A Simple Approach to Find the Address Mapping Scheme of USB Flash Drives

Jaehyuck Cha (chajh@hanyang.ac.kr)
Sooyong Kang (sykang@hanyang.ac.kr)

Hanyang University, Korea

Hanyang University
Database Laboratory

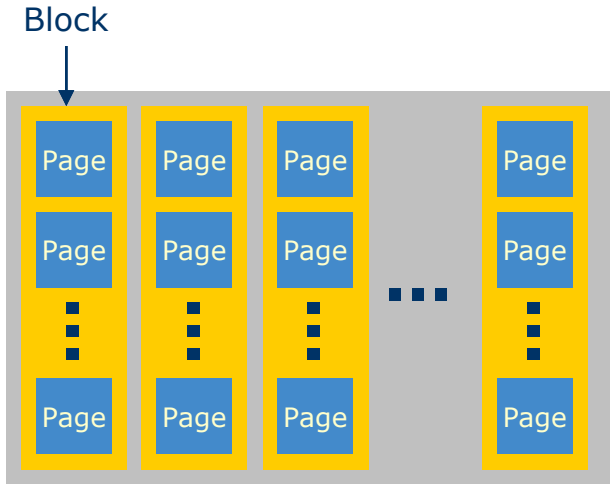
Outline

- **Backgrounds**
  - NAND Flash Memory
  - USB Flash Drives
- **Motivation**
- **Mapping Algorithms of FTL**
- **Mapping Algorithm Identification Mechanism**
- **Experiments**
- **Conclusion & Further Issues**

NAND Flash Memory

Physical organization

- An array of blocks (erase units)
- A block is an array of pages
- Example
  - 16 Kbyte block
    = 512 Byte page x 32
  - 128 Kbyte block
    = 2 Kbyte page x 64
  - 512 Kbyte block
    = 4 Kbyte page x 128
- Spare area
  - Small space for ECC and application's metadata
  - Example
    - 16 byte for 512 Byte page
    - 64 byte for 2 Kbyte page
    - 128 byte for 4 Kbyte page



NAND Flash Memory (con'd)

- Read/Write and erase operations

*Samsung Datasheets K9XXG08UXM (www.datasheet4u.com)

Operation	Data Unit	Time (us)
Read	Page	60
Write	Page	800
Erase	Block	1500
Copy-back*	Page	~800

* : Optional operation

- Asymmetric I/O in terms of its speed and data unit
- Not in-place-update: A block should be erased to re-write a page
  → (Logical to physical) Address mapping

Example of NAND Flash Operations

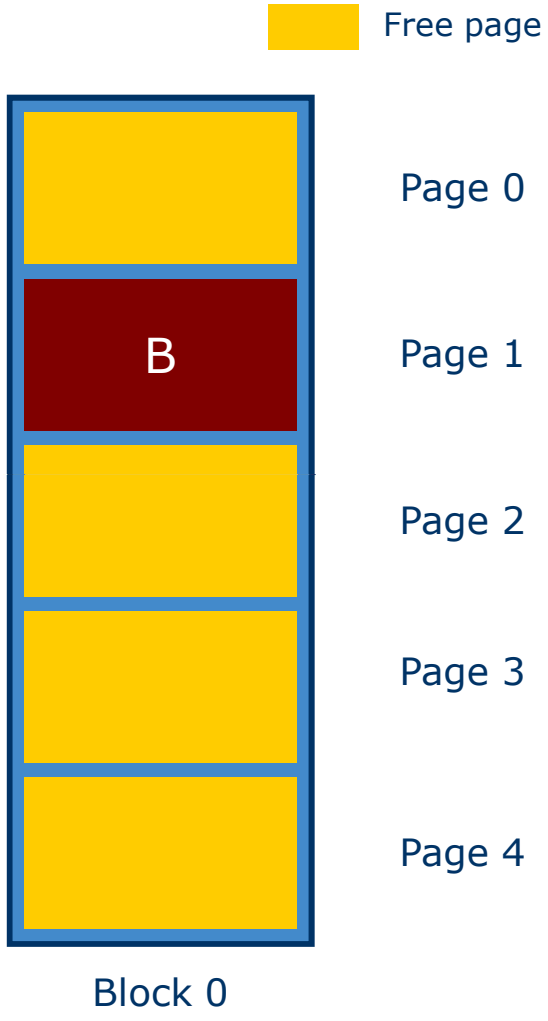
❖ Modifying a data page

- Write data 'A' to sector 1
- Write data 'B' to sector 1
  - Re-write to sector 1

Write page 0 again

Every update operation to a data page causes a block erase operation.

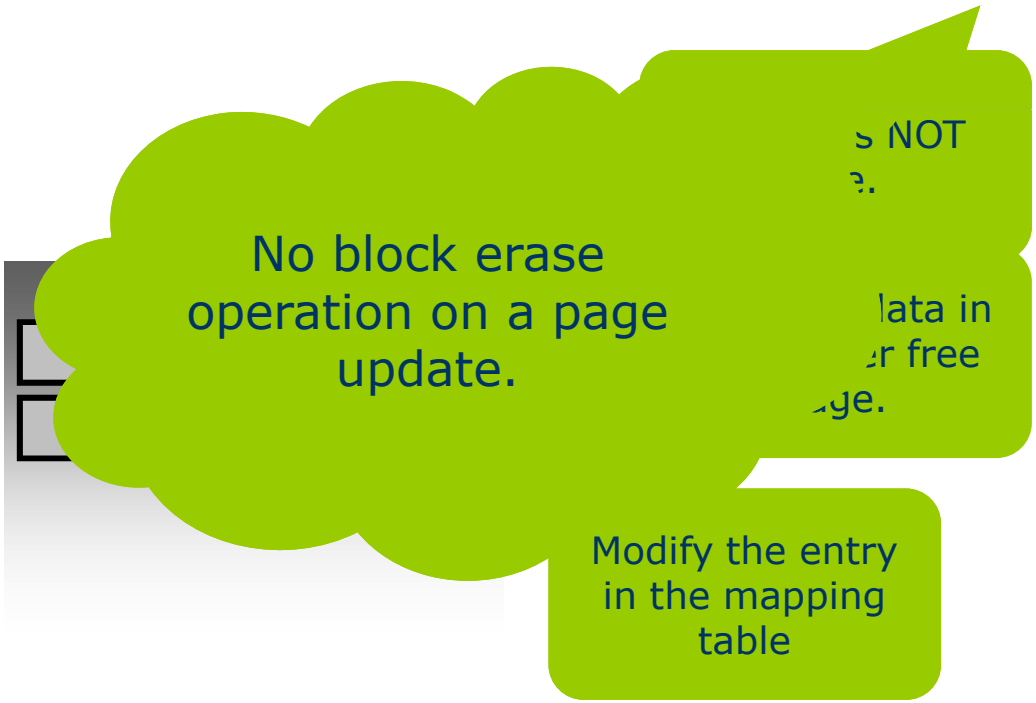
Now, erase the block

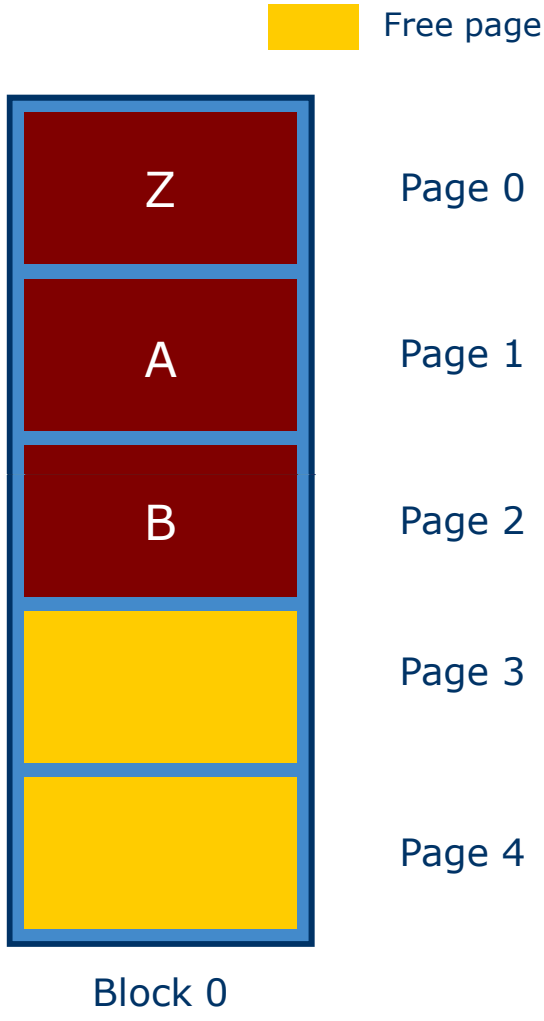


Example of NAND Flash Operations (cont'd)

❖ A better approach

- Write data 'A' to sector 1
- Write data 'B' to sector 1





NAND Flash Memory (con'd)

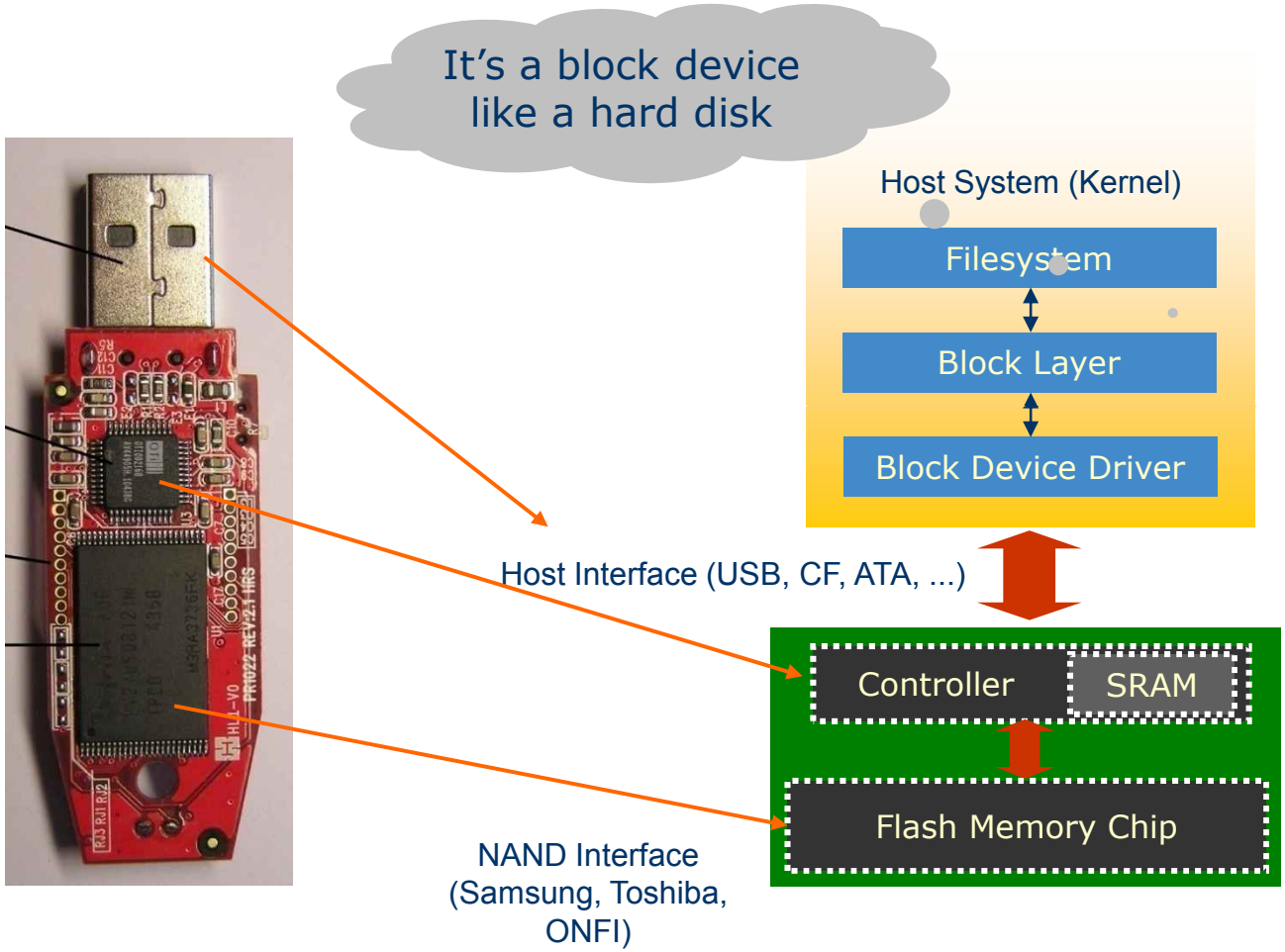
❖ Characteristics (con'd)

- Bad blocks, bit-flips
  → Bad block management
- A block can be worn-out after limited # of erase operations
  → Wear-leveling

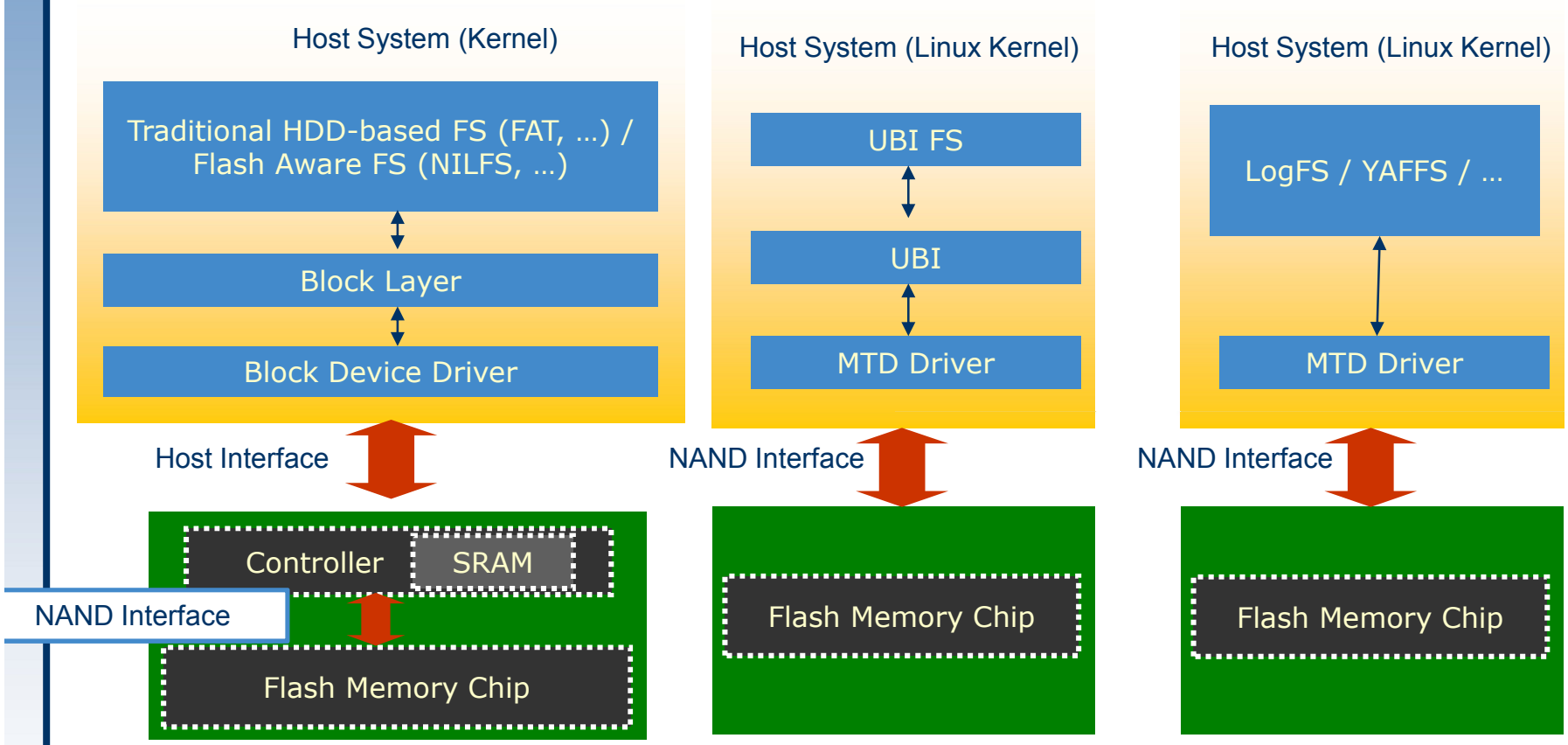
- Large block flash memories (2K and 4K page)
  - Pages should be written in ascending order in a block
  - Applications cannot use spare area (due to large ECC)
  - Shorter life cycle

