# Designing Mixed Criticality Applications on Modern Heterogeneous MPSoC Platforms

**Giovani Gracioli**, Rohan Tabish, Renato Mancuso, Reza Mirosanlou, Rodolfo Pellizzoni, and Marco Caccamo


***Technical University of Munich***

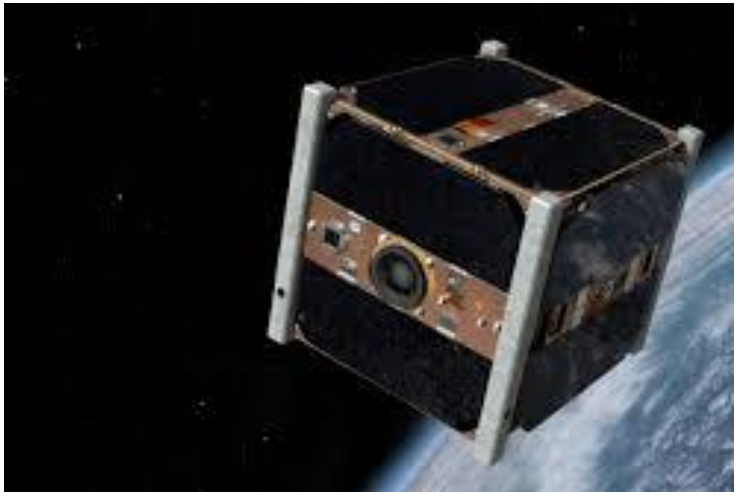

***University of Illinois at Urbana-Champaign***


***University of Waterloo***

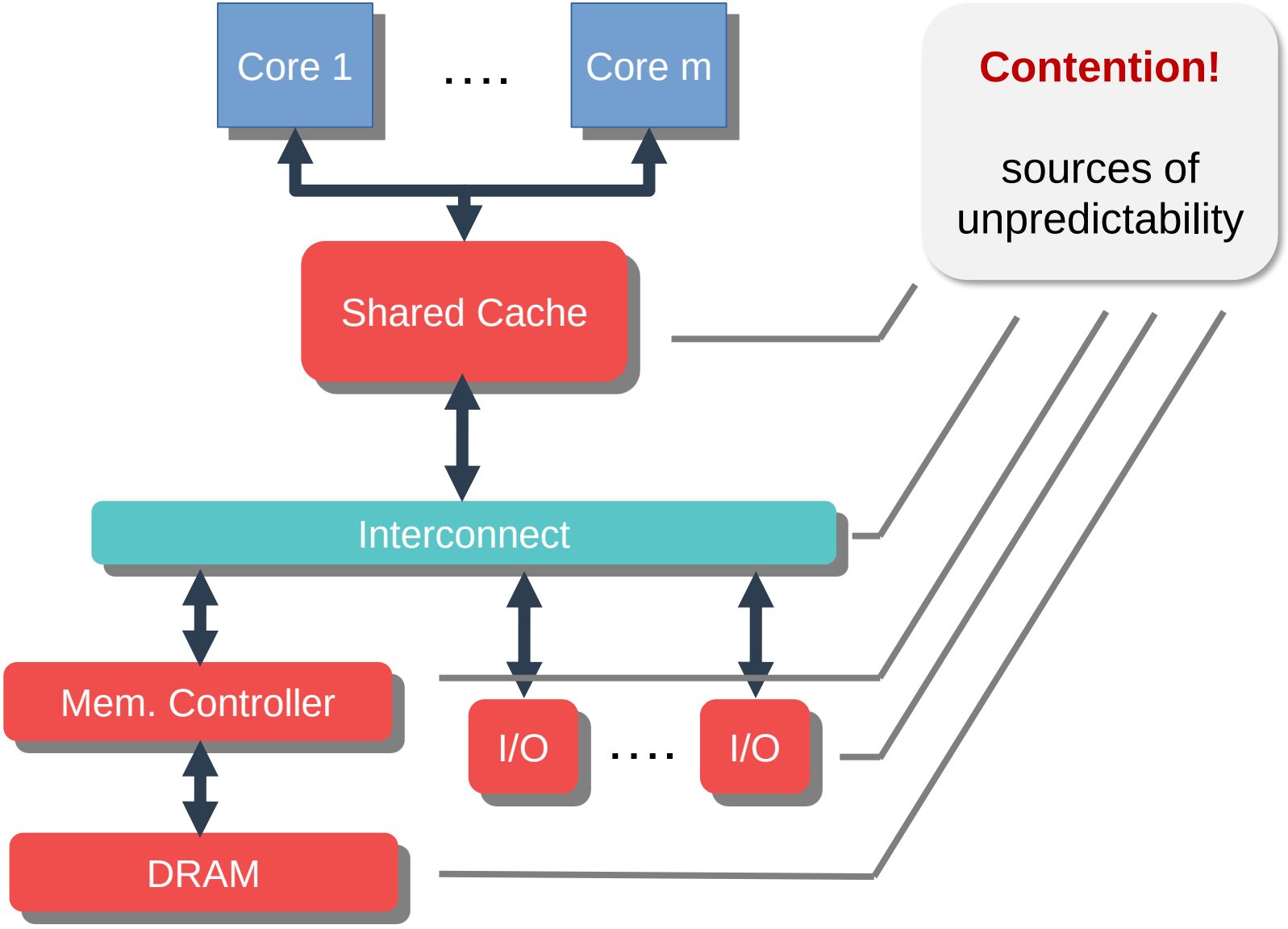

***Boston University***

**ECRTS 2019** @ Stuttgart

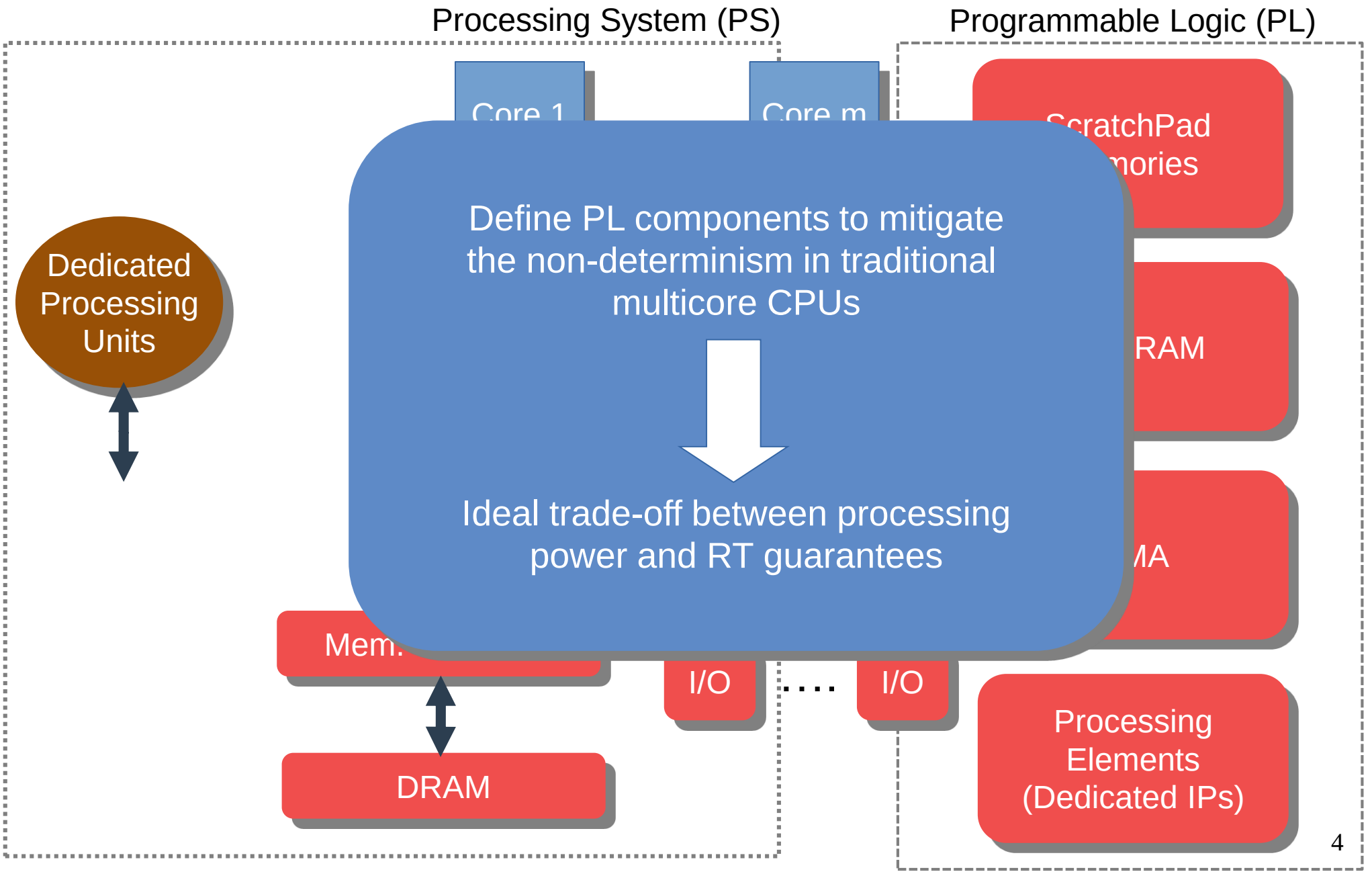
# Introduction



# Multicore Processors



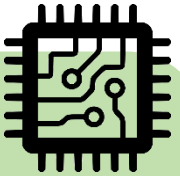
