Formalizing the ARTS MPSoC Model in UPPAAL

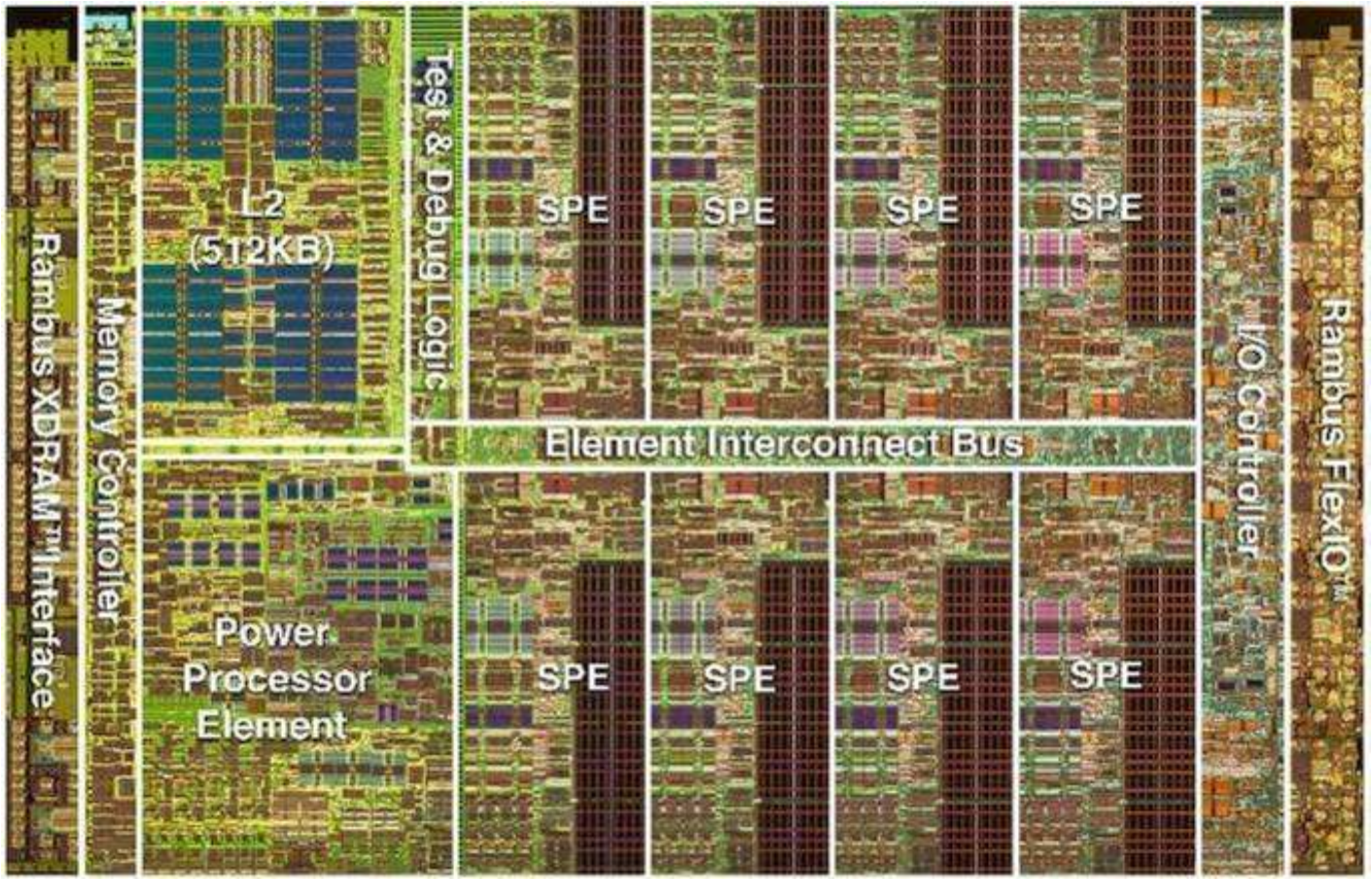
Jan Madsen

Embedded Systems Engineering Group

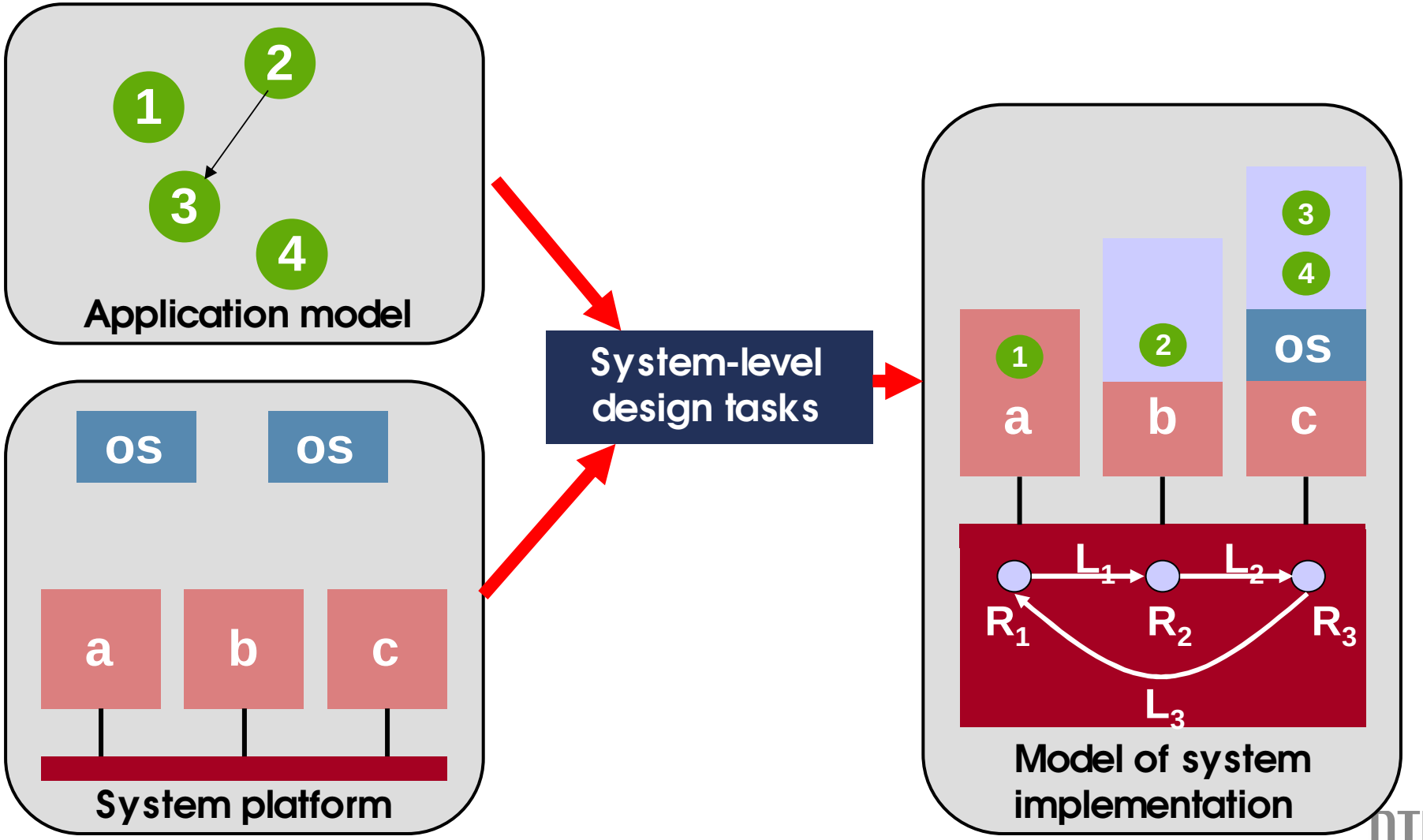
Informatics and Mathematical Modeling
Technical University of Denmark

