Collaborative Resource Management for Multi-Core AUTOSAR OS

Renata Martins Gomes
Fabian Mauroner
Marcel Baunach

Graz University of Technology
Institute for Technical Informatics

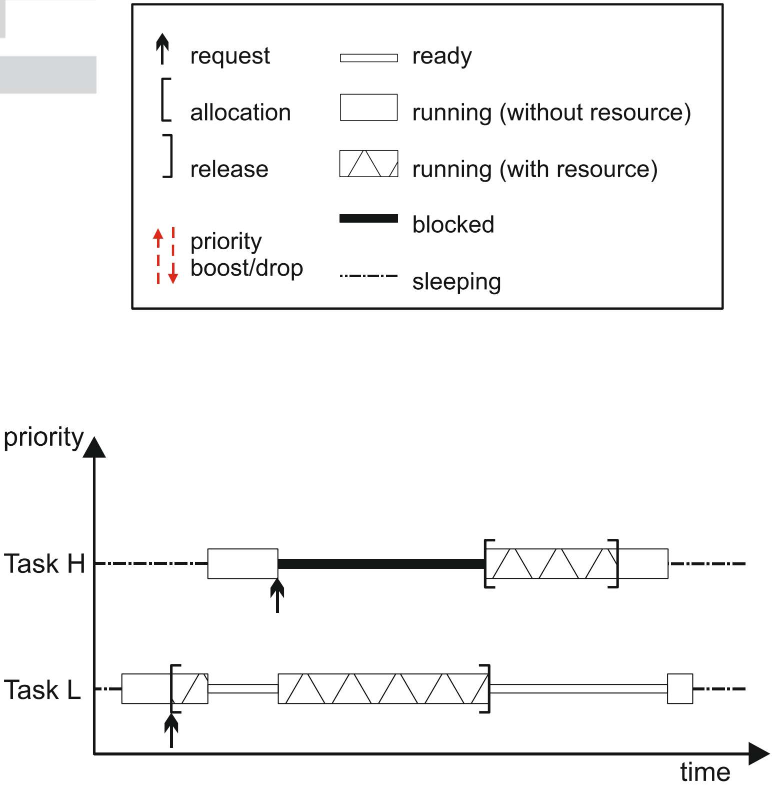
Echtzeit 2015 - Betriebssysteme und Echtzeit
Boppard, November 12-13, 2015

Agenda

- General Concepts
- AUTOSAR
- DynamicHinting
- Proposed Improvements
- Summary and Outlook

