FAST'26

# Cylon: Fast and Accurate Full-System Emulation of CXL-SSDs

Dongha Yoon*, *Hansen Idden**, Jinshu Liu, Berkay Inceisci, Sam H. Noh, Huaicheng Li

VIRGINIA TECH

* *Equal contribution*

# Convergence of Memory and Storage

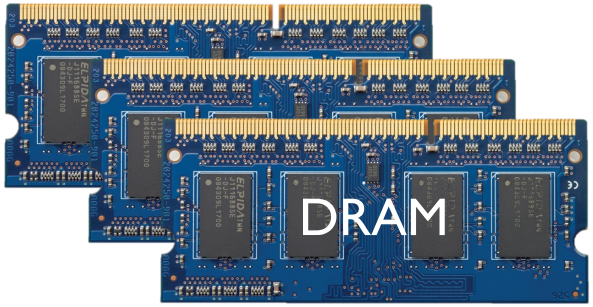


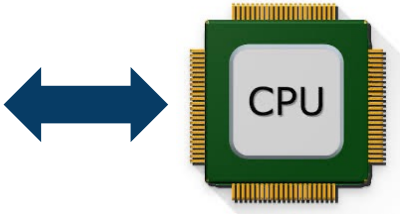
Memory-intensive applications exploding in size faster than DRAM can scale



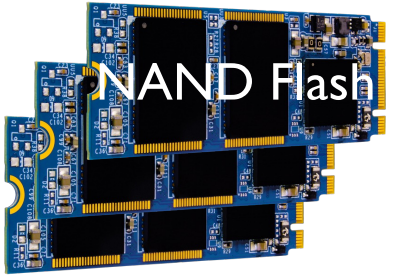



CXL-SSD




CXL




Expensive ($11-15/GB), Limited Size

Low memory access latency (~100ns)

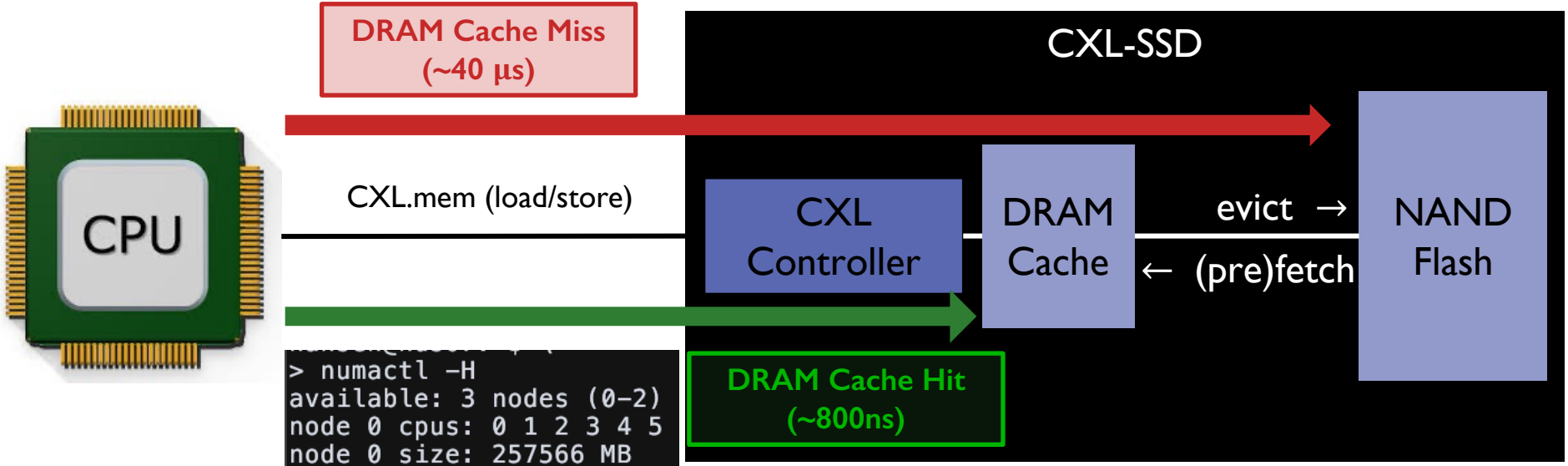
Cheap, Multi-TB capacity

Slow (tens of µs)

