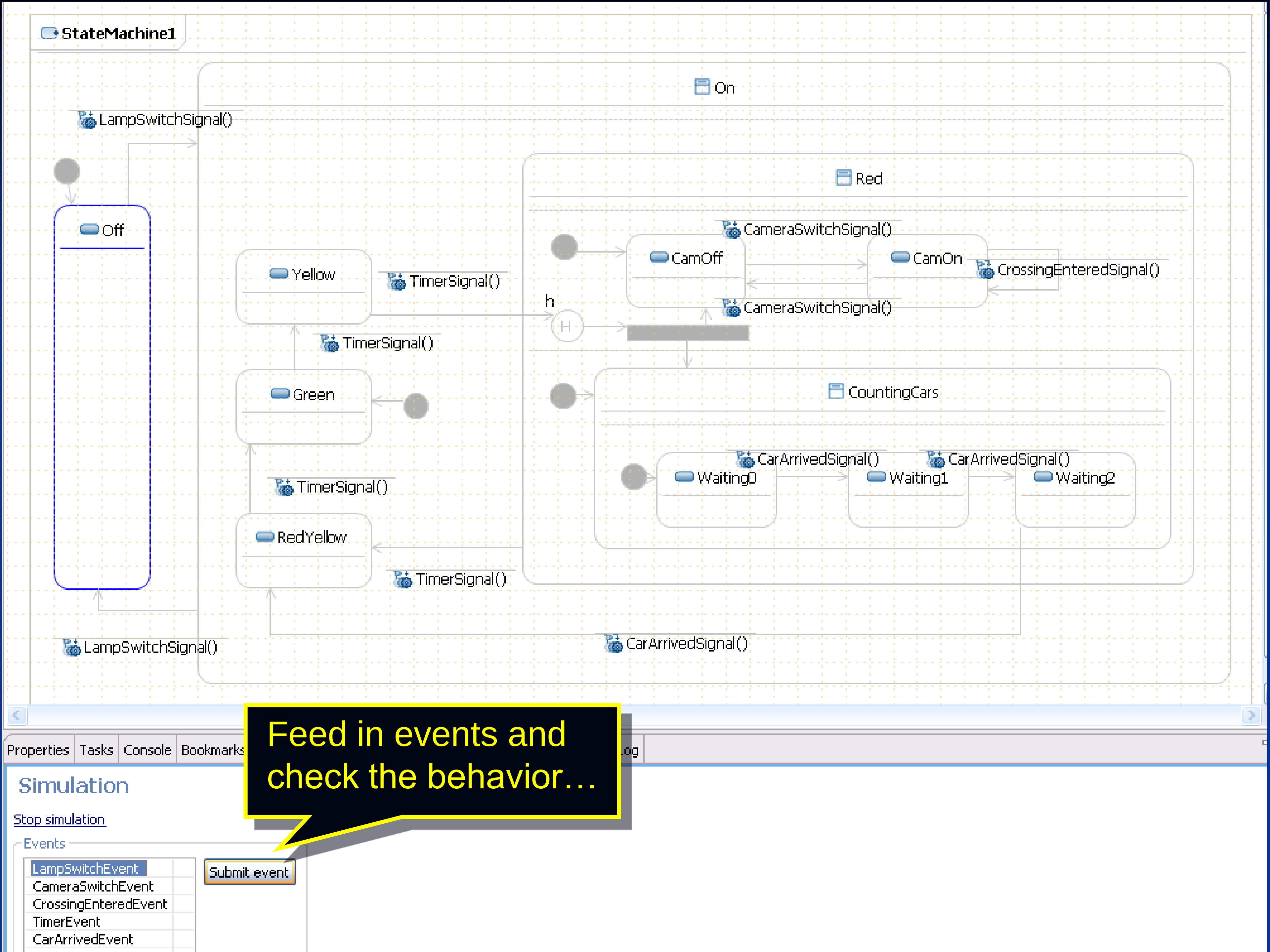


Simulation

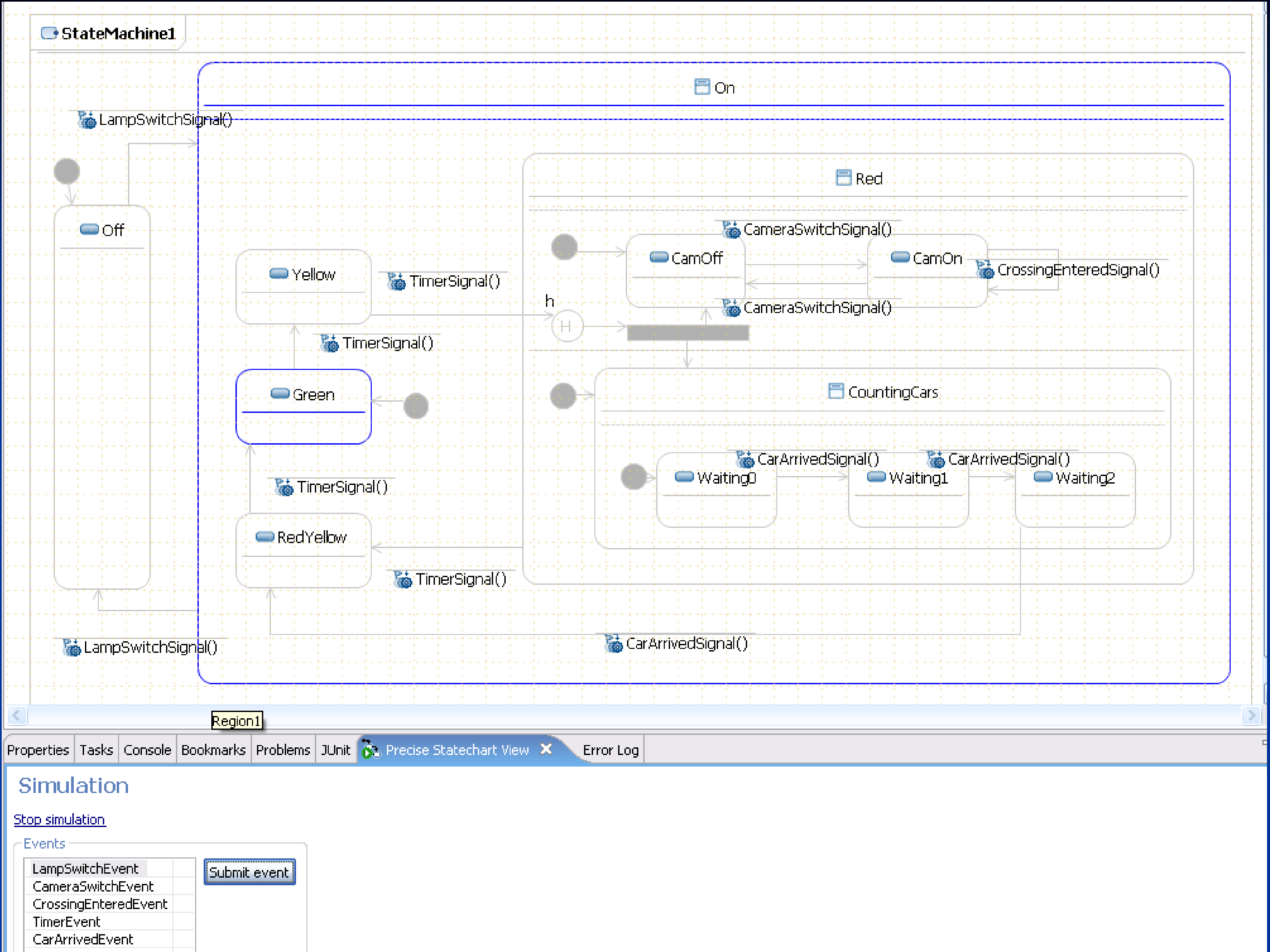
Stop simulation

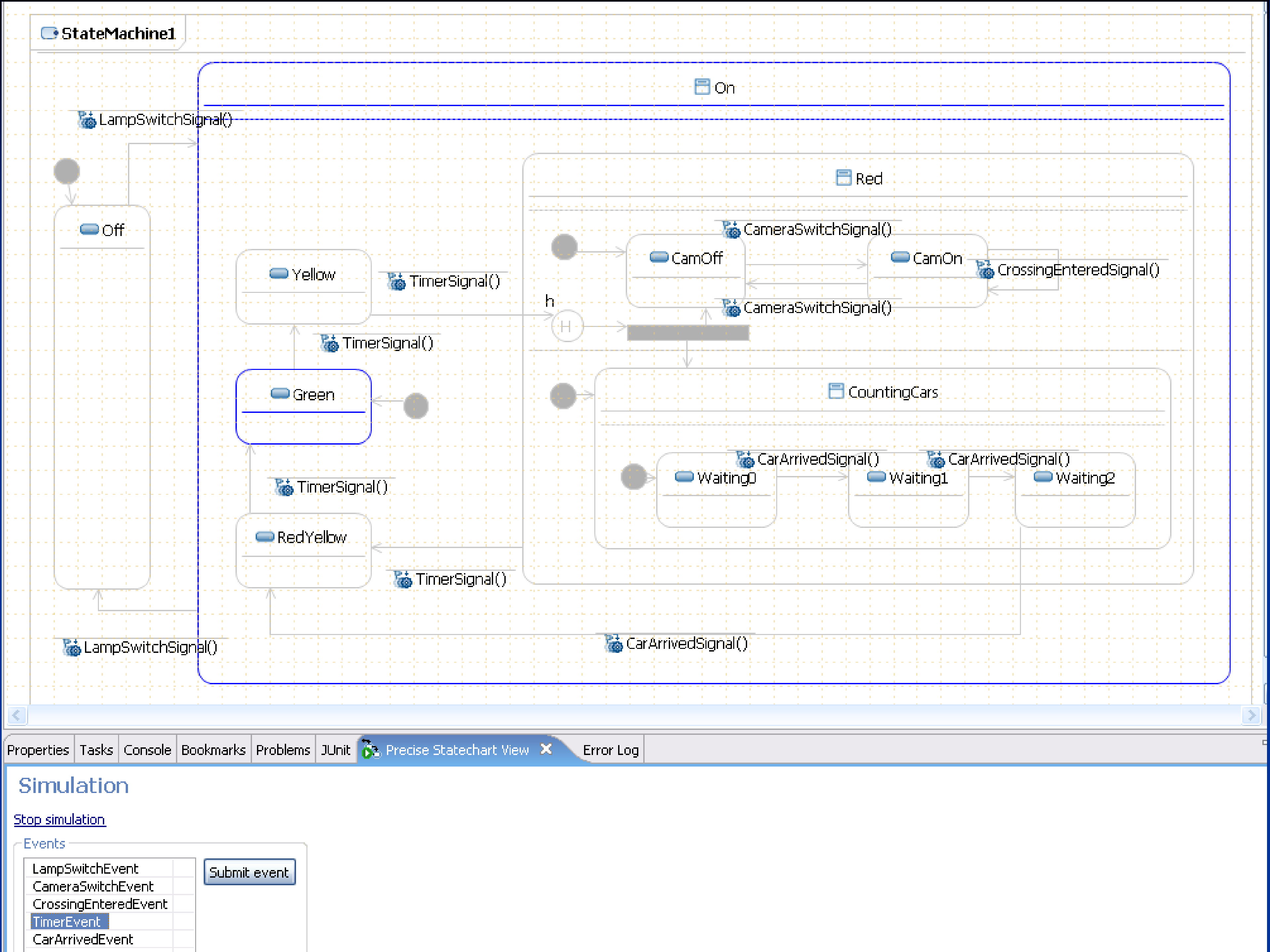