Motivation

CELL processor

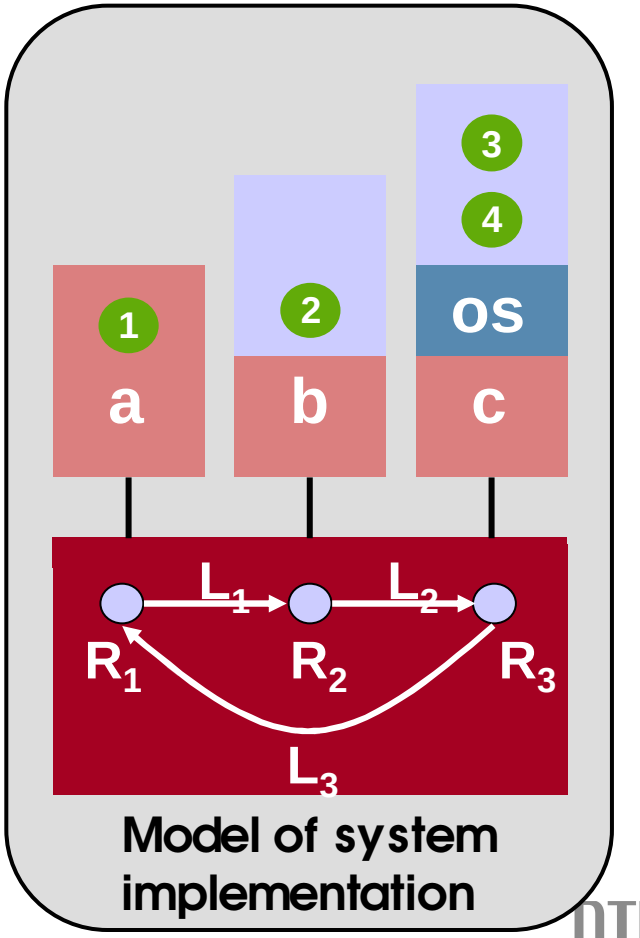


Motivation


1
2
3
4
Application model
os
os
a
b
c
System platform
System-level design tasks
1
a
2
b
3
4
os
c
L_1
L_2
L_3
R_1
R_2
R_3
Model of system implementation