USB Flash Drives

❖ Internals of a USB flash drive



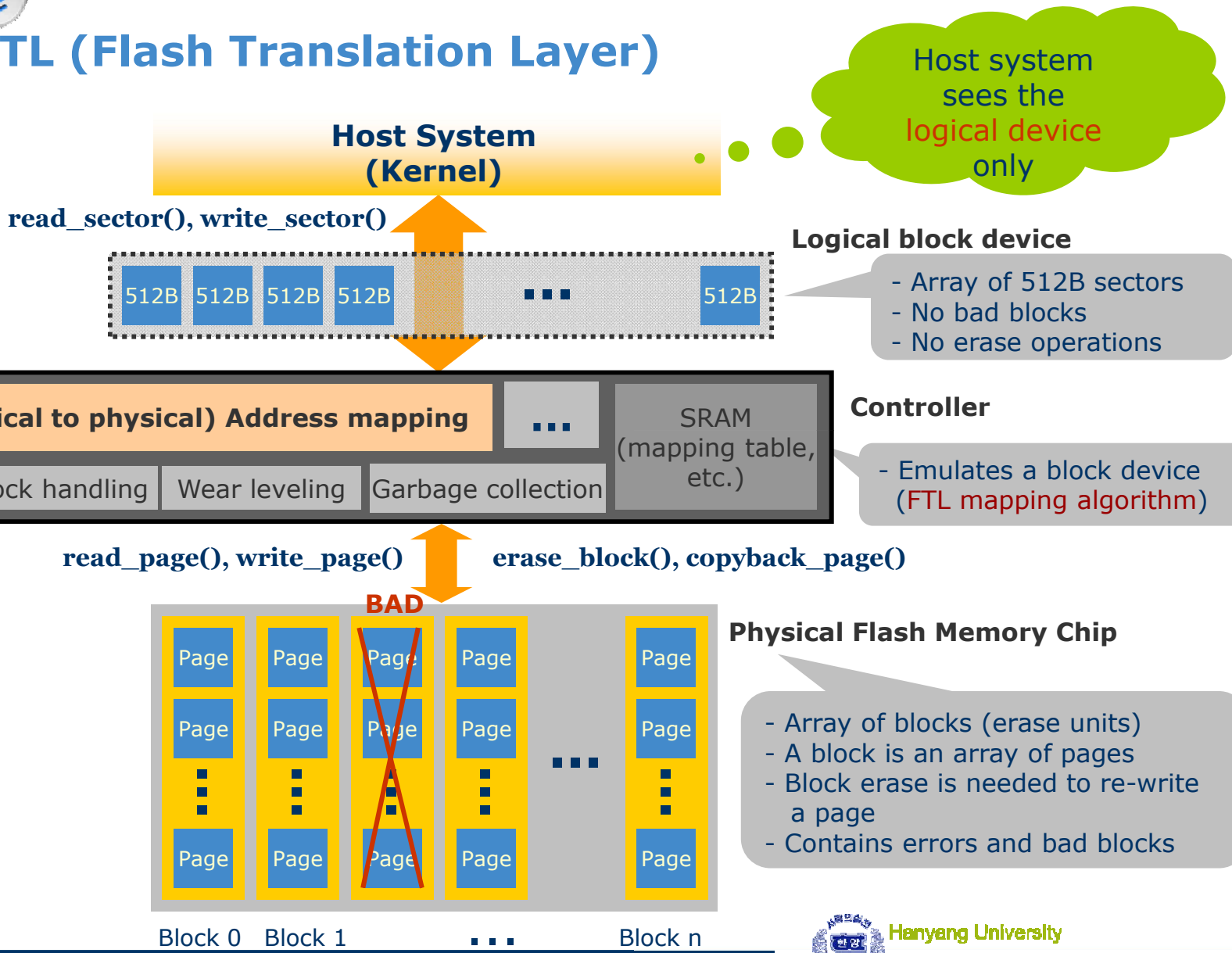
Comparison of Flash-based File System Architecture



File System	TPC-B Result (tps)
Ext2	16.26
NILFS	55.03

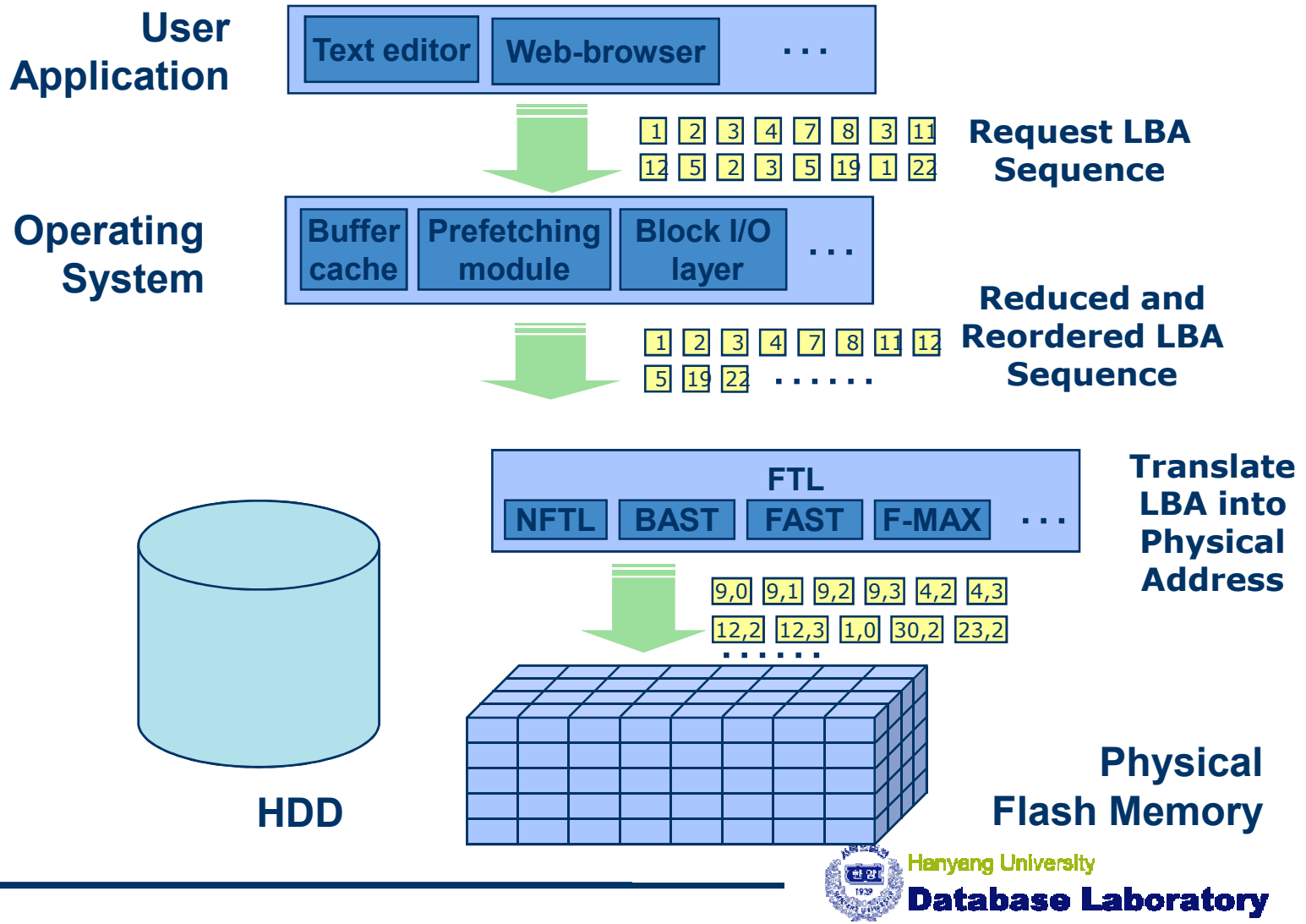
USB Flash Drives (cont'd)

❖ FTL (Flash Translation Layer)



Motivation

❖ For Performance Tuning,
the Address mapping of a FTL is important!!



Motivation

❖ Performance according to the various Address mapping schemes and the data traces

Effect of Buffer Replacement Algorithms on BAST with GCC trace			
Algorithm	Hit ratio	# write	# erase
LRU	21.44%	9,479	269
FAB	20.14%	8,600	232

TPC-C Result on various FTL					
FTL	# read	# write	# erase	#copy-back	Response Time
FMAX	1745054	900476	26601	618497	0.55
Mitsubishi	14403668	887344	26232	959039	1.76
BAST	48909061	887344	1502727	47922889	17.95

Linux Desktop Trace on Various FTL				
FTL	# read	# write	# erase	Elapsed Time
FMAX	32976540	2029560	62845	86.13
Mitsubishi	25522819	3536466	109890	118.23
BAST	2011545	2005865	62561	54.64

Address Mapping Schemes of FTL

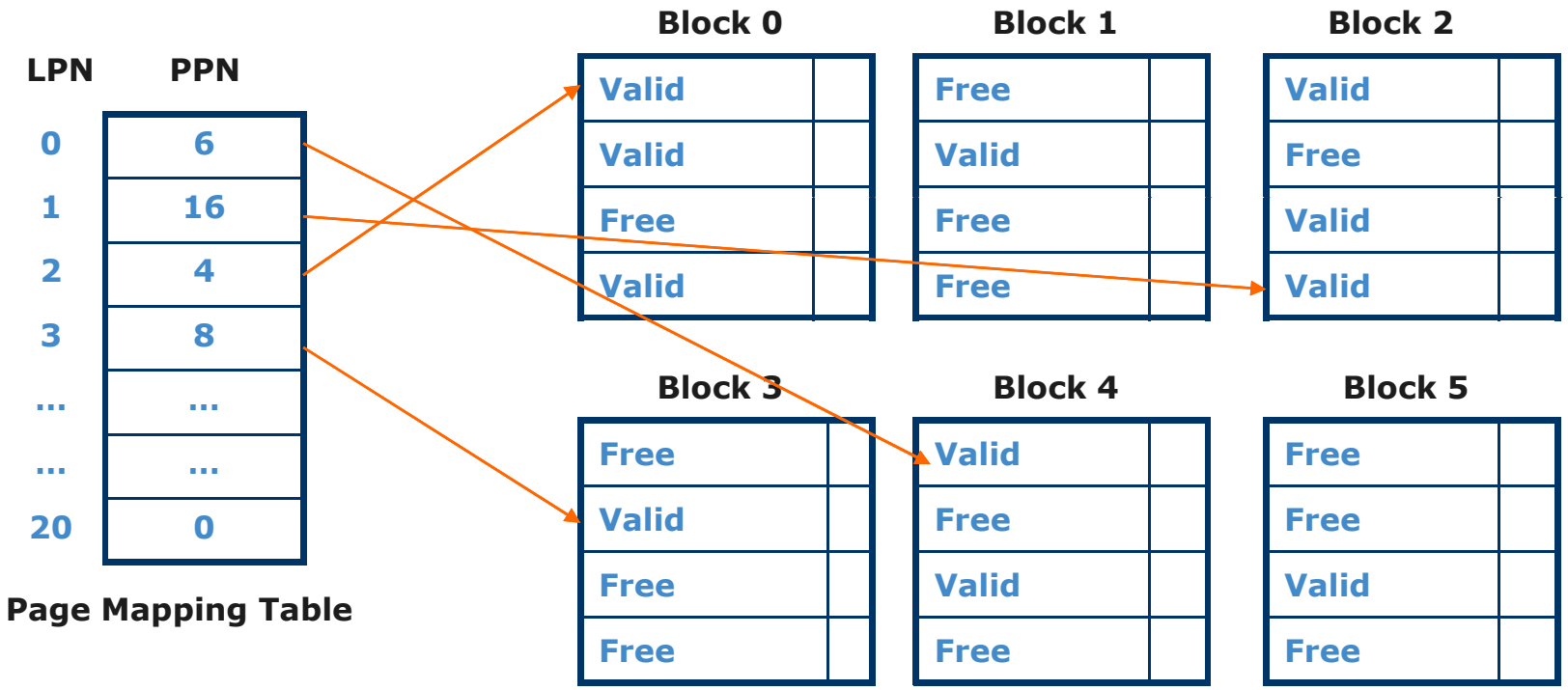
❖ Address Mapping schemes

- Page-level mapping scheme
- Block-level mapping scheme
- Hybrid mapping scheme

Mapping Schemes (con'd)

❖ Page-level mapping scheme

- Keep mapping entries for each page
  - Mapping entry (Logical page # : Physical page #)

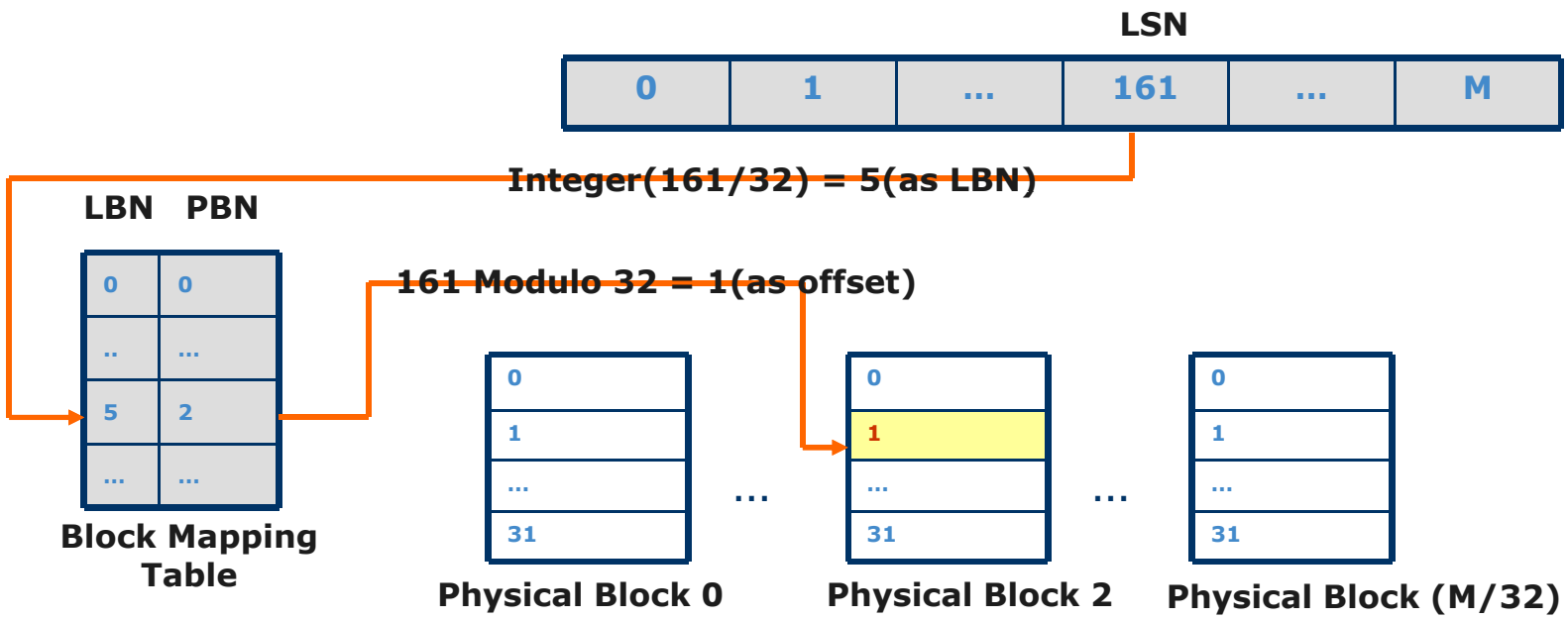


It's not used for USB flash drives because huge mapping table is needed!

