ICE3028: Embedded Systems Design
- Project 2: Page Mapping FTL on NAND simulator

Prof. Jinkyu Jeong (jinkyu@skku.edu)
TA -- Minwoo Ahn (minwoo.ahn@csl.skku.edu)
TA -- Donghyun Kim (donghyun.kim@csl.skku.edu)

Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
Descriptions

• Develop a page mapping FTL simulator
  – Simulate the operations of page mapping FTL
    • Manage mapping with L2P table
  – Assumption
    • Uniform size of write request: 4KB
    • Initial state of flash blocks: empty
  – Misc.
    • GC policy: greedy policy
    • GC triggering condition: when # of free blocks becomes one
    • Make Over-Provisioning (OP) ratio adjustable
  – Draw a graph on OP ratio vs. WAF
Data Structures

• L2P table
  – Index: logical page number (LPN)
  – Value: physical page number (PPN)
  – Updated on write and GC

• Store the LPN into the spare area of written page
  – Why?

• Per-block information
  – Which page is valid? (optional)
  – # of valid pages (optional)

• How to find the victim block for GC?
Configurations

- **Flash memory**
  - Total capacity: 4GB
  - Page size: 4KB
  - Pages per block: 128 (block size: 512KB)

- **SSD**
  - OP ratio: 7% (default), 10%, 13%, 16%, 19%, 22%, 25%, 28%

- **Workload**
  - Size of each write request: 4KB
  - Total number of write requests: x10 of the total SSD capacity visible to the user
Configurations (cont’d)

Total 4GB
\( (N_{\text{BLOCKS}} = 8192 \text{ blocks, } 512\text{KB/block}) \)

User-visible space
\( (N_{\text{USER BLOCKS}} = 7656 \text{ blocks}) \)

Over-provisioning space
\( (N_{\text{OP BLOCKS}} = 536 \text{ blocks, } 7\%) \)

PPN:
Block #  Page #
Code Analysis

```c
#define SSD_SHIFT 32
#define PAGE_SHIFT 12
#define PAGES_PER_BLOCK_SHIFT 7
#define OP_RATIO 7
#define N_RUNS 10
#define PPN_SHIFT (SSD_SHIFT - PAGE_SHIFT)
#define BLOCKS_SHIFT (PPN_SHIFT - PAGES_PER_BLOCK_SHIFT)
#define N_PAGE_SIZE (1 << PAGE_SHIFT)
#define N_PAGES_PER_BLOCK (1 << PAGES_PER_BLOCK_SHIFT)
#define N_PPNS (1 << PPN_SHIFT)
#define N_BLOCKS (1 << BLOCKS_SHIFT)
#define N_USER_BLOCKS (N_BLOCKS * 100 / (100 + OP_RATIO))
#define N_OP_BLOCKS (N_BLOCKS - N_USER_BLOCKS)
#define N_LPNS (N_USER_BLOCKS * N_PAGES_PER_BLOCK)
#define LPN_COUNTS (N_LPNS * N_RUNS)
```
## Code Analysis (cont’d)

### Execution result

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSD capacity</td>
<td>4GB</td>
</tr>
<tr>
<td>Page size</td>
<td>4KB</td>
</tr>
<tr>
<td>Pages / Block</td>
<td>128 pages</td>
</tr>
<tr>
<td>Block size</td>
<td>512KB</td>
</tr>
<tr>
<td>OP ratio</td>
<td>7%</td>
</tr>
<tr>
<td>Physical Blocks</td>
<td>8K (8192)</td>
</tr>
<tr>
<td>User Blocks</td>
<td>7656</td>
</tr>
<tr>
<td>OP Blocks</td>
<td>536</td>
</tr>
<tr>
<td>PPNs</td>
<td>1M (1048576)</td>
</tr>
<tr>
<td>LPNs</td>
<td>979968</td>
</tr>
<tr>
<td>Total runs</td>
<td>x10</td>
</tr>
<tr>
<td>Actual capacity</td>
<td>4013948928 Bytes</td>
</tr>
</tbody>
</table>

### Results

- **[Run 1]** host 979968, valid page copy 0, GC# 0, WAF=1.00
- **[Run 2]** host 1959936, valid page copy 0, GC# 0, WAF=1.00
- **[Run 3]** host 2939904, valid page copy 0, GC# 0, WAF=1.00
- **[Run 4]** host 3919872, valid page copy 0, GC# 0, WAF=1.00
- **[Run 5]** host 4899840, valid page copy 0, GC# 0, WAF=1.00
- **[Run 6]** host 5879808, valid page copy 0, GC# 0, WAF=1.00
- **[Run 7]** host 6859776, valid page copy 0, GC# 0, WAF=1.00
- **[Run 8]** host 7839744, valid page copy 0, GC# 0, WAF=1.00
- **[Run 9]** host 8819712, valid page copy 0, GC# 0, WAF=1.00
- **[Run 10]** host 9799680, valid page copy 0, GC# 0, WAF=1.00

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host writes</td>
<td>9799680</td>
</tr>
<tr>
<td>GC writes</td>
<td>0</td>
</tr>
<tr>
<td>Number of GCs</td>
<td>0</td>
</tr>
<tr>
<td>Valid pages per GC</td>
<td>-nan pages</td>
</tr>
<tr>
<td>WAF</td>
<td>1.00</td>
</tr>
</tbody>
</table>
pm.c

• ftl_read(long lpn, u32 *read_buffer)
• ftl_write(long lpn, u32 *write_buffer)
• garbage_collection()

• Write above APIs based on NAND simulator written at Project 1
Submission

• Compress your folder as YourStudentID-2.tar.gz
  – With codes and graph
  – Send email to minwoo.ahn@csl.skku.edu
  – Please send mail with uniformized title
    • [ICE3028]YourStudentID-2

• PLEASE DO NOT COPY
  – YOU WILL GET F GRADE IF YOU COPIED

• Due date: 11/1 (Wed.), 23:59:59 PM
  – -20% per day for delayed submission
Questions

• If you have questions, please email to TAs

• You can also visit Semiconductor Building #400509
  – Please send email before visiting