SOS : Software-based Out-of-Order Scheduling for High-Performance NAND Flash-Based SSDs

Sangwook Shane Hahn, Sungjin Lee and Jihong Kim

Computer Architecture & Embedded Systems Laboratory

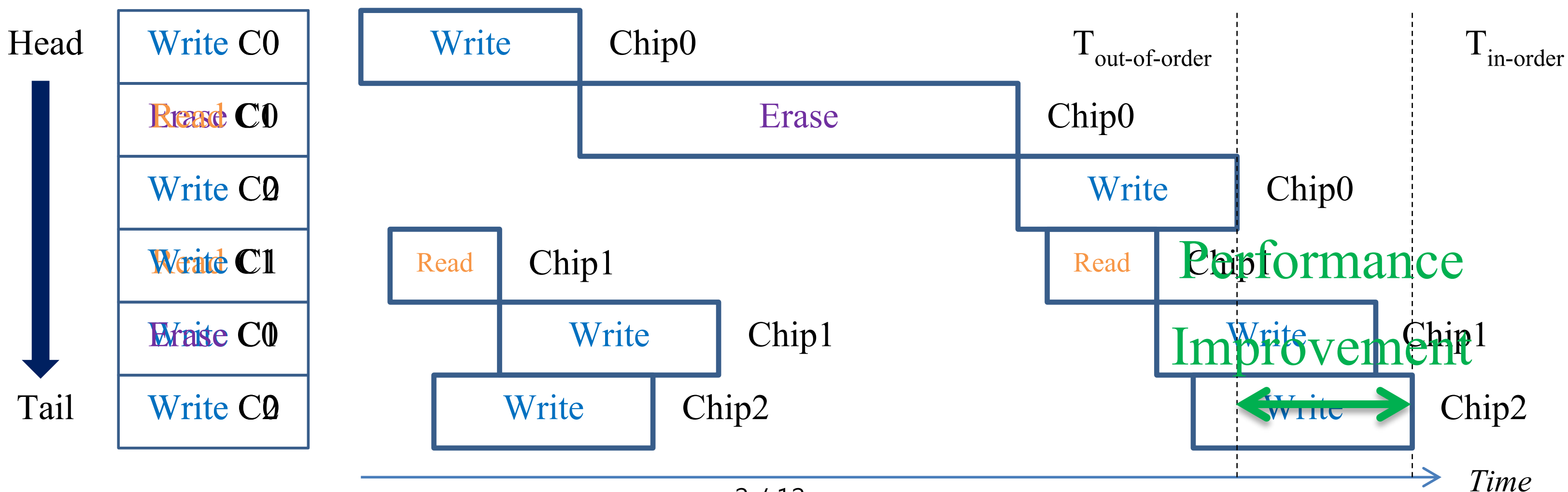
School of Computer Science and Engineering
Seoul National University

Massive Storage Systems and Technologies 2013 (MSST 2013)

