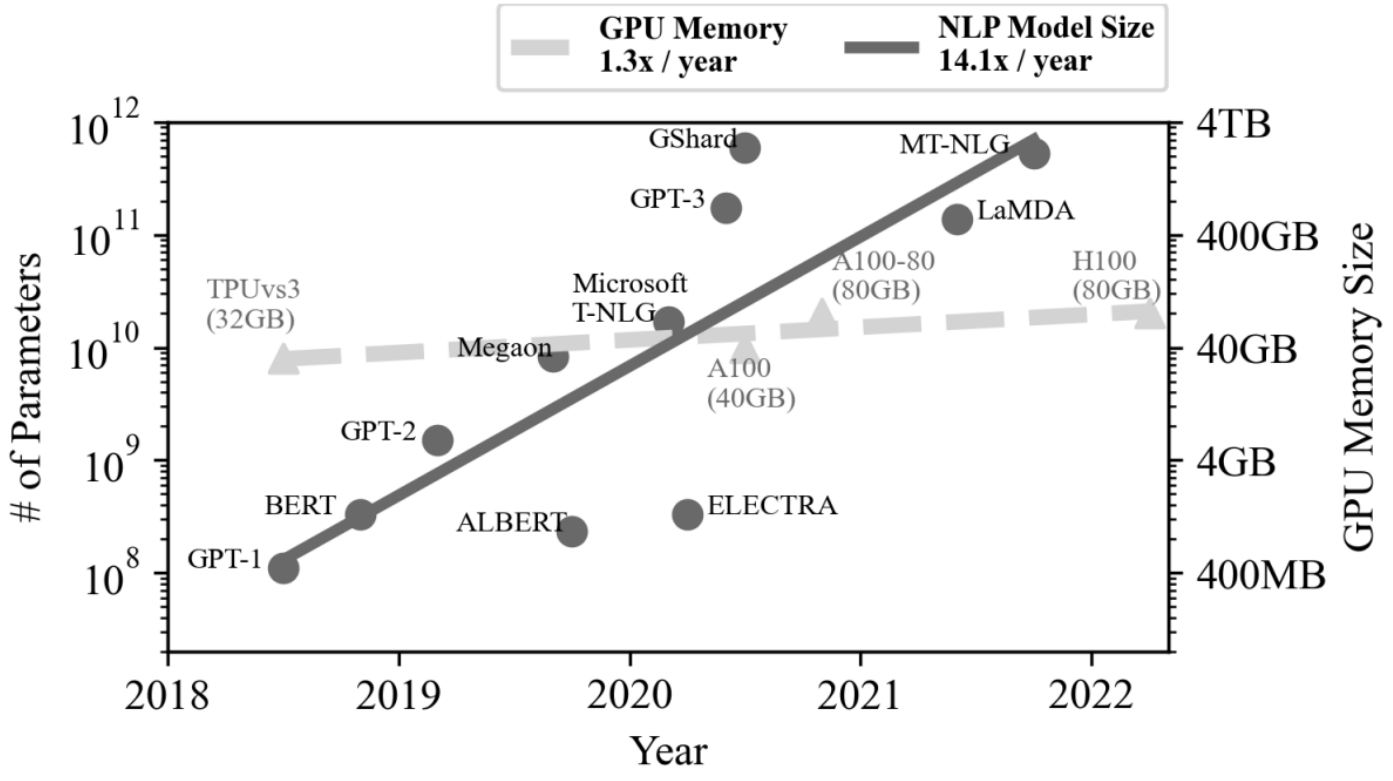
Yang, Shao-Peng, et al.

Overcoming the Memory Wall with CXL-Enabled SSDs

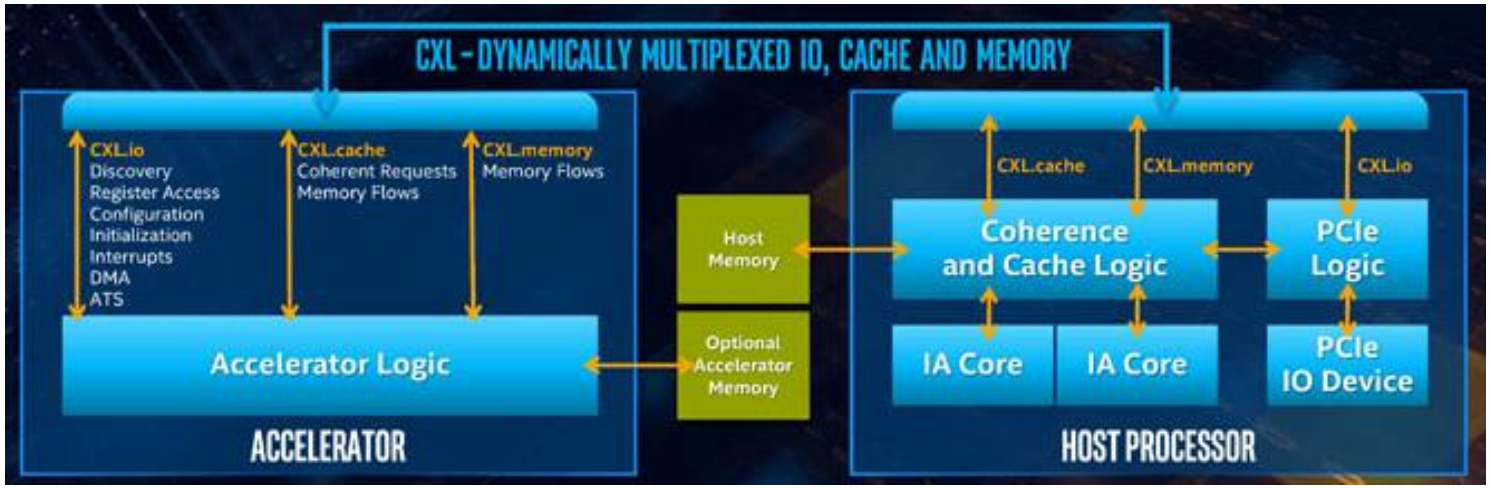
(ATC 23')

Presented by Minguk Choi

B1. Memory Wall

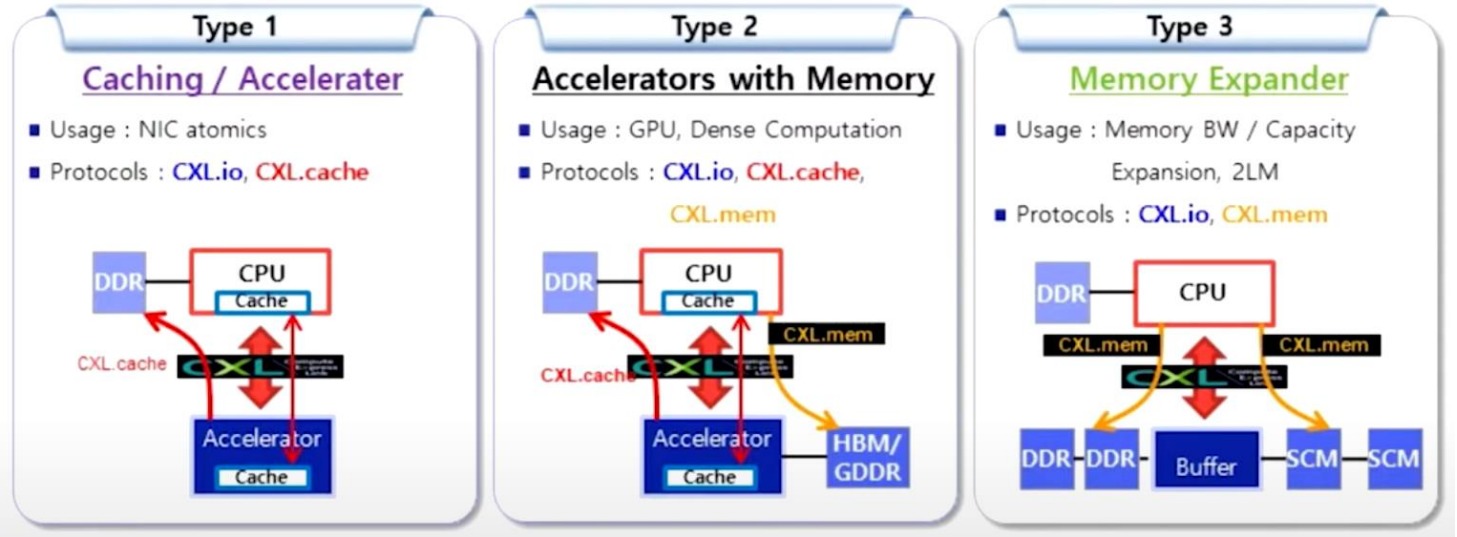


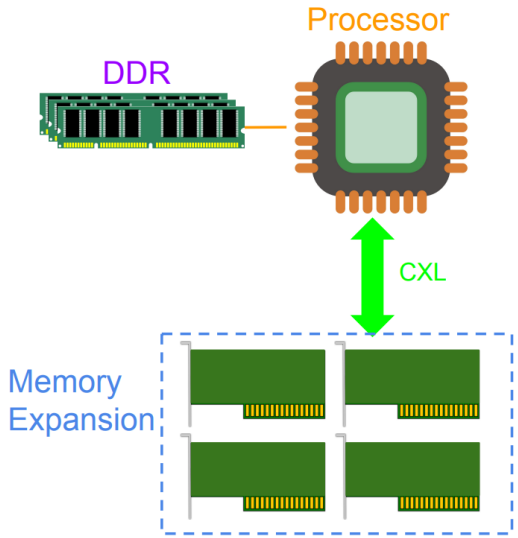
B2. Compute Express Link (CXL)



