MÄLARDALEN UNIVERSITY
SWEDEN

Towards WCET Analysis of Multi-core Architectures Using UPPAAL

Andreas Gustavsson[1], Andreas Ermedahl, Björn Lisper, and Paul Pettersson

[1]andreas.sg.gustavsson@mdh.se

Outline

- Motivation and Contribution
- Timed Automata and UPPAAL
- WCET analysis using UPPAAL
- Example WCET estimation
- Future work

Motivation

- Parallel software

```
thread1                thread2
{                      {

    ...                    ...

    wait for               tell thread1
      thread2                I'm here

    ...                    ...

}                      }
```

Motivation

- Parallel software
- Parallel hardware

```
thread1                thread2
{                      {
    ...                    ...
    wait for               tell thread1
      thread2                I'm here
    ...                    ...
}                      }
```

Motivation

- Parallel software
- Parallel hardware

```
thread1                 thread2
{                       {
    ...                     ...
    wait for                tell thread1
      thread2                 I'm here
    ...                     ...
}                       }
```

What is the WCET???

Motivation

- Parallel software
- Parallel hardware

```
thread1                 thread2
{                       {
    ...                     ...
    wait for                tell thread1
      thread2                 I'm here
    ...                     ...
}                       }
```

What is the WCET???

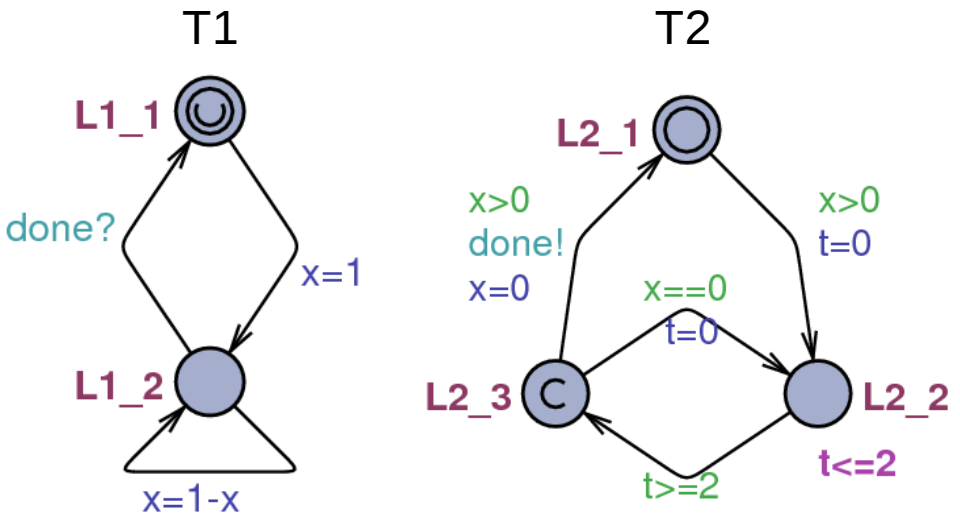
- Need new methods for WCET analysis

Contribution

- Parallel software
  - Threaded program
  - Mutual exclusion/Synchronization (spinlock)
- Parallel hardware
  - Shared Memory Multicore
- WCET analysis method: Model-checking of Timed Automata

Timed Automata in UPPAAL

- Parallel system
  - Synchronization
  - Variables
- Time is included in the formalism
- Analysis Tool
  - Query language

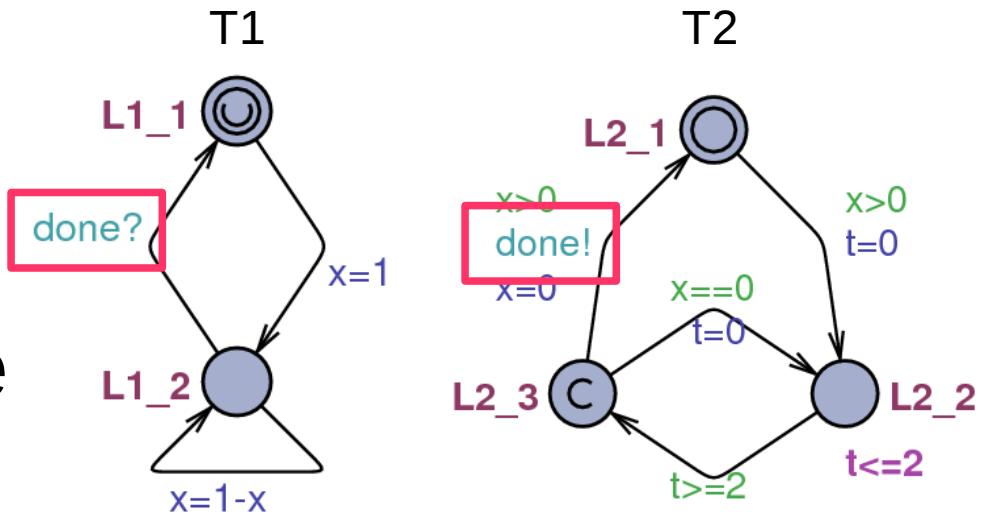


A[] x==1 imply T1.L1_2

E<> deadlock

Timed Automata in UPPAAL

- Parallel system
  - Synchronization
  - Variables
- Time is included in the formalism
- Analysis Tool
  - Query language

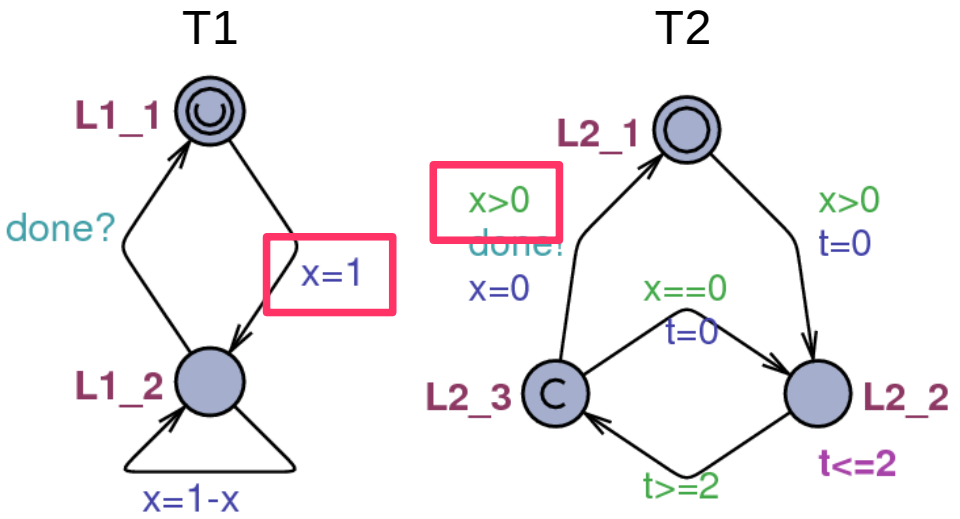


A[] x==1 imply T1.L1_2

E<> deadlock

Timed Automata in UPPAAL

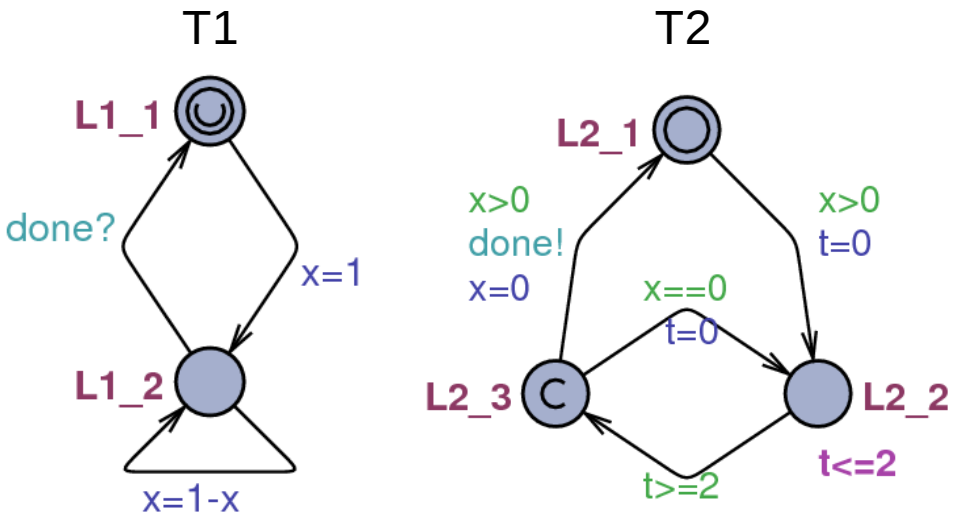
- Parallel system
  - Synchronization
  - Variables
- Time is included in the formalism
- Analysis Tool
  - Query language




```
A[] x==1 imply T1.L1_2
E<> deadlock
```