May 9th, 2013

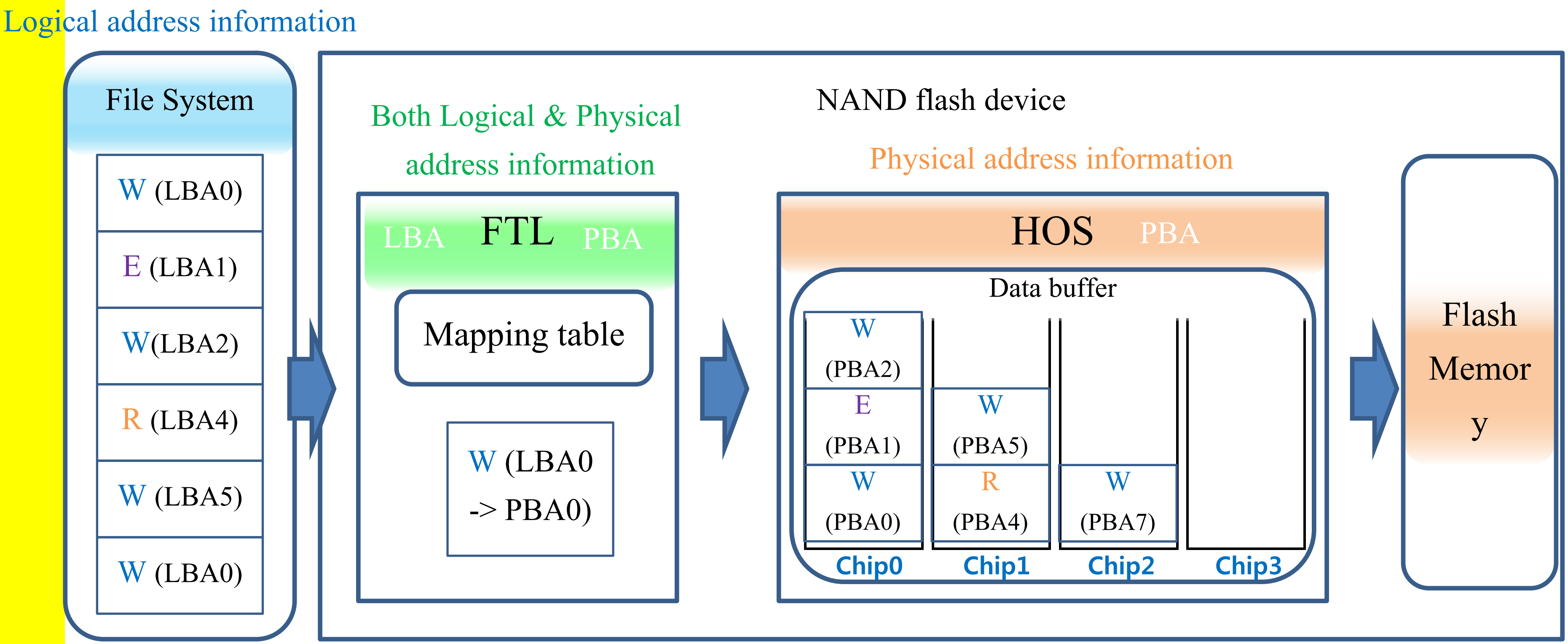
Introduction

- NAND flash memory based devices
  - Become more popular because of their performance
  - Consist of multiple flash chips
    - Each chip can perform only one flash operation at a time
- In order to increase the performance of NAND-based devices
  - Exploiting multichip parallelism is a key
  - Out-of-order execution model is ideal for multichip parallelism