Mapping Schemes (cont'd)

❖ Block-level mapping scheme

- Keep mapping entries for each block
- Mapping Entry (Logical block #, Physical block #)

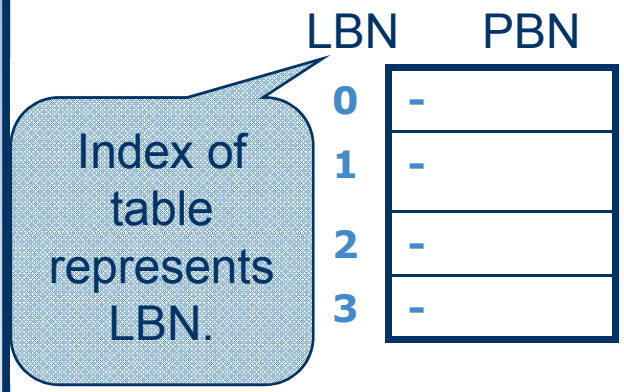


Example of Modified Block-level Mapping

❖ Developed by Lexar

- Has special spare block called "Replacement Block"
- Any free block can be a replacement block
- Block in Block Map is called "Data Block"

# of Pages/Block=4
total # of effective Blocks=4



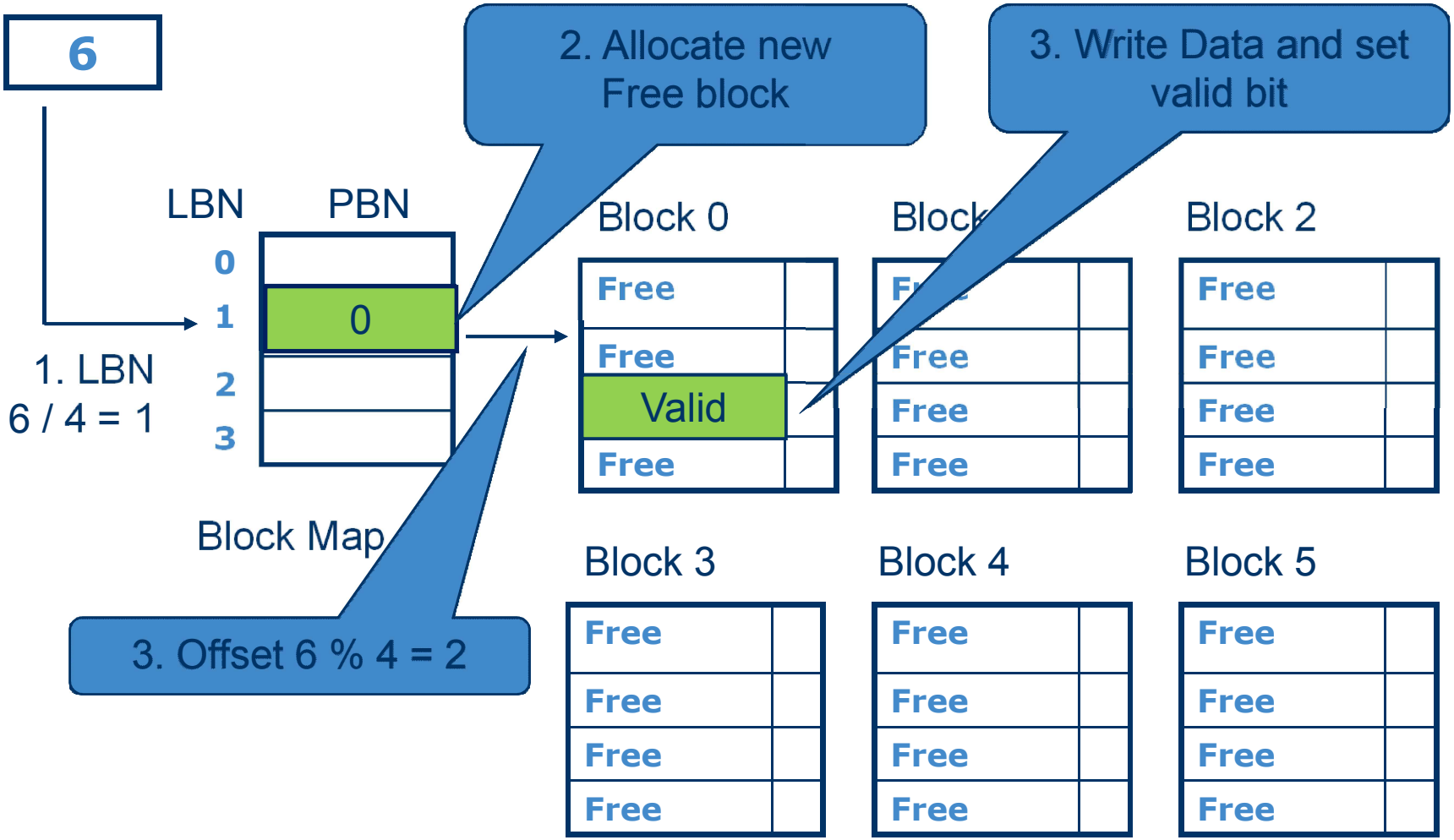
Block Map (SRAM)

1 bytes is enough for addressing all blocks, thus total RAM table size is **4 *1 = 4 bytes**

Initial state of NAND FTL

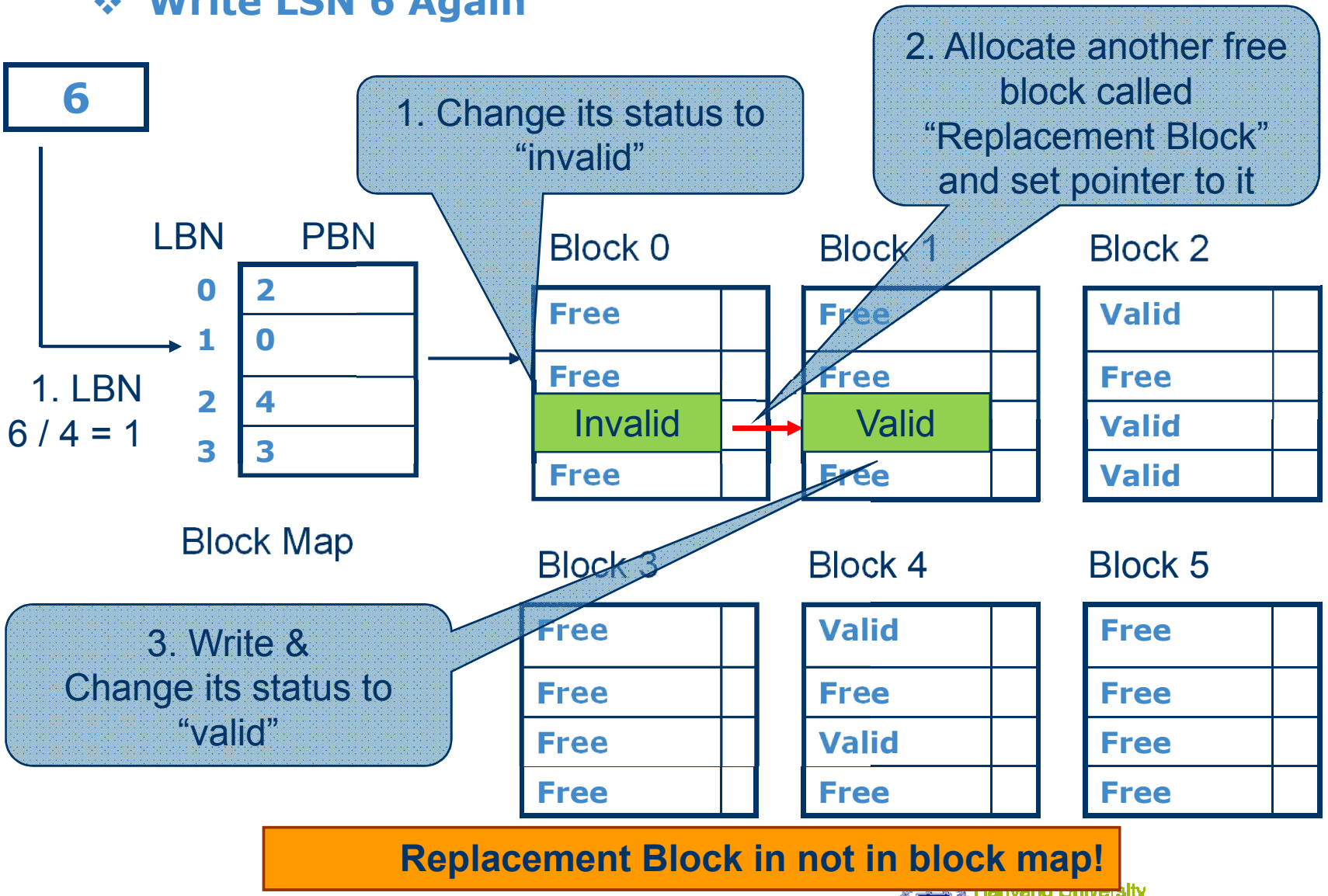
Example of Modified Block-level Mapping (cont'd)

❖ **Write LSN 6**



Example of Modified Block-level Mapping (cont'd)

❖ Write LSN 6 Again

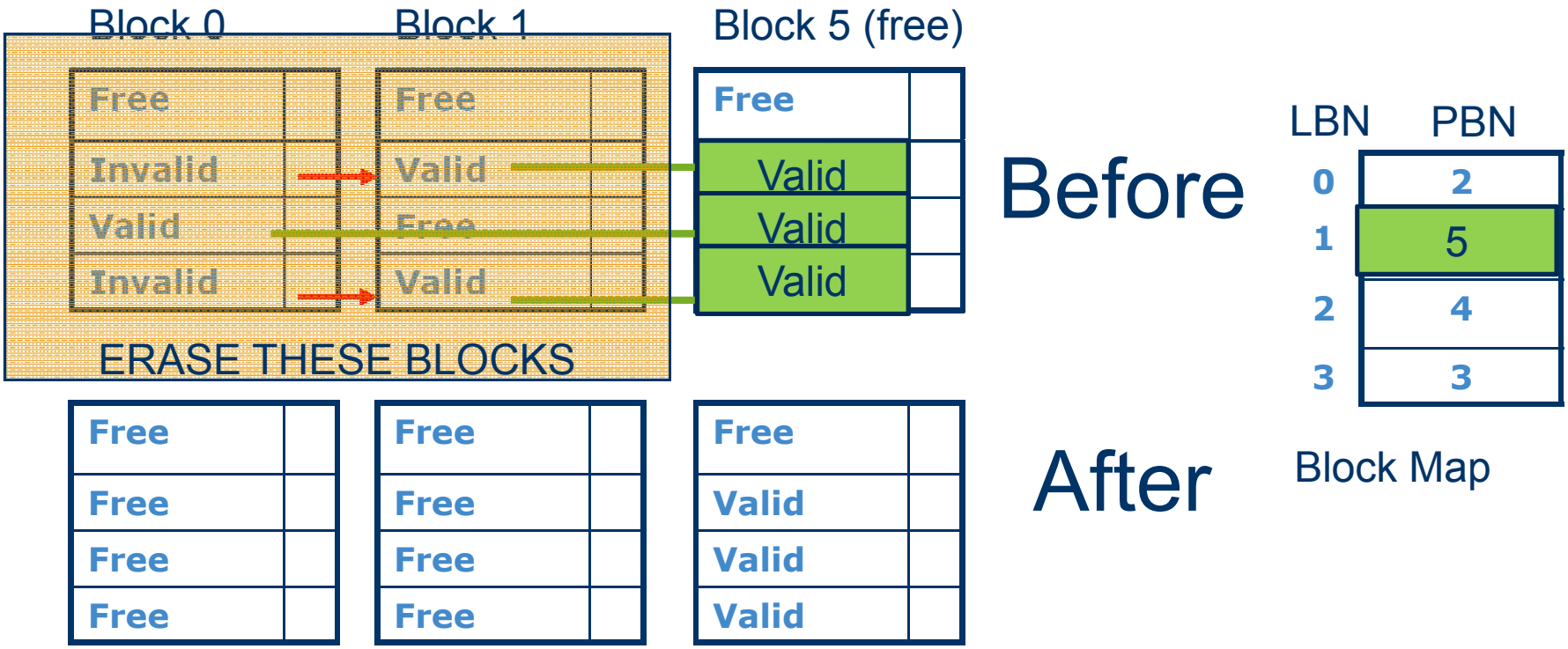


Replacement Block in not in block map!

Example of Modified Block-level Mapping (cont'd)

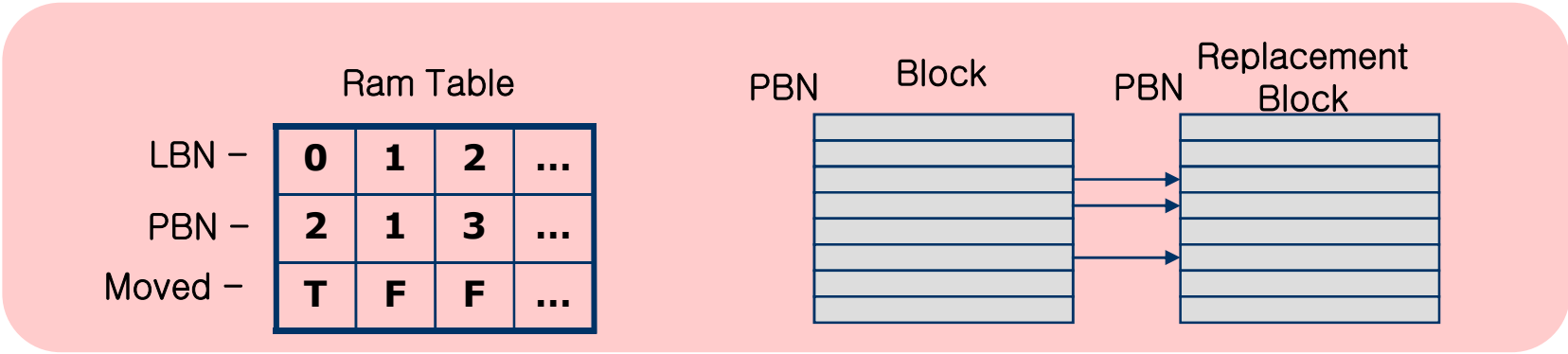
❖ Merge Operation

- When there's not enough space in replacement block, merge operation should be done.
- Choose a block (or blocks) and copy valid pages to new block and erase old block(s)



