TECHNISCHE UNIVERSITÄT DRESDEN

cfaed CENTER FOR ADVANCING ELECTRONICS DRESDEN

Real-Time Systems
Mixed-Criticality Systems

Marcus Völp

Motivation

Source: NASA

Flight

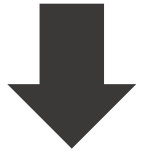
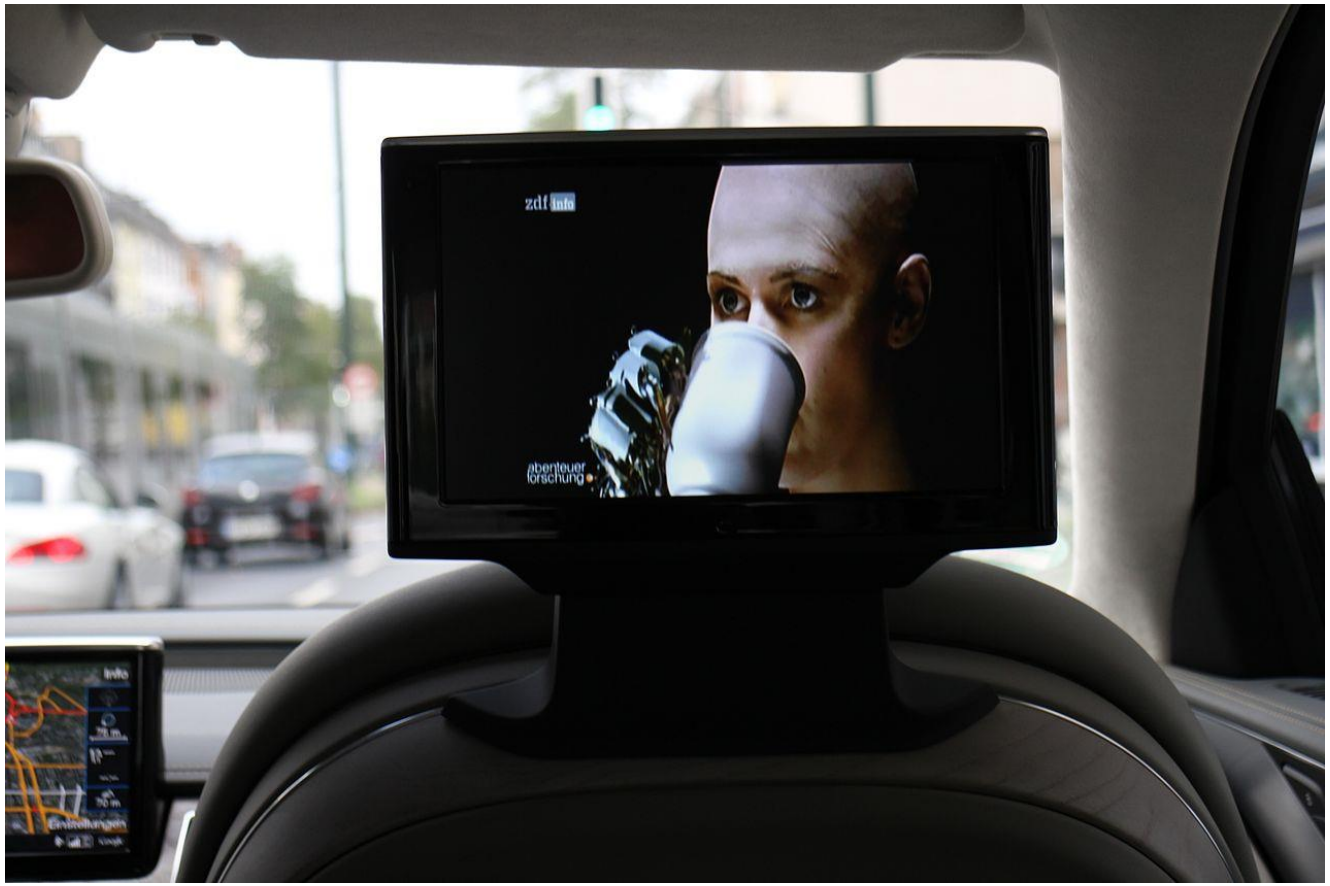


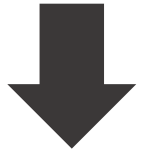
Image Processing

Motivation



Source: Magnus Manske (CC-BY 2.0)

Drive / Break



Car Entertainment

Motivation

DO 178c Certification

A: Catastrophic

B: Hazardous

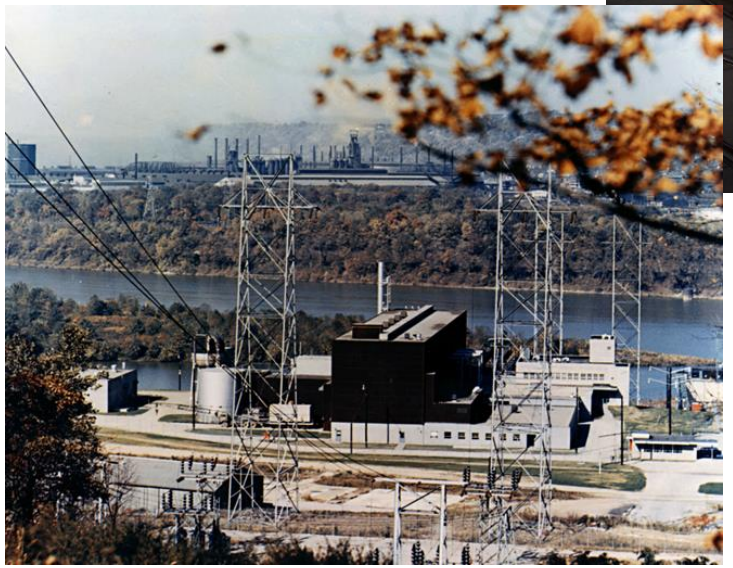
C: Major

D: Minor

E: No Safety Effect

Motivation

IEC EN 61508: Safety Certification



Probability for Faults per Hour
(continuous operation)

- SIL 4: 10^{-8} – 10^{-9} PFH
- SIL 3: 10^{-7} – 10^{-8} PFH
- SIL 2: 10^{-6} – 10^{-7} PFH
- SIL 1: 10^{-5} – 10^{-6} PFH

Motivation

How can we consolidate applications with differing certification requirements into a single system?

… to safe energy, weight, costs.
… without having to reevaluate the entire system.