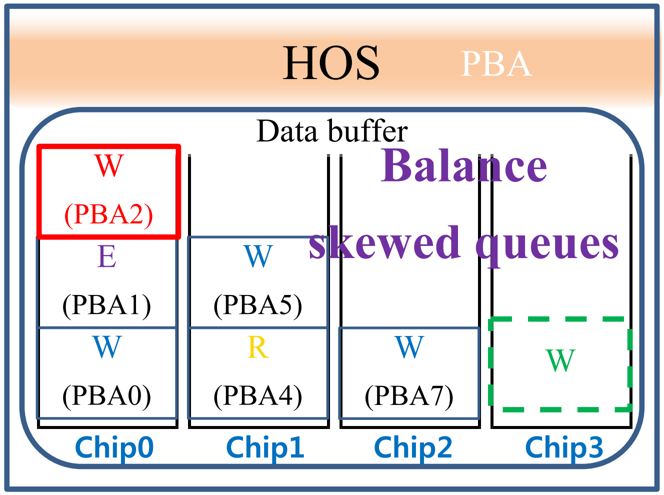
Out-of-order Support in SSDs

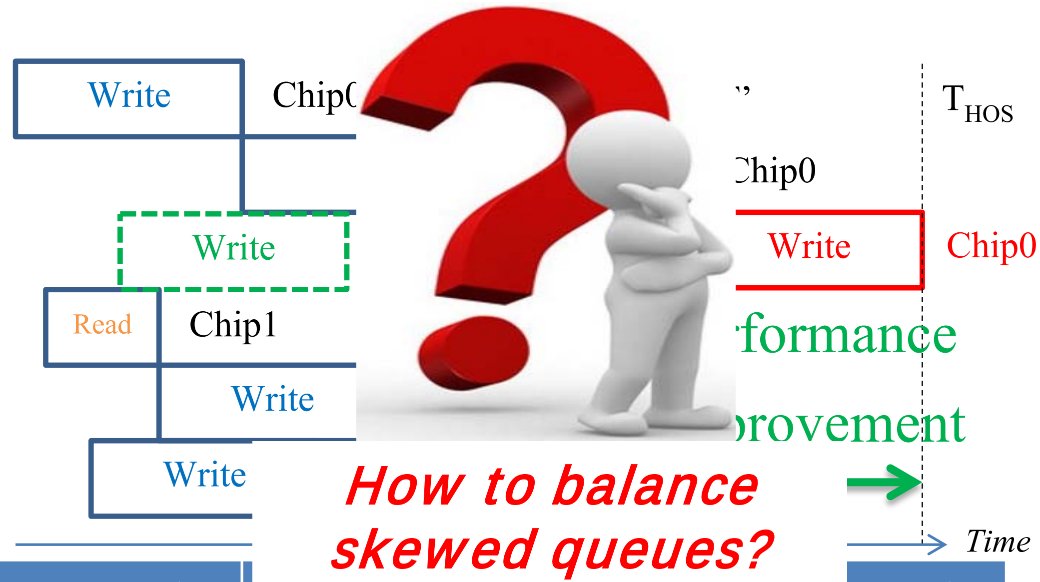
- Hardware-based Out-of-Order Scheduling (HOS)
  - Receive requests with *only physical address* information translated by a flash translation layer (FTL)
  - Execute requests in an out-of-order manner



HOS Weakness #1 : *Skewed Queue Problem*

- Data locality & different operation latencies induce the skewed queue problem





Benchmark	Bonnie++	Postmark	Financial1	Financial2	Websearch
$\frac{\#\ of\ reallocatable\ writes}{\#\ of\ total\ writes}$	29%	32%	18%	11%	9%