Events

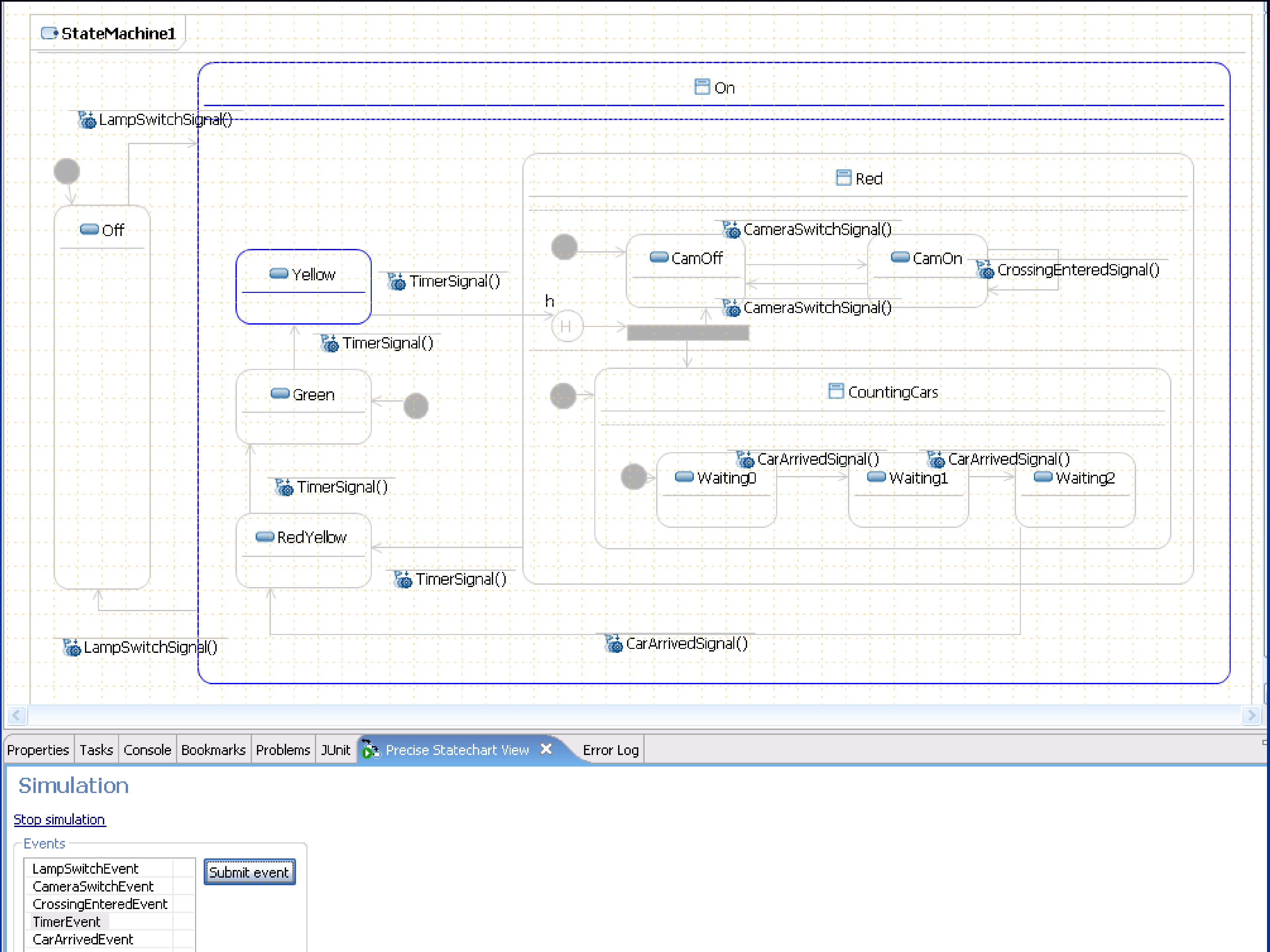
LampSwitchEvent
CameraSwitchEvent
CrossingEnteredEvent
TimerEvent
CarArrivedEvent

