

---

# Power Analysis of Memory Hierarchies in **RISC-V** Systems using gem5

Raghav Donakanti, Roja Sahoo

---

# Introduction

Evaluate power-performance trade-offs in RISC-V memory hierarchies using gem5.

Analyze the impact of cache parameters (size, associativity, block size) on power and performance.

## Possible Key Insights:

- Cache parameters significantly affect power and performance.
- Compute vs. memory-bound workloads exhibit distinct cache and memory behavior.
- Optimal configurations minimize power without compromising performance.

# Gem5

## What is Gem5?

An open-source, event-driven simulator for microarchitectural and system-level modeling.

### Modes of Operation:

- *Full System (FS)*: Simulates complete systems, including devices and OS.
- *Syscall Emulation (SE)*: Emulates system services for user-space programs.

### Usage in Project:

Configured RISC-V cores with the se.py script and customized cache and memory hierarchy parameters for analysis.

# McPAT

## What is McPAT?

A framework for power, area, and timing modeling of manycore processors and system components.

### Capabilities:

- Models power and energy using hardware parameters and ITRS projections, with detailed breakdowns across components

### Usage in Project:

Processed gem5 statistics into XML for McPAT and generated power consumption data by cache and memory hierarchy.

# Overview

**gem5 Configuration:** Simulate RISC-V cores with adjustable L1/L2 caches and DRAM configurations.

## Workload Selection:

- AES: Compute-bound benchmark.
- Tiled MatMul: Memory-bound benchmark.

**Power Estimation:** Integrate McPAT for power profiling based on gem5 activity statistics.

**Simulations:** Run benchmarks across varied cache configurations. Collect cache hit/miss rates, power usage, and simulation time.

**Trade-off Analysis:** Compare configurations to identify optimal cache setups for power efficiency and performance balance.

# Our Main Objectives

## **Q1.** How Does Power Consumption and Cache Behavior Vary with Tile Sizes in Tiled MatMul for Different Matrix Sizes?

Analyzing the impact of tile sizes on cache hits/misses and power consumption, with a focus on optimizing tile size for larger matrices.

## **Q2.** Which Program, AES or Tiled MatMul, Uses More Power and How Does DRAM Type Affect Their Cache Usage?

Comparing power consumption patterns of AES and Tiled MatMul, and examining the effect of different DRAM types on cache behavior.

## **Q3.** How Does Power Consumption Vary Between AES and MatMul with Different Cache Size and Associativity Configurations?

Investigating the effect of cache size and associativity on power consumption and cache efficiency for both AES and MatMul.

# Our Main Parameters

Parameter	Description
--l1i_size	Sets the size of the Level 1 instruction cache (L1I).
--l1d_size	Sets the size of the Level 1 data cache (L1D).
--l2_size	Sets the size of the Level 2 cache (L2).
--l1d_assoc	Defines the associativity of the L1 data cache (how many locations in the cache a particular block of memory can be stored in).
--l1i_assoc	Defines the associativity of the L1 instruction cache.
--l2_assoc	Defines the associativity of the Level 2 cache.
--cacheline_size (block size)	Sets the size of cache lines in bytes (amount of contiguous memory data that is transferred between the main memory and the cache in a single operation).
--mem-size	Specifies the total memory size for the simulated system.
--num-cpus	Defines the number of CPUs in the simulation.
--cpu-clock	Sets the clock frequency of each CPU core.
--smt	Enables or disables simultaneous multi-threading (SMT) for CPUs.
--cpu-type	Sets CPU types among:
--mem-type	Defines the type of memory to use in the simulation.
--caches	Enable to use caching (L1 cache).
--l2cache	Enable to use L2 cache.

# How we varied parameters?

## MatMul Params

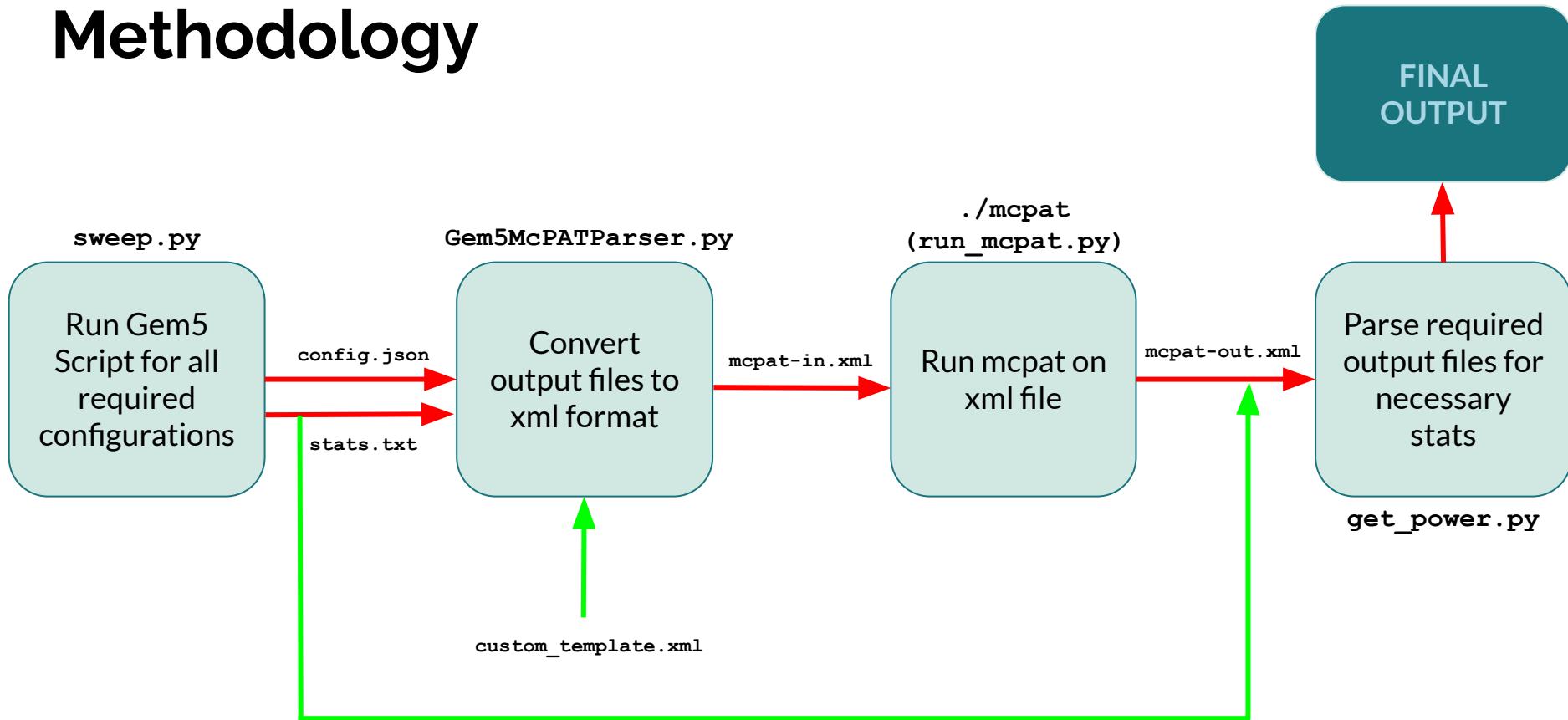
```
{  
    "program_path": "matmul.cpp",  
    "parameters": [  
        {"name": "N", "values": [64, 128]},  
        {"name": "TILE_SIZE", "values": [16, 32, 64]},  
    ]  
}
```

## Default Cache Params

### Bash ▾

```
{"name": "l1d_size", "values": ["32kB"]},  
{"name": "l1i_size", "values": ["32kB"]},  
{"name": "l1d_assoc", "values": [2]},  
{"name": "l1i_assoc", "values": [2]},  
{"name": "l2_size", "values": ["2MB"]},  
{"name": "l2_assoc", "values": [8]},  
{"name": "cacheline_size", "values": [64]},  
{"name": "mem-size", "values": ["8192MB"]},  
{"name": "mem-type", "values": ["DDR5_8400_4x8"]},  
{"name": "cpu-type", "values": ["Riscv03CPU"]},  
{"name": "num-cpus", "values": [1]},  
{"name": "cpu-clock", "values": ["2GHz"]}}
```

# Methodology



# Our Output Stats

## 3.4.1 From stats.txt:

- system.cpu.icache.overallHits::total
- system.cpu.icache.overallMisses::total
- system.cpu.dcache.overallHits::total
- system.cpu.dcache.overallMisses::total
- system.l2.overallHits::total
- system.l2.overallMisses::total
- system.mem\_ctrls.dram.rank0.averagePower

## 3.4.2 From McPAT Output (all in Watts):

- Instruction Cache Runtime Dynamic Power
- Data Cache Runtime Dynamic Power
- L2 Cache Runtime Dynamic Power
- Bus Cache Runtime Dynamic Power

# Experiment 2

To compare the power consumption of AES and Tiled MatMul, analyze the impact of DRAM type on cache usage, and investigate how these factors influence their power and performance characteristics.

# Expectations

## 1. MatMul vs AES: Memory and CPU Dependence

*MatMul:* Memory-bound for small inputs, compute-bound for large matrices, dependent on matrix size and cache.

*AES:* Cache-dependent with high locality and smaller data, less impacted by high-bandwidth RAM.

## 2. Impact of DRAM Types

*MatMul:* Gains from high-bandwidth RAM (DDR5, HBM) for reduced time and power per byte, especially with larger matrices.

*AES:* Minimal benefit from high-bandwidth RAM due to lower operational intensity.

## 3. Memory Access Behavior

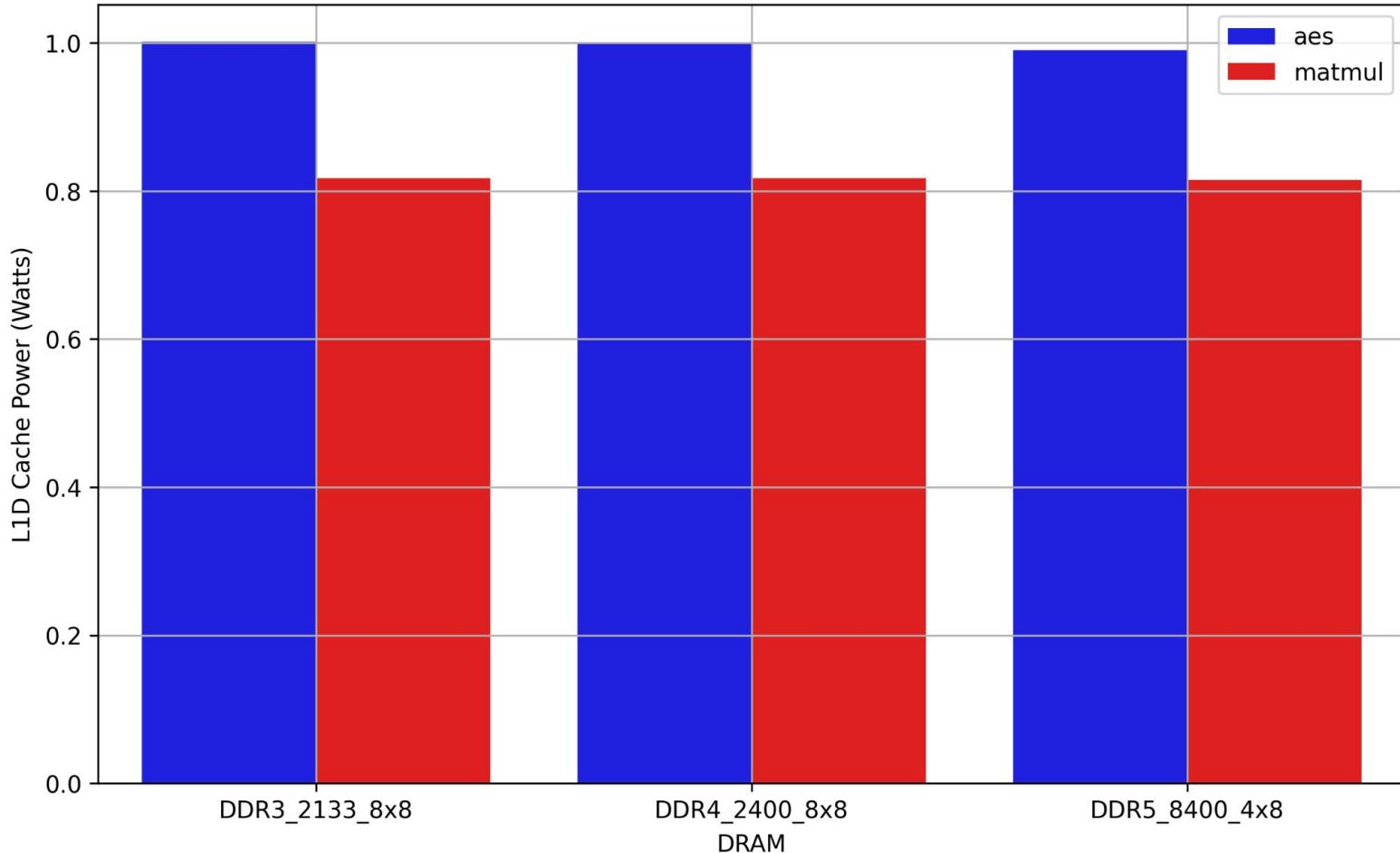
*MatMul:* More main memory accesses as data exceeds cache capacity.