# Multiprocessor System-on-a-Chip (MPSoC)



# Contributions

High-performance and time-sensitive applications to co-exist under strict temporal isolation

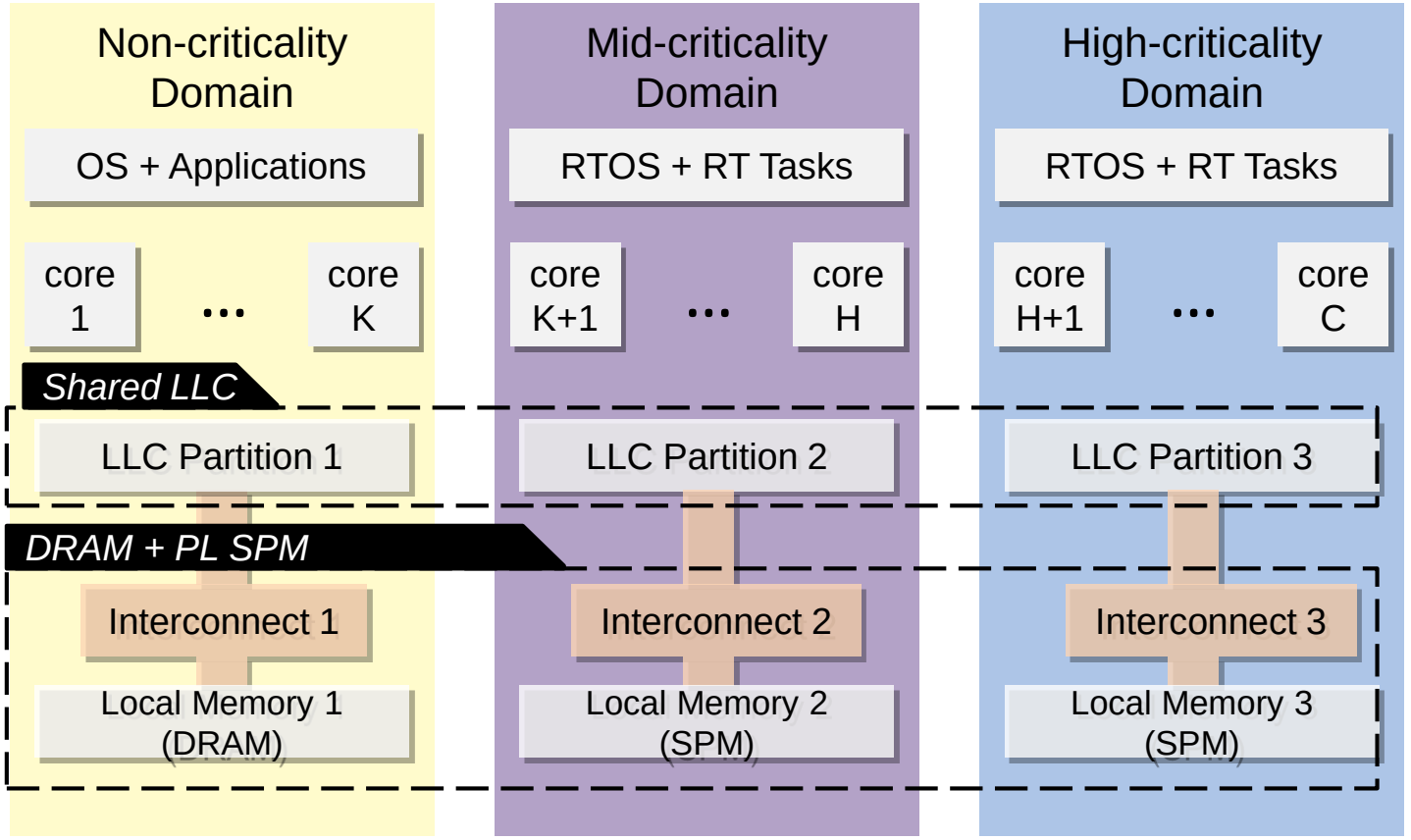
Set of SW and HW Techniques:
Hypervisor with coloring & code relocation,
PL-side SPM,
Variable TDMA slot size



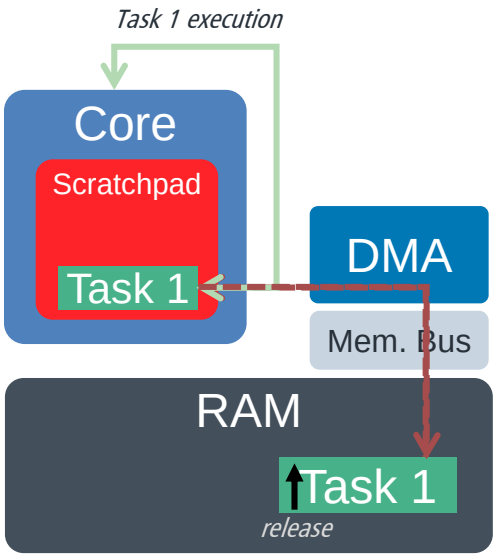
Hardware IP to prevent the problem of memory waste when cache coloring is used

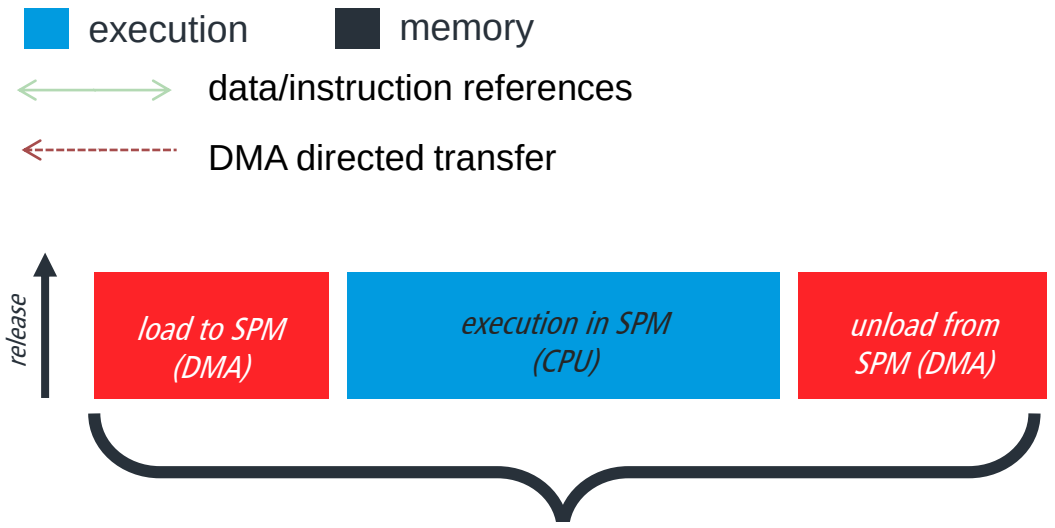
Full-stack implementation on one of the latest-generation MPSoC

