Integration of Mixed-Criticality CPS with Criticality Layers

AVICPS 2012

[1]Dionisio de Niz and [2]Anthony Rowe

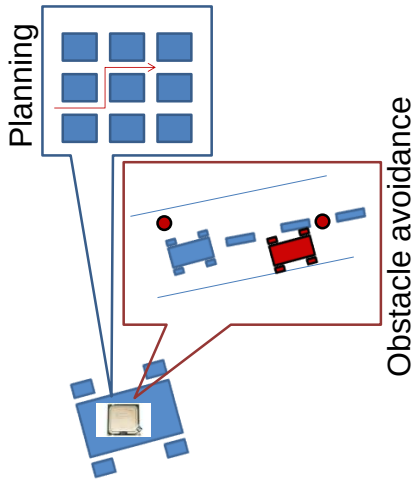
[1]Software Engineering Institute
[2]Electrical & Computer Engineering

Carnegie Mellon University

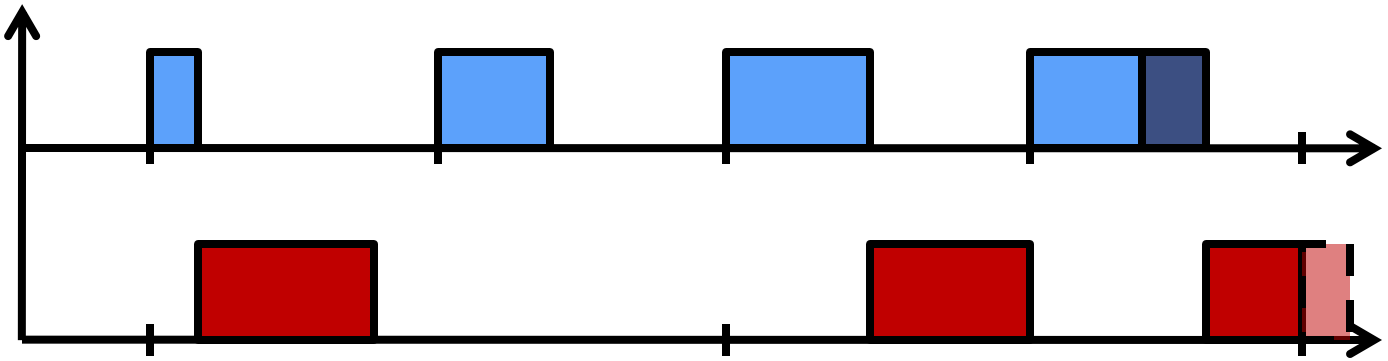
This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM-0000097

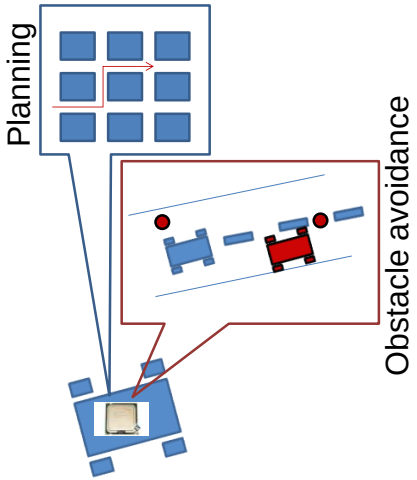
Integration of Mixed-Criticality Tasks



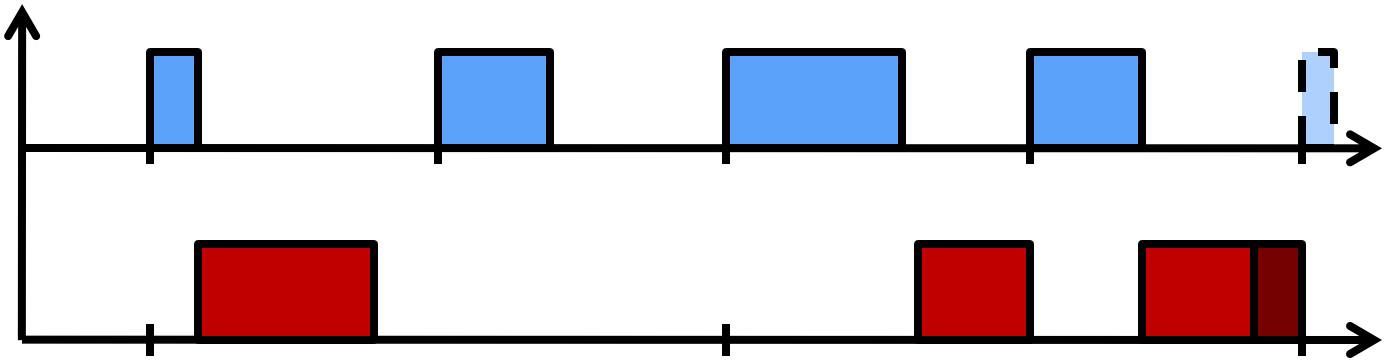
Shared Hardware
Can lead to cycle stealing



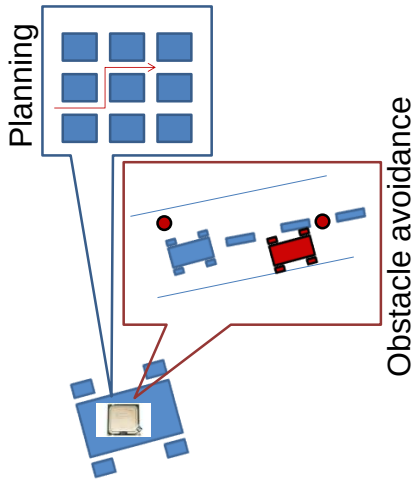
Integration of Mixed-Criticality Tasks



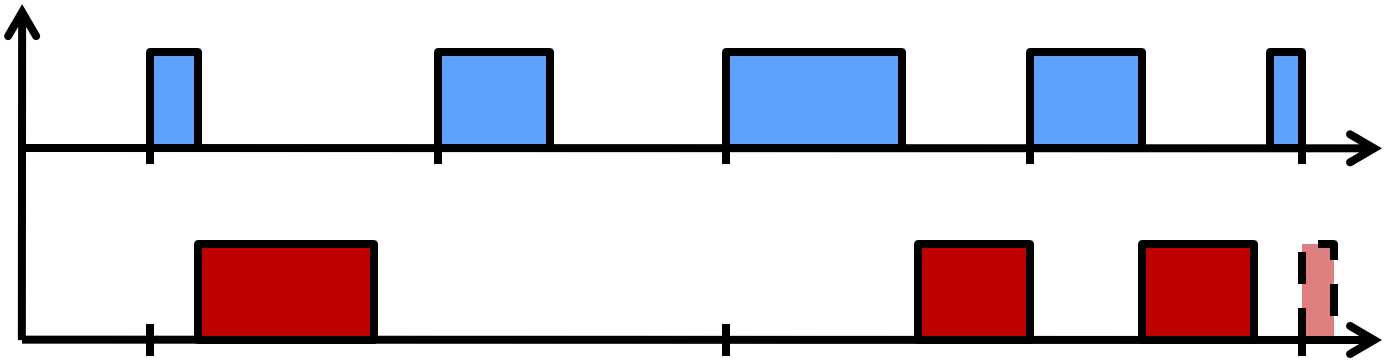
To avoid interference
add temporal protection



