Tutorial FTL

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

Contents

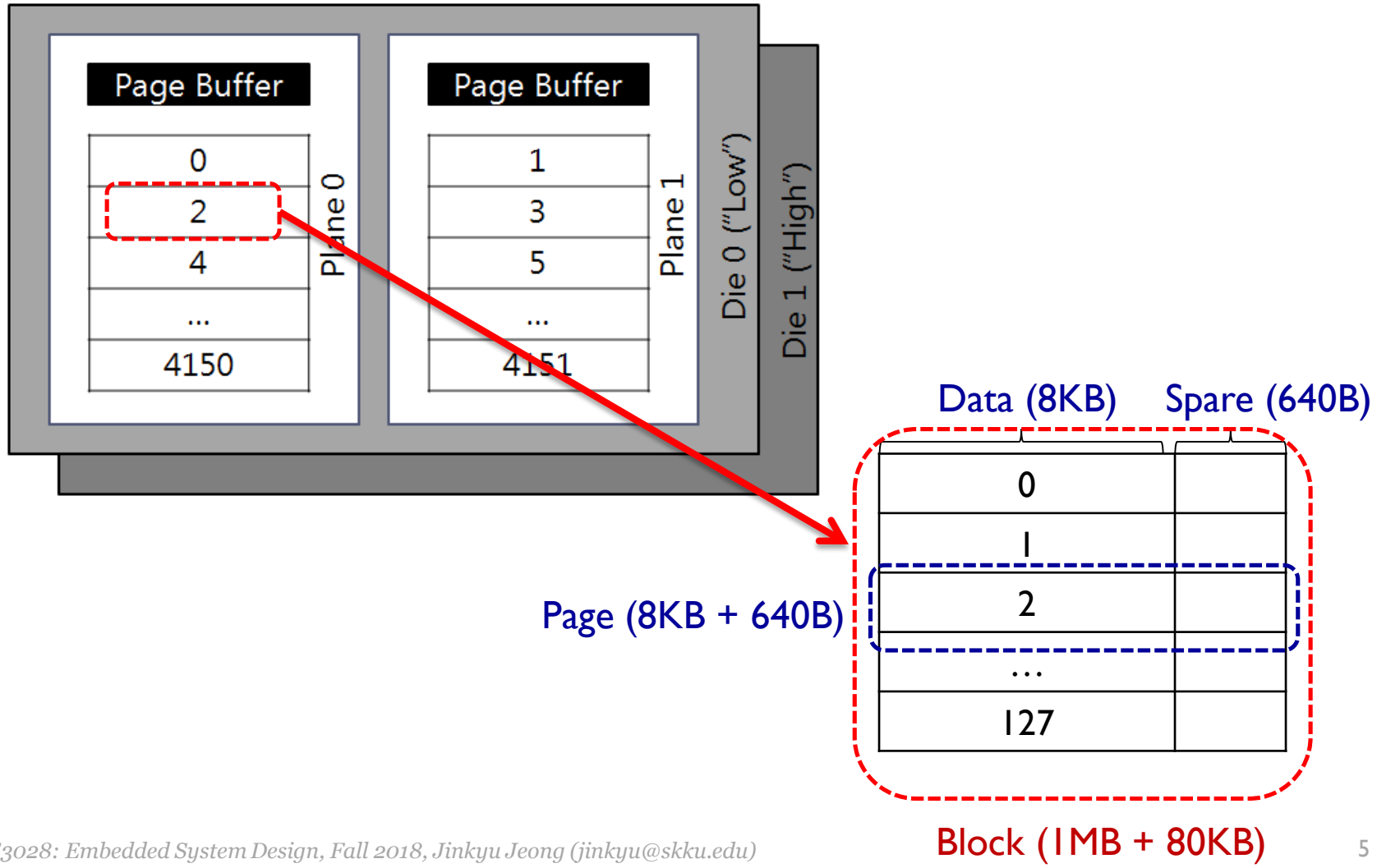
- Deep dive to the Jasmine
  - NAND flash memory
    - NAND flash memory configuration
    - NAND flash memory layout
  - Flash memory abstraction in Jasmine
  - Flash interface layer in Jasmine
    - NAND flash controller
- Intro. to Tutorial FTL
  - Read / Write in Tutorial FTL