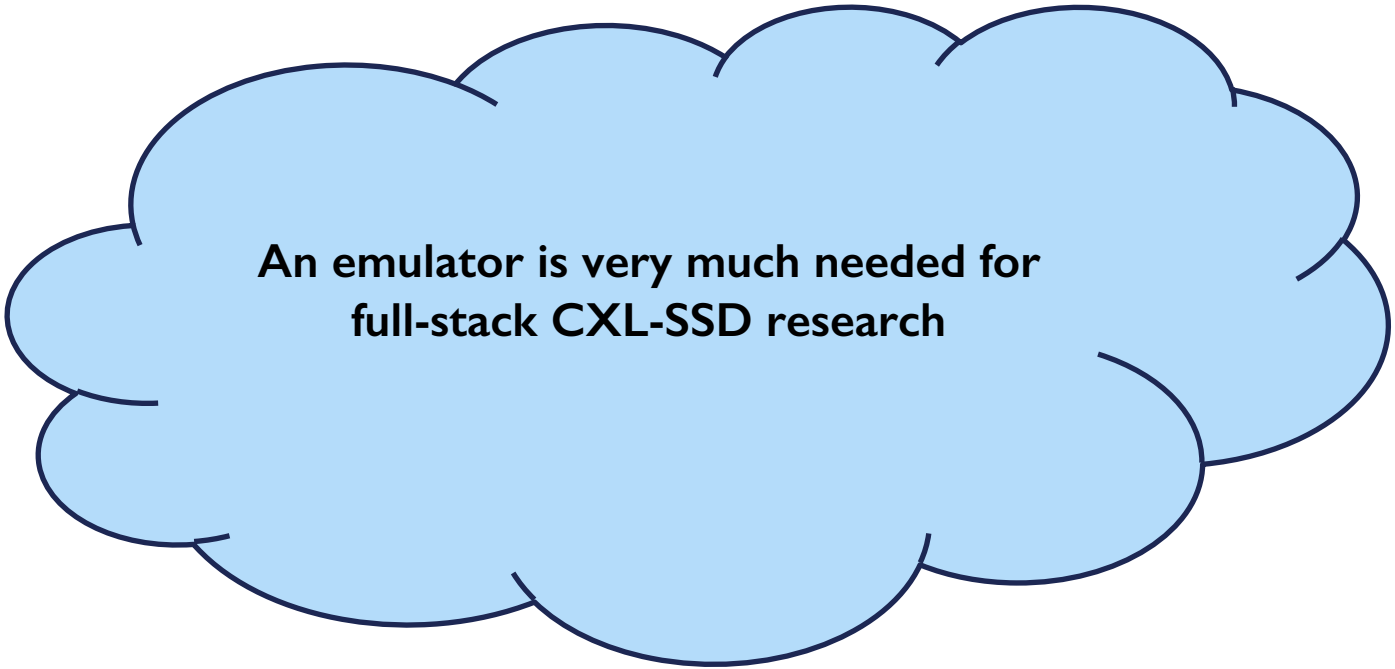
**CXL-SSDs promise DRAM-like programmability at storage-like cost**

# CXL-SSDs - The Best of Both Worlds






**50x gap! This makes cache management absolutely critical**


An emulator is very much needed for
full-stack CXL-SSD research

# CXL-SSD Emulator Requirements



**Full-Stack Execution**

Unmodified OS & apps

**Near Bare-Metal Speed**

Sub-µs hits, no VM-exit

**Flexible Device Modeling**

Configurable NAND timing and Caching policy

**There is a lack of research platform that delivers all three requirements**

Simulators (MQSim-CXL)
**No Full-stack**

Gem5, QEMU-CXL
**Too slow**

Prototypes (CMM-H)
Opaque, **scarce**

**We need an open, fast, and accurate emulator to enable CXL-SSD research**

# State of the Art Fall Short

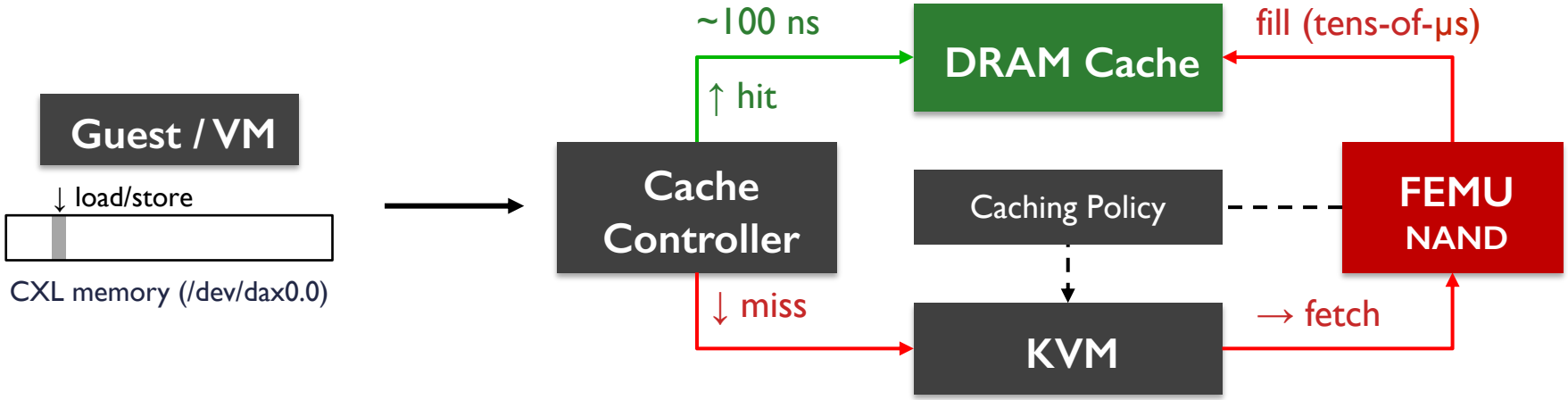


| | Validated | Fast | Full-stack | Transparent | Extensible |
|---|---|---|---|---|---|
| MQSim-CXL | ✗ | ✗ | ✗ | - | ✓ |
| ESF | ✗ | ✗ | ✗ | - | ✓ |
| CXL-SSD-Sim | ✗ | ✗ | ✗ | - | ✗ |
| CXL-DMSim | ✗ | ✗ | ✗ | - | ✓ |
| OpenCXD | - | ✗ | ✓ | ✓ | - |
| QEMU | - | ✗ | ✓ | ✓ | - |
| **Cylon** | ✓ | ✓ | ✓ | ✓ | ✓ |

