HPCL State Key Laboratory of High Performance Computing

ExtraCC: Improving Performance of Secure NVM with Extra Counters and ECC

Zhengguo Chen[1], Youtao Zhang[2], Nong Xiao[3]

1: National University of Defense Technology

2: University of Pittsburgh

3: Sun Yat-sen University

Email: zgchen.nudt@foxmail.com

Outline

- Background
- Motivation
- Design
- Evaluation

Information security on NVM

- Information security on Non-volatile Memory (NVM)
  - Confidentiality attack: bus snooping and memory scanning attacks
  - Integrity attack: splicing, spoofing and replay attacks
- Confidentiality protection
  - Counter mode encryption
- Integrity protection
  - Message authentication code（MAC）
  - Merkle tree

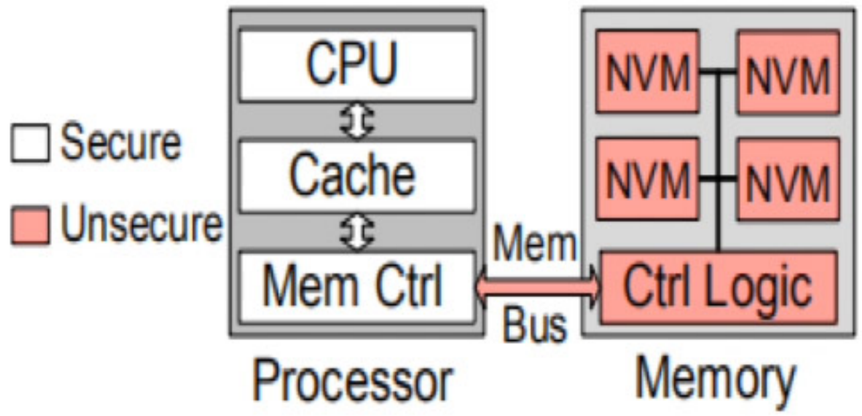
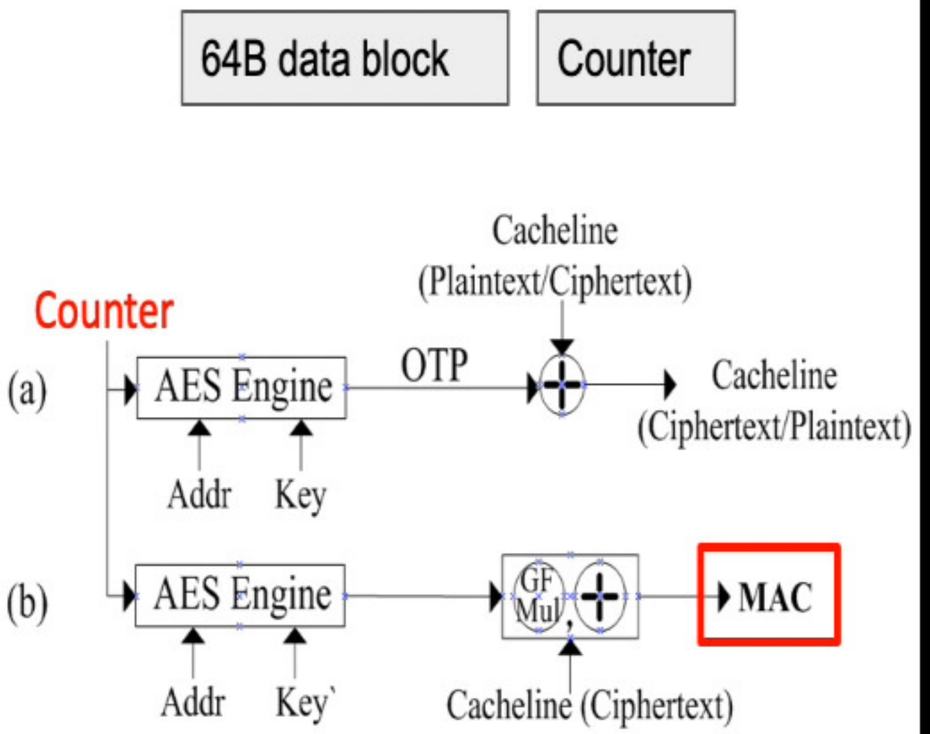
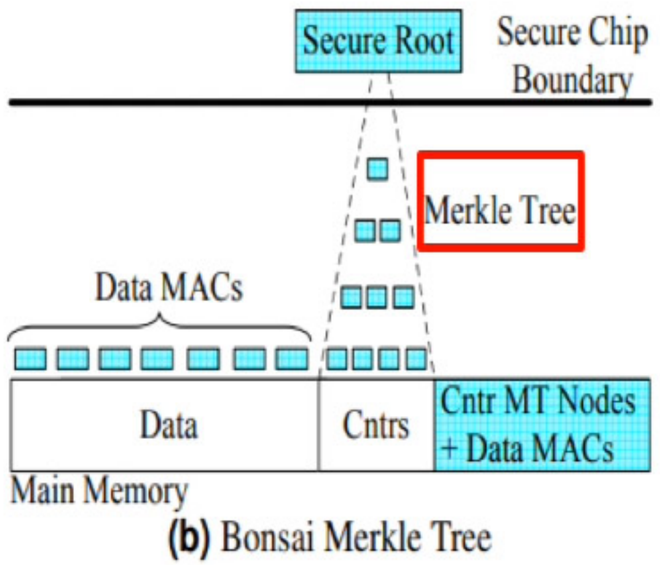