General Concepts

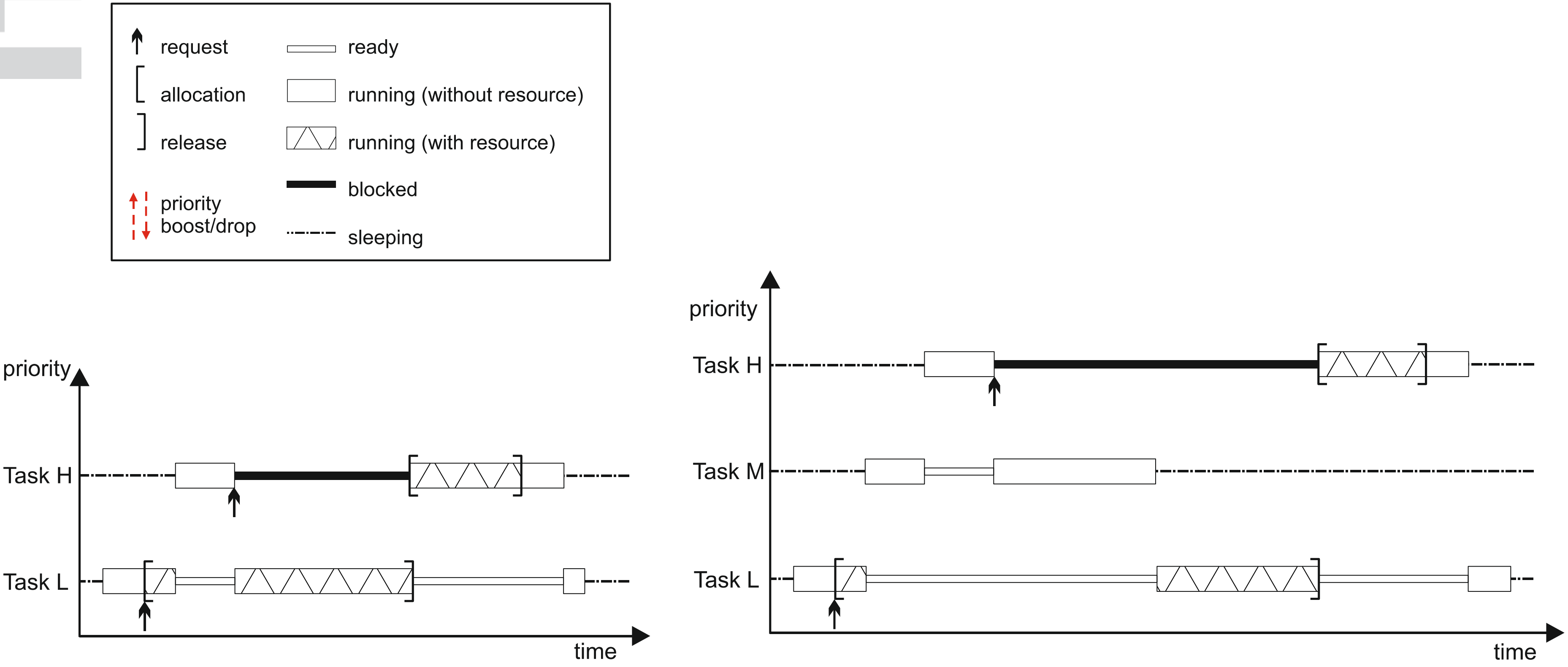
Priority Inversion



Bounded

General Concepts

Priority Inversion



Bounded

Unbounded

AUTOSAR OS

- Based on OSEK (**O**ffene **S**ysteme und deren Schnittstellen für die **E**lektronik im **K**raftfahrzeug)
- Designed with single-core in mind
- Extended for multi-core support
- For strictly static systems

AUTOSAR OS

Multi-Core Support

AUTOSAR has been extended to support multi-core processors. One example is in the resource sharing mechanisms:

In AUTOSAR OS, there was no reformulation of the concepts behind the resource management.

Instead, a second sharing mechanism was created to deal only with the resource sharing across cores.

→ Two incompatible concepts for sharing resources

AUTOSAR OS

Resources

Sharing resources among tasks within **one** core

- Defined by OSEK
- OSEK-PCP (highest-locker protocol)
  - Each resource has a ceiling priority
  - Task priority is immediately raised to ceiling upon allocation
  - Avoids unbounded priority inversion and deadlocks

  → Sub-ceiling priority tasks won't run, even if they do not need (yet) the resource
- Restricted system calls while holding a resource
  - Tasks cannot suspend themselves (wait for event, sleep, etc)
- Deadlock free

AUTOSAR OS

Spinlocks

Sharing resources among tasks only **across** cores

- Not allowed in same core
- Tasks priorities are meaningless across cores
- No priority-related resource manager defined
  - Spinlock allocation order is hw/sw dependent
- Busy waiting for already allocated spinlocks
  - Advice: disable interrupts while waiting/holding a spinlock
- Nesting only in globally well-defined order

→ Deadlocks are avoided, but unbounded priority inversion may occur across cores

DynamicHinting

Collaborative Resource Management

- Resource management approach
- Tasks collaborate with each other when sharing exclusive resources
- Hints sent to blocking tasks to indicate their spurious influence
- Tasks are free to ignore or follow a hint (release the resource)
    - Contracts can be specified, e.g. for enforcing behavior and real-time constraints
- Hint receiving and handling:
    - Explicit query
    - Early wakeup
    - Hint handlers

