Write-hotness Aware Retention Management: Efficient Hot/Cold Data Partitioning for SSDs

Saugata Ghose
ghose@cmu.edu ▪ http://ece.cmu.edu/~saugatag/

joint work with Yixin Luo, Yu Cai, Jongmoo Choi, Onur Mutlu

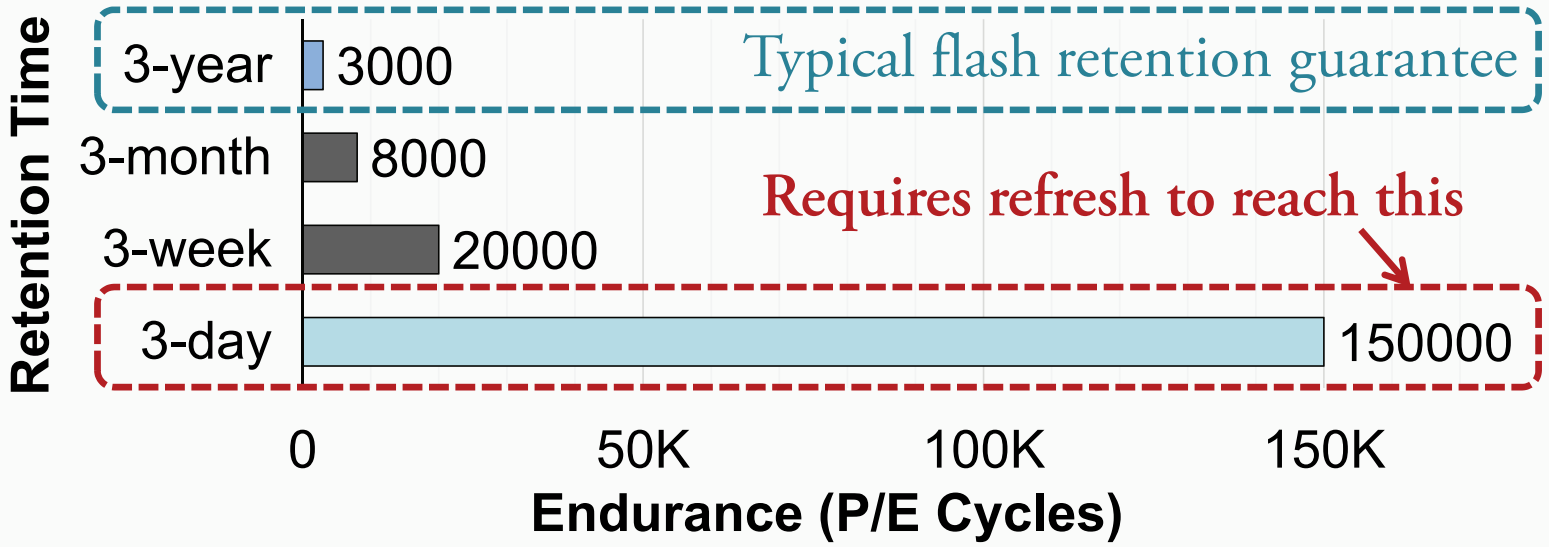
August 10, 2016
Santa Clara, CA

Executive Summary

- Flash memory can achieve up to **50x endurance improvement by relaxing retention time using refresh** [Cai+ ICCD '12]
  - Problem: **Refresh consumes majority of endurance improvement**
  - Goal: Reduce refresh overhead to increase flash memory lifetime
- Key Observation: **Refresh unnecessary for *write-hot data***
- Write-hotness Aware Retention Management (WARM)
  - Physically **partition write-hot pages and write-cold pages** within flash
  - Apply *different* policies (garbage collection, wear-leveling, refresh) to each group
- WARM w/o refresh **improves lifetime by 3.24x** w/ 1.05KB overhead
- WARM w/ adaptive refresh **improves lifetime by 12.9x** (1.21x over refresh)

Retention Time Relaxation for Flash Memory

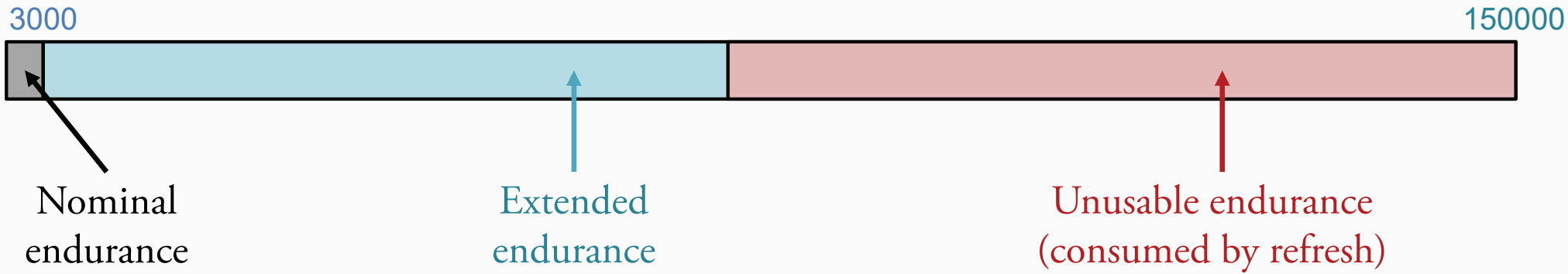
- Flash memory has limited write endurance
- Retention time
  - Duration for which flash memory can correctly hold data
  - Significantly affects endurance



[Cai+ ICCD '12]

Recovering Retention Time with NAND Flash Refresh

- Flash Correct and Refresh (FCR), Adaptive Rate FCR (ARFCR) [Cai+ ICCD '12]

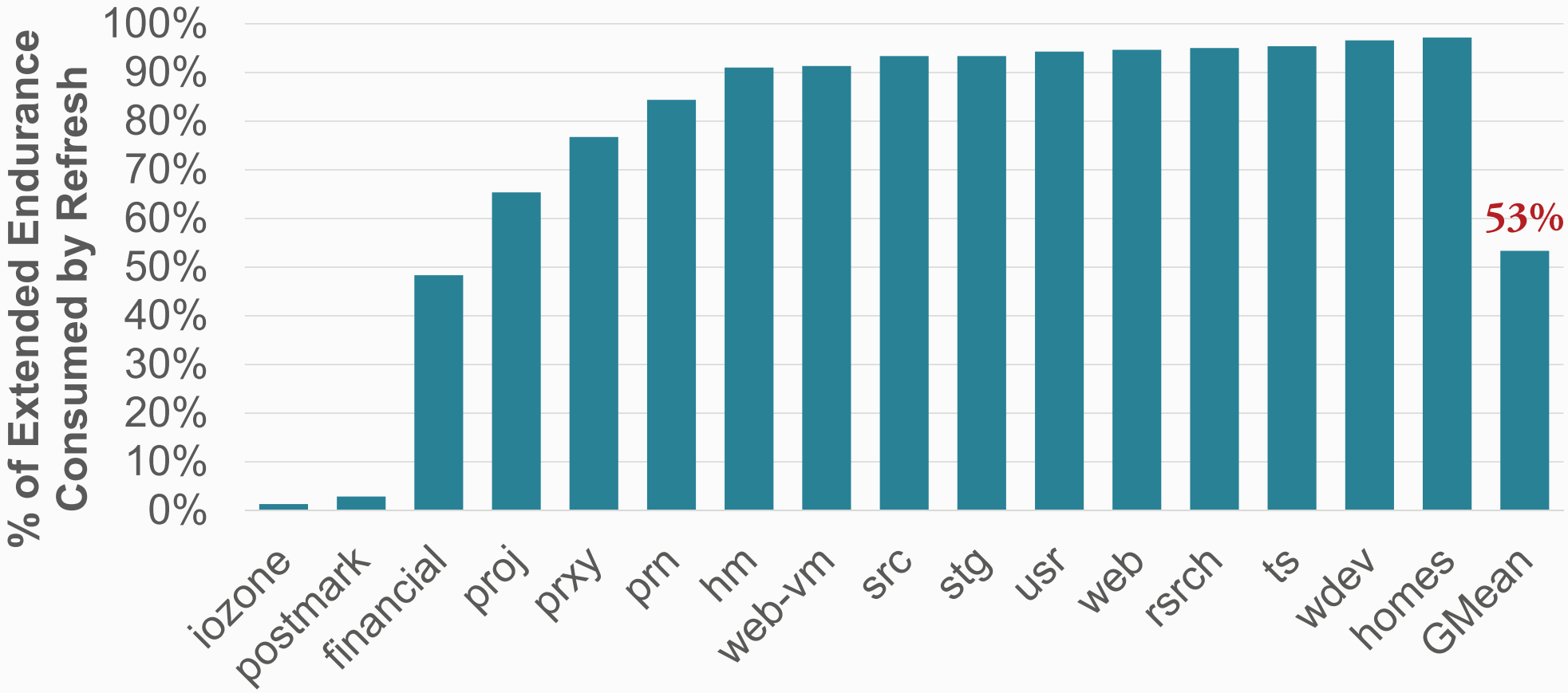


- Problem: Flash refresh operations reduce extended lifetime

OUR GOAL

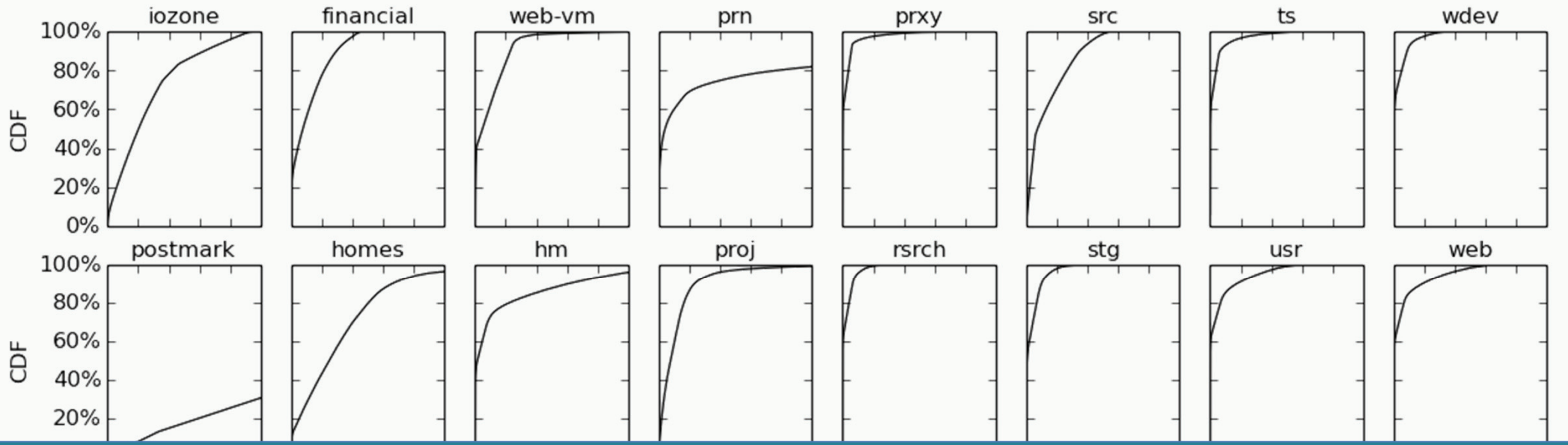
Reduce refresh overhead, improve flash lifetime

Observation 1: High Refresh Overhead



Observation 2: Skewed Distribution of Write Activity

SAFARI



Small amount of *write-hot* data generates large fraction of writes

Observation 3: Write-hot Pages Can Safely Skip Refresh

SAFARI

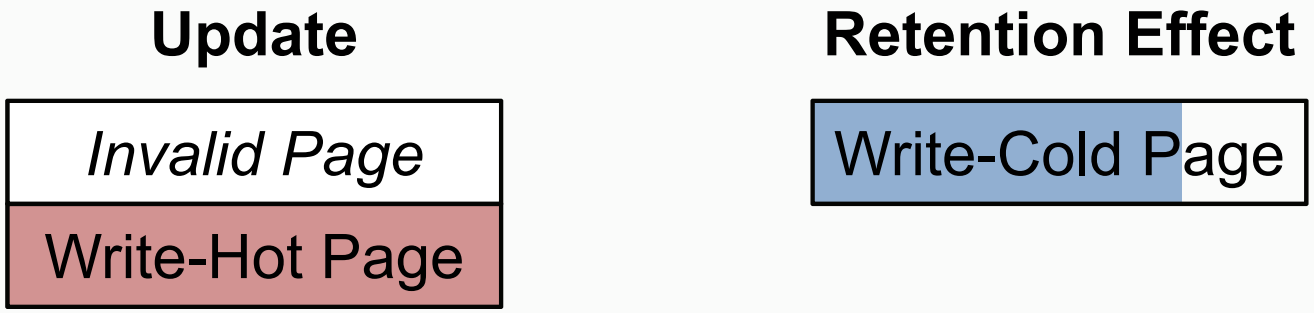
Write-Hot Page

Write-Cold Page

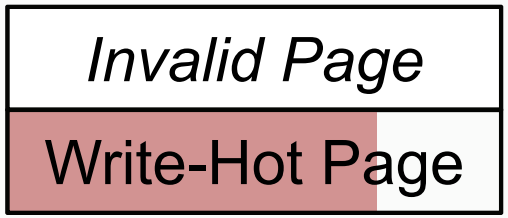
Observation 3: Write-hot Pages Can Safely Skip Refresh

Observation 3: Write-hot Pages Can Safely Skip Refresh

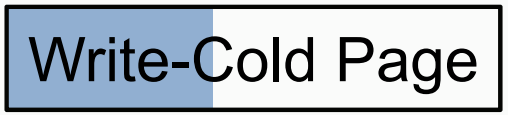
SAFARI



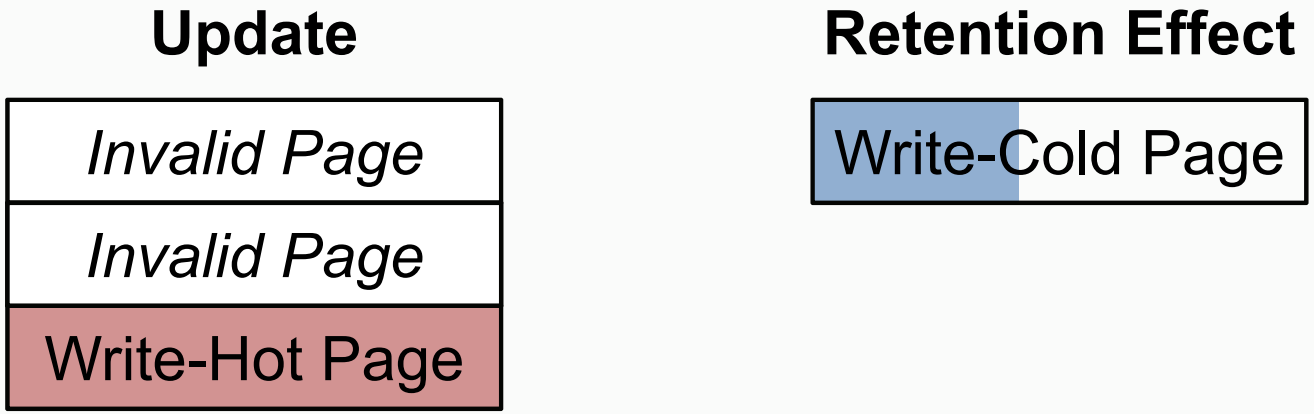
Observation 3: Write-hot Pages Can Safely Skip Refresh



