Implementation of Memory Centric Scheduling for COTS Multi-Core Real-Time Systems

Juan M. Rivas, Joël Goossens, **Xavier Poczekajlo**, Antonio Paolillo

Faculté des Sciences of the Université libre de Bruxelles, Belgium

ULB UNIVERSITÉ LIBRE DE BRUXELLES

PARTS

Multi-core: a challenge?

[. . .] multi-core hardware raises substantial **challenges** *[. . .]* Maiza et al. 2018

This poses new **challenges** *for scheduling [. . .]*, Agrawal et al. 2017

[. . .] engineers now face serious **challenges** *[. . .]*, Chang et al. 2013

One of the major **challenges** *[. . .] is the interference [. . .]*, Freitag et al. 2018

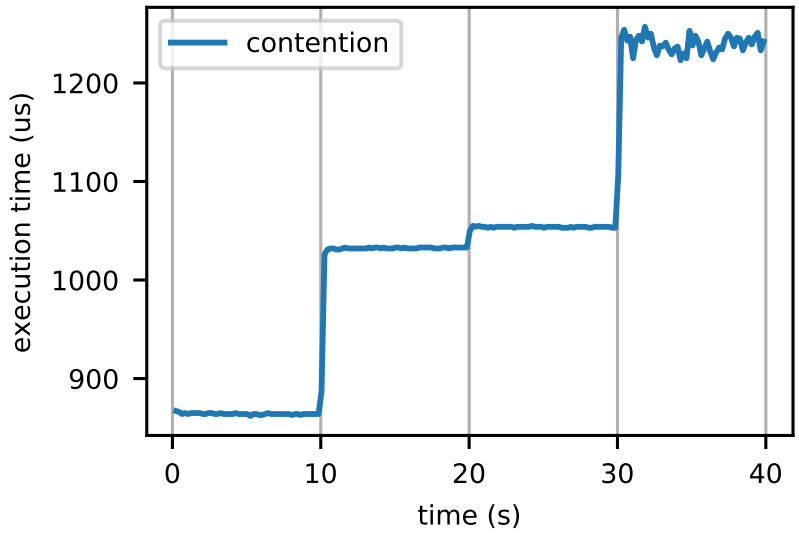
Multi-core: a challenge?

[. . .] multi-core hardware raises substantial **challenges** *[. . .]* Maiza et al. 2018

This poses new **challenges** *for scheduling [. . .]*, Agrawal et al. 2017

[. . .] engineers now face serious **challenges** *[. . .]*, Chang et al. 2013

One of the major **challenges** *[. . .] is the interference [. . .]*, Freitag et al. 2018



From 1 to 4 tasks

Existing approaches

- Time Division Multiple Access
- Memory Centric scheduling
- Bandwidth Regulator
- More advanced analysis
- ...

Existing approaches

- Time Division Multiple Access
- Memory Centric scheduling
- Bandwidth Regulator
- More advanced analysis
- ...

× Specialised hardware

× High analytical complexity

× Not implemented

Our work: Memory Centric

- ~~Specialised hardware~~ COTS

Our work: Memory Centric

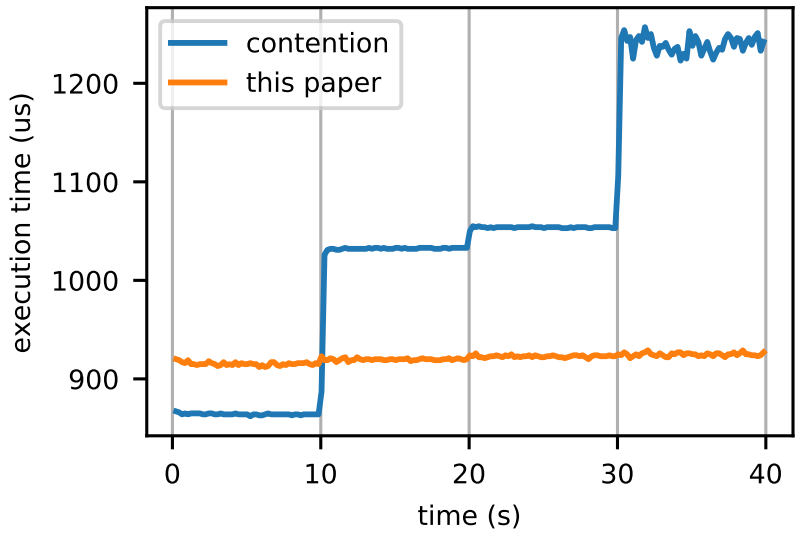
- ~~Specialised hardware~~ COTS
- ~~High analytical complexity~~ Easier to analyse

Our work: Memory Centric

- ~~Specialised hardware~~ COTS
- ~~High analytical complexity~~ Easier to analyse
- ~~Not implemented~~ Implemented in a RTOS

Our work: Memory Centric

- ~~Specialised hardware~~ COTS
- ~~High analytical complexity~~ Easier to analyse
- ~~Not implemented~~ Implemented in a RTOS