Deep Dive to the Jasmine

NAND Flash Memory

- K9LCG08U1M (Dual die)
  - Samsung 35 nm 2-bit MLC flash
  - Configuration
    - 16 sectors per page (8 KB + 640 B)
    - 128 pages per block (1 MB + 80 KB)
    - 4096 + 56 blocks per die
  - Average performance
    - Page read : 250 us
    - Page program : 1.3 ms
    - Block erase : 1.5 ms

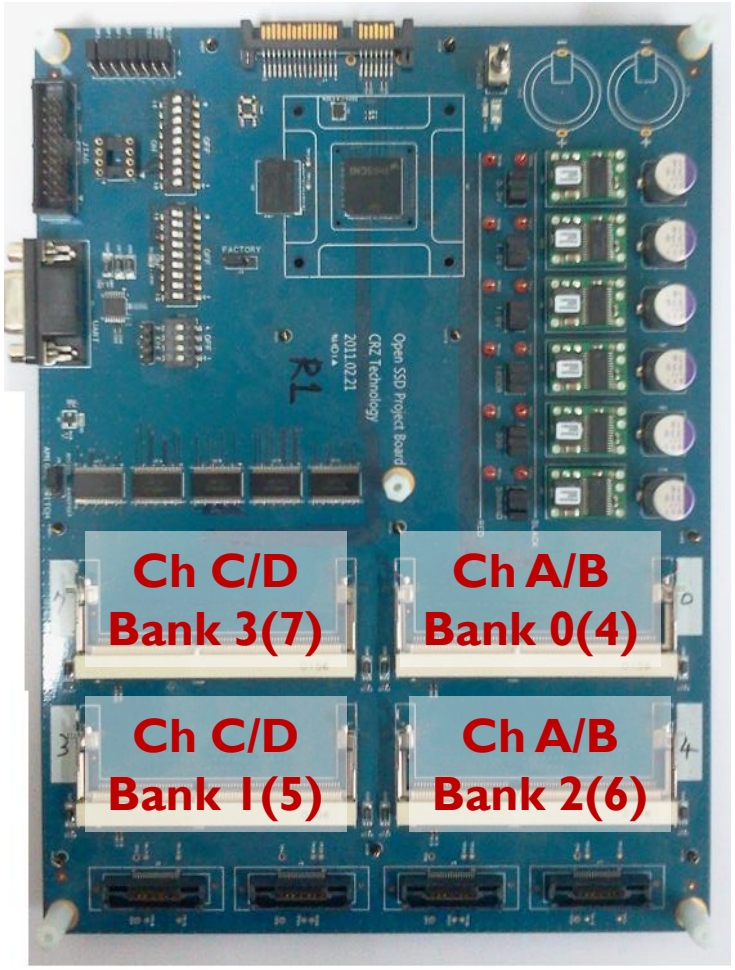
NAND Flash Memory(cont'd)

