Memory Interference Analysis on Heterogeneous MPSoCs

Alfonso Mascareñas González
ISAE-SUPAERO / ONERA

Workshop CAPITAL 2023

Toulouse

13/06/2023

ONERA
THE FRENCH AEROSPACE LAB

ISAE
Institut Supérieur de l'Aéronautique et de l'Espace
SUPAERO

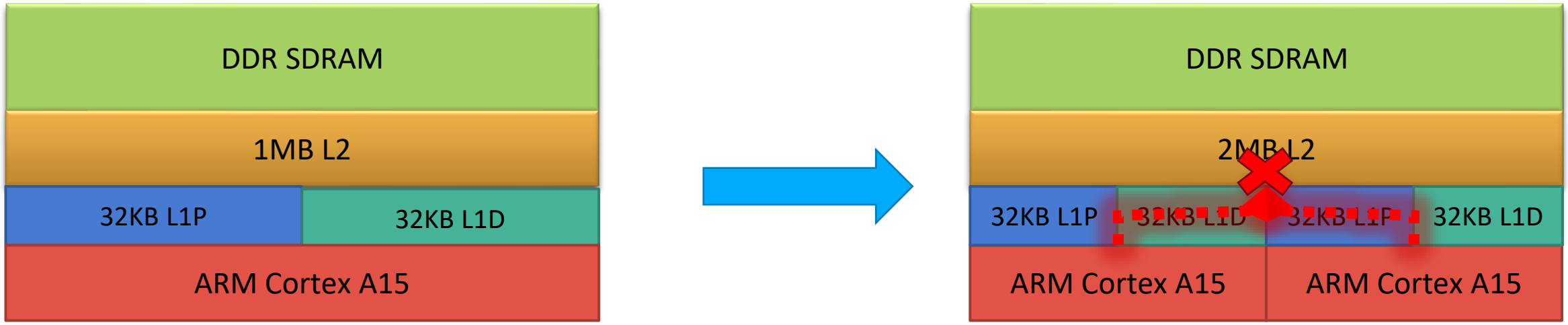
Content

Content

1. **Context**
2. On-board profiling
3. Memory interference analysis
4. Conclusions

Context: Safety-critical real-time systems

- High performance demanding real-time application, e.g., Autonomous drones
- Traditionally, monocore platforms have been used
- Multicore platforms → SWaP-C + throughput[1]
- Resources sharing imply interference → Predictability loss
- Execution deadlines can be missed → Certification problem[2]

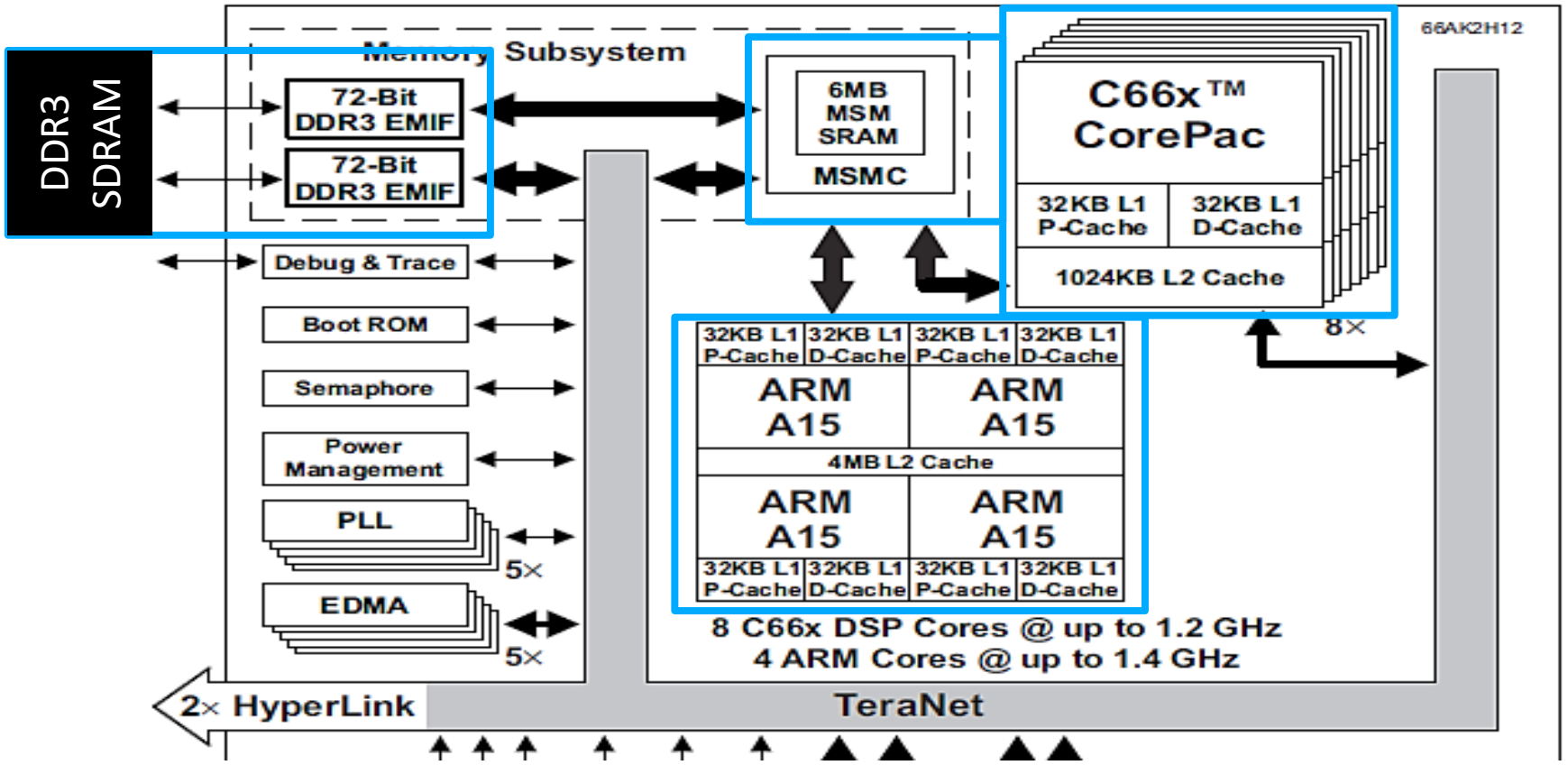


[1]R. Ramanathan, "White paper: Intel® multi-core processors making the move to quadcore and beyond"

[2]S. A. Jacklin, "Certification of safety-critical software under do-178c and do-278a," 2012.