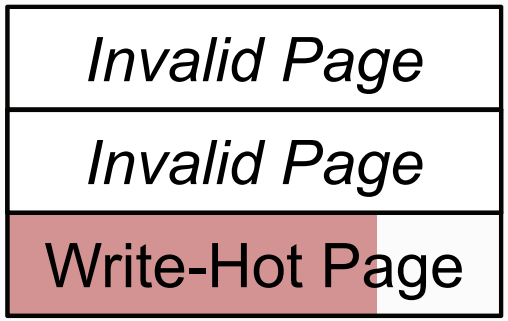
Retention Effect



Observation 3: Write-hot Pages Can Safely Skip Refresh

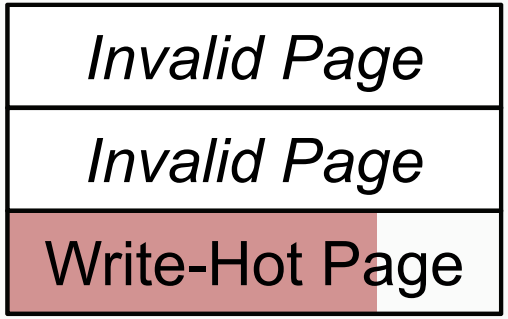


Observation 3: Write-hot Pages Can Safely Skip Refresh

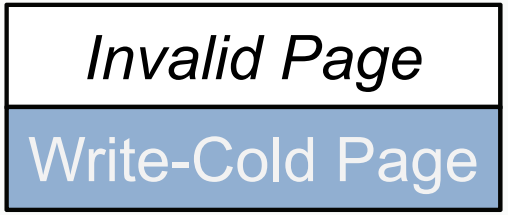


Retention Effect

Observation 3: Write-hot Pages Can Safely Skip Refresh

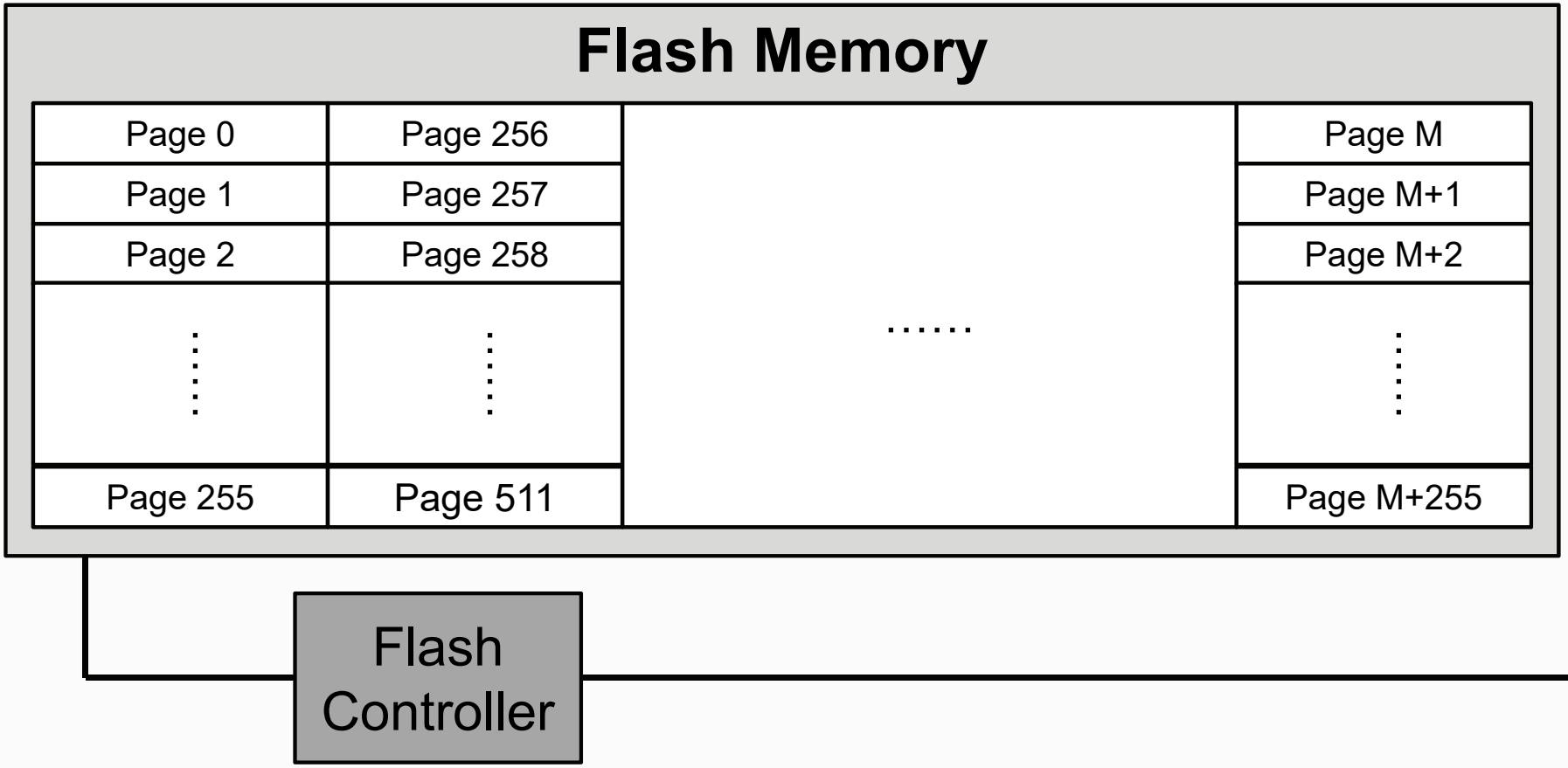


Refresh Unnecessary

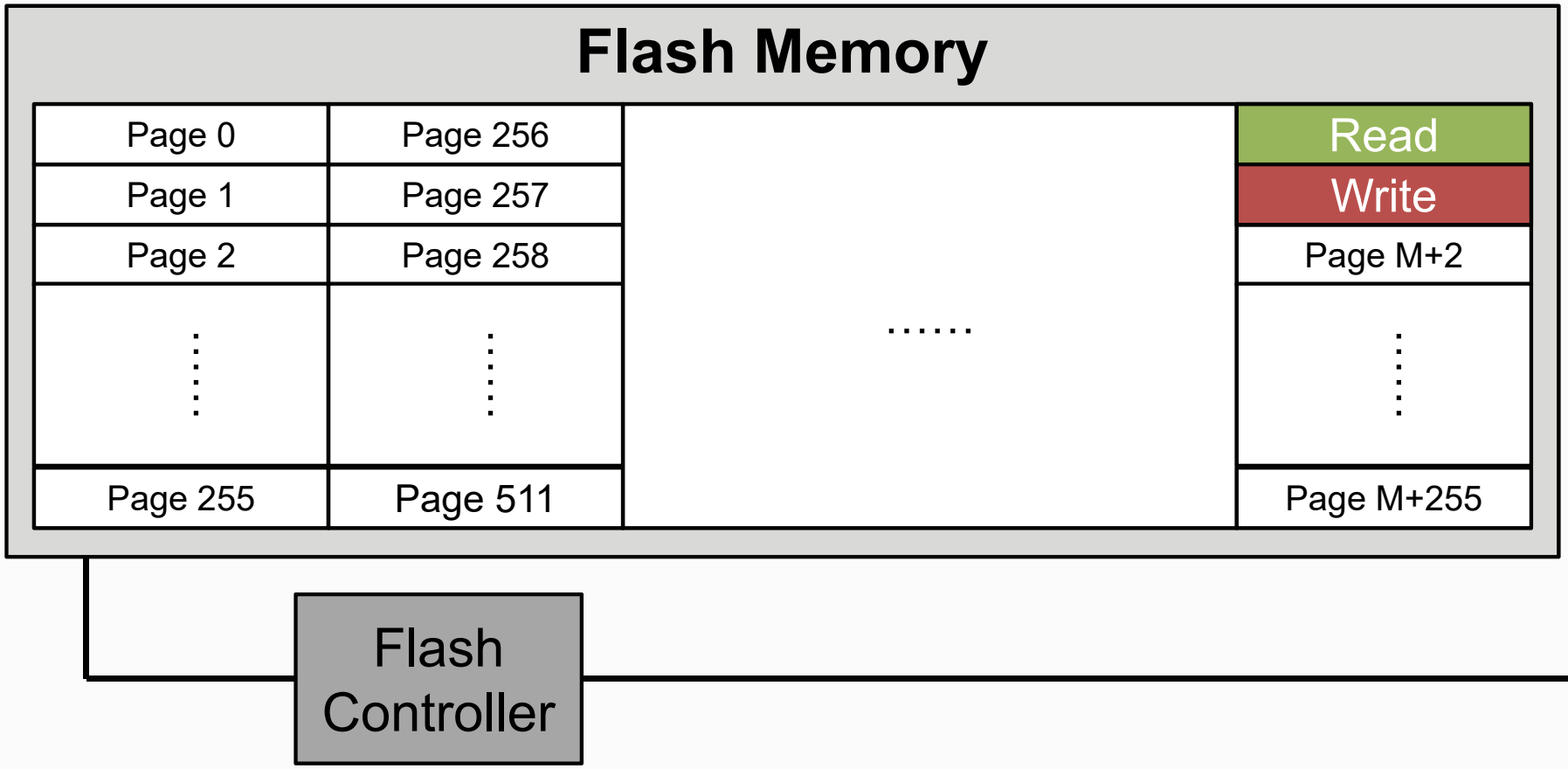


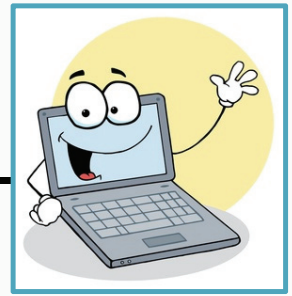
Needs Refresh

Conventional Write-hotness Oblivious Management

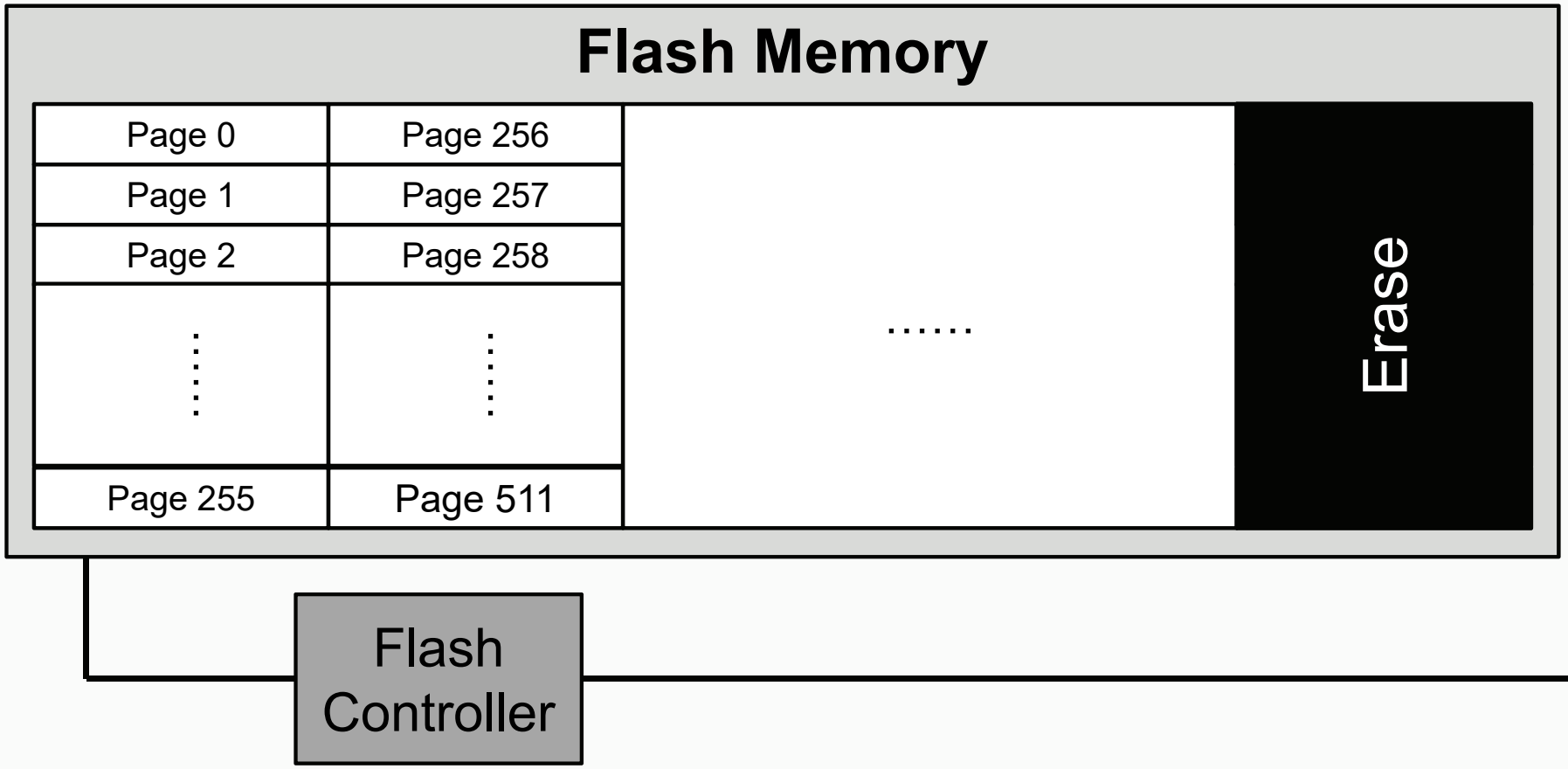


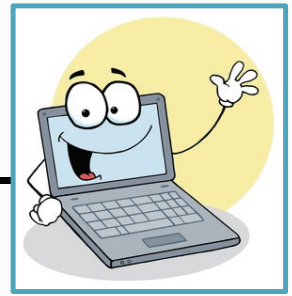
Conventional Write-hotness Oblivious Management



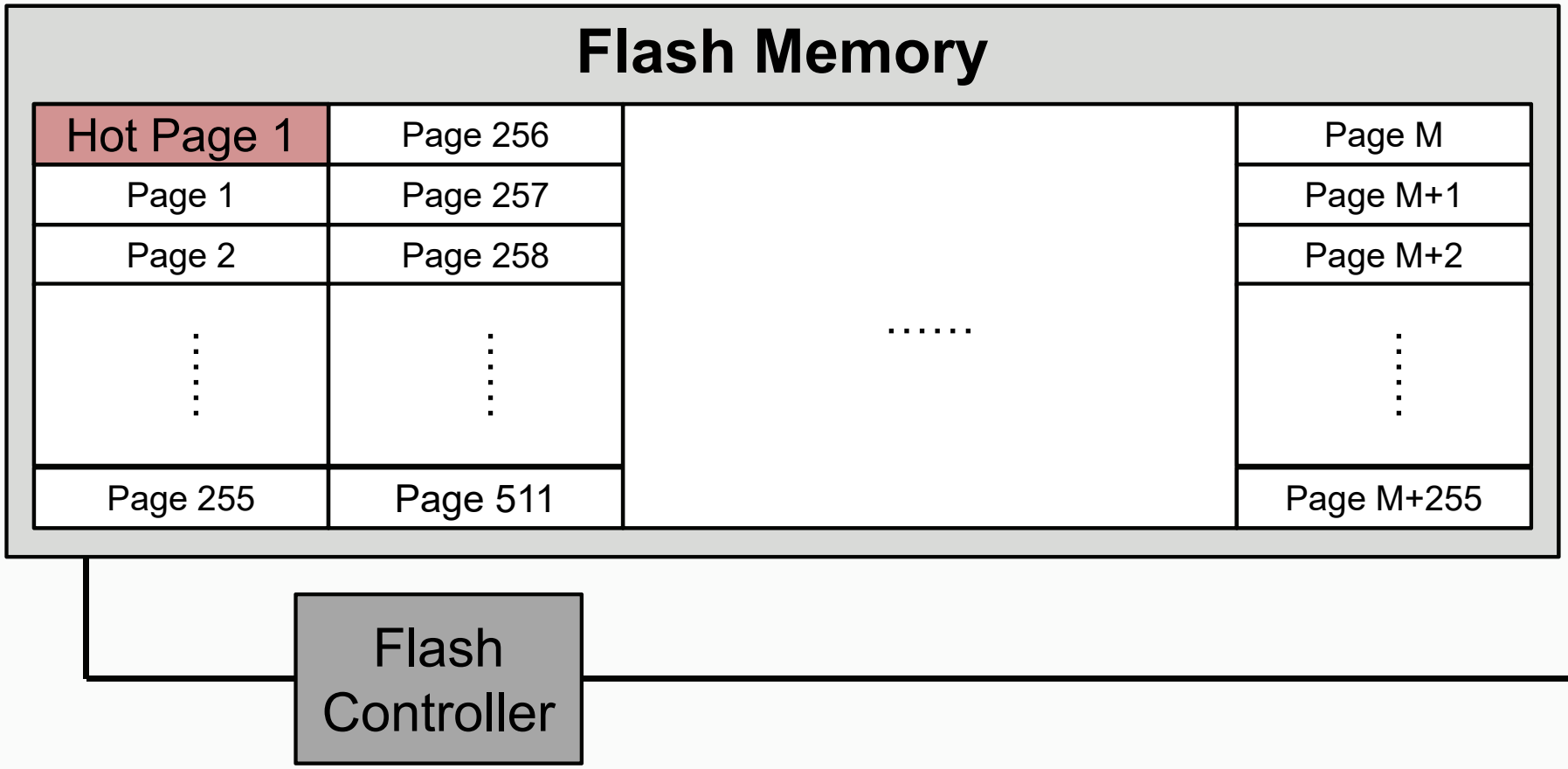


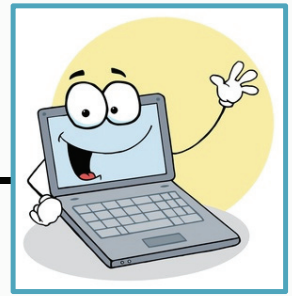
Conventional Write-hotness Oblivious Management



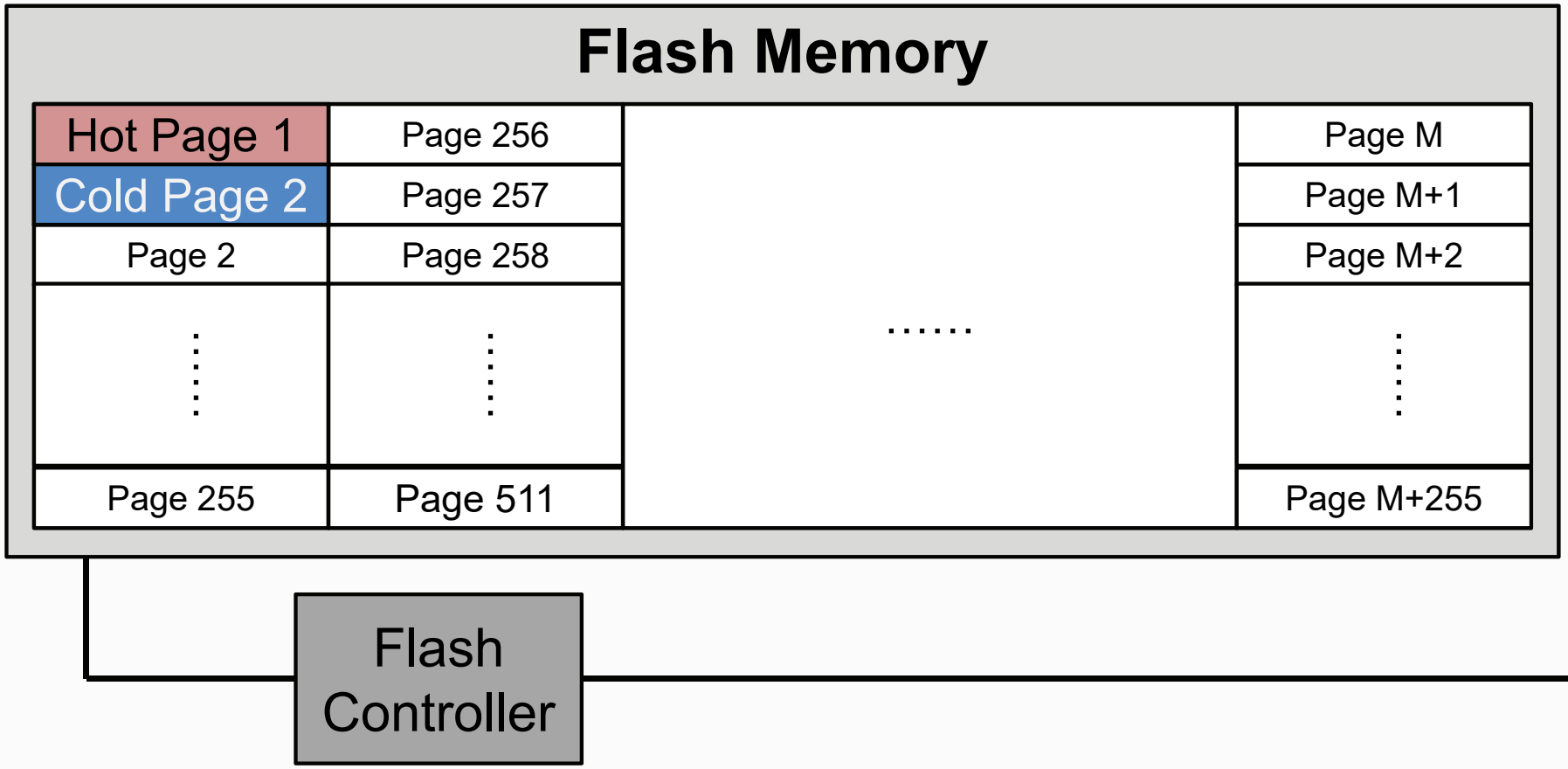


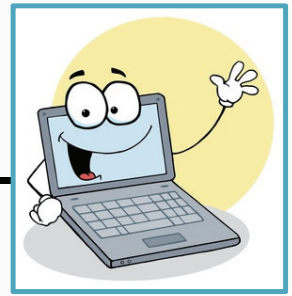
Conventional Write-hotness Oblivious Management



