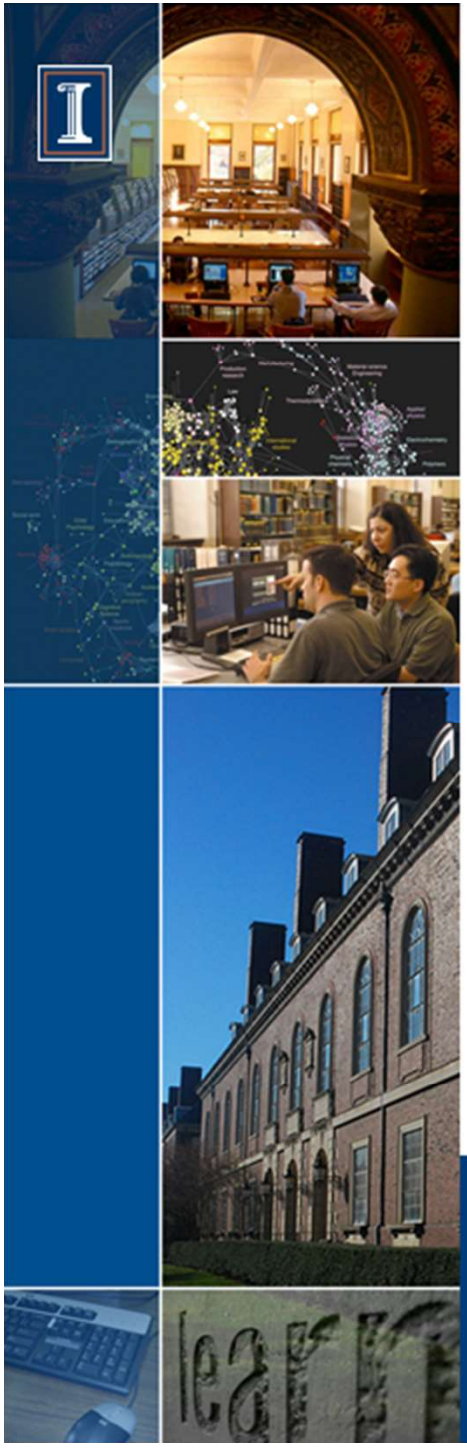


A Real-time Reliability Challenge:

Reduce Interactive Complexity
Reduce Coupling

A Multicore Case Study

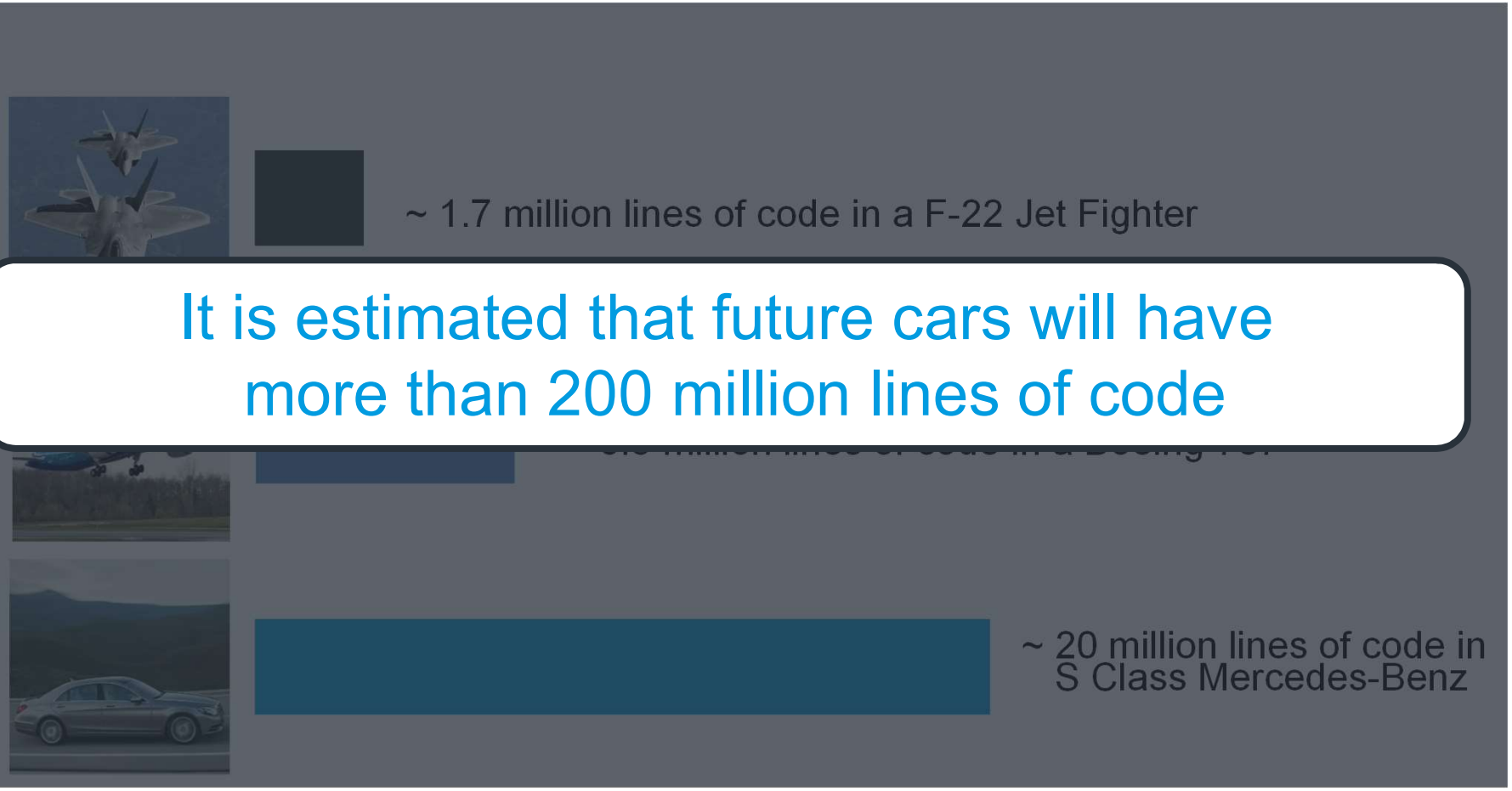
By: Rohan Tabish

Department of Computer Science
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Reminders of Definitions