- Compute Express Link™ (CXL™)
  - Based on PCIe Physical layer, Add CXL "Memory Access“ and “Cache Coherence“
  - Effective access mechanism of "shared memory pool“ of Heterogeneous computing
  - Extended data flow and effective resource sharing between Accelerator and CXL devices
  - Reduce access latency of distributed memory

B3. CXL-flash

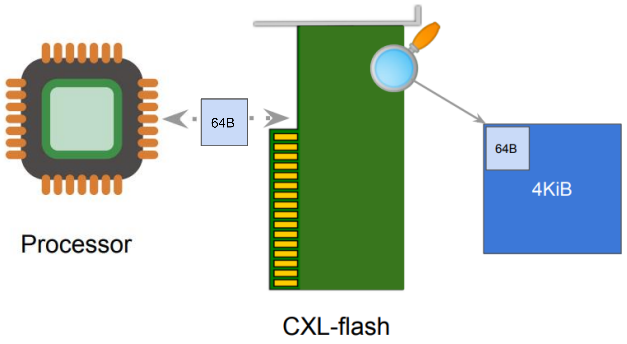




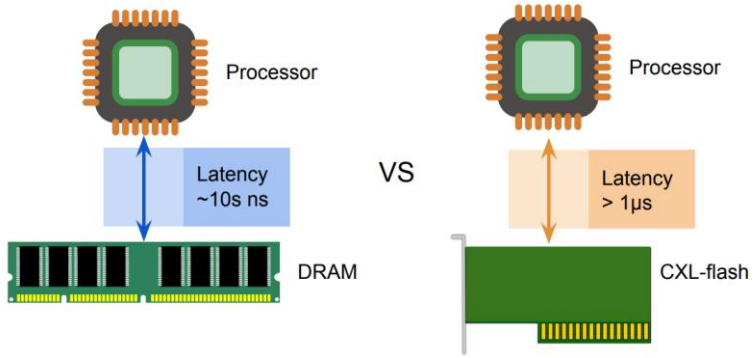
- CXL Type 3
  - allows the host CPU to directly manipulate the device memory via **load/store** instructions
  - currently only considers DRAM and PMEM
- Why CXL Type 3 with flash device?
  - the high capacity and better scaling of flash-based SSDs
  - enabled by stacking in 3D and storing multiple bits in a cell

B4. Challenges of CXL-flash

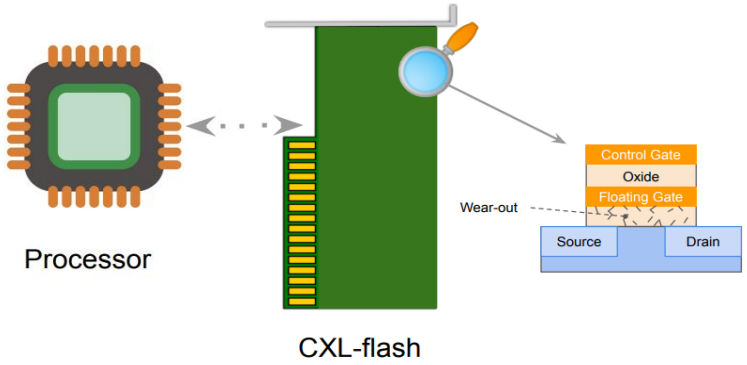
Challenge #1. Granularity mismatch



Challenge #2. Microsecond latency



Challenge #3. Limited endurance

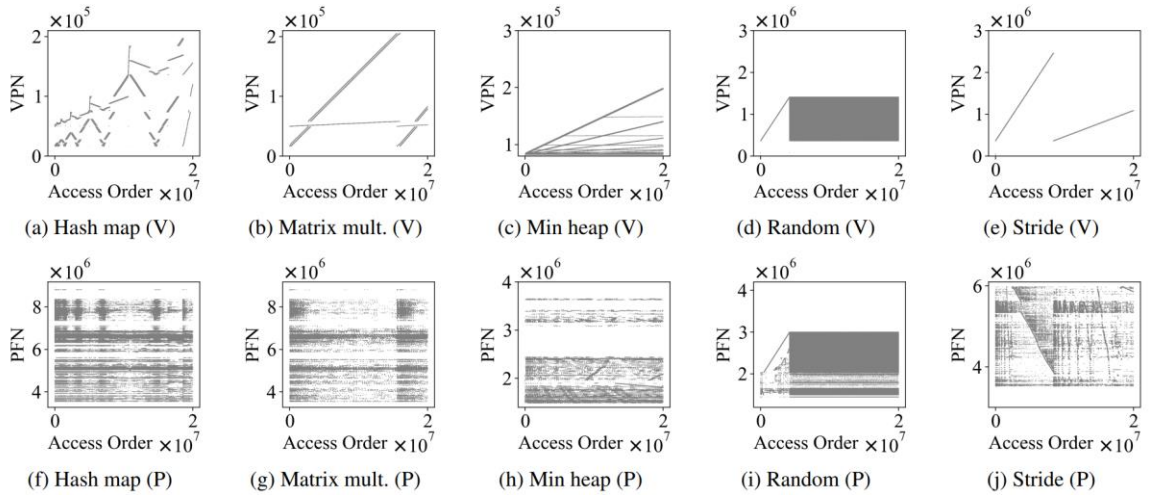


B5. Virtual vs Physical memory trace

- Memory access trace with CXL-flash simulator
  - 5 synthetic workload
  - VPN : Virtual Page Number
  - PFN : Physical Frame Number
- Mismatch between virtual and physical
  - Differences in access pattern and performance
  - Should follow physical address trace

Table 2: Synthetic workload characteristics.

Workload	Inter-arrival time (ns)	Read-write ratio	Footprint (GiB)
Hash map	329	53:47	<1
Matrix multiply	38	55:45	<1
Min heap	72	50:50	1
Random	76	50:50	4
Stride	146	50:50	8



Workload	% of sub-μs latency (virtual)	% of sub-μs latency (physical)					Error (%)				
		1	2	3	4	5	1	2	3	4	5
Hash map	96.9%	86.7%	88.3%	74.5%	63.8%	63.9%	10.2%	8.6%	22.4%	33.1%	33.0%
Matrix mult.	98.2%	72.7%	57.4%	59.2%	48.1%	47.9%	25.5%	40.8%	39.0%	50.1%	50.3%
Min heap	97.8%	92.1%	96.0%	75.6%	69.1%	69.4%	5.7%	1.8%	22.2%	28.7%	28.4%
Random	32.2%	26.4%	27.1%	28.0%	22.4%	21.8%	5.8%	5.1%	4.2%	9.8%	10.6%
Stride	64.7%	64.3%	59.4%	64.5%	51.9%	52.0%	0.4%	5.3%	0.2%	12.6%	12.7%

Summary of Background

- Memory Wall
  - HW memory size << ML(Data) Size → CXL Interface
- CXL-flash
  - the high capacity and better scaling of flash-based SSDs
- Challenges of CXL-flash

  1) Granularity mismatch, 2) Micro-second Latency, 3) Limited endurance
- Virtual vs Physical memory trace
  - Different access pattern → should follow physical access pattern.

Design of CXL-flash

- Challenges of CXL-flash
  1) Granularity mismatch
  2) Micro-second Latency
  3) Limited endurance

- Design objectives
  - How effective is caching in improving performance?
  - How can we effectively reduce flash memory traffic?
  - How effective is prefetching in hiding the long flash memory latency?
  - What are the appropriate flash memory technology and parallelism for CXL-flash?