Outline

Motivation

Mixed Criticality

Non-Optimality of RMS and EDF

Scheduling Algorithms

- *Criticality Monotonic*
- *OCBP*
- *EDF-VD*

Period Transformation

Resources

our MCS Research: *Flattening, QAS-MC, …*

Reference:

A. Burns, R. Davis "***Mixed-Criticality Systems – A Review***", 4th ed, July 2014

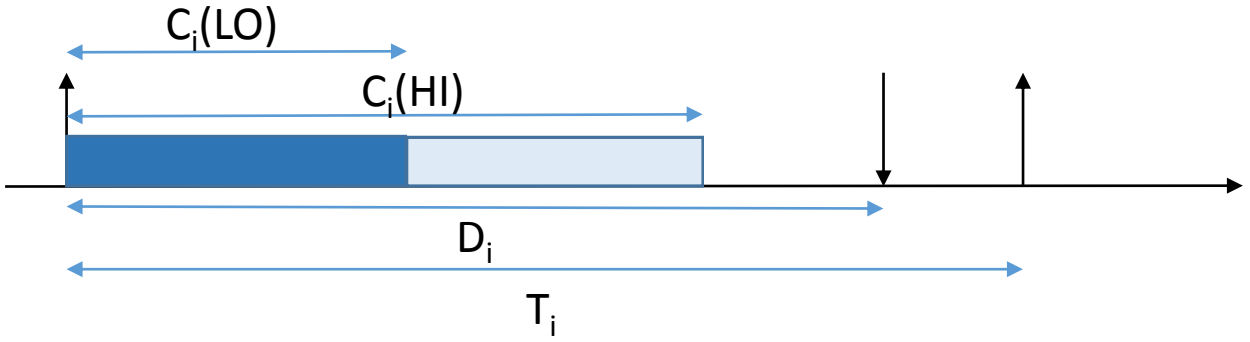
http://www-users.cs.york.ac.uk/burns/review.pdf

Mixed Criticality

How can we consolidate applications with differing certification requirements into a single computing system to safe energy, weight and costs?

Steve Vestal

1. Assign each task a criticality level l_i (e.g., SIL 1-4 / Arinc A-E / LO, HI).
2. Estimate task parameters according to the requirements of each level.*
3. Make sure a higher criticality task can meet the guarantees of the next highest level if it fails on a low level.



S. Vestal. **"Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance."**
IEEE *Real-Time Systems Symposium* (RTSS), pages 239–243, 2007.

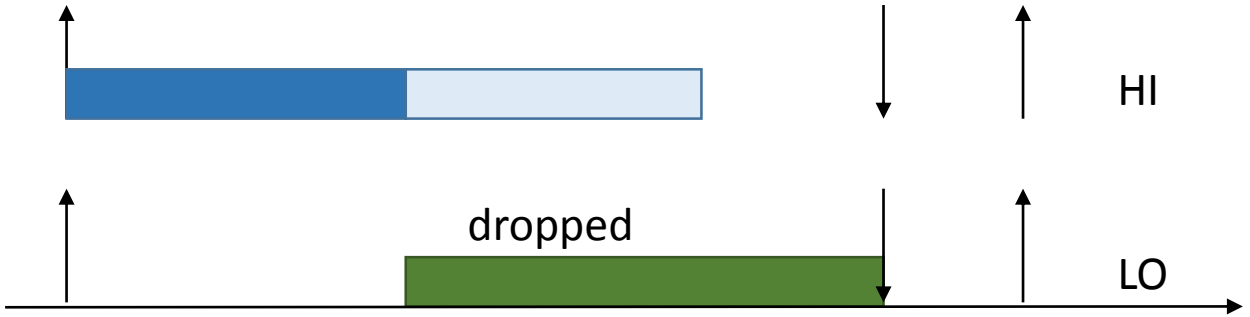
Mixed Criticality

How can we consolidate applications with differing certification requirements into a single computing system to safe energy, weight and costs?

Steve Vestal

Mixed-Criticality Guarantees:

Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



S. Vestal. **"Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance."**
IEEE *Real-Time Systems Symposium* (RTSS), pages 239–243, 2007.

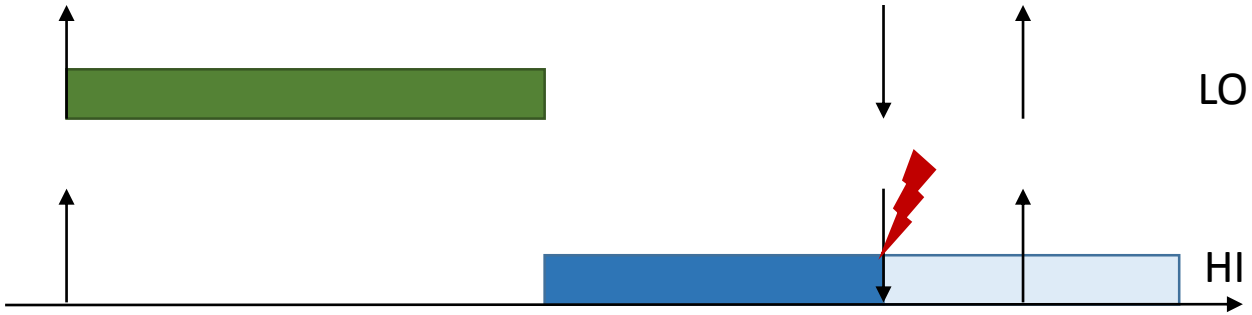
Mixed Criticality

How can we consolidate applications with differing certification requirements into a single computing system to safe energy, weight and costs?

Steve Vestal

Mixed-Criticality Guarantees:

Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



S. Vestal. **"Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance."**
IEEE *Real-Time Systems Symposium* (RTSS), pages 239–243, 2007.

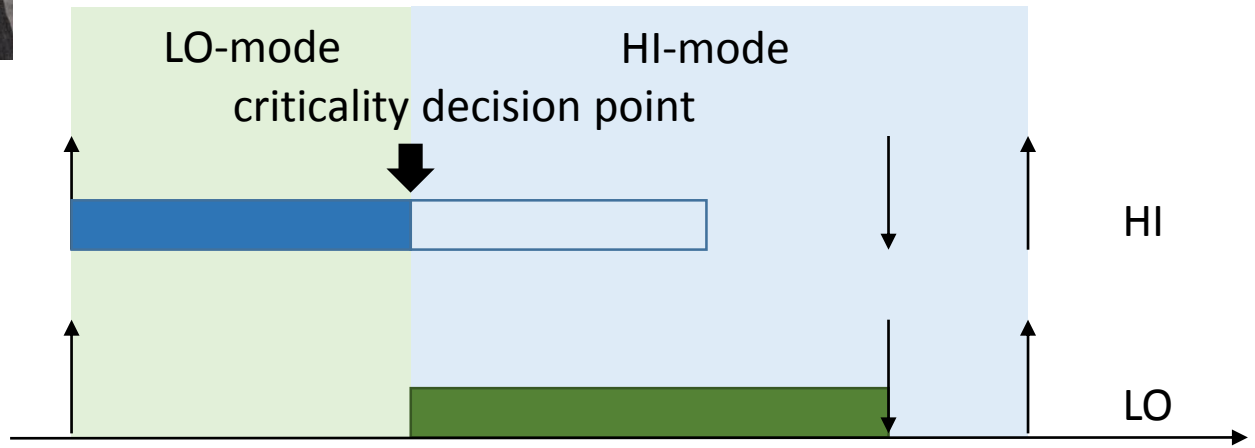
Mixed Criticality

How can we consolidate applications with differing certification requirements into a single computing system to safe energy, weight and costs?

