Berkeley Architecture Research

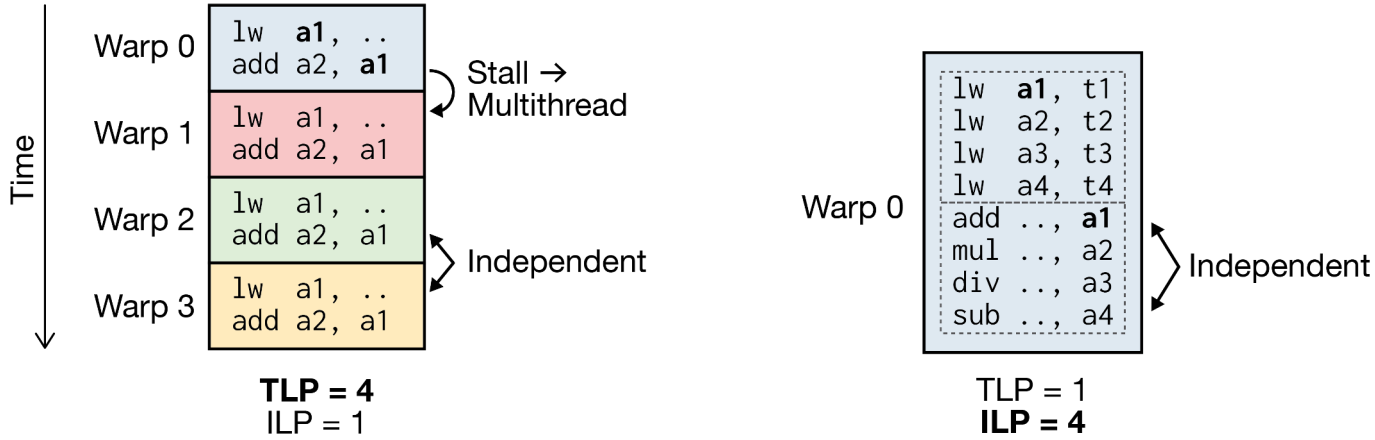
Muon: An ASIC-optimized SIMT Core Designed for Register-heavy GPU Kernels

Hansung Kim*, Richard Yan*, Shashank Anand*,
Joshua You, Amanda Shi, Christopher Fletcher, Yakun Sophia Shao

OSCAR Workshop | Raleigh, NC | June 28, 2026

Latency Hiding in GPUs

GPUs hide latencies in two ways: **TLP (Occupancy)** and **ILP**



❶ Modern GPU kernels increasingly rely on ILP, not just TLP
❷ Existing SIMT cores lack support for ILP-based latency hiding
❸ **Muon** is a new open-source SIMT core designed for ILP, targeting ASIC

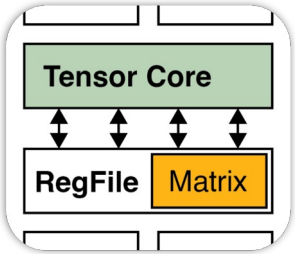
GPU Kernels are Occupancy-limited

Modern AI GPU kernels have high register pressure → **low occupancy / TLP**

GEMM-heavy

Tensor Cores compute and store tensors in registers

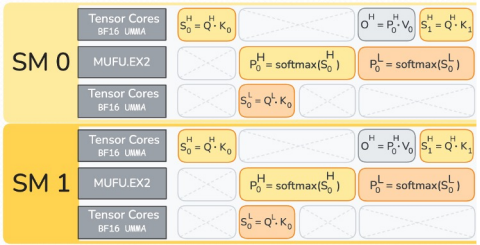
Ex) CUTLASS: Occ ≤ 15%[1]



Kernel Fusion

Stages intermediate tensors in registers for pipelining

Ex) FA-3: Occ ≤ 20%[1,2]



FlashAttention-4 SW Pipelining[2]

Warp Specialization

Warp slots designated to control MMA / TMA

Warp Role	Warp
MMA	0
Scheduler	1
Mainloop Load	2
Epilogue Load	3
Epilogue	4, 5, 6, 7

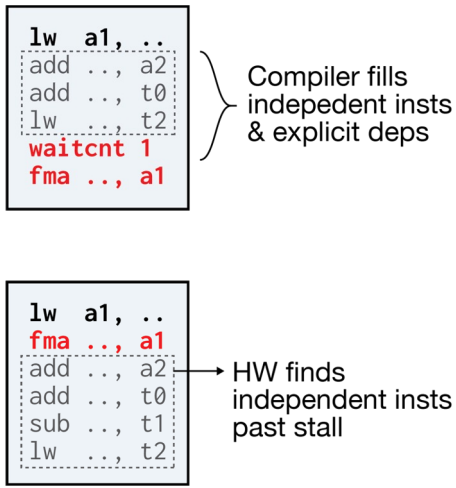
Warp Specialization in Blackwell CUTLASS[3]