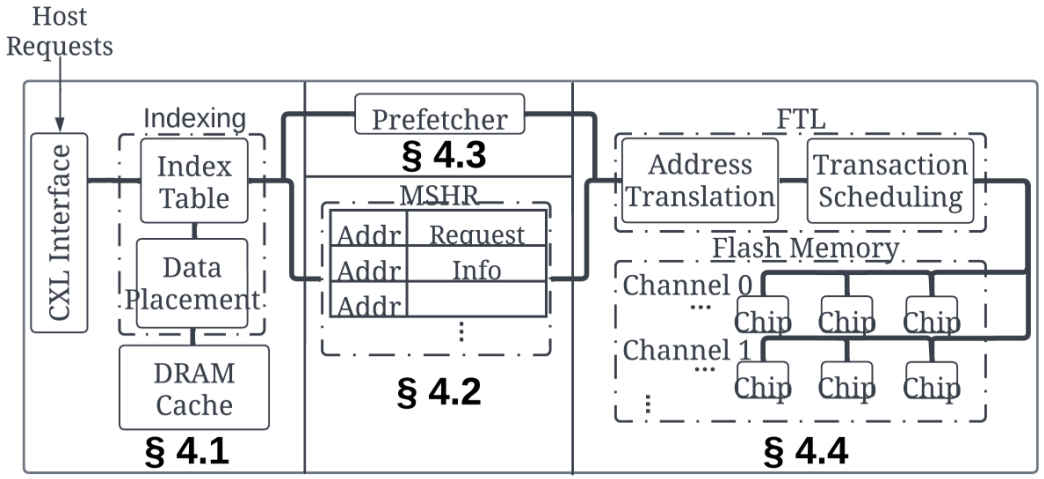
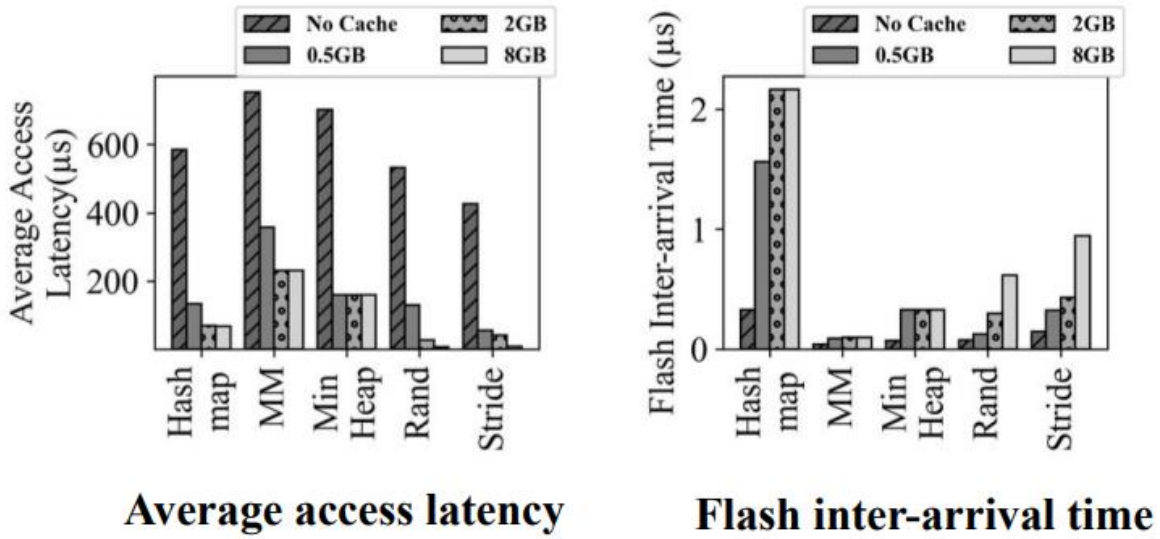
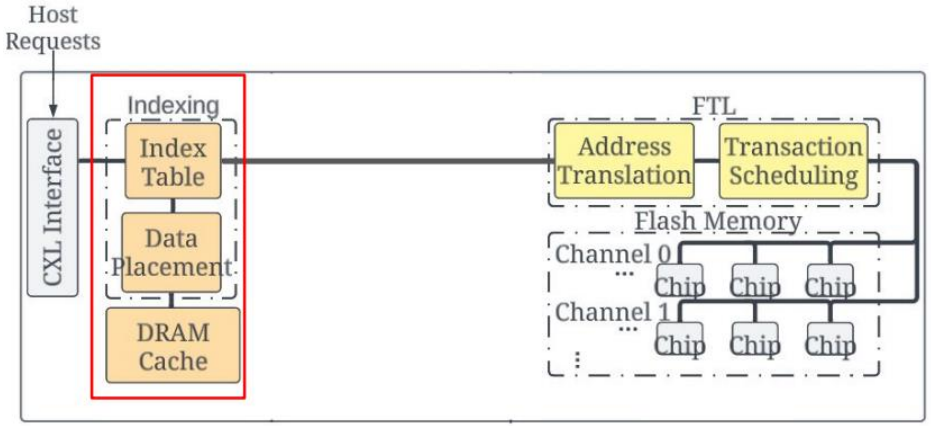
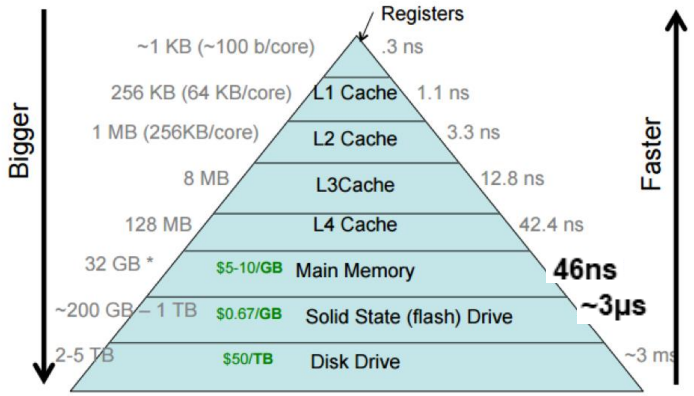


Figure 4: Architecture of the CXL-flash

D1. Cache

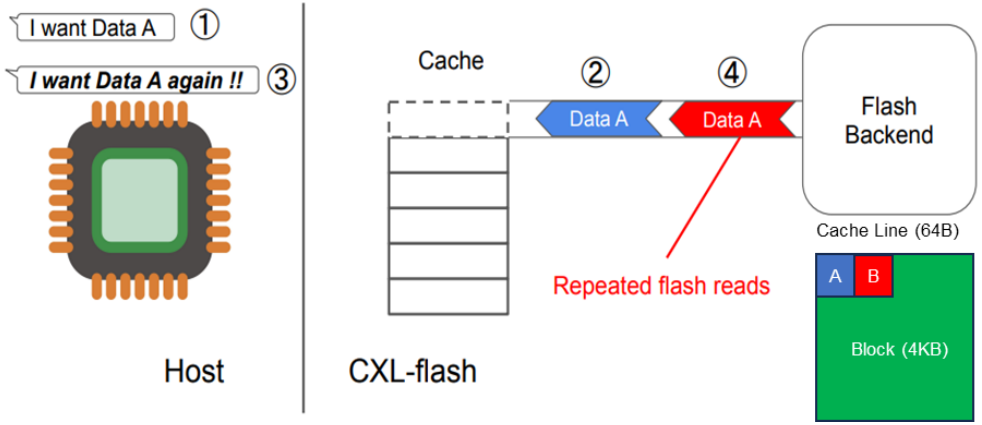




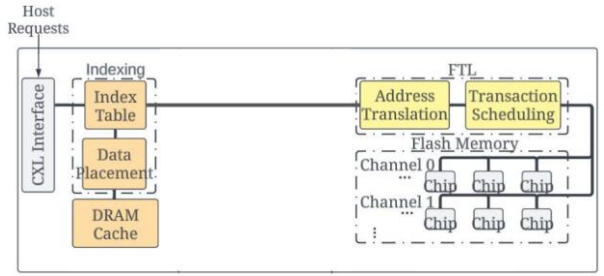


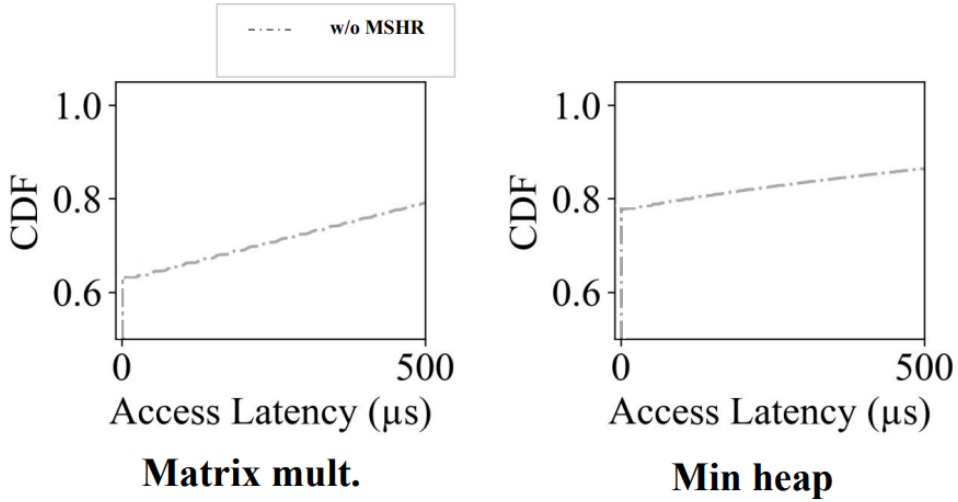
- Without cache
  - high latency and short inter-arrival time (high traffic)
- With cache
  - Low latency and high inter-arrival time (low traffic)