Steve Vestal

Mixed-Criticality Guarantees / Feasibility:

Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.

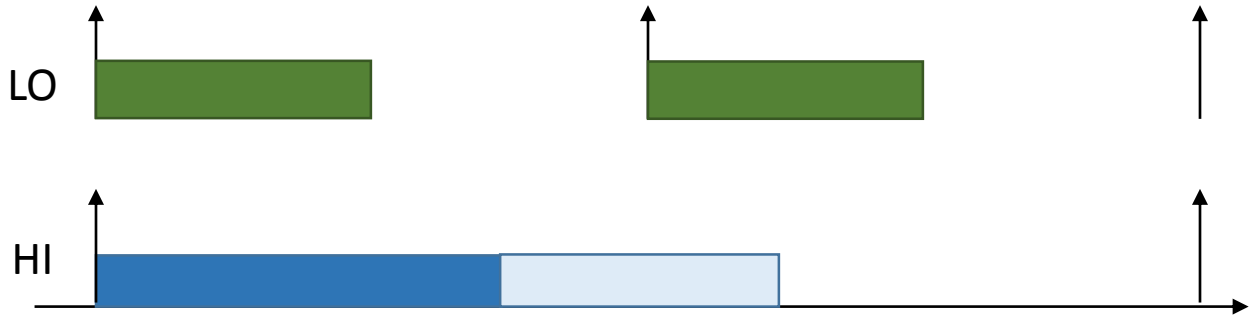


S. Vestal. **"Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance."** IEEE *Real-Time Systems Symposium* (RTSS), pages 239–243, 2007.

Mixed Criticality Scheduling

RMS is not optimal:

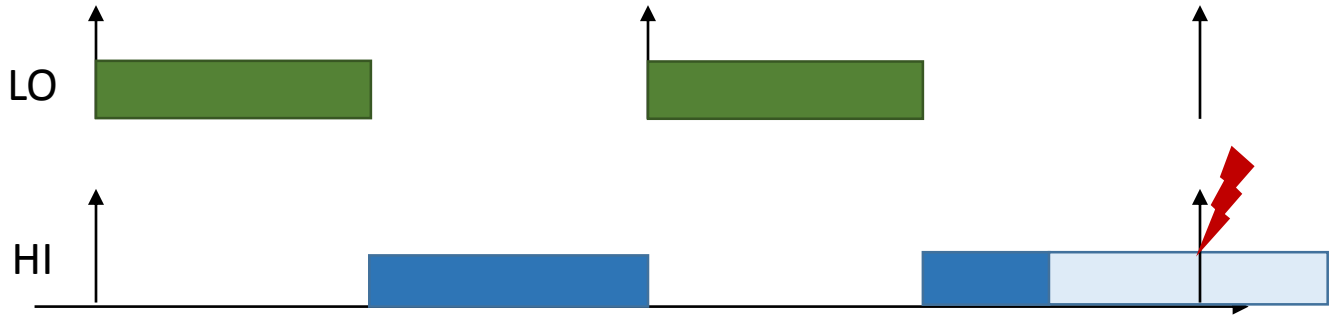
Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



Mixed Criticality Scheduling

RMS is not optimal:

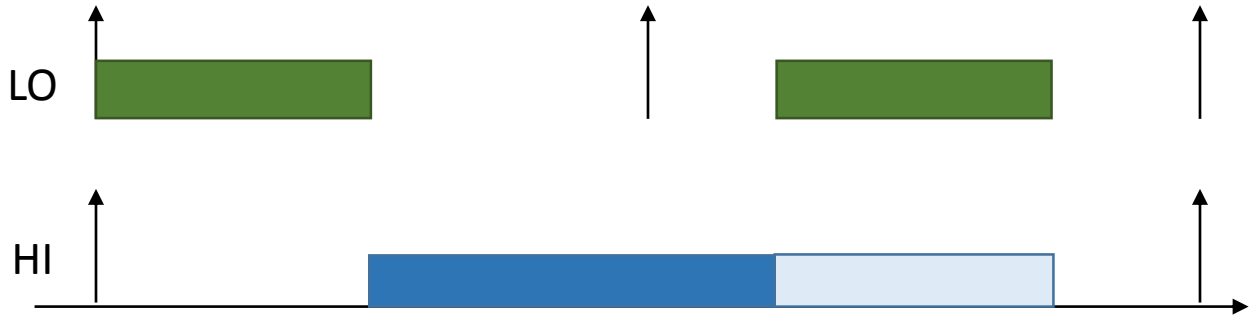
Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



Mixed Criticality Scheduling

RMS is not optimal:

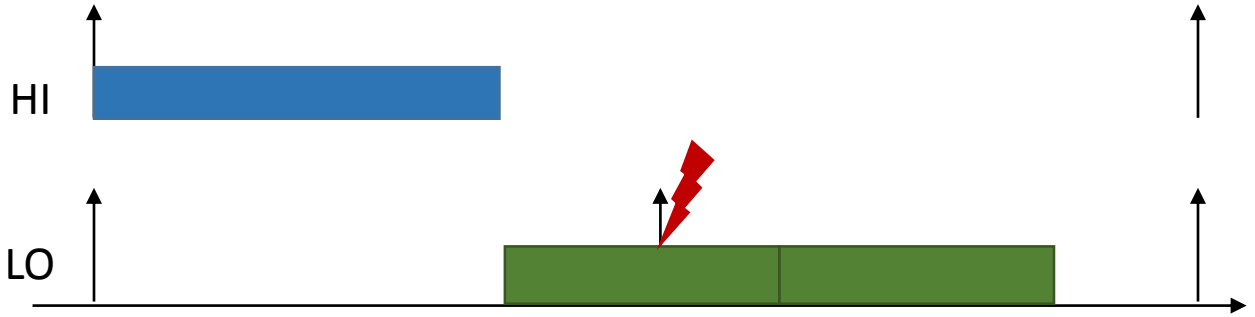
Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



Mixed Criticality Scheduling

RMS is not optimal:

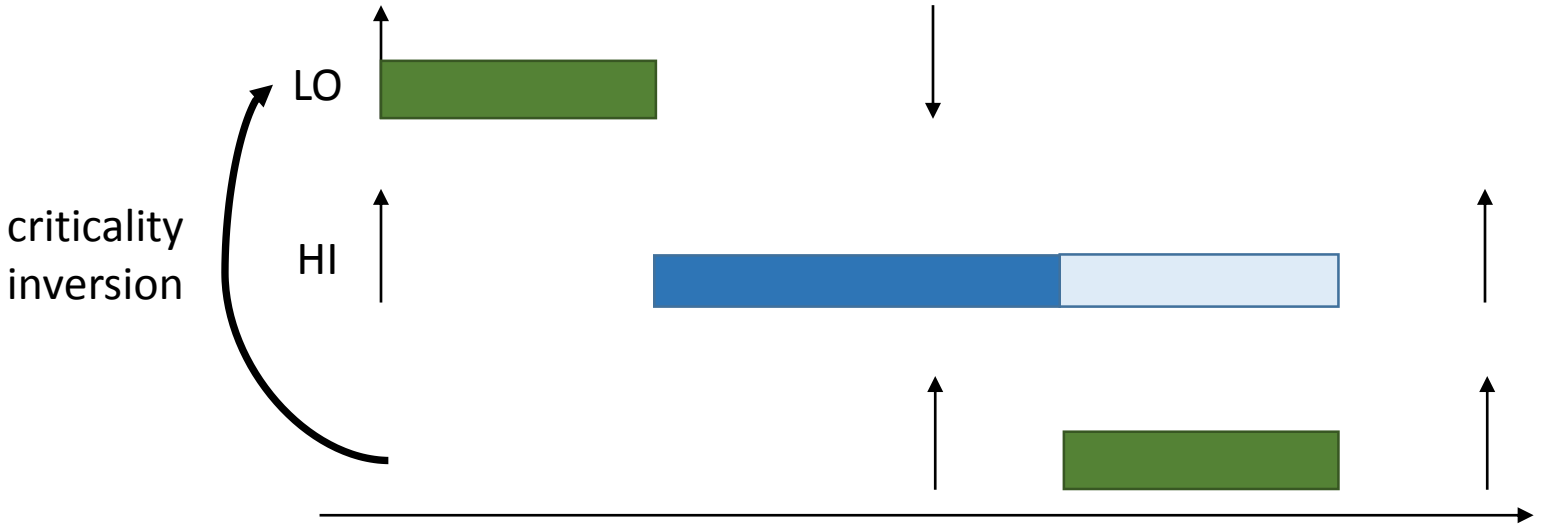
Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



Mixed Criticality Scheduling

RMS is not optimal:

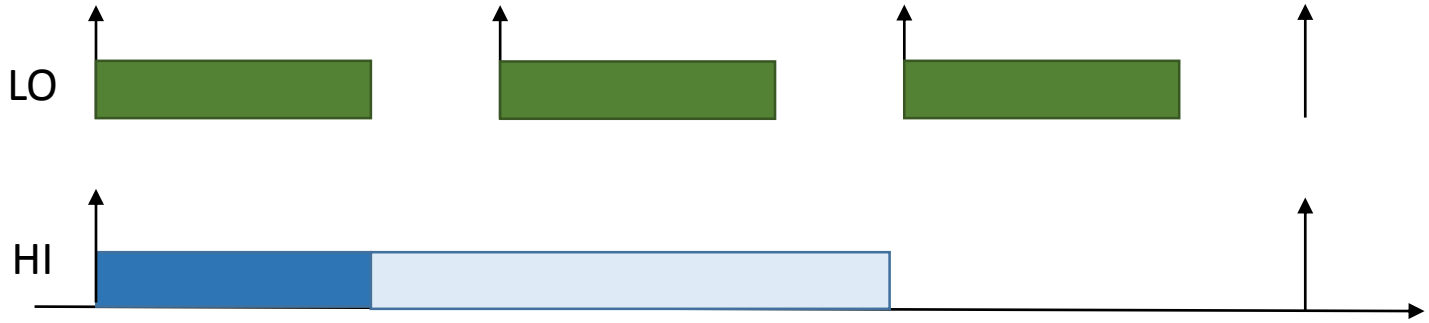
Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



Mixed Criticality Scheduling

EDF is not optimal:

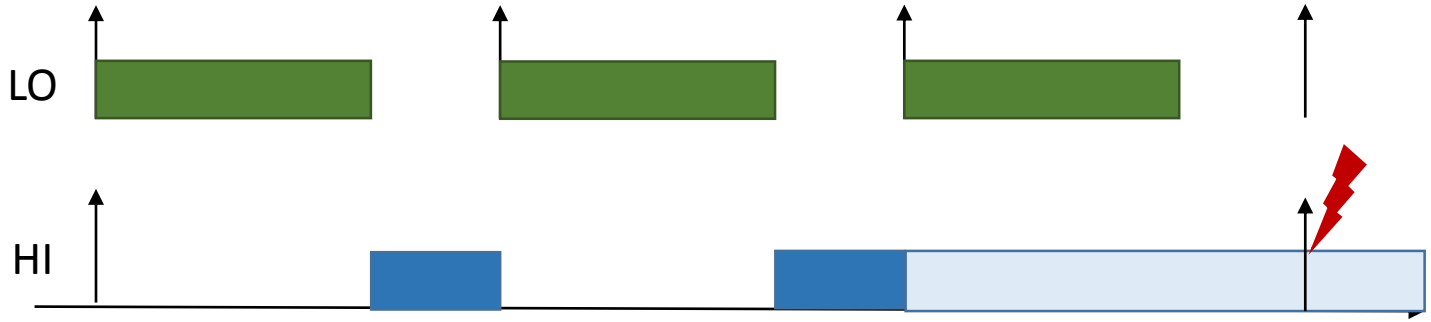
Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



Mixed Criticality Scheduling

EDF is not optimal:

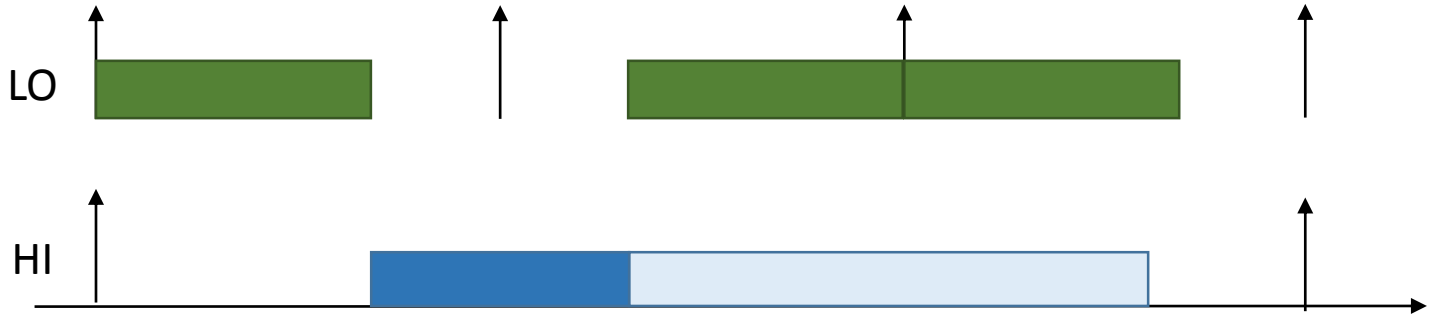
Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



Mixed Criticality Scheduling

EDF is not optimal:

Each job rejeives $C_i(l_i)$ time in between its release $r_{i,j}$ and its deadline $r_{i,j} + D_i$, provided all jobs of higher criticality tasks τ_h complete within $C_h(l_i)$.