Context: Keystone II MPSoC

Keystone II TCI6636K2H



Content

1. Context
2. **On-board profiling**
3. Memory interference analysis
4. Conclusions

On-board profiling: Hardware performance counters

Performance counters are used for monitoring hardware-related events. We highlight the following two advantages:

- Low-overhead access
- No code modification

The access pattern to the counters are important for correlating metrics. We point out these two:

- Reset-Start-Stop-Read
- Start-Read

On-board profiling: Hardware performance counters

Three measurement systems:

- ARM Performance Monitor Counters → Reset-Start-Stop-Read pattern
- DSP Time Stamp Register → Start-Read pattern
- DDR3 Controller Performance Counters → Start-Read pattern

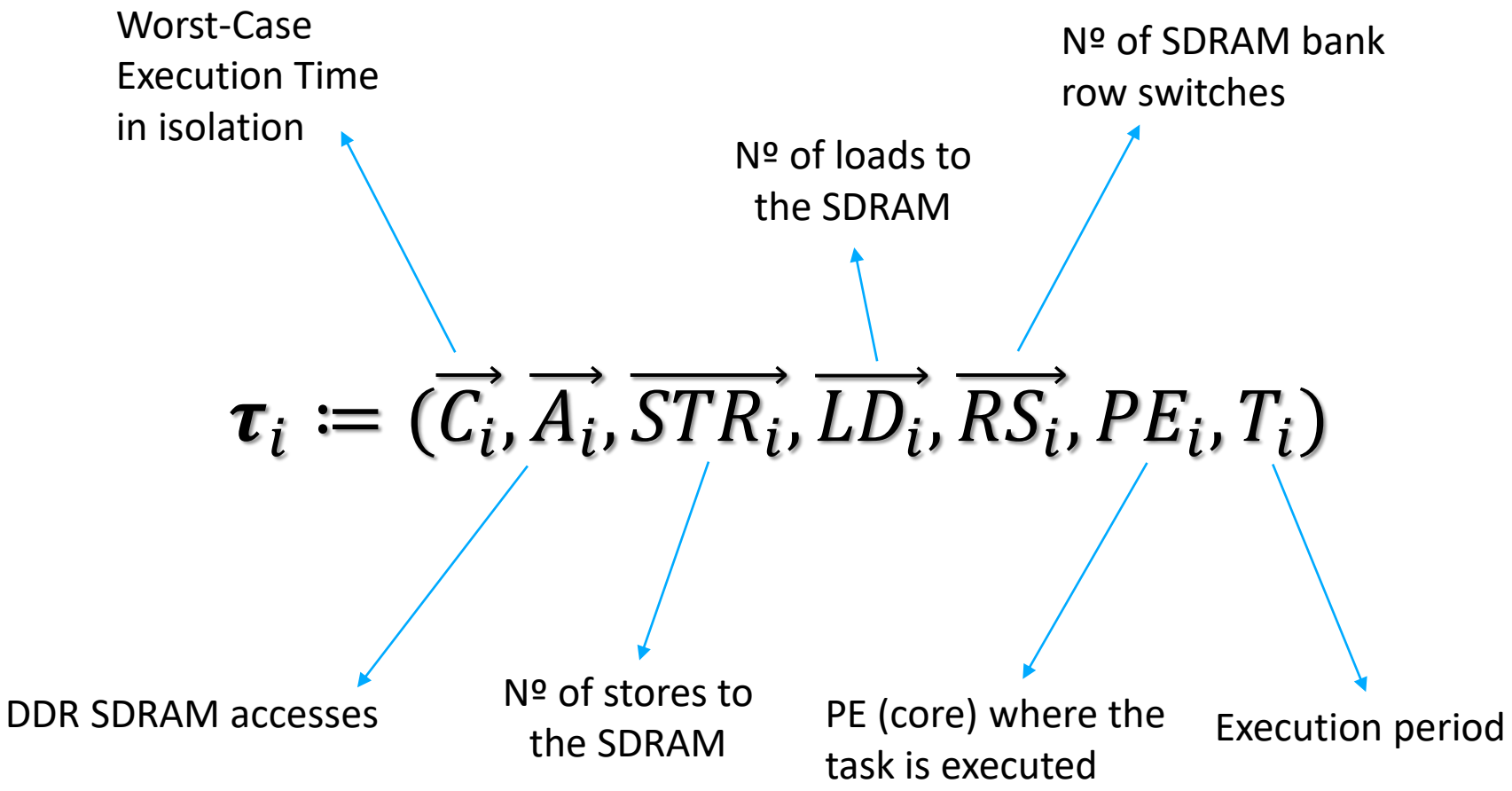
ARM Events
- Time
- L1 Cache Miss
- L1 Cache Access
- L2 Cache Miss
- L2 Cache Access
- Bus Access
- Branch miss prediction
- etc.

DSP Events
- Time

TI DDR3 SDRAM Events
- Time
- Total Accesses
- Row Activates
- Writes
- Reads
- Priority elevations
- etc.

System or core perspective

On-board profiling: Task profiling



On-board profiling: Task properties

The properties of the tasks change according to the PE they are executed on

$$\boldsymbol{\tau}_i := (\overrightarrow{C_i}, \overrightarrow{A_i}, \overrightarrow{STR_i}, \overrightarrow{LD_i}, \overrightarrow{RS_i}, PE_i, T_i)$$