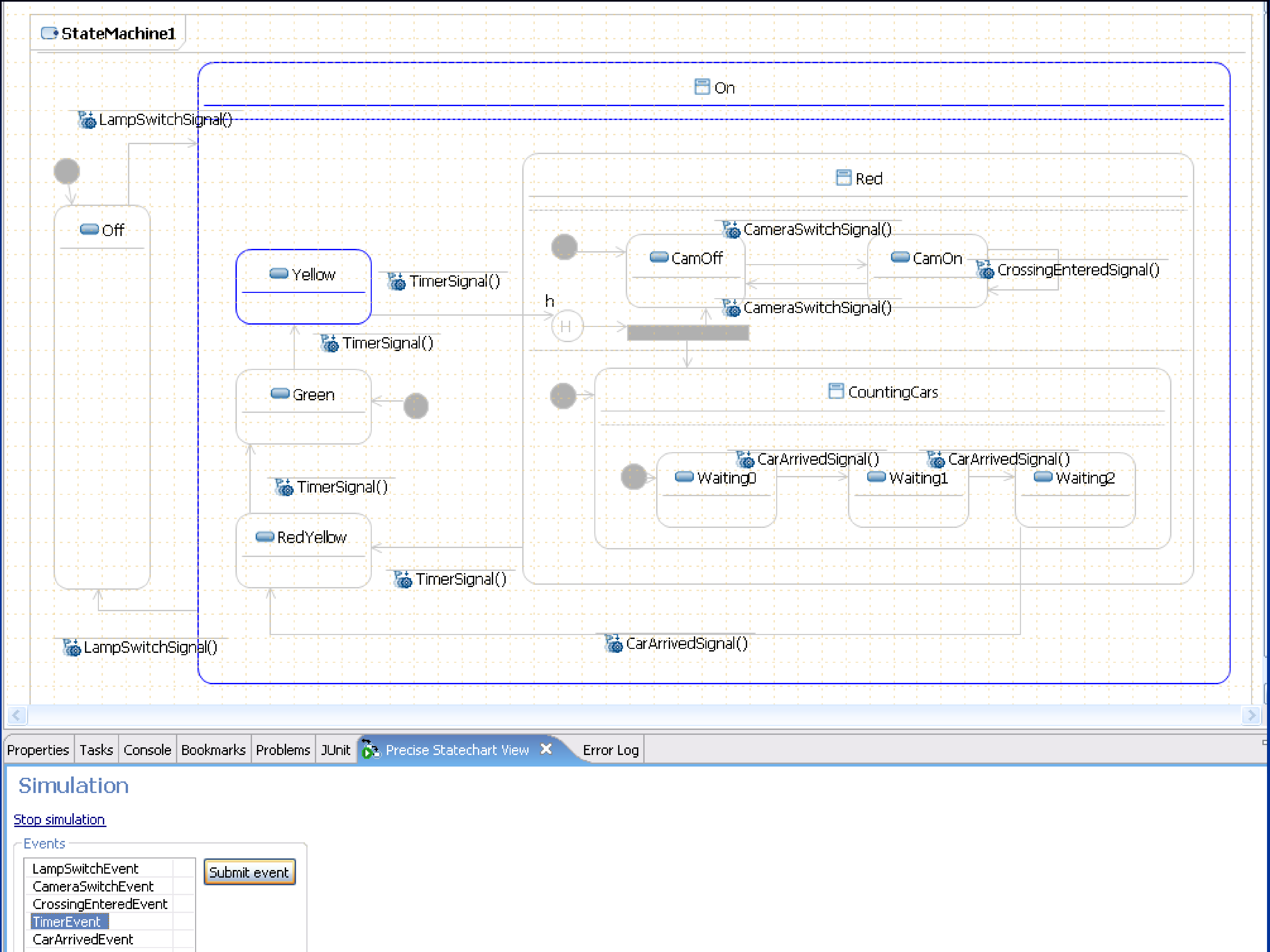


Feed in events and check the behavior…


StateMachine1
On
LampSwitchSignal()
Off
Red
CameraSwitchSignal()
CamOff
CamOn
CrossingEnteredSignal()
CameraSwitchSignal()
Yellow
TimerSignal()
h
H
TimerSignal()
Green
CountingCars
CarArrivedSignal()
CarArrivedSignal()
Waiting0
Waiting1
Waiting2
TimerSignal()
RedYellow
TimerSignal()
LampSwitchSignal()
CarArrivedSignal()
Region1
Properties
Tasks
Console
Bookmarks
Problems
JUnit
Precise Statechart View
Error Log
Simulation
Stop simulation
Events
LampSwitchEvent
CameraSwitchEvent
CrossingEnteredEvent
TimerEvent
CarArrivedEvent
Submit event


StateMachine1
On
LampSwitchSignal()
Off
Yellow
TimerSignal()
Green
TimerSignal()
TimerSignal()
RedYellow
TimerSignal()
Red
CameraSwitchSignal()
CamOff
CamOn
CrossingEnteredSignal()
CameraSwitchSignal()
h
H
CountingCars
CarArrivedSignal()
CarArrivedSignal()
Waiting0
Waiting1
Waiting2
CarArrivedSignal()
LampSwitchSignal()
Properties
Tasks
Console
Bookmarks
Problems
JUnit
Precise Statechart View
Error Log
Simulation
Stop simulation
Events
LampSwitchEvent
CameraSwitchEvent
CrossingEnteredEvent
TimerEvent
CarArrivedEvent
Submit event


StateMachine1
On
LampSwitchSignal()
Off
Yellow
TimerSignal()
h
H
Red
CameraSwitchSignal()
CamOff
CamOn
CrossingEnteredSignal()
CameraSwitchSignal()
TimerSignal()
Green
CountingCars
CarArrivedSignal()
CarArrivedSignal()
Waiting0
Waiting1
Waiting2
TimerSignal()
RedYellow
TimerSignal()
LampSwitchSignal()
CarArrivedSignal()
Properties
Tasks
Console
Bookmarks
Problems
JUnit
Precise Statechart View
Error Log
Simulation
Stop simulation
Events
LampSwitchEvent
CameraSwitchEvent
CrossingEnteredEvent
TimerEvent
CarArrivedEvent
Submit event

