Flexible Cache Error Protection using an ECC FIFO

Doe Hyun Yoon and Mattan Erez

Dept. Electrical and Computer Engineering
The University of Texas at Austin

ECC FIFO

- Goal: **to reduce on-chip ECC overhead**
  - Two-tiered error protection
    - T1EC: light-weight on-chip error code
    - T2EC: strong error correcting code
  - **Off-load T2EC overhead to FIFO in DRAM**

- Why FIFO? It's **simple to manage**

- **15-25**% LLC area reduction
- **10-17**% LLC power saving
- Just **1**% performance penalty

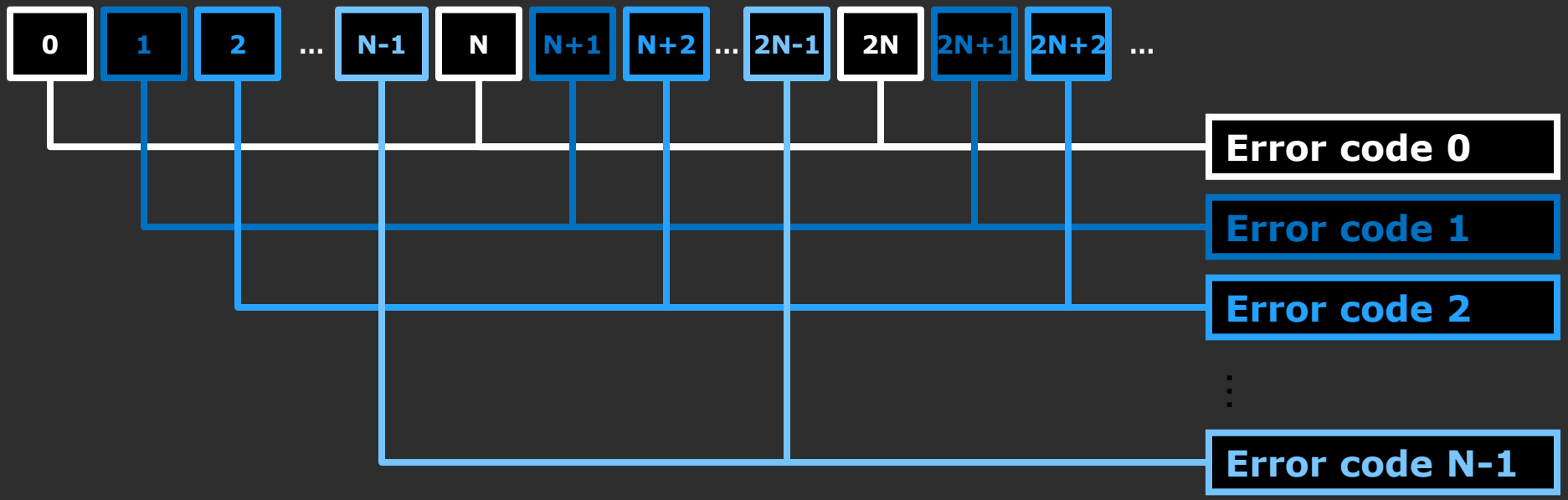
BACKGROUND

Error Correcting Codes

- 1-bit parity for error detection
- SEC-DED (Hamming) codes
  - Single-bit Error Correction and Double-bit Error Detection
  - 8bit ECC for 64bit data
- DEC-TED
  - Double-bit Error Correction and Triple-bit Error Detection
  - 15bit ECC for 64bit data

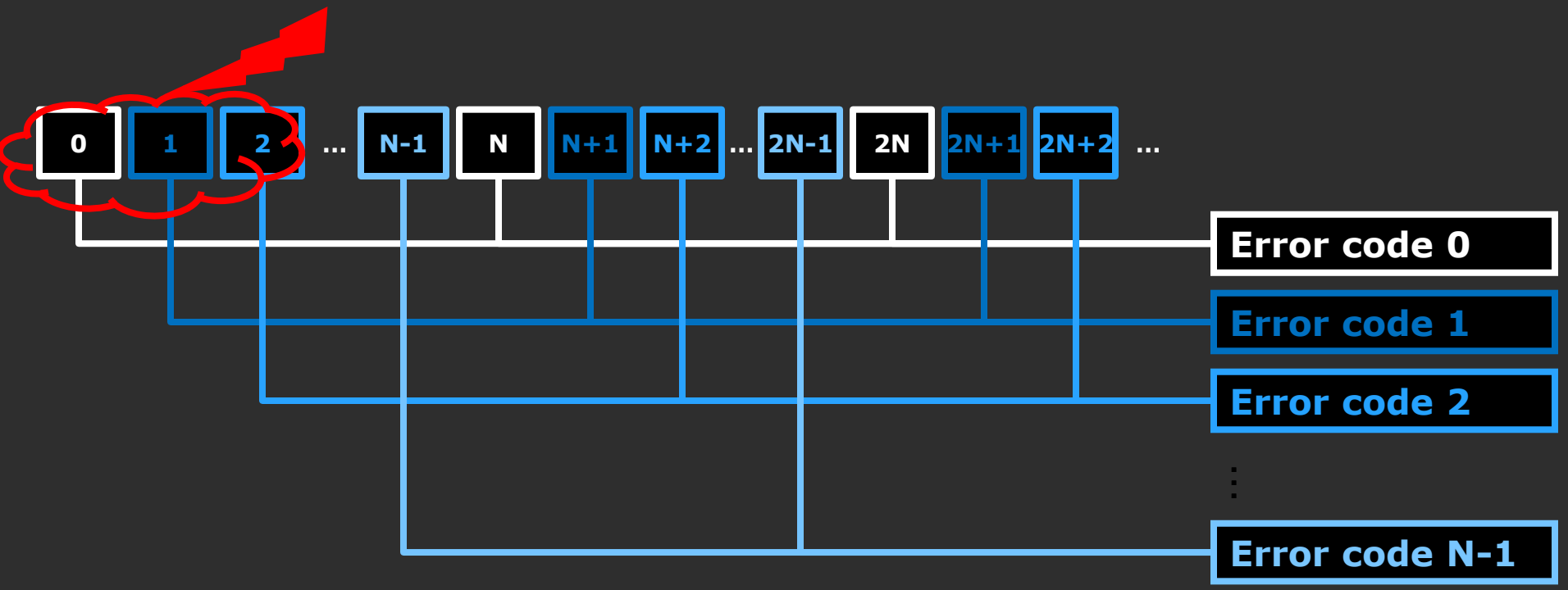
Interleaving

- To detect and correct burst errors
  - N-way interleaving converts an N-bit burst error to N single-bit errors



Interleaving

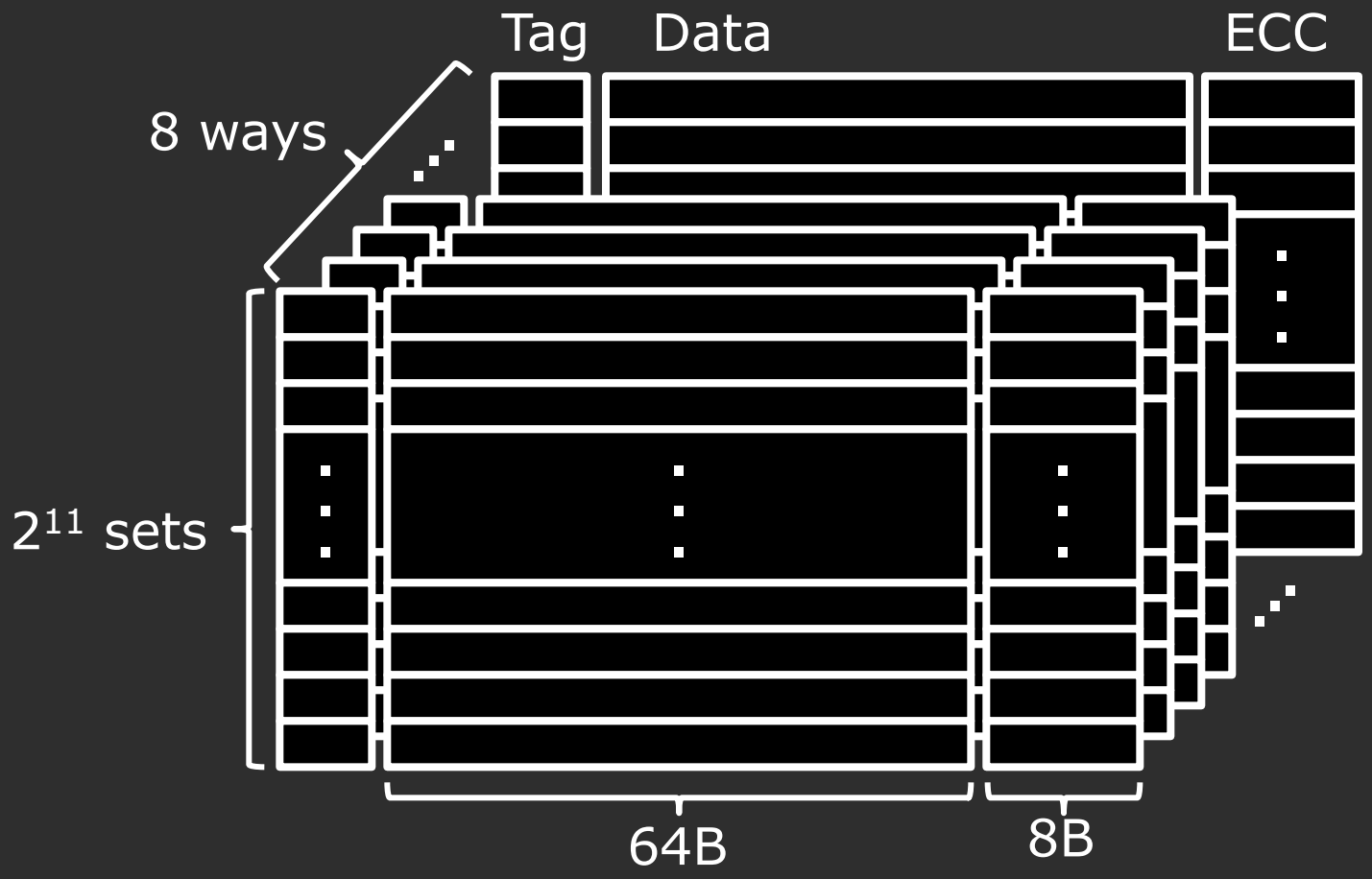
- To detect and correct burst errors
  - N-way interleaving converts an N-bit burst error to N single-bit errors



Interleaving

- To detect and correct burst errors
  - N-way interleaving converts an N-bit burst error to N single-bit errors
- Baseline cache error protection
  - 8 way interleaved SEC-DED
    - Can correct up to 8-bit burst errors
    - 8B ECC per 64B cache line

Uniform Error Protection



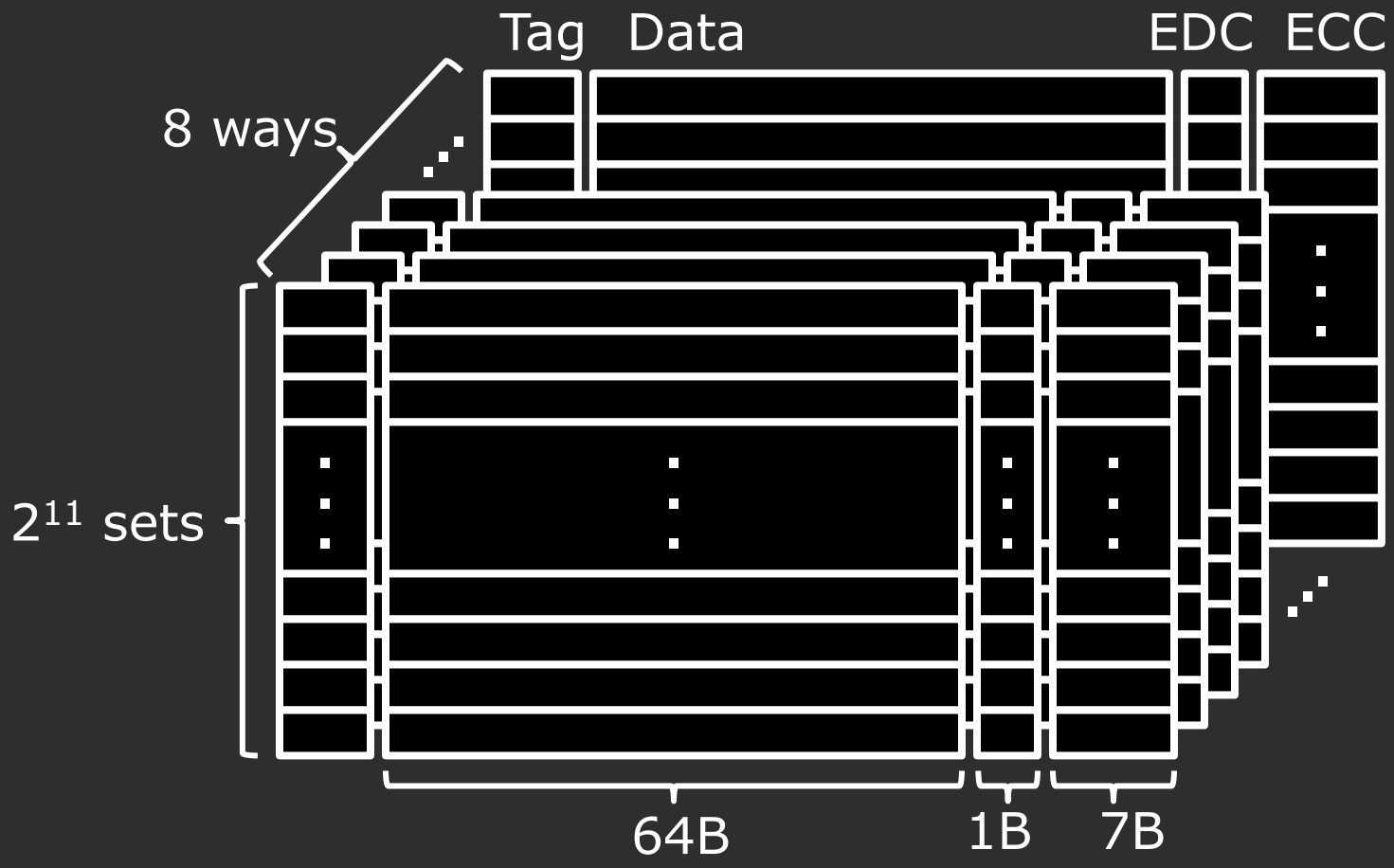
ECC increases area AND leakage/dynamic power

RELATED WORK

Soft Errors: Observations

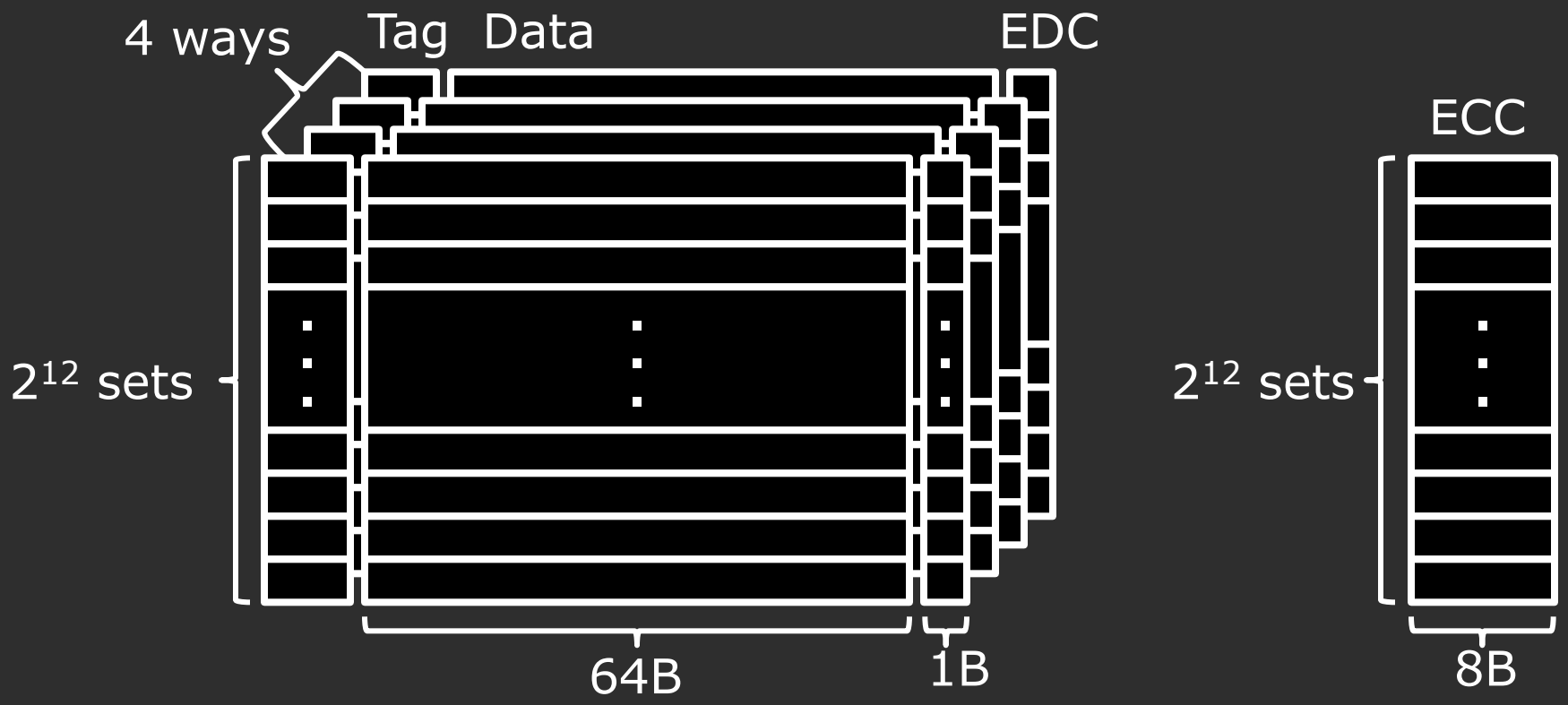
- Still, Soft Error Rate (SER) is low
  - Every cache access tries to detect errors, but finds no error in most cases
- Error Detection – Common case
  - Need a low cost, low overhead error detection mechanism
- Error Correction – Uncommon case
  - Correction can be slow
  - But, still need to maintain error correction info somewhere
- Memory hierarchy provides redundancy inherently for clean data
  - Only dirty lines need error correcting codes