*AES:* Efficient within cache, fewer main memory accesses.

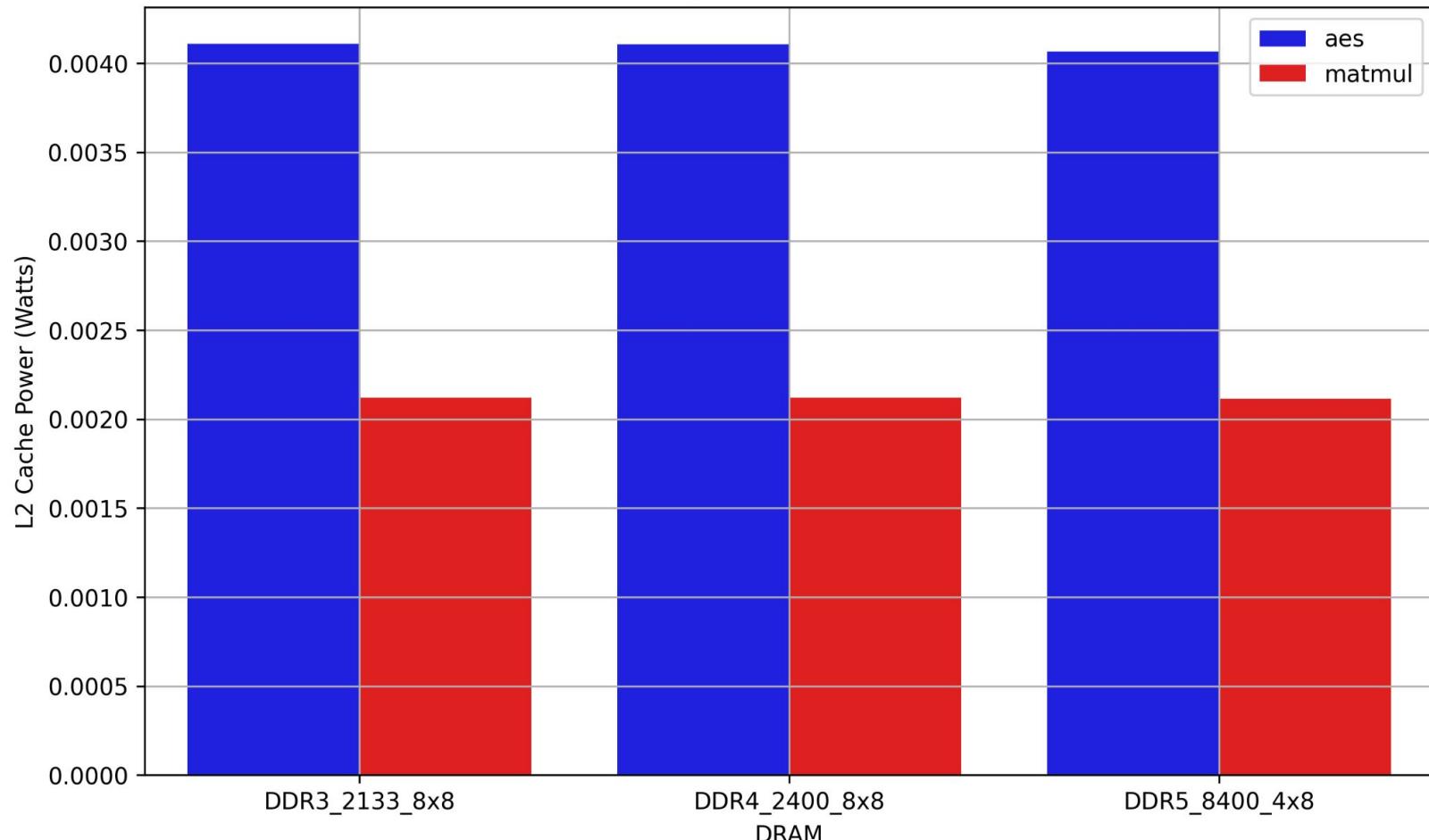
## 4. Modern DRAM (DDR5)

Reduces latency and power consumption, enhancing performance.

