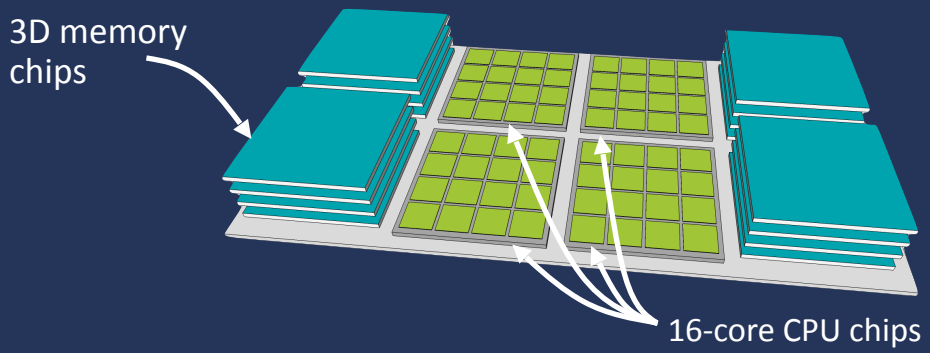
UNIVERSITY OF TORONTO | Engineering

Architecting Chiplet-Based Systems



Natalie Enright Jerger
Percy Edward Hart Professor of Electrical and Computer Engineering
enright@ece.utoronto.ca
www.eecg.toronto.edu/~enright

Large *Cost-Effective* SoCs through Disintegration

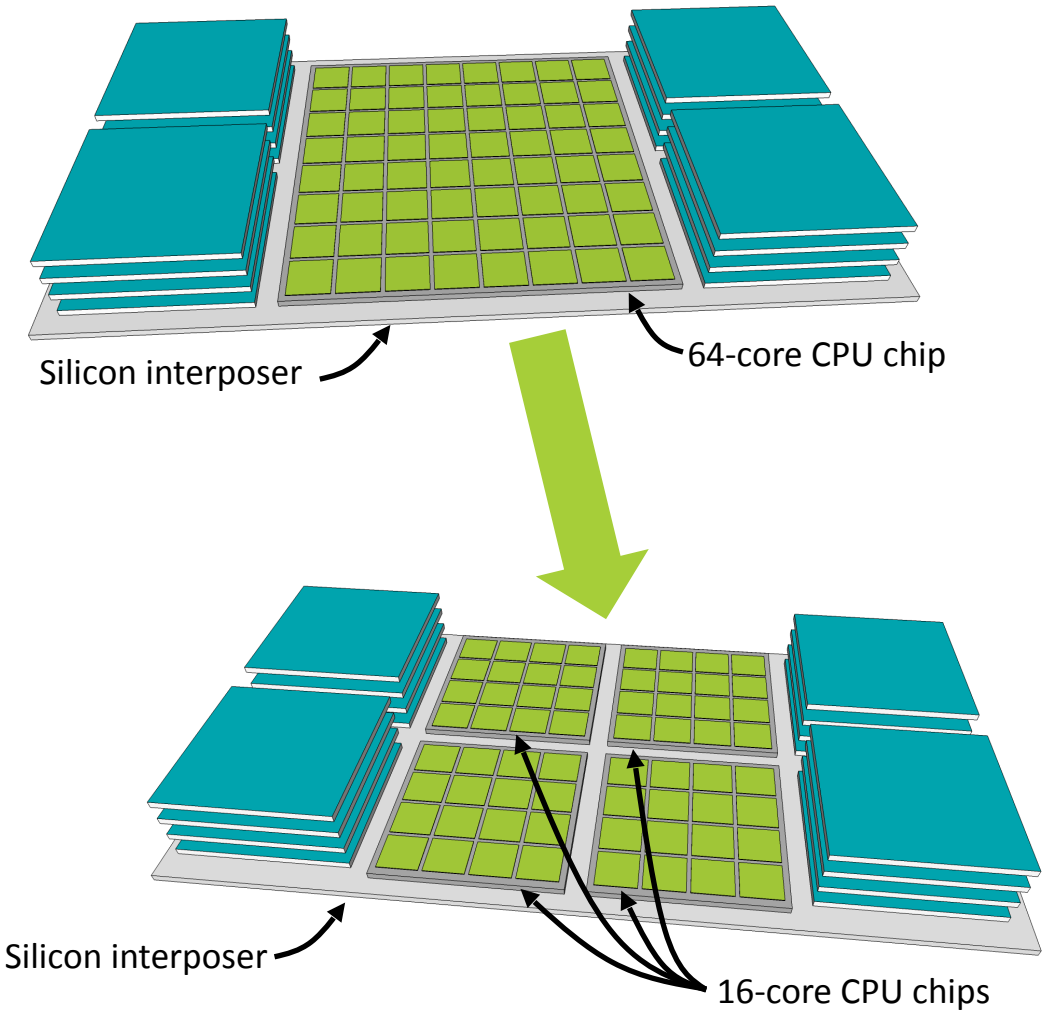
Why disintegrate?

Want more functionality, but…

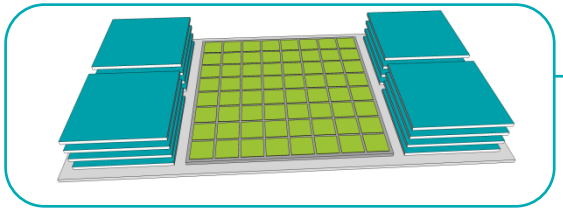
Big chips are expensive

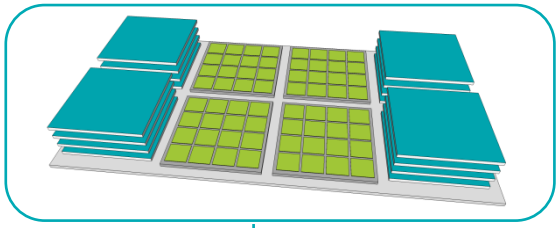
Break (disintegrate) into several smaller pieces