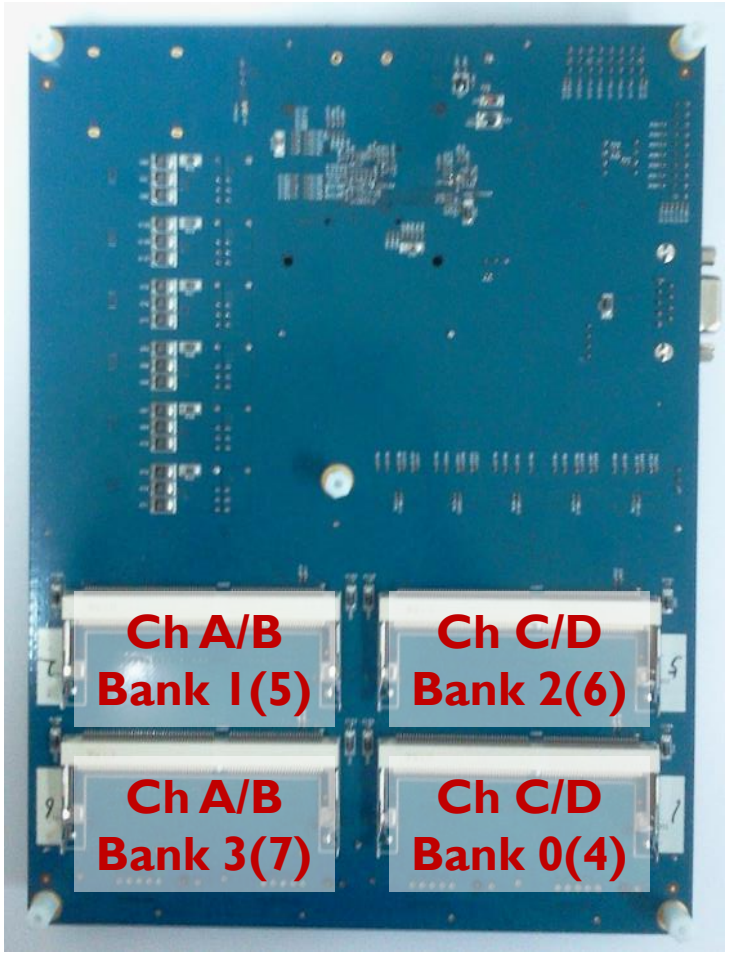


NAND Flash Operations

- Page read
  - Cell -> Page Buffer -> RAM
- Page program
  - RAM - > Page Buffer -> Cell
- Page copy-back
  - Cell (src) -> Page Buffer -> Cell (dst)