PERC [Sorin'06] / Energy Efficient [Li'04]



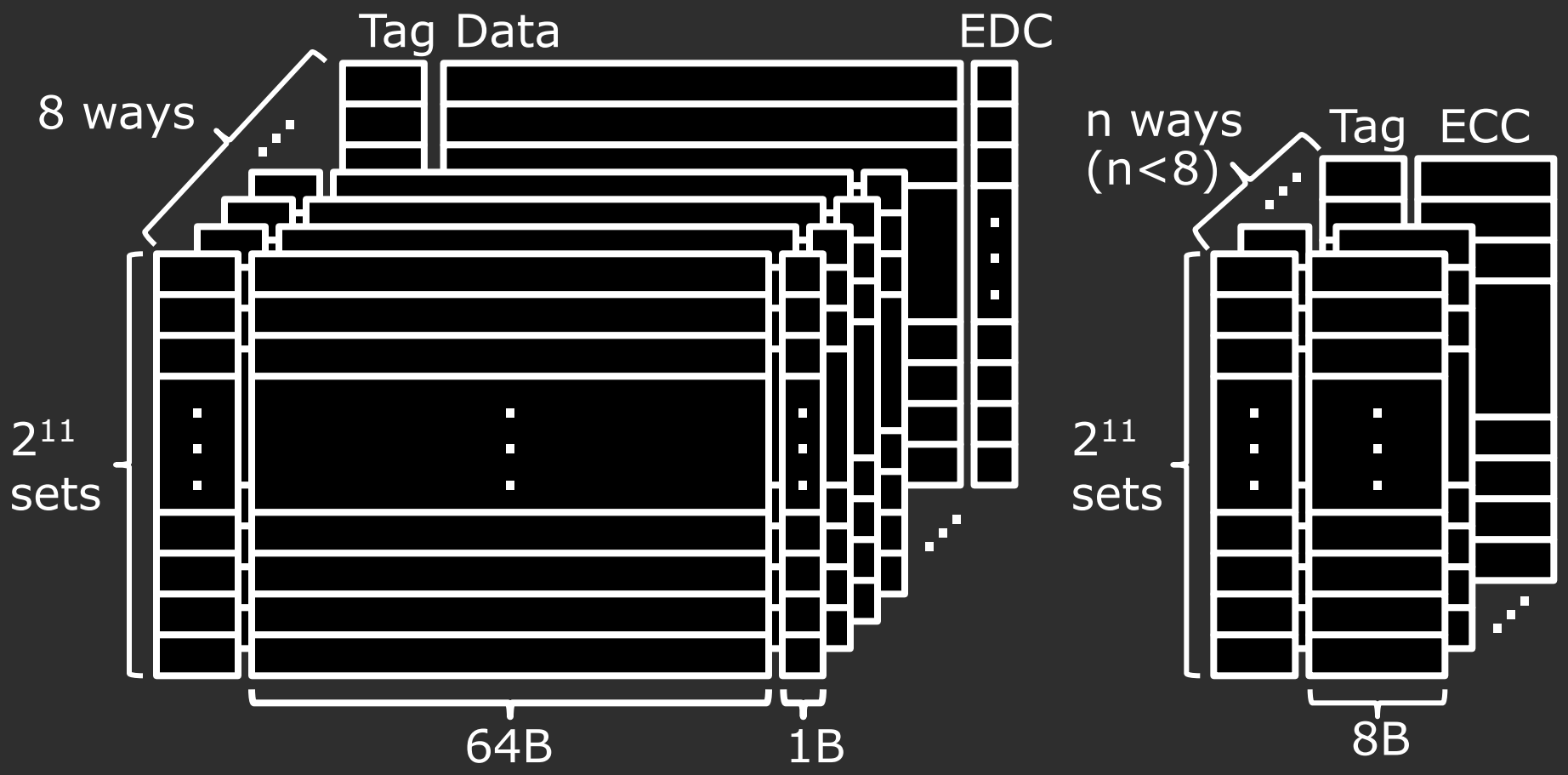
Read only Data and EDC – saves dynamic power
Power gate ECC of clean lines – saves static power

Area Efficient [Kim'06]



Allow only 1 dirty line per set

MAXn scheme



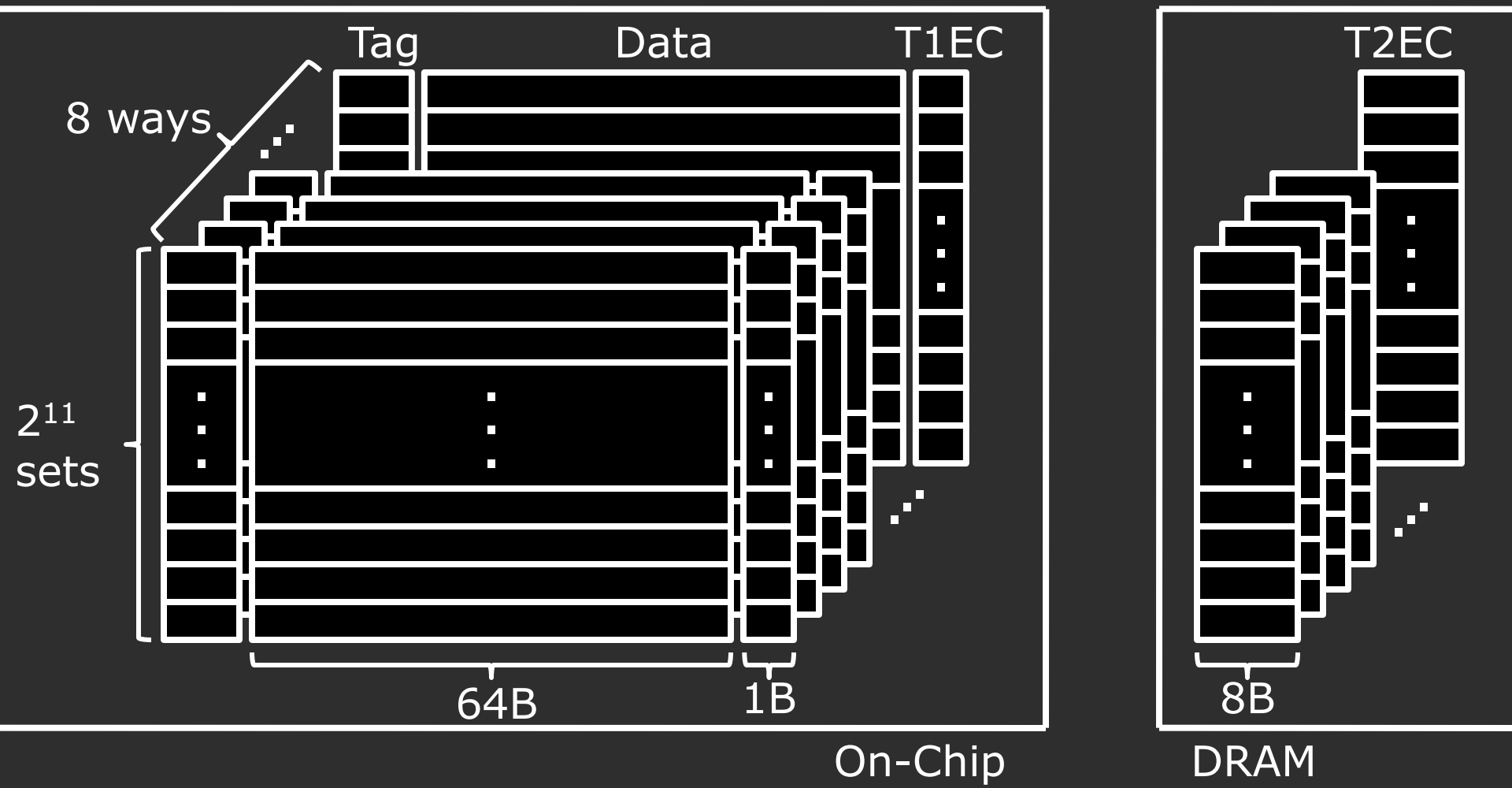
Allow only n dirty lines per set
May cause detrimental cleaning traffic

Two-tiered error protection

- Tier-1 Error Code (T1EC)
  - On-chip light-weight error code
  - Uniform error protection

- Tier-2 Error Code (T2EC)
  - Strong error codes only for dirty lines
  - Corrects
    Detected but Uncorrected Errors (DUE)
    of T1EC

Memory Mapped ECC [Yoon'09]