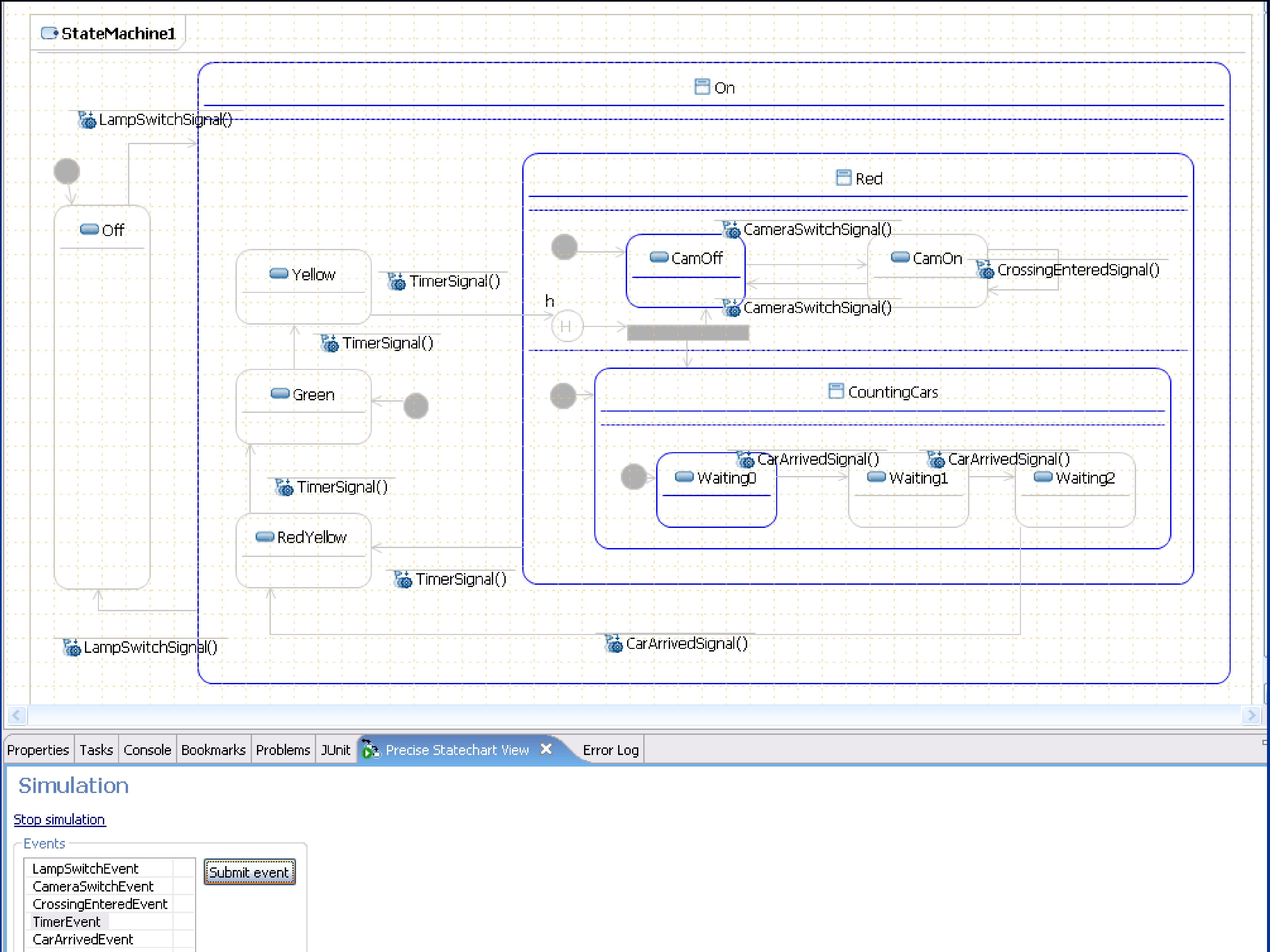


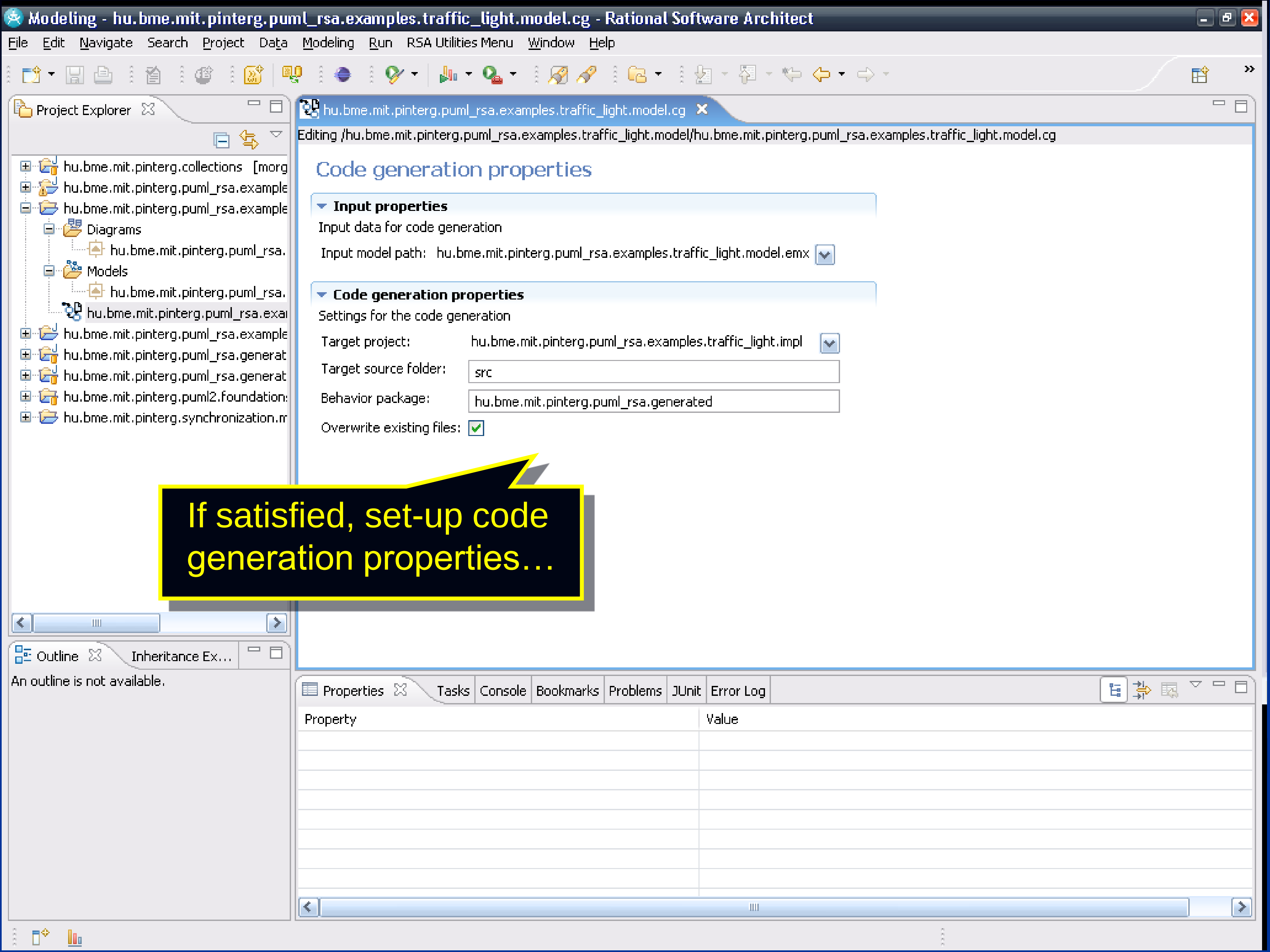
Simulation

Stop simulation

Events

LampSwitchEvent
CameraSwitchEvent
CrossingEnteredEvent
TimerEvent
CarArrivedEvent


StateMachine1
On
LampSwitchSignal()
Off
Red
CameraSwitchSignal()
CamOff
CamOn
CrossingEnteredSignal()
CameraSwitchSignal()
Yellow
TimerSignal()
h
H
TimerSignal()
Green
CountingCars
CarArrivedSignal()
CarArrivedSignal()
Waiting0
Waiting1
Waiting2
TimerSignal()
RedYellow
TimerSignal()
LampSwitchSignal()
CarArrivedSignal()
Properties
Tasks
Console
Bookmarks
Problems
JUnit
Precise Statechart View
Error Log
Simulation
Stop simulation
Events
LampSwitchEvent
CameraSwitchEvent
CrossingEnteredEvent
TimerEvent
CarArrivedEvent
Submit event


Modeling - hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.model.cg - Rational Software Architect

Editing /hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.model/hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.model.cg

Code generation properties

Input properties

Input data for code generation

Input model path: hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.model.emx

Code generation properties

Settings for the code generation

Target project: hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.impl

Target source folder: src

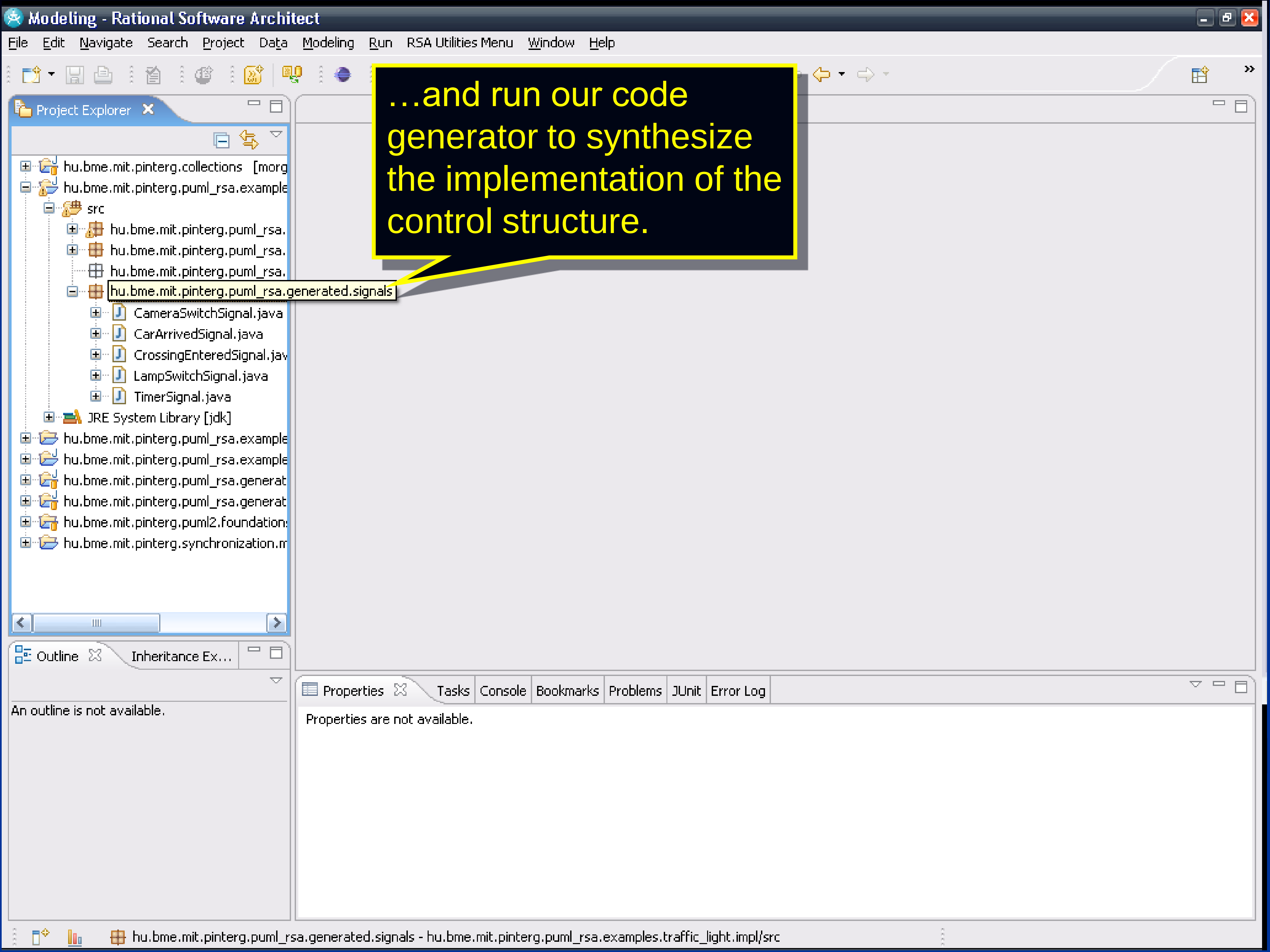
Behavior package: hu.bme.mit.pinterg.puml_rsa.generated

Overwrite existing files: ✓

An outline is not available.

If satisfied, set-up code generation properties…

File Edit Navigate Search Project Data Modeling Run RSA Utilities Menu Window Help
Project Explorer
hu.bme.mit.pinterg.collections [morg
hu.bme.mit.pinterg.puml_rsa.example
hu.bme.mit.pinterg.puml_rsa.example
Diagrams
hu.bme.mit.pinterg.puml_rsa.
Models
hu.bme.mit.pinterg.puml_rsa.
hu.bme.mit.pinterg.puml_rsa.exa
hu.bme.mit.pinterg.puml_rsa.example
hu.bme.mit.pinterg.puml_rsa.generat
hu.bme.mit.pinterg.puml_rsa.generat
hu.bme.mit.pinterg.puml2.foundation
hu.bme.mit.pinterg.synchronization.m
Outline
Inheritance Ex...
Properties Tasks Console Bookmarks Problems JUnit Error Log
Property
Value



…and run our code generator to synthesize the implementation of the control structure.

Modeling - Rational Software Architect

File Edit Navigate Search Project Data Modeling Run RSA Utilities Menu Window Help

Project Explorer

- hu.bme.mit.pinterg.collections [morg
- hu.bme.mit.pinterg.puml_rsa.example
  - src
    - hu.bme.mit.pinterg.puml_rsa.
    - hu.bme.mit.pinterg.puml_rsa.
    - hu.bme.mit.pinterg.puml_rsa.
    - hu.bme.mit.pinterg.puml_rsa.generated.signals
      - CameraSwitchSignal.java
      - CarArrivedSignal.java
      - CrossingEnteredSignal.jav
      - LampSwitchSignal.java
      - TimerSignal.java
  - JRE System Library [jdk]
- hu.bme.mit.pinterg.puml_rsa.example
- hu.bme.mit.pinterg.puml_rsa.example
- hu.bme.mit.pinterg.puml_rsa.generat
- hu.bme.mit.pinterg.puml_rsa.generat
- hu.bme.mit.pinterg.puml2.foundation
- hu.bme.mit.pinterg.synchronization.m