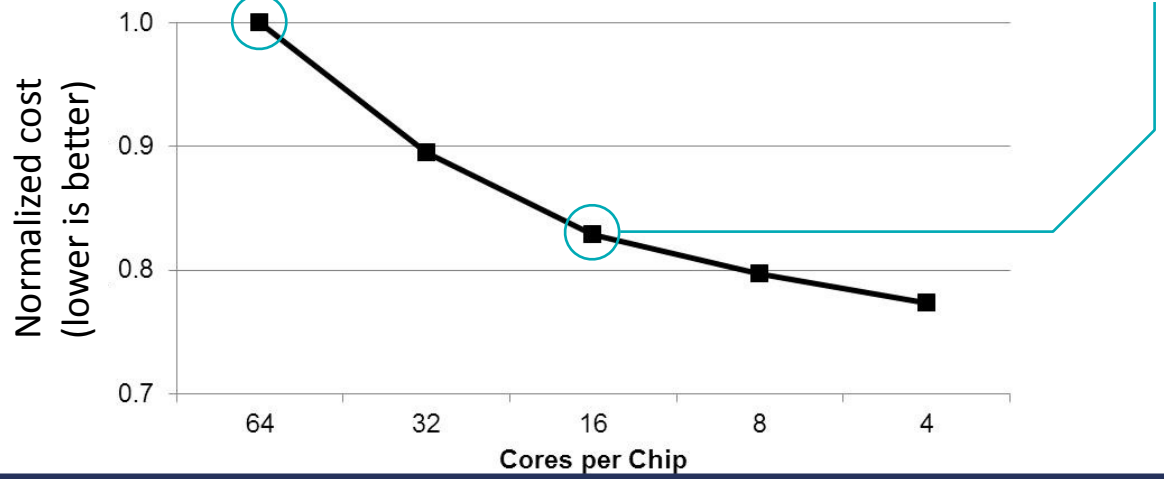
Cheaper to manufacture



Disintegration → Cheaper SoCs

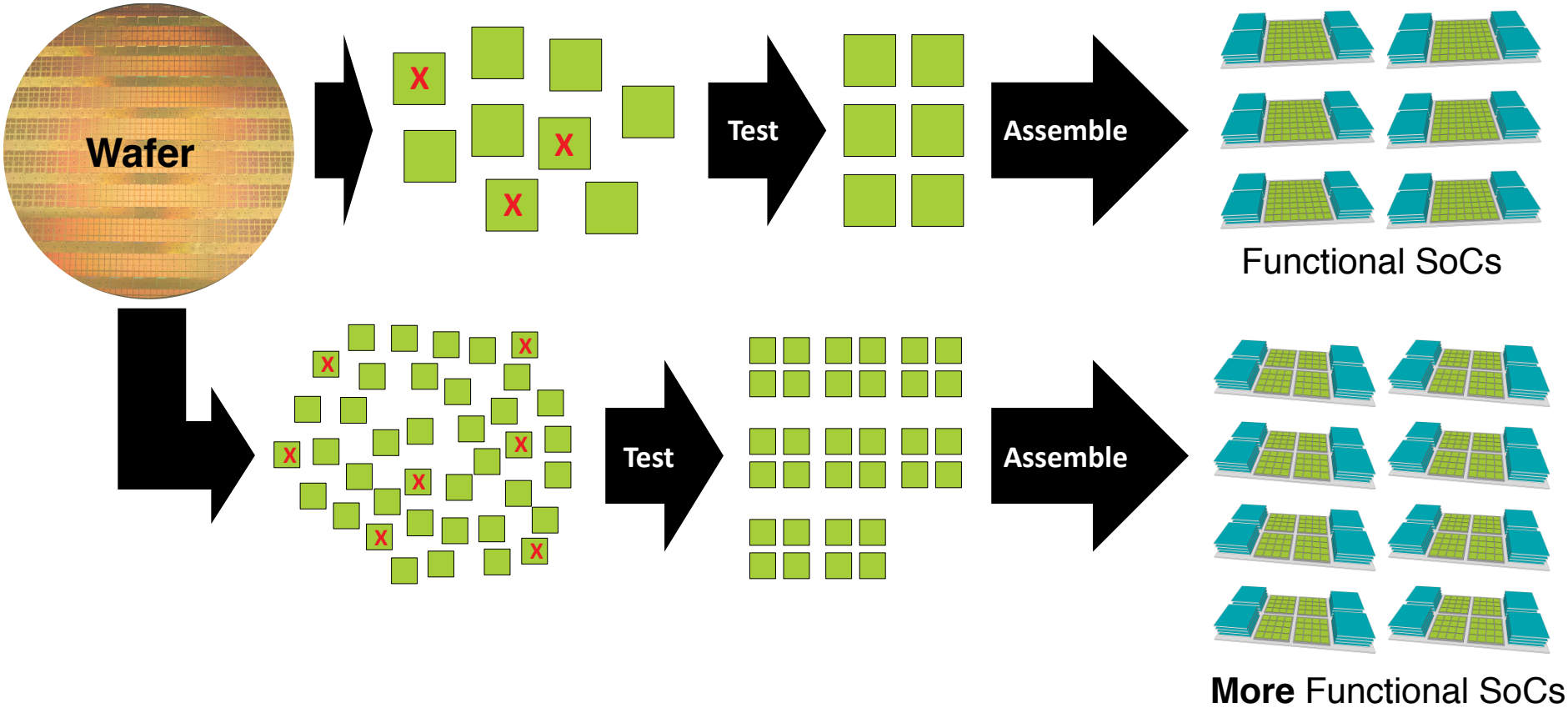






Disintegrated SoCs have potential for reducing costs of large chips while maintaining functionality

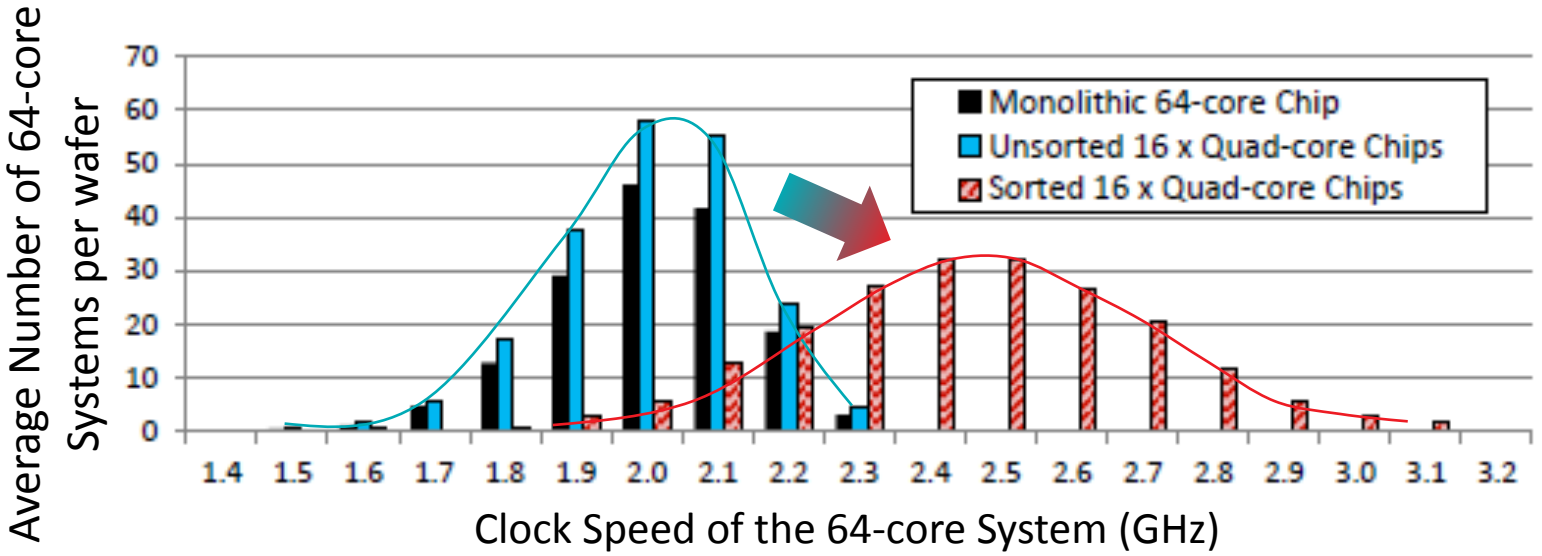
Cost Argument: High-Level Idea



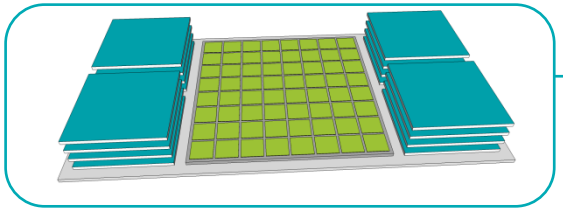
Cheaper Chips + Larger Profit Margins

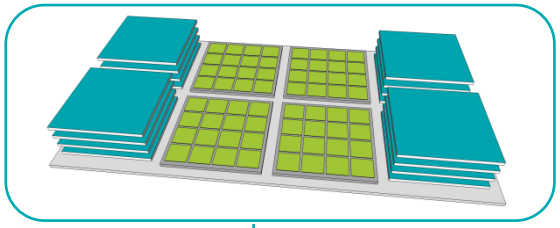
Sort chips before assembly to improve speed binning