D2. MSHR



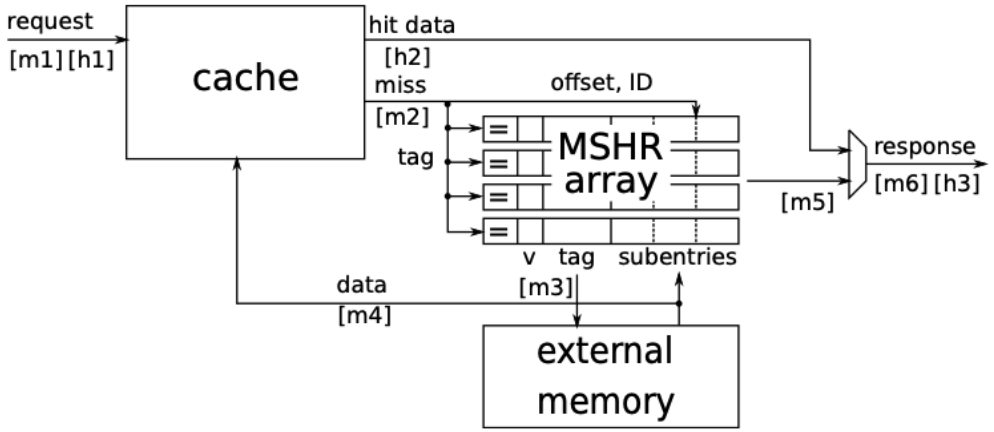
- Repeated flash reads on same block
- Non-blocking Cache
  - do not stalls the pipeline on a cache miss
- Memory-level parallelism
  - A miss-under-miss cache coupled with a parallel lower-level memory system

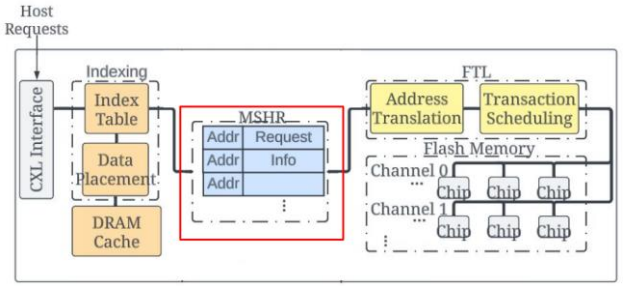


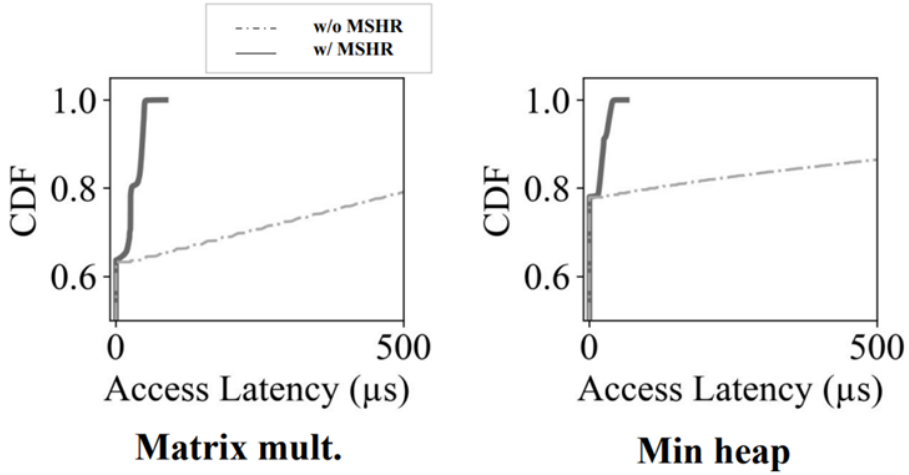


- Even with a large cache size (8GB),
  - the average access latency is still high with cache

D2. MSHR



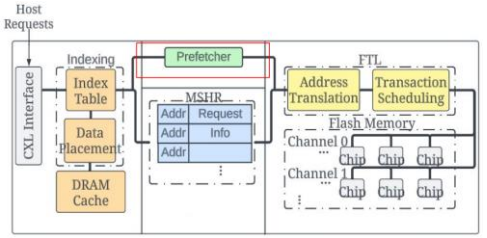


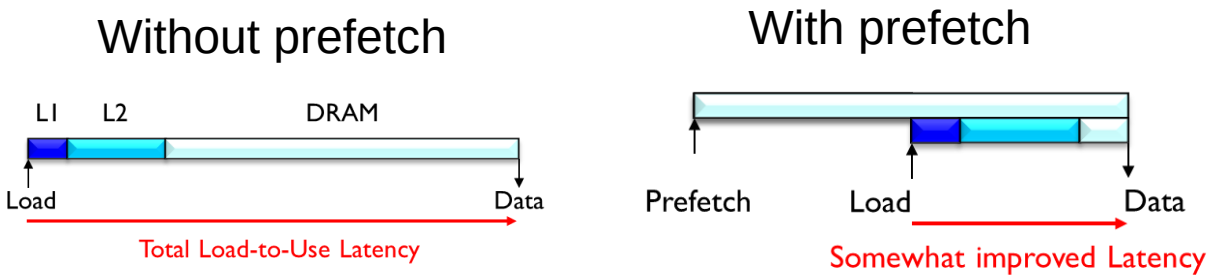


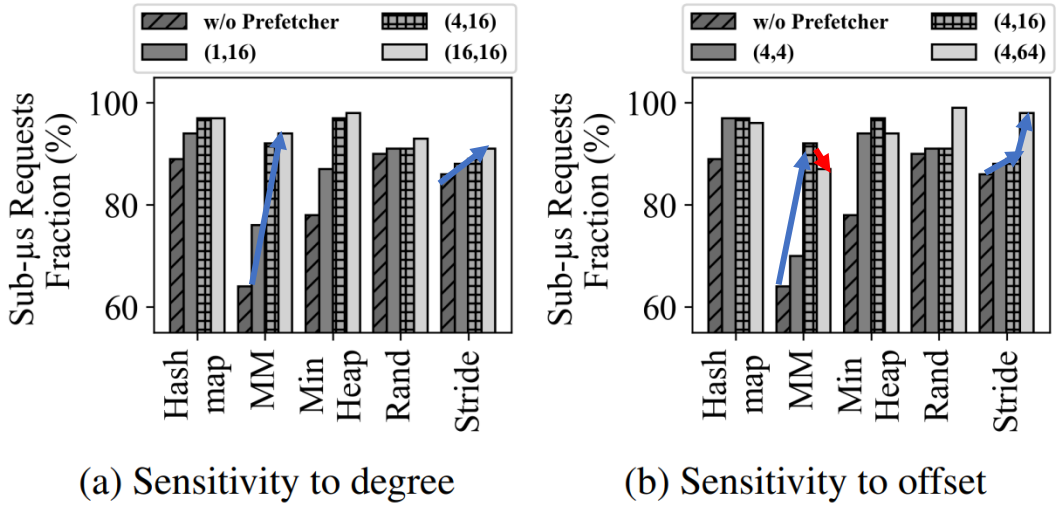
- MSHRs: miss status holding registers
  - If cache miss, the MSHRs are looked up determine if the cache block is already being fetched
  - If MSHR hit, then a cache miss is merged with the primary miss
  - If MSHR miss, read data from flash

