Performance Impact and Interplay of SSD Parallelism through Advanced Commands, Allocation Strategy and Data Granularity

Yang Hu, Hong Jiang, Dan Feng, Lei Tian, Hao Luo, Shuping Zhang
Outline

• Introduction
• SSD Simulator
• Flash Page Size
• Allocation Scheme
• Advanced Commands
• Parallelism Inside SSDs
• Conclusion
Introduction

• Some SSD internal behaviors with potentially important impacts on the system performance and endurance

• Four features: flash page size, allocation schemes, advanced commands, priority order of SSD parallelism

• Designed and implemented a new SSD simulator, called SSDsim
SSD Simulator

• Provides the detailed and accurate simulation of each level of SSD, including hardware, FTL and buffer layer
• Provides four levels of parallelism
• Supports all the advanced commands that adhere to all the aforementioned restrictions
• It is directly validated against a real SSD prototype
SSD Simulator

- Workloads

<table>
<thead>
<tr>
<th>Workloads</th>
<th>Abb.</th>
<th>Avg. req. size read/write(KB)</th>
<th>Read(%)</th>
<th>Int. arrv. Time(ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Financial1</td>
<td>Fin1</td>
<td>2.25/3.75</td>
<td>23.2</td>
<td>8.19</td>
</tr>
<tr>
<td>Financial2</td>
<td>Fin2</td>
<td>2.3/2.9</td>
<td>82.3</td>
<td>11.08</td>
</tr>
<tr>
<td>Websearch</td>
<td>Web</td>
<td>15.15/8.6</td>
<td>99.9</td>
<td>2.99</td>
</tr>
<tr>
<td>Exchange</td>
<td>Ex</td>
<td>15.15/14.5</td>
<td>30.8</td>
<td>1179</td>
</tr>
<tr>
<td>MSN</td>
<td>MSN</td>
<td>9.6/11.1</td>
<td>67.2</td>
<td>513</td>
</tr>
<tr>
<td>Develop</td>
<td>Dev</td>
<td>18.45/10.95</td>
<td>88.6</td>
<td>1985</td>
</tr>
<tr>
<td>Radius</td>
<td>Rad</td>
<td>124.25/12.45</td>
<td>17.1</td>
<td>9475</td>
</tr>
</tbody>
</table>

Table 1. Workload characteristics of the traces
Figure 6. The average-response-time deviation of SSDsim from the prototype.
SSD Simulator

Figure 7. The Cumulative Distribution Function (CDF) of the response time.
## SSD Simulator

**Table 2. Configuration parameters used in SSDsim** (Channel-Chip-Die-Plane-Block-Page indicates the number of channels in the SSD, chips in a channel, dies in a chip, planes in a die, blocks in a plane and pages in a block, respectively. Unless otherwise noted, they are default experiment parameters)

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page read to register</td>
<td>20us</td>
</tr>
<tr>
<td>Page write from register</td>
<td>200us</td>
</tr>
<tr>
<td>Block erase</td>
<td>1.5ms</td>
</tr>
<tr>
<td>Read one byte data from register</td>
<td>25ns</td>
</tr>
<tr>
<td>Write one byte data to register</td>
<td>25ns</td>
</tr>
<tr>
<td>Channel-Chip-Die-Plane-Block-Page</td>
<td>4-4-2-2-2048-64</td>
</tr>
<tr>
<td>Page size</td>
<td>2KB</td>
</tr>
</tbody>
</table>
Flash Page Size

• Two scenarios in which a logical page is written to the flash memory
  – The logical page is written for the very first time
  – The logical page is rewritten or *updated* in the flash memory

• The update operation can lead to two types of SSD internal data movement

![Diagram](image)

*Figure 9. The covered and un-covered update operations.*
Flash Page Size

• An un-covered operation requires one more flash read operation than a covered update operation.

• Given the same average request size, the larger the page size is, the more likely it is for uncovered update operations to be induced.

• A large page-size can fetch/send more data to/from the register, allows a large read/write request to be more efficiently executed.
Figure 10. The percentage of un-covered update operations and normalized average response time as functions of flash page-size under different workloads.
Flash Page Size

• Insight
  – Enlarging the storage capacity of SSD by means of increasing the page size may not be a wise choice under some workloads
  – overlap more chips of an appropriate page size in a flash package
Allocation Scheme

• Allocation schemes are classified into two categories: dynamic and static

• Static allocation
  – Assigns a logical page to a *pre-determined* channel, package, chip, die and plane

• Dynamic allocation
  – Assigns a logical page to any free physical page of the entire SSD
  – Consider several factors