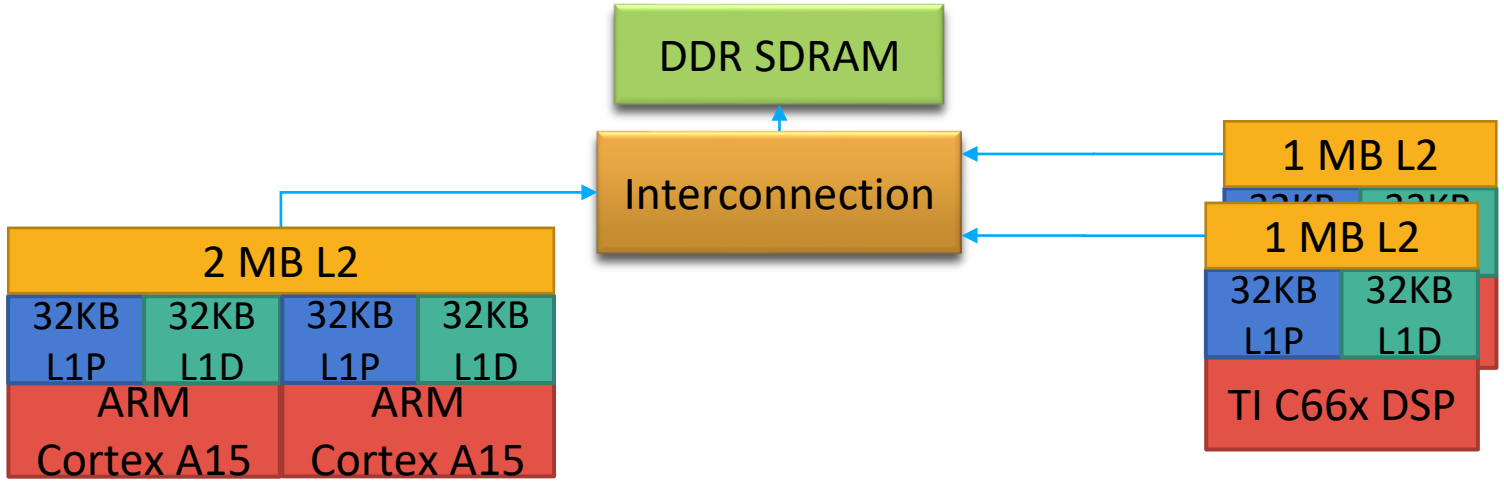
Depends on PE

ARM Cortex A15 TI C66x DSP

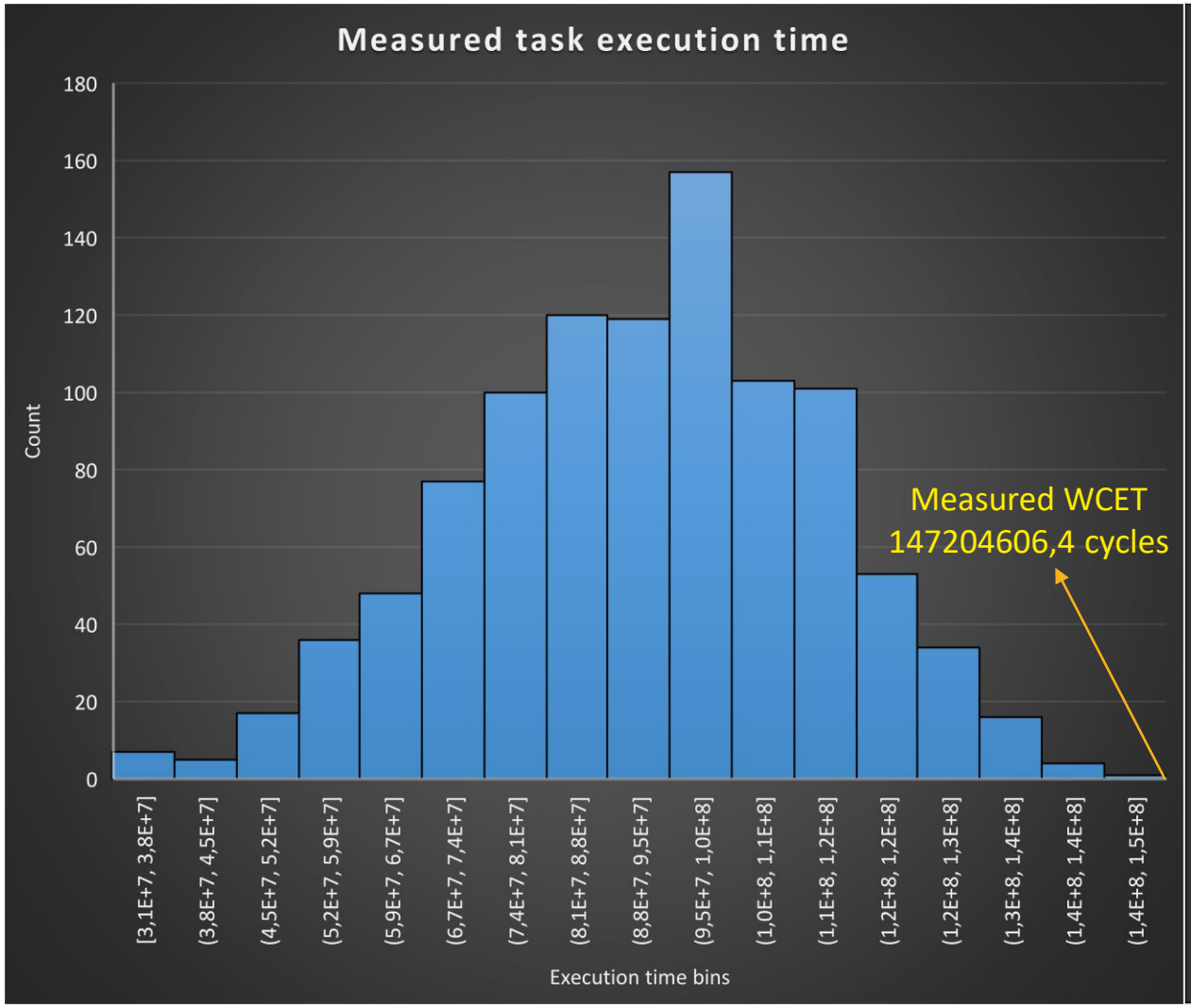
For example,

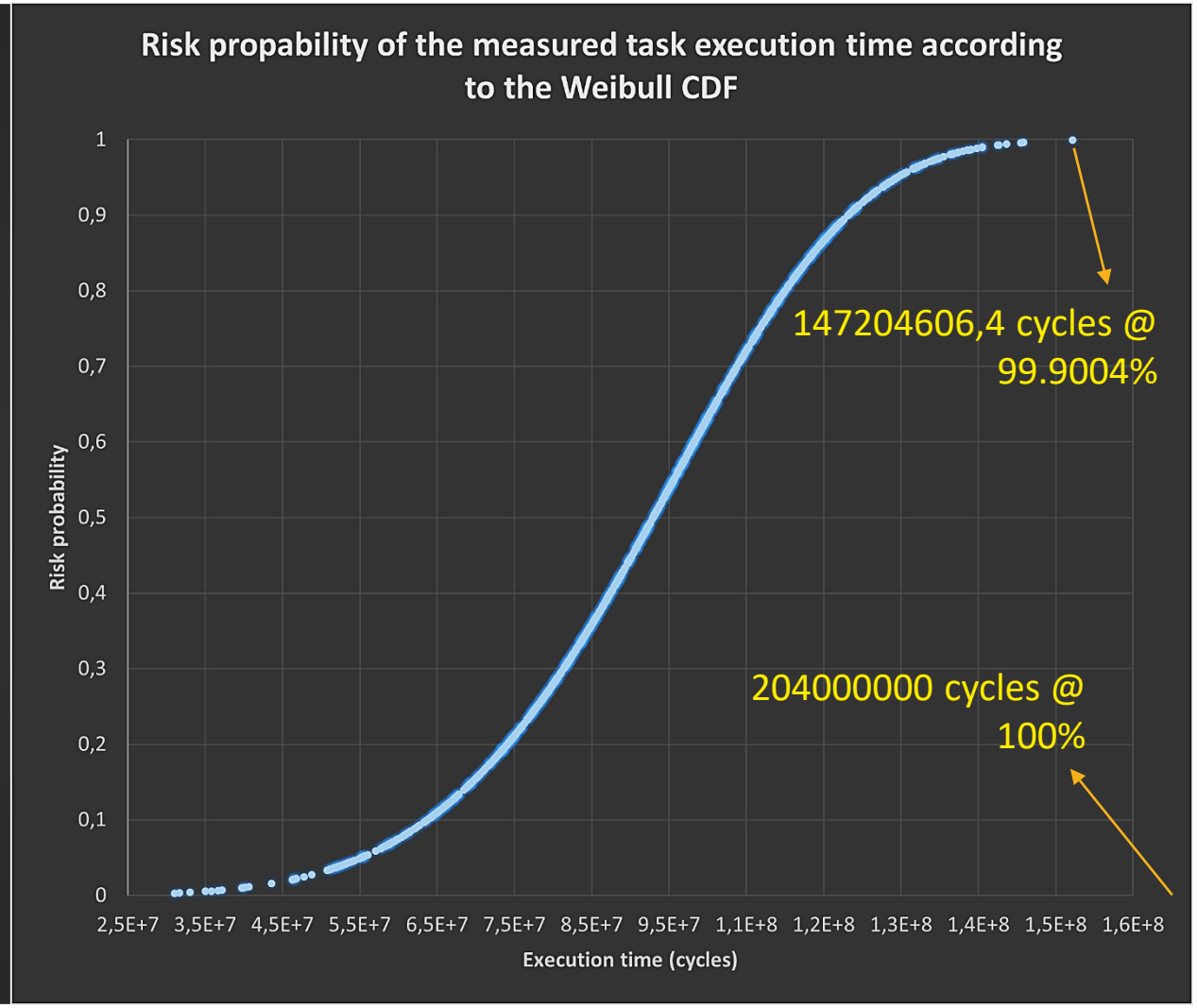
$\boldsymbol{\tau}_i$ is mapped on ARM iff $PE_i \in \{0,1\}$

$\boldsymbol{\tau}_i$ is mapped on DSP iff $PE_i \in \{2,3\}$



