# Understanding Hardware Interference Channels in Multicore

Computer architecture basics for multicore real-time systems

Heechul Yun
Associate Professor
University of Kansas
https://www.ittc.ku.edu/~heechul

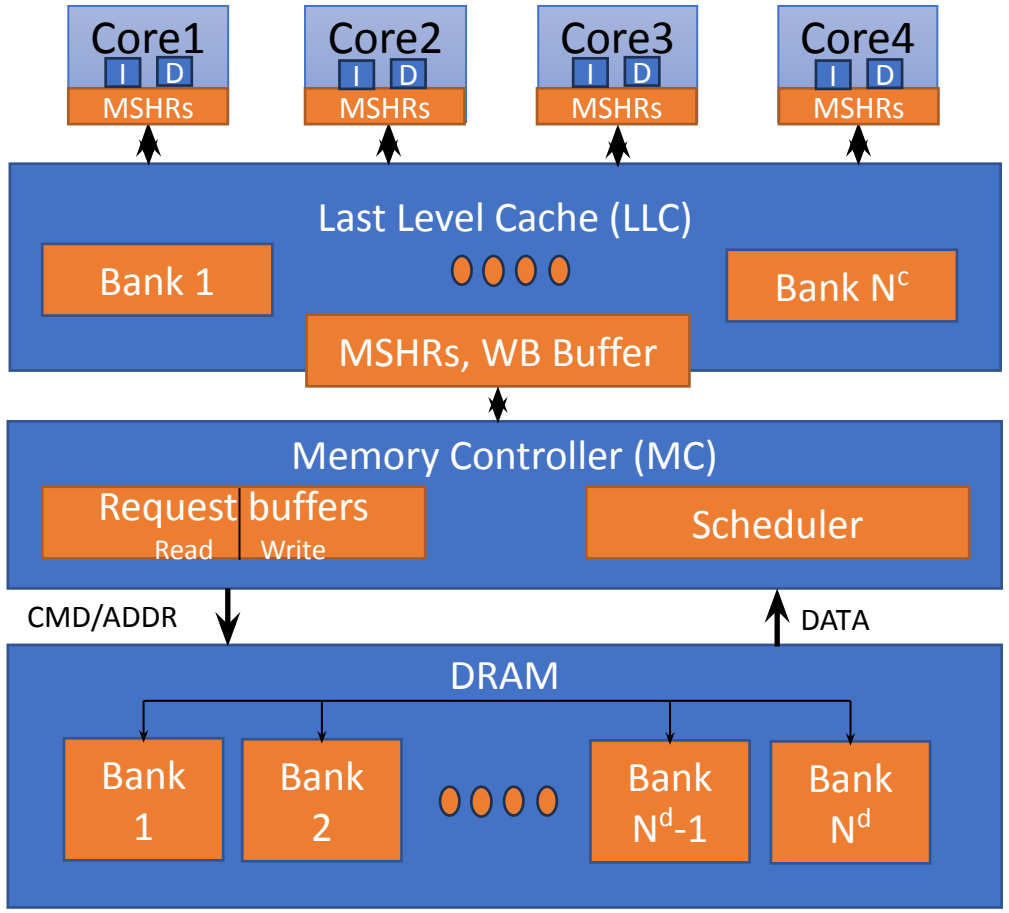
# Agenda

- Memory-level parallelism
- Split-transaction bus
- Non-blocking cache
- Banked cache and DRAM organization
- Memory controller
- Effective “attack” strategies to cause massive cross-core interference

# Memory-level parallelism (MLP)

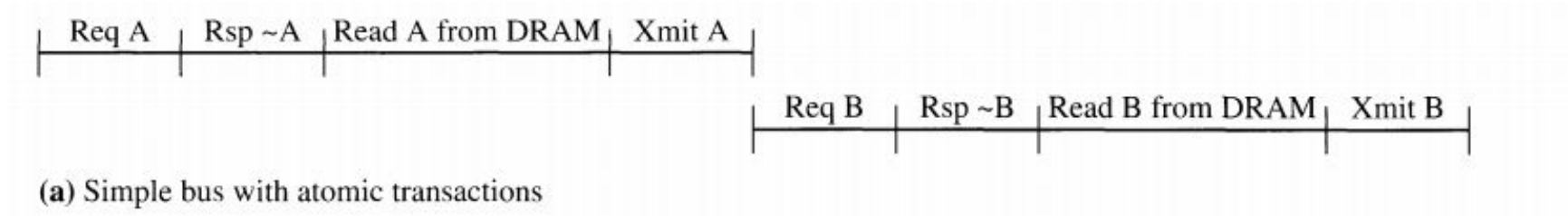
- MLP is the key to understand modern multicore processors (MCP)
  - essential for performance (throughput)
- A core can request multiple concurrent memory accesses at a time
  - times the number of cores (and accelerators)
- Interconnect (bus) supports split-transactions
  - multiple outstanding transactions can occur simultaneously
- Non-blocking cache can handle multiple outstanding cache misses
  - it can continue to serve hits under multiple misses
- Cache and DRAM are composed of multiple independent resources
  - cache/dram banks can be accessed simultaneously in parallel

