PEN: Design and Evaluation of Partial-Erase for 3D NAND-Based High Density SSDs

Chun-Yi Liu, Jagadish B. Kotra, *Myoungsoo Jung, Mahmut T. Kandemir

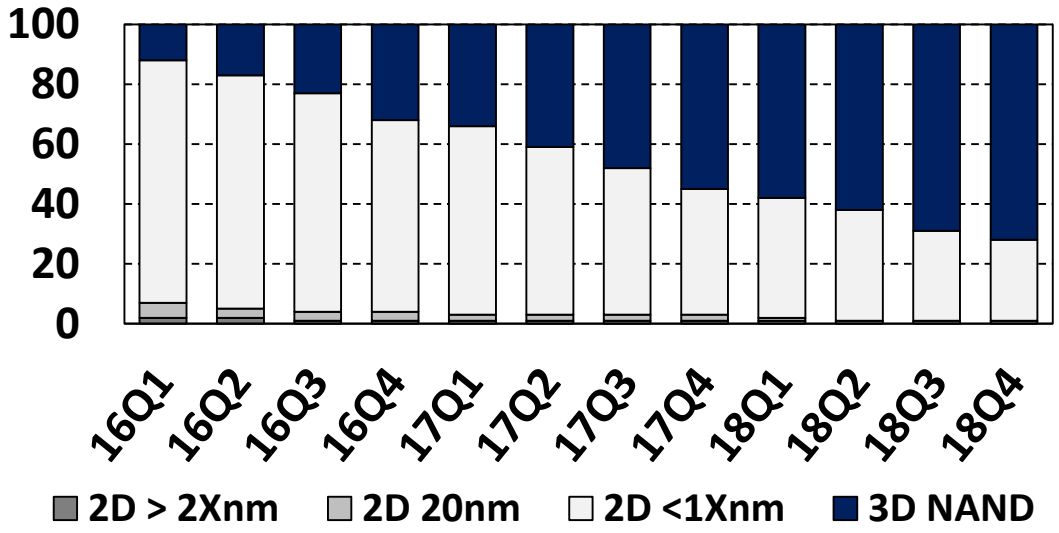
The Pennsylvania State University, *Yonsei University

Outline

- **Introduction and Background**
- **Motivation: Impact on Block Size**
- **Controller Hardware for Partial-Erase**
- **FTL for Partial-Erase**
- **Evaluation**
- **Conclusion**

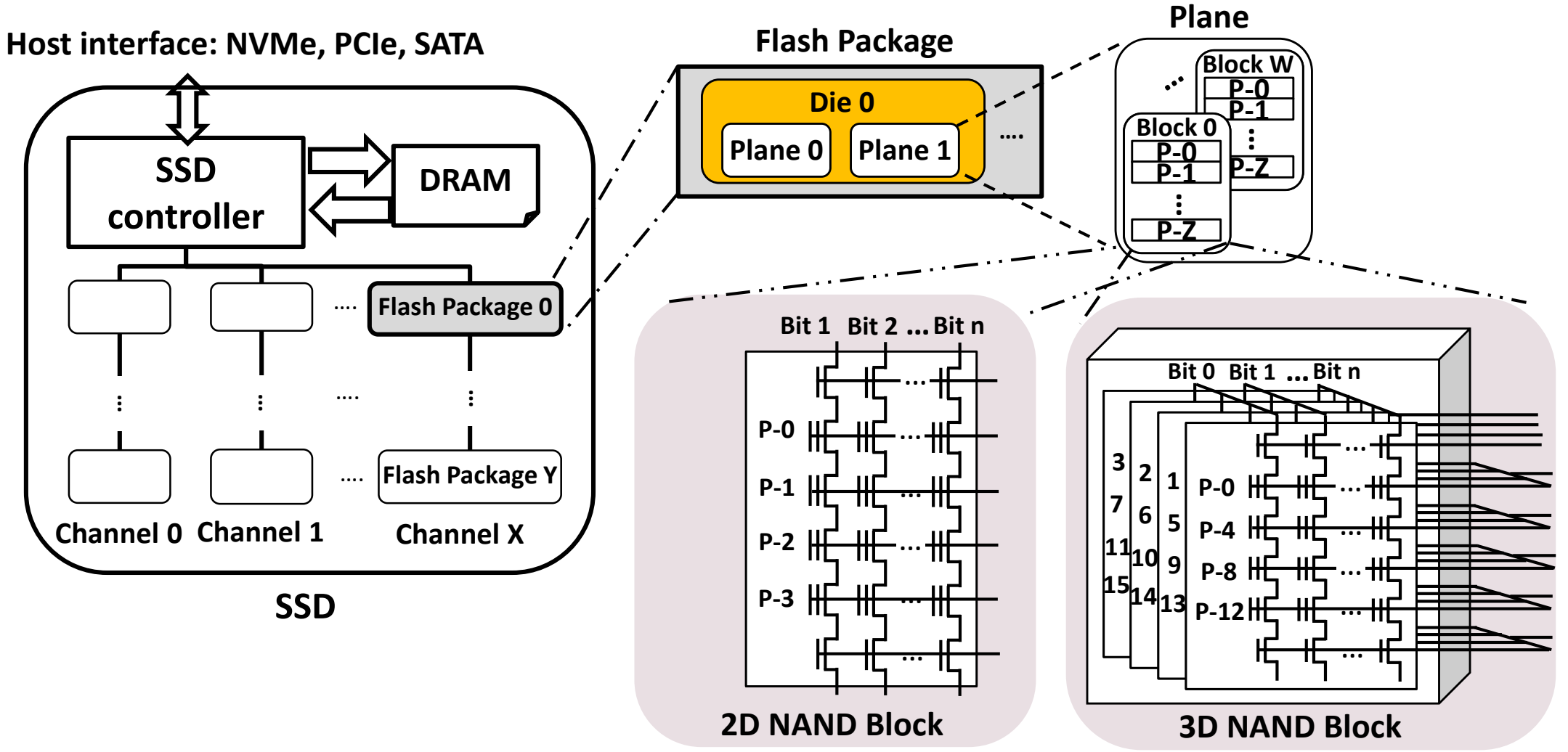
The trend of NAND Flash

- The issues of increasing 2D NAND flash density (smaller cells)
  - Reliability - various serious disturbances
    - Program (write) disturbances, read disturbances, and retention errors.
  - Performance – longer program (write) time
    - Cells are more sensitive to the program voltage.
- 3D NAND can highly mitigate those issues.
  - Larger stakability cells
  - Increasing layers to increase density.
    - 32 -> 48 -> 64 -> 96 layers
- 3D NAND will dominate MLC / TLC NAND market.

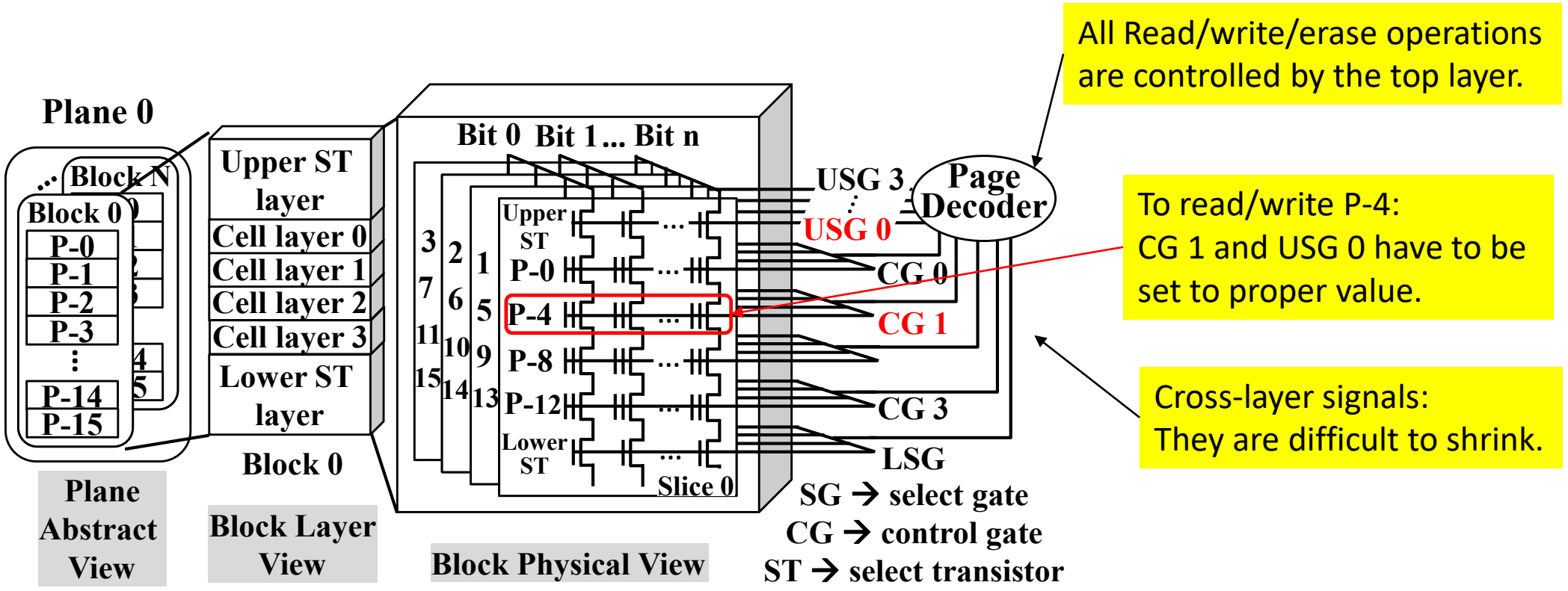


Overview of 2D/3D NAND-based SSDs

2D NAND dies are replaced by 3D NAND dies.



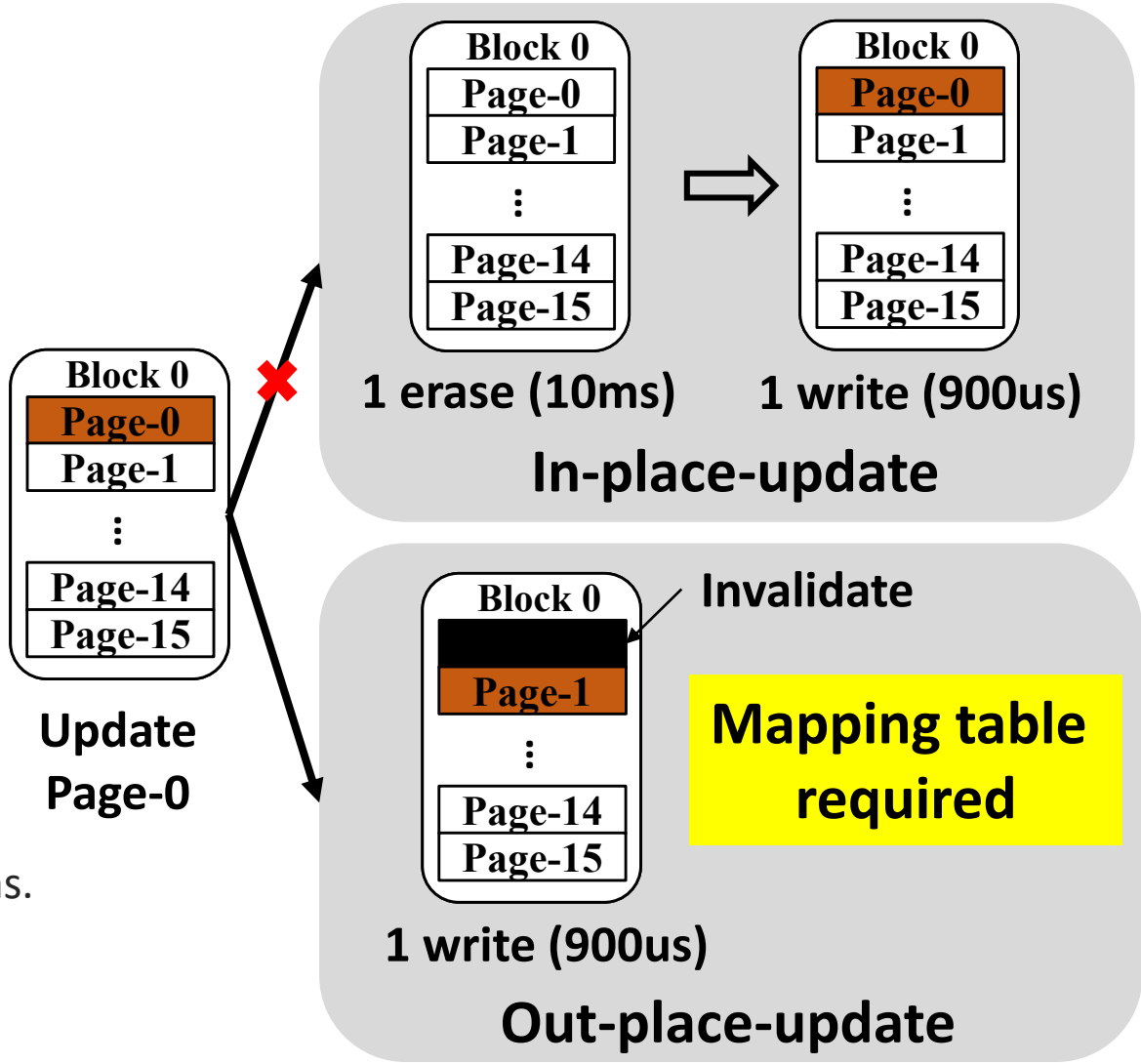
The Big Block Problem



Multiple slices have to share the control circuits due to cross-layer signals.
As the number of layers (n) increases, pages per page in 3D NAND increases in $O(n^2)$, not only O(n).