Integration of Mixed-Criticality Tasks



BUT
Symmetric protection[1]
leads to *criticality inversion*



[1]ARINC 653

Zero-Slack Scheduling

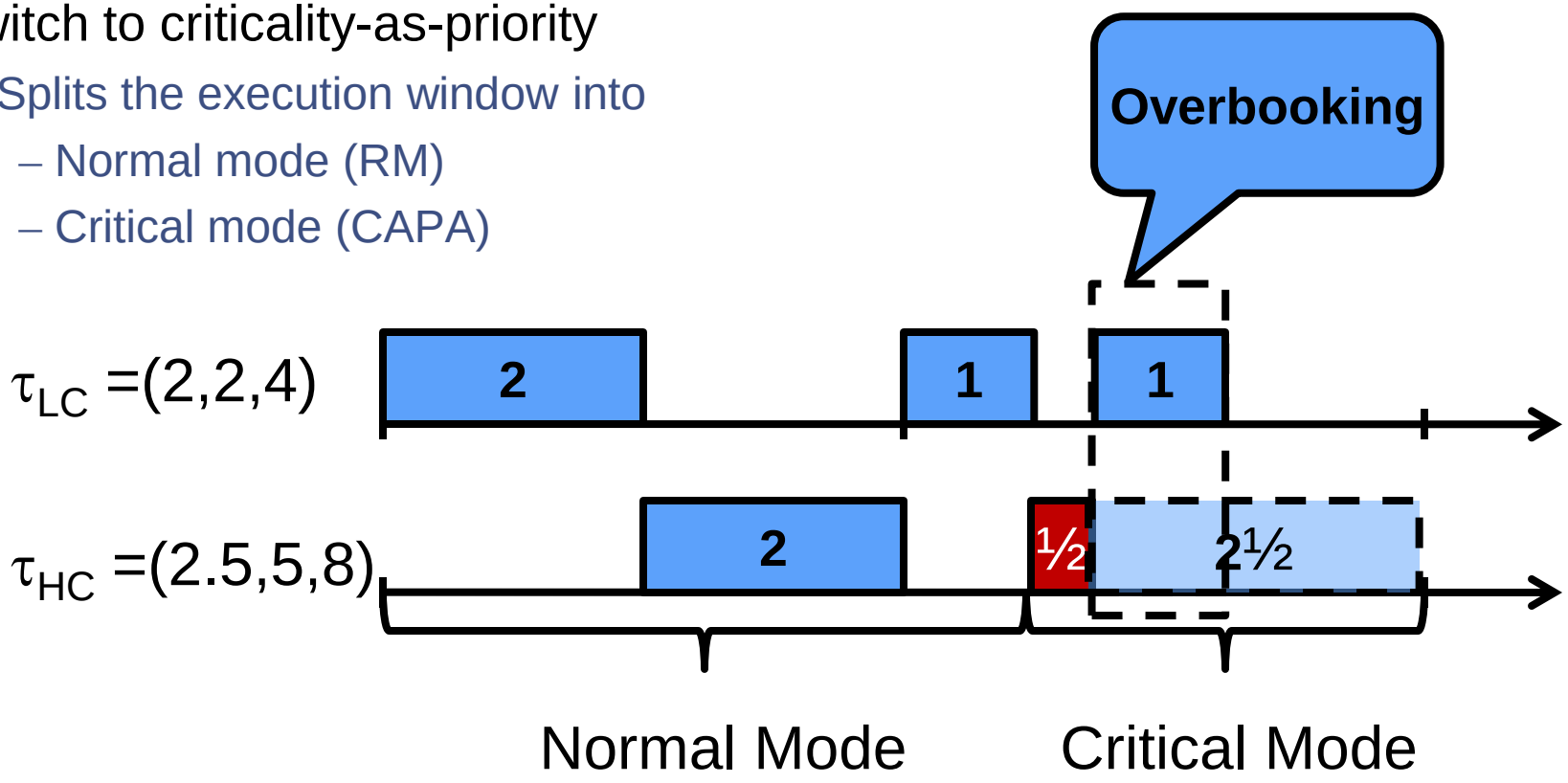
Start with RM

Calculate the last instant before τ_{HC} misses its deadline

- this is called the ***zero-slack*** instant

Switch to criticality-as-priority

- Splits the execution window into
  - Normal mode (RM)
  - Critical mode (CAPA)



Guarantee

A task is guaranteed to execute for C^o
if no higher-criticality task overloads

Layered guarantee

Engineer:

- No task overloads => all task meet their deadlines (Internal testing)

Certification authority

- A task is guarantee to execute for C^o at its criticality level (safety margin)
  - Disregarding lower-criticality tasks
  - Assuming no higher-criticality task overloads

Mission

$$Value = (\wedge_i Safety\_Critical_i) * \sum_j Mission\_Critical_j$$

0

true

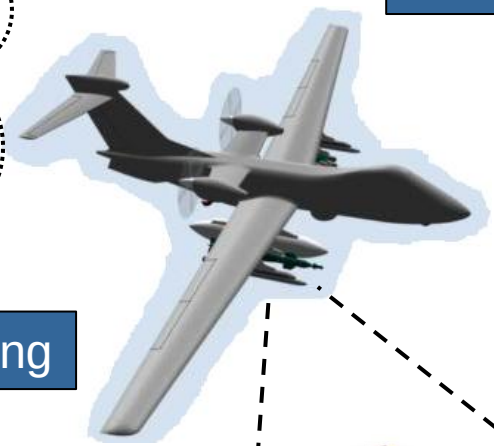
0

Safety-Critical
(Prevent Damage)

Flight Control

Collision Avoidance

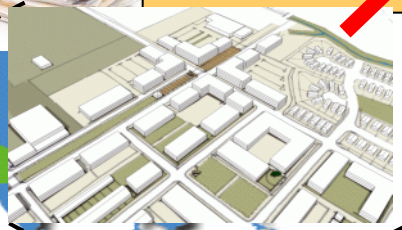
Route Planning



Mission-Critical
(Provide Value)