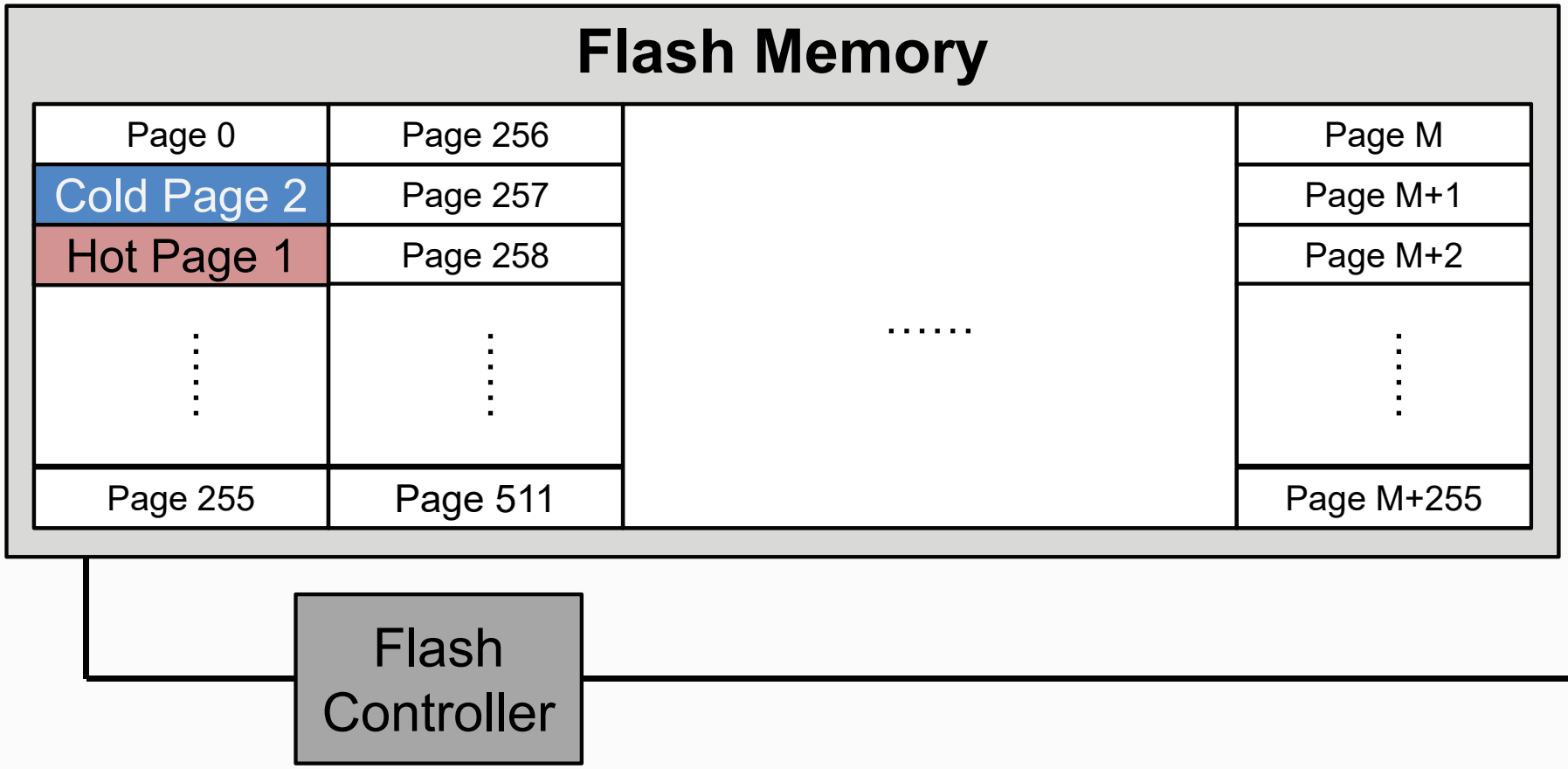


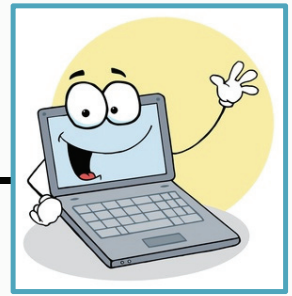
Conventional Write-hotness Oblivious Management



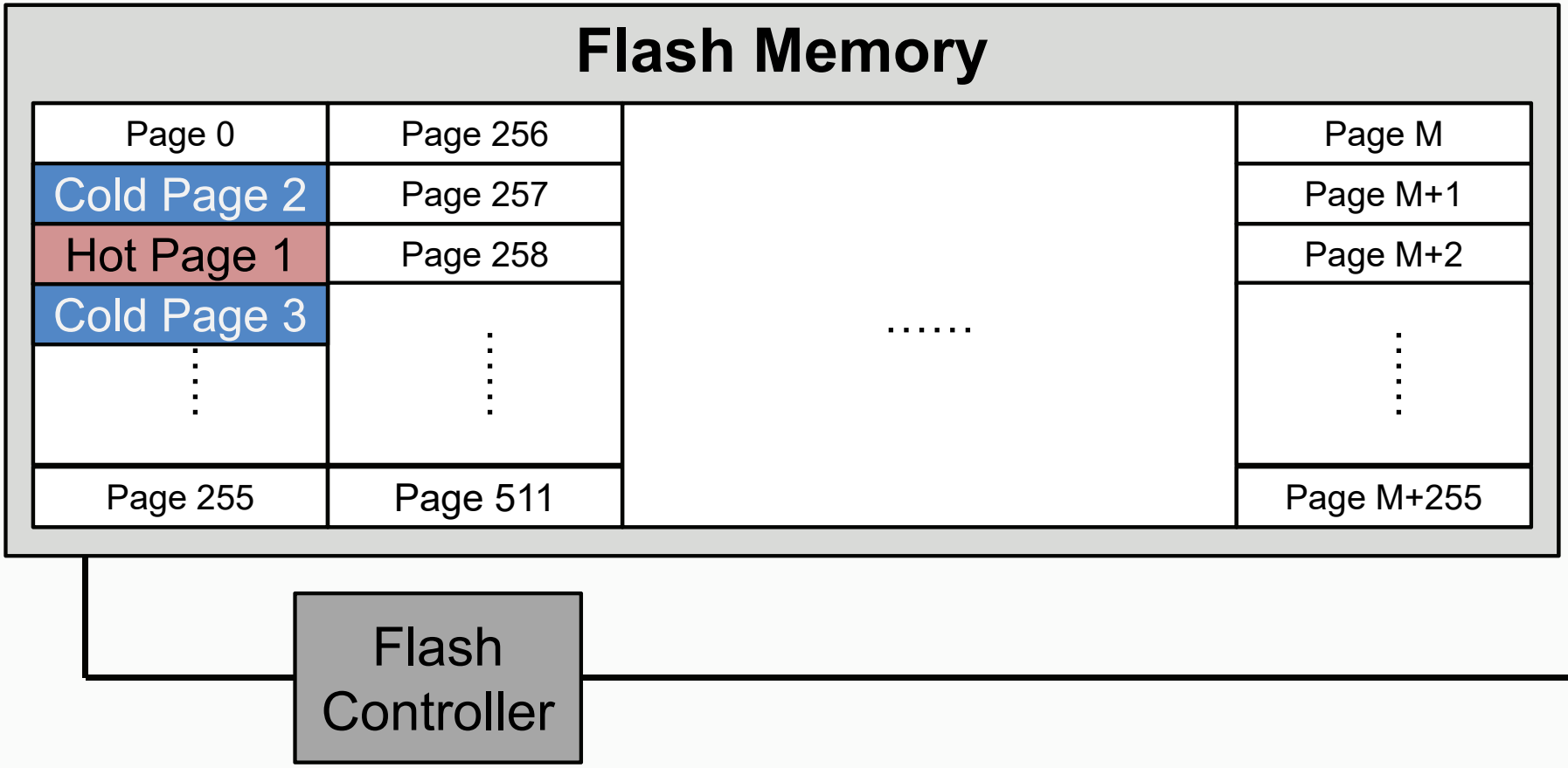


Conventional Write-hotness Oblivious Management

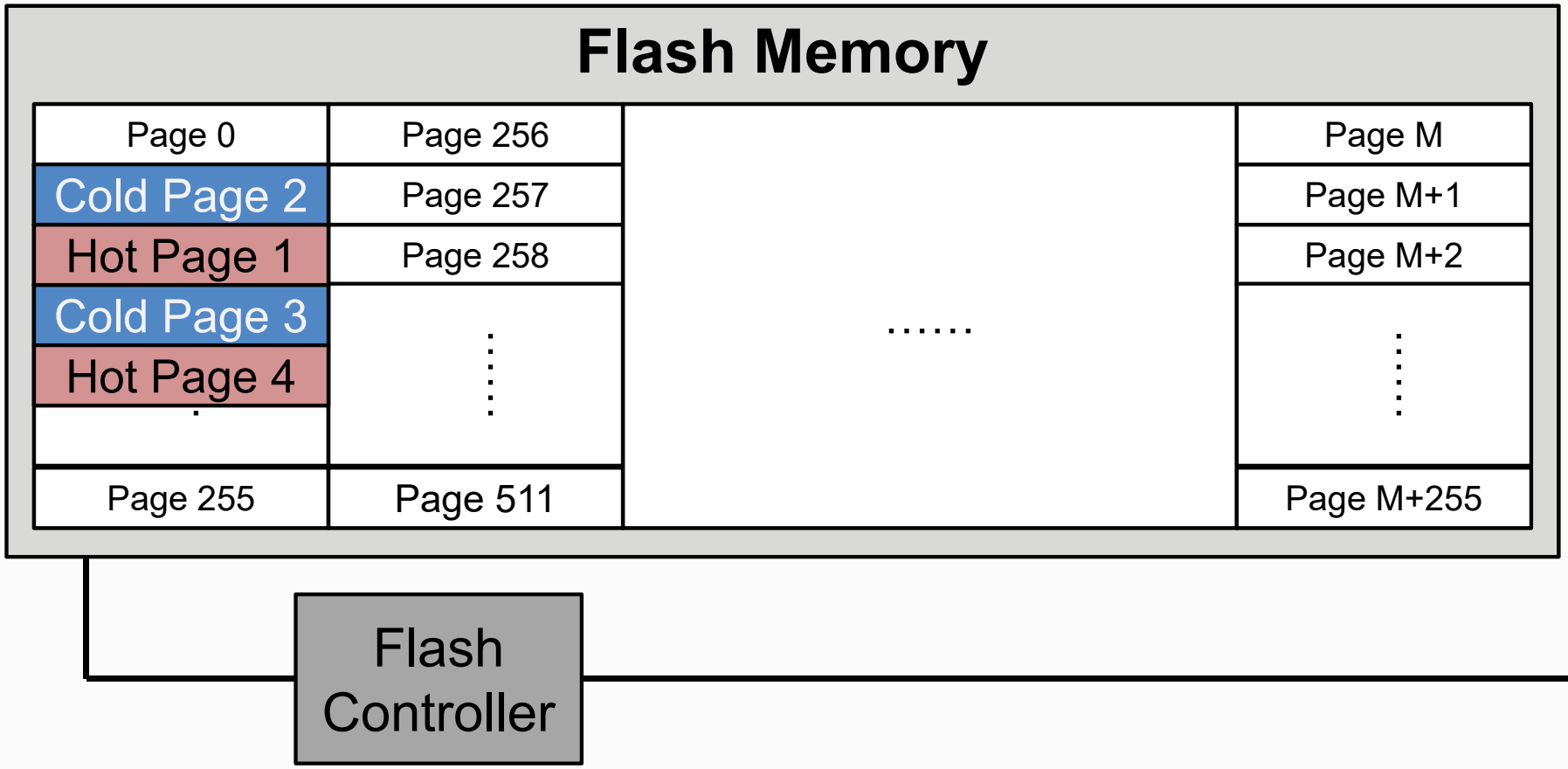


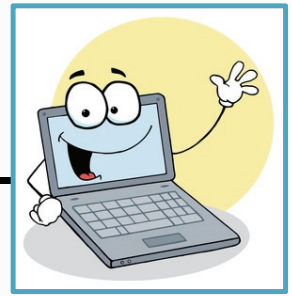


Conventional Write-hotness Oblivious Management

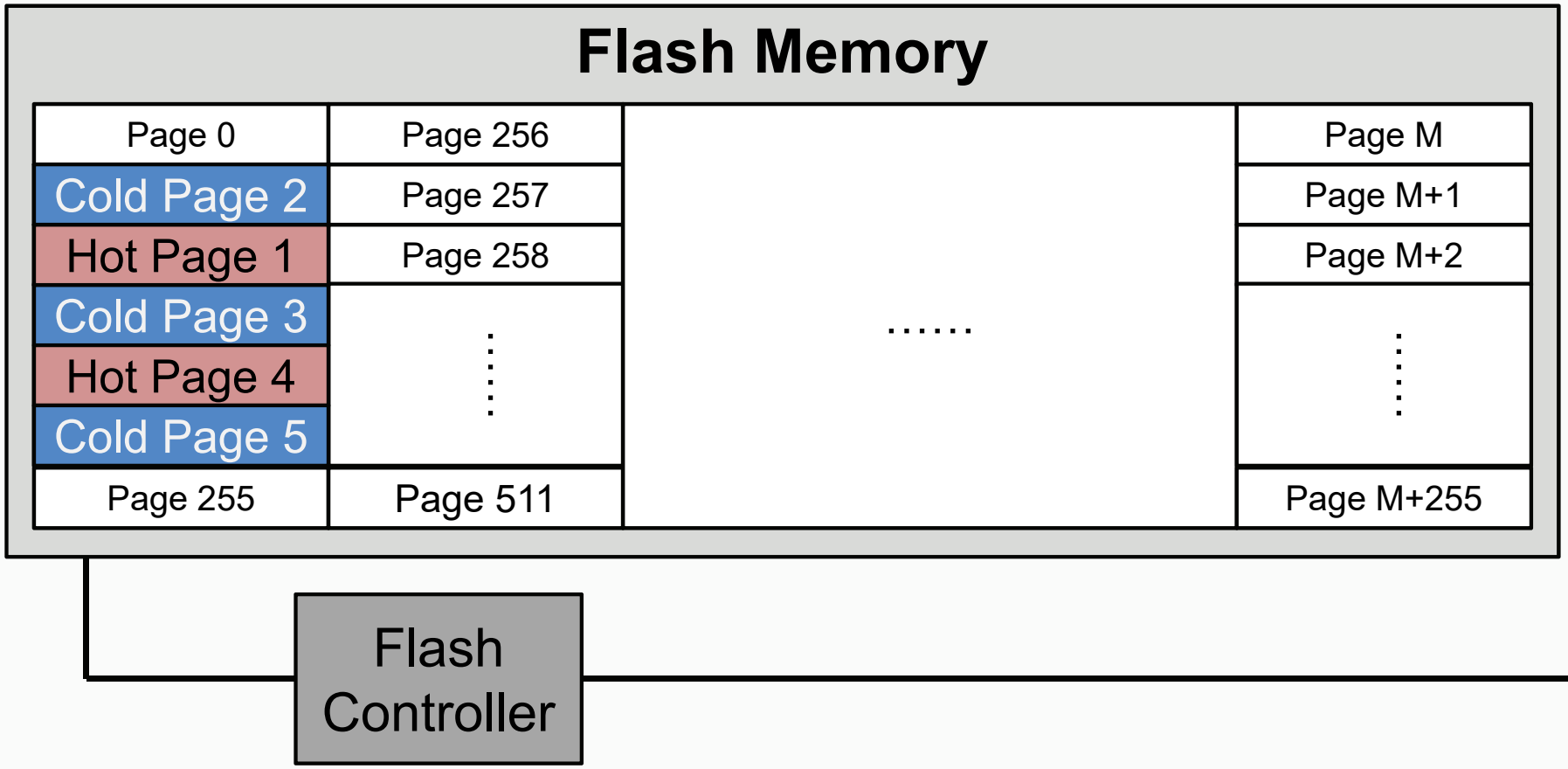


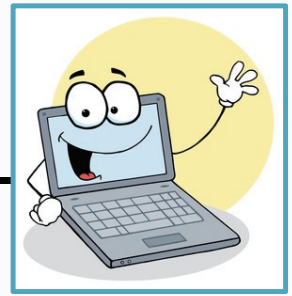
Conventional Write-hotness Oblivious Management



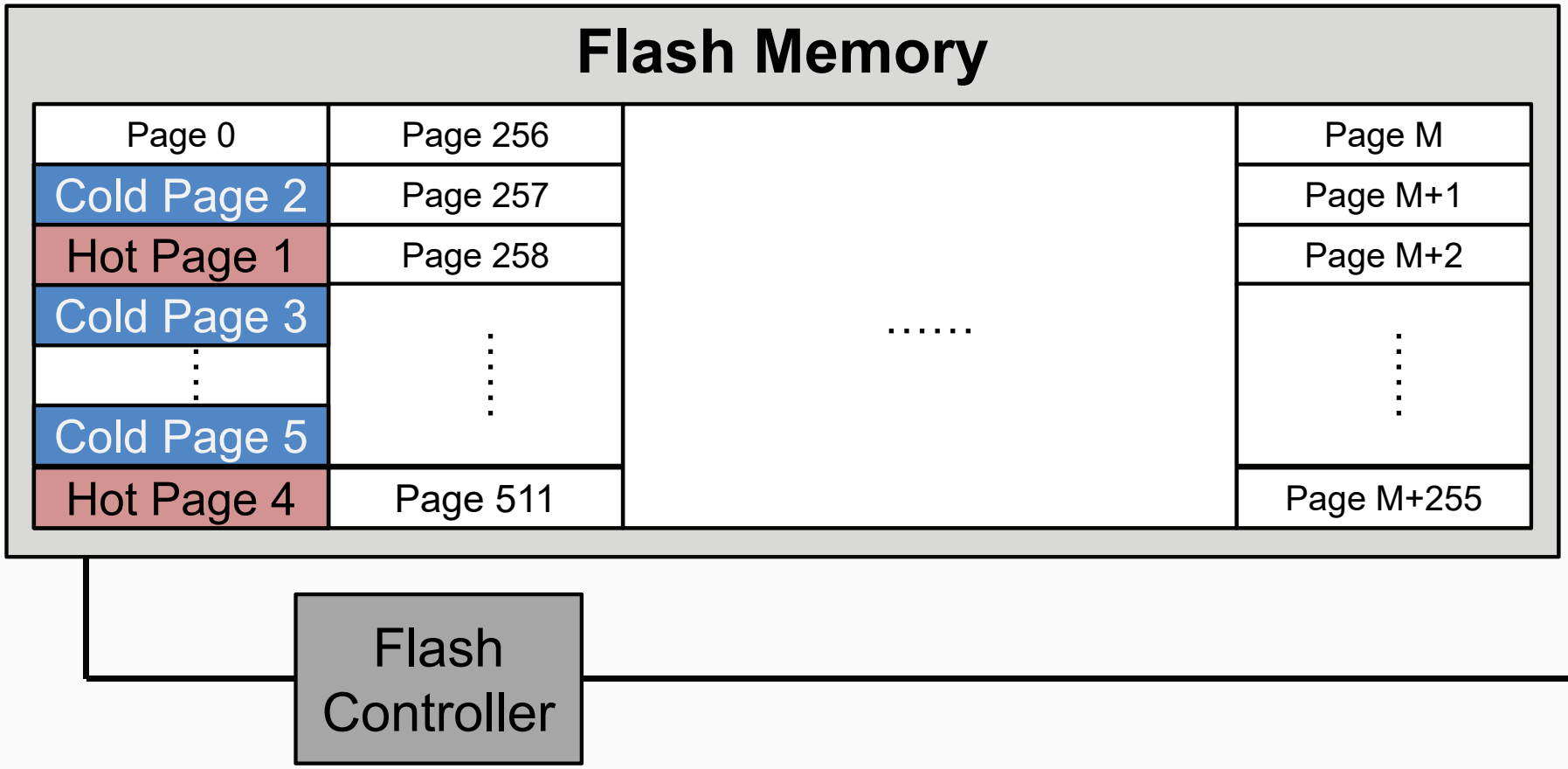


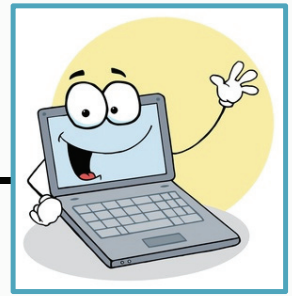
Conventional Write-hotness Oblivious Management



