SuperMem: Enabling Application-transparent Secure Persistent Memory with Low Overheads

Pengfei Zuo[1,2], Yu Hua[1], Yuan Xie[2]

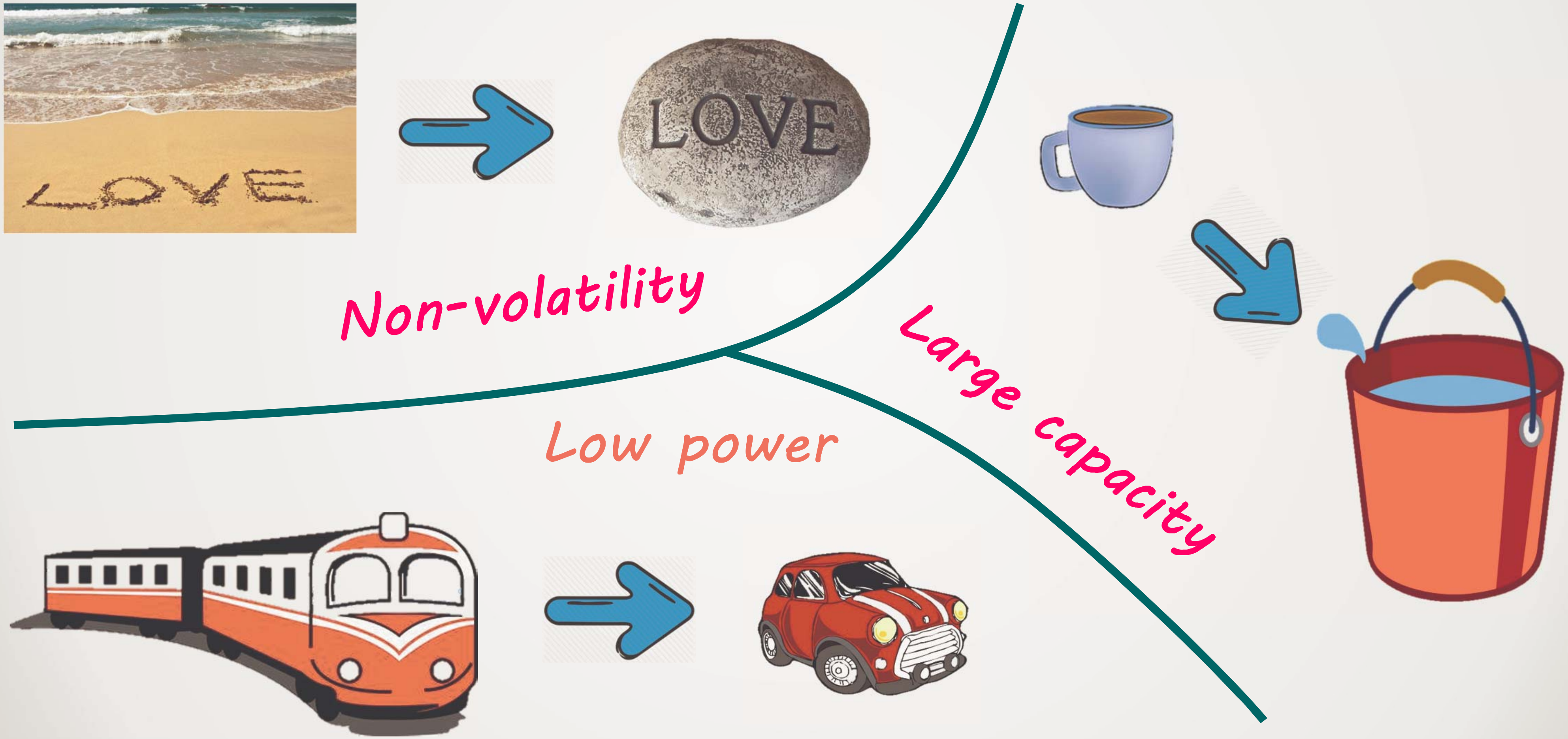
[1] *Huazhong University of Science and Technology, China*