**Cylon is validated, fast, full-stack, transparent, and extensible**

# Cylon Overview






Cache Hit : No VM-exit, remote DRAM speed

Cache Miss: VM-exit → FEMU → NAND

# Outline



Overview

Cylon Hypervisor Design

Extensibility

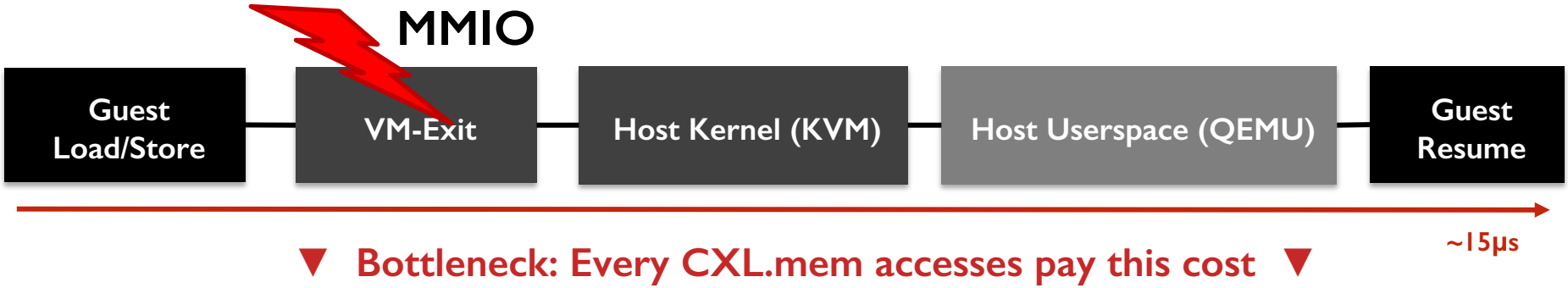
Evaluation

# Challenge #1: QEMU Problem – Everything is Slow



Cylon build on FEMU for accurate timing