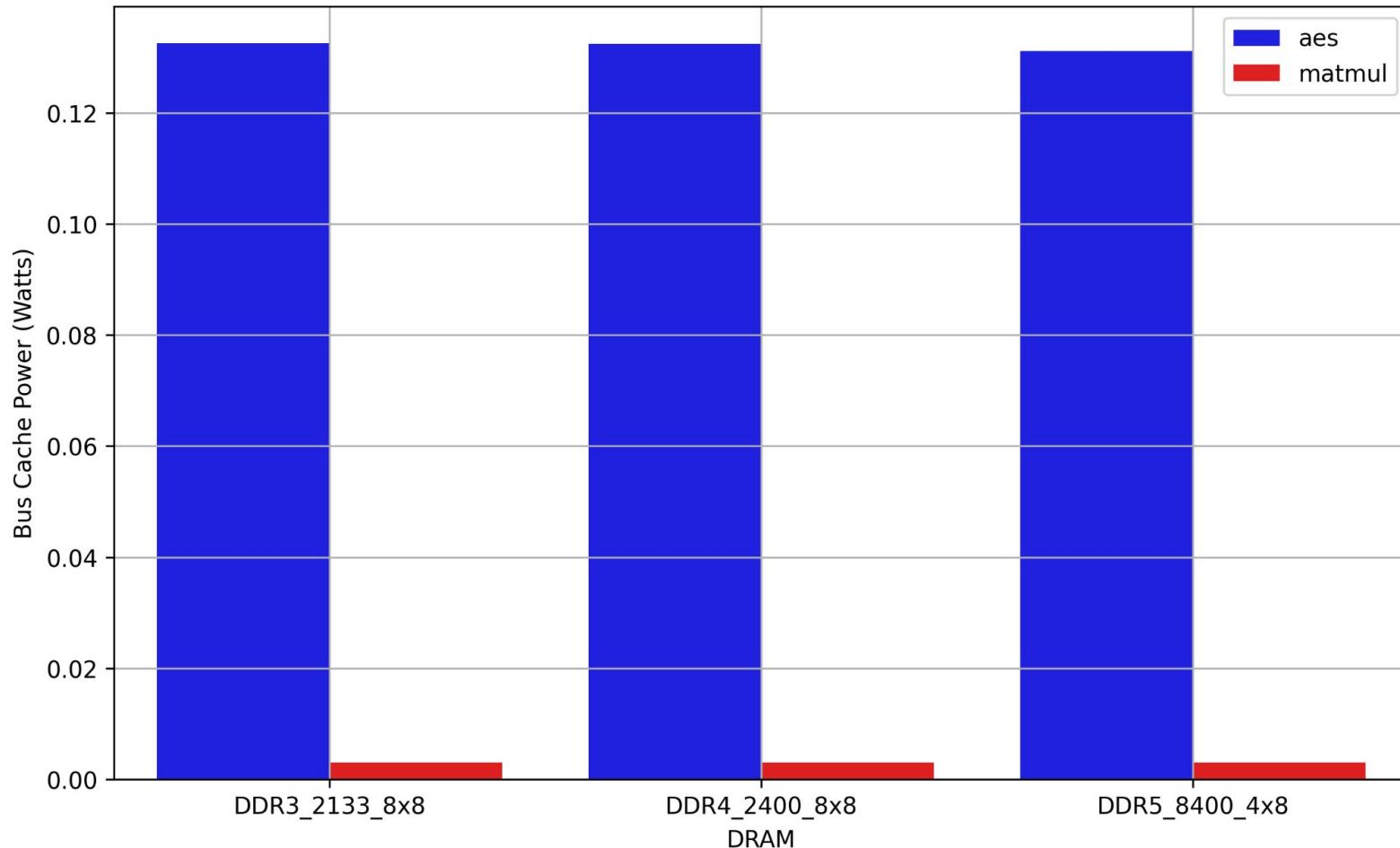
### Matmul/AES L1D Cache Power



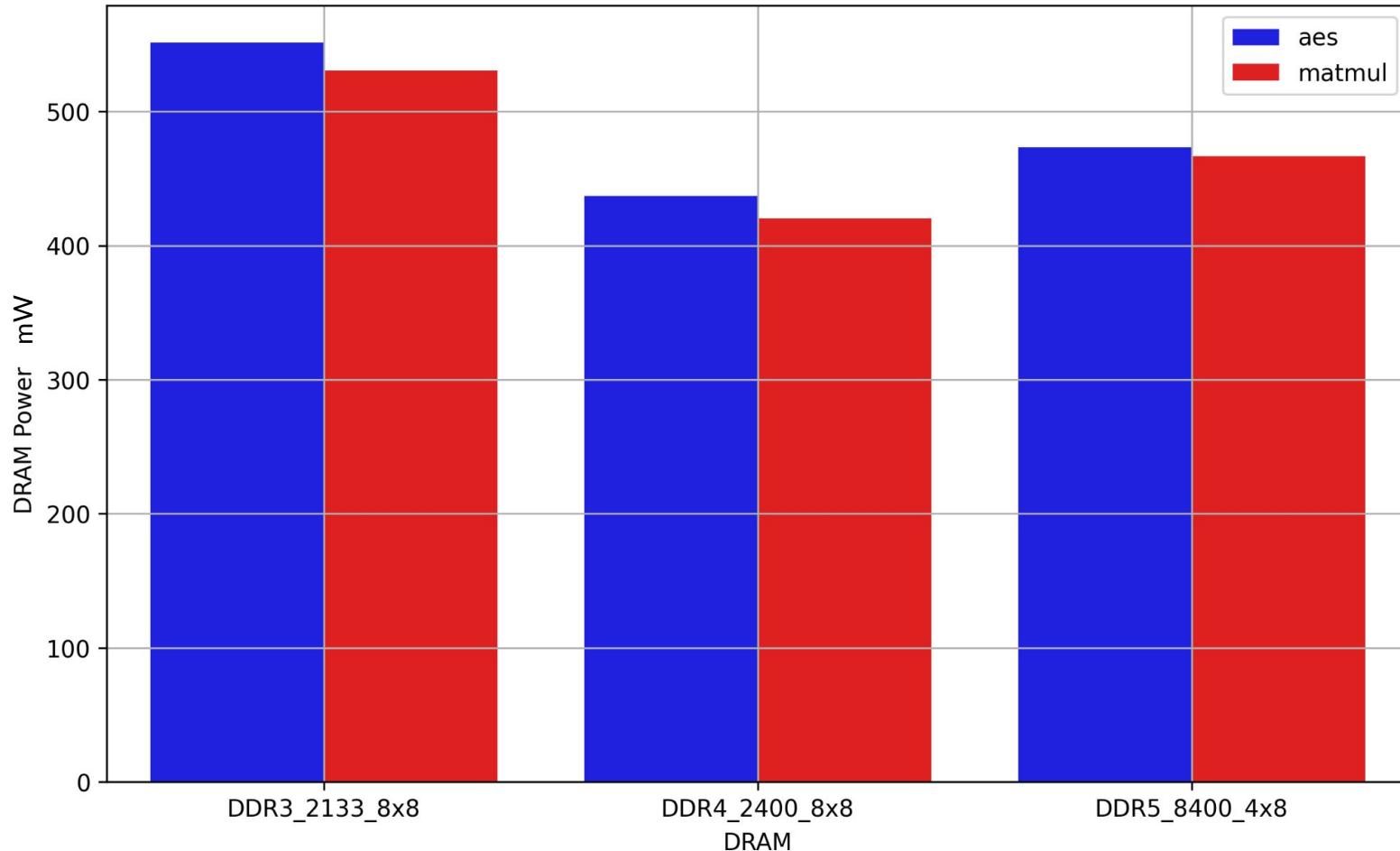
### Matmul/AES L2 Cache Power



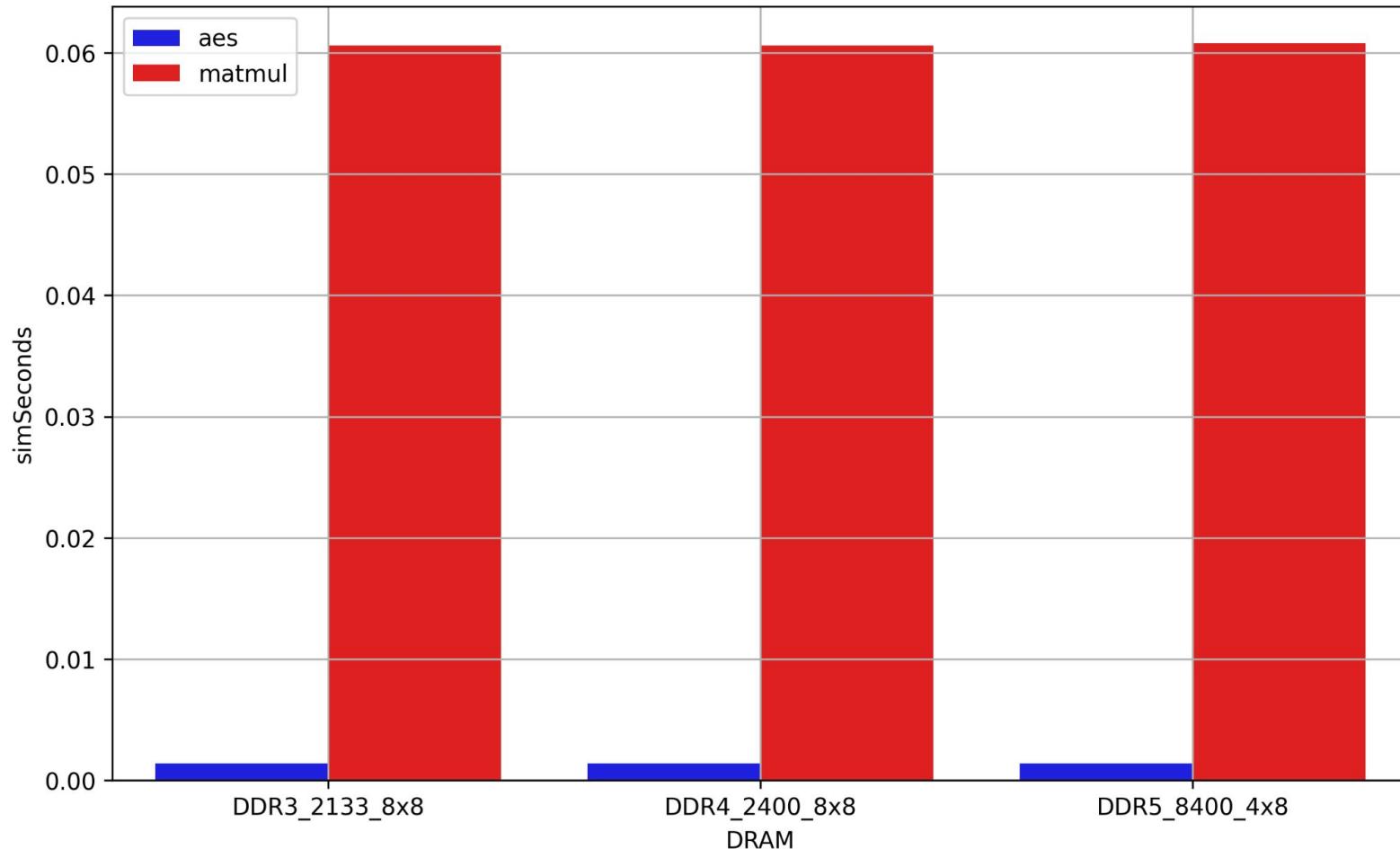
### Matmul/AES Bus Cache Power



### Matmul/AES and DRAM Power



### Matmul/AES simSeconds



# Experiment 1

To investigate how power consumption and cache behavior (hits and misses) are influenced by varying tile sizes in Tiled MatMul, and how these effects differ for large and small matrix sizes.

# Expectations

## 1. Tiled Matrix Multiplication Optimization

- Reduces data movement from DRAM and speeds up memory access.
- Minimizes cache misses by fetching smaller square matrices (tiles).
- Tile size must fit within L1D or L2 cache for optimal performance.

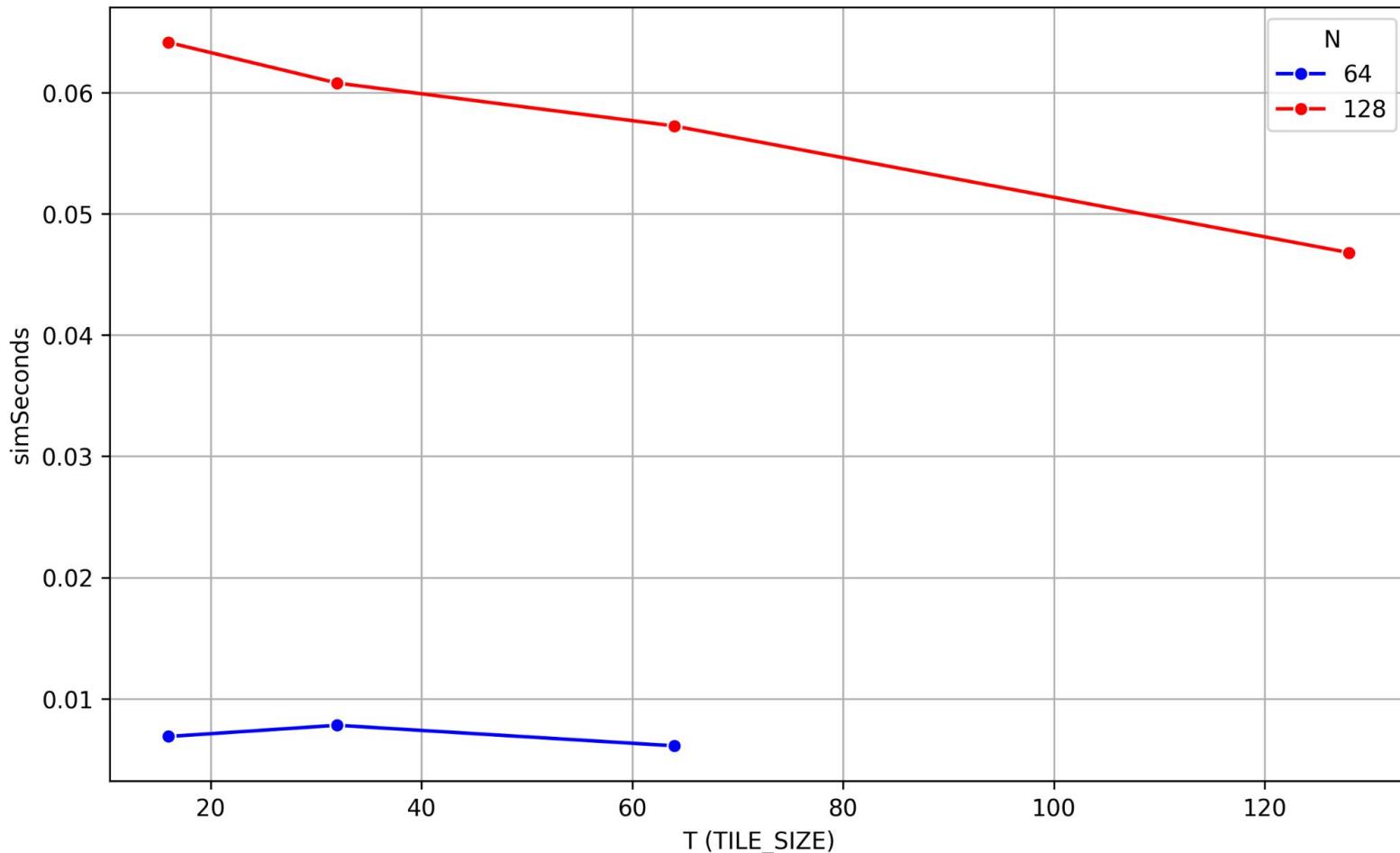
## 2. Effect of Matrix Size and Tile Size on Cache Performance

- Larger matrices increase operations, affecting cache performance.
- Tile size impacts cache miss rates, influencing power usage.
- Too large a tile size may lead to more cache misses and higher power consumption.

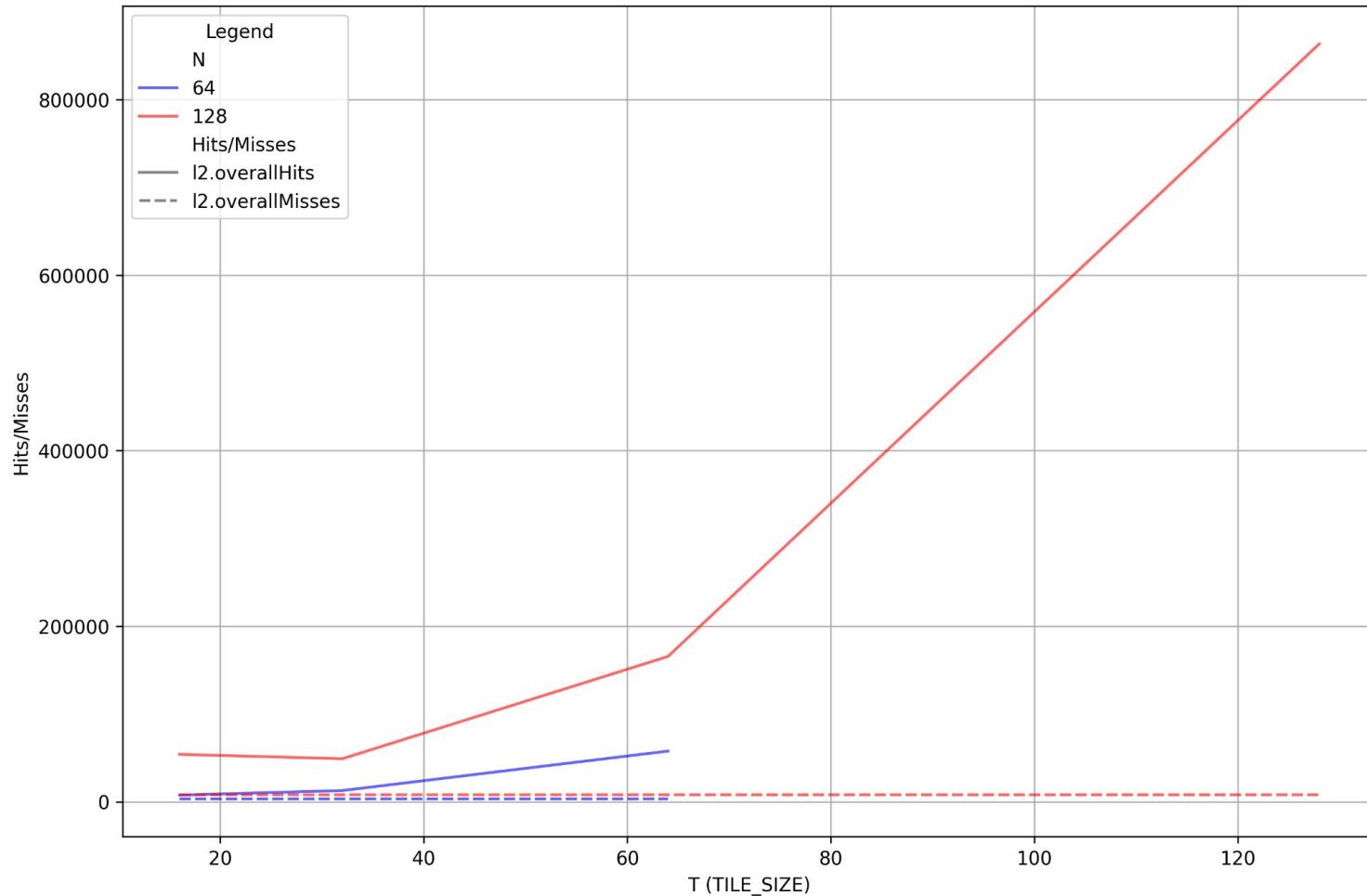
## 3. Optimizing Tile Size for Power Efficiency

- Properly optimized tile sizes reduce memory accesses and improve cache efficiency, leading to lower power consumption.
- Balancing tile size, cache misses, and power is key to performance optimization.