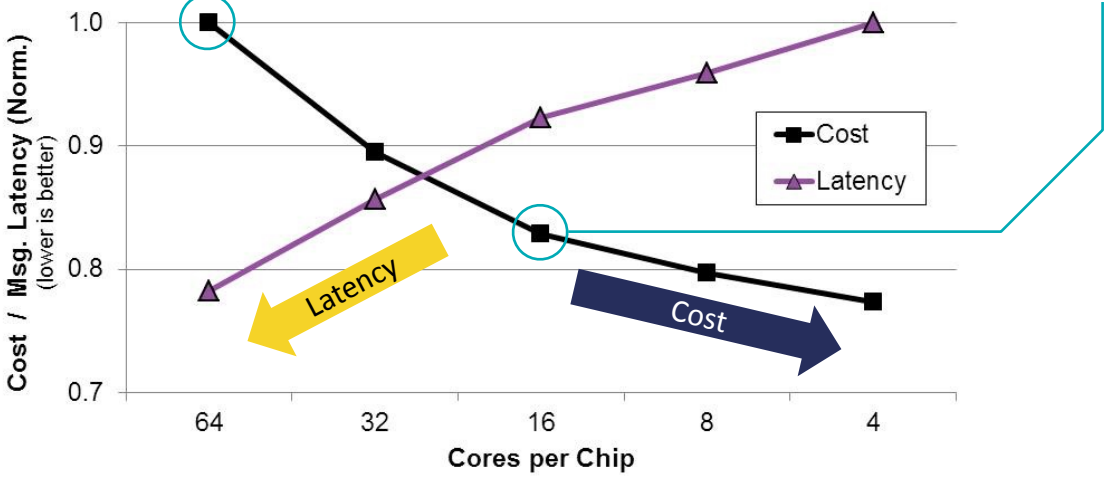
Within die variations hurt performance of large monolithic chips



Fragmented Architecture

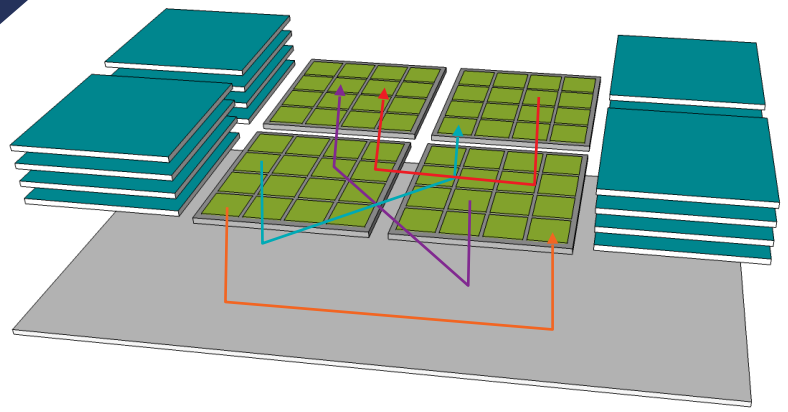






Disintegrated SoCs have potential for reducing costs of large chips

But performance degrades with disintegration granularity



How to integrate chiplets?

What are we looking for when reintegrating?

Enable small/simple chiplets

High bandwidth/low latency connections between chiplets

Ease of manufacturing

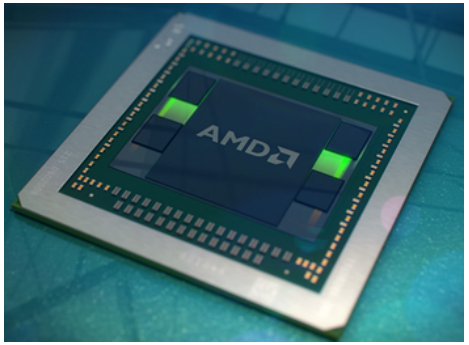
How to integrate?

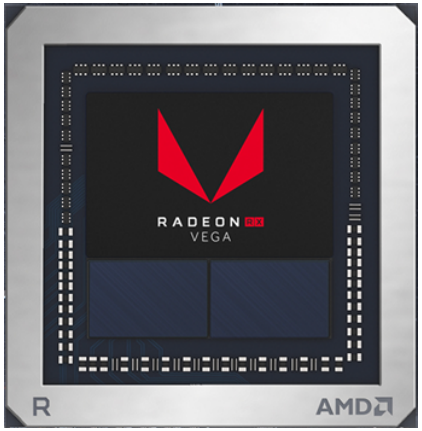
Multi-chip modules (MCM)

Embedded Multi-Chip Interconnect Bridge (EMIB)

Silicon Interposer (2.5D)

- 😁 Technology maturation (high volume passive interposer production — 3 years)





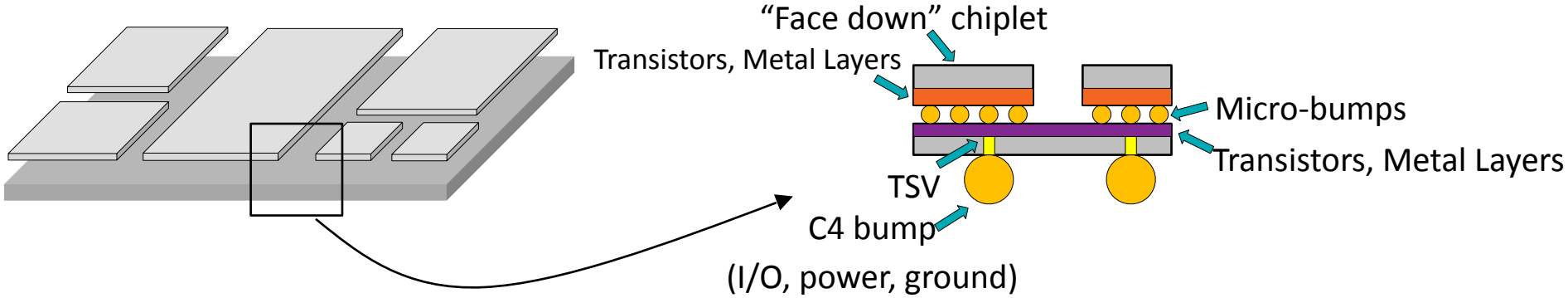
How to integrate?

Multi-chip modules (MCM)

Embedded Multi-Chip Interconnect Bridge (EMIB)

Active Silicon Interposer (2.5D)

- Simple, small chiplets
- Move SoC functionality into interposer
- Implement in older technology node



