FTL Testing

Prof. Dongkun Shin (dongkun@skku.edu)
TA – Junho Lee (crow6316@skku.edu)
TA – Somm Kim (sommkim@skku.edu)
Embedded Software Laboratory
Sungkyunkwan University
http://nyx.skku.ac.kr

Contents

- FTL Testing
  - With SATA interface
    - Windows: IOmeter
    - Linux: FIO
  - Without SATA interface
    - ftl_test (Emulation)

- Project 2: DFTL Porting on Jasmine

FTL Testing

FTL Testing: ftl_test

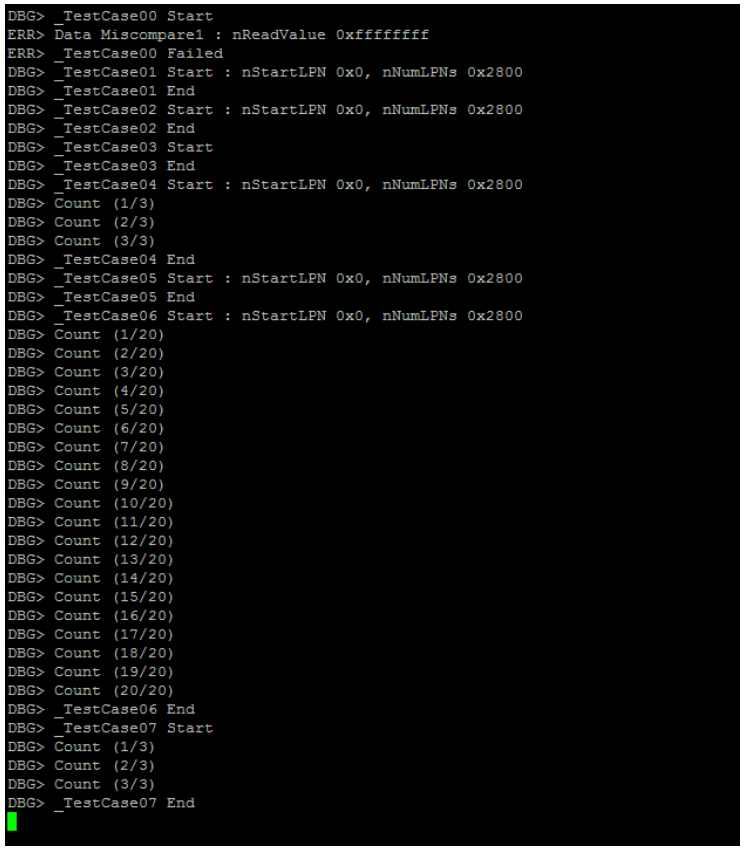
- FTL verification without using SATA interface
- Emulate the command inside the SSD.
- Test 8 I/O sub-tests right after *ftl_open*
- Generate *ftl_write* and *ftl_read* internally
  - Not from host
  - Almost same as ftl_sim.c in past lab.

FTL Testing: ftl_test

- How to?
  - Download ftl_test.zip at course homepage
  - Unzip it into *ftl_xxx* folder
  - Add *ftl_test.c* to '*SRCS=*' in *build_gnu/Makefile*
  - Edit *include/jasmine.h*
    - Set OPTION_FTL_TEST and OPTION_UART_DEBUG
  - Edit *ftl_xxx/ftl.h*
    - #define DRAM_BYTES_OTHER (… + FTL_TEST_BYTES)
    - #define FTL_TEST_ADDR (…previous define value…)
    - #define FTL_TEST_BYTES (4*1024*1024)

FTL Testing: ftl_test

- Result will be shown on PuTTY
- Eight I/O patterns are generated
  - 0th : Specific behaviors
  - 1st : [SW-SR]s
  - 2nd : SWs-SRs
  - 3rd : Similar with 1st one
  - 4th : Similar with 3rd one
  - 5th : SWs-RWs-Rs
  - 6th : Partial page RWs-RRw (hard)
  - 7th : Similar with 5th



- Try it with Greedy FTL
- You can also add other workloads by creating them.

Project II
DFTL Porting on Jasmine

Project II : DFTL Porting on Jasmine

- Porting the DFTL simulator (Project 1) on Jasmine Board
  - Store page mapping table on flash memory.
  - Dynamically load/unload map page into DRAM.
  - GTD is managed in DRAM.