From 1 to 4 tasks

Table of Contents

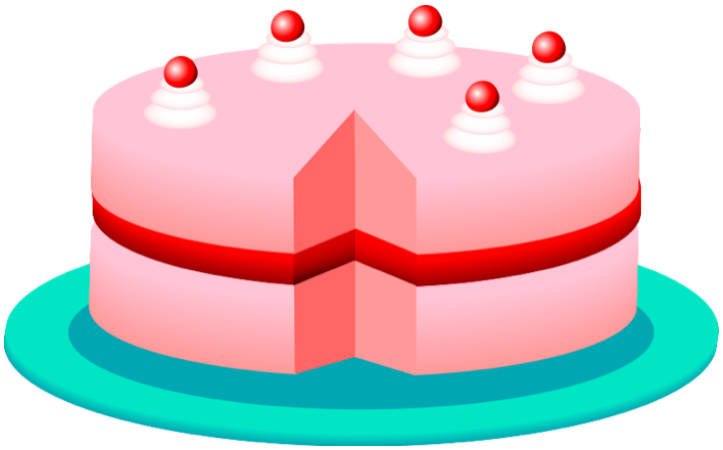
1. Memory Centric

2. Experimentation

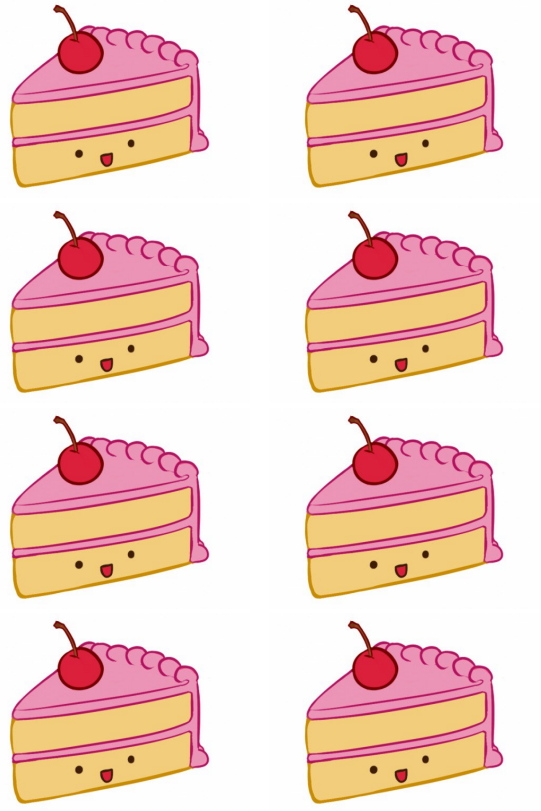
3. Schedulability analysis

Memory Centric

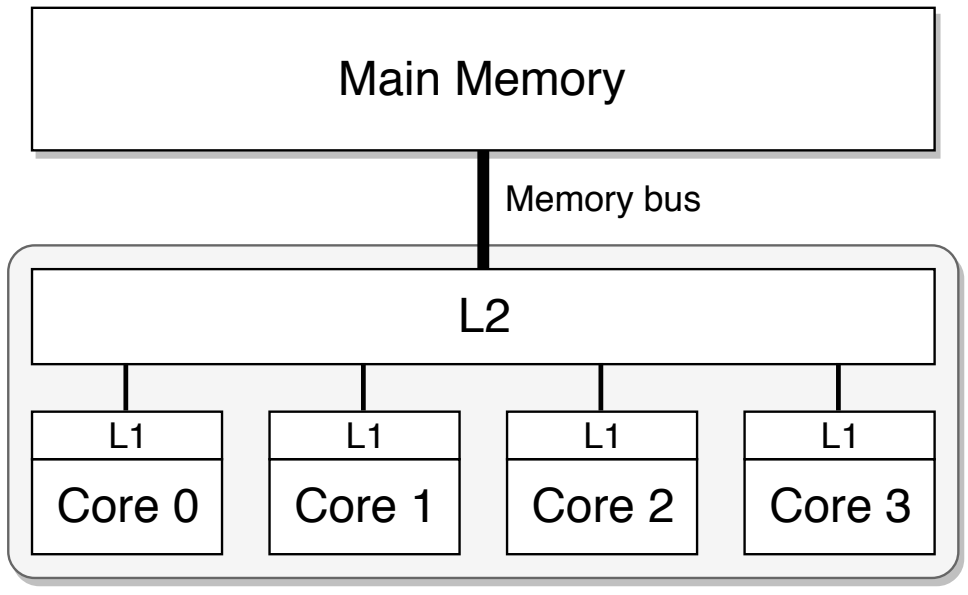
Assumptions: data



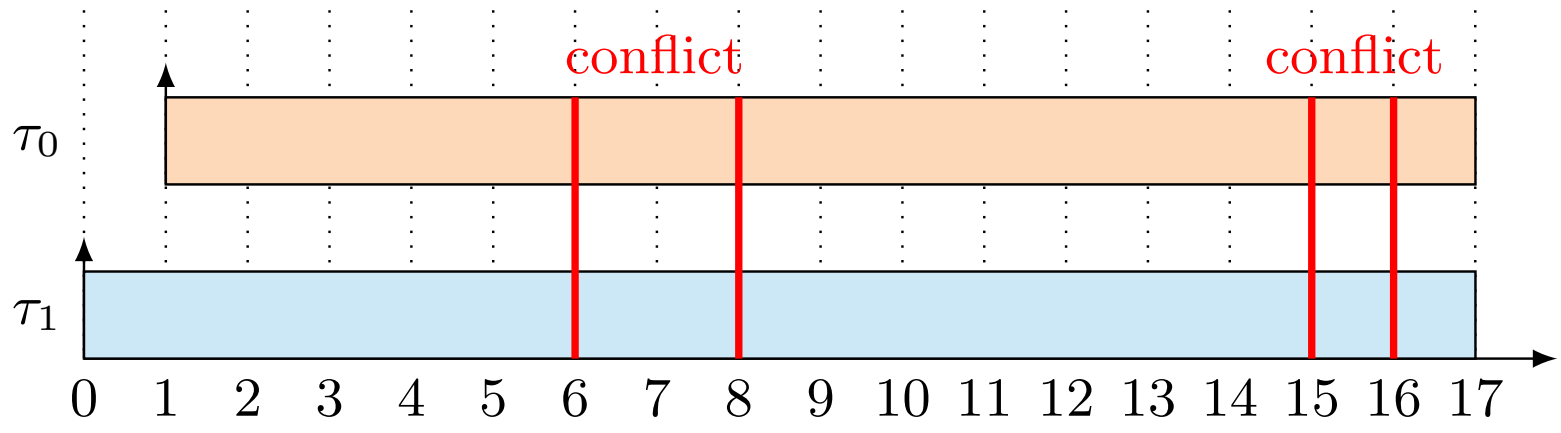
$\Rightarrow$