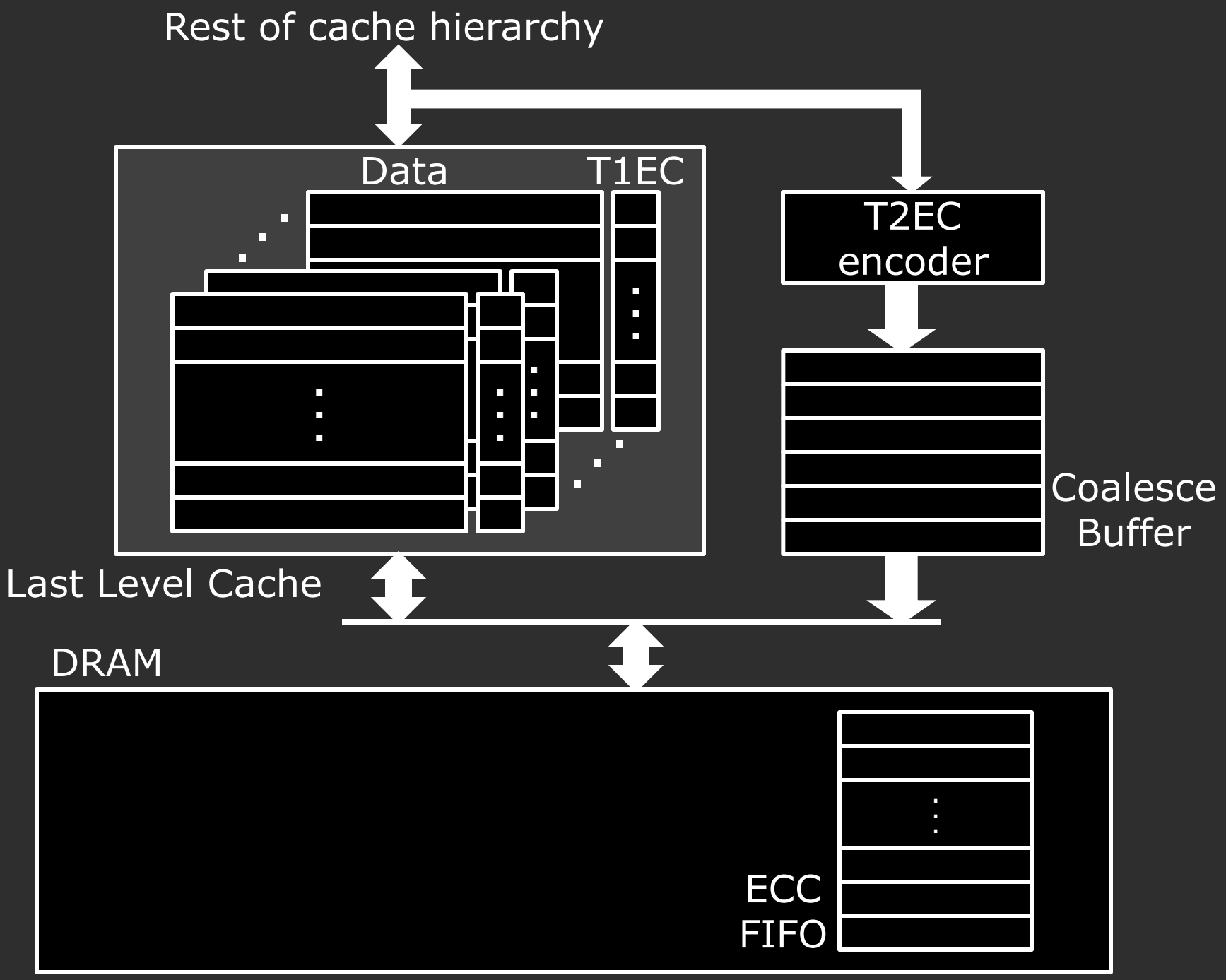


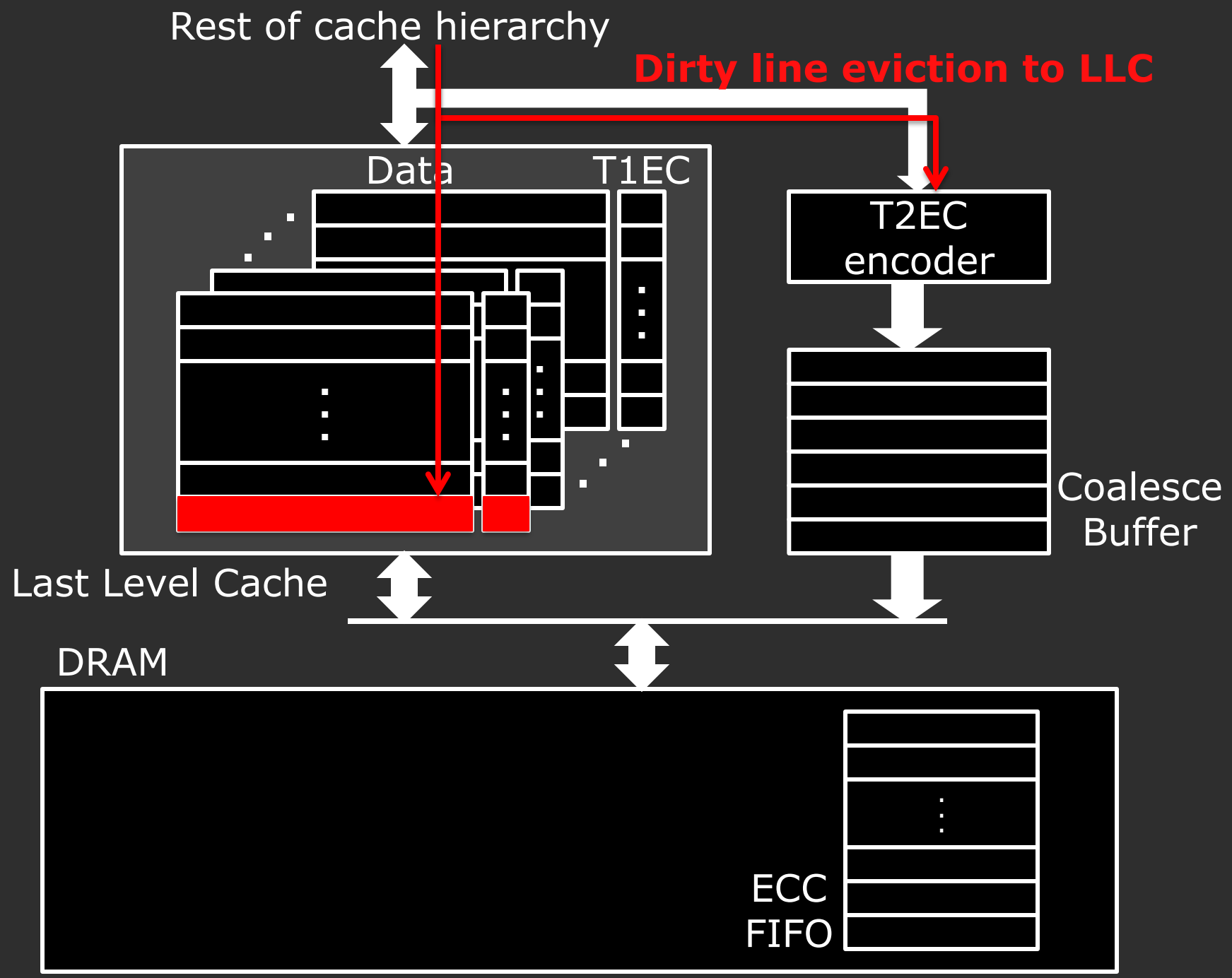
T2EC is memory mapped AND cached

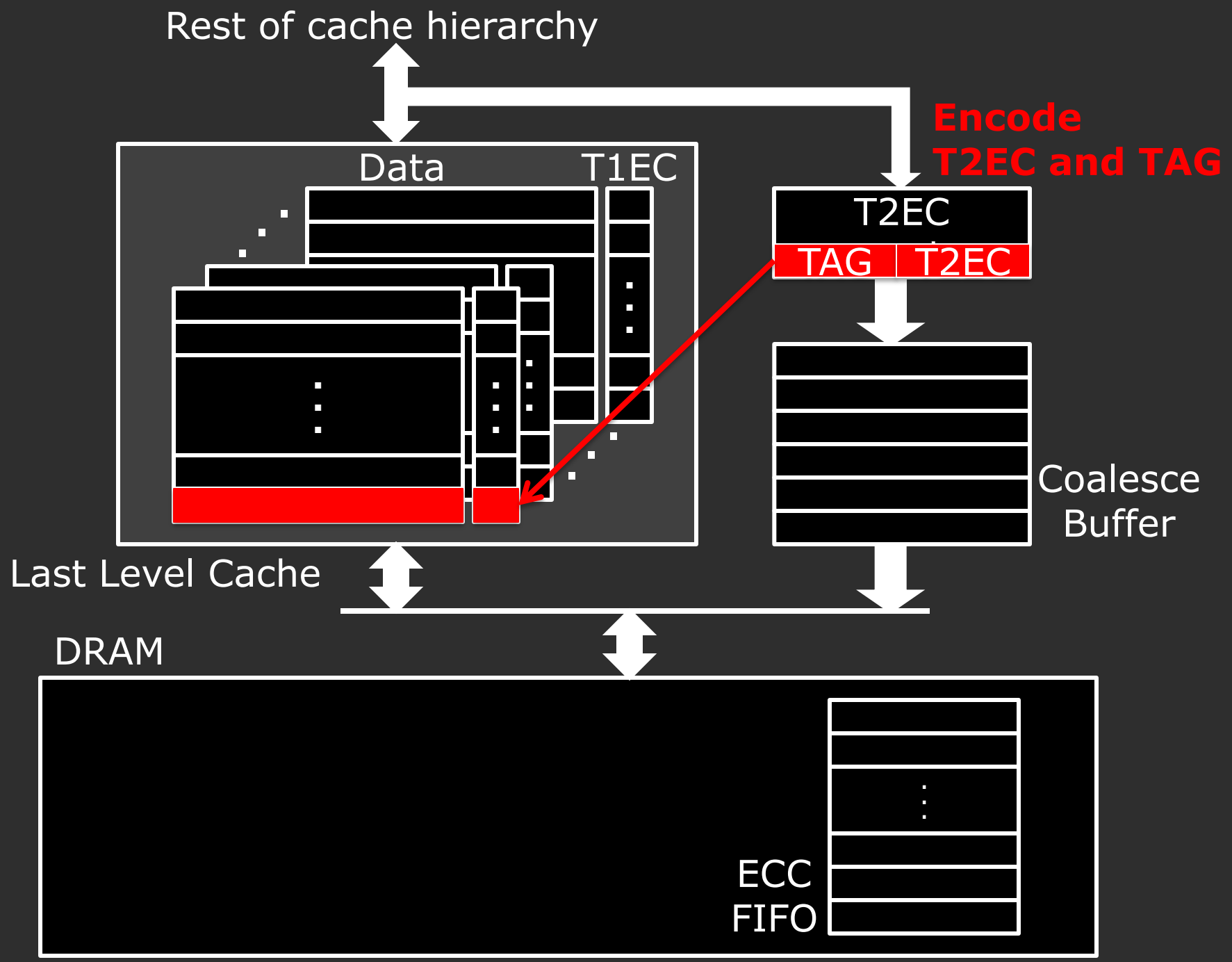
ECC FIFO

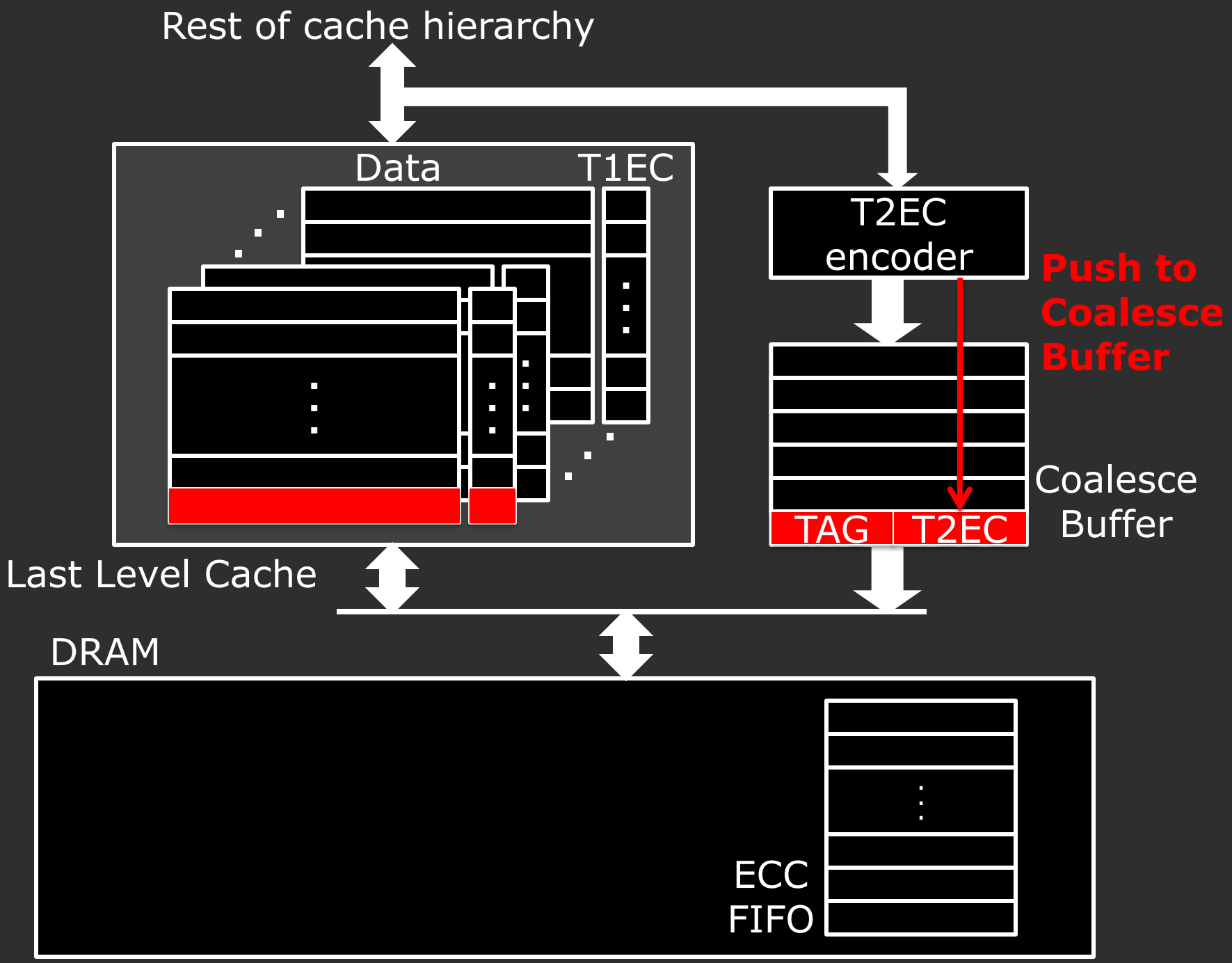
ECC FIFO

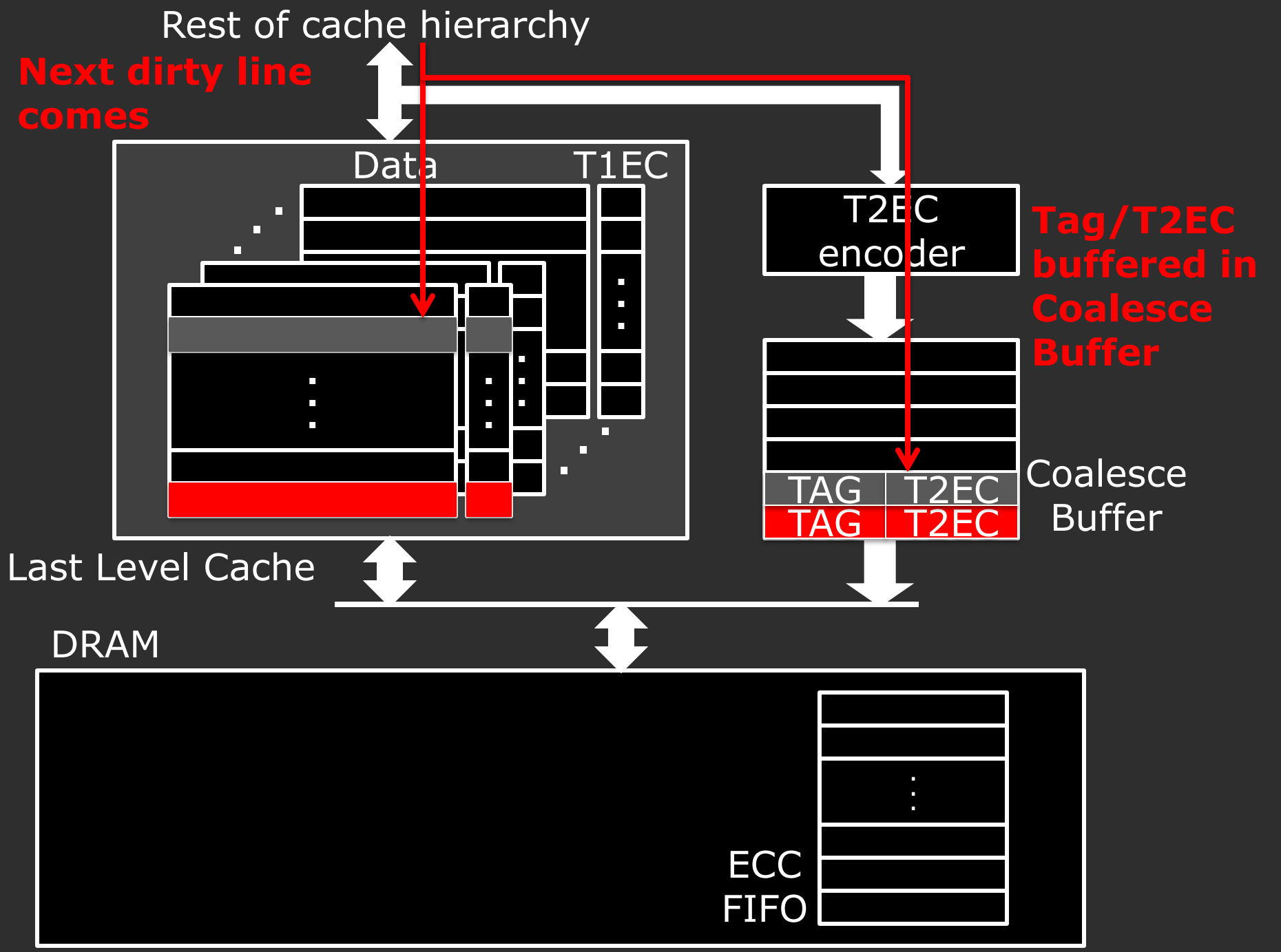
- Use Two-tiered error protection
- T2EC is off-loaded to FIFO in DRAM
  - LLC caching behavior is unaffected
- FIFO
  - Simple to manage
- Coalesce buffer
  - To better utilize DRAM channel

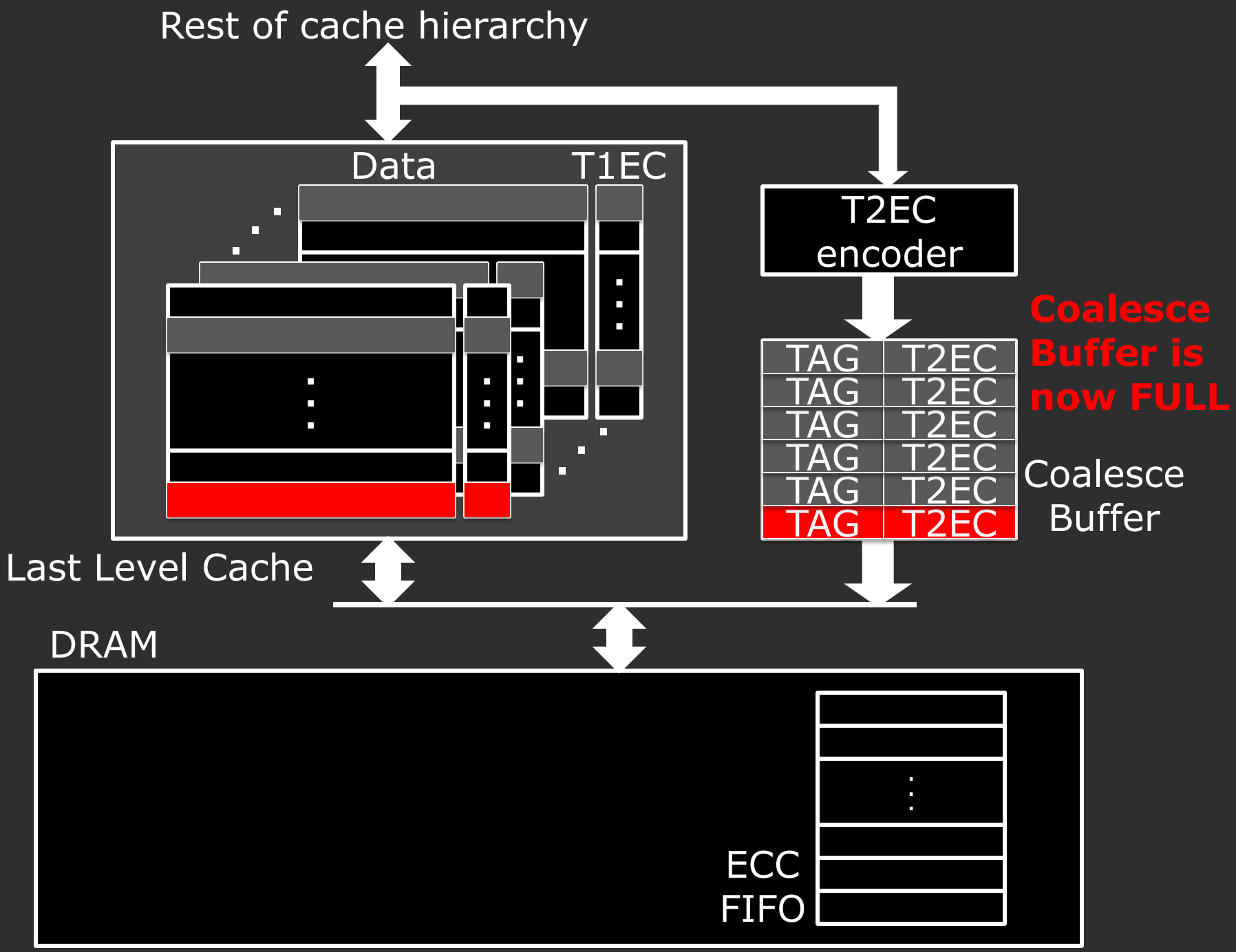

Rest of cache hierarchy
Data
T1EC
T2EC
encoder
Coalesce
Buffer
Last Level Cache
DRAM
ECC
FIFO