On-board profiling: Measurement-based (Probabilistic) Timing Analysis





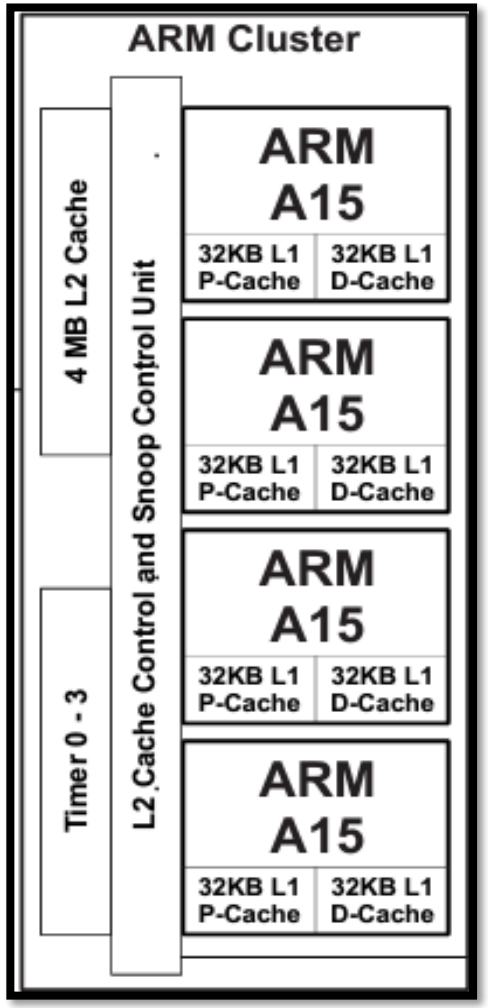
We can probabilistically assure that any future execution is below 204000000

Content

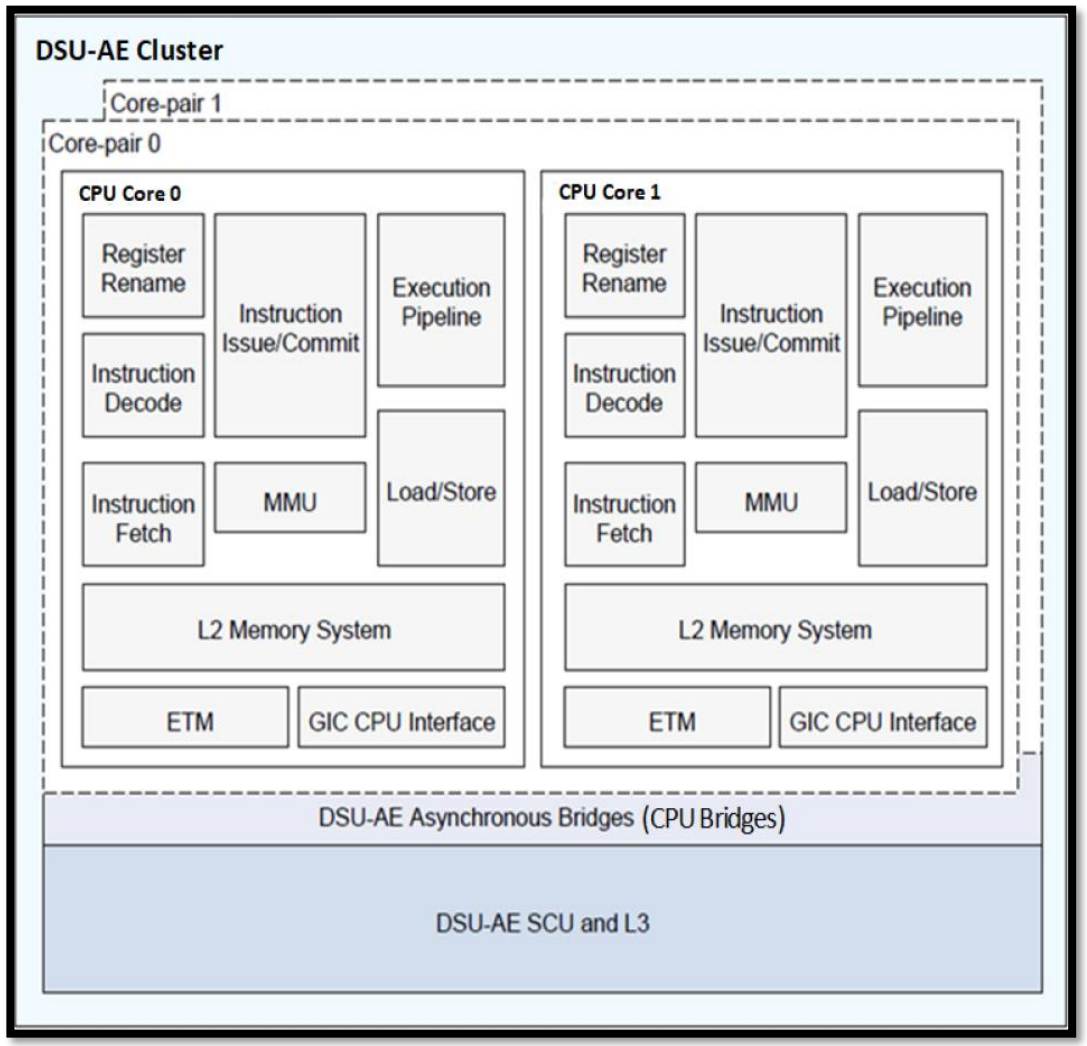
1. Context
2. On-board profiling
3. **Memory interference analysis**
4. Conclusions

