The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Ghostwriter: Cache Coherence for Error-Tolerant Applications

Henry Kao **Joshua San Miguel** **Natalie Enright Jerger**

Overview

- Cache coherence is used to keep data coherent in multicores
- Communication of data is expensive within processors
- Limited power budget makes us look to new processor paradigms – Approximate Computing

Contribution: Ghostwriter Coherence Protocol

- Approximate stores instr. and approximate coherence states
- Up to 50.1% savings in dynamic energy
- Up to 37.3% speedup

Cache Coherence

- Mechanism to keep data coherent between private caches in a multiprocessor

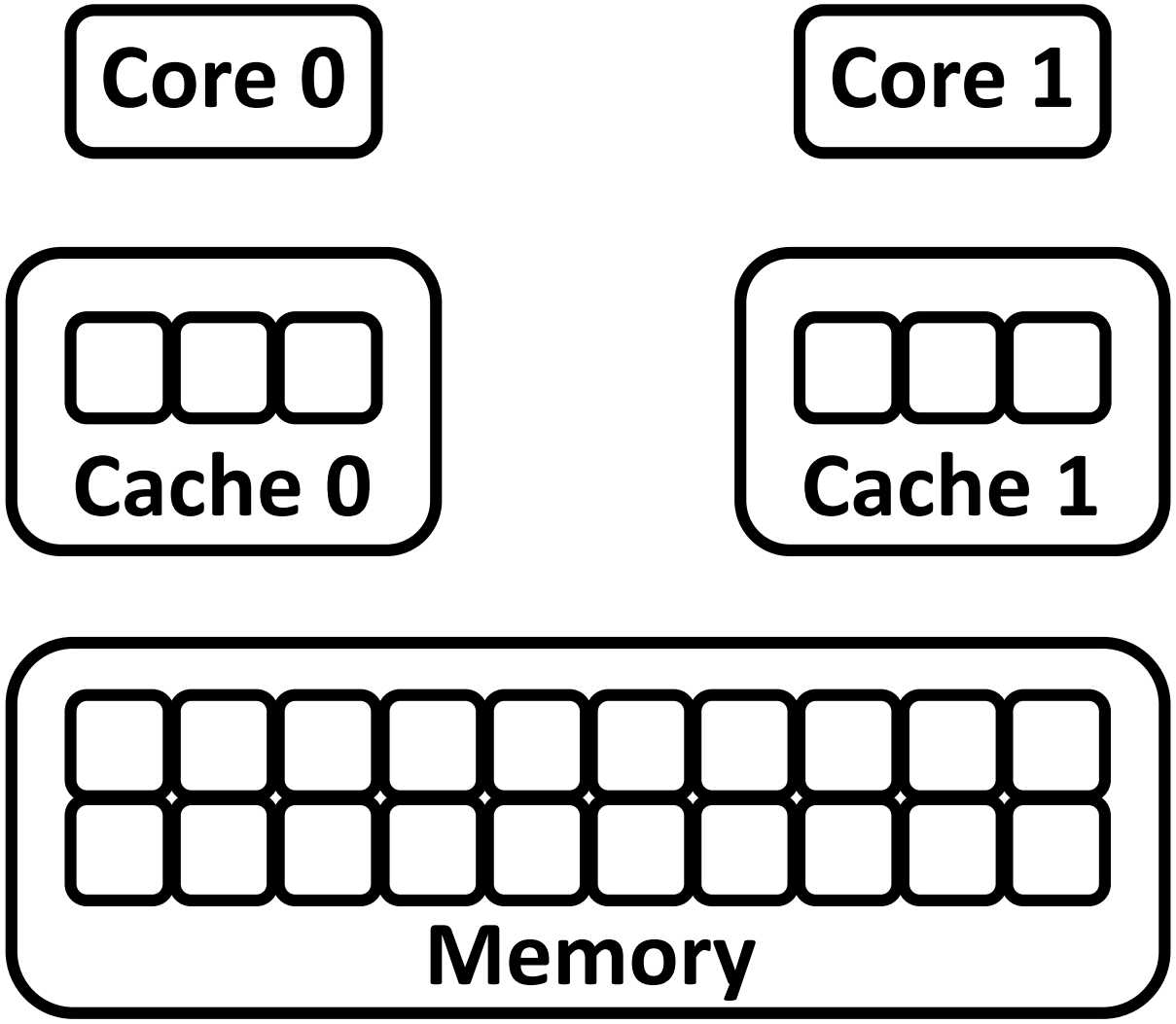
Single-Writer Multiple-Reader Invariant

- Each data can have multiple readers, but only a single writer

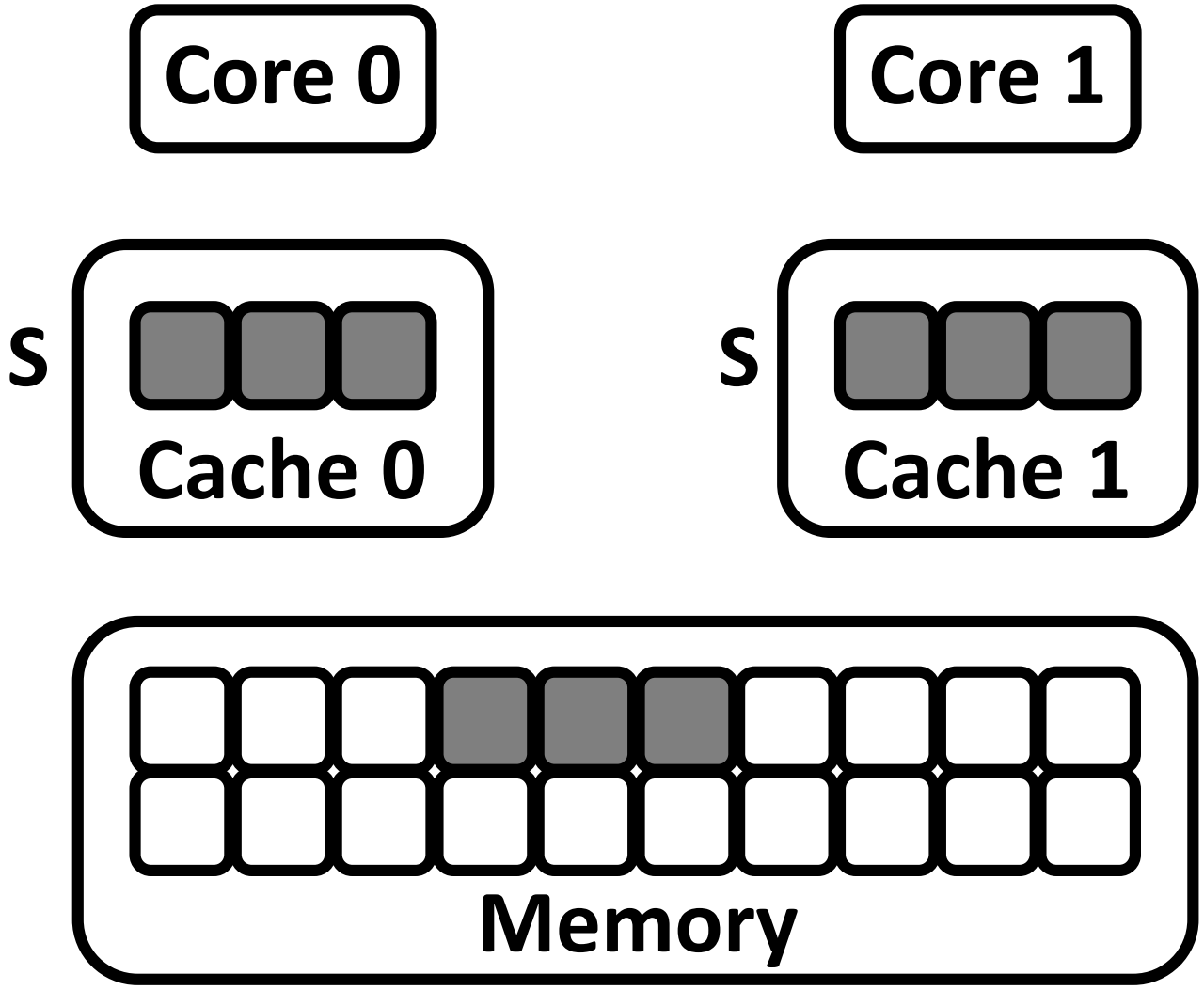
Data-Value Invariant

- The start of every read must have data from the previous write