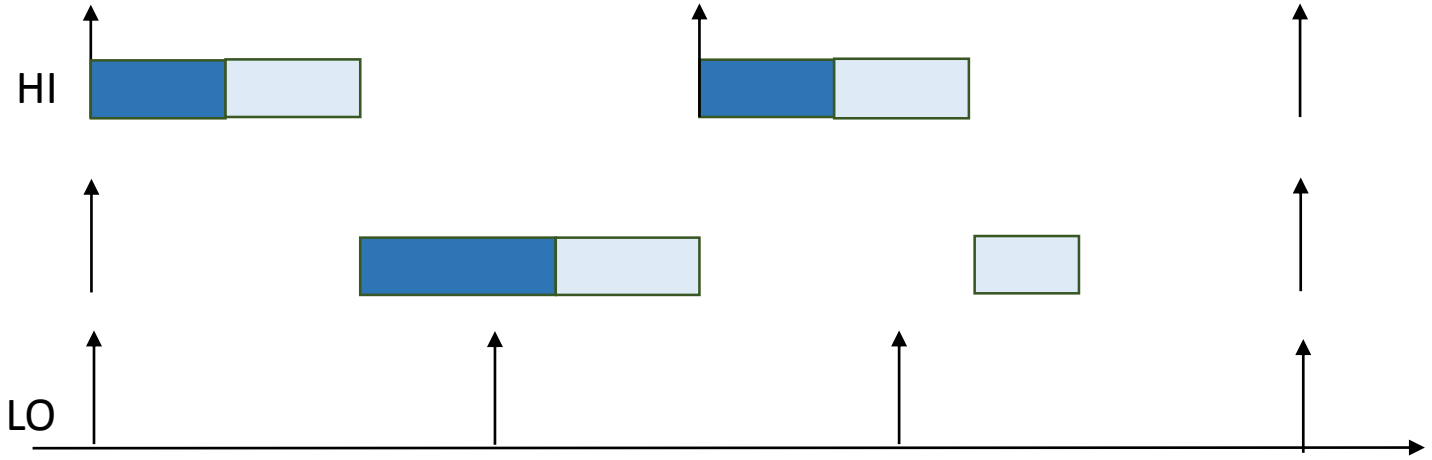
Mixed Criticality Scheduling is NP-Hard

(Proof: see S.K. Baruah. Mixed criticality schedulability analysis is highly intractable. Technical report, University of North Carolina at Chapel Hill, 2009)

Mixed Criticality Scheduling

Criticality Monotonic Scheduling:

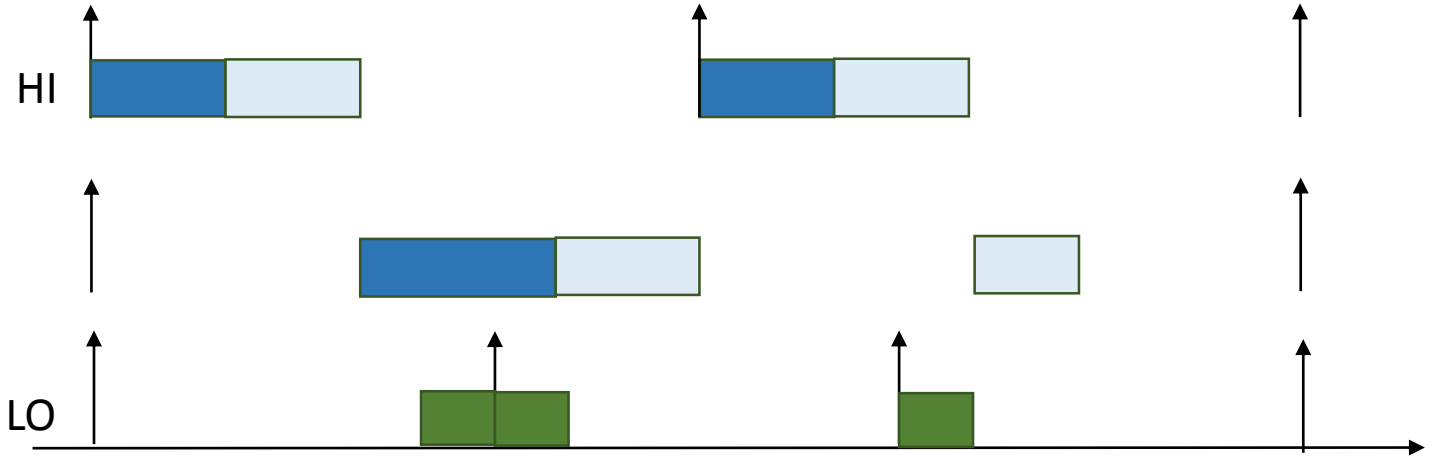
- family of scheduling algorithms
- higher criticality tasks receive strictly higher priorities
- apply standard algorithm within priority band



Mixed Criticality Scheduling

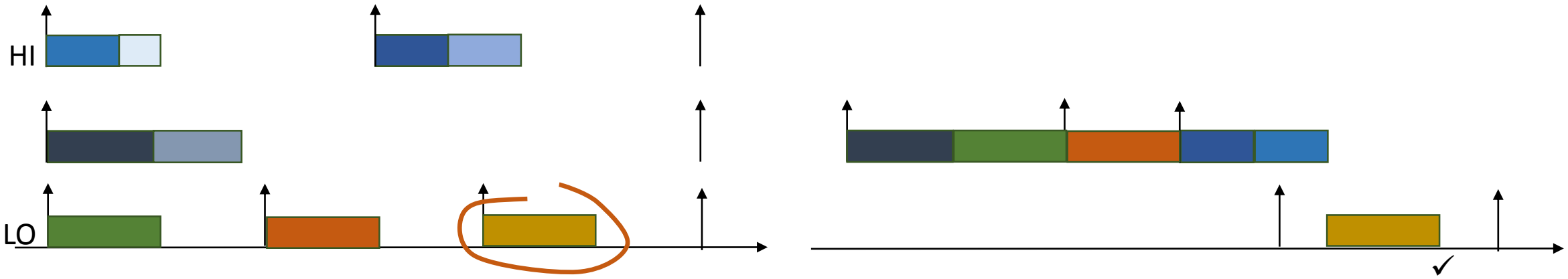
Criticality Monotonic Scheduling:

- family of scheduling algorithms
- higher criticality tasks receive strictly higher priorities
- apply standard algorithm within priority band