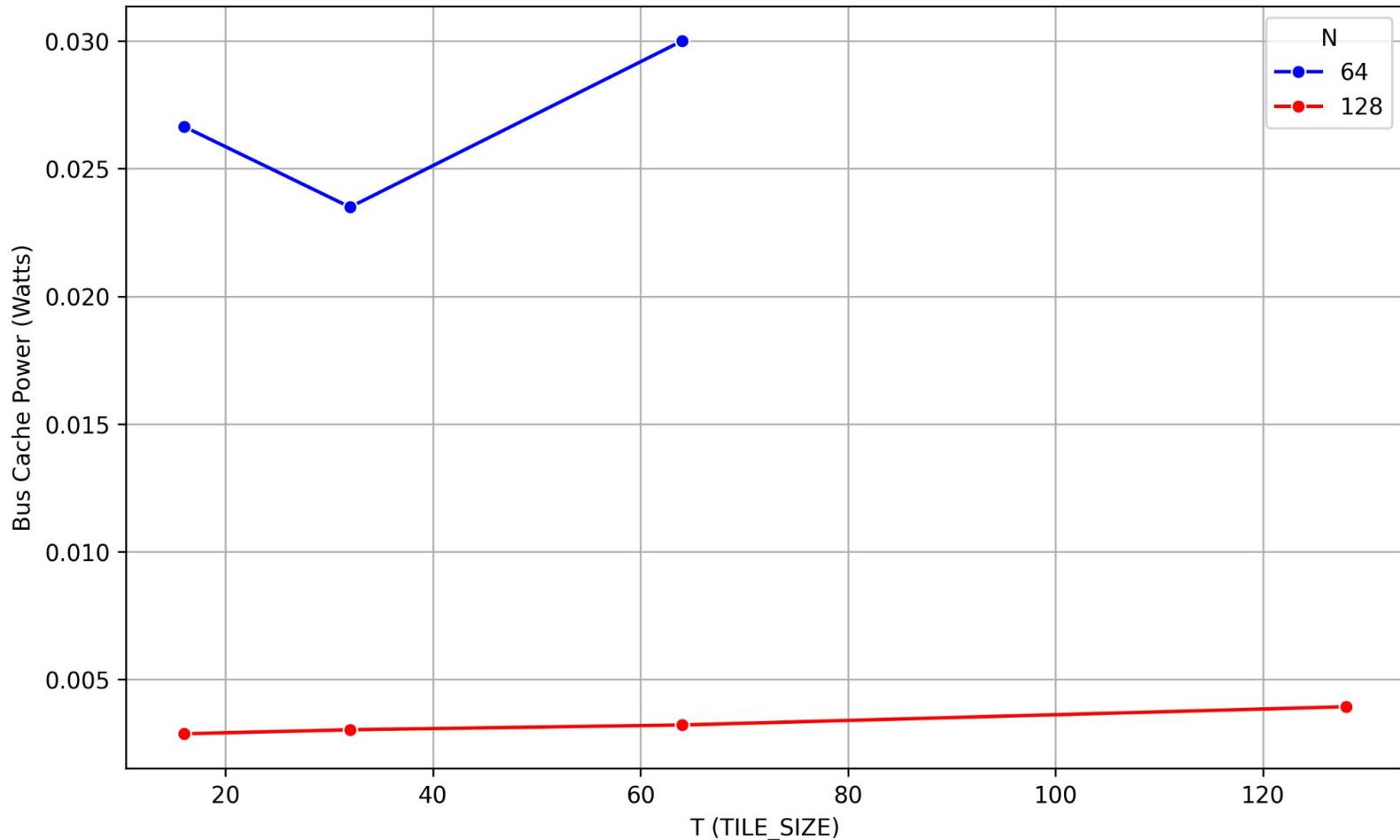
simSeconds vs T



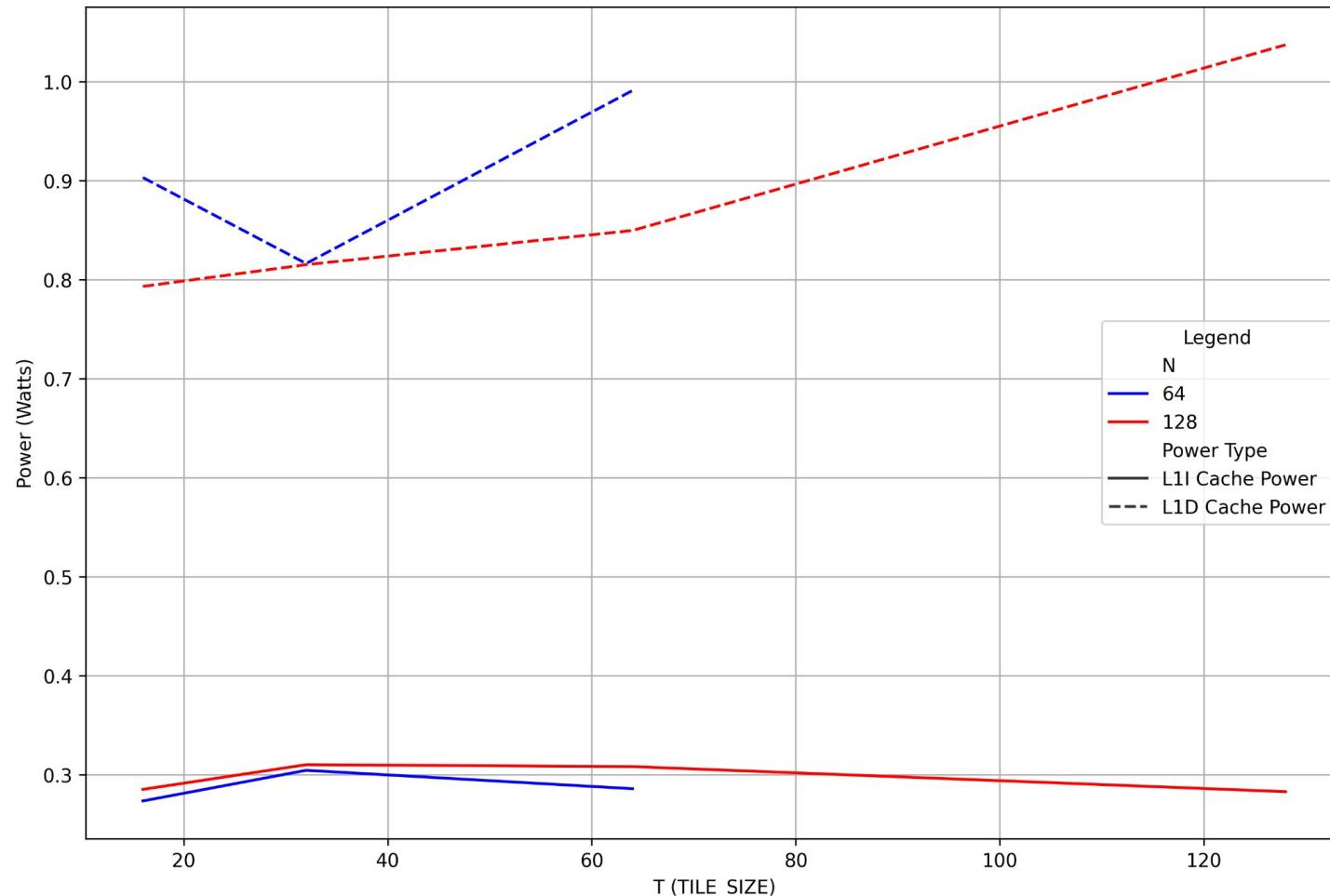
### L2 Cache Hits and Misses vs T



### Bus Cache Power vs T



Various Powers vs T



# Experiment 3

To analyze how cache size and associativity configurations affect power consumption and cache efficiency in AES and MatMul, focusing on cache hits, misses, and the resulting performance-power trade-offs.

# Expectations

## Impact of Cache Size:

- Increasing cache size generally reduces misses and increases hits, improving performance.
- Larger caches consume more power due to higher storage capacity.

## Impact of Cache Associativity:

- Higher associativity reduces conflict misses but slightly increases power consumption due to added hardware complexity.

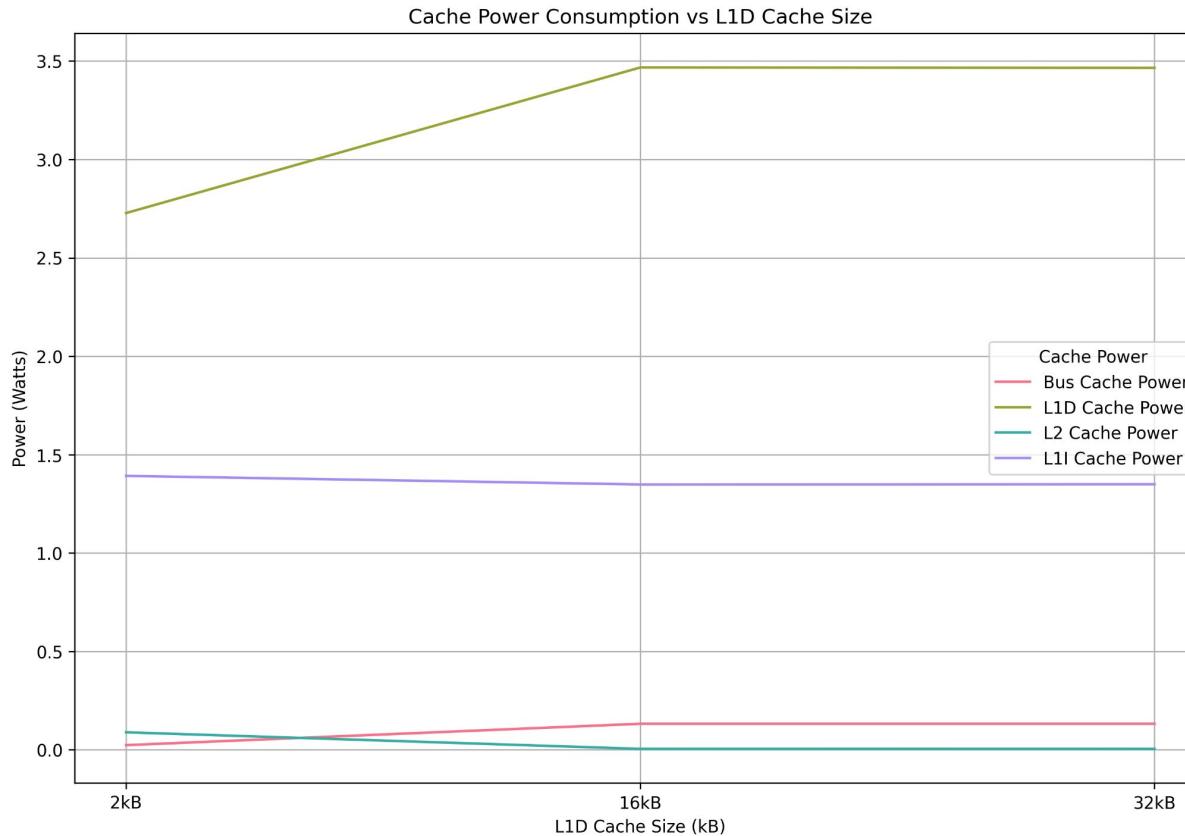
## L1 Cache Dynamics:

- L1 cache services most hits due to proximity to the CPU, leading to faster memory access.
- Shielding L2 from frequent hits reduces L2's power consumption.

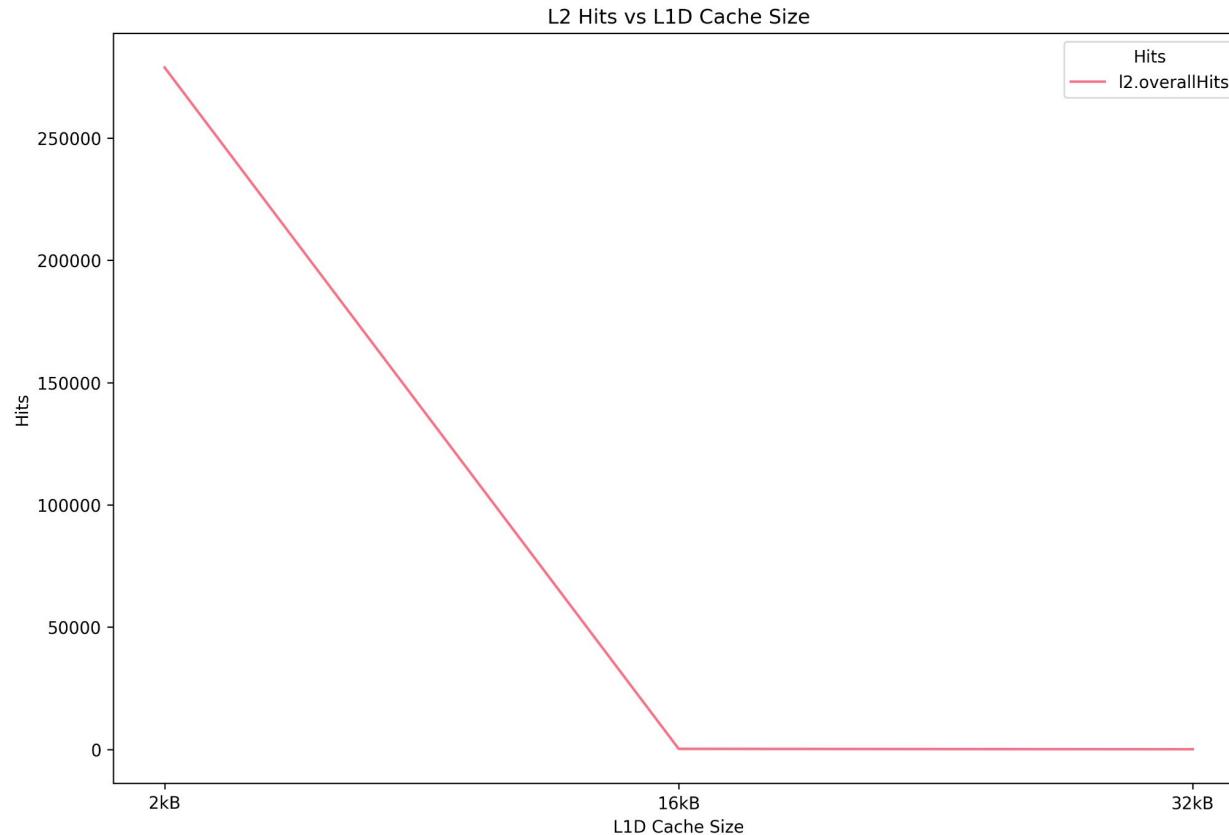
## L2 Cache Dynamics:

- L2 cache's power usage may decrease as L1 absorbs most of the traffic.
- This hierarchical caching enhances overall program performance by efficiently distributing memory operations.