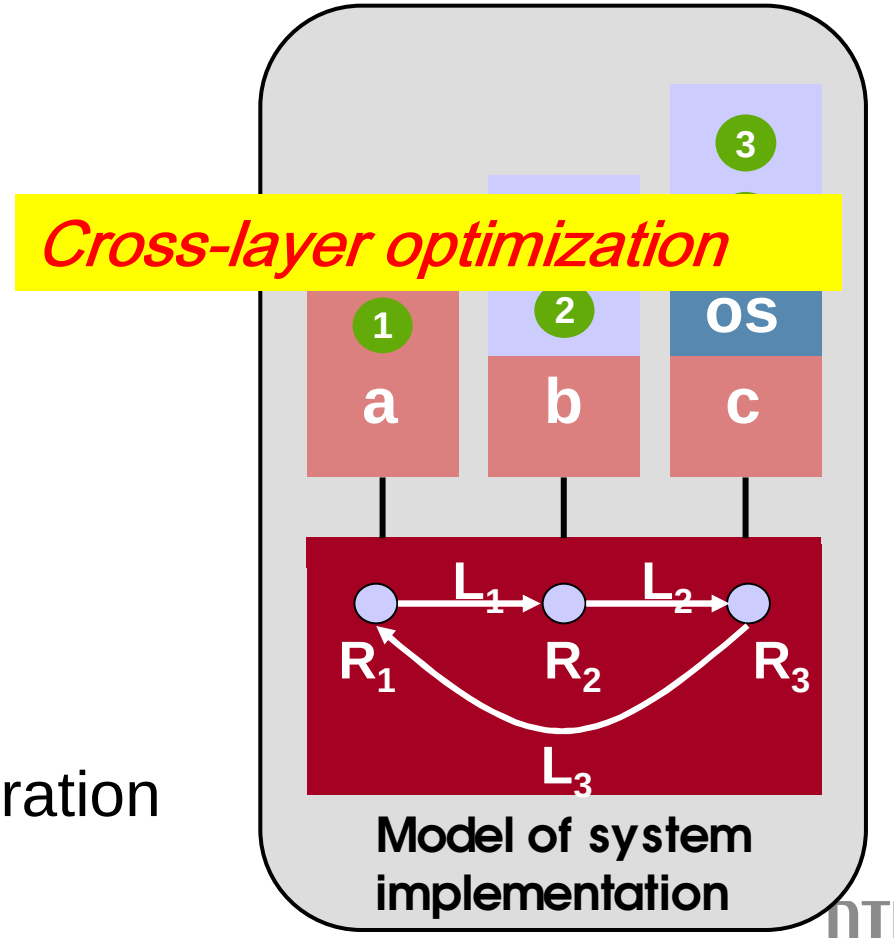
Outline

- The ARTS framework
- Formalizing ARTS
- MOVES
- Example

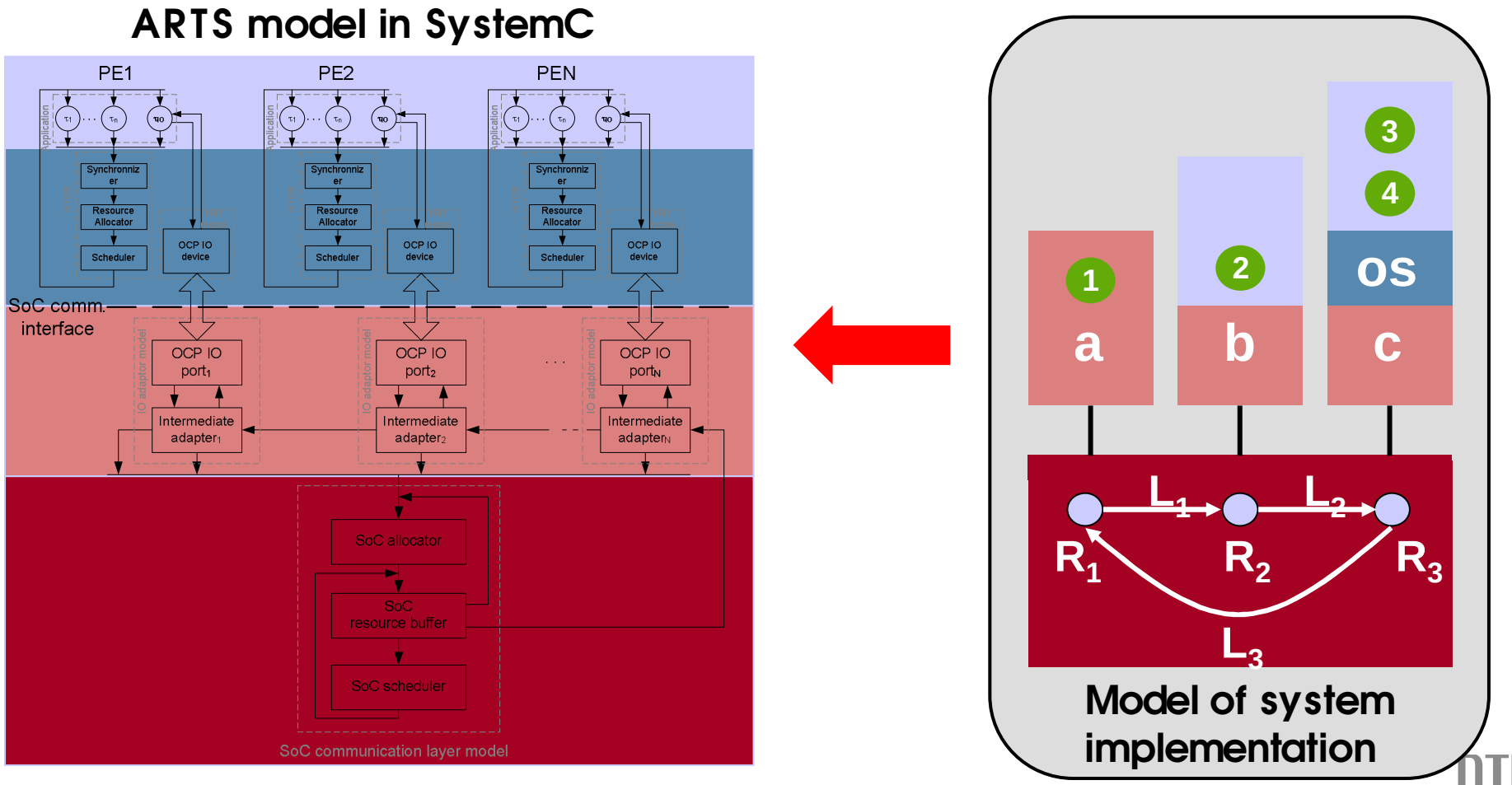


ARTS objectives

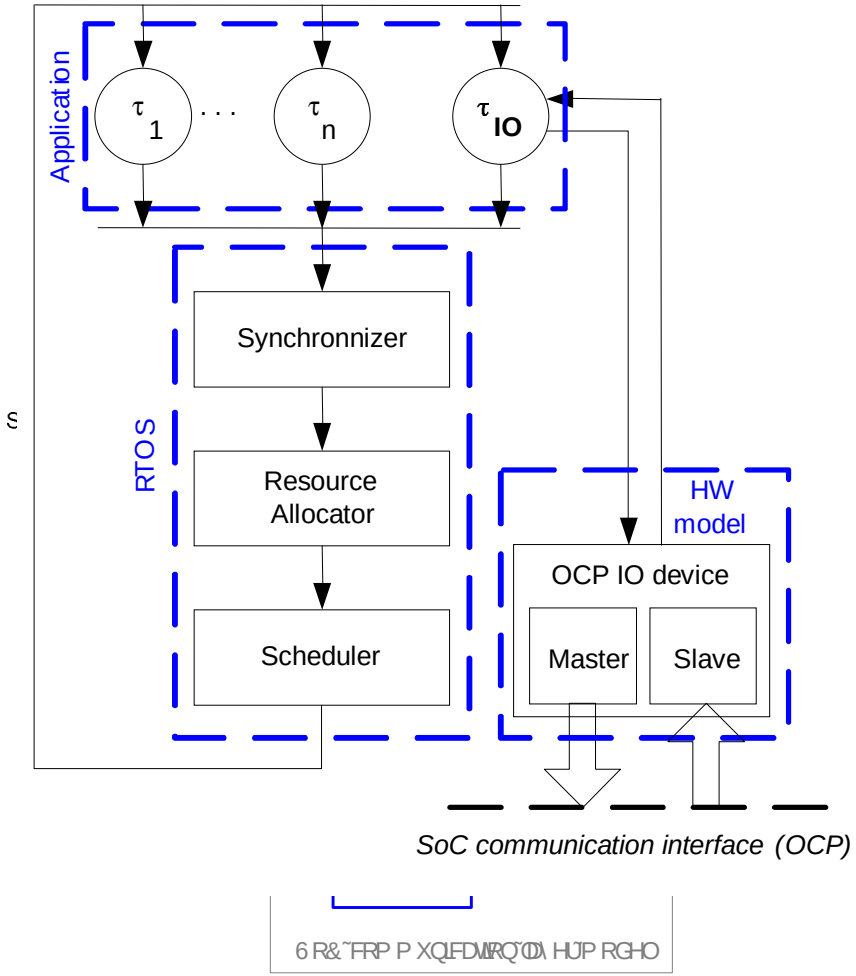
- System-level modeling framework
- Bridging,
  - Application
  - RTOS
  - Execution platform
    - Processing elements
    - NoC
- Supporting
  - System-level analysis
  - Early design space exploration