# Memory-level parallelism (MLP)

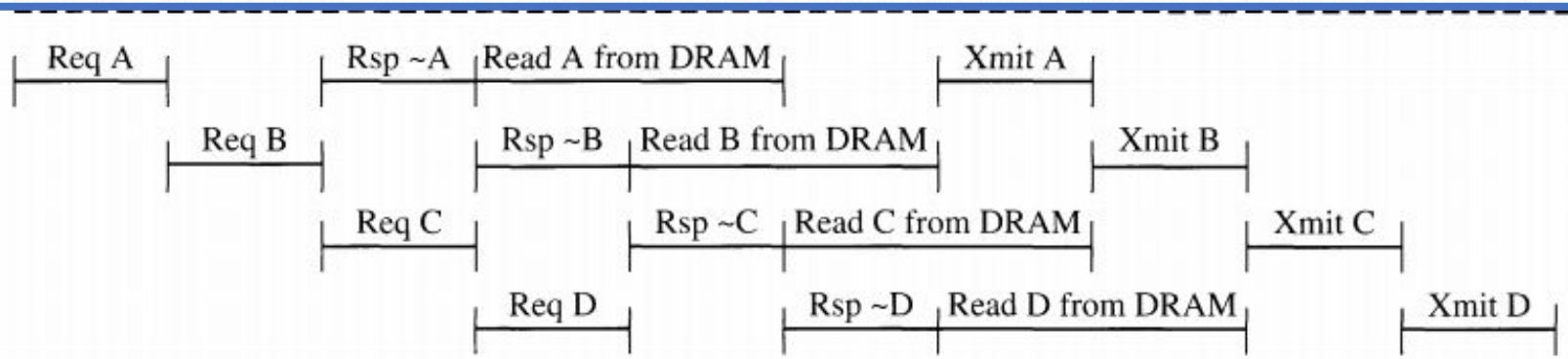



- Out-of-order core:
  Multiple memory requests
- Non-blocking caches:
  Multiple cache-misses
- Memory controller:
  Request buffering, re-ordering
- DRAM:
  Multiple banks serve multiple requests

# Split-transaction bus




(a) Simple bus with atomic transactions




(b) Split-transaction bus with separate requests and responses

A split-transaction bus enables higher throughput by pipelining requests, responses, and data transmission.

**Figure 11.9**
Simple Versus Split-Transaction Busses.

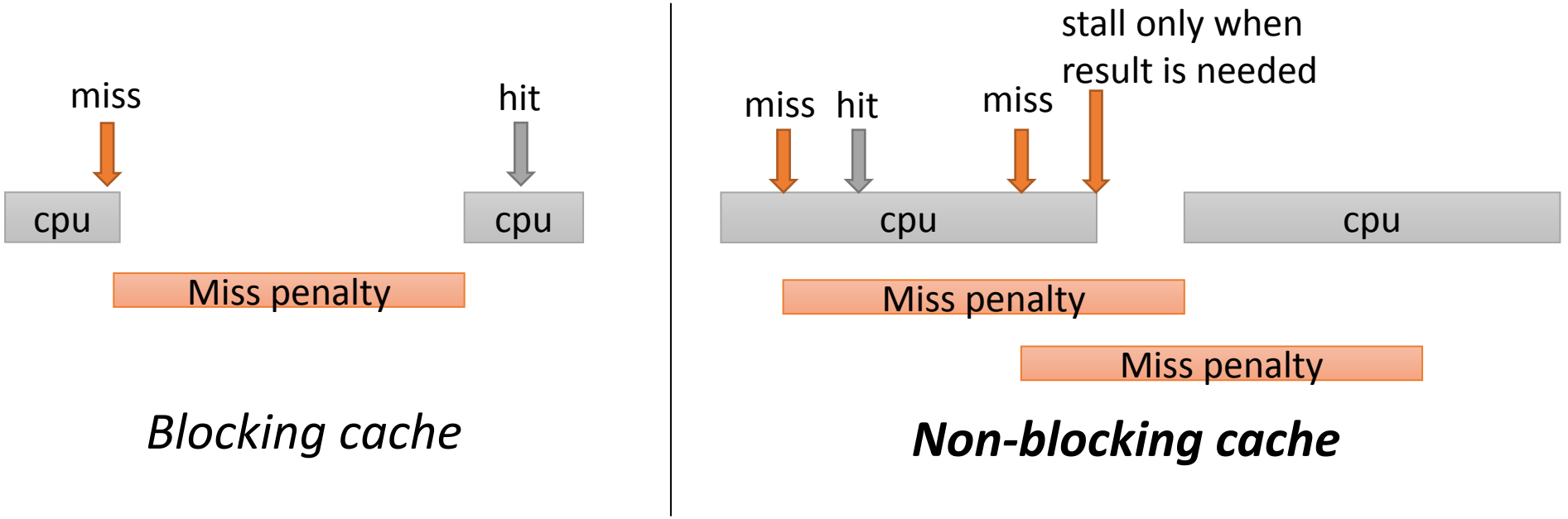
Interconnect is usually not a bottleneck