[2] *University of California at Santa Barbara, USA*

52nd IEEE/ACM International Symposium on Microarchitecture (**MICRO**), 2019

DRAM → Persistent Memory

Images from Internet

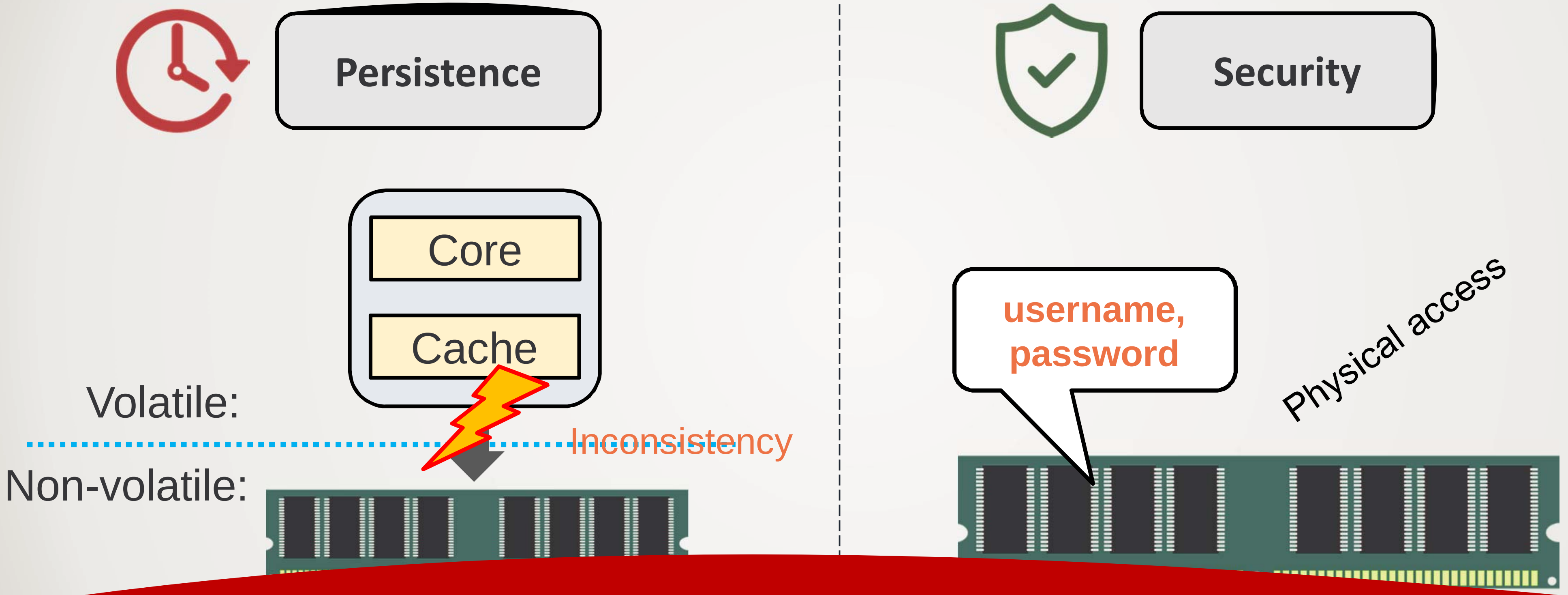


Non-volatility

Low power

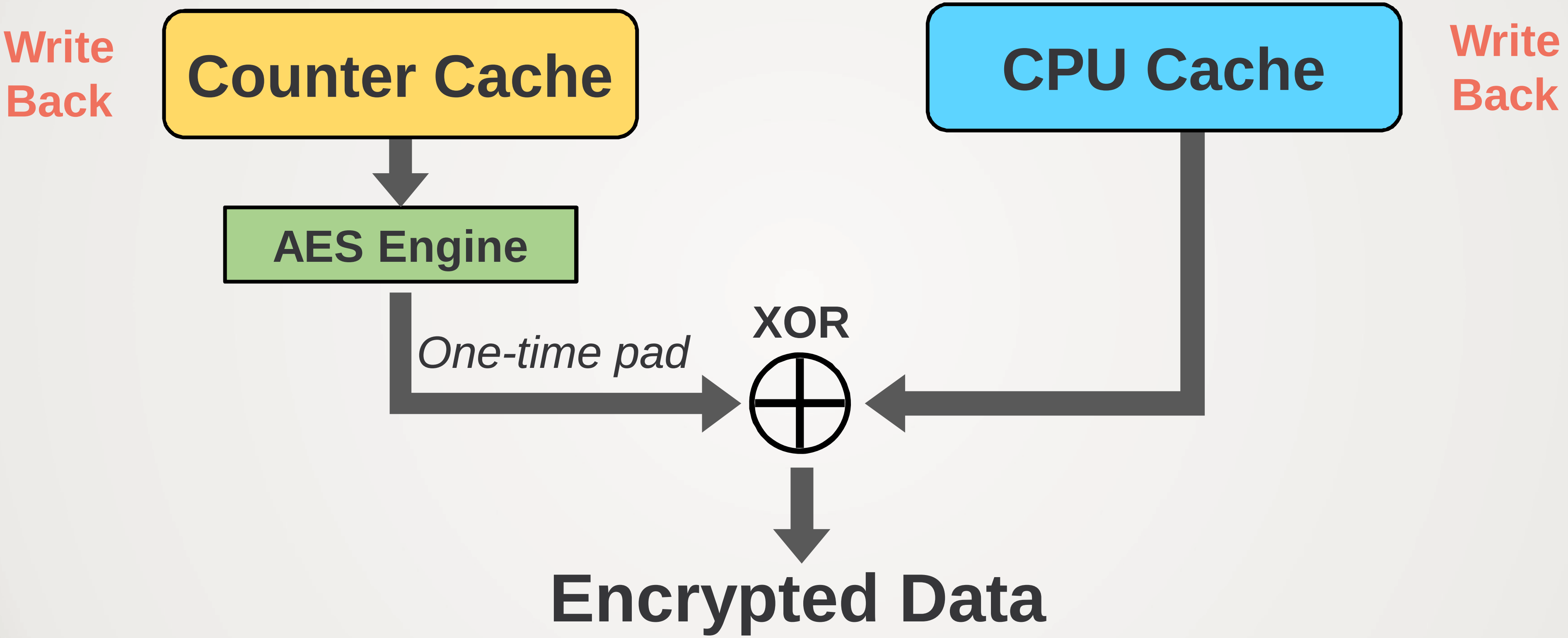
Large capacity

Two Key Challenges for Persistent Memory

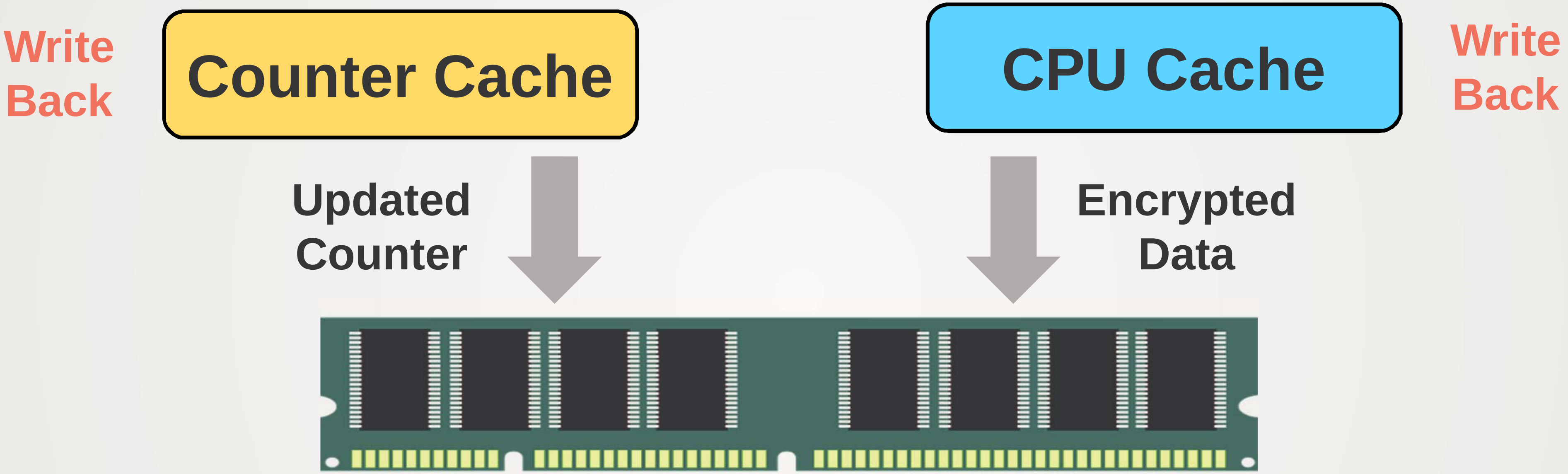


Gap between persistence and security:
Encryption incurs new inconsistency problem