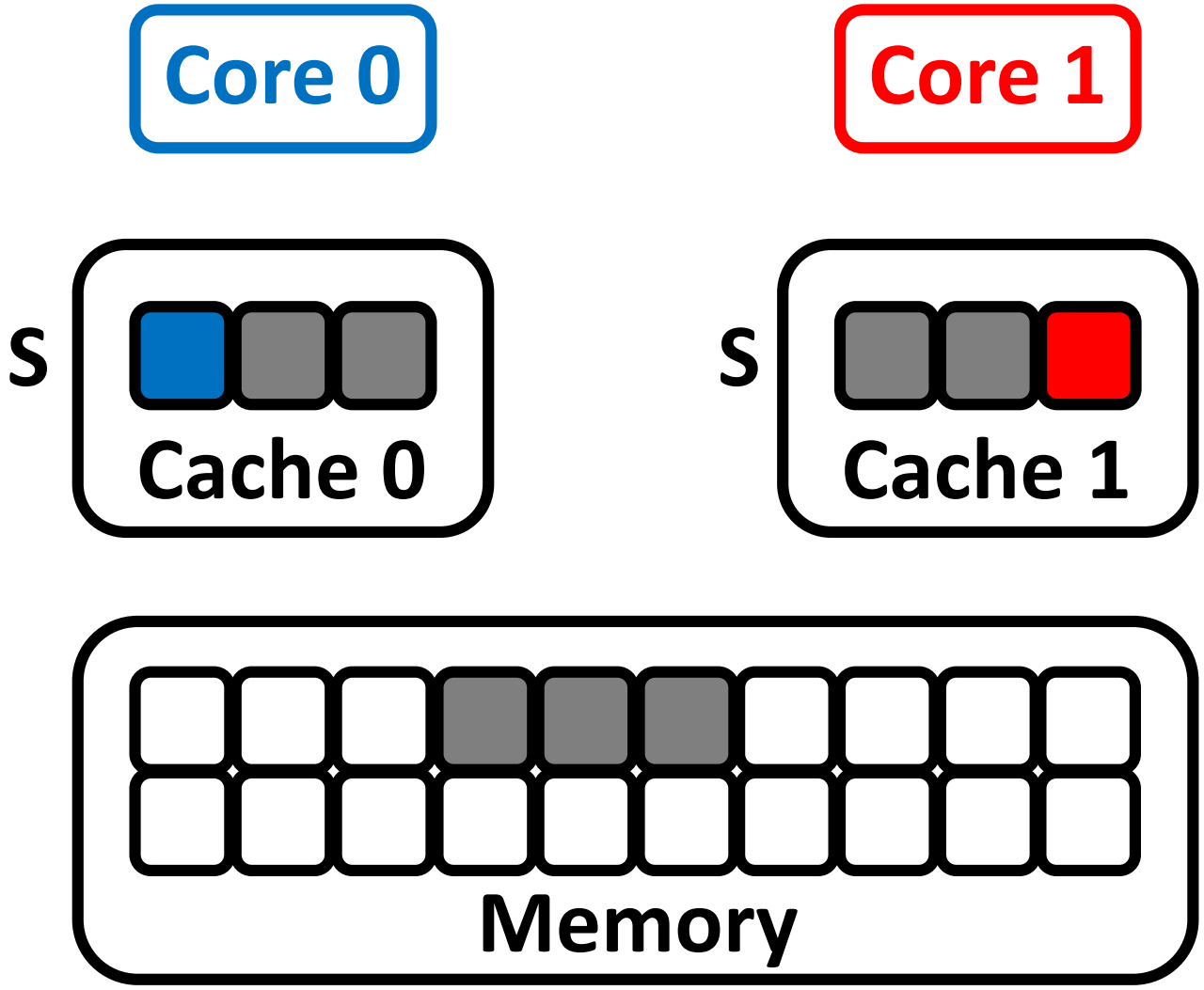
False Sharing is Bad



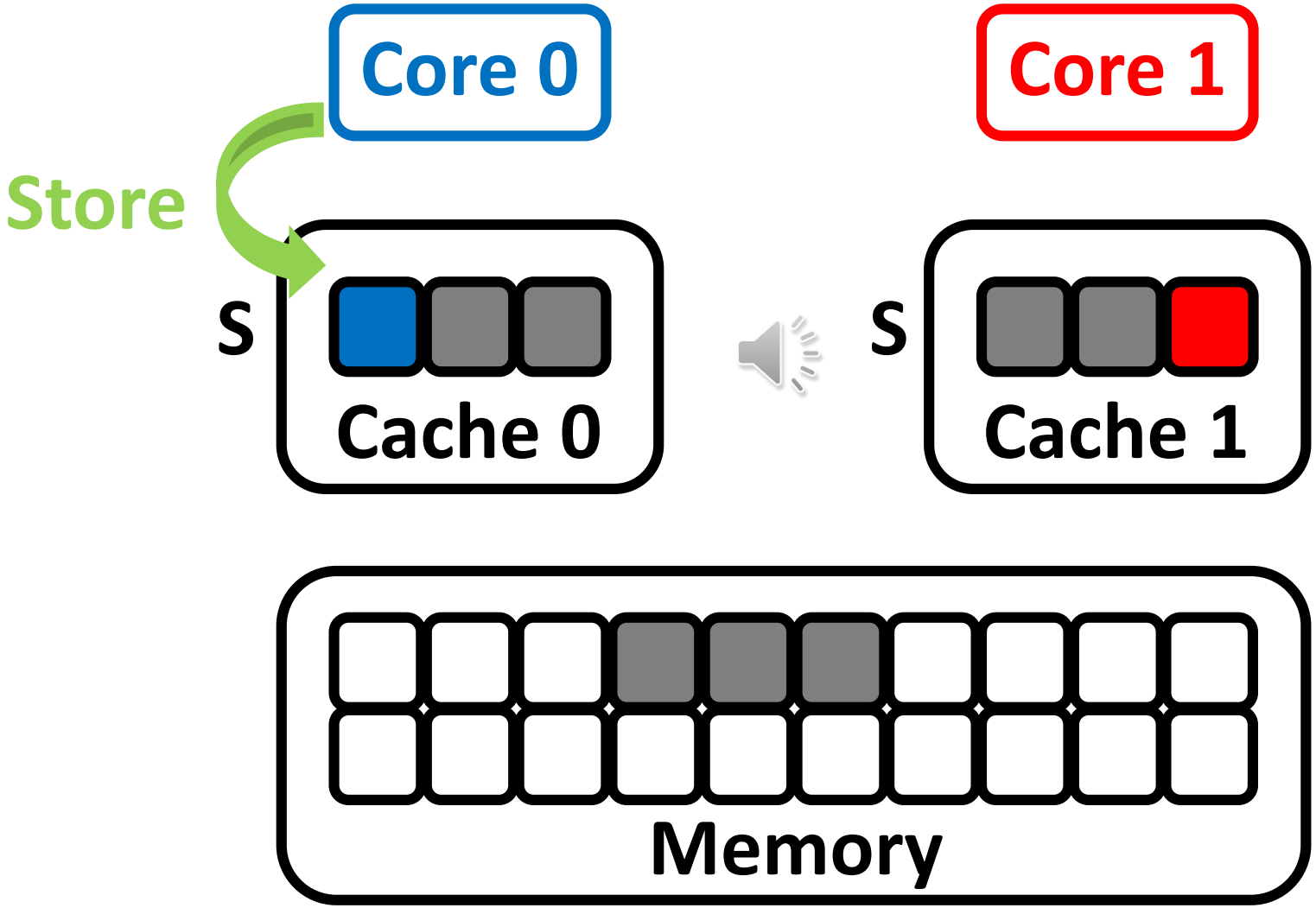
False Sharing is Bad



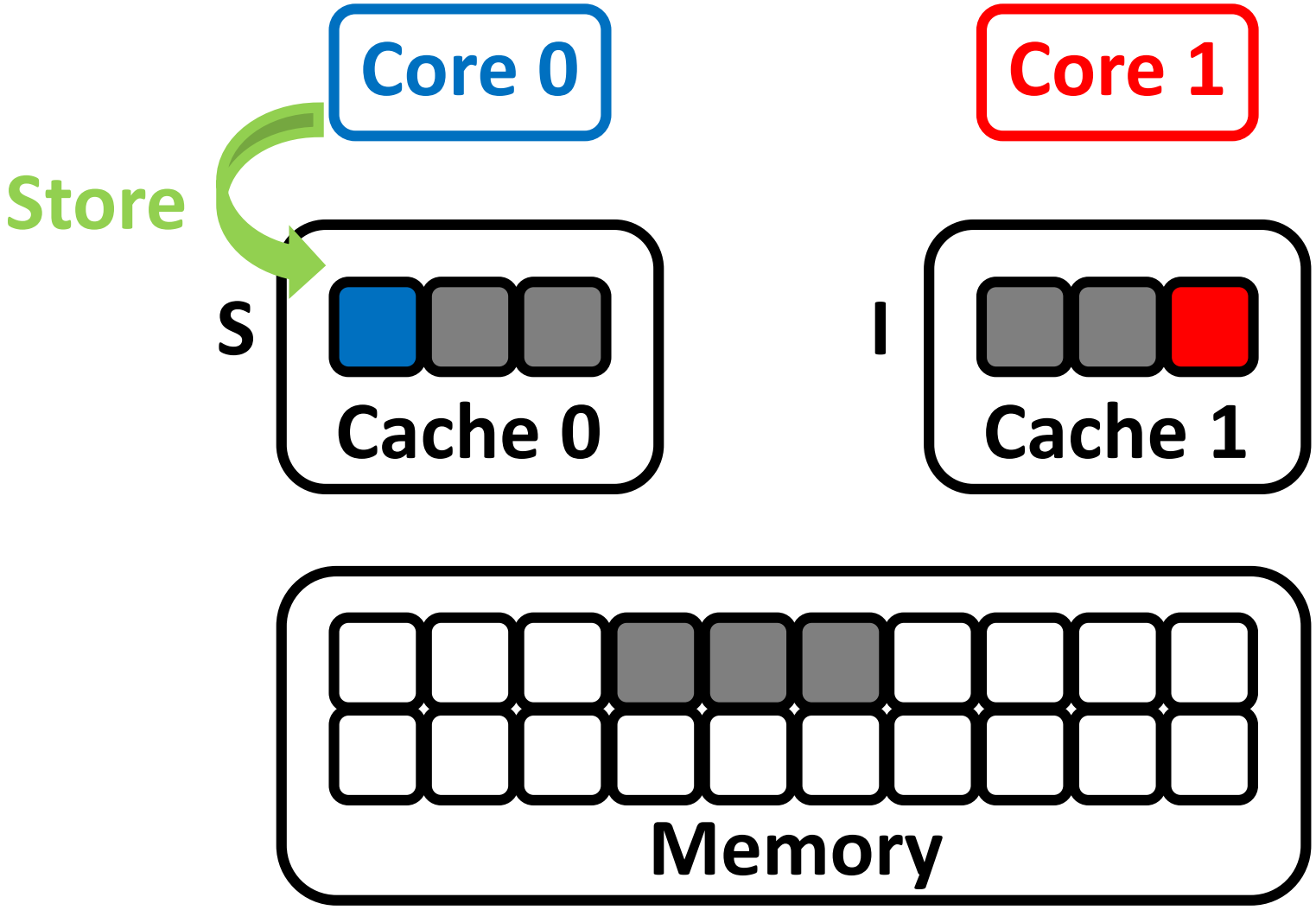
False Sharing is Bad



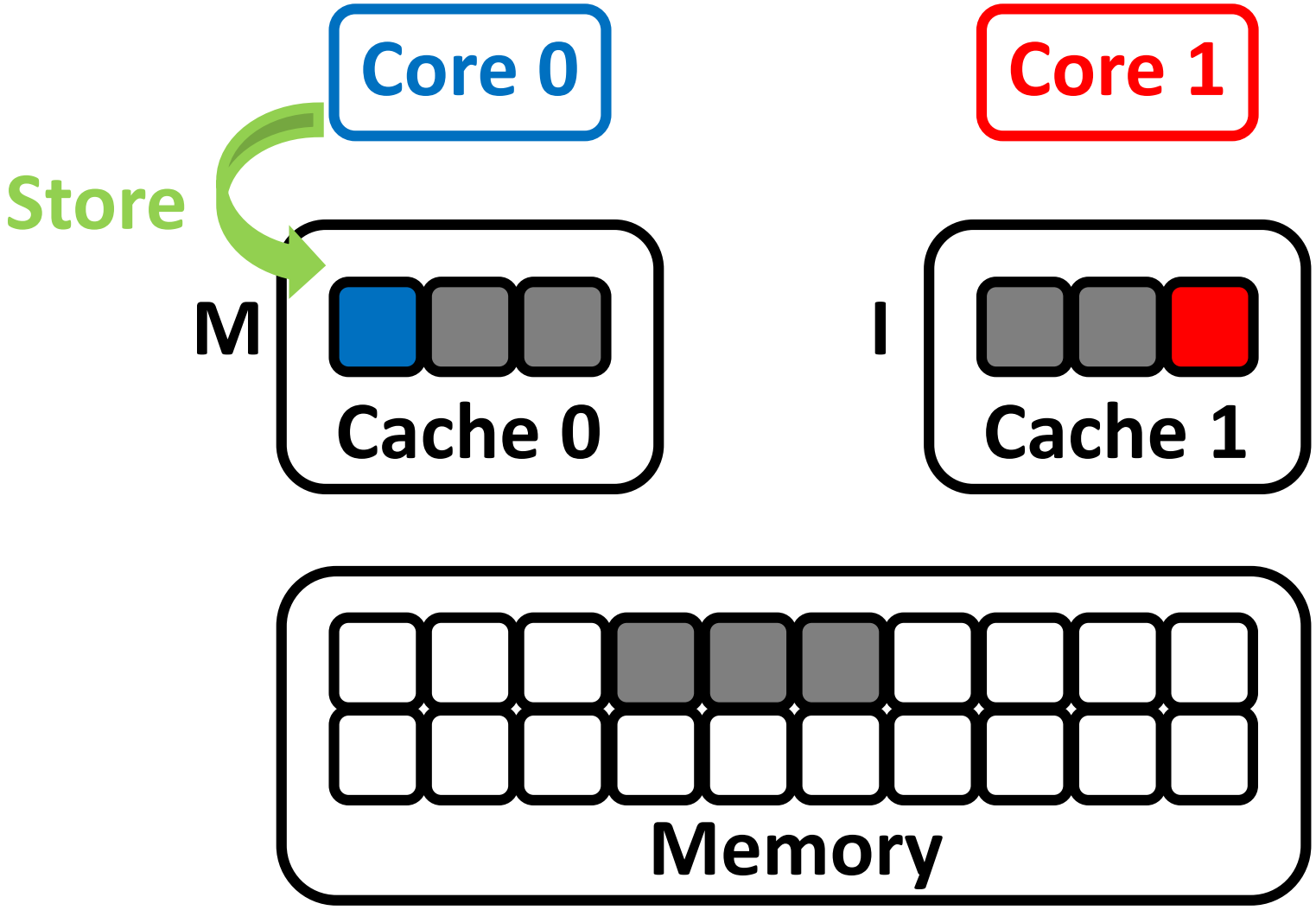
False Sharing is Bad



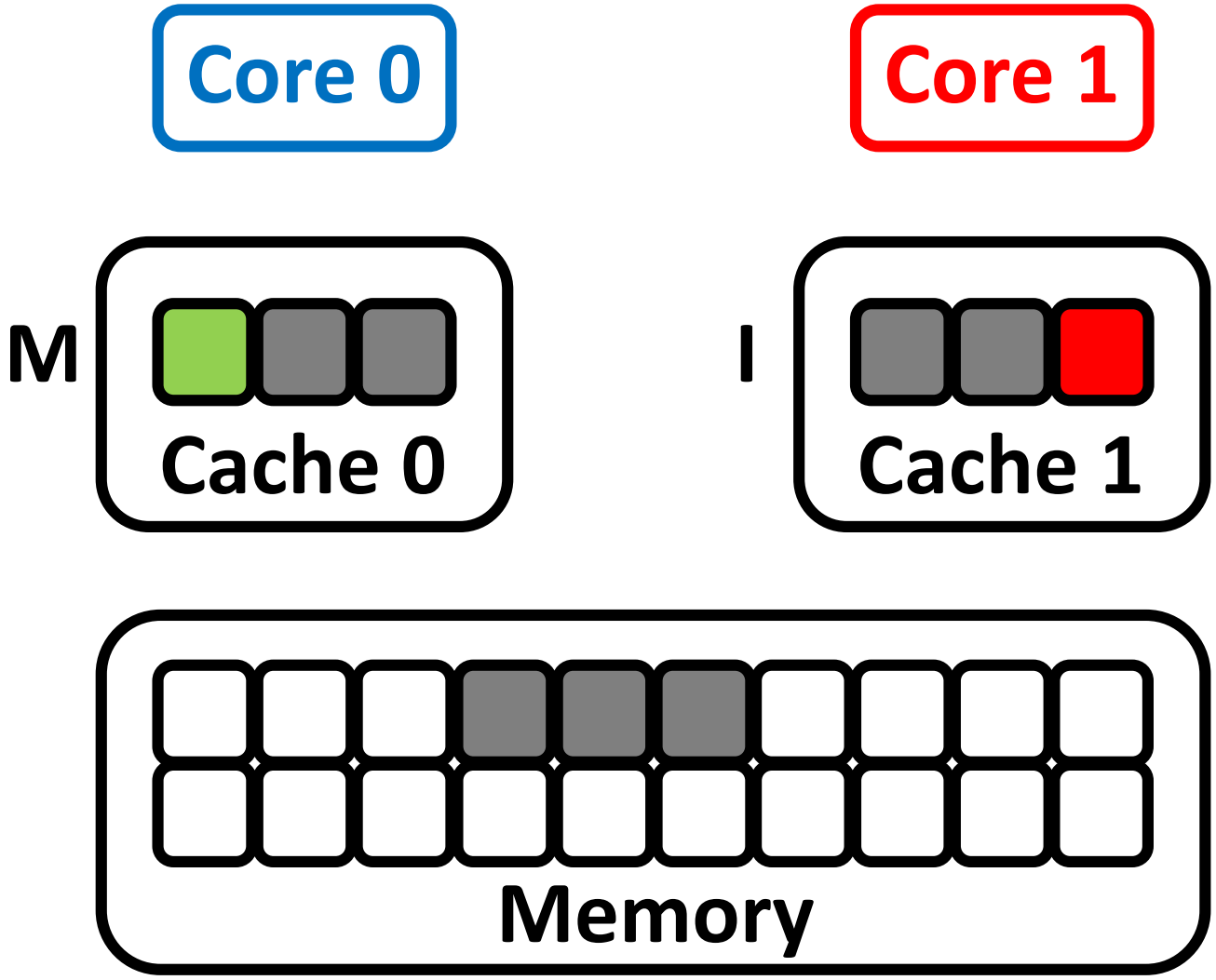
False Sharing is Bad



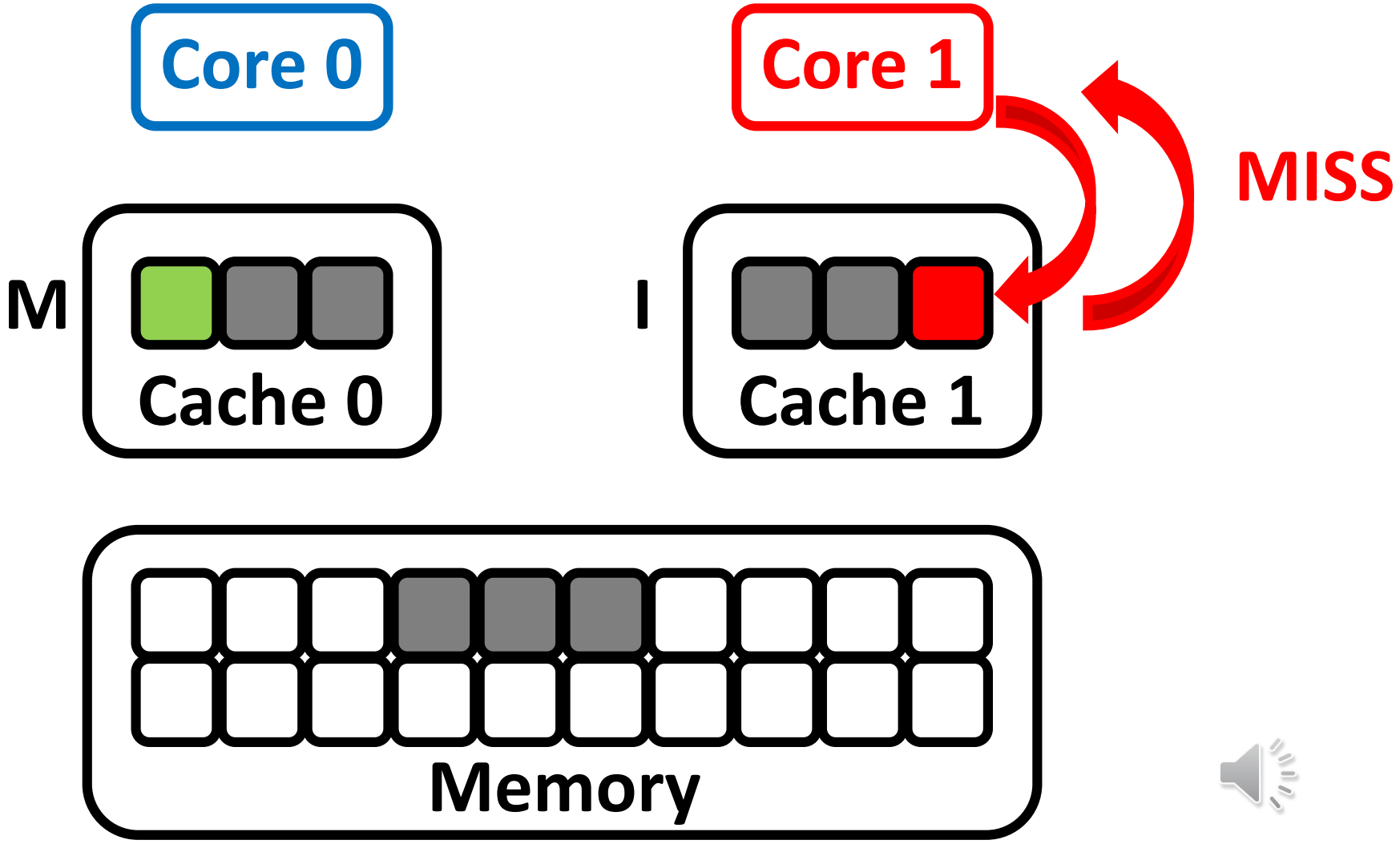
False Sharing is Bad



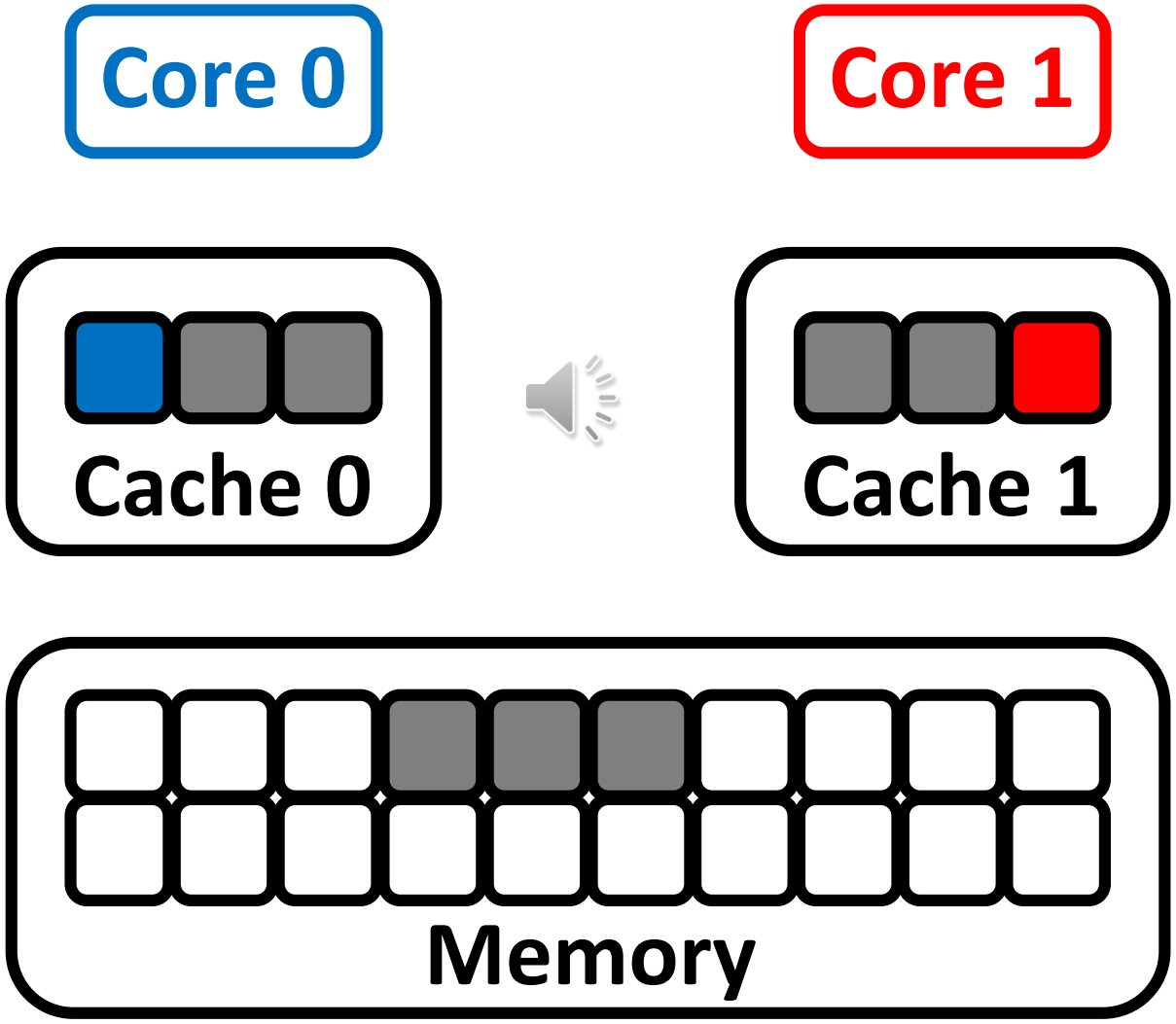
False Sharing is Bad



False Sharing is Bad



False Sharing is Bad



Communication is Expensive

Energy Consumption [Keckler, Micro'11]

- L1 – 1.14x
- L2/L3 – 6.2x
- DRAM – 200x

Latency [Intel Xeon 5500]

- L1 – 4 cycles
- L2 – 10 cycles
- L3 – 75-300 cycles
- DRAM – 600-1000 cycles

Communication is Expensive

Energy Consumption [Keckler, Micro'11]

- L1 – 1.14x
- L2/L3 – 6.2x
- DRAM – 200x

Latency [Intel Xeon 5500]

- L1 – 4 cycles
- L2 – 10 cycles