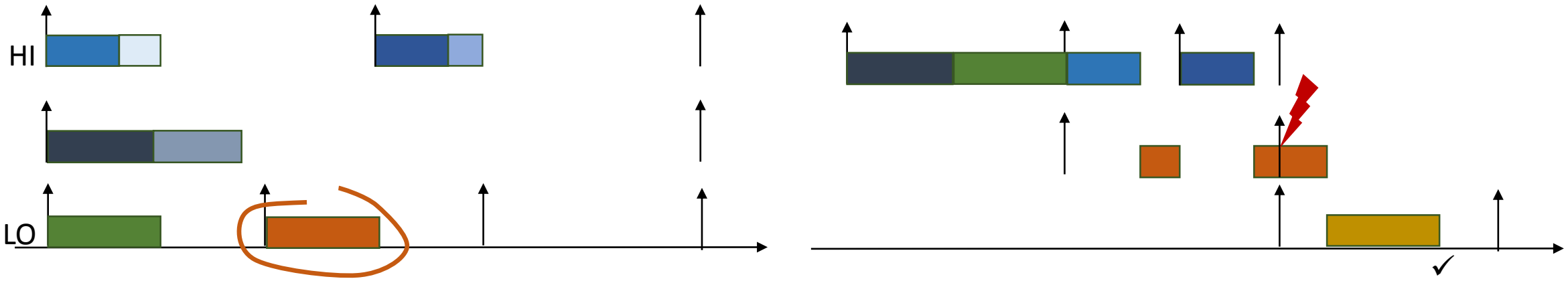
Own Criticality Based Scheduling (OCBP)

- Based on Audsley‘s Algorithm:
  1. pick job J_i
  2. check whether J_i can run at lowest priority, assuming $C_k(l_i)$ for all other jobs J_k (ignoring D_i)
  3. if ok, remove J_i from set of jobs and goto 1.



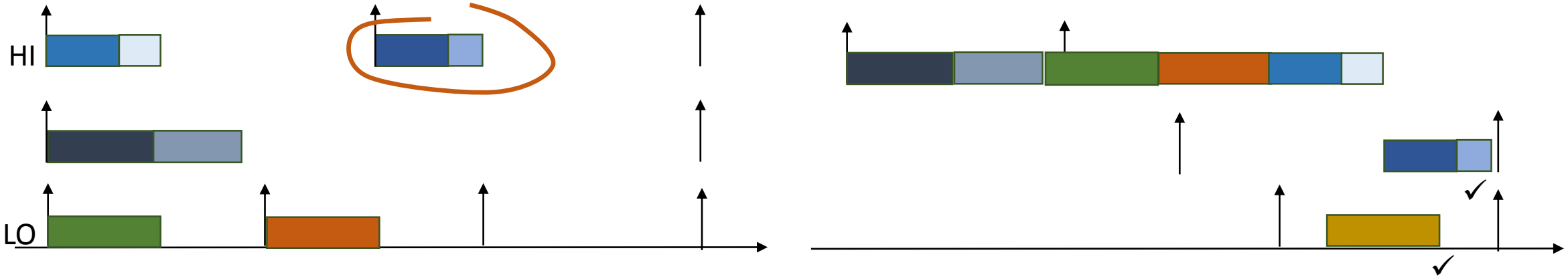
Own Criticality Based Scheduling (OCBP)

- Based on Audsley's Algorithm:
  1. pick job J_i
  2. check whether J_i can run at lowest priority, assuming $C_k(l_i)$ for all other jobs J_k (ignoring D_i)
  3. if ok, remove J_i from set of jobs and goto 1.



Own Criticality Based Scheduling (OCBP)

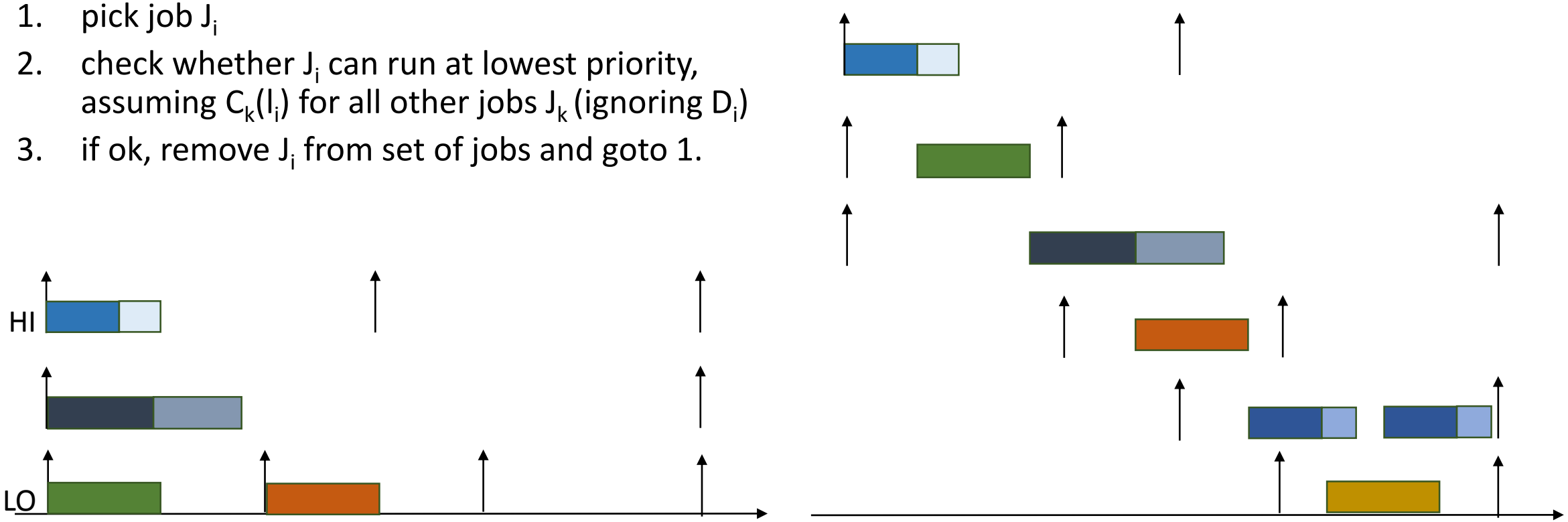
- Based on Audsley‘s Algorithm:
  1. pick job J_i
  2. check whether J_i can run at lowest priority, assuming $C_k(l_i)$ for all other jobs J_k (ignoring D_i)
  3. if ok, remove J_i from set of jobs and goto 1.



Mixed Criticality Scheduling

Own Criticality Based Scheduling (OCBP)

- Based on Audsley‘s Algorithm:
  1. pick job J_i
  2. check whether J_i can run at lowest priority, assuming $C_k(l_i)$ for all other jobs J_k (ignoring D_i)
  3. if ok, remove J_i from set of jobs and goto 1.



Best algorithm (in terms of speedup) in the class of single fixed priority per job.