Amount of Intelligence

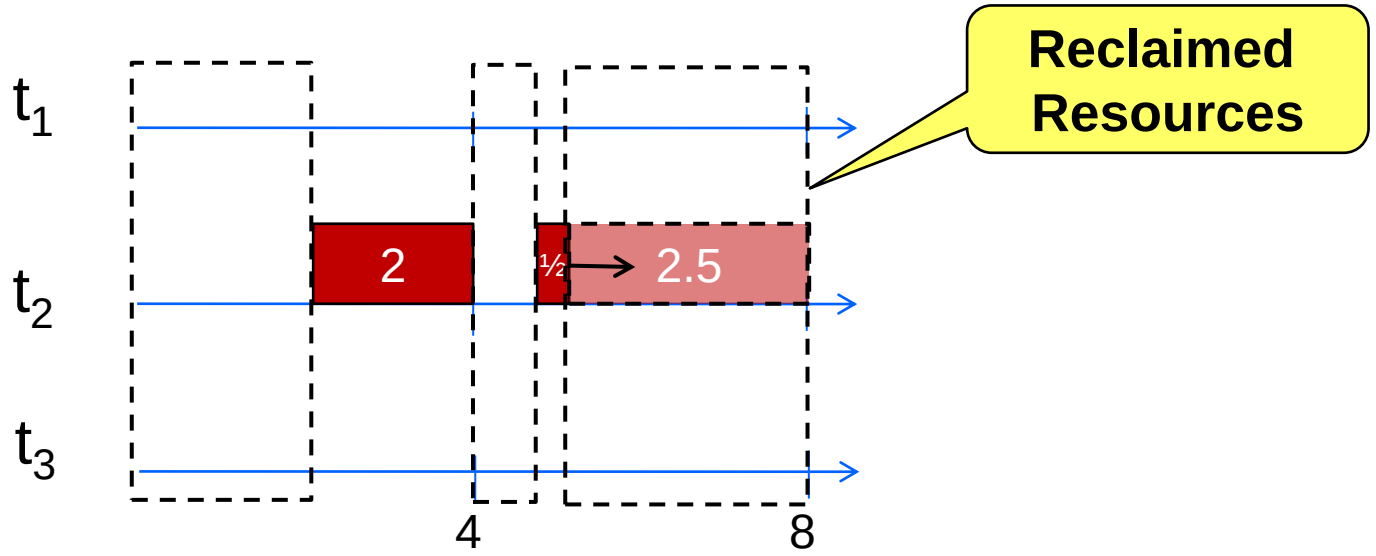
Takeoff-to-Surveillance Distance



Surveillance Coverage

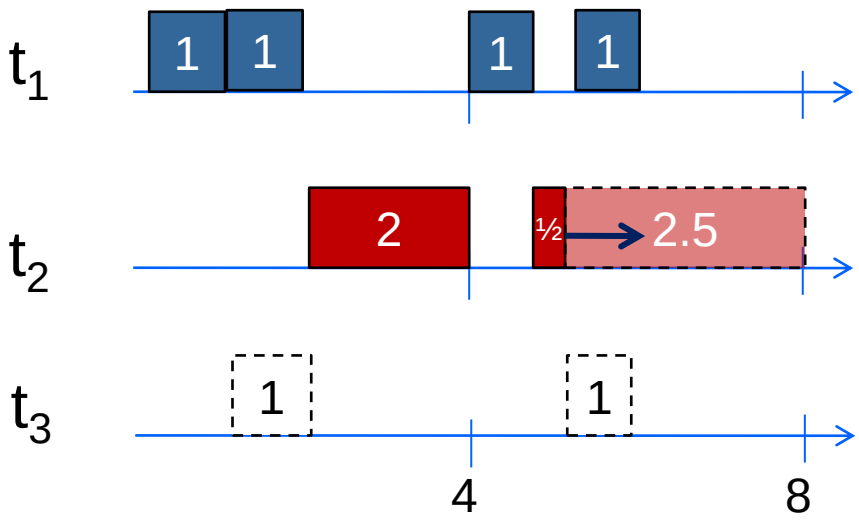
Reclaiming Resources in Mixed-Criticality Systems

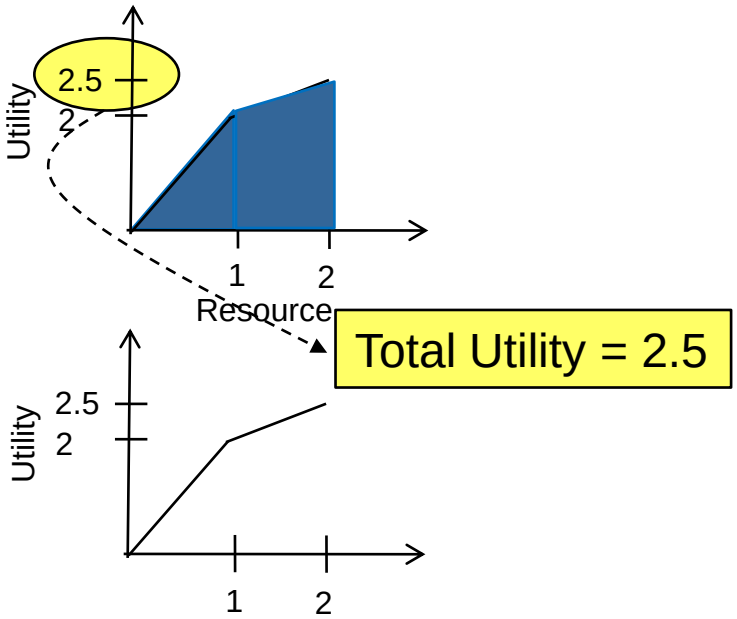
Task	Period	Criticality	WCET	NCET	Utility
t_1 Surveillance Cov.	4	Mission	2	2	{2,2.5}
t_2 Collision Avoid.	8	Safety	5	2.5	
t_3 Amount of Intelligence	4	Mission	2	2	{2,2.5}



Using Reclaimed Resources to Maximized Utility

Task	Period	Criticality	WCET	NCET	Utility Levels
t_1 Surveillance Cov.	4	Mission	2	2	{2,2.5}
t_2 Collision Avoid.	8	Safety	5	2.5	
t_3 Amount of Intelligence	4	Mission	2	2	{2,2.5}