- L3 – 75-300 cycles
- DRAM – 600-1000 cycles

Keep data in L1

Communication is Expensive

Energy Consumption [Keckler, Micro'11]

- L1 – 1.14x
- L2/L3 – 6.2x
- DRAM – 200x

Latency [Intel Xeon 5500]

- L1 – 4 cycles
- L2 – 10 cycles
- L3 – 75-300 cycles
- DRAM – 600-1000 cycles

Keep data in L1

Cold Misses

Capacity Misses

Conflict Misses

Communication is Expensive

Energy Consumption [Keckler, Micro'11]

- L1 – 1.14x
- L2/L3 – 6.2x
- DRAM – 200x

Latency [Intel Xeon 5500]

- L1 – 4 cycles
- L2 – 10 cycles
- L3 – 75-300 cycles
- DRAM – 600-1000 cycles

Keep data in L1

Cold Misses

Capacity Misses

Conflict Misses

COHERENCE MISSES

Related Work

False Sharing Detection and Repair

- Detecting false sharing at software level

  [T. Lui et al., PPoPP'14], [T. Lui , E. D. Berger, OOPSLA'14], [L. Lou et al., HPCA'16]

Silent Stores

- Terminate stores that stores exact same value

  [K. M. Lepak, M. H. Lipasti, ISCA'00], [K. M. Lepak, M. H. Lipasti, Micro'00], [K. M. Lepak, M. H. Lipasti, ASPLOS'02]

Coherence Optimizations

- Varying granularity to minimize false sharing

  [J. Huh et al. ASPLOS'04], [H. Zhao et al. ISCA'13], [S. Kumar ISCA'12]

Approximate Computing

- For error-tolerant applications
- Tradeoff output accuracy for better performance/energy

- Applied across the stack

Software examples

- Approximate Kernels, Code Perforation, etc…

Hardware examples

- Load-Value Approximation, Voltage Scaling, etc…

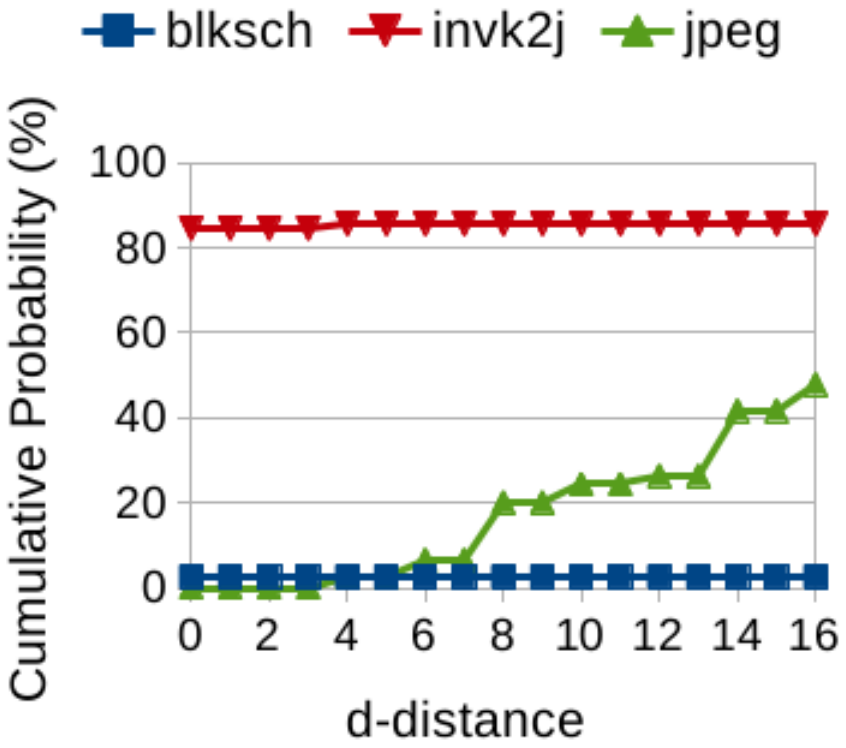
Value Similarity in Approximate Applications

- Occurrence of similar values within an application
- d-distance: allow ‘d’ lower bits to be different

Example using 1-distance:

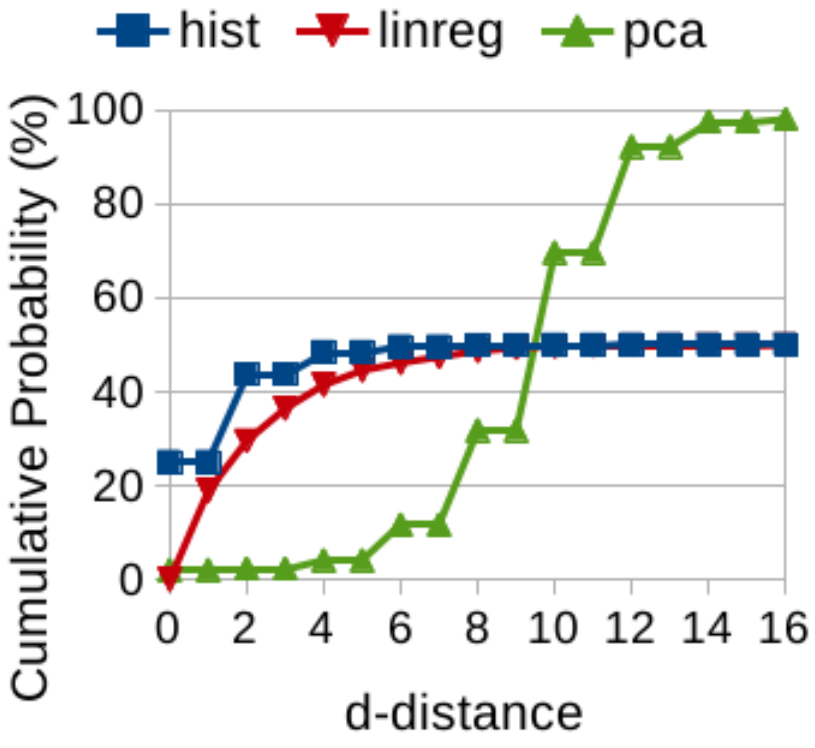
- 0b111**0** (14) and 0b111**1** (15) are similar
- 0b11**01** (13) and 0b11**11** (15) are **NOT** similar