Fig. 1. Threat Model.

Confidentiality and integrity protection

- Counter mode encryption
  - Counter increments for data update
    - Same OTP incur info leakage
  - Counter overflow
    - Change key
    - Decrypt and re-encrypt user data
- Integrity protection
  - Message authentication code（MAC）
  - Merkle tree
    - Defend replay attack



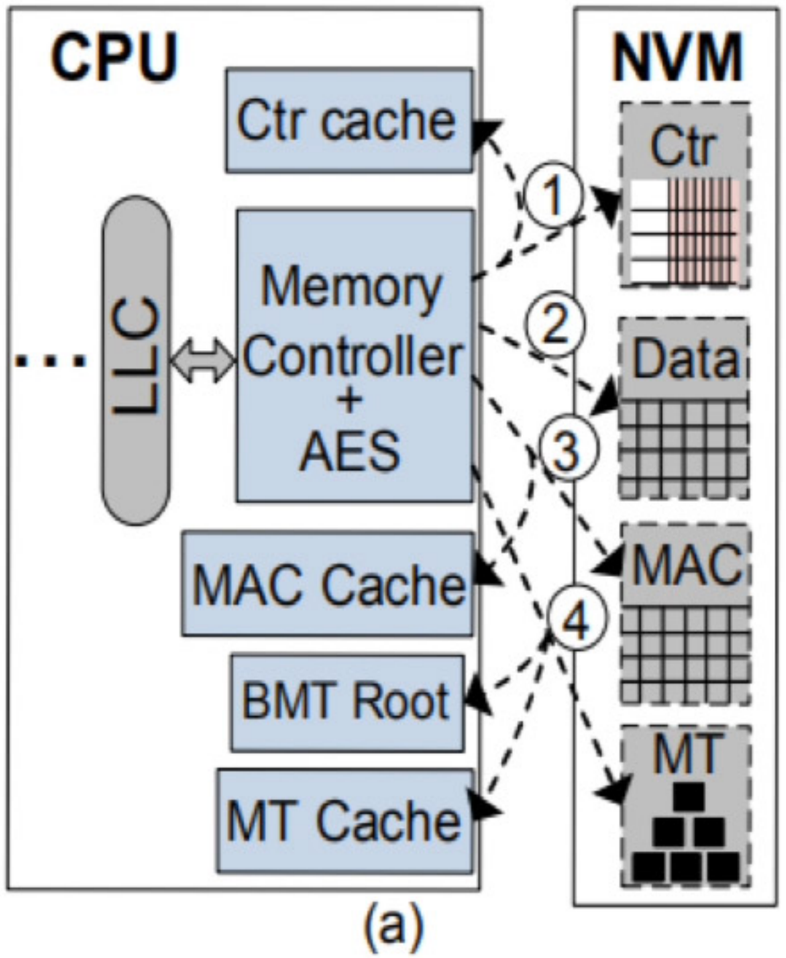


(b) Bonsai Merkle Tree

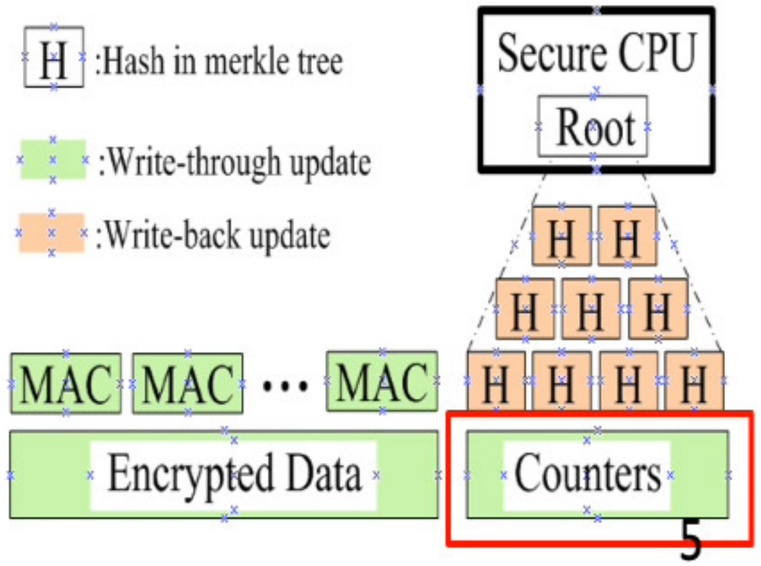
Secure NVM system

- Secure NVM system
  - Instant data recovery after system crash
  - Exposed to data confidentially and integrity attacks
- Overhead
  - Persistent update data and metadata
  - Counter：write-through updata
  - MAC：write-through update
  - Incur many NVM writes
  - Reduce lifetime and performance

Focus on solving counter update overhead!



(a)

