MASSIVE
IEEE Conference on Massive Data Storage

MSST 2013
2013/5/10

OSSD: A Case for Object-based Solid State Drives

Young-Sik Lee
Sang-Hoon Kim, Seungryoul Maeng, KAIST
Jaesoo Lee, Chanik Park, Samsung
Jin-Soo Kim, Sungkyunkwan Univ.

KAIST

SAMSUNG

SUNG KYUN KWAN UNIVERSITY

SSD

Desktop

Laptop

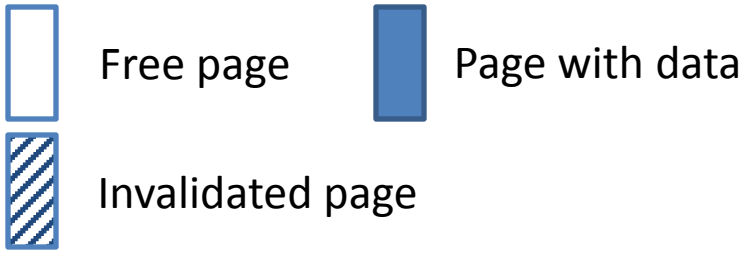
SSD

Server

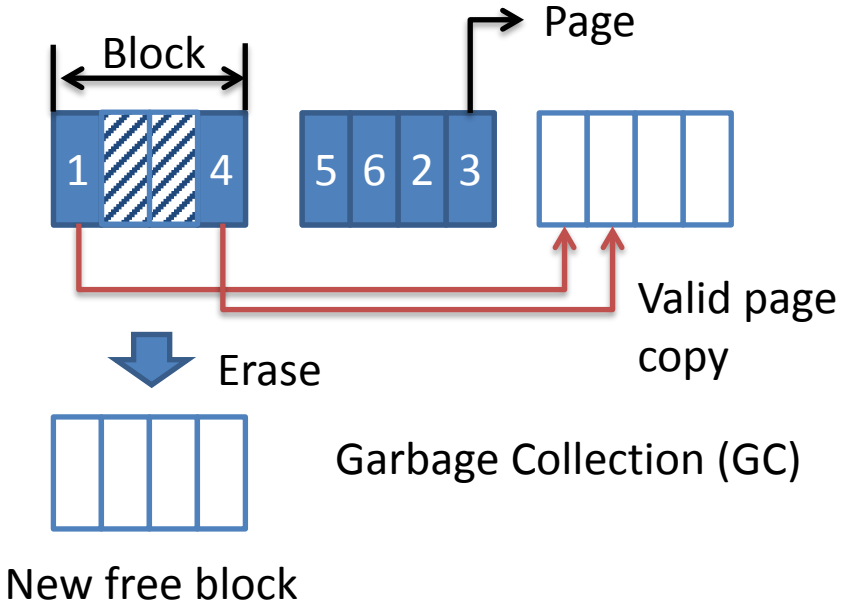
- High performance
- Better shock & vibration resistance
- Low power consumption
- Light weight

Inside of SSDs

- NAND flash memory



- Flash Translation Layer (FTL)
  - Space allocation
  - Address mapping
  - Garbage collection (GC)
  - Wear-leveling

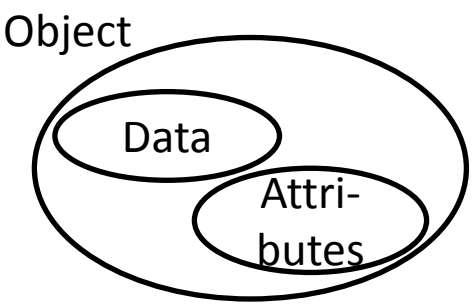


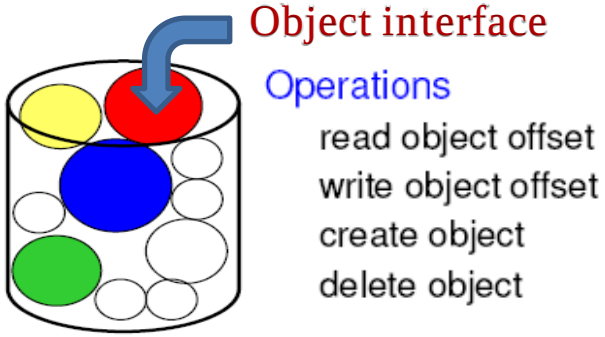
Block Interface

- Connects the host machine with SSDs
- Limitations
  - Space management done in two different layers
    - File system: <file, offset> → file system blocks
    - FTL: file system blocks → flash pages
  - Hard to know liveness information
    - TRIM, but large overhead for highly fragmented files
  - No file-level information
    - Relationship among blocks (e.g., which block from which file)
    - File size, file attributes, etc.
  - Hard to manage per-block data properties
    - e.g., hot vs. cold, metadata vs. data, etc.
    - Additional operations & large memory space

Object-based Storage Device (OSD)

- Virtualizes physical storage as a pool of objects
  - Object = Data + Attributes
- Provides general abstraction layer to manage objects
  - Create object, delete object, read object, write object, etc.
- Enables storage management based on objects
  - Offloads space management to storage devices



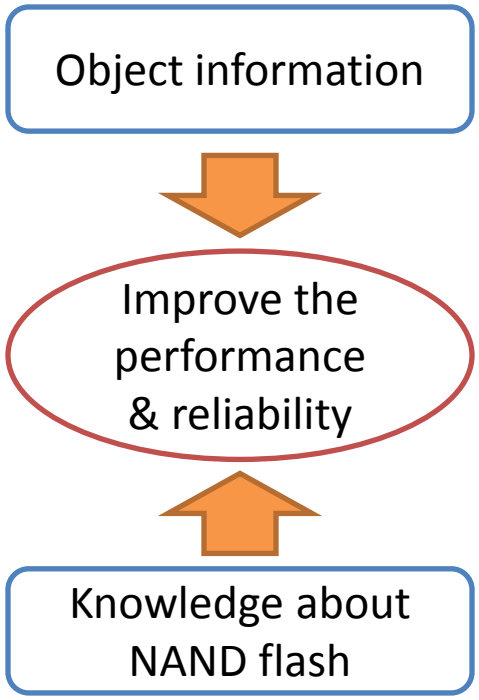


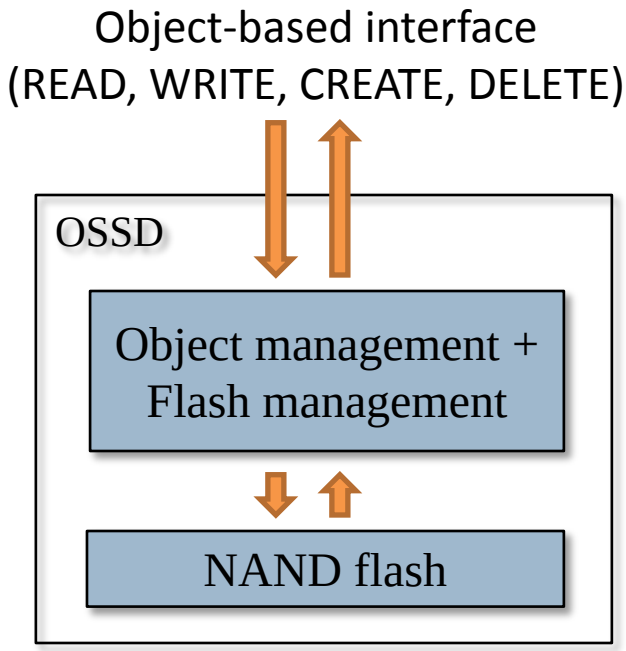
Related Work

- Nagle et al. [IBM Journal of Research and Development 2003]
  - Showed the benefits of OSDs based on HDDs
- Rajimwale et al. [USENIX ATC 2009]
  - Suggested the notion of OSD is well suited to SSDs
- Kang et al. [MSST 2011]
  - Employed OSD for SCM (Storage Class Memory)