Assumptions: hardware

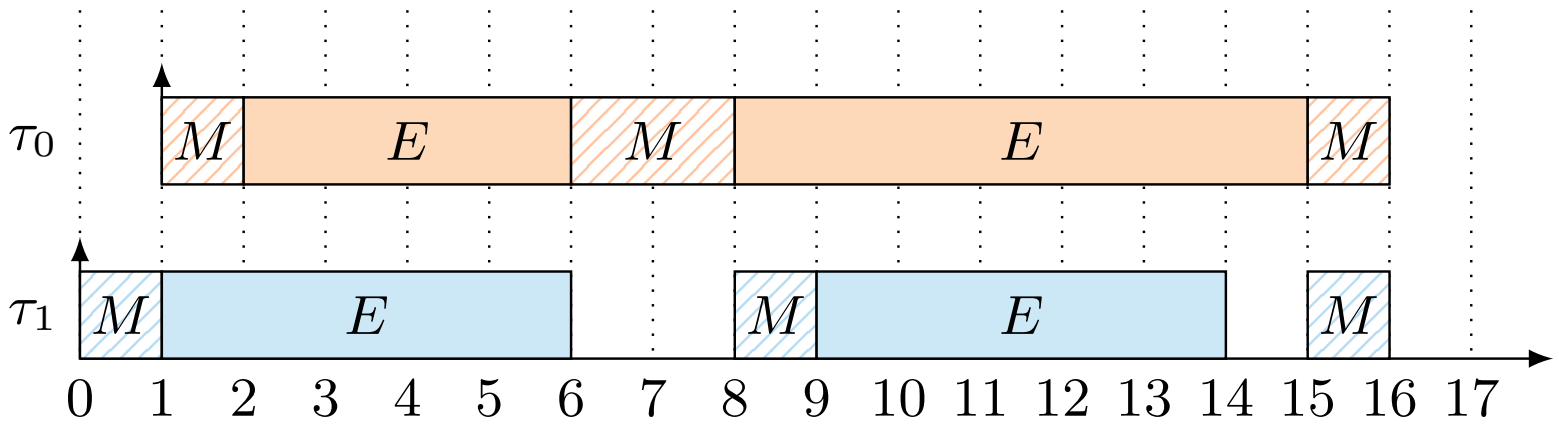


Contention vs Memory centric



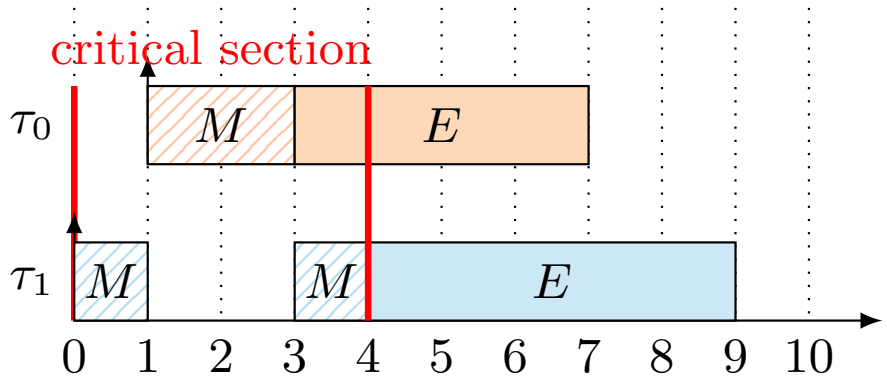
Contention approach

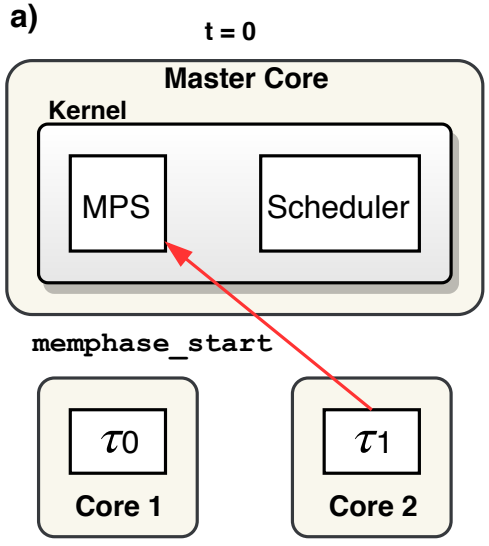
Contention vs Memory centric



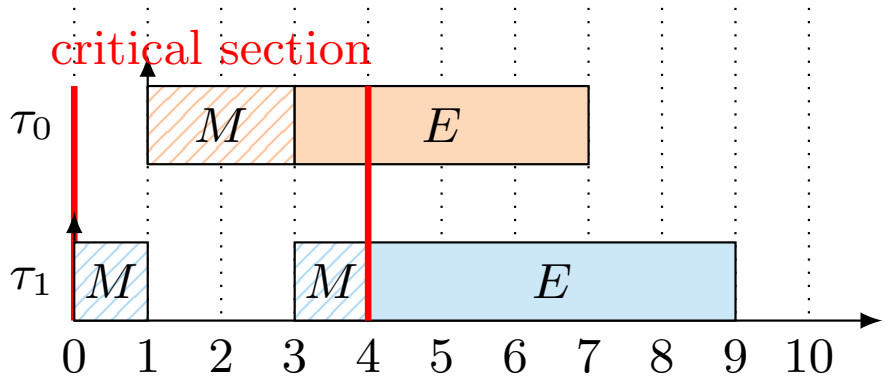
Memory centric approach, $\tau_0 \succ \tau_1$