Value Similarity in Approximate Applications



AxBench

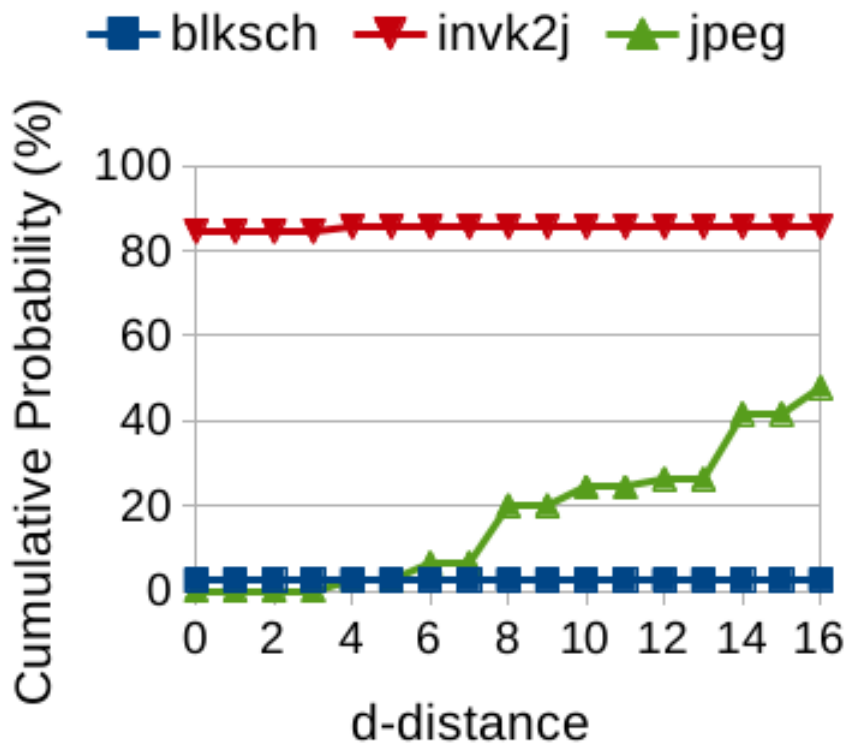
[A. Yazdanbakhsh, IEEE Design & Test ’16]

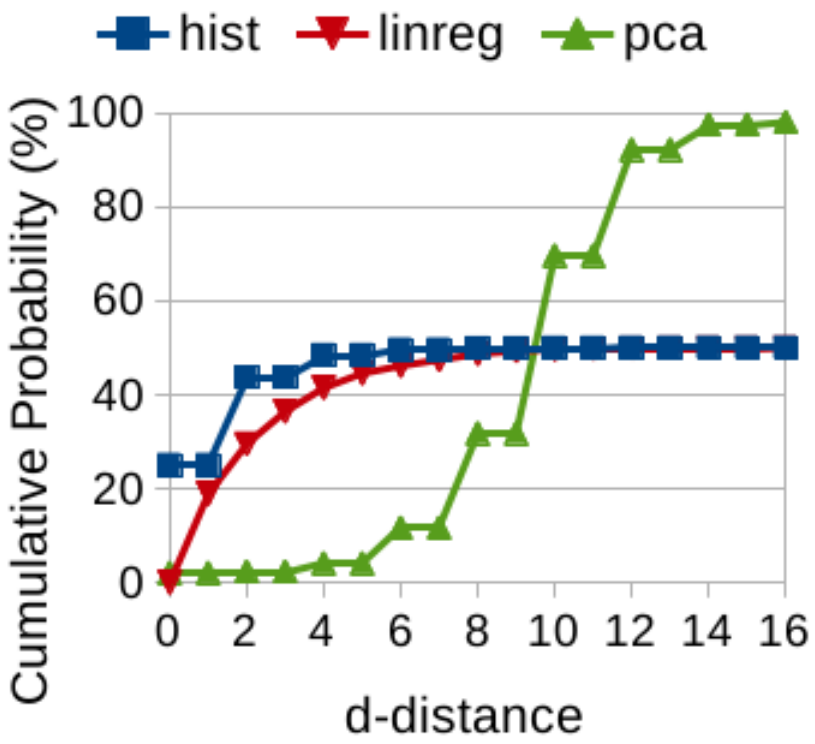


Phoenix

[C. Ranger, HPCA’07]

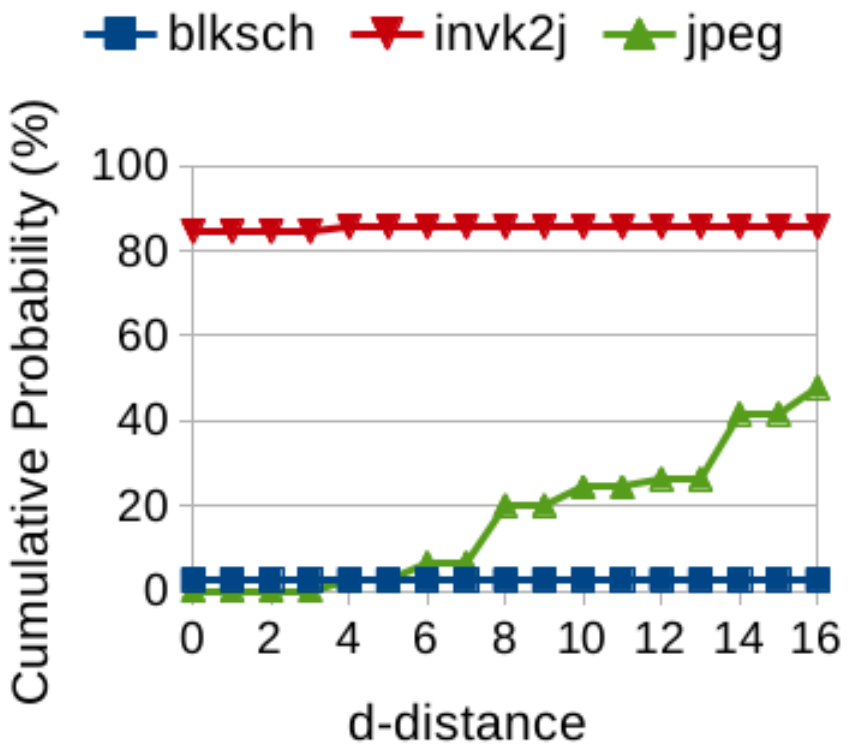
Value Similarity in Approximate Applications

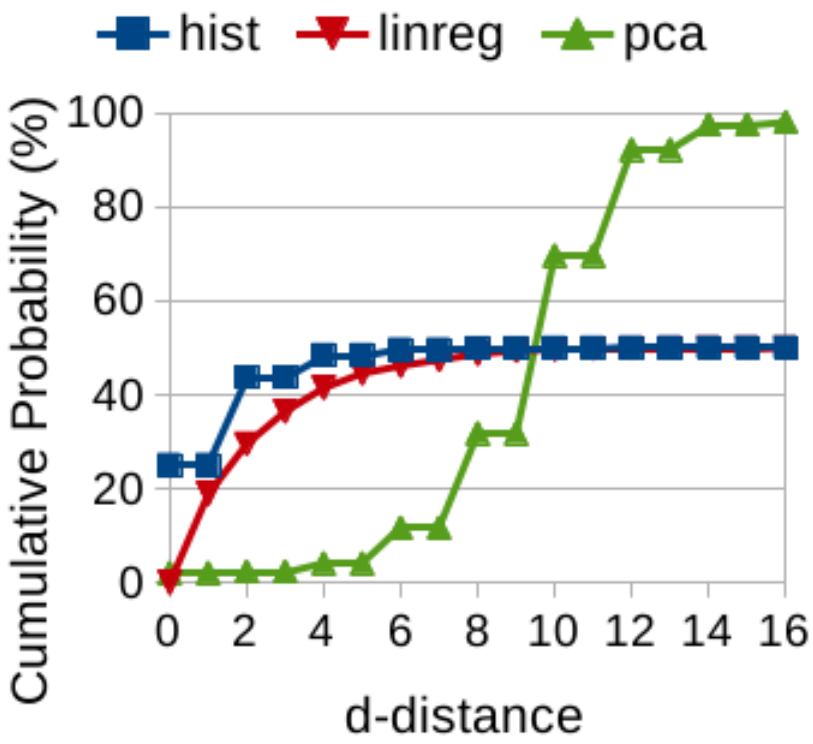




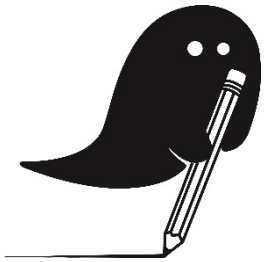
On average, 22.8% of store values have 0-distance

Value Similarity in Approximate Applications





Exploit value similarity to mitigate coherence misses



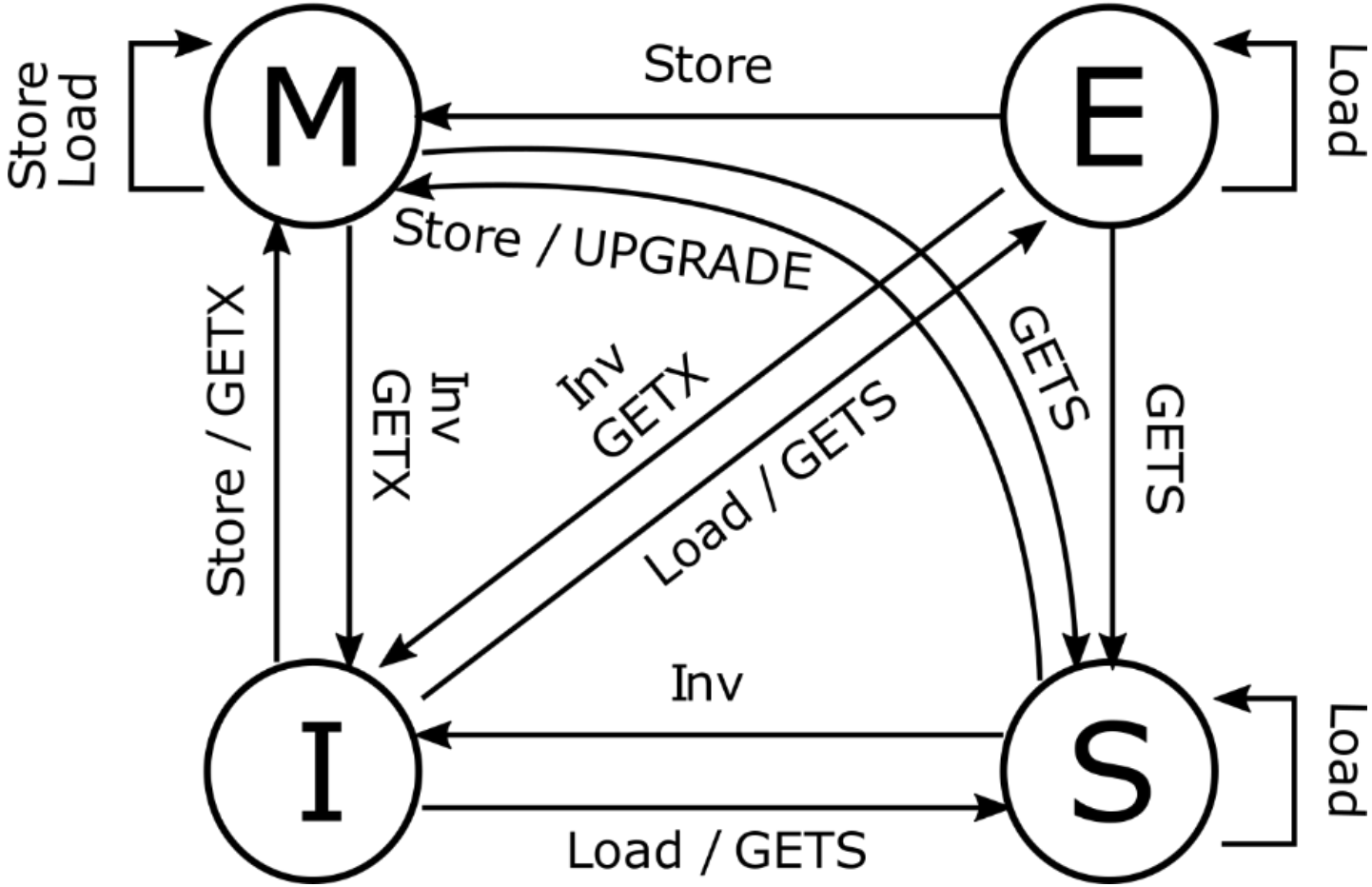
Ghostwriter

Ghostwriter

- Cache Coherence Protocol for Error-Tolerant Applications
- Exploits inherent value-similarity to mitigate false sharing