ARTS framework

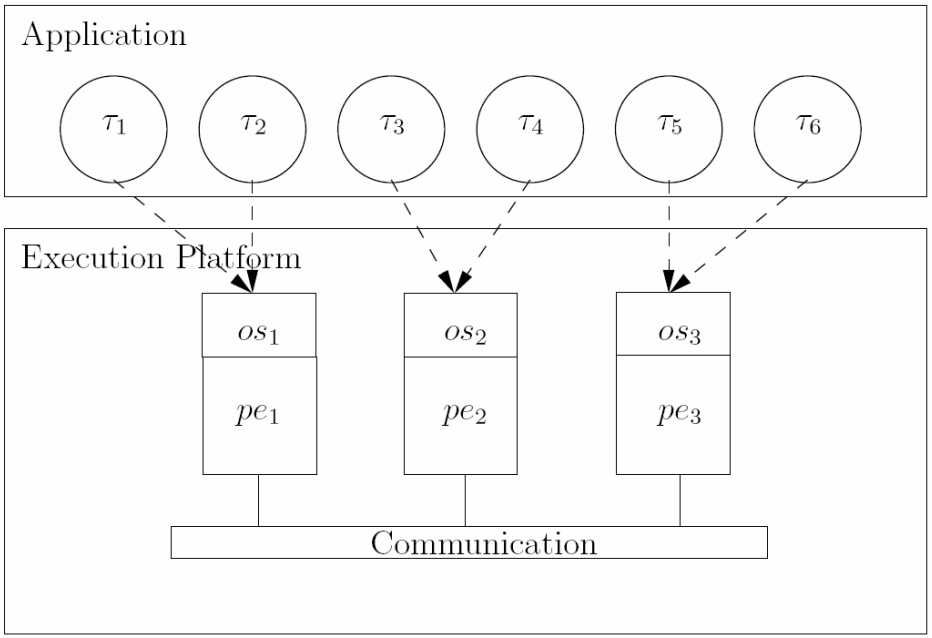


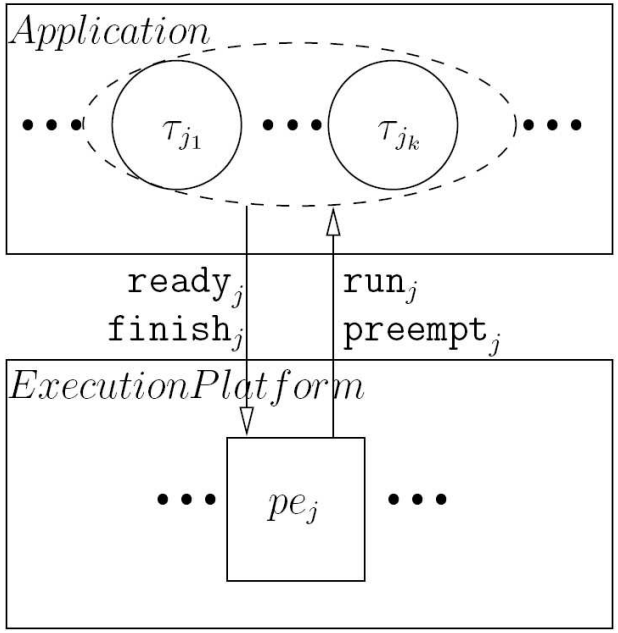
ARTS Framework



- **ARTS** Simulation framework based on SystemC
- **ARTS** PE module:
  - Application
  - OS
  - IO ports (OCP 2.0 interface)
  - IO device drivers
- **ARTS** Communication module:
  - Network topology and protocol
  - Network adapters
  - IO ports (OCP 2.0 interface)
- Applications of **ARTS**:
  - MPSoC (NoC exploration)
  - Wireless sensor networks
  - Automotive systems (TT vs. ET)
  - Dynamic reconfiguration