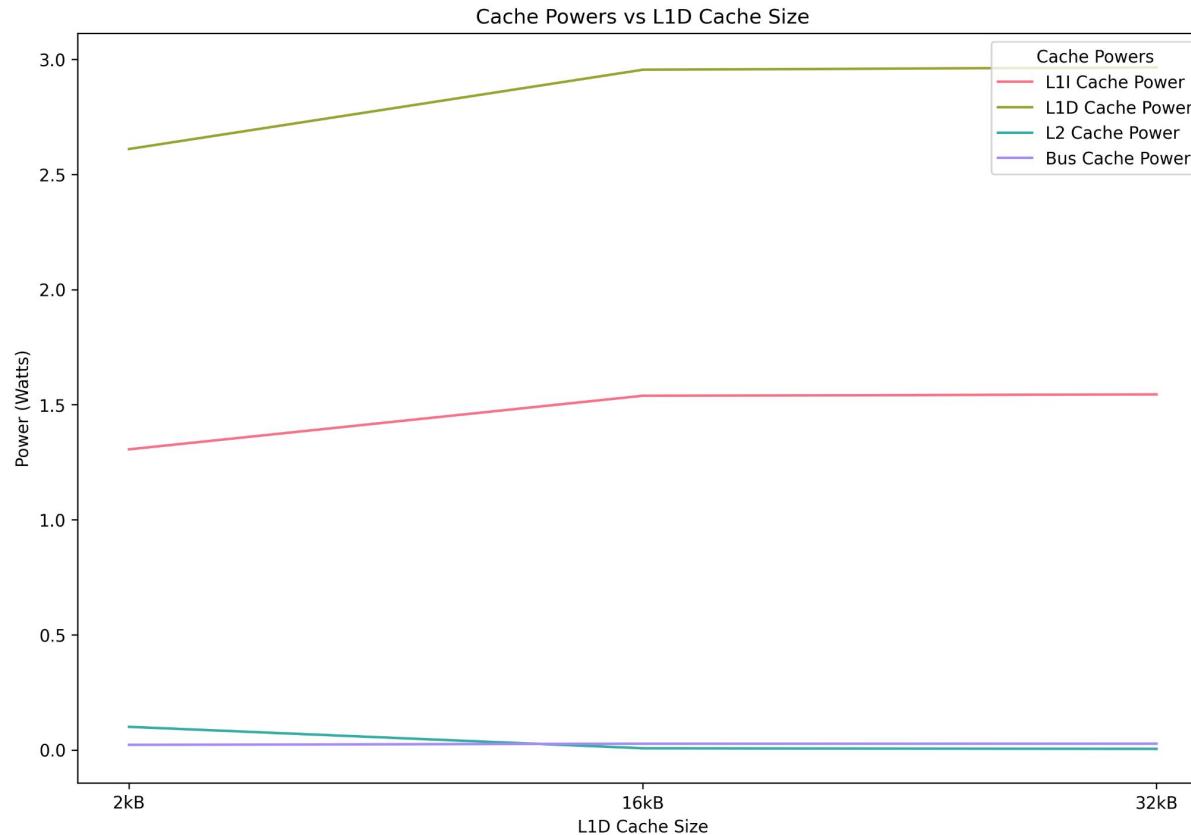
# AES



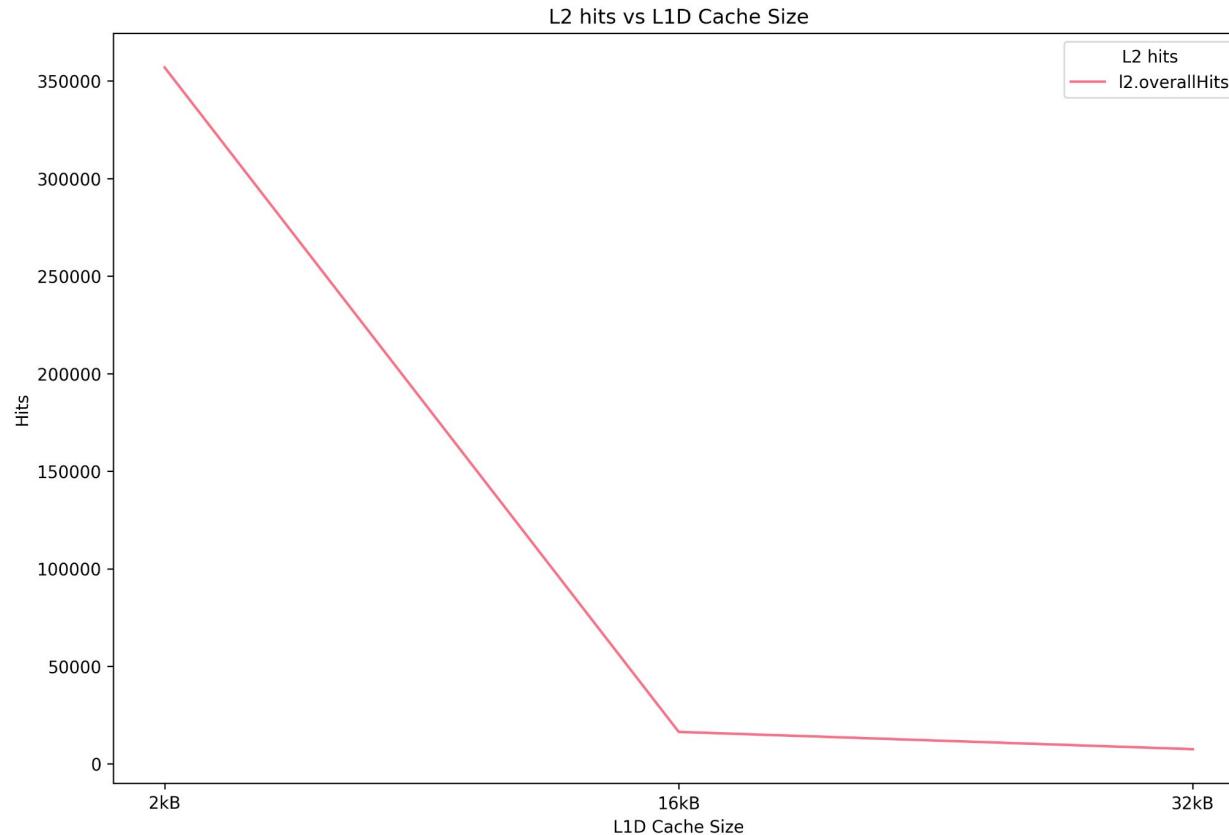
# AES



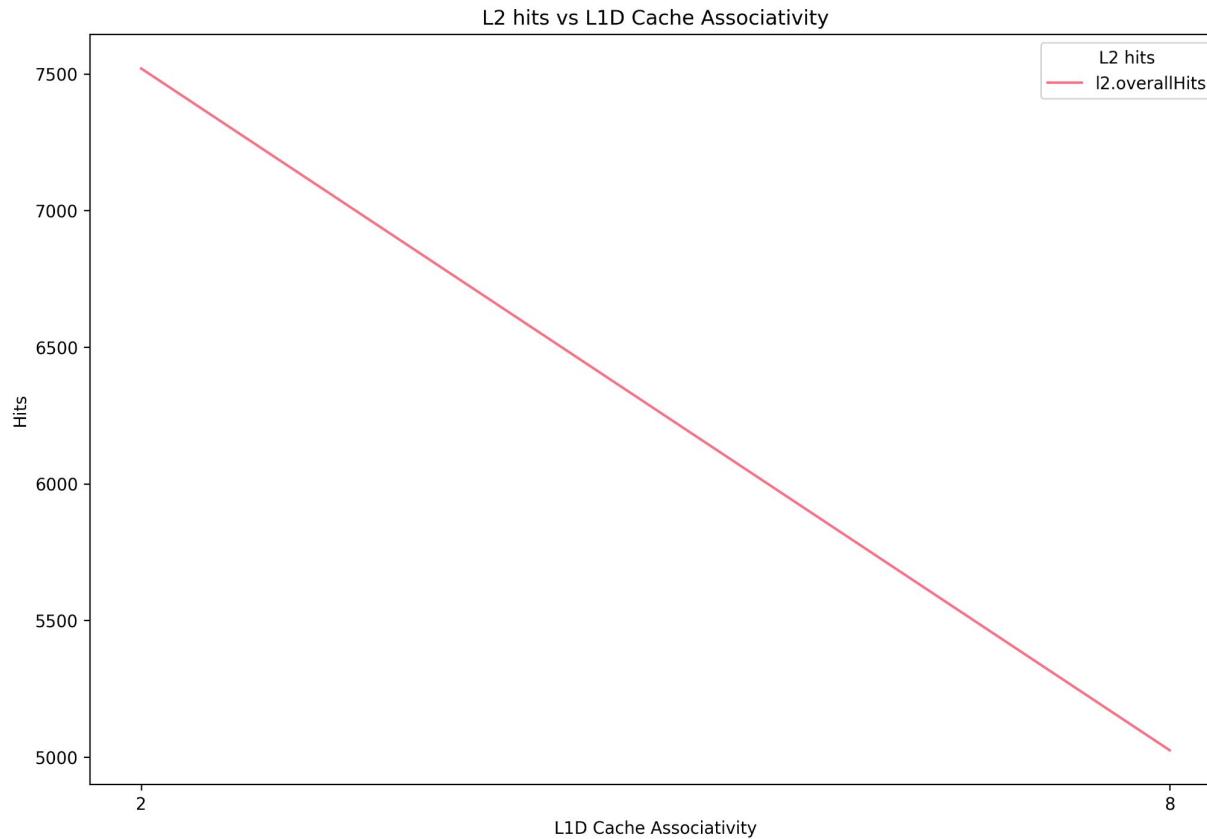
# Matmul



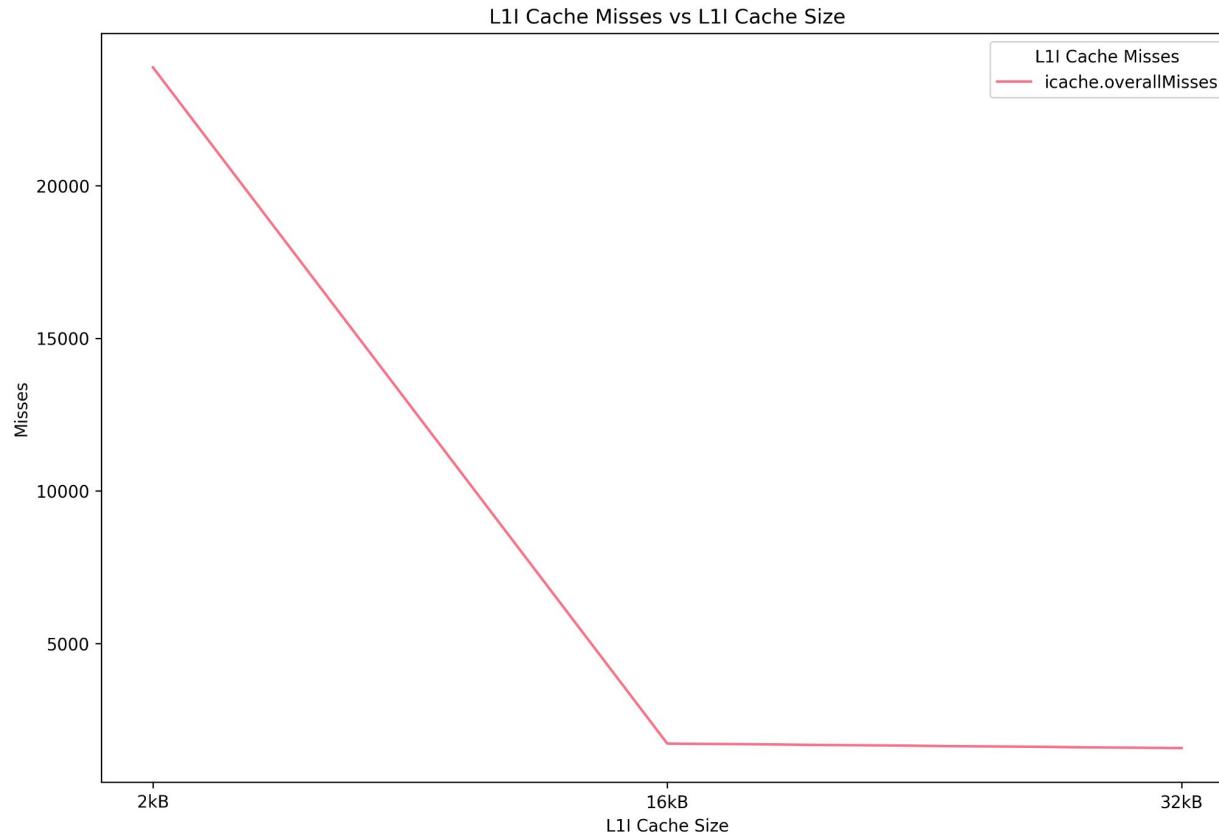
# Matmul



# Matmul



# Matmul



# Key Conclusions

**Tiled MatMul:** Memory-bound, requires more cache than available, leading to higher main memory reliance, lower cache power, fewer hits, and more misses.

**AES:** Compute-bound, fits well within the available cache, experiencing fewer cache misses.

**Execution Time:** Tiled MatMul took longer than AES due to higher memory demands and increased data movement.

**Cache Power:** More cache usage results in higher power consumption, emphasizing the need for cache-power balance.

**Cache Optimization:** Increasing associativity and cache size improves hits, reduces misses, and boosts performance and power efficiency.

**L1 Cache Size:** Larger L1 cache reduces reliance on L2, benefiting memory-bound programs.

## Some Anomalies:

- **L1D hits:** AES decreases, MatMul increases with larger L1D cache.