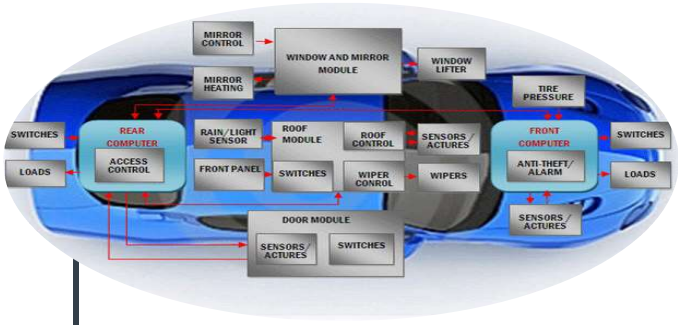
- Interactive complexity
  - ➢ Unexpected interactions between seemingly correct components (e.g., between one CPU core and another)
- Coupling
  - ➢ Fast propagation of effects of failure from one component to another

The Size of the Software Complexity Problem



It is estimated that future cars will have more than 200 million lines of code

The Software Complexity Problem: Where is this code?



infotainment

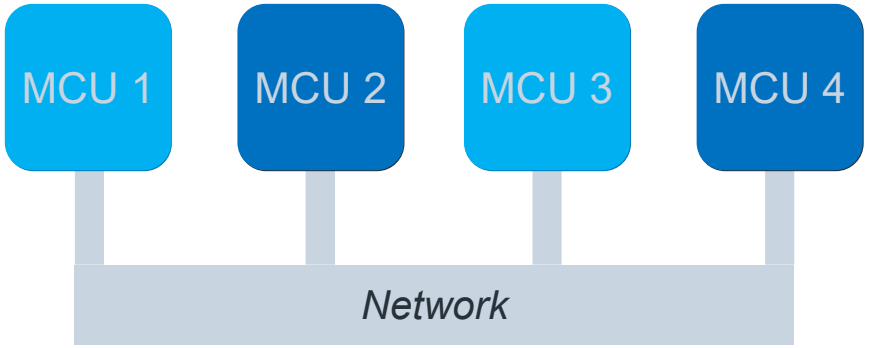
engine control

body

= tens of interconnected MCUs

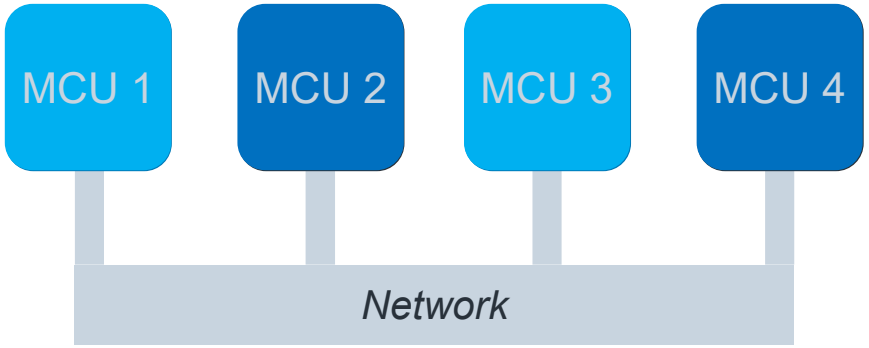
Migrating to Multicore has Benefits

From **multiple** single-core systems