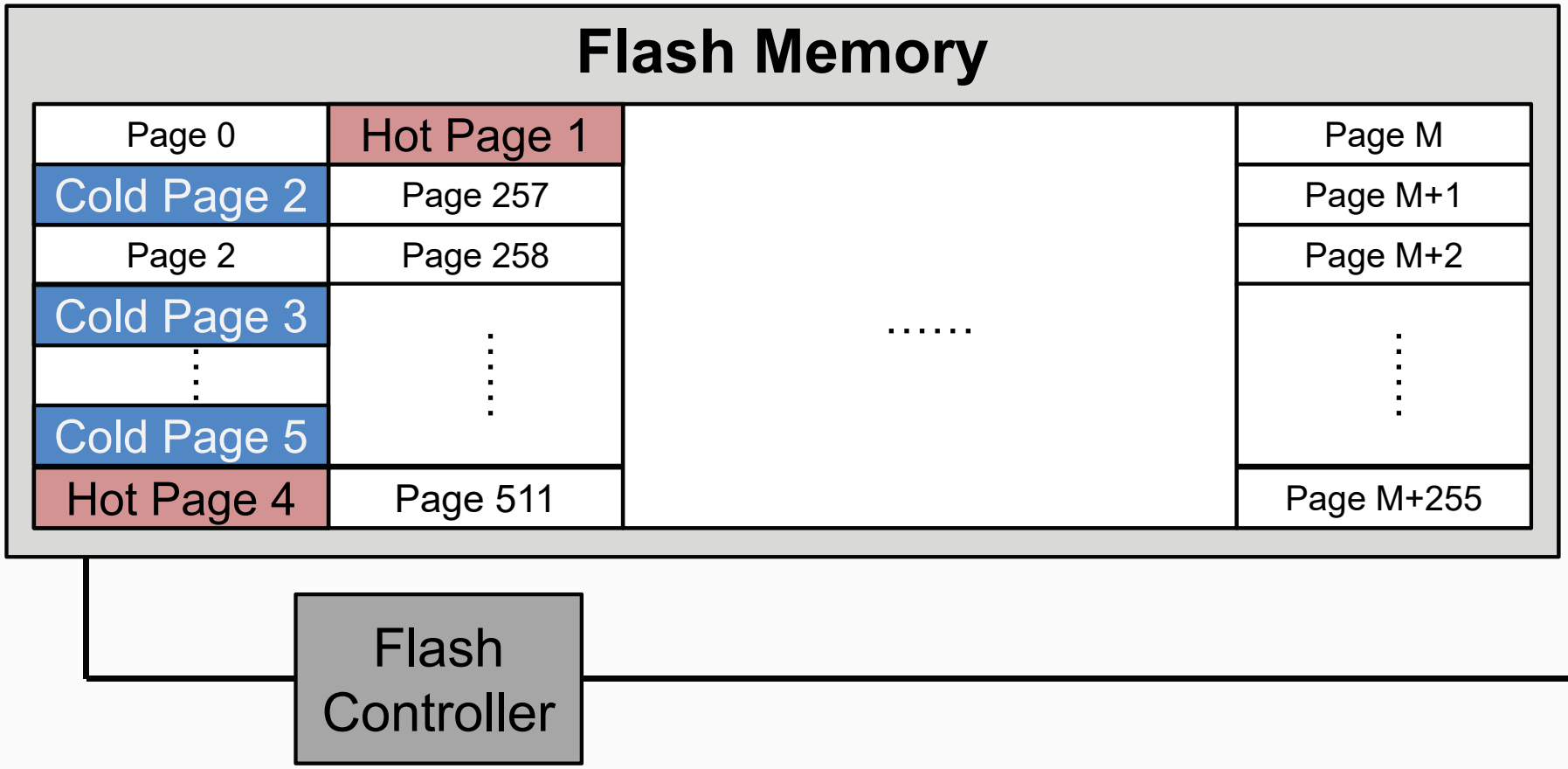


Conventional Write-hotness Oblivious Management

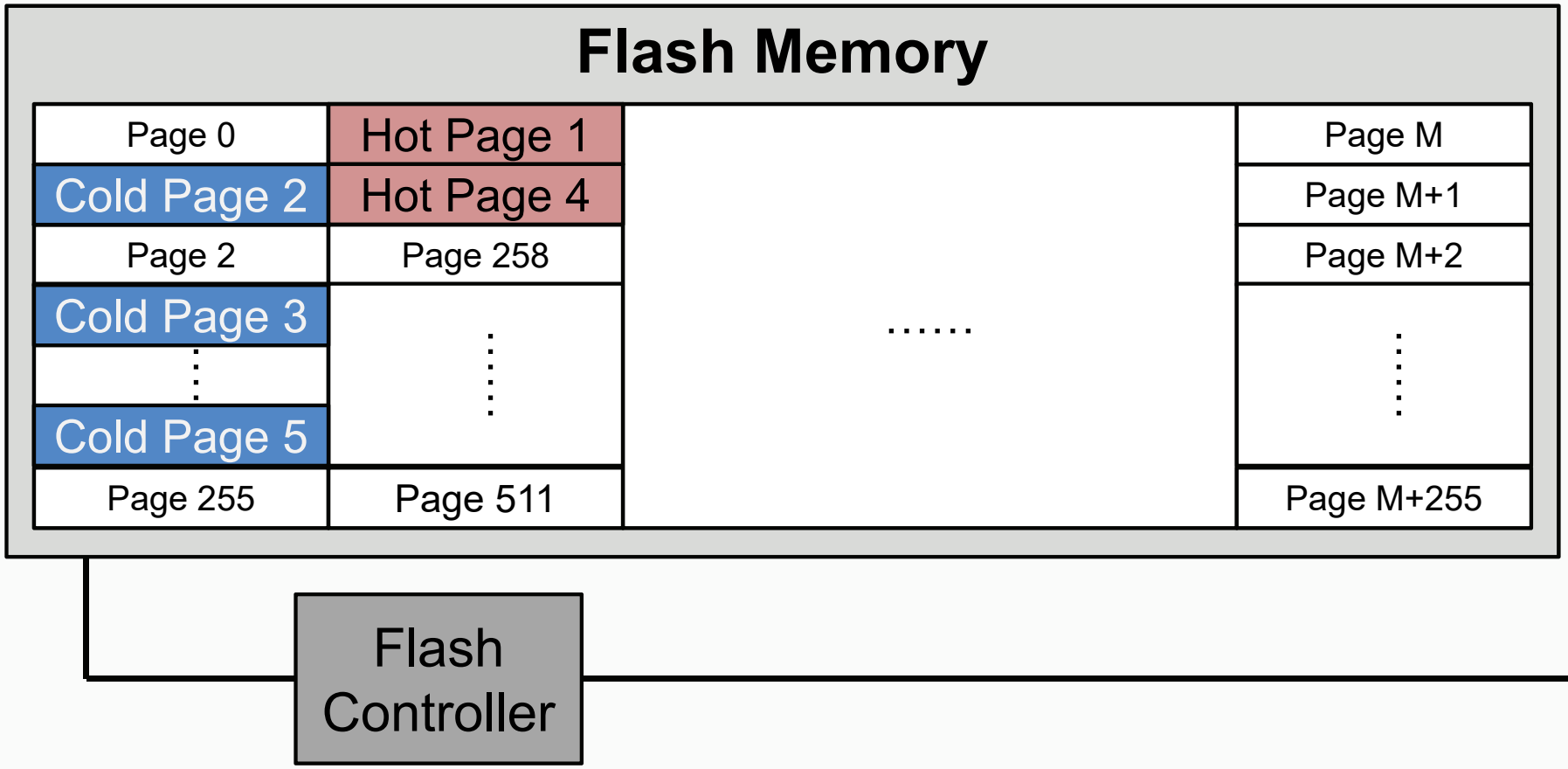




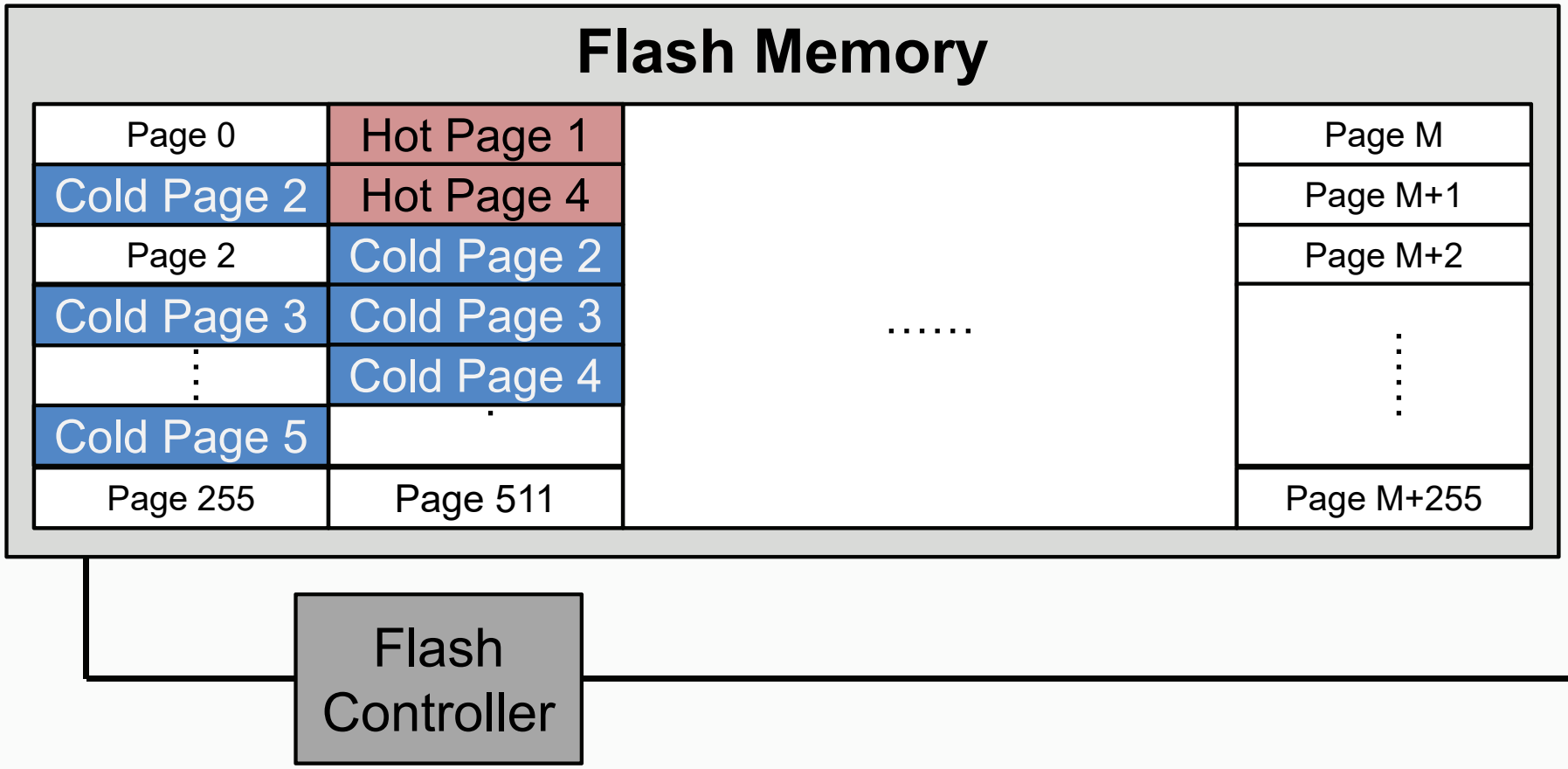
Conventional Write-hotness Oblivious Management



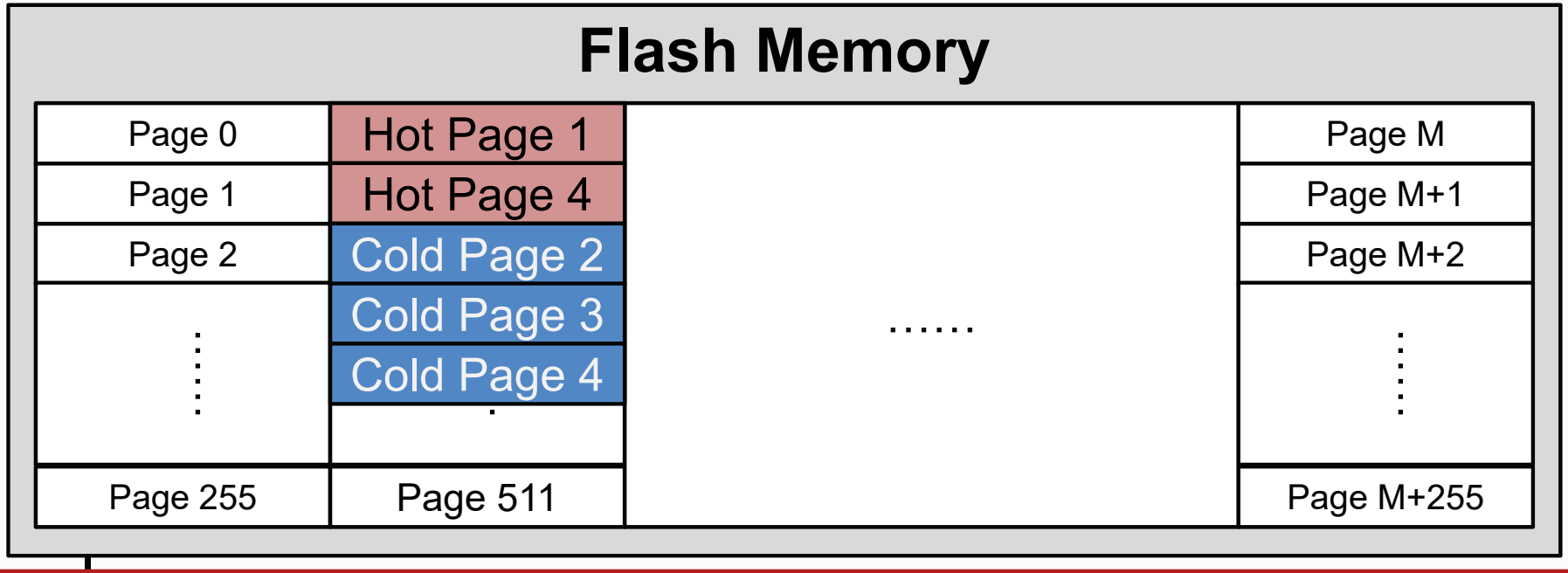
Conventional Write-hotness Oblivious Management



Conventional Write-hotness Oblivious Management

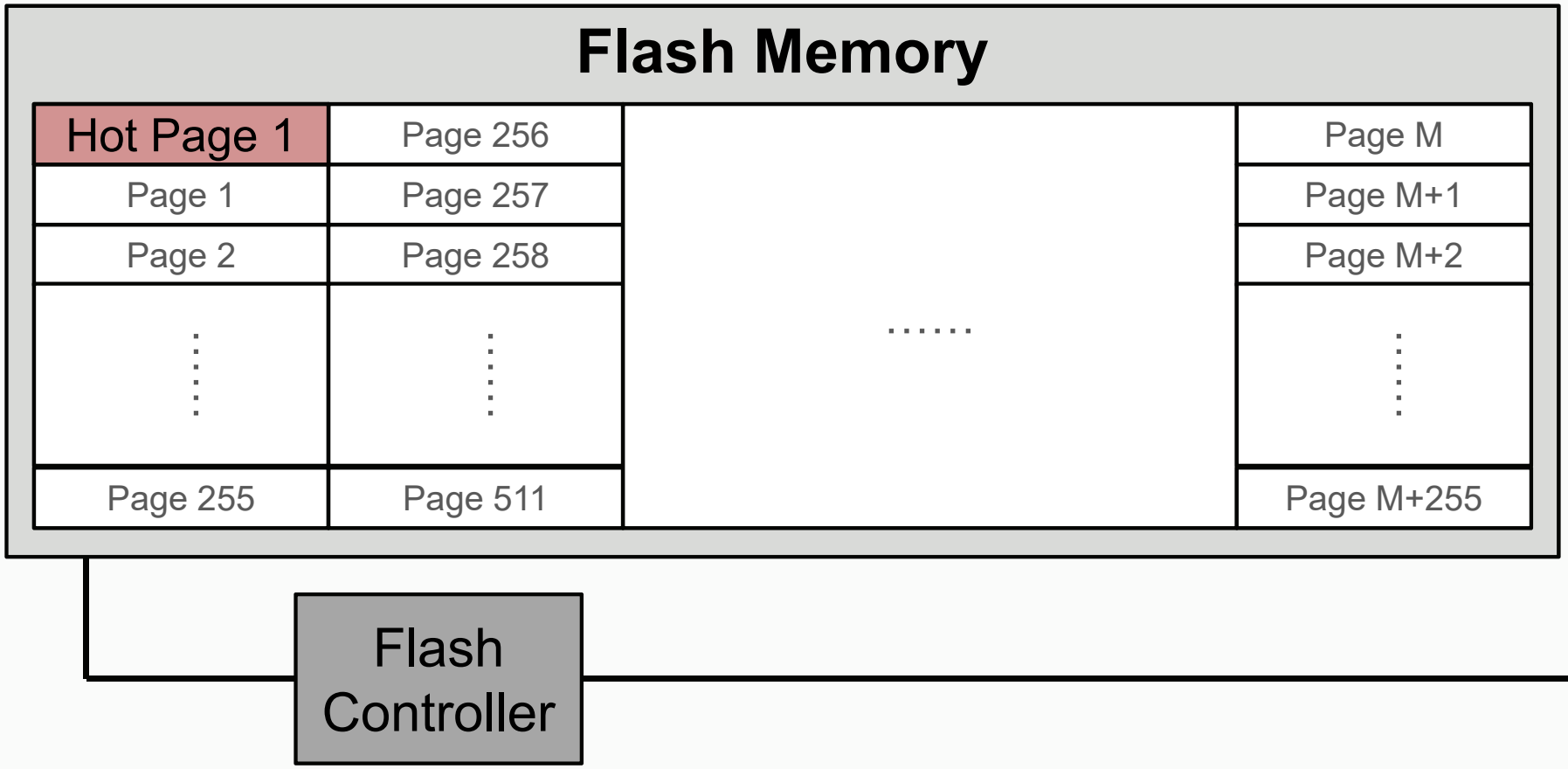


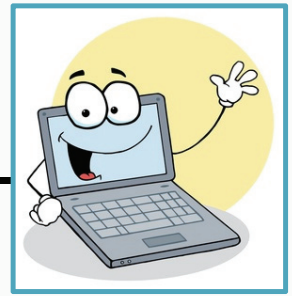
Conventional Write-hotness Oblivious Management