Figure source: John Paul Shen and Mikko H Lipasti. “Modern processor design: fundamentals of superscalar processors.” Waveland Press, 2013

# Non-blocking cache

- A core can generate multiple simultaneous accesses to a cache
- Multiple cores/accelerators can simultaneously access a shared cache
- So, a shared cache can get lots of parallel requests
- A non-blocking shared cache is essential for performance

# Non-blocking cache



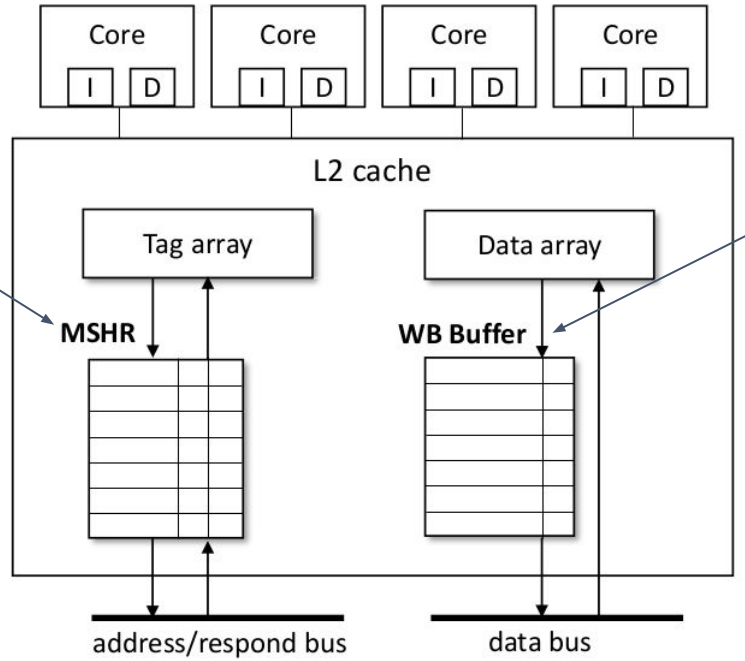

- Can serve cache hits under multiple cache misses
- Essential for performance in multicore

D. Kroft. “Lockup-free instruction fetch/prefetch cache organization,” ISCA’81

# Non-blocking cache

**Miss Status Holding Registers**

- Track outstanding cache misses.
- Allow high memory-level parallelism




**Writeback Buffer.**

- Hold evicted dirty lines (writebacks).
- Prevent cache refills from waiting.

- Cache internal structures are **potential interference channels**

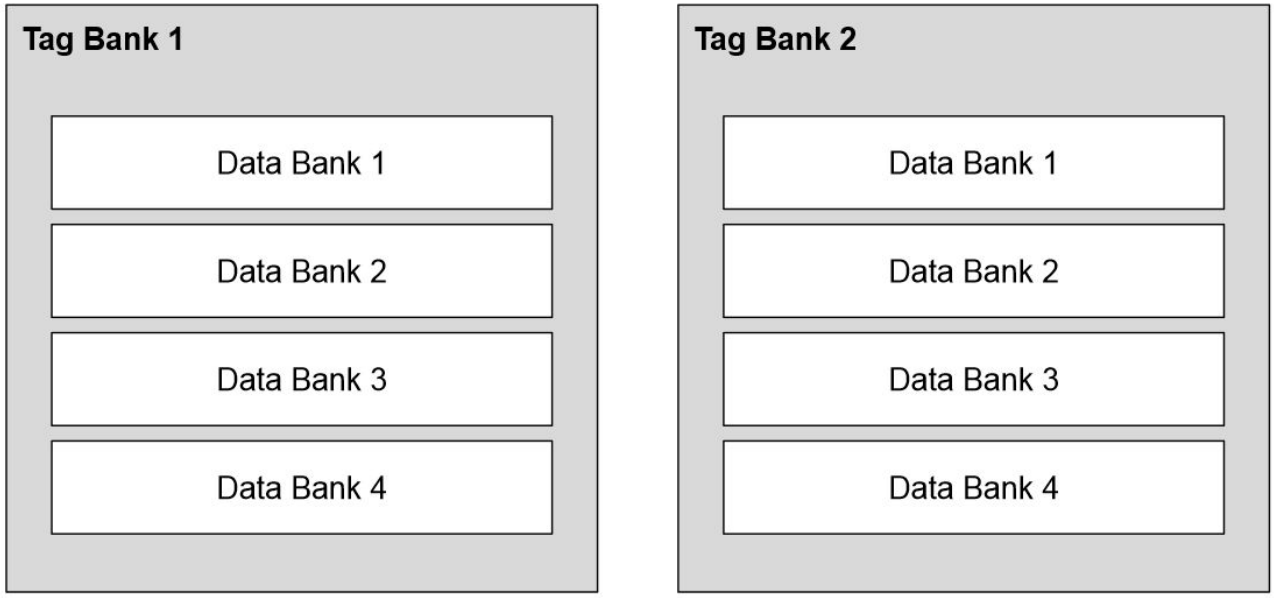
Prathap Kumar Valsan, Heechul Yun, Farzad Farshchi. "Taming Non-blocking Caches to Improve Isolation in Multicore Real-Time Systems." In *IEEE RTAS*, 2016 **(Best Paper Award)**
Michael G. Bechtel and Heechul Yun. "Denial-of-Service Attacks on Shared Cache in Multicore: Analysis and Prevention." In *IEEE RTAS*, 2019 **(Outstanding Paper Award)**