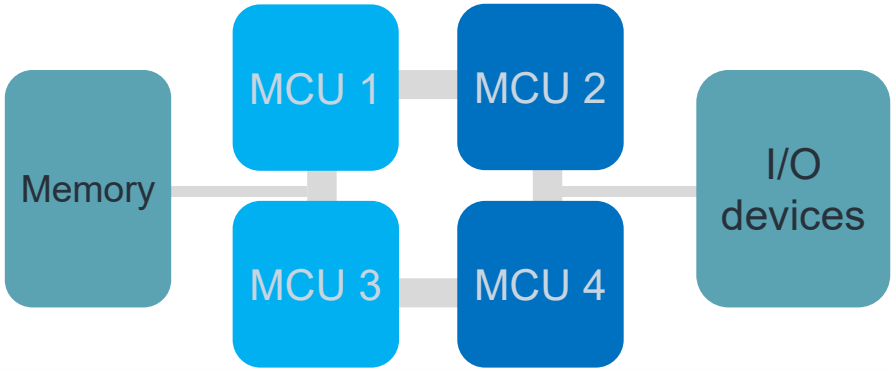


Migrating to Multicore has Benefits

From **multiple** single-core systems

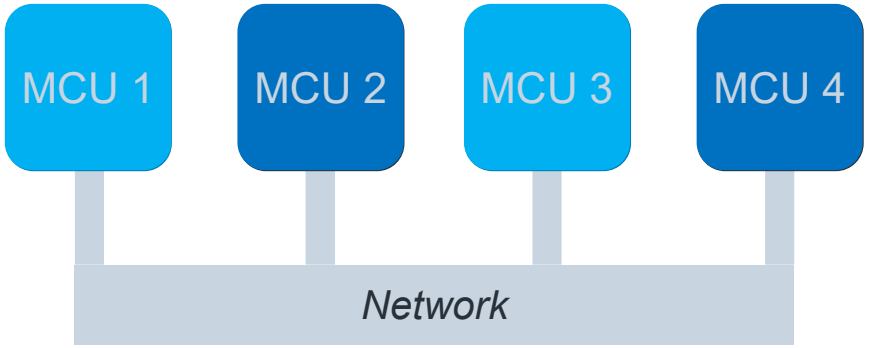


To a **single** multi-core system

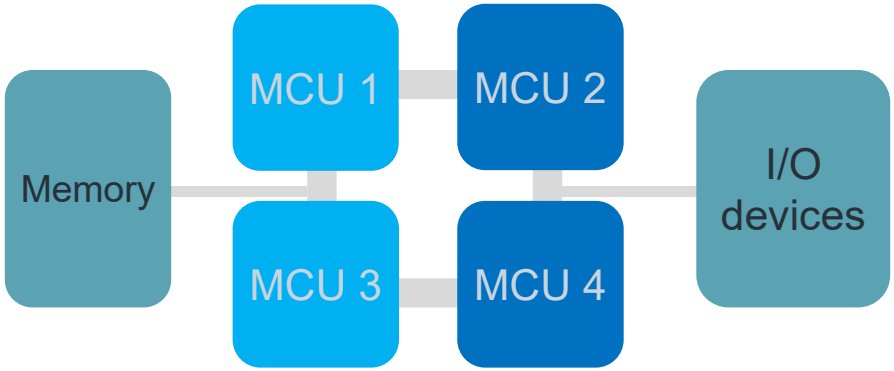


Migrating to Multicore has Benefits

From **multiple** single-core systems



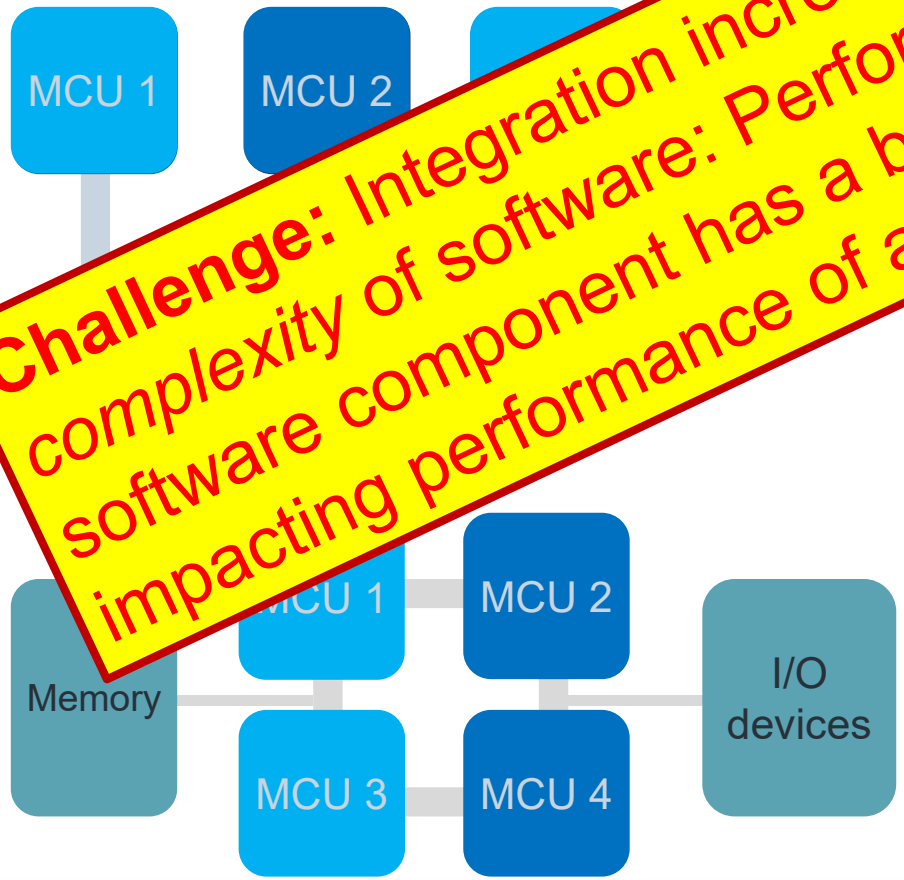
To a **single** multi-core system



- ✓ Integration
- ✓ Cost reduction

Migrating to Multicore I… enefits

From **multiple** single-core …



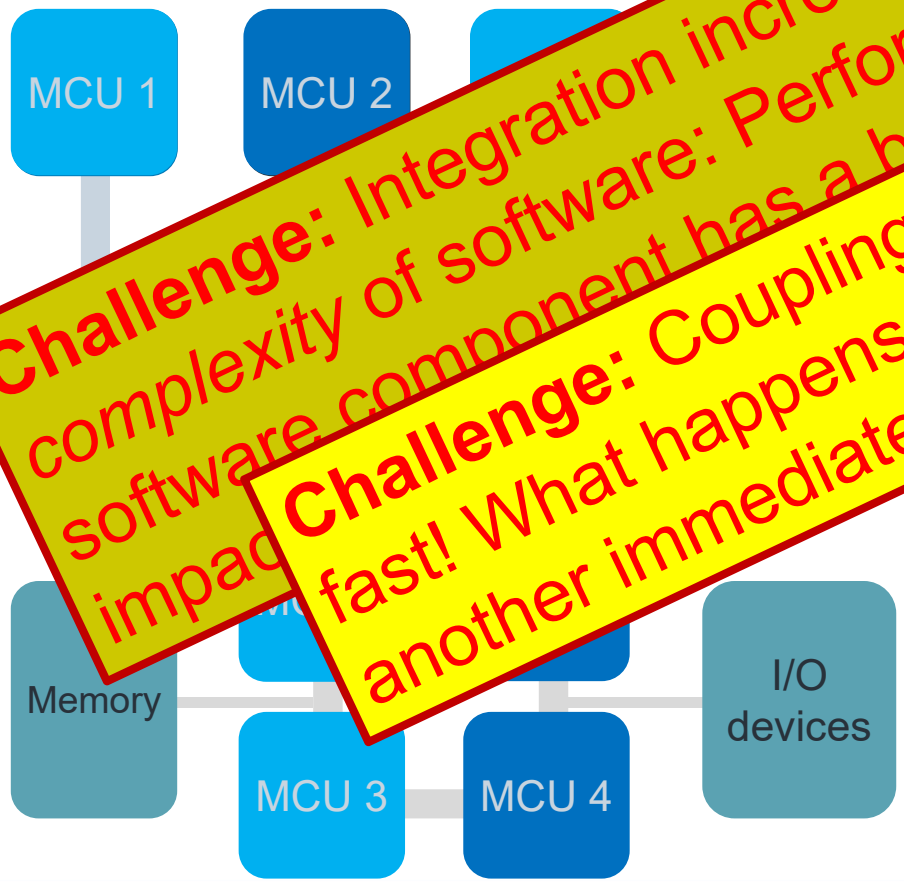
Challenge: Integration increases interactive complexity of software: Performance of one software component has a better chance of impacting performance of another

