Flash Memory Summit

Designing Storage ATC using real life workloads

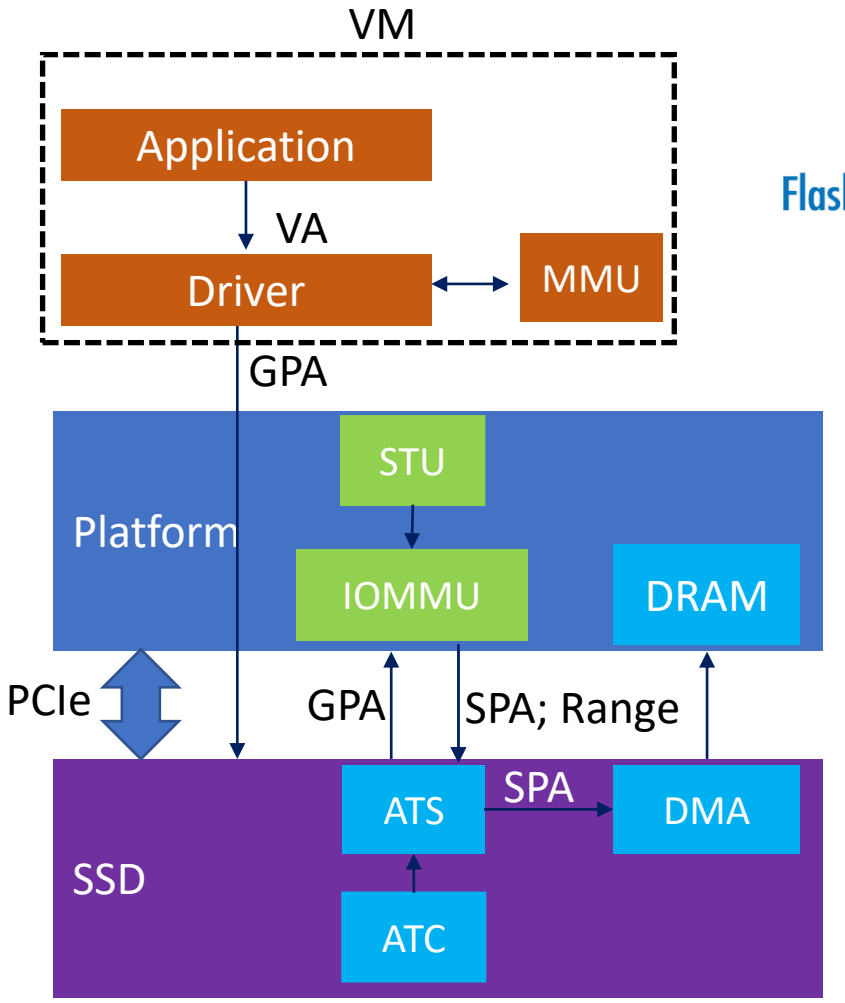
Luca Bert, Micron Technology, Inc.

Problem statement

- New systems will support ATS and ATC protocols
  - An effective way to better handle Address Translations for DMA agents
- We need to design SSD accordingly, but:
  - These are DMA addresses caching subsystems where there is no history on requirements
  - Will have typical cache metrics but there is no usable data to evaluate them
  - They will highly depend on applications DMA maps
- How can SSD design/ size/ configure ATC to maximize their impact to SSD performance?

ATS/ATC Basics

- ATS: Address Translation Service
  - With ATS, IO Device asks for GPA→ SPA translation and uses it for all accesses in that range
  - Translation validity range defined by STU (Smallest Translation Unit).
  - Most common STU values:
    - 4KB (legacy)
    - 2MB (most common with Virtual Machines)
    - 1GB (Huge Pages from User Space).
- ATC: Address Translation Cache
  - Cache holding recent translations
  - Subject to typical cache metrics (locality, eviction, associativity,…)



Legend:
- VA: Virtual Address
- GPA: Guest Physical Address
- SPA: System Physical Address
- STU: Smallest Translation Unit