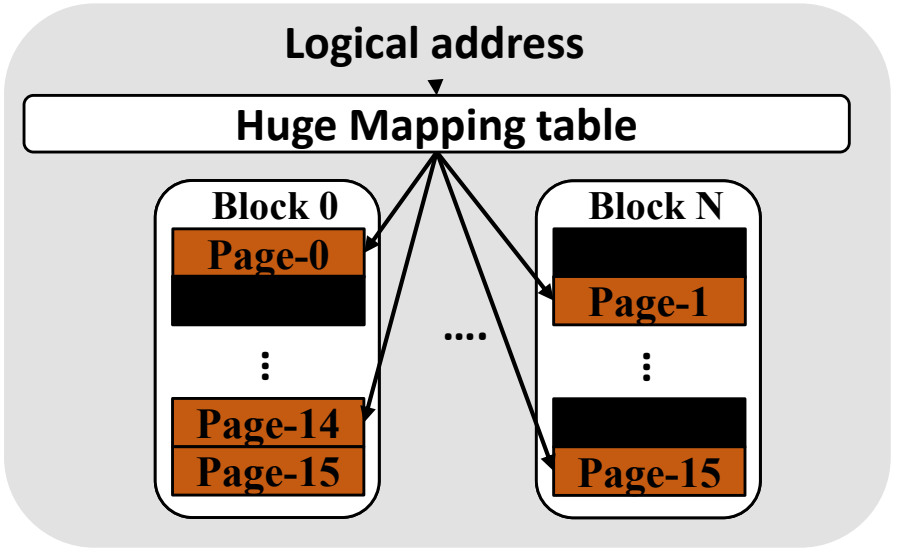
SSD Management Software: Flash Translation Layers (FTLs)

- NAND flash properties:
  - Read/Write operation: at a page unit
    - Typically, hundred of microsecond (us)
  - Erase: at a block unit
    - Typically, milliseconds (ms) or tens of ms
- Out-place-update:
- Flash Translation Layers (FTLs):
  - Address mapping
    - Maintain a mapping table
  - Garbage Collection (GC)
    - Different FTLs have their own GC algorithms.

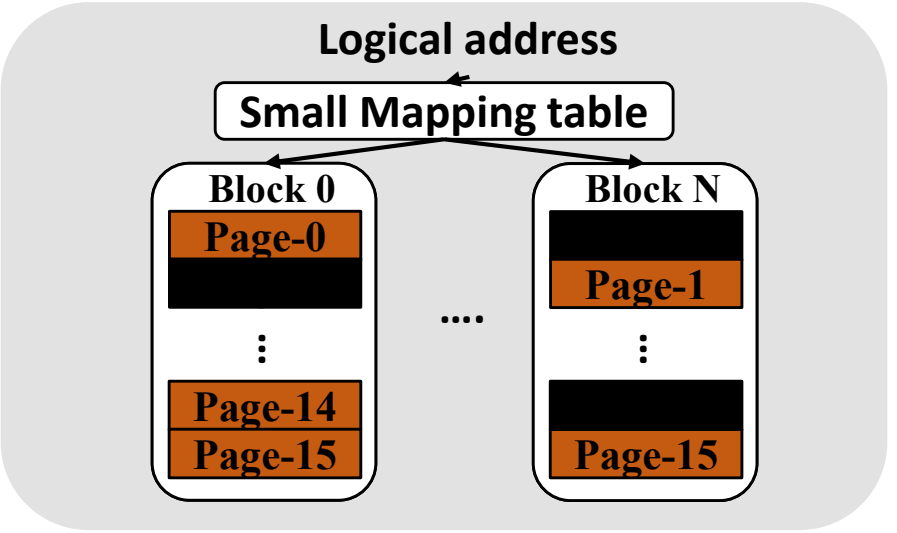


Flash Translation Layers (FTLs)

- Three categories of FTLs:
  - **Page-level mapping:**
    - Pro: providing the best performance
    - Con: huge mapping table required
      - 1 TB SSDs requires 1 GB mapping table.
  - **Block-level mapping:**
    - Pro: small mapping table
    - Con: performance degradation
    - A well-known implementation: **NFTL**
  - **Hybrid mapping:**
    - Combining the best of previous two mappings
    - We focus on the **Superblock FTL**.

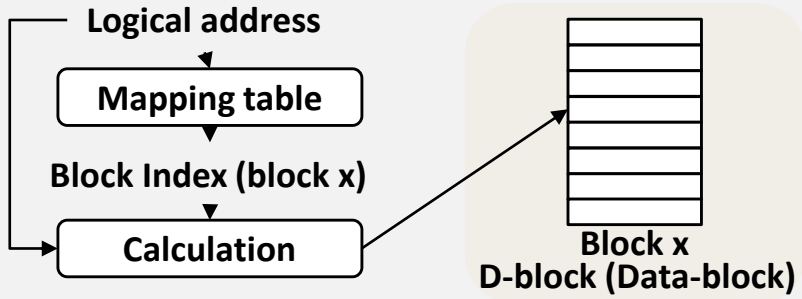


Page-level mapping

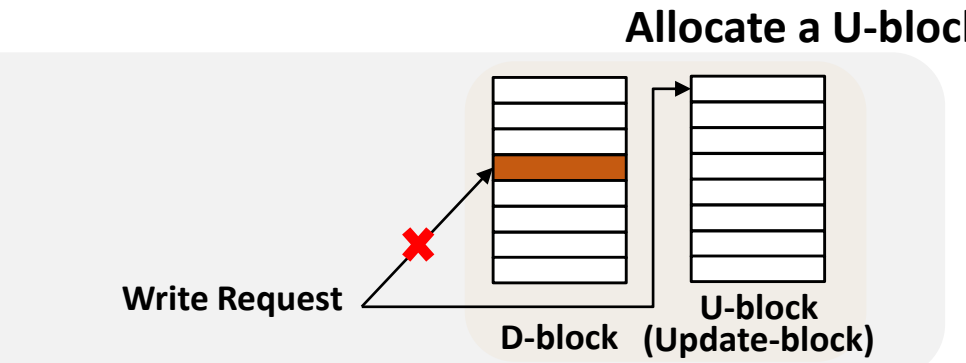


Block-level mapping

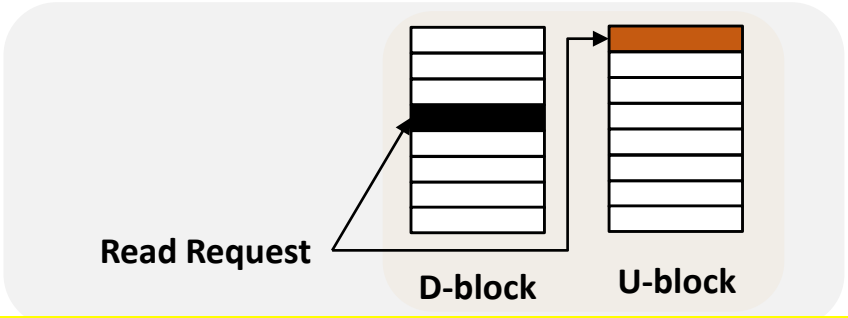
Introduction of NFTL



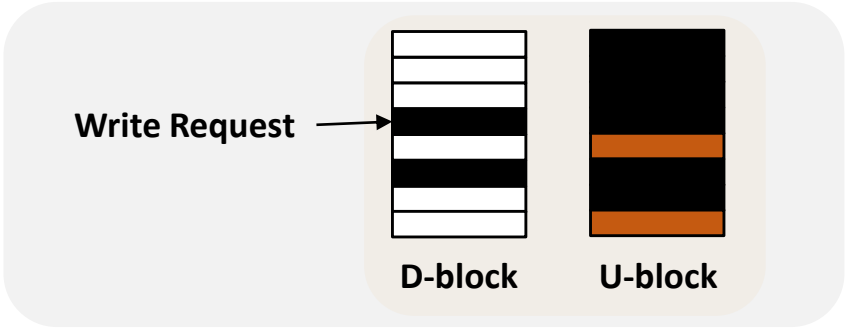
Write request to a new address



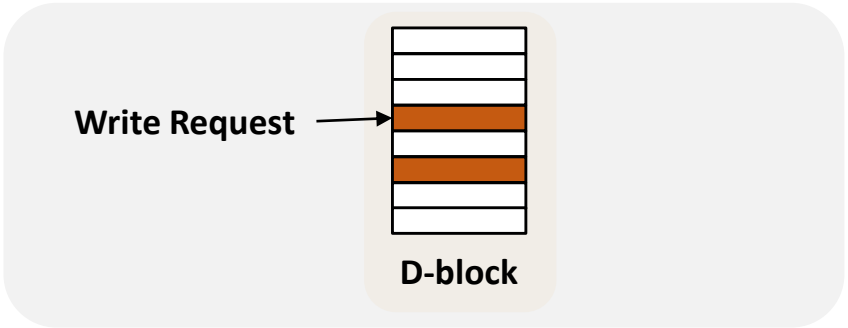
Update request to the same address
The data is updated (logged) in U-block.



Read request to the same address
(We may need to sequentially search U-block.)



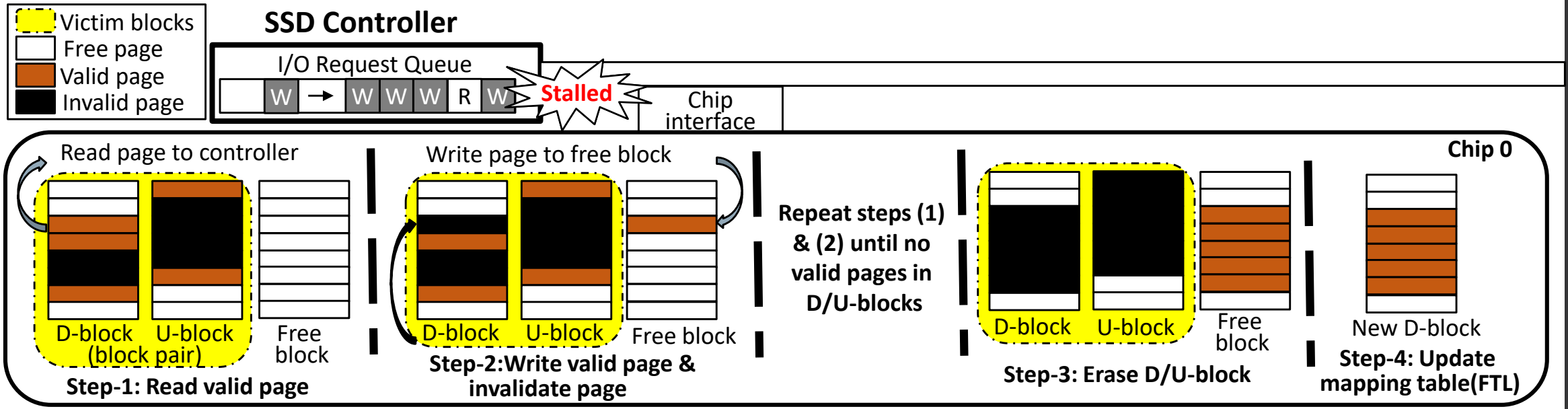
GC scenario 1: fully-utilized U-block



GC scenario 2: unpaired D-block
(The number of U-blocks is mush smaller than that of D-blocks.)