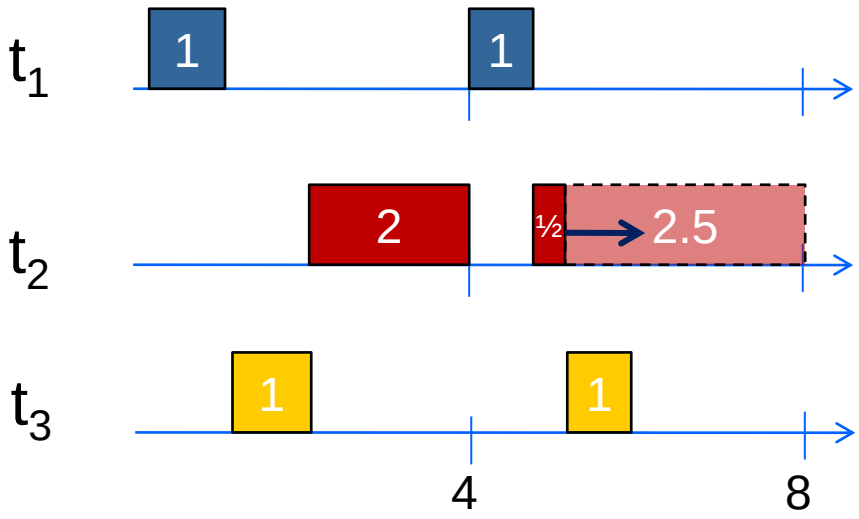


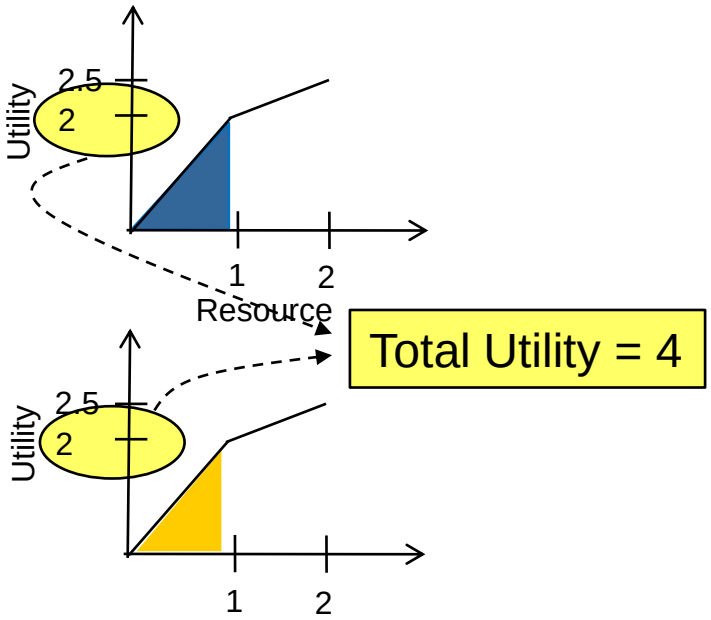


Utility Diminishes: Utility ≠ Criticality

Using Reclaimed Resources to Maximized Utility

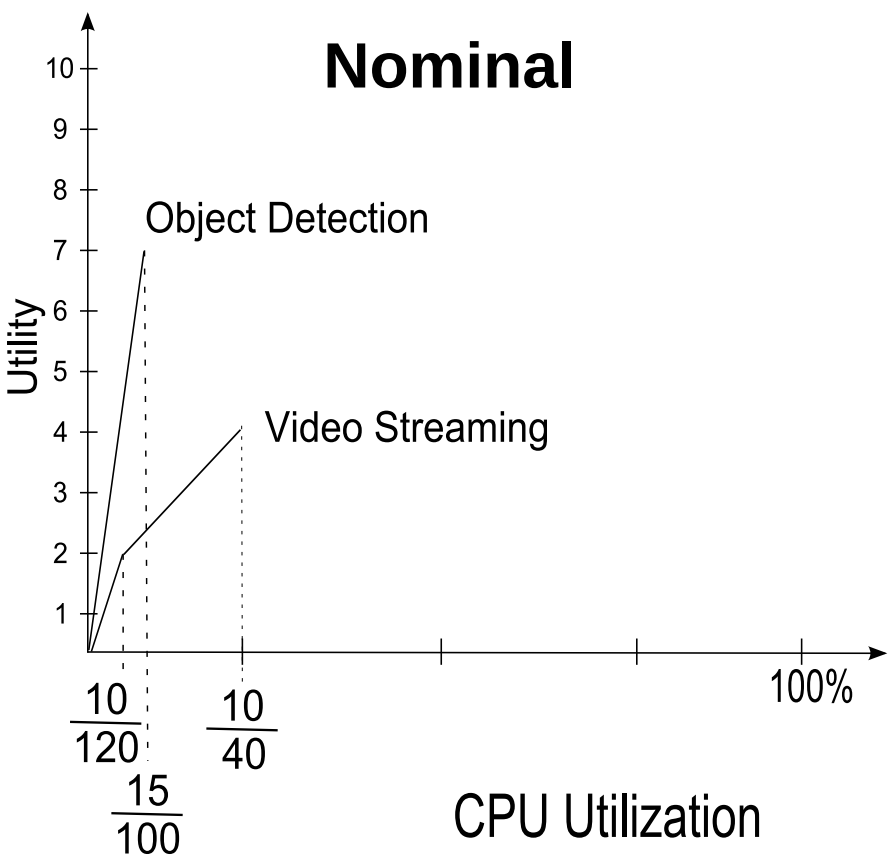
Task	Period	Criticality	WCET	NCET	Utility Levels
t_1 Surveillance Cov.	4	Mission	2	2	{2,2.5}
t_2 Collision Avoid.	8	Safety	5	2.5	
t_3 Amount of Intelligence	4	Mission	2	2	{2,2.5}