Baseline MESI Protocol

- Stores miss on I and S
- Store misses send GETX and UPGRADE requests
- Invalidates other copies causing misses on subsequent accesses

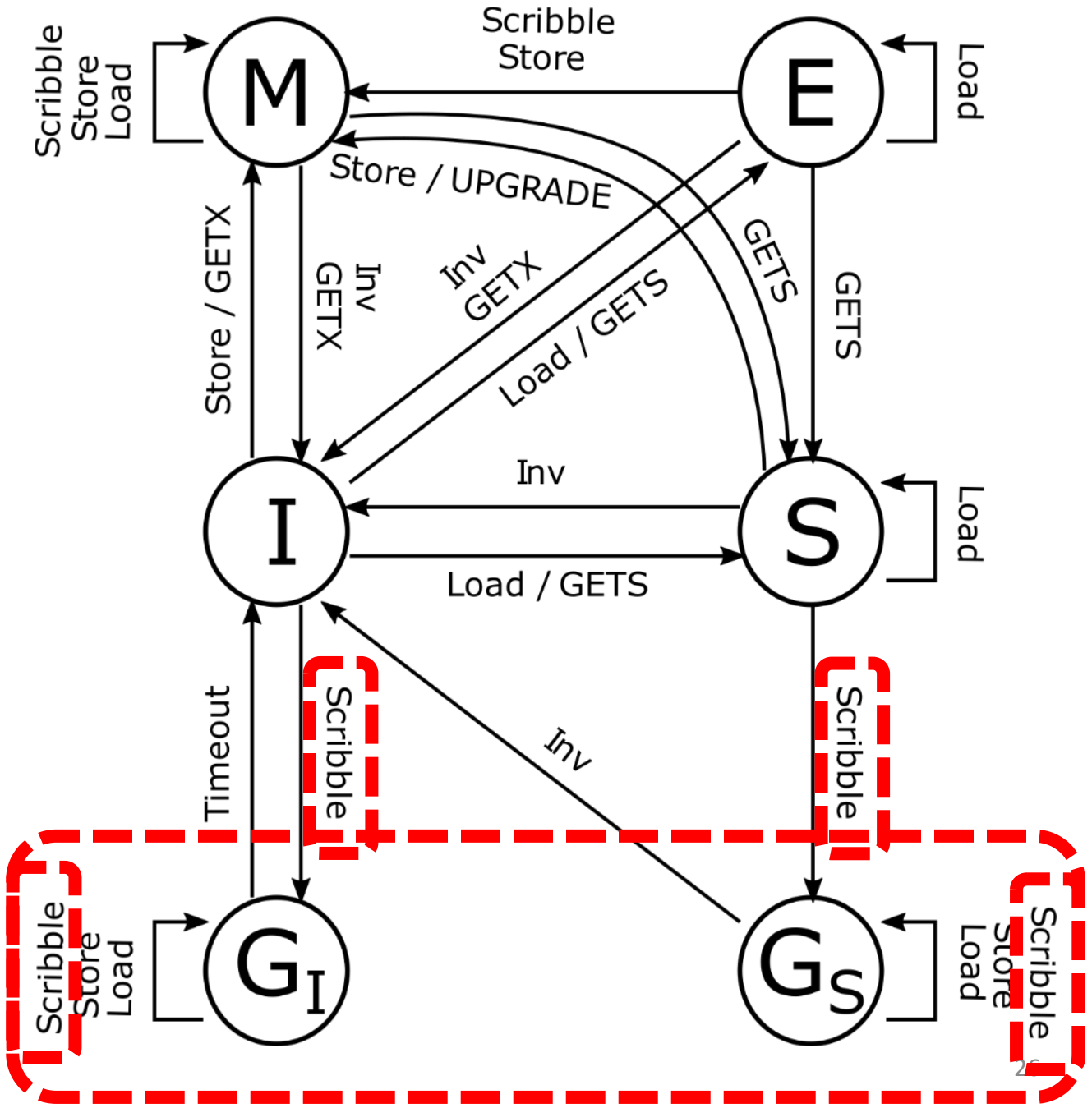


Ghostwriter Protocol

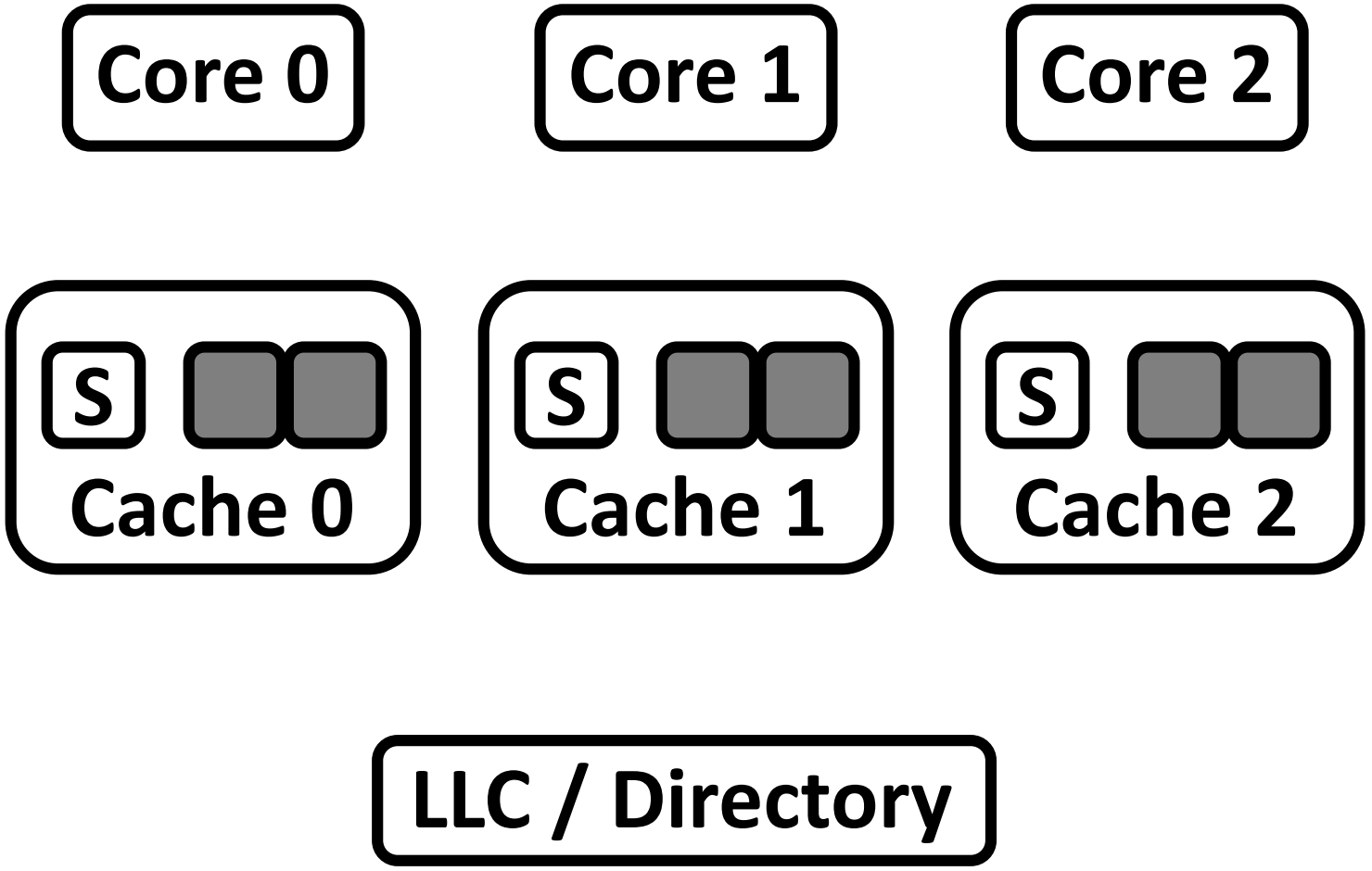
- Two approximate states
  - G_I
  - G_S