Introduction of NFTL (cont'd)

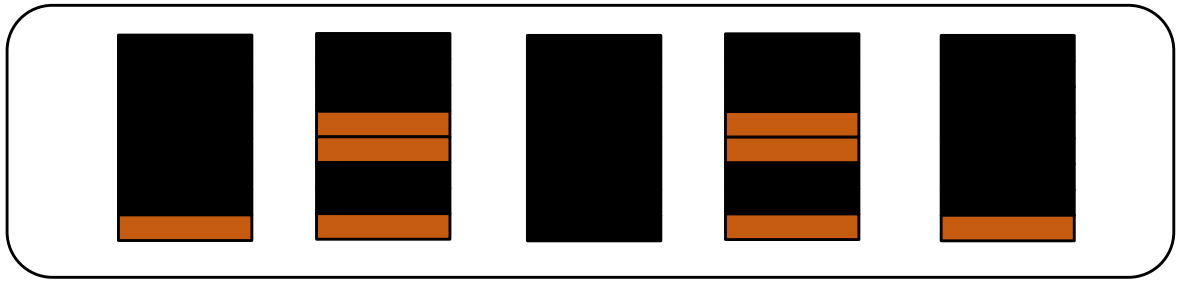
- NFTL GC (Merge) overview



The number of copied valid pages increase, as the number of pages per block increase.

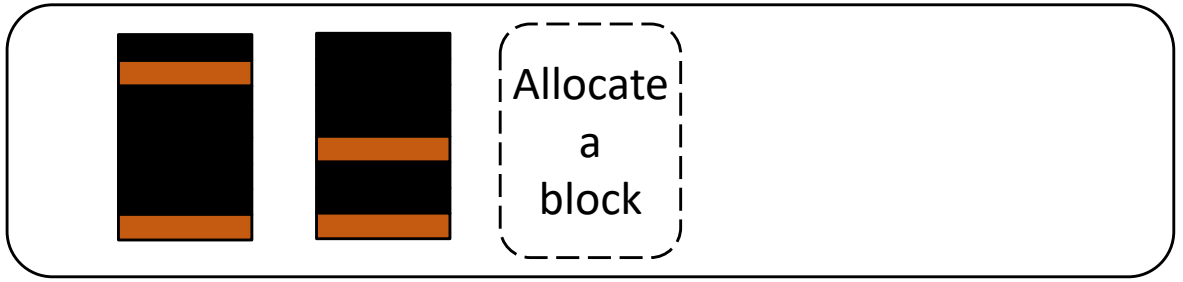
Introduction of Superblock FTL

- Superblock FTL GC overview
  - Intra-superblock GC (similar to fully-utilized U-block in NFTL)



A superblock

  - Inter-superblock GC (similar to unpaired D-block in NFTL)



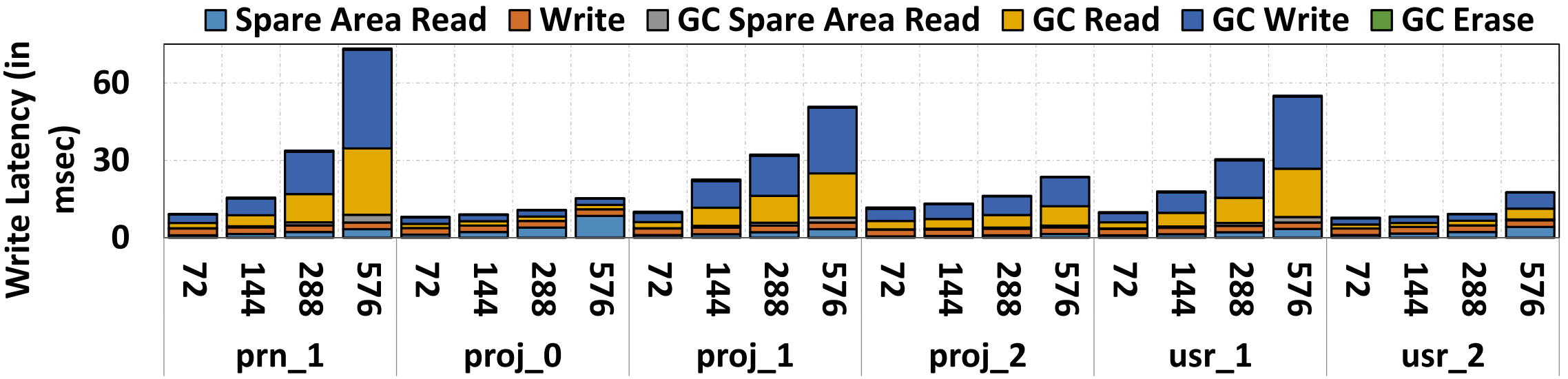
A superblock

Outline

- Introduction and Background
- **Motivation: Impact on Block Size**
- Controller Hardware for Partial-Erase
- FTL for Partial-Erase
- Evaluation
- Conclusion

Impacts on block size

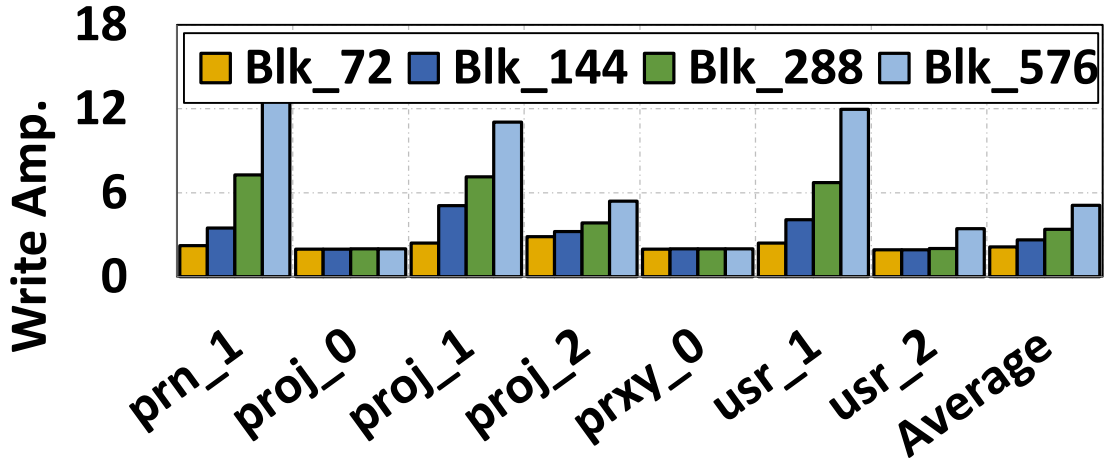
- We use four iso-capacity SSD configurations to show the impacts.
  - To prevent that the capacity affects the GC overheads (GC scenario 2 in NFTL)
- (blocks per plane, **pages per block**)
  - (15014, **72**), (7552, **144**), (3776, **288**), (1888, **576**)

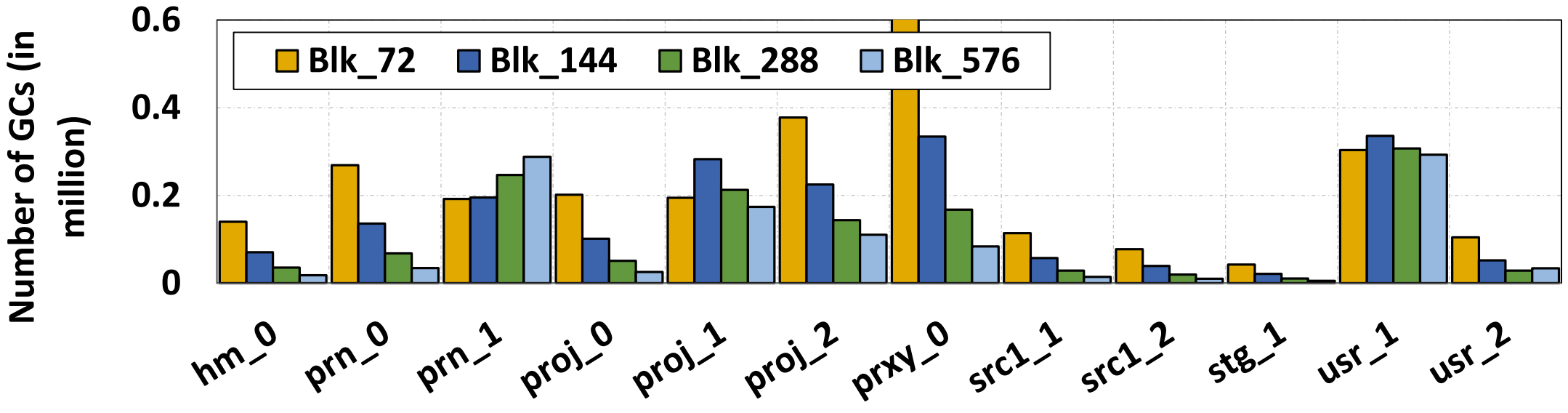


As number of pages per block increases, the time spending in reading/writing valid pages during GC increases.

Impacts on block size (cont'd)

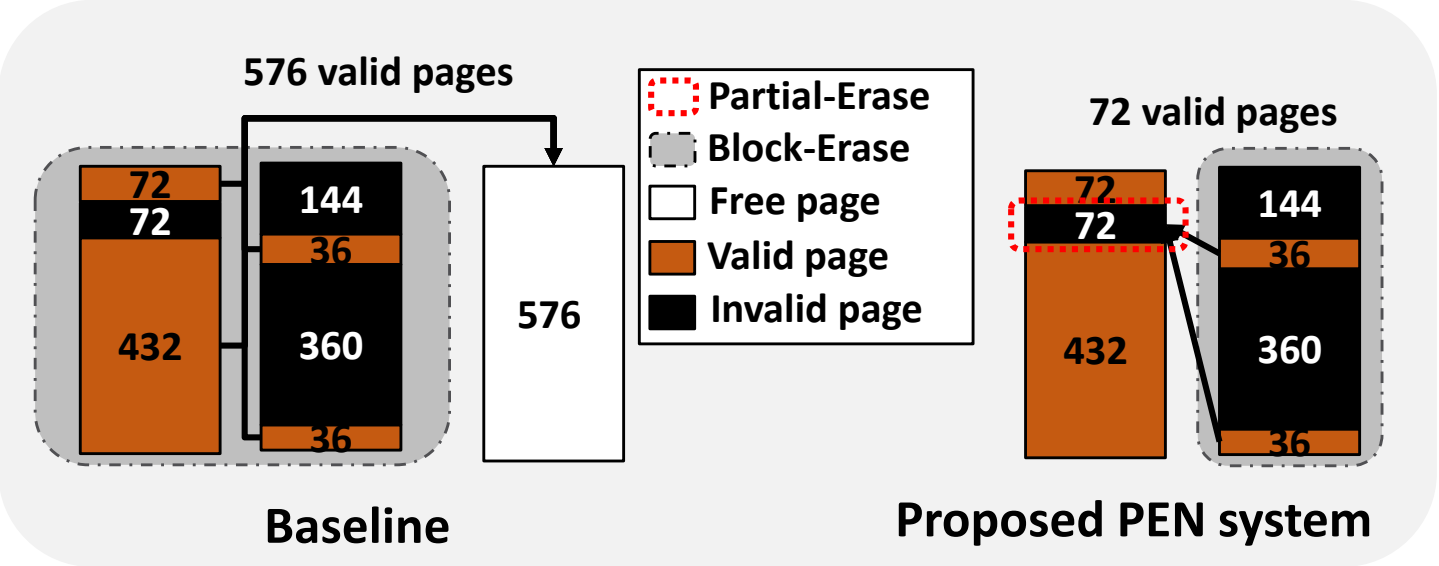
- The write amplifications slightly increase.
- GC triggered frequencies increase due to a few number of blocks.