## QEMU-CXL Emulation Path



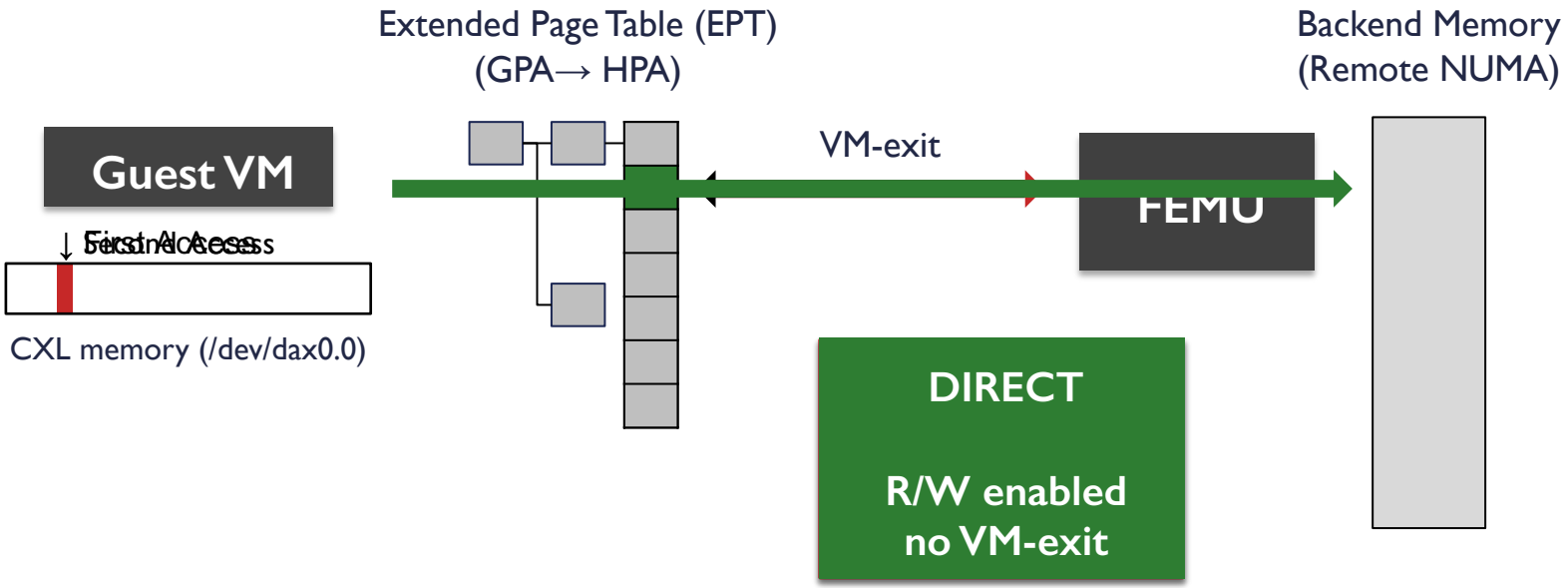

Acceptable for NAND flash emulation, but too high for CXL-attached DRAM

We need to optimize QEMU/KVM to reproduce sub-µs cache hit latency

# Cylon Dynamic EPT Remapping (DER)



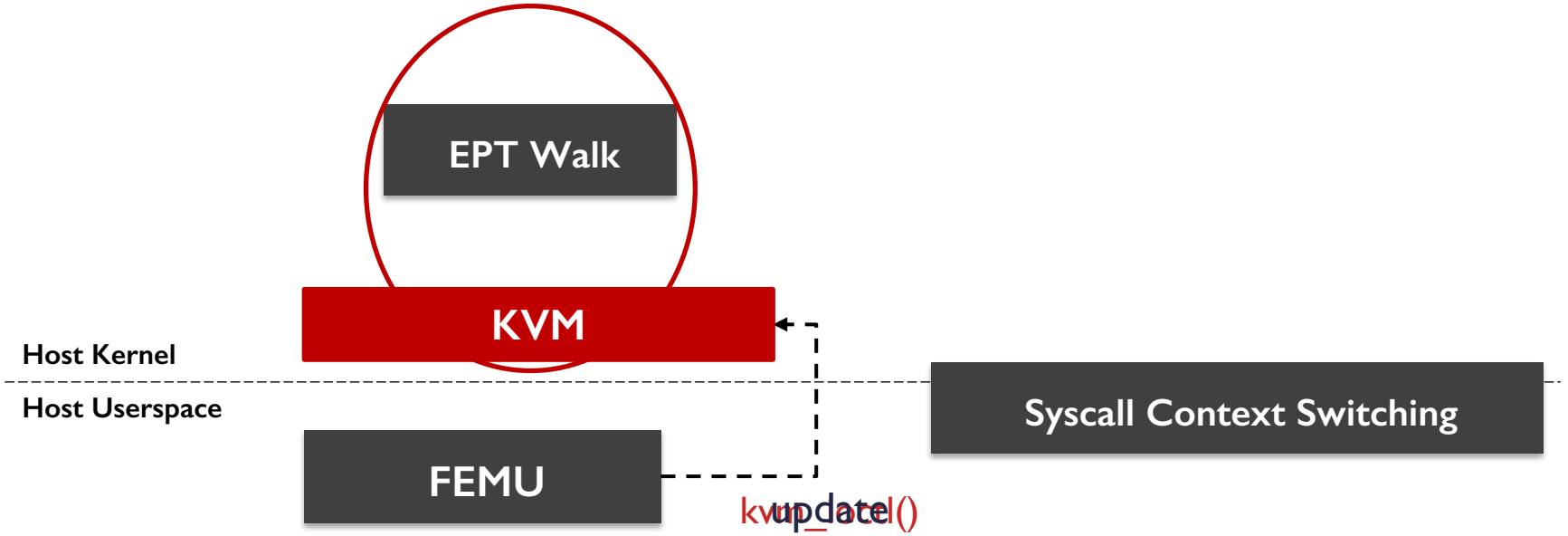
Dynamically toggling EPT entry between TRAP and DIRECT state