- MSHR prevents repeated flash reads
  - Leads better access (tail) latency

D3. Prefetcher







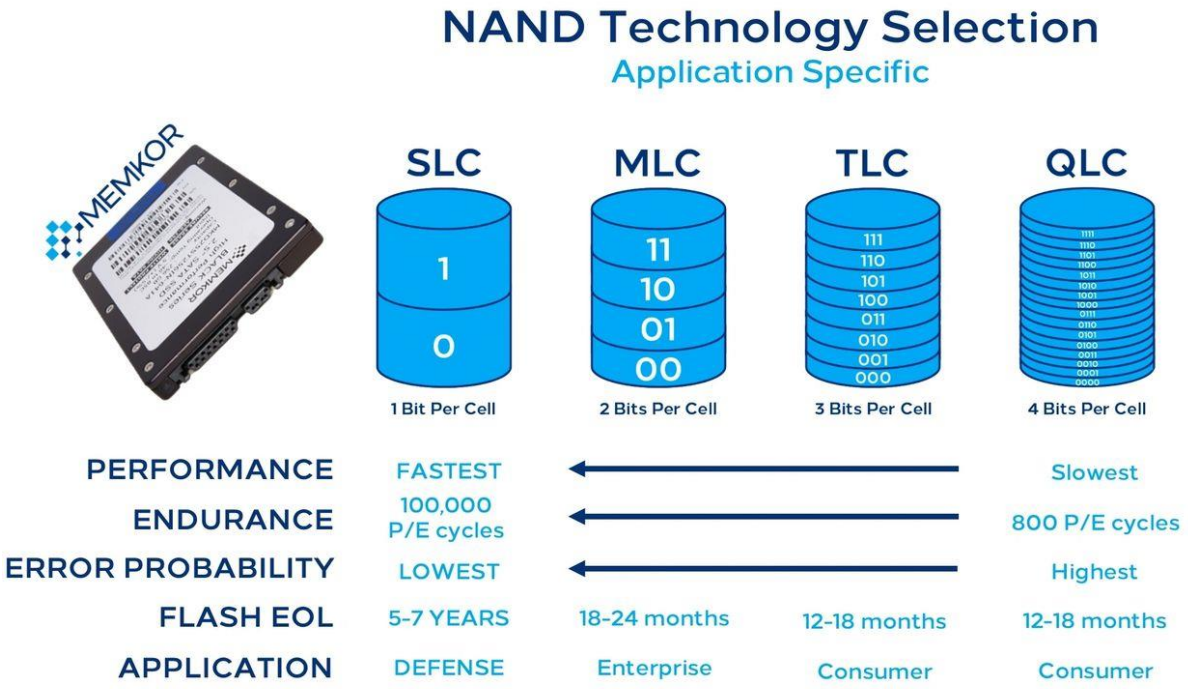
(a) Sensitivity to degree

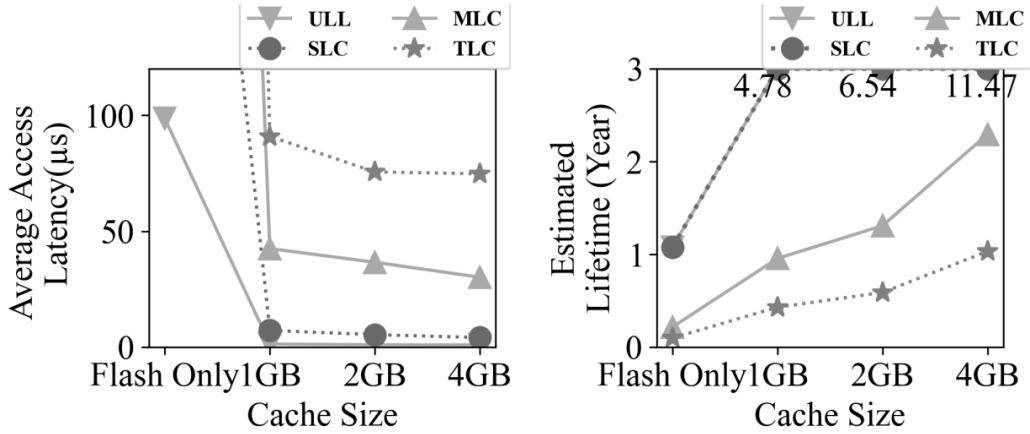
(b) Sensitivity to offset

- Prefetch
  - Hide the long latency of a load
- Next-N-line prefetcher
  - Degree(N): the aggressiveness of the prefetcher
  - Offset(O): how far ahead the prefetcher is fetching.
  - If request X, prefetch X+O+1, X+O+2, ..., X+O+N

- Higher prefetch degree, better latency
- Higher prefetch offset, better(or worse) latency

D4. Flash technology

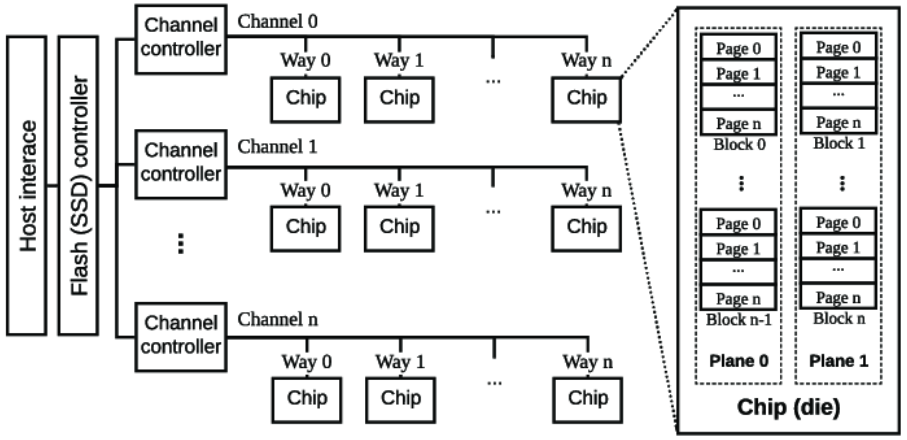


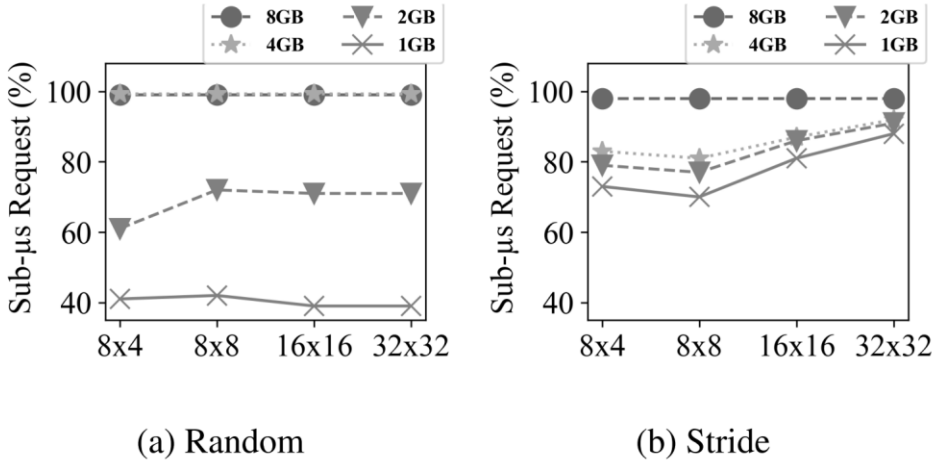


(a) Average access latency (b) Estimated lifetime

- With (larger) DRAM cache
  - Lower average access latency
  - Longer estimated lifetime

D5. Parallelism





- SSD
  - Channel: separate interface between the SSD controller and the NAND flash memory chips.
  - Way: number of dies that can be accessed simultaneously within a single channel.

- Higher parallelism, better latency
  - Parallelism works well in CXL-flash, too.

Summary of Design

- Challenges and Design of CXL-flash
  1) Granularity mismatch: Cache, MSHRs, Prefetcher
  2) Micro-second Latency: Cache, MSHRs, Prefetcher, Parallelism
  3) Limited endurance : Cache, Flash technology

- Design Details
  - DRAM Cache : Lower latency and lower traffic
  - MSHR : Lower latency and prevent repeated flash read
  - Prefetcher : hide long flash read latency
  - Flash technology: with DRAM Cache, better latency & lifetime
  - Parallelism: higher parallelism, better latency

Evaluation

- Objectives
  - How effective are the cache policies?
  - How effective are the prefetchers?
  - Is CXL-flash a good memory expansion option?
  - How is the performance difference between virtual and physical traces?

Workload	Category	Description
BERT [18]	NLP	Infers using a transformer model
Page rank [6]	Graph	Computes the page rank score
Radiosity [17]	HPC	Computes the distribution of light
XZ [21]	SPEC	Compresses data in memory
YCSB F [22]	KVS	Read-modify-writes on Redis [14]

Table 6: Default parameters for the CXL-flash in § 5.

Parameters	Value
DRAM size	64MiB
DRAM latency	46ns
Flash parallelism	8 × 8
Flash technology	ULL (Table 1)

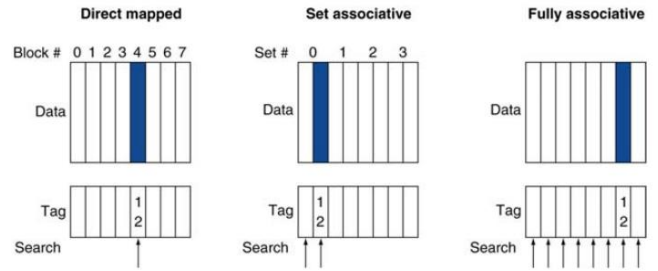
Table 1: Overview of memory technology characteristics.

Technology	Read latency	Program latency	Erase latency	Endurance limit
DRAM [50]	46ns	46ns	N/A	N/A
ULL [46, 76]	3μs	100μs	1000μs	100K
SLC [24]	25μs	200μs	1500μs	100K
MLC [24]	50μs	600μs	3000μs	10K
TLC [24]	75μs	900μs	4500μs	3K

E1. Cache replacement policy

- How effective are the cache policies?
  - CFLRU outperforms which prioritizes evicting clean cache lines
  - Higher set associativity
    → Higher cache hit rate and performance.