Overview of Partial-Erase Operation

- We propose **PEN** (**P**artial-**E**rase for 3D **N**AND flash) to address the 3D NAND performance degradation.
  - The number of copied valid pages can be reduced.
- The PEN system contains two parts:
  - Hardware part:
    - Partial-erase operation
    - Indexing the partial-blocks
  - Software part:
    - FTL for partial-erase
      - M-merge
      - Program disturbance
      - Wear-leveling

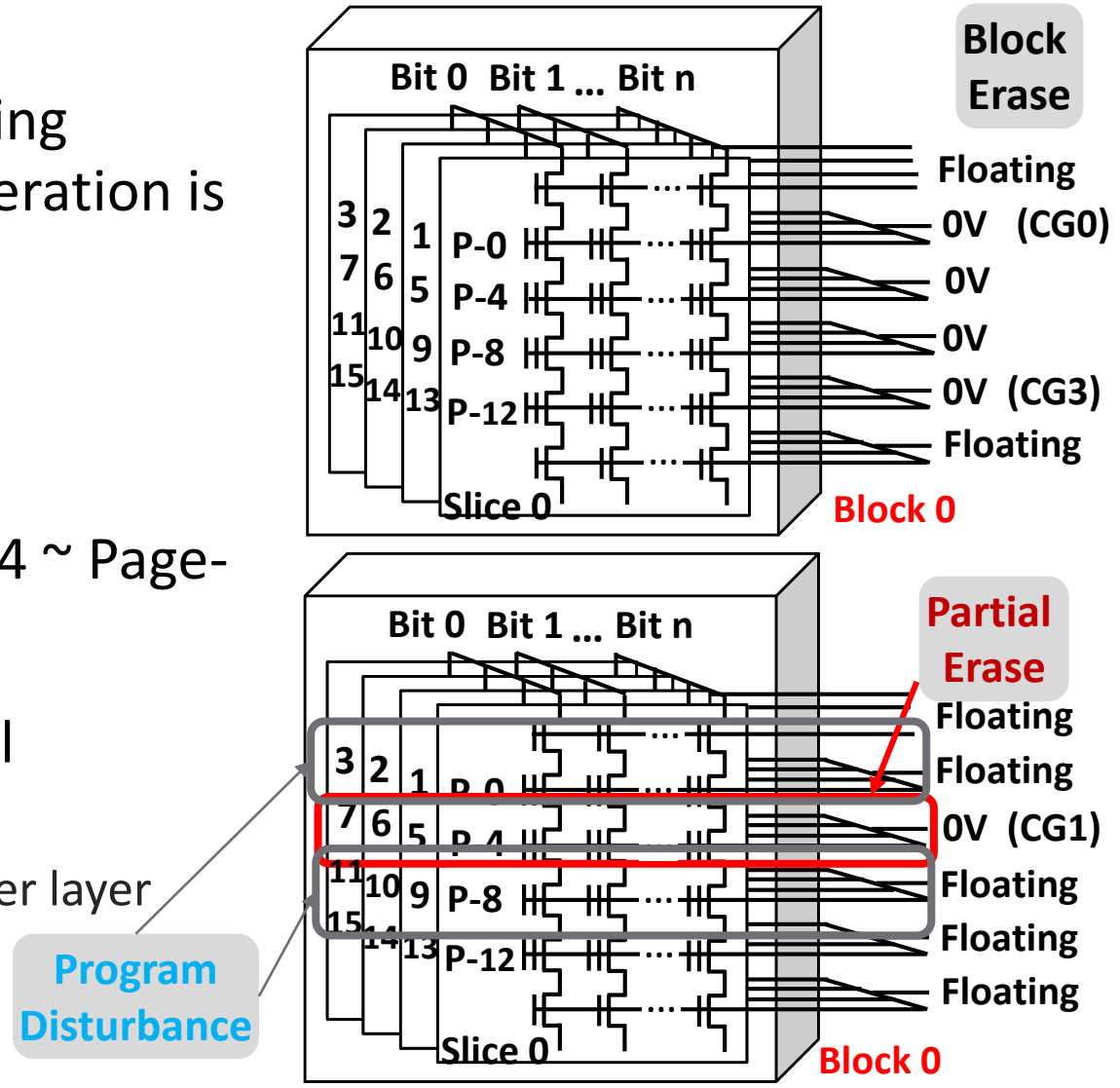


Outline

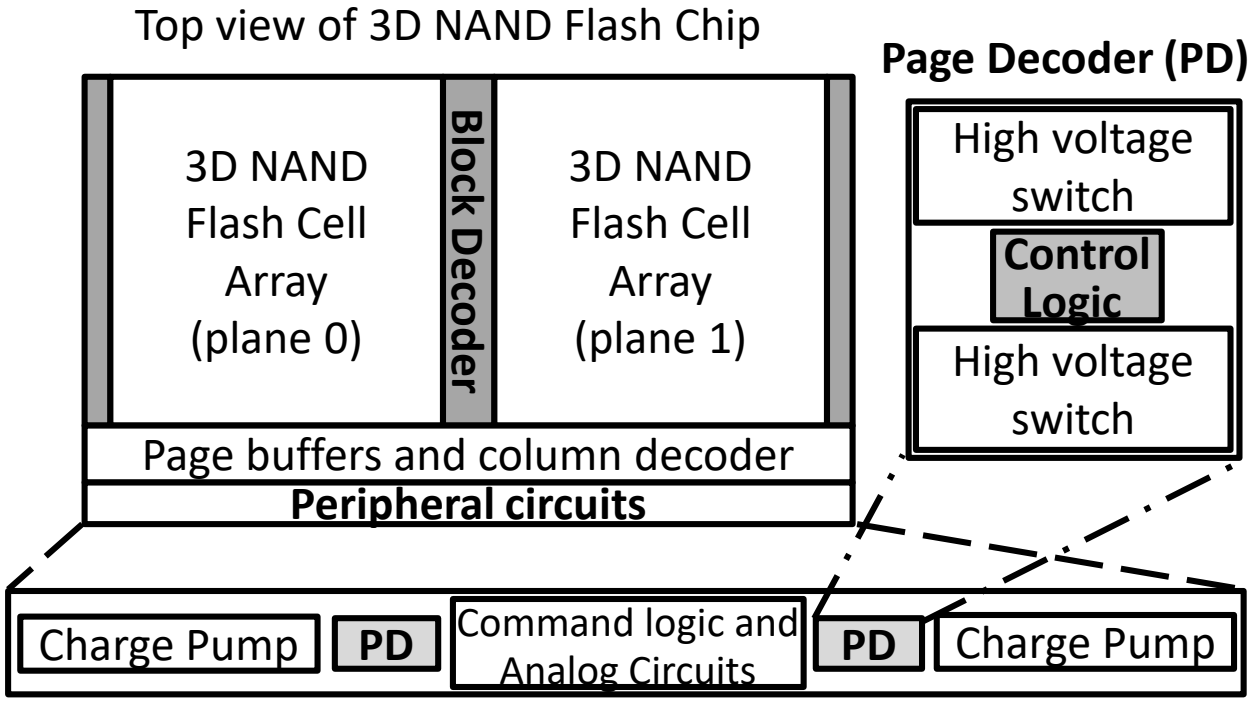
- Introduction and Background
- Motivation: Impact on Block Size
- **Controller Hardware for Partial-Erase**
- FTL for Partial-Erase
- Evaluation
- Conclusion

Controller Hardware for Partial-Erase

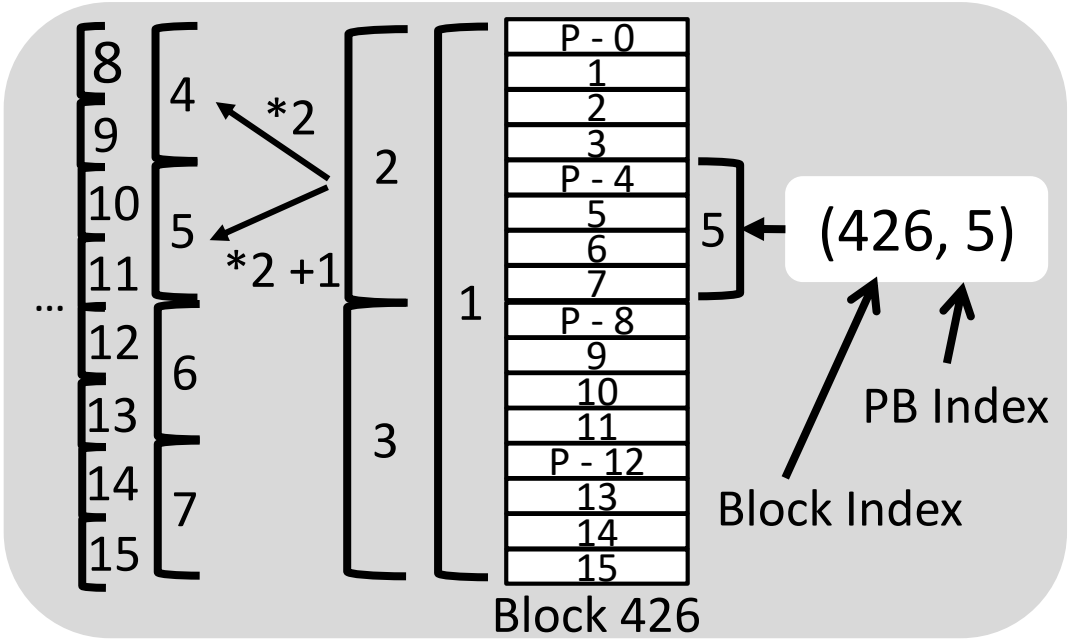
- PEN enables the partial-erase by setting proper control signals while erase operation is performed.
  - E.g. CG0~CG3 in the figure.
- The minimum unit is a layer of page.
- In the bottom figure, the layer (Page-4 ~ Page-7) controlled by CG1 is erased.
- The partial-erase introduce additional program disturbances.
  - E.g. upper layer (Page-0 ~ Page-3) and lower layer (Page-8 ~ Page-11).



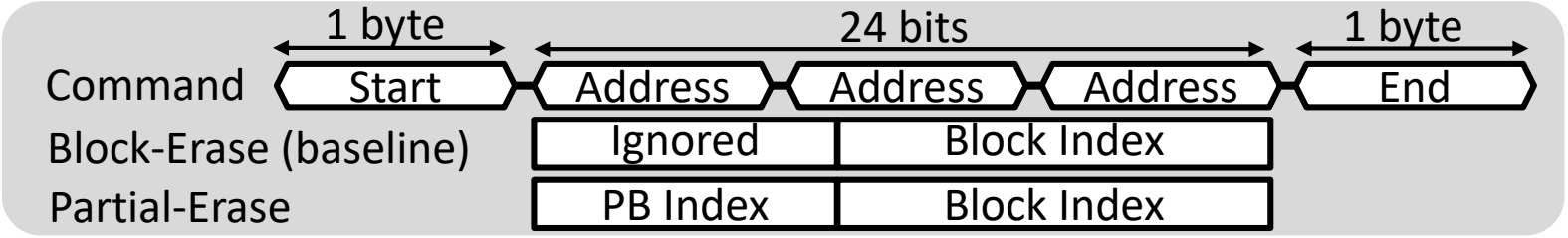
Hardware Overheads & Indexing the Partial Blocks



Control logics in page decoders(PD) need to be duplicated.



Partial-block index mechanism.



Partial-erase command format.

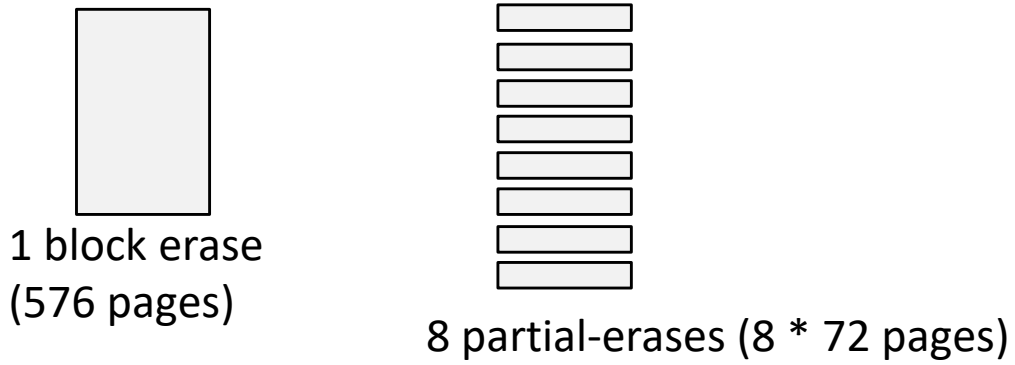
The latency of the partial-erase operation is only slightly faster than the block erase.