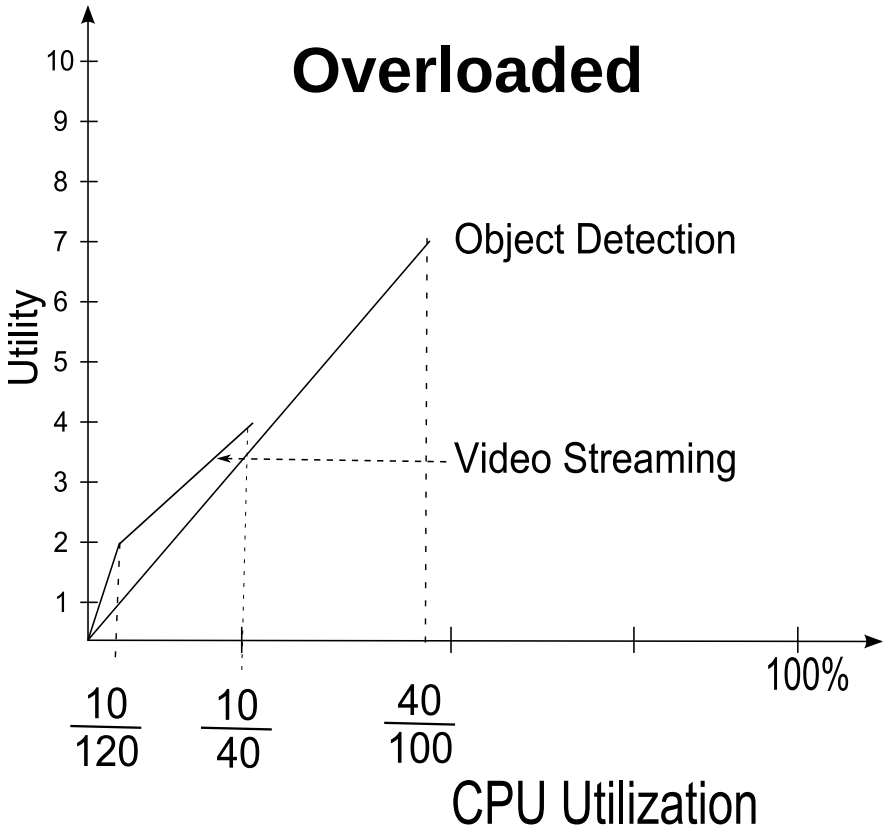




ZS-QRAM: More mission-critical utility from same resources

Periods as allocation points

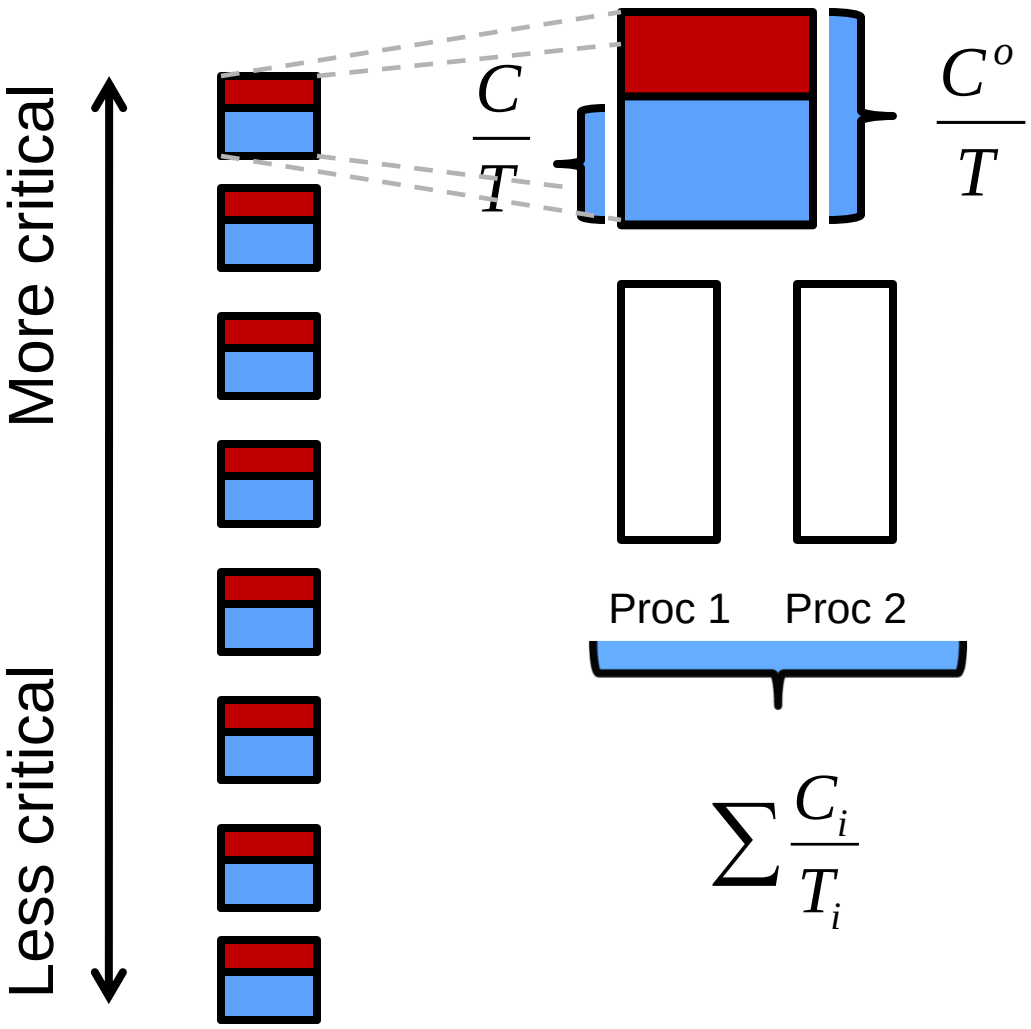




Initial allocation with nominal functions

On overload: period degradation with overloaded functions

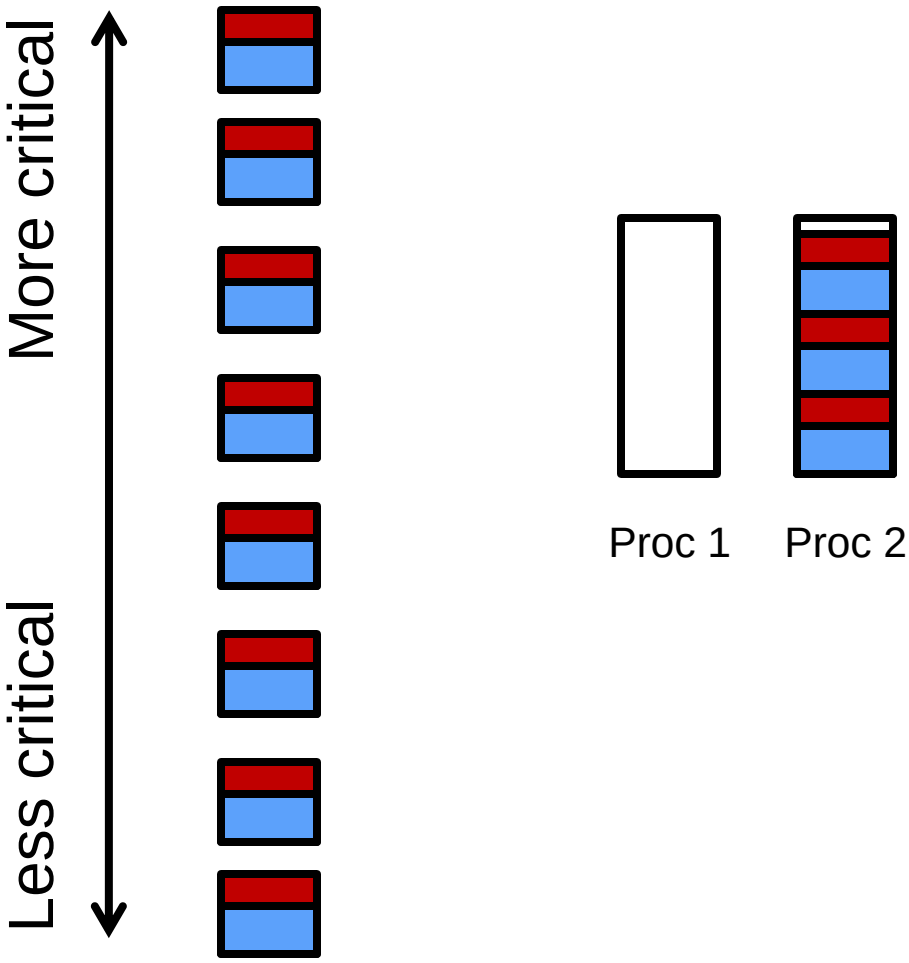
Compress-on-Overload Packing (COP)



Compress-on-Overload Packing (COP)