- DDR5 showed higher time and power consumption due to optimization overhead.

# Limitations & Future Work

- MatMul fails to run for matrix sizes greater than 256 due to CPU usage constraints in the current setup.
- The bounds for AES and MatMul could not be pushed further due to the constraints of the Gem5 simulation environment.
- The circumstances under which the CPU-bound AES program might consume more power than the memory-bound MatMul (ridge point) remain unclear and need further investigation.
- Identifying the optimal configuration for compute and memory-bound programs, and determining the settings that enable the best performance with the least power consumption, remains an open challenge.

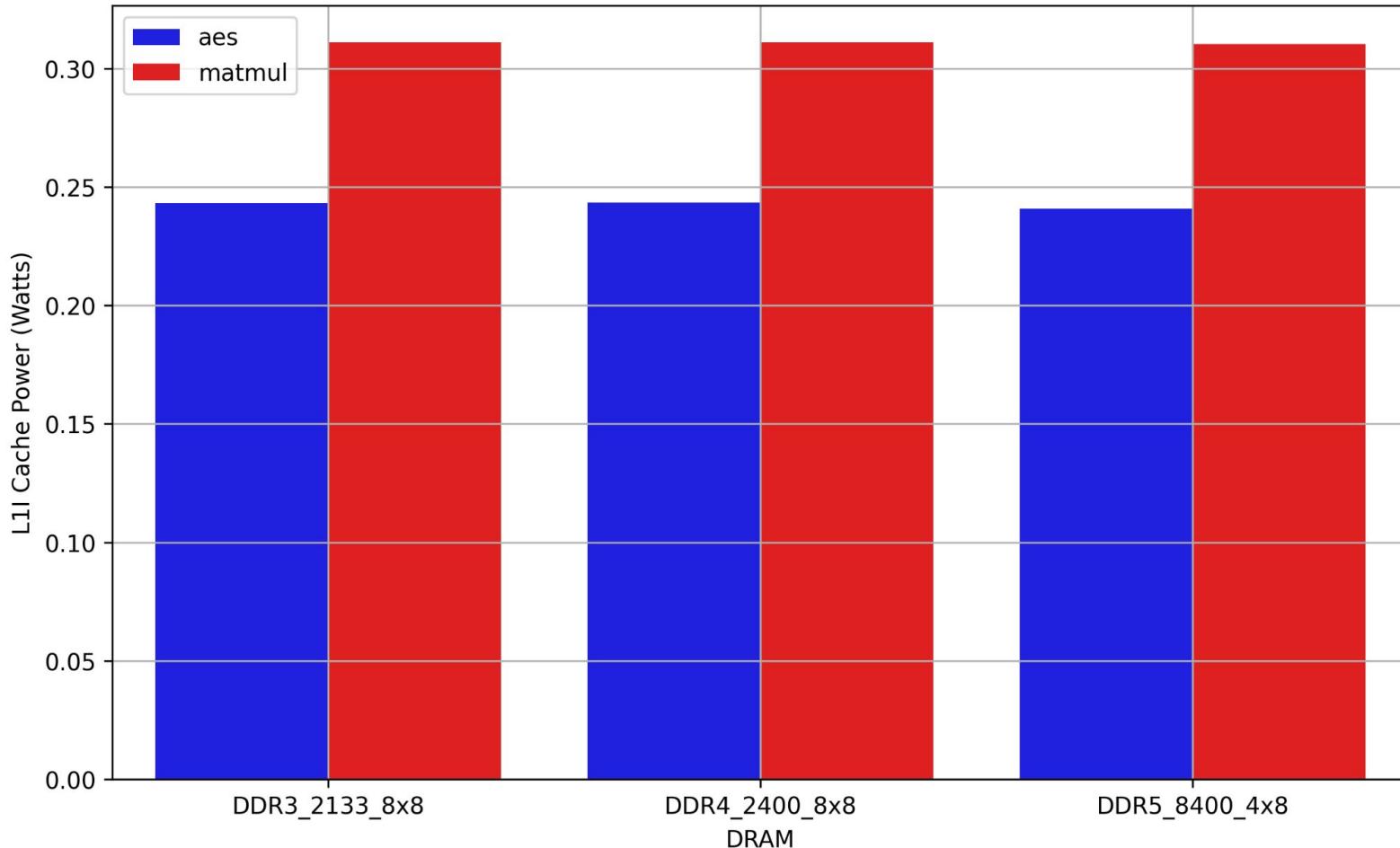
# Resources

[Extensive Results](#)

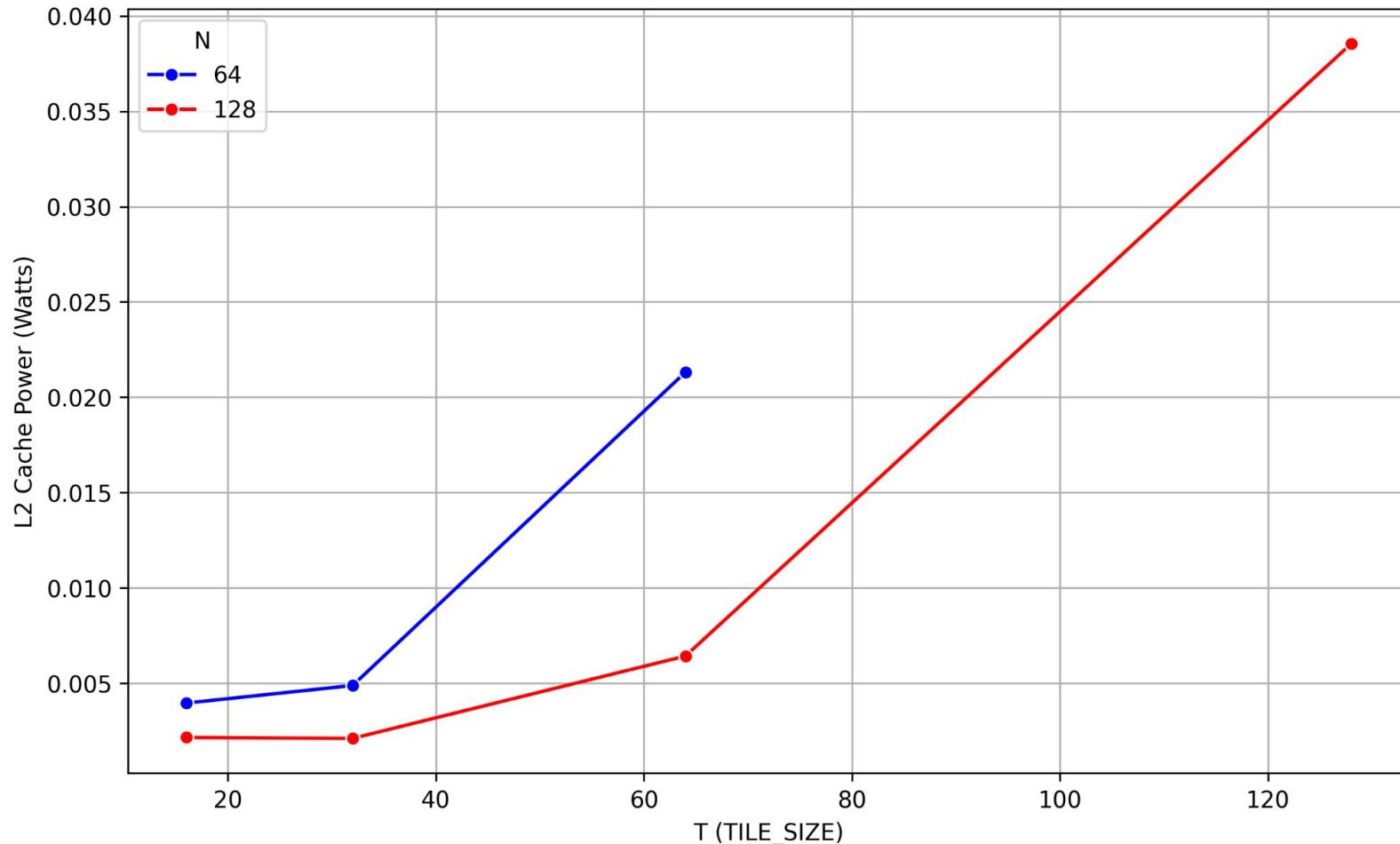
[Code and Stat files](#)