Counter Mode Encryption



Counter Mode Encryption



Write Back

Counter Cache

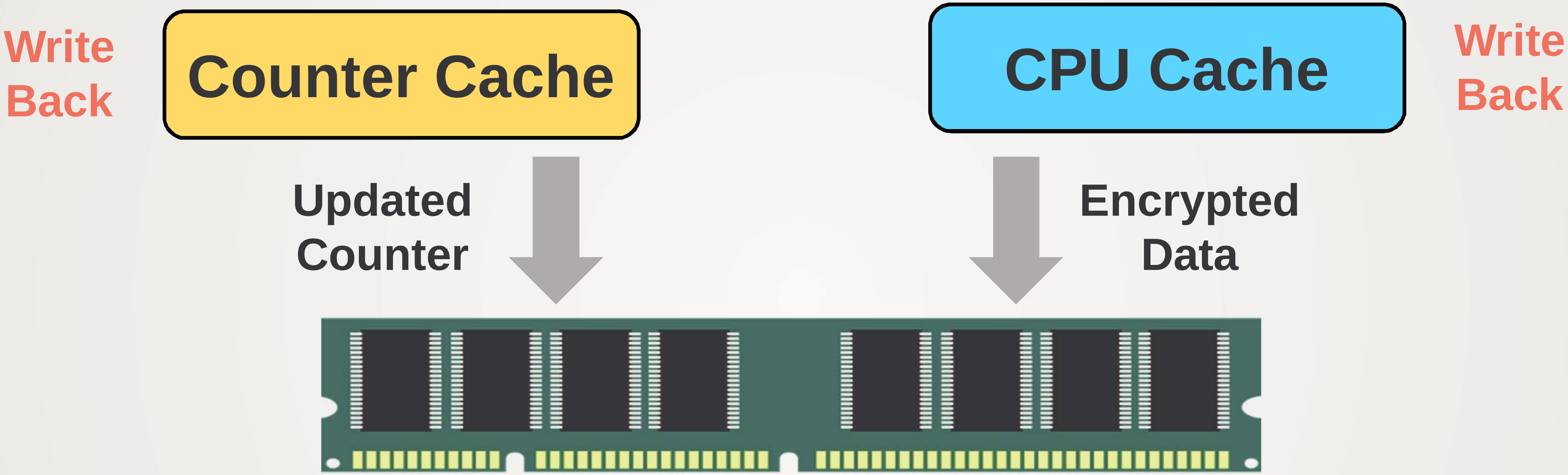
Updated Counter

CPU Cache

Write Back

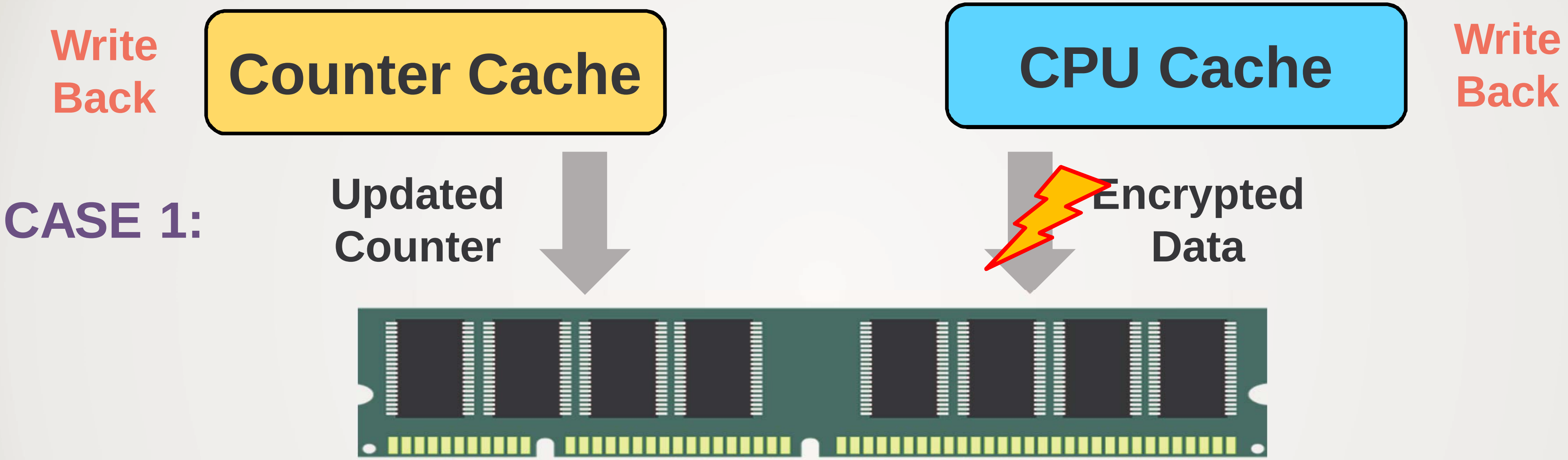
Encrypted Data

Crash Inconsistency Caused by Encryption



- Data and counter cannot reach NVM at the same time

Crash Inconsistency Caused by Encryption



Write Back

Counter Cache

CPU Cache

Write Back

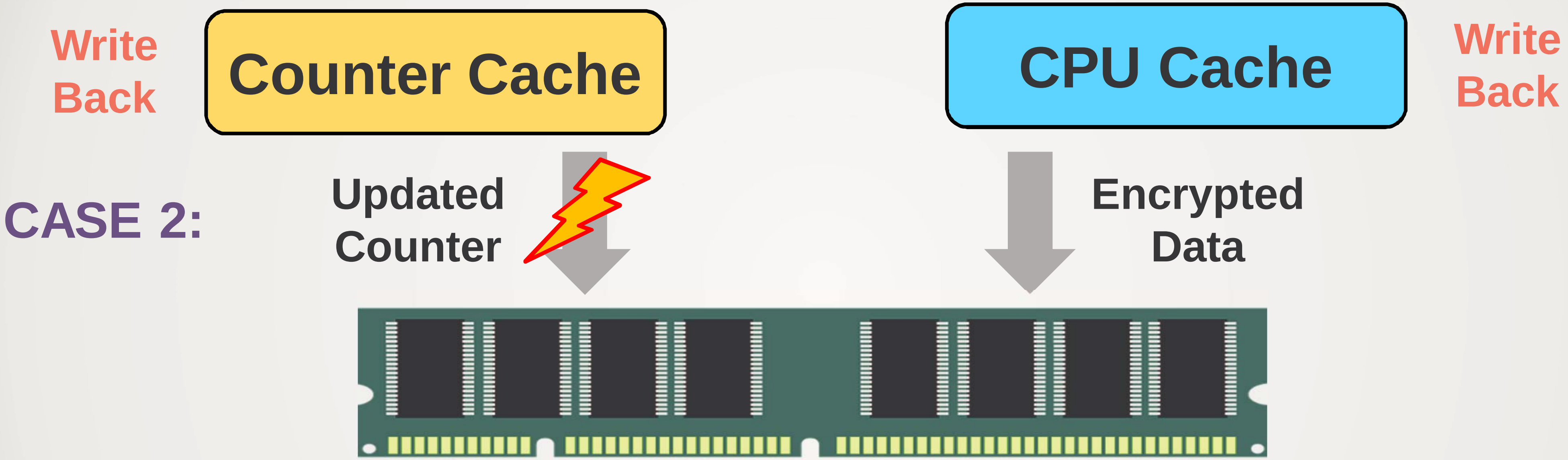
CASE 1:

Updated Counter

Encrypted Data

- ➢ Data and counter cannot reach NVM at the same time

Crash Inconsistency Caused by Encryption



Write Back

Counter Cache

CPU Cache

Write Back

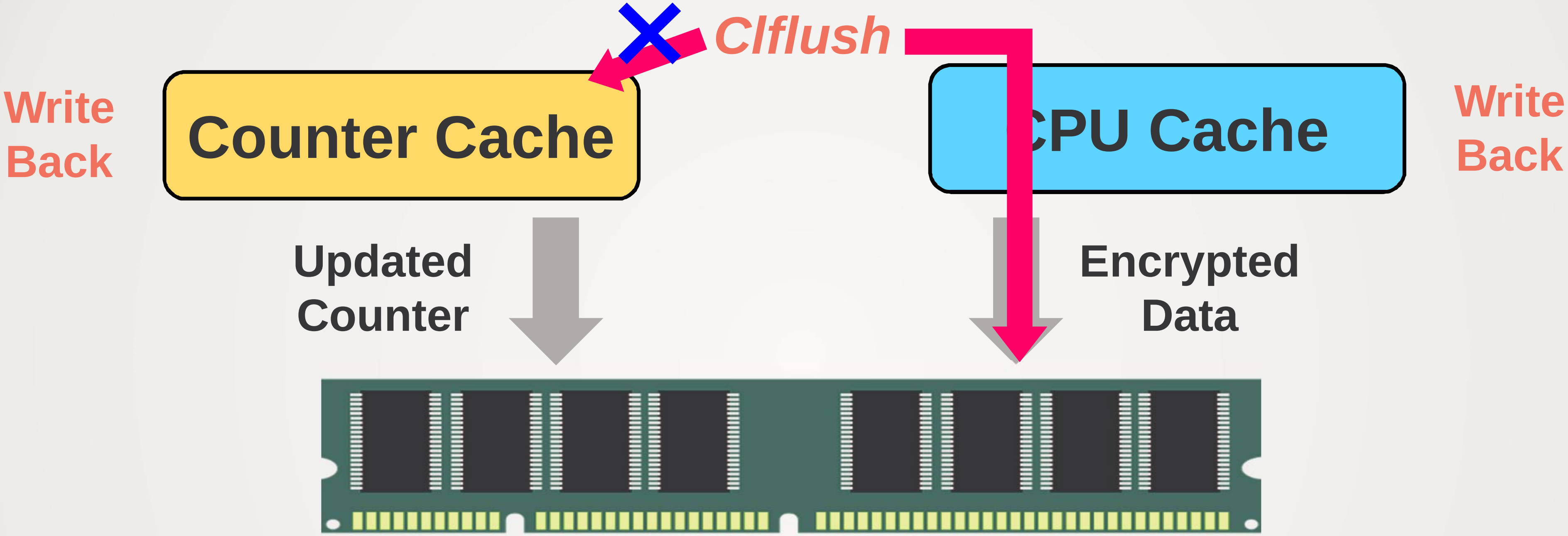
CASE 2:

Updated Counter

Encrypted Data

- ➢ Data and counter cannot reach NVM at the same time

Crash Inconsistency Caused by Encryption

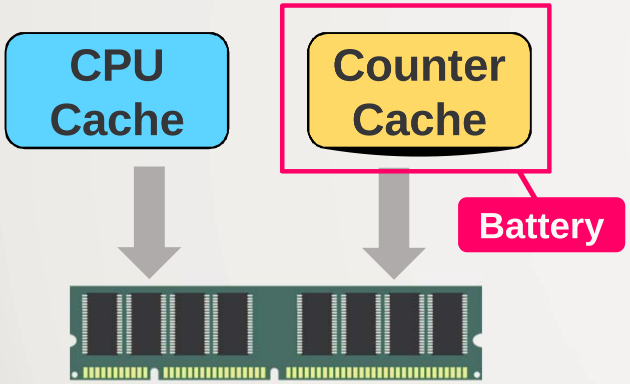


- Data and counter cannot reach NVM at the same time
- *Clflush* and *mfence* cannot operate the counter cache

Existing Solutions (Write-back Counter Cache)

Large Battery Backup

[Awad et al., ASPLOS’16]
[Zuo et al., MICRO’18]



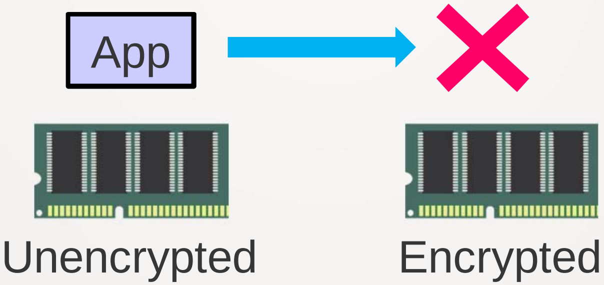
Expensive

Software-level Modification

[Liu et al., HPCA’18]

New programming primitives

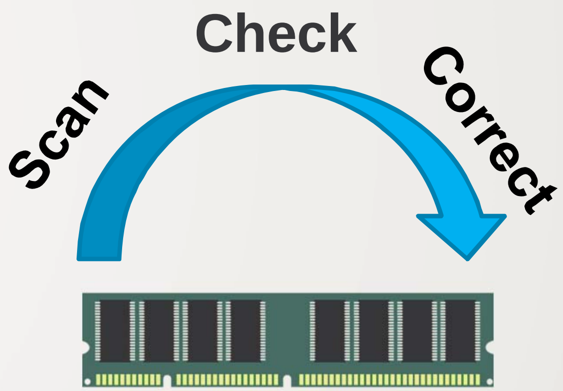
- *counter_cache_writeback()*
- *CounterAtomic*