Typical Block-level mapping scheme

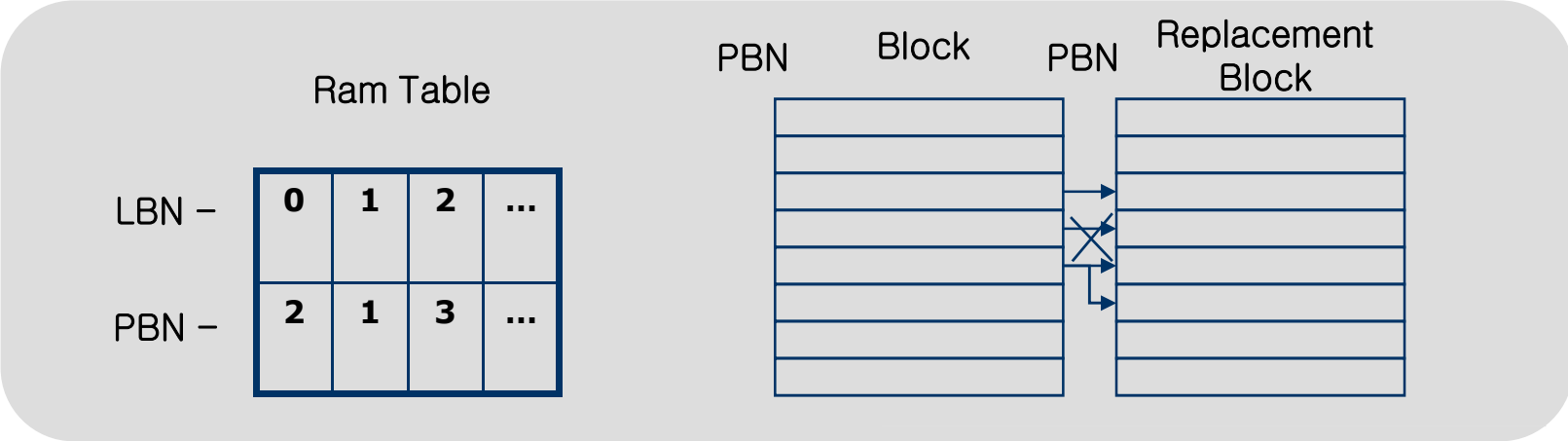
❖ Lexar Media (LM)



- The total # of effective data blocks
  - is approximately the half of the total # of blocks
- Each effective data block has at most one replacement block
  - If so, each page in an effective block has exactly one page with same offset in a corresponding replacement block

modified Block-level mapping scheme

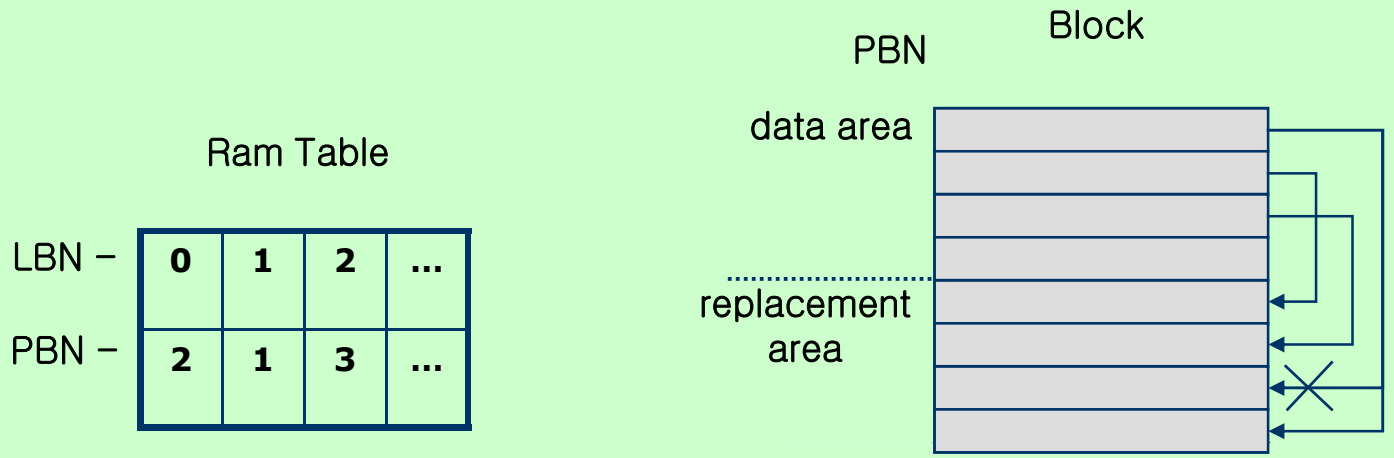
❖ M-Systems (FMAX)



- The total # of effective data block
  - is the half of the total # of blocks
- Each effective data block has at most one replacement block
  - If so, each page in an effective block can have several pages (among them, only one valid page exist!) in a corresponding replacement block

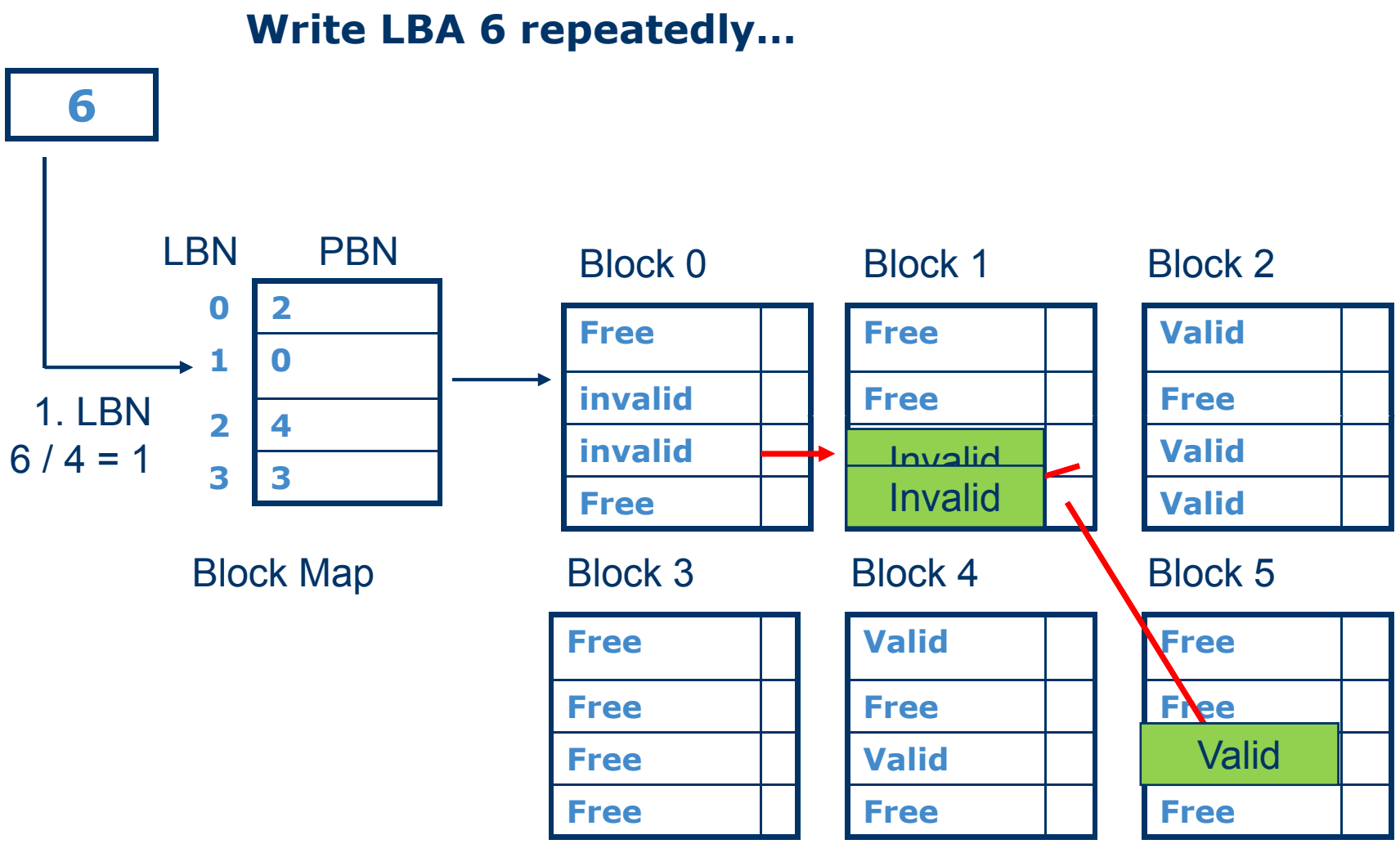
modified Block-level mapping scheme

❖ Mitsubishi (MSBS)



- The total # of effective data blocks
  - is approximately same as the total # of blocks
- Each effective data page in a data block
  - has several corresponding pages (among them, only one valid page exits!) in the replacement area of a same data block

Example of Modified Block-level Mapping – FMAX2

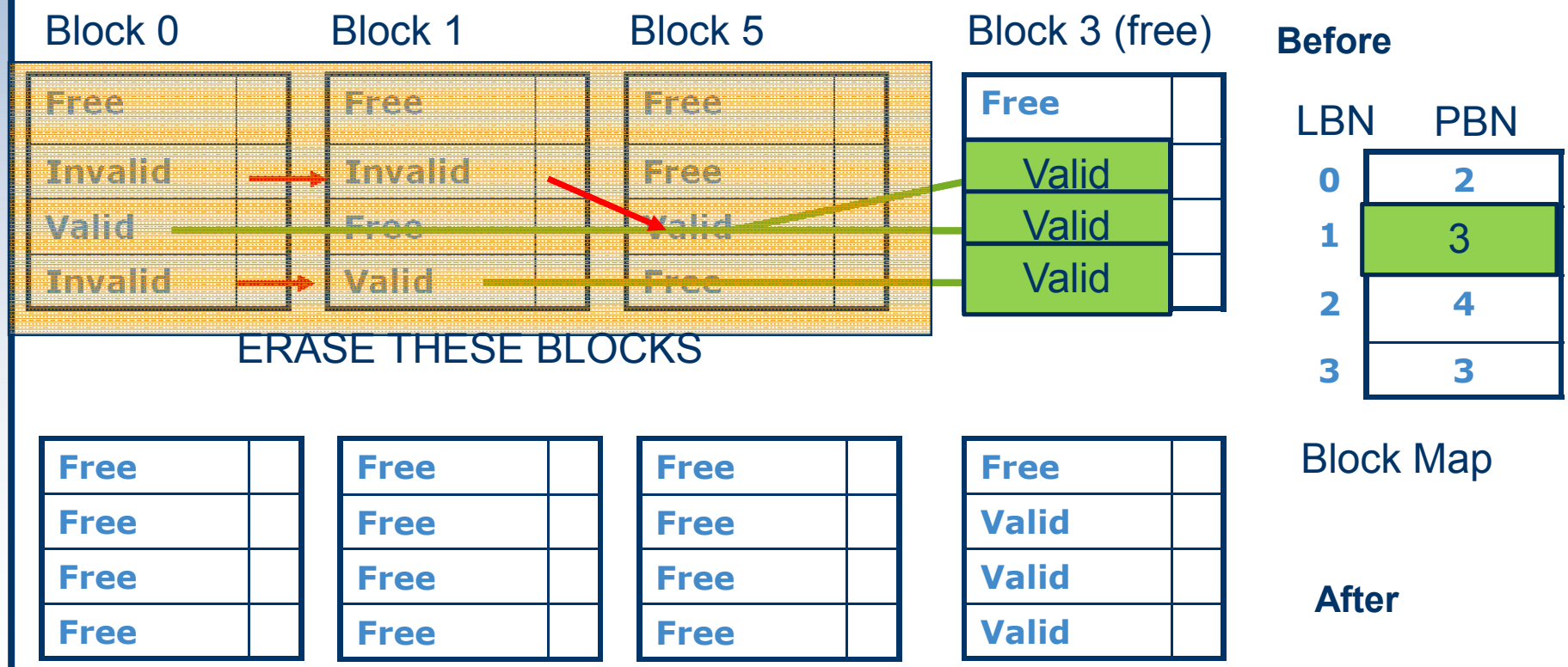


❖ FMAX2 allows maximum 2 replacement blocks per each page.

Example of Modified Block-level Mapping – FMAX2 (cont'd)

❖ Merge Operation of FMAX2

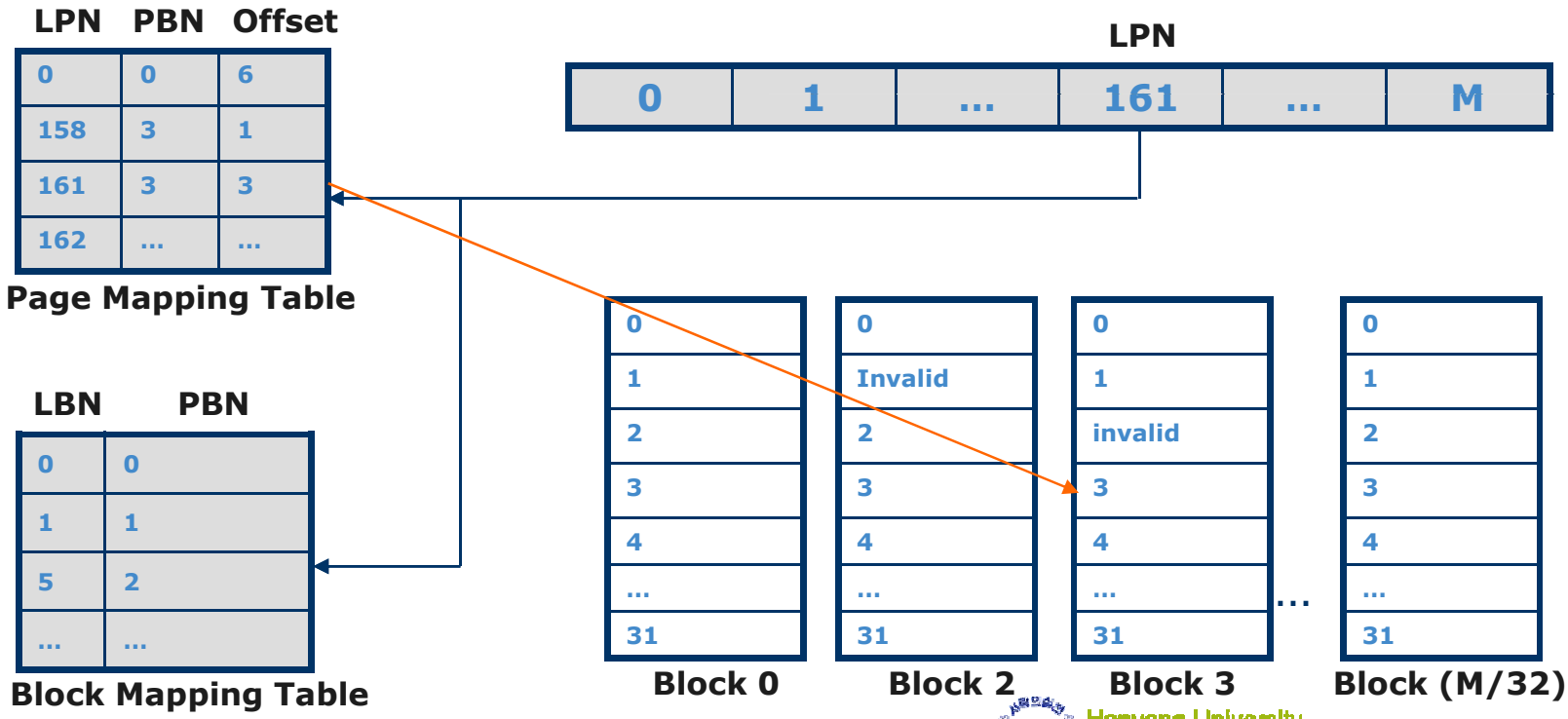
- 3 blocks are associated with a single merge operation!