- Cache replacement policy
  - **FIFO**: evicts the oldest data
  - **Random**: selects data arbitrarily to evict
  - **LRU**: kicks out the least recently used data
  - **CFLRU**: prefers to evict clean cache lines



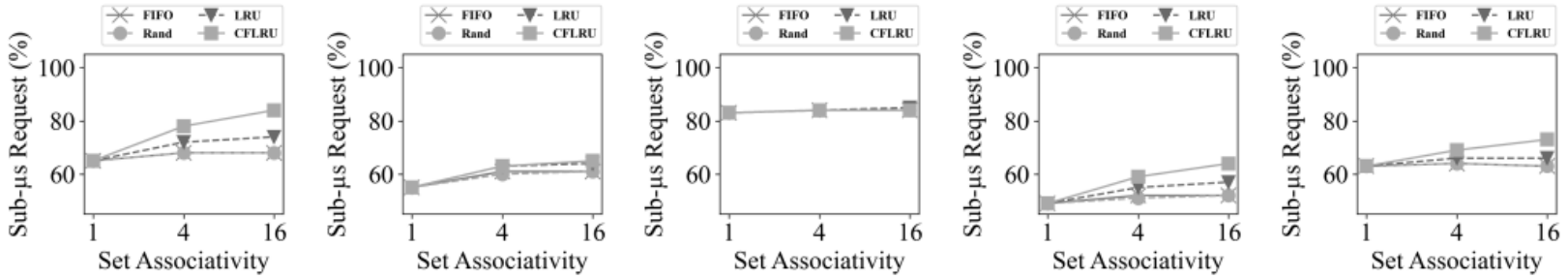


Figure 11: Percentage of CXL-flash latencies smaller than a microsecond with respect to cache replacement policies and set associativity. In general, increasing associativity reduces the latency and CFLRU performs better than the others.

E1. Cache replacement policy

- Workloads with **high** localities: Radiosity
  - insensitive to cache replacement policies
- Workloads with **low** localities: Page rank, XZ
  - perform poorly & less sensitive to policies.

- Cache replacement policy
  - **FIFO**: evicts the oldest data
  - **Random**: selects data arbitrarily to evict
  - **LRU**: kicks out the least recently used data
  - **CFLRU**: prefers to evict clean cache lines

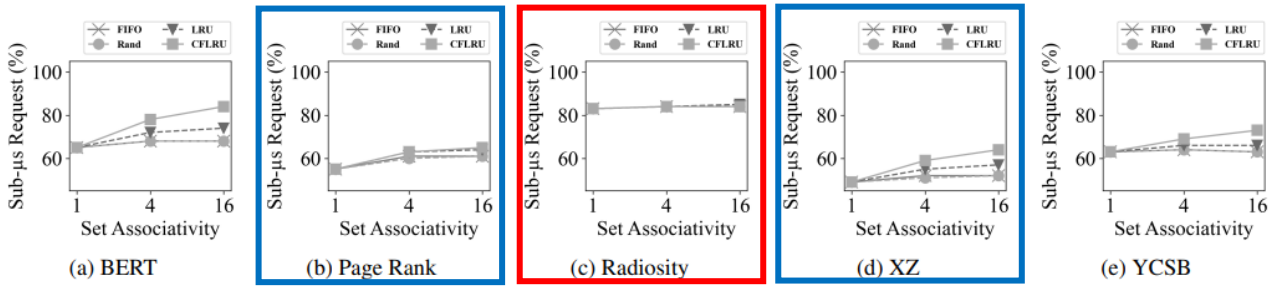


Figure 11: Percentage of CXL-flash latencies smaller than a microsecond with respect to cache replacement policies and set associativity. In general, increasing associativity reduces the latency and CFLRU performs better than the others.

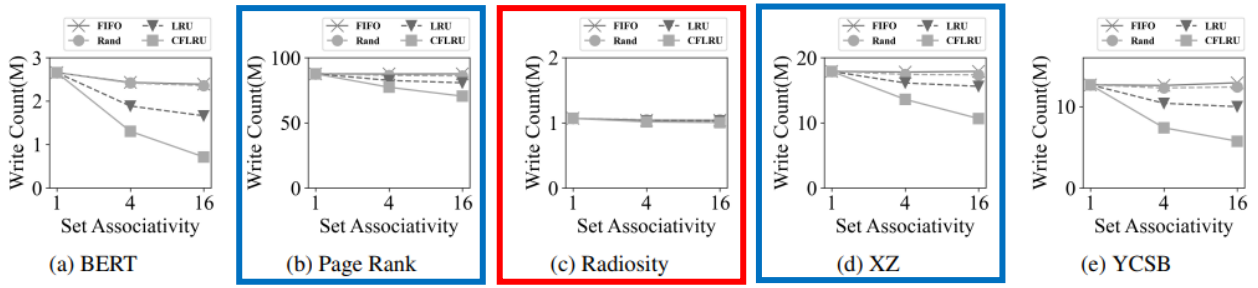
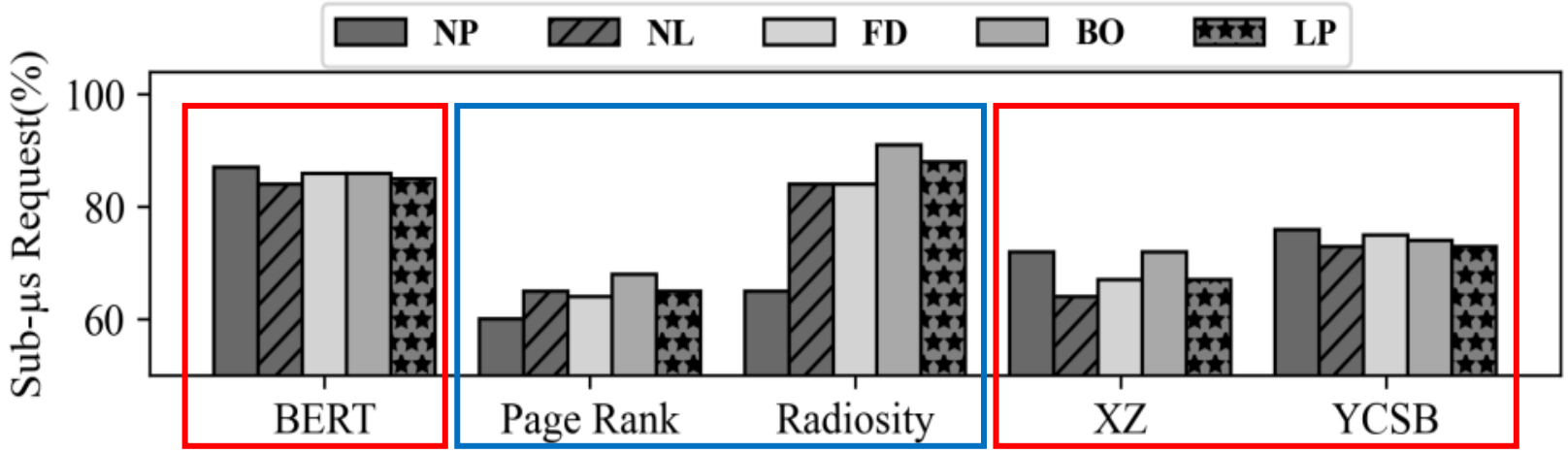


Figure 12: Number of flash memory write requests with respect to cache replacement policies and set associativity. CFLRU noticeably reduces the number of writes as the associativity increases.

E2. Prefetching policy

- How effective are the prefetchers?
  - Using a prefetcher can sometimes **improve** or **hurt** performance

- Prefetch Policy
  - **NP (No prefetch)**: does not prefetch any data.
  - **NL (Next-N-line)**: brings in the next N data upon a demand miss or prefetch hit.
  - **FD (Feedback-directed)**: dynamically adjusts the aggressiveness of the prefetcher by tracking prefetcher accuracy, timeliness, and pollution.
  - **BO (Best-offset)**: learns the deltas between consecutive accesses by tracking the history of recent requests.
  - **LP (Leap)**: implements a majority-based prefetching with dynamic window size adjustment.