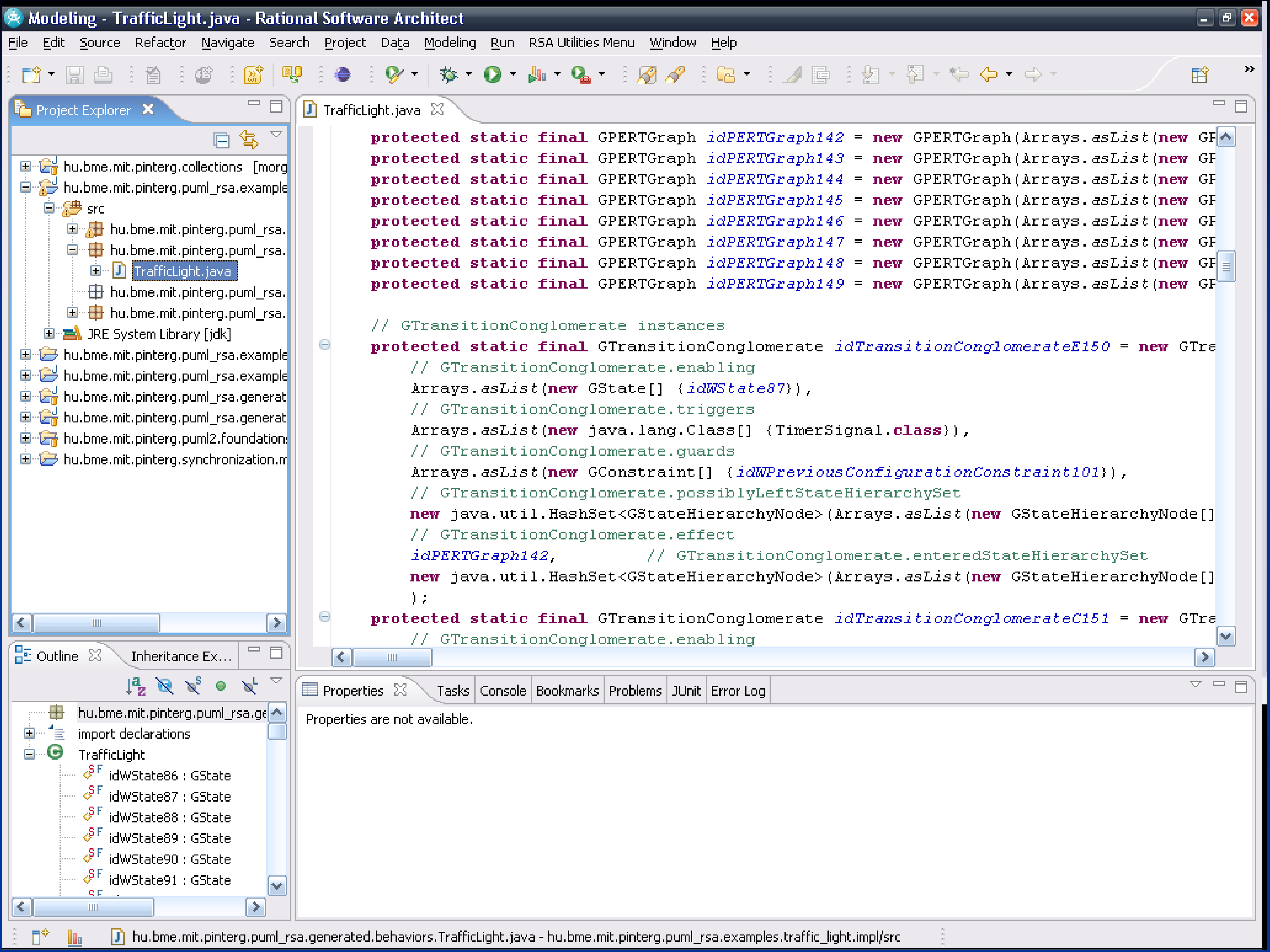
Outline | Inheritance Ex...

An outline is not available.

Properties | Tasks | Console | Bookmarks | Problems | JUnit | Error Log

Properties are not available.

hu.bme.mit.pinterg.puml_rsa.generated.signals - hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.impl/src



Modeling - TrafficLight.java - Rational Software Architect

File Edit Source Refactor Navigate Search Project Data Modeling Run RSA Utilities Menu Window Help

Project Explorer

```
⊞ hu.bme.mit.pinterg.collections  [morg
⊟ hu.bme.mit.pinterg.puml_rsa.example
   ⊟ src
      ⊞ hu.bme.mit.pinterg.puml_rsa.
      ⊟ hu.bme.mit.pinterg.puml_rsa.
         ⊞ TrafficLight.java
         hu.bme.mit.pinterg.puml_rsa.
      ⊞ hu.bme.mit.pinterg.puml_rsa.
   ⊞ JRE System Library [jdk]
⊞ hu.bme.mit.pinterg.puml_rsa.example
⊞ hu.bme.mit.pinterg.puml_rsa.example
⊞ hu.bme.mit.pinterg.puml_rsa.generat
⊞ hu.bme.mit.pinterg.puml_rsa.generat
⊞ hu.bme.mit.pinterg.puml2.foundation:
⊞ hu.bme.mit.pinterg.synchronization.m
```

TrafficLight.java

```java
    protected static final GPERTGraph idPERTGraph142 = new GPERTGraph(Arrays.asList(new GF
    protected static final GPERTGraph idPERTGraph143 = new GPERTGraph(Arrays.asList(new GF
    protected static final GPERTGraph idPERTGraph144 = new GPERTGraph(Arrays.asList(new GF
    protected static final GPERTGraph idPERTGraph145 = new GPERTGraph(Arrays.asList(new GF
    protected static final GPERTGraph idPERTGraph146 = new GPERTGraph(Arrays.asList(new GF
    protected static final GPERTGraph idPERTGraph147 = new GPERTGraph(Arrays.asList(new GF
    protected static final GPERTGraph idPERTGraph148 = new GPERTGraph(Arrays.asList(new GF
    protected static final GPERTGraph idPERTGraph149 = new GPERTGraph(Arrays.asList(new GF

    // GTransitionConglomerate instances
    protected static final GTransitionConglomerate idTransitionConglomerateE150 = new GTra
        // GTransitionConglomerate.enabling
        Arrays.asList(new GState[] {idWState87}),
        // GTransitionConglomerate.triggers
        Arrays.asList(new java.lang.Class[] {TimerSignal.class}),
        // GTransitionConglomerate.guards
        Arrays.asList(new GConstraint[] {idWPreviousConfigurationConstraint101}),
        // GTransitionConglomerate.possiblyLeftStateHierarchySet
        new java.util.HashSet<GStateHierarchyNode>(Arrays.asList(new GStateHierarchyNode[]
        // GTransitionConglomerate.effect
        idPERTGraph142,         // GTransitionConglomerate.enteredStateHierarchySet
        new java.util.HashSet<GStateHierarchyNode>(Arrays.asList(new GStateHierarchyNode[]
        );
    protected static final GTransitionConglomerate idTransitionConglomerateC151 = new GTra
        // GTransitionConglomerate.enabling
```

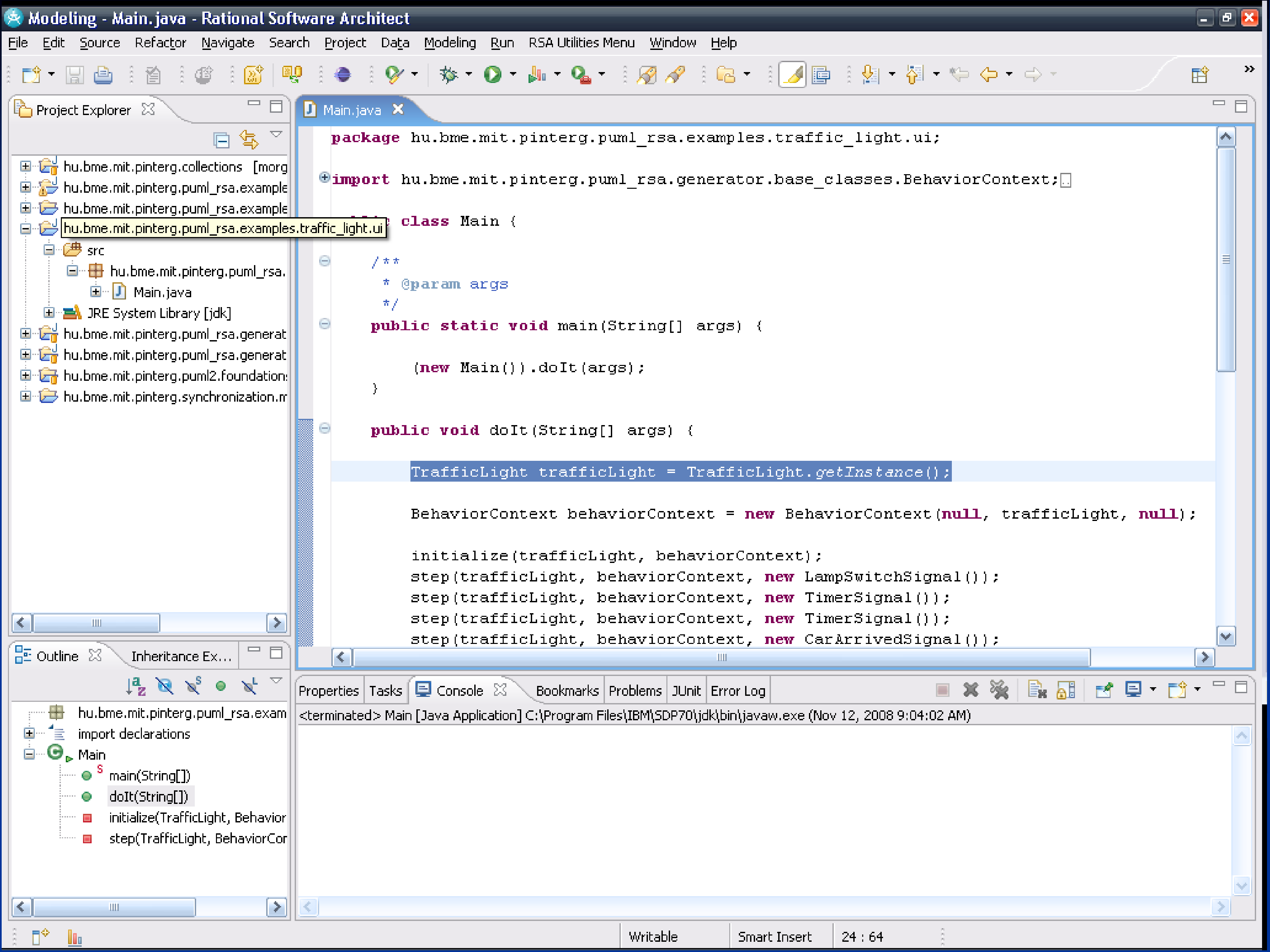
Outline | Inheritance Ex...

```
hu.bme.mit.pinterg.puml_rsa.ge
⊞ import declarations
⊟ TrafficLight
    idWState86 : GState
    idWState87 : GState
    idWState88 : GState
    idWState89 : GState
    idWState90 : GState
    idWState91 : GState
```