# System Model



# Background
# 3-Phase Task Execution Model

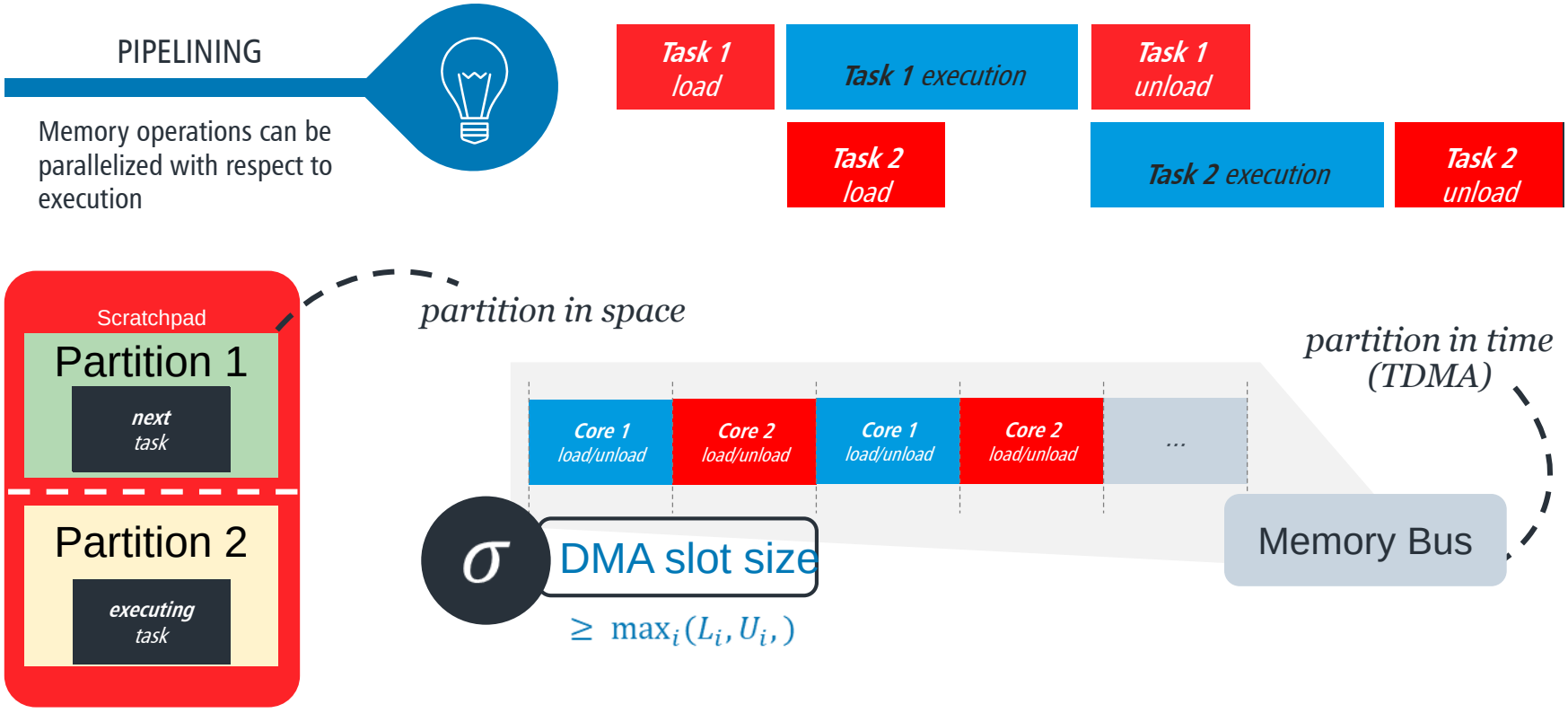


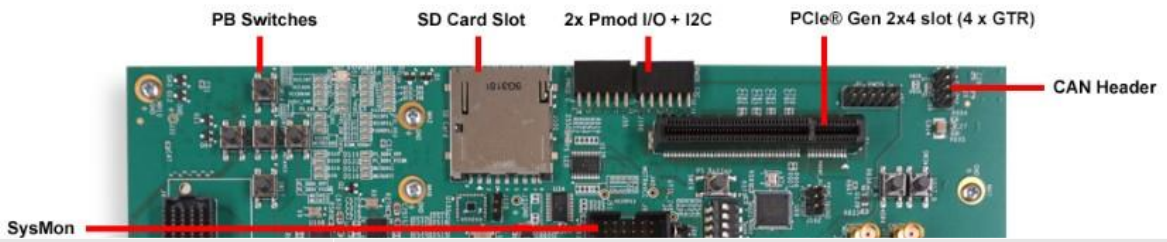

3 *stages*, 2 *resources*

$$\tau_i = \{L_i, C_i, U_i\}$$

# Pipelining and Memory Bus Scheduling



# Chosen Platform




| | |
|---|---|
| CPU units | 4x A53 1.2 Ghz, 2x R5 600 Mhz |
| A53 Memories | 32KB private I/D caches, 1MB LLC |
| R5 Memories | 32KB private I/D caches, 128KB TCM |
| PS-PL interfaces | 2x HPM, 1x LPD (PS→ PL)<br>2x HPC, 4x HP (PL→ PS) |
| Memories | DDR 4GB 64-bit (PS), OCM 256KB (PS)<br>DDR 512MB 16-bit (PL), BRAM 3MB (PL) |