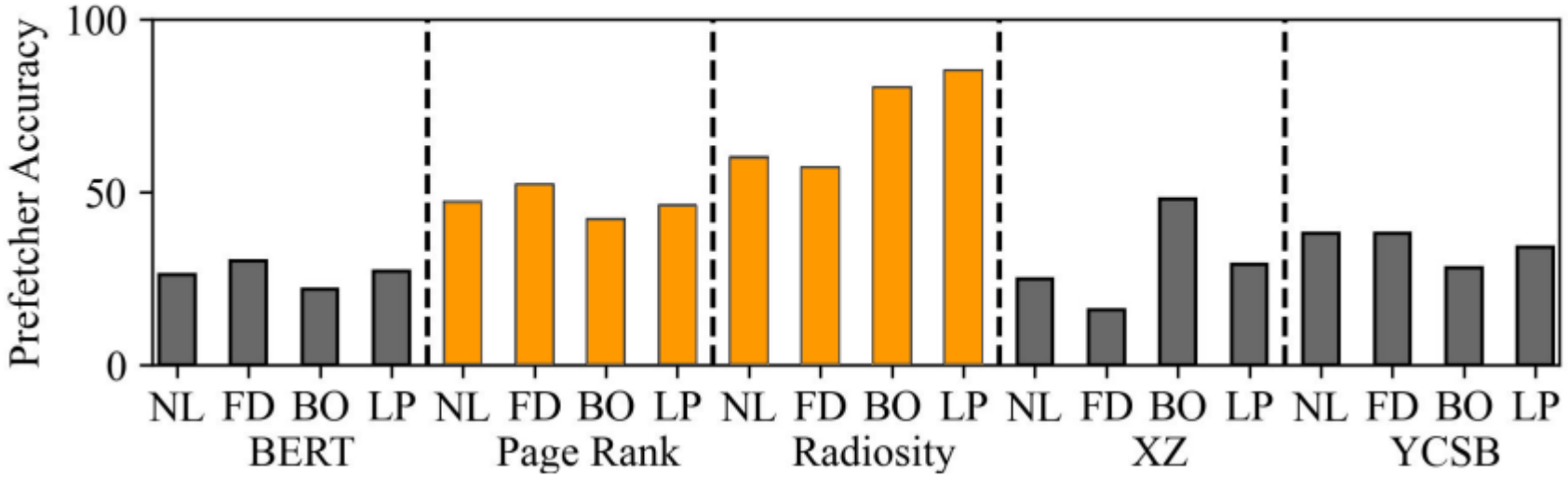
(a) Percentage of sub-μs requests

E2. Prefetching policy

- Why does prefetcher improve performance?
  - it is due to achieving high accuracy

- Prefetch Policy
  - **NP (No prefetch)**: does not prefetch any data.
  - **NL (Next-N-line)**: brings in the next N data upon a demand miss or prefetch hit.
  - **FD (Feedback-directed)**: dynamically adjusts the aggressiveness of the prefetcher by tracking prefetcher accuracy, timeliness, and pollution.
  - **BO (Best-offset)**: learns the deltas between consecutive accesses by tracking the history of recent requests.
  - **LP (Leap)**: implements a majority-based prefetching with dynamic window size adjustment.

$$\text{Accuracy} = \frac{\text{Accessed prefetched data}}{\text{Total prefetched data}}$$

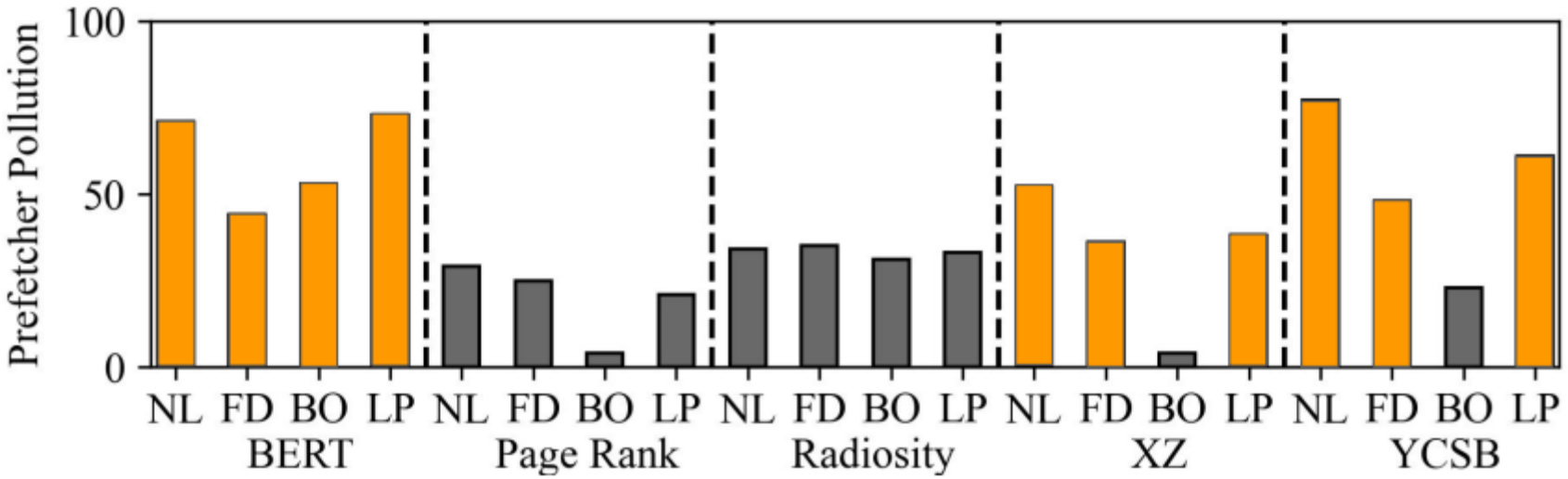


E2. Prefetching policy

- Why does prefetcher degrade performance?
  - it is due to cache pollution

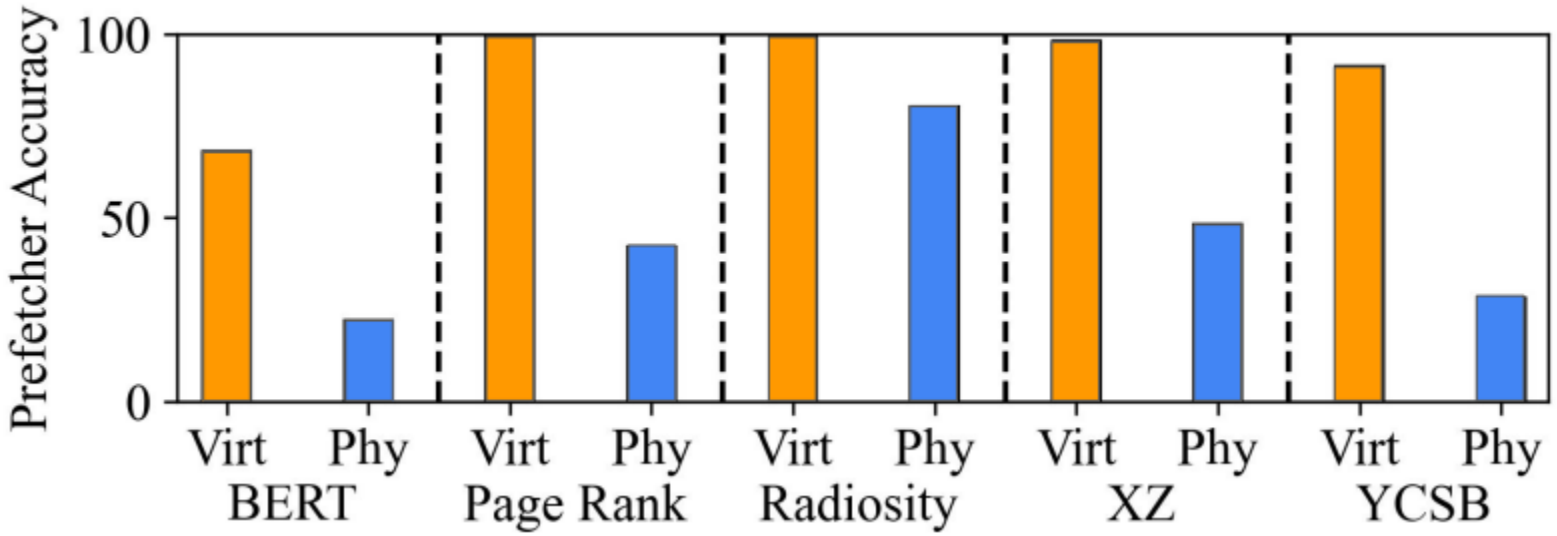
- Prefetch Policy
  - **NP (No prefetch)**: does not prefetch any data.
  - **NL (Next-N-line)**: brings in the next N data upon a demand miss or prefetch hit.
  - **FD (Feedback-directed)**: dynamically adjusts the aggressiveness of the prefetcher by tracking prefetcher accuracy, timeliness, and pollution.
  - **BO (Best-offset)**: learns the deltas between consecutive accesses by tracking the history of recent requests.
  - **LP (Leap)**: implements a majority-based prefetching with dynamic window size adjustment.

$$\text{Pollution} = \frac{\text{Cache miss due to prefetching}}{\text{Total cache miss}}$$