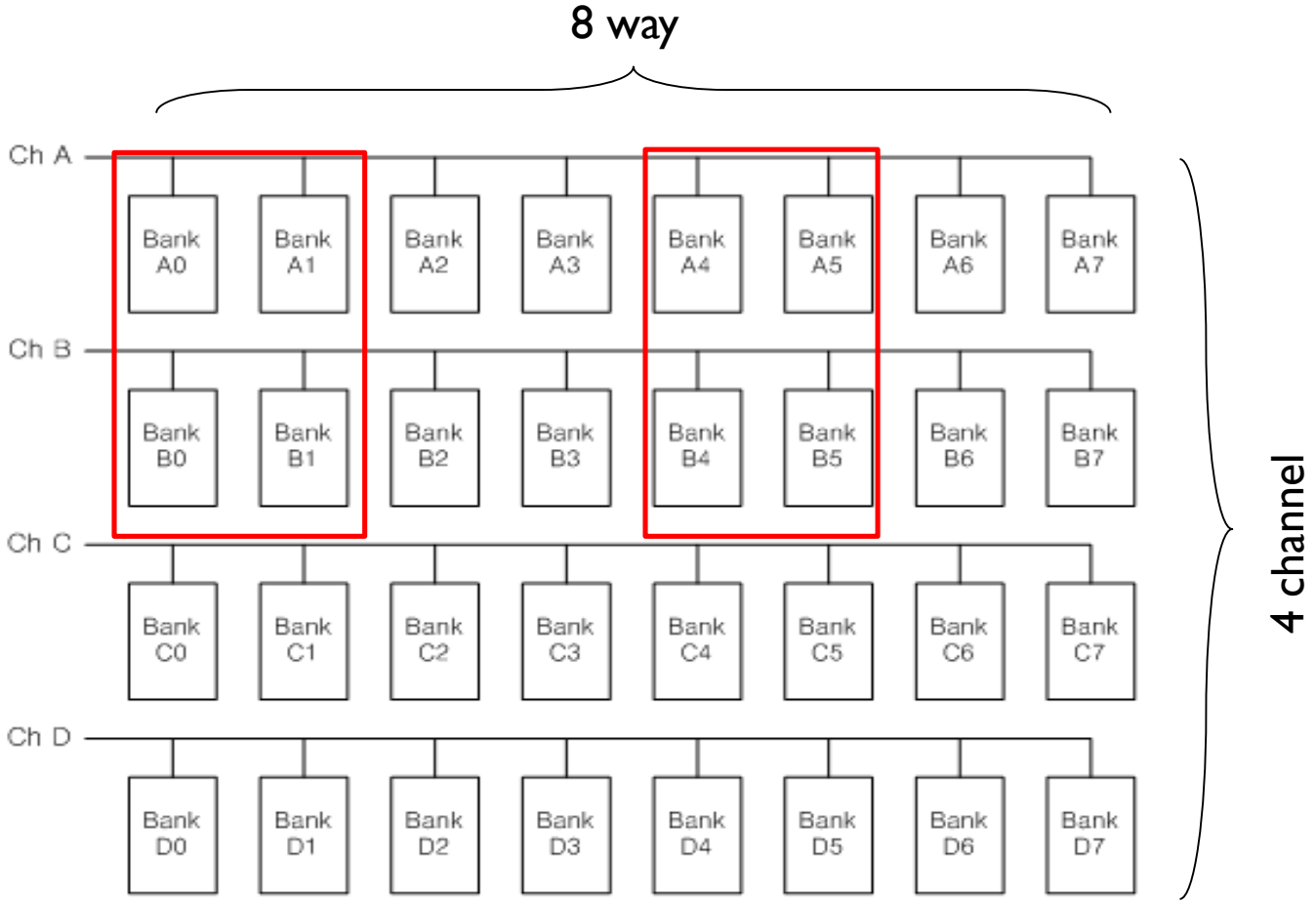
NAND Flash Physical Layout





NAND Flash Logical Layout

- Four channels + eight banks in each channel

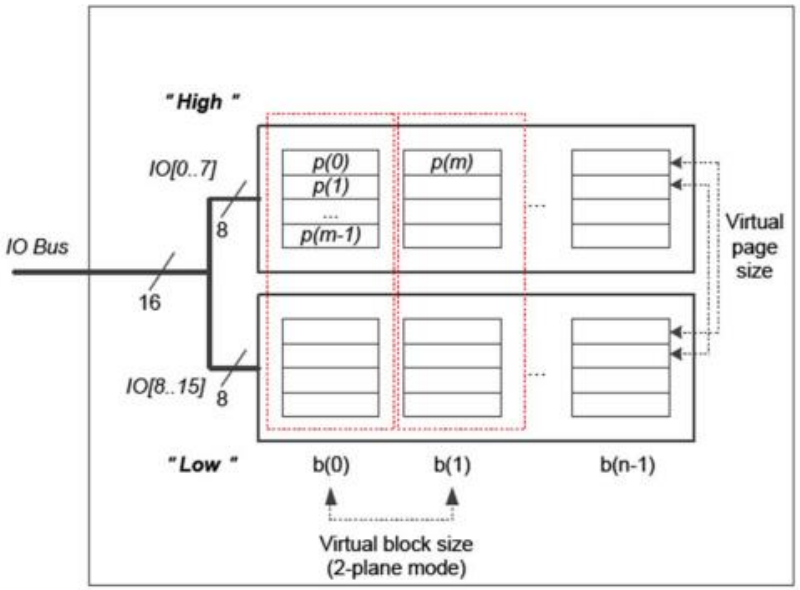


NAND Flash Logical Layout(cont'd)

- Channel
  - Banks in same channel share I/O bus
    - Only single I/O operation per a channel
- Bank
  - 16 bit I/O bus
  - High / Low die (dual die)
  - Each bank can perform cell operations in parallel
    - Internal operations not using I/O bus can run parallel
  - Barefoot has only 4 R/B signal inputs per channel
    - (0,4) (1,5) (2,6) (3,7) chip binding in a same channel
    - Maximum 4 way interleaving