Interposers: An enabling integration technology

Facilitates modular SoC design

But what about…

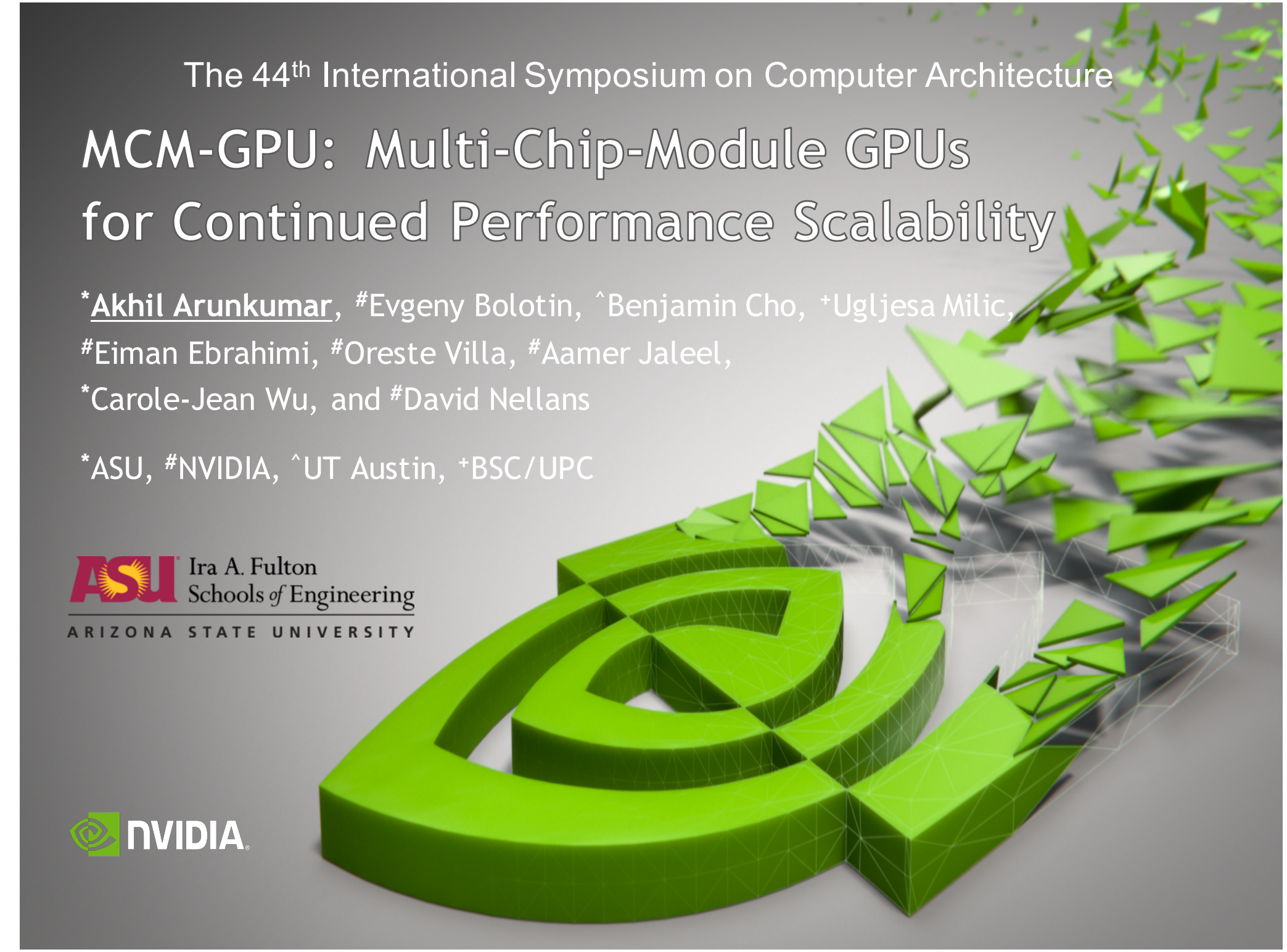
- Cost — Aren't interposers expensive?
- Communication — How do we reintegrate?
- How to maximize modularity while reintegrating?

The 44th International Symposium on Computer Architecture

MCM-GPU: Multi-Chip-Module GPUs for Continued Performance Scalability

*Akhil Arunkumar, #Evgeny Bolotin, ^Benjamin Cho, +Ugljesa Milic,
#Eiman Ebrahimi, #Oreste Villa, #Aamer Jaleel,
*Carole-Jean Wu, and #David Nellans

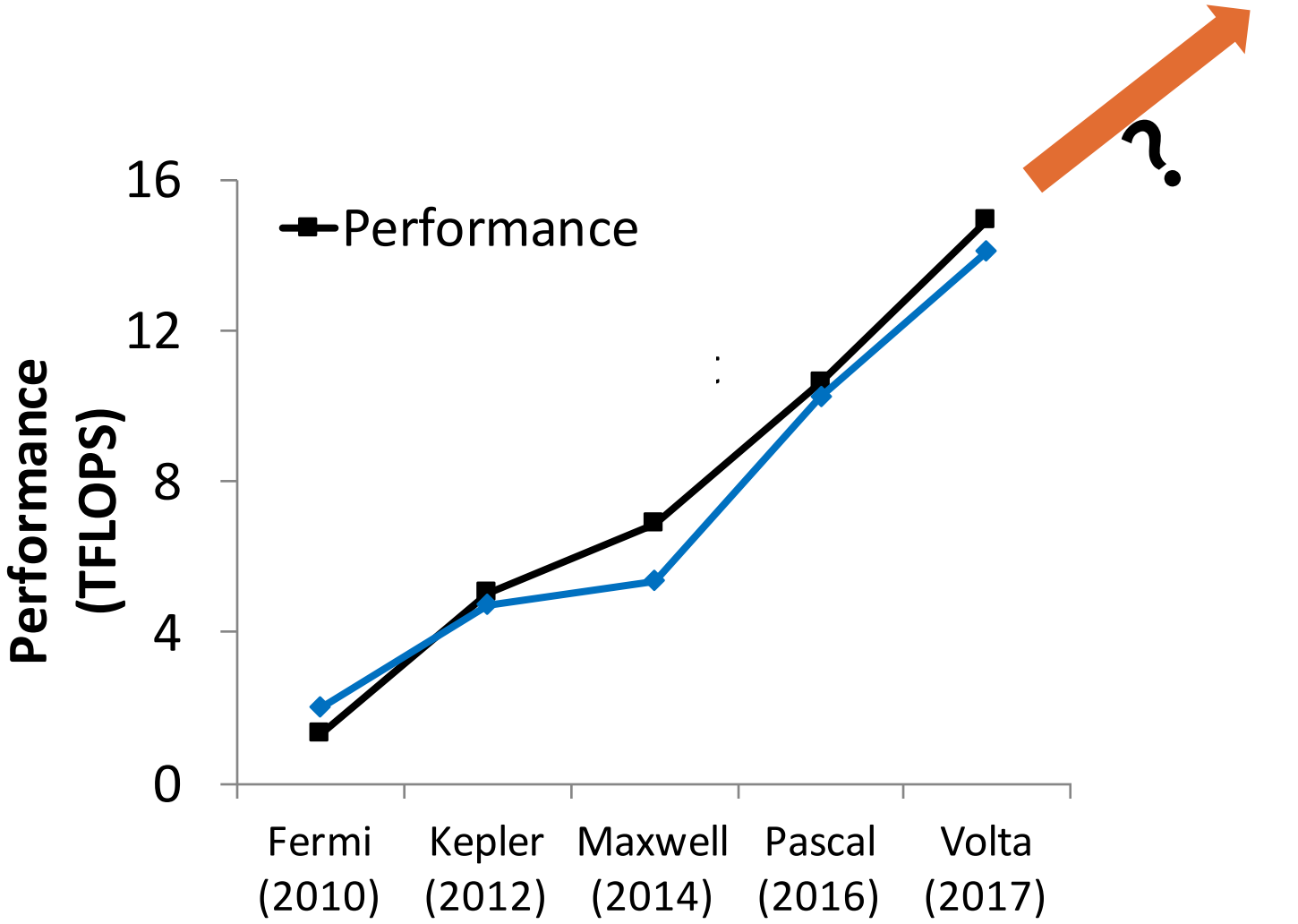
*ASU, #NVIDIA, ^UT Austin, +BSC/UPC



Outline

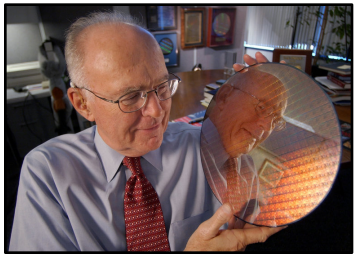
- The Problem: Performance Scalability of Future GPUs
- Solution: MCM-GPU Multi-Chip-Module GPU
  - Basic MCM-GPU Architecture
  - Optimized MCM-GPU Architecture
- Evaluation Results

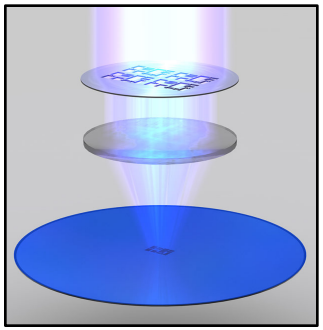
GPU Performance Scaling Trend



The Problem: Performance Scalability of Future GPUs

- End of Moore's law
  - Transistor scaling has slowed down
  - Expected to come to a halt at 7nm
- Photolithography limitations
  - GPU dies are reticle limited
  - Maximum die size is limited by optics (~800 mm^2)

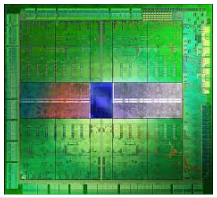




How do we continue to scale GPU performance?