Allow stores on approx. data in states I and S

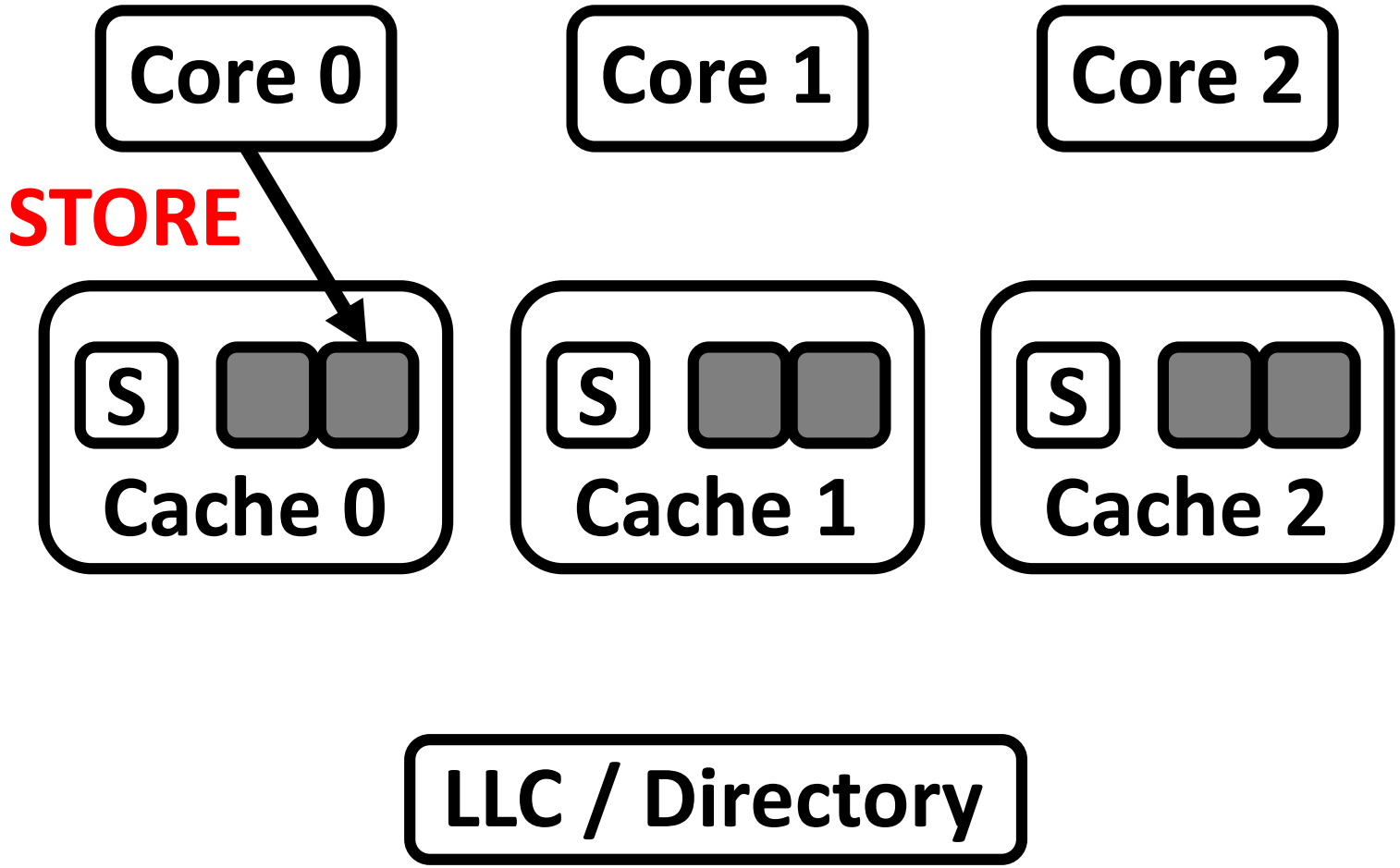
- Approximate Store instr.
  - **Scribble**
  - Programmer Annotations