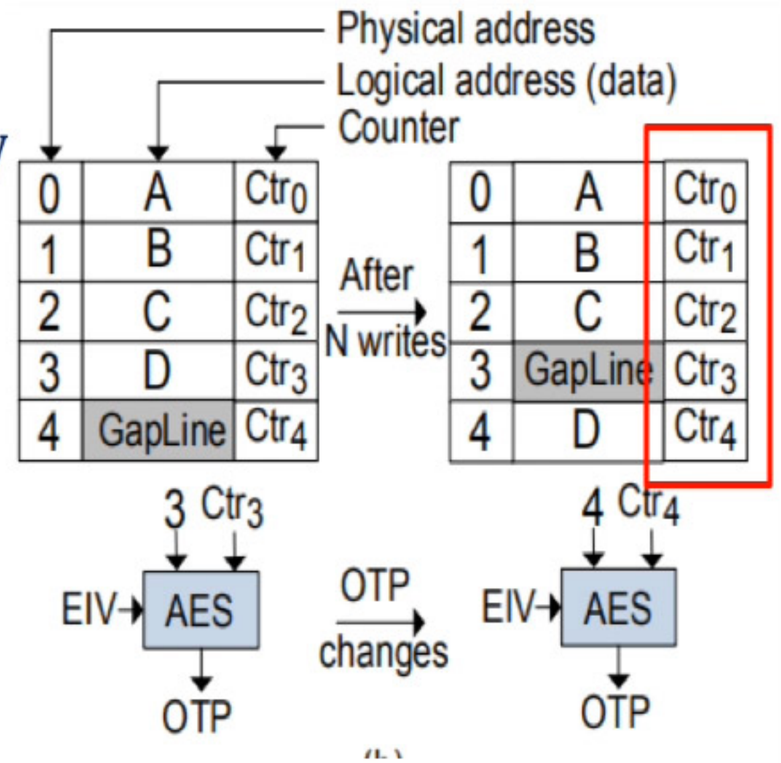


Outline

- Background
- Motivation
- Design
- Evaluation

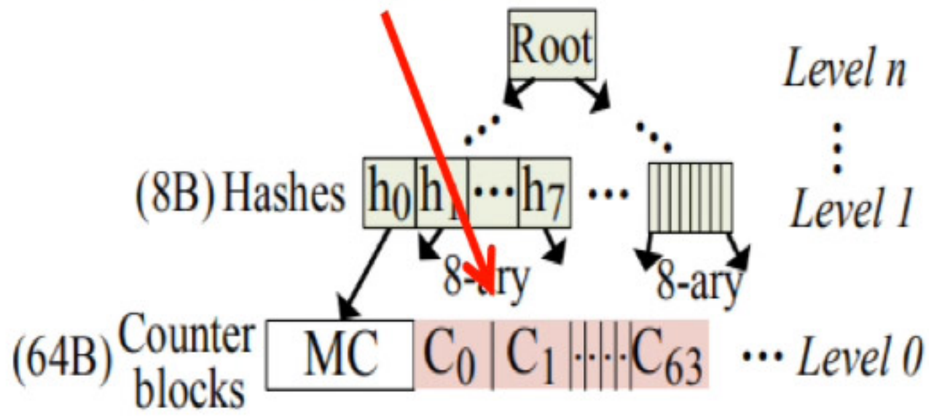
Motivation

- Counter associates with logic address
  - Ctr of hot data tends to overflow frequently
- ACME[DAC'18]
  - Associate ctr with physical address
    - Achieve ctr wear leveling
  - Ctr is stored with user data
    - Update data and ctr within one NVM write
    - Reduce Ctr update overhead