Formalizing ARTS



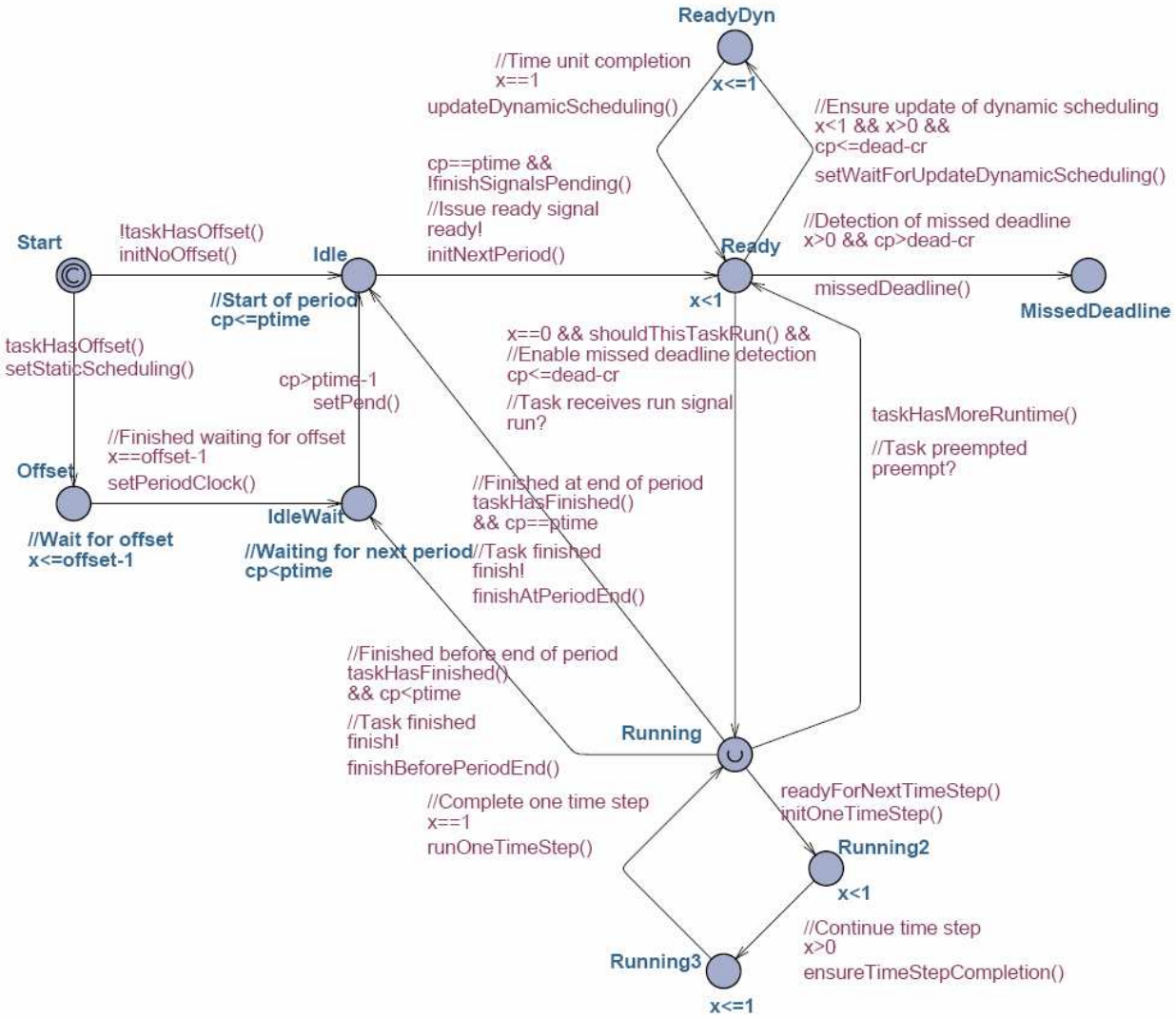


$$System = Application \parallel ExecutionPlatform$$

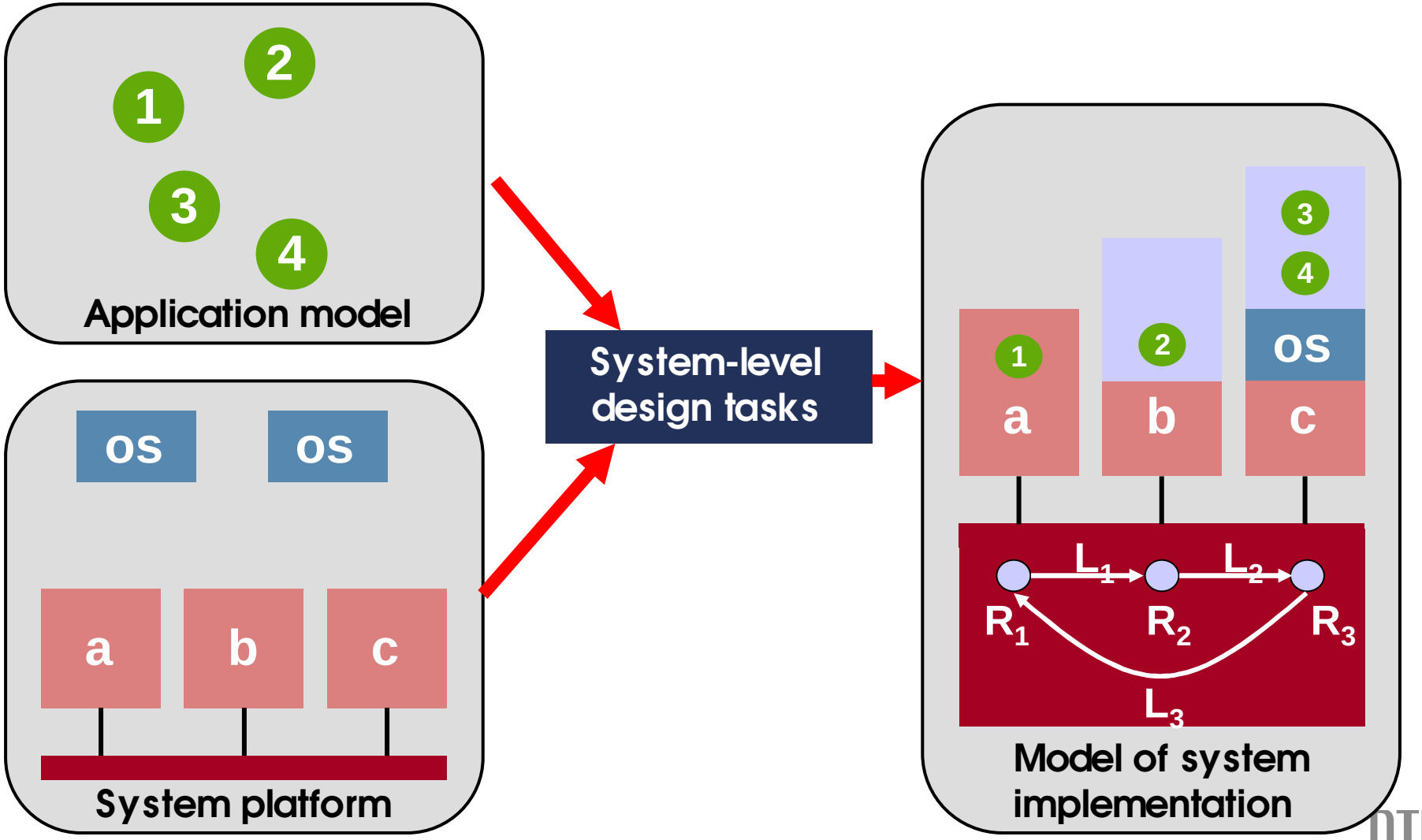
$$Application = \|_{i=1}^{n} \tau_i$$

$$ExecutionPlatform = \|_{j=1}^{m} pe_j$$

Timed Automata for a task



MOVES: Hiding UPPAAL!