Timed Automata in UPPAAL

- Parallel system
  - Synchronization
  - Variables
- Time is included in the formalism
- Analysis Tool
  - Query language

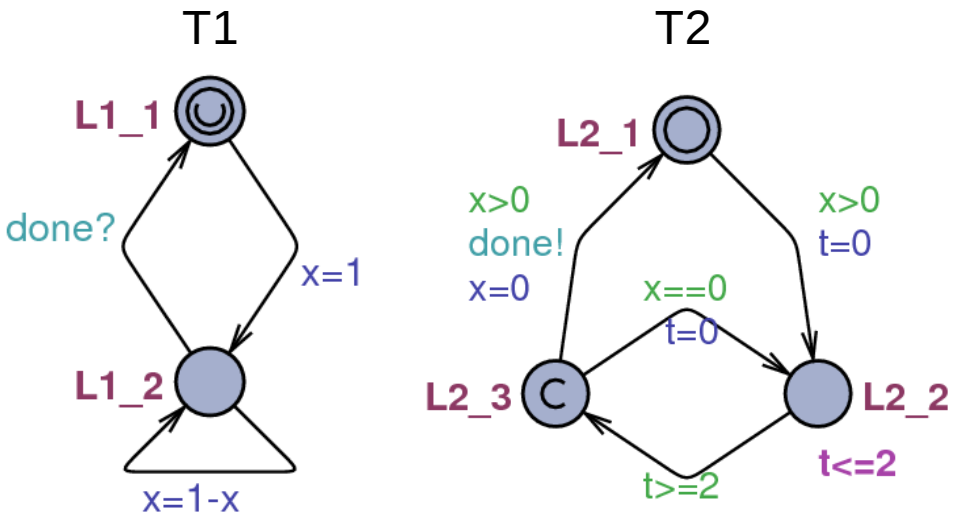


A[] x==1 imply T1.L1_2

E<> deadlock

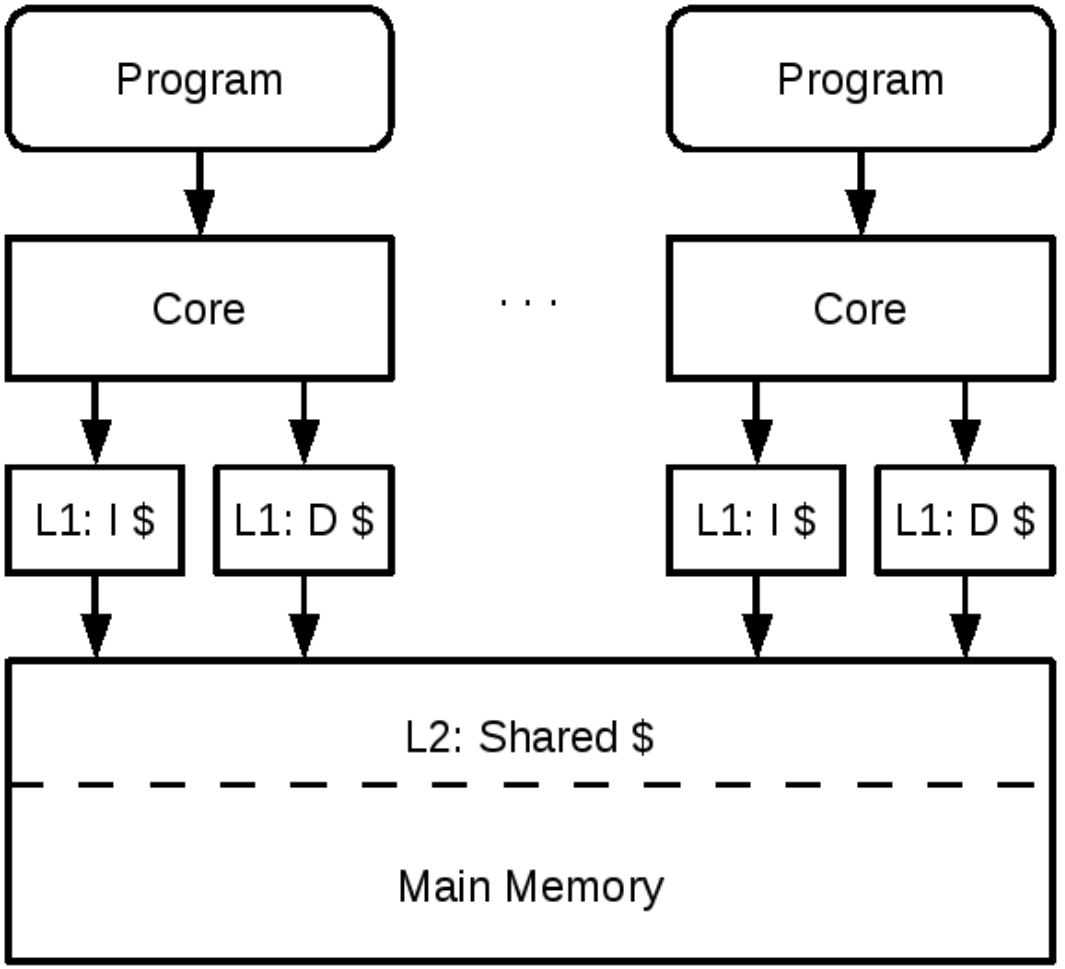
Timed Automata in UPPAAL

- Parallel system
  - Synchronization
  - Variables
- Time is included in the formalism
- Analysis Tool
  - Query language



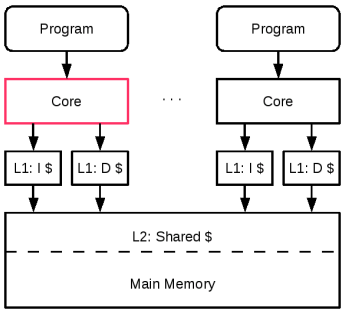

```
A[] x==1 imply T1.L1_2
E<> deadlock
```