Memory interference analysis: Shared cache

ARM Cortex A15 L2 shared cache memory



ARM Cortex A78AE L3 shared cache memory



To analyze:

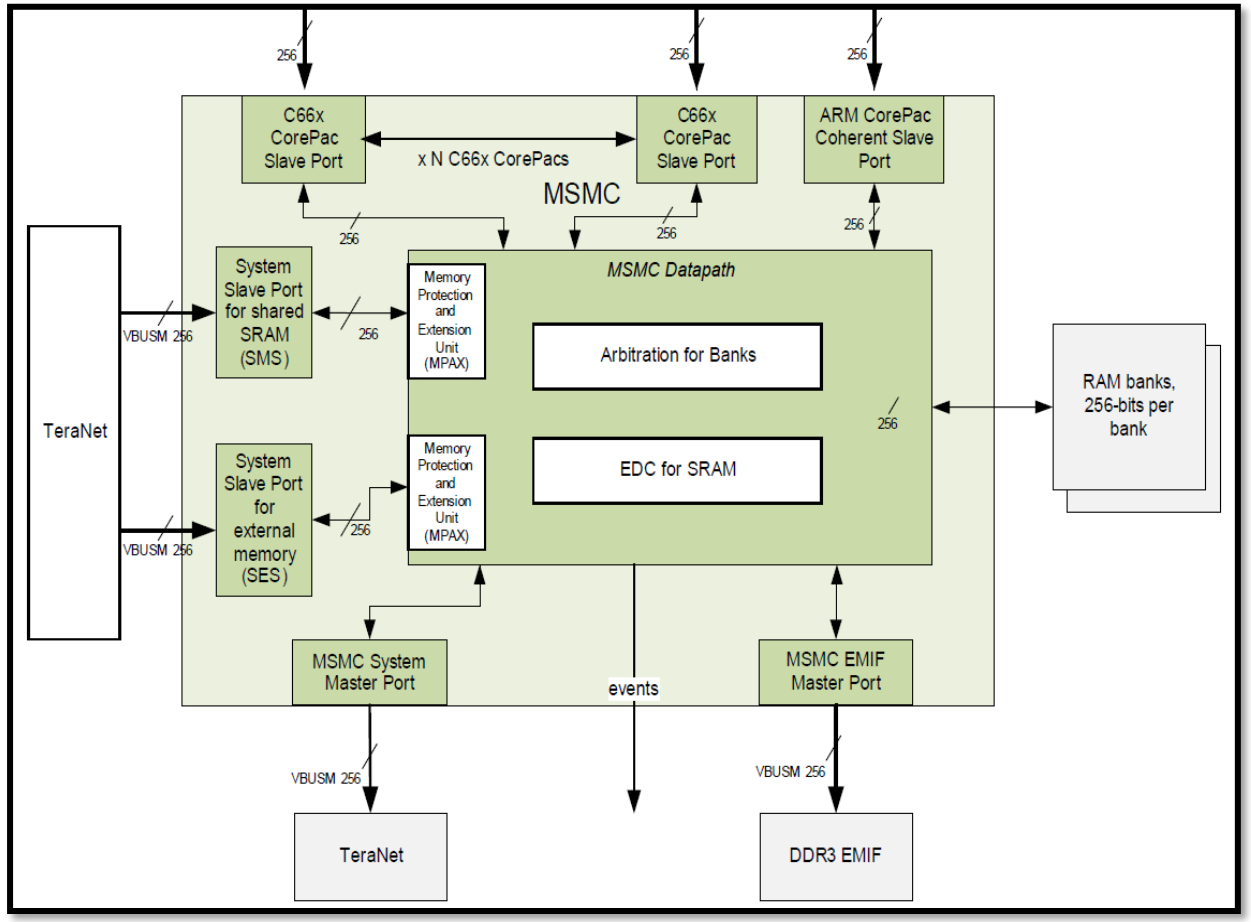
- Execution time overhead
- Nº of L2 cache accesses
- Nº of L2 cache evictions
- Bus accesses
- Etc.

When:

- Default conditions
- Cache partitioning
- Cache locking
- Bandwidth regulator
- Etc.

Memory interference analysis: Multicore Shared SRAM

Keystone II MPSoC Multicore Shared Memory Controller. Note that the ARMs are packed together



To analyze:

- Execution time in isolation
- Execution time overhead
- Effect of slave ports priority
- Effect of slave ports starvation counters
- Etc.

Reverse-engineering:

- Interconnection requests scheduling

Curiosity:

- Hidden registers (e.g., ARM Cortex slave configuration)