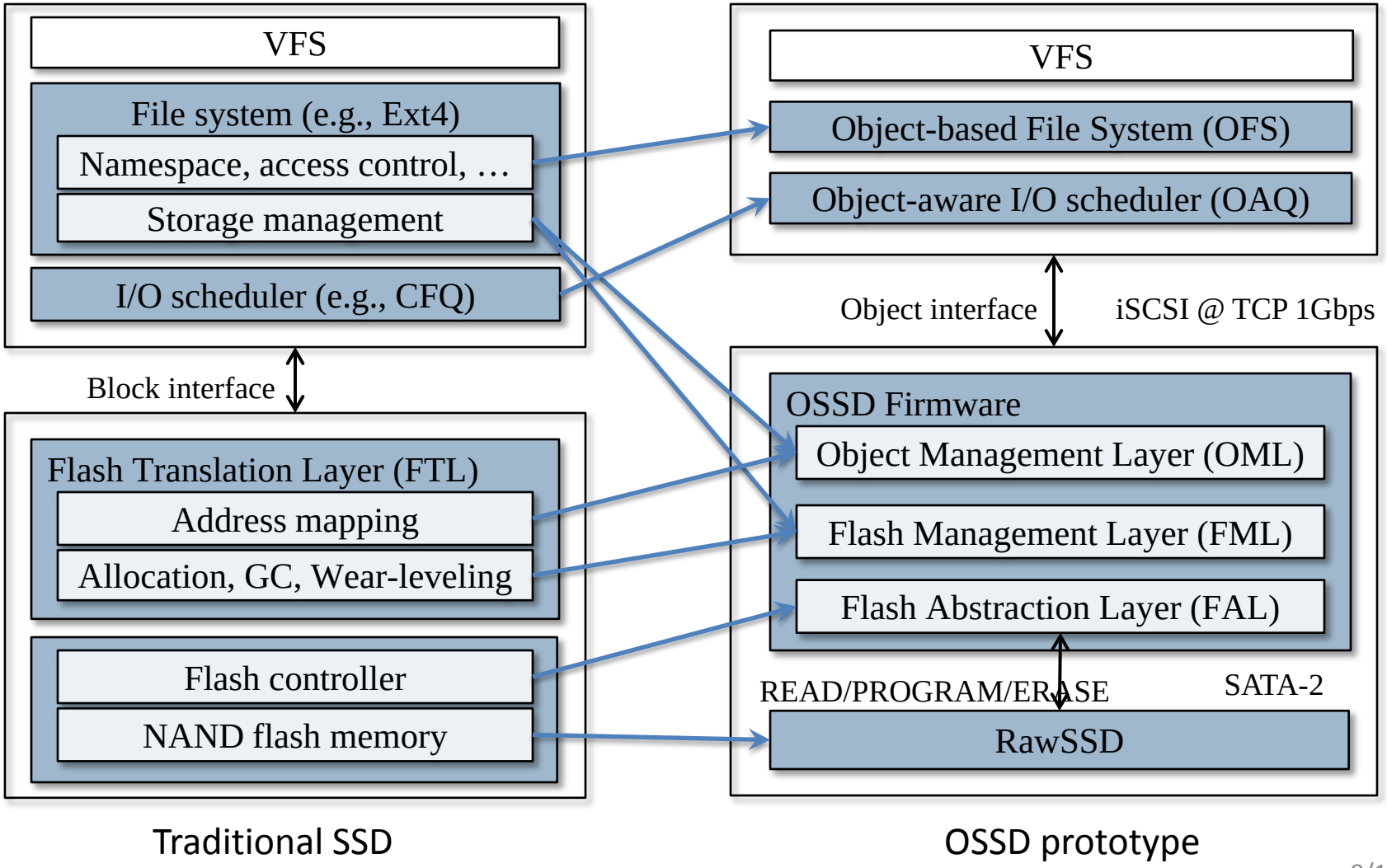
Object-based SSDs (OSSDs)