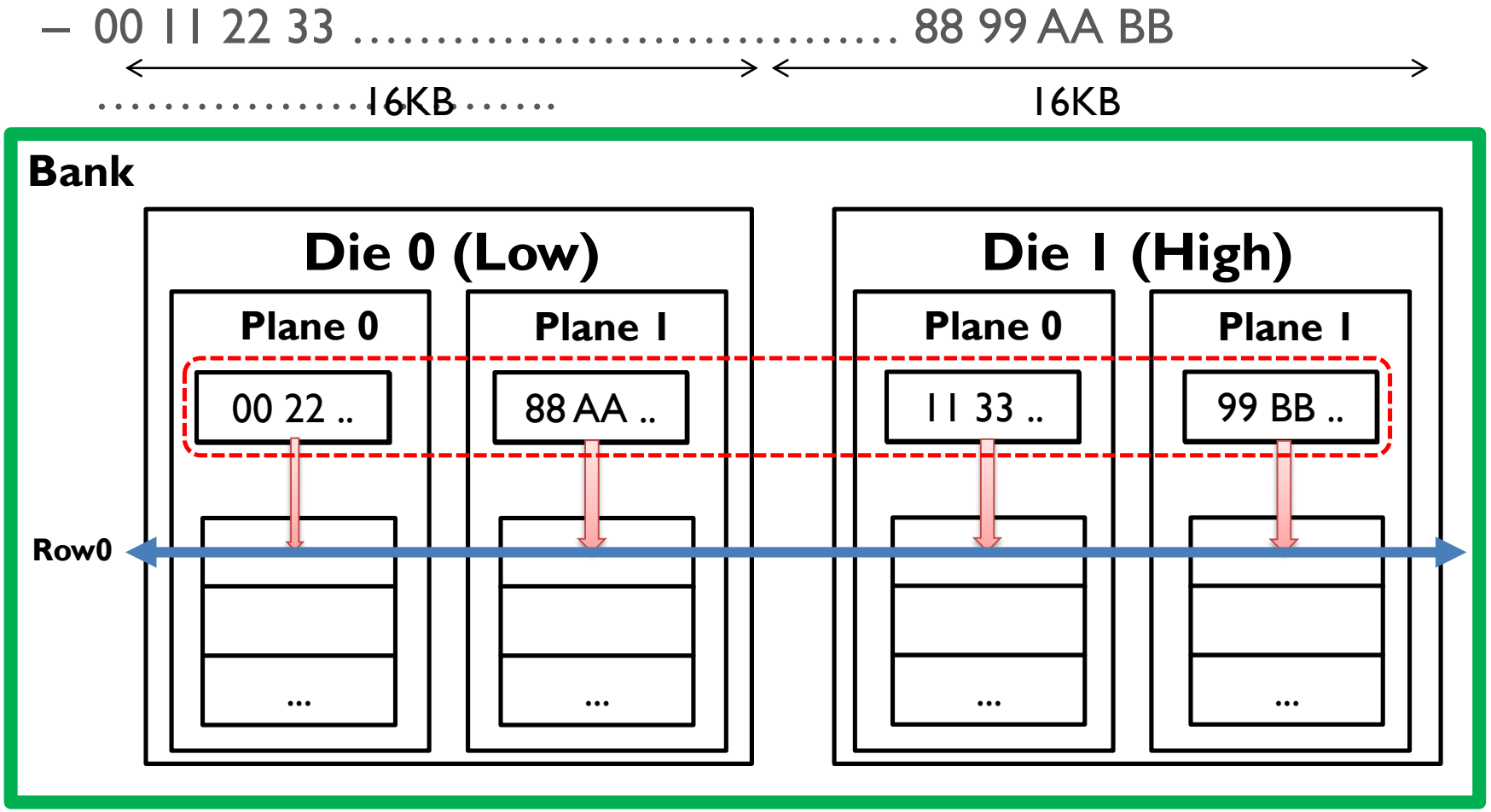
Flash Memory Abstraction

- Bank interleaved operation (dual die)
  - Example
    - DRAM buffer data = 01 23 45 67 89 AB CD EF 01 … (512 bytes)
    - 'Low' chip <- 01 45 89 CD 01 … (256 bytes)
    - 'High' chip <- 23 67 AB EF … (256 bytes)



Flash Memory Abstraction(cont'd)

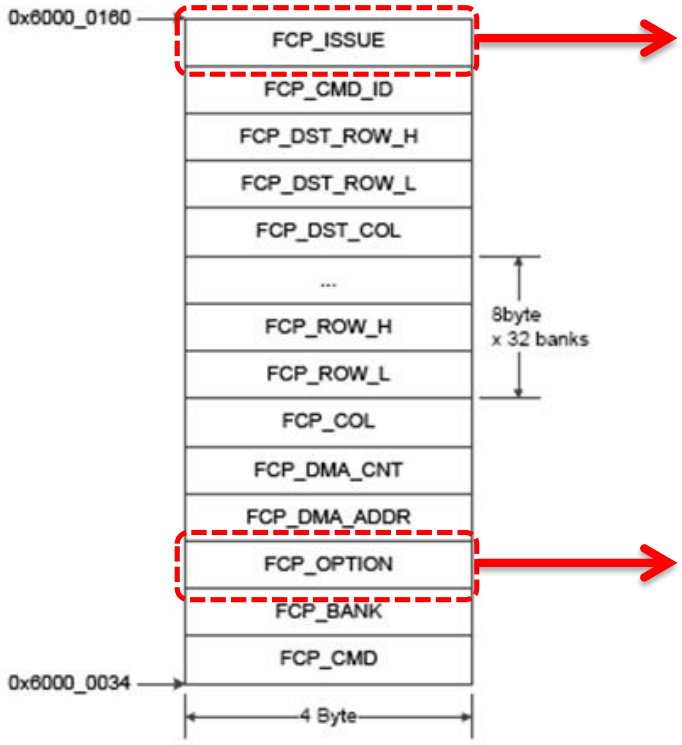
- 2-plane mode
  - 00 11 22 33 ..... 88 99 AA BB