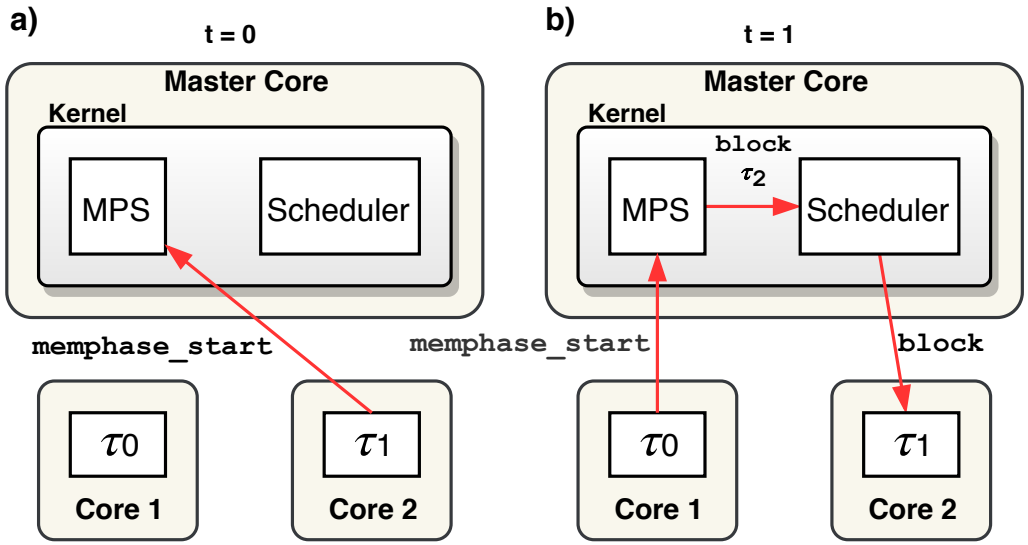
Critical section and preemptive mutex



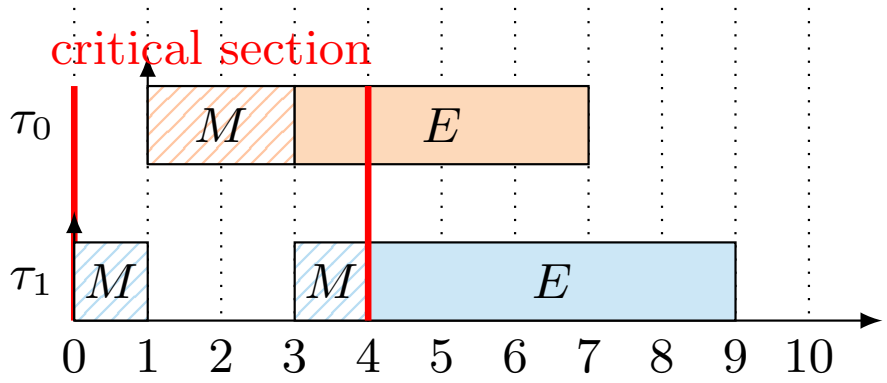


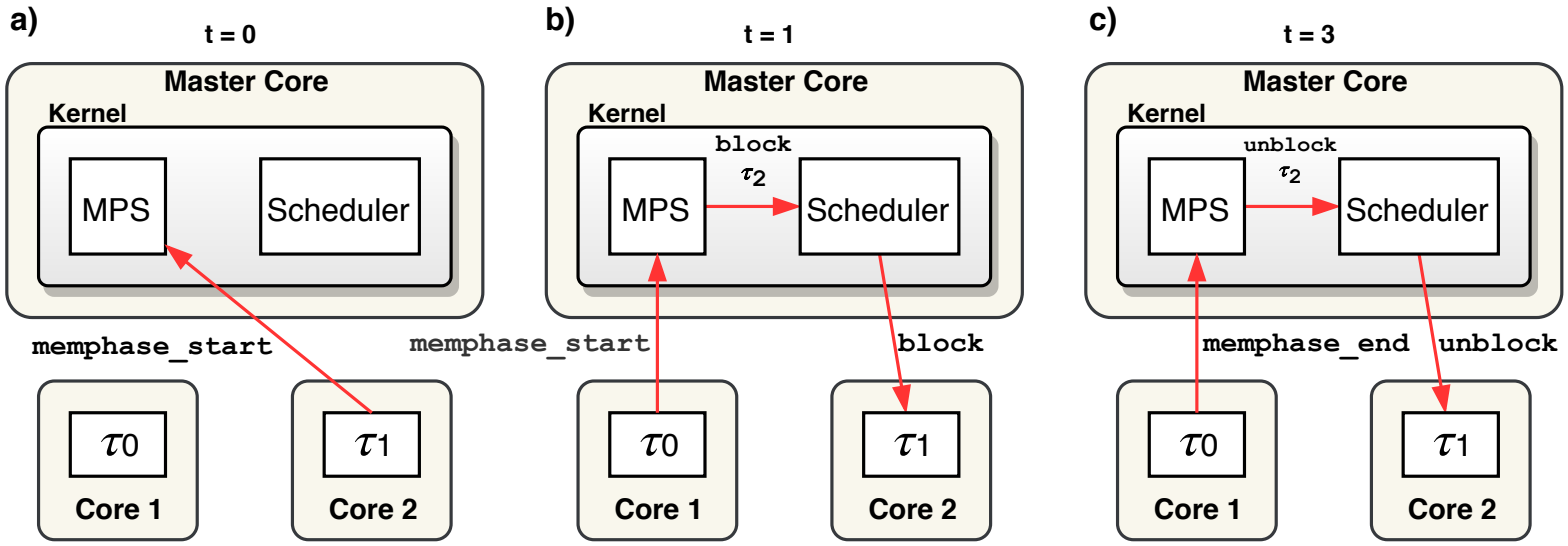
Critical section and preemptive mutex



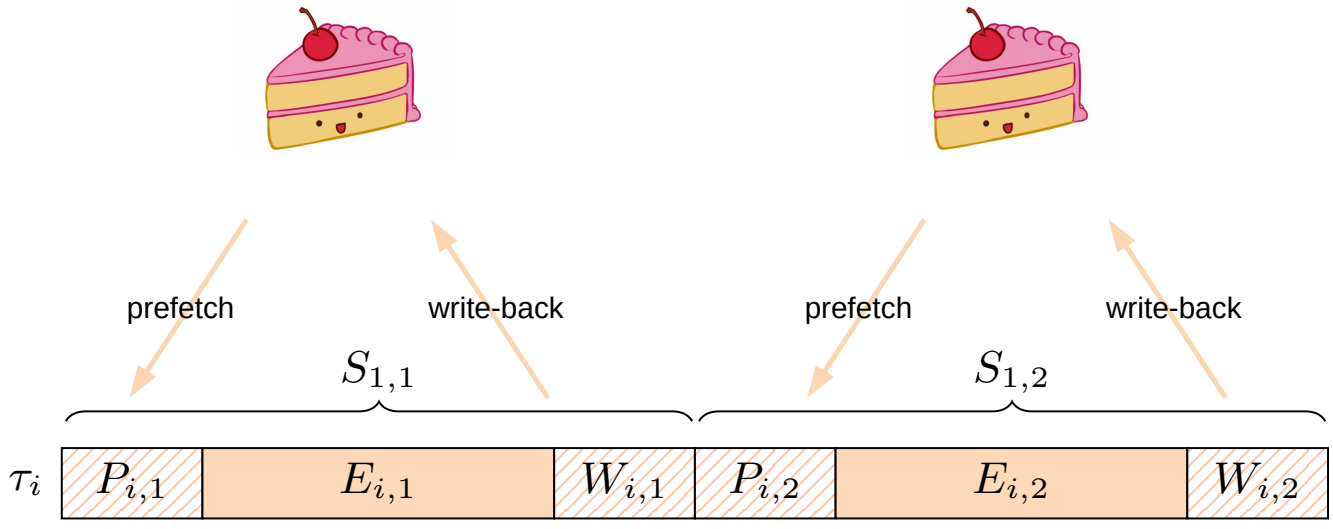


Critical section and preemptive mutex

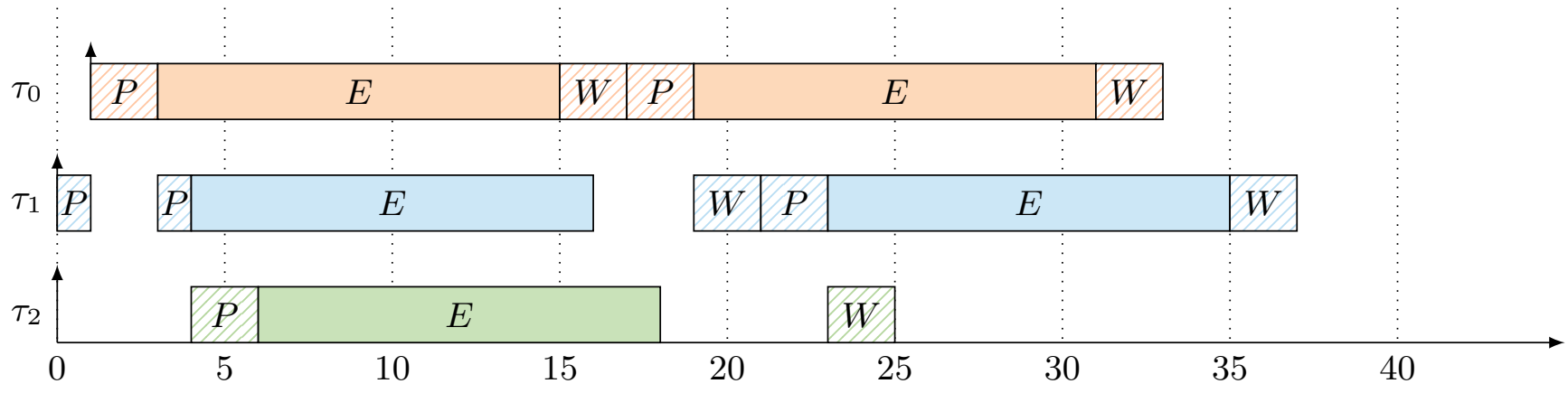




Adding semantics to memory phases

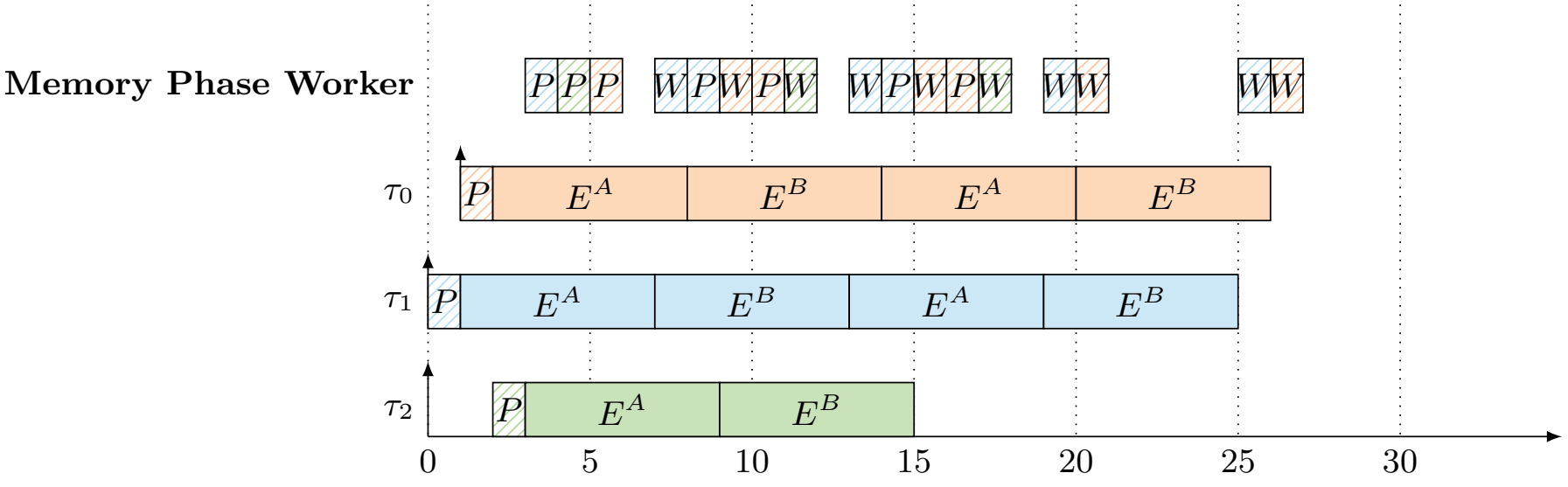


S-MP and A-MP approaches