- OSD + SSD



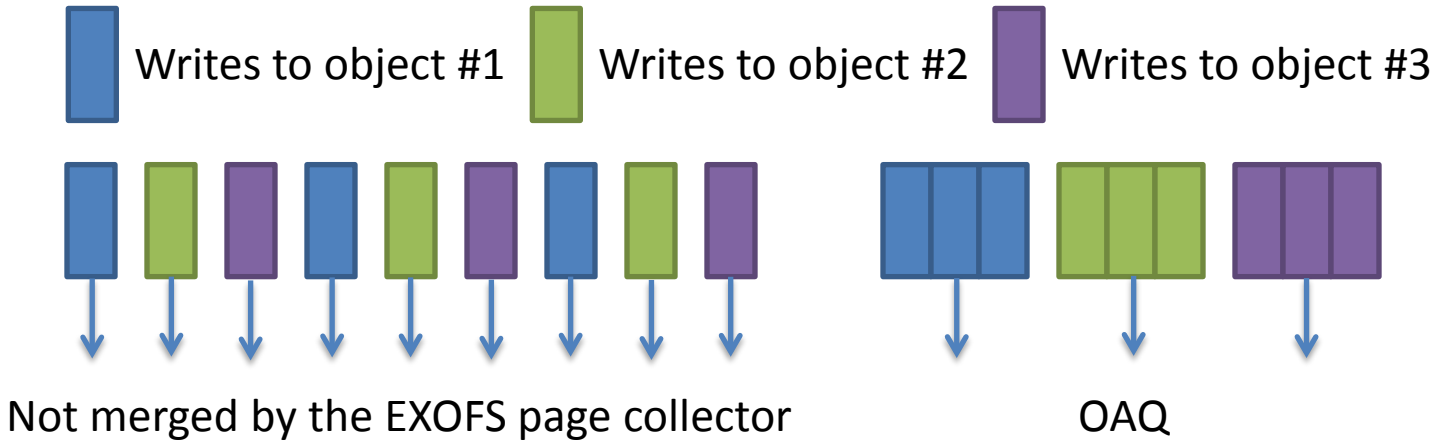


Overall Architecture



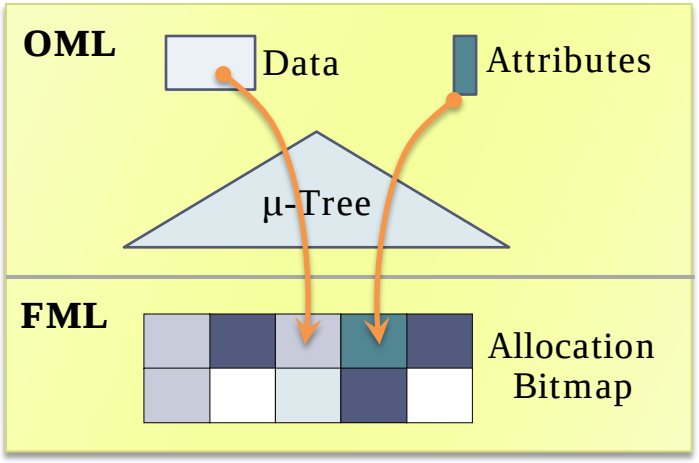
Host

- Object-based File System (OFS)
  - Based on EXOFS
  - One file → one object
- Object-aware I/O Scheduler (OAQ)
  - Replaces the page collector of EXOFS
  - Merges requests and supports priority on an object basis



Target

- Object Management Layer (OML)
  - Uses μ-Tree [EMSOFT 2007] for object data and attributes mapping
- Flash Management Layer (FML)
  - Allocates space
  - Considers data properties
  - Handles prioritized objects

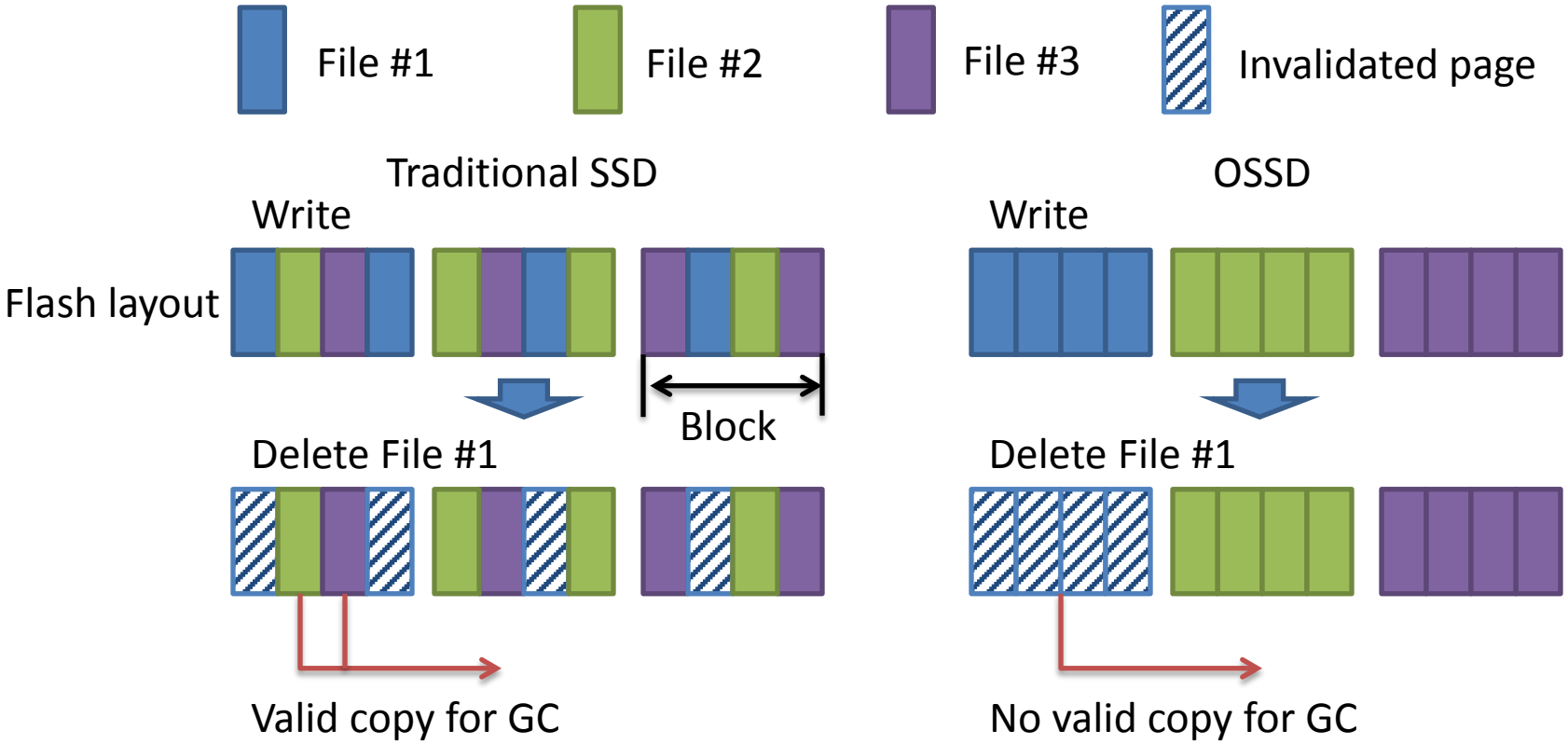


Benefits of OSSDs

- Object-aware data placement
- Hot/cold data separation
- QoS support for prioritized objects