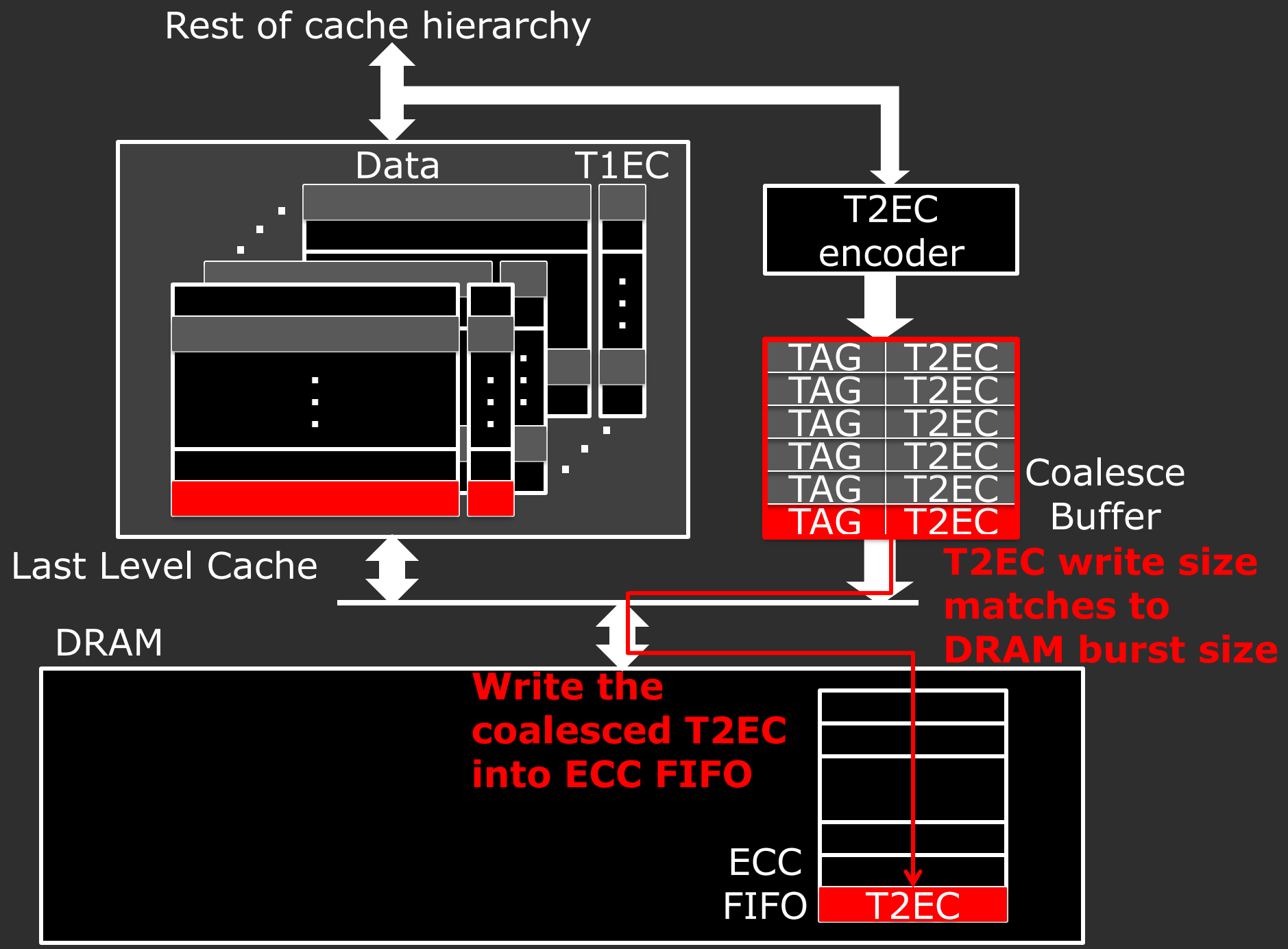

Rest of cache hierarchy
Dirty line eviction to LLC
Data
T1EC
T2EC encoder
Coalesce Buffer
Last Level Cache
DRAM
ECC FIFO

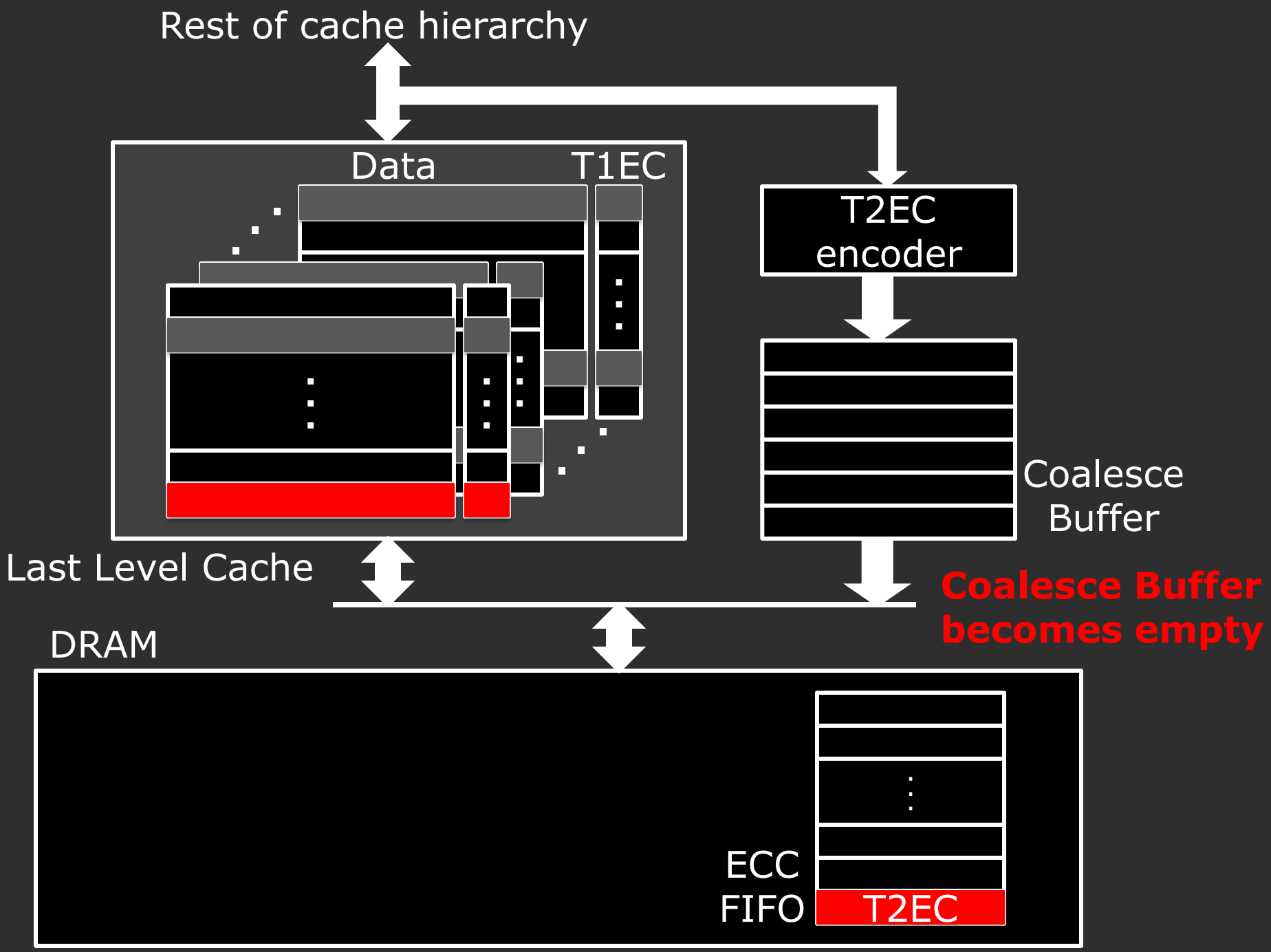

Rest of cache hierarchy
Encode
T2EC and TAG
Data
T1EC
T2EC
TAG
T2EC
Coalesce
Buffer
Last Level Cache
DRAM
ECC
FIFO


Rest of cache hierarchy
Data
T1EC
T2EC
encoder
Push to
Coalesce
Buffer
Coalesce
Buffer
TAG
T2EC
Last Level Cache
DRAM
ECC
FIFO


Rest of cache hierarchy
Next dirty line comes
Data
T1EC
T2EC encoder
Tag/T2EC buffered in Coalesce Buffer
TAG
T2EC
TAG
T2EC
Coalesce Buffer
Last Level Cache
DRAM
ECC FIFO


Rest of cache hierarchy
Data
T1EC
T2EC encoder
Coalesce Buffer is now FULL
TAG T2EC
TAG T2EC
TAG T2EC
TAG T2EC
TAG T2EC
TAG T2EC
Coalesce Buffer
Last Level Cache
DRAM
ECC FIFO


Rest of cache hierarchy
Data
T1EC
T2EC encoder
TAG T2EC
TAG T2EC
TAG T2EC
TAG T2EC
TAG T2EC
TAG T2EC
Coalesce Buffer
Last Level Cache
T2EC write size matches to DRAM burst size
DRAM
Write the coalesced T2EC into ECC FIFO
ECC FIFO
T2EC


Rest of cache hierarchy
Data
T1EC
T2EC encoder
Coalesce Buffer
Last Level Cache
Coalesce Buffer becomes empty
DRAM
ECC FIFO
T2EC

More on ECC FIFO

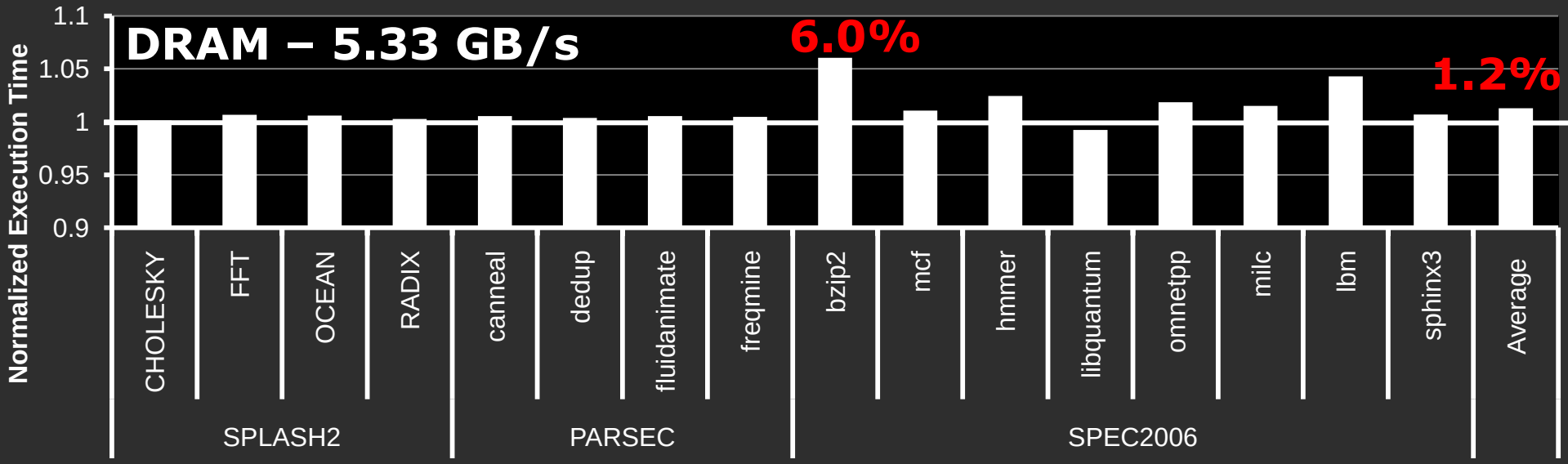
- Write-back data, but write-through ECC
- Potential performance degradation
  - Increased DRAM traffic due to T2EC writes
- Error correction
  - Search the matching TAG in coalesce buffer AND ECC FIFO
- May take a long time
  - Not a problem since SER is low
- Sometimes, may not find the matching TAG
  - ECC FIFO is finite
  - Potentially unprotected dirty lines – discussed later

EVALUATION

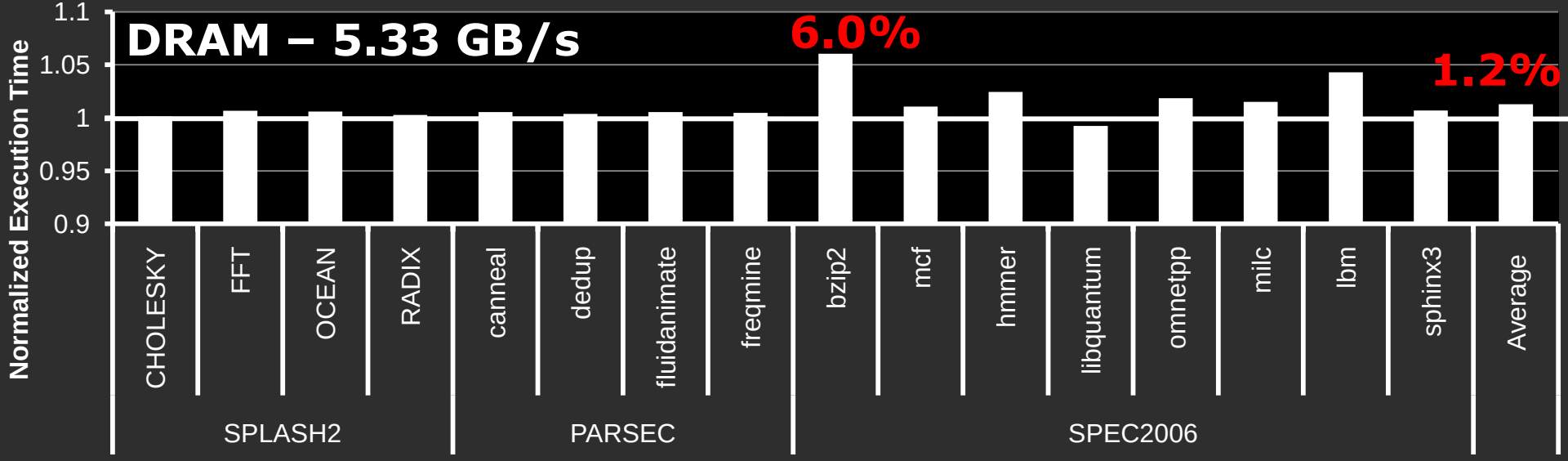
Performance Evaluation

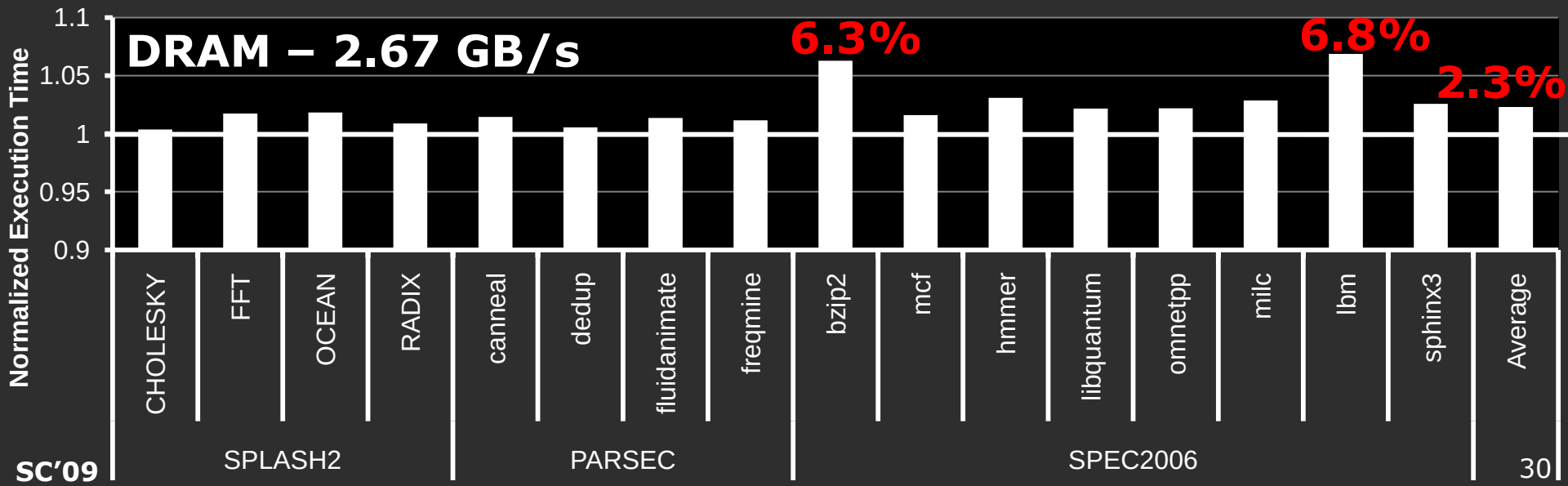
- GEMS + DRAMsim
  - An out-of-order SPARC V9 core
  - Exclusive two-level cache hierarchy
  - DDR2 667MHz – 5.33GB/s
  - Eager write-back
    - Clean dirty lines periodically
- Workloads
  - 16 data intensive applications
  - SPEC CPU 2006, PARSEC, and SPLASH2