Mixed Criticality Scheduling

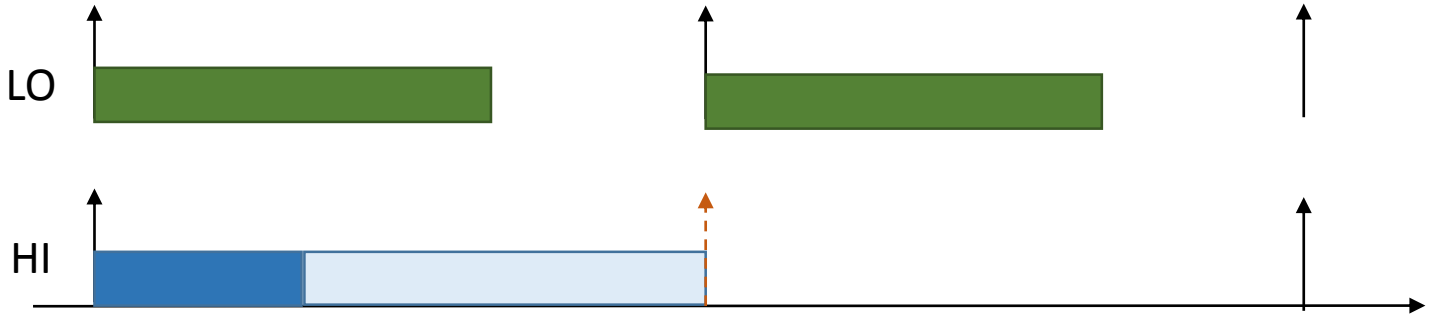
Earliest Deadline First – Virtual Deadlines

- Assign multiple priorities to each job (one per mode)
- Schedule higher criticality tasks with earlier deadline to make room for low tasks

$$U_i(k) = \sum_{j \in \{1,\ldots,n \mid l_j = i\}} \frac{C_j(k)}{T_j}$$

$Case\ 1$: $U_{LO}(LO) + U_{HI}(HI) \leq 1$
apply EDF

$Case\ 2$: $U_{LO}(LO) + \frac{U_{HI}(LO)}{1 - U_{HI}(HI)} \leq 1$
schedule HI tasks in LO mode with relative deadlines $\frac{U_{HI}(LO)}{1 - U_{LO}(LO)} D_i$



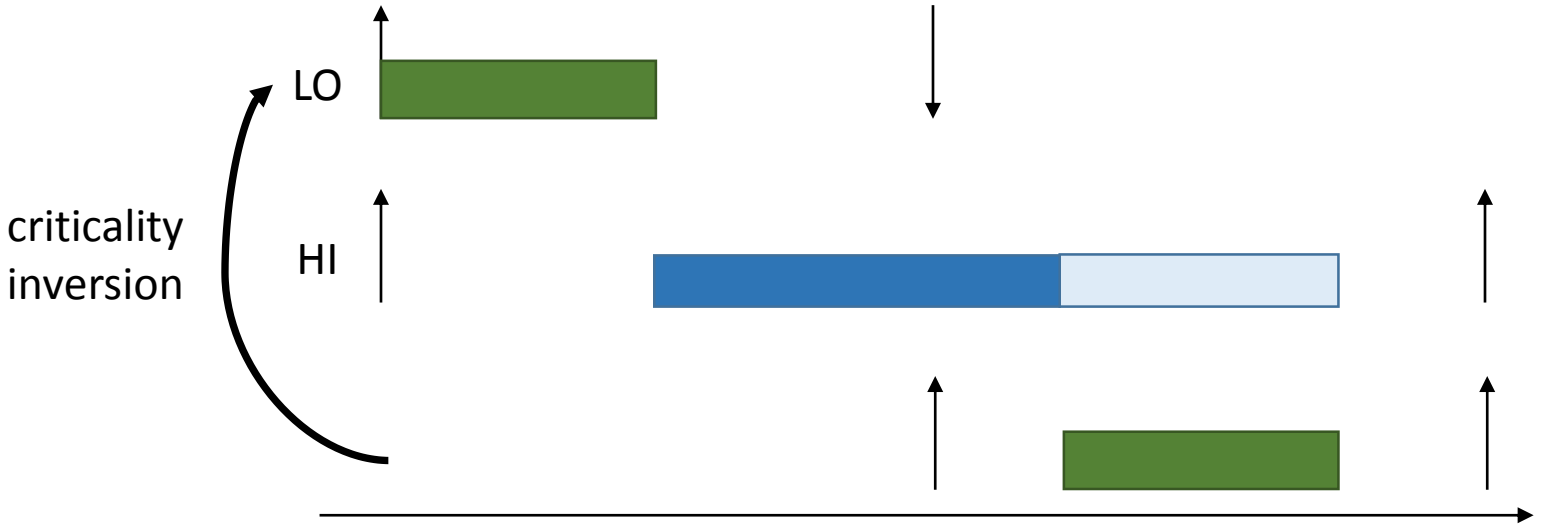
$\tau_{LO} = (LO, 3, (2))$ $U_{LO}(LO) = \frac{2}{3} = \frac{4}{6}$

$\tau_{HI} = (HI, 6, (1, 3))\, U_{HI}(LO) = \frac{1}{6}$ $U_{HI}(HI) = \frac{3}{6}$ $D'_{HI} = \frac{\frac{1}{6}}{\frac{2}{6}} 6 = 3$

Mixed Criticality Scheduling

Period Transformation

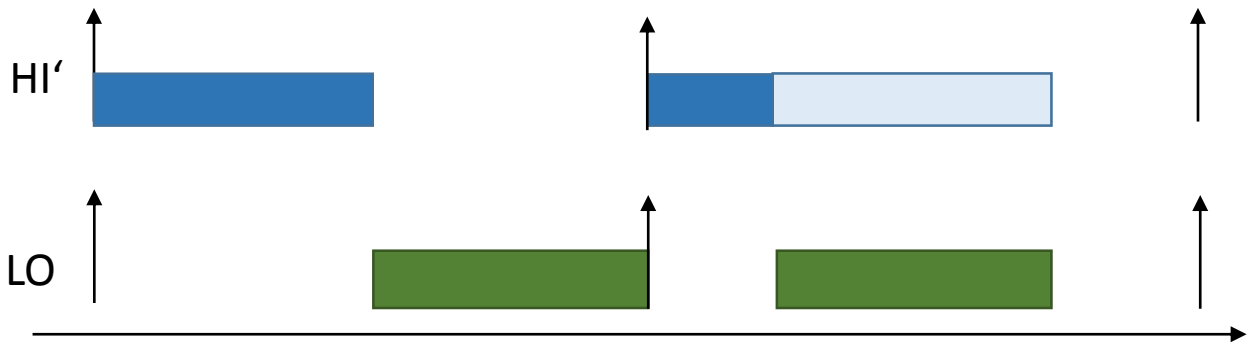
- split HI tasks into smaller ones to avoid criticality inversion