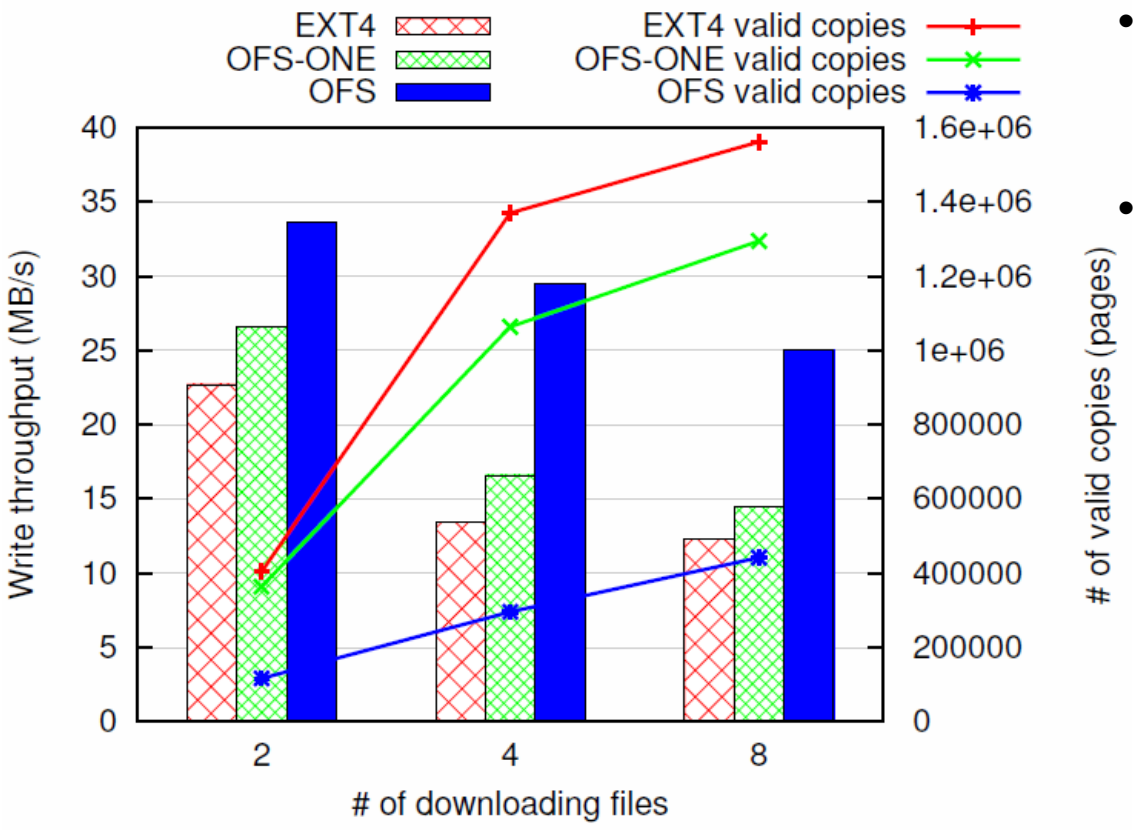
Object-aware Data Placement

- Lower fragmentation of object data
- Improve GC performance



Object-aware Data Placement

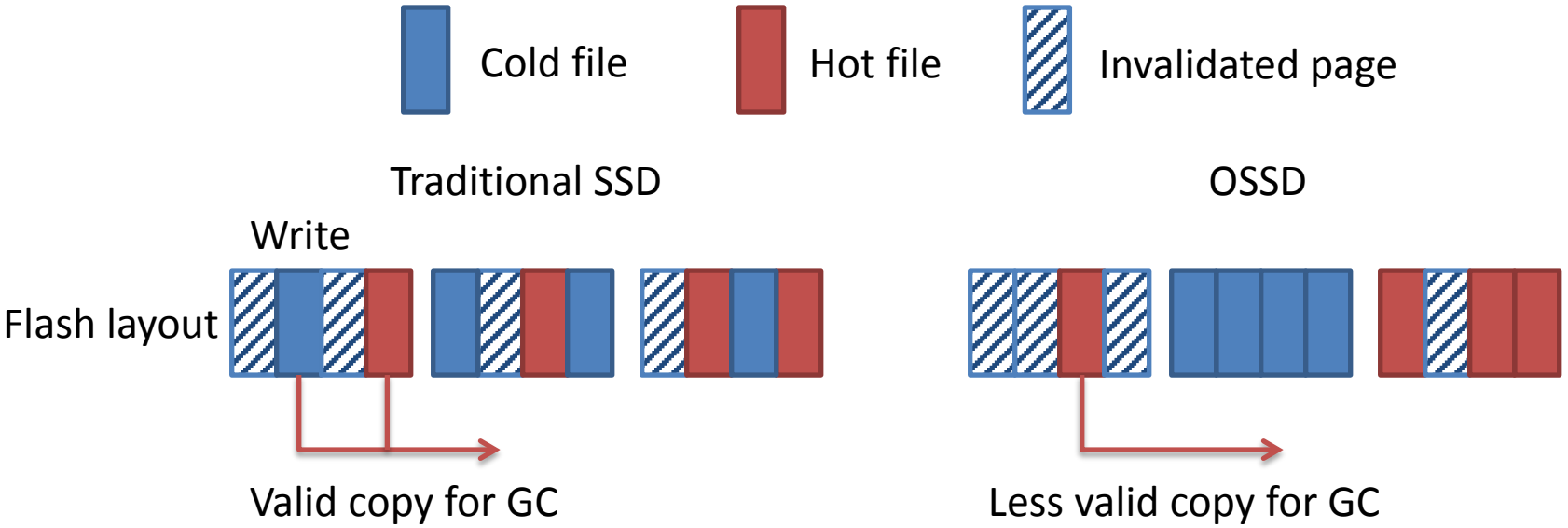
- Downloading 2, 4, or 8 files in Torrent



- EXT4
  - Page mapping FTL on the same H/W
- OFS-ONE
  - No object-aware data placement
  - Shares a single update block for all data