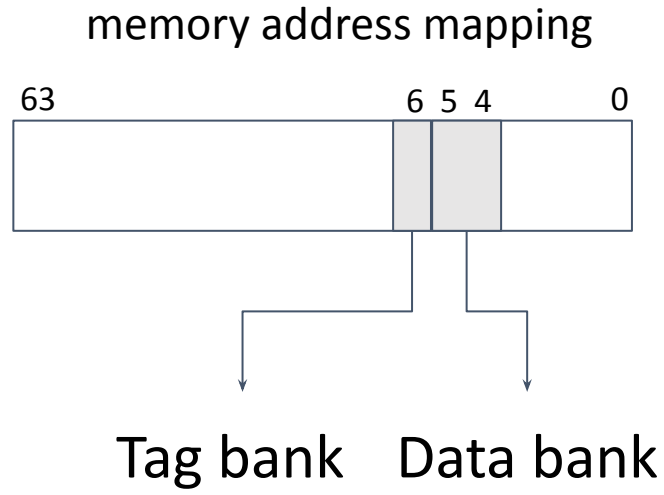
# Multi-bank cache/DRAM organizations

- Shared cache and DRAM are not a single resource
- Each is composed of multiple resources---banks
- Banks are (largely) independent and can be accessed in parallel
- Generally, **more banks = more parallelism/throughput**