Synchronous memory phases (S-MP), $\tau_0 \succ \tau_1 \succ \tau_2$

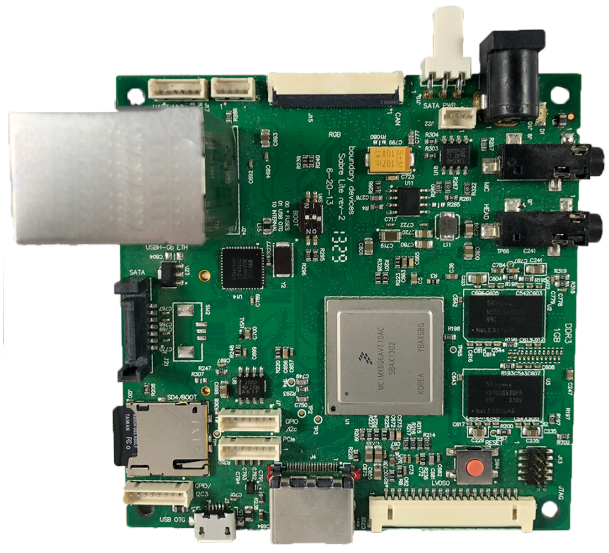
S-MP and A-MP approaches



Asynchronous memory phases (A-MP), $\tau_0 \succ \tau_1 \succ \tau_2$

Experimentation

Experimental setup



+

BD-SL-I.MX6
4 Cortex-A9 cores
1GB of DDR3 RAM

HIPPEROS

=

3 applications, 4 tasks, 430000 executions

Experimental setup

$= \quad 0.2\text{MB}$

$$\tau_0 \succ \tau_1 \succ \tau_2 \succ \tau_3$$