Outline

- Introduction and Background
- Motivation: Impact on Block Size
- Controller Hardware for Partial-Erase
- **FTL for Partial-Erase**
- Evaluation
- Conclusion

Partial Block vs. Smaller Block

- Pretending the block size is 72 pages, instead of 576 pages.
- The drawback of smaller block (72 pages) through the partial-erase.
  - 8x of the mapping table
    - Reducing the block size will increase the number of blocks.
  - Inefficient partial-erase operations

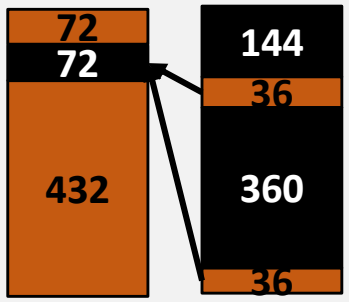


  - Not aware of disturbance

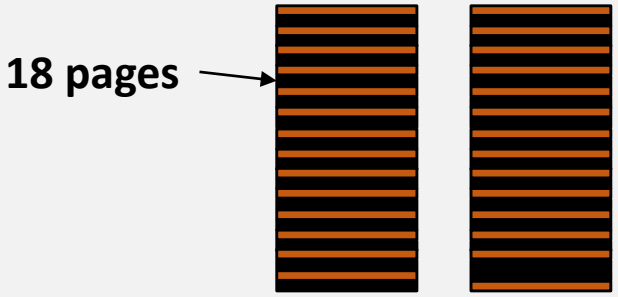
Different Block Pair Scenarios

How/Should we use the partial-erase in the following scenarios?

(1)

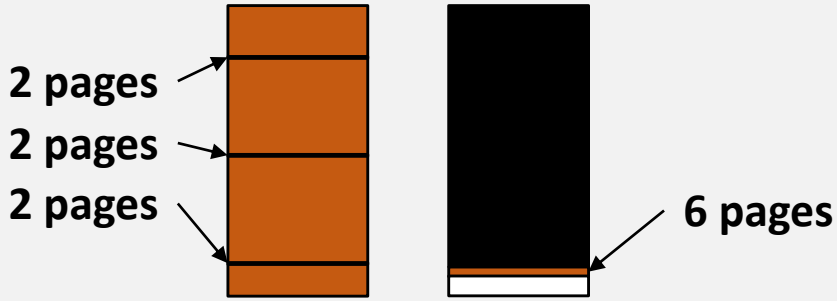


(2)



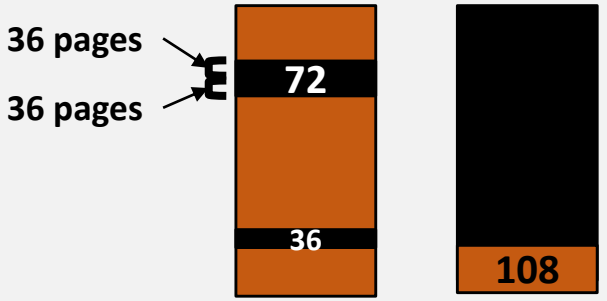
Partial-erases may not be beneficial.

(3) Assume that minimum partial-block has 18 pages.



Partial-erase may still introduce valid page copy.

(4)

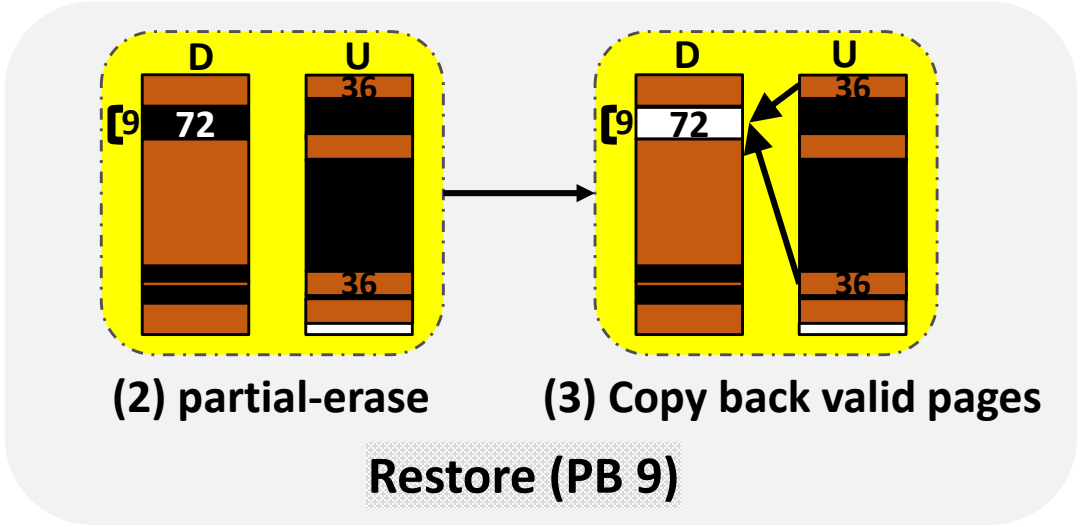


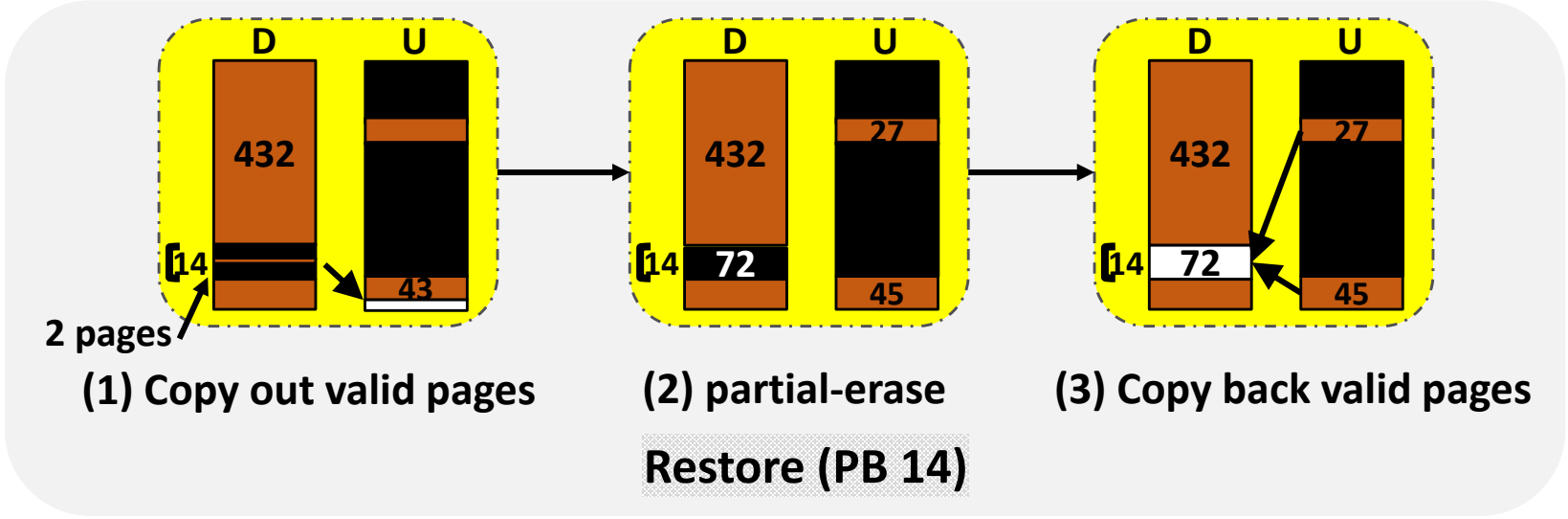
Should select 72 + 36 pages ,not 36 + 36 + 36 pages.

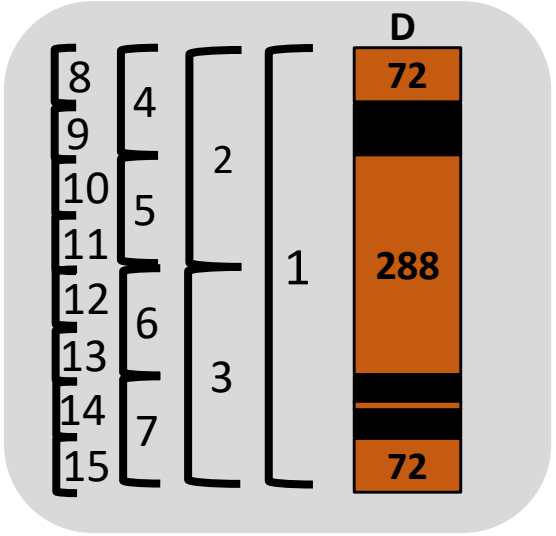
Deciding partial-block size is important.

M-Merge (Modified-Merge) Algorithm

- Restore operation:
  - (1) copy out valid pages
  - (2) **partial-erase** operation
  - (3) copy back valid pages

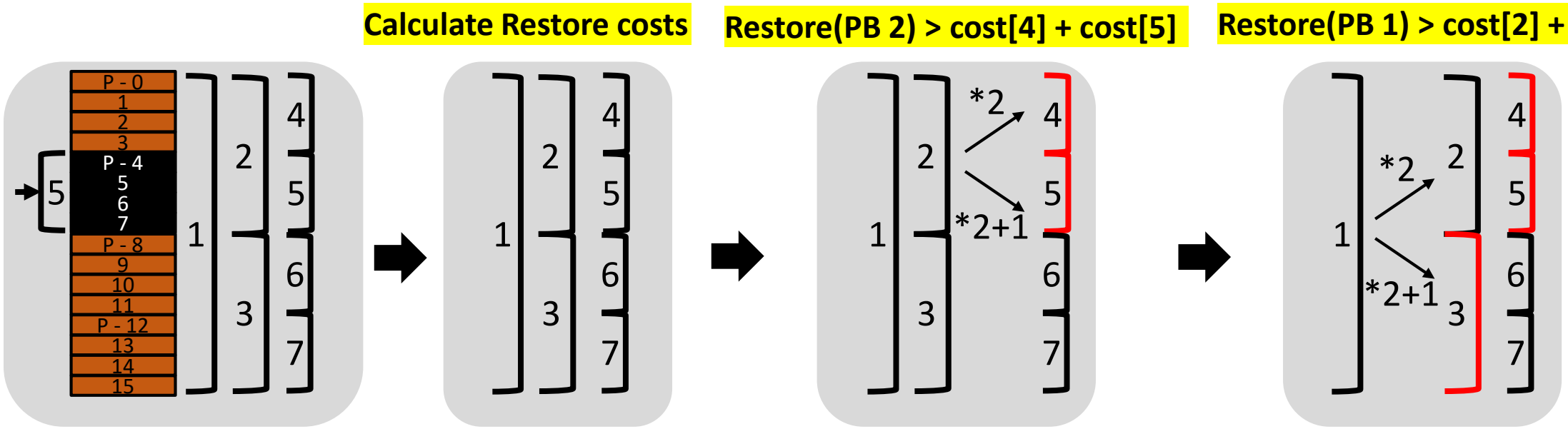






M-Merge Algorithm (cont'd)

- Recursive relation to decide the restored partial-blocks (pb)
  - $cost[pb] = \min(restore(pb), cost[pb * 2] + cost[pb * 2 + 1])$
  - If (pb = the smallest PB) $cost[pb] = restore(pb)$
- Check whether the Cost(M-Merge) < cost(Merge)