Performance Penalty

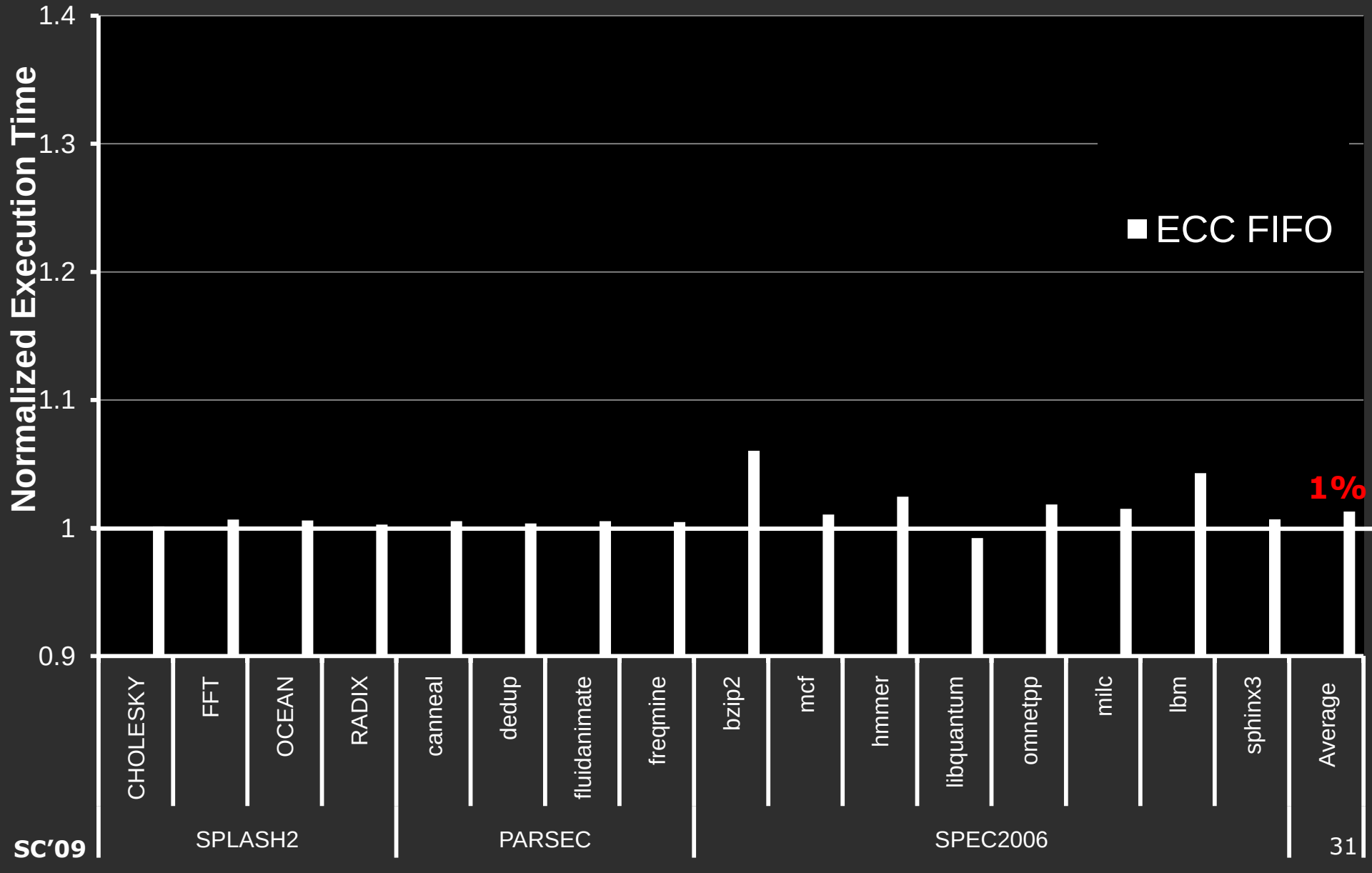


Performance Penalty

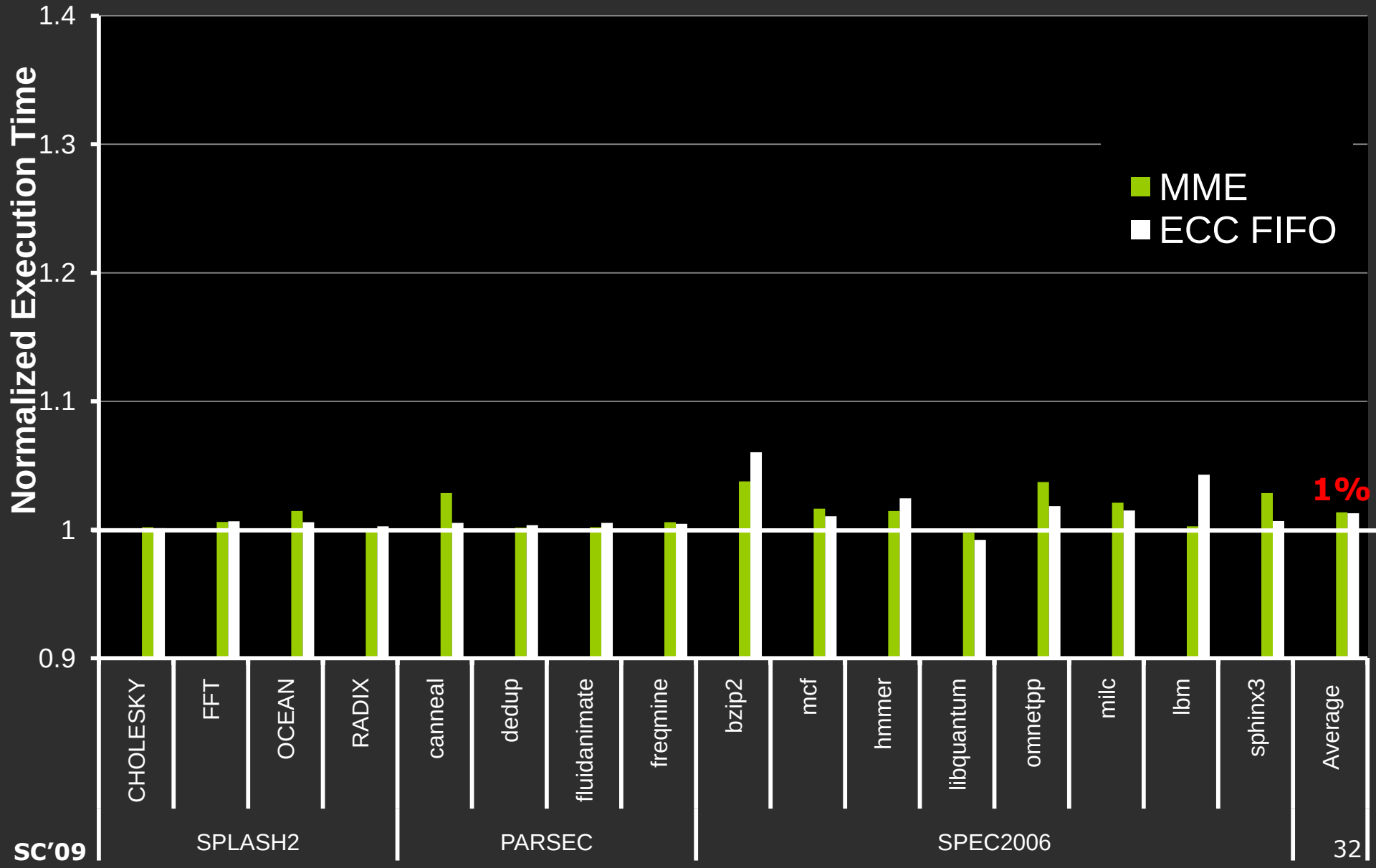




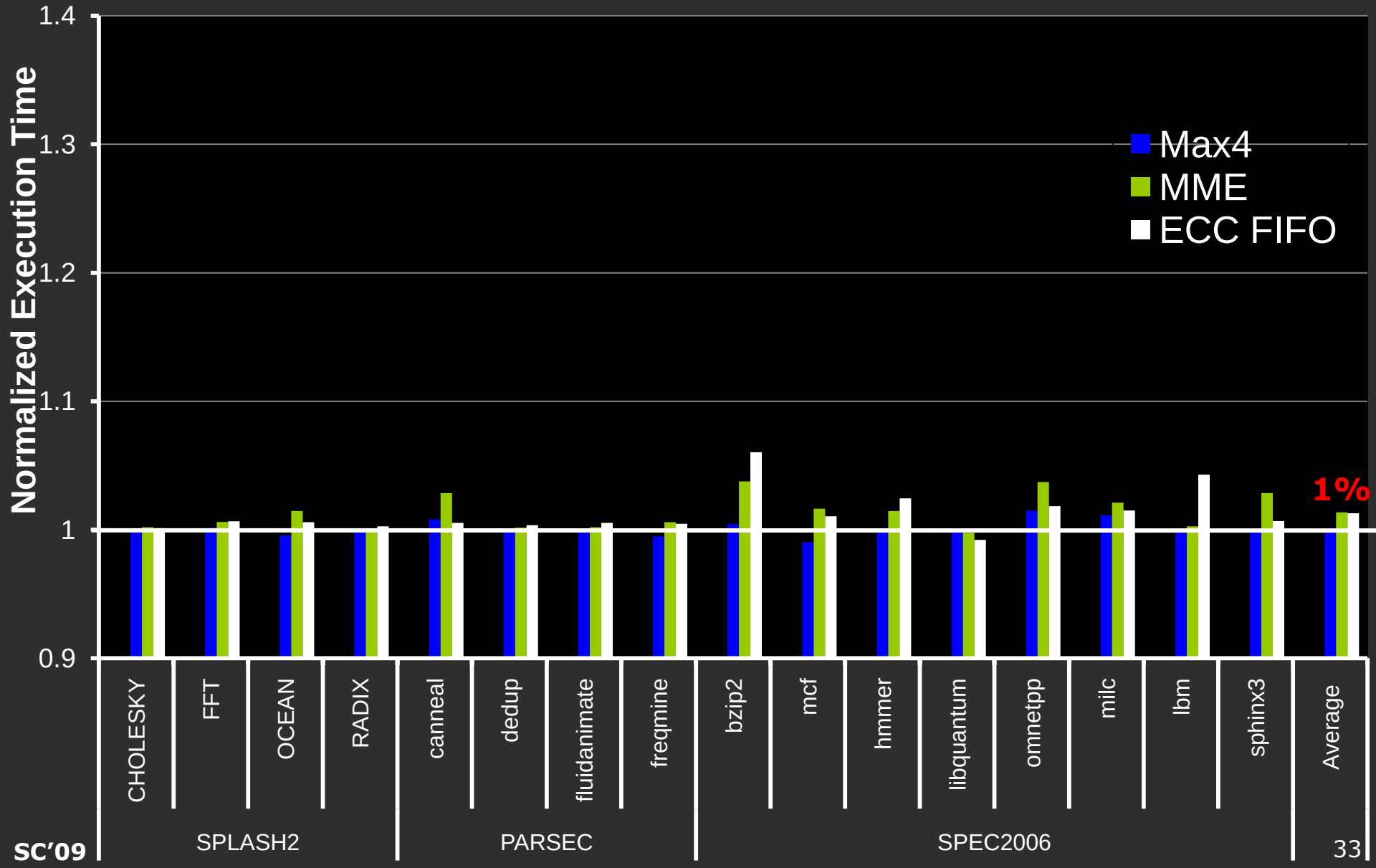
Comparison to MAXn and MME



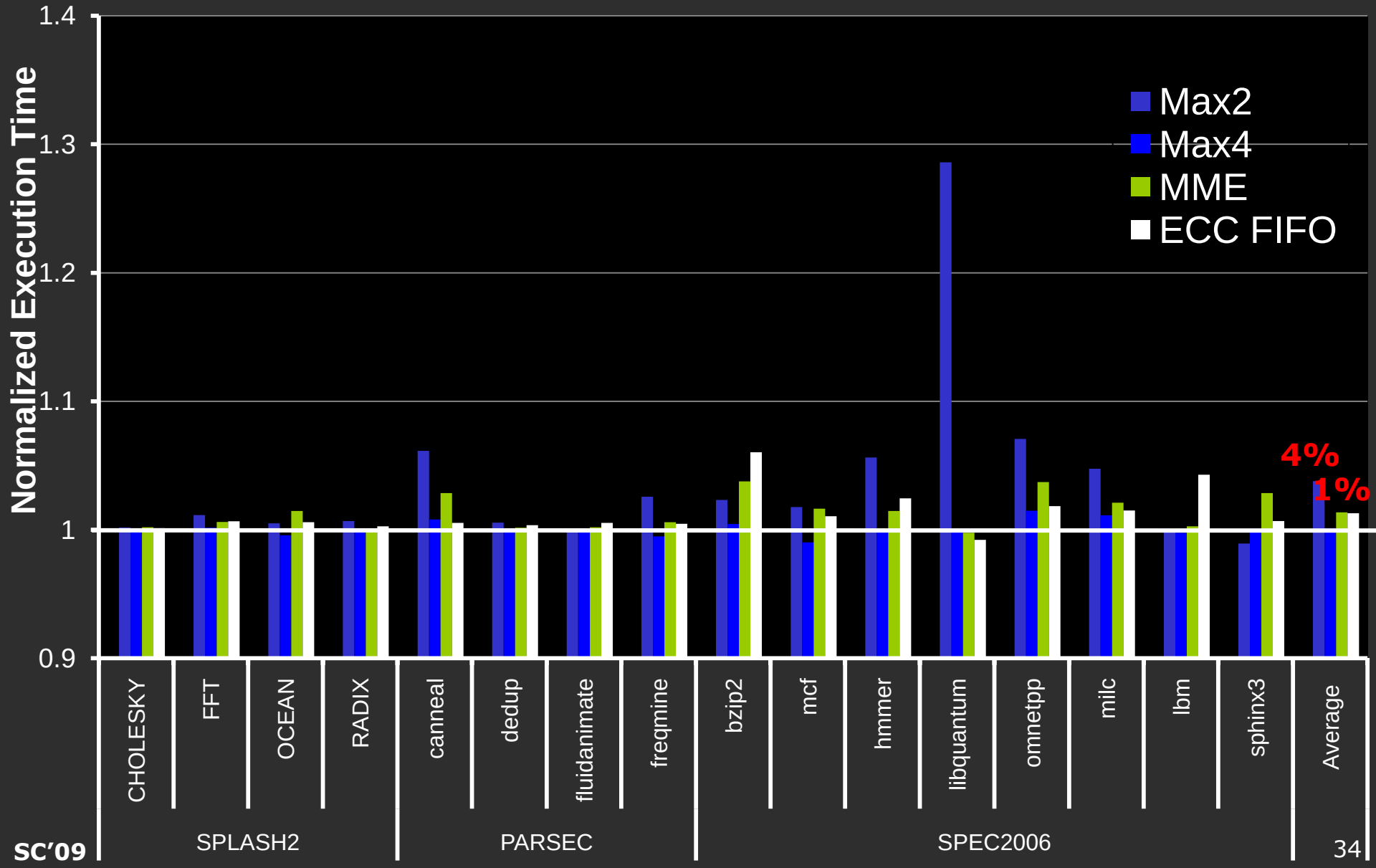
Comparison to MAXn and MME



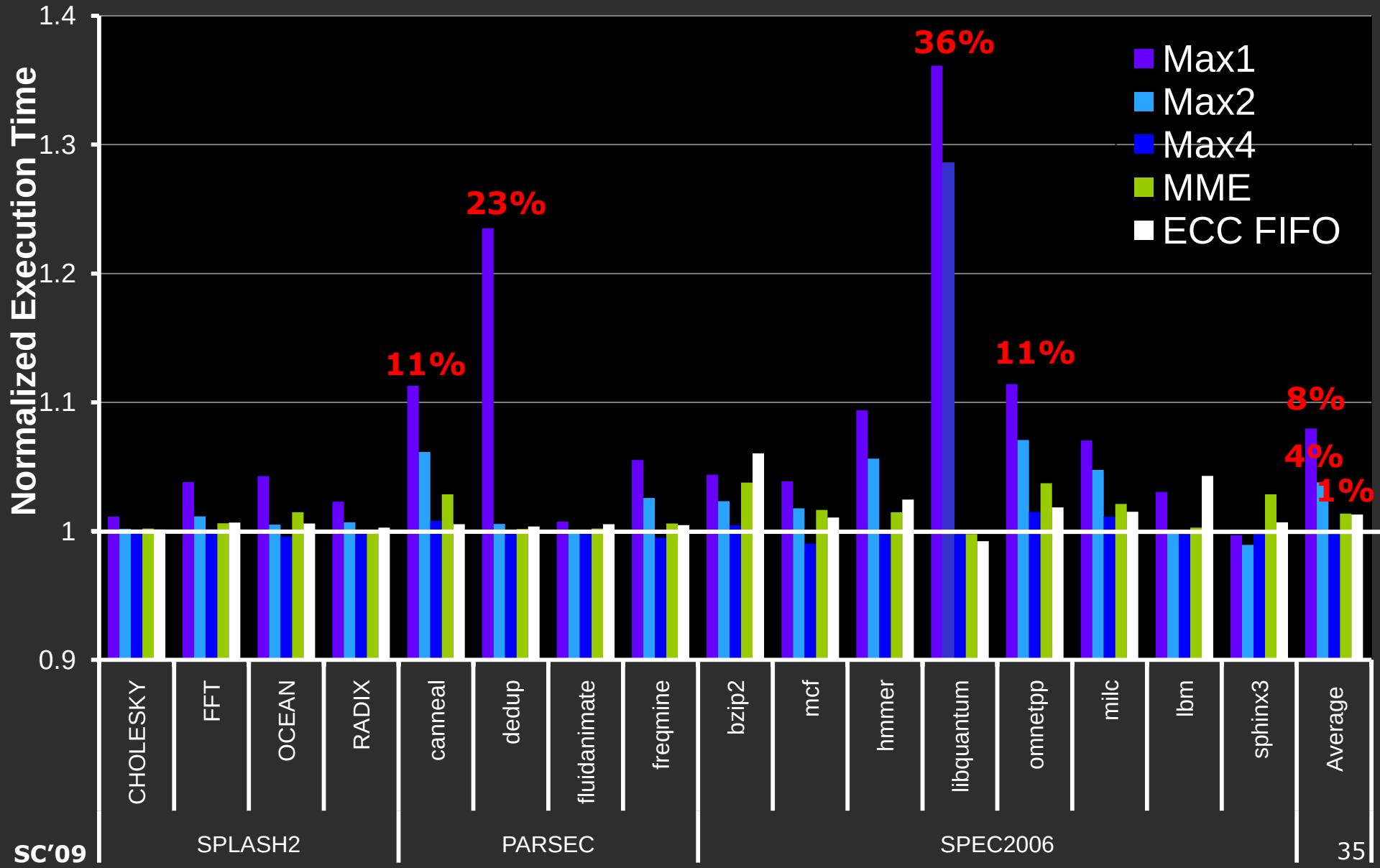
Comparison to MAXn and MME



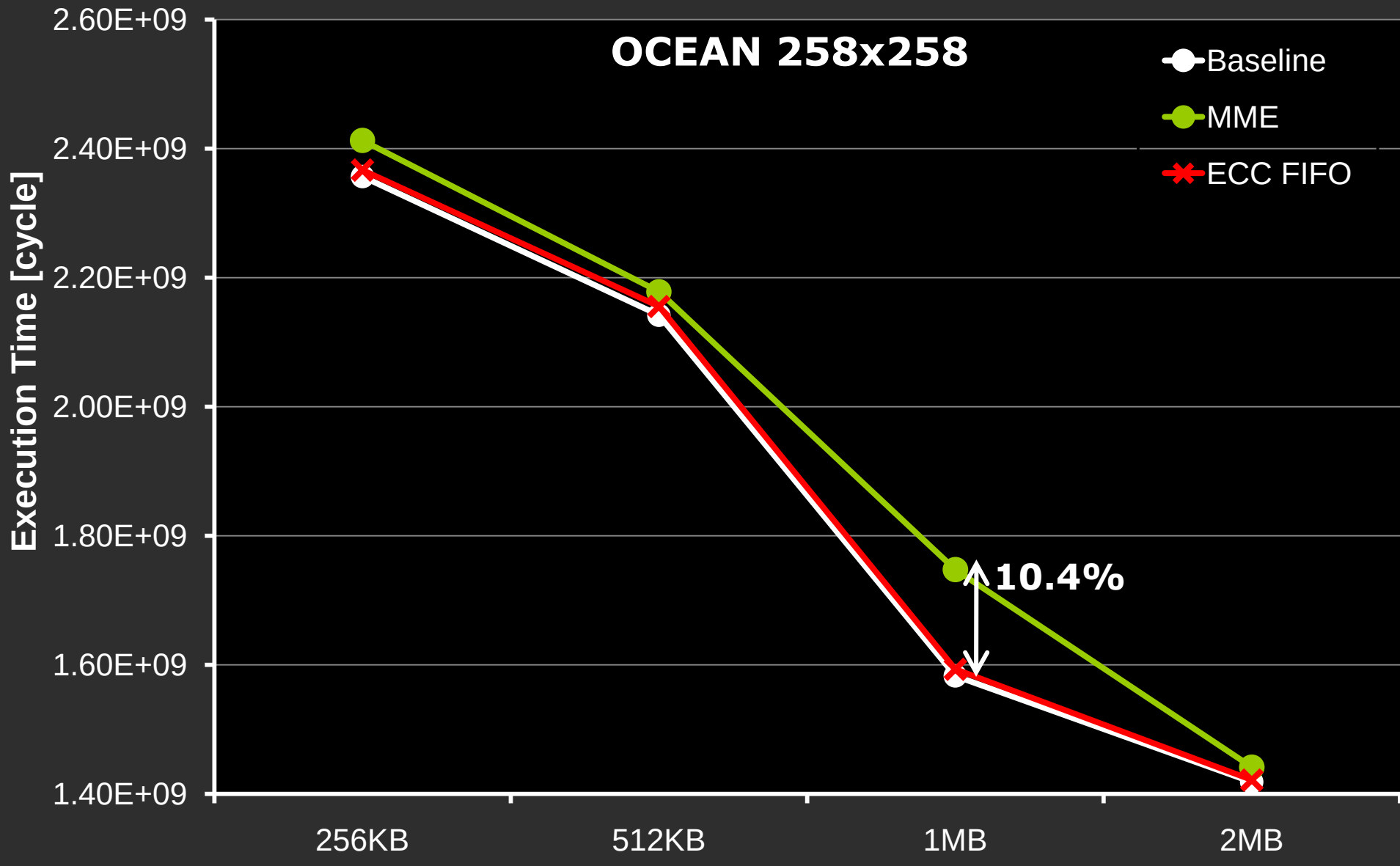