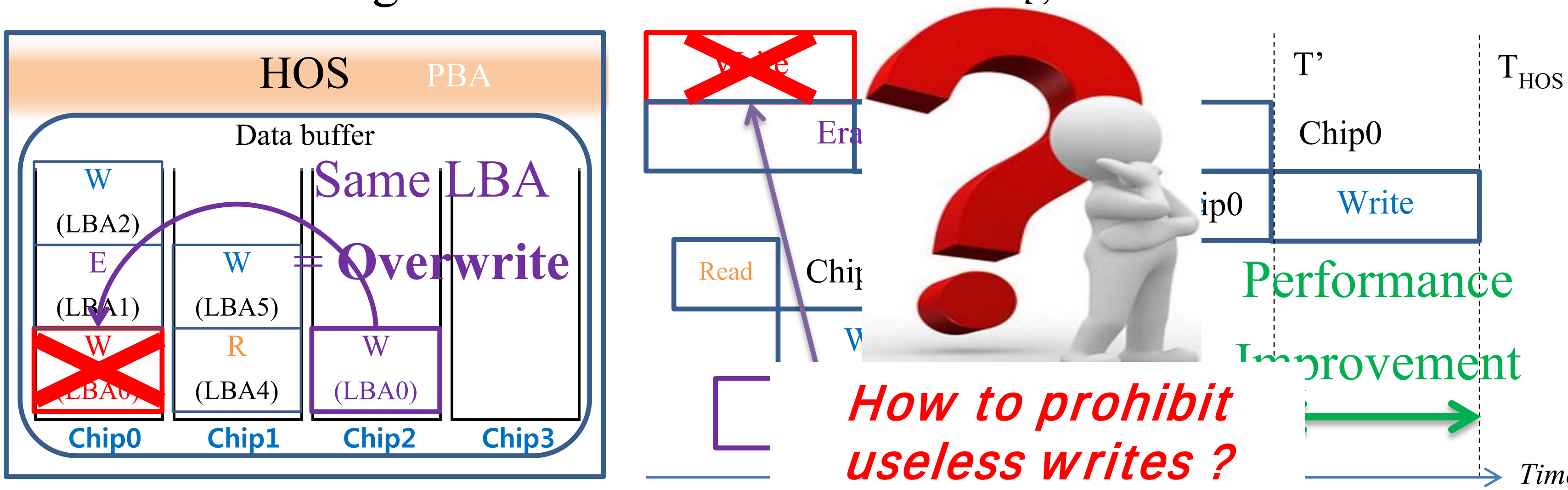
When at least one of chips is idle

- In order to *reallocate requests*, mapping table update process is inevitable

*Modifying mapping table is **hard** to hardware-based scheduler and **easy** to software-based one*

HOS Weakness #2 : *Useless Write Problem*

- *Useless Writes* means overwrites at the data buffer
- HOS can't recognize useless writes without logical address



Benchmarks	Bonnie++	Postmark	Financial1	Financial2	Websearch
# of overwrites / *# of total writes*	11.7%	14.3%	17.6%	9.2%	7.1%

- In order to *cancel useless writes*,

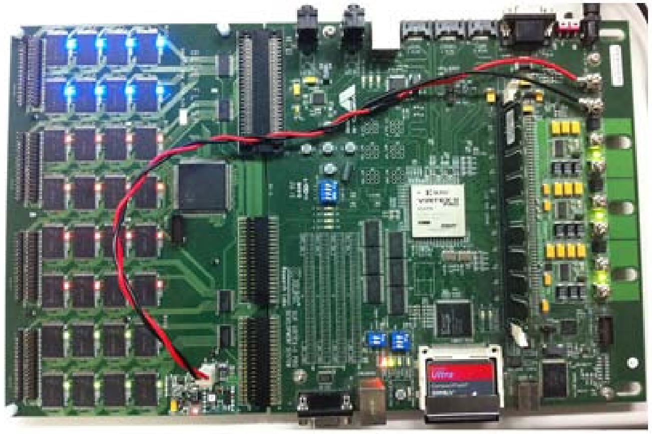
  logical address information of requests is essential

*Access logical address information of request is **hard** to hardware-based scheduler and **easy** to software-based one*