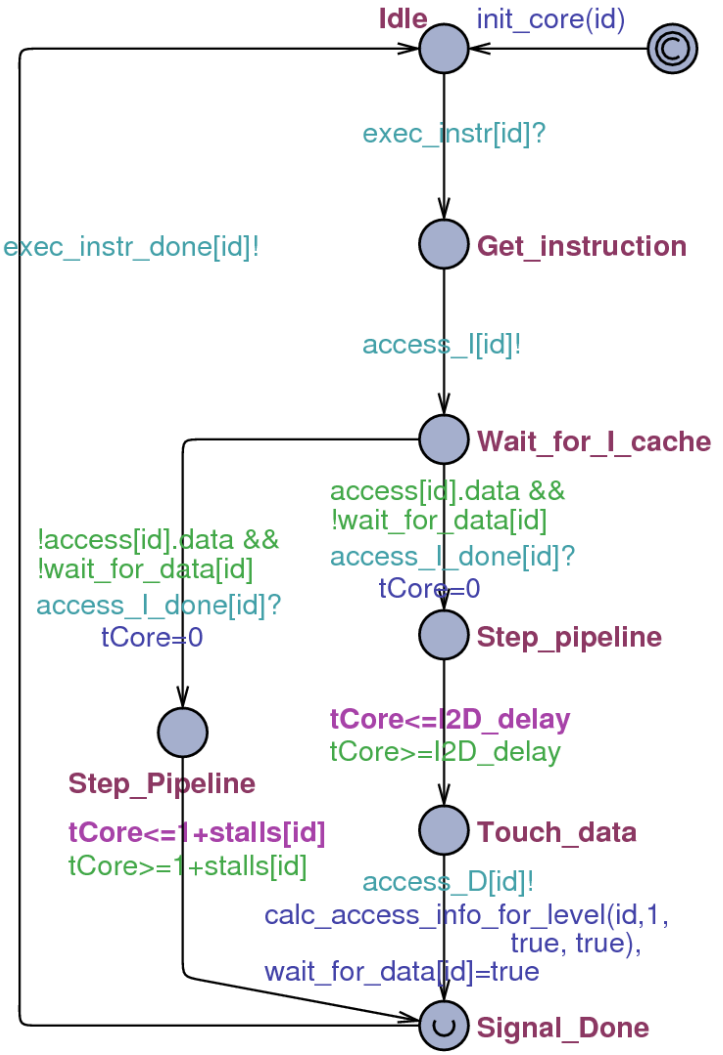
The Modeled System



The Core

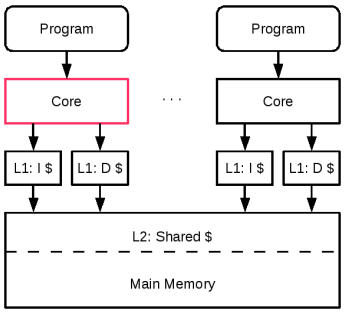
- In-order execution
- The core can overlap instruction execution and data accessing

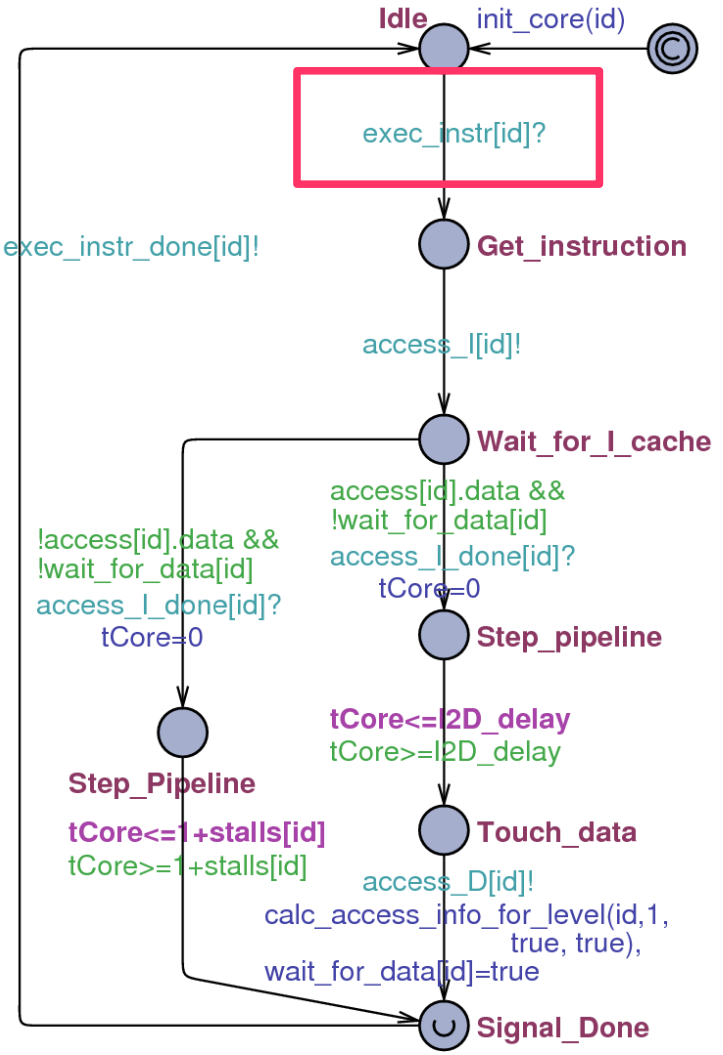




The Core

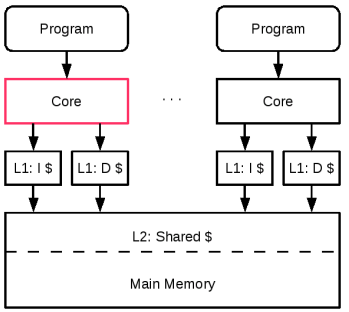
- In-order execution
- The core can overlap instruction execution and data accessing

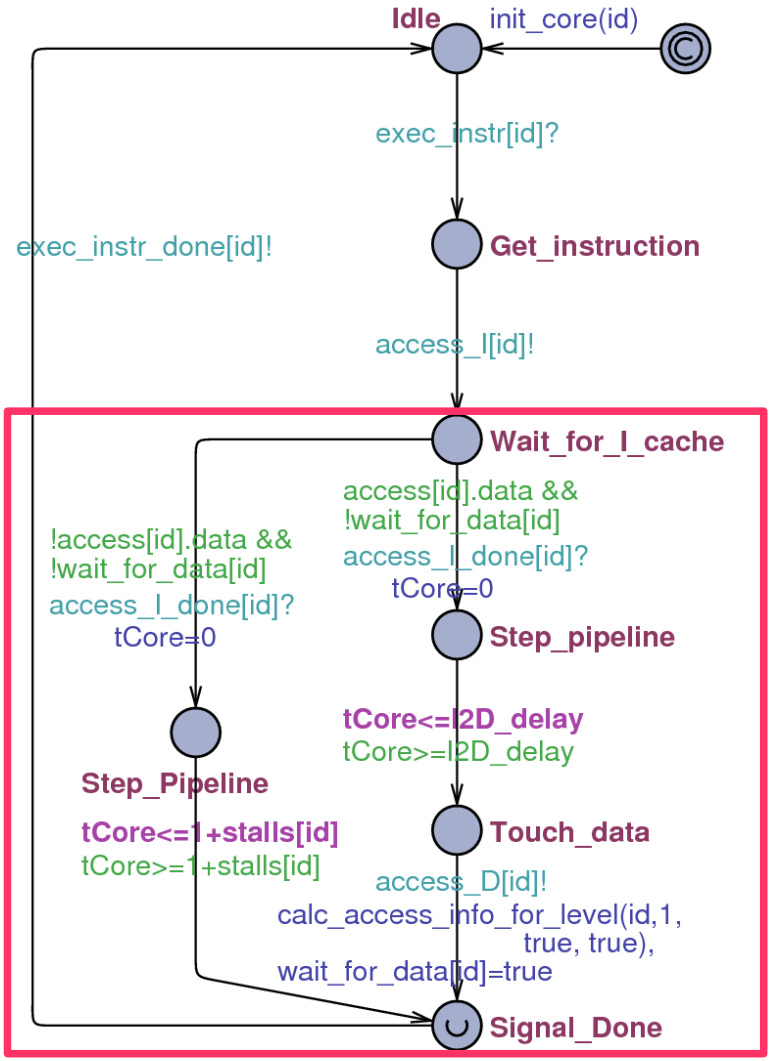




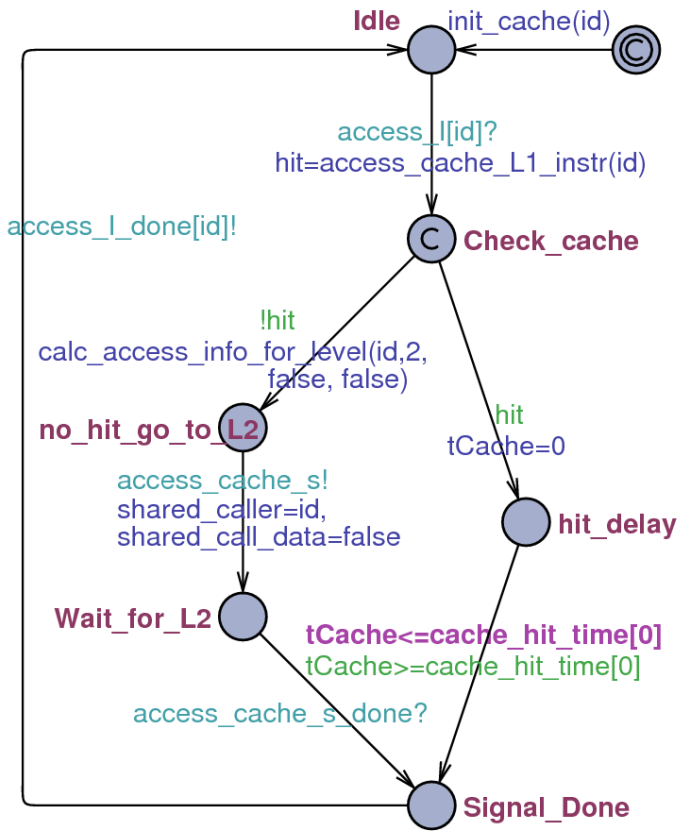
The Core

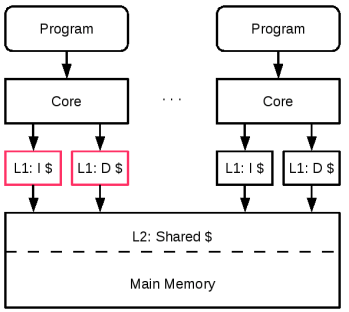
- In-order execution
- The core can overlap instruction execution and data accessing