**An EPT entry bit-flip per page eliminates all VM-exit overhead on cache hits**

# Challenge #2: The ioctl() Bottleneck



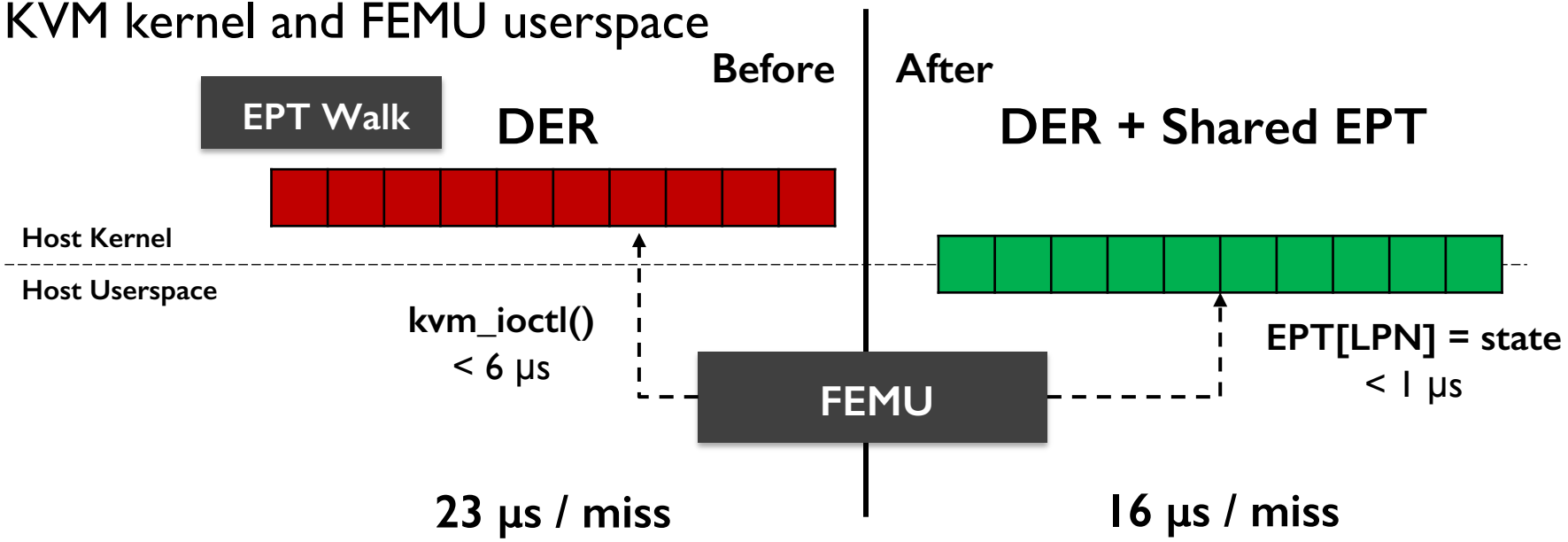



ioctl() calls for every EPT updates make Cylon less accurate and unpredictable

# Cylon Shared EPT Memory



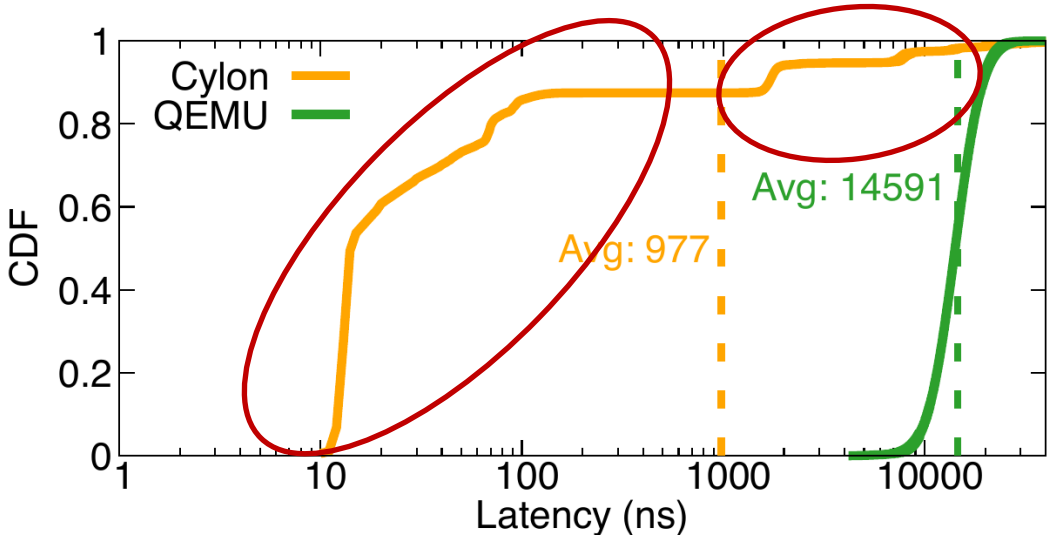
Pre-allocates all EPT entries in a single contiguous region mapped into both KVM kernel and FEMU userspace




**29% lower VM-exit overhead, no syscall, O(1) EPT walk**

# Realistic Latency Distribution






**8GB working set > 4.8GB cache**
(exercises both hits and misses)

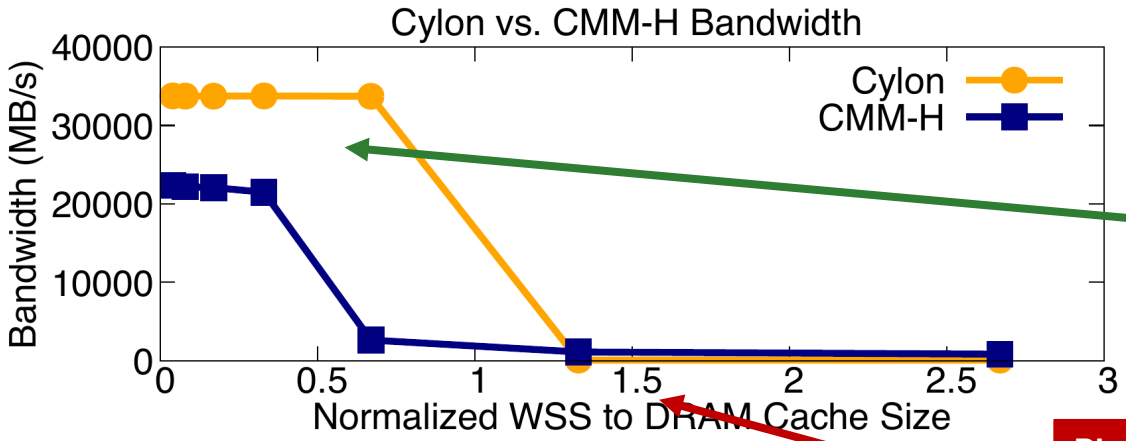
**QEMU (green):**
- Single inflated mode at **14.6µs**