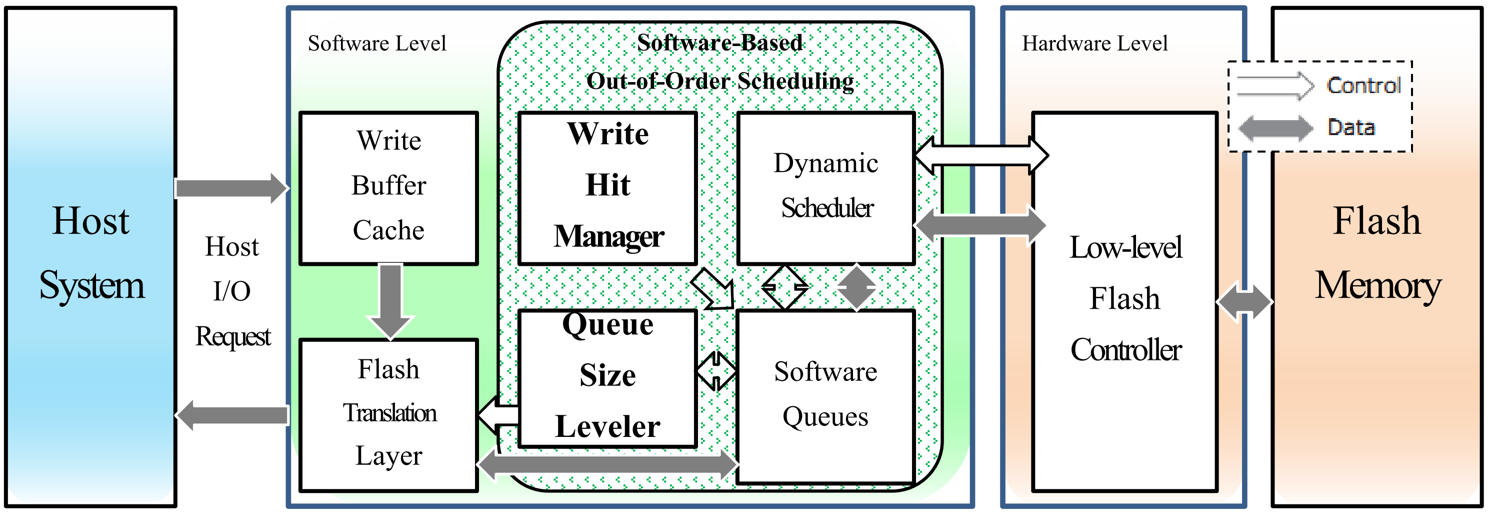
Our Contributions

- Propose software-based out-of-order scheduling (SOS)
  - SOS can overcome the skewed queue problem & useless write problem without additional hardware resources and high design cost

- SOS was implemented at a prototype SSD, BlueSSD
  - SOS improves the average I/O response time by up to *42%* over HOS