Performance Scaling via Integration

- We scale via hierarchy of integration tiers:

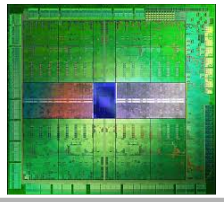
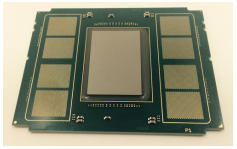


	Chip	Board	System
Interconnect BW	10 TB/s	160 GB/s	12.5 GB/s
Interconnect energy	80 fJ/bit	10 pJ/bit	250 pJ/bit
Cost	Low	High	Very High

- Progressively less bandwidth at higher cost

Performance Scaling via Integration

- We scale via hierarchy of integration tiers:

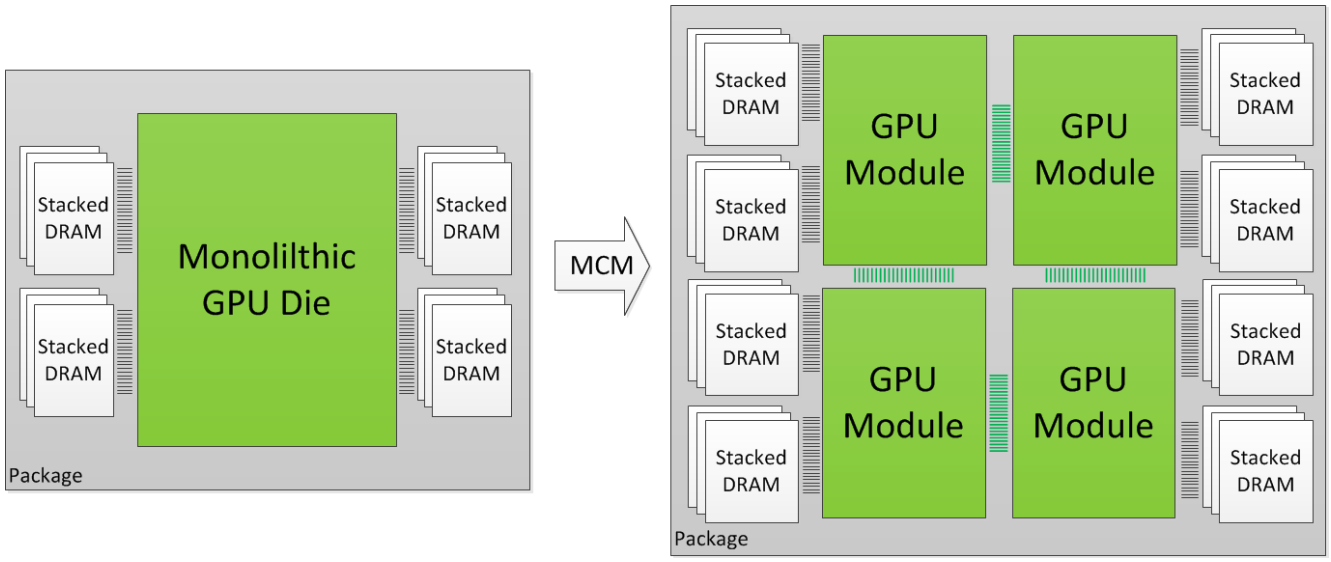
	Chip	Package	Board	System
Interconnect BW	10 TB/s	3 TB/s	160 GB/s	12.5 GB/s
Interconnect energy	80 fJ/bit	1 pJ/bit	10 pJ/bit	250 pJ/bit
Cost	Low	Medium	High	Very High

- Progressively less bandwidth at higher cost

Efficient package-level integration is possible

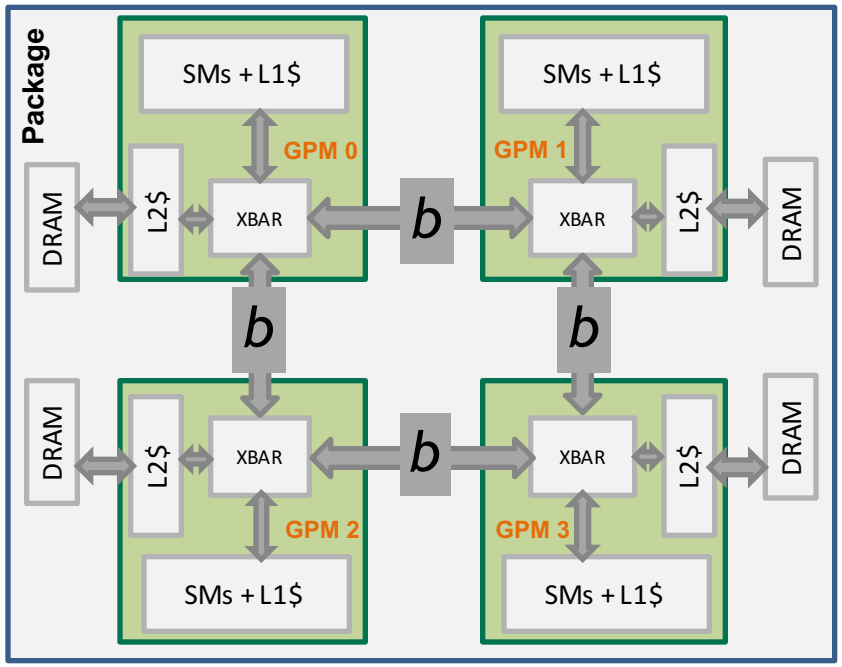
MCM-GPU: Multi-Chip-Module GPU

- Key Idea: Integrate multiple GPU Modules (GPMs) on a package
  - Use on-package signaling for inter-GPM interconnect
  - Programmer-transparent single large logical GPU