Phase 1: Pack by criticality then size

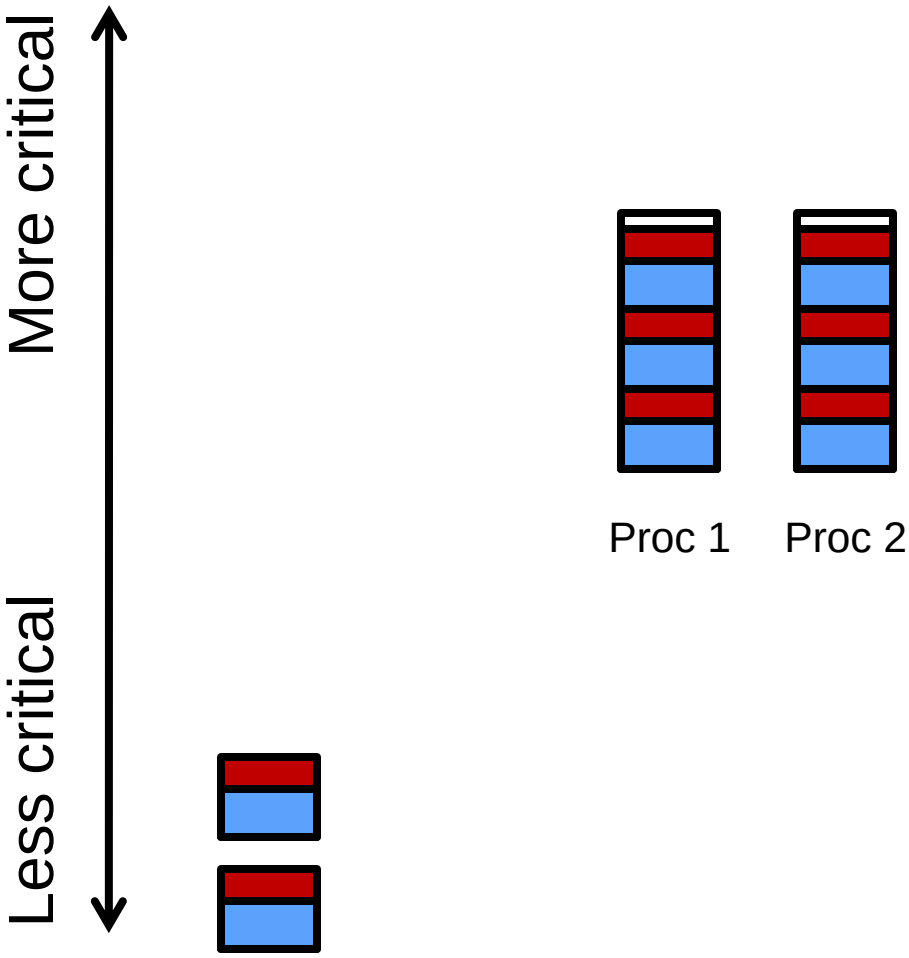
object size = $\frac{C_i^{\,o}}{T_i}$



Compress-on-Overload Packing (COP)

Phase 2: Pack by criticality then size

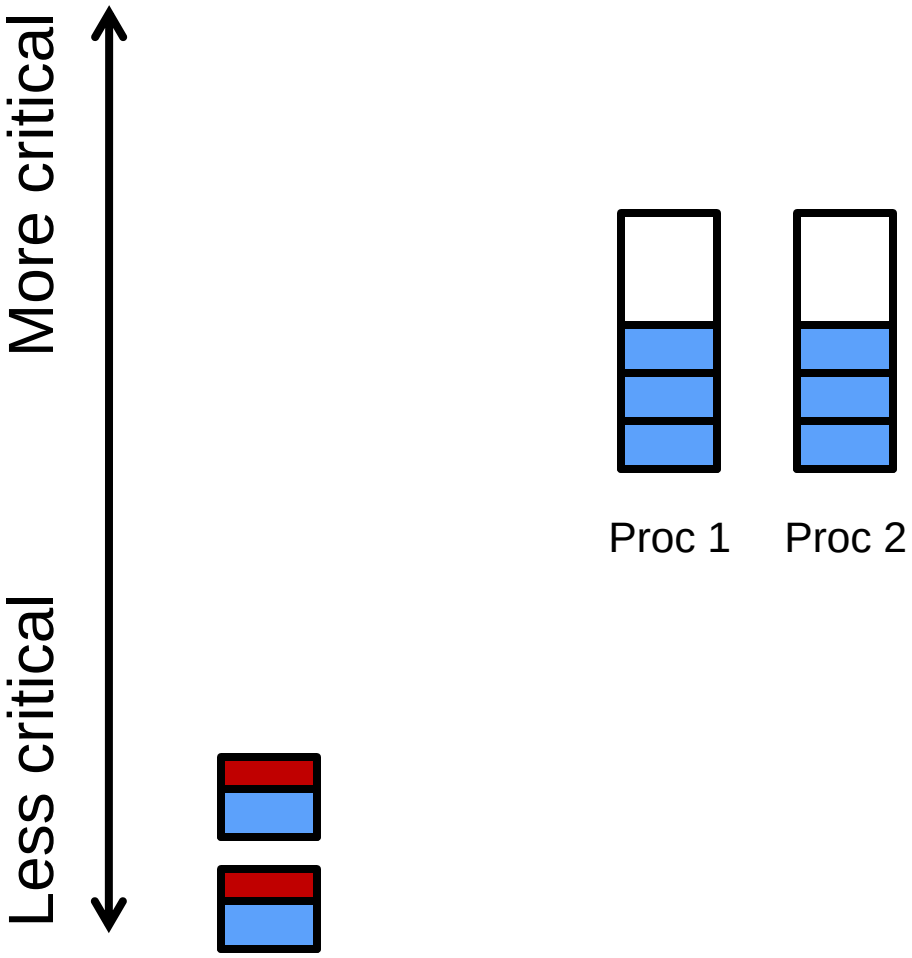
object size = $\frac{C_i}{T_i}$



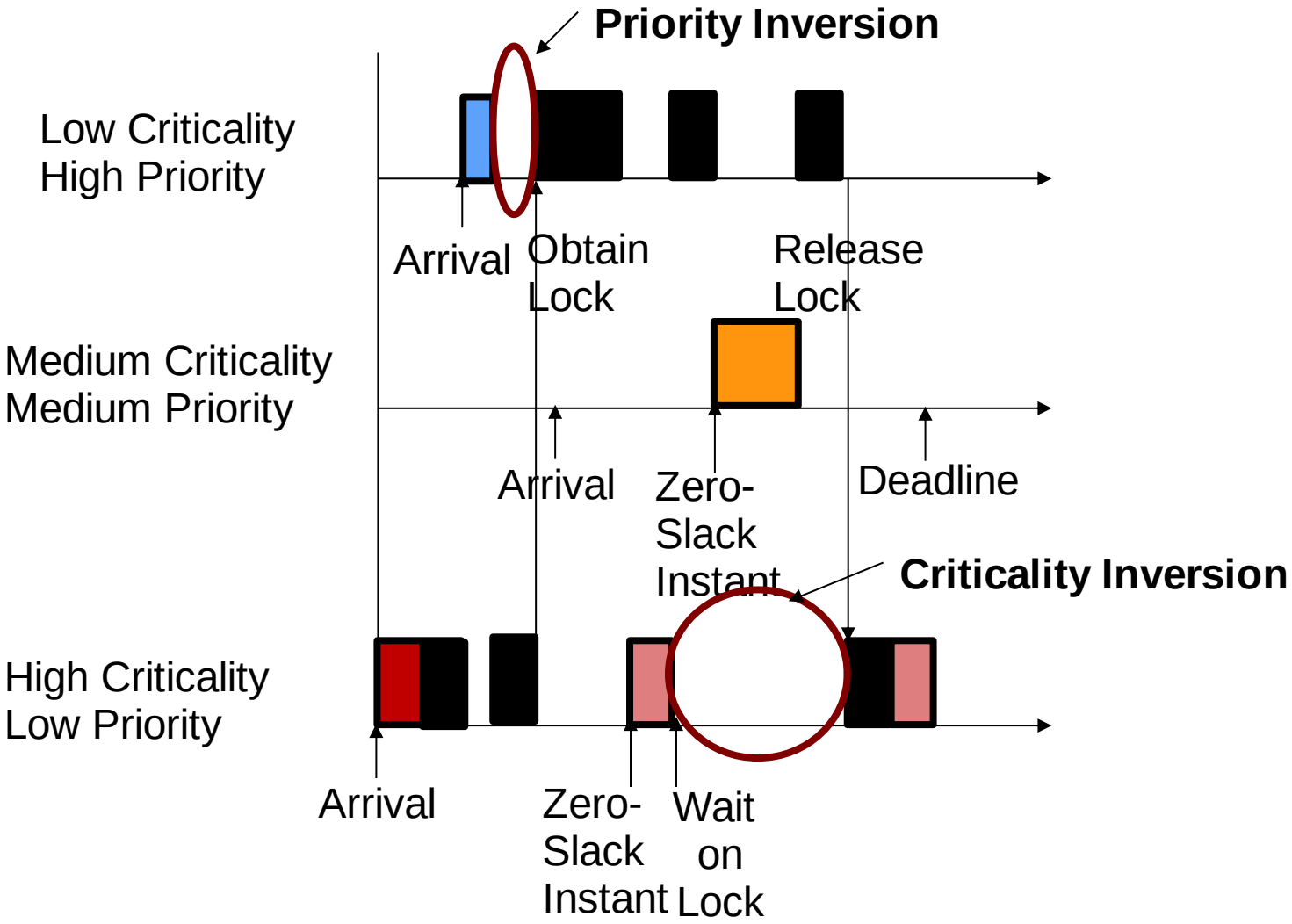
Compress-on-Overload Packing (COP)