Comparison to MAXn and MME



Comparison to MAXn and MME



Comparison to MME



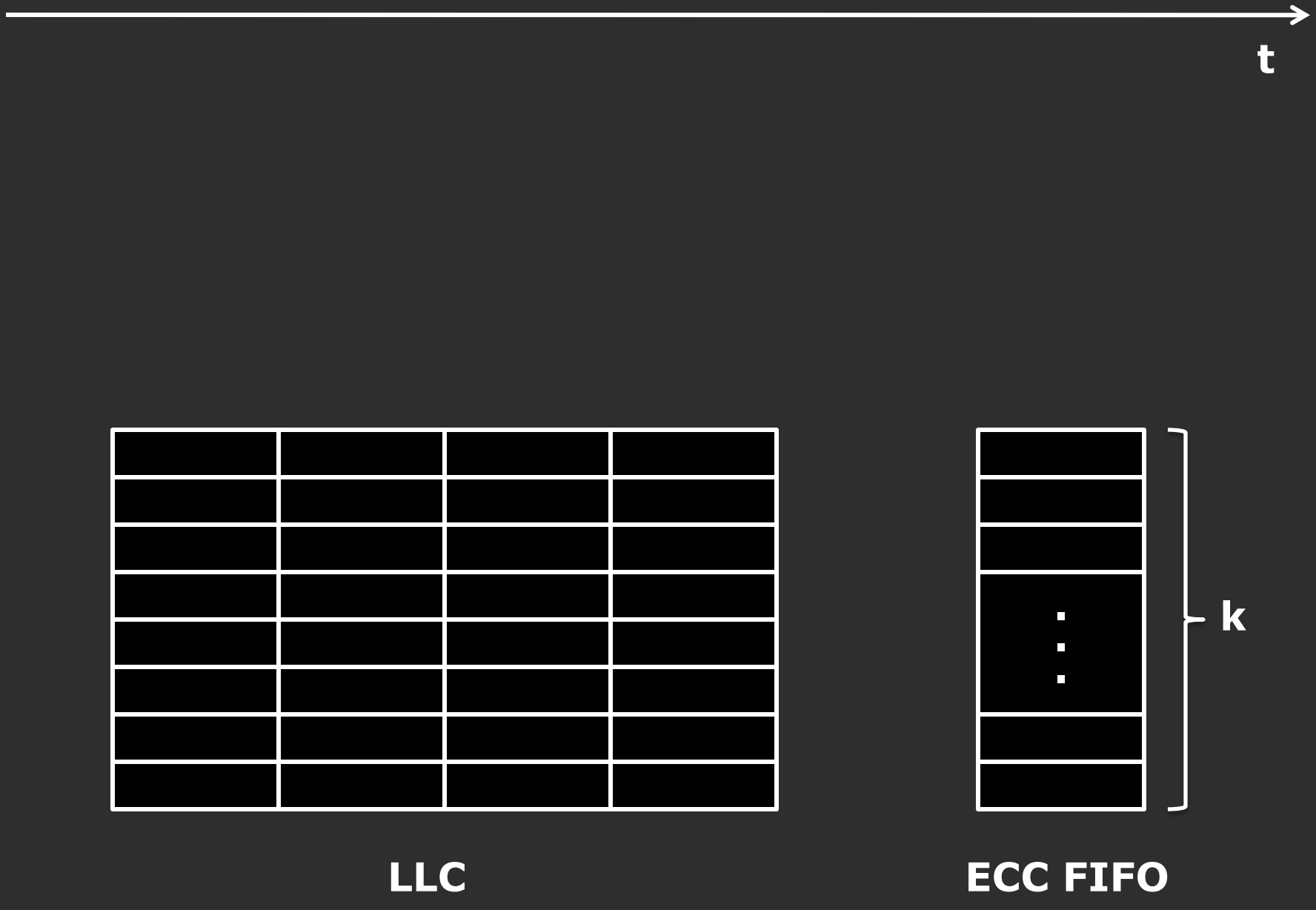
LLC Area and Power

- CACTI 5 model of a 45nm process
- Parity T1EC + SEC-DED T2EC
  - 15% area and 9% power saving
    - Compared to SEC-DED baseline
- SEC-DED T1EC + DEC-TED T2EC
  - 15% area and 10% power saving
    - Compared to DEC-TED baseline
- More error code examples in the paper