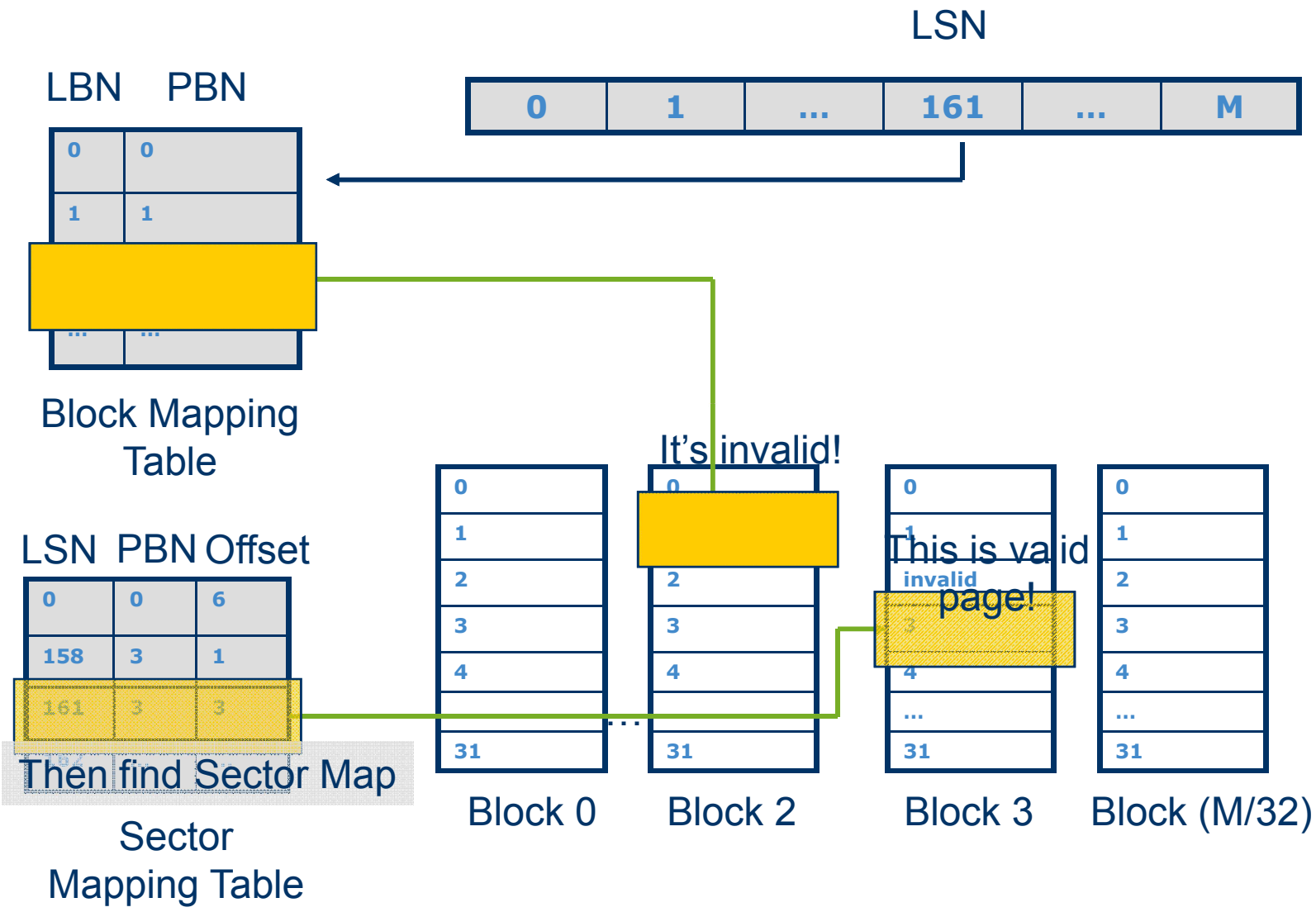
Mapping Schemes (cont'd)

❖ Hybrid mapping scheme

- Most of blocks called "data block" are allocated by block mapping algorithm
- A few blocks called "log block" are allocated by page mapping algorithm



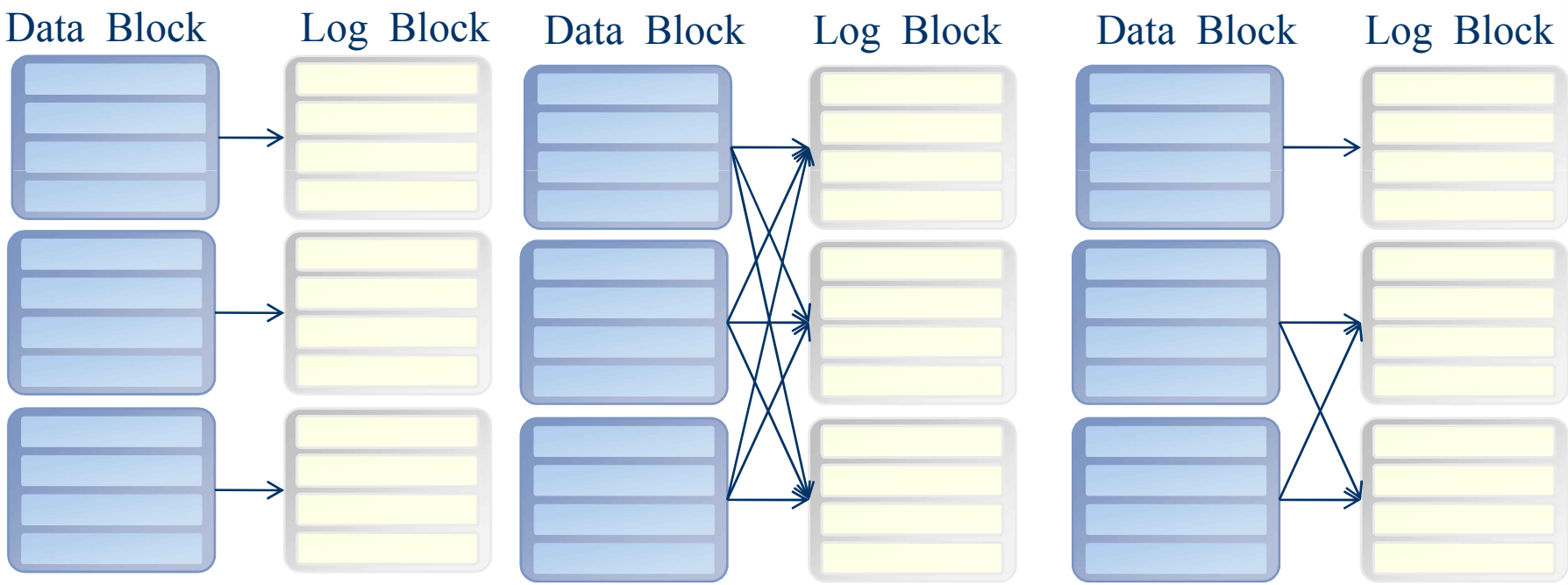
Simple Hybrid Mapping Example



Hybrid Mapping

❖ BAST, FAST, Super Block

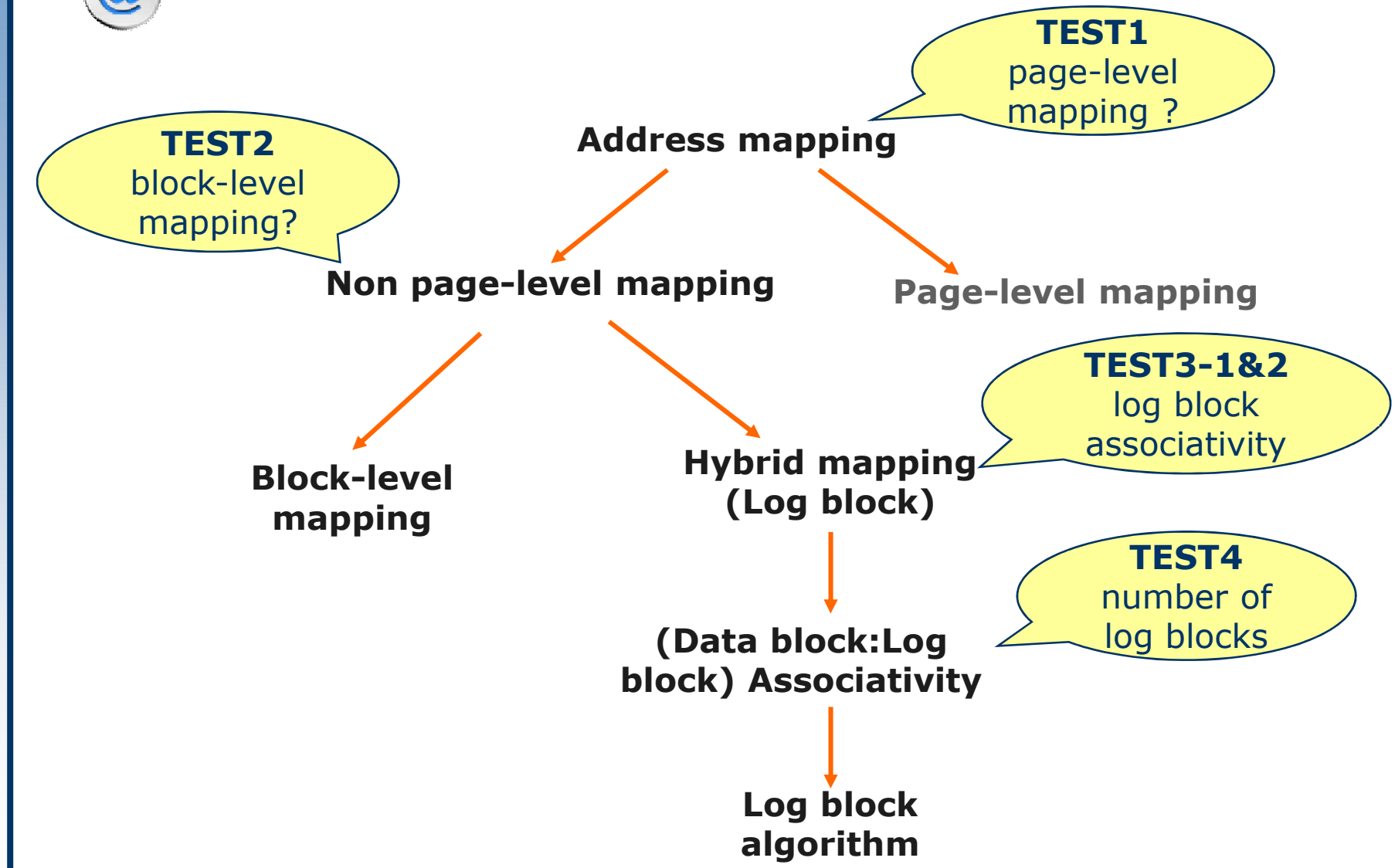
- Data Block : Log Block = N : M
  - P: the total # of effective data pages
  - L: the total # of log blocks



Identification of the Mapping Scheme
Target Device Assumption

- **USB flash drives that have large-block NAND flash memories**
  - We can get the number of pages per block and page size from datasheets
  - A block has 64/128 pages
  - The size of a page is 2/4KB

Identification of the Mapping Scheme
Classification of Major Mapping Schemes



Identification of Mapping Schemes
Time Interval Measurements

❖ Timing measurements

- Open a character device file with O_DIRECT
- Send requests to the device with page-oriented I/Os
- Measure the time interval between device *issue* and *complete*
  - Block IO trace facility of linux (*blktrace*)
  - Time interval is calculated in the unit of usec

```
int fd = open ("/dev/sda", O_RDWR | O_DIRECT);
char buf[PAGE_SIZE];
off_t offset = pageno * PAGE_SIZE;
…

/* Write a page. */
pwrite (fd, buf, count, offset);

…
```

Identification of the Mapping Schemes
Time Interval Measurements (cont'd)

root@srnote:~

```
1578    [W]     2          1155.622
1579    [W]     3          1168.168
1580    [W]     4          1170.077
1581    [W]     5          1197.659
1582    [W]     6          1215.932
1583    [W]     7          1231.724
1584    [W]     8          1248.315
1585    [W]     9          1263.141
1586    [W]     0          1279.237
1587    [W]     1          1294.268
1588    [W]     2          1307.268
1589    [W]     3          1309.057
1590    [W]     4          1339.124
1591    [W]     5          1353.418
1592    [W]     6          1369.884
1593    [W]     7          1385.641
1594    [W]     8          1400.322
1595    [W]     9          1415.994
1596    [W]     0          1402.280
1597    [W]     1         45446.889
1598    [W]     2          1461.066
1599    [W]     3           451.664
1600    [W]     4          1489.599
1601    [W]     5           508.064
1602    [W]     6          1524.941
1603    [W]     7           539.176
1604    [W]     8           553.513
1605    [W]     9           569.557

[Flash Shell @ /dev/sdb]
[ srnote ] 0 kernel message  1 blktrace  2 test_shell
```

Merge operations can be detected by comparing I/O latency

Identification of the Mapping Schemes
Test Methods 1

❖ Page-level mapping?

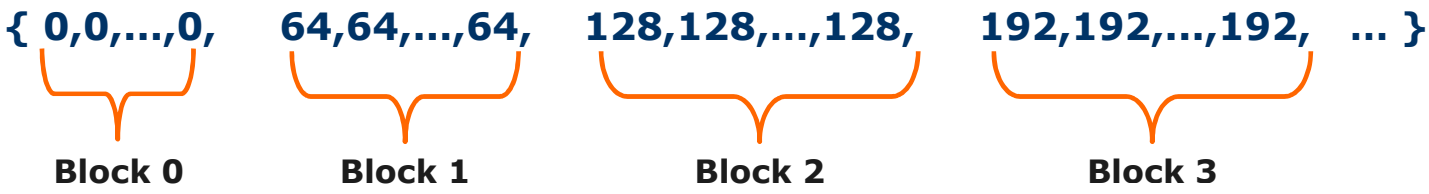
- Write pages in the following sequence
- **{ 0,0,…,0}**

  N= number of whole pages
- get the average merge cycle
  - Merge cycle = the # of page write requests between the nth merge operation and the (n+1)th merge operation
- Page-level mapping if the average merge cycle cannot determined or approximately same as N
- Block-level / Hybrid mapping if the average merge cycle is below the threshhold(20% of total # of pages)

Identification of the Mapping Schemes
Test Methods 2

❖ Block-level mapping?