DynamicHinting

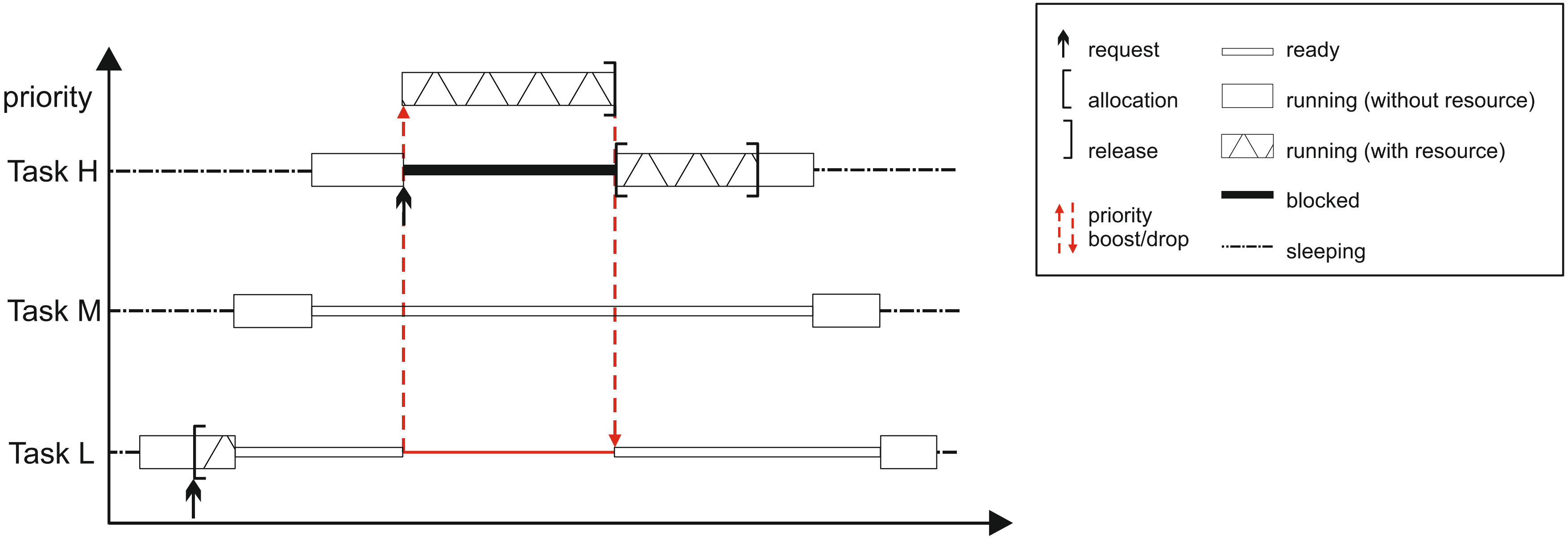
Collaborative Resource Management

- Reduces priority inversion
- Requires collaboration between tasks

DynamicHinting

Collaborative Resource Management

- Reduces priority inversion
- Requires collaboration between tasks

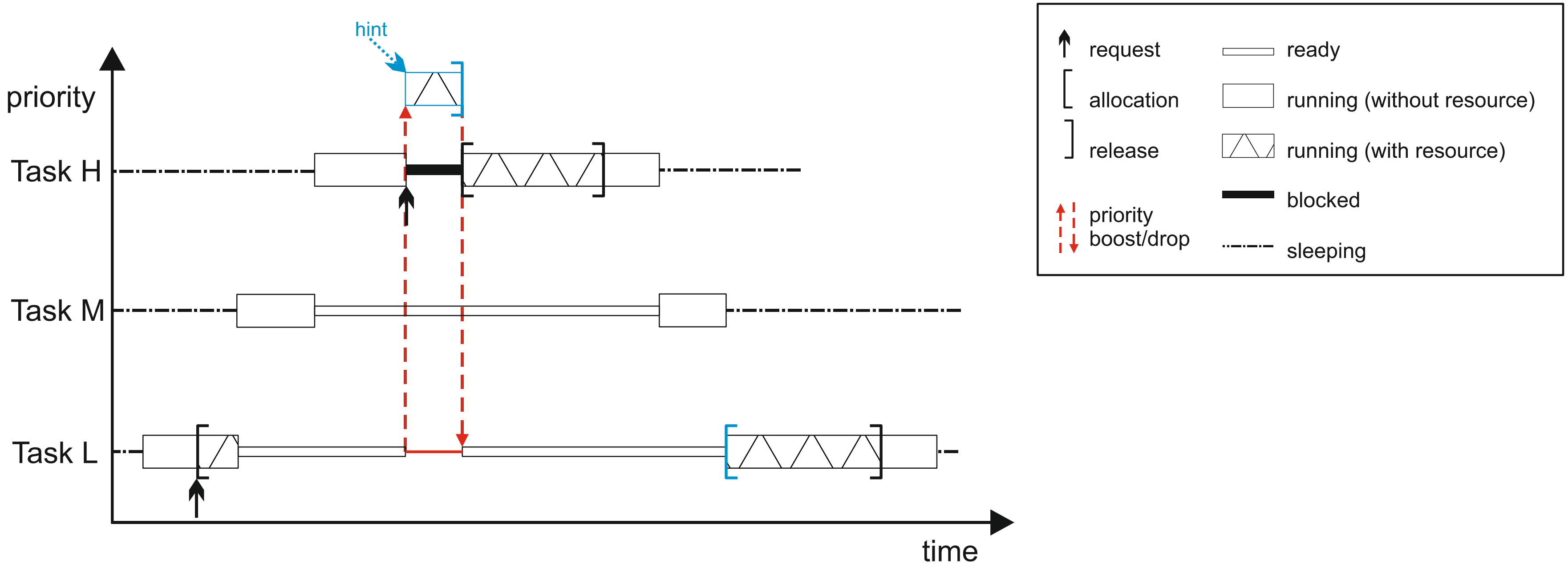


Priority Inheritance Protocol (PIP)

DynamicHinting

Collaborative Resource Management

- Reduces priority inversion
- Requires collaboration between tasks



PIP with DynamicHinting

Proposed Improvements

in AUTOSAR

- Introduce DynamicHinting
- Establish cross-core priority awareness
- Define priority-based allocation order for spinlocks
- Remove unbounded priority inversion for spinlocks
- Reduce bounded priority inversion