Properties | Tasks | Console | Bookmarks | Problems | JUnit | Error Log

Properties are not available.

hu.bme.mit.pinterg.puml_rsa.generated.behaviors.TrafficLight.java - hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.impl/src



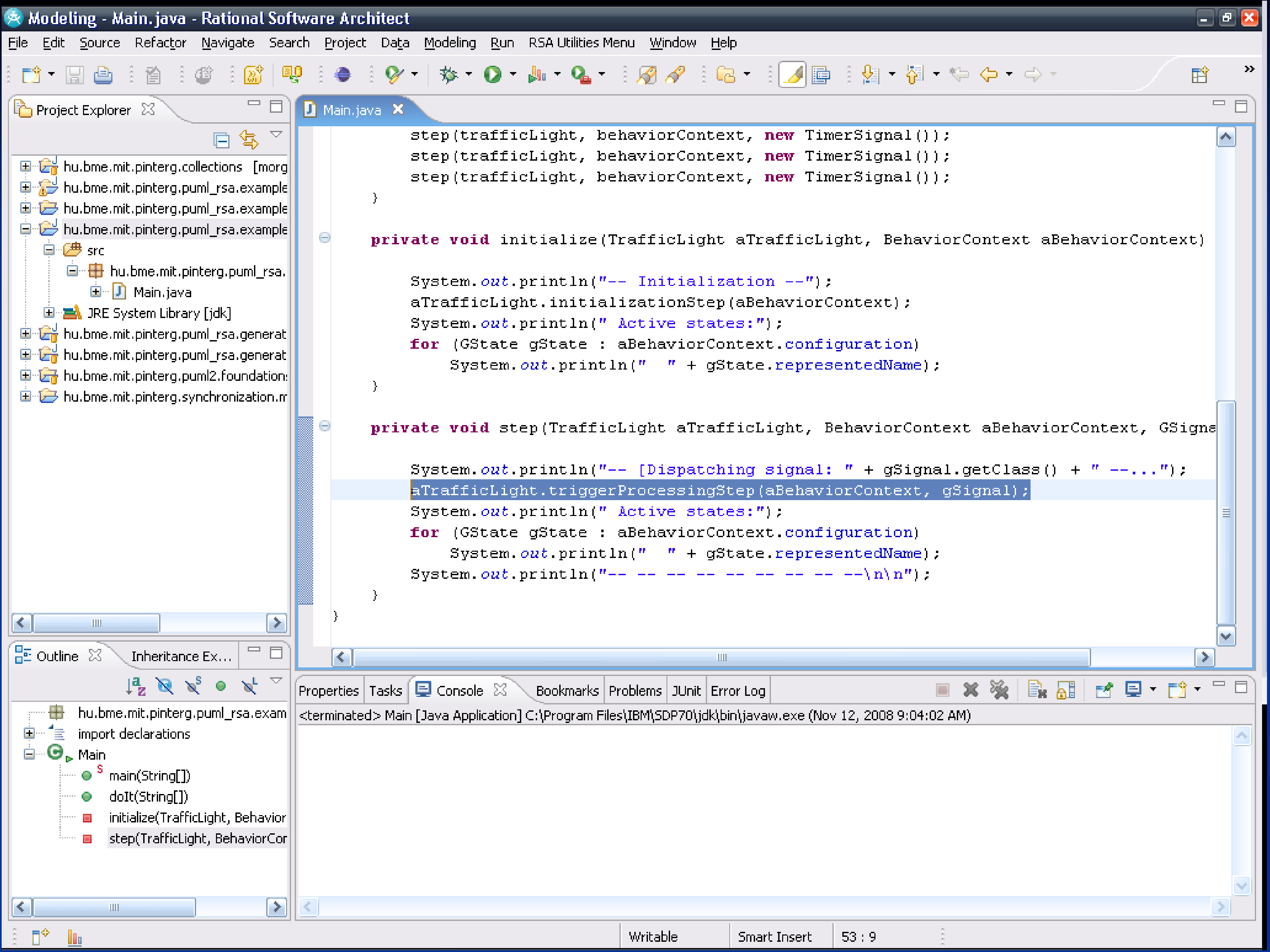
Modeling - Main.java - Rational Software Architect
File Edit Source Refactor Navigate Search Project Data Modeling Run RSA Utilities Menu Window Help
Project Explorer
Main.java
Outline
Inheritance Ex...
Properties Tasks Console Bookmarks Problems JUnit Error Log
Writable
Smart Insert
24 : 64

```
hu.bme.mit.pinterg.collections  [morg
hu.bme.mit.pinterg.puml_rsa.example
hu.bme.mit.pinterg.puml_rsa.example
hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.ui
    src
        hu.bme.mit.pinterg.puml_rsa.
            Main.java
    JRE System Library [jdk]
hu.bme.mit.pinterg.puml_rsa.generat
hu.bme.mit.pinterg.puml_rsa.generat
hu.bme.mit.pinterg.puml2.foundation:
hu.bme.mit.pinterg.synchronization.m
```

```java
package hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.ui;

import hu.bme.mit.pinterg.puml_rsa.generator.base_classes.BehaviorContext;

      class Main {

    /**
     * @param args
     */
    public static void main(String[] args) {

        (new Main()).doIt(args);
    }

    public void doIt(String[] args) {

        TrafficLight trafficLight = TrafficLight.getInstance();

        BehaviorContext behaviorContext = new BehaviorContext(null, trafficLight, null);

        initialize(trafficLight, behaviorContext);
        step(trafficLight, behaviorContext, new LampSwitchSignal());
        step(trafficLight, behaviorContext, new TimerSignal());
        step(trafficLight, behaviorContext, new TimerSignal());
        step(trafficLight, behaviorContext, new CarArrivedSignal());
```

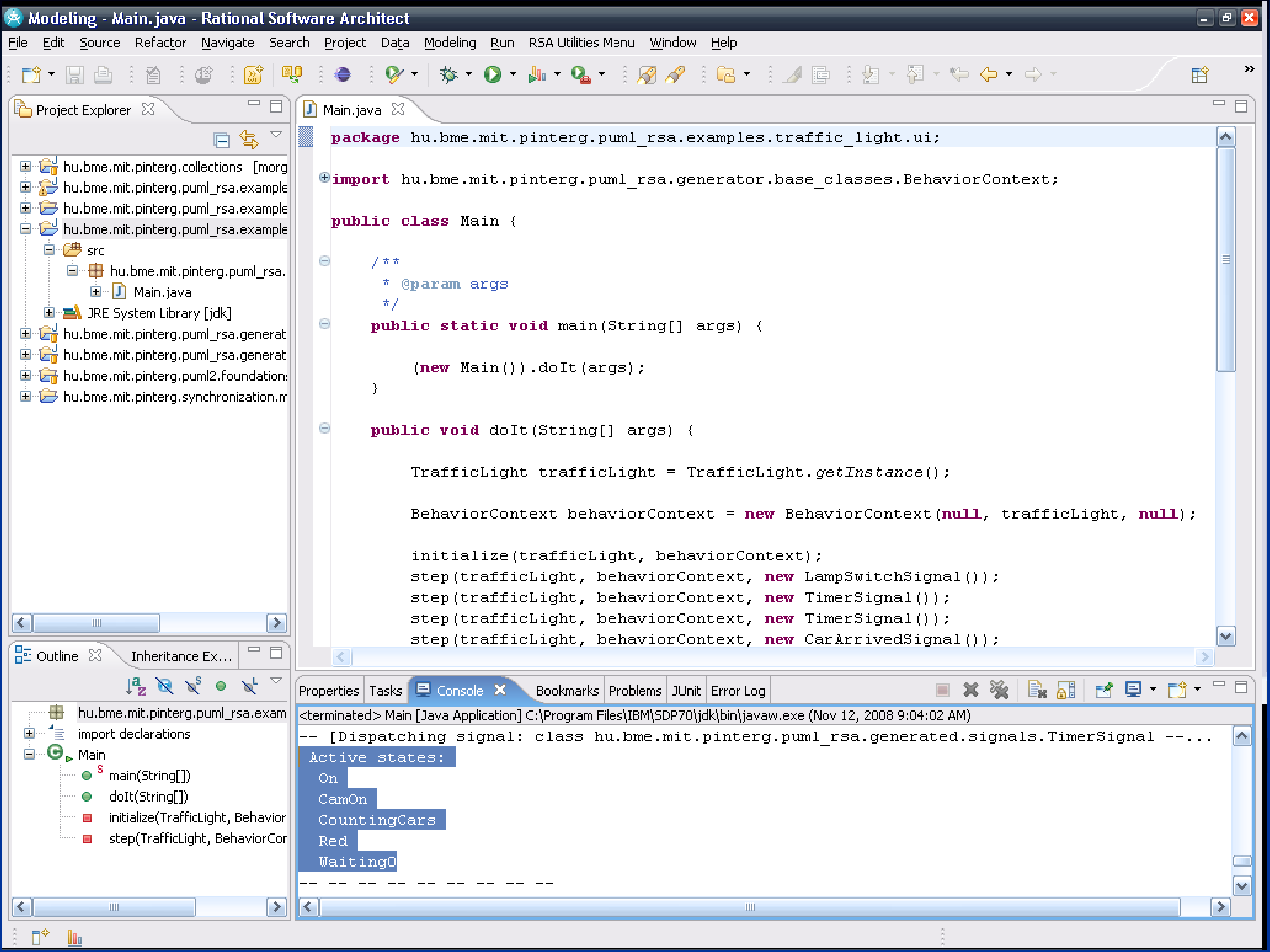
```
hu.bme.mit.pinterg.puml_rsa.exam
import declarations
Main
    main(String[])
    doIt(String[])
    initialize(TrafficLight, Behavior
    step(TrafficLight, BehaviorCor
```

```
<terminated> Main [Java Application] C:\Program Files\IBM\SDP70\jdk\bin\javaw.exe (Nov 12, 2008 9:04:02 AM)
```