**Cylon (orange):**
- Clear bimodal distribution

**First emulator to reproduce realistic CXL-SSD latency distributions**

# Validated Against CMM-H in Bandwidth






**Intel MLC bandwidth test**

WSS (working set size) normalized to DRAM cache size

**Phase 1: WSS ≤ cache size**

- Near-memory bandwidth
- Cylon sustains **32 GB/s**

**Phase 2: WSS > cache size**

- Sharp drop to NAND-bound
- Both converge to same throughput

CMM-H drops earlier due to opaque controller overhead (FPGA artifacts)

**Cylon reproduces the fundamental DRAM-to-NAND transition**

# Outline



Overview

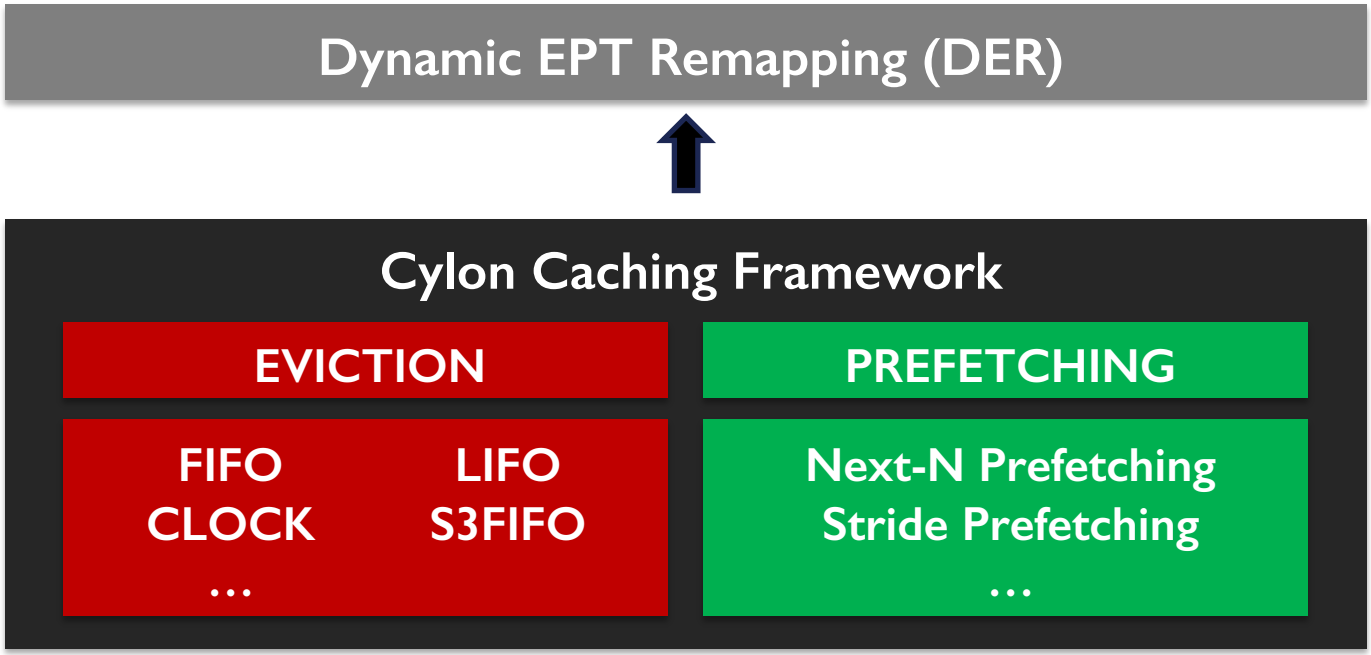
Cylon Hypervisor Design

Extensibility

Evaluation

# Cylon Extensible: Caching Framework

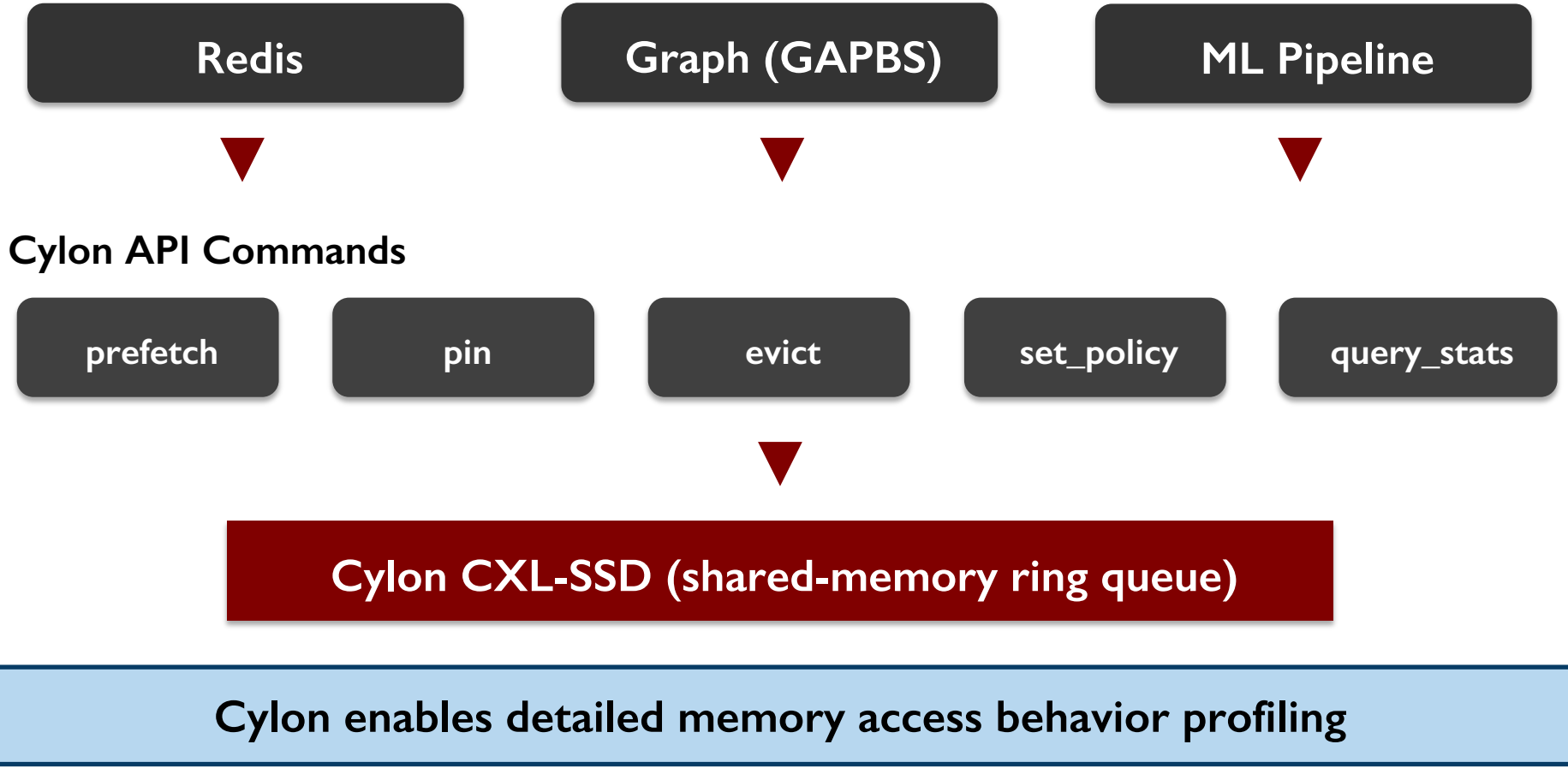





Cylon enables pluggable eviction and prefetching policies to explore CXL-SSD cache management design space