# Cache bank organization

- Multiple banks can be accessed simultaneously

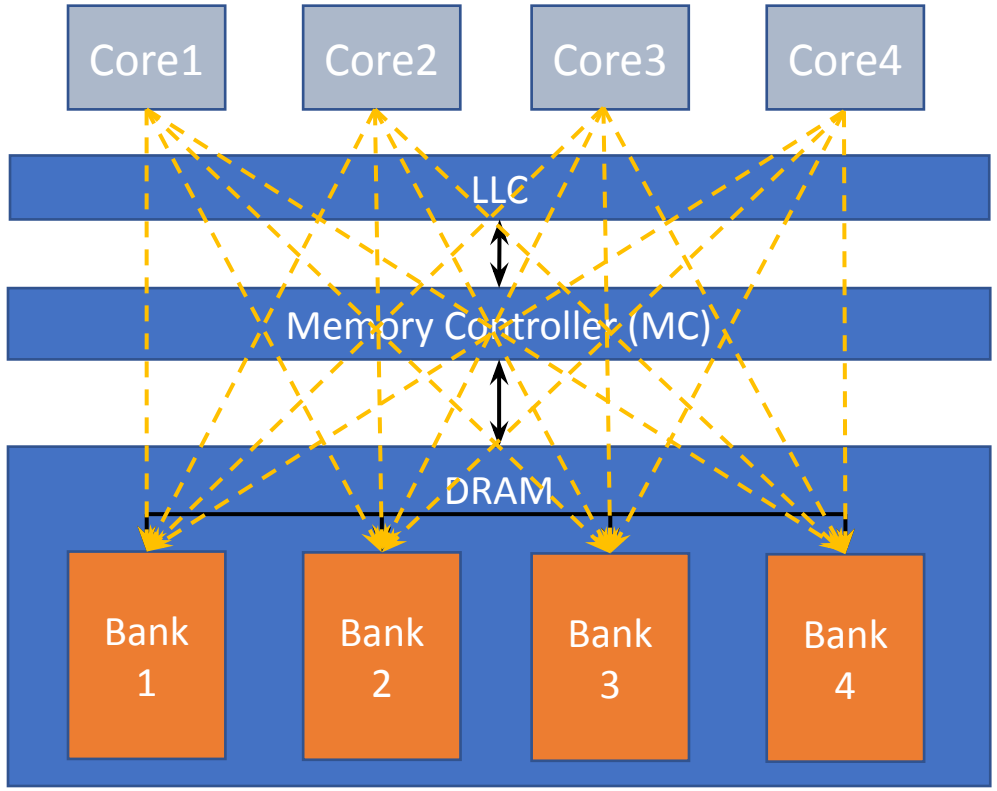


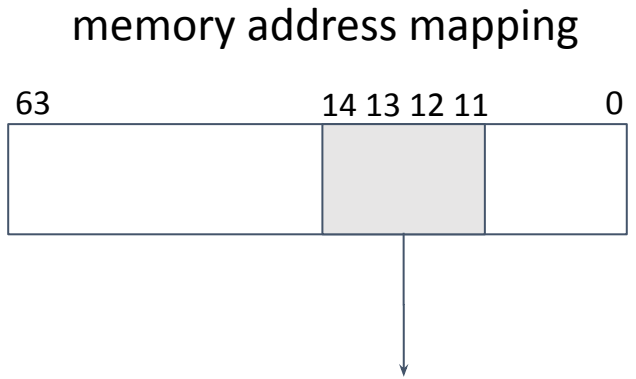




ARM Cortex A72/A57 LLC Bank Organization

# DRAM bank organization

- Multiple banks can be accessed simultaneously

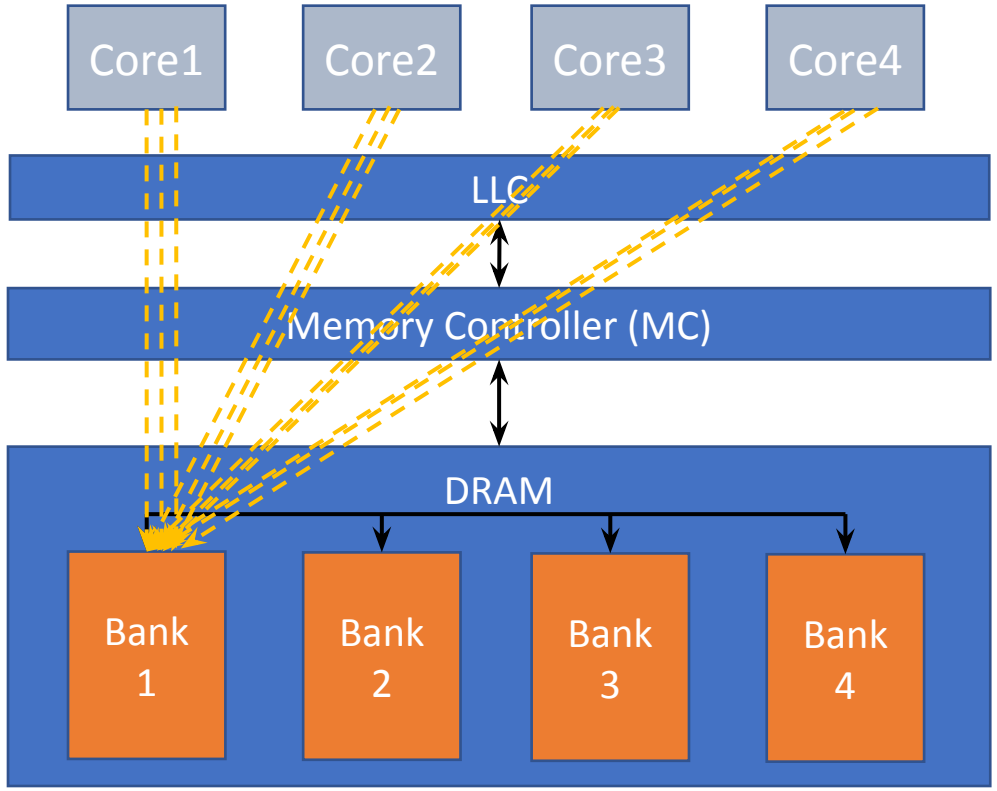



Raspberry Pi 4 DRAM bank mapping (**16 banks**)