[1] H. Kim et al., “Virgo: Cluster-level Matrix Unit Integration in GPUs for Scalability and Energy Efficiency,” ASPLOS 2025.
[2] T. Zadouri et al., “FlashAttention-4: Algorithm and Kernel Pipelining Co-Design for Asymmetric Hardware Scaling,” arXiv 2026.
[3] NVIDIA CUTLASS Documentation, https://docs.nvidia.com/cutlass/latest/media/docs/cpp/blackwell_cluster_launch_control.html#blackwell-warp-specialized-persistent-kernel

ILP is Crucial for Modern GPUs

Because occupancy is scarce, kernels increasingly rely on ILP[1]

However, SIMT cores are usually designed in-order



Compiler-guided static scheduling

Pros: Simple, cheap hardware
Cons: Complex compiler; SW stack challenges

HW-assisted dynamic scheduling ← **Muon**

Pros: Performance latitude, Easy adoption to existing stack
Cons: Complex HW — *By how much?*

[1] V. Volkov, “Better Performance at Lower Occupancy,” https://www.nvidia.com/content/gtc-2010/pdfs/2238_gtc2010.pdf

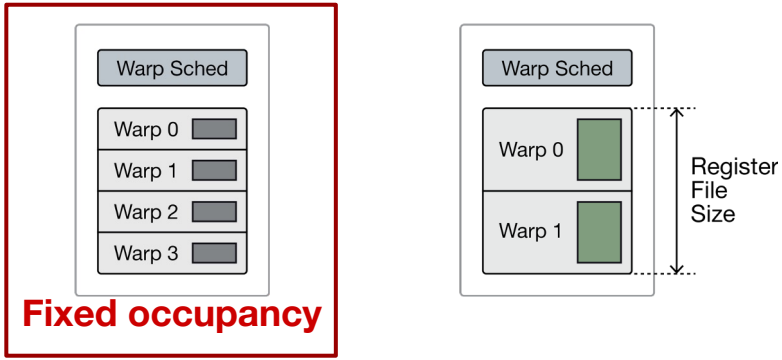
Limitations with Open SIMT Cores

Existing SIMT cores do not support **variable occupancy** + **ILP** latency-hiding

Cf) Vortex, Ventus, SIMTight, Nyuzi[1-4]

Except *Ventus*, occupancy is fixed at design-time; Per-warp register space fixed small

All in-order; lack of static scheduling to leverage ILP at compile-time



→ Fails to model **ILP** ↔ **TLP tradeoff** in modern kernels

[1] B. Tine et al., “Vortex: Extending the risc-v isa for gpgpu and 3d-graphics,” MICRO 2021.
[2] J. Li et al., “Ventus: A High-performance Open-source GPGPU Based on RISC-V and Its Vector Extension,” ICCD 2024.
[3] M. Naylor et al., “Advanced Dynamic Scalarisation for RISC-V GPGPUs,” ICCD 2024.
[4] J. Bush et al., “Nyuzi: An Open Source GPGPU for Graphics, Enhanced with OpenCL Compiler for Calculations,” DATE 2021.

Limitations with Open SIMT Cores

Existing cores are optimized for **FPGA**, with vendor-specific IP and primitives

- Memories realized as BRAM/LUT; No SRAM macro or PDK integration
  — Vortex the only exception

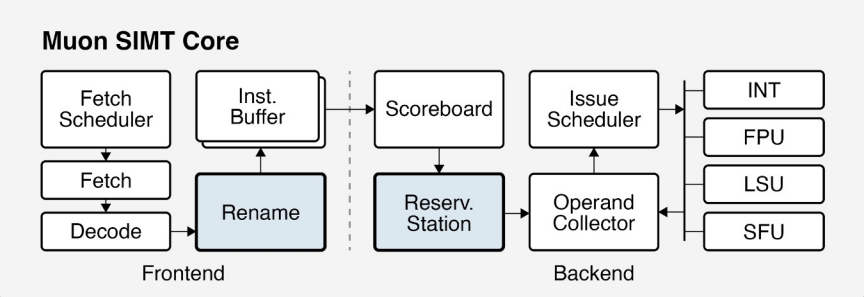
No reported silicon **tape-outs**; Limited silicon-measured PPA study

	Vortex[1]	Ventus[2]	SIMTight[3]	**Muon**
Microarchitecture	❌ Fixed occupancy; no ILP focus	⚠️ Variable occupancy; no ILP focus	❌ Fixed occupancy, no ILP focus (focus on scalarization)	✅ Variable occupancy + ILP w/ OoO issue
ASIC optimization	✅ ASIC PDK / SRAM; No reported tapeout	❌ Xilinx FPGA; No ASIC flow	❌ Intel FPGA; No ASIC flow	✅ ASIC PDK / SRAM; Taped out to **TSMC 16nm** (April ‘26)

[1] B. Tine et al., “Vortex: Extending the risc-v isa for gpgpu and 3d-graphics,” MICRO 2021.
[2] J. Li et al., “Ventus: A High-performance Open-source GPGPU Based on RISC-V and Its Vector Extension,” ICCD 2024.
[3] M. Naylor et al., “Advanced Dynamic Scalarisation for RISC-V GPGPUs,” ICCD 2024.