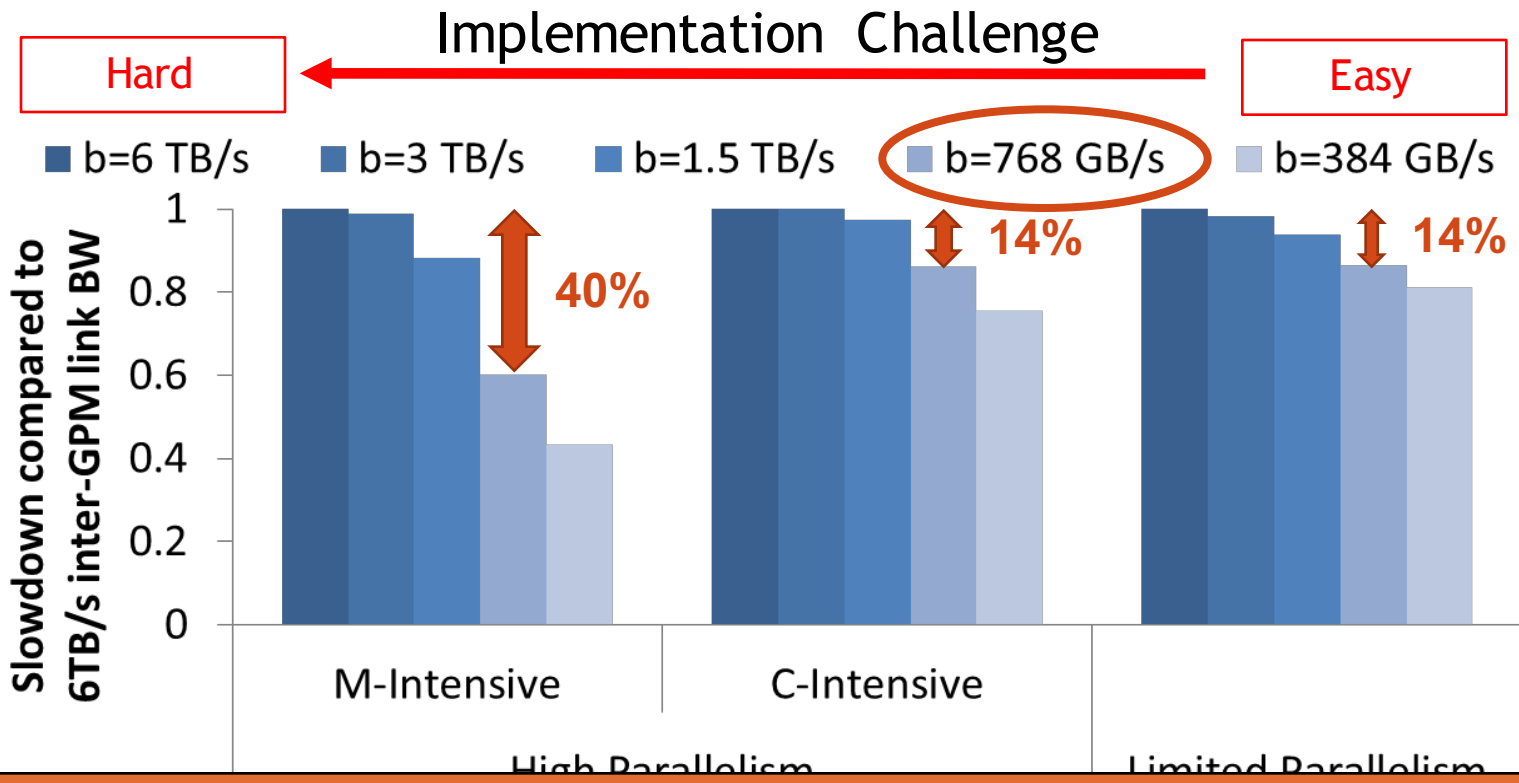


Basic MCM-GPU Architecture

- Modular – 4 GPMs on package example
  - 256 SMs total, 3TB/s DRAM BW
- On-package inter-GPM interconnect
- Other aspects unchanged
  - Cache hierarchy, thread-block (CTA) scheduling, memory interleaving
- Challenges:
  - Inter-GPM bandwidth is performance-critical
  - In-package NUMA design



Inter-GPM Bandwidth Sensitivity



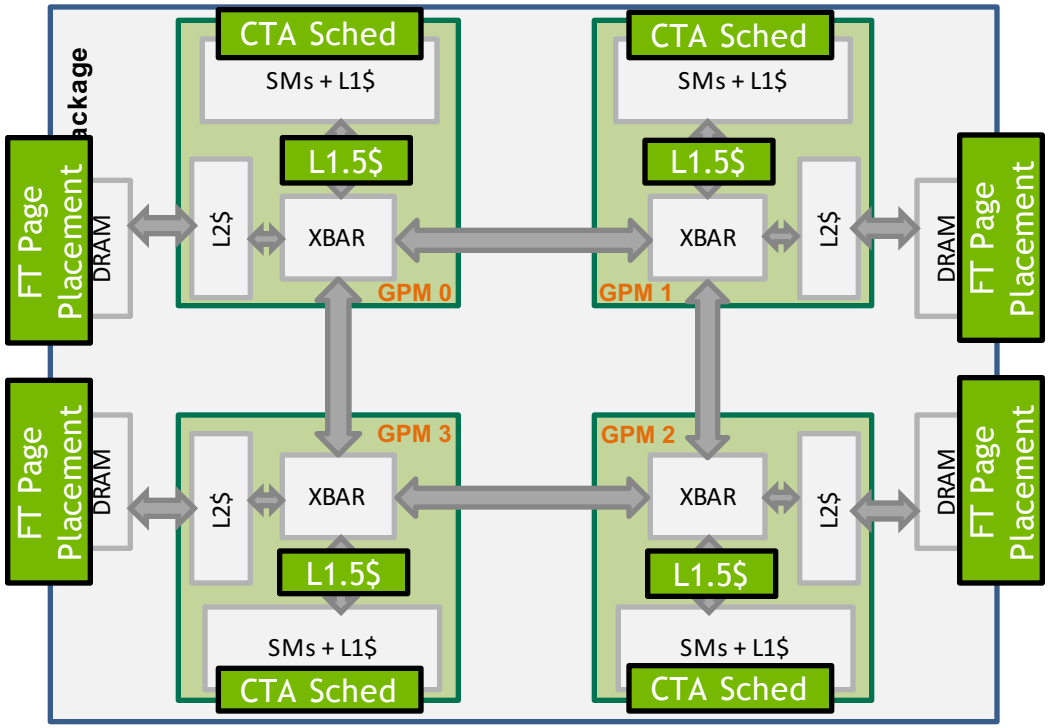
Bridge performance gap with architectural optimizations

Optimized MCM-GPU Architecture

Goal: Reduce inter-GPM traffic

- Exploit ***locality*** within GPMs.

1. New L1.5 cache
2. Distributed and batched CTA scheduling
3. First Touch Page Placement



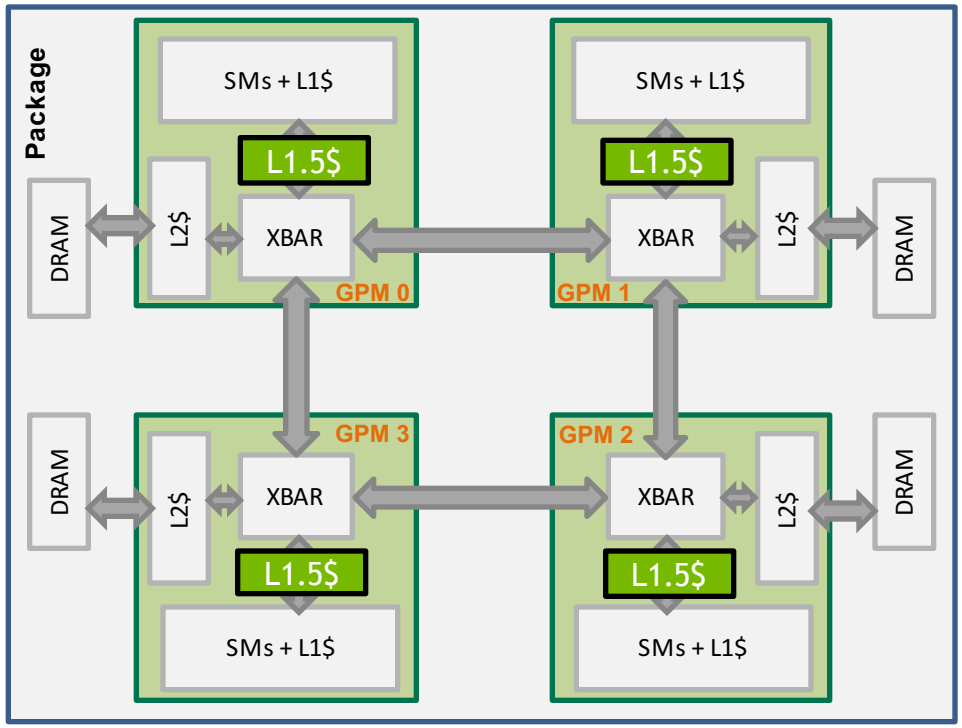
1. GPM-Side L1.5 Cache

Remote traffic shows locality

- Not captured in SM's L1 caches

Proposal:

- Add L1.5 cache
  - Reduce L2 cache
- Remote-only cache allocation
- SW based coherence