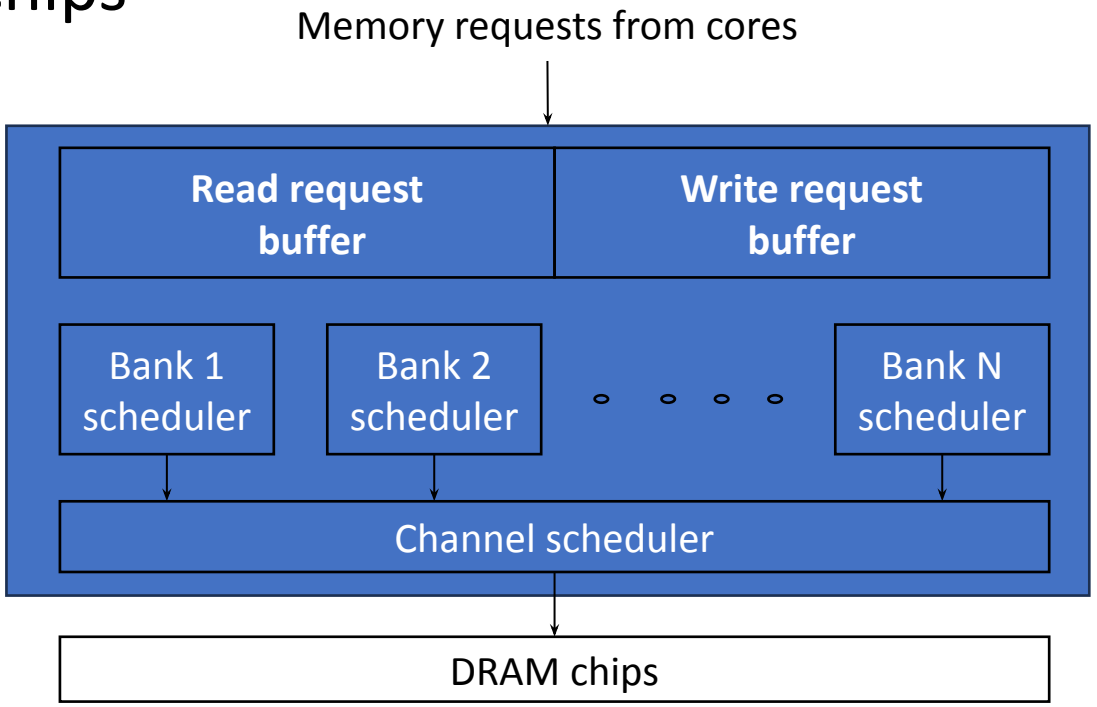
# Multi-bank cache/DRAM organizations

- Can be a problem when all try to access the same cache/dram bank



# Memory controller (MC)

- Schedule memory requests on DRAM chips
- Subject to DDR timing constraints
- Can re-order the requests to maximize memory throughput
- Often prioritize reads over writes unless too many writes are pending
- Scheduling algorithms can greatly impact worst-case timing

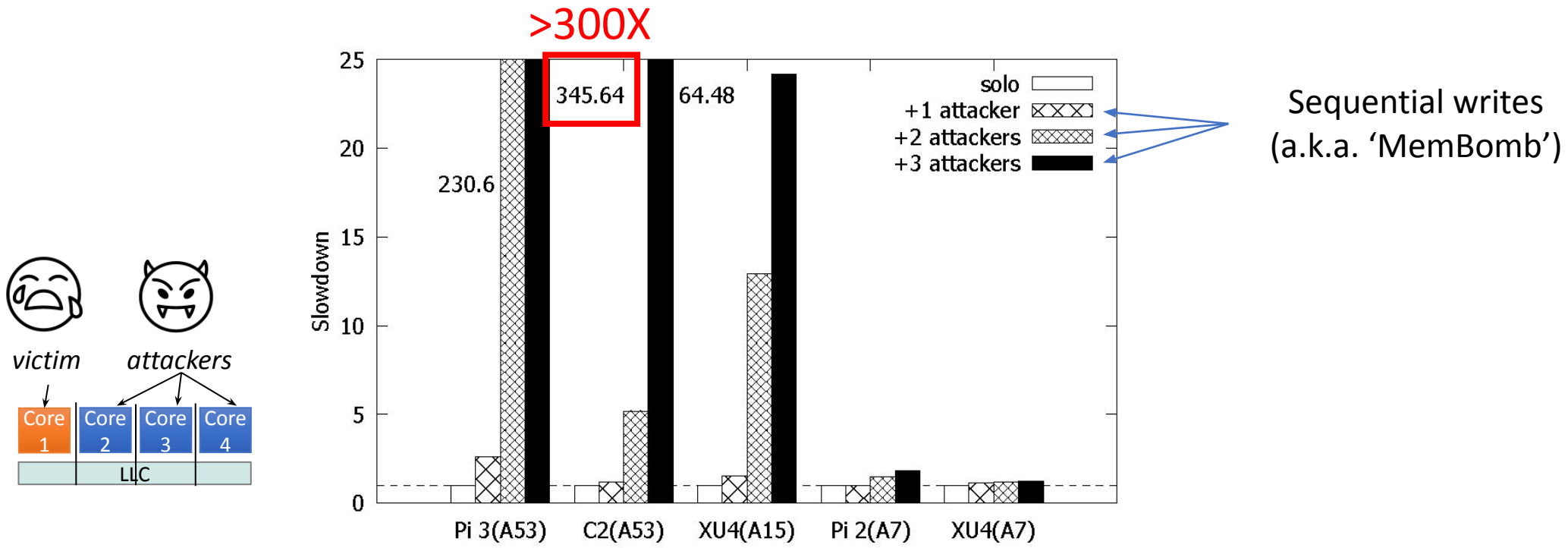



Heechul Yun, Rodolfo Pellizzoni, Prathap Kumar Valsan. “Parallelism-Aware Memory Interference Delay Analysis for COTS Multicore Systems.” In *ECRTS*, 2015