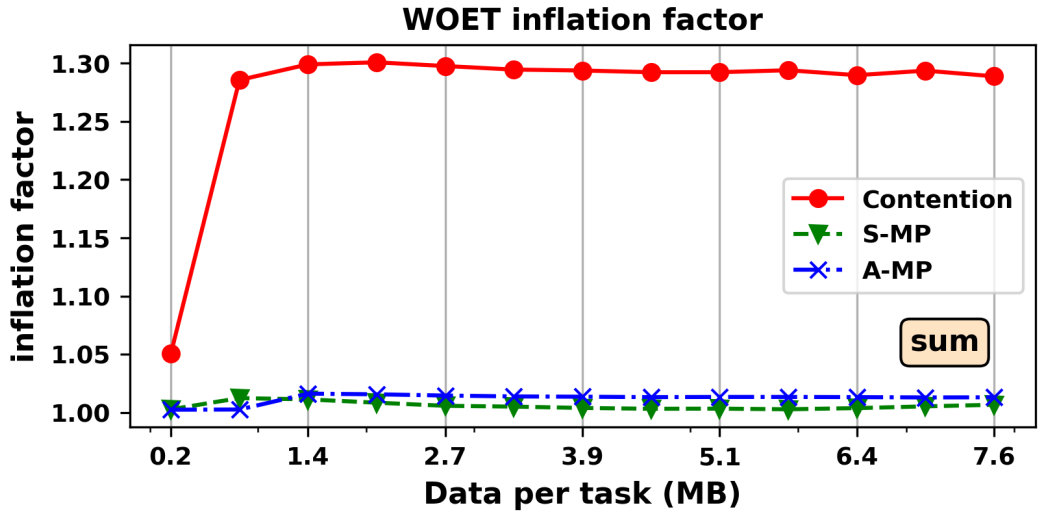
WOET Inflation factor τ_0

τ_0	$\emptyset$
$\emptyset$	$\emptyset$

WOET

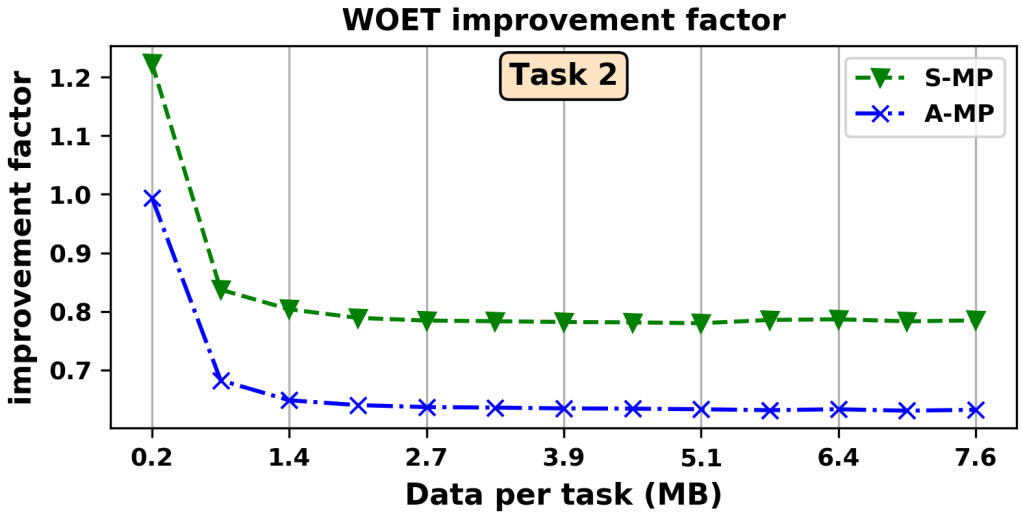
τ_0	τ_1
τ_2	τ_3

WOET



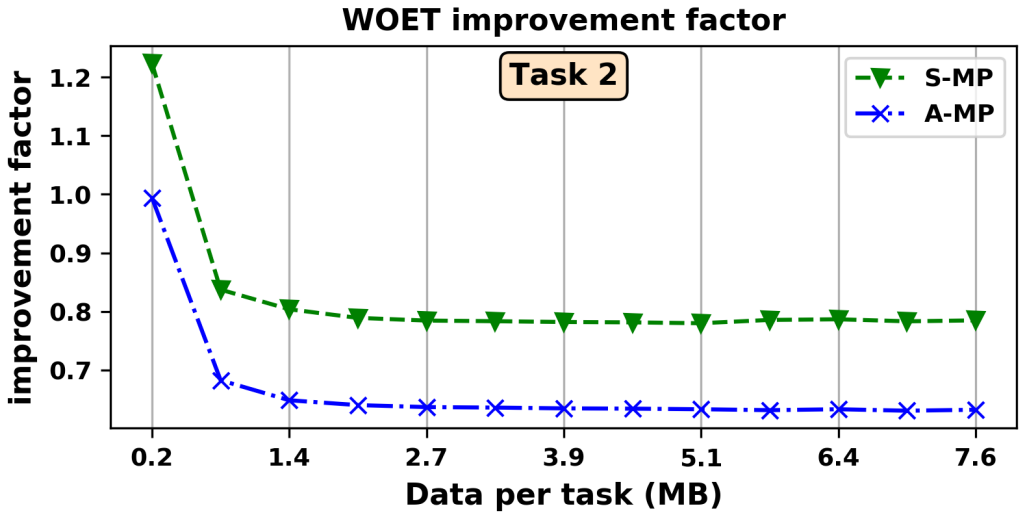
Improvement factor τ_2

$$\frac{\text{Mem. Centric WOET}}{\text{Contention WOET}}$$



Improvement factor τ_2

$$\frac{\text{Mem. Centric WOET}}{\text{Contention WOET}}$$

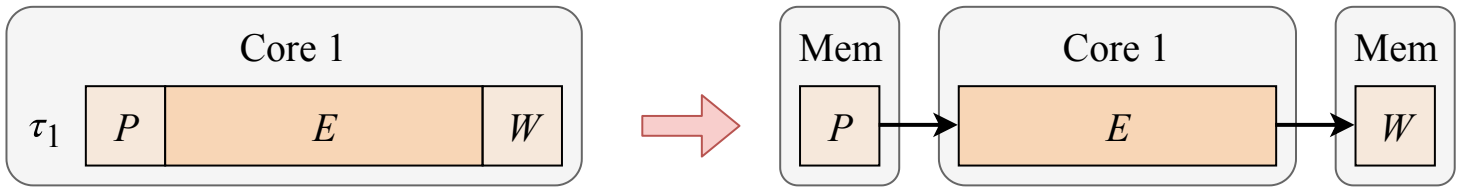


Up to 40% gain!