- Weakness：break Ctr locality
  - Integrity verification need

Read multiple Ctr



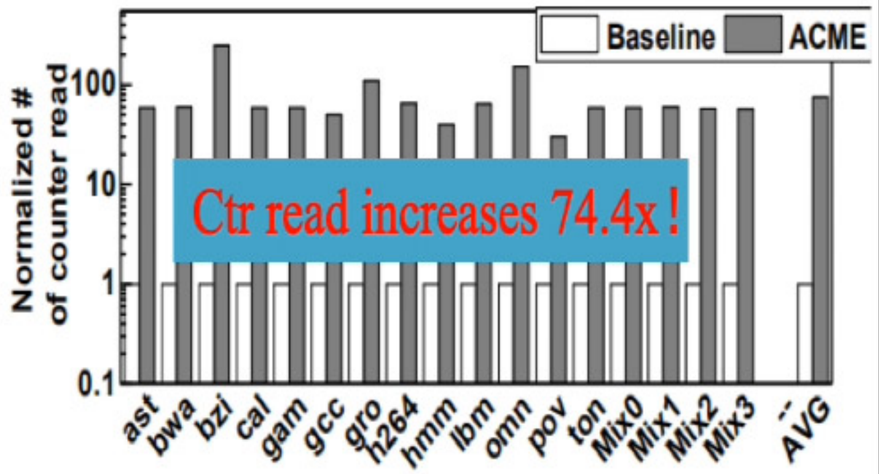


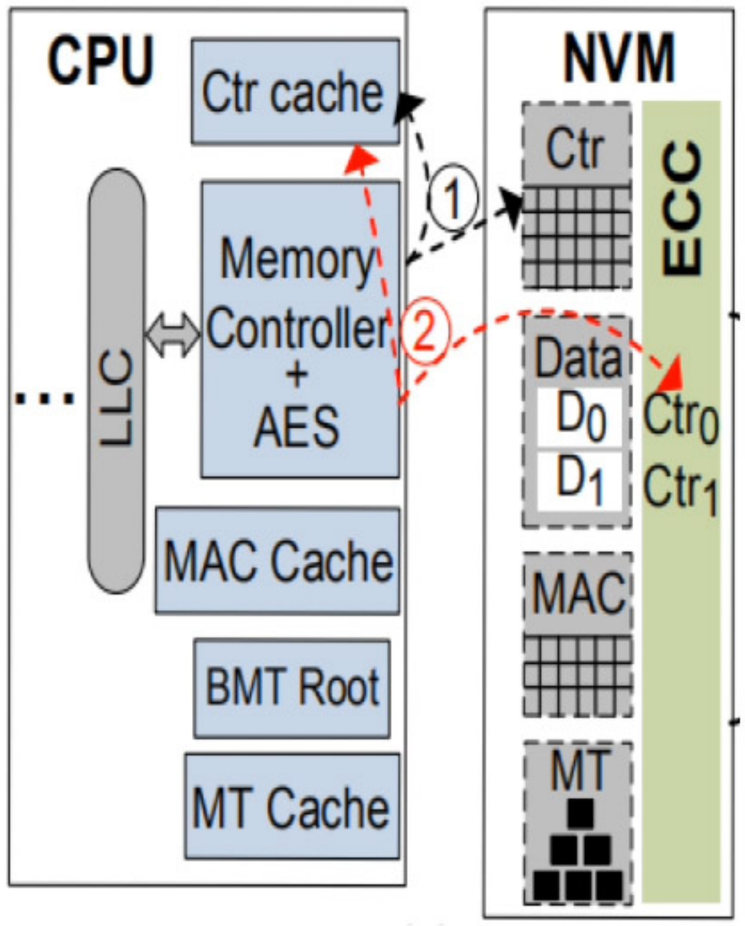
Fig. 5. Normalized number of counter read from NVM.

Outline

- Background
- Motivation
- Design
- Evaluation

Our design

- Propose ExtraCC
  - A permanent Ctr
    - Used for encryption/decryption (Step 1)
    - Keep Ctr locality
    - Reduce integrity check overhead
    - Improve memory read operation
  - A working Ctr
    - Record updated Ctr (Step 2)
    - Utilize ECC bus
    - Updated with user data in one NVM write
    - Improve memory write operation



Permanent Ctr utilize write-back update,
working Ctr is written with user data.
Thus, reduce counter update overhead!

Our design: ExtraCC

- Two-tiered ECC
  - ECC of user data -> T1EC (detect errors) + T2EC (detect and correct errors)
  - Make room for working counter
  - T2EC is written with MAC