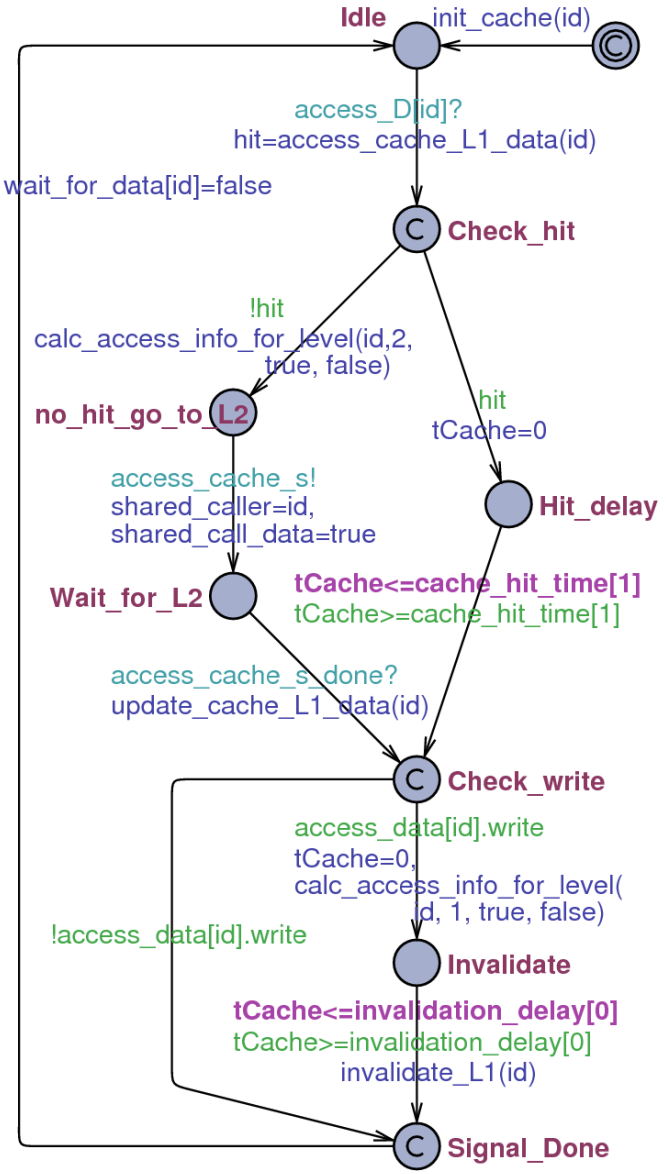




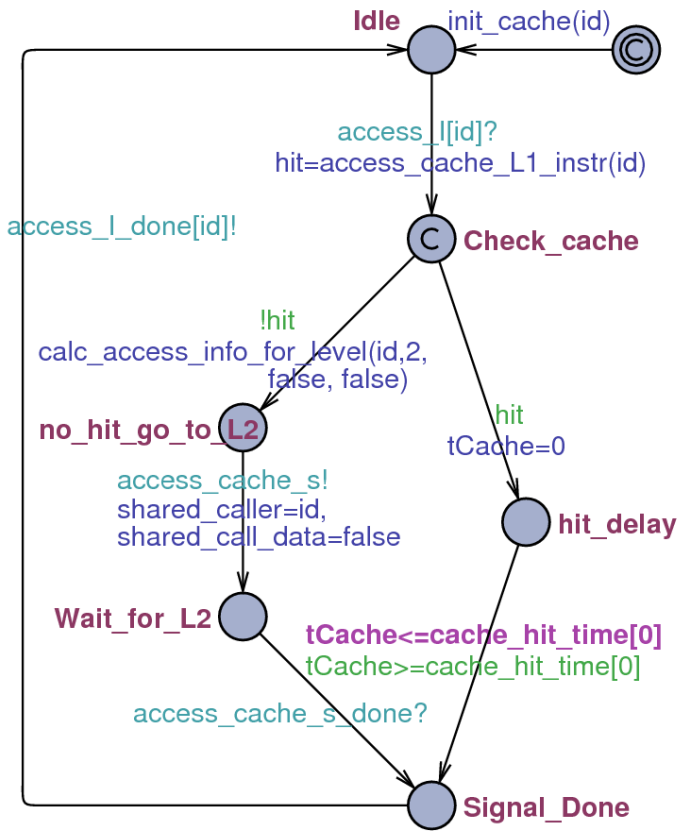
The L1 Caches

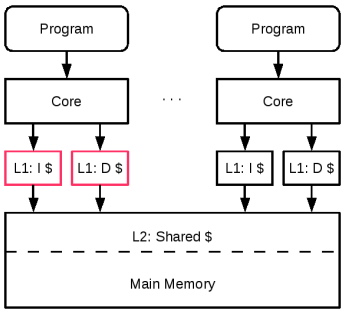


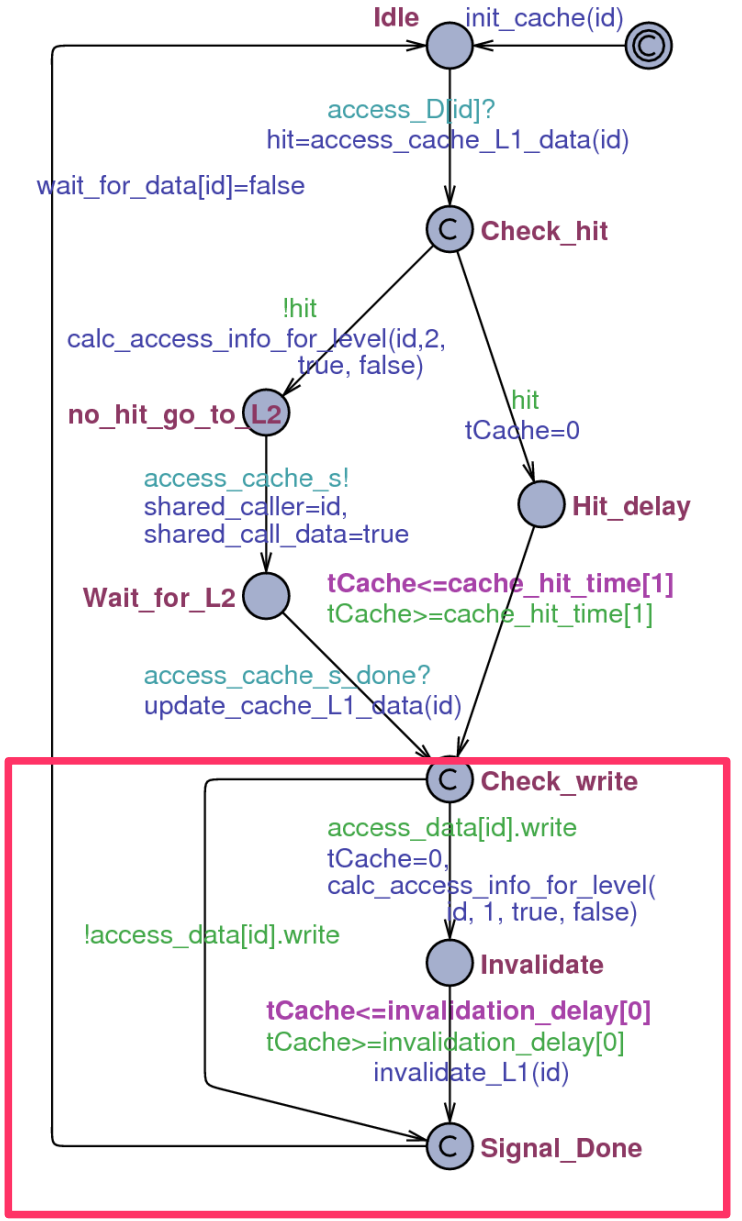




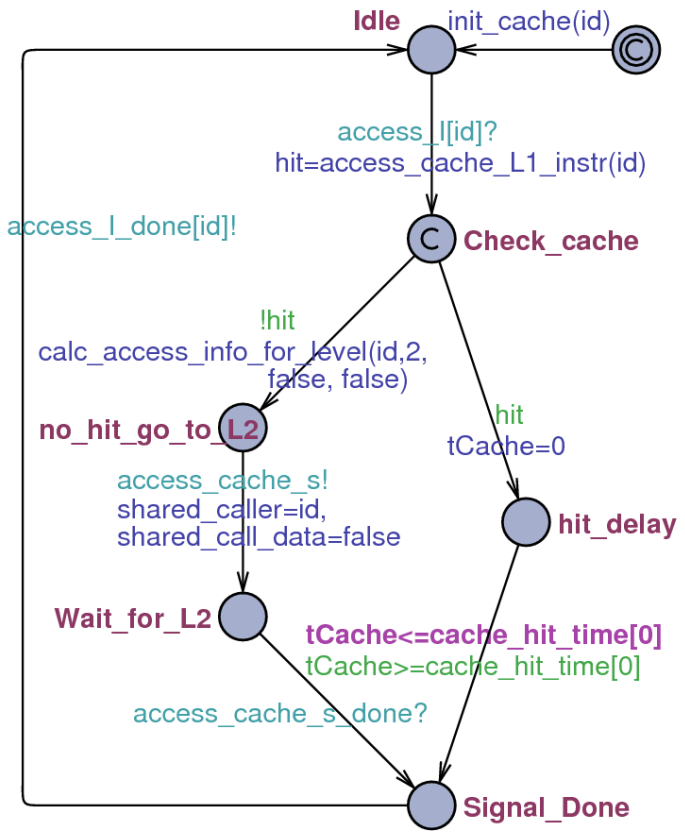
The L1 Caches

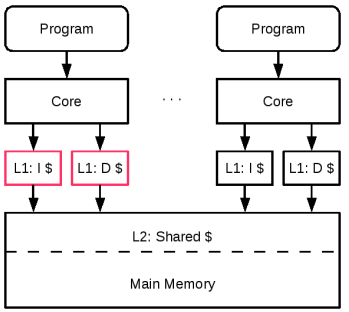