- Feature of a modified block
- Write pages in the following sequence (e.g. 64 pages per block)



- Check if several consecutive merge operations are generated
  - If so, block-level mapping method
  - Otherwise, hybrid mapping method or modified block-level mapping method

Identification of the Mapping Schemes
Test Methods 3-1&2

❖ Log block associativity

- # of data blocks : # of log blocks = M : N

❖ TEST 3-1

- A log block set = a set of log blocks associated with a data block
- Find N = # of log blocks associated with a data block
  - Write a page continuously { 0, 0, 0, … , 0 }
  - Check the cycle of a merge operation
  - N = (the average merge cycle / # of pages per block) through the Test Method 1&2&3-1

Identification of the Mapping Schemes
Test Methods 3-1&2 (con'd)

❖ TEST 3-2

- Get all the data block sets
  - A data block set = A set of data blocks associated with a log block set

- Get all the data block sets, i.e., disjoint sets with UNION-FIND algorithm
  1. Make each data block a set
  2. For each two set, test if the two representative elements belong to same log block set
     1. If yes, UNION the two sets
  3. During the step 2, at least one UNION happen, then goto step 2

- Find M = # of data blocks associated with a log block set

Identification of the Mapping Schemes
Test Methods 4

❖ Number of log blocks

1. Assume that # of log blocks, N = 2
2. Fill the log blocks with data pages associative with the log blocks
3. If only N merge operations are generated
   1. N = N + 1
   2. Goto step 2.
4. If more than N merge operations are generated
   1. # of log blocks = N - 1

Experimental Tests on Real Devices

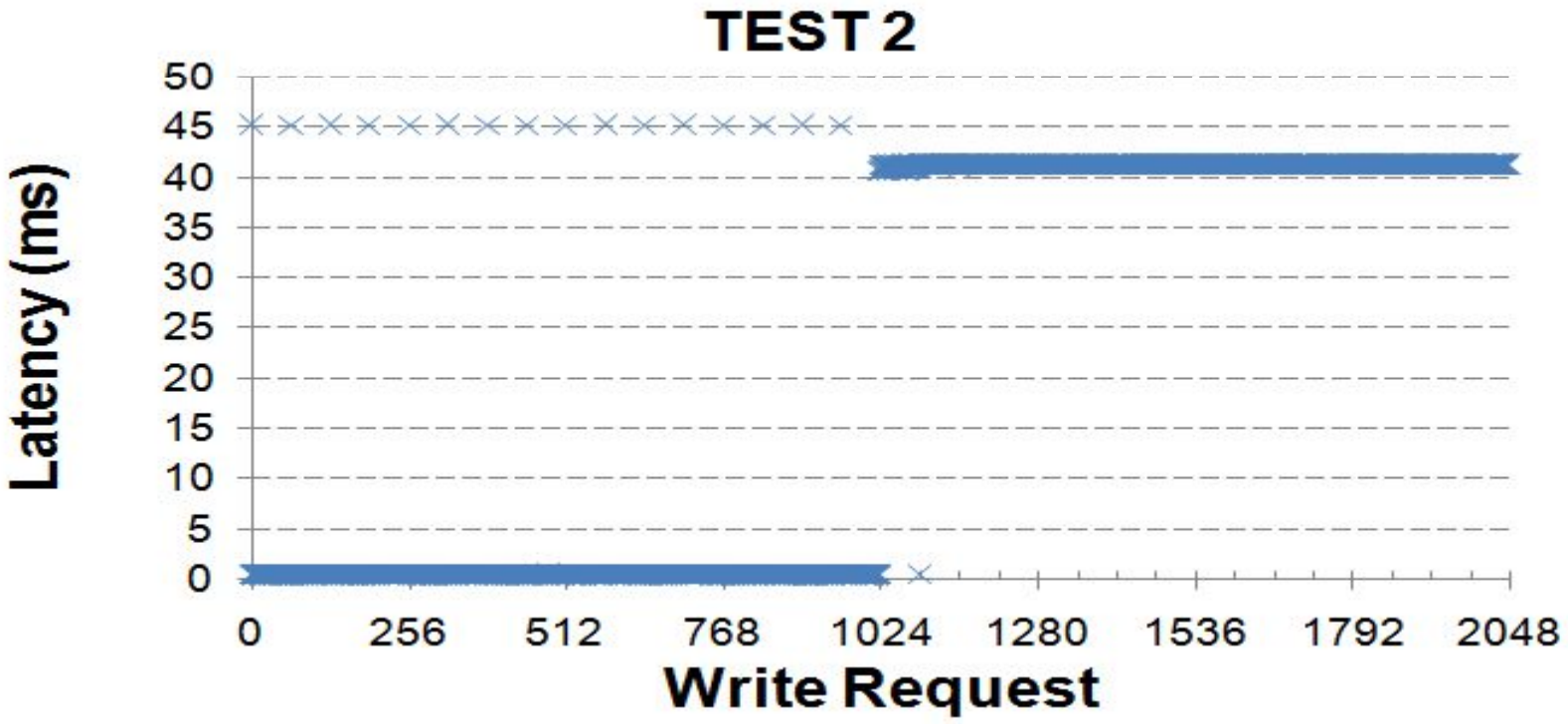
❖ Target devices

- Samsung USB Flash Drive SUB-1G
  - OTI's OTI002168-G controller
  - Samsung's K9K8G08U0A NAND flash memory (SLC)
- SKY digital Swing Solo 1G White
  - Silicon Motion's SM3210F controller
  - Hynix' HY27UG088G5M NAND flash memory (SLC)
- SKY digital Swing Solo 1G Black
  - Silicon Motion's SM3210F controller
  - Samsung's K9G8G08U0M NAND flash memory (MLC)
- Transcend
  - Samsung'sK9LBG08U0M NAND flash memory (MLC)

Experimental Tests on Real Devices (cont'd)

❖ Samsung 1G SLC

- TEST1 – Non page-level mapping
- TEST2 – Block-level mapping?
  - Logical blocks are divided by two regions
    - Region1: Hybrid-mapping scheme (0~15)
    - Region2: Block-level mapping scheme (16~)

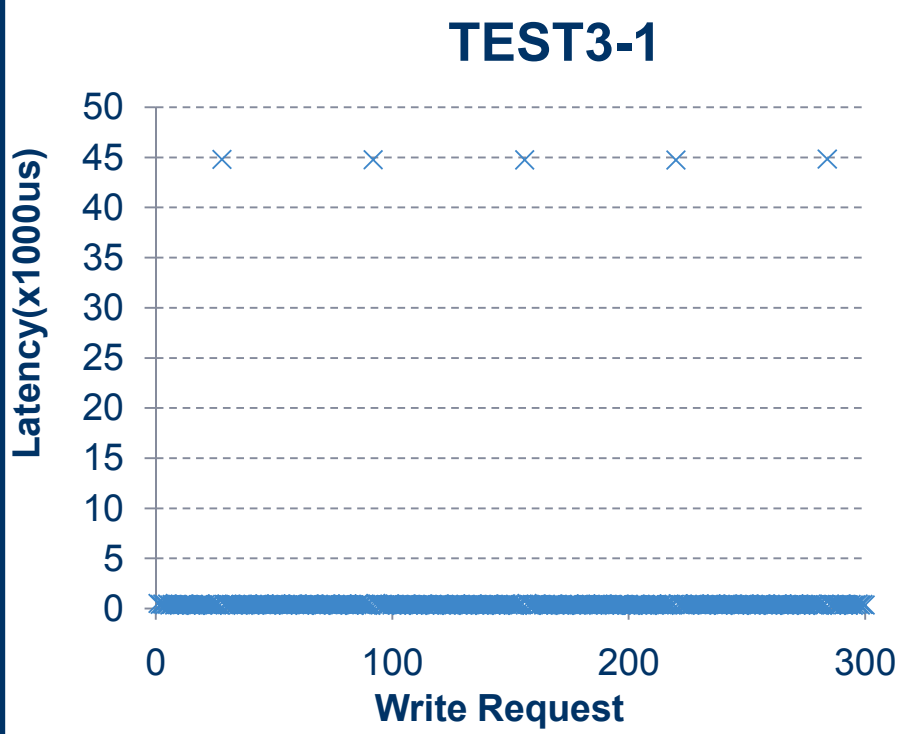


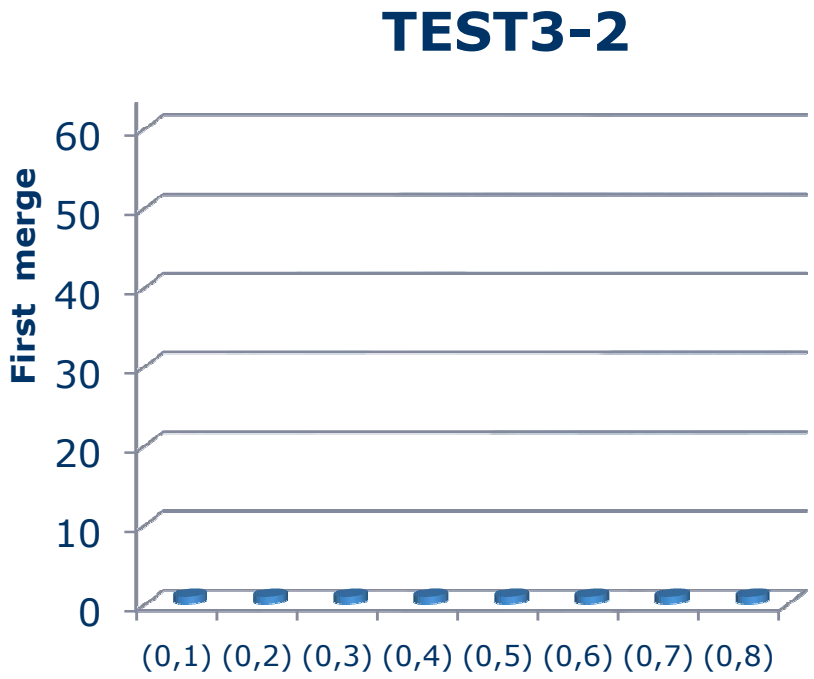
Experimental Tests on Real Devices (cont'd)

- **TEST 3/4 on Samsung 1G SLC (cont'd)**
- **(# data blocks : # of block blocks) = M:N**
  - TEST3-1 – # of log blocks associated with a data block
    - A log block set
      - a set of log blocks associated with a data block
    - N = the # of log blocks in a log block set
  - TEST3-2 - # of data blocks associated with a log block set
    - A data block set
      - A set of data blocks associated with a log block set
    - M = the # of data blocks in a data block set
    - If M:N = 1:1, Block-associative log block mapping
    - If M:N, Fully Associative log block mapping
  - TEST 4 – total # of log blocks
    - 16 log blocks for 16 data blocks

Experimental Tests on Real Devices (cont'd)

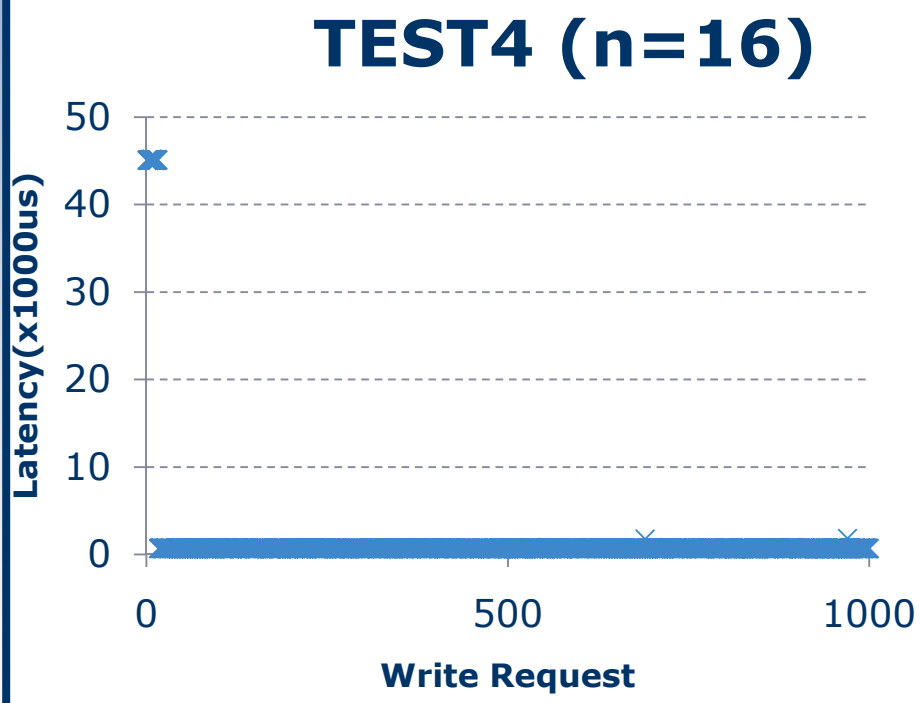
❖ TEST 3 on Samsung 1G SLC (cont'd)

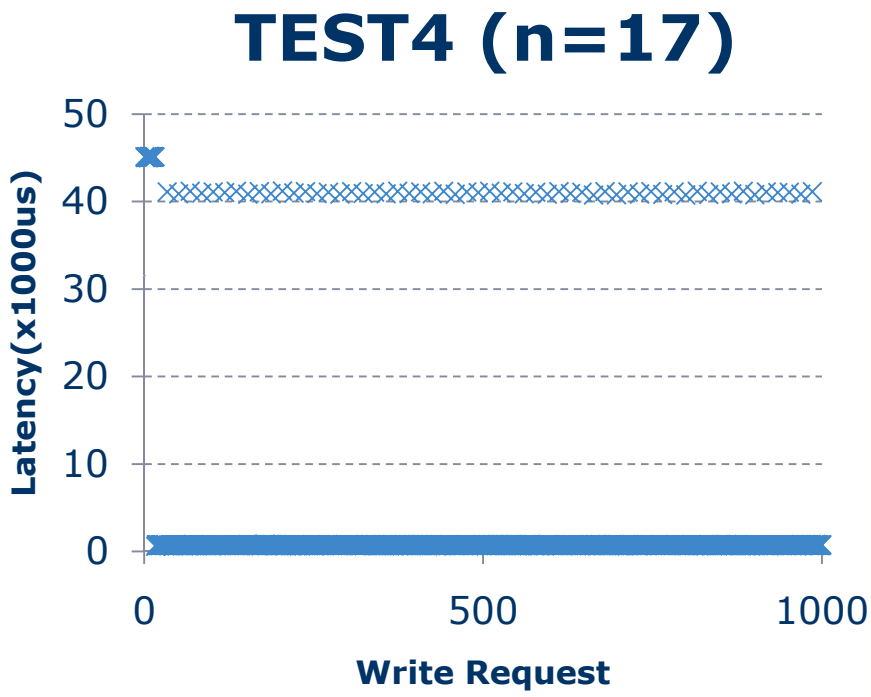




Experimental Tests on Real Devices (cont'd)

❖ TEST 4 on Samsung 1G SLC (cont'd)

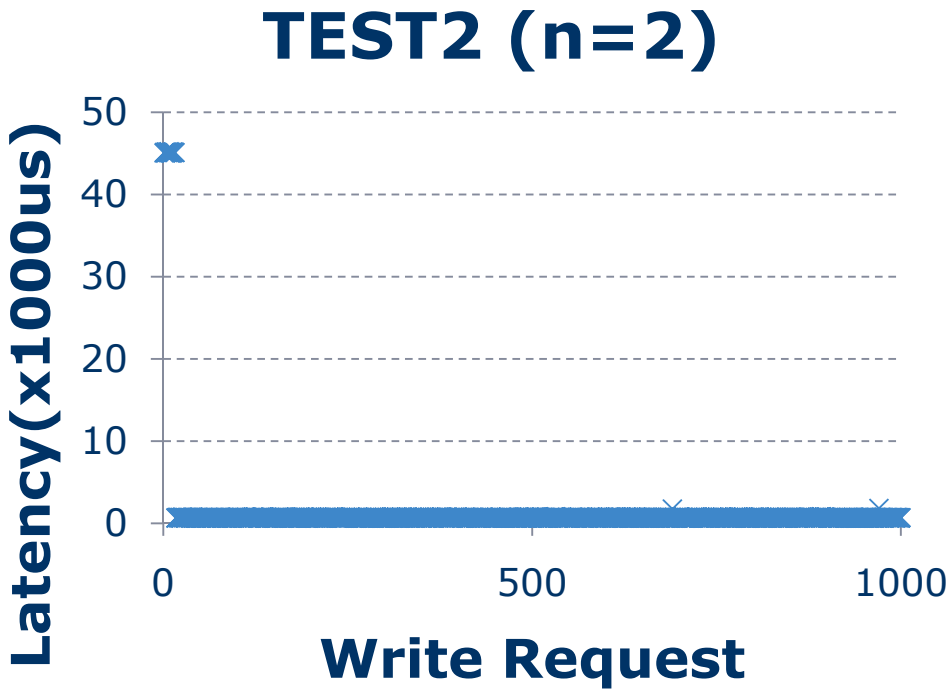




Experimental Tests on Real Devices (cont'd)

❖ SKY digital 1G SLC

- TEST1 – Non page-level mapping
- TEST2 – Block-level mapping?
  - Logical blocks are divided by two regions
    - Region1: Hybrid-mapping method (0 ~ 7)
    - Region2: Block-level mapping scheme (8 ~)