Phase 2: Pack by criticality then size

object size = $\frac{C_i}{T_i}$



Shared Resources: Priority & Criticality Inversion



Priority & Criticality Inheritance Protocols

Priority and Criticality Inheritance Protocol (PCIP):

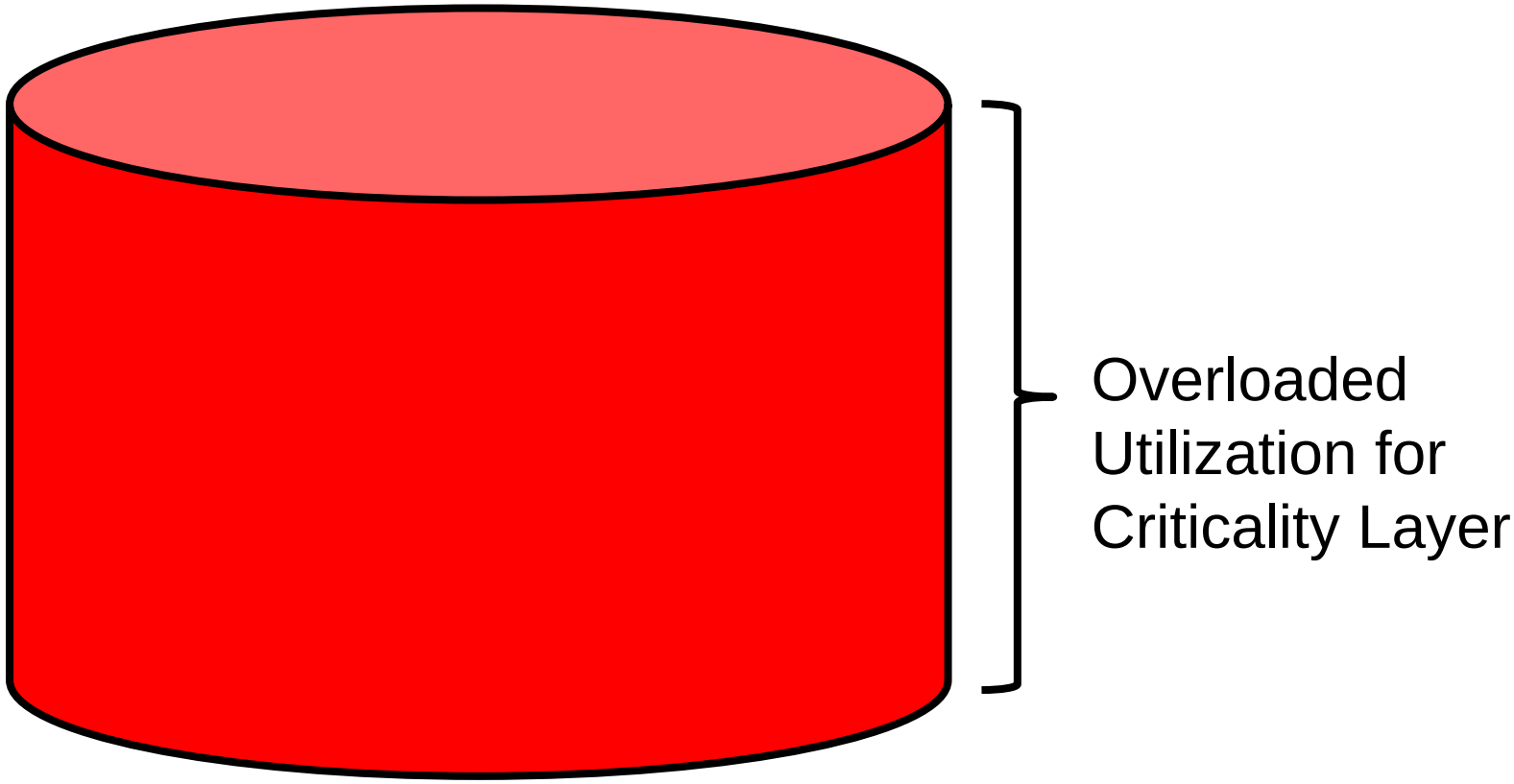
- A task **holding** a lock inherits the highest priority of all tasks **requesting** the lock AND inherits the highest criticality among all tasks that **request** the lock (could be different or the same)

Priority and Criticality Ceiling Protocol (PCCP):

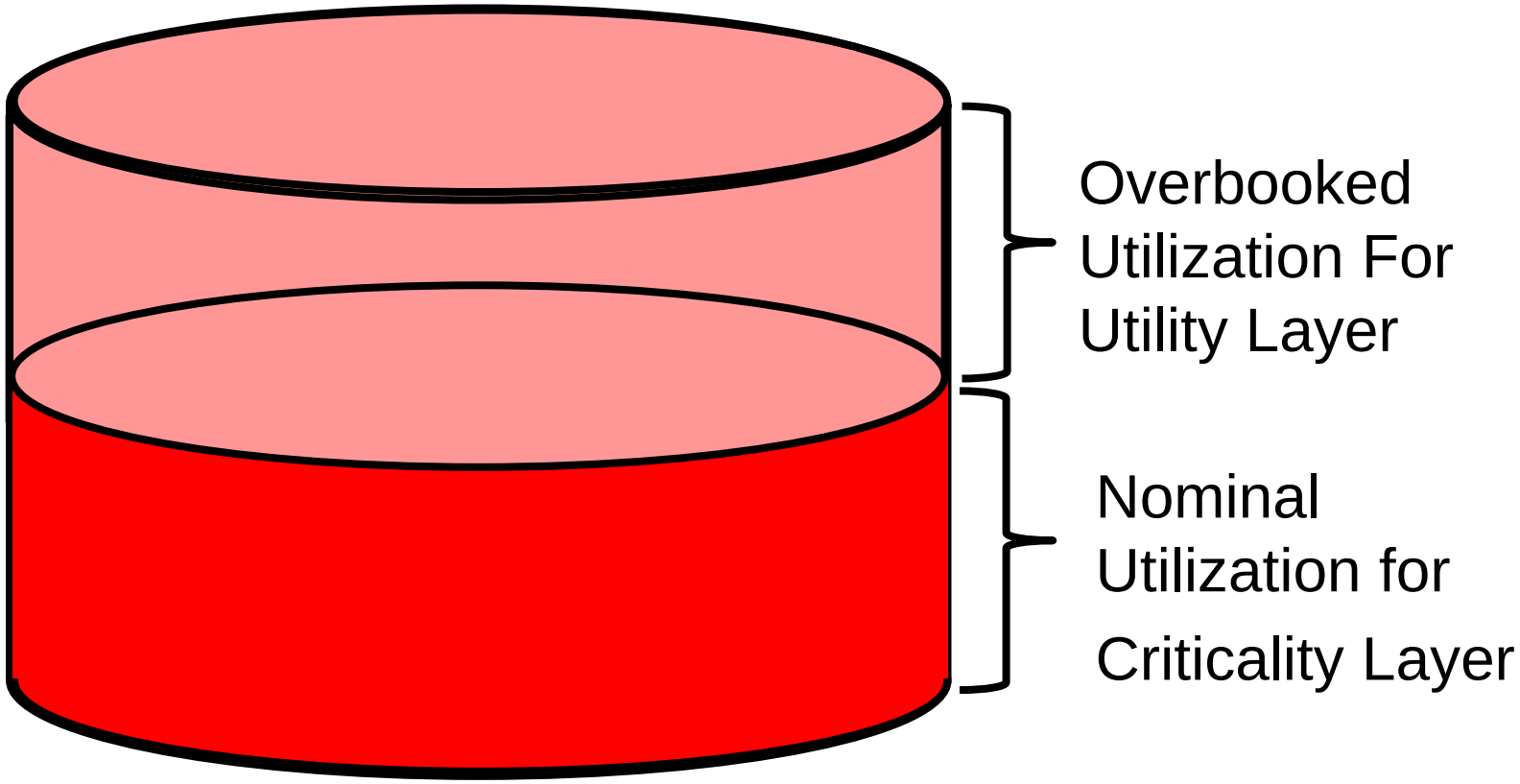
- Each lock is assigned both a:
  - Priority ceiling that is the highest priority among all tasks that can request the lock
  - Criticality ceiling that is the highest criticality among all tasks that can request the lock
- Both the priority ceiling and the criticality ceiling are inherited by the task that hold the lock