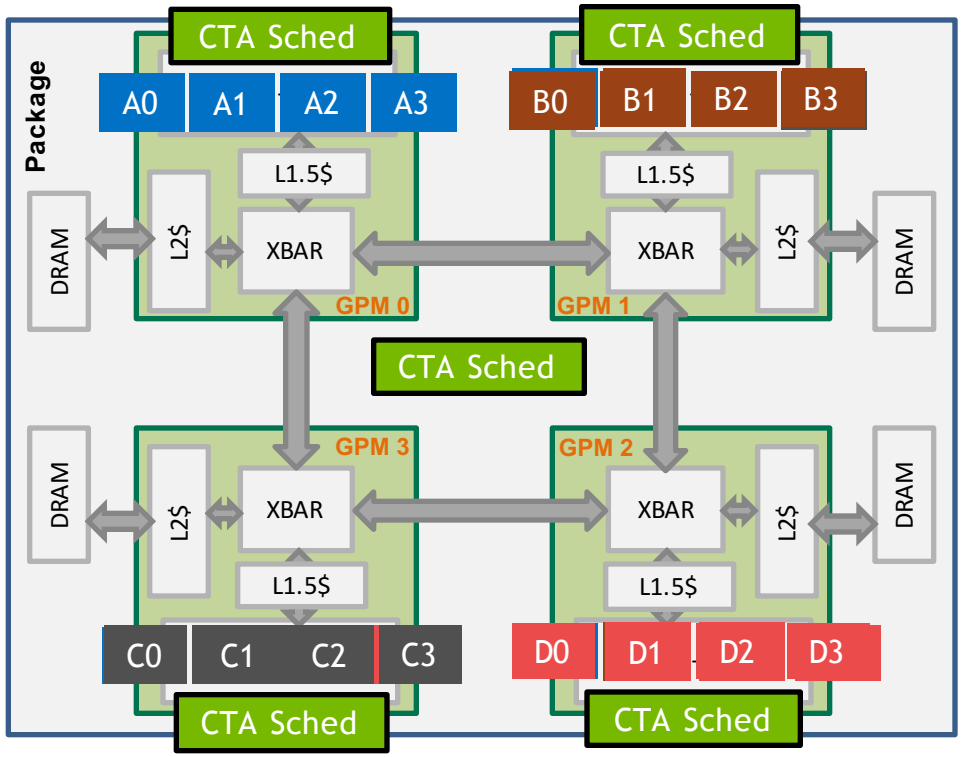
2. Distributed CTA Scheduling

Contiguous CTAs possess locality

- Lost in basic MCM-GPU

Proposal:

- Distributed & Batched CTA scheduling
- Contiguous CTAs → same GPM



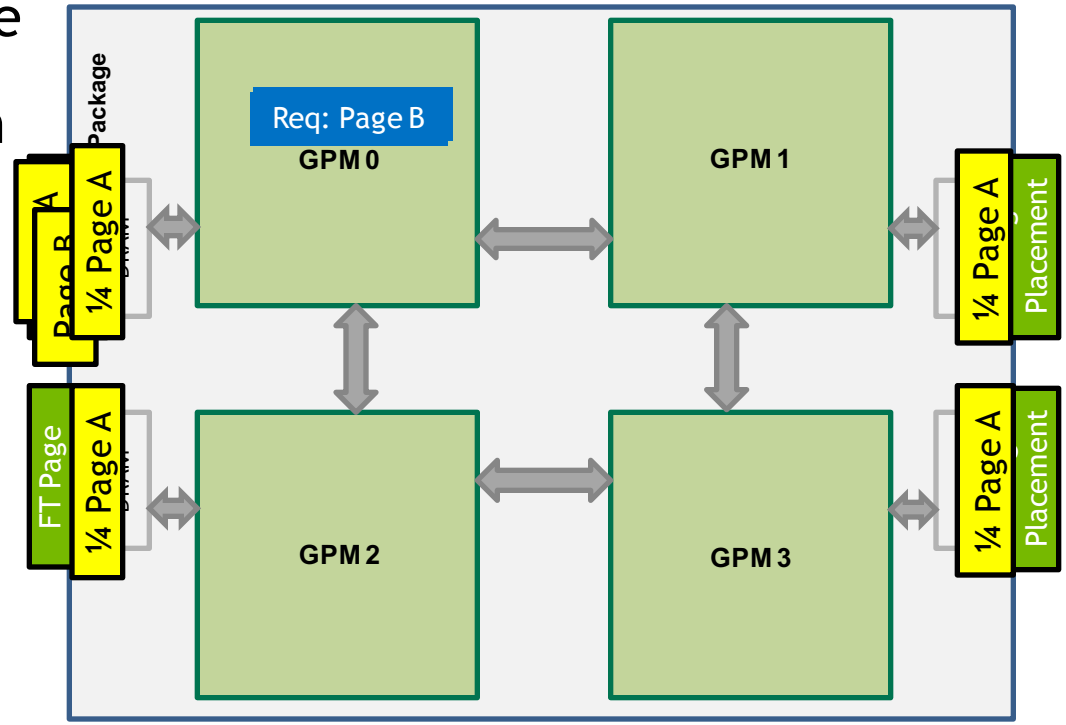
3. First Touch Page Placement

Contiguous CTAs access same page

- Locality lost due to fine-grain memory interleaving

Proposal:

- First touch page placement across memory partitions



Experimental Methodology

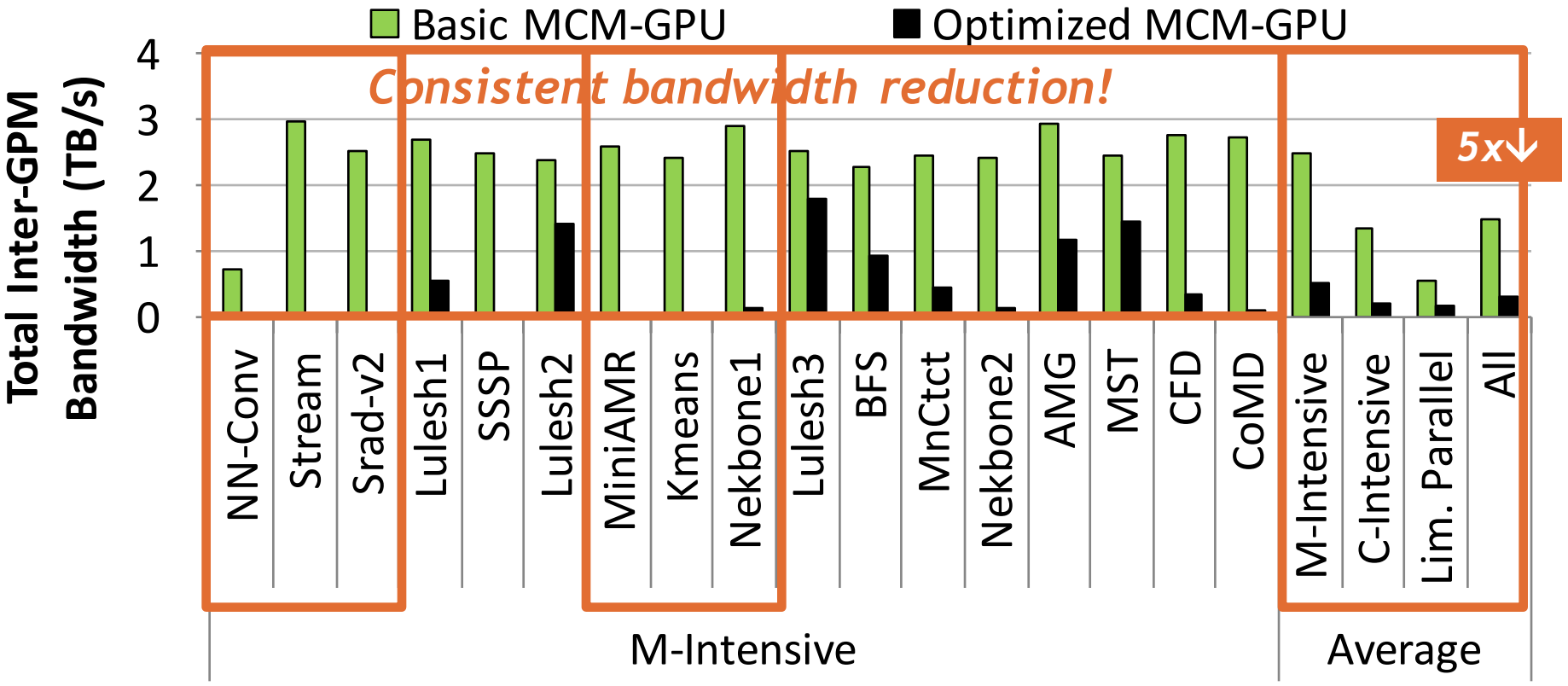
- Simulator: NVIDIA In-house GPU Simulator
- 256 SM GPU w/ 4 GPMs
  - 3 TB/s Memory BW
  - 16 MB (L1.5 + L2) Cache
  - 768 GB/s Inter-GPM Link BW
- Benchmarks:
  - CORAL [1], Rodinia [2], Lonestar [3], and NVIDIA In-house Benchmarks
  - 48 Benchmarks:
    - 33 High Parallelism
      - 17 Memory Intensive
      - 16 Compute Intensive
    - 15 Limited Parallelism

