The Dirty-Block Index

Vivek Seshadri
Abhishek Bhowmick · Onur Mutlu
Phillip B. Gibbons · Michael A. Kozuch · Todd C. Mowry
Summary

• Problem: Dirty bit organization in caches does not match queries
  – Inefficiency and performance loss

• The Dirty-Block Index (DBI)
  – Remove dirty bits from cache tag store
  – DRAM row-oriented organization of dirty bits

• Efficiently respond to queries
  – Get all dirty blocks of a DRAM row; Is block B dirty?

• Enables efficient implementation of many optimizations
  – DRAM-aware writeback, bypassing cache lookup, reducing ECC cost, ...

• Improves performance while reducing overall cache area
  – 28% performance over baseline, 6% over state-of-the-art (8-core)
  – 8% cache area reduction
Mismatch leads to inefficiency
Mismatch between Organization and Query

Sorted by title

Get all the books written by author X

Bad organization for the query
Metadata: Information About a Cache Block

- **Block Address**
- **Valid Bit**
- **Dirty Bit** (Writeback cache)
- **Sharing Status** (Multi-cores)
- **Replacement Policy** (Set-associative cache)
- **Error Correction** (Reliability)
Block-Oriented Metadata Organization

- Sharing Status (Multi-cores)
- Replacement Policy (Set-associative cache)
- Block Address
- V (Valid Bit)
- D (Dirty Bit)
- Sh (Sharing Status)
- Repl (Replacement Policy)
- ECC (Error Correction)
- Dirty Bit (Writeback cache)
- Error Correction (Reliability)
Block-Oriented Metadata Organization

- Block Address
- V
- D
- Sh
- Repl
- ECC

- Tag Entry

- Cache
- Tag Store

- Simple to Implement
- Scalable

Any metadata query requires an expensive tag store lookup
Is this the best organization?
Block-Oriented Metadata Organization

- Block Address
- V
- D
- Sh
- Repl
- ECC

- Tag Entry
- Cache
- Tag Store

- Simple to Implement
- Scalable

Any metadata query requires an expensive tag store lookup.

Is this the best organization?
Focus of This Work

Dirty Bit
Queried by many operations and optimizations
Is putting the dirty bit in the tag entry the best approach?
Outline

✓ Introduction
  • Shortcomings of Block-Oriented Organization
  • The Dirty-Block Index (DBI)
  • Optimizations Enabled by DBI
  • Evaluation
  • Conclusion
DRAM-Aware Writeback

Virtual Write Queue [ISCA 2010], DRAM-Aware Writeback [TR-HPS-2010-2]

1. Buffer writes and flush them in a burst
2. Row buffer hits are faster and more efficient than row misses
DRAM-Aware Writeback

Virtual Write Queue [ISCA 2010], DRAM-Aware Writeback [TR-HPS-2010-2]

Last-Level Cache ➔ Dirty Block

Proactively write back all other dirty blocks from the same DRAM row

Significantly increases the DRAM write row hit rate

Get all dirty blocks of DRAM row ‘R’
Shortcoming of Block-Oriented Organization

Get all dirty blocks of DRAM row ‘R’
Get all dirty blocks of DRAM row ‘R’

Set of blocks co-located in DRAM
~8KB = 128 cache blocks

Cache Tag Store

Is block 1 of Row R dirty?
Is block 2 of Row R dirty?
Is block 3 of Row R dirty?

Is block 128 of Row R dirty?
Get all dirty blocks of DRAM row ‘R’

Requires many expensive (possibly unnecessary) tag lookups

Inefficient

Significantly increases tag store contention
Many Cache Optimizations/Operations

- DRAM-aware Writeback
- Bulk DMA
- Cache Flushing
- DRAM Write Scheduling
- Bypassing Cache Lookup
- Metadata for Dirty Blocks
- Load Balancing Memory Accesses
Queries for the Dirty Bit Information

Get all dirty blocks that belong to a coarse-grained region

Block-based dirty bit organization is inefficient for both queries

Is block ‘B’ dirty?
The Dirty-Block Index

Outline

✓ Introduction
✓ Shortcomings of Block-Oriented Organization
  • The Dirty-Block Index (DBI)
  • Optimizations Enabled by DBI
  • Evaluation
  • Conclusion
The Dirty-Block Index

DRAM row-oriented organization of dirty bits
The Dirty-Block Index

- Block Address
- V
- Sh
- Repl
- ECC

- DRAM row address
- V
- D D D D D

DBI entry valid bit
Dirty bit vector
(one bit per block)
A block in the cache is dirty \textit{if and only if}\n
1. The DBI has a valid entry for the DRAM row that contains the block, and

2. The dirty bit for the block in the bit vector of the corresponding DBI entry is set
DBI Semantics by Example

Even if it is present in the cache, it is not dirty.
Benefits of DBI

Get all dirty blocks of DRAM row ‘R’

A single lookup to Row R in the DBI

Compared to 128 lookups with existing organization

Is block ‘B’ dirty?