Muon SIMT Core

Muon is an ASIC-optimized SIMT core that enables **ILP-TLP tradeoff** in GPU kernels



Dynamic occupancy support via register renaming

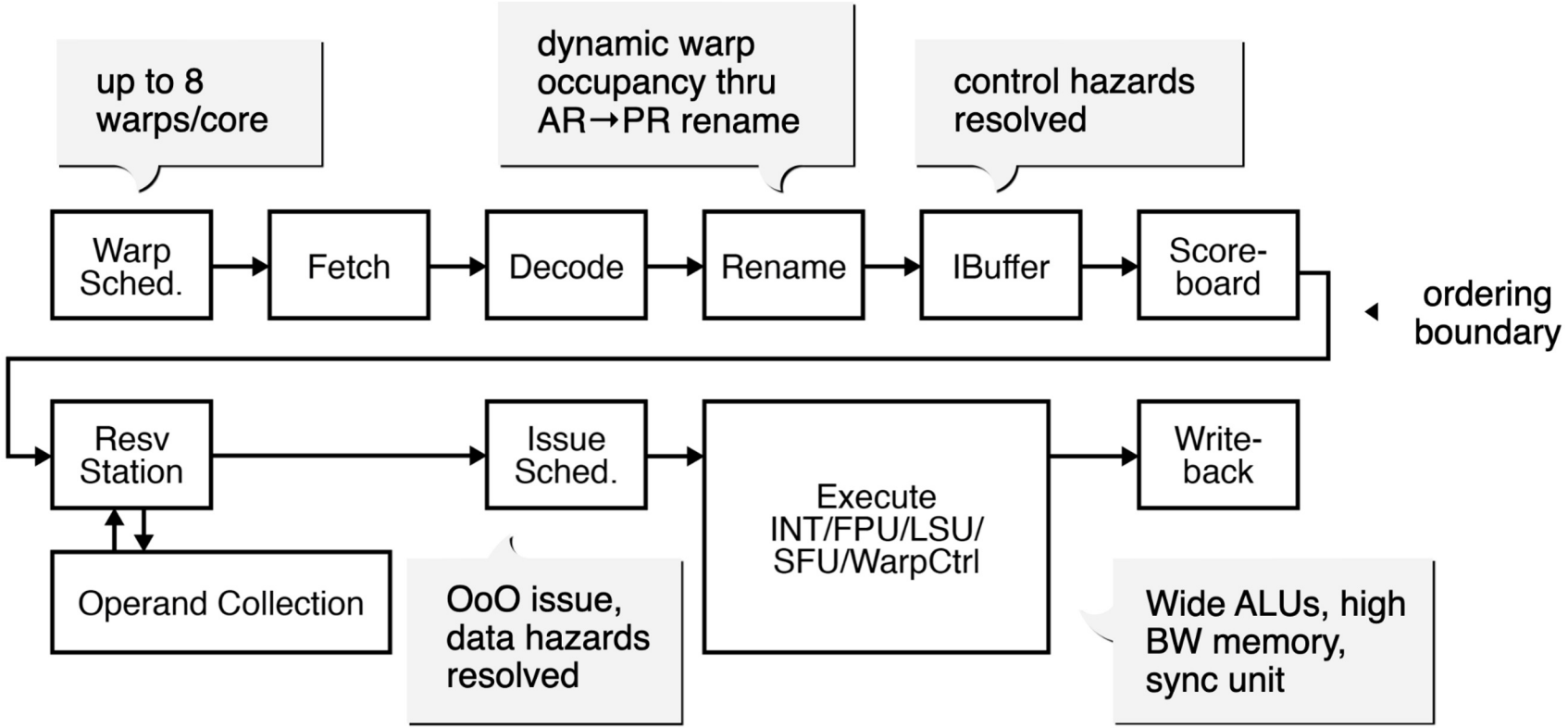
Lightweight out-of-order issue to leverage ILP for low-TLP kernels