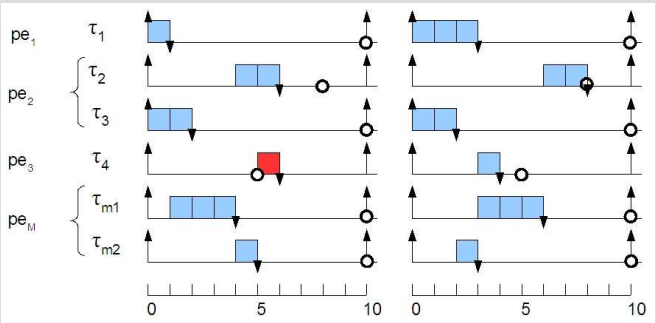


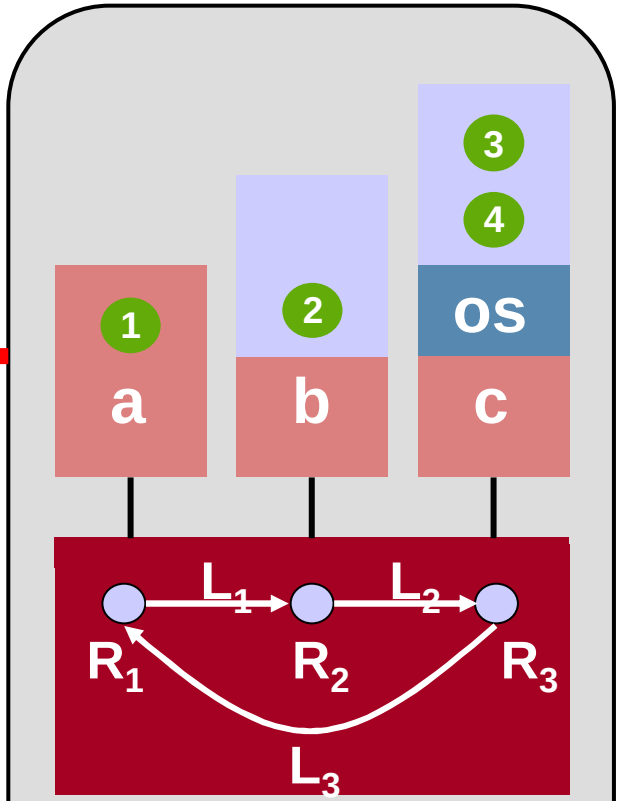
MOVES

```
E<>missedDeadline

E<>totalCostUsed(Memory)>=23

E<>totalCostUsed(Energy)>=15
```

Required specification

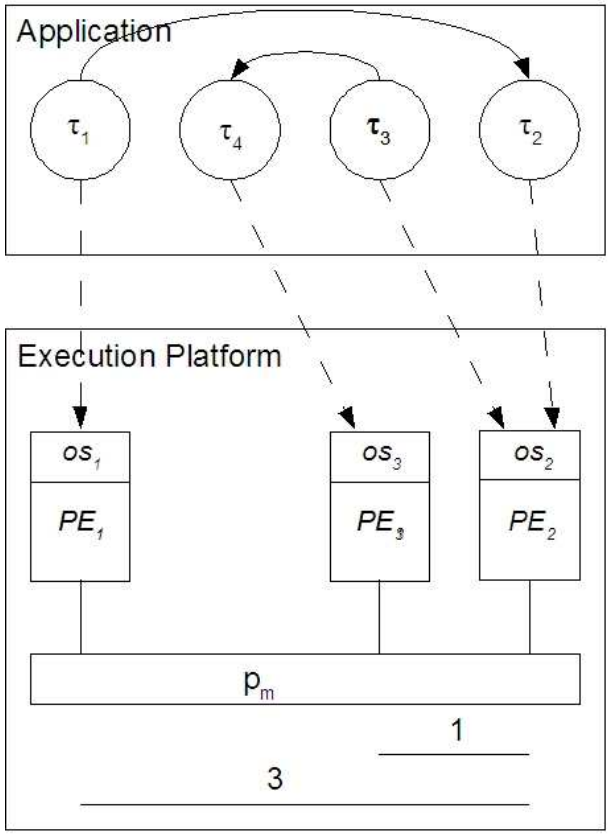
Model checking





Model of system implementation

Example: MPSoC specification



Task	π	δ	o
τ_1	10	10	0
τ_2	10	8	0
τ_3	10	10	0
τ_4	10	5	0

PE_i f = 1 MHz, os = RM

Task	*bcet*	*wcet*	*sm*	*dm*	*pw*
τ_1	1	3	1	3	5
τ_2	2	2	1	7	5
τ_3	2	2	1	6	10
τ_4	1	1	2	9	10

π : period [s]
δ : deadline [s]
ω : offset [s]
bcet : best case execution time [cycles]
wcet : worst case execution time [cycles]
sm : static memory [Byte]
dm : dynamic memory [Byte]
pw : power [*m*W]
f : frequency [Hz]
os: operating system[{*FP,RM,DM,EDF*}]

Specifying the application

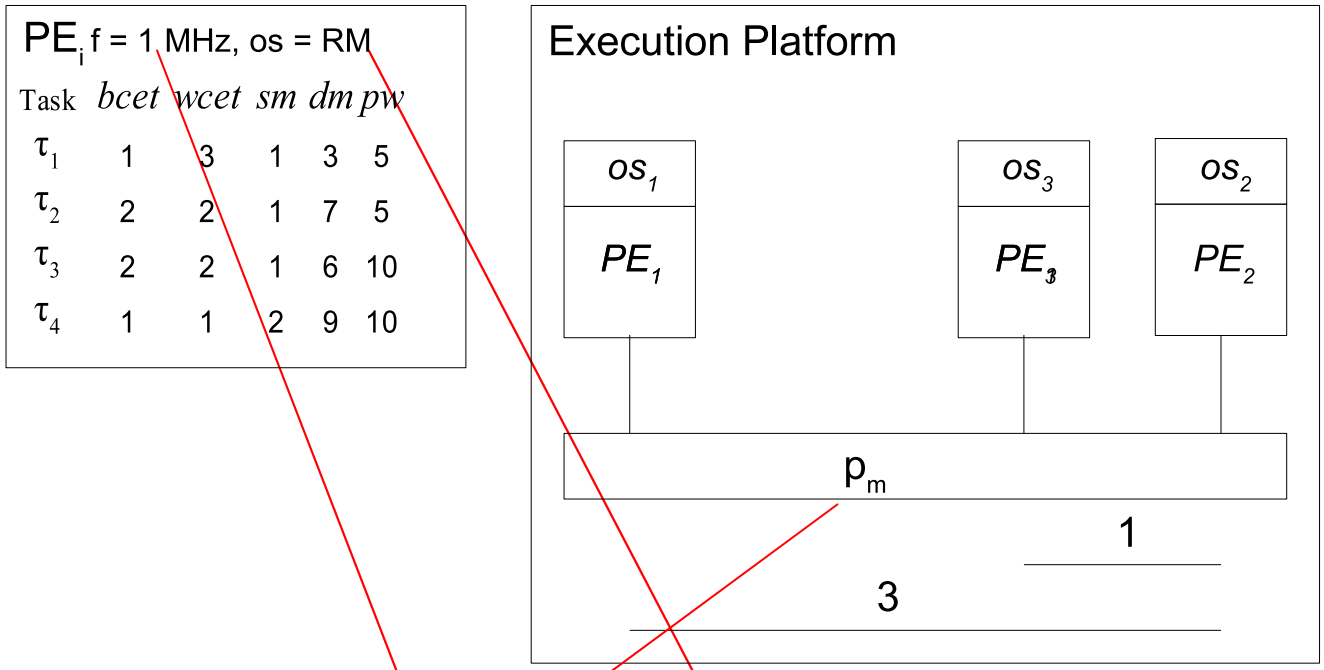
```
Task t1  = new Task(1, 3, 10, 10, 0, 1);
Task t2  = new Task(2, 2, 8, 10, 0, 2);
Task t3  = new Task(2, 2, 10, 10, 0, 3);
Task t4  = new Task(1, 1, 5, 10, 0, 4);
```

