Flash Translation Layers III

Jin-Soo Kim (jinsookim@skku.edu)
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
Contents

▪ Superblock FTL
▪ DFTL
▪ μ-FTL
Superblock FTL

Reference: Superblock FTL: A Superblock-based Flash Translation Layer with a Hybrid Address Translation Scheme
Superblock FTL

- A superblock shares log blocks
- Up to M log blocks per superblock
- Block mapping at the superblock level, Page mapping within a superblock
- Hot/cold pages separation
- Map cache
- The amount of mapping information increased
Example

- \( W = \langle \{1,2\}, \{8\}, \{1,2\}, \{12\}, \{13\}, \{9\} \rangle \)
  - Write (\{1,2\}, AB)
  - Write (\{8\}, C)
  - Write (\{1,2\}, DE)
  - Write (\{12\}, F)
  - Write (\{13\}, G)
  - Write (\{9\}, H)
Mapping Information

- PGD (Page Global Directory)
- PMD (Page Middle Directory)
- PTE (Page Table Entry)
## Comparison

<table>
<thead>
<tr>
<th></th>
<th>ANAND</th>
<th>Log block scheme</th>
<th>FAST</th>
<th>Superblock FTL</th>
<th>Page mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data blocks</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Terminology</td>
<td></td>
<td>Data blocks</td>
<td>Data blocks</td>
<td>Data blocks</td>
<td>Data blocks</td>
</tr>
<tr>
<td>Management scheme</td>
<td>In-order</td>
<td>In-order</td>
<td>In-order</td>
<td>Out-of-order</td>
<td>Out-of-order</td>
</tr>
<tr>
<td>The degree of sharing</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Min(P, S)</td>
<td>P</td>
</tr>
<tr>
<td><strong>Update blocks</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Terminology</td>
<td></td>
<td>Replacement blocks</td>
<td>Log blocks</td>
<td>Sequential / random log blocks</td>
<td>U-blocks</td>
</tr>
<tr>
<td>Management scheme</td>
<td>In-order</td>
<td>Out-of-order</td>
<td>Out-of-order</td>
<td>Out-of-order</td>
<td>Out-of-order</td>
</tr>
<tr>
<td>The degree of sharing</td>
<td>1</td>
<td>1</td>
<td>1 (sequential) or P (random)</td>
<td>Min(P, S)</td>
<td>P</td>
</tr>
</tbody>
</table>
DFTL

- Page mapping
- Keeps full mapping on flash to reduce SRAM use
- Demand-based selective caching of page-level address mappings in SRAM
- Exploits temporal locality in workloads
- Data pages vs. Translation pages
DFTL Architecture

- **Cached Mapping Table**: Store active address mappings.
- **Global Translation Directory**: Store logical to physical address translations.
- **Flash**: Fetch mapping entry.
- **SRAM**: Evict mapping entry for synchronization.
- **Translation Blocks**: Consult location of translation pages on flash.
- **Data Blocks**: Tracks translation pages on flash.
- **Data OOB**: Store real data from I/O requests.
DFTL Example

- Read request to $D_{LPN} \ 1280$

![Diagram showing DFTL example with steps 1 to 11]
Garbage Collection

- Current data block: updated data pages
- Current translation block: updated translation pages

- GC invoked if the number of free blocks < GC_threshold
- Selecting a victim block: simple cost-benefit policy
μ-FTL

Reference: μ-FTL: A Memory-efficient Flash Translation Layer Supporting Multiple Mapping Granularities
μ-FTL

- Minimally-updated FTL
- Supports multiple mapping granularities
  - Based on extents
  - Reduce the amount of mapping information
- Requires more sophisticated index structure
  - μ-Tree is used to store the mapping info.
- Tunable memory footprint
  - Frequently accessed mapping info. cached
Example

- $W = \langle \{1\}, \{2\}, \{8\}, \{1\}, \{2\}, \{12,13\}, \{9\} \rangle$
  - write ($\{1\}$, A)
  - write ($\{2\}$, B)
  - write ($\{8\}$, C)
  - write ($\{1\}$, D)
  - write ($\{2\}$, E)
  - write ($\{12,13\}$, FG)
  - write ($\{9\}$, H)
μ-FTL Architecture