Portability limitation

Error Correction

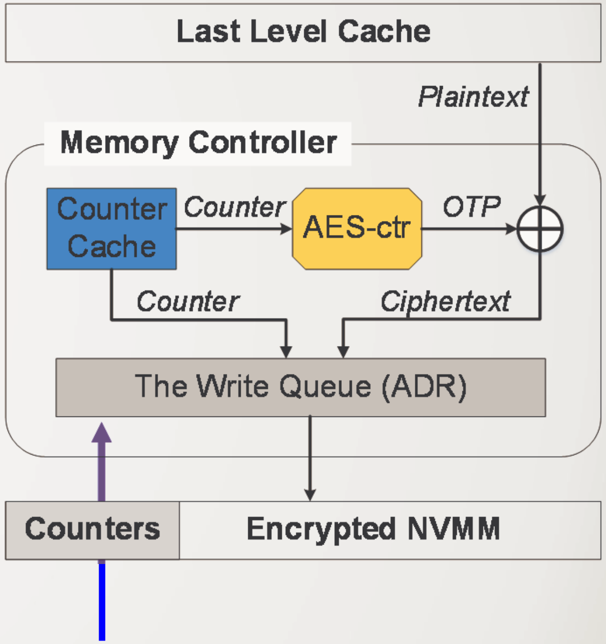
[Ye et al., MICRO’18]



Long recovery time

SuperMem: Secure and Persistent Memory

- **Exploit a write-through counter cache**
  - No large battery backup
  - No software-level modifications
  - No need to correct counters
  - Double writes
- **A counter write coalescing scheme**
  - Reduce the number of write requests
- **A cross-bank counter storage scheme**
  - Speedup memory writes



Asynchronous DRAM refresh (ADR): cache lines reaching the write queue can be considered durable.

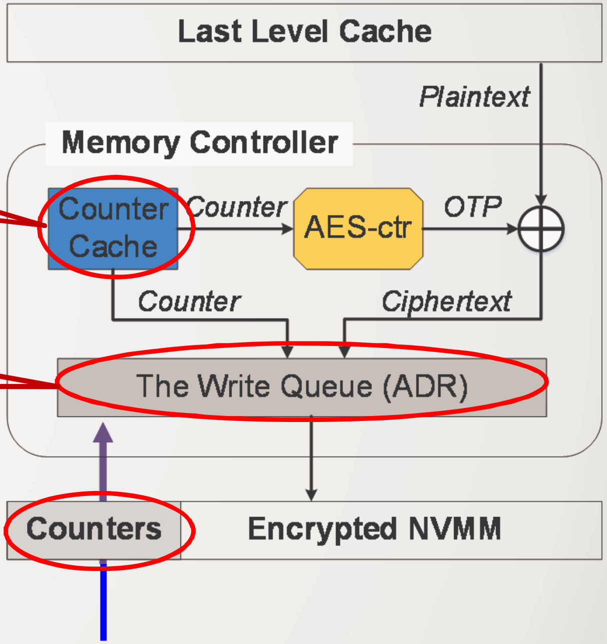
SuperMem: Secure and Persistent Memory

Application-transparent

Write-through counter cache (*Guarantee consistency*)

Counter write coalescing (*Reduce writes*)

Cross-bank counter storage (*Speedup writes*)



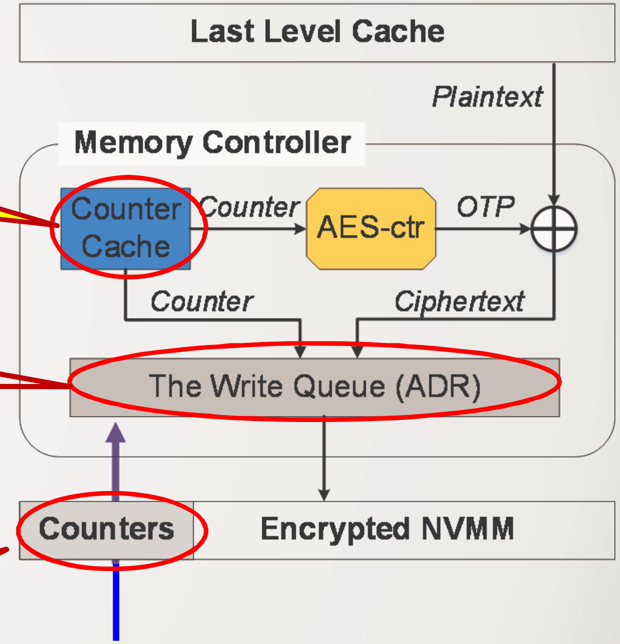
Asynchronous DRAM refresh (ADR): cache lines reaching the write queue can be considered durable.

SuperMem: Secure and Persistent Memory

Write-through counter cache
(*Guarantee consistency*)

Counter write coalescing
(*Reduce writes*)

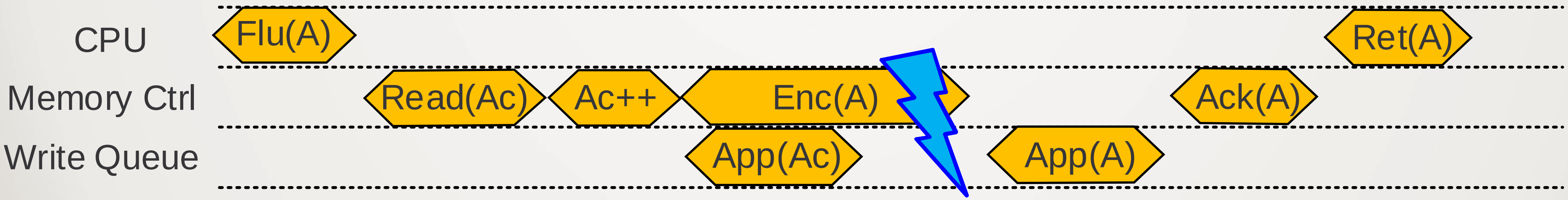
Cross-bank counter storage
(*Speedup writes*)



Asynchronous DRAM refresh (ADR): cache lines reaching the write queue can be considered durable.

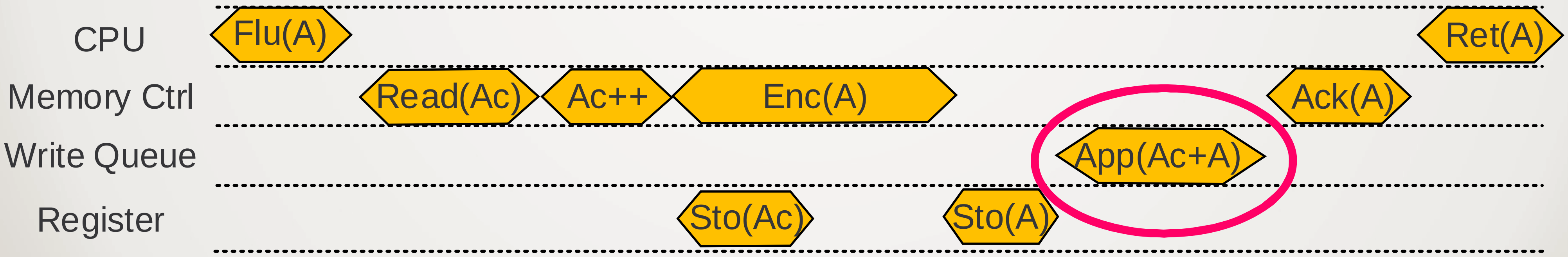
Write-through Counter Cache

- Ensure that data and its counter reach the write queue in the same time
  - Write through counter cache



Write-through Counter Cache

- Ensure that data and its counter reach the write queue in the same time
  - Write through counter cache
  - Add a register

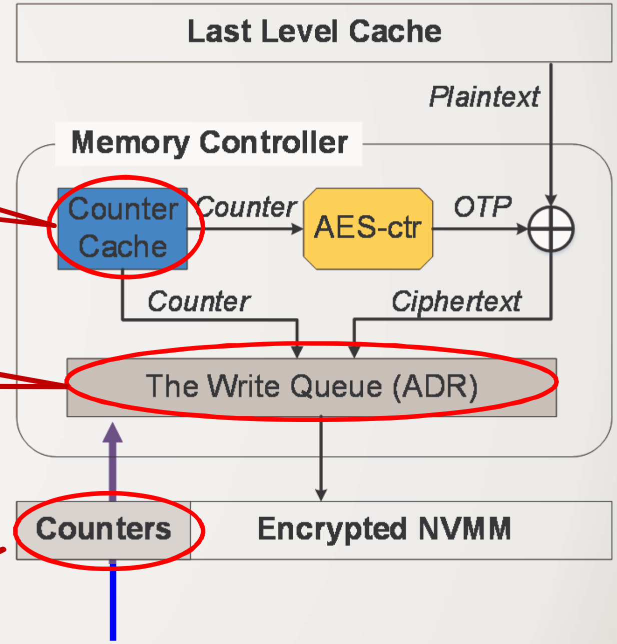


SuperMem: Secure and Persistent Memory

Write-through counter cache
(*Guarantee consistency*)