Open-source and parameterized in Chisel with SoC integration

Tapeout in TSMC 16nm with pending bring-up

Microarchitecture of Muon

Pipeline Microarchitecture

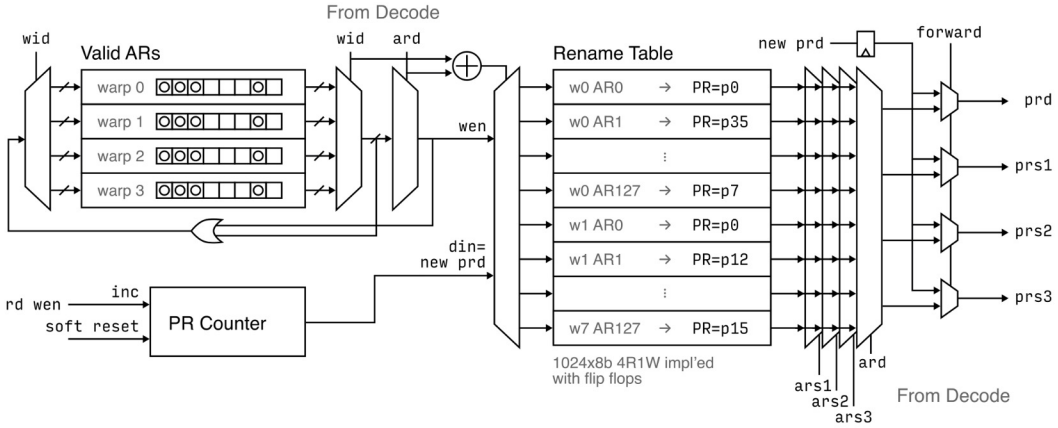


Register Renaming

Architectural register to physical register rename

- **At compile time**
  limit per warp reg usage → higher warp occupancy
- **At run time**
  first seen arch RD→ assign new PR
  - *reset at kernel boundary*

ILP-TLP Tradeoff

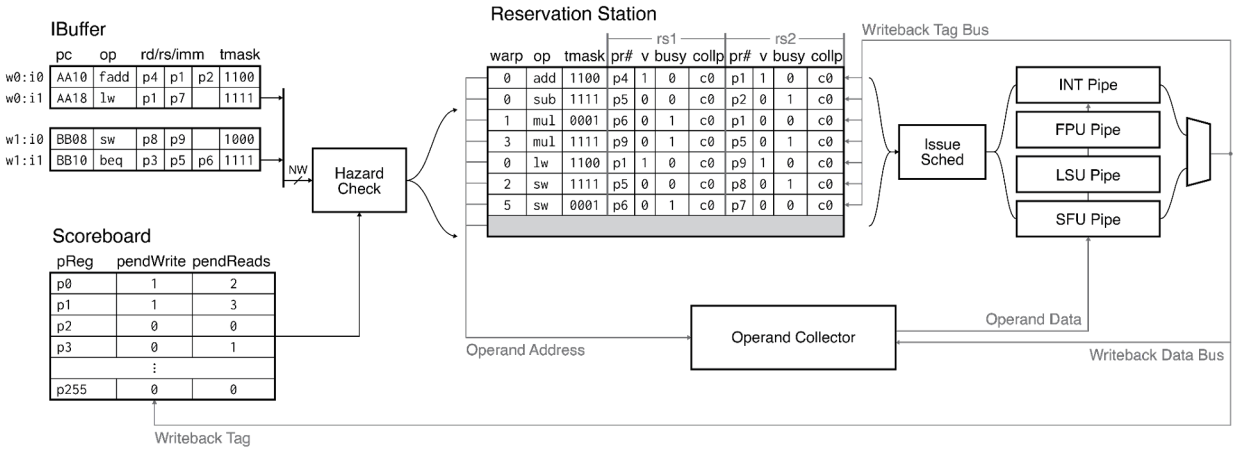


Out-of-Order Issue

Out-of-order issue

- **Scoreboard**
  bookkeeps pending register ops, informs RS hazard gating & forwarding
- **Reservation Station**
  issues instructions past blocked head-of-line *within* the warp