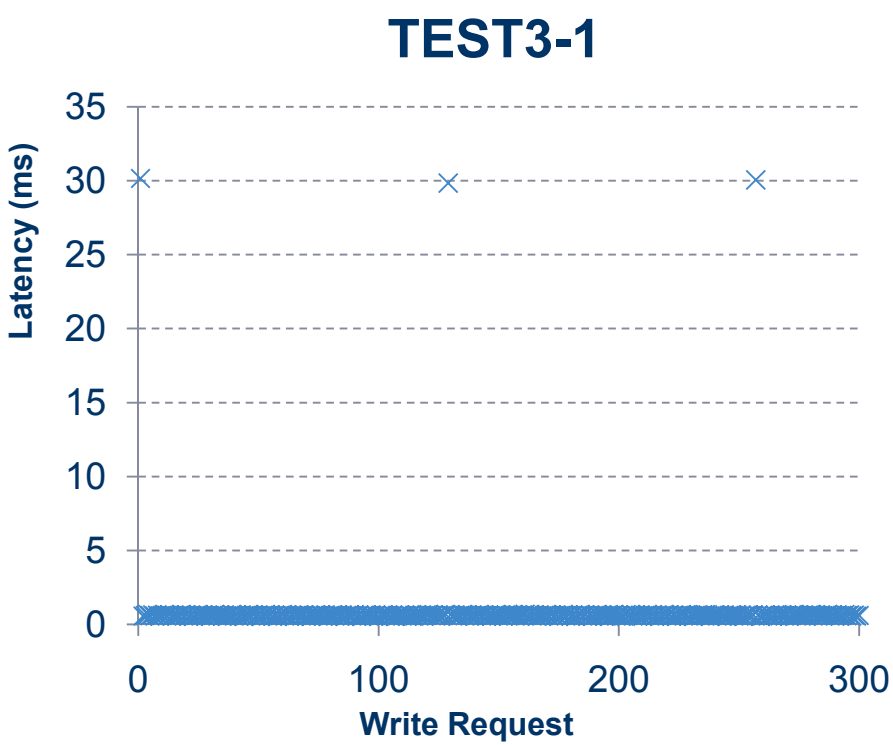


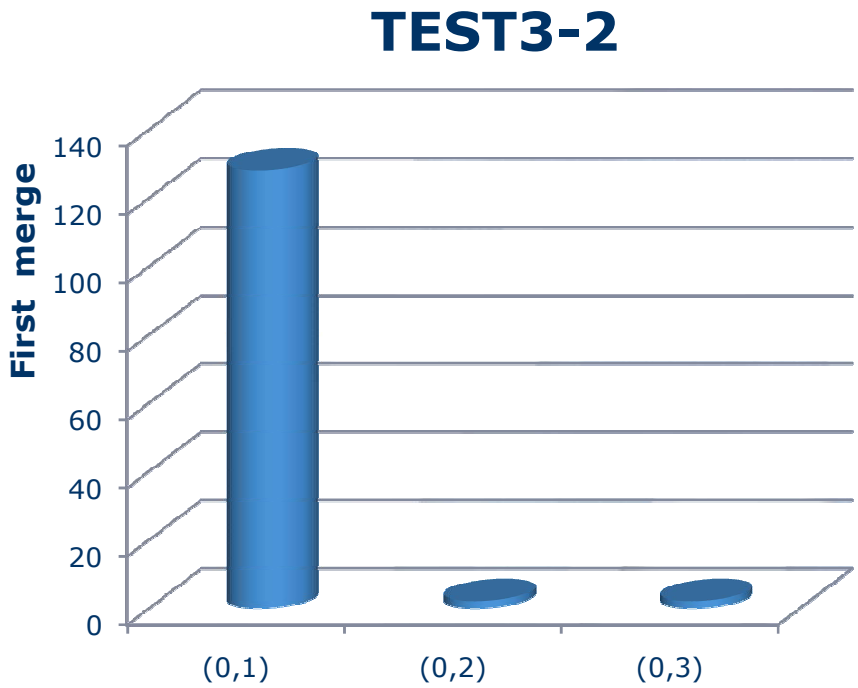
Experimental Tests on Real Devices (cont'd)

- **TEST3/4 on SKY digital 1G SLC**
- **(# data blocks : # of block blocks) = M:N**
  - TEST3-1 – # of log blocks associative with a data block
    - Merge cycle = 128 (2 blocks)
    - N = Log block set = 2 blocks
  - TEST3-2 - # of data blocks associative with a log block set
    - M = Data block set = 2 physically consecutive blocks
  - TEST 4 - # of log blocks
    - 2 log block sets for 4 data block sets
    - A block set consist of two physical blocks
    - 4 log blocks for 8 data blocks

Experimental Tests on Real Devices (cont'd)

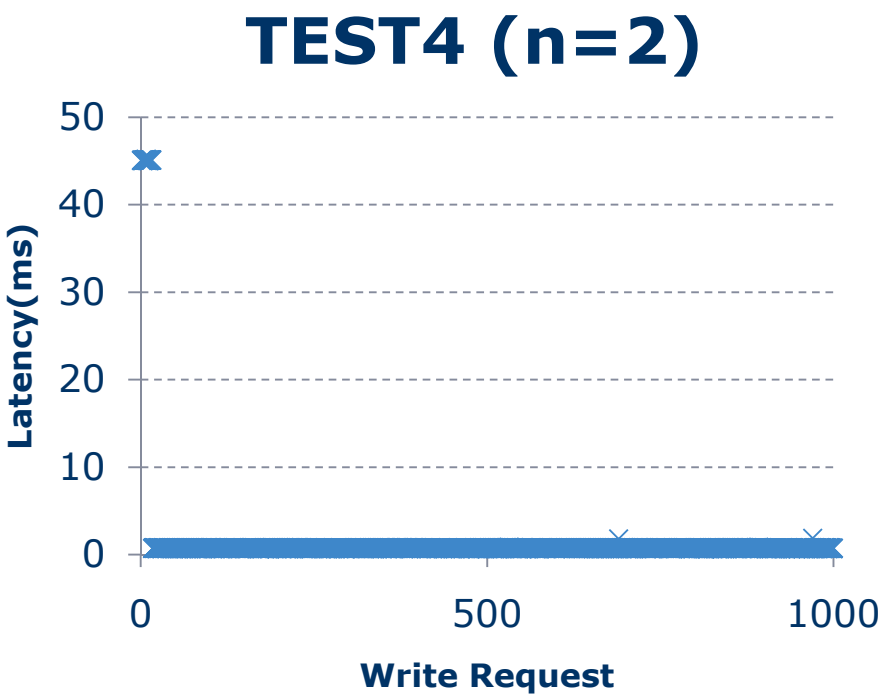
❖ TEST3 on SKY digital 1G SLC (cont'd)

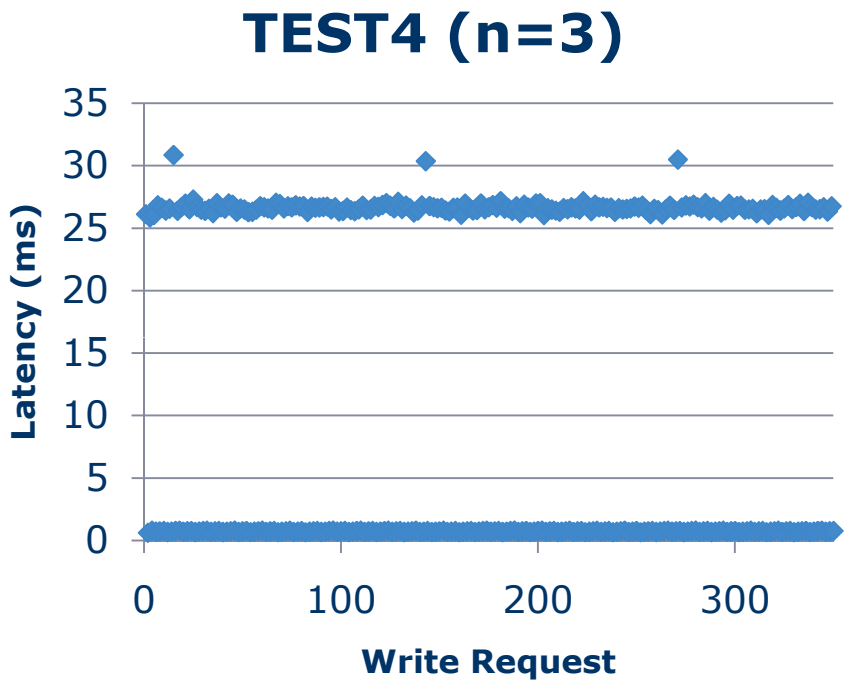




Experimental Tests on Real Devices (cont'd)

❖ TEST4 on SKY digital 1G SLC (cont'd)

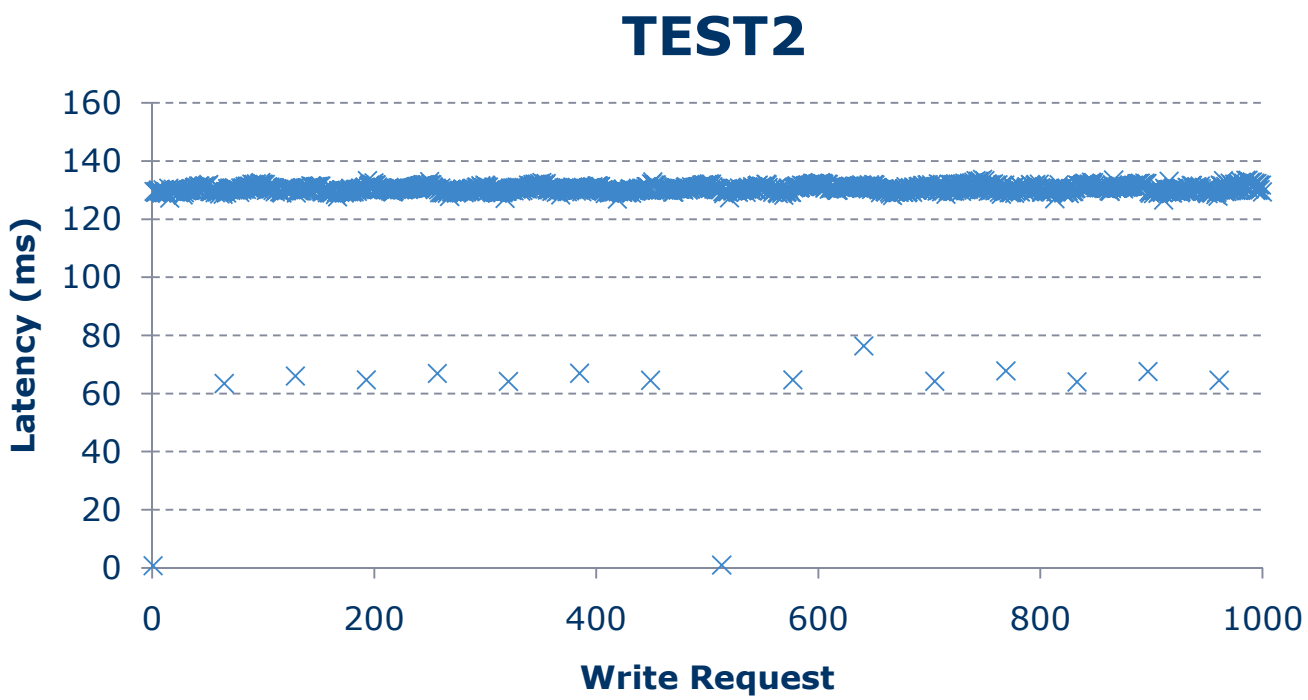




Experimental Tests on Real Devices (cont'd)

❖ SKY digital 1G MLC

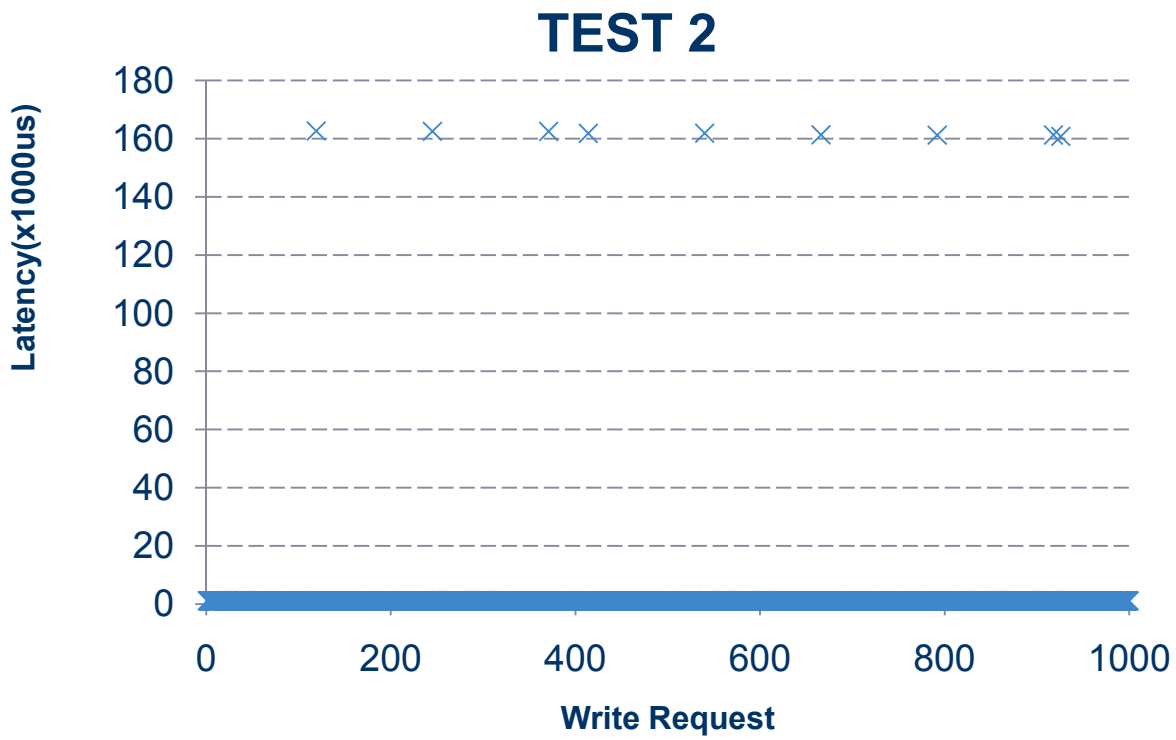
- TEST1 – Non page-level mapping
- TEST2 – Block-level mapping?
  - Every write operation generates merge
    - Simple block-level mapping scheme



Experimental Tests on Real Devices (cont'd)

Transcend 4G MLC

- 4K page NAND flash memory (128 pages per block)
- TEST1 – Non page level mapping scheme
- TEST2 – Block-level mapping?
  - All blocks are mapped by hybrid mapping scheme