Memory interference analysis: Multicore Shared SRAM

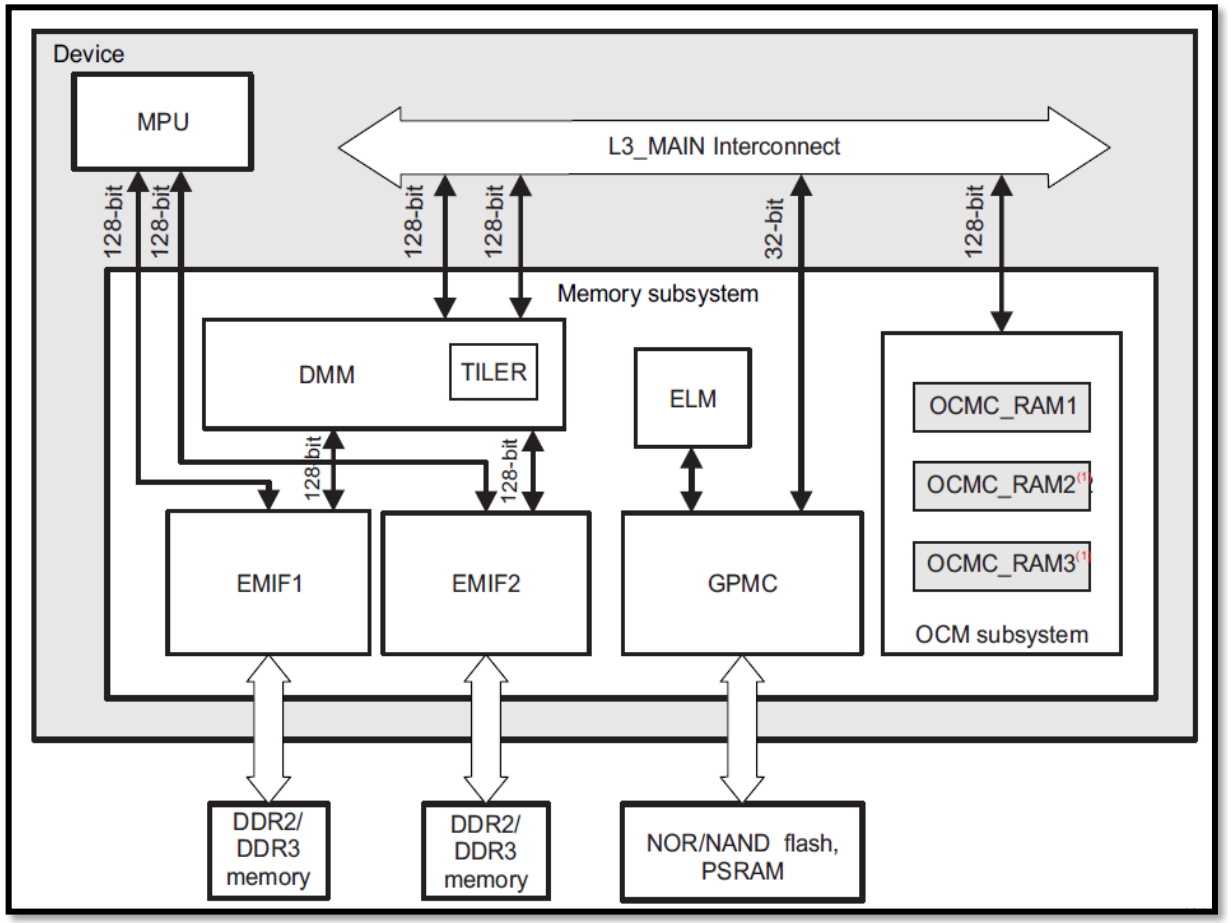
Curiosity:

- On Sitara AM5728, the ARM Cortex A15 takes more time accessing the On Chip RAMs than the DDR SDRAM. According to Texas Instruments, TI Sitara AM5728 ≈ TI DRA74x/75x.

TI DRA74x/75x timer based results (Source: Texas Instruments)

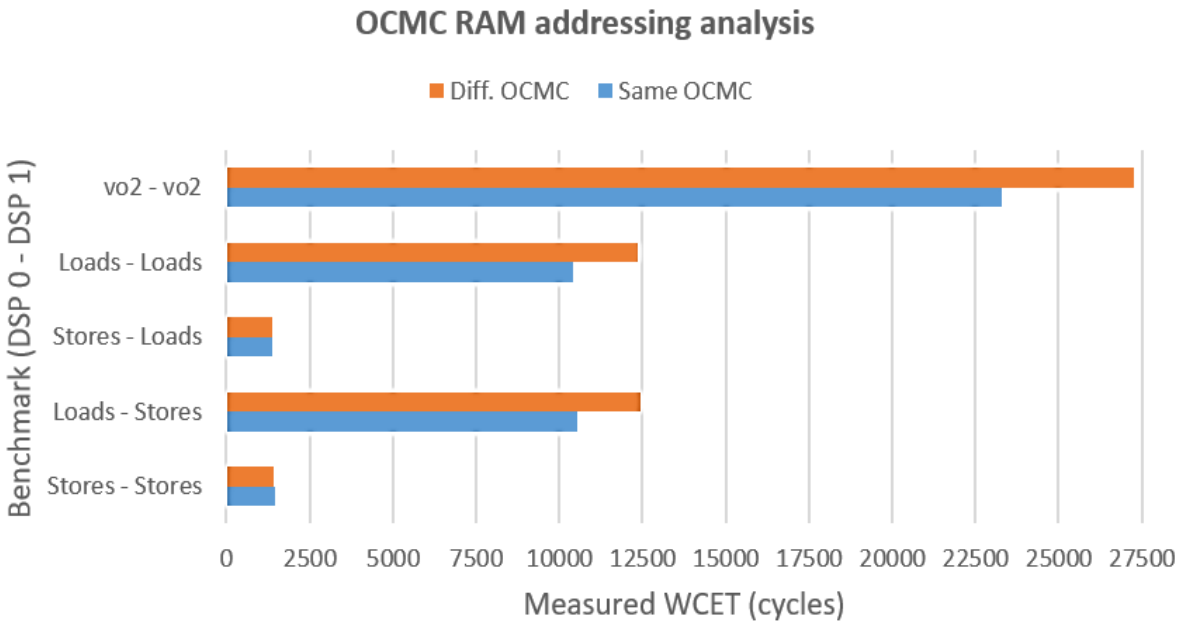
Operations for Cortex-A15	Source	Destination	Bandwidth with Cache Policy - WB WA (MBps)
CPU_WR	CPU Register	DDR	3642
	CPU Register	OCMC	1392
CPU_RD	DDR	CPU Register	2203
	OCMC	CPU Register	2173