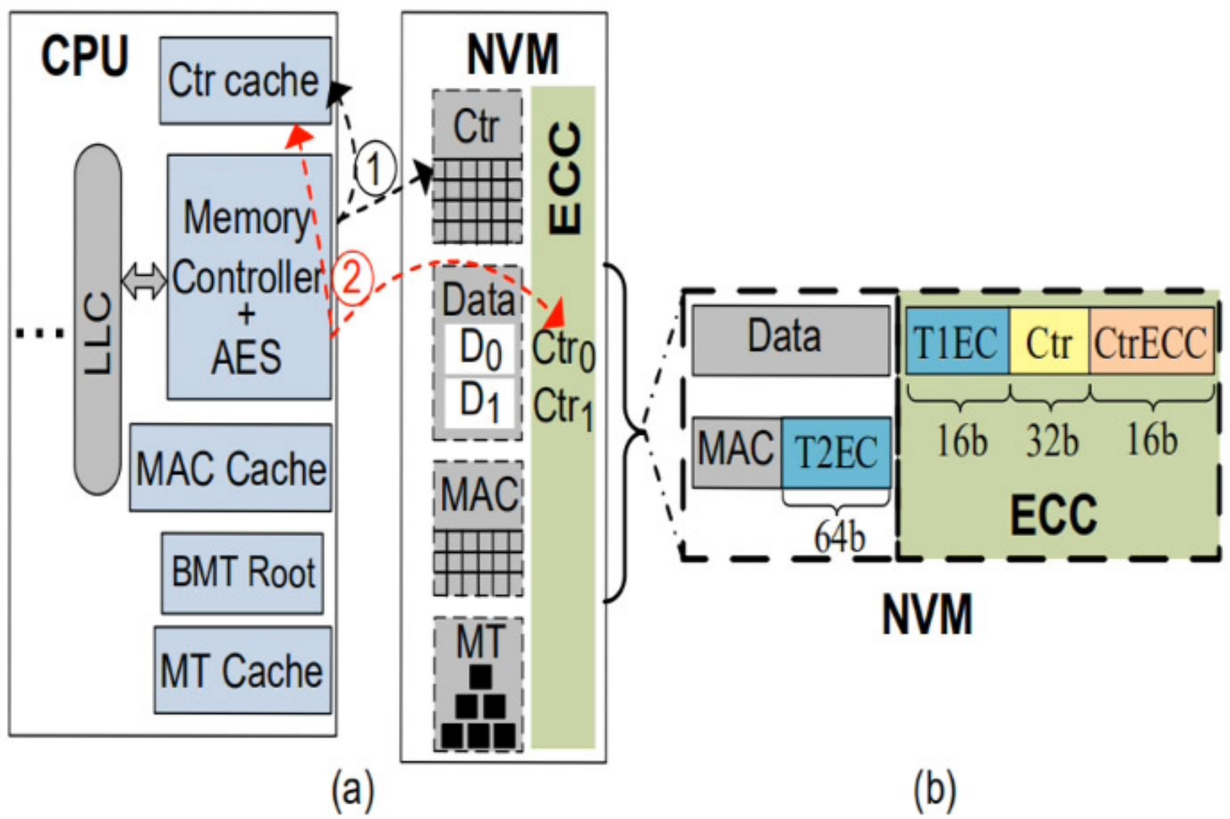


Fig. 6. Overview of ExtraCC (a) and two-tiered ECC (b).

Our design: ExtraCC

- LAPA counter scheme
  - logical-addressed-physical-associated counter scheme
  - Associate counters with physical address of user data
  - Use logical address to locate permanent counters