<table>
<thead>
<tr>
<th>channel</th>
<th>package</th>
<th>chip</th>
<th>die</th>
<th>plane</th>
<th>block</th>
<th>page</th>
<th>In-page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>In flash package address</td>
<td></td>
<td>Full SSD address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 3. The format of a full SSD address.*
Allocation Scheme

- Compare the performances of the static allocation and dynamic allocation schemes
- Six different static allocation schemes

(A > B > C > D means the priority order of allocating logical page. In other word, it is striping address to A first, then to B, then to C, and finally to D.)
(A > B > C > D means the priority order of allocating logical page. In other word, it is striping address to A first, then to B, then to C, and finally to D.)
Allocation Scheme

(e) Static Allocation 5 (s5)
channel > plane > die > chip

(f) Static Allocation 6 (s6)
channel > die > plane > chip

(A > B > C > D means the priority order of allocating logical page. In other word, it is striping address to A first, then to B, then to C, and finally to D.)
Figure 12. The normalized average response time when using static allocation and dynamic allocation schemes, in non-aged and aged SSDs, under the six workloads.
Allocation Scheme

• Insight
  – The static allocation scheme consistently outperforms the dynamic allocation scheme in serving read requests

  – In a aged SSD, the dynamic allocation scheme consistently outperforms of the static allocation scheme, with the only exception being the read-only workloads
Allocation Scheme

• Wear-leveling algorithms
  – Used to distribute the erase operations evenly to the entire SSD
  – Usually writes hot data to the least frequently erased blocks
  – Migrates cool data to blocks with higher erasure counts

• Insight
  – The dynamic allocation scheme outperforms the static wear-leveling performance

<table>
<thead>
<tr>
<th>Workloads</th>
<th>Static allocation</th>
<th>Dynamic allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dev</td>
<td>284.9</td>
<td>2.5</td>
</tr>
<tr>
<td>Ex</td>
<td>409.1</td>
<td>39.5</td>
</tr>
<tr>
<td>Fin</td>
<td>207.3</td>
<td>3.9</td>
</tr>
<tr>
<td>MSN</td>
<td>3534.4</td>
<td>112.6</td>
</tr>
<tr>
<td>Rad</td>
<td>7.5</td>
<td>2.6</td>
</tr>
</tbody>
</table>
Advanced Commands

• Improve the performance of SSD

• Are extensions of the basic read, program and erase commands but with some usage restrictions

• The pages must be programmed consecutively in the increasing order of page address. Random-page-address programming is prohibited -Restriction a
Advanced Commands

• Copy-back
  – Moves one page of data from one page to another in the same plane, without occupying the I/O bus
  – The source page and the target page must have the same chip, die and plane addresses
  – The addresses of the source page and destination page must be both odd or both even -Restriction b
Advanced Commands

- **Restriction a** and **Restriction b** must be adhered to.

- It is obvious that using the copy-back command blindly will lead to a waste of flash pages.

![Diagram](image)

**Figure 13.** The exemplar process of executing a copy-back command.
Advanced Commands

Figure 14. The performance comparison of different the methods of using copy-back command.
Advanced Commands

• Insight
  – The copy-back command should only be used wisely, otherwise the I/O performance and endurance of SSD can be significantly reduced
Advanced Commands

• Multi-Plane
  – Activates multiple read, program or erase operations in all planes of the same die
  – The pages executing a multi-plane read/write operation must have the same chip, die, block and page addresses – Restriction c
Advanced Commands

• Multi-Plane
  – We analyze the multi-plane command for reads and writes, which we call MPW (Multi-Plane Write) and MPR (Multi-Plane Read)
  – When using multi-plane write command, **Restriction a** and **Restriction c** must be adhered to
Advanced Commands

- Multi-Plane Write

*Figure 15. The exemplar process of executing a MPW.*
Advanced Commands

Figure 16. Performance comparison of different the methods of using MPW.
Advanced Commands

• Multi-Plane Read
  – When using MPR, **Restriction c** must be adhered to
  – We found that the performance gains are negligible under a majority of the workloads

<table>
<thead>
<tr>
<th>Table 4. The performance gain due to MPR (RS is short for response time speedup. Baseline is based on basic command alone.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic RS</td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>1.15</td>
</tr>
<tr>
<td>Static RS</td>
</tr>
</tbody>
</table>
Advanced Commands

• Insight
  – Using MPW blindly improves the I/O performance but reduces the endurance
  – The impact of the wise MPW is negligible
  – MPR cannot provide significant performance improvement
  – Workloads are read-dominant and comprise of large reads, MPR can help improve I/O performance
Advanced Commands