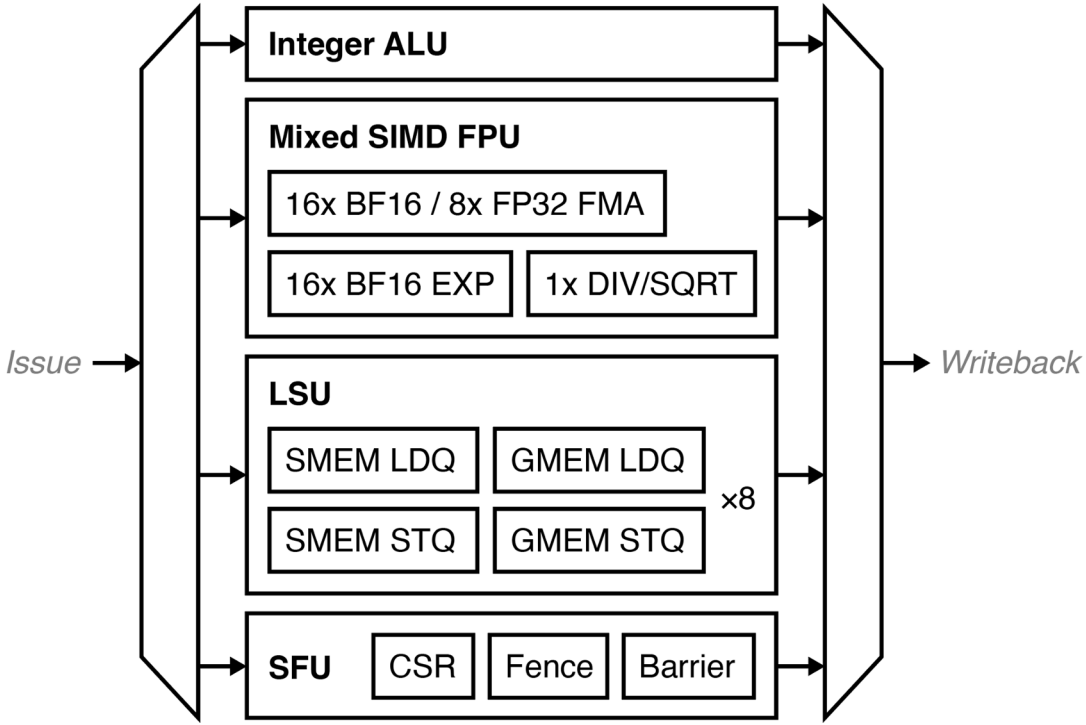
Recovers ILP at low-TLP regime



Execution Pipeline

Execute

- Mixed SIMD BF16/FP32 **FPU** pipeline
- Dedicated LUT based BF16 **EXP** pipeline
- **SFU** for core or SM-wide sync, fences, cache flushes
- **LSU** with relaxed R→R ordering, attached **memory coalescer**
- **Parameterized** per-pipe rates



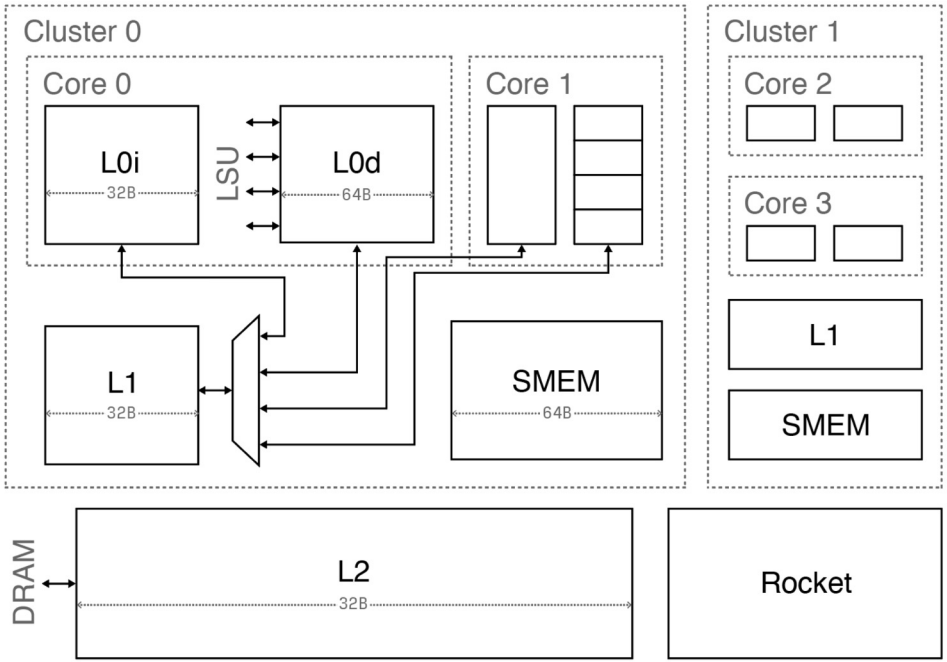
Memory Subsystem

Global Memory

- SM-level weak memory consistency with fence + sync
- Write-back L0, incoherent with flush