- **μ-Tree Cache**
- **Bitmap update flush**
- **Bitmap Cache**
- **Cache full**
- **Extent update**
- **Data**
- **Cache miss**
- **Data Extent update**
- **Partition 1**
- **Partition 2**
- **Partition 3**
- **Partition 4**
- **Free block list**
Conclusion

- Garbage collection issues
- Mapping information management issues
- Platform-dependent issues
CAFTL (Content-Aware FTL)

Reference: CAFTL - A Content-Aware FTL Enhancing the Lifespan of Flash Memory based Solid State Drives (2011)
CAFTL

- Deduplication
  - Reduce write traffic to flash memory by removing unnecessary duplicate writes

![Bar chart showing the percentage of redundant data in disks.](image)

**Figure 1:** The percentage of redundant data in disks.
Identifying duplicate blocks

- **Fingerprint**
  - Byte-by-byte comparison is slow
  - Identify duplicate blocks by comparing SHA-1 hash values

- **Fingerprint Store**
  - Store and search in the most likely-to-be-duplicated fingerprints
Architecture

Figure 3: An illustration of CAFTL architecture.
Two-level indirect mapping

Mapping Table

<table>
<thead>
<tr>
<th>LPN 0</th>
<th>LPN 1</th>
<th>LPN 2</th>
<th>LPN 3</th>
<th>LPN 4</th>
<th>LPN 5</th>
<th>LPN 6</th>
<th>LPN 7</th>
<th>LPN 8</th>
<th>LPN 9</th>
<th>LPN 10</th>
<th>LPN 11</th>
<th>LPN 12</th>
<th>LPN 13</th>
<th>LPN 14</th>
<th>LPN 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>F</td>
<td>T</td>
<td>I</td>
<td>L</td>
<td>L</td>
<td>C</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>E</td>
<td>H</td>
<td>W</td>
<td>R</td>
<td>B</td>
<td>Z</td>
</tr>
</tbody>
</table>

Flash Memory

Conventional FTL

Primary Map

<table>
<thead>
<tr>
<th>LPN 0</th>
<th>LPN 1</th>
<th>LPN 2</th>
<th>LPN 3</th>
<th>LPN 4</th>
<th>LPN 5</th>
<th>LPN 6</th>
<th>LPN 7</th>
<th>LPN 8</th>
<th>LPN 9</th>
<th>LPN 10</th>
<th>LPN 11</th>
<th>LPN 12</th>
<th>LPN 13</th>
<th>LPN 14</th>
<th>LPN 15</th>
</tr>
</thead>
</table>

Secondary Map

<table>
<thead>
<tr>
<th>VPN 0</th>
<th>VPN 1</th>
<th>VPN 2</th>
<th>VPN 3</th>
<th>VPN 4</th>
<th>VPN 5</th>
</tr>
</thead>
</table>

Flash Memory

CAFTL

<table>
<thead>
<tr>
<th>F</th>
<th>A</th>
<th>E</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>Z</td>
<td>L</td>
<td>R</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>I</td>
<td>W</td>
</tr>
</tbody>
</table>
Example (1)

\[ W = \langle \{1\}, \{2\}, \{0\}, \{11\}, \{4\}, \{7\} \rangle \]

- write (\{1\}, F)
- write (\{2\}, F)
- write (\{0\}, A)
- write (\{11\}, E)
Example (2)

- \( W = \langle \{1\}, \{2\}, \{0\}, \{11\}, \{4\}, \{7\} \rangle \)
  - write (\{1\}, F)
  - write (\{2\}, F)
  - write (\{0\}, A)
  - write (\{11\}, E)
  - write (\{4\}, T)
Example (3)

- \( W = \langle \{1\}, \{2\}, \{0\}, \{11\}, \{4\}, \{7\} \rangle \)
  - write (\(\{1\}\), F)
  - write (\(\{2\}\), F)
  - write (\(\{0\}\), A)
  - write (\(\{11\}\), E)
  - write (\(\{4\}\), T)
  - write (\(\{7\}\), T)
Performance Evaluation

Figure 8: Perc. of removed duplicate writes.

Figure 9: Perc. of extended flash space.