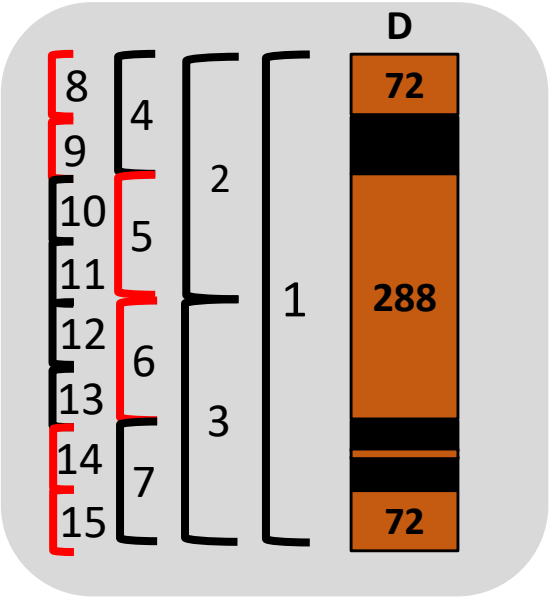


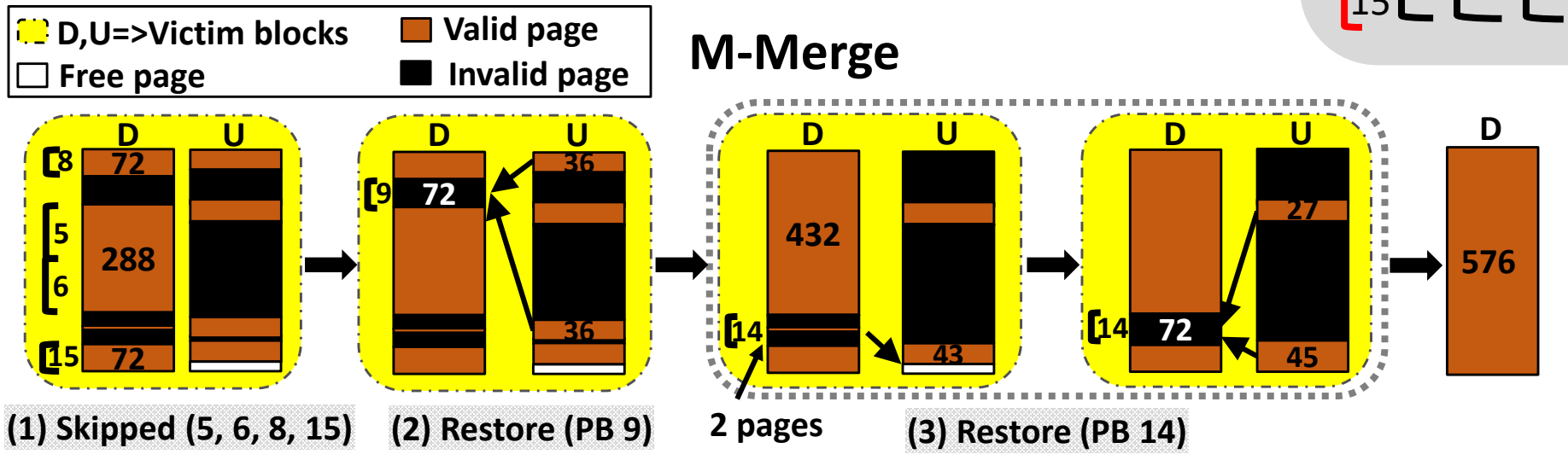
Restore(PB 1) = 12 copies + 1 erase + 16 copies
Restore(PB 2) = 4 copies + 1 erase + 8 copies
Restore (PB 5) = 1 erase + 4 copies
Restore (other PBs) = 0

Cost[PB 1] is the final cost of the M-merge.
Merge algorithm will restore PBs 3, 4, 5 to complete merge.
Note restore(PB 3) and restore(PB 4) are 0.

M-Merge for Block-level Mapping (NFTL)

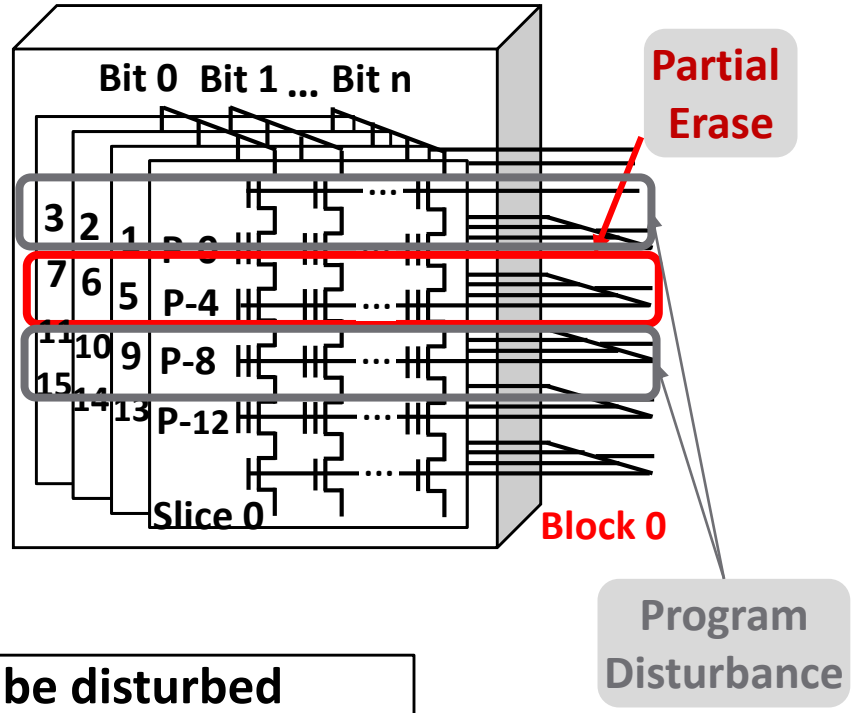
- M-Merge use the recursive relation to decide the restored partial-block.
  - PBs 5, 6, 8, 9, 14, and 15
- Apply restore to those PBs
  - PBs 5, 6, 8, and 15 can be skipped.

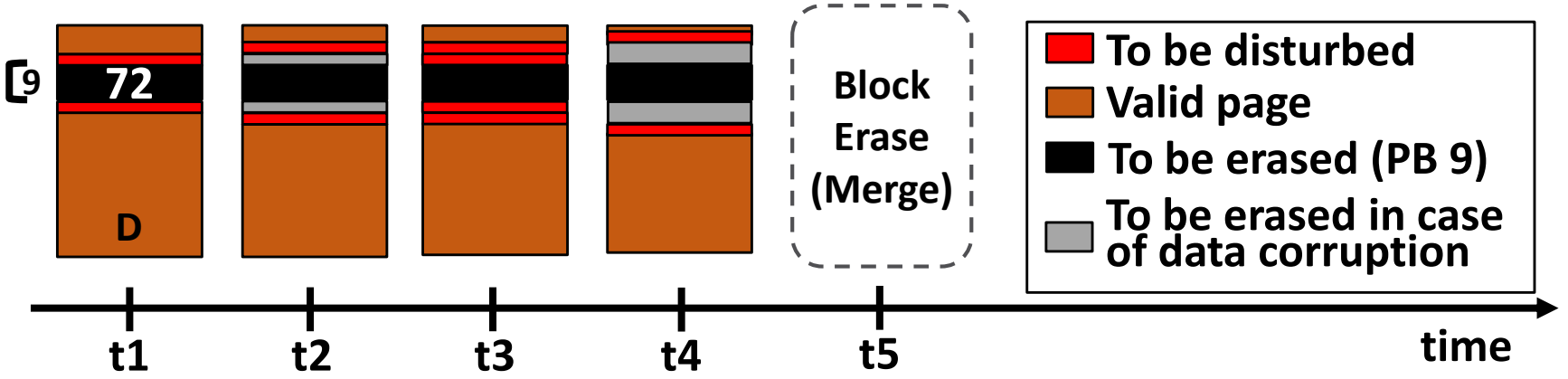




M-Merge with Program Disturbance

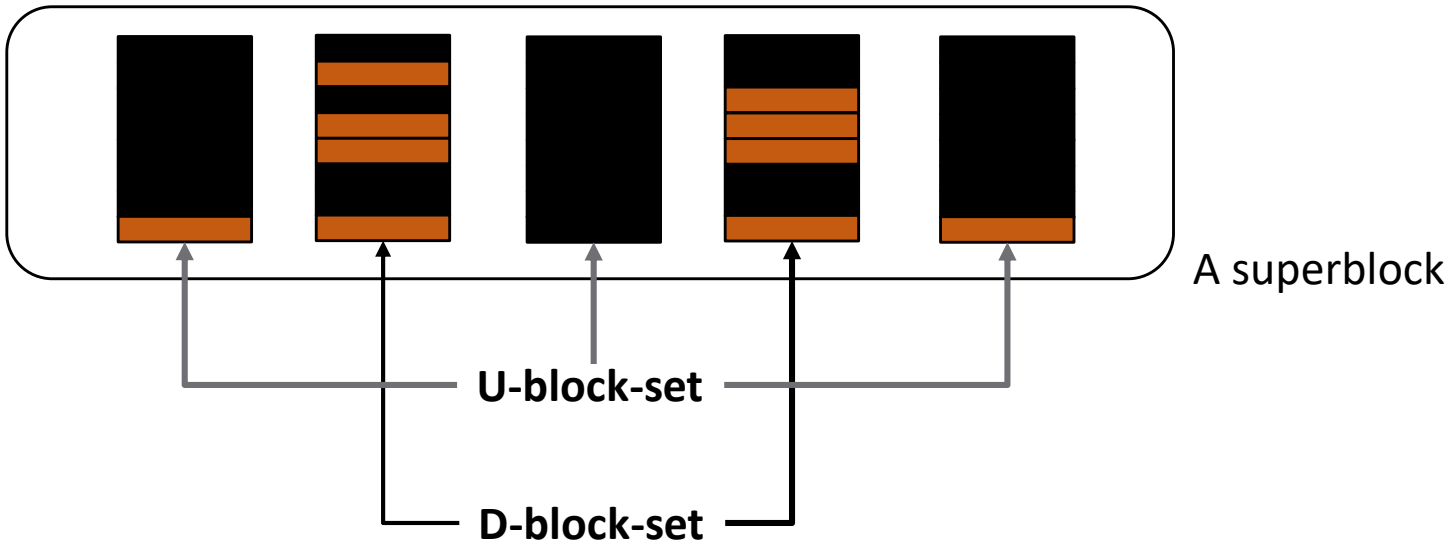
- To prevent the data corruption, M-Merge restored the previous disturbed pages.
- At time t2 and t4, neighbored partial-blocks are erased.
- To prevent wear-unleveling, a block can only be M-Merged limited times.





M-Merge for Hybrid Mapping (Superblock FTL)

- Superblock FTL has no D-/U-block concept
  - Before M-Merge, we assign D-/U-block-sets, based on number of valid pages.



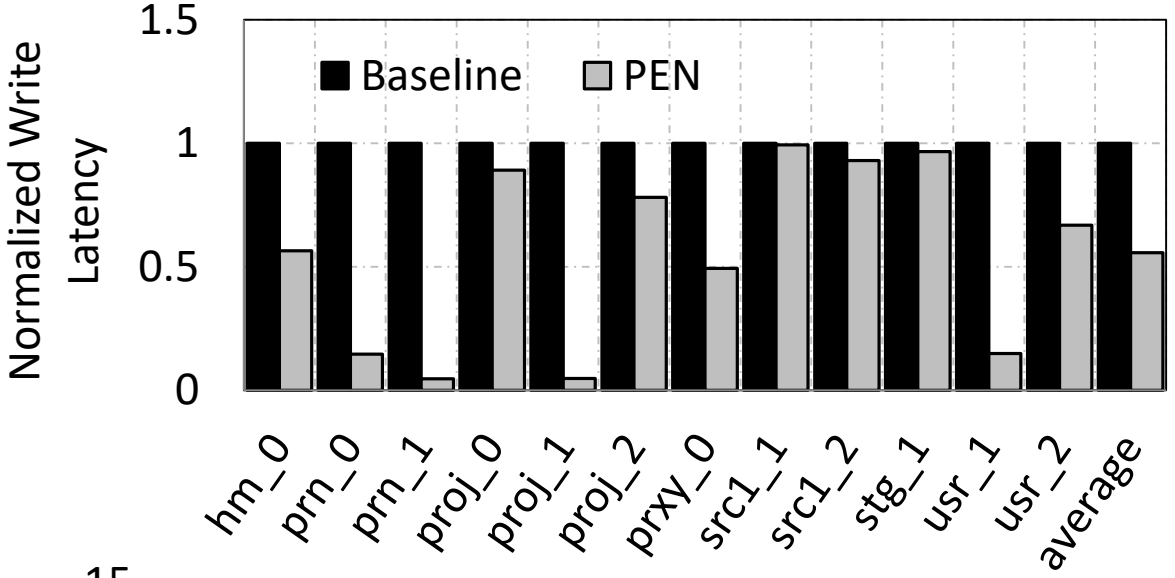
Outline

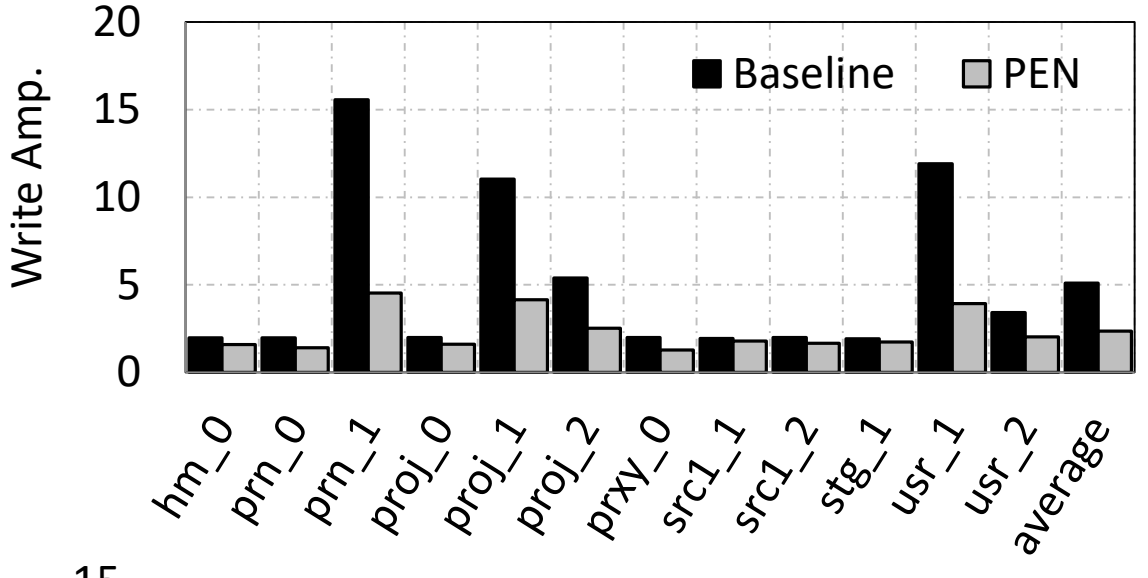
- Introduction and Background
- Motivation: Impact on Block Size
- Controller Hardware for Partial-Erase
- FTL for Partial-Erase
- **Evaluation**
- Conclusion