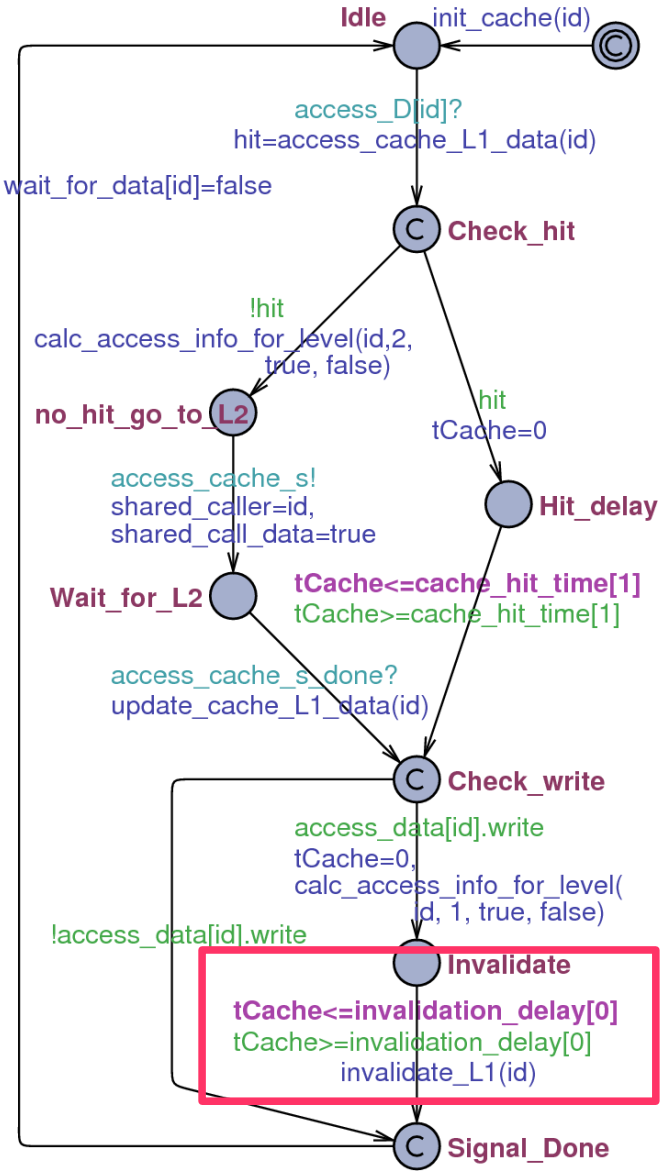




The L1 Caches

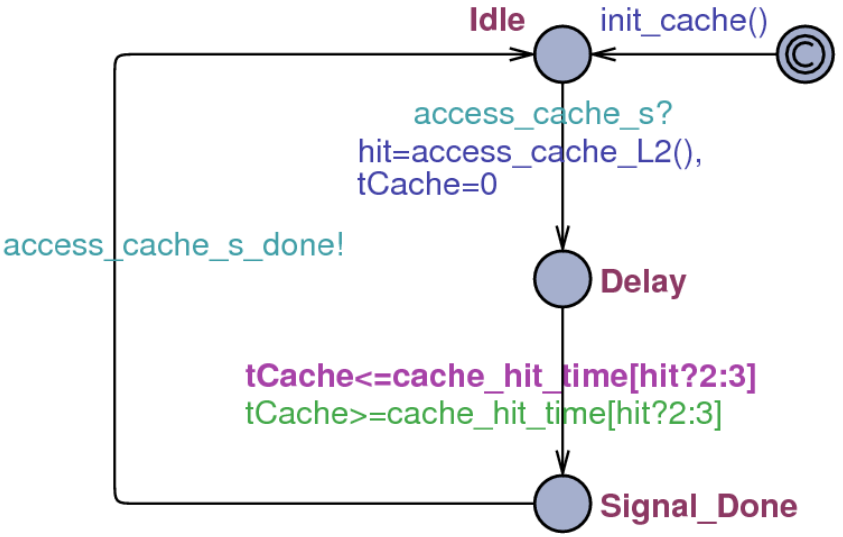


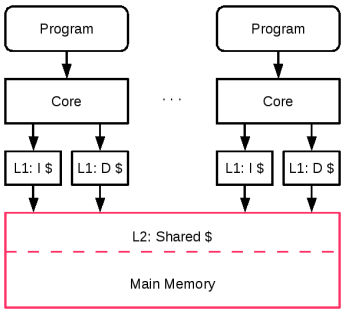




The L2 Cache

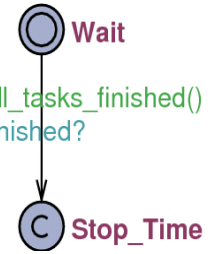
- L1 caches compete for access to the L2 cache
- Main memory is assumed to always hit