# Effective strategies to cause interference

- Try to exhaust various hardware queues/buffers
- Try to generate many requests targeting a single resource (bank)
- Writes often cause worse contention than reads

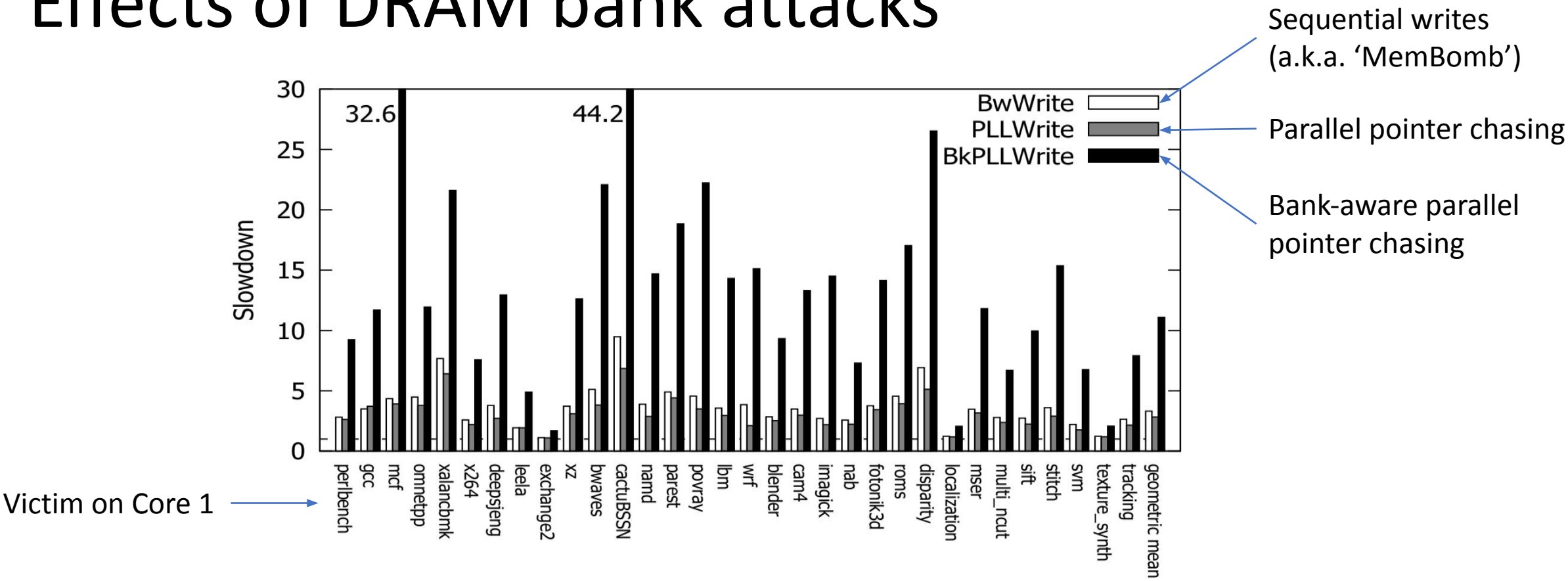
# Effects of cache internal buffer attacks




- Observed worst-case: >300X (times) slowdown on popular multicores
- Even when the cache is *partitioned* to protect the victim

M. G. Bechtel and H. Yun. "Denial-of-Service Attacks on Shared Cache in Multicore: Analysis and Prevention." In *IEEE RTAS*, 2019