Task	π	δ	o
τ_1	10	10	0
τ_2	10	8	0
τ_3	10	10	0
τ_4	10	5	0

PE_i f = 1 MHz, os = RM

Task	*bcet*	*wcet*	*sm*	*dm*	*pw*
τ_1	1	3	1	3	5
τ_2	2	2	1	7	5
τ_3	2	2	1	6	10
τ_4	1	1	2	9	10

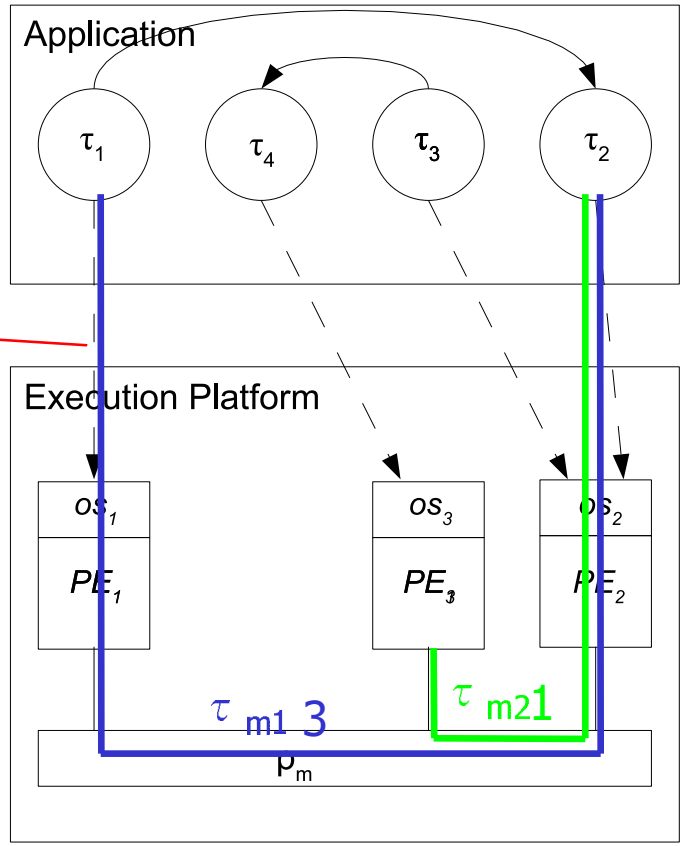
Execution platform




```
Processor p1 = new Processor(1, Processor.RM);
Processor p2 = new Processor(1, Processor.RM);
Processor p3 = new Processor(1, Processor.RM);
Processor pm = new Processor(1, Processor.RM);
Resource r1 = new Resource();
```

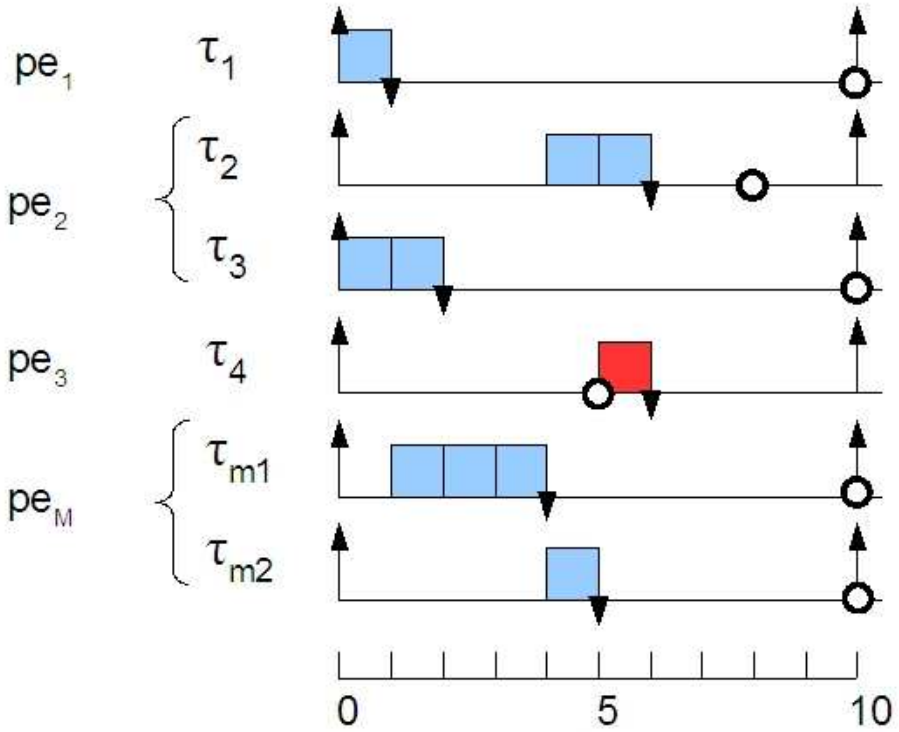
Mapping application onto platform

```
Task[][] tasks =
        {{t1},{t2,t3},{t4},{tm1,tm2}};

Task tm1 = new Task(3, 3, 10, 5);
Task tm2 = new Task(1, 1, 10, 6);

apps.useResource(tm1,r1);
apps.useResource(tm2,r1);

apps.addDep(t1,tm1);
apps.addDep(tm1,t2);
apps.addDep(t3,tm2);
apps.addDep(tm2,t4);
```

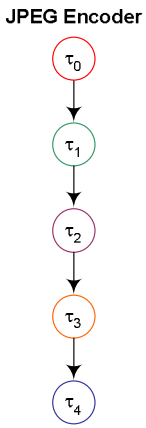


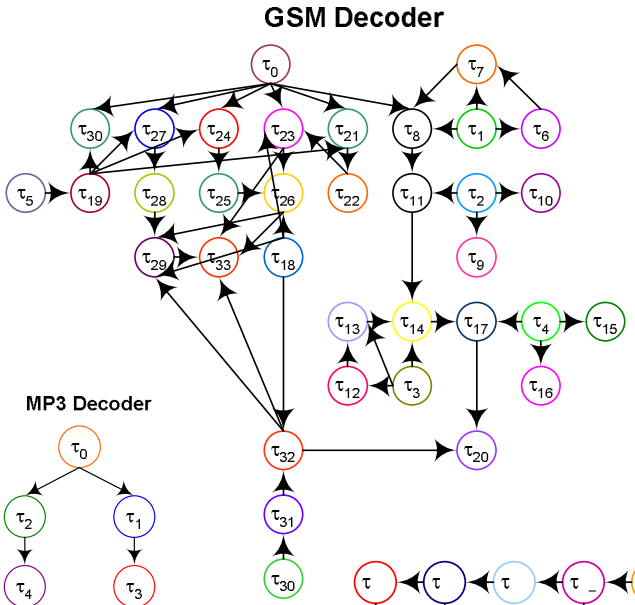
Traces

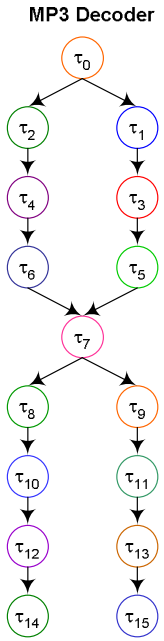
$e(\tau_1)$= [1:3]