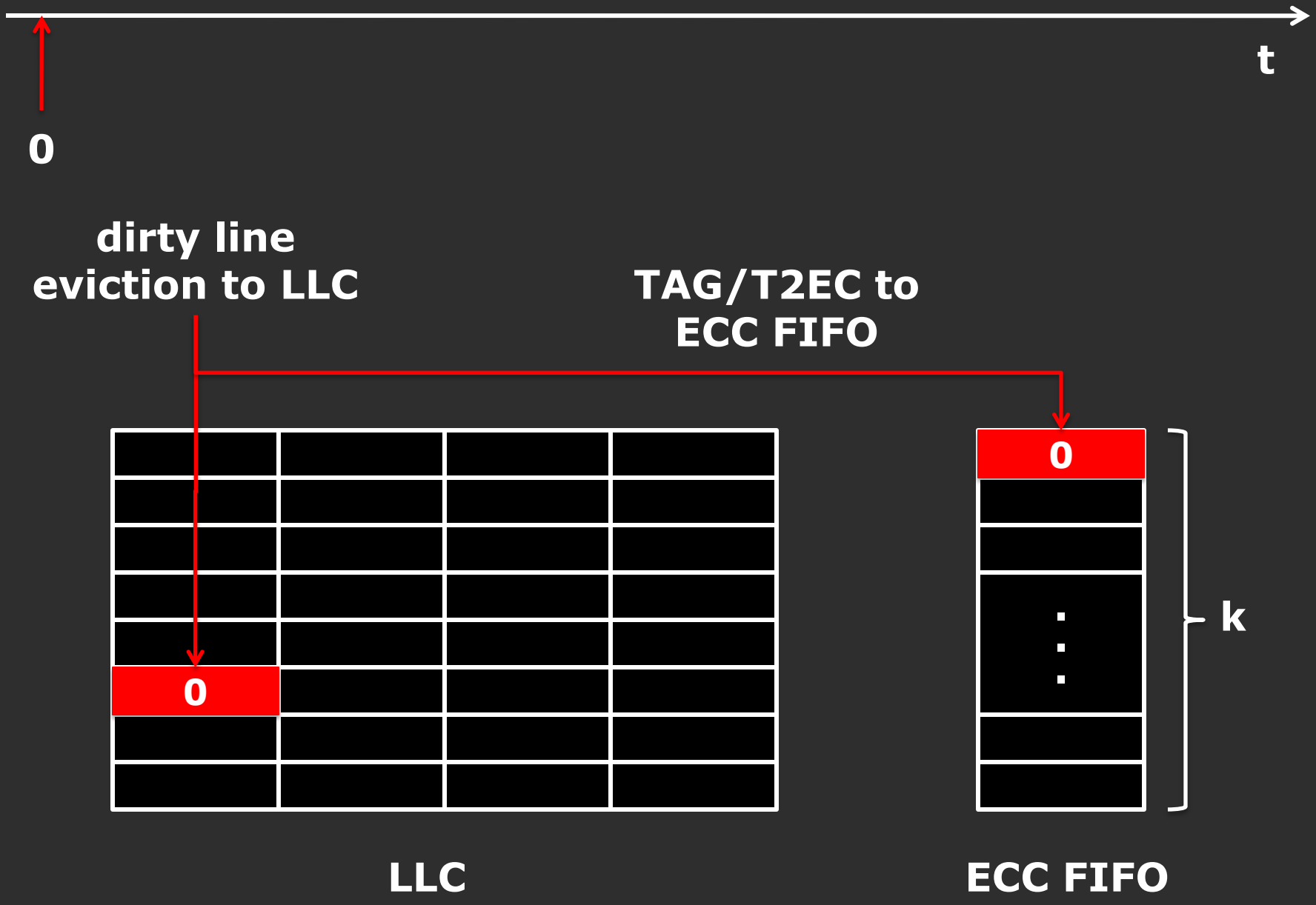
FIFO OVERWRITE

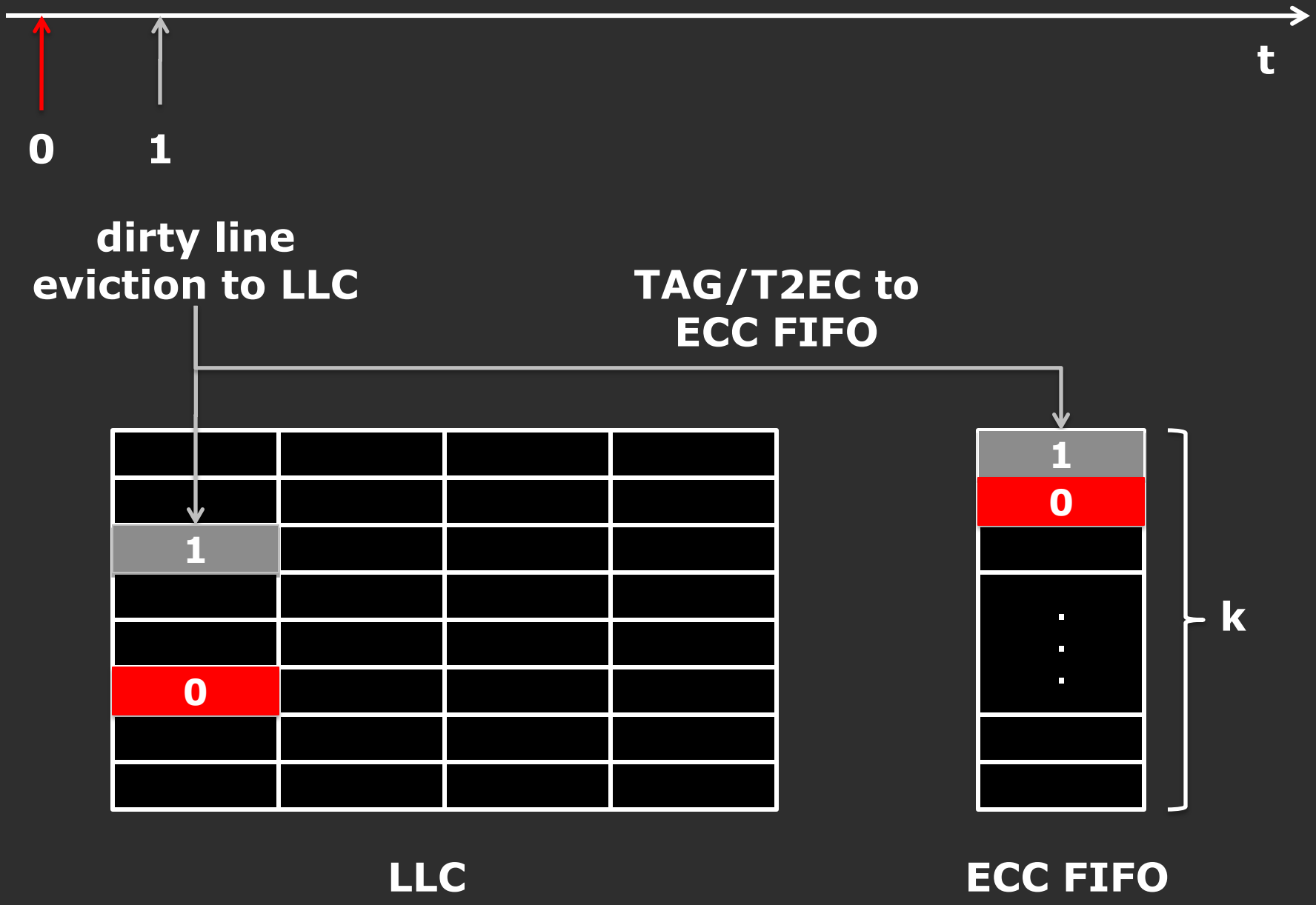
ECC FIFO is Finite

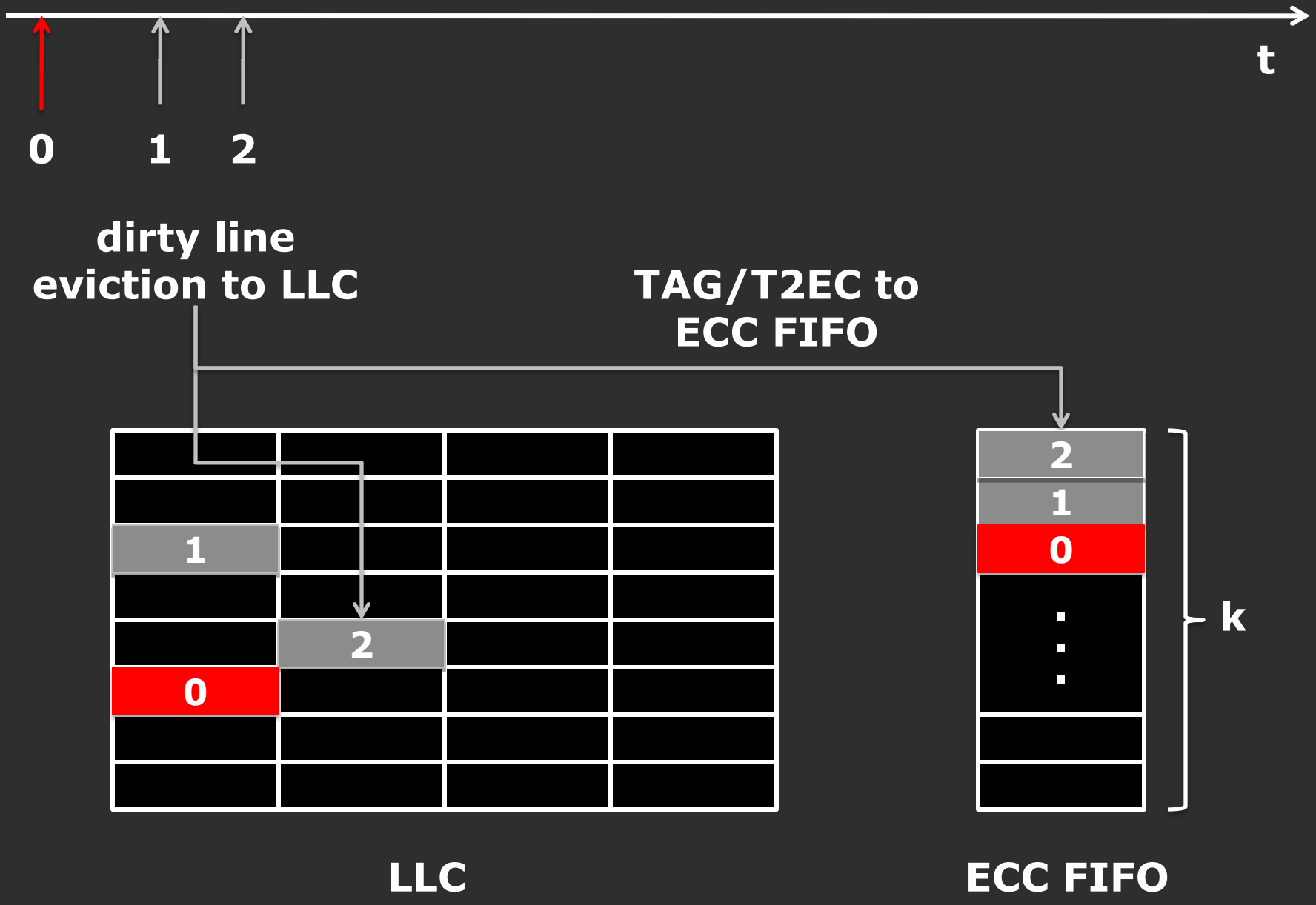
- ECC FIFO is implemented as a circular buffer
  - Each T2EC push overwrites the oldest entry in the FIFO
- If the associated data is still valid even after T2EC is overwritten
  - The cache line is no longer protected
  - Errors on this line cannot be corrected using T2EC

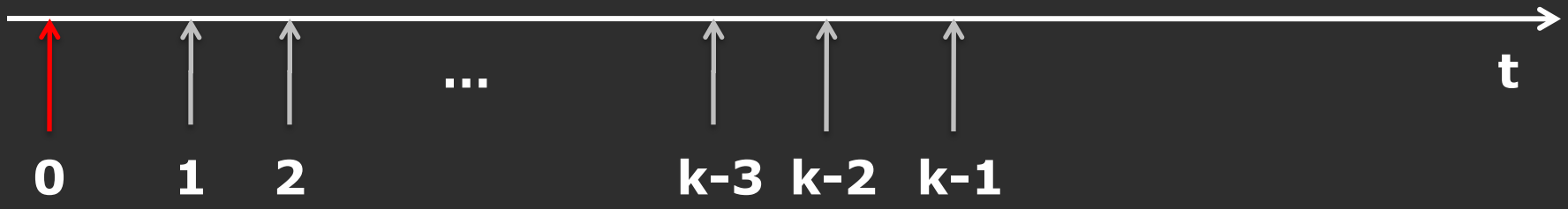


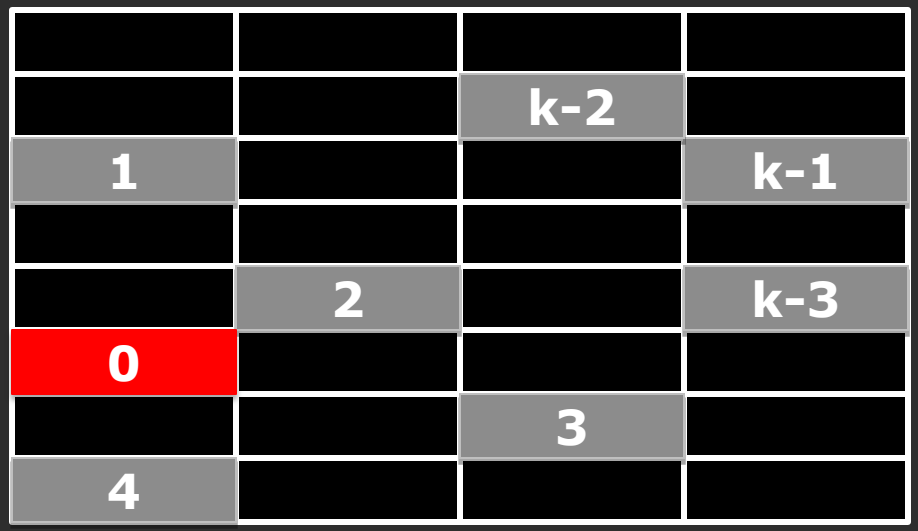
A small LLC and an ECC FIFO with k entries (no coalesce buffer for the simplicity)


t
0
dirty line
eviction to LLC
TAG/T2EC to
ECC FIFO
0
0
k
LLC
ECC FIFO

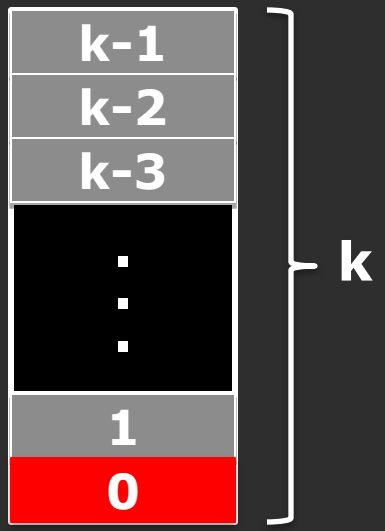

t
0
1
dirty line
eviction to LLC
TAG/T2EC to
ECC FIFO
1
0
1
0
k
LLC
ECC FIFO


t
0
1
2
dirty line
eviction to LLC
TAG/T2EC to
ECC FIFO
1
2
0
2
1
0
k
LLC
ECC FIFO

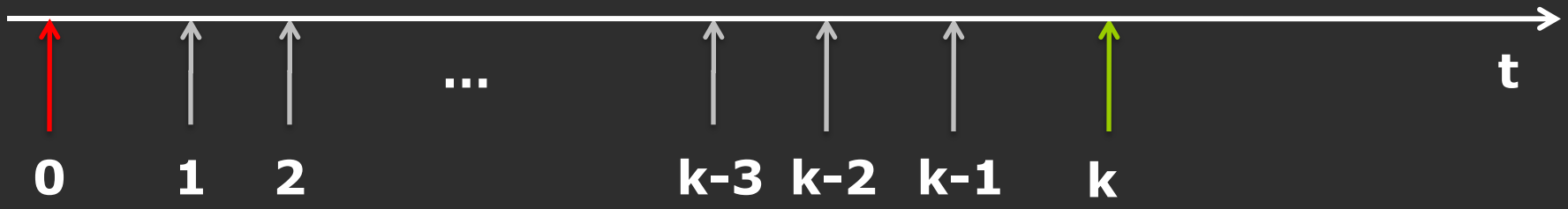




LLC

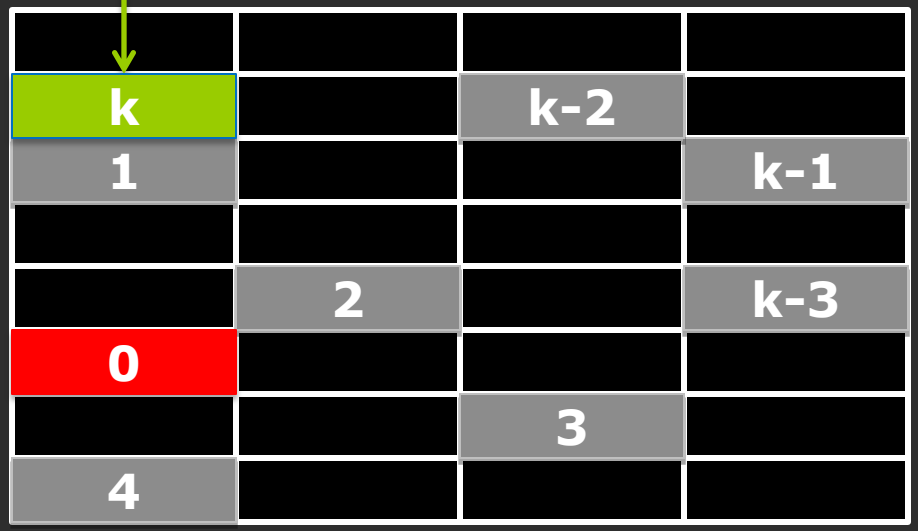


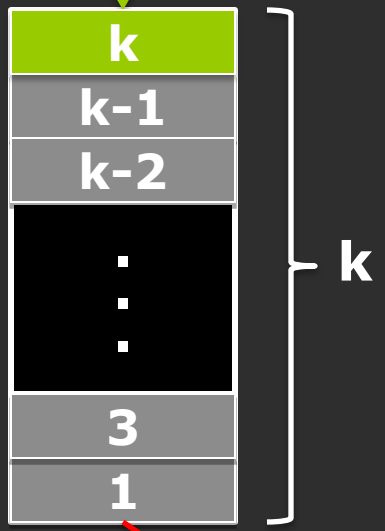
ECC FIFO



dirty line eviction to LLC

TAG/T2EC to ECC FIFO



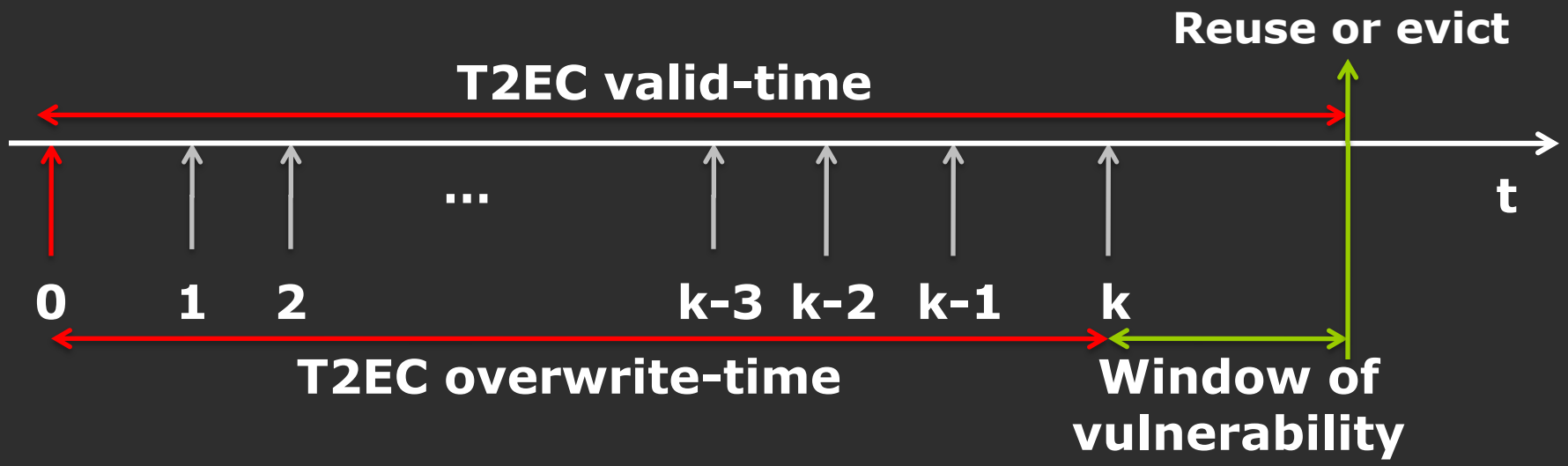


LLC

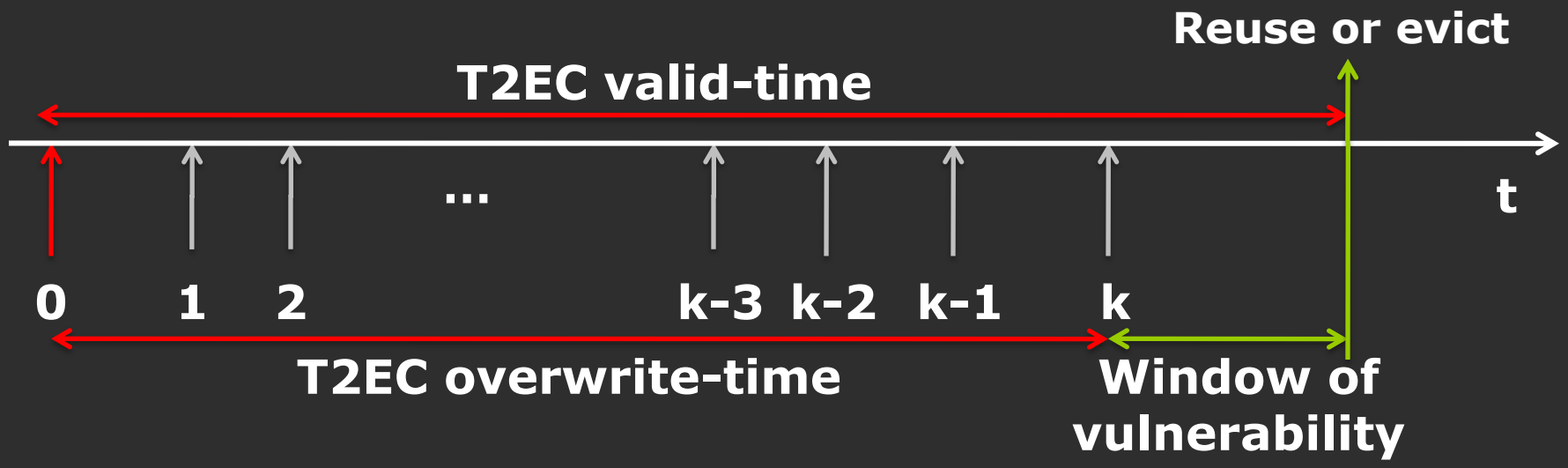
ECC FIFO

0

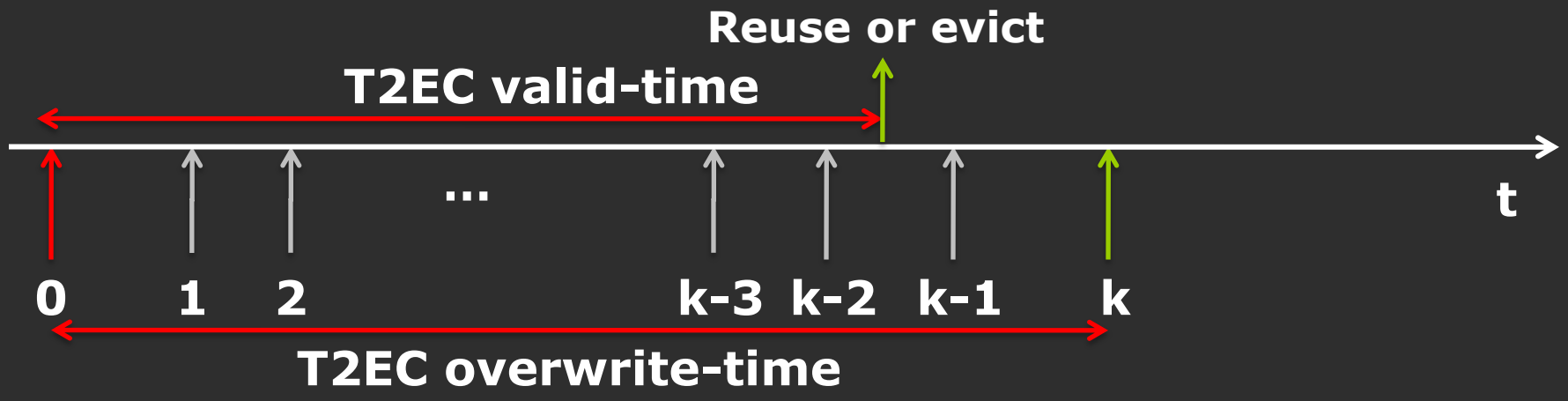
T2EC 0 is evicted from the FIFO
dirty line 0 is no longer protected