Shared Memory

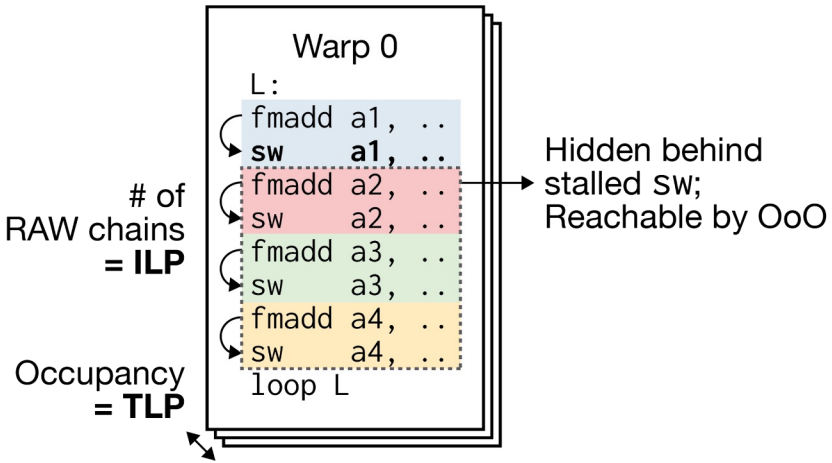
- 128KB/cluster combined SMEM memory
- High BW with lane-bank crossbar for random SIMT access



Evaluation of Muon

Performance: ILP × TLP

Q: Can Muon leverage ILP to hide latencies at low-occupancy regime?

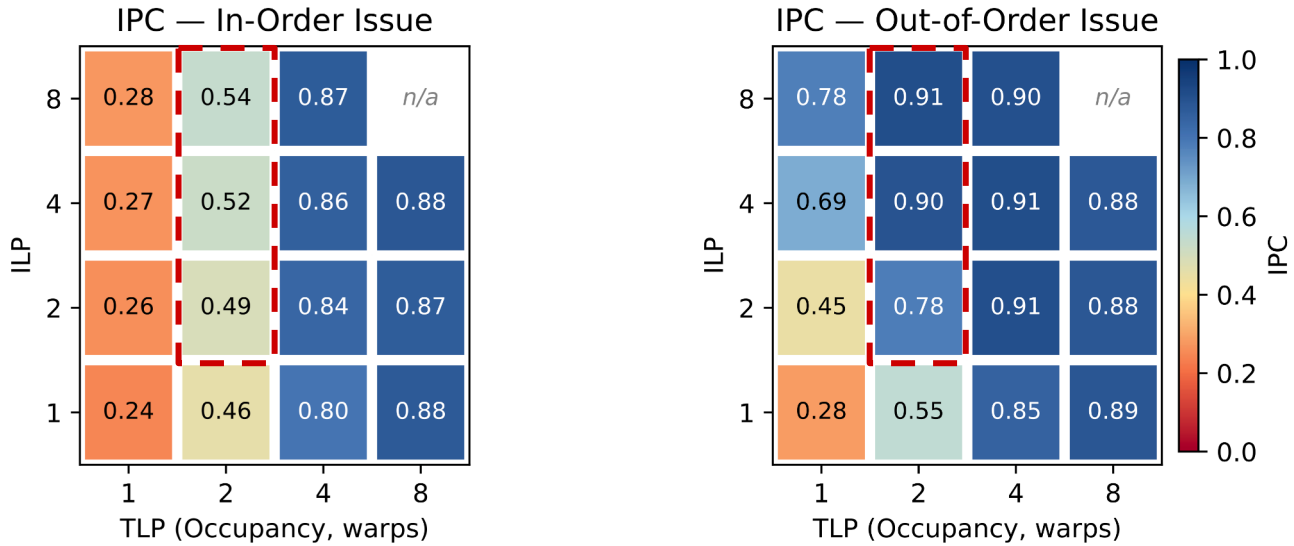


µbench: Loop of `fmadd-sw` RAW chains

- In-order stalls at every `sw`, exposing `fmadd` latency
- OoO issues past **head-of-line** and hides latency even at TLP=1

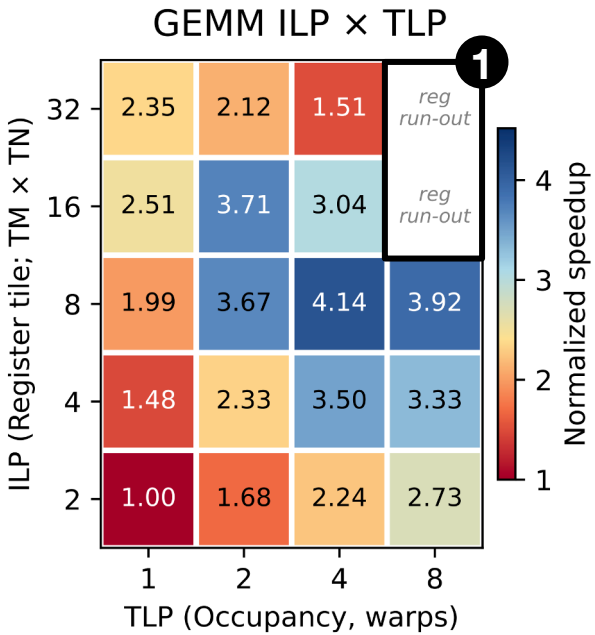
In-order: Modeled by gating issue logic to program order