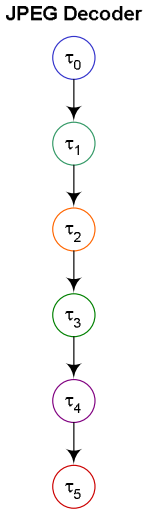


Handling realistic applications?

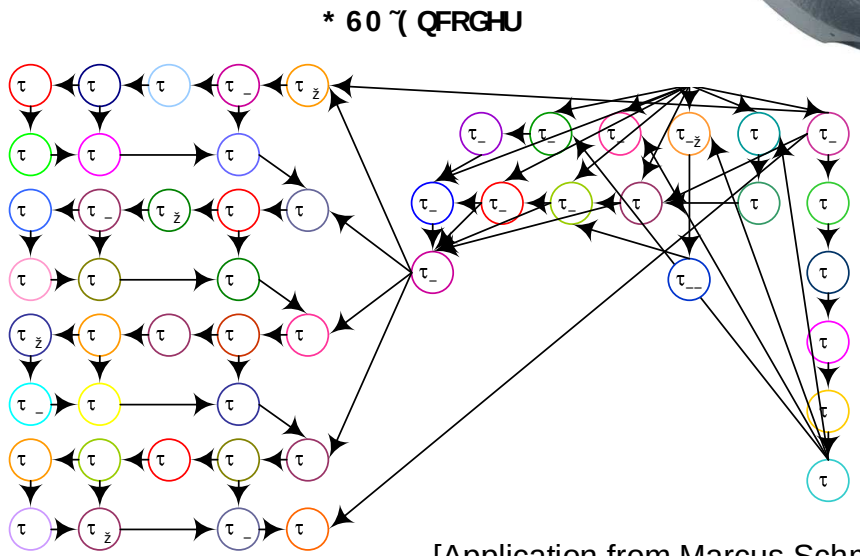






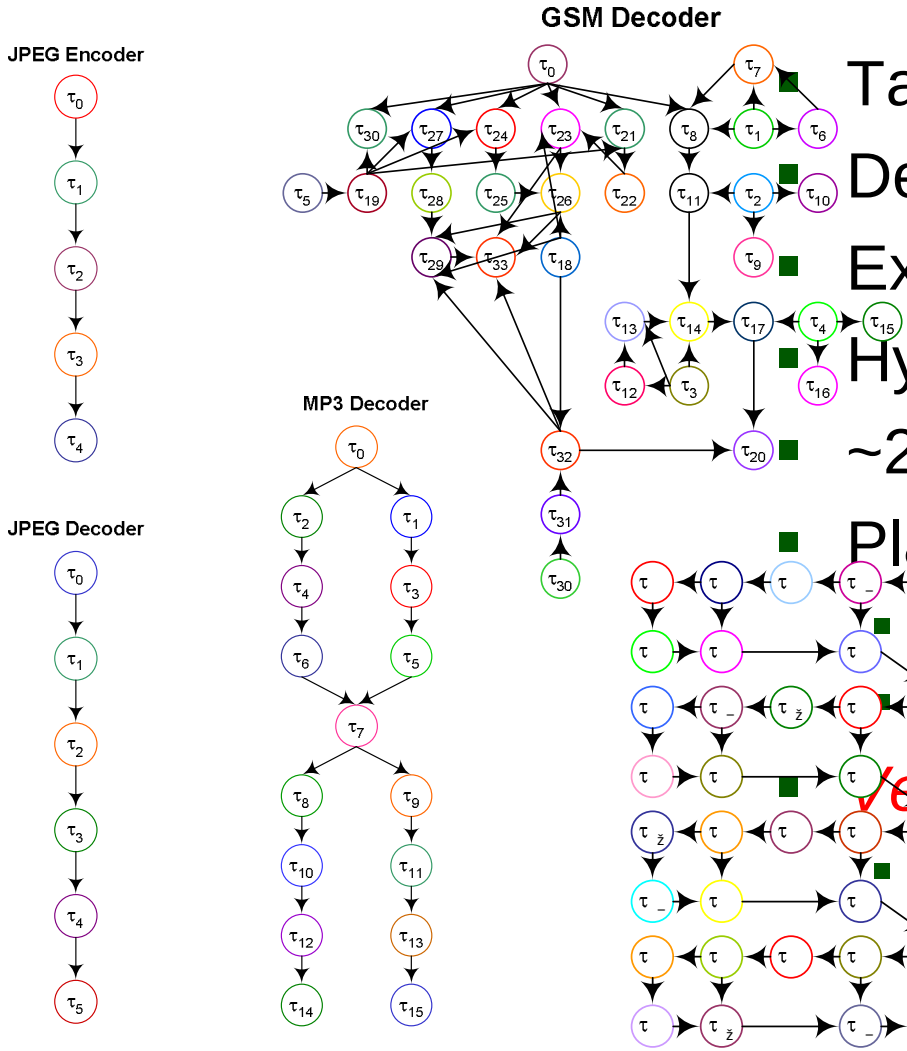


Smart phone:



[Application from Marcus Schmitz, TU Linkoping]

Smart phone



- Tasks: 114
- Deadlines: [0.02: 0.5] sec
- Execution: [52 : 266.687] cycles
- Hyperperiod: 12.500.000 cycles
- ~2.500 task executions
- Platform:
  - 6 processors, 2 MHz
  - 1 bus
- *Verified in 2.5 hours!*
- Using a granularity of 400 cycles

Acknowledgements

MOVES

- Michael R. Hansen
- Aske Brekling
- Jens Ellebæk
- Kristian S. Knudsen

ARTS

- Shankar Mahadevan
- Kehuai Wu
- Kashif Virk
- Michael Storgaard
- Mercury Gonzalez