Modeling - Main.java - Rational Software Architect
File Edit Source Refactor Navigate Search Project Data Modeling Run RSA Utilities Menu Window Help
Project Explorer
hu.bme.mit.pinterg.collections [morg
hu.bme.mit.pinterg.puml_rsa.example
hu.bme.mit.pinterg.puml_rsa.example
hu.bme.mit.pinterg.puml_rsa.example
src
hu.bme.mit.pinterg.puml_rsa.
Main.java
JRE System Library [jdk]
hu.bme.mit.pinterg.puml_rsa.generat
hu.bme.mit.pinterg.puml_rsa.generat
hu.bme.mit.pinterg.puml2.foundation:
hu.bme.mit.pinterg.synchronization.m
Main.java
Outline
Inheritance Ex...
hu.bme.mit.pinterg.puml_rsa.exam
import declarations
Main
main(String[])
doIt(String[])
initialize(TrafficLight, Behavior
step(TrafficLight, BehaviorCor
Properties Tasks Console Bookmarks Problems JUnit Error Log
Writable
Smart Insert
53 : 9

```
        step(trafficLight, behaviorContext, new TimerSignal());
        step(trafficLight, behaviorContext, new TimerSignal());
        step(trafficLight, behaviorContext, new TimerSignal());
    }

    private void initialize(TrafficLight aTrafficLight, BehaviorContext aBehaviorContext)

        System.out.println("-- Initialization --");
        aTrafficLight.initializationStep(aBehaviorContext);
        System.out.println(" Active states:");
        for (GState gState : aBehaviorContext.configuration)
            System.out.println("   " + gState.representedName);
    }

    private void step(TrafficLight aTrafficLight, BehaviorContext aBehaviorContext, GSigna

        System.out.println("-- [Dispatching signal: " + gSignal.getClass() + " --...");
        aTrafficLight.triggerProcessingStep(aBehaviorContext, gSignal);
        System.out.println(" Active states:");
        for (GState gState : aBehaviorContext.configuration)
            System.out.println("   " + gState.representedName);
        System.out.println("-- -- -- -- -- -- -- -- --\n\n");
    }
}
```

```
<terminated> Main [Java Application] C:\Program Files\IBM\SDP70\jdk\bin\javaw.exe (Nov 12, 2008 9:04:02 AM)
```

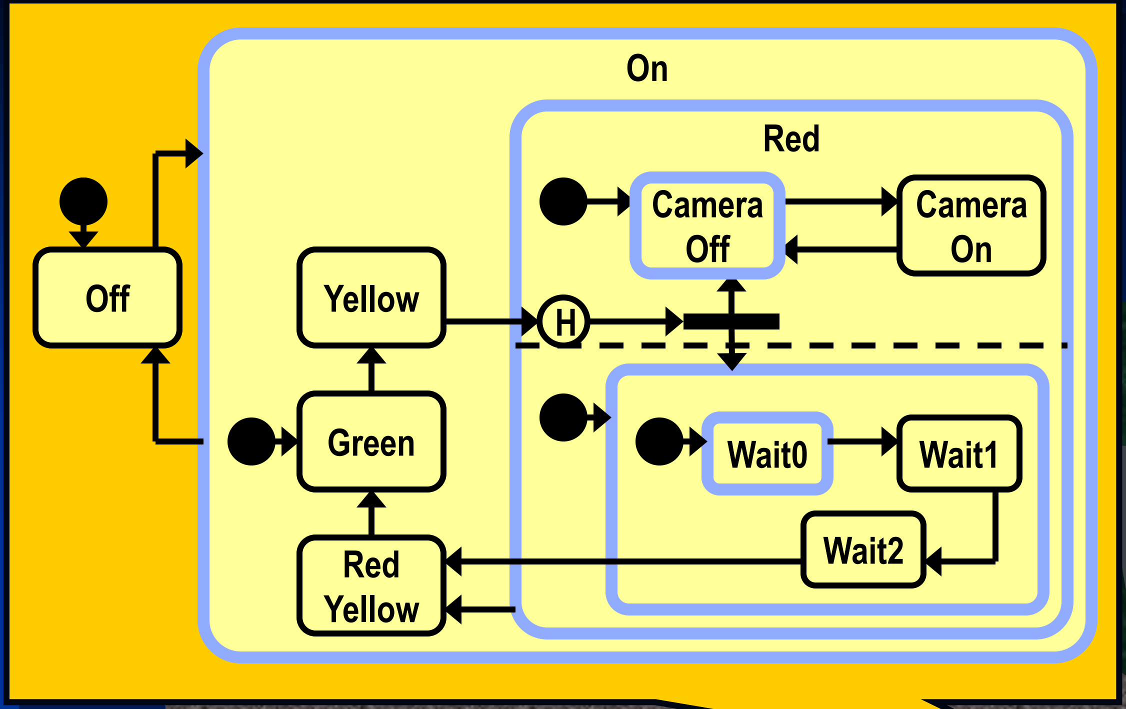

Modeling - Main.java - Rational Software Architect
File Edit Source Refactor Navigate Search Project Data Modeling Run RSA Utilities Menu Window Help
Project Explorer
Main.java
Outline
Inheritance Ex...
Properties Tasks Console Bookmarks Problems JUnit Error Log

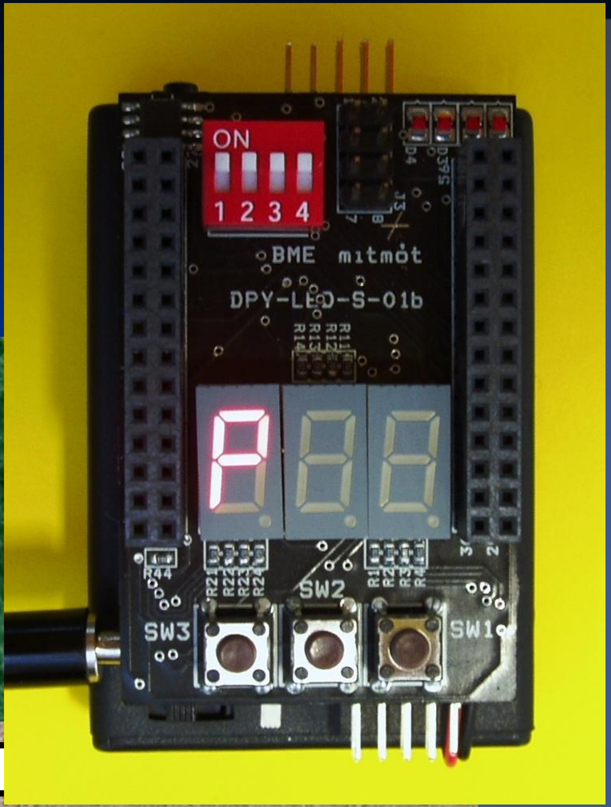
```
hu.bme.mit.pinterg.collections  [morg
hu.bme.mit.pinterg.puml_rsa.example
hu.bme.mit.pinterg.puml_rsa.example
hu.bme.mit.pinterg.puml_rsa.example
    src
        hu.bme.mit.pinterg.puml_rsa.
            Main.java
    JRE System Library [jdk]
hu.bme.mit.pinterg.puml_rsa.generat
hu.bme.mit.pinterg.puml_rsa.generat
hu.bme.mit.pinterg.puml2.foundation:
hu.bme.mit.pinterg.synchronization.m
```