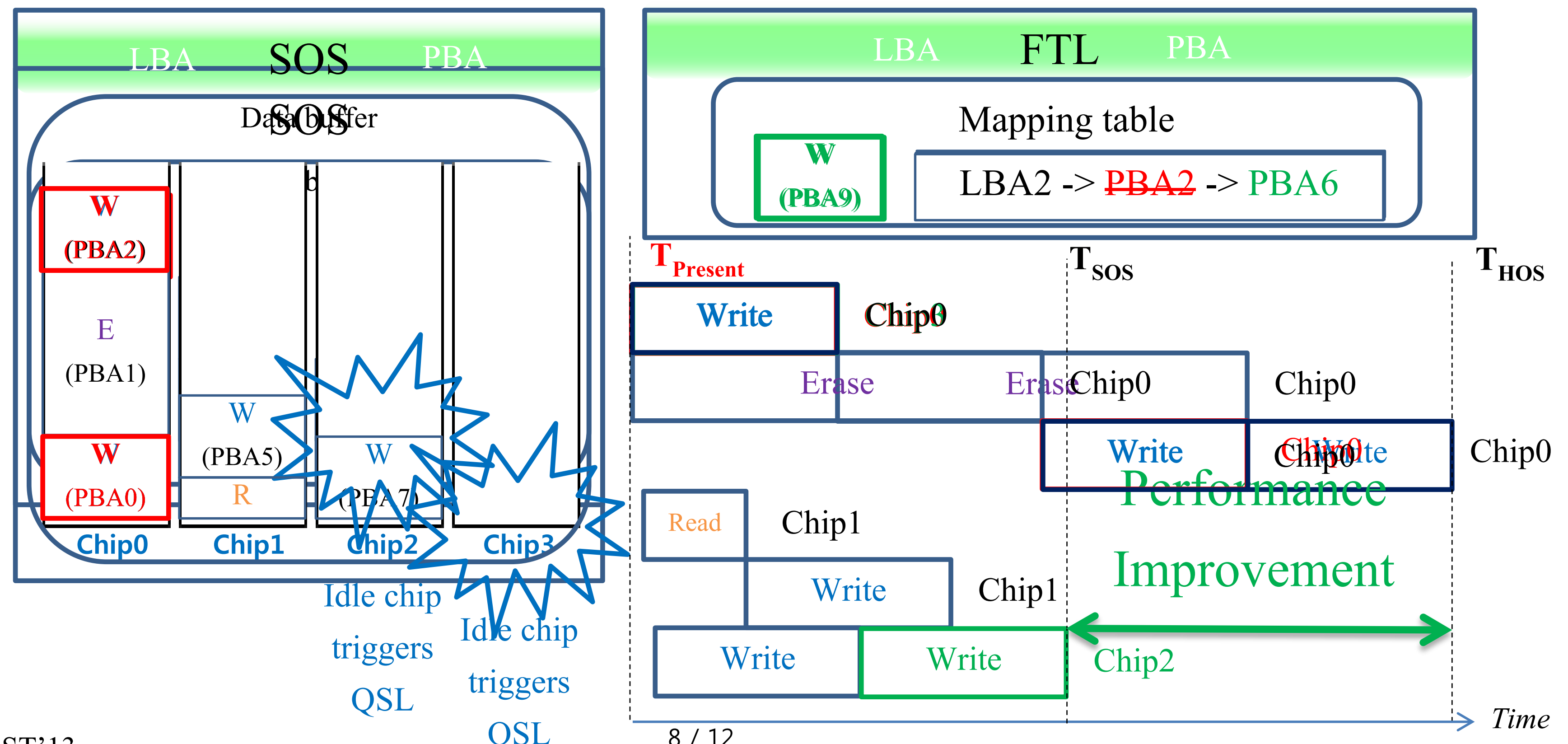
Overview of SOS



- SOS handles requests at the software queues with logical & physical address information
  - Queue size leveler : detect the skewed queues and then rearranges requests
  - Write hit manager : eliminate useless writes by canceling unnecessary writes

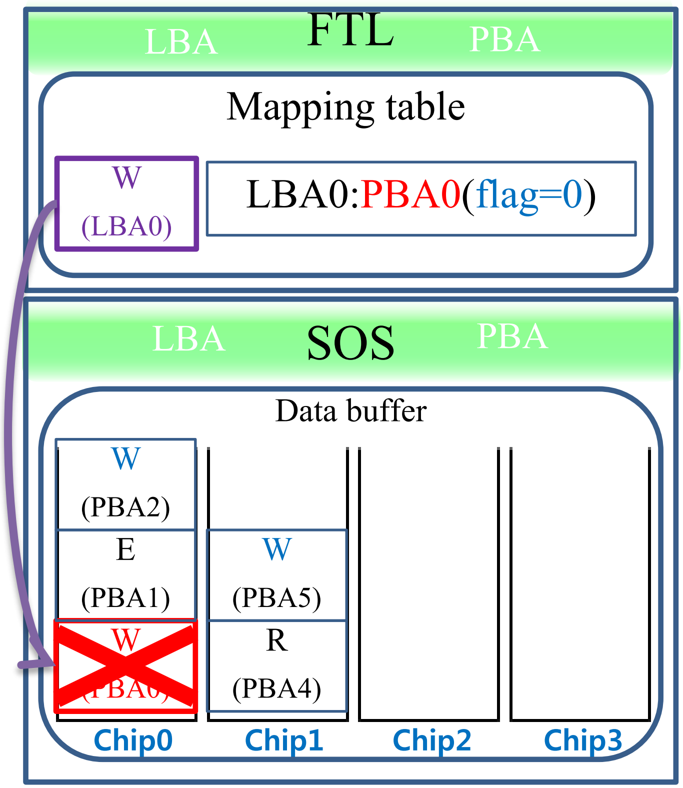
Queue Size Leveler (QSL)

- Balance the size of multiple I/O queues by reallocating write requests to idle chips
  - Consider different latencies of each flash operations
  - Triggered when one of chips become idle