# Cylon Application-Level Interface






Cylon enables detailed memory access behavior profiling

# Outline



Overview

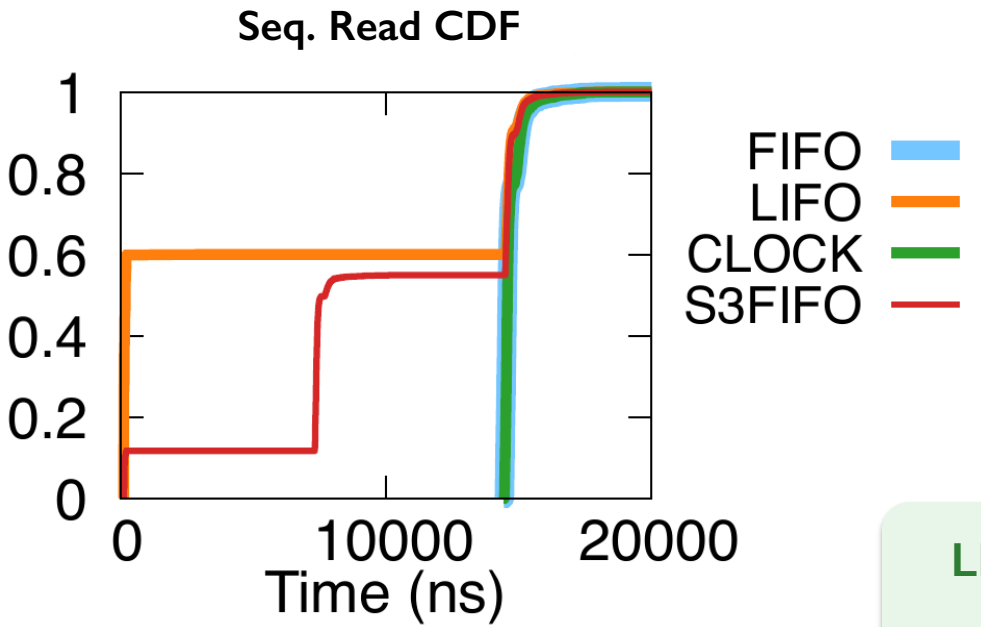
Cylon Hypervisor Design

Extensibility

Evaluation

# Flexible Policy Exploration - Eviction






## Cache Hit Rate Stride-1024

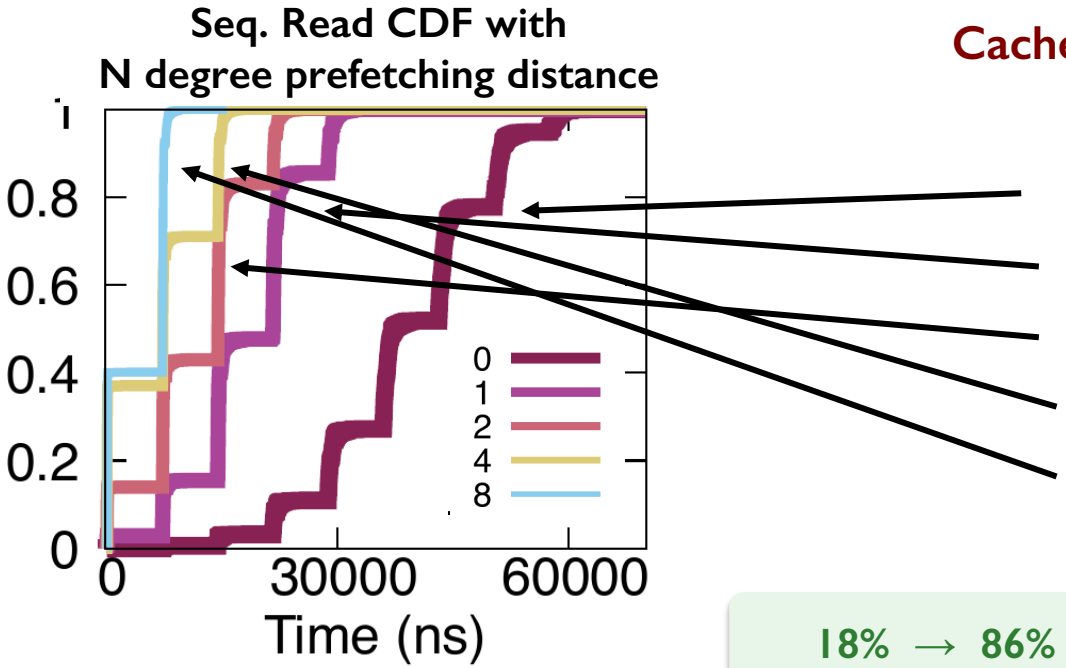
| Policy | Hit Rate |
|---|---|
| FIFO | 75% |
| LIFO | 90% |
| CLOCK | 75% |
| S3FIFO | 80% |

LIFO → 15% higher hit rate than FIFO/CLOCK

**Cylon enables rapid eviction-policy comparison**

# Flexible Policy Exploration - Prefetching






**Cache Hit Rate Stride-4096 (S3FIFO)**

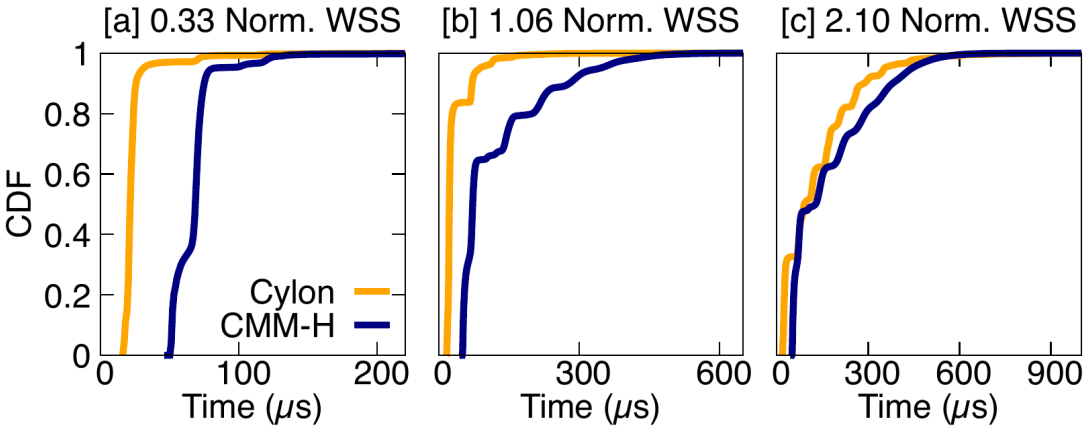
| Degree | Hit Rate |
|---|---|
| N=0 | 18% |
| N=1 | 35% |
| N=2 | 56% |
| N=4 | 77% |
| N=8 | 86% |

18% → 86% just by tuning prefetch degree

**Cylon shows increasing prefetch degree shifts accesses from NAND to DRAM**

# Real Application - Redis






**The gap between Cylon and CMM-H is due to FPGA overhead in CMM-H**

**Redis YCSB-C (100% reads)**

**(a) Small WSS (0.33× cache):**
*All in DRAM, fast hits*
*Cylon: hundreds of ns*

**(b) Medium WSS (1.06× cache):**
*Mix of hits + misses*
*Gap narrows*

**(c) Large WSS (2.10× cache):**
*Mostly NAND, both converge to tens–hundreds of μs*

**Cylon can replicates cache-dominated to NAND-dominated performance**

## Cylon Performance Optimizations

**Dynamic EPT Remapping (DER)**

Eliminates VM-exits on cache hits by toggling EPTE flags

**Shared EPT Memory**

O(1) constant-time cache residency updates without syscalls or page-table walks

## Cylon Flexibility Features

**Configurable Caching Framework**

Pluggable eviction and prefetching policies

**Application-Level Cache Control**

Apps issue prefetch / pin / evict commands for HW-SW co-design

# Cylon: https://github.com/MoatLab/FEMU