Performance: ILP × TLP



- When occupancy runs out, in-order falls apart
- OoO issue **saturates IPC at low-occupancy** (ILP=4, TLP=2)
  → Muon can hide latencies well when TLP is scarce

Performance: GEMM



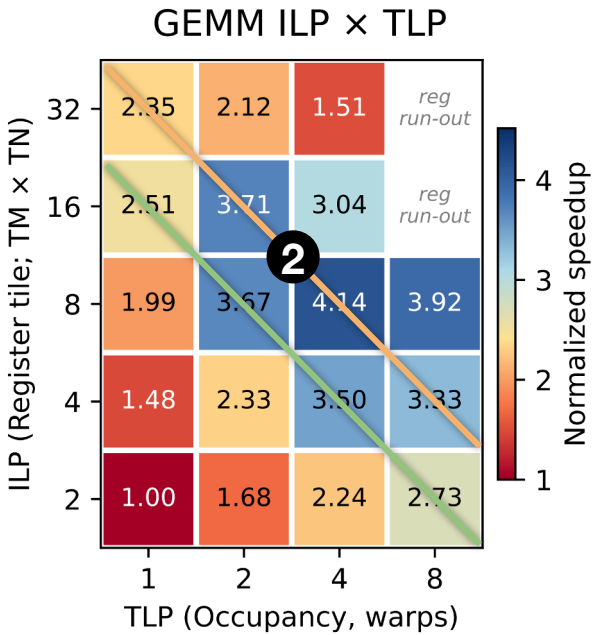
Q: Do these findings generalize to a substantial GEMM kernel[1]?

1 **Occupancy ↔ register space tradeoff:**
High ILP cannot run at high TLP
due to register pressure

Muon's register renamer enables
trading occupancy for
more register space & ILP headroom

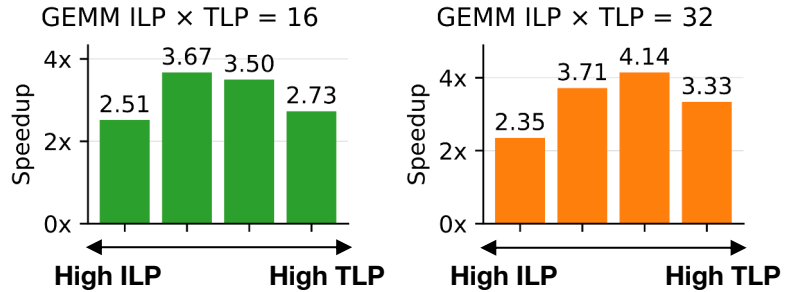
[1] S. Boehm, "How to Optimize a CUDA Matmul Kernel for cuBLAS-like Performance: a Worklog," https://siboehm.com/articles/22/CUDA-MMM

Performance: GEMM



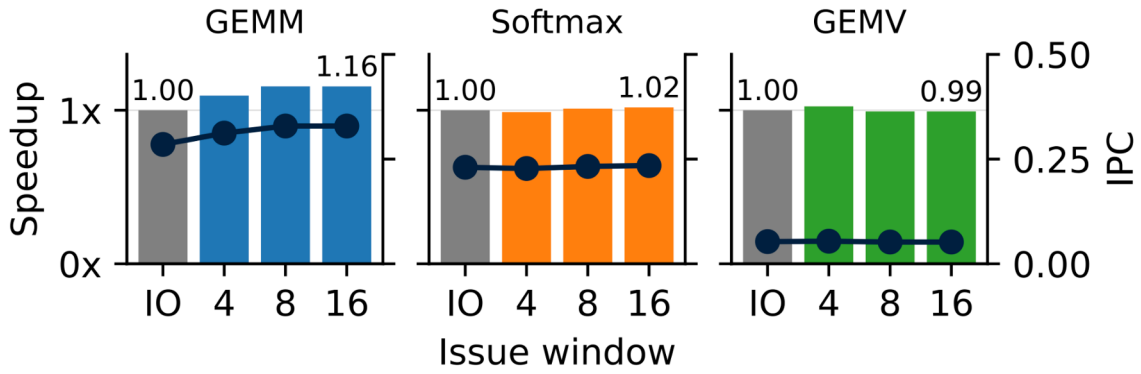
2 **ISO-parallelism frontier**:
ILP-TLP has a *sweet-spot*

High ILP (= larger TMxTN tiles, more data reuse) help performance, but extremely low TLP cancels its benefit