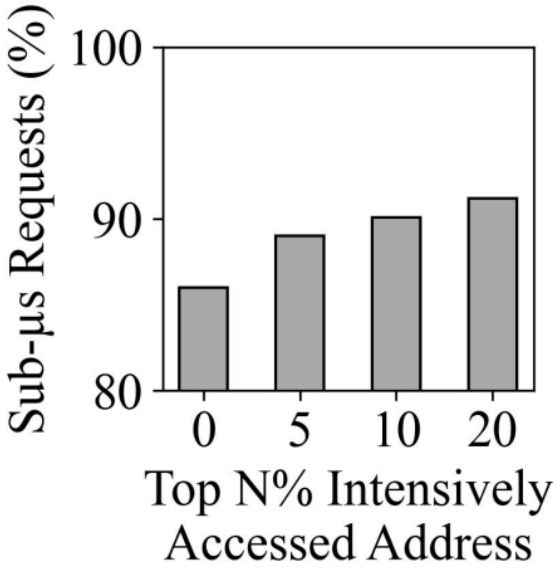
E3. Virtual vs Physical

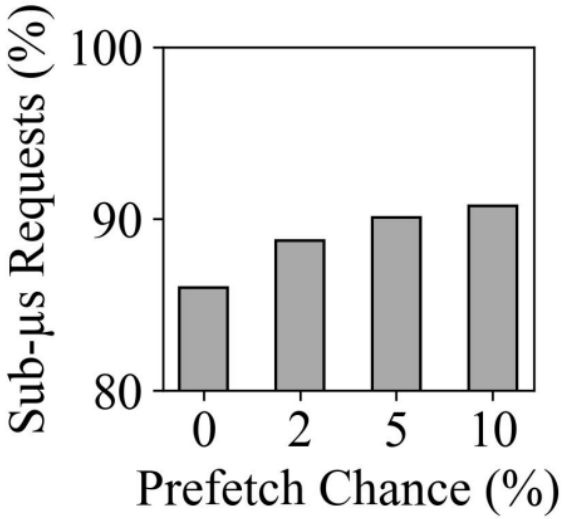
- How is the performance difference between traces?
  - The V2P address translation makes it difficult to accurately prefetch data



E3. Virtual vs Physical

- How can the performance be further improved?
  - Host-generated access pattern hints can improve performance
    - the kernel has information on the top intensively accessed physical frames
    - pass hints to the device prior to their actual accesses
  - Data-intensive applications often iterate multiple times and their behaviors can be profiled.





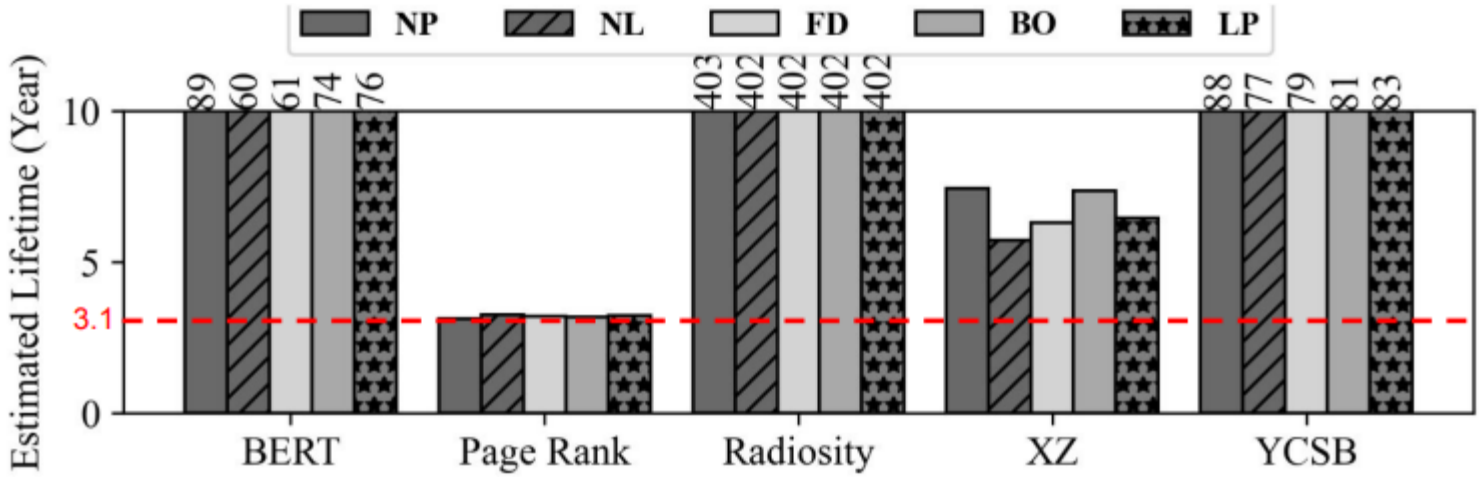
*With BERT

E4. Lifetime

- Does CXL-flash have a reasonable lifetime?
  - CXL-flash can last for at least 3.1 years

Table 1: Overview of memory technology characteristics.

Technology	Read latency	Program latency	Erase latency	Endurance limit
DRAM [50]	46ns	46ns	N/A	N/A
ULL [46, 76]	3μs	100μs	1000μs	100K
SLC [24]	25μs	200μs	1500μs	100K
MLC [24]	50μs	600μs	3000μs	10K
TLC [24]	75μs	900μs	4500μs	3K



E5. Cost-Benefit

- Is CXL-flash a good memory expansion option?

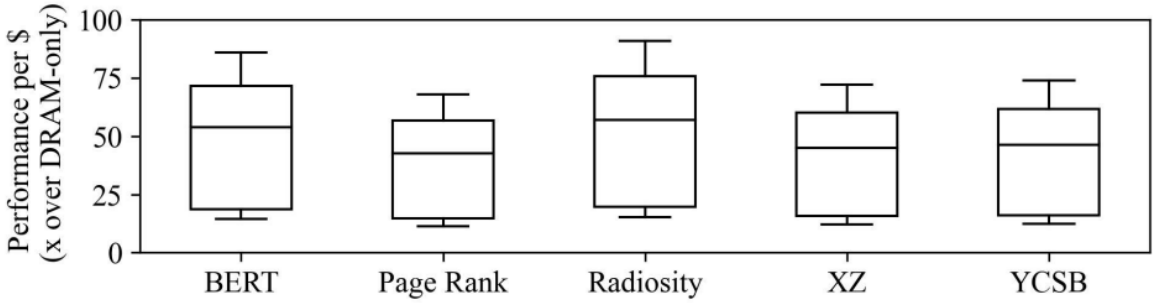
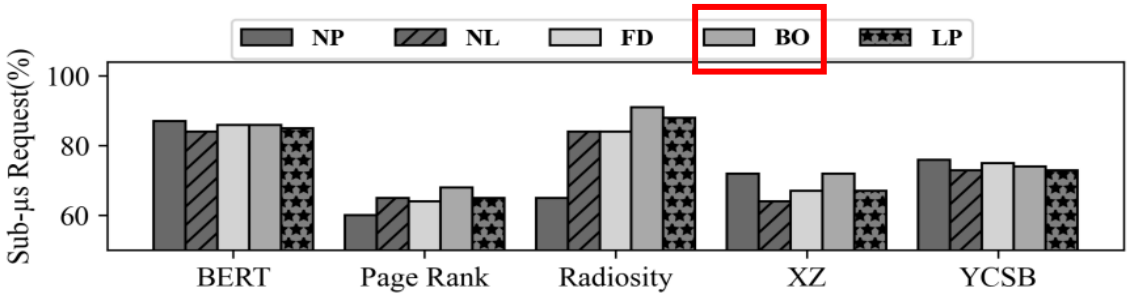


Figure 14: Performance-per-cost benefits of a CXL-flash with BO prefetcher over a DRAM-only device.



(a) Percentage of sub-μs requests

- Cost
  - CXL-flash: 0.05 ~ 0.30 $/GB
  - DRAM: 5$/GB
- Metrics = $\frac{\frac{Sub\text{-}\mu s\ request\ \%_{CXL\_flash}}{Cost_{ULL\ flash}}}{\frac{Sub\text{-}\mu s\ request\ \%_{DRAM\_Only}}{Cost_{DRAM}}}$

Summary of Evaluation

- Cache Efficiency
  - Cache CFLRU outperforms
- Prefetcher
  - Higher prefetch accuracy, higher performance (lower accuracy pollutes cache)
- Virtual vs Physical
  - V2P address translation makes prefetch difficult → kernel hint
- Lifetime
  - CXL-flash can last for at least 3.1 years
- Cost-Benefit
  - 11 - 91x performance-per-cost benefit

Conclusion

- CXL-flash can address the Memory Wall through its cost-effective high capacity and scalability.

- However, CXL-flash faces challenges such as 1) granularity mismatch, 2) micro-second latency, and 3) limited endurance, which can be partially mitigated through caching and prefetching.

- Despite these mitigations, there are still performance limitations compared to DRAM, and it raises the question of whether workloads that define a memory wall without considering GPU accelerators actually exist.

Open Questions

- What is differences between flash device using:
  - OS Block interface / CXL-flash interface / SPDK interface
- What is differences between KV-Store with flash device:
  - In-memory KV-Store with CXL-flash interface
  - Disk-based KV-Store with Block interface
- What is benefit of CXL-flash with ZNS SSD?
  - Is it efficient for sequential write & read workload? (e.g., compaction)

Thank you