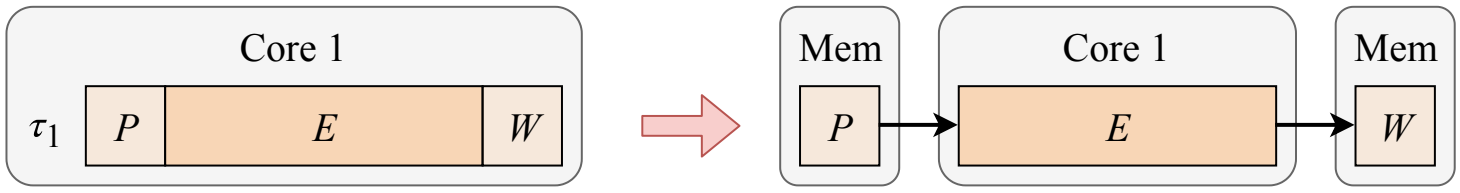
Schedulability analysis

Yao et al. 2016 validation

From PEW-tasks to distributed tasks



From PEW-tasks to distributed tasks



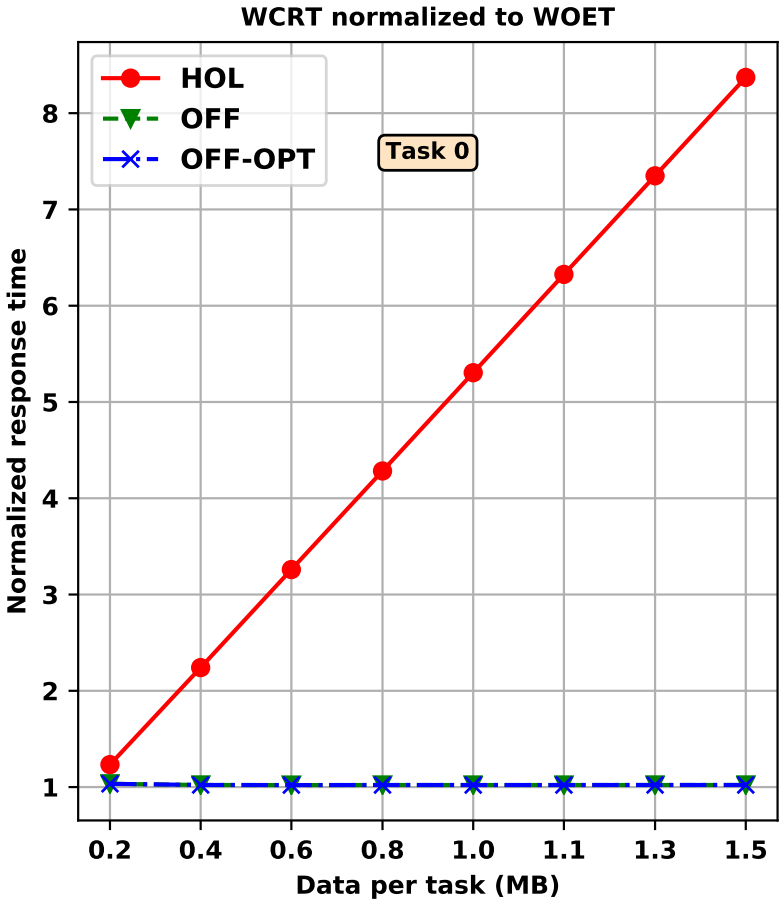
MAST tool

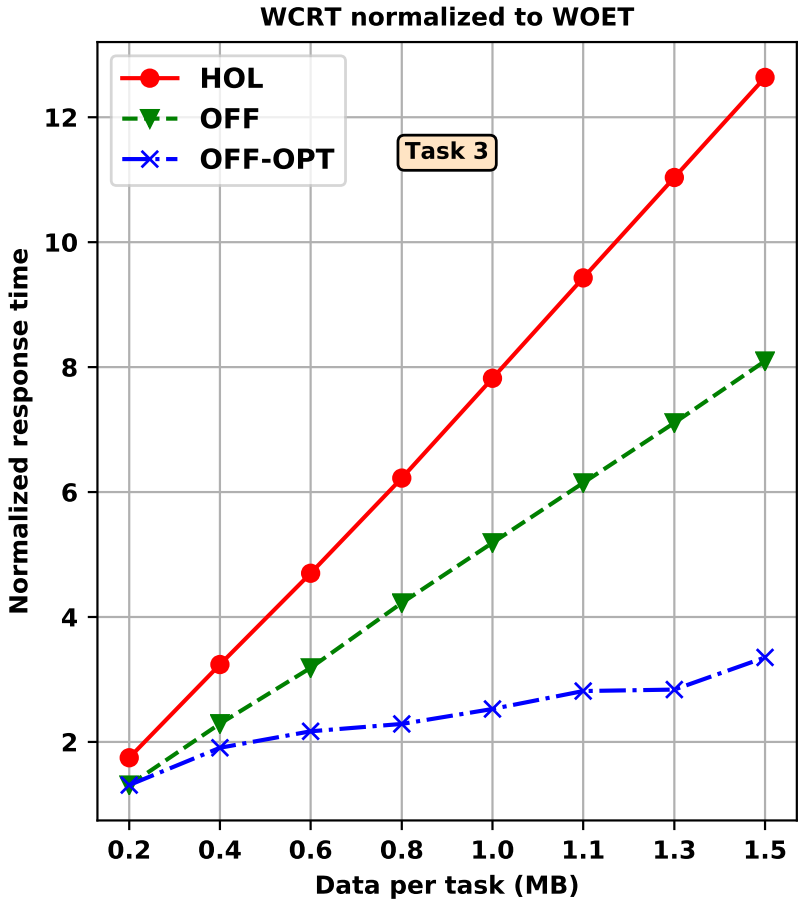
`mast.unican.es`

+

- Hollistic
- Offset
- Offset-optimised

Practical results





Paper's details

Implementation of Memory Centric Scheduling for COTS Multi-Core Real-Time Systems

Juan M. Rivas[1]
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
jrivasco@ulb.ac.be

Joël Goossens
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
joel.goossens@ulb.ac.be

Xavier Poczekajlo
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
xavier.poczekajlo@ulb.ac.be

Antonio Paolillo
HIPPEROS S.A., Louvain-la-Neuve, Belgium
antonio.paolillo@hipperos.com

Abstract

The demands for high performance computing with a low cost and low power consumption are driving a transition towards multi-core processors in many consumer and industrial applications. However, the adoption of multi-core processors in the domain of real-time systems faces a series of challenges that has been the focus of great research intensity during the last decade. These challenges arise in great part from the non real-time nature of the hardware arbiters that schedule the access to shared resources, such as the main memory. One solution proposed in the literature is called Memory Centric Scheduling, which defines a separate software scheduler for the sections of the tasks that will access the main memory, hence circumventing the low level unpredictable hardware arbiters. Several Memory Centric schedulers and associated theoretical analyses have been proposed, but as far as we know, no actual implementation of the required OS-level underpinnings to support dynamic event-driven Memory Centric Scheduling has been presented before. *In this paper* we aim to fill this gap, targeting cache based COTS multi-core systems. We will confirm via measurements the main theoretical benefits of Memory Centric Scheduling (e.g. task isolation). Furthermore, we will describe an effective schedulability analysis using concepts from distributed systems.

2012 ACM Subject Classification Computer systems organization → Real-time systems

Keywords and phrases real-time, multi-core, memory centric, predictability, implementation, rtos

Digital Object Identifier 10.4230/LIPIcs.ECRTS.2019.1

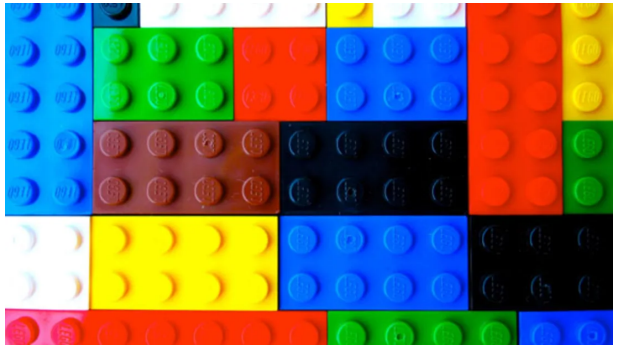
Funding *Juan M. Rivas*: This research was funded by Wallonia Region (Belgium) BEWARE grant PARTITA (1610375)

1 Introduction

Advancements in the manufacturing process of integrated electronics, in addition to the sheer size of the general computing market, are increasingly widening the offer and lowering the costs of commercial off-the-shelf (COTS) multi-core processors. While these commercial processors are generally designed with *average* performance in mind, their wide availability and low cost are pushing their adoption into real-time applications where a different set of requirements such as predictability and worst-case guarantees are needed.

[1] Corresponding author

31st Euromicro Conference on Real-Time Systems (ECRTS 2019).
Editor: Sophie Quinton; Article No. 1; pp. 1:1–1:23
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Paper's details

Implementation of Memory Centric Scheduling for COTS Multi-Core Real-Time Systems

Juan M. Rivas[1]
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
jrivasco@ulb.ac.be

Joël Goossens
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
joel.goossens@ulb.ac.be

Xavier Poczekajlo
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
xavier.poczekajlo@ulb.ac.be

Antonio Paolillo
HIPPEROS S.A., Louvain-la-Neuve, Belgium
antonio.paolillo@hipperos.com

Abstract

The demands for high performance computing with a low cost and low power consumption are driving a transition towards multi-core processors in many consumer and industrial applications. However, the adoption of multi-core processors in the domain of real-time systems faces a series of challenges that has been the focus of great research intensity during the last decade. These challenges arise in great part from the non real-time nature of the hardware arbiters that schedule the access to shared resources, such as the main memory. One solution proposed in the literature is called Memory Centric Scheduling, which defines a separate software scheduler for the sections of the tasks that will access the main memory, hence circumventing the low level unpredictable hardware arbiters. Several Memory Centric schedulers and associated theoretical analyses have been proposed, but as far as we know, no actual implementation of the required OS-level underpinnings to support dynamic event-driven Memory Centric Scheduling has been presented before. *In this paper* we aim to fill this gap, targeting cache based COTS multi-core systems. We will confirm via measurements the main theoretical benefits of Memory Centric Scheduling (e.g. task isolation). Furthermore, we will describe an effective schedulability analysis using concepts from distributed systems.