NAND Flash Controller

- To issue NAND flash operation
  - include/flash.h

Registers	Offset
FCP_ISSUE	87
FCP_CMD_ID	86
FCP_DST_ROW_H	85
FCP_DST_ROW_L	84
FCP_DST_COL	83
FCP_ROW31_H	82
FCP_ROW31_L	81
...	...
FCP_ROW0_H	20
FCP_ROW0_L	19
FCP_COL	18
FCP_DMA_CNT	17
FCP_DMA_ADDR	16
FCP_OPTION	15
FCP_BANK	14
FCP_CMD(FCP_BASE)	13
WR_BANK	12
WR_STAT	11

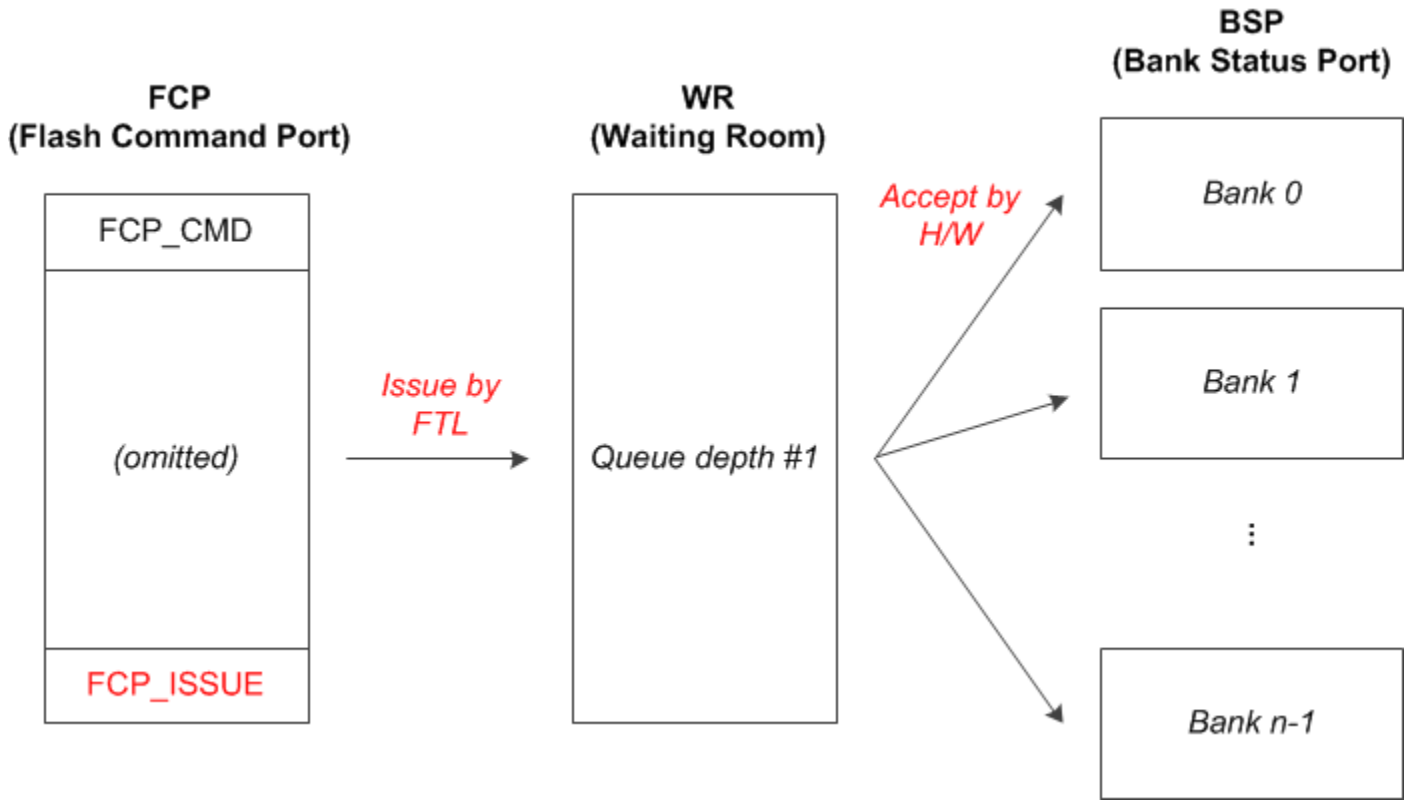


FC_COL_ROW_READ_OUT
FC_COL_ROW_IN_PROG
FC_COPYBACK

FO_P
FO_E
FO_B_SATA_W
FO_B_SATA_R

NAND Flash Controller (cont'd)

- To issue NAND flash operation
  - include/flash.h



Dive to the code: FCP Setting

```
void nand_page_ptread_to_host(UINT32 const bank, UINT32 const vblock,
UINT32 const page_num, UINT32 const sect_offset, UINT32 const num_sectors)
{
…
    row = (vblock * PAGES_PER_BLK) + page_num;

    SETREG(FCP_CMD, FC_COL_ROW_READ_OUT);
    SETREG(FCP_DMA_ADDR, RD_BUF_PTR(g_ftl_read_buf_id));
    SETREG(FCP_DMA_CNT, num_sectors * BYTES_PER_SECTOR);

    SETREG(FCP_COL, sect_offset);
#if OPTION_FTL_TEST == TRUE
    SETREG(FCP_OPTION, FO_P | FO_E);
…
    SETREG(FCP_ROW_L(bank), row);
    SETREG(FCP_ROW_H(bank), row);

    g_ftl_read_buf_id = (g_ftl_read_buf_id + 1) % NUM_RD_BUFFERS;


…
    flash_issue_cmd(bank, RETURN_ON_ISSUE);
}
```

Dive to the code: FCP Issue

```
void flash_issue_cmd(UINT32 const bank, UINT32 const sync)
{
        UINT32 rbank = REAL_BANK(bank);