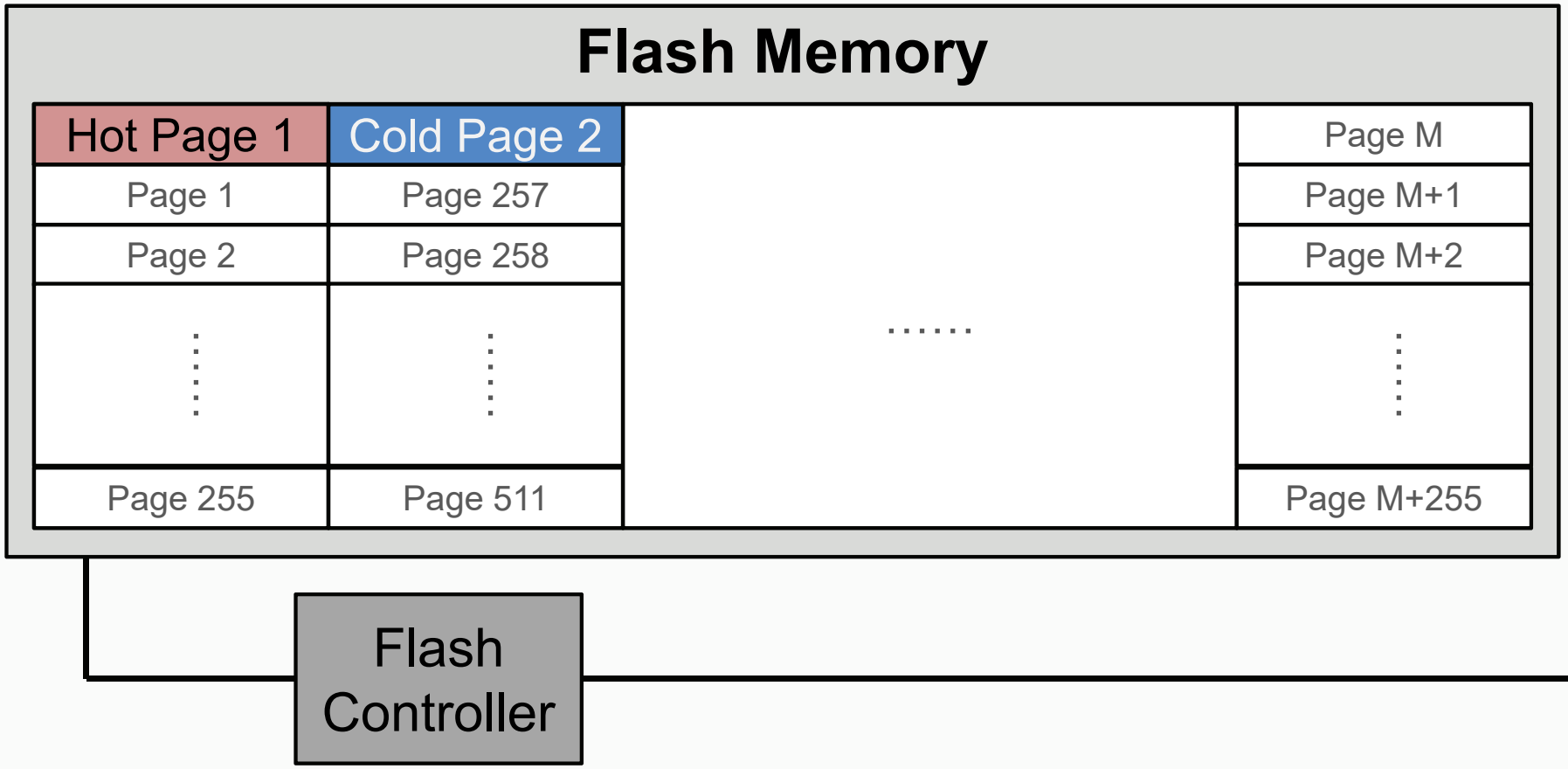
Unable to relax retention time for blocks with both write-hot and write-cold pages

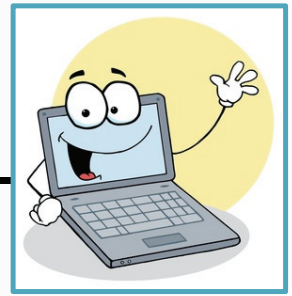
Key Idea: Write-hotness Aware Management