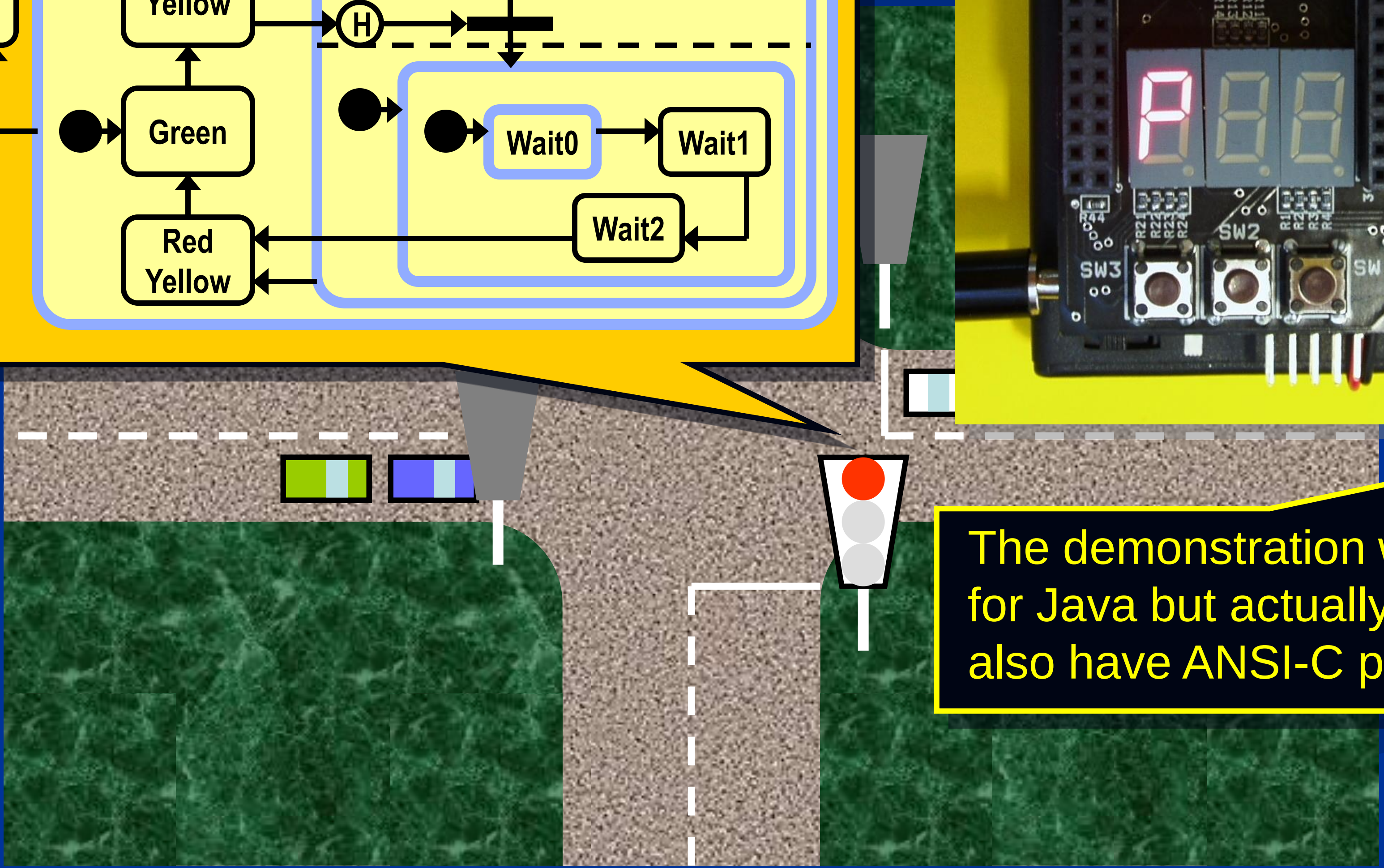
```java
package hu.bme.mit.pinterg.puml_rsa.examples.traffic_light.ui;

import hu.bme.mit.pinterg.puml_rsa.generator.base_classes.BehaviorContext;

public class Main {

    /**
     * @param args
     */
    public static void main(String[] args) {

        (new Main()).doIt(args);
    }

    public void doIt(String[] args) {

        TrafficLight trafficLight = TrafficLight.getInstance();

        BehaviorContext behaviorContext = new BehaviorContext(null, trafficLight, null);

        initialize(trafficLight, behaviorContext);
        step(trafficLight, behaviorContext, new LampSwitchSignal());
        step(trafficLight, behaviorContext, new TimerSignal());
        step(trafficLight, behaviorContext, new TimerSignal());
        step(trafficLight, behaviorContext, new CarArrivedSignal());
```

```
hu.bme.mit.pinterg.puml_rsa.exam
import declarations
Main
    main(String[])
    doIt(String[])
    initialize(TrafficLight, Behavior
    step(TrafficLight, BehaviorCor
```

```
<terminated> Main [Java Application] C:\Program Files\IBM\SDP70\jdk\bin\javaw.exe (Nov 12, 2008 9:04:02 AM)
-- [Dispatching signal: class hu.bme.mit.pinterg.puml_rsa.generated.signals.TimerSignal --...
 Active states:
  On
  CamOn
  CountingCars
  Red
  Waiting0
-- -- -- -- -- -- -- -- --
```







The demonstration was for Java but actually we also have ANSI-C ports.

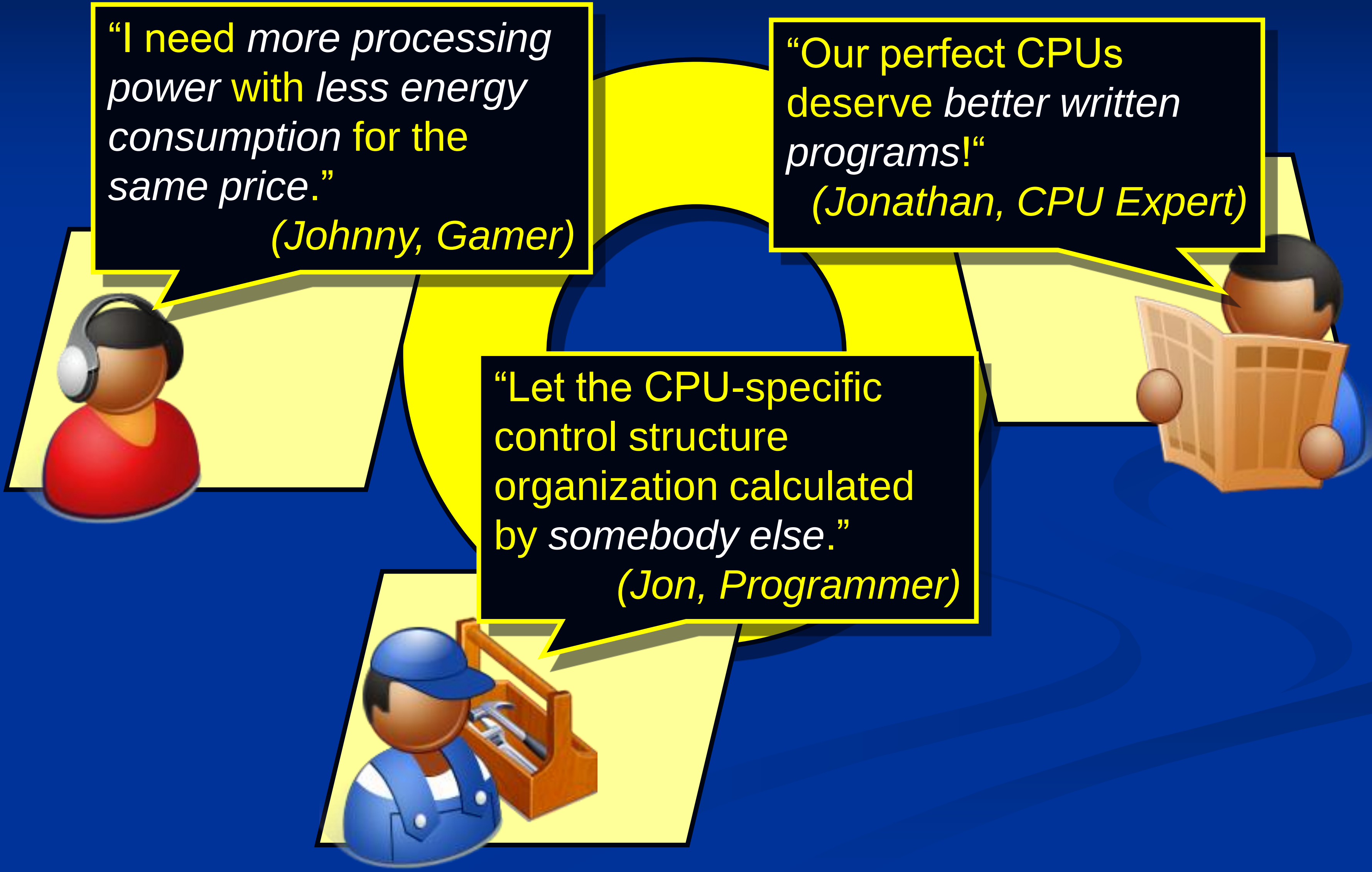
Presentation Structure

Wide Context and Future Research Goals

Achievements Until Now

Demonstration

Summary



“I need *more processing power* with *less energy consumption* for the *same price*.”
(Johnny, Gamer)

“Our perfect CPUs deserve *better written programs*!“
(Jonathan, CPU Expert)

“Let the CPU-specific control structure organization calculated by *somebody else*.”
(Jon, Programmer)