Counter write coalescing
(*Reduce writes*)

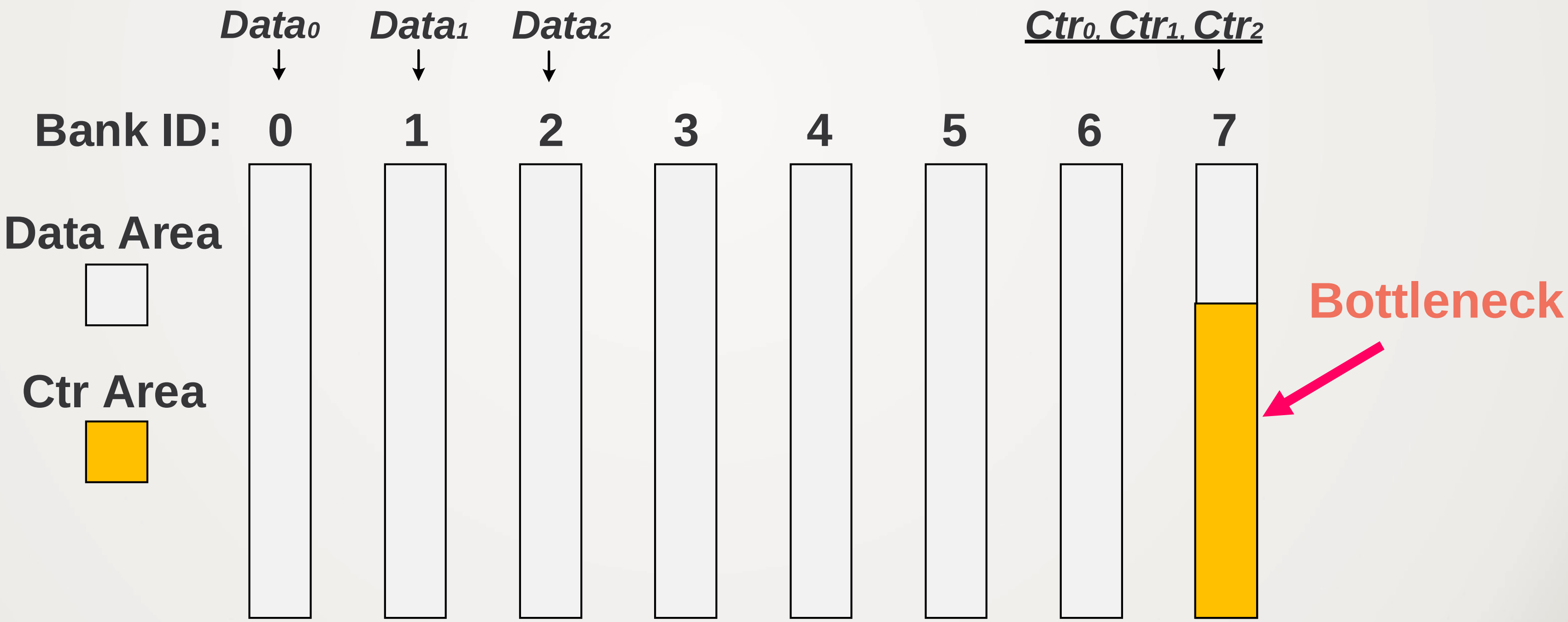
Cross-bank counter storage
(*Speedup writes*)



Asynchronous DRAM refresh (ADR): cache lines reaching the write queue can be considered durable.

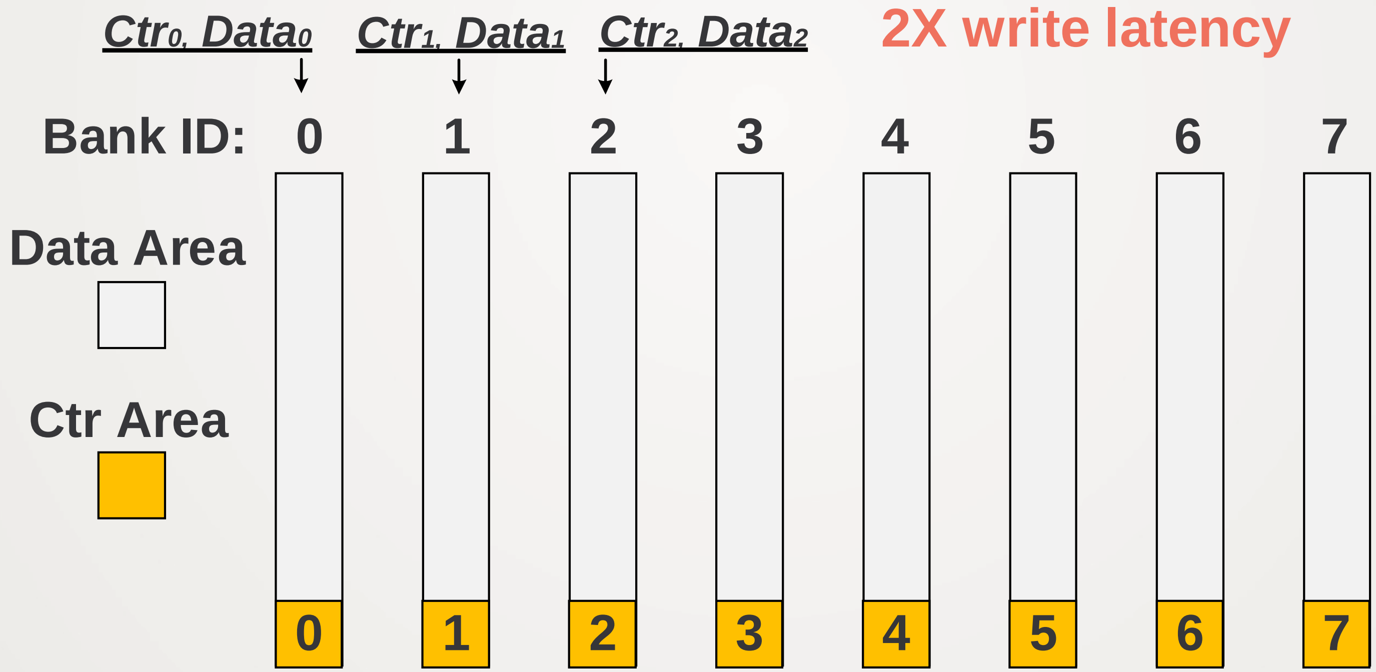
Cross-bank Counter Storage

- SingleBank: Counters are stored in a continuous area in NVM [ASPLOS'15, ASPLOS'16, HPCA'18]