Data ATC evaluation for Storage

- **Characterization Method**:
  - Assume VM running standard workloads.
  - Trace unique buffers addresses for each workload
  - Map them into STU (2 MB) lower pages
  - Build Python model of ATC
  - Replay traces to model to validate hit rate
  - Repeat for as many workload and config as reasonable

Example: YCSB (Yahoo Cloud Server Benchmark) on RocksDB – Full system vs. VM configurations (16 VM/system)

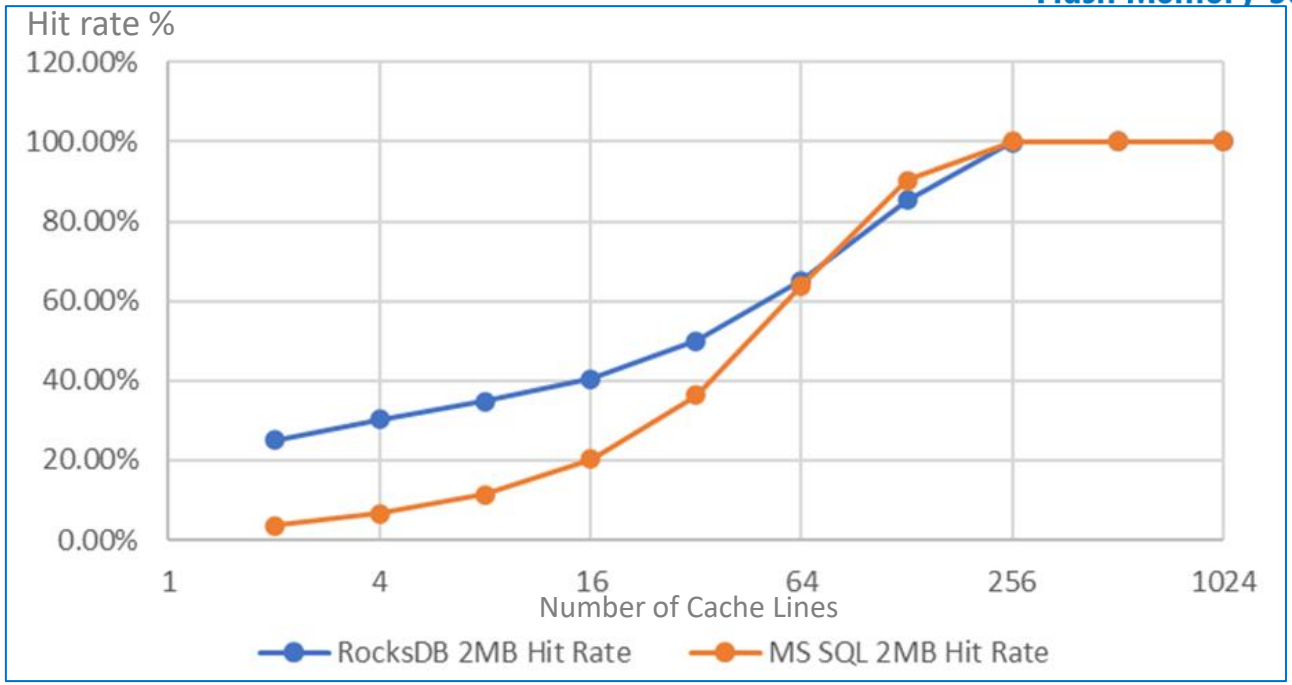
YCSB / RocksDB/ XFS					
Configuration	**Ops/s**	**Total IOs**	**Unique add.**	**Unique ATC pages**	**ATC size for >85% hit**
128 cores, 1TB, 256 th, YCSB WL C	667K	142M	21,659	1,349	128
8c, 64GB, 32 th, YCSB WL A	95K	371M	10,141	322	32 (x16 = 512)
8c, 64GB, 32 th, YCSB WL B	71K	245M	9,133	357	32 (x16 = 512)
8c, 64GB, 32 th, YCSB WL C	95K	200M	8,741	363	32 (x16 = 512)
8c, 64GB, 32 th, YCSB WL D	76K	236M	8,064	320	32 (x16 = 512)