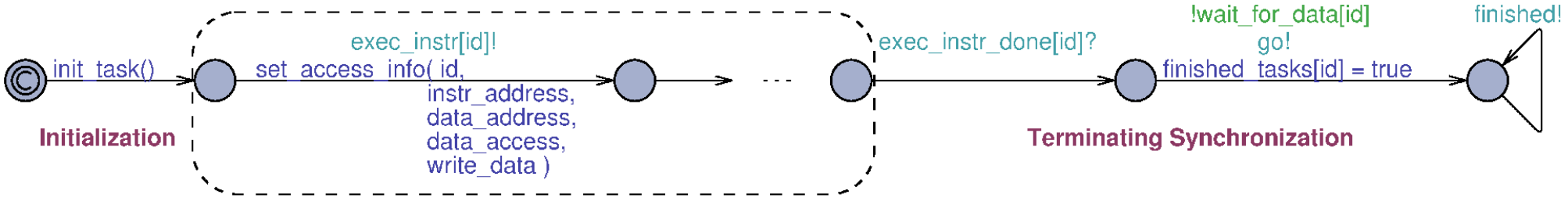


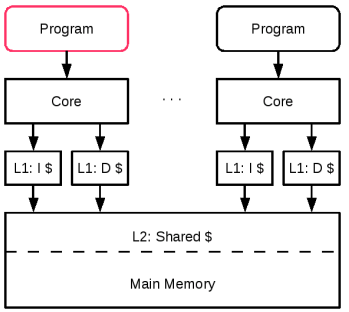


Program Model Interface

- When all programs/threads have finished, the time is stopped

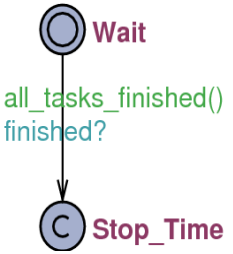


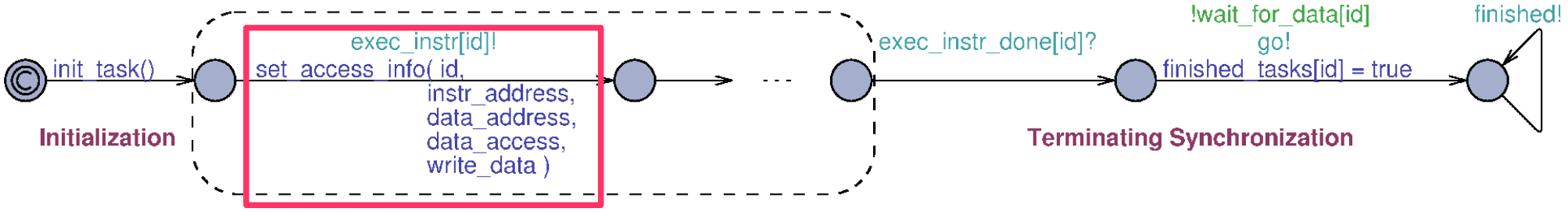


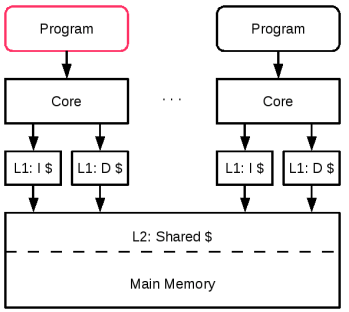


Program Model Interface

- When all programs/threads have finished, the time is stopped

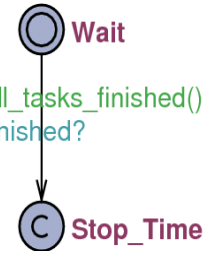


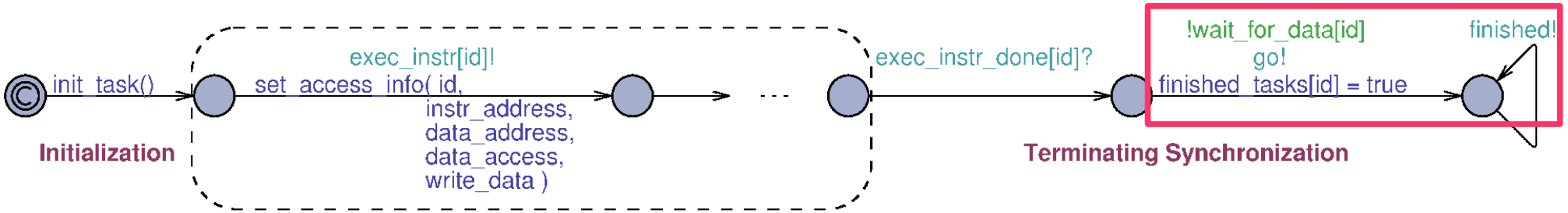


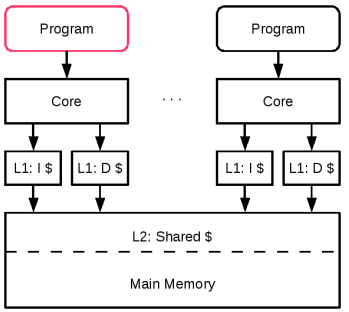


Program Model Interface

- When all programs/threads have finished, the time is stopped

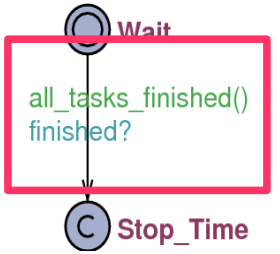


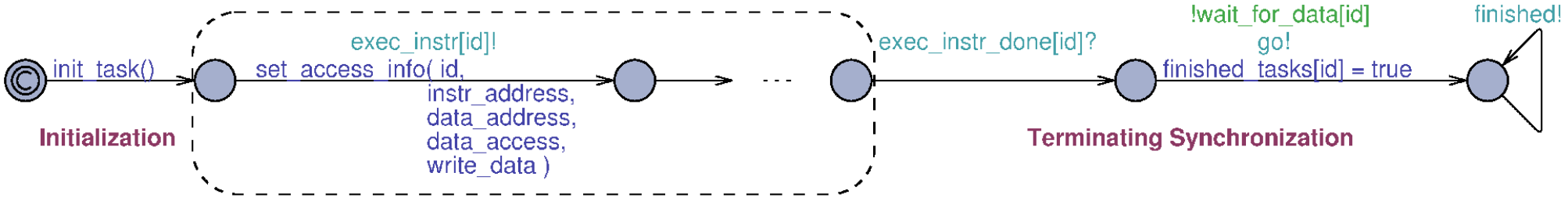


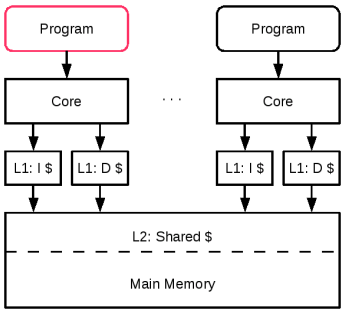


Program Model Interface

- When all programs/threads have finished, the time is stopped