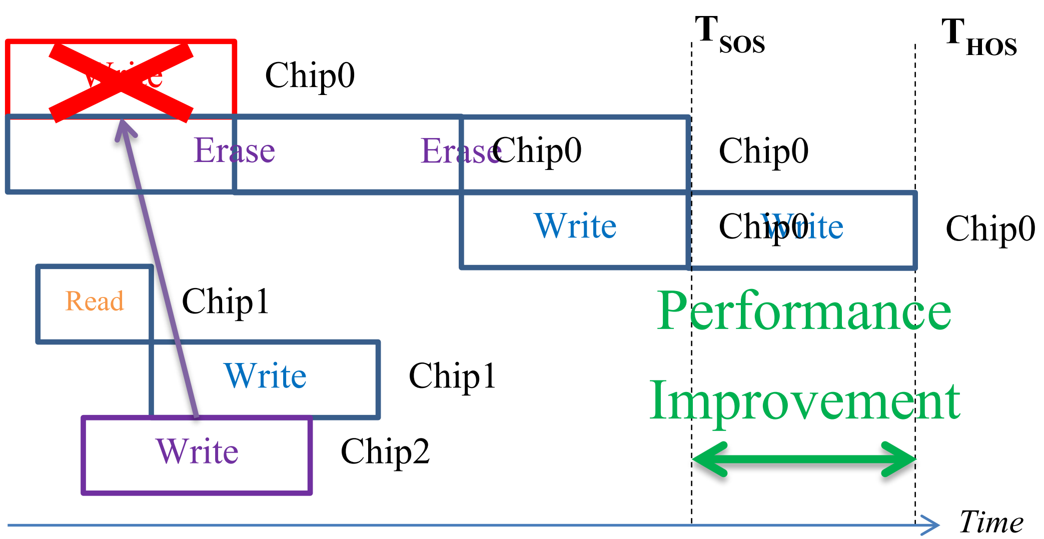
Write Hit Manager (WHM)

- Detect overwrites and cancel them
  to eliminate unnecessary writes and invalidations
  - Additional flag at mapping table implemented for detection
    - Detect useless writes without full search



Flag 0 means “Previous write request still exists at data buffer”

Overwrite occurs & it triggers WHM

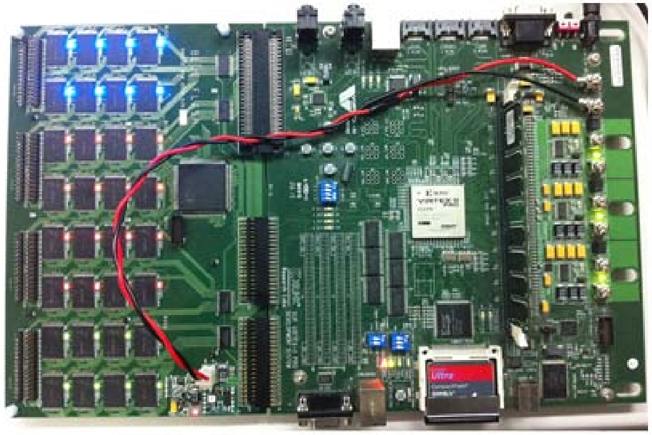


MSST’13

Experimental Settings

- We implemented the SOS in SSD prototype, BlueSSD
  - BlueSSD supports 4 buses and 4 ways (Total 16 chips)
  - PowerPC 405 processor (@100Mhz) on BlueSSD runs Linux 2.6.25.3 kernel

- Realize HOS by rearranging the sequence of requests according to the out-of-order scheduling algorithm
  - The rearranged I/O traces were replayed, using the in-order scheduling algorithm