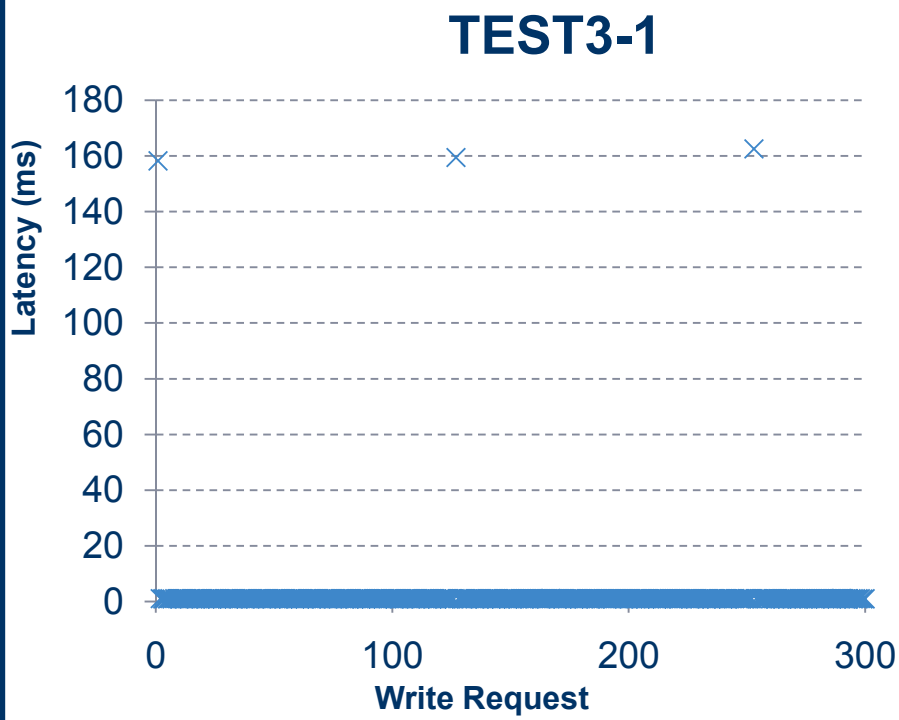


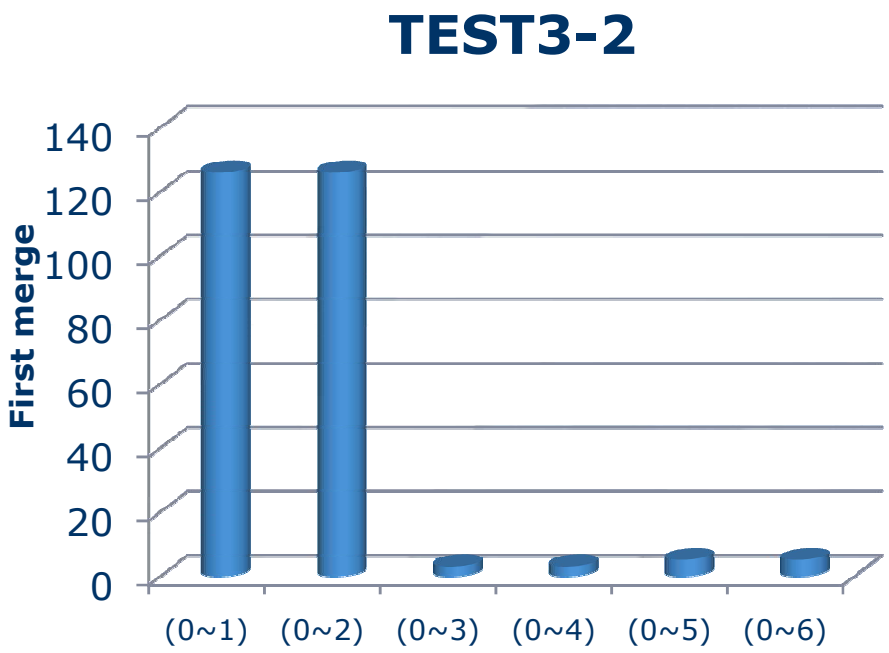
Experimental Tests on Real Devices (cont'd)

- **TEST3/4 on Transcend 4G MLC**
- **(# data blocks : # of block blocks) = M:N**
  - TEST3-1 – # of log blocks associative with a data block
    - N = Log block set = 1 block
  - TEST3-2 - # of data blocks associative with a log block set
    - M = Data block set = 3 block
  - TEST 4 - # of log blocks
    - 4 log blocks

Experimental Tests on Real Devices (cont'd)

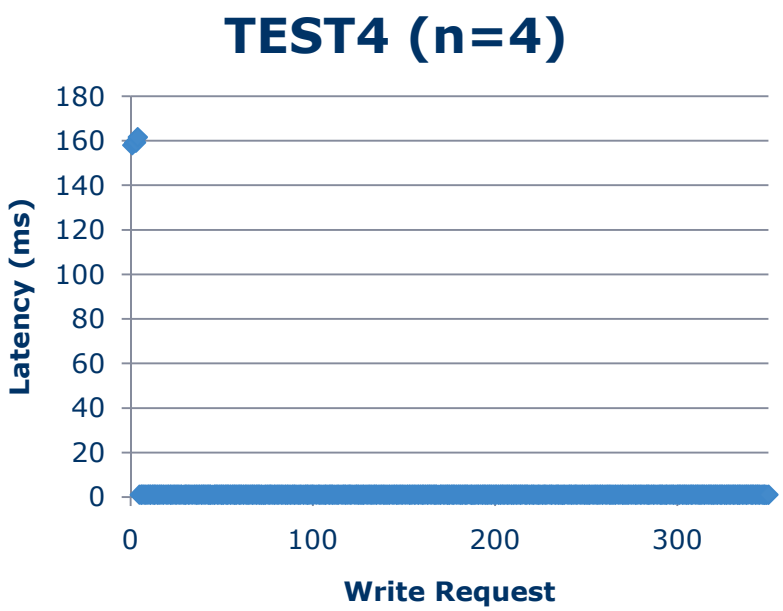
❖ TEST3 on Transcend 4G MLC

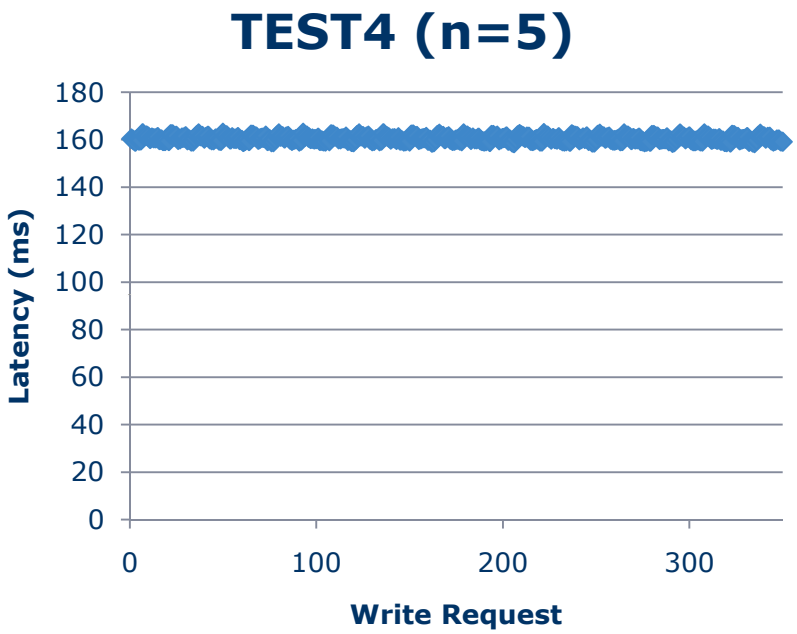




Experimental Tests on Real Devices (cont'd)

❖ TEST3 on Transcend 4G MLC

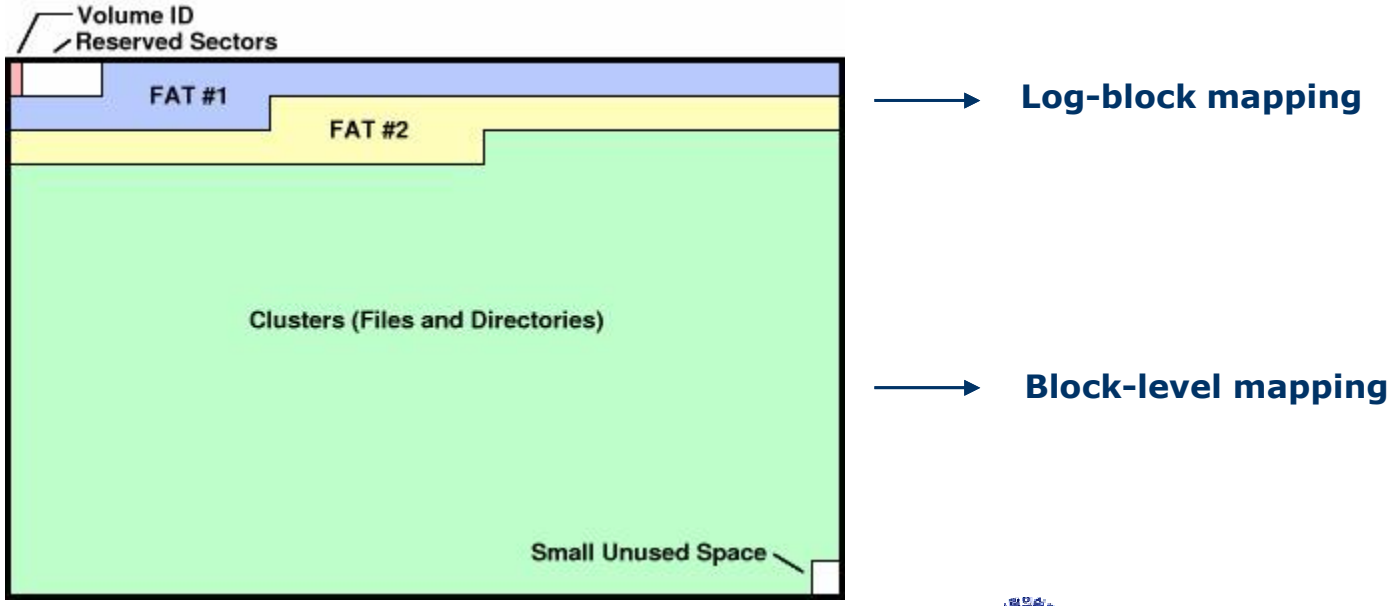




Experimental Tests on Real Devices (cont'd)

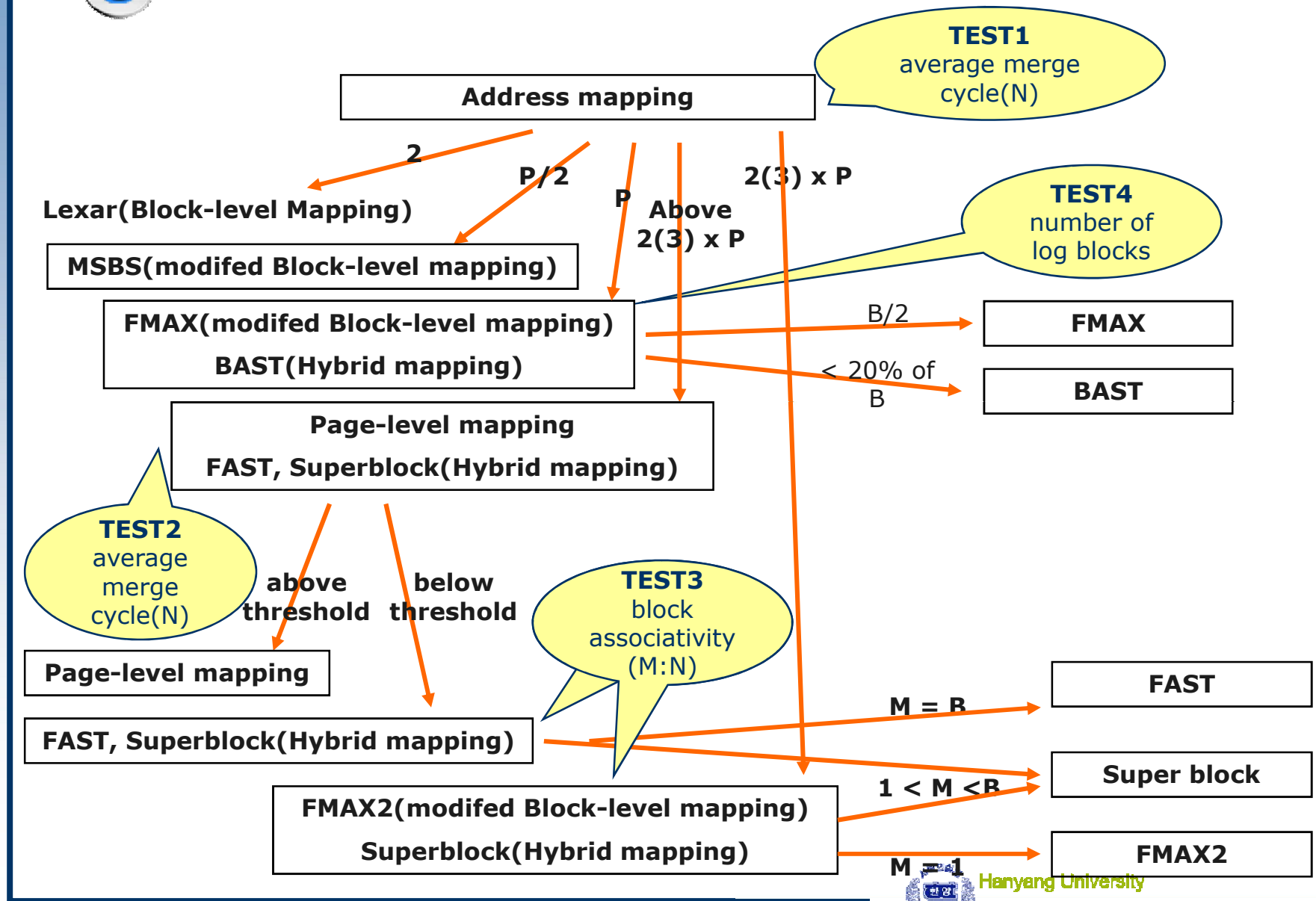
❖ Why does two separate regions exist?

- Samsung 1G SLC, SKY digital 1G SLC
- Generally, FAT filesystem is used for USB flash drives
- FAT filesystem
  - FAT area
    - Frequently accessed and modified
  - Data area
    - Sequentially accessed and rarely modified



Identification of the Mapping Scheme

new Classification of Major Mapping Schemes



Identification of the Mapping Schemes
new Test Methods 1

❖ What kind of Mapping is used in each block?

- Write pages in the following sequences for each block
  (e.g. if total # of pages per block is 64,
  then write the same page 64x7 times for each block)

{ 0,0,…,0, 64,64,…,64, 128,128,…,128, 192,192,…,192, … }

Block 0 **Block 1** **Block 2** **Block 3**

- For each sequence, get the average merge cycle
  - Merge cycle = the # of page write requests between the nth merge operation and the (n+1)th merge operation

- Partition whole area into sub-areas, each of which consists of consecutive blocks of same average merge cycle

❖ What Mapping for each block?

For each sequence (P: total # of pages per block),
switch(average merge cycle)

- Case 1: the Block-level mapping
- Case 2: the modified Block-level mapping, **Lexar**,
- Case P/2: the modified Block-level mapping, **MSB**S,
- Case P: the modified Block-level mapping, **FMAX**, or
  the Hybrid mapping, **BAST**, are used
- Case 2(or 3) x P:
  the modified Block-level mapping, **FMAX2**, or
  the Hybrid mapping, **Superblock**,
- **Default**: **Unknown**
  the Page-level mapping or
  the Hybrid mappings, **FAST / Superblock**,

❖ Page-level mapping?

- For each block with unknown address mapping
  - Write the first page of the block N times
  - If the address mapping for block 0 is unknown, then write page 0 N times: **{ 0,0,…,0}**

    N= number of whole pages
  - if the average merge cycle > threshold(e.g.:0.2 x N), then the Page-level mapping
    else the Hybrid mapping, FAST / Superblock is used
- Partition whole area into sub-areas, each of which consists of consecutive blocks of same average merge cycle

Identification of the Mapping Schemes
new Test Methods 3

❖ Log block associativity

- # of data blocks : # of log blocks = M : N
- Assumption:
  N = (the average merge cycle / # of pages per block)
  through the Test Method 1&2

❖ TEST 3

- Find a data block set associated with a log block set
- Each set have to have One and only one representative element
- Make all the data block sets, i.e., disjoint sets with UNION-FIND algorithm
  1. Make each data block a set
  2. For each two set, test if the two representative elements belong to same log block set
     1. If yes, UNION the two sets
  3. During the step 2, UNION's happen, then goto step 2
- Find M, # of data blocks associated with a log block set

Identification of the Mapping Schemes
new Test Methods 4

❖ Log block associativity

- # of data blocks : # of log blocks = M : N
- Assumption:

  N = (the average merge cycle / # of pages per block)
  through the Test Method 1&2

❖ TEST 4 : Number of log blocks

- The # of data block set = (the total # of data block / M)
- the total # of log block, L = the # of data block set * N
- If M:N = 1:1 and L < the threshold(20% of data blocks),
  then BAST
  else FMAX

Conclusion & Further Issues

- **Each USB flash device has its own mapping algorithm**
- **According to the address mapping algorithms,**
  - Performance differs
- **What about the SSD?**
- **Flash Drive/SSD abstraction is possible?**

Questions ?

Thank you !