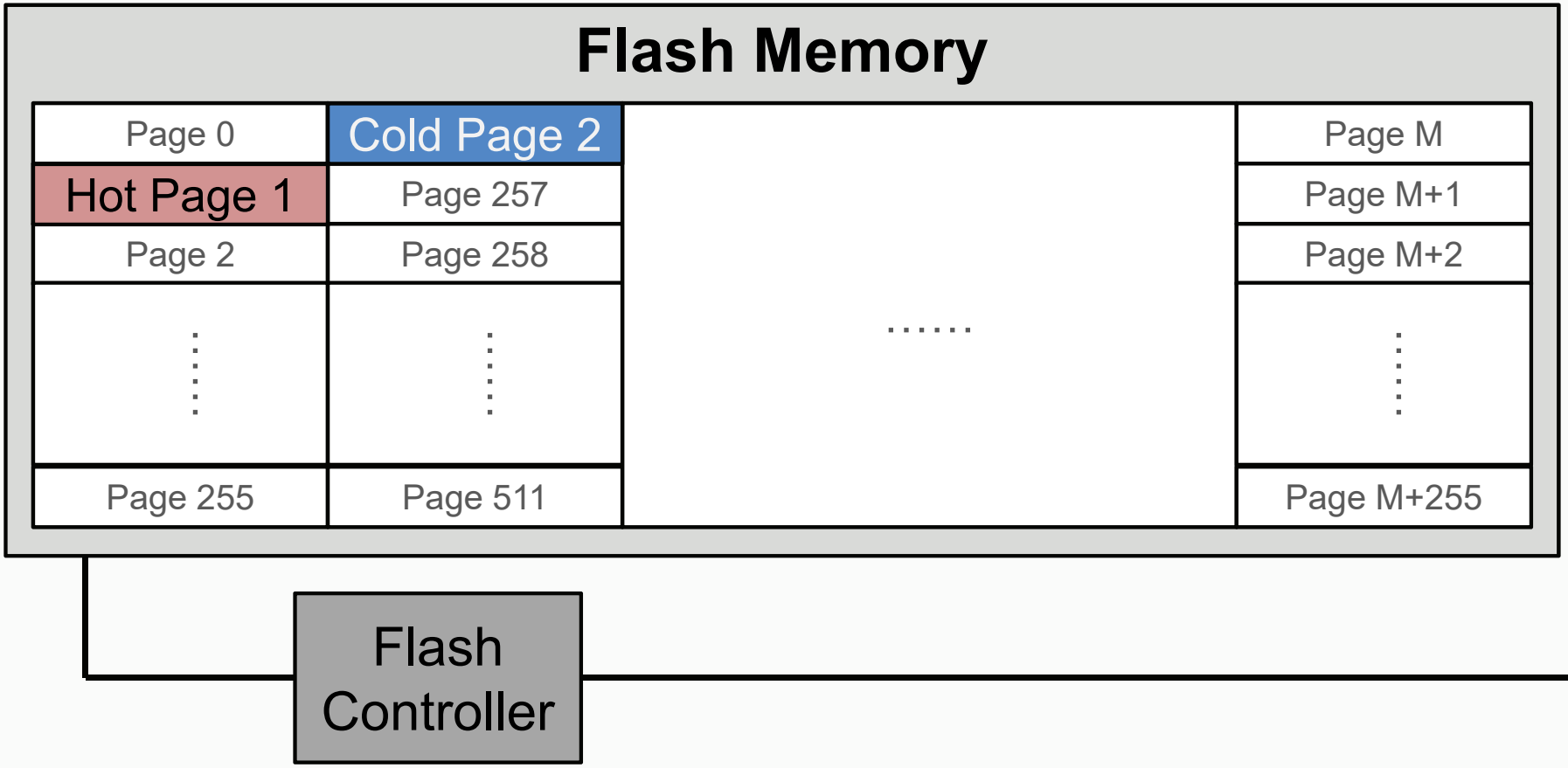


Key Idea: Write-hotness Aware Management

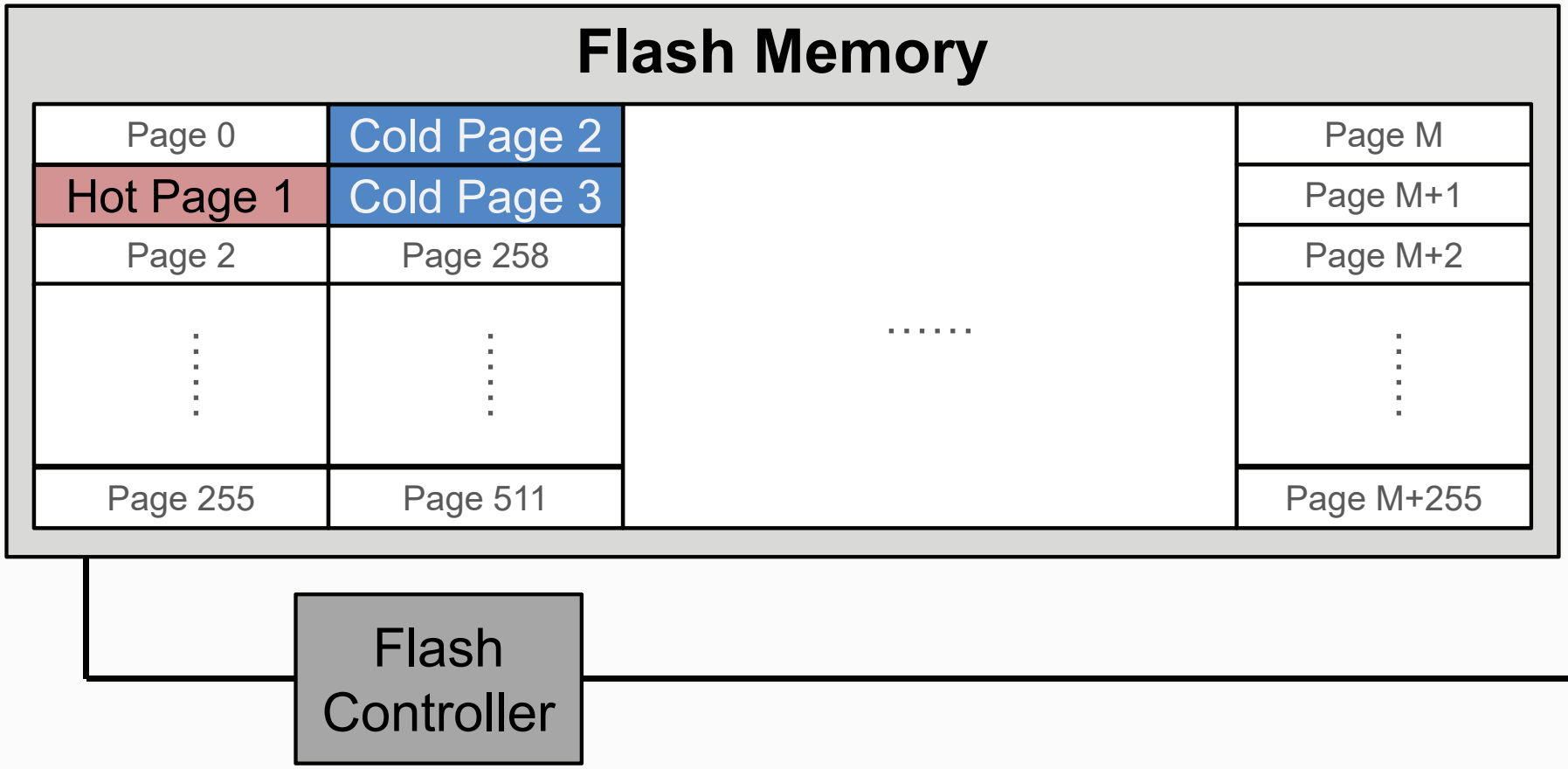


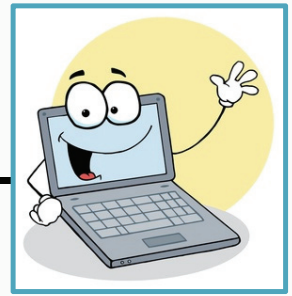


Key Idea: Write-hotness Aware Management

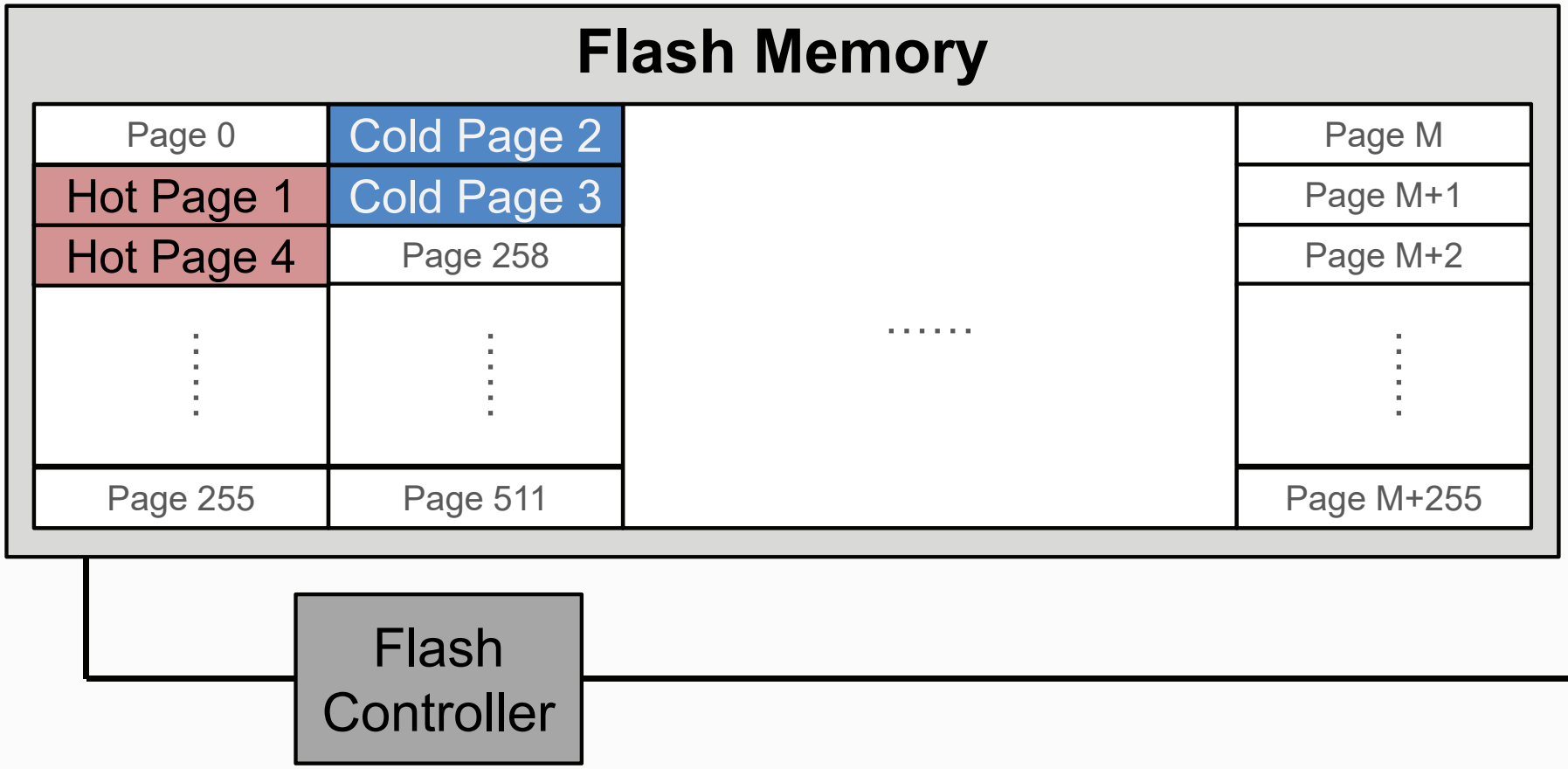


Key Idea: Write-hotness Aware Management

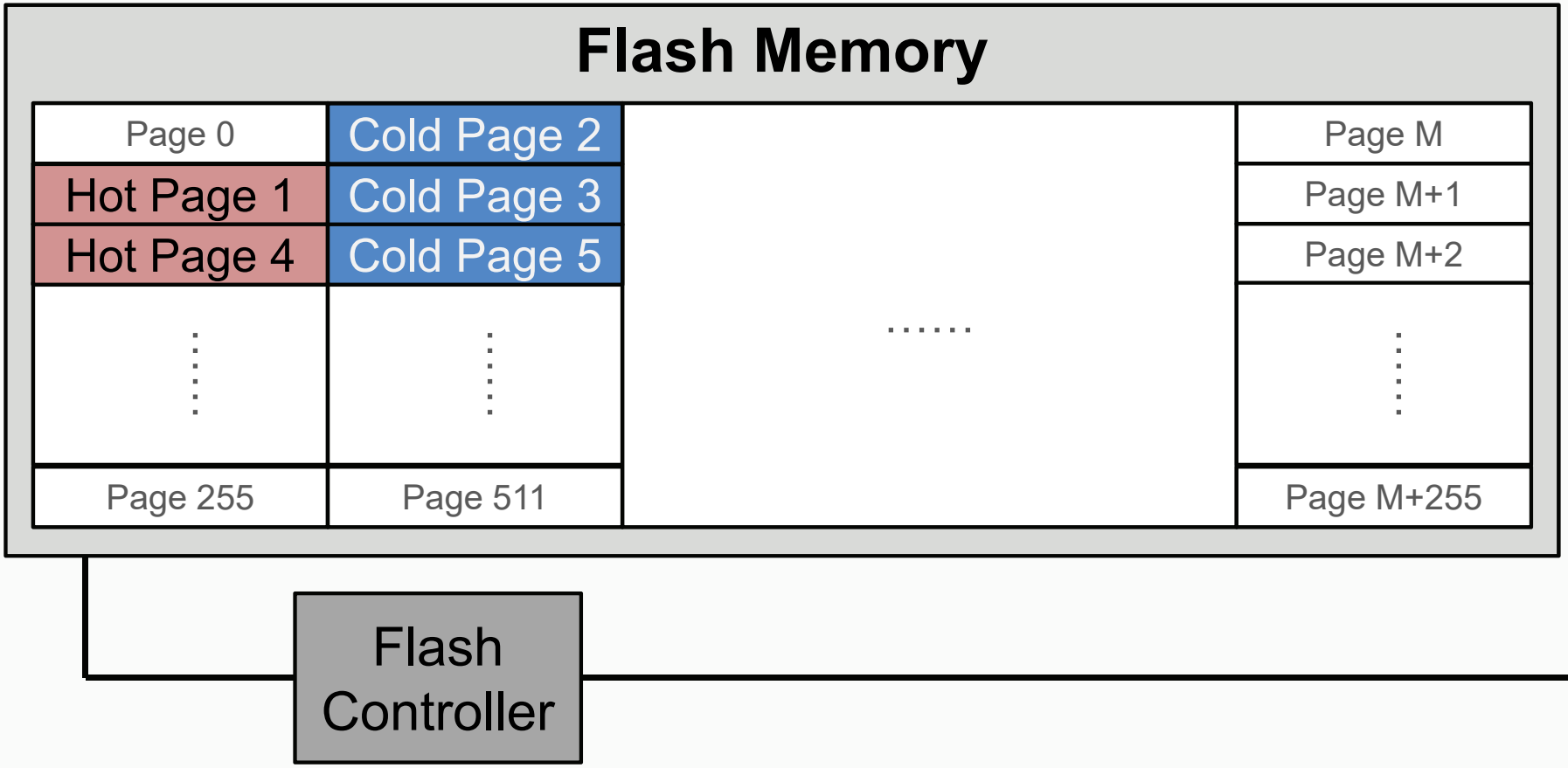




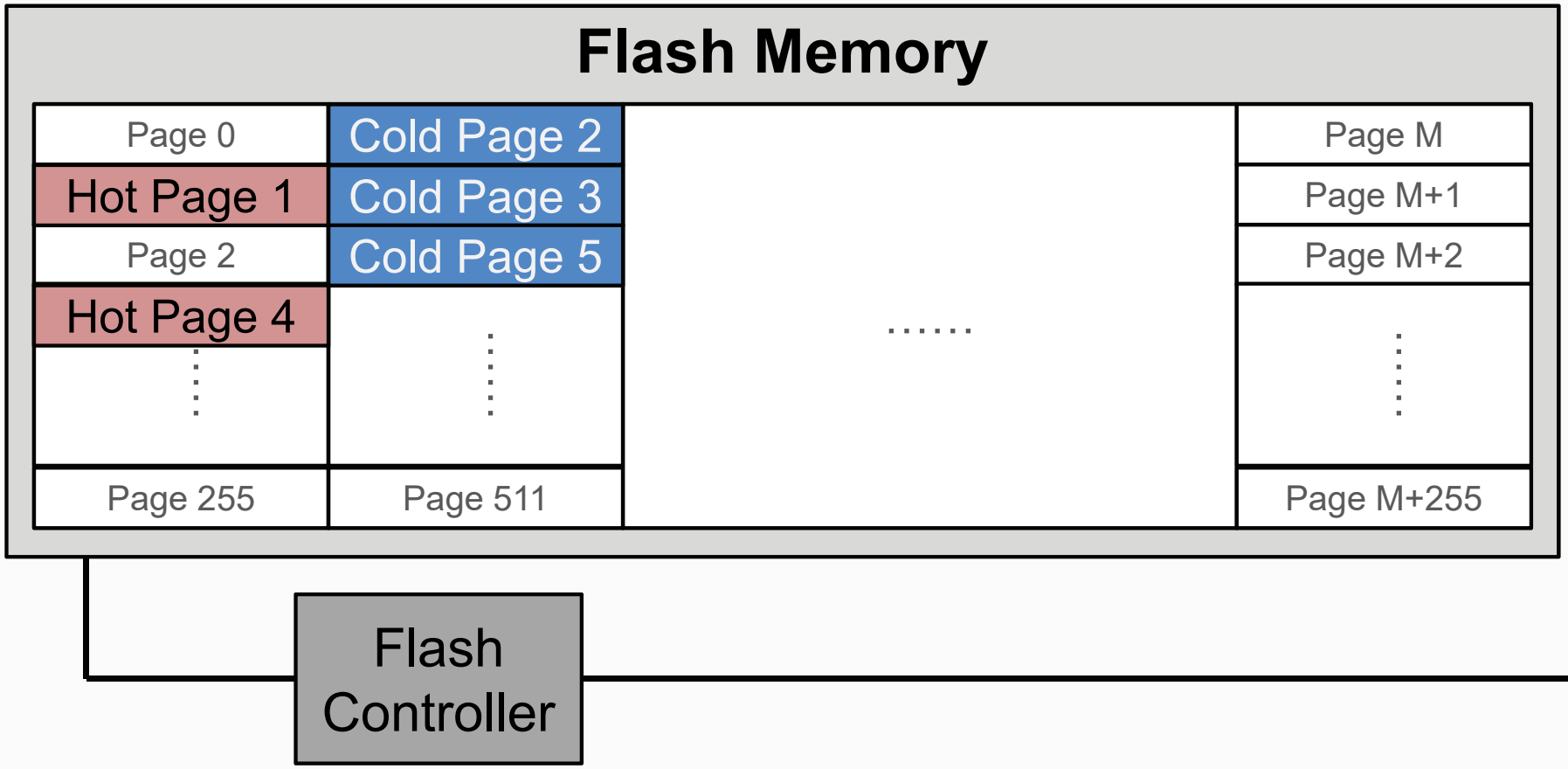
Key Idea: Write-hotness Aware Management

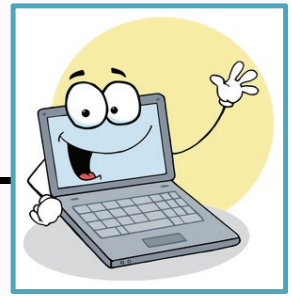


Key Idea: Write-hotness Aware Management

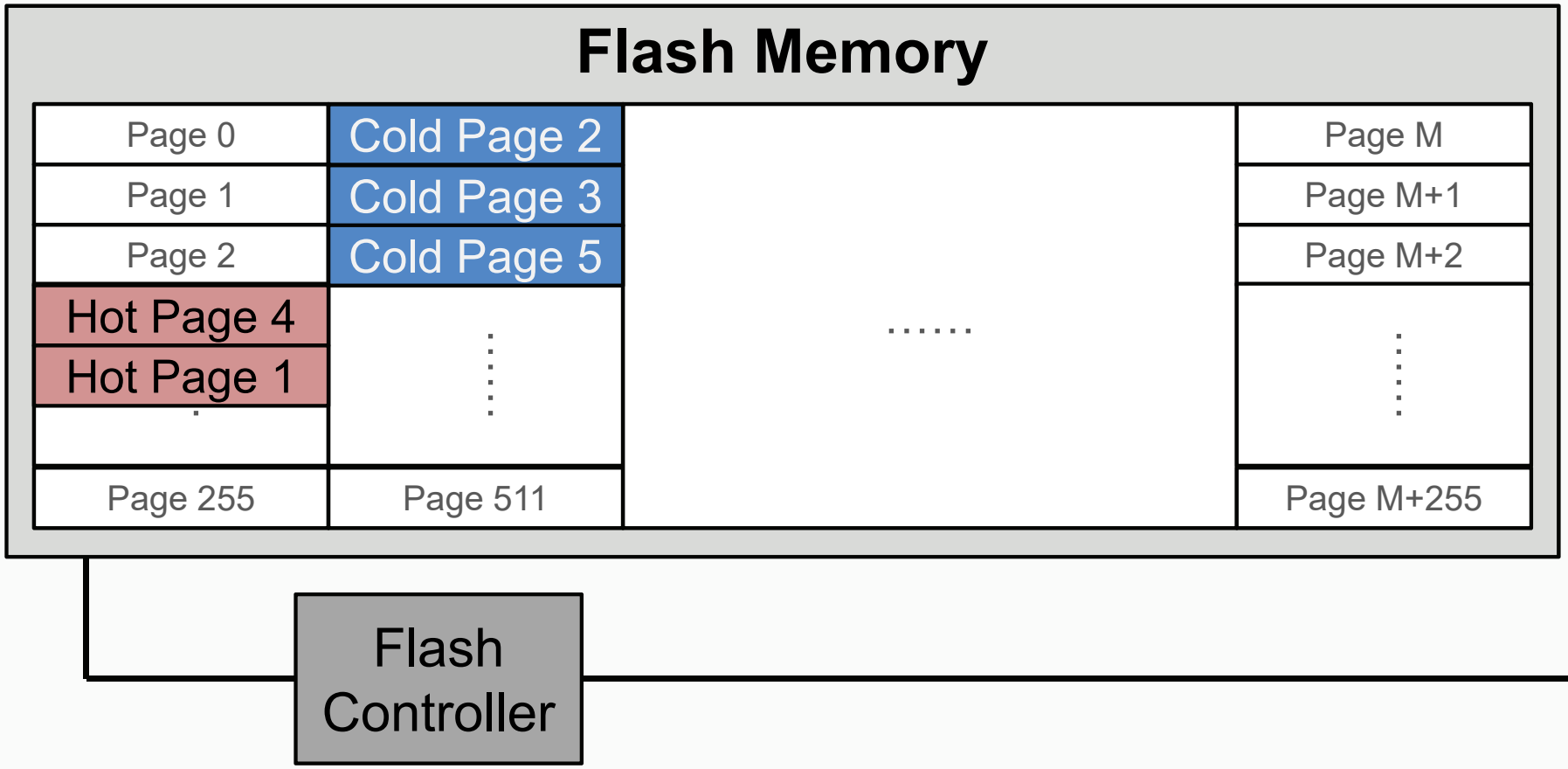


Key Idea: Write-hotness Aware Management

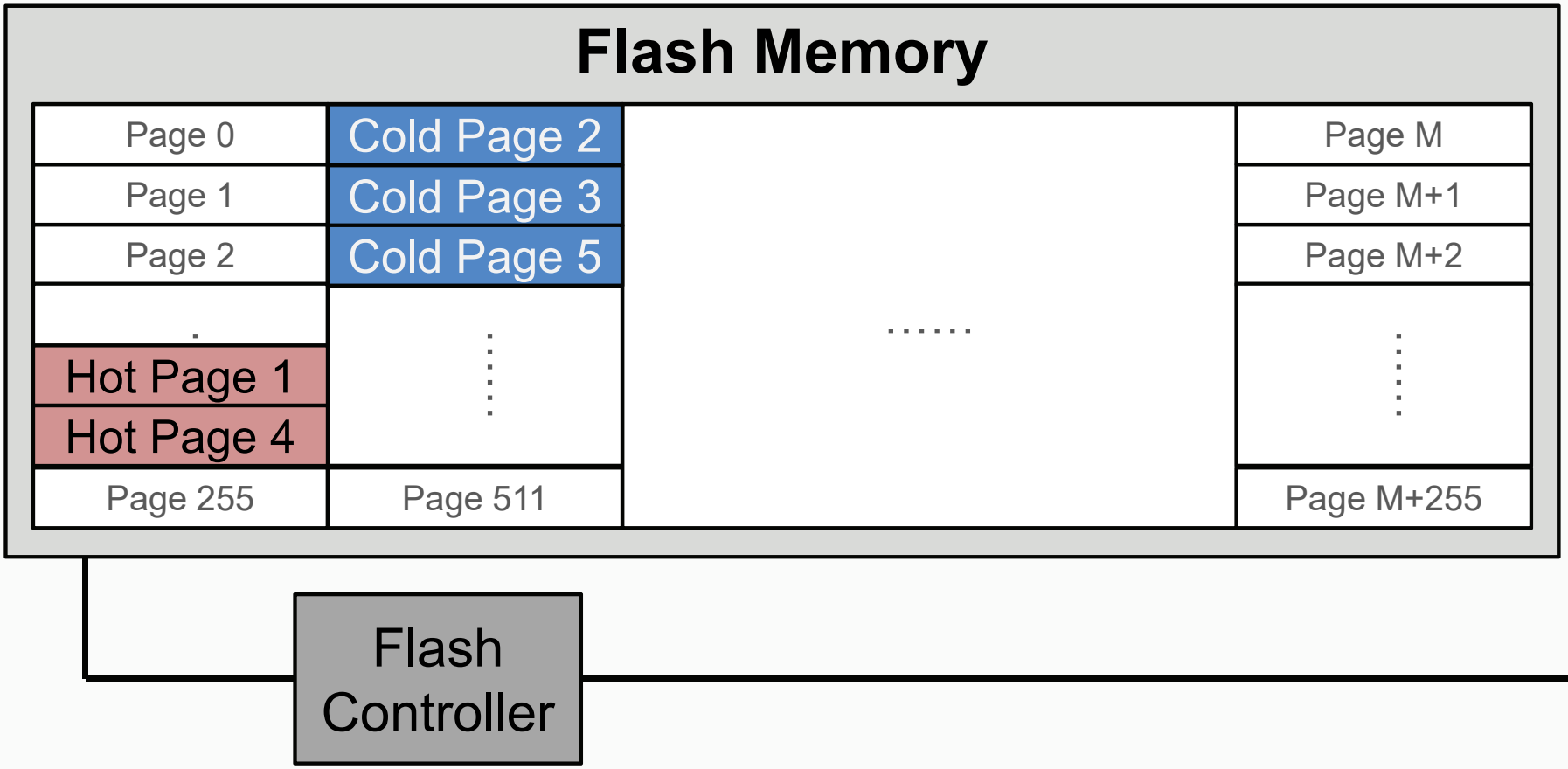




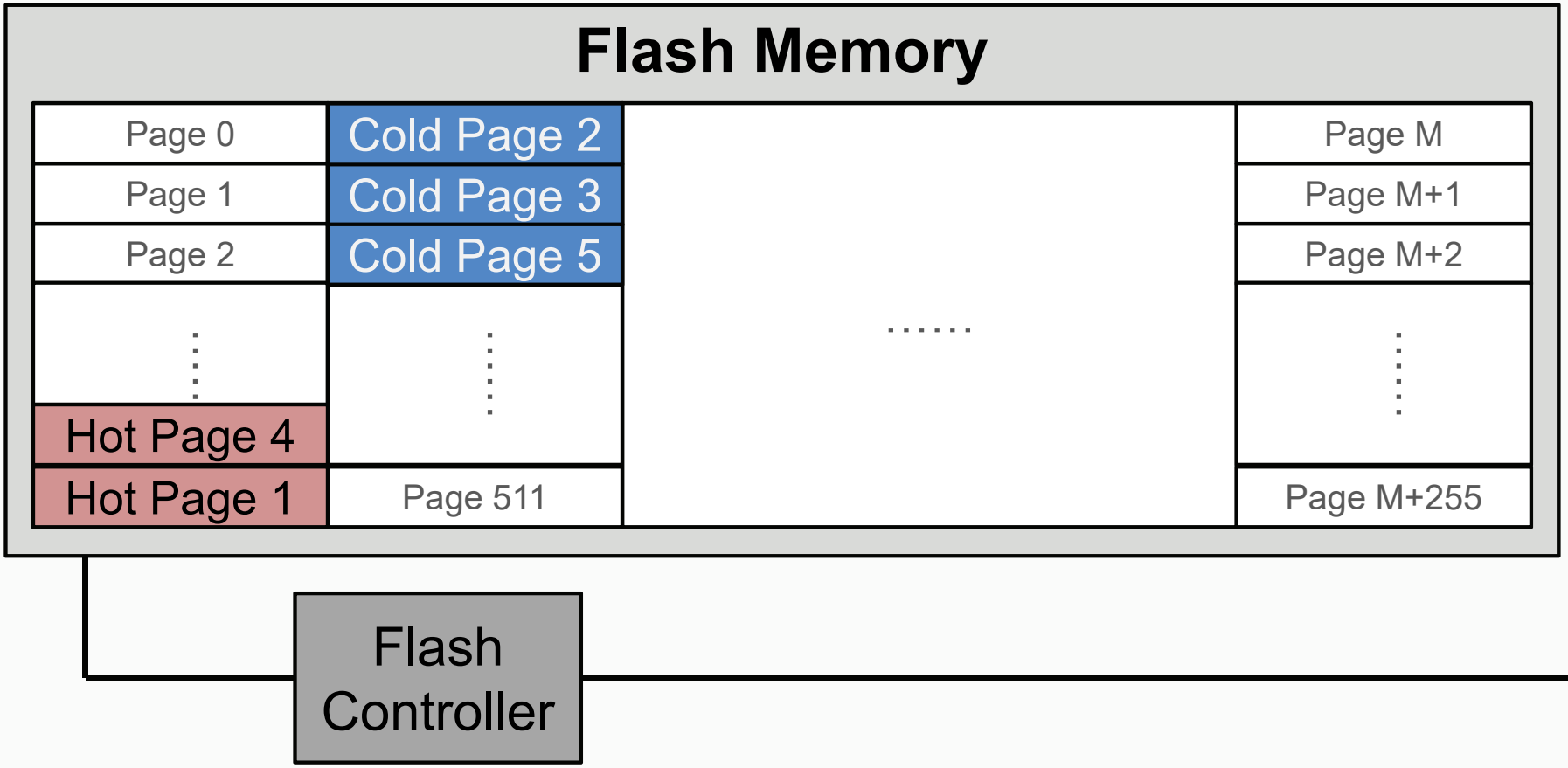
Key Idea: Write-hotness Aware Management



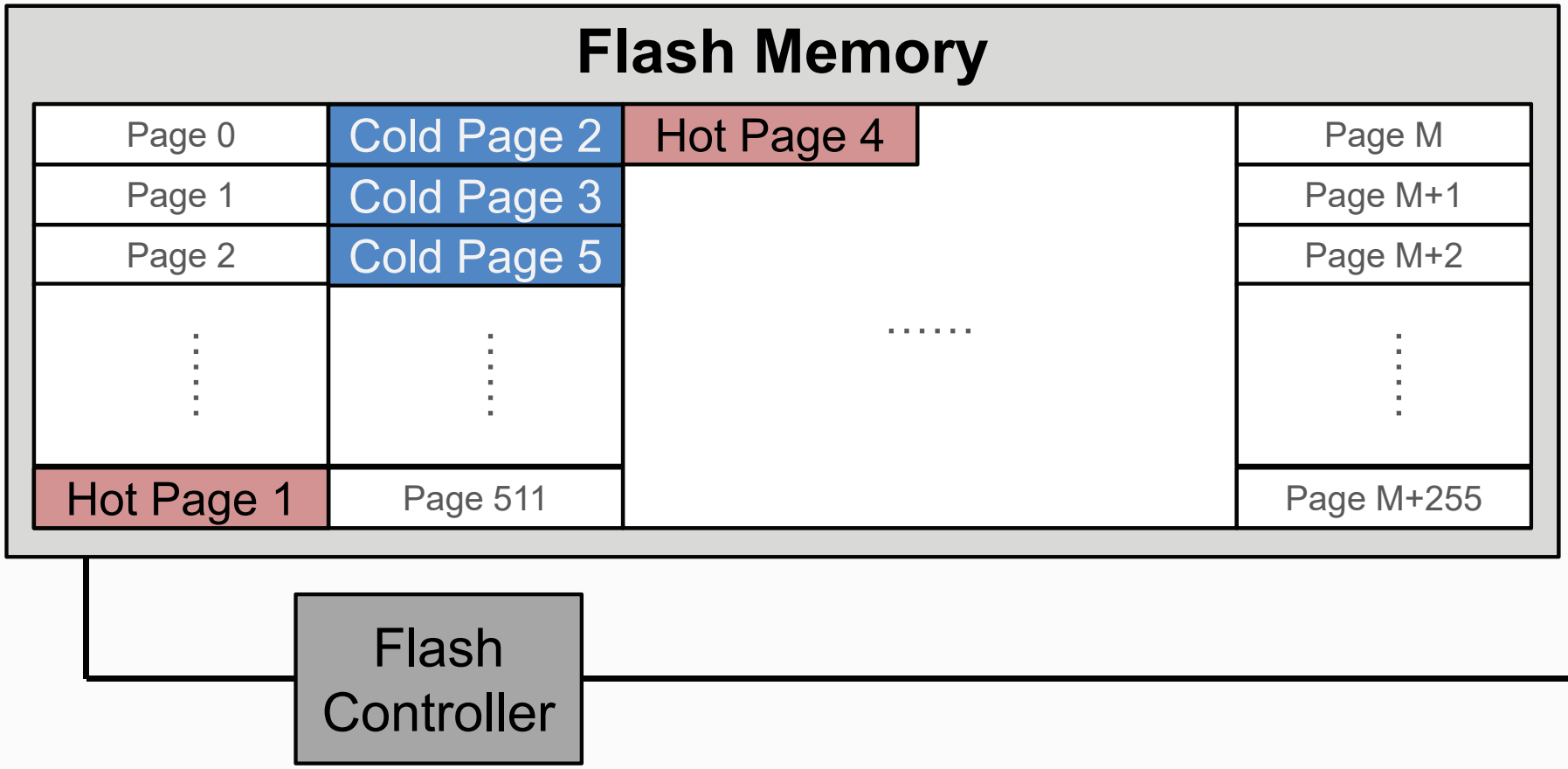
Key Idea: Write-hotness Aware Management