Hot/Cold Data Separation

- Improve GC performance
- Lower overhead for managing hot/cold information

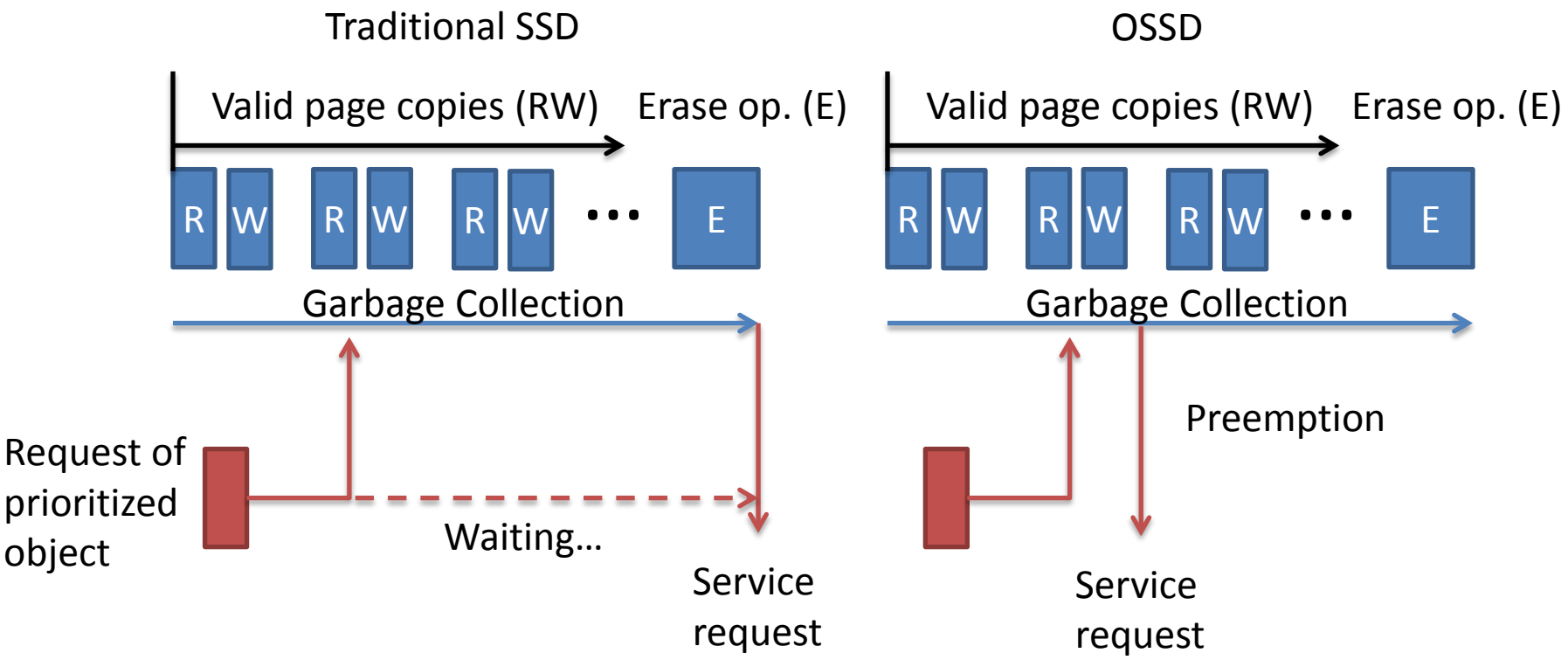


Hot/Cold Data Separation

- 90% write to 10% files, 10% write to 90% files

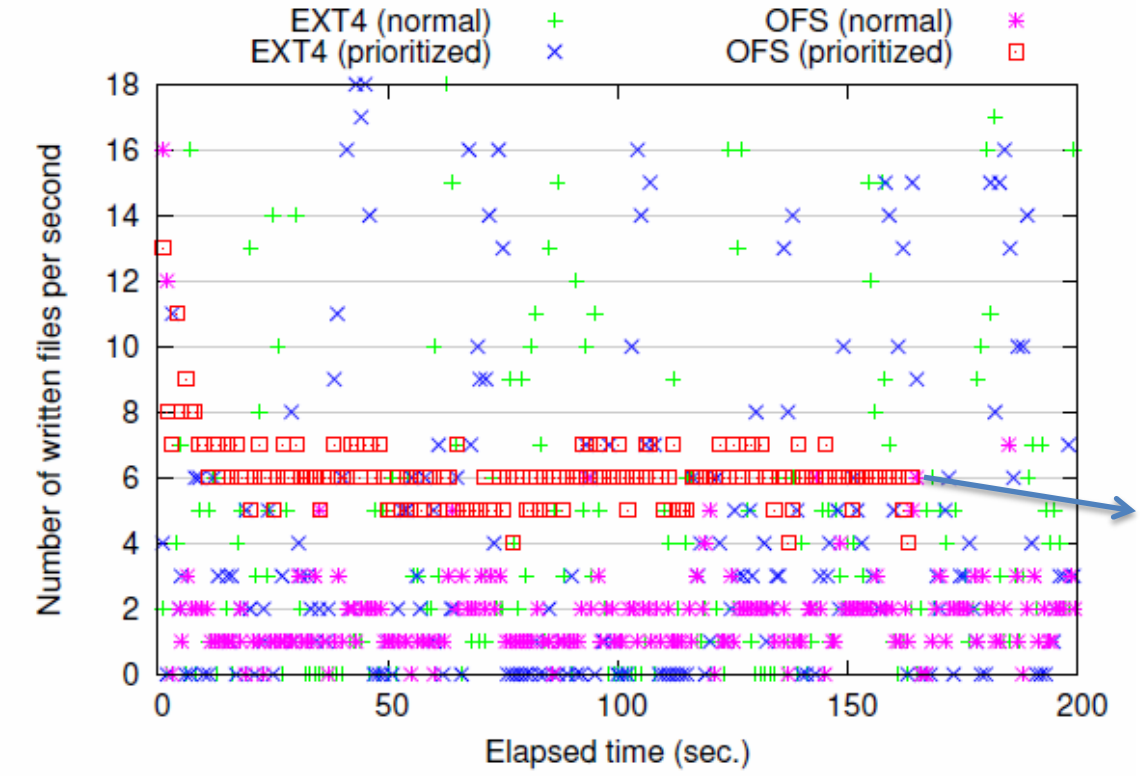
QoS Support for Prioritized Objects

- Provide a low latency service for prioritized objects



QoS Support for Prioritized Objects

- Background: 4 threaded write benchmark
- Foreground (high priority): write 2MB files



- OFS (prioritized) is finished in 164s
- EXT4 (prioritized) is finished in 230s

Conclusion

- We present the design and prototype implementation of the object-based SSDs.

- Benefits of OSSDs
  - Object-aware data placement
  - Hot/cold data separation
  - QoS support for prioritized objects

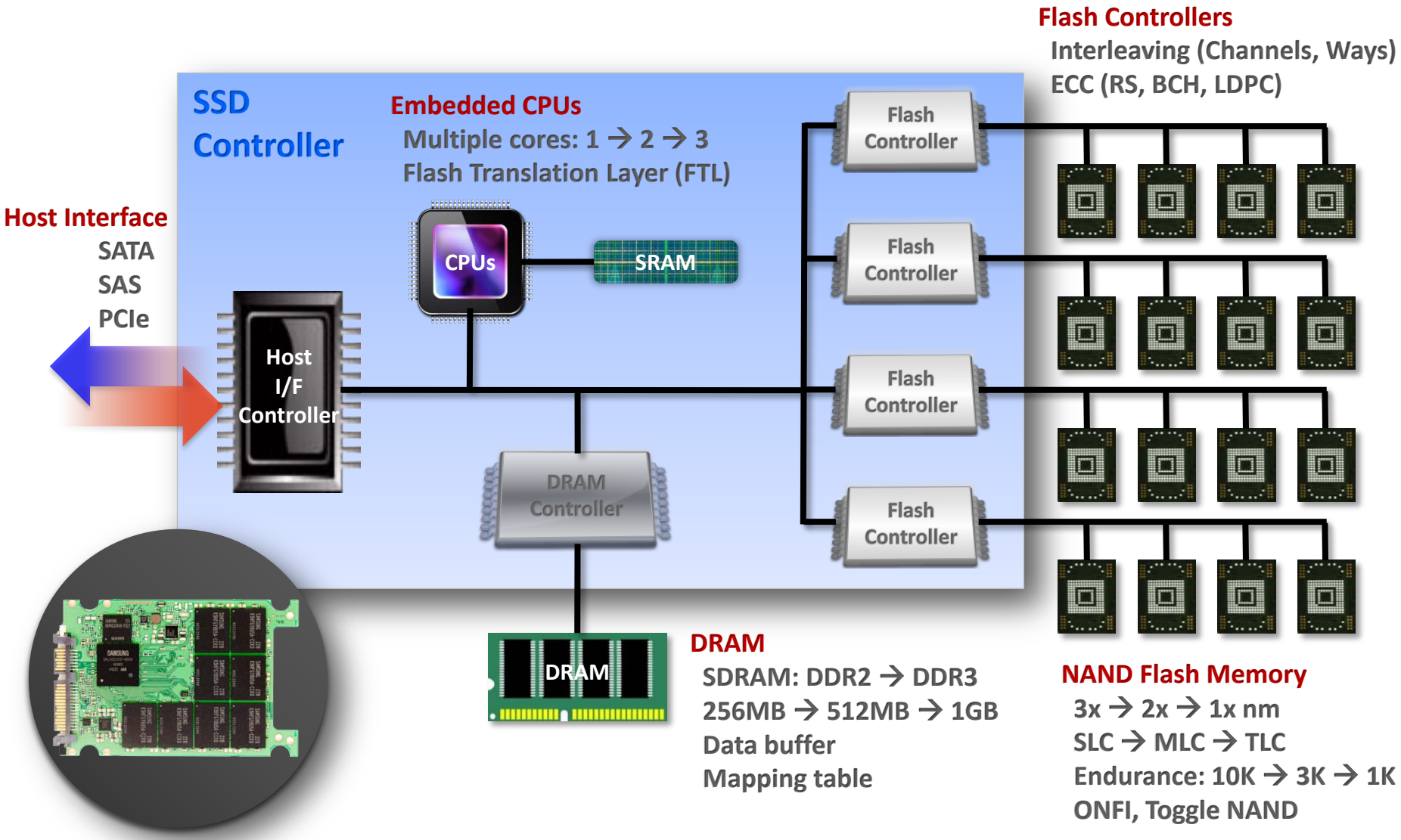
- Future work
  - Ensure metadata reliability (journaling)
  - Find other scenarios to show the benefits of OSSDs