2012 ACM Subject Classification Computer systems organization → Real-time systems

Keywords and phrases real-time, multi-core, memory centric, predictability, implementation, rtos

Digital Object Identifier 10.4230/LIPIcs.ECRTS.2019.1

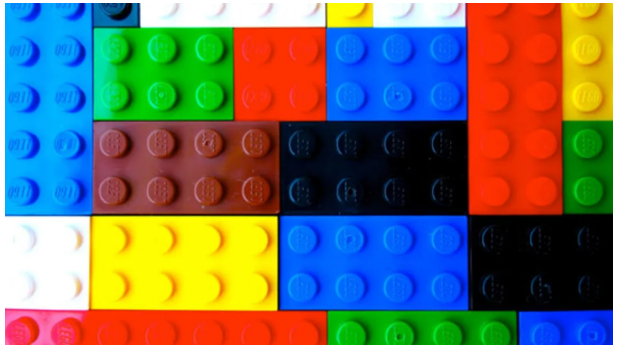
Funding *Juan M. Rivas*: This research was funded by Wallonia Region (Belgium) BEWARE grant PARTITA (1610375)

1 Introduction

Advancements in the manufacturing process of integrated electronics, in addition to the sheer size of the general computing market, are increasingly widening the offer and lowering the costs of commercial off-the-shelf (COTS) multi-core processors. While these commercial processors are generally designed with *average* performance in mind, their wide availability and low cost are pushing their adoption into real-time applications where a different set of requirements such as predictability and worst-case guarantees are needed.

[1] Corresponding author

31st Euromicro Conference on Real-Time Systems (ECRTS 2019).
Editor: Sophie Quinton; Article No. 1; pp. 1:1–1:23
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



```
void h_memphase_writeback(
        memphase_start();
        writeback(addr, by
        memphase_end();
}
```

Paper's details

Implementation of Memory Centric Scheduling for COTS Multi-Core Real-Time Systems

Juan M. Rivas[1]
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
jrivasco@ulb.ac.be

Joël Goossens
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
joel.goossens@ulb.ac.be

Xavier Poczekajlo
PARTS Research Centre, Université libre de Bruxelles, Brussels, Belgium
xavier.poczekajlo@ulb.ac.be

Antonio Paolillo
HIPPEROS S.A., Louvain-la-Neuve, Belgium
antonio.paolillo@hipperos.com

Abstract

The demands for high performance computing with a low cost and low power consumption are driving a transition towards multi-core processors in many consumer and industrial applications. However, the adoption of multi-core processors in the domain of real-time systems faces a series of challenges that has been the focus of great research intensity during the last decade. These challenges arise in great part from the non real-time nature of the hardware arbiters that schedule the access to shared resources, such as the main memory. One solution proposed in the literature is called Memory Centric Scheduling, which defines a separate software scheduler for the sections of the tasks that will access the main memory, hence circumventing the low level unpredictable hardware arbiters. Several Memory Centric schedulers and associated theoretical analyses have been proposed, but as far as we know, no actual implementation of the required OS-level underpinnings to support dynamic event-driven Memory Centric Scheduling has been presented before. *In this paper* we aim to fill this gap, targeting cache based COTS multi-core systems. We will confirm via measurements the main theoretical benefits of Memory Centric Scheduling (e.g. task isolation). Furthermore, we will describe an effective schedulability analysis using concepts from distributed systems.

2012 ACM Subject Classification Computer systems organization → Real-time systems

Keywords and phrases real-time, multi-core, memory centric, predictability, impl

Digital Object Identifier 10.4230/LIPIcs.ECRTS.2019.1

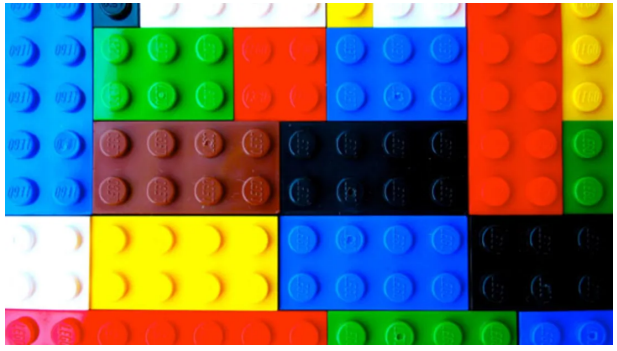
Funding *Juan M. Rivas*: This research was funded by Wallonia Region (Belgium) PARTITA (1610375)

1 Introduction

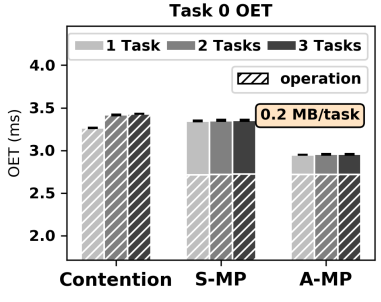
Advancements in the manufacturing process of integrated electronics, in addi
size of the general computing market, are increasingly widening the offer a
costs of commercial off-the-shelf (COTS) multi-core processors. While th
processors are generally designed with *average* performance in mind, their
and low cost are pushing their adoption into real-time applications where
requirements such as predictability and worst-case guarantees are needed.

[1] Corresponding author

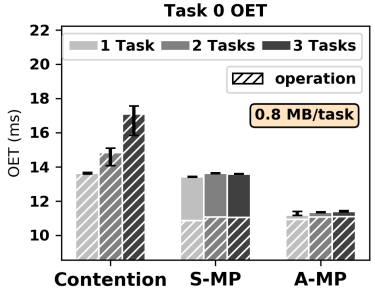
31st Euromicro Conference on Real-Time Systems (ECRTS 2019).
Editor: Sophie Quinton; Article No. 1; pp. 1:1–1:23
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



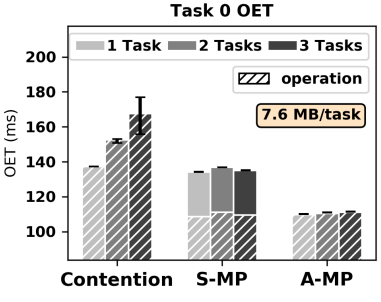
```
void h_memphase_writeback(
        memphase_start();
        writeback(addr, by
        memphase_end();
}
```



(a)

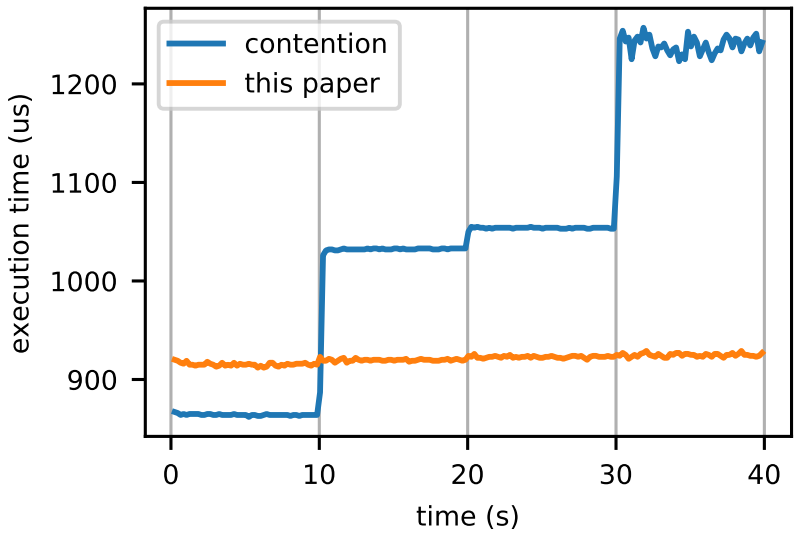


(b)



(c)

Conclusion


contention
this paper
execution time (us)
1200
1100
1000
900
0
10
20
30
40
time (s)

? > jrivasco@ulb.ac.be