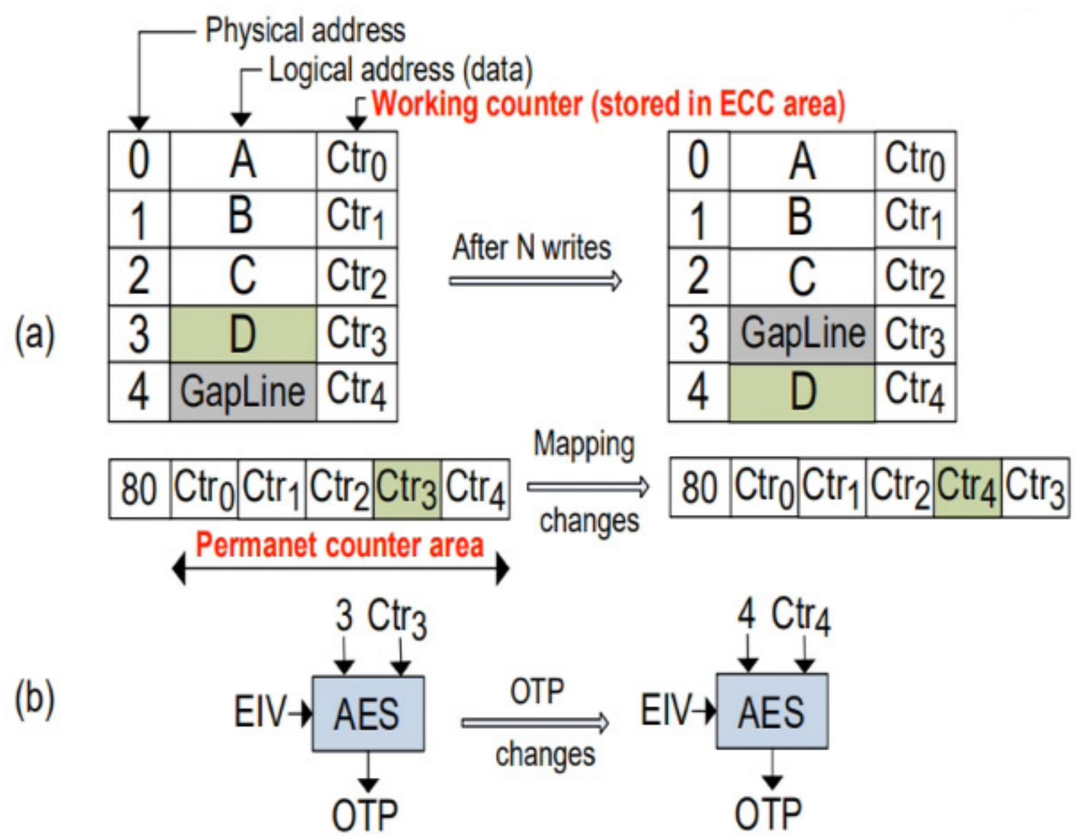


Fig. 7. LAPA counter scheme during wear leveling.

Outline

- Background
- Motivation
- Design
- Evaluation

Evaluations

- Methodology
  - PIN tool to collect ins
  - In-house simulator
    - 2 level caches
    - PCM
- Workloads
  - SPEC CPU2006
- Comparison
  - Baseline
  - ACME[DAC'18]
  - Osiris[MICRO'18]

TABLE I
CONFIGURATIONS [2], [3], [21] AND MIXED WORKLOADS

CPU	4 cores single issue in-order CMP, 4GHz
L1 I/D-cache	Private, 32KB, 2-way, 64B block, 2 cycles
L2 cache	Shared, 2MB, 4-way, 64B block, 10 cycles
Counter Cache	Shared, 128KB, 4-way, 64B block, 2 cycles
MAC Cache	Shared, 32KB, 4-way, 64B block, 2 cycles
MT Cache	Shared, 16KB, 4-way, 64B block, 2 cycles
PCM memory	8GB, 1 channel, 2 ranks, 8 banks/rank, 8-entry write queue/bank, Latency: Read: 100ns, Set: 200ns , Reset: 100ns, energy: Read: 1.49 nJ, Set:6.76 nJ, Reset: 6.73 nJ
Workload	Benchmarks
Mix0	astar, bwaves, bzip2, calculix
Mix1	bwaves, calculix, gcc, gamess
Mix2	lbm, h264, tonto, gromacs
Mix3	hmmer, lbm, omnetpp, povray

Results

Results

- Performance
  - Improve performance by 16.1%
  - Reduce read response time by 49.9%
- Lifetime
  - Reduce write traffic by 20.5%