[1] "CORAL Benchmarks" - https://asc.llnl.gov/CORAL-benchmarks/

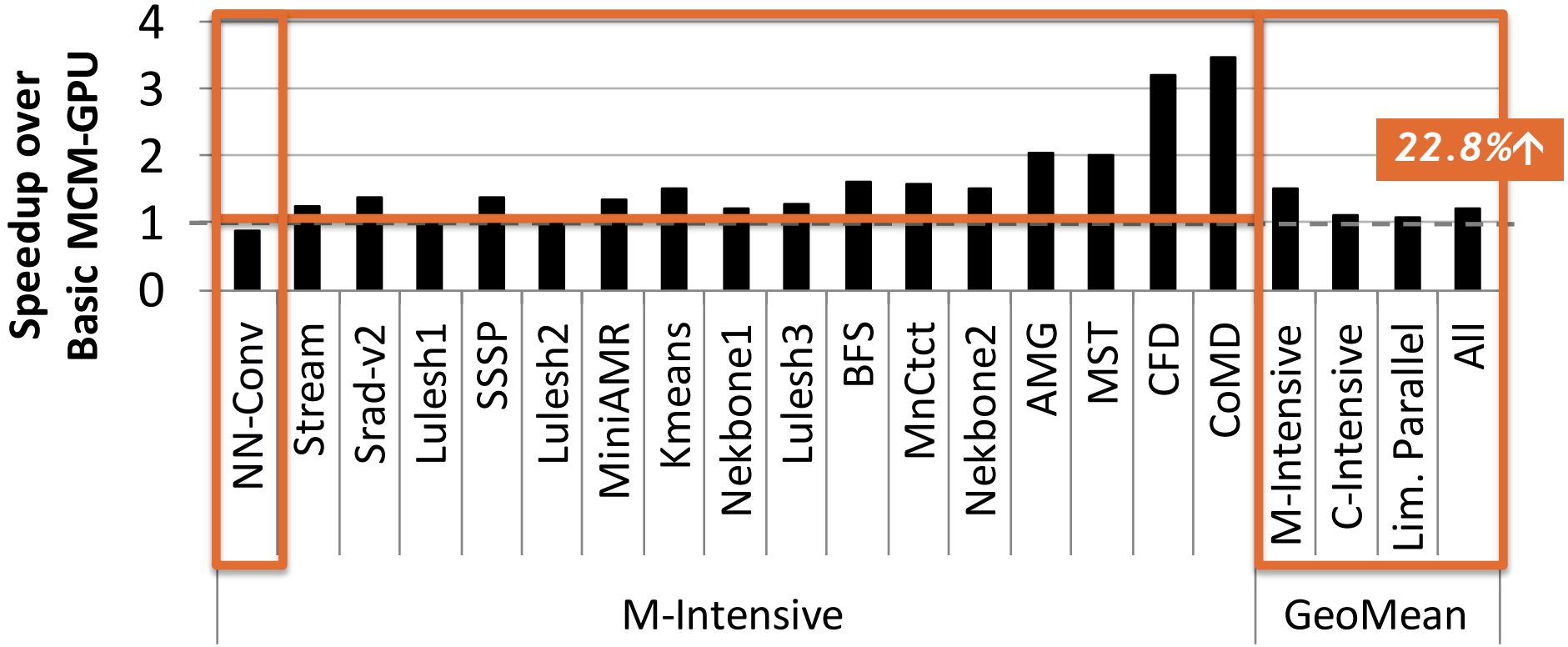
[2] "Rodinia: A Benchmark Suite for Heterogeneous Computing" S. Che et al., IISWC 2009

[3] "A Quantitative Study of Irregular Programs on GPUs" M. Burtscher et al., IISWC 2012

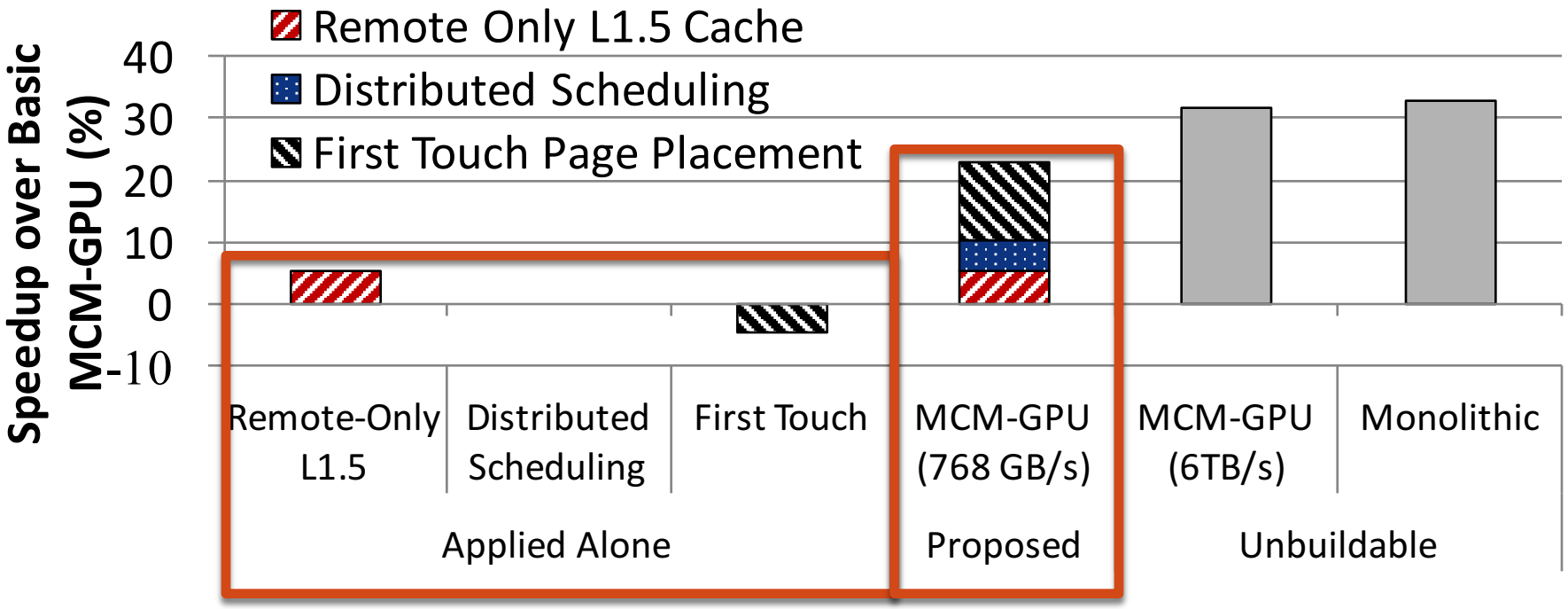
Results: Inter-GPM Bandwidth



MCM-GPU Performance Improvement



Anatomy of Performance Improvement



- The proposed optimizations work well in combination
- MCM-GPU is within 8% of ideal monolithic GPU

Summary

- Problem: How do we continue scaling single GPU performance?
- Proposed solution: Multi-Chip-Module GPU architecture
  - Basic and optimized MCM-GPU architectures
- Optimized MCM-GPU:
  - 5x Inter-GPM bandwidth reduction
  - 22.8% performance improvement
  - Within 8% of performance of ideal monolithic GPU

Thank You