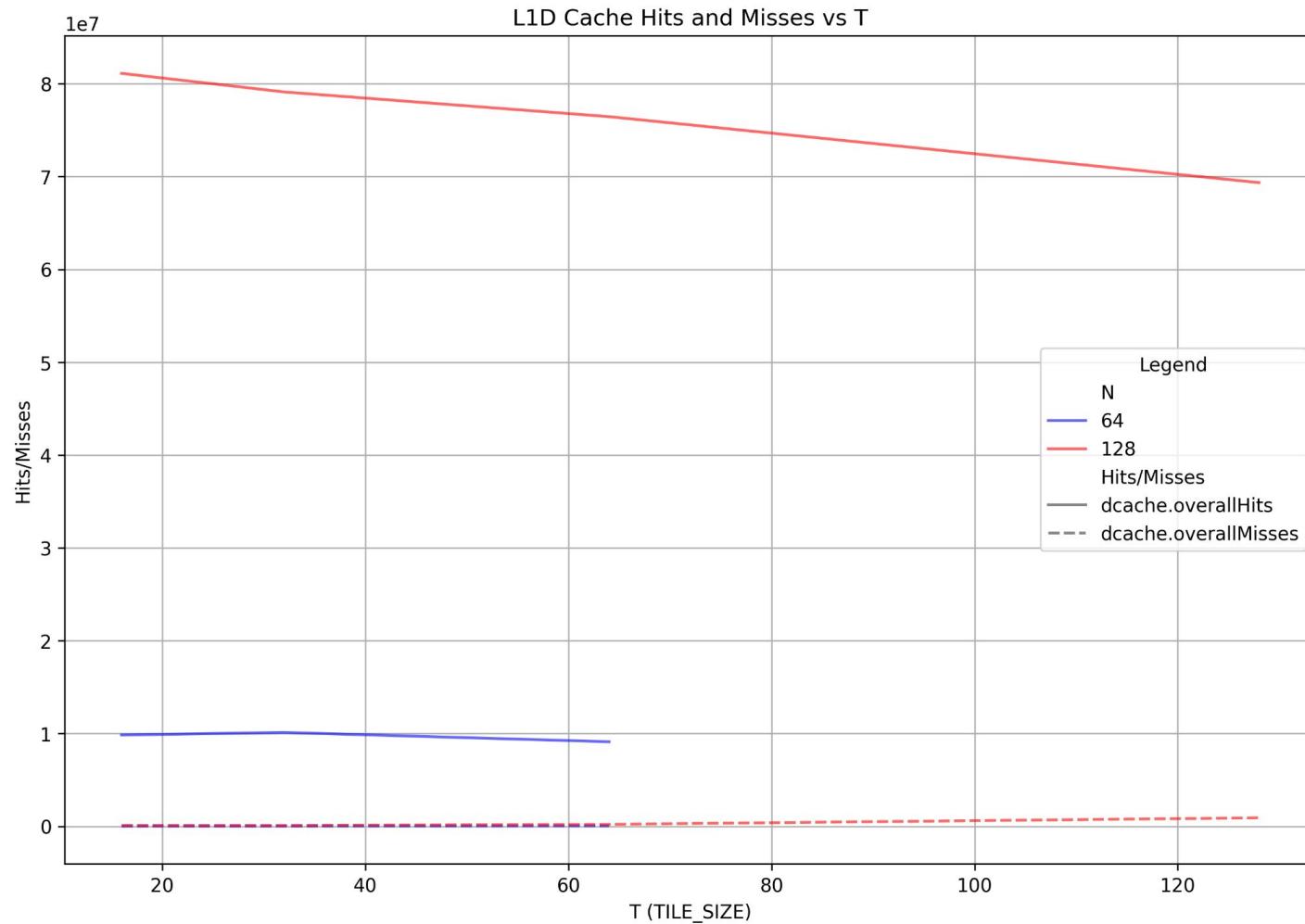
**Thank You!**

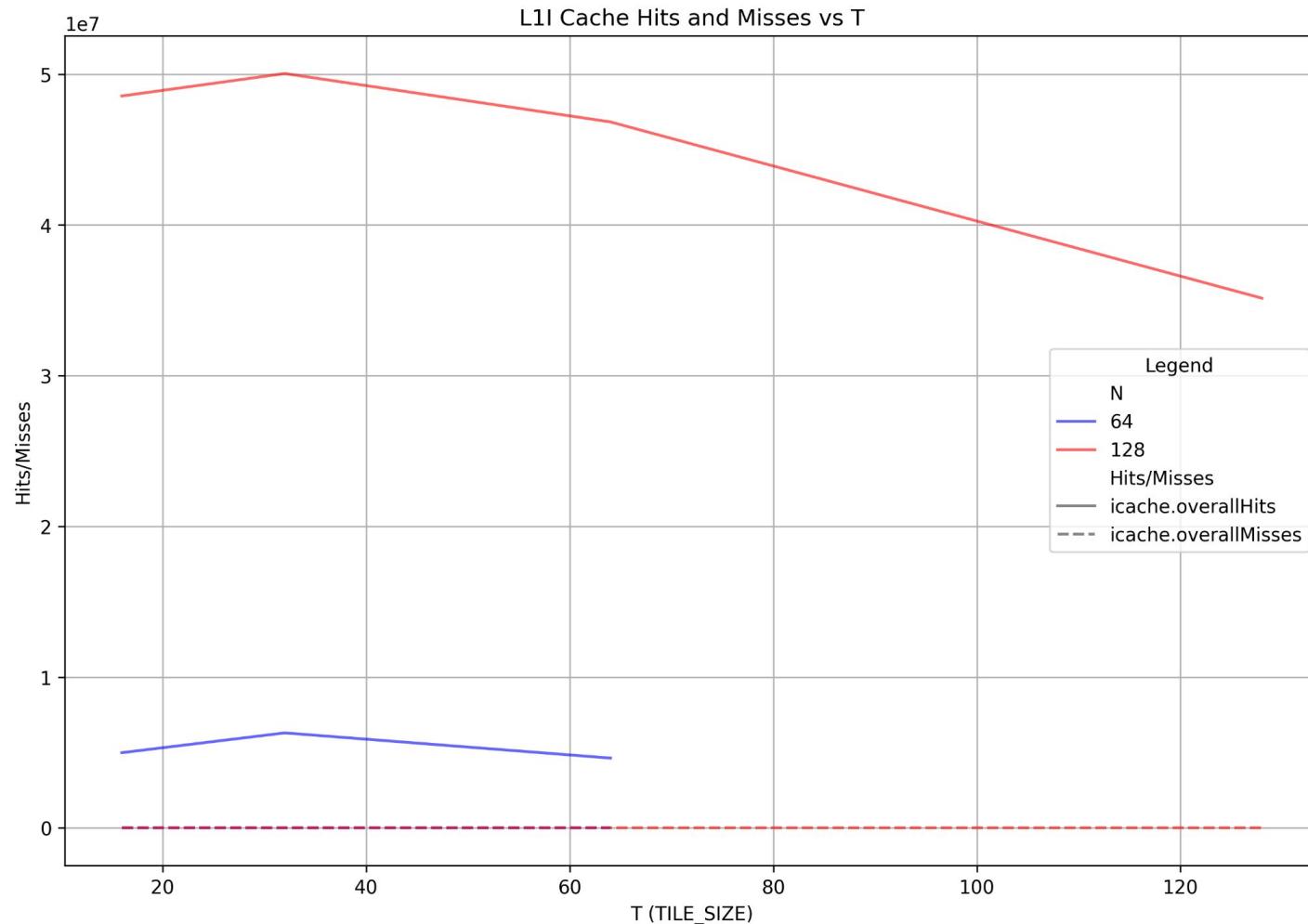
### Matmul/AES L1I Cache Power



### L2 Cache Power vs T



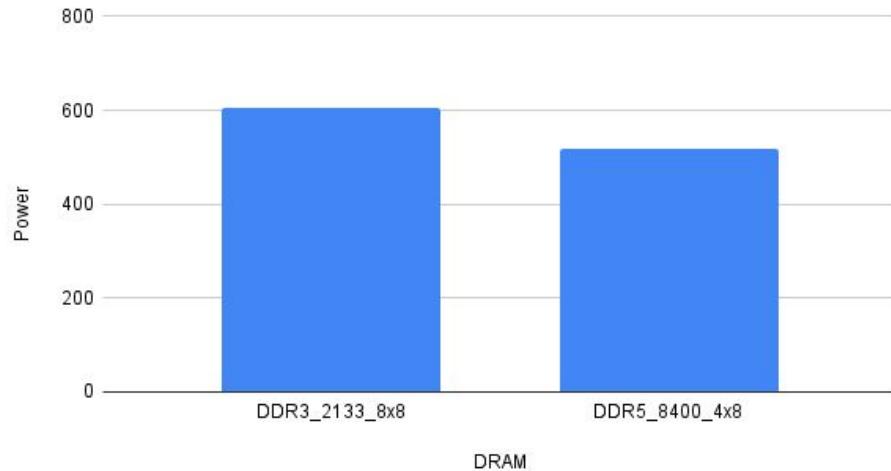




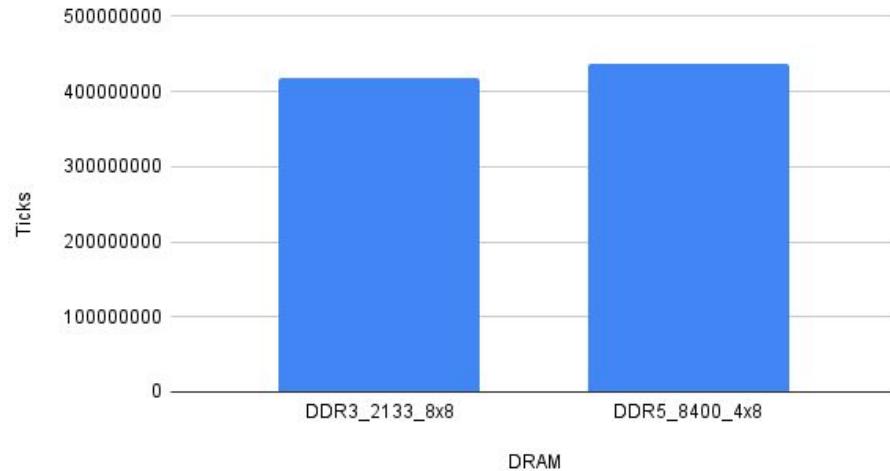
—

# Changing DRAM types?

Power vs. DRAM

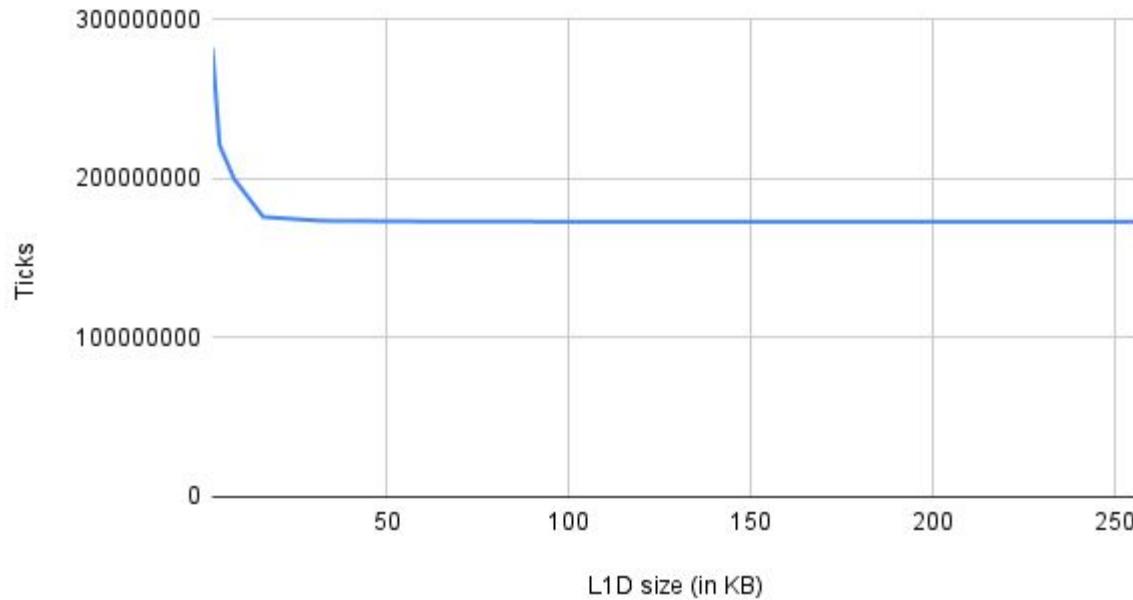


Ticks vs. DRAM



Varying **L1D** size?