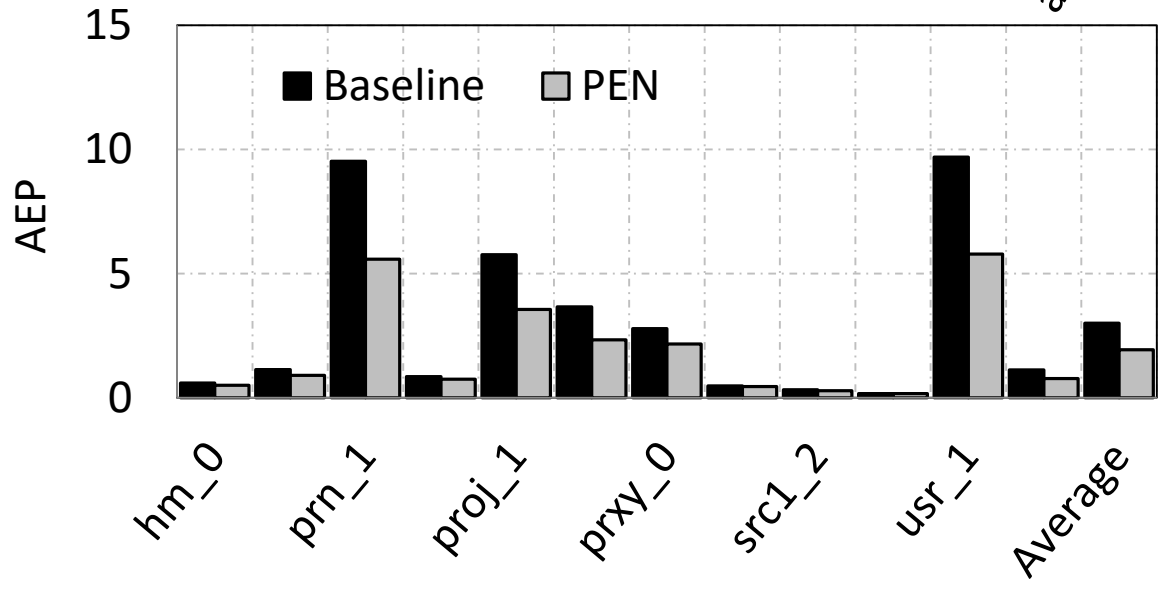
Experimental Setup

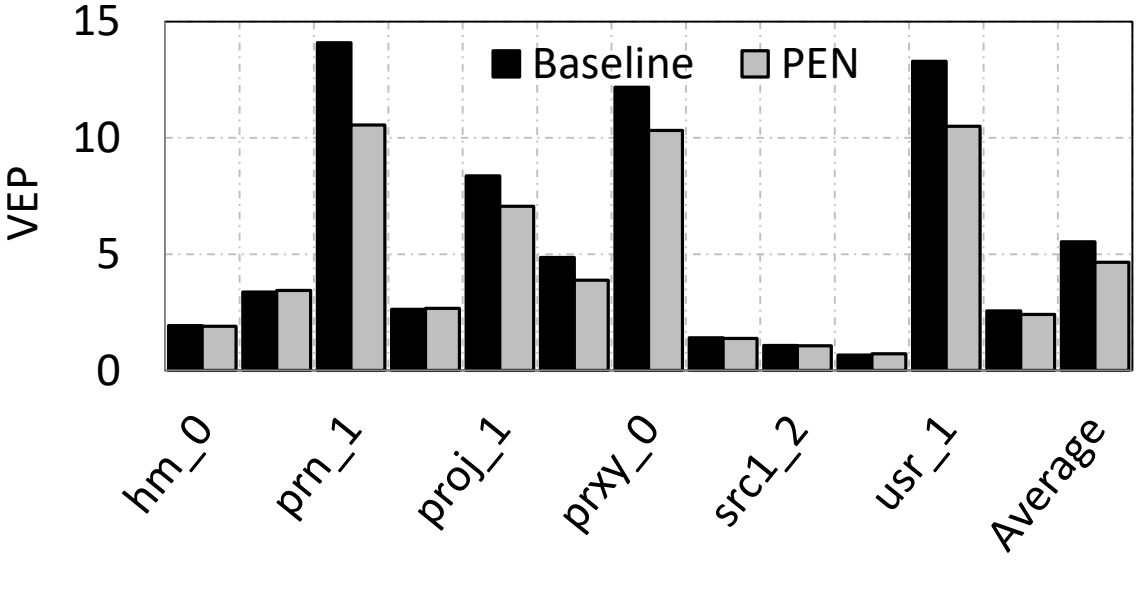
- We use 12 write-dominant workload traces.
- Four metrics
  - Average write latency
  - Write amplification
  - AEP – the average number of erase operations per page
  - VEP – the variance number of erase operations per page
    - Typically, AEB (average number of erase operation per block) are used.
    - Due to the partial-erase operation, a finer measurement is required.

