Project #2: Page Mapping FTL on NAND simulator

Dong-Yun Lee(dongyun.lee@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

- **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)

User-visible space
(N_USER_BLOCKS = 7656 blocks)

Over-provisioning space
(N_OP_BLOCKS = 536 blocks, 7%)

Total 4GB
(N_BLOCKS = 8192 blocks, 512KB/block)

PPN:
Block #  Page #
Dive to the Code

```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)
```
Dive to the Code (cont’d)

SSD capacity: 4GB
Page size: 4KB
Pages / Block: 128 pages
Block size: 512KB
OP ratio: 7%
Physical Blocks: 8K (8192)
User Blocks: 7656
OP Blocks: 536
PPNs: 1M (1048576)
LPNs: 979968
Total runs: x10
Actual capacity: 4013948928 Bytes

[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

Results ------
Host writes: 9799680
GC writes: 0
Number of GCs: 0
Valid pages per GC: -nan pages
WAF: 1.00
jinsoo@ubuntu:~$


Miscellaneous

▪ Skeleton code is available at course homepage
  • Based on a NAND simulator written at Project 1 that you made
  • Linux is strongly recommended for coding environment

▪ Submission
  • Hand in your work via e-mail (dongyun.lee@csl.skku.edu)
  • Attach two files
    - [Student ID].tar.gz (e.g. 2010310166.tar.gz)
    - [Student ID].pdf (graph file)
  • Your submission status will also be noticed at the homepage

▪ Due : 4/18 or 23 23:59:59
Any Questions?