*** PCCP prevents deadlocks**

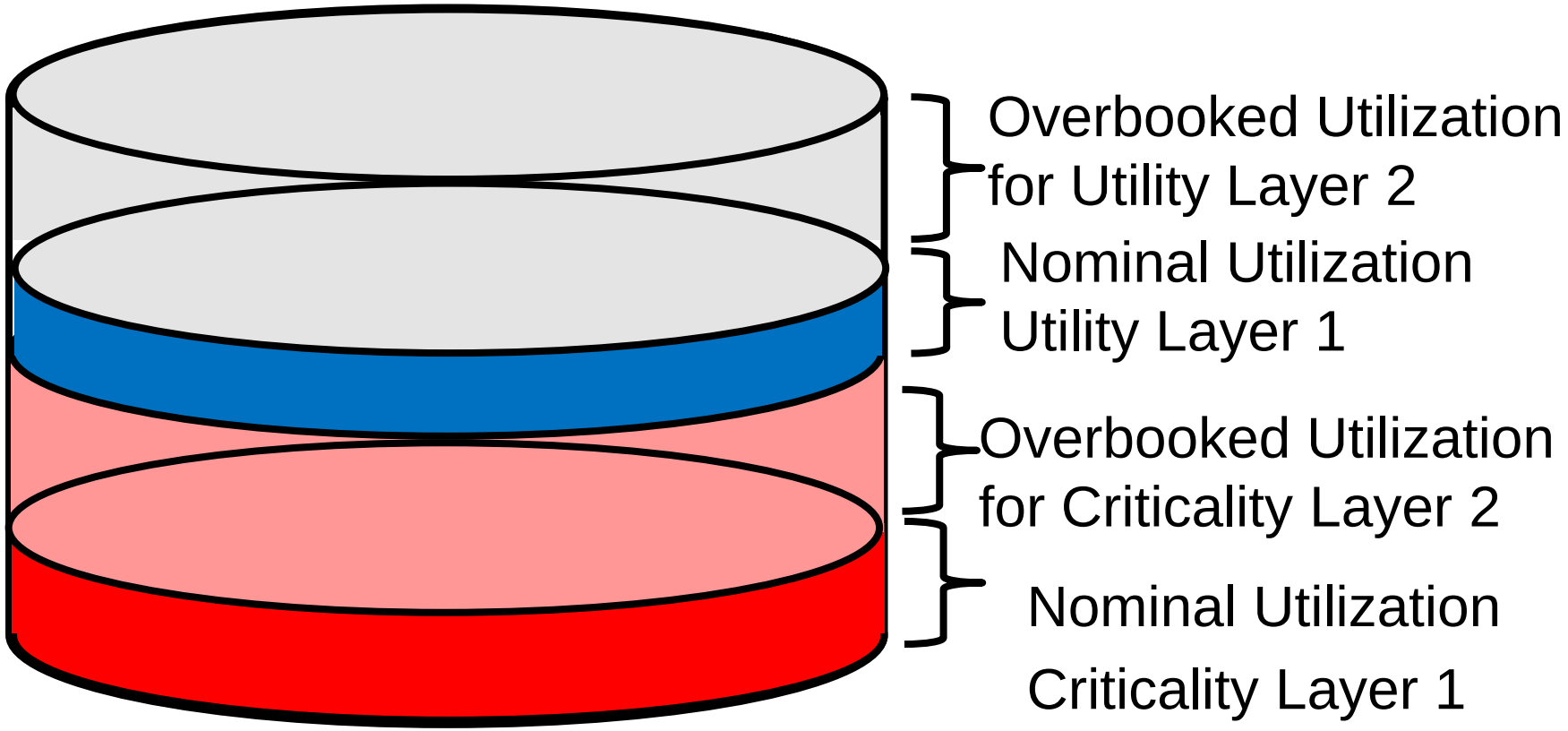
Integration with Criticality Layers



Integration with Criticality Layers



Integration with Criticality Layers



Certification Standards

DO-178-B/C

- Design Assurance Levels (A-E)
- Based on criticality of failure
- Higher-criticality => stringent validation / larger safety margin

ISO-26262

- [1]Automotive Safety Integrity Levels (ASIL) : A-D
- Classification based on risk analysis
- Should protect higher-critical from lower ones

Certification at highest-level if temporal protection cannot avoid it

Incremental certification

- A criticality layer can be certified ignoring lower-criticality

1. Ficek, et al. “Applying the AUTOSAR timing protection to build safe and efficient ISO 26262 mixed-criticality systems”. ERTS 2012

Open Issues

Input / Output

- Interrupts
- DMA transactions

Network bandwidth

- Single hop
- Multiple hop

Memory hierarchy

- Cache
- Memory banks
- Memory bus

Shared resources (locking protocols)

- Extend protocols to multi-processor (multi-cores)

Concluding Remarks

Asymmetric protection (ZS-QRAM)

- Efficient overbooking without compromising safety

Layered overbooking

- Efficient scheduling between critical and non-critical tasks

Sub-layers overbooking within criticality and utlility layers increases adaptability of systems

- Zero-Slack Packing algorithms (COP) allows efficient overbooking in partitioned scheduling of multi-core/multi-processor systems
- Priority and Criticality Inheritance Protocols enable predictable locking protocols for resource sharing

Enables Layered Certification

- Incremental by layers
- Layered safety margins