✔ Integration

✔ Cost reduction

Migrating to Multicore I… enefits

From **multiple** single-core …



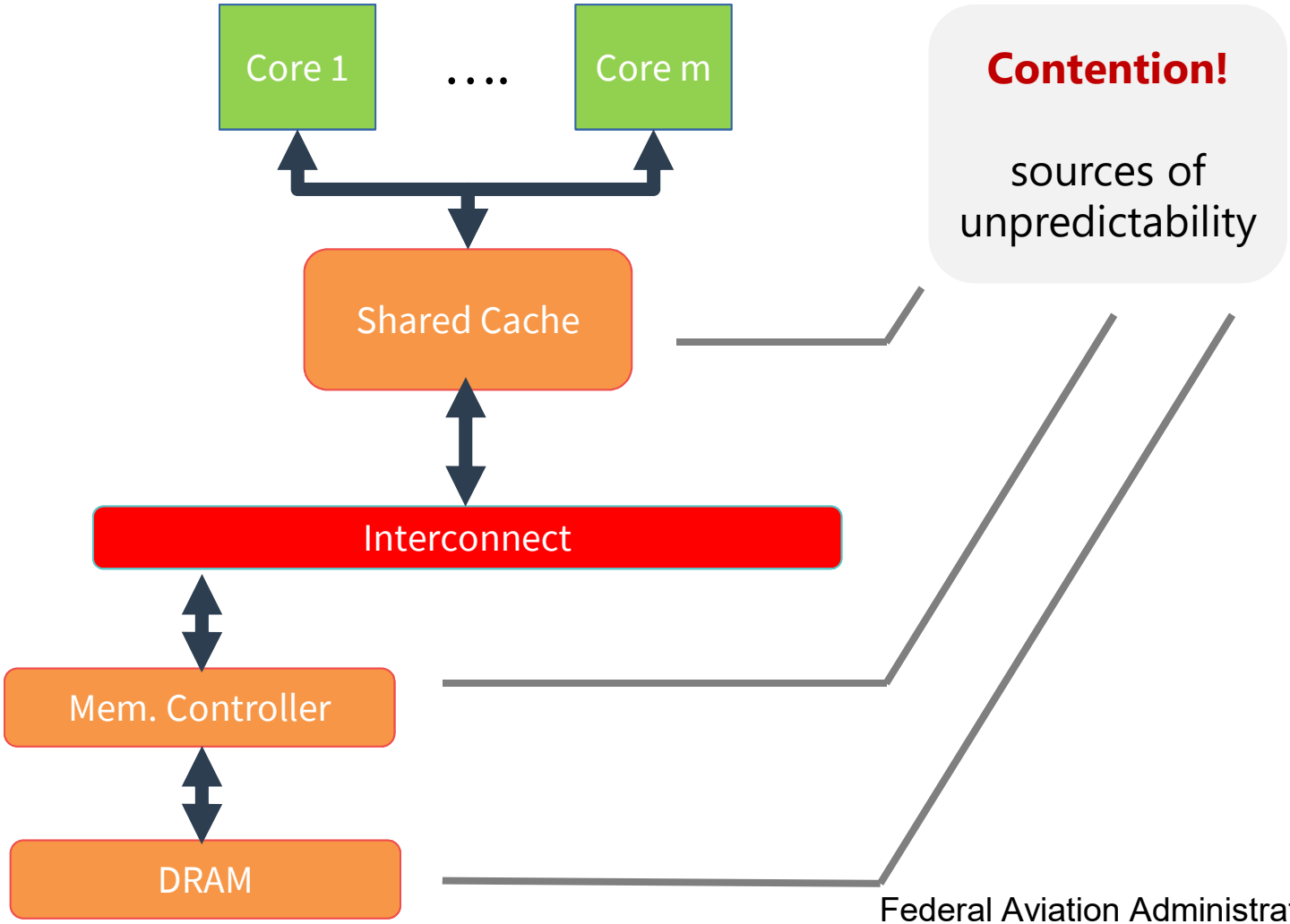
Challenge: Integration increases interactive complexity of software: Performance of one software component has a better chance of impac…

Challenge: Coupling among cores is very fast! What happens on one core can impact another immediately!