NFTL Results

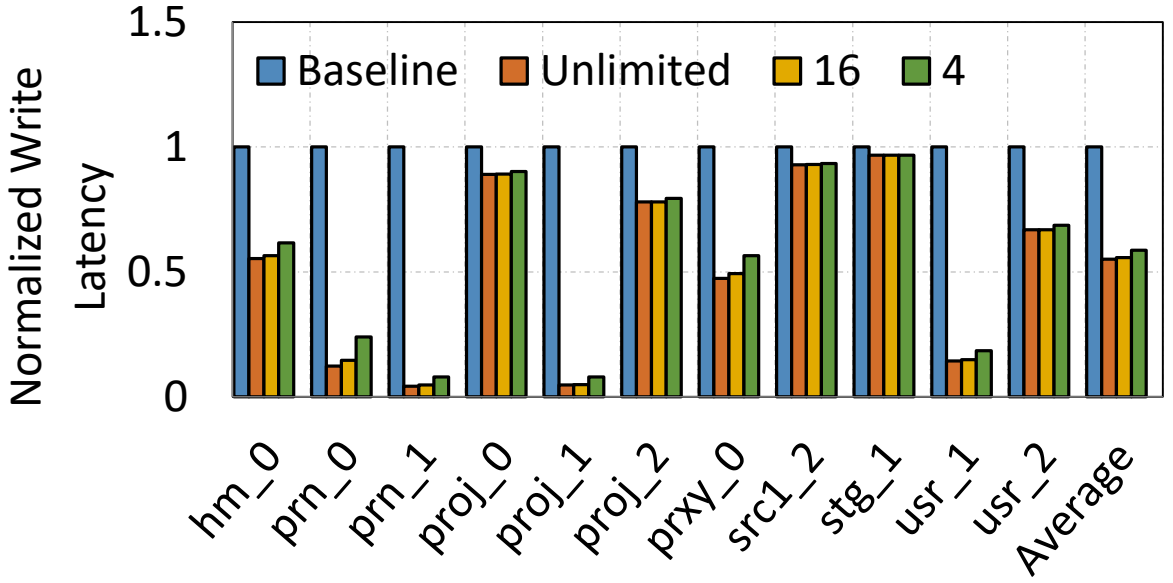


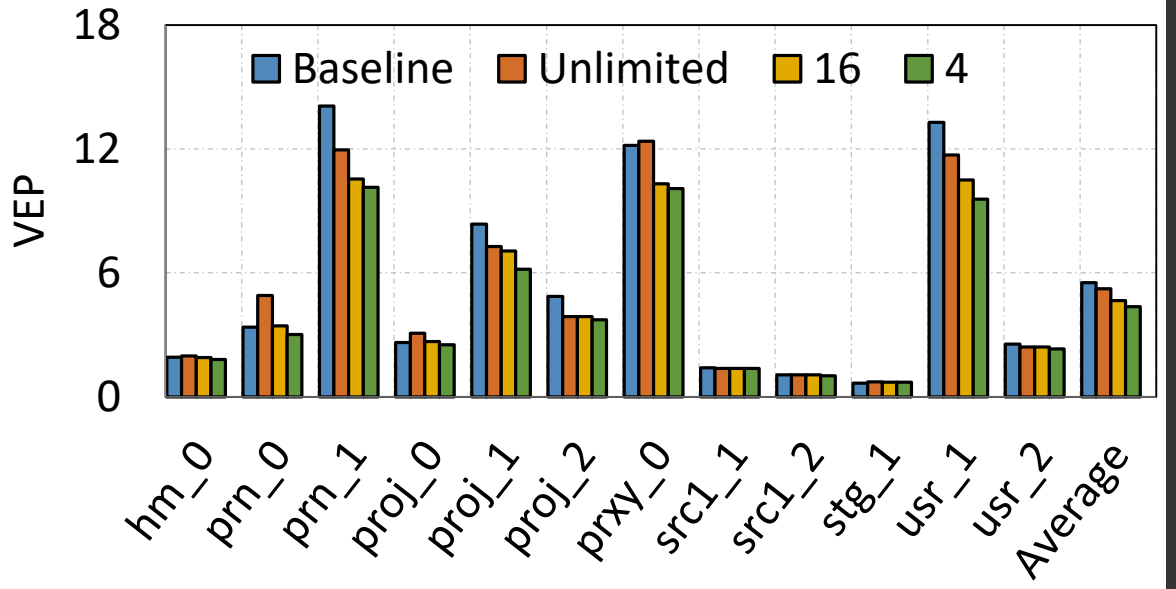


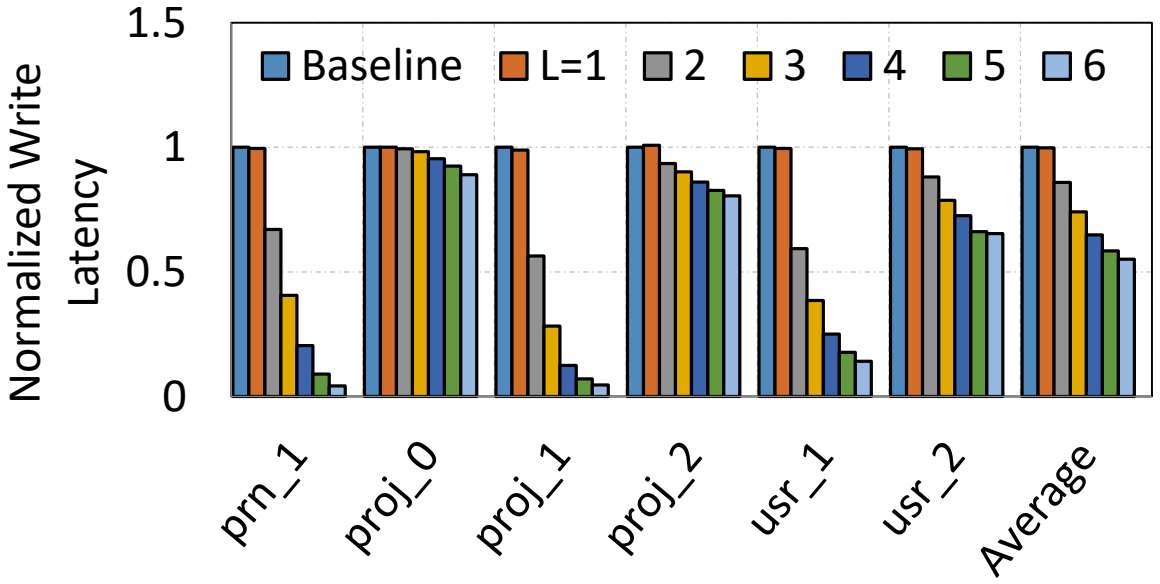


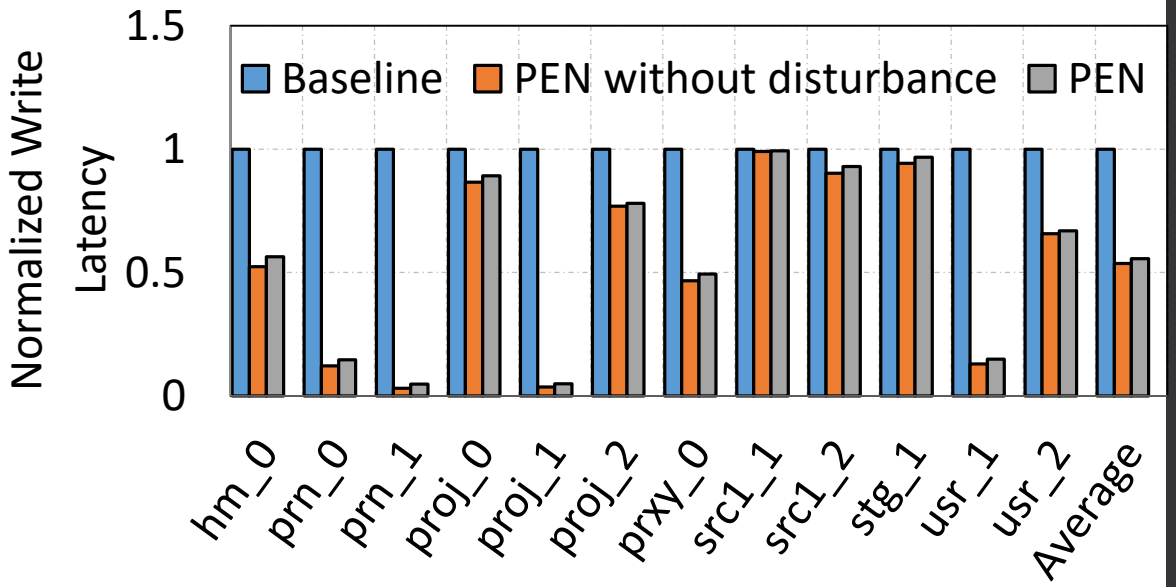


Sensitivity Results

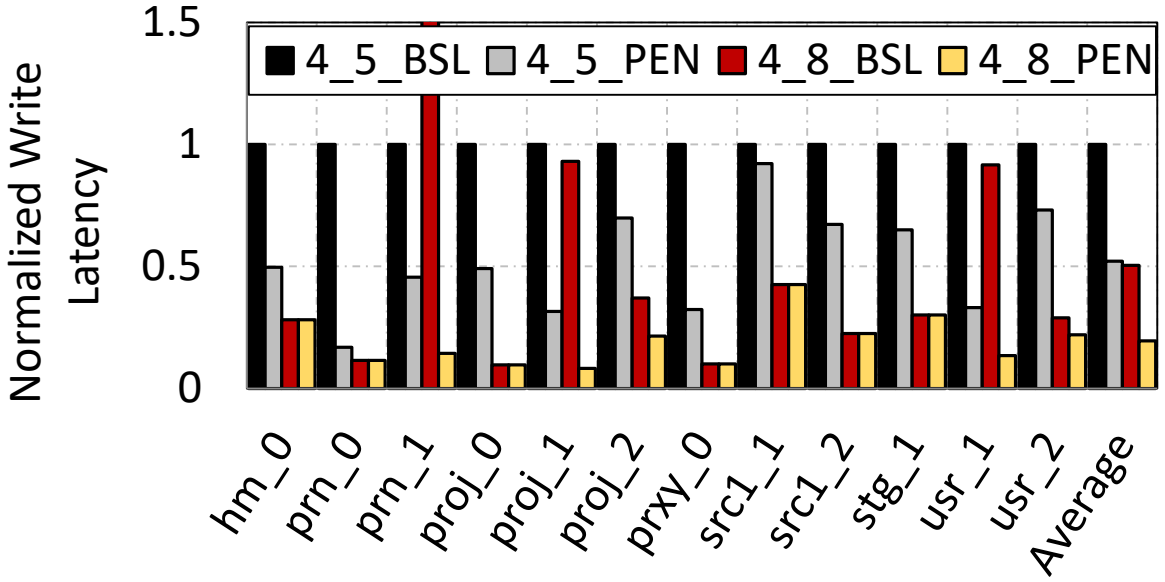


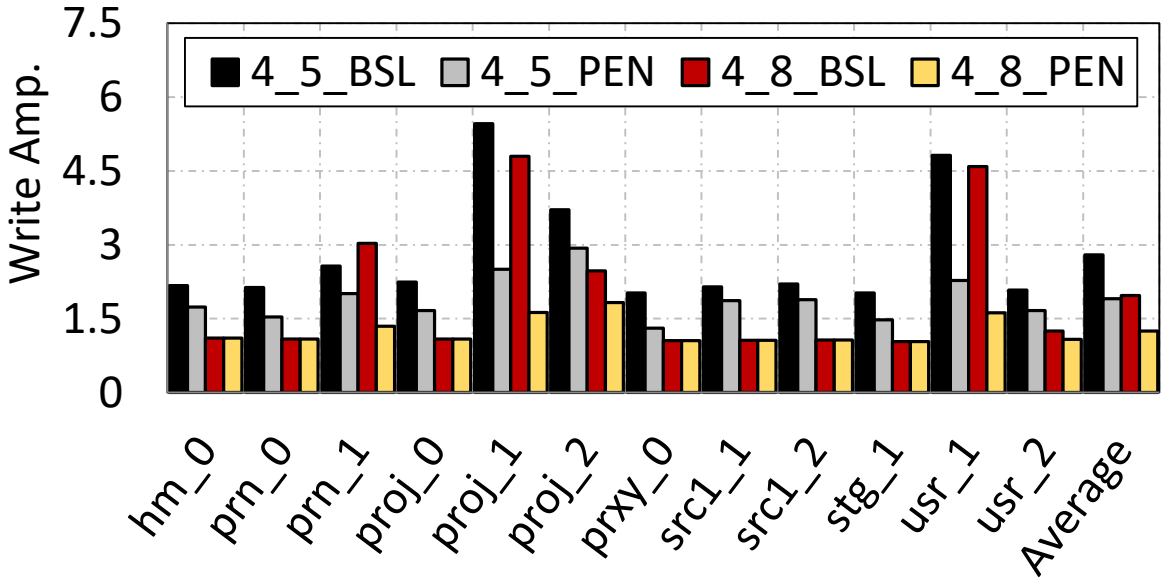


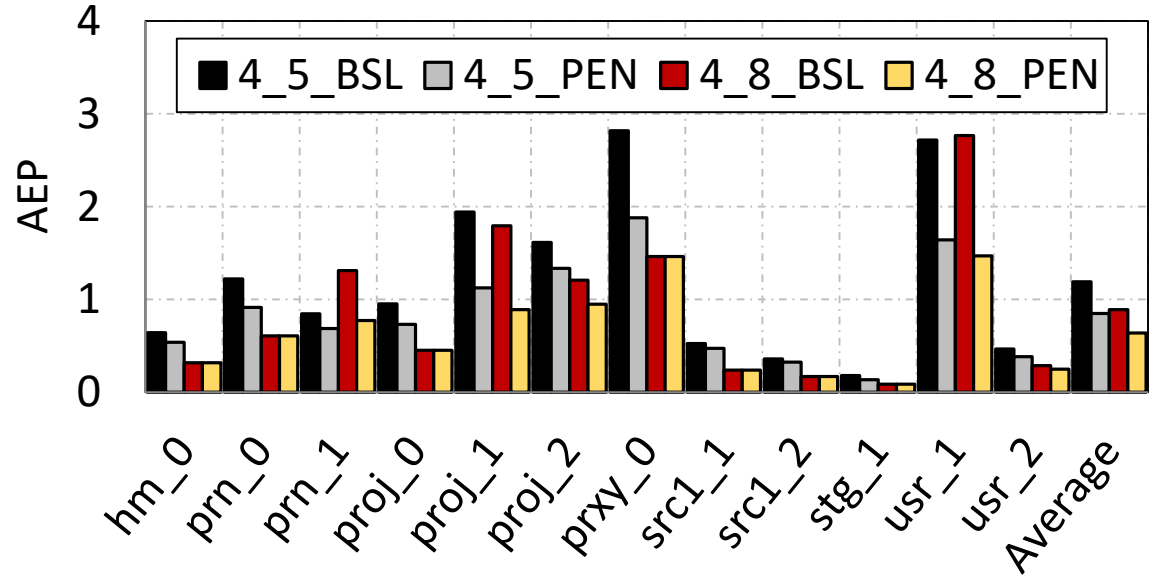


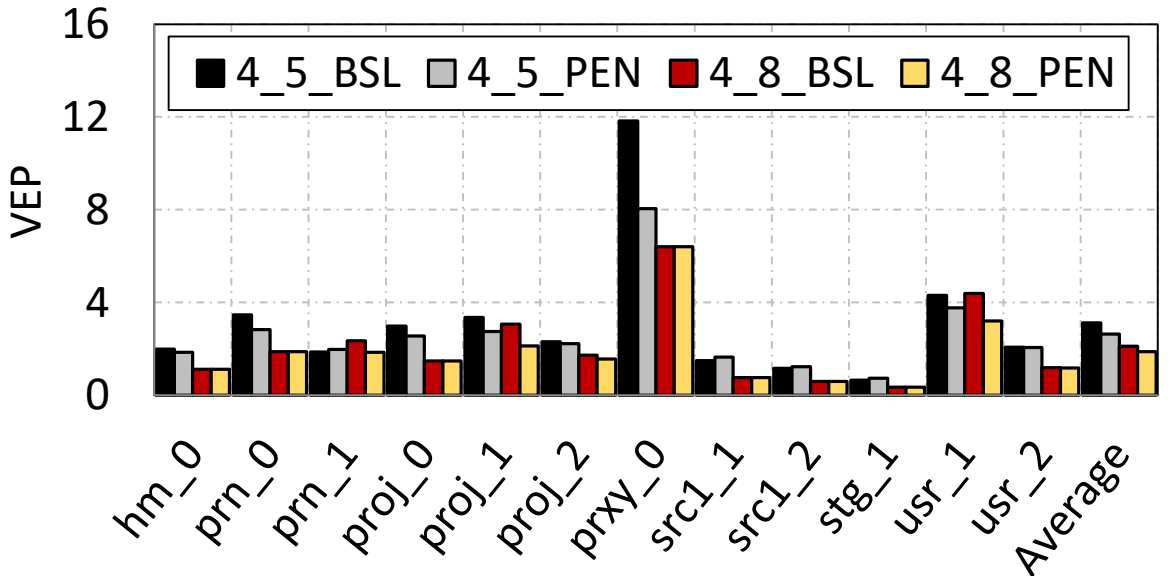


Superblock FTL Results









Related Works

- Partial-erase proposals
  - Hardware
    - 2D NAND partial-erase proposal – not beneficial due to a smaller block
    - Partial-block erase (PBE) – reduce the whole capacity
    - Subblock management – provide only three partial-blocks
  - Software
    - Subblock-erase – designed for page-level mapping
- Partial-GC proposals
  - Those redistribute the GC overheads to idle time, cannot reduce GC overheads.
  - Those can be combined with our partial-erase operation.

Conclusion

- we propose and evaluate a novel partial-erase based PEN architecture in emerging 3D NAND flashes, which minimizes the number of valid pages copied during a GC operation.

- To show the effectiveness of our proposed partial-erase operation, we introduce our M-Merge algorithm that employs our partial-erase operation for NFTL and Superblock FTL.

- Our extensive experimental evaluations show that the average write latency under the proposed PEN system is reduced up to 47.9%.

Q & A

Thank You!