DBI is faster than the tag store
Outline

- Introduction
- Shortcomings of Block-Oriented Organization
- The Dirty-Block Index (DBI)
  - Optimizations Enabled by DBI
  - Evaluation
  - Conclusion
The Dirty-Block Index

1 DRAM-Aware Writeback

Virtual Write Queue [ISCA 2010], DRAM-Aware Writeback [TR-HPS-2010-2]

Dirty Block

Proactively write back all other dirty blocks from the same DRAM row

DBI achieves the benefit of DRAM-aware writeback without increasing contention for the tag store!

Look up the cache only for these blocks
Bypassing Cache Lookups

Mostly-No Monitors [HPCA 2003], SkipCache [PACT 2012]

If an access is likely to miss, we can bypass the tag lookup!

Reduces access latency/energy; Reduces tag store contention

DBI seamlessly enables *simpler and more aggressive miss predictors!*

1. No false negatives
2. Write through

Forward to next level
Reducing ECC Overhead

ECC-Cache [IAS 2009], Memory-mapped ECC [ISCA 2009], ECC-FIFO [SC 2009]

Dirty block – Requires error correction
Clean block – Requires only error detection

ECC for dirty blocks in some other structure.
Complex mechanism to identify location of ECC.
3 Reducing ECC Overhead

ECC-Cache [IAS 2009], Memory-mapped ECC [ISCA 2009], ECC-FIFO [SC 2009]

Dirty block – Requires error correction
Clean block – Requires only error detection

DBI enables a *simpler mechanism to reduce ECC cost.*
8% reduction in overall cache area!
DBI – Other Optimizations

- Load balancing memory accesses in hybrid memory
- Better DRAM write scheduling
- Fast cache flushing
- Bulk DMA coherence

(Discussed in paper)
The Dirty-Block Index

Outline

✓ Introduction
✓ Shortcomings of Block-Oriented Organization
✓ The Dirty-Block Index (DBI)
✓ Optimizations Enabled by DBI
  • Evaluation
  • Conclusion
Evaluation Methodology

- 2.67 GHz, single issue, OoO, 128-entry instruction window
- Cache Hierarchy
  - 32 KB private L1 cache, 256 KB private L2 cache
  - 2MB/core Shared L3 cache
- DDR3-1066 DRAM
  - 1 channel, 1 rank, 8 banks, 8KB row buffer, FR-FCFS, open row policy
- SPEC CPU2006, STREAM
- Multi-core
  - 102 2-core, 259 4-core, and 120 8-core workloads
  - Multiple metrics for performance and fairness
Mechanisms

- Dynamic Insertion Policy *(Baseline)* (ISCA 2007, PACT 2008)
- DRAM-Aware Writeback *(DAWB)* (TR-HPS-2010-2 UT Austin)
- Virtual Write Queue (ISCA 2010)
- Skip Cache (PACT 2012)
- Dirty-Block Index
  - + No Optimization
  - + Aggressive Writeback
  - + Cache Lookup Bypass
  - + Both Optimizations *(DBI+Both)*

Difficult to combine
DBI achieves almost all the benefits of DAWB with significantly lower tag store contention.
System Performance

Reduced tag store contention due to DBI translates to significant performance improvement.
Other Results in Paper

• Detailed cache area analysis (with and without ECC)
• DBI power consumption analysis
• Effect of individual optimizations
• Other multi-core performance/fairness metrics
• Sensitivity to DBI parameters
• Sensitivity to cache size/replacement policy
Conclusion

• The Dirty-Block Index
  – Key Idea: DRAM-row oriented dirty-bit organization
• Enables efficient implementation of several optimizations
  – DRAM-Aware writeback, cache lookup bypass, Reducing ECC cost
  – 28% performance over baseline, 6% over best previous work
  – 8% reduction in overall cache area
• Wider applicability
  – Can be applied to other caches
  – Can be applied to other metadata (e.g., coherence)
The Dirty-Block Index

Vivek Seshadri
Abhishek Bhowmick · Onur Mutlu
Phillip B. Gibbons · Michael A. Kozuch · Todd C. Mowry
Cache Coherence

Exclusive unmodified  Shared Unmodified

Exclusive modified  Shared modified

The Dirty-Block Index
Operation of a Cache with DBI

1. Read Access
   Look up tag store

2. Writeback
   Update tag store. Update DBI to indicate the block is dirty.

3. Cache Eviction
   Check DBI. Write back if block is dirty

4. DBI Eviction
   Write back all blocks marked dirty by the entry
**DBI Design Parameters**

**DBI Granularity** ($g$)
Number of blocks tracked by each entry

**DBI Size** ($\alpha$)
Total number of blocks tracked by the DBI
Represented as a fraction of number of blocks in cache
DBI Design Parameters – Example

Cache tracks 16384 blocks

1MB Cache
64B Blocks

DBI tracks 4096 blocks
Each entry tracks 64 blocks
DBI has 64 entries

DBI
$\alpha = \frac{1}{4}$
$g = 64$
Effect on Writes and Tag Lookups

- Memory Writes
- Write Row Hits
- Tag Lookups

Baseline, DAWB, DBI, +AWB, +CLB, +Both
The Dirty-Block Index

System Performance

- **Baseline**
- **DAWB**
- **DBI**
- **+AWB**
- **+CLB**
- **+Both**

System Performance

1-Core 2-Core 4-Core 8-Core

The Dirty-Block Index