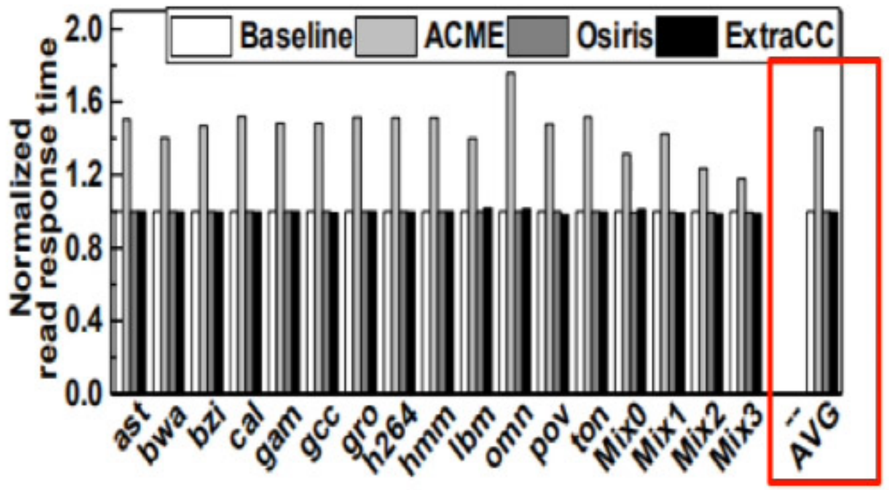


Fig. 10. Normalized read response time in various schemes.

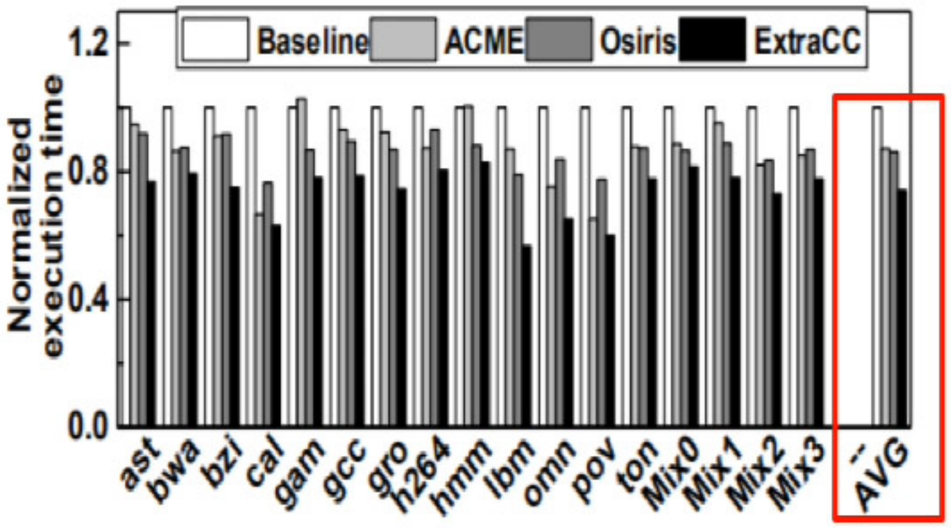


Fig. 9. Normalized execution time in various schemes.

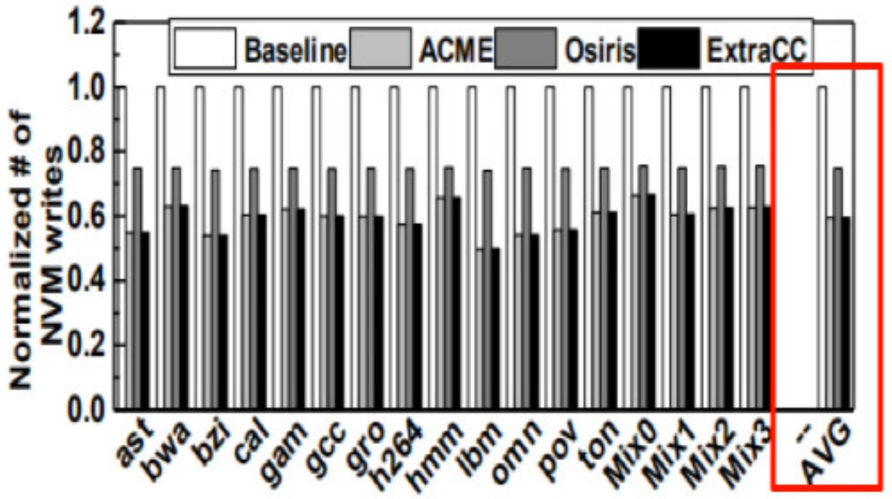


Fig. 11. Normalized number of NVM writes in various schemes.

HPCL

Thanks!

Q&A