Configurations

- FTL Configure
  - Configure to use 1 plane mode (include/jasmine.h)

```
#define OPTION_2_PLANE              0   // 1 = 2-plane mode, 0 = 1-plane mode
```

  - Do not modify # of Bank, # of Block, # of Page, etc.
- DFTL Configure
  - # of Map Blocks (per bank)

```
#define MAP_ENTRY_SIZE          sizeof(UINT32)
#define N_MAP_ENTRIES_PER_PAGE  ((SECTORS_PER_PAGE * BYTES_PER_SECTOR) / MAP_ENTRY_SIZE)
#define N_MAP_PAGES_PB          (((NUM_LPAGES / NUM_BANKS) + N_MAP_ENTRIES_PER_PAGE) / N_MAP_ENTRIES_PER_PAGE)
#define TMP_MAP_BLOCKS_PB       ((N_MAP_PAGES_PB + PAGES_PER_BLK) / PAGES_PER_BLK)

#define MAP_OP_RATIO            (7)
#define REMAINING_BLOCKS        (VBLKS_PER_BANK - 4)    // except block #0, #1, #2, #3
#define N_MAP_OP_BLOCKS_PB      (REMAINING_BLOCKS * MAP_OP_RATIO / 100)
#define N_MAP_BLOCKS_PB         (TMP_MAP_BLOCKS_PB + N_MAP_OP_BLOCKS_PB)
```

  - # of Cached Map page (per bank)

```
#define CMT_RATIO               (10)
#define N_CACHED_MAP_PAGE_PB    ((N_MAP_PAGES_PB * 100) / (100 + (100 - CMT_RATIO)))
```

    - Manage dirty bit separately.

Description

- Block layout
  - block #0: scan list, firmware binary image, etc.
  - block #1: FTL misc. metadata
    - If more than 1 block is needed, use next block
  - block #2: initial gc block for user block
    - Perform GC when free block is 1 in the total user blocks
  - block #3: initial gc block for map block
    - Perform GC when free block is 1 in the total map blocks
  - block #4~: user & map data blocks
    - Total number of map blocks: *N_MAP_BLOCKS_PB + 1* (gc block)
    - Total number of user blocks: *REMAINING_BLOCKS – N_MAP_BLOCKS_PB + 1* (gc block)

- Do not use the page mapping table (*PAGE_MAP_ADDR*) used in Greedy FTL.
  - Do not use the associated function.
  - You must initialize the page mapping table in the NAND.

Description

- Garbage Collection
  - GC policy: Greedy
  - User block GC
    - Select victim block
    - Copy valid pages
    - Update Corresponding translation page (NAND)
    - Update GTD
    - Update CMT (If the map page is present)
  - Map block GC
    - Select victim block
    - Copy valid pages
    - Update GTD

- CMT hit, miss counting
  - When no page read or write operation required → Cache hit
  - Other → Cache miss

Description

- Note
  - Spare area management
    - Manage map page number list of current block for spare area management.
      - Store map page number list of current block to last page of the block.
      - See *g_misc_meta[bank].lpn_list_of_cur_vblock*.
  - Flash wrapper function
    - Do not use *xxx_from_host()*, *xxx_to_host()* if it only changes to DRAM.
    - Use *RETURN_ON_ISSUE*, *RETURN_WHEN_DONE* according to the operation.
  - SRAM data management
    - Add the data to the *ftl_statistics* and *misc_metadata* structures.
      - ex) *cache_hit*, *cache_miss*, ...

Miscellaneous

- Recommended environment : Windows
- Team Project
- You should submit a report
  - Describe your code in detail.
  - Analyze the performance vs. CMT ratio (Draw a graph)
  - Analyze the performance vs. # of Map OP blocks (Draw a graph)
    - Configure MAP_OP_RATIO
  - Measure the performance with an IOmeter
    - Direct connect to SATA, not through a USB
    - Sequential Write 16KB, Sequential Read 16KB, Random Write 16KB, Random Read 16KB
- Submit to the icampus
  - Due: 11/27(Wed.) 23:59:59
  - File to submit: ftl_dftl (folder)
    - ftl.h, ftl.c, report.pdf
  - File name: team_$NUMBER.zip (ex. team_01.zip)
- Note: Project II has a large percentage of score, unlike Lab!

Any Questions?