Proposed Improvements

Testbed

Tests conducted with widely-used hardware and software in the automotive domain:

- Triple-core Infineon AURIX^{TM} TC297B microcontroller
- Open-source OSEK and AUTOSAR compliant ERIKA Enterprise OS V2.5.0

→ External oscilloscope measures digital-output lines driven by tasks

Proposed Improvements

Allocation Order for Spinlocks

In case two or more tasks wait for a spinlock, which one receives it first?

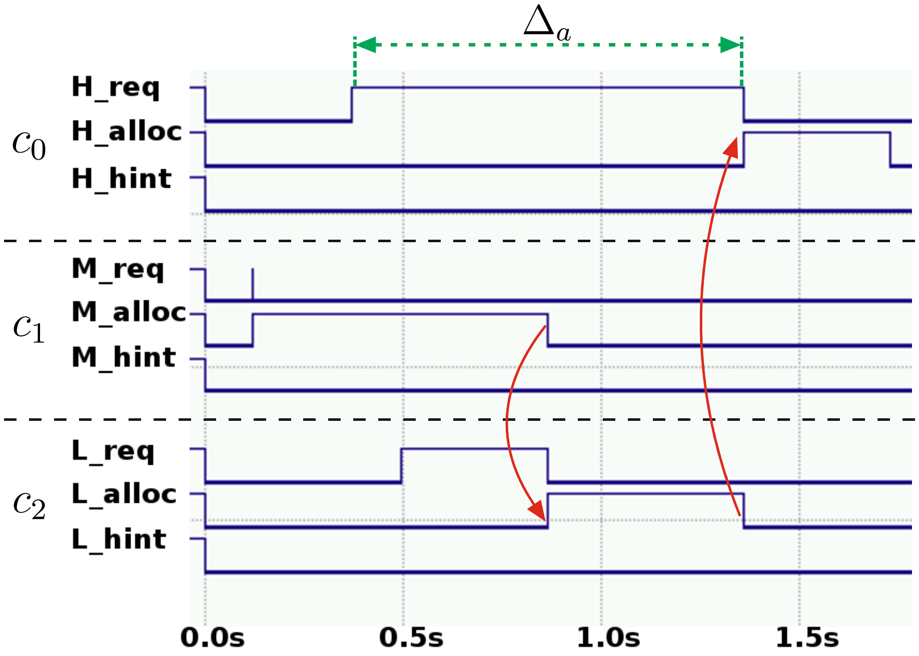
- Tasks waiting for a spinlock should receive it in order of task priority
- AUTOSAR does not define this order, leaving it dependent on the implementation
  - E.g. Test system:
    - Software (OS): FIFO queue
    - Hardware: core-specific priority queue