Thank you!

Q&A

BACKUP SLIDE

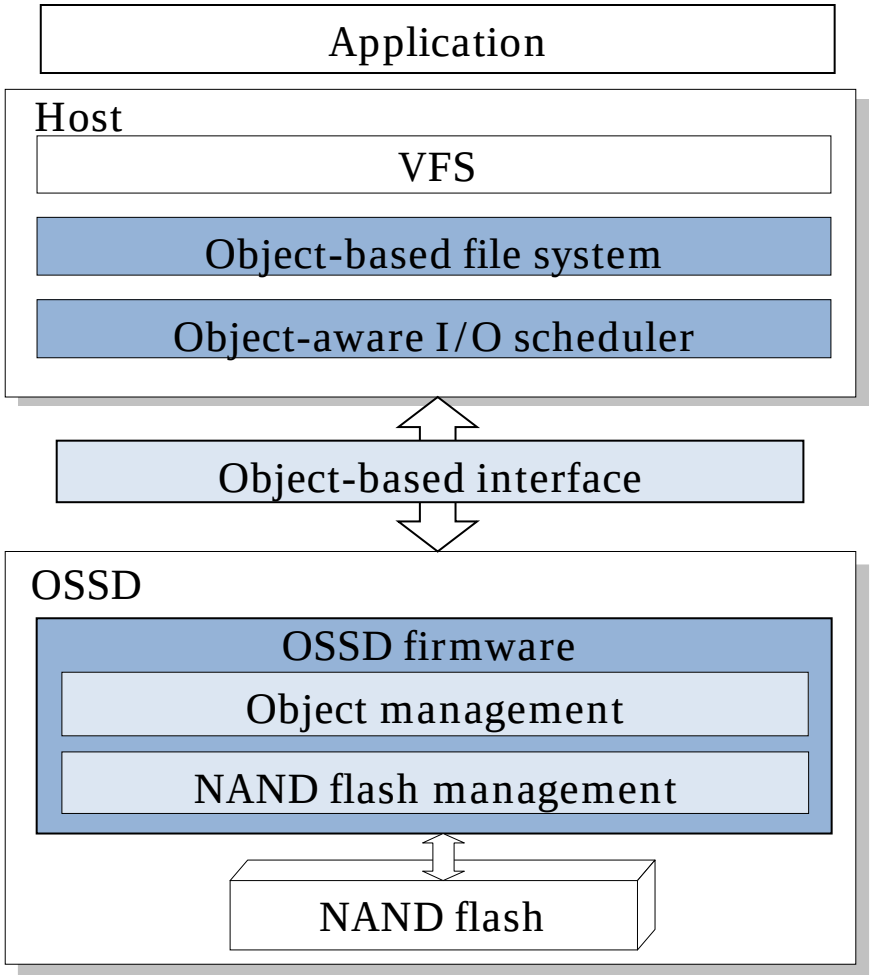
Inside of SSD



Benefits of Object-based SSDs (OSSDs)

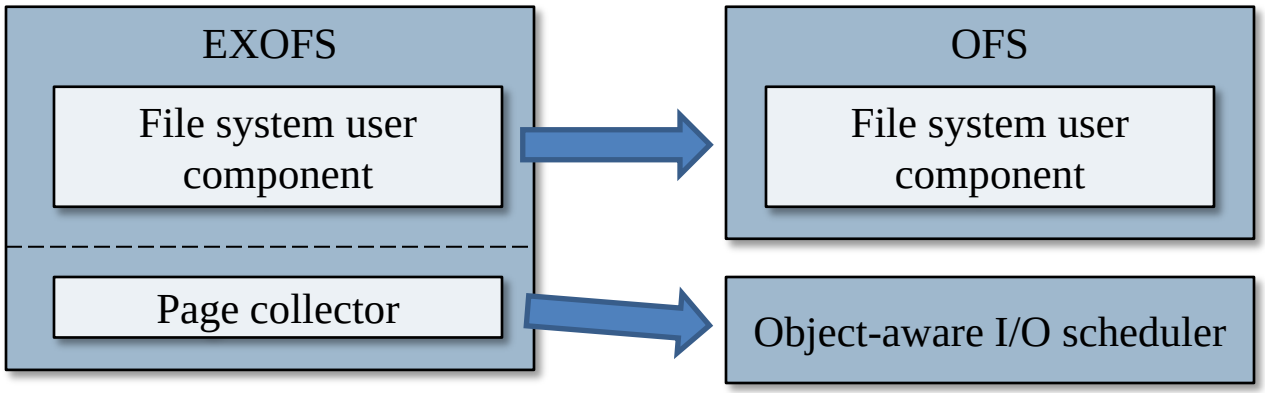
- Simplified host file system
  - Space management in OSSDs
- Utilizing liveness information
  - No valid copies for deleted data
- Metadata management
  - GC performance improvement by metadata separation
- Object-aware data placement
  - Fragmentation reduction of object data
- Hot/cold data separation
  - GC performance improvement
- QoS support for prioritized objects
  - Special services for prioritized objects

Object-based Solid State Drives (OSSDs)