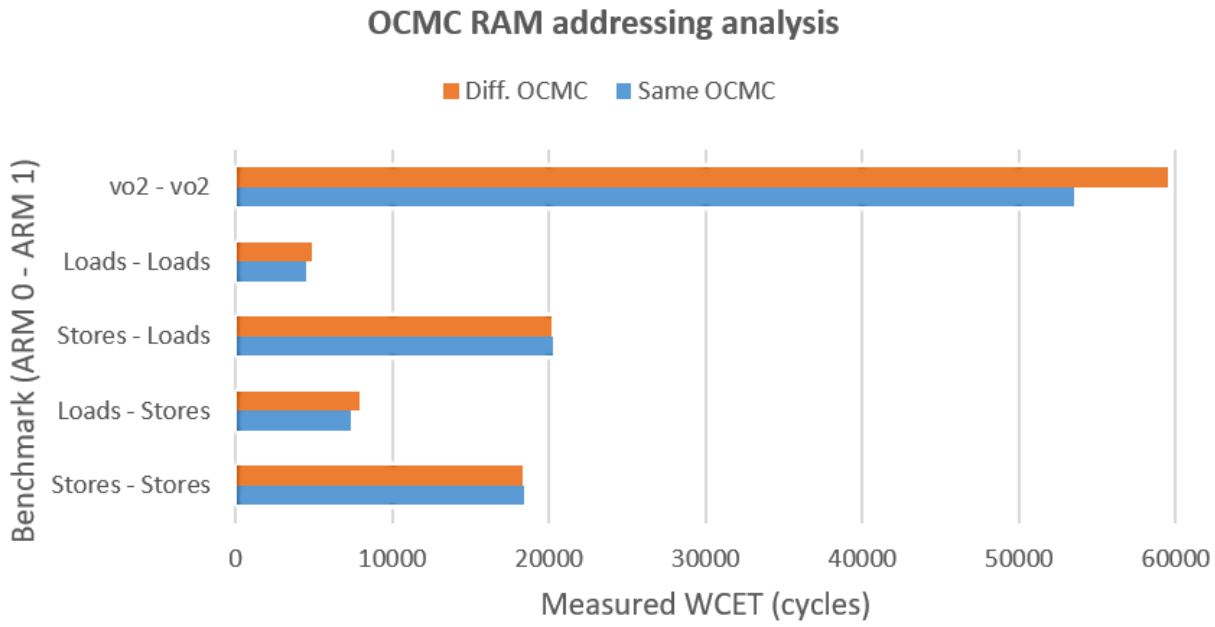
Sitara AM5728 MPSoC Memory Subsystem. The ARMs (MPU) connects directly to the EMIFs



Memory interference analysis: Multicore Shared SRAM

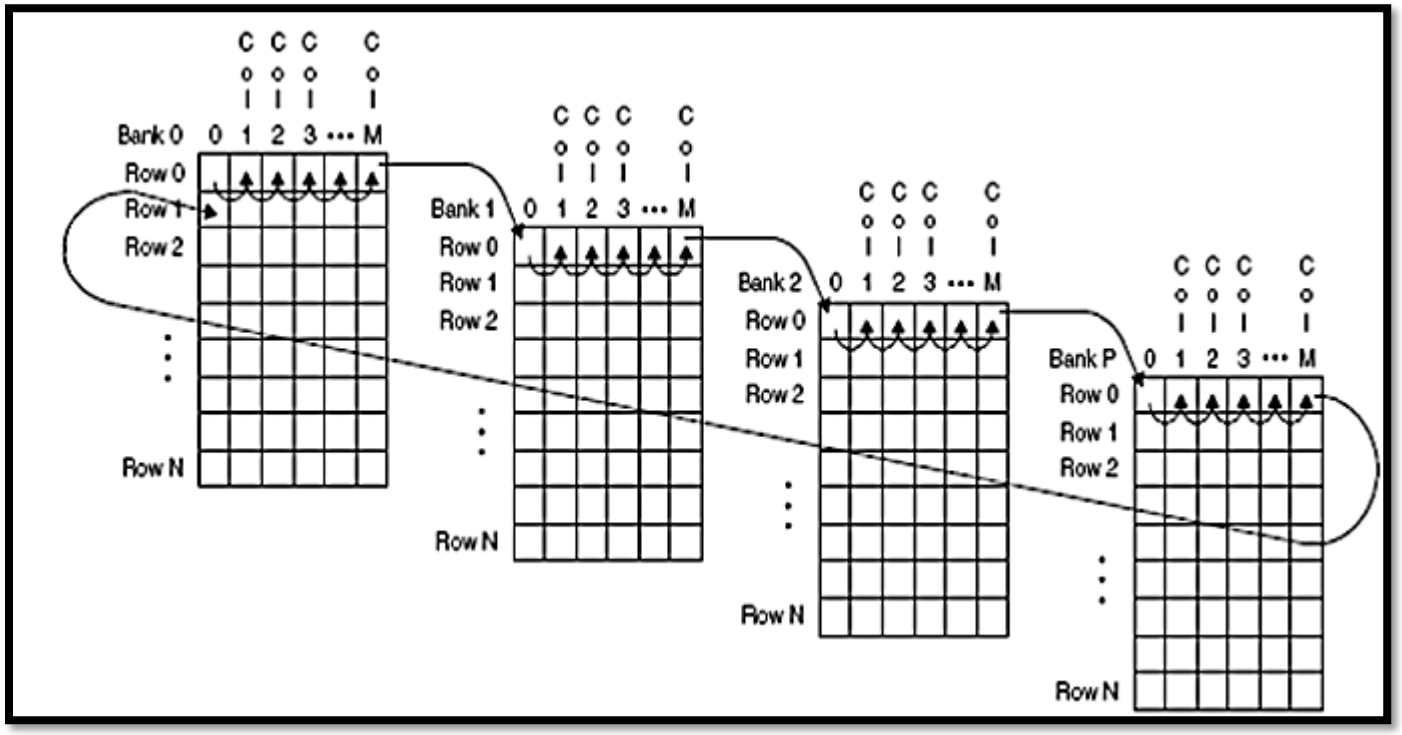
vo2 = Vector operations benchmark
Loads = Stream of reads microbenchmark
Stores = Stream of writes microbenchmark





