DRAM-less FTL II

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

- DRAM-less FTL
  - Remove L2P Table in DRAM
- Multi-stream Support FTL

DRAM-less FTL

DRAM-less FTL

- Eliminate Page Mapping Table in DRAM
  - Delete the DRAM area for L2P Table in *ftl.h*

- L2P tables are now permanently preserved.
  - *logging_pmap_table()* and *load_pmap_table()* functions are no longer needed.
  - Initialize L2P table in NAND in *format()* of *ftl.c*

```
//----------------------------------------
// initialize SRAM metadata
//----------------------------------------
init_metadata_sram();

// flush metadata to NAND
format_logging_pmap_table();
logging_misc_metadata();

write_format_mark();
led(1);
uart_print("format complete");
```

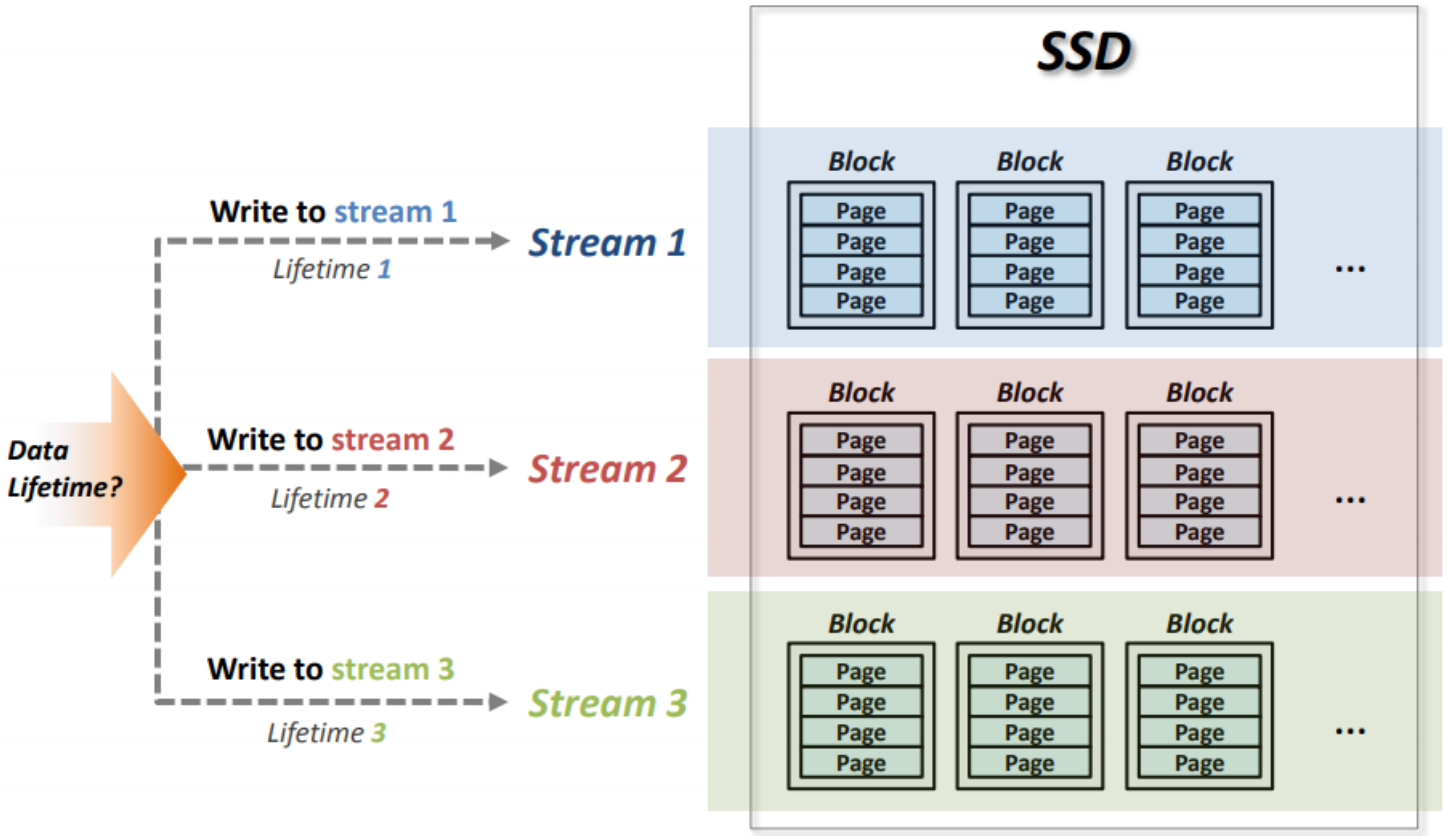
Multi-stream Support FTL

Lab 6 : Multi-stream Support FTL Simulator

- Develop an Multi-stream Support FTL simulator
  - Based on page mapping FTL (Lab5)
  - Host issues additional "stream ID" to help GC
    - Before: <start lba, # of sectors, write buffer)
    - After: <stream ID, start lba, # of sectors, write buffer>
  - Assumption
    - Sector size: 4B
    - Page size: 32B(data area) + 4B(spare area)
    - Initial state of flash blocks: empty
    - Host requests a continuous sector belonging to the same stream