Changes on the busy-waiting mechanism:

- Only the highest-priority task waiting actually polls lock variable
- Others wait until they have highest priority in waiting queue

Proposed Improvements

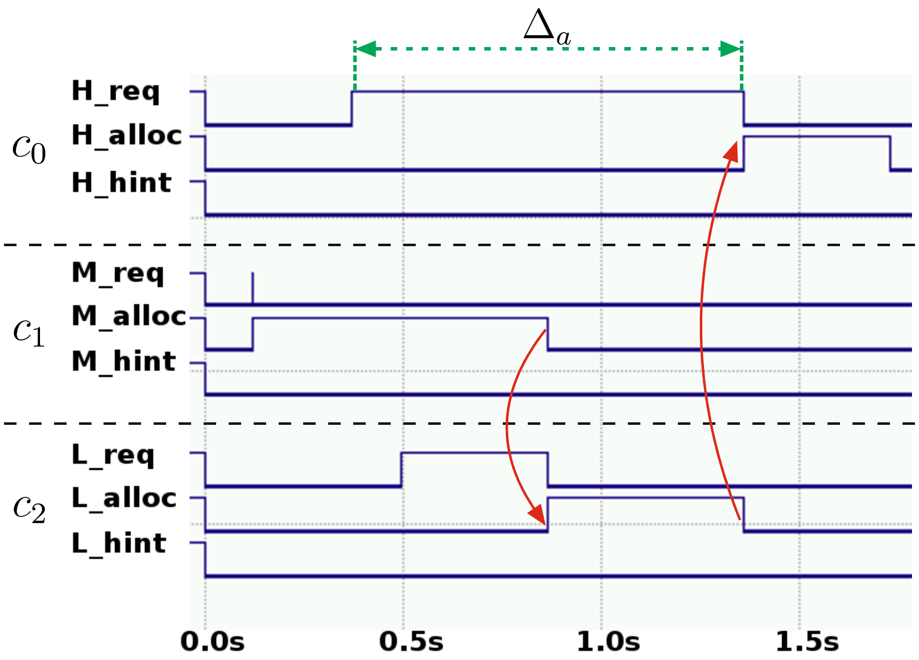
Allocation Order for Spinlocks



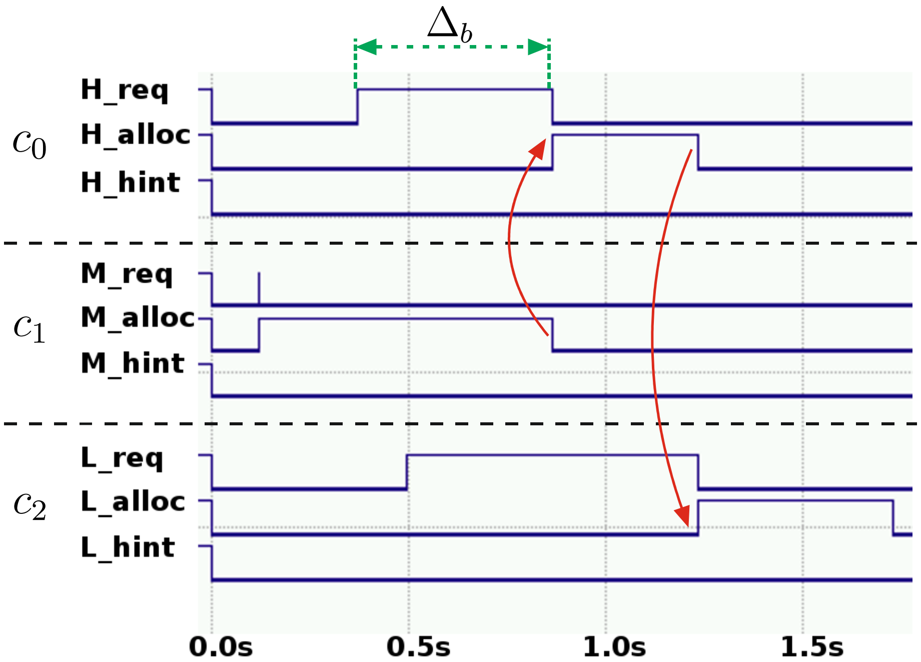
Unmodified ERIKA

Proposed Improvements

Allocation Order for Spinlocks



Unmodified ERIKA



Modified ERIKA

Proposed Improvements

Handling Unbounded Priority Inversion

No unbounded priority inversion should ever occur in a real-time system, but so far in AUTOSAR:

- No resource management protocol implemented across cores
- No priorities known across cores
- Unbounded priority inversion might occur

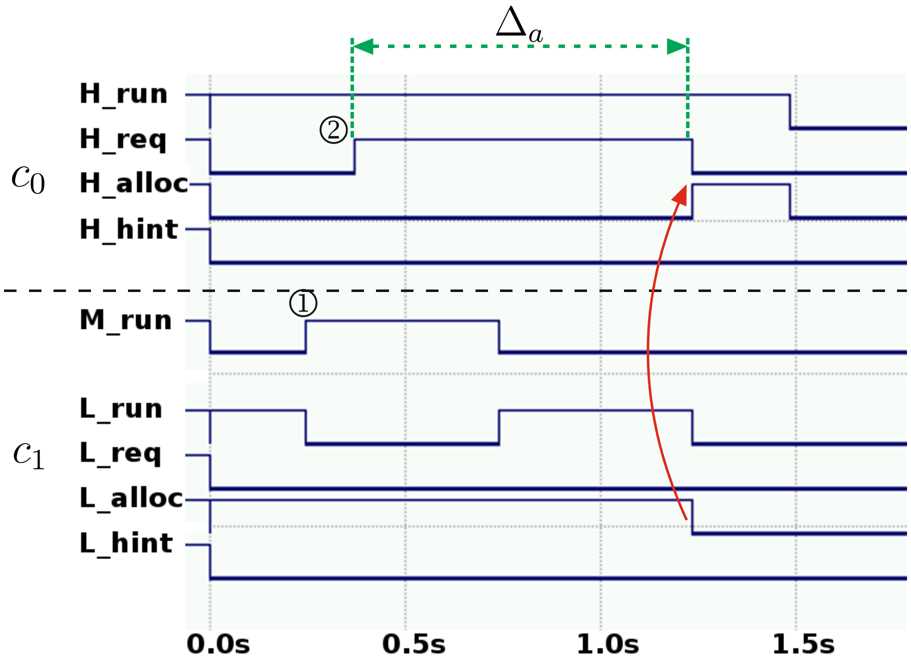
Communicate priority inversion to other cores:

- Set an OSEK event when cross-core priority inversion occurs
- Current owner (core) receives event, and raises the priority of the blocking task for the time while it holds the spinlock

→ Ideally implemented in kernel, proof of concept implemented at application level