Source: Micron DCWE Lab

Observation: Multiple VM, as expected, has less locality that single image but scaling in not linear (4x size for 16x #VM)

Correlation with TPC-H

Benchmark	YCSB WL C RocksDB XFS	TPC-H SS MS SQL XFS
Config	256 Logical Cores; 1TB DRAM	
STU Size	2MB	
# IOs traced	142M	264M
Unique addresses	21,659	69,493
Unique 2MB pages	1,349	969
Unique 1GB pages	2	3
128 pages hit rate	88%	90%

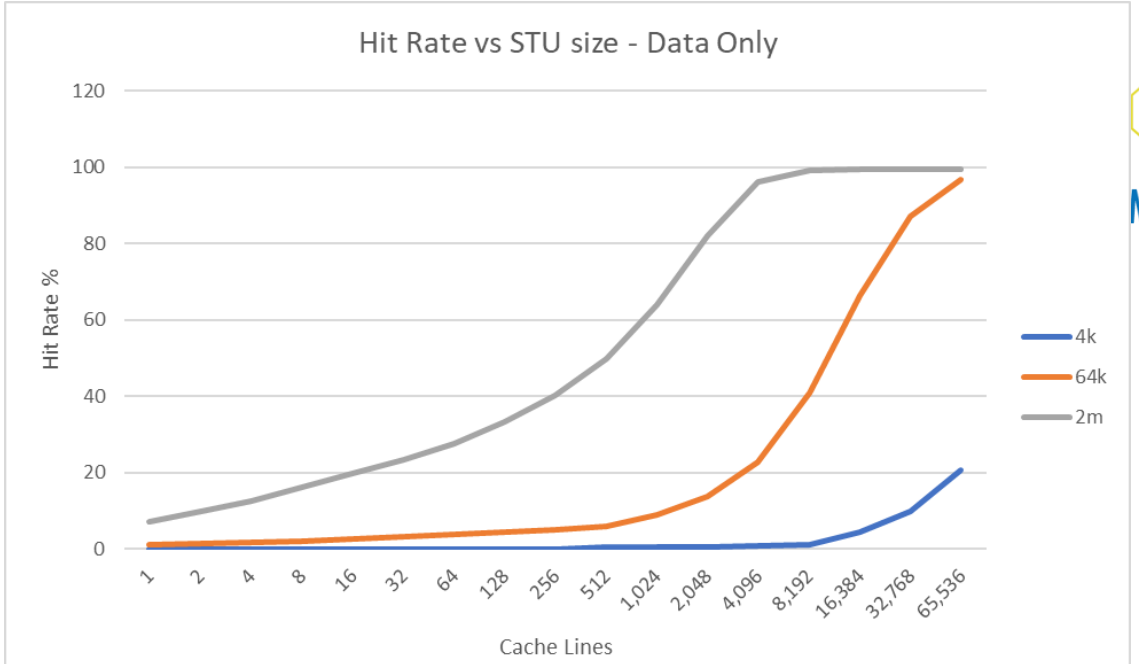


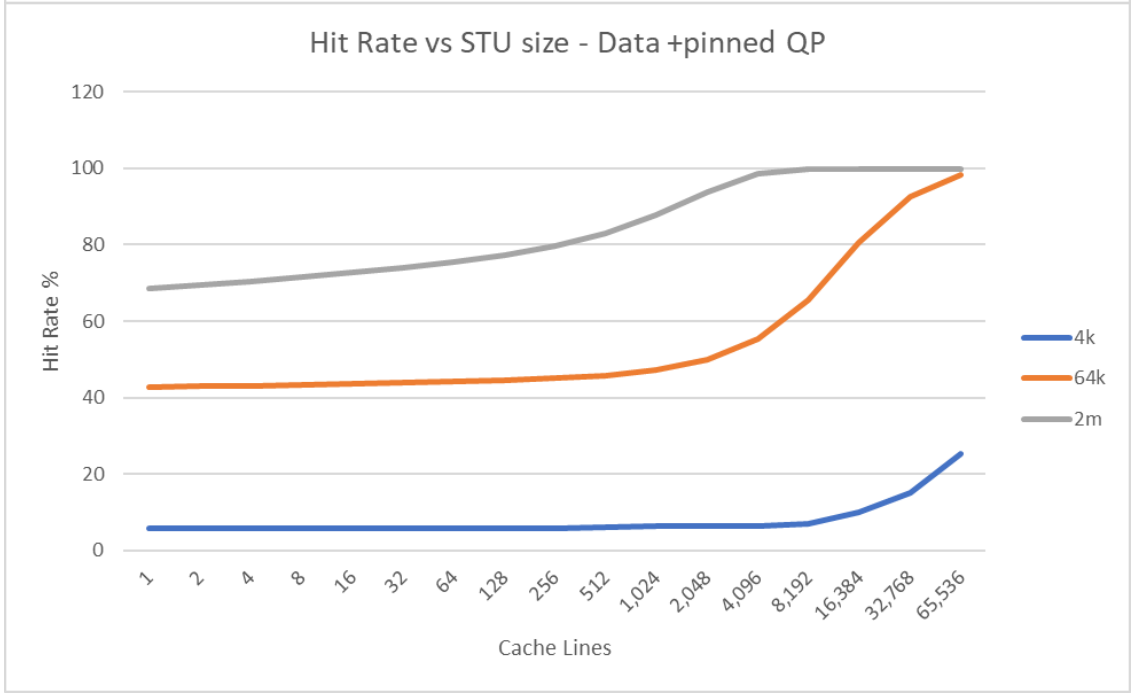
Source: Micron DCWE Lab

- IO distribution very different:
  - 3.2x as many unique address…
  - … but distributed in 70% of STU → Higher locality
- Cache hit rate converging around 64 entries, consistent with RocksDB

Size dependency

- Benchmark used: YCSB WL B with Cassandra
- Cache: 4 way Set Associative
- Algorithm: Round Robin
- Observations:
  - As expected, hit rate is highly dependent on STU size
  - The larger the STU size the better hit rate
  - Not all data are created equal: every NVMe command need access to SQ and CQ so such addresses have an outsized impact on hit rate