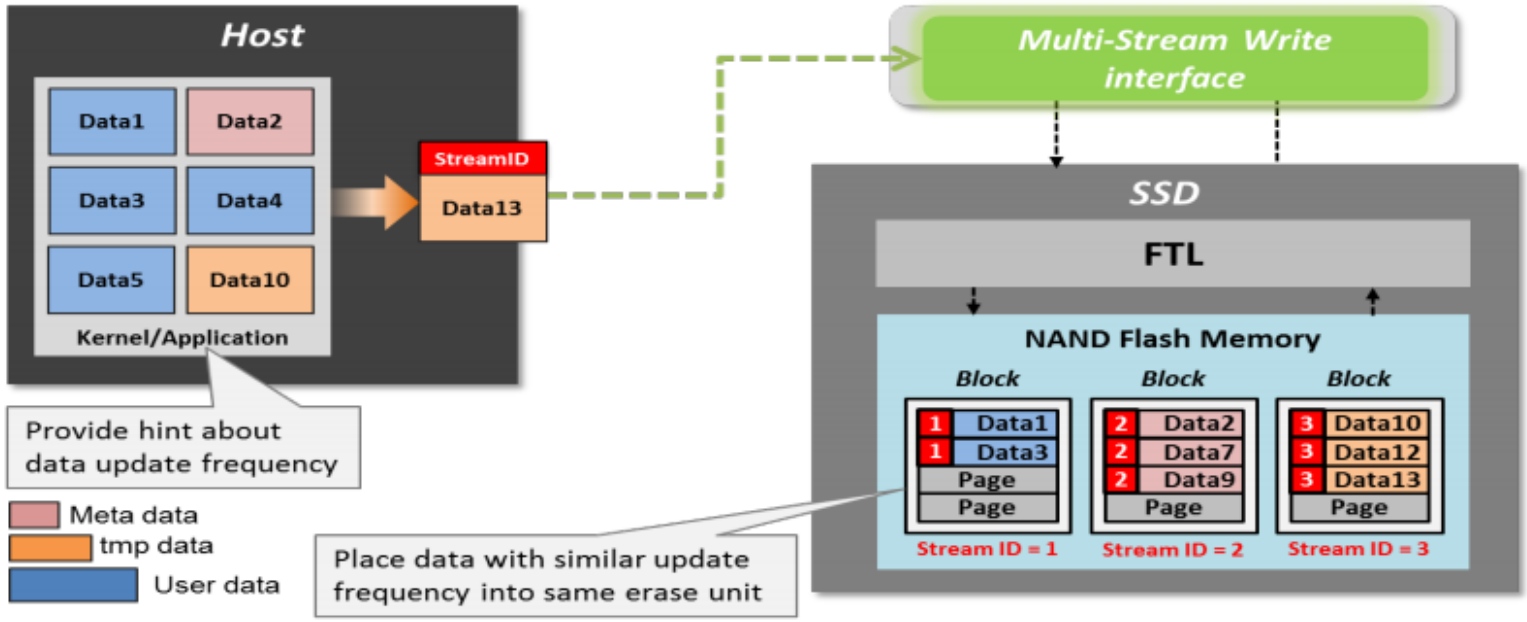
Multi-streamed SSD

- Stream Management
  - FTL manages streams with different physical blocks



Multi-streamed SSD

- Stream Management
  - Mapping data with different update frequency to different streams



https://www.flashmemorysummit.com/English/Collaterals/Proceedings/2016/20160809_FC12_Choi.pdf

ftl.c

- ftl_open()
- ftl_read(u32 lba, u32 num_sectors, u32 *read_buffer)
  - *lba*: start sector #
  - *num_sectors*: # of sectors
- ftl_write(u32 streamid, u32 lba, u32 num_sectors, u32 *write_buffer)
  - *streamid*: stream ID
  - *lba*: start sector #
  - *num_sectors*: # of sectors
  - If the stream ID is different, it should be written in a different block
  - You should add *s.ftl_write* for every *nand_write* call
- garbage_collection(u32 bank, u32 streamid)
  - GC should be performed per stream ID
  - GC policy: Greedy
  - You should add *s.gc_write* for every *nand_write* call

Miscellaneous

- Recommended environment : Linux (Ubuntu is ok!)
  - You can do it in Windows, but be sure that your work also runs in Linux (I'll score all the works only in Ubuntu 16.04)
- Personal Project
- You should submit a report
  - Describe your code in detail.
    - Block Management
    - Problems that occur during GC & Your solution
  - Capture and analyze the WAF of Multi-stream Support FTL Simulator
- Submit to the icampus
  - Due: 11/13(Wed.) 23:59:59
  - File to submit: ftl_sim.c, ftl.h, ftl.c, nand.h, nand.c, Makefile, report.pdf
  - File name: $(STUDENT_ID).tar.gz
- Late penalty: -20% per day (Up to 3 days)

Any Questions?