T2EC valid-time > T2EC overwrite-time
The cache line becomes T2EC unprotected



T2EC valid-time > T2EC overwrite-time
The cache line becomes T2EC unprotected



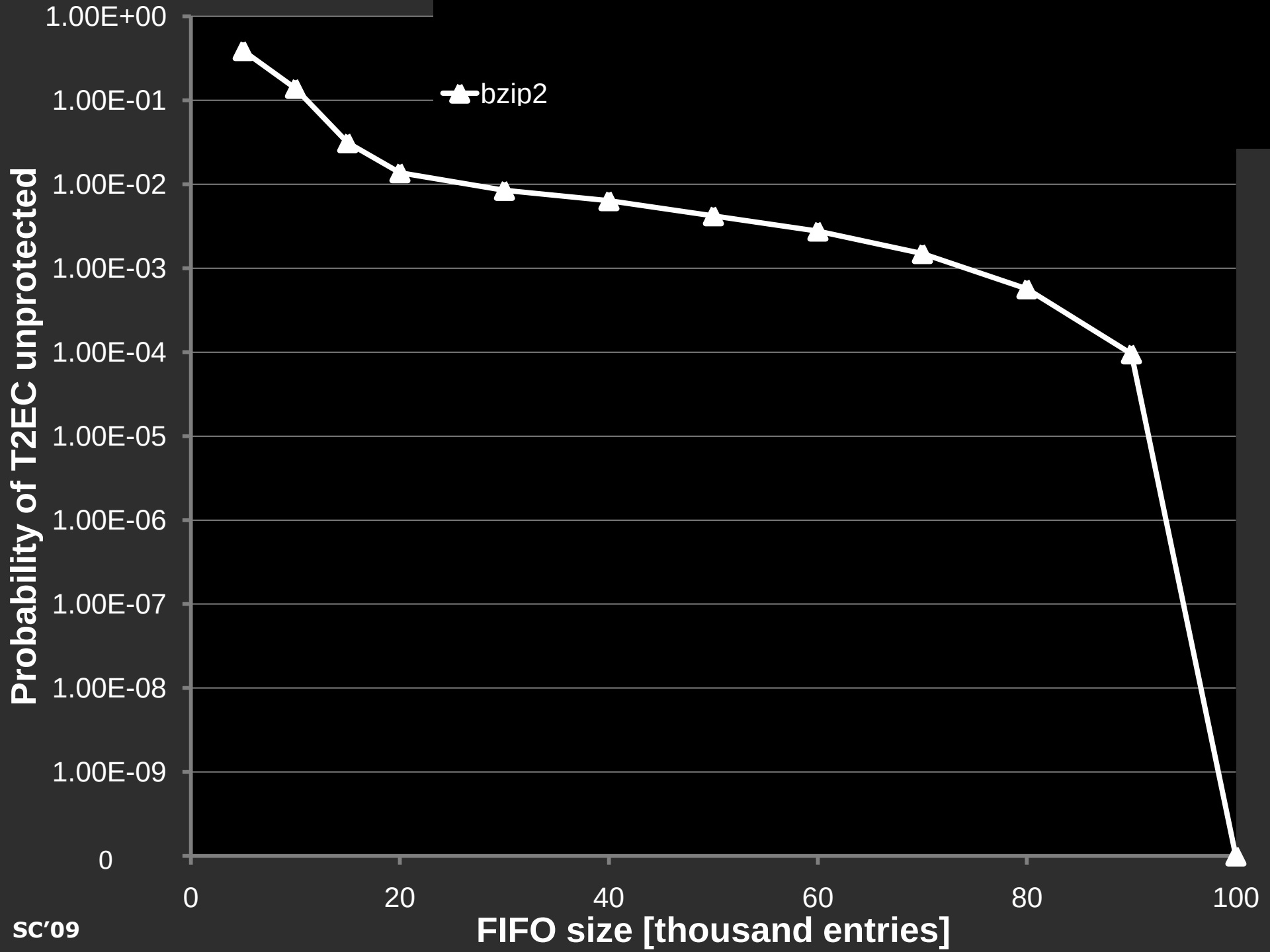
T2EC valid-time < T2EC overwrite-time
No window of vulnerability

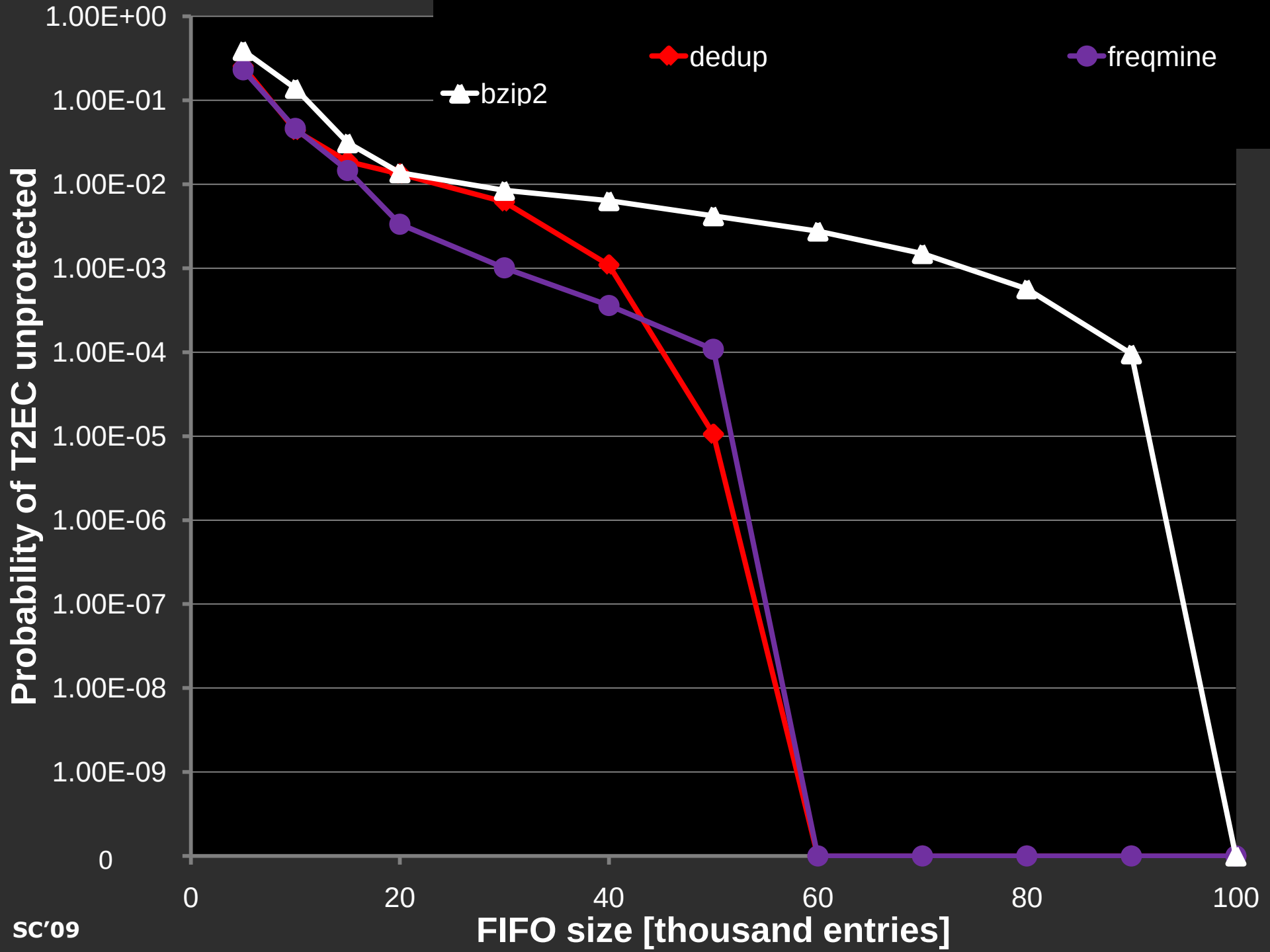
How to Avoid (or Mitigate) It?

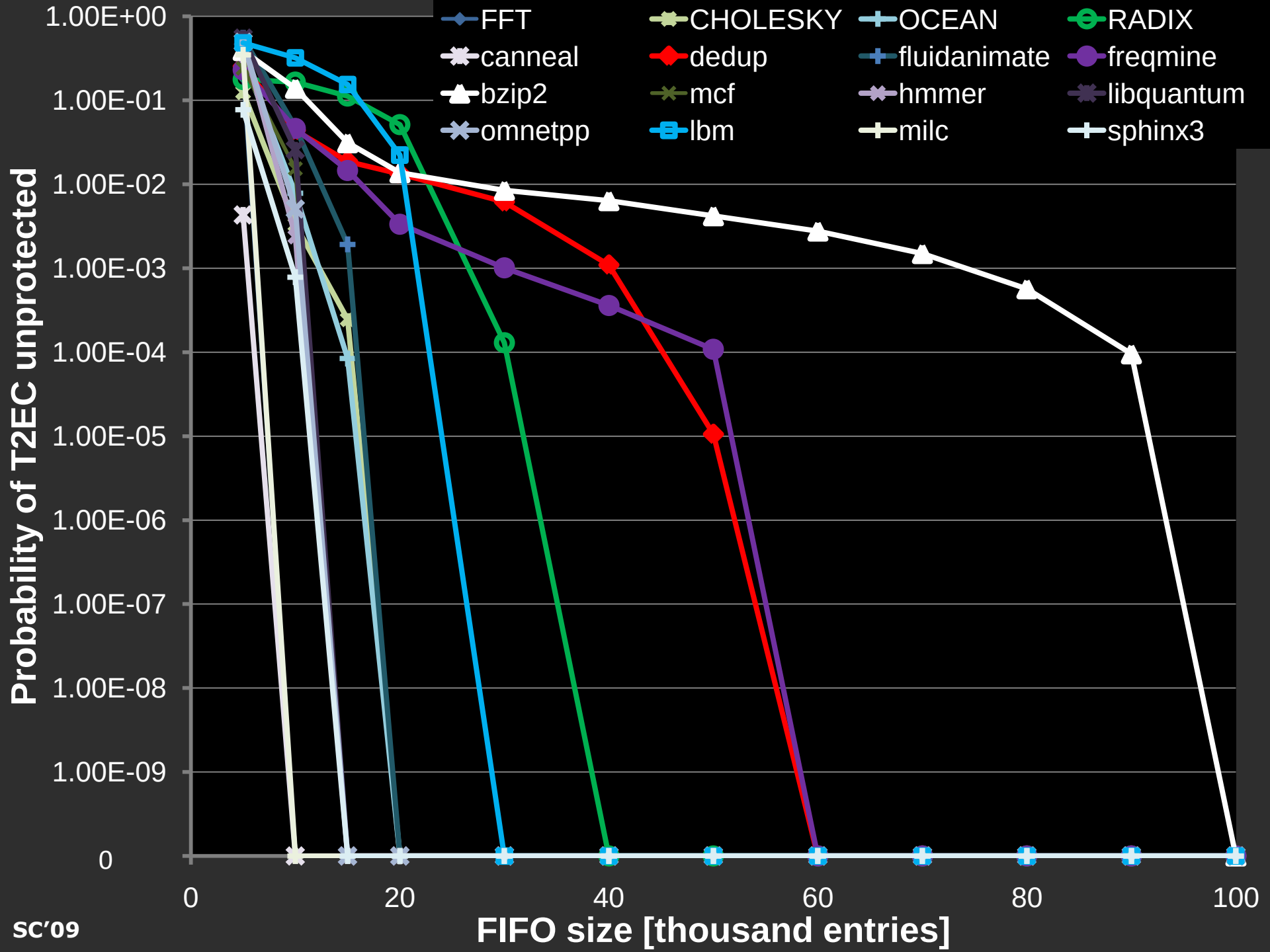
- T2EC overwrite-time↑
  - Make the FIFO bigger
    - The bigger the FIFO, the longer T2EC lifetime
    - Not sure how big it has to be
- T2EC valid-time↓
  - Inherent memory access pattern
    - A dirty line is re-used or evicted
    - Make the T2EC obsolete before it gets overwritten
  - Eager write-back
    - Limit the lifetime of dirty lines

Eager Write-Back

- Scan LLC lines periodically
  - Eagerly write-back dirty lines older than the period
    - It's cleaning, so it doesn't affect hit-rate
    - 1M cycle period
  - Limit the T2EC valid-time (max: 2xEWB period)
    - More T2ECs become obsolete before it gets overwritten
- Eager write-back improves performance
  - 6-10% in general, 26% in libquantum


1.00E+00
1.00E-01
1.00E-02
1.00E-03
1.00E-04
1.00E-05
1.00E-06
1.00E-07
1.00E-08
1.00E-09
0
Probability of T2EC unprotected
bzip2
0
20
40
60
80
100
FIFO size [thousand entries]


1.00E+00
1.00E-01
1.00E-02
1.00E-03
1.00E-04
1.00E-05
1.00E-06
1.00E-07
1.00E-08
1.00E-09
0
Probability of T2EC unprotected
dedup
freqmine
bzip2
0
20
40
60
80
100
FIFO size [thousand entries]


FFT
CHOLESKY
OCEAN
RADIX
canneal
dedup
fluidanimate
freqmine
bzip2
mcf
hmmer
libquantum
omnetpp
lbm
milc
sphinx3
1.00E+00
1.00E-01
1.00E-02
1.00E-03
1.00E-04
1.00E-05
1.00E-06
1.00E-07
1.00E-08
1.00E-09
0
Probability of T2EC unprotected
0
20
40
60
80
100
FIFO size [thousand entries]
SC'09

Required FIFO size

- 100k entry FIFO, the worst case
  - 1MB of storage in DRAM
- More in the paper
  - A simple analytic model on Probability of unprotected
  - Required FIFO size with regards to eager write-back period
  - how to guarantee zero probability of unprotected

Conclusion

- ECC FIFO is an efficient low cost error protection mechanism
  - A simple FIFO off-loads the overhead of strong T2EC
    - 15-25% LLC area reduction
    - 10-17% LLC power saving
  - Does not affect LLC caching behavior
  - Choosing error code for T1EC/T2EC is flexible
    - See more error code examples in the paper
- Penalties
  - Average 1% performance degradation – negligible
  - Increased error correction latency – but SER is low
  - Potentially unprotected cache lines
    - Can make this quite low or even guarantee not to occur

Flexible Cache Error Protection using an ECC FIFO

Doe Hyun Yoon and Mattan Erez

Dept. Electrical and Computer Engineering
The University of Texas at Austin