• Interleave
  – Executes several page read, page write, block erase in different dies of the same chip simultaneously
  – Only Restriction a must be adhered to
  – There is no endurance loss when using the interleave command

(b) Interleave write in a chip

- Command and Address transfer
- Data transfer
- Write data to target page
Advanced Commands

Figure 17. The performance comparisons of employing interleave command or not.
Advanced Commands

• Insight
  – The interleave command can help improve the I/O performance without any endurance degradation
Advanced Commands

• The combinations of the three advanced commands
  – There are two recommended approaches to using the advanced commands
    • Use the copy-back command, the MPW command, and the interleave command wisely - Approach (1)
      • Use the copy-back command wisely, the MPW command blindly, and the interleave command ubiquitously - Approach (2)
Advanced Commands

![Graph showing performance comparison](image)

**Figure 18.** The performance comparison of two approaches to using the advanced commands.
Parallelism Inside SSDs

• Each read/write operation consists of two steps
  – (1) data transfer
  – (2) reading/writing data from/to the target page to/from the data register of the plane

Four levels of parallelism

• Channel-level
• Chip-level
• Die-level
• Plane-level

• channel > die > plane > chip
Parallelism Inside SSDs

- **Evaluation of priority order of SSD parallelism under the dynamic allocation**

Table 5. Six kinds of configured SSDs. (A>B in the “Priority” field signifies that choosing a free page from A is preferred to choosing one from B. Cl.-Cp.-D.-P. indicates the numbers of channels in the SSD, chips in a channel, dies in a chip, and planes in a die, respectively. The “AC” row indicates whether advanced commands are used (Yes) or not (No)).

<table>
<thead>
<tr>
<th>SSD</th>
<th>Cl.-Cp.-D.-P.</th>
<th>AC</th>
<th>Page</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSD1</td>
<td>8-4-2-2</td>
<td>Yes</td>
<td>2KB</td>
<td>chip &gt;die&gt;plane&gt;channel</td>
</tr>
<tr>
<td>SSD2</td>
<td>8-4-2-2</td>
<td>Yes</td>
<td>2KB</td>
<td>channel&gt;chip&gt;die&gt;plane</td>
</tr>
<tr>
<td>SSD3</td>
<td>1-4-2-2</td>
<td>Yes</td>
<td>2KB</td>
<td>channel&gt;chip&gt;die&gt;plane</td>
</tr>
<tr>
<td>SSD4</td>
<td>1-4-2-2</td>
<td>Yes</td>
<td>2KB</td>
<td>channel&gt;die&gt;chip&gt;plane</td>
</tr>
<tr>
<td>SSD5</td>
<td>1-4-2-2</td>
<td>Yes</td>
<td>2KB</td>
<td>channel&gt;die&gt;plane&gt;chip</td>
</tr>
<tr>
<td>SSD6</td>
<td>1-4-2-2</td>
<td>Yes</td>
<td>2KB</td>
<td>channel&gt;plane&gt;die&gt;chip</td>
</tr>
</tbody>
</table>
Parallelism Inside SSDs

**Figure 20.** The normalized average response time of SSD1, SSD2, SSD3, SSD4, SSD5 and SSD6.
Parallelism Inside SSDs

- **Evaluation of priority order under the static allocation**
  - Use six SSDs, SSD-s1, SSD-s2, SSD-s3, SSD-s4, SSD-s5 and SSD-s6
  - Cl.-Cp.-D.-P. => 8-4-2-2

<table>
<thead>
<tr>
<th>Workload</th>
<th>Write ratio</th>
<th>Req. size</th>
<th>Interval time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syn1</td>
<td>100%</td>
<td>16KB</td>
<td>30us (75%)</td>
</tr>
<tr>
<td>Syn2</td>
<td>25%</td>
<td>16KB</td>
<td>30us (75%)</td>
</tr>
<tr>
<td>Syn3</td>
<td>100%</td>
<td>20KB</td>
<td>200us (75%)</td>
</tr>
<tr>
<td>Syn4</td>
<td>25%</td>
<td>20KB</td>
<td>200us (75%)</td>
</tr>
</tbody>
</table>
Parallelism Inside SSDs

Figure 21. The normalized average response time of SSD-s1, SSD-s2, SSD-s3, SSD-s4, SSD-s5, SSD-s6.
Conclusion

• The request size and the percentage of the uncovered update operations of the workload must be taken into considerations to choose an appropriate page size

• The static allocation is found to perform the best on read performance under all workloads

• The dynamic allocation performs the best on overall performance and endurance under the most of workloads in aged SSDs
Conclusion

• There are two recommended approaches to using the advanced commands

• The optimal priority order of parallelisms in SSD should be channel > die > plane > chip