Finding the WCET

- Binary search
  - `A[] t <= estimate`

Finding the WCET

- Binary search
  - `A[] t <= estimate`
- The `sup`-operator
  - `sup: t`

Experimental Evaluation

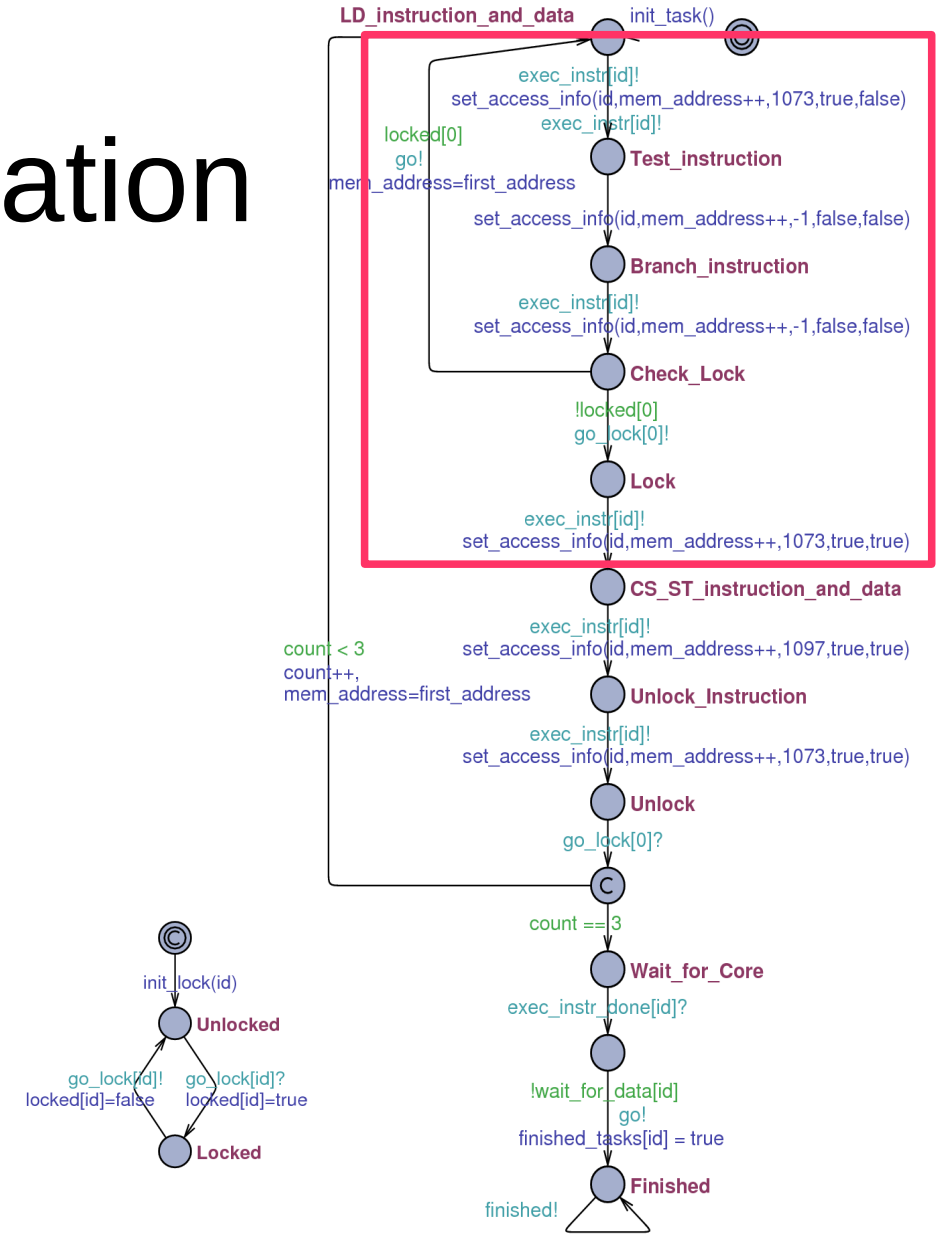
	L1 instr.	L1 data	L2 (shared)	RAM
Lines	4	4	8	∞
Words/Line	2	2	4	-
Sets	2	4	2	-
Latency	1	1	10	80
Replacement pol.	LRU	LRU	LRU	-

Experimental Evaluation

	L1 instr.	L1 data	L2 (shared)	RAM
Lines	4	4	8	∞
Words/Line	2	2	4	-
Sets	2	4	2	-
Latency	1	1	10	80
Replacement pol.	LRU	LRU	LRU	-

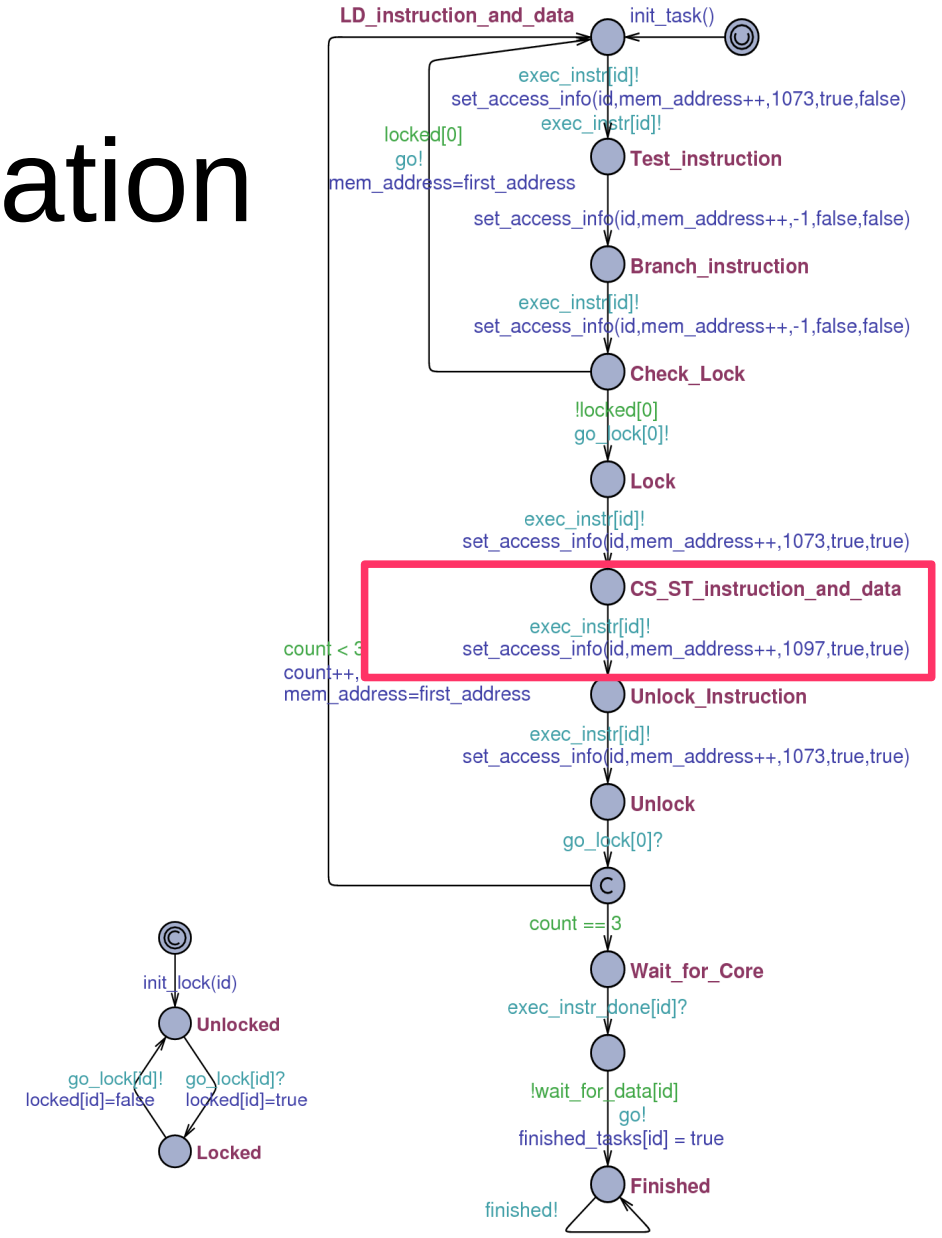
Experimental Evaluation

```
thread_body() {
    acquire_lock();
    shared_var = 0;
    release_lock();
    acquire_lock();
    shared_var = 1;
    release_lock();
    acquire_lock();
    shared_var = 2;
    release_lock();
}
```