Xilinx Ultrascale+ ZCU102

# Design Space Exploration

- MPSoCs allow many possible designs

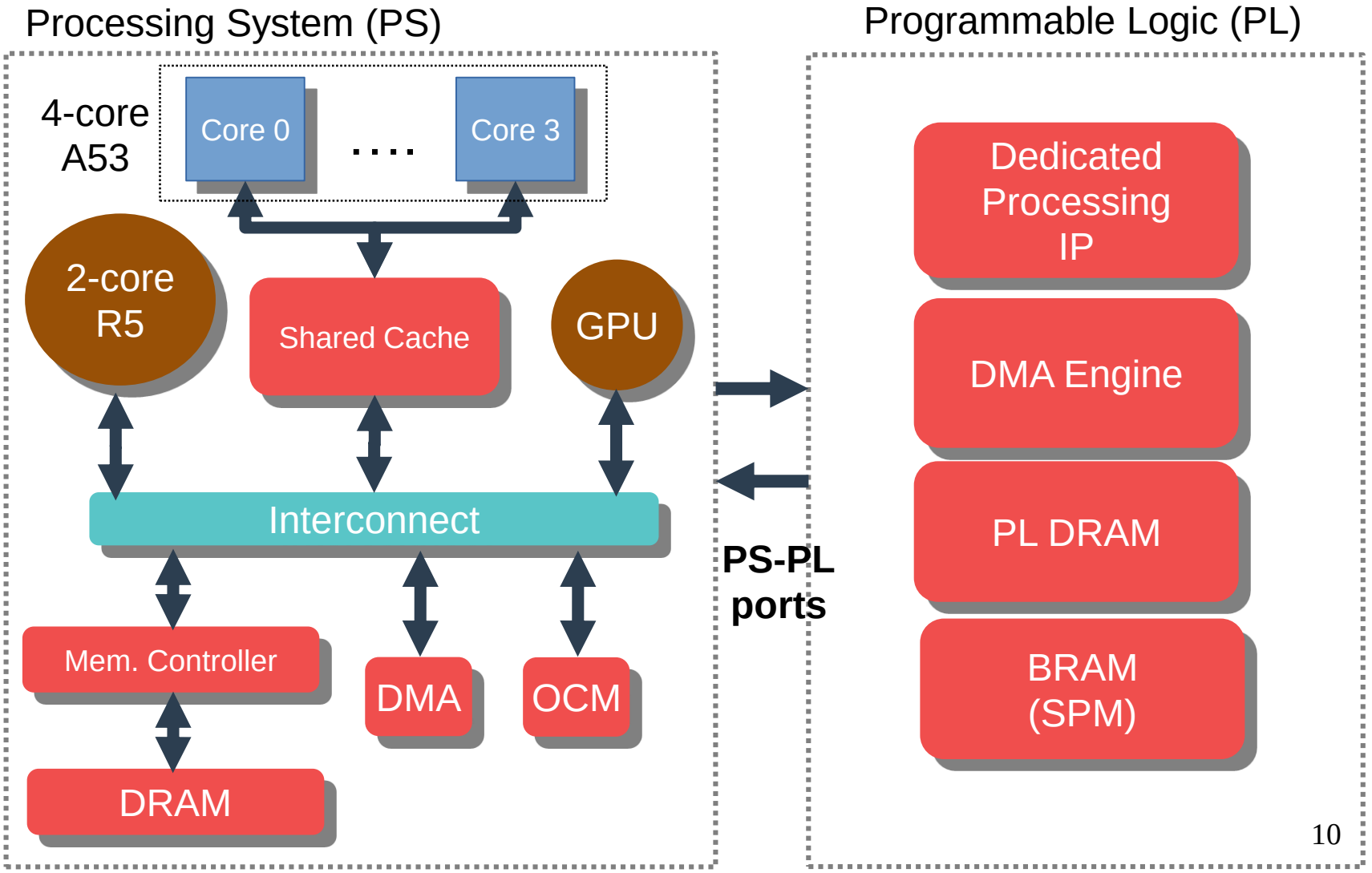
Where to execute tasks?

Where to implement the commu. engine?

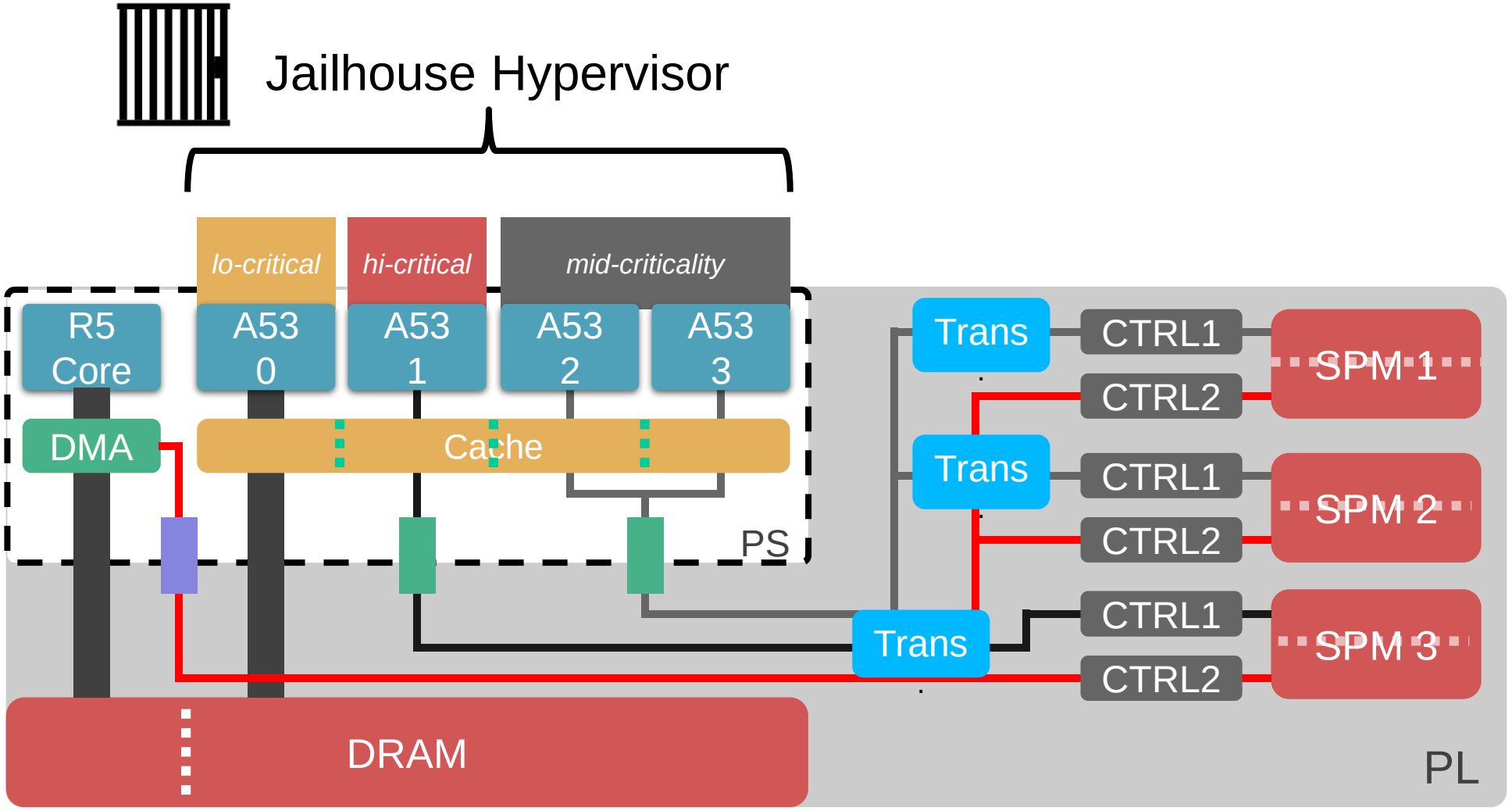
Which main memory?

SPM memory?

How to handle PS-PL communi.?