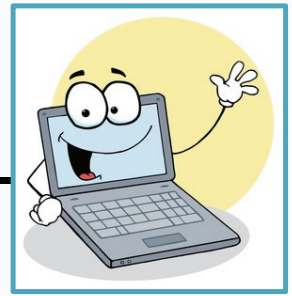


Key Idea: Write-hotness Aware Management

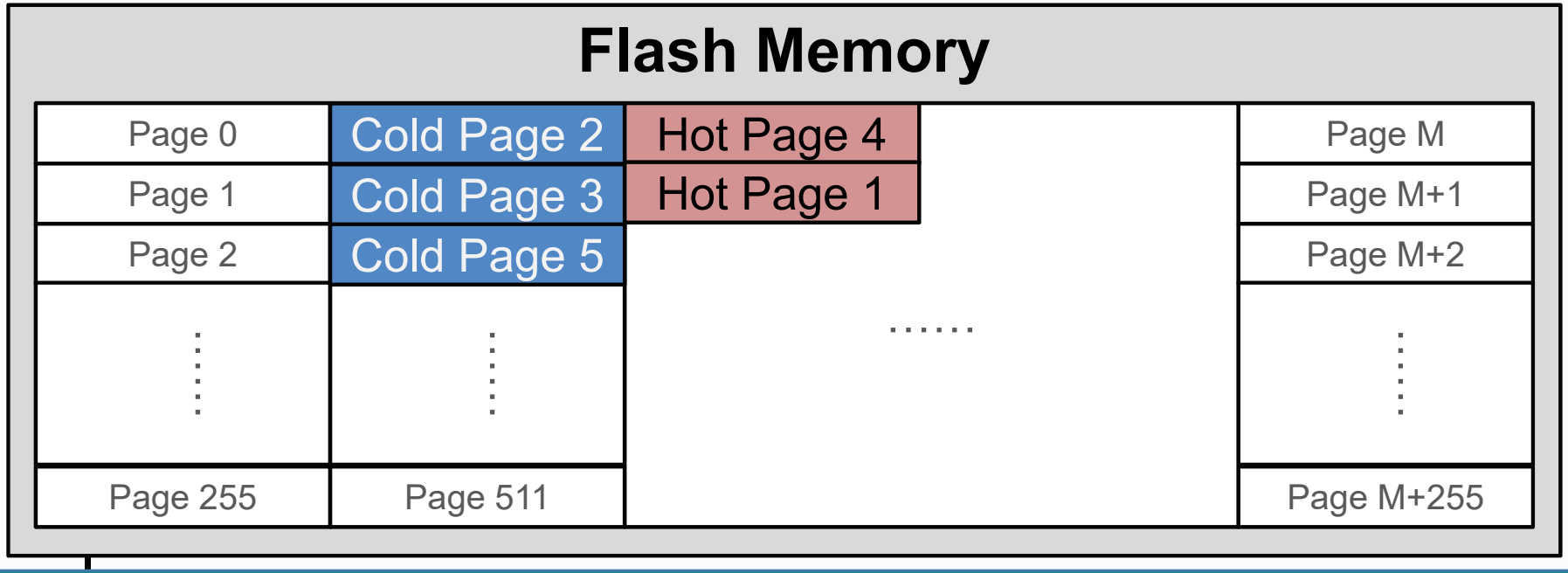


Key Idea: Write-hotness Aware Management





Key Idea: Write-hotness Aware Management

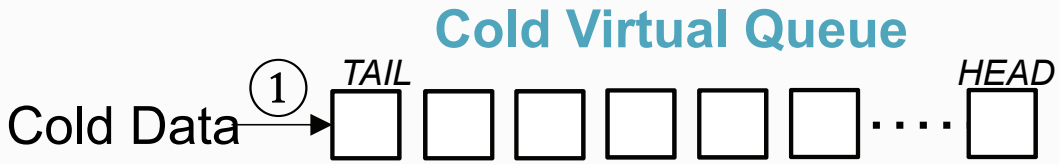


Can relax retention time for blocks with *only* write-hot pages

WARM: Write-hotness Aware Retention Management

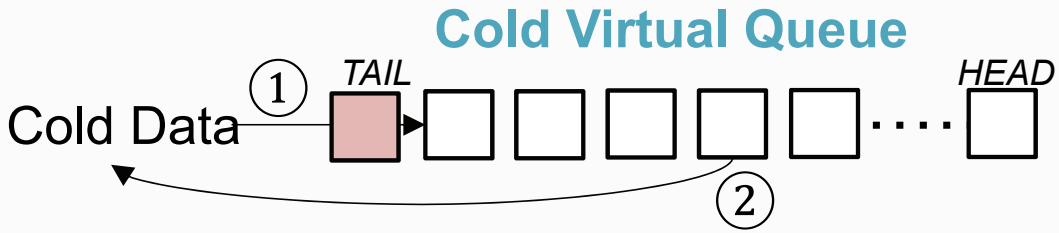
- Design Goal: Relax retention time w/o refresh for blocks that only hold **write-hot data**
- Write-hot/write-cold data partitioning algorithm
- Write-hotness aware flash policies
  - Partition write-hot and write-cold data into separate blocks
  - Skip refreshes for write-hot blocks
  - More efficient garbage collection and wear-leveling

Write-hot/Write-cold Data Partitioning Algorithm



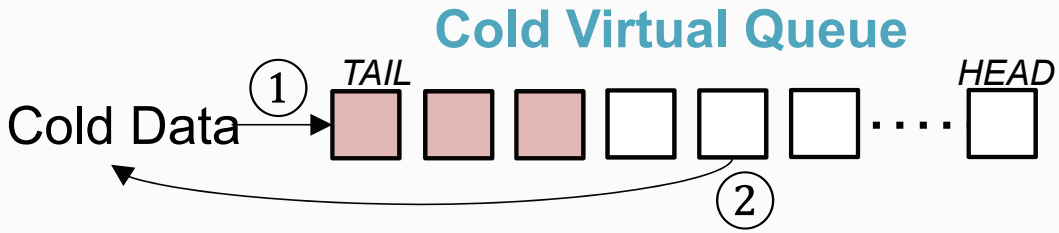
1. Initially, all data is cold and is stored in *cold virtual queue*.

Write-hot/Write-cold Data Partitioning Algorithm



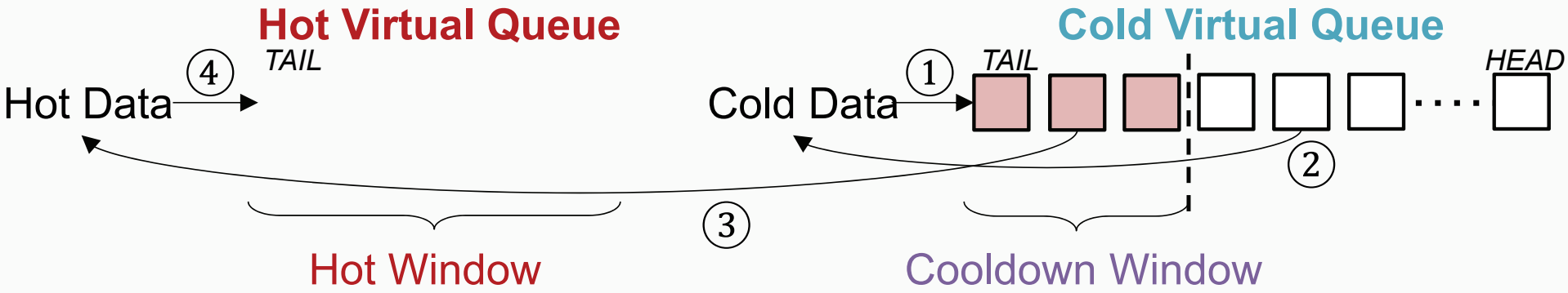
2. On write operation, data is pushed to tail of cold virtual queue.

Write-hot/Write-cold Data Partitioning Algorithm



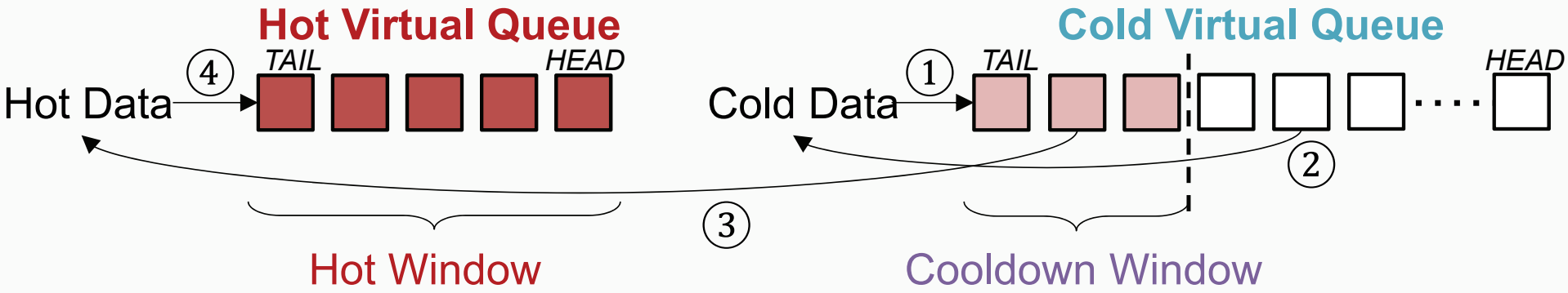
Recently written data is at tail of cold virtual queue.

Write-hot/Write-cold Data Partitioning Algorithm



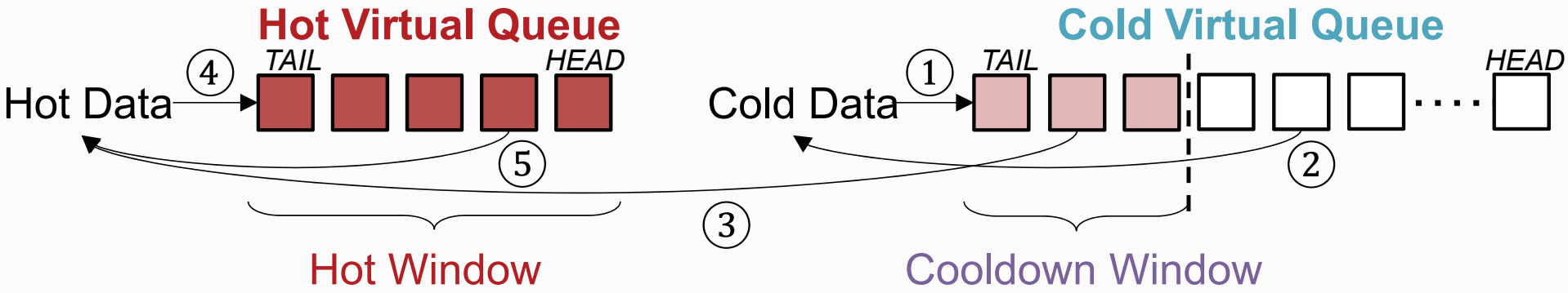
3, 4. On write hit in *cooldown window*, data is promoted to *hot virtual queue*.

Write-hot/Write-cold Data Partitioning Algorithm



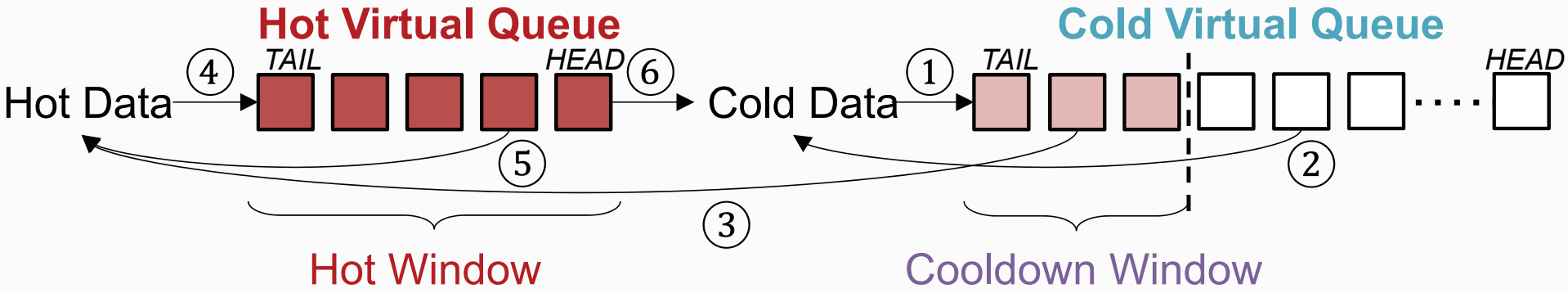
Data is sorted by write-hotness in hot virtual queue.

Write-hot/Write-cold Data Partitioning Algorithm



5. On write hit in hot virtual queue, data is pushed to the tail.

Write-hot/Write-cold Data Partitioning Algorithm

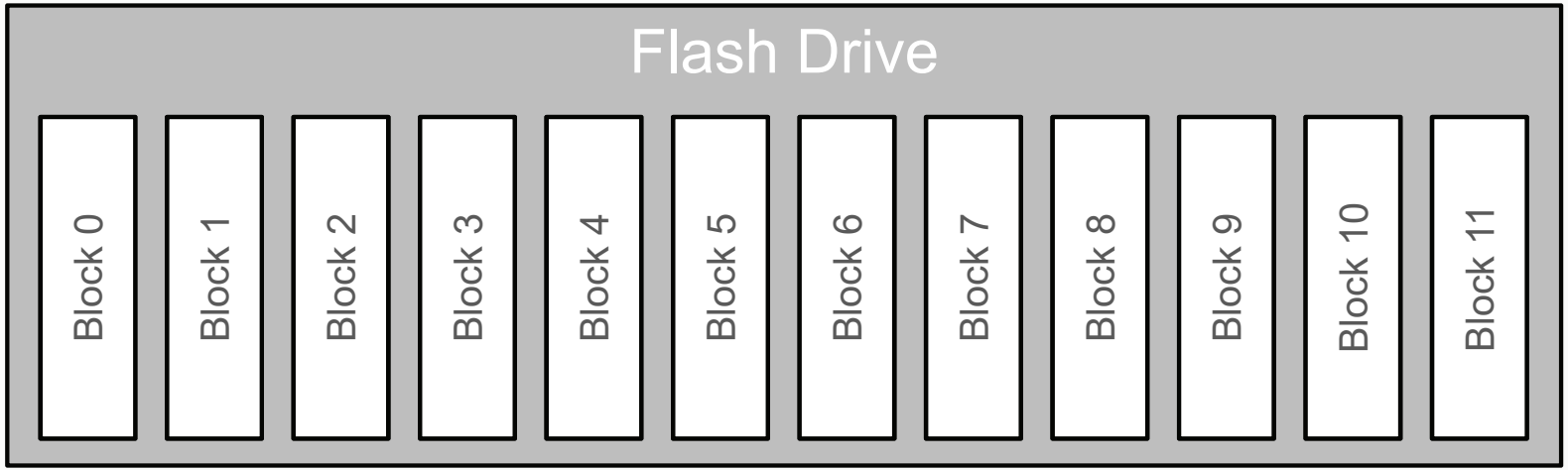


6. Unmodified hot data will be demoted to cold virtual queue.

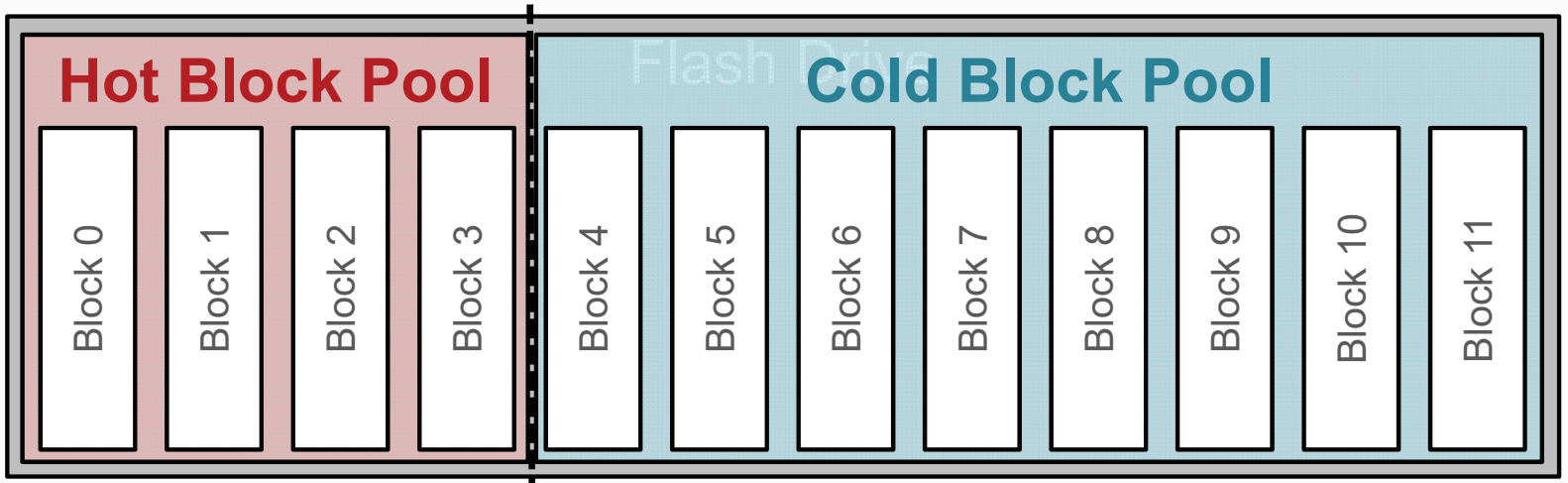
Conventional Flash Management Policies

- **Flash Translation Layer (FTL)**
  - Map data to erased blocks
  - Translate logical page number to physical page number
- **Garbage Collection**
  - Triggered before erasing a victim block
  - Remap all valid data on the victim block
- **Wear-leveling**
  - Triggered to balance wear-level among blocks

Write-hotness Aware Flash Policies



Write-hotness Aware Flash Policies



- **Write-hot data** → naturally relaxed retention time

- Program in block order
- Garbage collect in block order
- All blocks naturally wear-leveled

- **Write-cold data** → lower write frequency, less wear-out

- Conventional garbage collection
- Conventional wear-leveling algorithm

Dynamically Sizing Hot & Cold Pools

- All blocks are divided between the **hot** and **cold** block pools

1. Find the maximum **hot pool size**
2. Reduce hot virtual queue size to maximize **cold pool lifetime**
3. Size **cooldown window** to minimize ping-ponging of data between the two pools

WARM Overhead

- **Dynamic window tuning counters:** 4 × 32-bit counters
- **Cooldown window tracking:** 128 × 8-byte block IDs
- **Hot pool tracking:** 4 × 8-byte block IDs
- **Total Storage Overhead: 1.05KB**