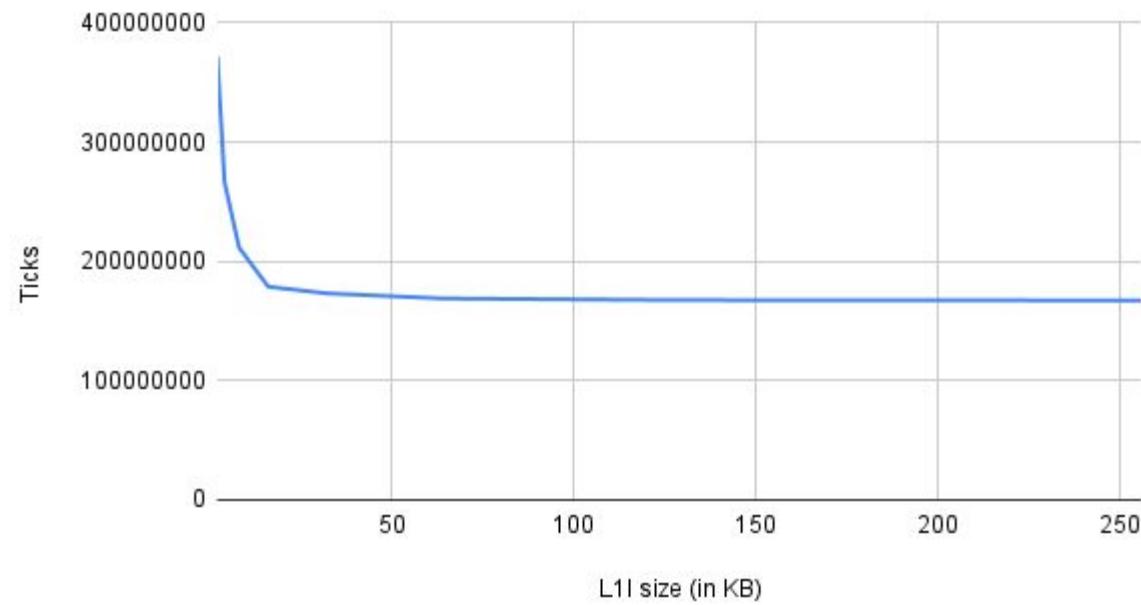
## Varying L1D size



—

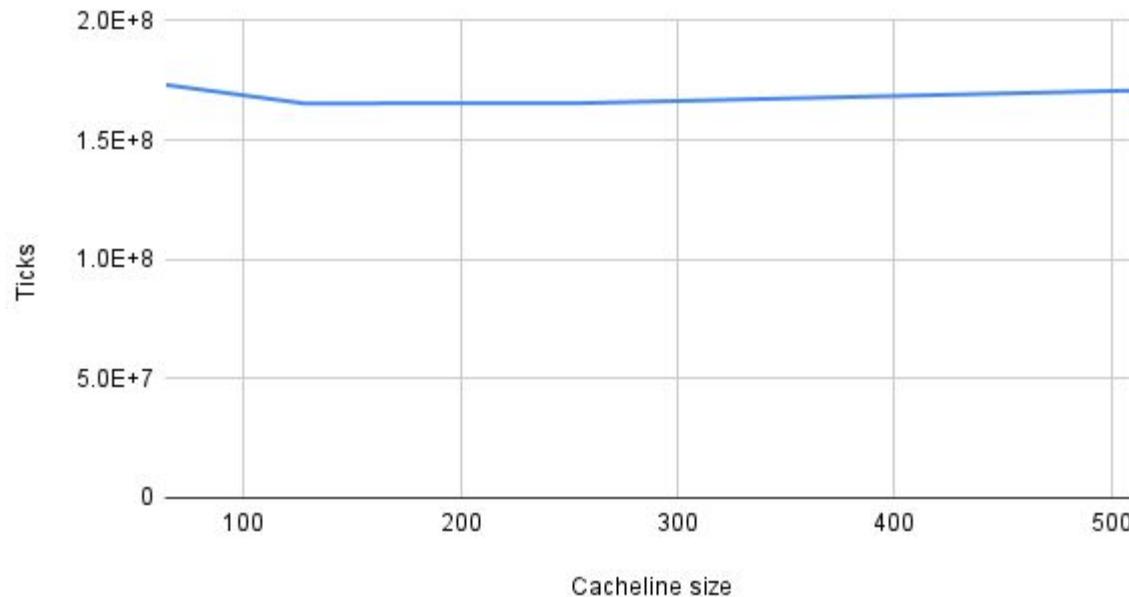
# Changing **L1** size?

## Varying L1I size



Varying **Cacheline** size?

## Varying Cacheline Size



Relationship b/w

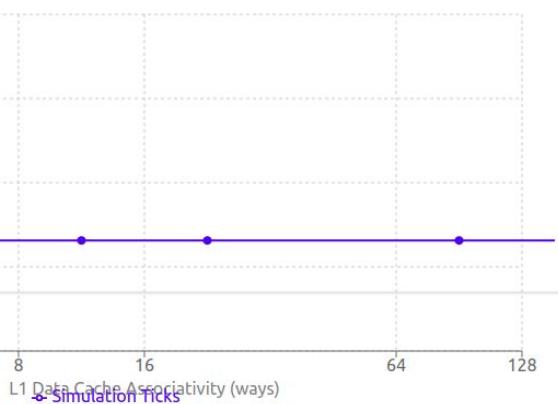
**Cache Size**

**Associativity**

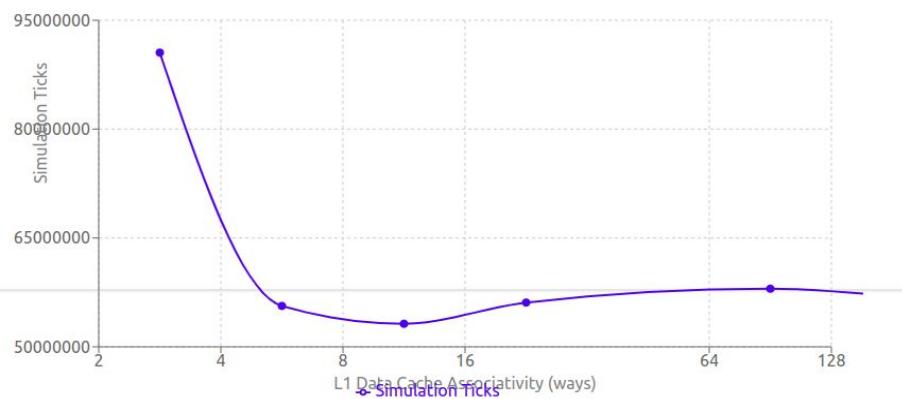
**Tick Count**

?

## AES



## MatMul



Lower L1D Cache Size ->  
Saturation occurs at higher L1D Associativity

# At Small L1D size

**Decreasing L1D Associativity → Higher conflict misses → Higher tick count**

**Increasing L1D Associativity → Fewer conflict misses → Lower tick count**

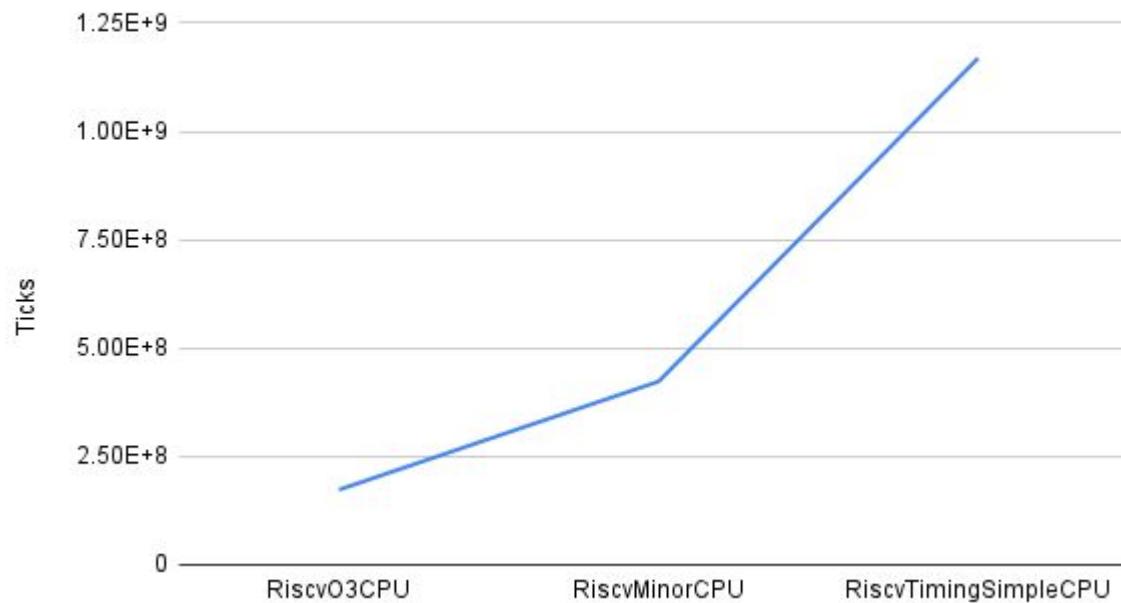
This **difference in tick count** when associativity changed was more **pronounced** for matmul as compared to AES.

—

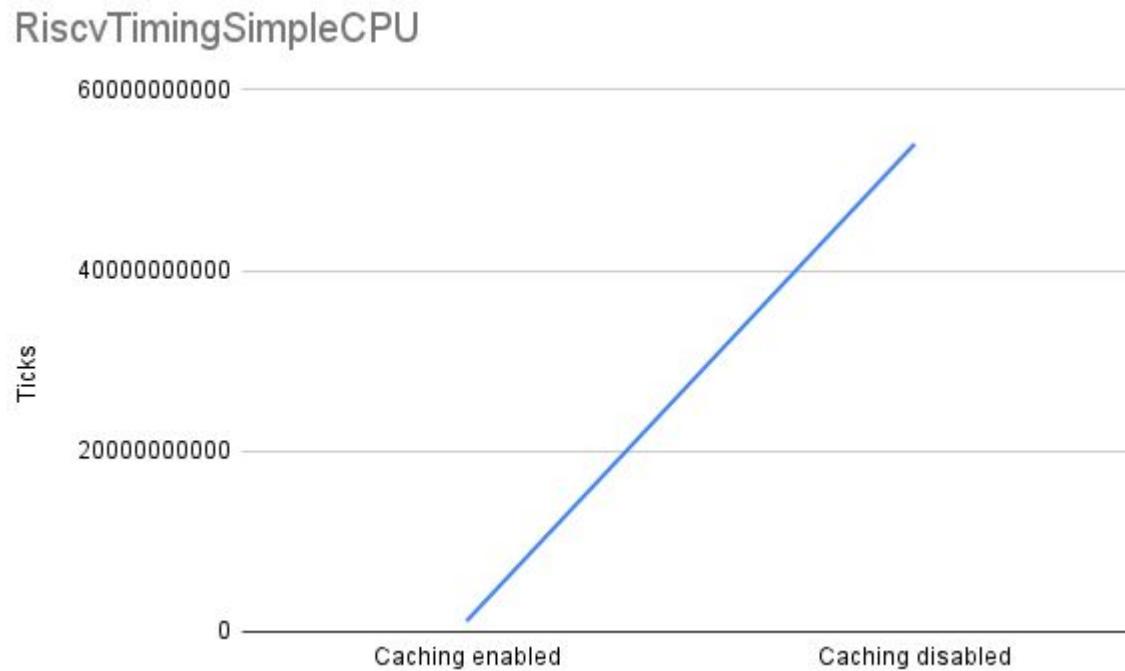
# Changing CPUs?

# The Good, The Bad, The Ugly

Ticks with Various CPUs (caching enabled)



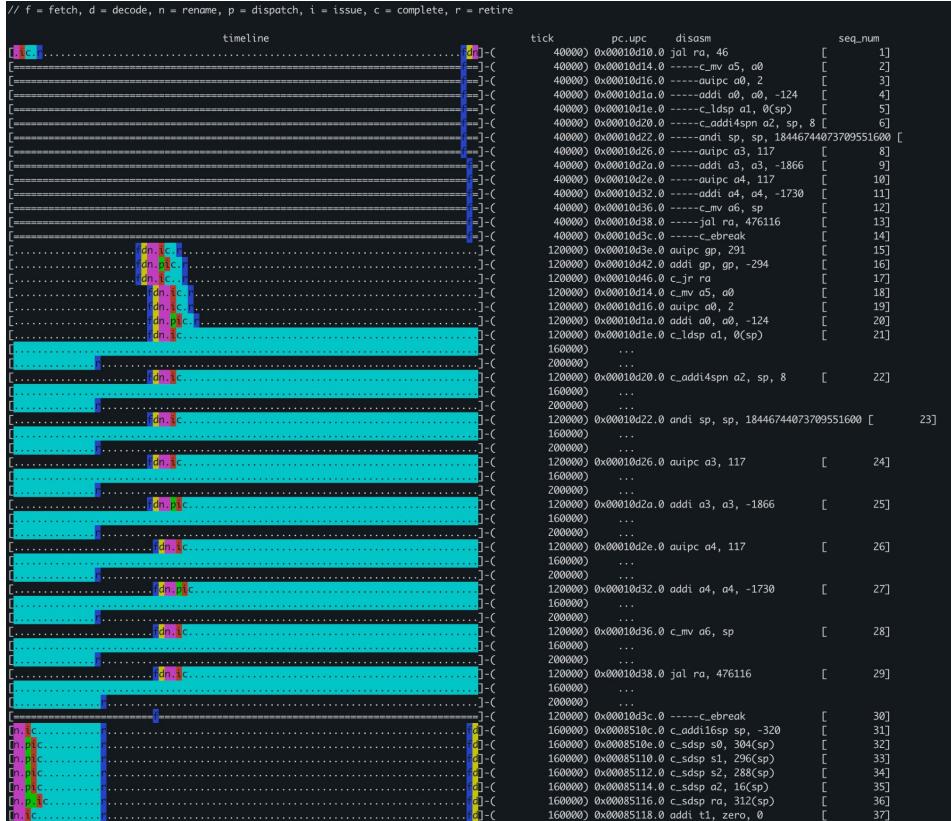
# Cache Effects



# Matmul - O3 Pipeline View



# AES - O3 Pipeline View



—

End to End system for rapid prototyping, testing, and visualizing various memory configs along with CPU params, supporting parameter sweeps and extensive on demand cross parameter visualizations.