        SETREG(FCP_BANK, rbank);

        while ((GETREG(WR_STAT) & 0x00000001) != 0);

        SETREG(FCP_ISSUE, NULL);

        if (sync == RETURN_ON_ISSUE)
                return;

        while ((GETREG(WR_STAT) & 0x00000001) != 0);

        if (sync == RETURN_ON_ACCEPT)
                return;

        while (_BSP_FSM(rbank) != BANK_IDLE);
}
```

Tutorial FTL

Tutorial FTL: Overview

- ./ftl_tutorial
  - ftl.c, ftl.h
- Page Mapping FTL
  - Write data from DRAM to NAND
  - Read data from NAND to DRAM
  - No garbage collection

Tutorial FTL: Page Mapping Table

- LPN to PPN map
  - LPN (Logical Page Number)
    = LBA (from host) / Sectors per Page
  - PPN (Physical Page Number)

Index (=LPN)	0	1	2	3	…	209715
Value (=PPN)	100	256	0	INVALID	…	20000

```
static UINT32 get_physical_address(UINT32 const lpage_addr)
{
    // Page mapping table entry size is 4 byte.
    return read_dram_32(PAGE_MAP_ADDR + lpage_addr * sizeof(UINT32));
}

static void update_physical_address(UINT32 const lpage_addr, UINT32 const new_bank, UINT32 const new_row)
{
    write_dram_32(PAGE_MAP_ADDR + lpage_addr * sizeof(UINT32), new_bank * PAGES_PER_BANK + new_row);
}
```

Assignment

- Describe functions in target_spw/flash_wrapper.c within three sentences
  - functions
    - nand_page_read
    - nand_page_program
  - Send it to donghyun.kim@csl.skku.edu
  - Due: Until next lab class

Any Questions?