- Minimal performance impact

Evaluation Methodology

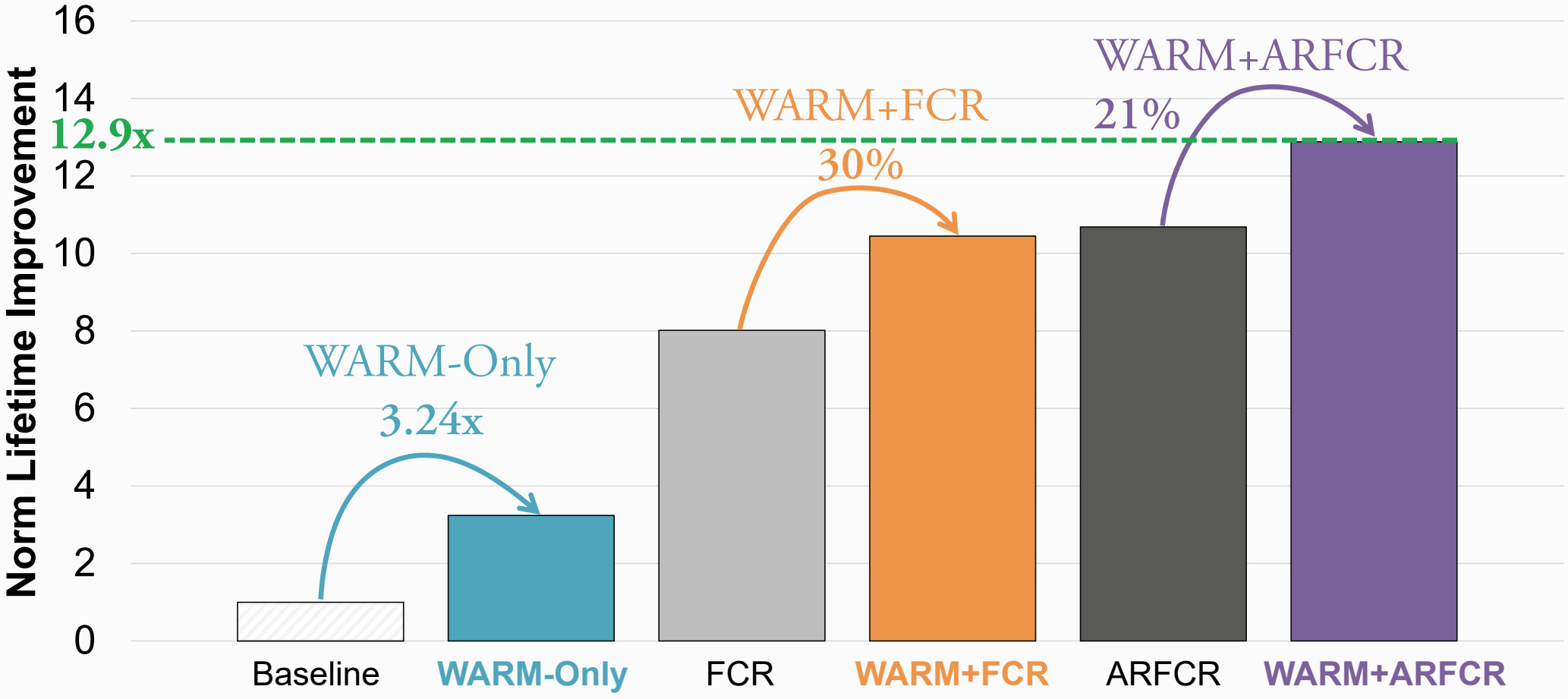
- DiskSim 4.0 + SSD model

Parameter	Value
Page read to register latency	25 µs
Page write from register latency	200 µs
Block erase latency	1.5 ms
Data bus latency	50 µs
Page/block size	8 KB/1 MB
Die/package size	8 GB/64 GB
Total capacity	256 GB
Over-provisioning	15%
Endurance for 3-year retention time	3,000 PEC
Endurance for 3-day retention time	150,000 PEC

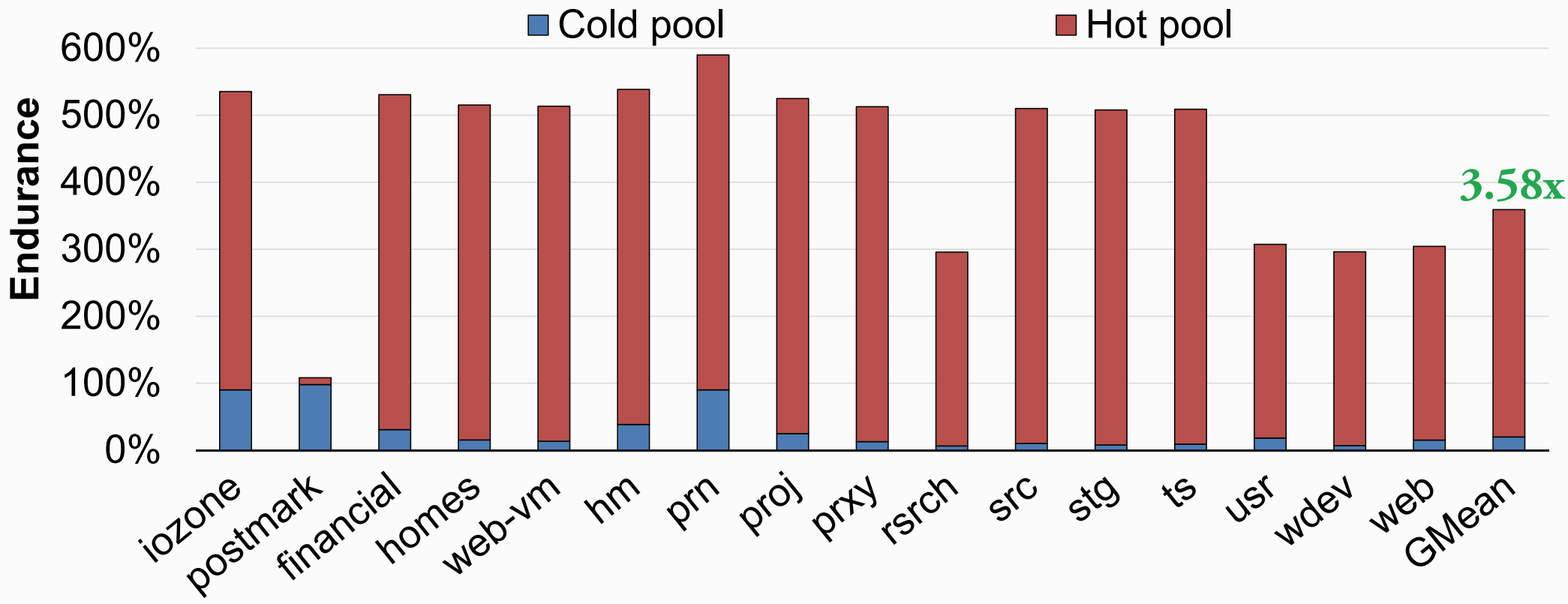
WARM Configurations

- **WARM-Only**
  - Relax retention time in hot block pool only
  - No refresh needed
- **WARM+FCR**
  - First apply **WARM-Only**
  - Then *also* relax retention time in cold block pool
  - Refresh cold blocks every 3 days
- **WARM+ARFCR**
  - Relax retention time in both hot and cold block pools
  - Adaptively increase the refresh frequency over time

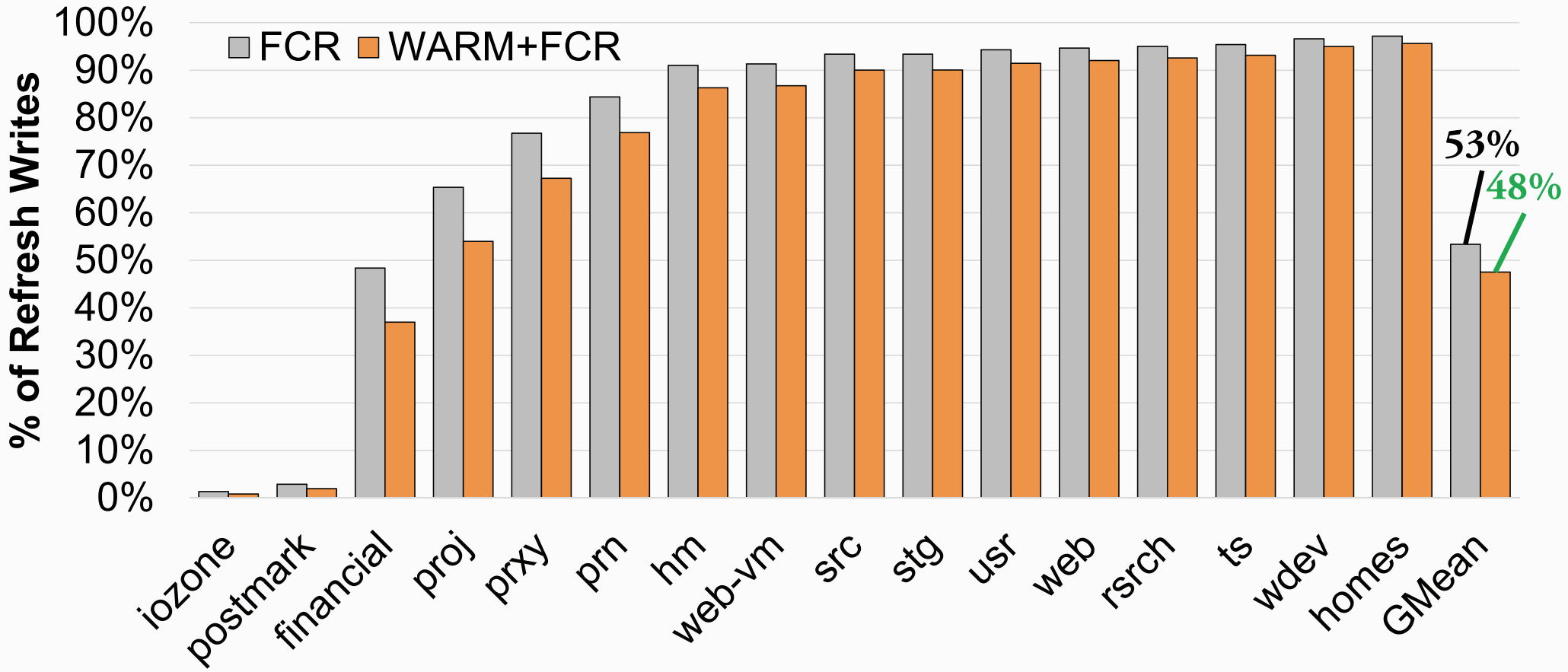
Flash Lifetime Improvements



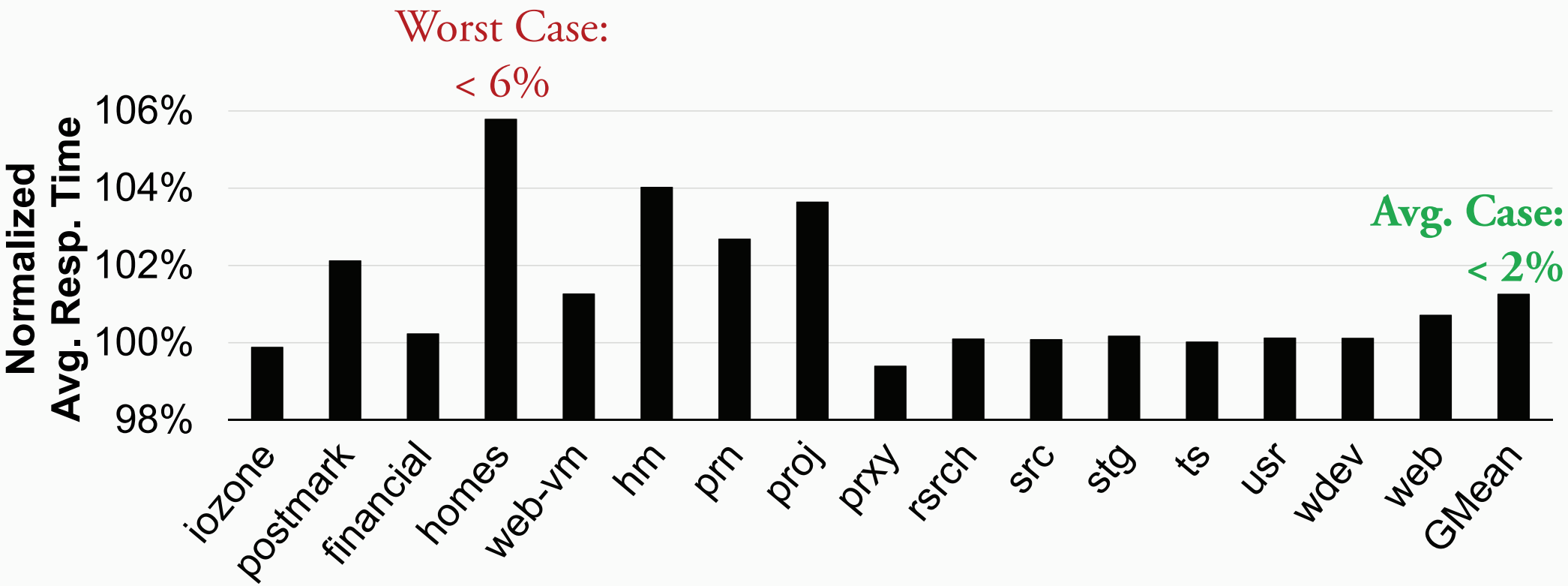
WARM-Only: Endurance Improvement



WARM+FCR: Reduction in Refresh Operations



WARM Performance Impact



Other Results in the Paper

- **Breakdown of write frequency** into host writes, garbage collection writes, refresh writes in hot/cold block pools
  - WARM reduces refresh writes significantly
  - Low garbage collection overhead
- **Sensitivity to different capacity over-provisioning amounts**
  - WARM improves flash lifetime more as over-provisioning increases
- **Sensitivity to different refresh intervals**
  - WARM improves flash lifetime more as refresh frequency increases

- Y. Luo et al., *Improving NAND Flash Memory Lifetime with Write-hotness Aware Retention Management*. MSST 2015.

Conclusion

- Flash memory can achieve up to **50x endurance improvement by relaxing retention time using refresh** [Cai+ ICCD '12]
  - Problem: **Refresh consumes majority of endurance improvement**
  - Goal: Reduce refresh overhead to increase flash memory lifetime
- Key Observation: **Refresh unnecessary for *write-hot data***
- Write-hotness Aware Retention Management (WARM)
  - Physically **partition write-hot pages and write-cold pages** within flash
  - Apply *different* policies (garbage collection, wear-leveling, refresh) to each group
- WARM w/o refresh **improves lifetime by 3.24x** w/ 1.05KB overhead
- WARM w/ adaptive refresh **improves lifetime by 12.9x** (1.21x over refresh)

Other SAFARI Research on NAND Flash

- Flash Memory Summit 2016 Talks
  - Pre-Conference Seminar C: **Onur Mutlu**, *Software-Transparent Crash Consistency for Persistent Memory* [Ren+ MICRO '15]
  - Forum E-22: **Yixin Luo**, *Practical Threshold Voltage Distribution Modeling* [Luo+ JSAC '16]
  - Forum F-22: **Onur Mutlu**, *Large-Scale Study of In-the-Field Flash Failures* [Meza+ SIGMETRICS '15]
- Two posters in **Booth 933**
- All available at http://www.ece.cmu.edu/~safari/

Other SAFARI Research on NAND Flash

- Retention noise study and management [Cai+ ICCD '12, Cai+ HPCA '15]
- Flash-based SSD prototyping and testing platform [Cai+ FCCM '11]
- Overall flash error analysis [Cai+ DATE '12, Cai+ ITJ '13]
- Program and erase noise study [Cai+ DATE '13]
- Cell-to-cell interference study and tolerance [Cai+ ICCD '13, Cai+ SIGMETRICS '14]
- Read disturb noise study and mitigation [Cai+ DSN '15]
- Flash errors in the field [Meza+ SIGMETRICS '15]
- Persistent memory [Ren+ MICRO '15]

- All available at http://www.ece.cmu.edu/~safari/

Write-hotness Aware Retention Management: Efficient Hot/Cold Data Partitioning for SSDs

Saugata Ghose
ghose@cmu.edu ▪ http://ece.cmu.edu/~saugatag/

joint work with Yixin Luo, Yu Cai, Jongmoo Choi, Onur Mutlu

August 10, 2016
Santa Clara, CA

References to Papers and Talks

Our FMS Talks and Posters

- Onur Mutlu, ThyNVM: Software-Transparent Crash Consistency for Persistent Memory, FMS 2016.
- Onur Mutlu, Large-Scale Study of In-the-Field Flash Failures, FMS 2016.
- Yixin Luo, Practical Threshold Voltage Distribution Modeling, FMS 2016.
- Saugata Ghose, Write-hotness Aware Retention Management, FMS 2016.
- Onur Mutlu, Read Disturb Errors in MLC NAND Flash Memory, FMS 2015.
- Yixin Luo, Data Retention in MLC NAND Flash Memory, FMS 2015.
- Onur Mutlu, Error Analysis and Management for MLC NAND Flash Memory, FMS 2014.

- FMS 2016 posters:
  - WARM: Improving NAND Flash Memory Lifetime with Write-hotness Aware Retention Management
  - Read Disturb Errors in MLC NAND Flash Memory
  - Data Retention in MLC NAND Flash Memory

Our Flash Memory Works (I)

- **Retention noise study and management**
  - Yu Cai, Gulay Yalcin, Onur Mutlu, Erich F. Haratsch, Adrian Cristal, Osman Unsal, and Ken Mai, Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime, ICCD 2012.
  - Yu Cai, Yixin Luo, Erich F. Haratsch, Ken Mai, and Onur Mutlu, Data Retention in MLC NAND Flash Memory: Characterization, Optimization and Recovery, HPCA 2015.
  - Yixin Luo, Yu Cai, Saugata Ghose, Jongmoo Choi, and Onur Mutlu, WARM: Improving NAND Flash Memory Lifetime with Write-hotness Aware Retention Management, MSST 2015.

- **Flash-based SSD prototyping and testing platform**
  - Yu Cai, Erich F. Haratsh, Mark McCartney, Ken Mai, FPGA-based solid-state drive prototyping platform, FCCM 2011.

Our Flash Memory Works (II)

- **Overall flash error analysis**
  - Yu Cai, Erich F. Haratsch, Onur Mutlu, and Ken Mai, Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis, DATE 2012.
  - Yu Cai, Gulay Yalcin, Onur Mutlu, Erich F. Haratsch, Adrian Cristal, Osman Unsal, and Ken Mai, Error Analysis and Retention-Aware Error Management for NAND Flash Memory, ITJ 2013.

- **Program and erase noise study**
  - Yu Cai, Erich F. Haratsch, Onur Mutlu, and Ken Mai, Threshold Voltage Distribution in MLC NAND Flash Memory: Characterization, Analysis and Modeling, DATE 2013.

Our Flash Memory Works (III)

- **Cell-to-cell interference characterization and tolerance**
  - Yu Cai, Onur Mutlu, Erich F. Haratsch, and Ken Mai, Program Interference in MLC NAND Flash Memory: Characterization, Modeling, and Mitigation, ICCD 2013.
  - Yu Cai, Gulay Yalcin, Onur Mutlu, Erich F. Haratsch, Osman Unsal, Adrian Cristal, and Ken Mai, Neighbor-Cell Assisted Error Correction for MLC NAND Flash Memories, SIGMETRICS 2014.

- **Read disturb noise study**
  - Yu Cai, Yixin Luo, Saugata Ghose, Erich F. Haratsch, Ken Mai, and Onur Mutlu, Read Disturb Errors in MLC NAND Flash Memory: Characterization and Mitigation, DSN 2015.

Our Flash Memory Works (IV)

- Flash errors in the field
  - Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu, A Large-Scale Study of Flash Memory Errors in the Field, SIGMETRICS 2015.

- Persistent memory
  - Jinglei Ren, Jishen Zhao, Samira Khan, Jongmoo Choi, Yongwei Wu, and Onur Mutlu, ThyNVM: Enabling Software-Transparent Crash Consistency in Persistent Memory Systems, MICRO 2015.

Referenced Papers and Talks

- All are available at
  - http://users.ece.cmu.edu/~omutlu/projects.htm
  - http://users.ece.cmu.edu/~omutlu/talks.htm

- And, many other previous works on
  - Challenges and opportunities in memory
  - NAND flash memory errors and management
  - Phase change memory as DRAM replacement
  - STT-MRAM as DRAM replacement
  - Taking advantage of persistence in memory
  - Hybrid DRAM + NVM systems
  - NVM design and architecture