…gration

Cost reduction

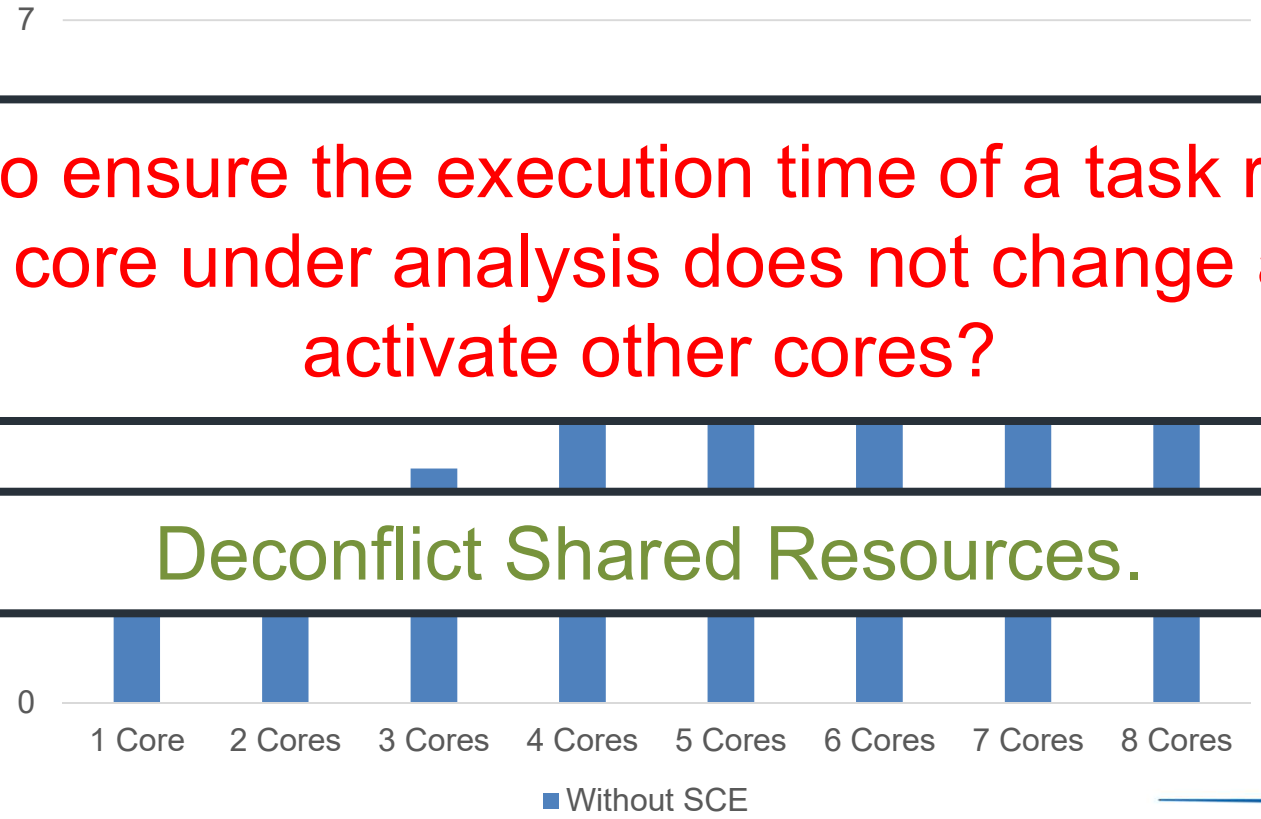
Use of Multicore Chips Is Regulated By FAA



Federal Aviation Administration (FAA)

An (Undesirable) Interaction Problem:

Execution Time Changes as Multiple Cores are Activated



How to ensure the execution time of a task running on a core under analysis does not change as we activate other cores?

Deconflict Shared Resources.

LOCKHEED MARTIN

Practical Challenge: How to Decouple the Cores?

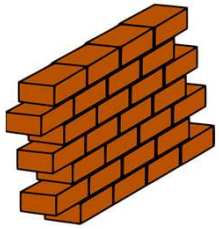
- Class discussion: How would you do it?
  - ➢ Understand what causes the interaction.
  - ➢ Propose solutions to remove it.
  - ➢ Try not to hurt performance too much
    - Note: Remember the fundamental trade-off between performance and reliability! Reducing interactive complexity and coupling prevents bad interactions, thereby increasing reliability. However, this may come at the expense of performance.

Practical Challenge: How to Decouple the Cores?

Note: Remember the fundamental trade-off between performance and reliability! Reducing interactive complexity and coupling prevents bad interactions, thereby increasing reliability. However, this may come at the expense of performance.

Isolation:

"A" has resources



"B" is hungry

- **"B" can't use A's resources:**
  Good isolation! Independent failures ☺
- **"B" might not operate well (starve)**
  (possible inefficiency) ☹

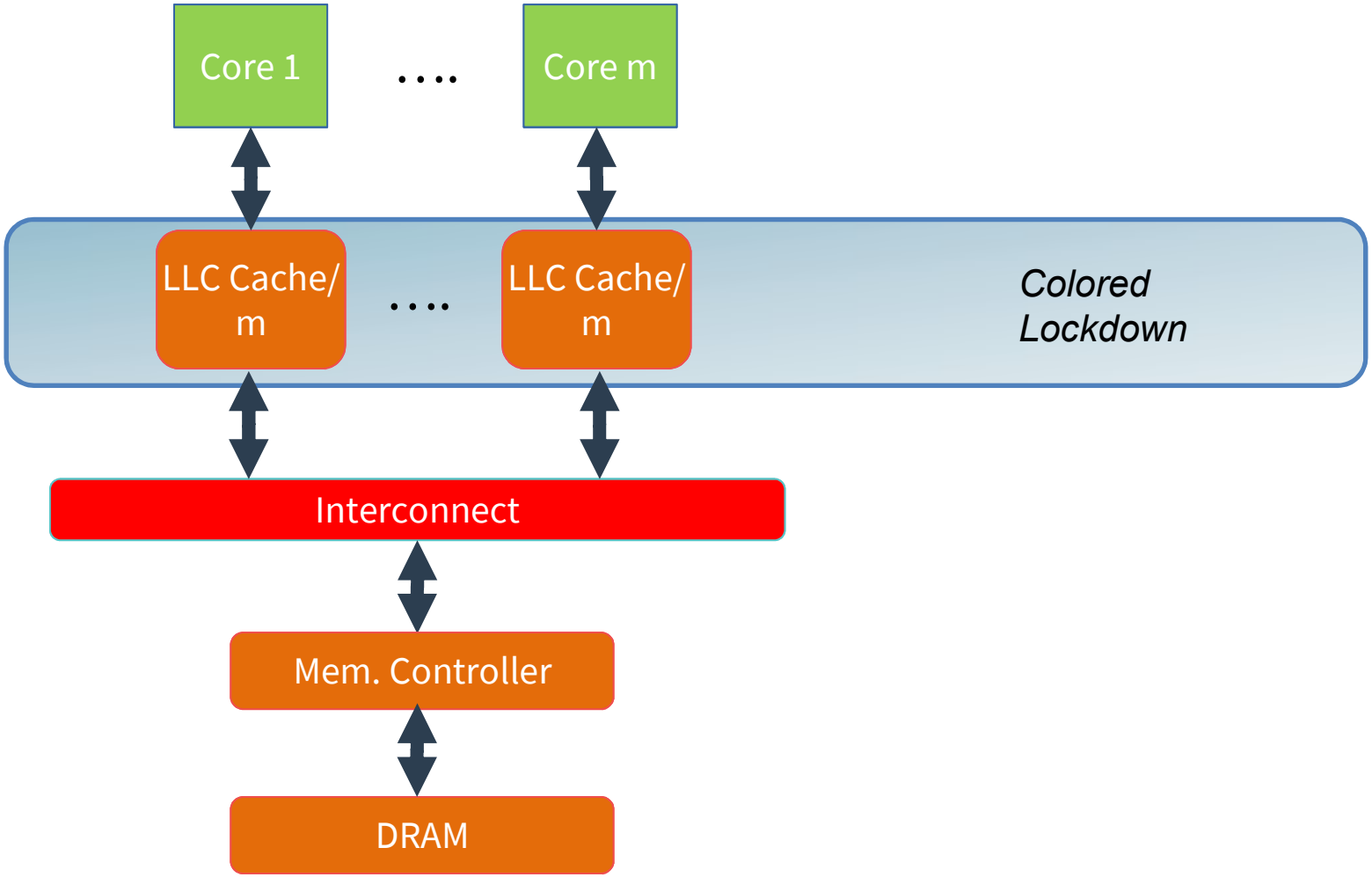
Sharing:

- **Whoever needs the resource can have it**
  Improved collective performance! ☺
- **"A" and "B" can interfere with each other**
  Higher interactive complexity/coupling ☹

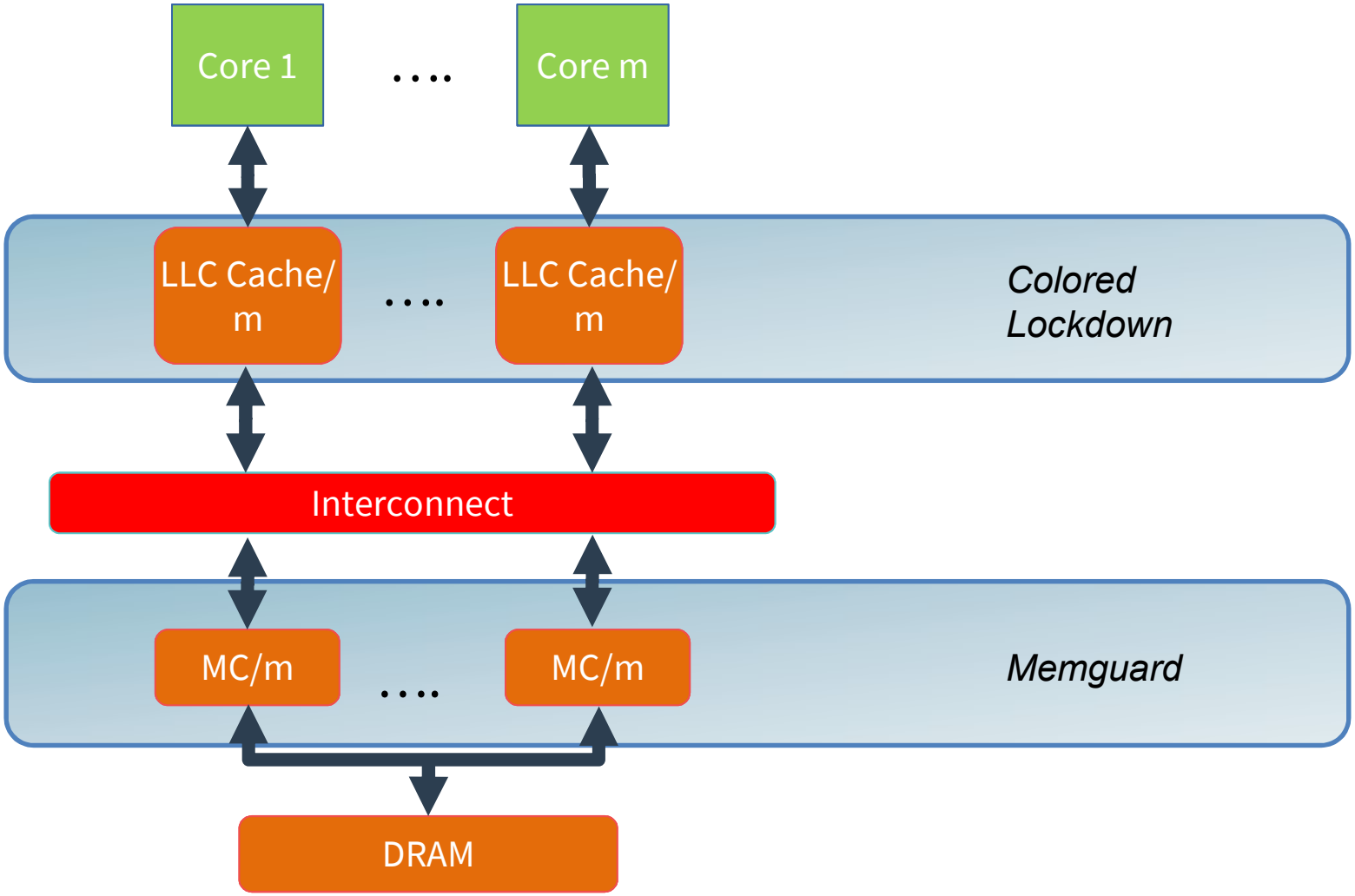
Solutions for Isolation in Multicore Systems

- Reliability comes first!