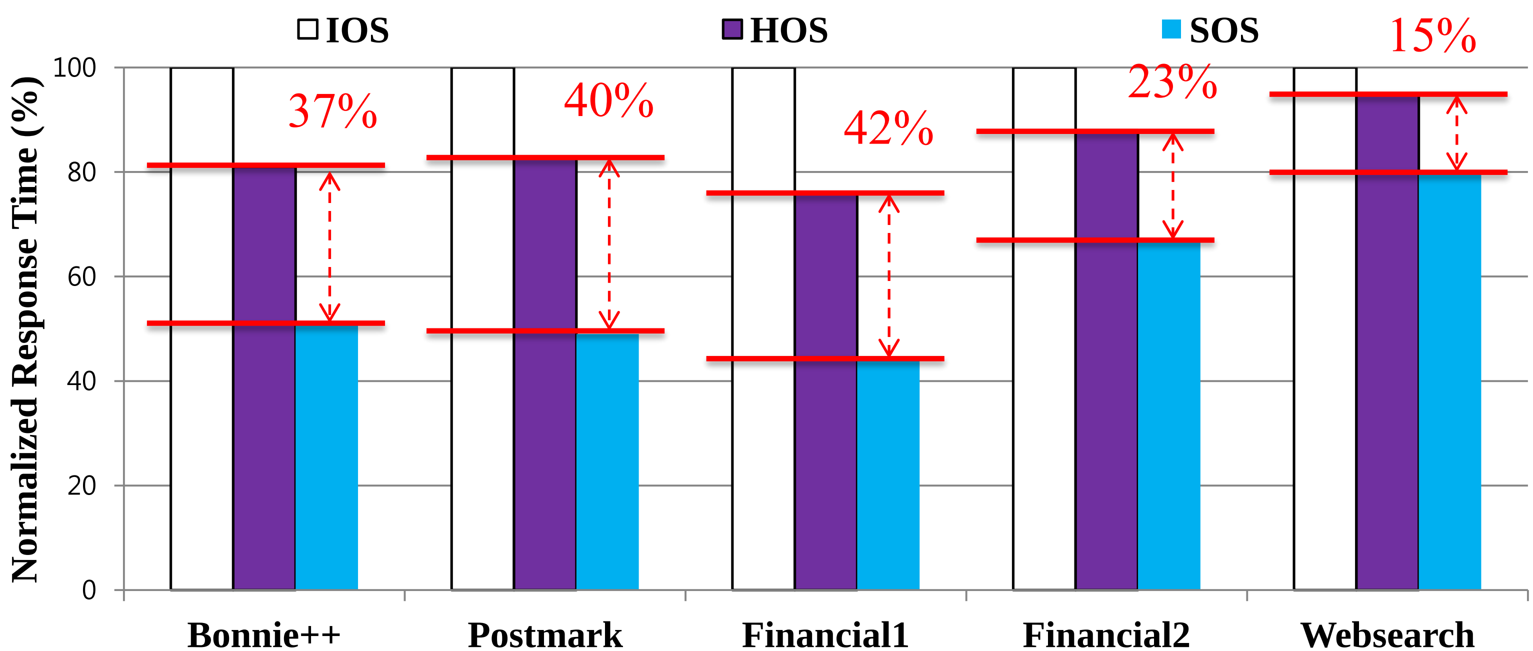


Experimental Results

- Characteristics of benchmarks

Benchmarks	Bonnie++	Postmark	Financial1	Financial2	Websearch
Read Ratio	52.1%	50.0%	32.8%	82.4%	91.1%
Write Ratio	47.9%	50.0%	67.2%	17.6%	8.9%

- SOS improves I/O response times by 15% to 42% over HOS



Conclusion & Future Work

- Software-based out-of-order scheduling
  - Exploits the multichip parallelism more effectively than hardware-based one
    - Queue size leveler addresses skewed queue problem
    - Write hit manager addresses useless write problem
  - Improves I/O response times by up to 42% over HOS

- Future work
  - More flexible request scheduling techniques
    - Reflect user-priority of requests from upper layer, etc.

End of Presentation

Thank you