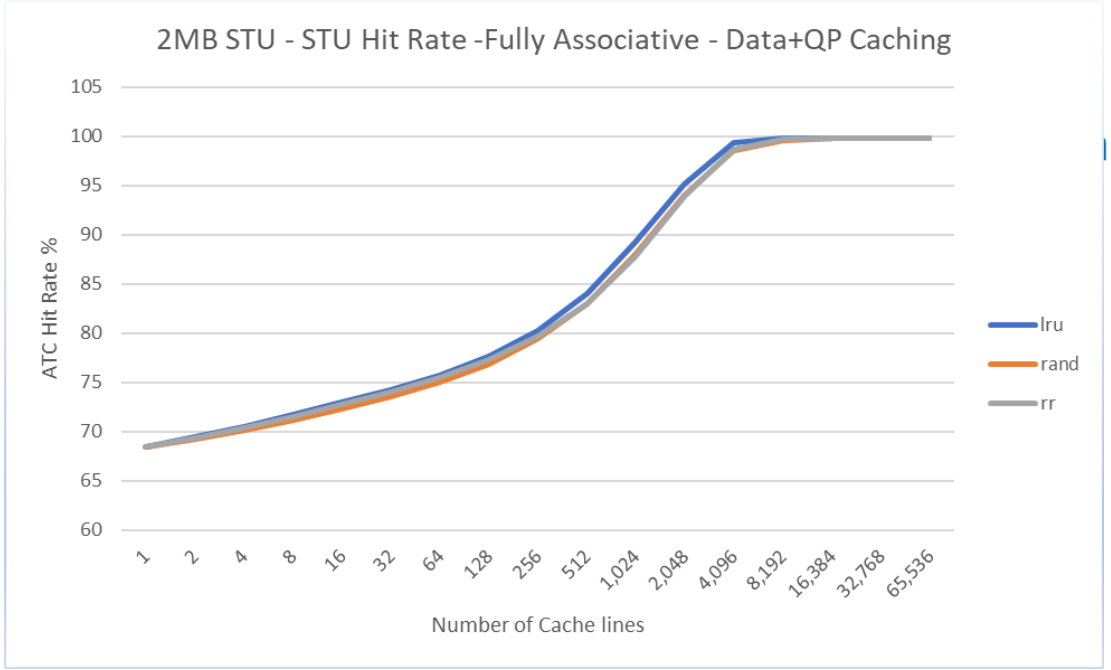


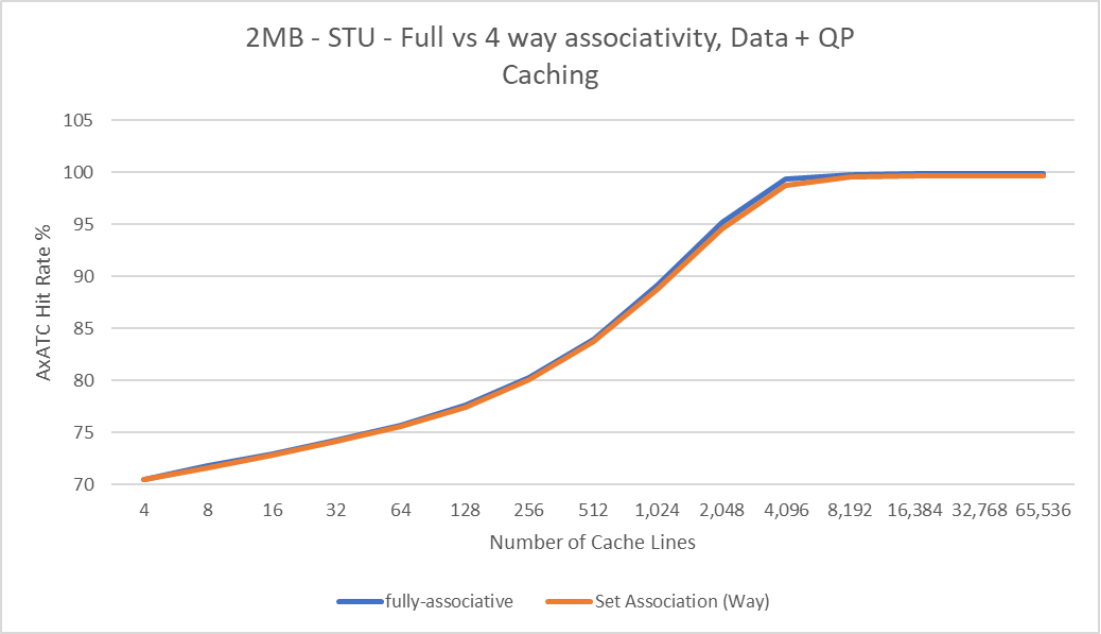


Source: Micron DCWE Lab

Algorithms dependency

- Benchmark used:
  - YCSB WL B with Cassandra
  - Part of a much larger set
- Associativity: Full and 4 ways
- Eviction algo: LRU, Rand and Round Robin
- Outcome:
  - Replacement algorithms do not make any visible difference
  - Selecting the simplest implementation may be the most effective approach





Conclusions

- ATS/ATC are mainly driven by system requirements to provide efficient data transfers
- ATC is an opportunity to improve high performance devices as it exhibits a high hit rate at a moderate silicon cost
- Characterization of current system is possible but:
  - New systems will scale #cores/ DRAM size/ OS capabilities
  - Further modeling work required to cast traces in new configurations
  - Modeling highly dependent on future (and largely unknown) system architectures

Questions?