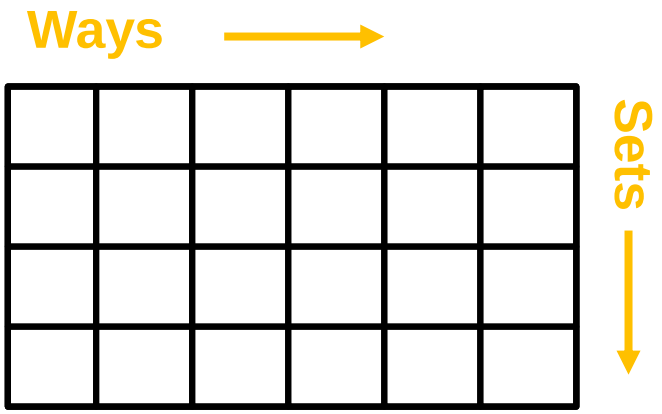
# Proposed Design



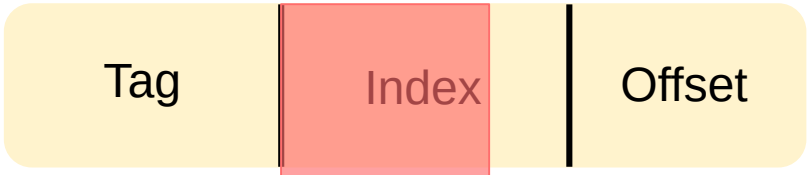

= High-Performance PS-to-PL Interfaces

= Low-Power Domain PS-to-PL Interface

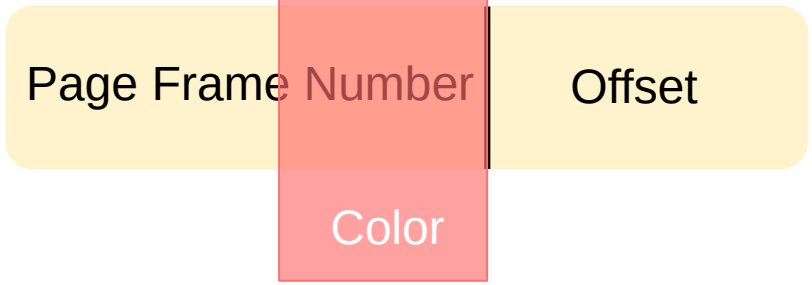
# Coloring Revisited



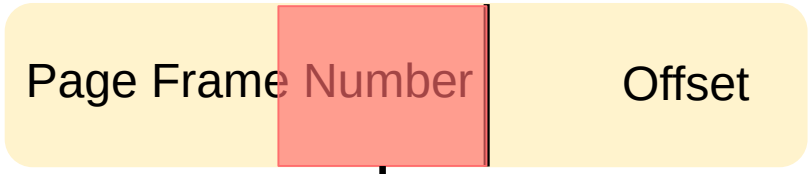

From cache controller's perspective

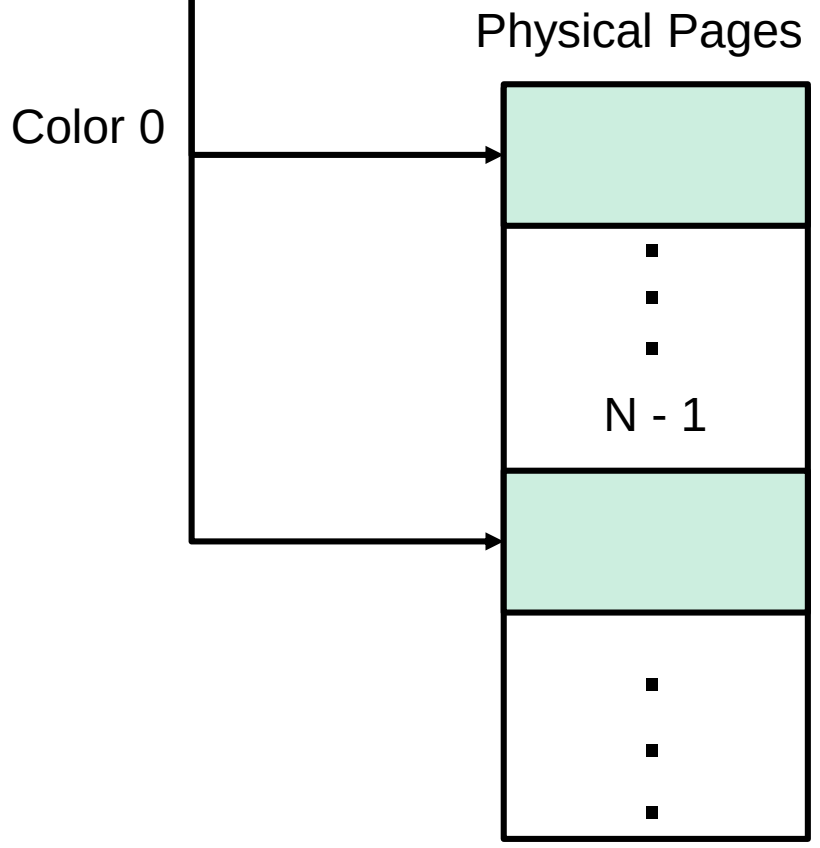



From OS's perspective

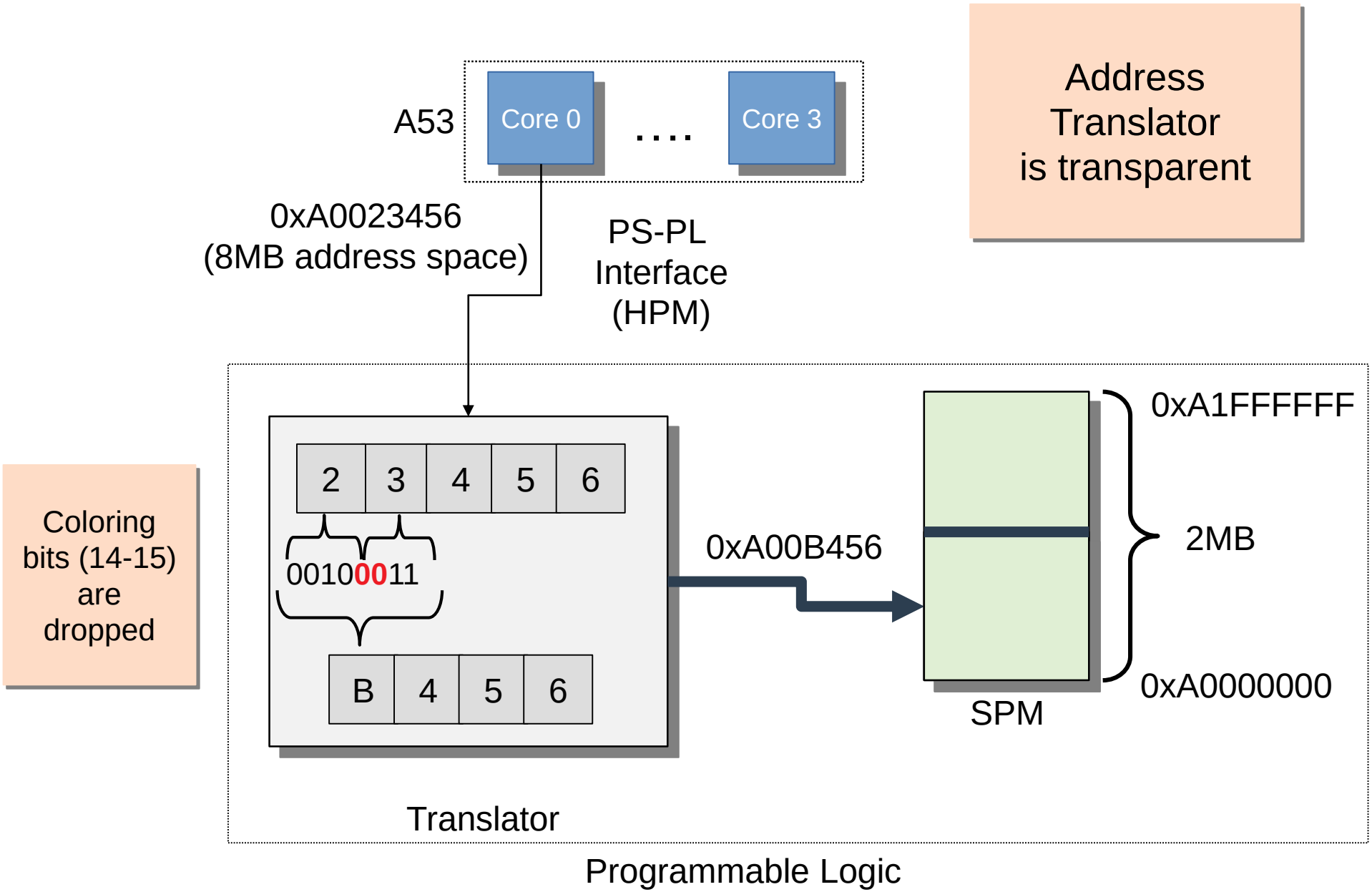



From OS's perspective



# Address Translator



# Designs Evaluation

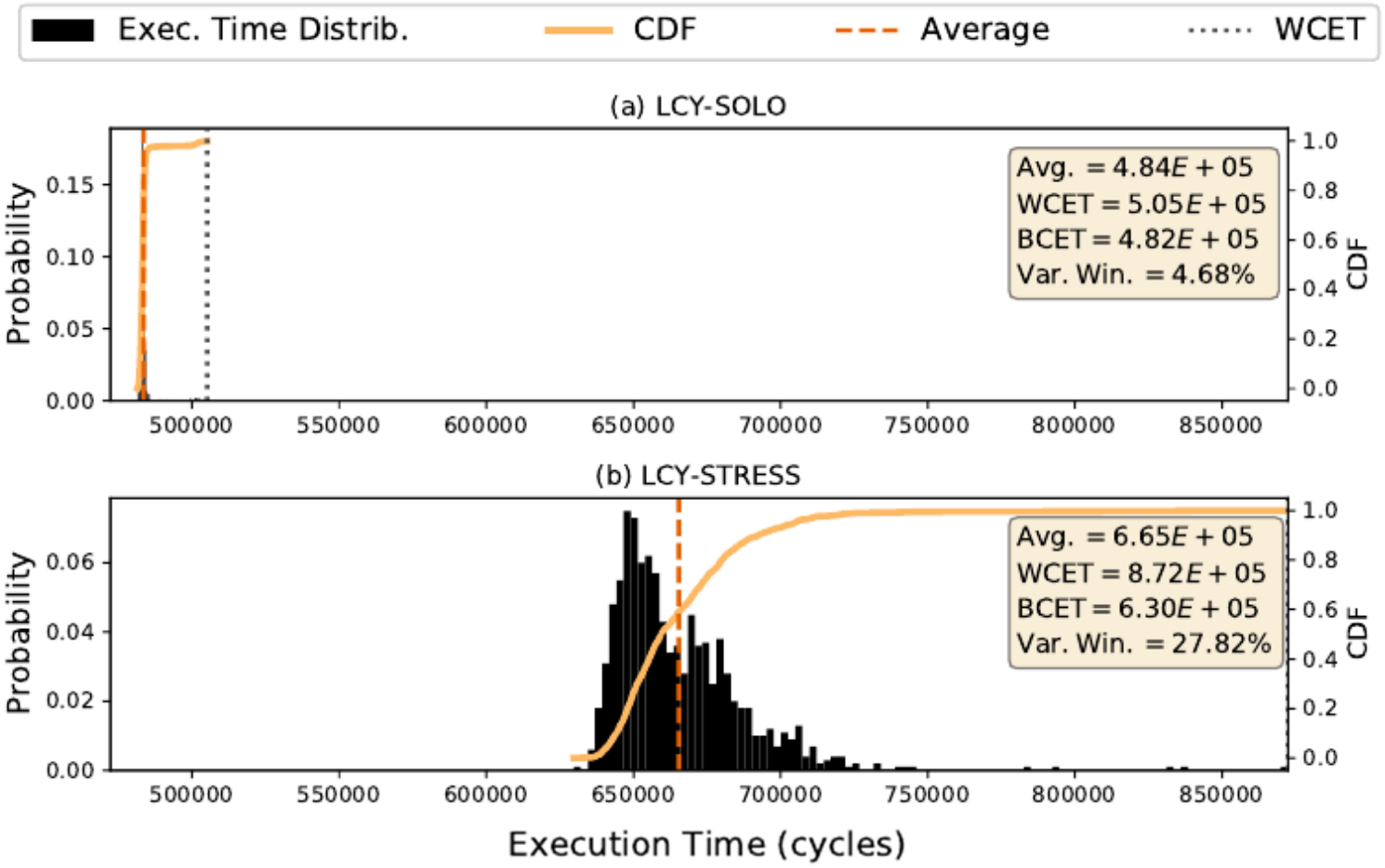