Proposed Improvements

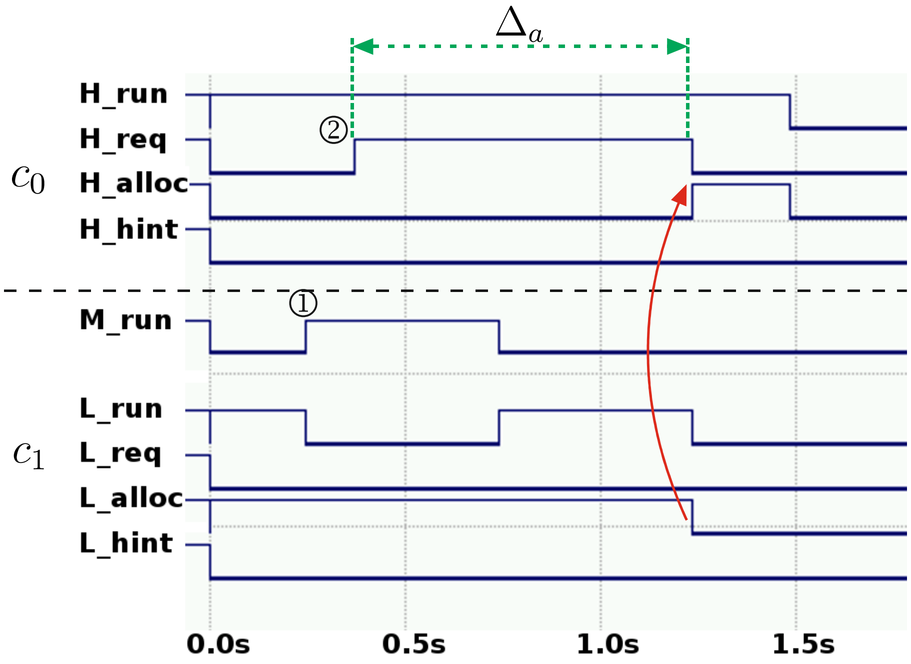
Handling Unbounded Priority Inversion



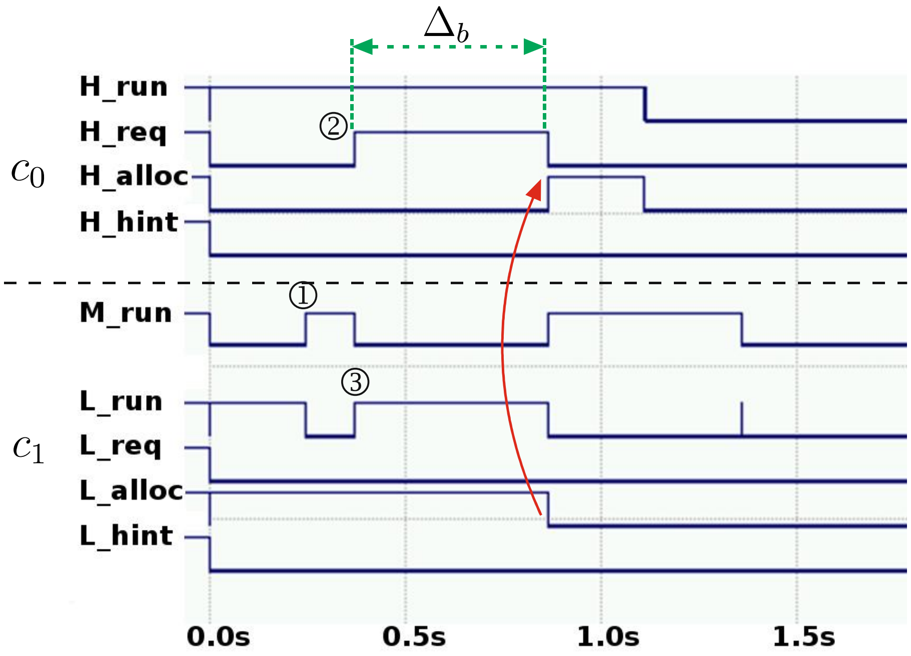
Unmodified ERIKA

Proposed Improvements

Handling Unbounded Priority Inversion



Unmodified ERIKA



Modified ERIKA

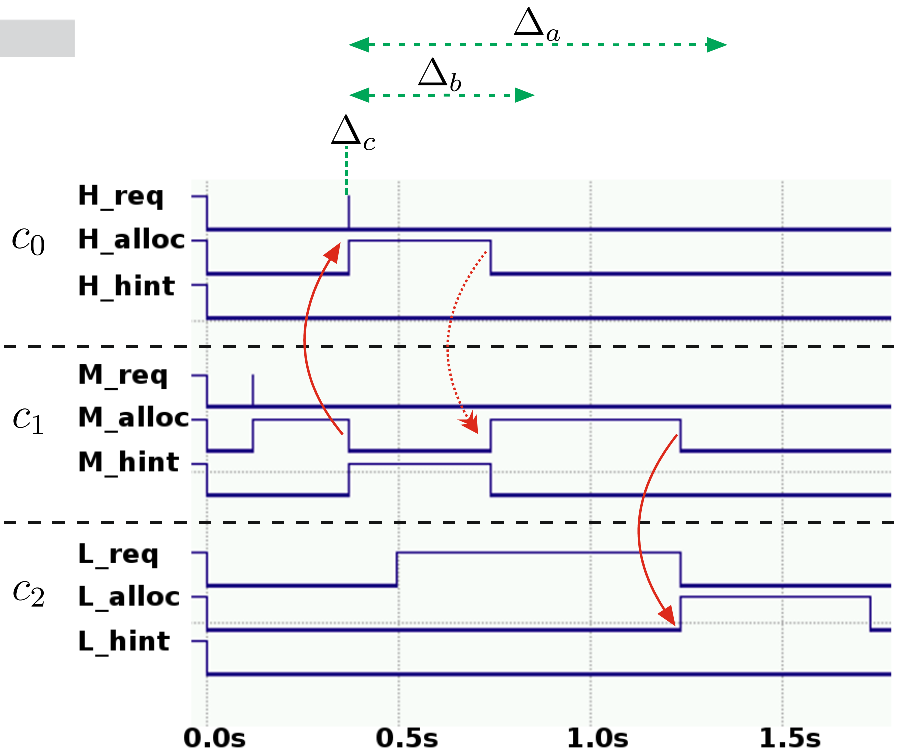
Proposed Improvements

DynamicHinting

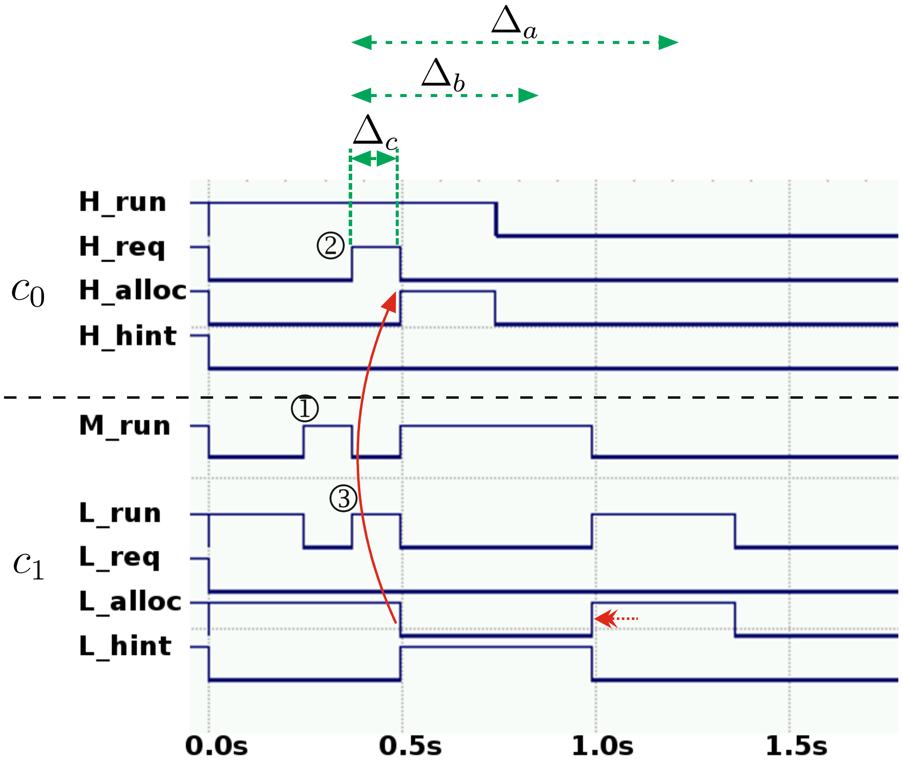
- Hints for spinlocks
- Hints received through explicit query
- In tests, tasks periodically ask the resource manager about their current hints
- In tests, tasks are always resource-collaborative

Proposed Improvements

DynamicHinting



Ex.: Allocation Order



Ex.: Unbounded Priority Inversion

Summary

AUTOSAR does not provide satisfactory resource management, specially for multi-core platforms.

Proposed improvements

- Removed unbounded priority inversion for spinlocks
- Defined priority-based allocation order for spinlocks
- Introduced DynamicHinting in AUTOSAR
- Reduced bounded priority inversion
- System kept deadlock free
- API unchanged

Outlook

- Modify kernel to support more flexible hint notification
- Unify AUTOSAR resources and spinlocks
- Schedulability analysis
- Dependability guarantees on real-time behavior
- Introduce DynamicHinting in other multi-core Oss
- Test in real application

Thank you!

Renata Martins Gomes
renata.gomes@tugraz.at

Graz University of Technology
Institute for Technical Informatics
Embedded Automotive Systems Group (EAS)