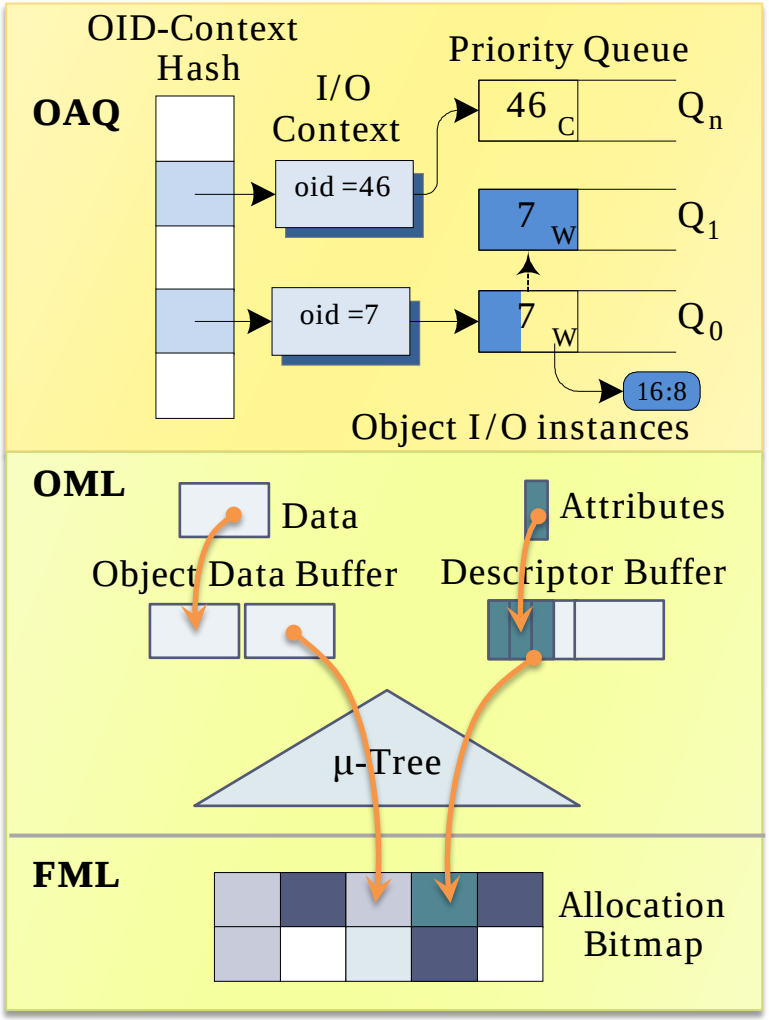
Host

- Object-based File System (OFS)
  - Based on EXOFS
  - One file → one object
- Object-aware I/O Scheduler (OAQ)
  - Replaces page collector of EXOFS
  - Merges requests and supports priority on a object basis

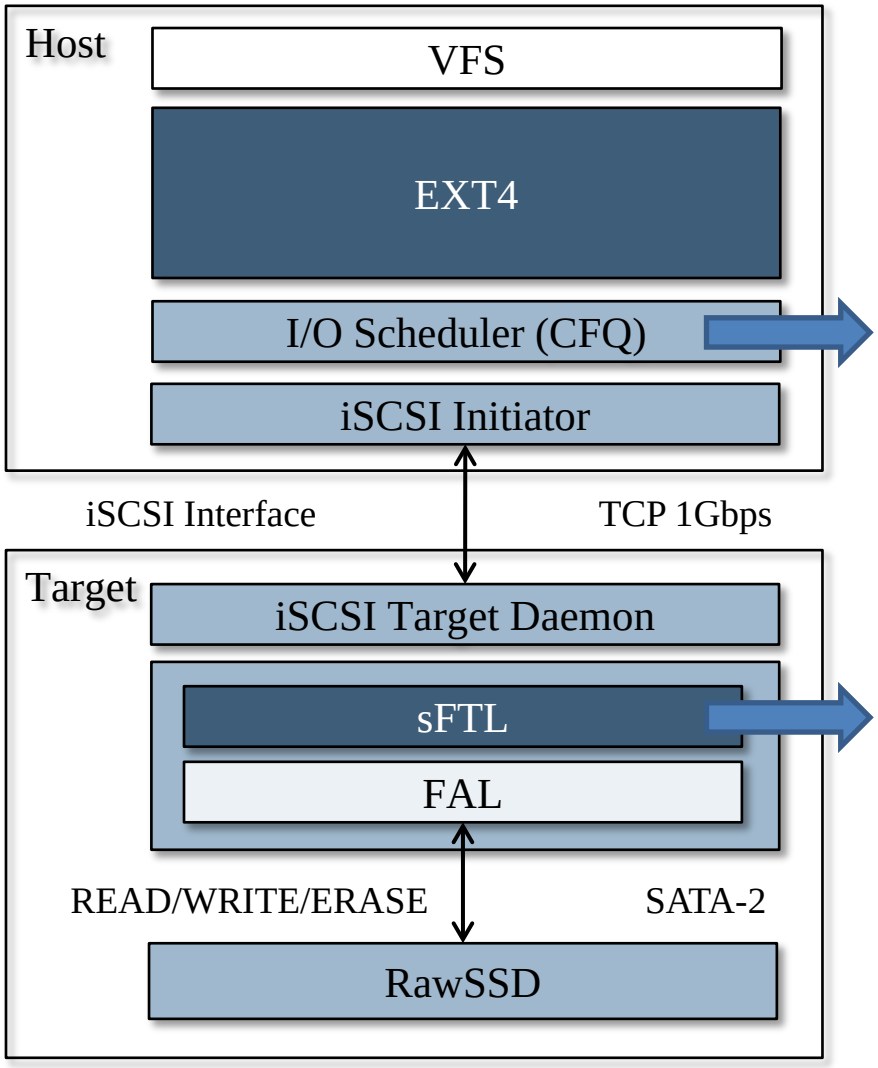


Implementation

- OFS
  - Based on the EXOFS
- OAQ
  - Merges requests by object basis using hash function
  - Supports priority using multiple queues
- OML
  - Uses μ-Tree [EMSOFT 2007] for object data and attributes mapping
- FML
  - Allocates space by data properties
  - Services for prioritized objects by garbage collection preemption