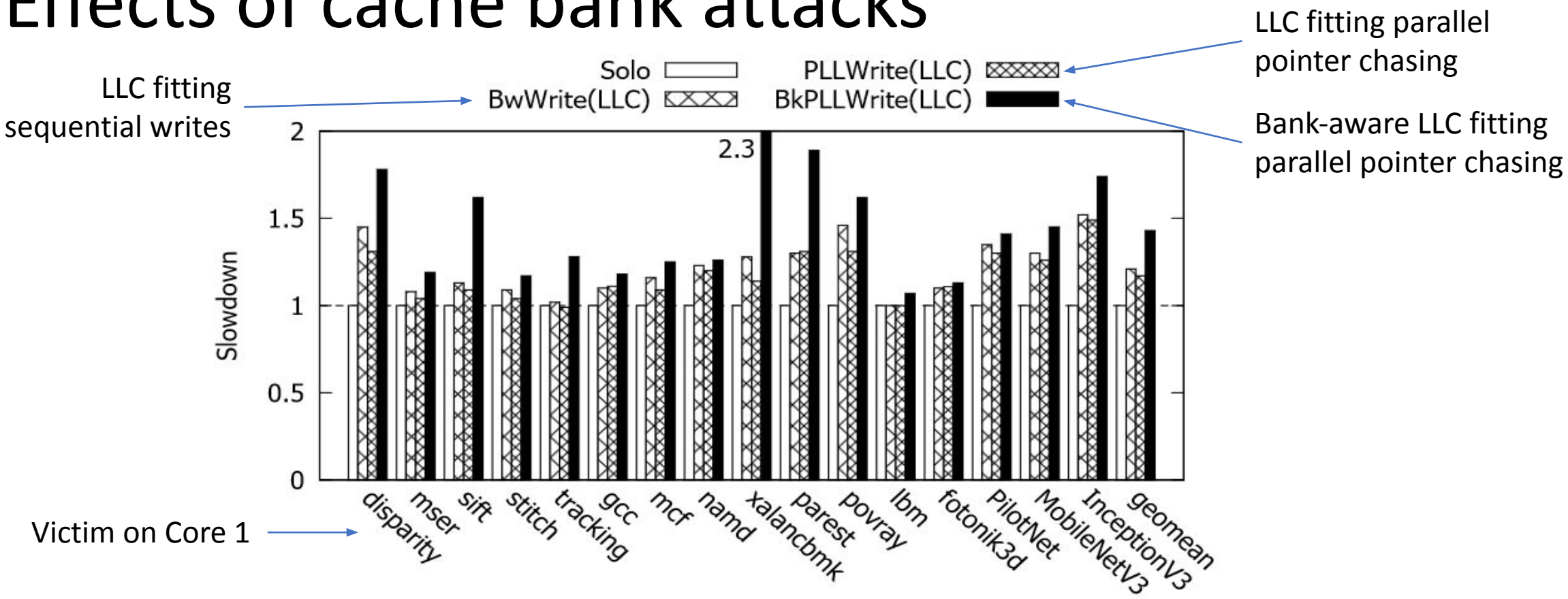
# Effects of DRAM bank attacks




- Targeting a single DRAM bank caused up to 44X slowdown in real apps
- LLC space partitioning was not effective

M. G. Bechtel and H. Yun. "Memory-Aware Denial-of-Service Attacks on Shared Cache in Multicore Real-Time Systems." In *IEEE Transactions on Computers*, 2021

# Effects of cache bank attacks




- Targeting a single cache bank caused up to 2.3X slowdown in real apps
- LLC space partitioning and DRAM bandwidth throttling were not effective

M. G. Bechtel and H. Yun. "Cache Bank-Aware Denial-of-Service Attacks on Multicore ARM Processors." In *IEEE RTAS,* 2023

# Takeaways

- Memory-level parallelism (MLP) is key to understand modern multicore processors (MCPs)
- High MLP designs at all levels of the memory hierarchy are essential for performance/throughput, but they also can be problematic **hardware interference channels** from a real-time perspective
- Contrary to popular beliefs, interconnects are usually not major interference channels in modern MCPs. Major ones are at the edges
- There are effective “attack” strategies to cause massive cross-core interference, which cannot be easily mitigated by existing software/hardware partitioning techniques

# References


- [C] Michael Garrett Bechtel and Heechul Yun. Cache Bank-Aware Denial-of-Service Attacks on Multicore ARM Processors. *IEEE Intl. Conference on Real-Time and Embedded Technology and Applications Symposium (RTAS)*, May 2023. [paper] [slides] [code]
- [J] Michael Garrett Bechtel and Heechul Yun. Memory-Aware Denial-of-Service Attacks on Shared Cache in Multicore Real-Time Systems. *IEEE Transactions on Computers*, 2021. [paper] [code]
- [C] Michael Garrett Bechtel and Heechul Yun. Denial-of-Service Attacks on Shared Cache in Multicore: Analysis and Prevention. *IEEE Intl. Conference on Real-Time and Embedded Technology and Applications Symposium (RTAS)*, April 2019. [paper] [arXiv] [slides] [code] [data] (**Outstanding Paper Award**)
- [C] Prathap Kumar Valsan, Heechul Yun, Farzad Farshchi. Taming Non-blocking Caches to Improve Isolation in Multicore Real-Time Systems. *IEEE Intl. Conference on Real-Time and Embedded Technology and Applications Symposium (RTAS)*, IEEE, 2016. [paper] [slides] [code] (**Best Paper Award**)
- [C] Heechul Yun, Rodolfo Pellizzoni, Prathap Kumar Valsan. Parallelism-Aware Memory Interference Delay Analysis for COTS Multicore Systems. *Euromicro Conference on Real-Time Systems (ECRTS)*, 2015. [paper] [slides]