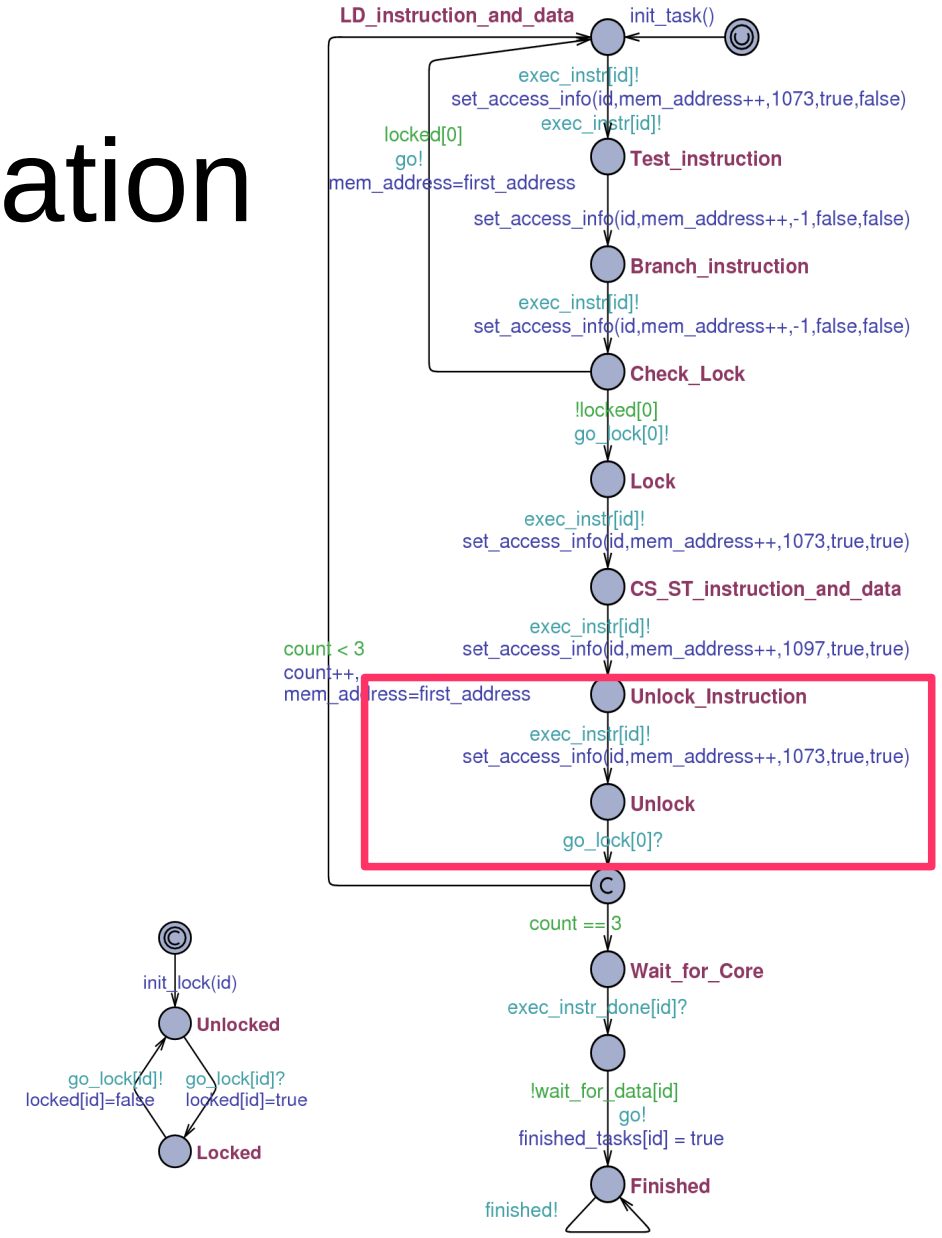
Experimental Evaluation

```
thread_body() {
    acquire_lock();
    shared_var = 0;
    release_lock();
    acquire_lock();
    shared_var = 1;
    release_lock();
    acquire_lock();
    shared_var = 2;
    release_lock();
}
```



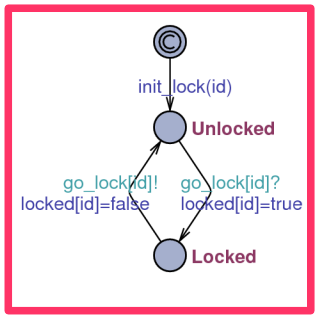
Experimental Evaluation

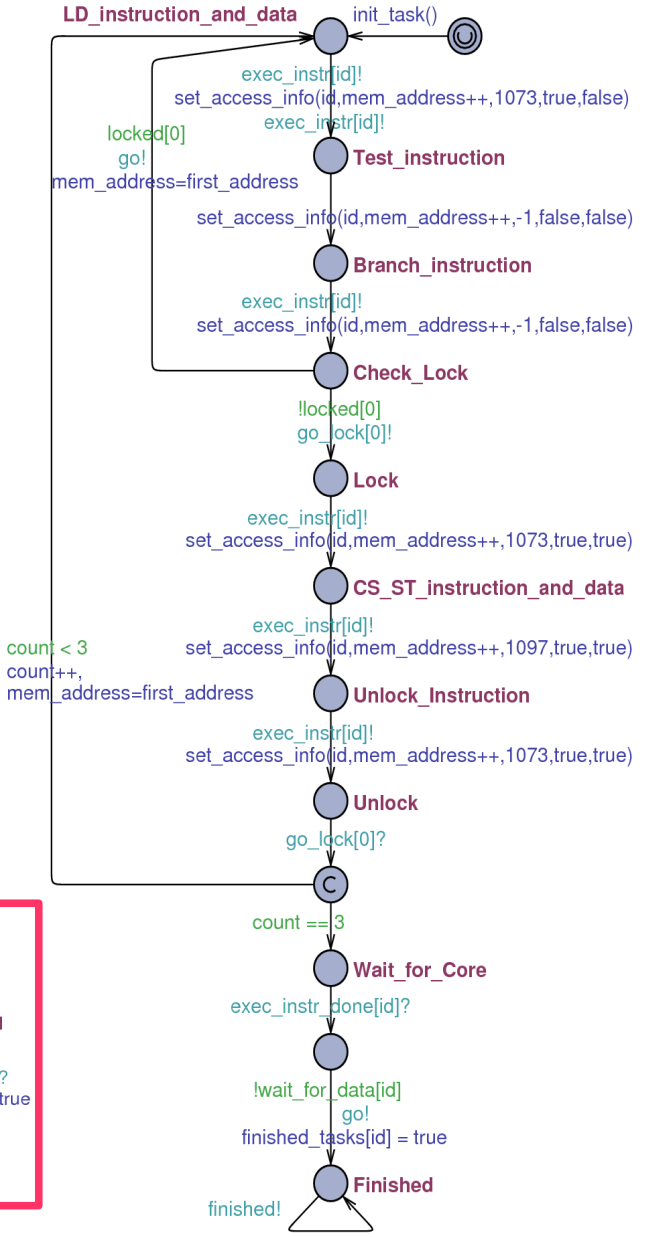
```
thread_body() {
    acquire_lock();
    shared_var = 0;
    release_lock();
    acquire_lock();
    shared_var = 1;
    release_lock();
    acquire_lock();
    shared_var = 2;
    release_lock();
}
```



Experimental Evaluation

```
thread_body() {
    acquire_lock();
    shared_var = 0;
    release_lock();
    acquire_lock();
    shared_var = 1;
    release_lock();
    acquire_lock();
    shared_var = 2;
    release_lock();
}
```





Experimental Evaluation

WCET = 636

- Base case:
  - 2 cores with corresponding caches
  - 1 thread of the program on each core
- Analysis Time:

	Binary Search	`sup`-Operator
1 step	1s	1s
Total	11s	1s

Experimental Evaluation

WCET = ???

- Scalability Test 1
  - 4 cores with corresponding caches
  - 1 thread of the program on each core
- Analysis Time:

	Binary Search	`sup`-Operator
1 step	>3h	>3h
Total	?	>3h

Experimental Evaluation

WCET = 1207

- Scalability Test 2
  - 2 cores and caches, 1 thread on each core
  - Thread 2 is released in the interval [0,1000]
- Analysis Time:

	Binary Search	`sup`-Operator
1 step	44s	42s
Total	~500s	42s

Experimental Evaluation

WCET = 619

- Scalability Test 3
  - 2 cores and caches, 1 thread on each core
  - # of cache lines are 2048 (L1-I,L1-D) and 8192 (L2)
- Analysis Time:

	Binary Search	`sup`-Operator
1 step	14s	14s
Total	~150s	14s

The Future...

- Scalability problem
  - Reduce complexity in models of HW and SW
  - Symmetry reduction
  - Partial order reduction
- Integration with SWEET
- Alternative approach
  - Model-checking as static analysis subroutine

Thank you! Questions?

- Andreas Gustavsson:
  andreas.sg.gustavsson@mdh.se
- Andreas Ermedahl:
  andreas.ermedahl@mdh.se
- Björn Lisper:
  bjorn.lisper@mdh.se
- Paul Pettersson:
  paul.pettersson@mdh.se