- We performed an experimental evaluation to evaluate the created designs. We used:
  - Two benchmarks from San Diego Visual Benchmark Suite (SD-VBS) – disparity and mser
  - Bandwidth benchmark (BW) to stress the memory subsystem
  - Main memory (DRAM in PS) and SPM (BRAM in PL)
    - PS-DRAM is faster than PL-DRAM
  - DMA on the PS-side
    - DMA on the PS-side is also faster than PL-side DMA
- Predictability for mixed criticality applications

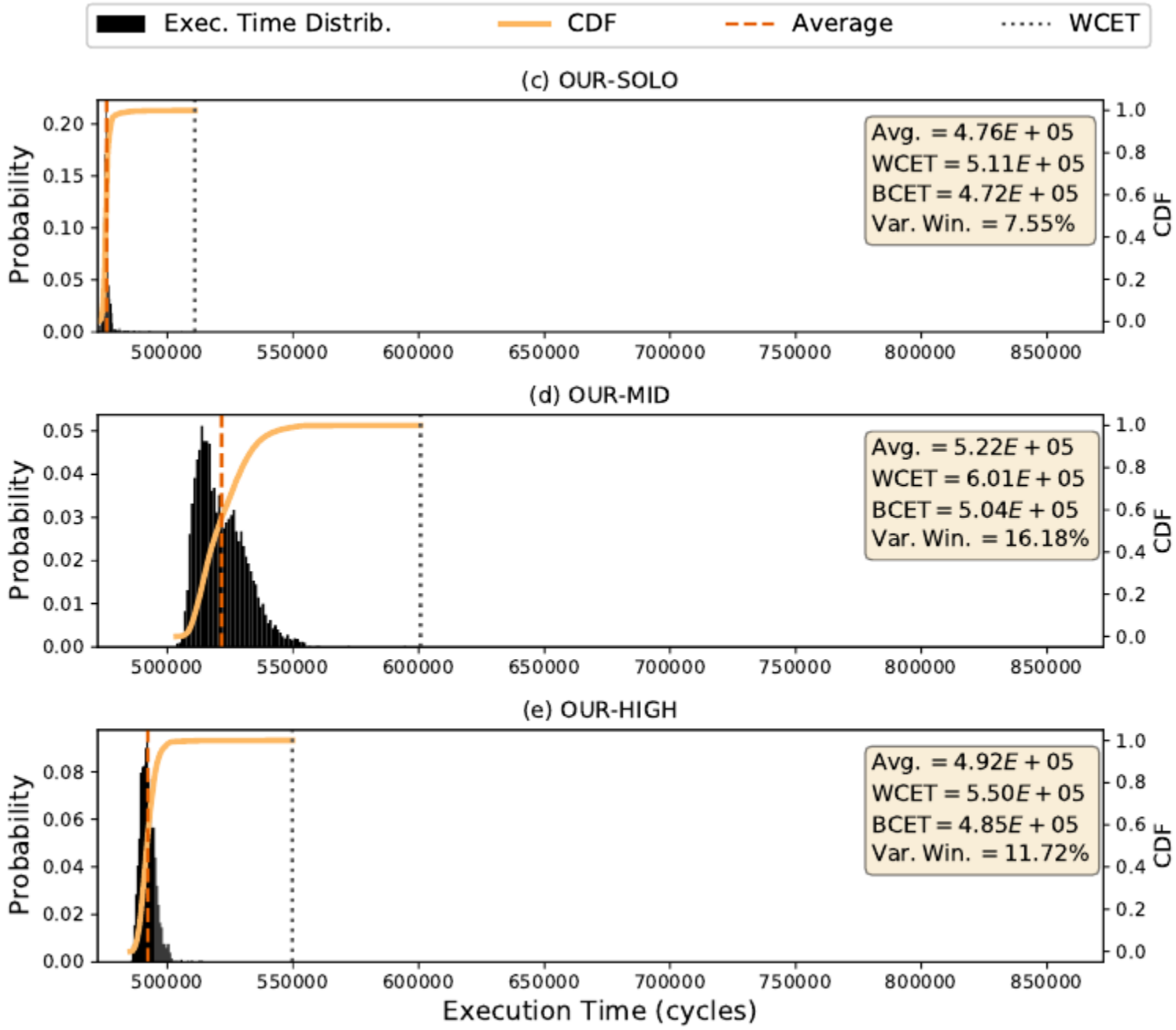
# Designs Evaluation

- We consider the following execution scenarios:

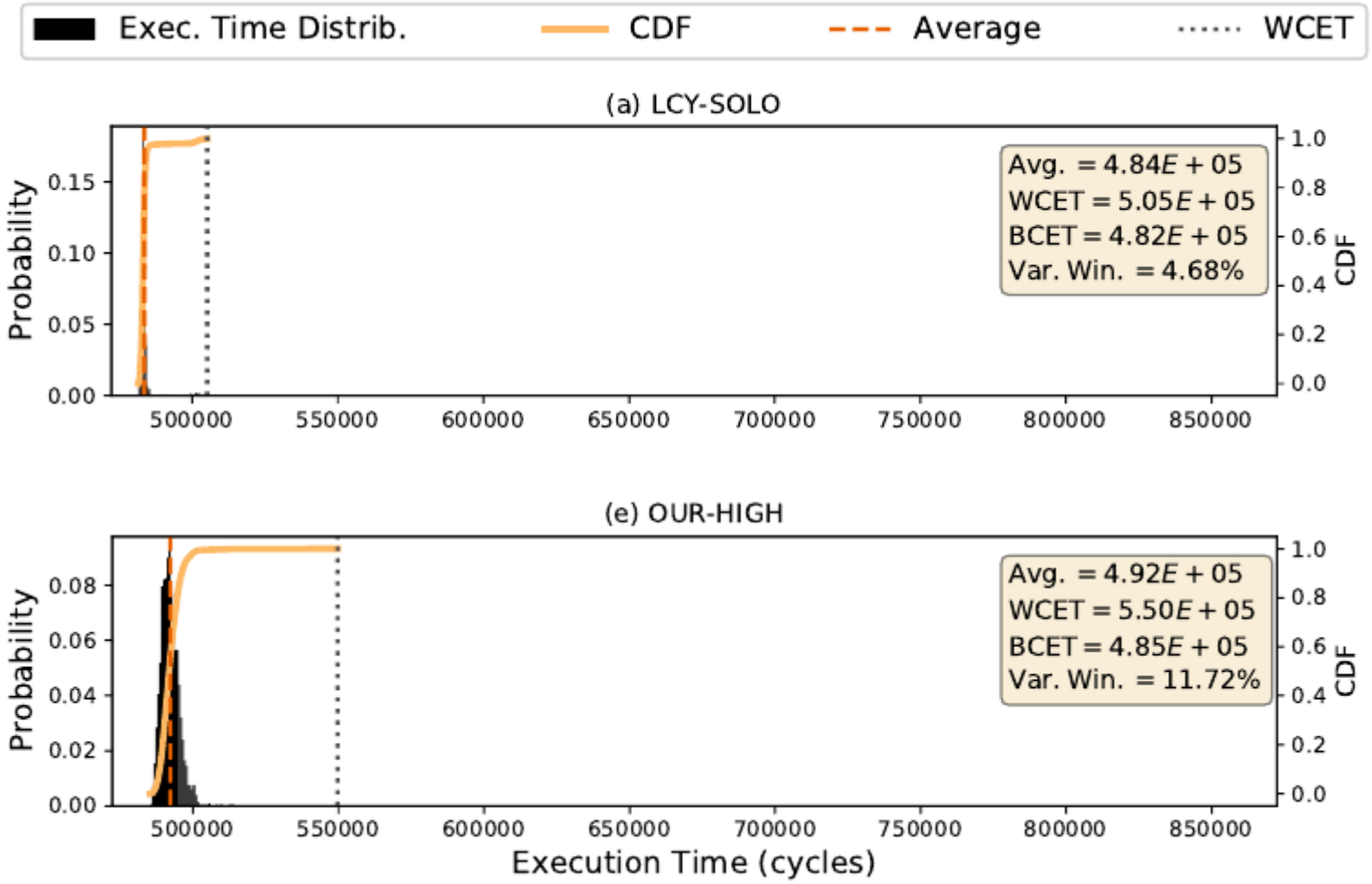
| Scenario | Experiment | Accessed Memory | Coloring | PS-PL Interface | Contention Type |
|---|---|---|---|---|---|
| **LCY-SOLO** | Solo | PS DRAM | No | Not used | None |
| **LCY-STRESS** | Contention | PS DRAM | No | Not used | 3x BW |
| **OUR-SOLO** | Solo | SPM | Yes | Dedicated | None |
| **OUR-HIGH** | Contention | SPM | Yes | Dedicated | 1x BW from low-crit.<br>2x BW from mid-crit. |
| **OUR-MID** | Contention | SPM | Yes | Shared | 1x BW from low-crit.<br>1x BW from mid-crit.<br>1x BW from high-crit. |

# Results for mser




Exec. Time Distrib.
CDF
Average
WCET
(c) OUR-SOLO
Avg. = 4.76E + 05
WCET = 5.11E + 05
BCET = 4.72E + 05
Var. Win. = 7.55%
(d) OUR-MID
Avg. = 5.22E + 05
WCET = 6.01E + 05
BCET = 5.04E + 05
Var. Win. = 16.18%
(e) OUR-HIGH
Avg. = 4.92E + 05
WCET = 5.50E + 05
BCET = 4.85E + 05
Var. Win. = 11.72%
Probability
CDF
Execution Time (cycles)

# Mser: LCY-SOLO vs. OUR-HIGH



# DMA Evaluation

- DMA transfer time
  - Different data sizes
  - 1000 repetitions
  - AVG, STD, WCET
  - Programming overhead

# DMA Evaluation

| Transfer Size | Transfer Time | | | Bandwidth (MB/s) |
|---|---|---|---|---|
| | Average ($\mu s$) | STD | Worst-case ($\mu s$) | |
| 2 KB | 4.92 | 0.057 | 5.11 | 397.0 |
| 4 KB | 7.15 | 0.04 | 7.27 | 546.3 |
| 8 KB | 11.63 | 0.01 | 12.01 | 671.8 |
| 9.1 KB | 12.91 | 0.05 | 13.11 | 688.4 |
| 16 KB | 20.62 | 0.08 | 20.96 | 757.8 |
| 22 KB | 27.42 | 0.10 | 27.72 | 783.5 |
| 32 KB | 38.52 | 0.05 | 38.81 | 811.3 |
| 1 MB | 1149.44 | 0.05 | 1149.78 | 870.0 |

- STD within range [0.057, 0.1]
- Programming overhead: **3.89 us**
- Programming overhead vs. small data size transfers
- Model behaves well as long as task execution times are longer than the time required to reload an SPM partition