Memory interference analysis: DDR3 SDRAM

DDR SDRAM logical organization



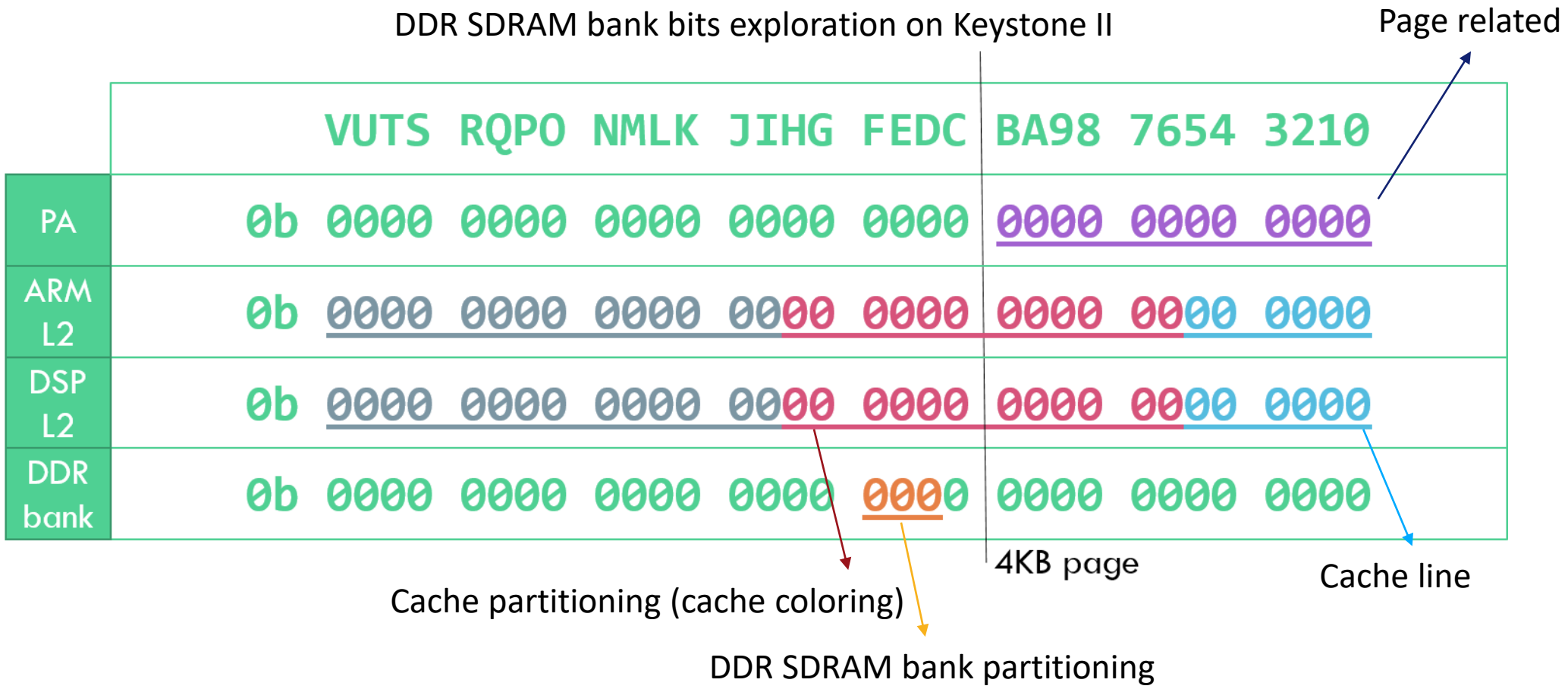
To analyze:

- Execution time in isolation
- Execution time overhead
- Nº of SDRAM accesses
- Nº of SDRAM row switches
- Nº of SDRAM priority elevations
- Etc.

When:

- Default conditions
- Bank partitioning
- Bandwidth regulator
- Etc.

Memory interference analysis: DDR3 SDRAM

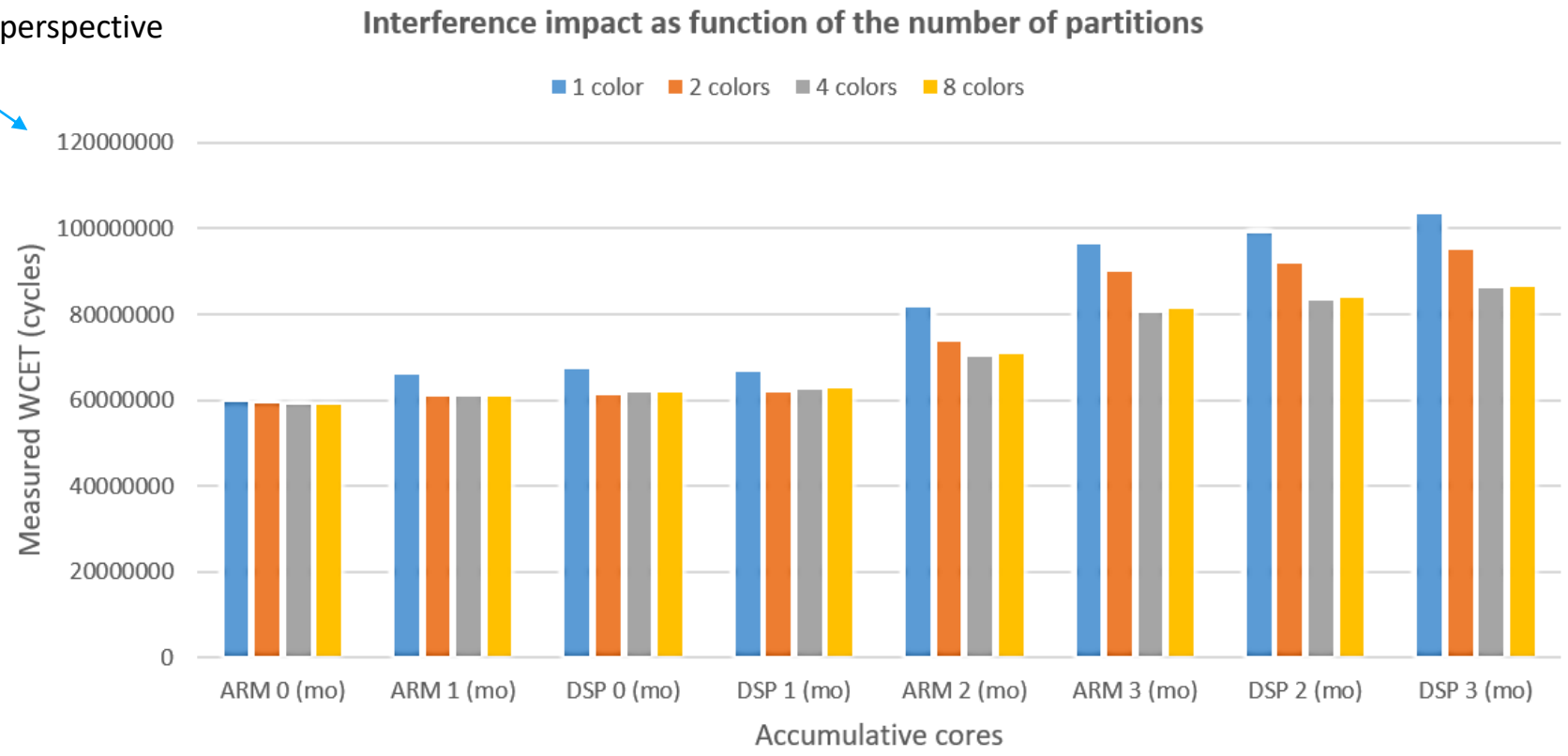


In some SoCs, SDRAM bank bits must be obtained via reverse-engineering. Many times, we will see intersection between cache and bank partitioning bits like in this example.

Memory interference analysis: DDR3 SDRAM & Shared cache

Bank and Cache partitioning effect on Keystone II

Arm Cortex A15 (ARM 0) perspective



Content

1. Context
2. On-board profiling
3. Memory interference analysis
4. **Conclusions**

Conclusions

Use on-board monitoring hardware to measure events for:

- Task profiling
- Interference analysis
- Interference mitigation techniques effectiveness
- Reverse-engineering
- Detect intrusive traffic
- Etc.

On different memory units of the platform:

- L1 Instruction cache / L1 Data cache
- L2/L3/Ln cache
- Shared on-chip SRAM
- DDRx SDRAM

Thank you for your attention