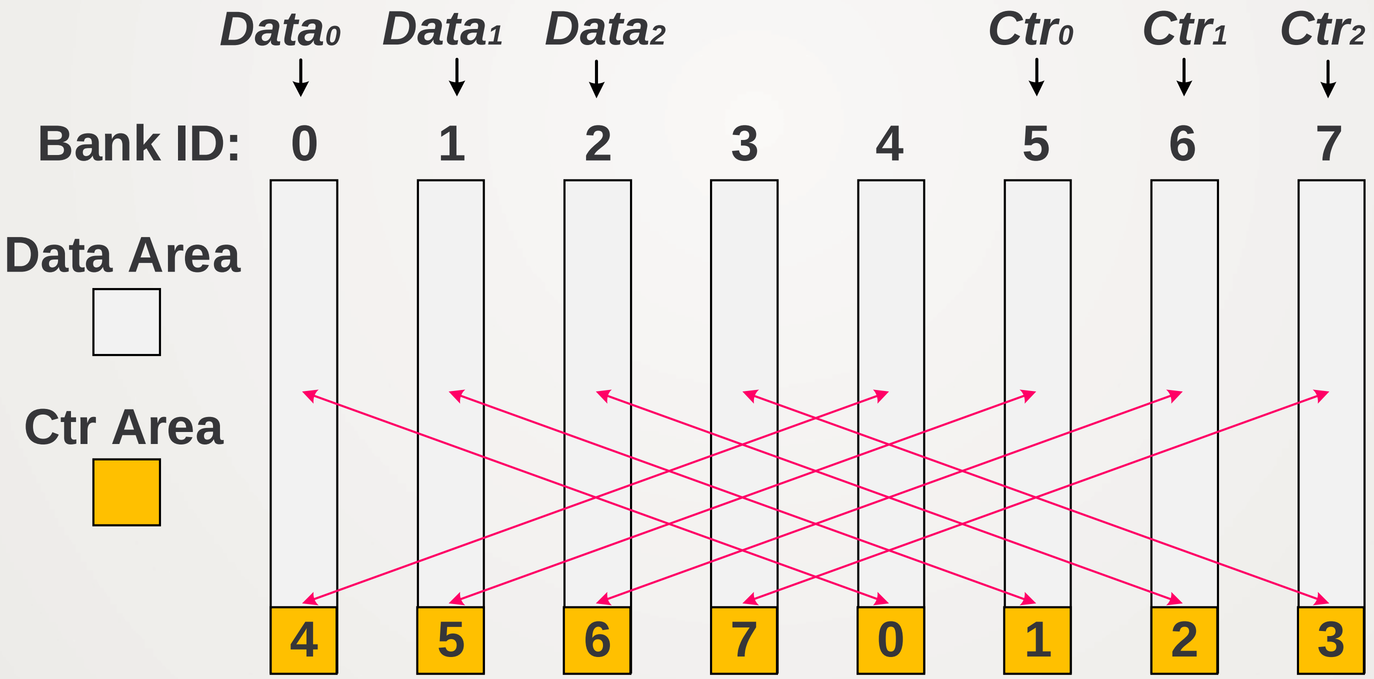
Cross-bank Counter Storage

- SameBank: Stores the counters of data into their local banks



Cross-bank Counter Storage

- XBank: Stores each data and its counter into different banks to leverage bank parallelism

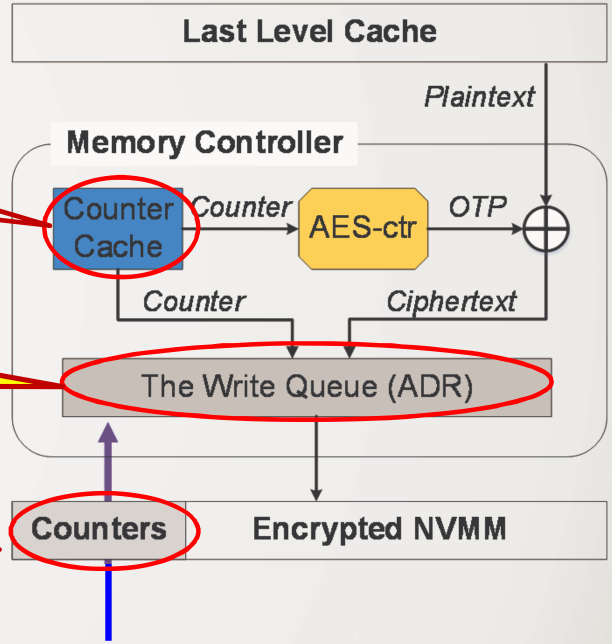


SuperMem: Secure and Persistent Memory

Write-through counter cache
(*Guarantee consistency*)

Counter write coalescing
(*Reduce writes*)

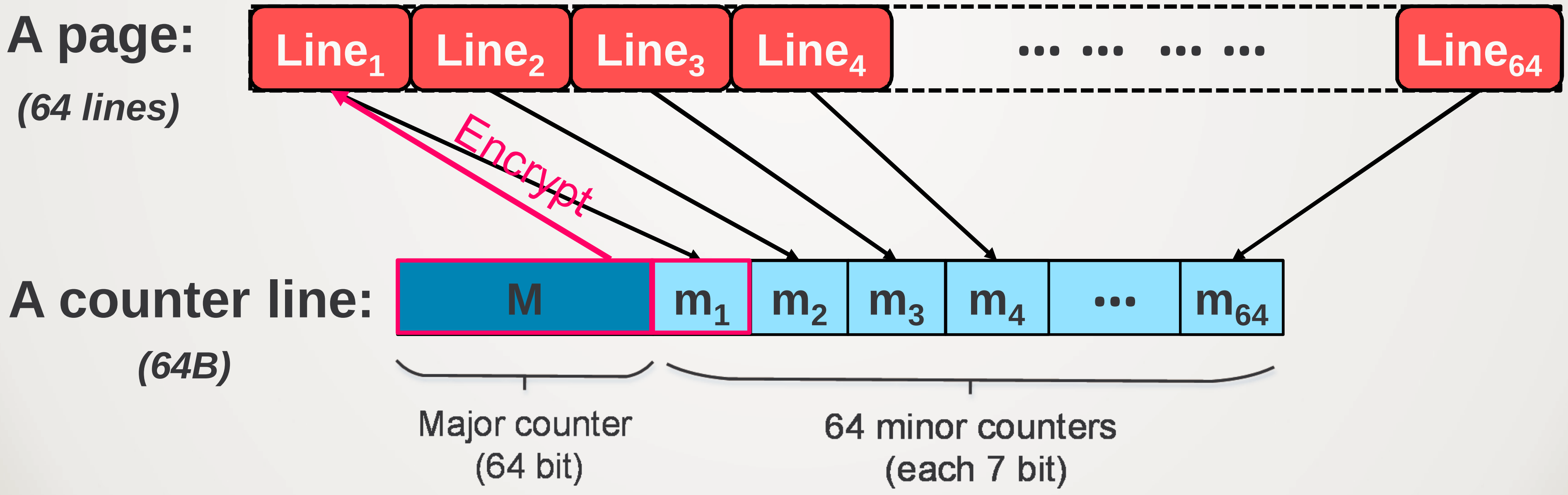
Cross-bank counter storage
(*Speedup writes*)



Asynchronous DRAM refresh (ADR): cache lines reaching the write queue can be considered durable.

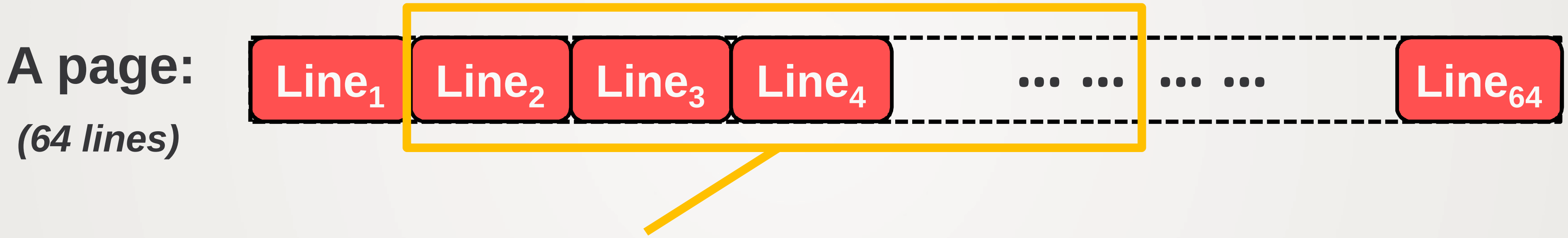
Locality-aware Counter Write Coalescing

- Spatial locality of counter storage
  - – All counters of a page are stored in a counter line



Locality-aware Counter Write Coalescing

- Spatial locality of counter storage
  - – All counters of a page are stored in a counter line



A log entry or the transaction data

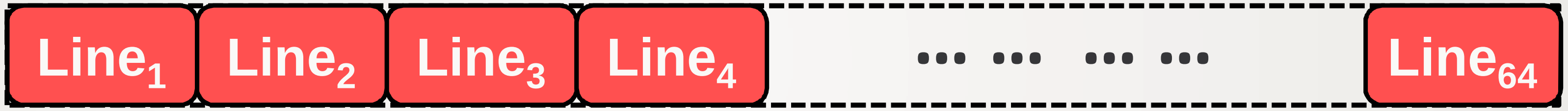
- Spatial locality of log and data writes

Locality-aware Counter Write Coalescing

- An example of writing 4 lines within a page

A page:

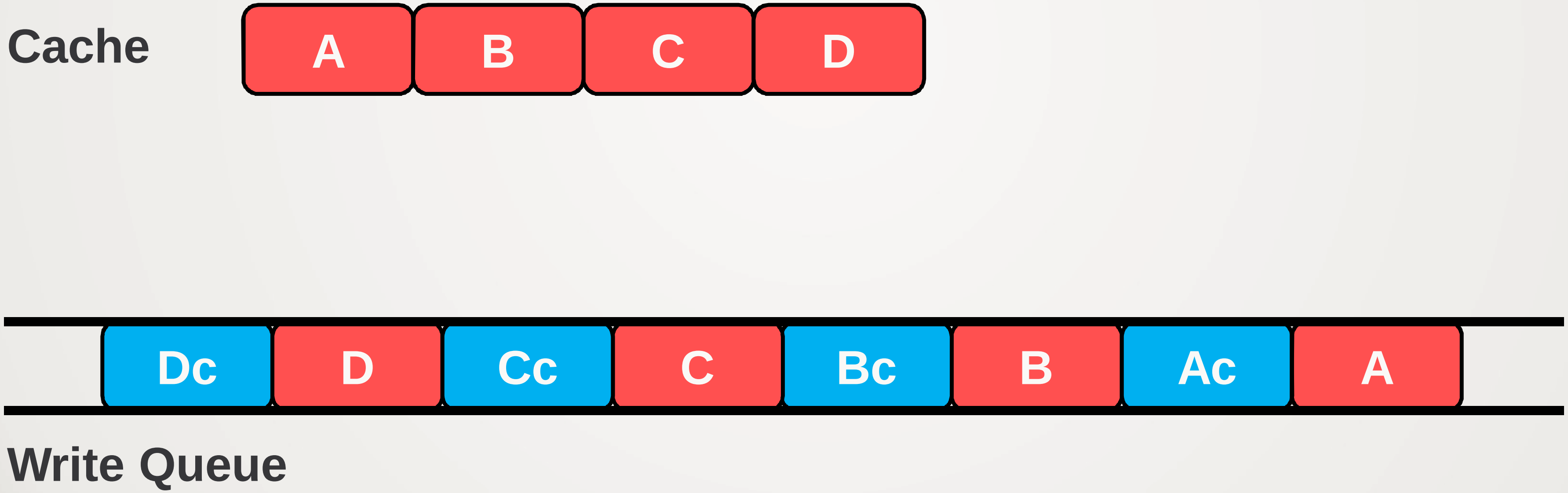
(64 lines)



Line$_1$ Line$_2$ Line$_3$ Line$_4$ … … … … Line$_{64}$

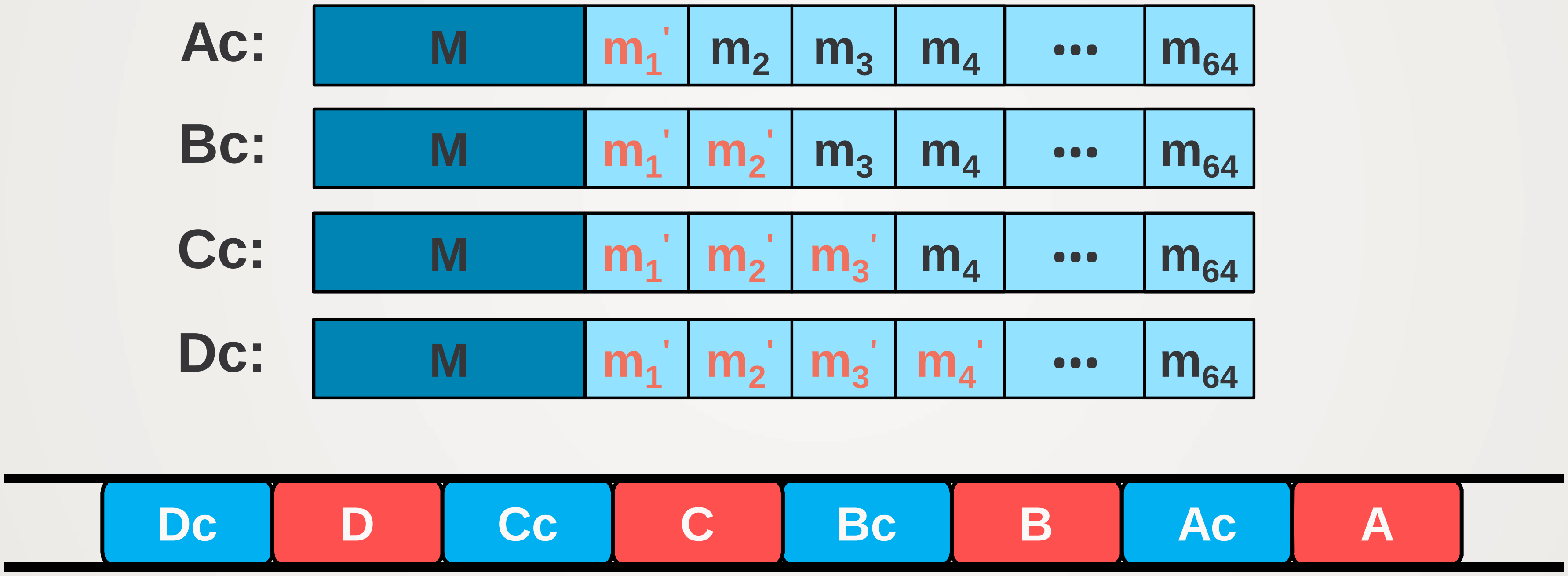
Locality-aware Counter Write Coalescing

- An example of writing 4 lines within a page



Locality-aware Counter Write Coalescing

- An example of writing 4 lines within a page



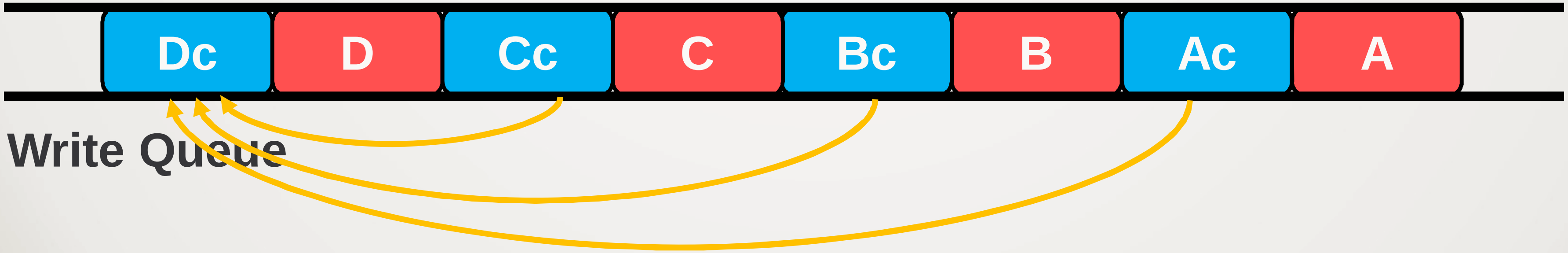
Ac:	M	m_1'	m_2	m_3	m_4	···	m_{64}
Bc:	M	m_1'	m_2'	m_3	m_4	···	m_{64}
Cc:	M	m_1'	m_2'	m_3'	m_4	···	m_{64}
Dc:	M	m_1'	m_2'	m_3'	m_4'	···	m_{64}

Write Queue

Locality-aware Counter Write Coalescing

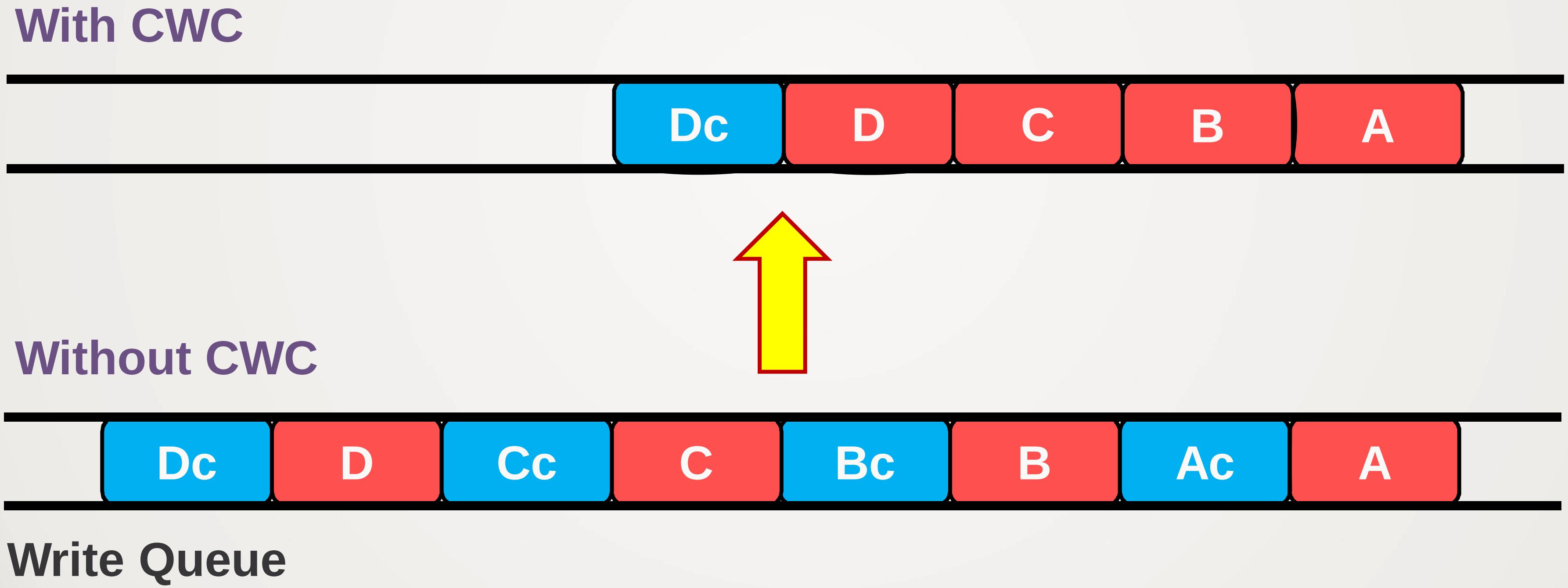
- Coalescing counter writes in the write queue