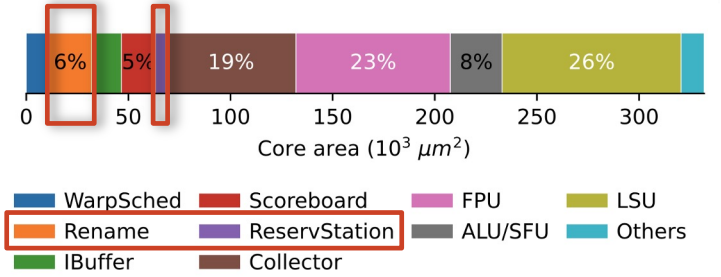
Muon enables modeling interesting **ILP vs TLP tradeoff** in GPU kernels

OoO vs In-Order Performance

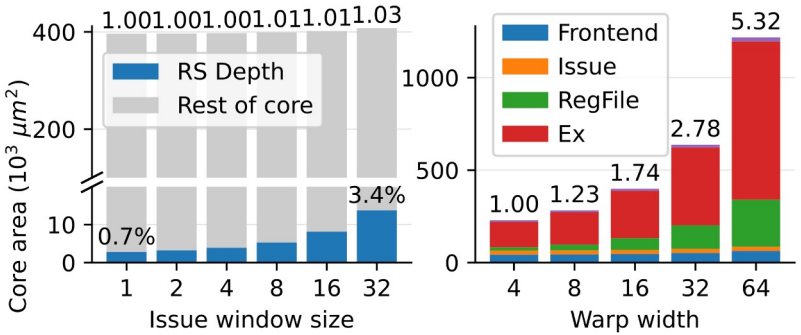


- Out-of-order gives **16%** speedup vs. in-order for GEMM
  → Muon uncovers additional ILP beyond manual best-effort scheduling
- Speed is less pronounced for **memory-bound kernels**;
  TLP remains the main leverage for hiding memory latency

Area Breakdown



Post-synthesis Area Breakdown



Area scaling behavior

- Occupancy/OoO logic consumes **<8%** of total core area
  - Reservation station scales linearly to depth; remains a small portion
- **Sub-linear** area scaling to SIMD warp width
  - Frontend/OoO area stays largely constant

Conclusion

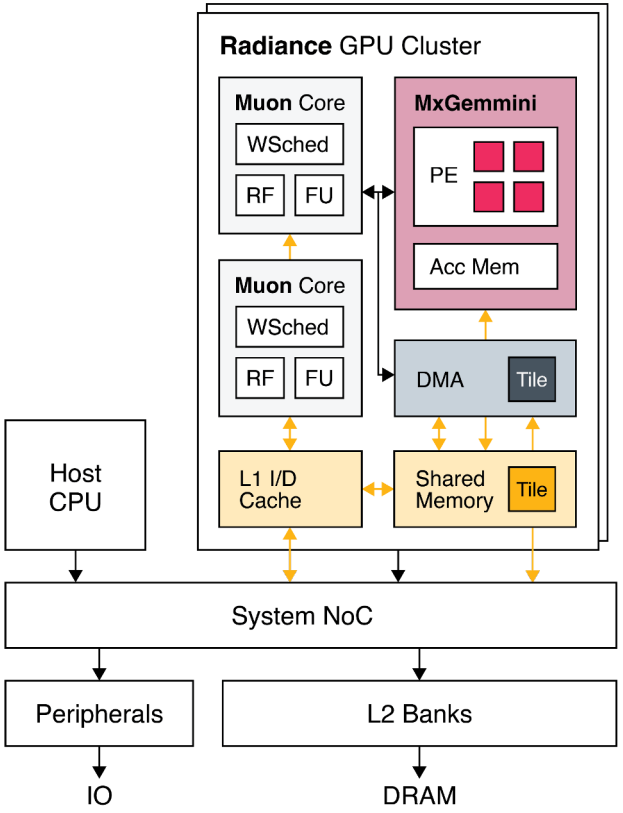
Muon is:

- An ASIC-optimized SIMT core that enables **ILP-TLP tradeoff** in GPU kernels
- Open-source in Chisel; SoC-integrated via Chipyard
- Taped-out as part of **Radiance**, an SoC with AI-accelerator-integrated GPU

Future work

- More kernels: FlashAttention-4, TinyLLama
- LSU/memory-system optimization
- Chip bring-up and power measurement

RTL & docs: `github.com/ucb-bar/radiance`
Kernels: `github.com/ucb-bar/radiance-kernels`



Radiance GPU SoC

Thank You!

Hansung Kim hansung_kim@berkeley.edu
Richard Yan yrh@berkeley.edu
Shashank Anand shashank.anand@berkeley.edu