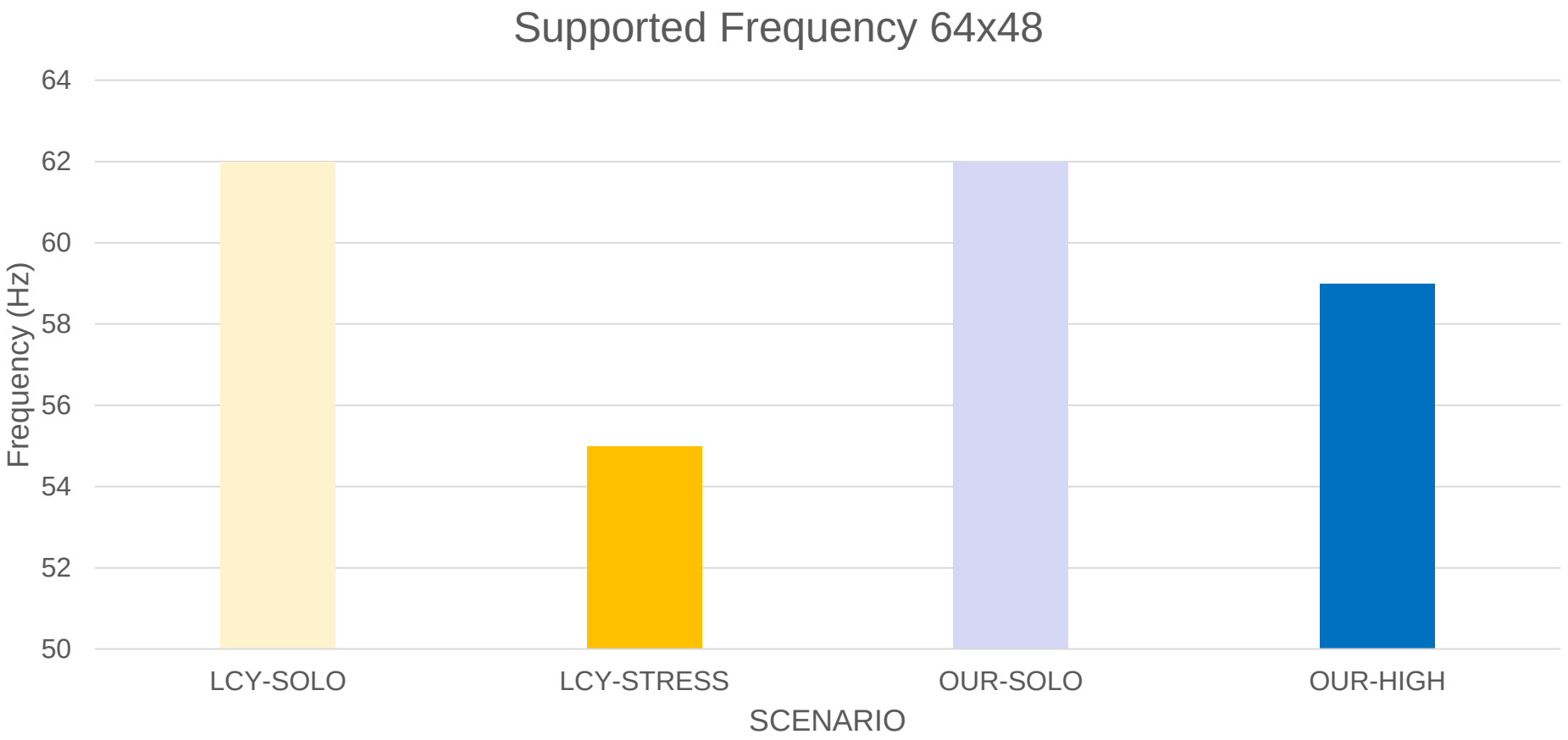
# Case study

- Videos frames captured from a camera are processed in a high-criticality domain
- Disparity: obtains relative positions of objects
  - Useful for cruise control, pedestrian tracking, and collision control
- Demonstrate how the system behaves in a realistic setup and show the limits in terms of achievable hard real-time guarantees

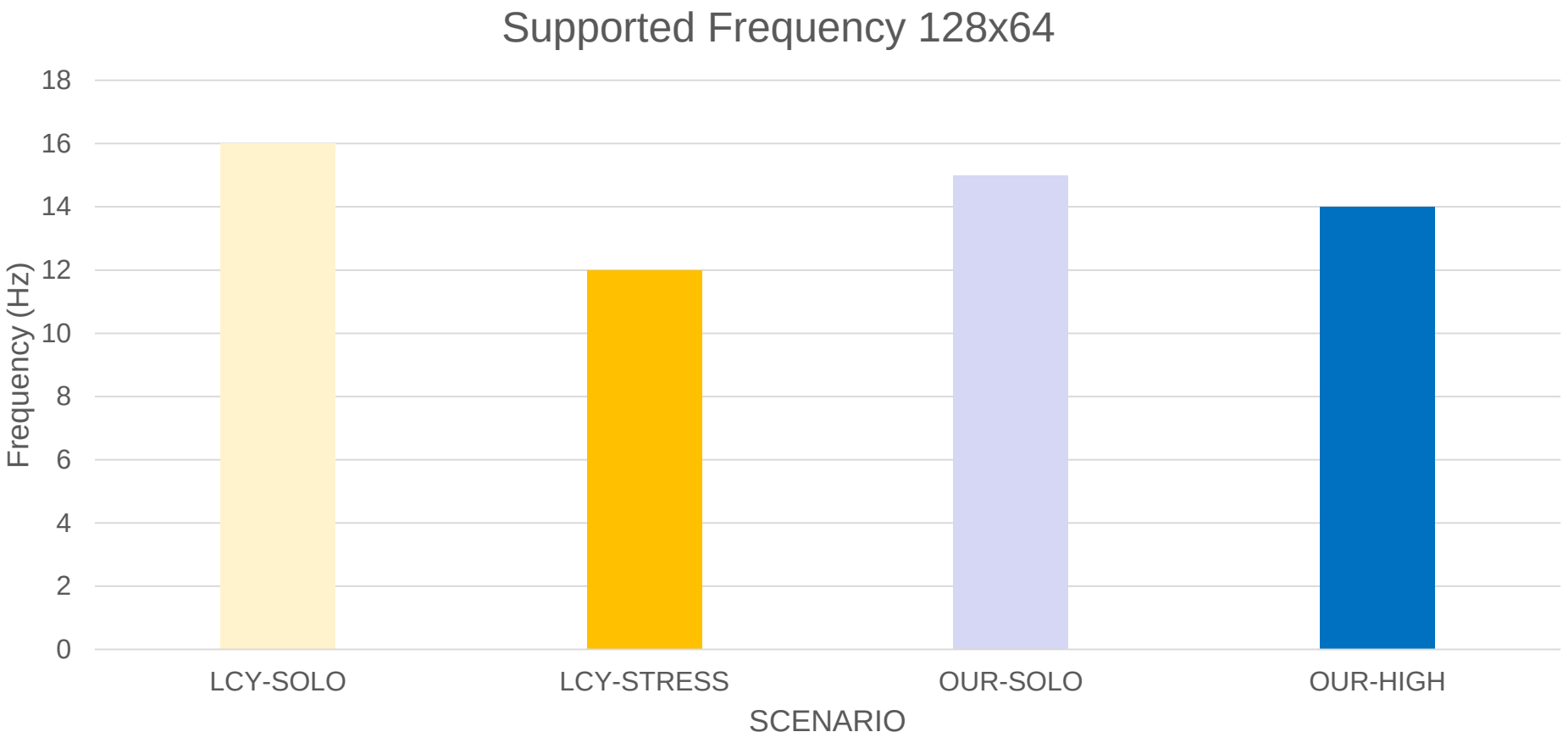
# Case study

- Two image sizes
  - 64x48 (9.1KB) and 128x64 (22KB)
  - Limitations on the SPM size and benchmark
- Images from the KITTI vision benchmark suite dataset
- Code size
  - Disparity 64x48: 349KB
  - Disparity 128x64: 745KB
  - Erika RTOS: 294KB
- Four out of the five scenarios previously described

# Case study: Disparity



# Case study: Disparity



# Summary and Future Work

**Multiple criticality domains on MPSoCs**

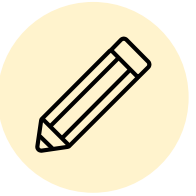
Software techniques (isolation, cache coloring, code/data relocation)

Hardware techniques (SPM, dedicated PS-PL interfaces, address translator IP)

Full-stack system implementation

Compiler Integration

Security



Schedulability Analysis


Thank you!