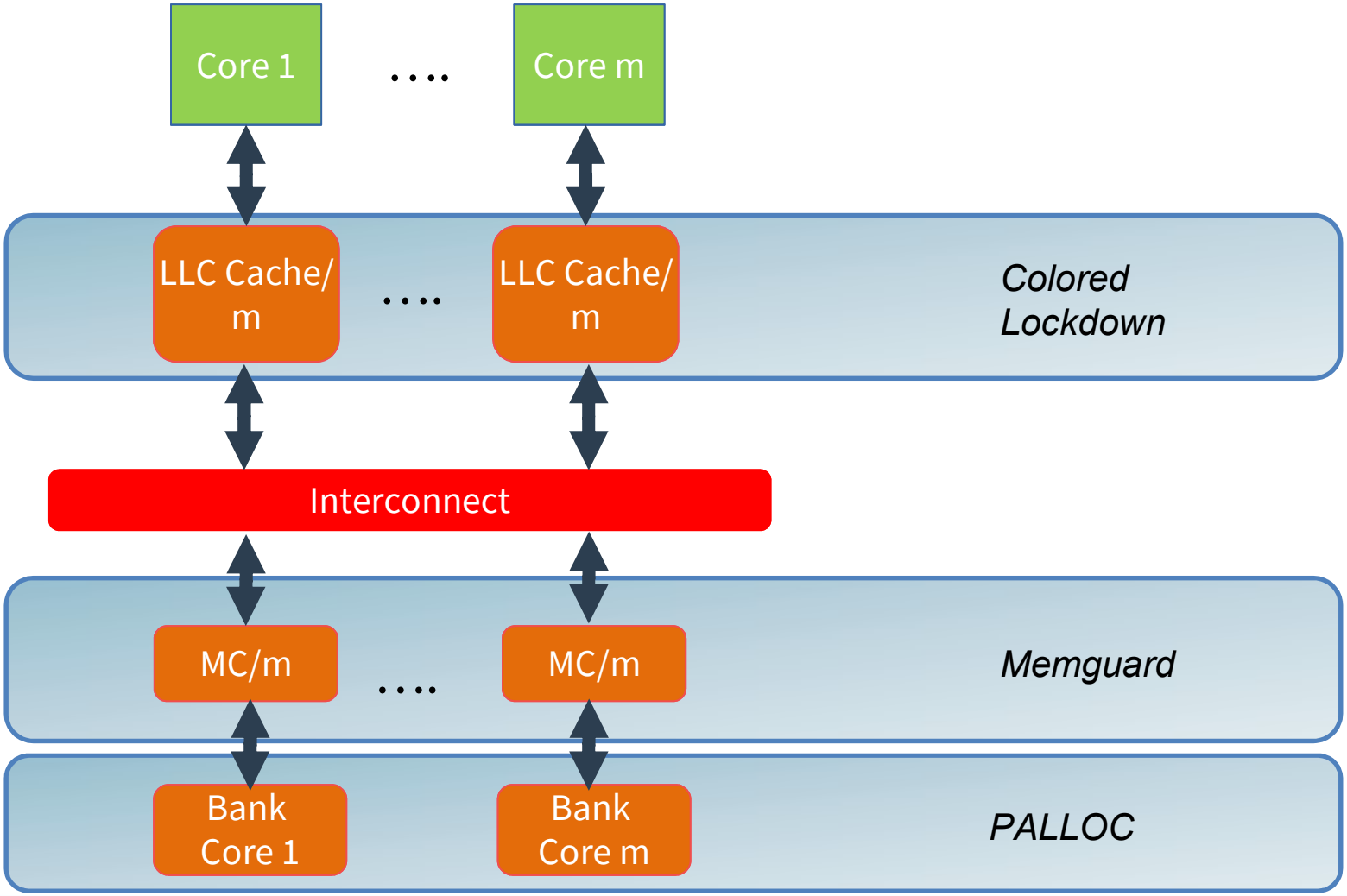
Partition the LLC and Lock Hot Pages



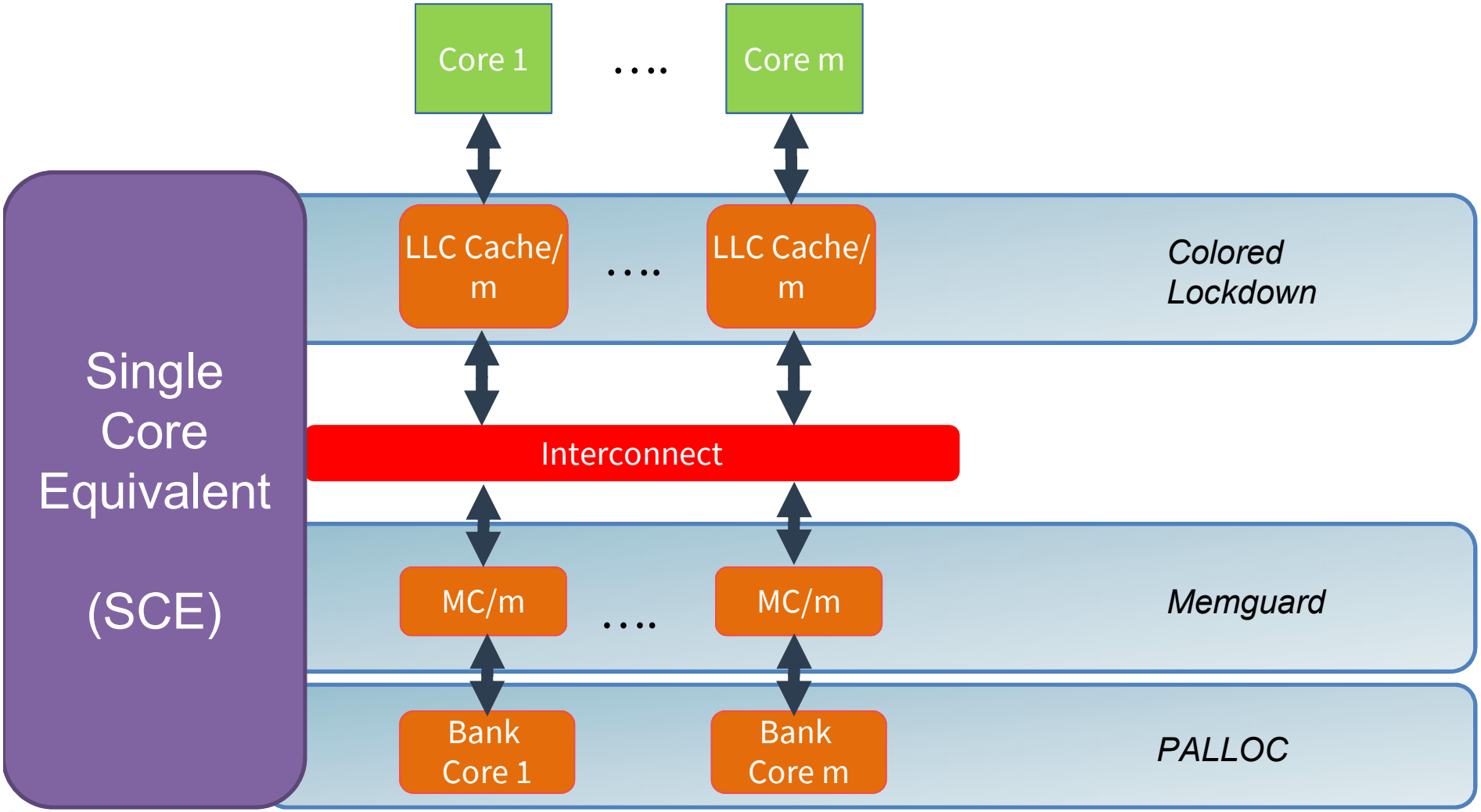
Split the BW of MC Equally Among Cores



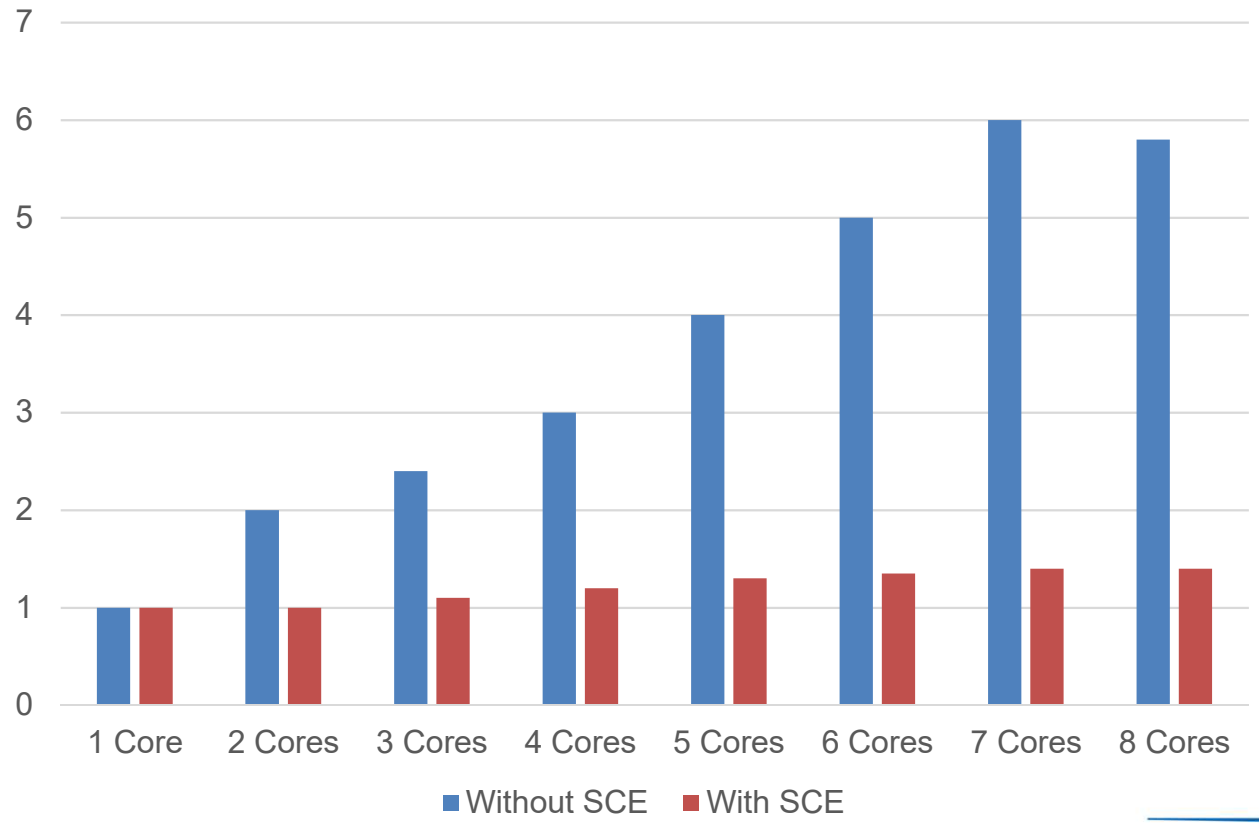
Assign each Core a Dedicated BANK



SCE: Deconflicts shared Resources



WCET of Task With and Without SCE



LOCKHEED MARTIN

Discussion

- What is the effect of the proposed solutions on reliability/predictability?
- What is the effect of the proposed solutions on interactive complexity and coupling?
- What is the effect of the proposed solutions on *average* performance (including less critical applications)?

References

- Mancuso, Renato, et al. "Real-time cache management framework for multi-core architectures." *2013 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2013.
- Yun, Heechul, et al. "PALLOC: DRAM bank-aware memory allocator for performance isolation on multicore platforms." *2014 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2014.
- Yun, Heechul, et al. "Memguard: Memory bandwidth reservation system for efficient performance isolation in multi-core platforms." *2013 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2013.
- Sha, Lui, et al. *Single core equivalent virtual machines for hard real—time computing on multicore processors*. 2014.