Legacy Support

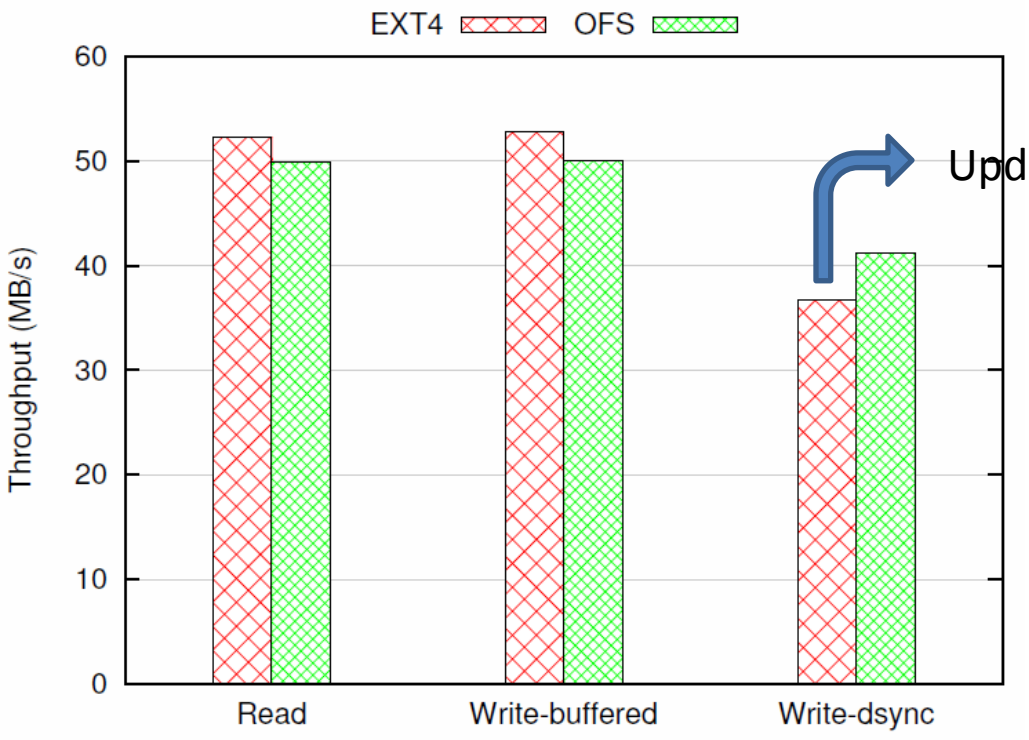


Journaling traffic to DRAM

- Simple page mapping
- The same policies for GC

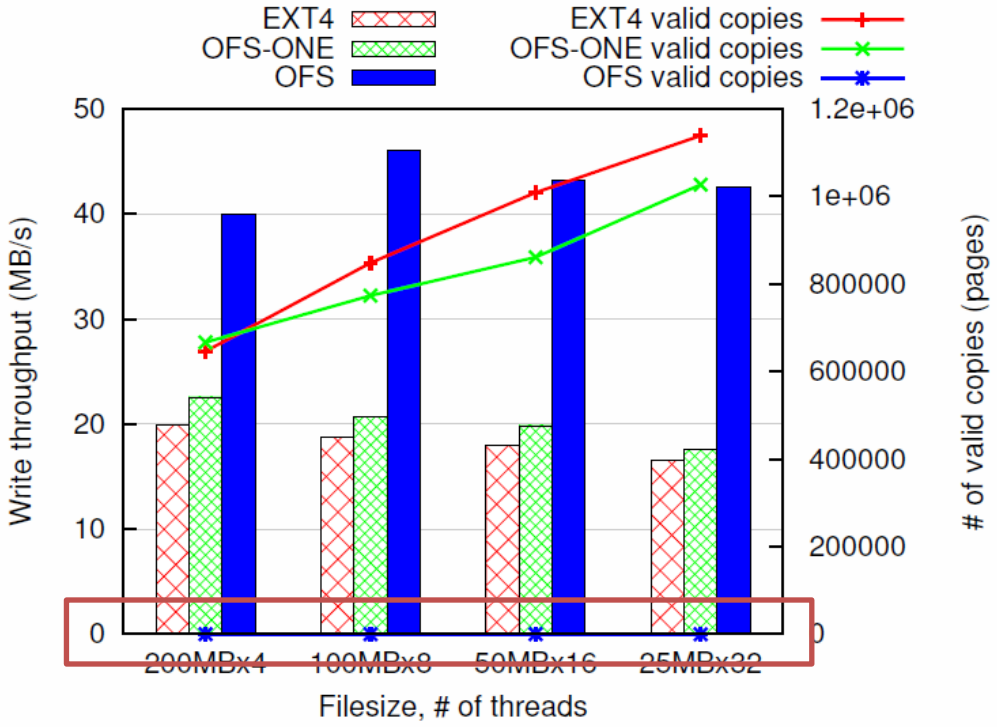
Base Performance

- Sequential accesses



Object-aware Data Placement

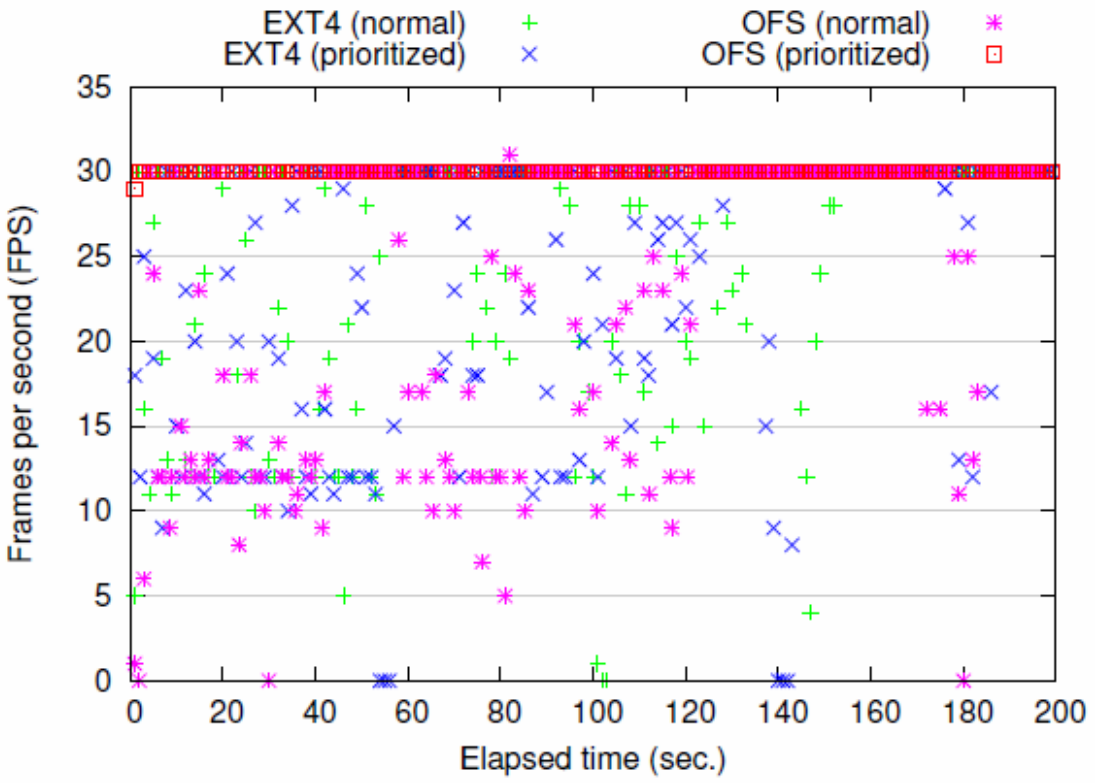
- Micro benchmark
  - Multi-threaded write



(a) Multi-threaded write benchmark.

QoS Support for Prioritized Objects

- Background: 4 threaded write benchmark
- READ (high priority): Playing a music video



File System Benchmarks

- Postmark, Postmark w/o create, Filebench (4 threaded write & delete)
- Aging: hot/cold benchmarks

Read-modify-write overhead for directory updates by create operations