Mixed Criticality Scheduling

Period Transformation

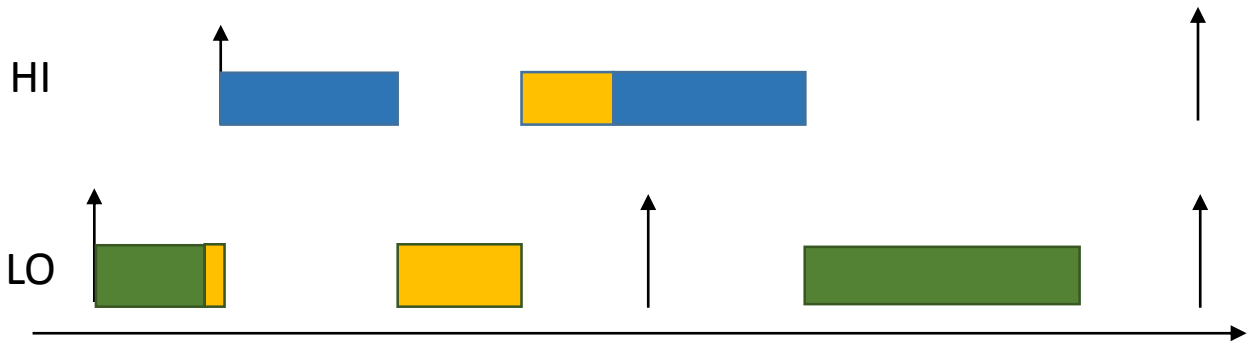
- split HI tasks into smaller ones to avoid criticality inversion



Resources

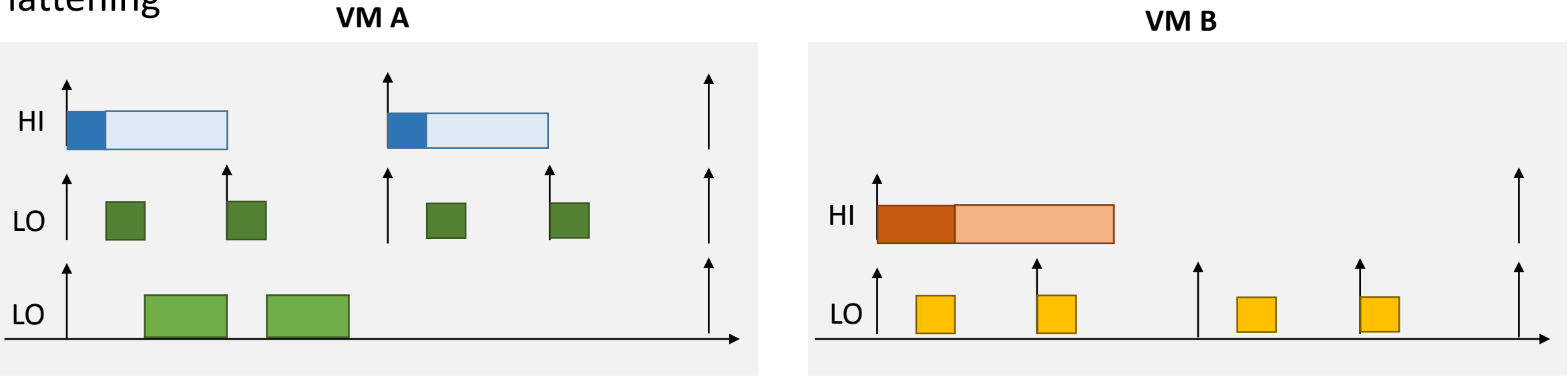
Facilitate sharing across criticality levels

- non-preemptible resources affect higher criticality levels even if the resource is not shared
- worst-case resource access time WCRT = max ($WCRT_k$)
  => evaluate LO resource access as if they were HI



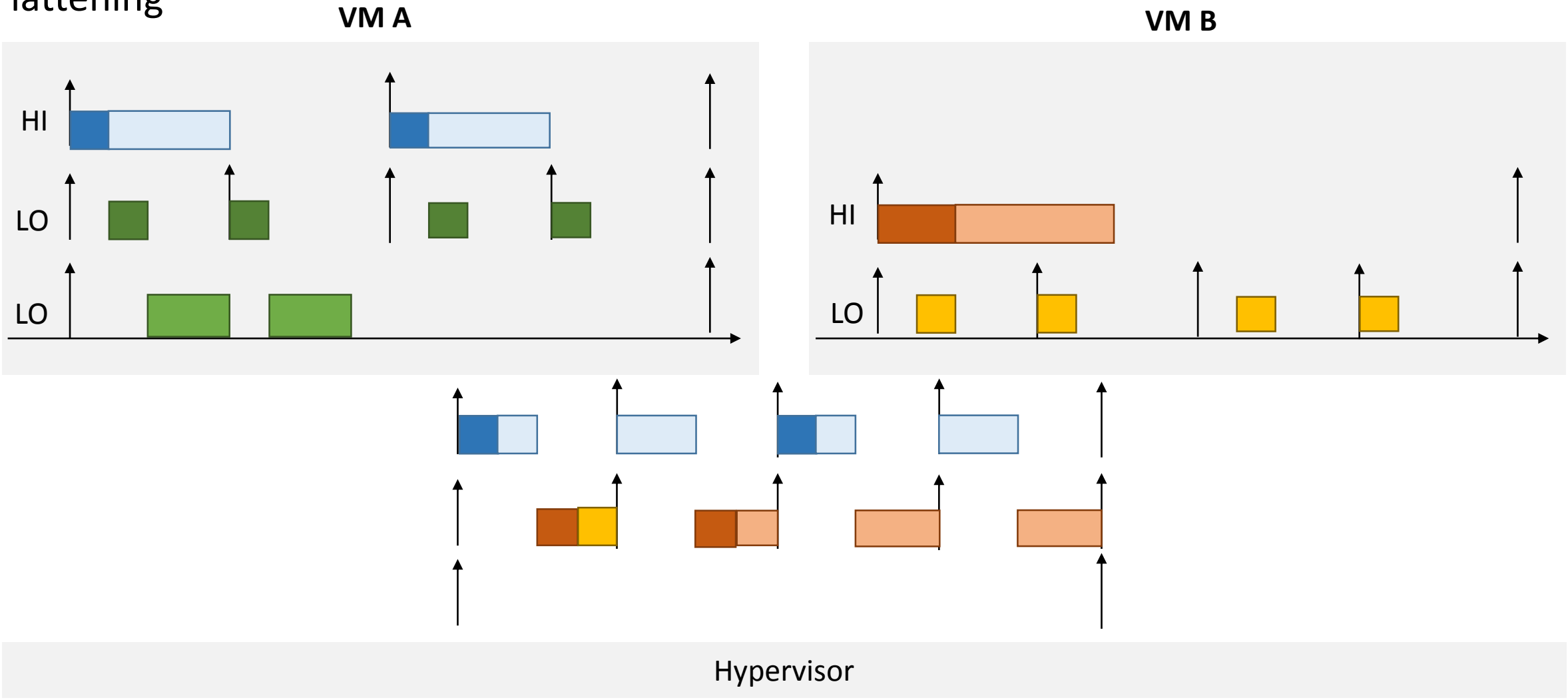
Our MCS Research

Flattening



Our MCS Research

Flattening



Our MCS Research

Flattening