Ac:	M	m_1'	m_2	m_3	m_4	•••	m_{64}
Bc:	M	m_1'	m_2'	m_3	m_4	•••	m_{64}
Cc:	M	m_1'	m_2'	m_3'	m_4	•••	m_{64}
Dc:	M	m_1'	m_2'	m_3'	m_4'	•••	m_{64}



Locality-aware Counter Write Coalescing (CWC)

- Coalescing counter writes in the write queue



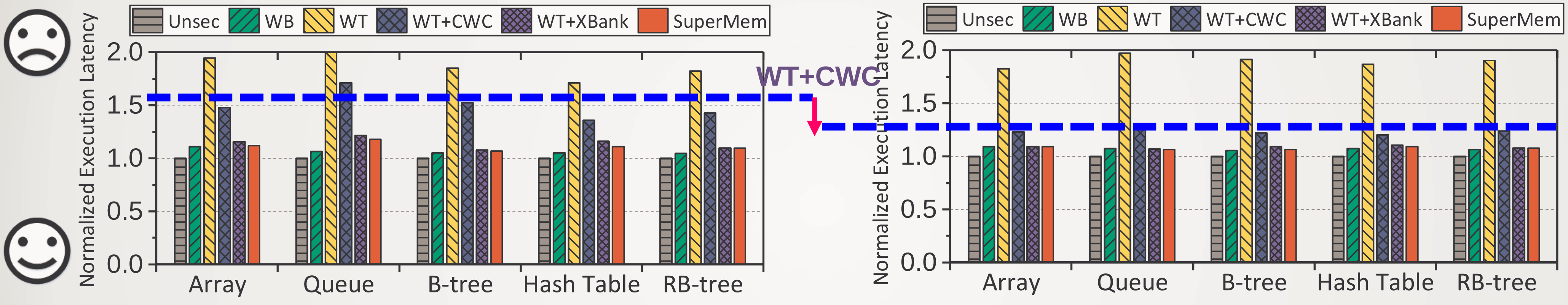
Performance Evaluation

- Model NVM using gem5 and NVMain

Comparisons
Unsec: An un-encrypted NVM
WB: An ideal write-back scheme
WT: A write-through scheme
WT+CWC: A write-through scheme with CWC
WT+Xbank: A write-through scheme with XBank
SuperMem

Benchmarks
Array: Randomly swapping entries
Queue: Randomly enqueueing and dequeueing
B-tree: Inserting random KVs
Hash Table: Inserting random KVs
RB-tree: Inserting random KVs

Transaction Execution Latency – Single-core

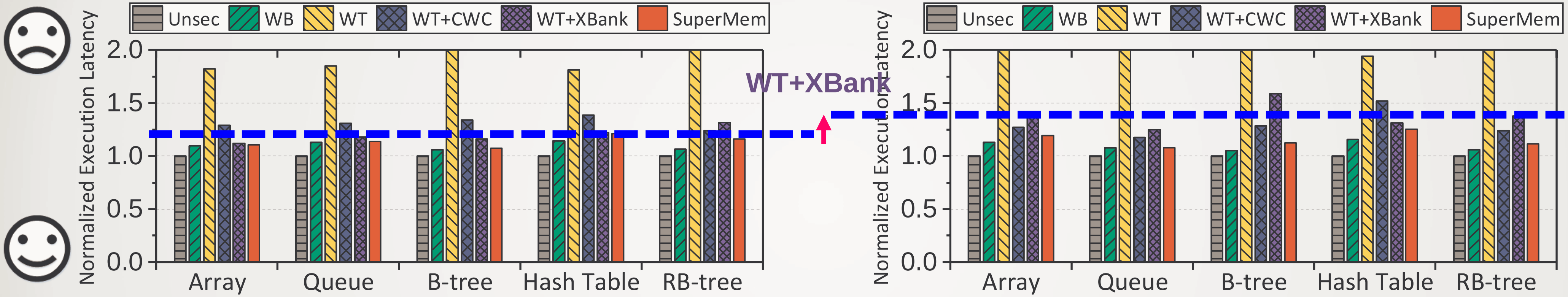


Transaction size: 256B

Transaction size: 4KB

- ➢ SuperMem achieves the performance comparable to a secure NVM with an ideal write-back cache (WB)

Transaction Execution Latency – Multi-core

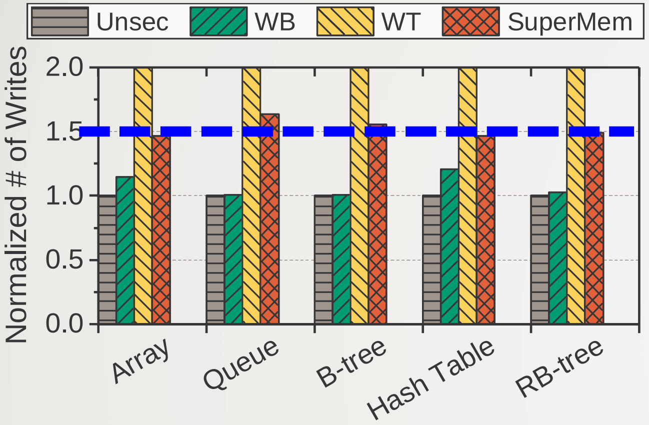


2 programs

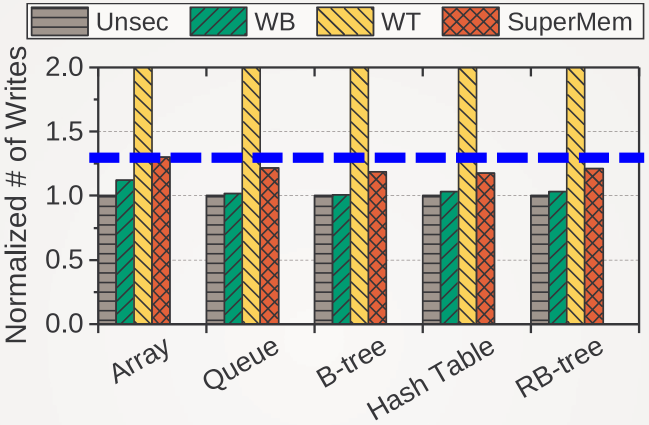
8 programs

- ➢ SuperMem achieves the performance comparable to a secure NVM with an ideal write-back cache (WB)

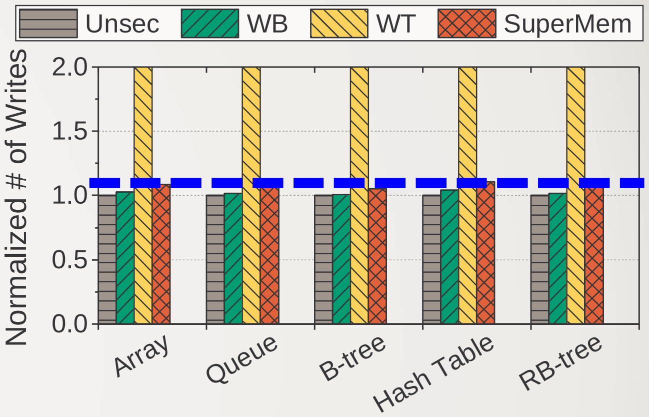
The Number of Write Requests



Transaction size: 256B



Transaction size: 1KB



Transaction size: 4KB

- SuperMem reduces up to 50% of write requests by using the CWC scheme

Conclusion

Problem

- Memory encryption incurs crash inconsistency issue

Existing Work

- Using a write-back counter cache
- Large battery backup, software-level modification, or error correction

Our Solution

- SuperMem: exploit a write-through counter cache
    - ~~Large battery backup~~, ~~software-level modification~~, ~~error correction~~
    - Counter write coalescing for reducing writes
    - Cross-bank counter storage for speeding up writes

Thanks! Q&A