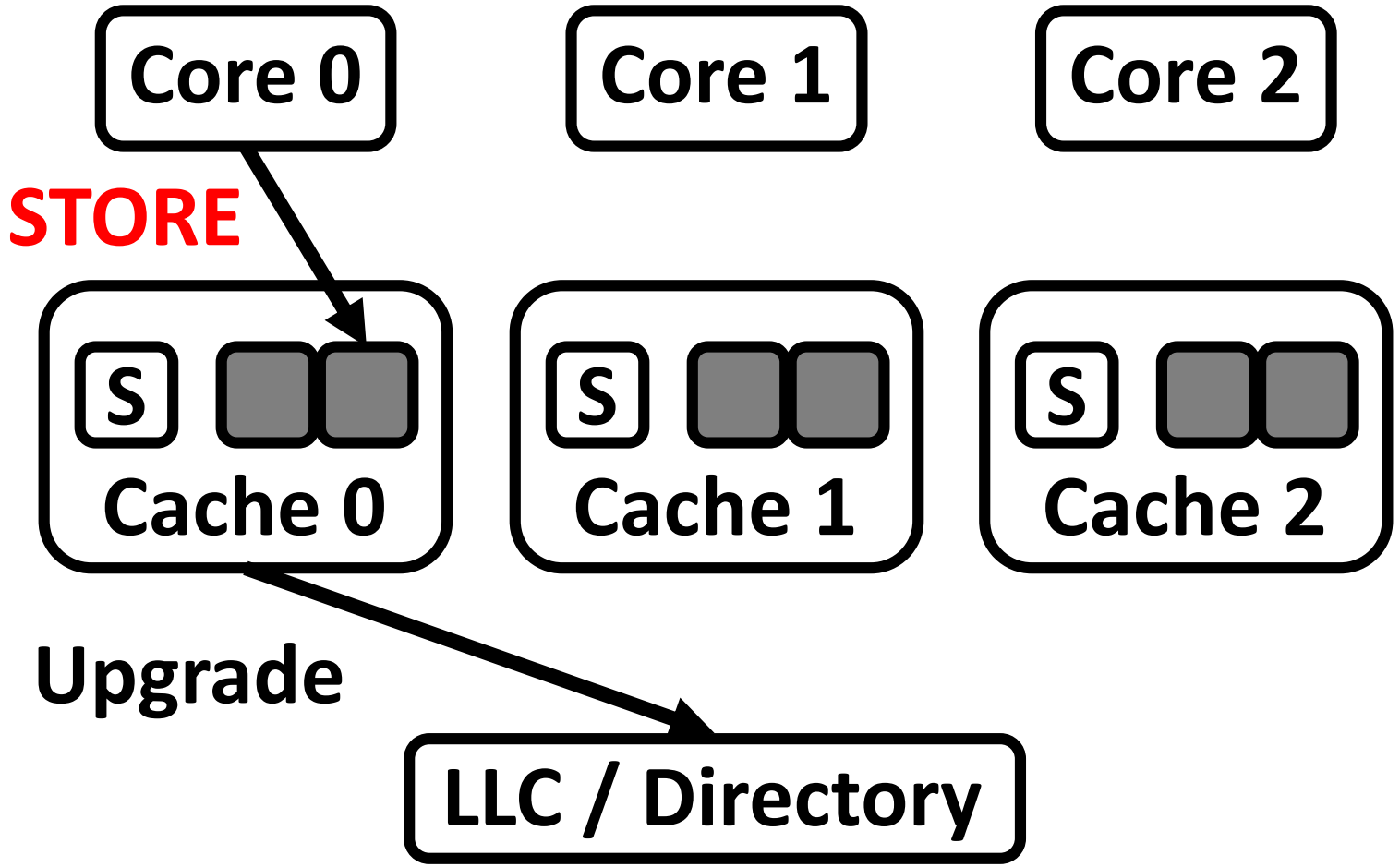
Example: Migratory Sharing



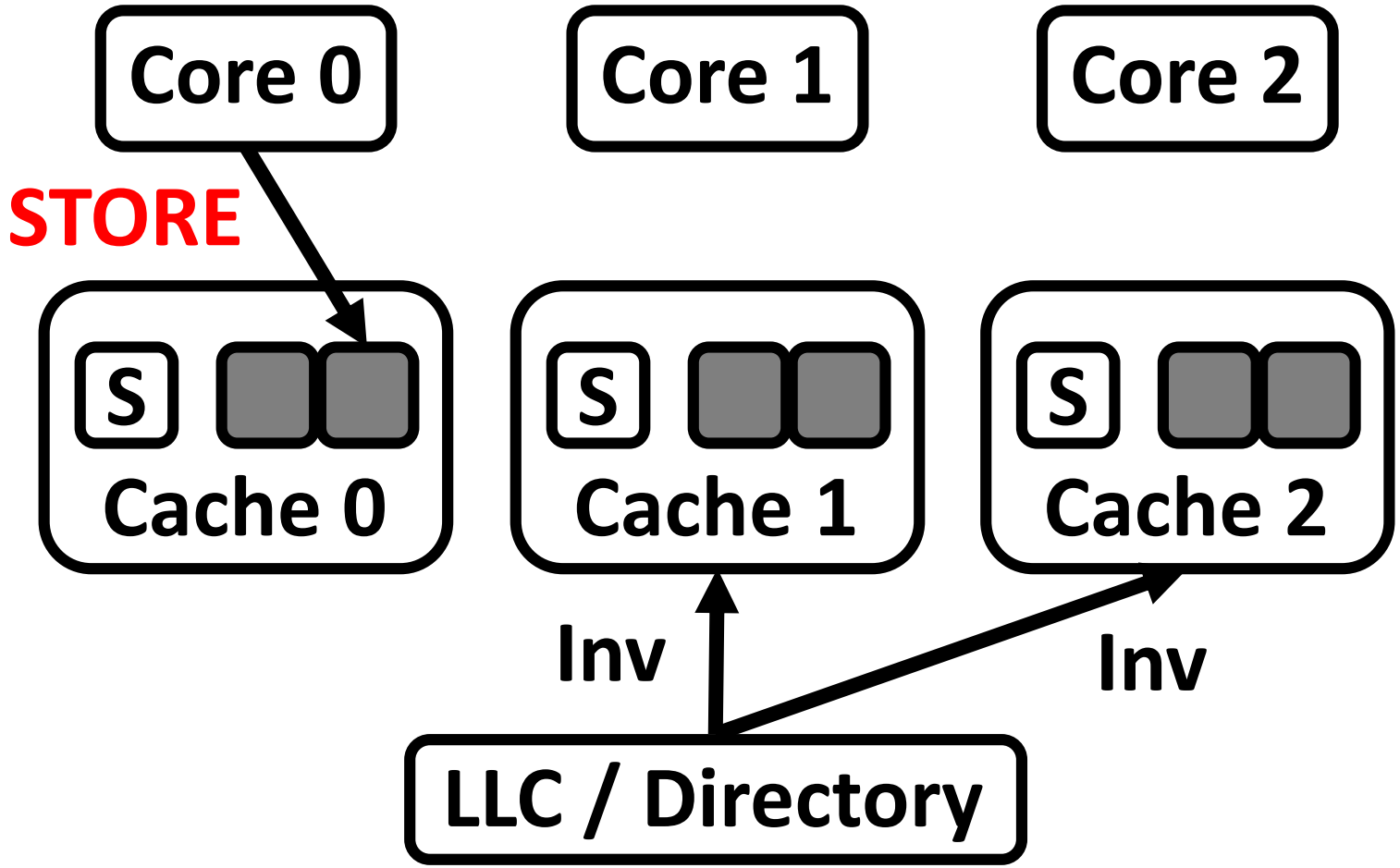
Example: Migratory Sharing



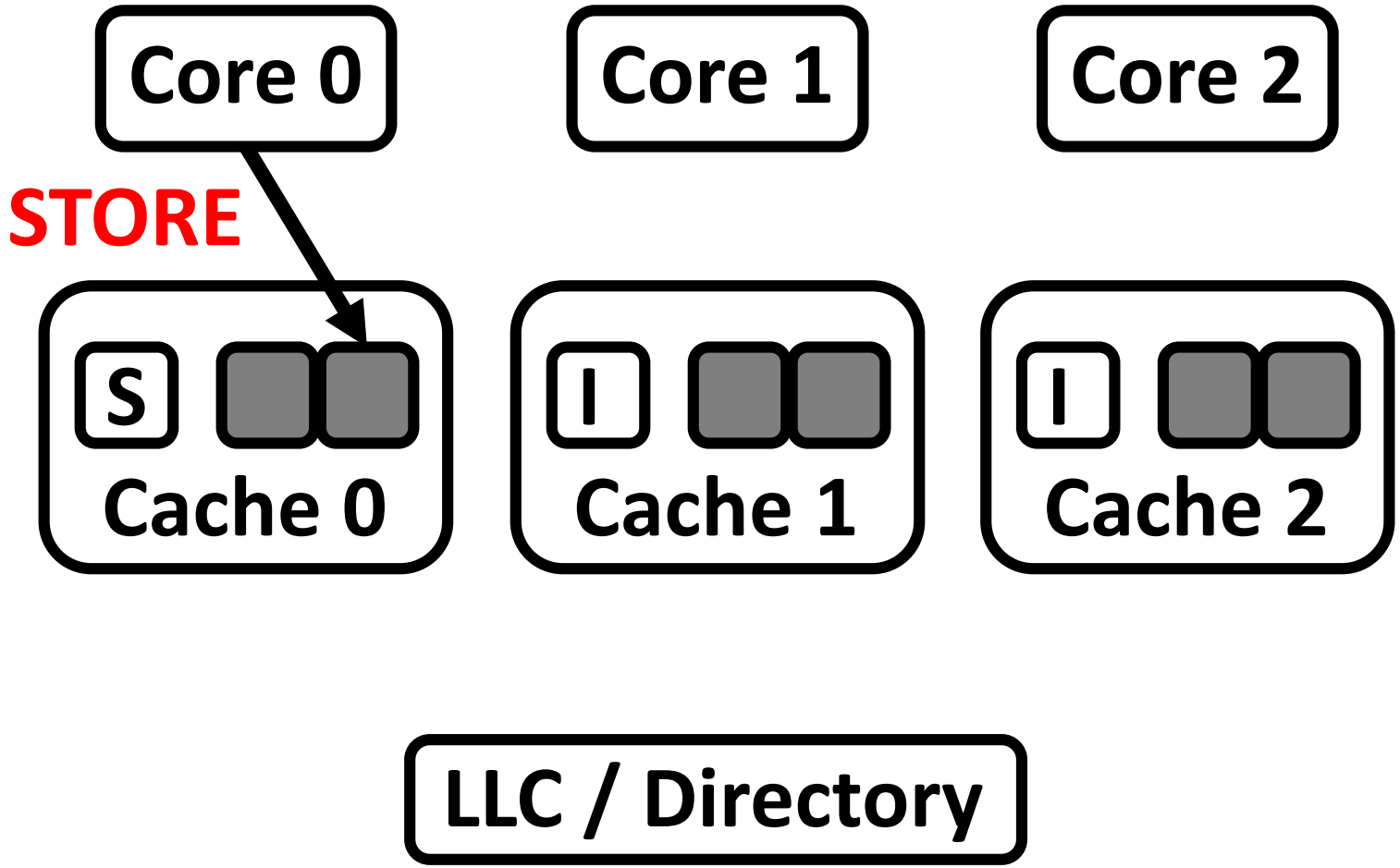
Example: Migratory Sharing



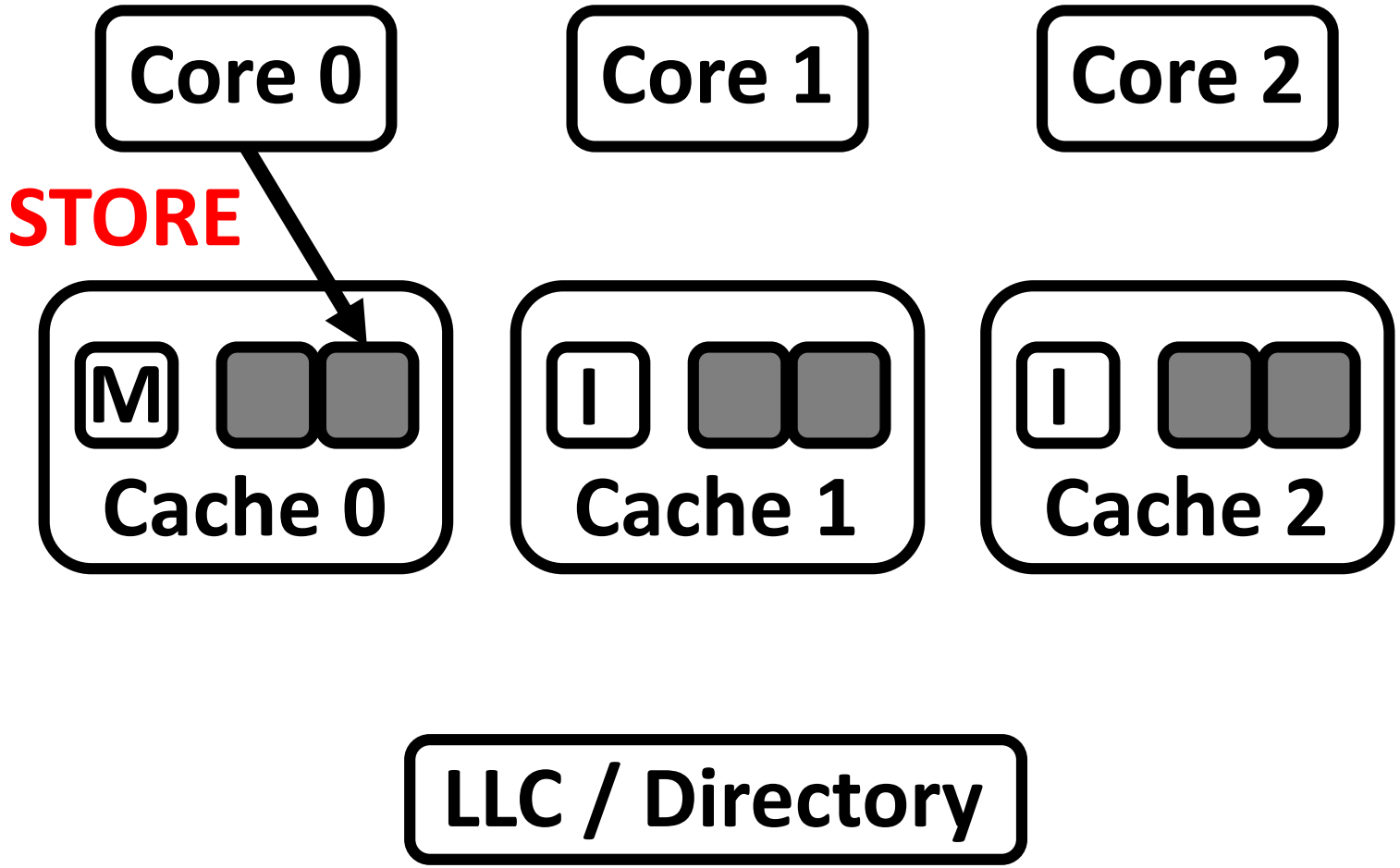
Example: Migratory Sharing



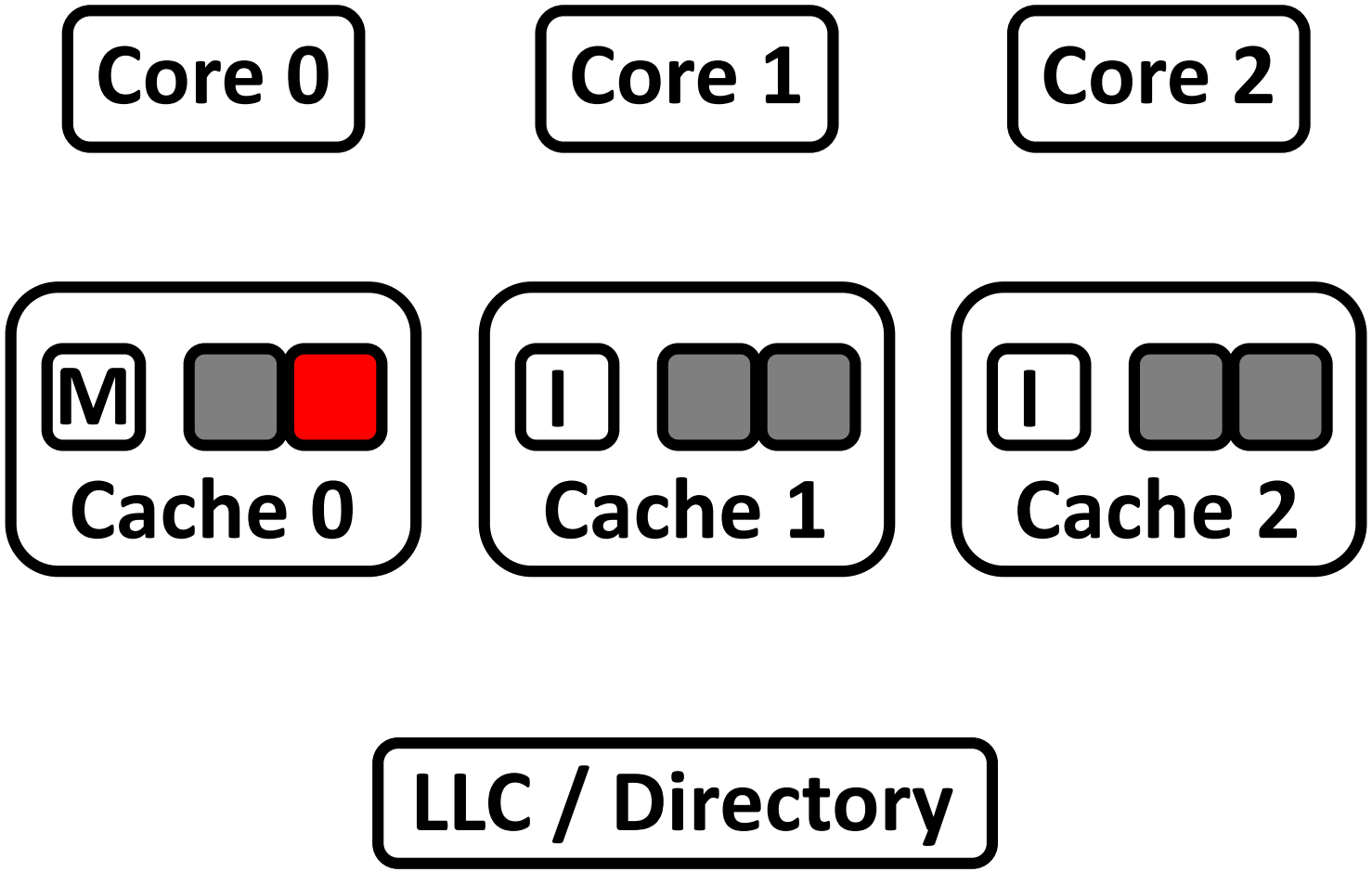
Example: Migratory Sharing



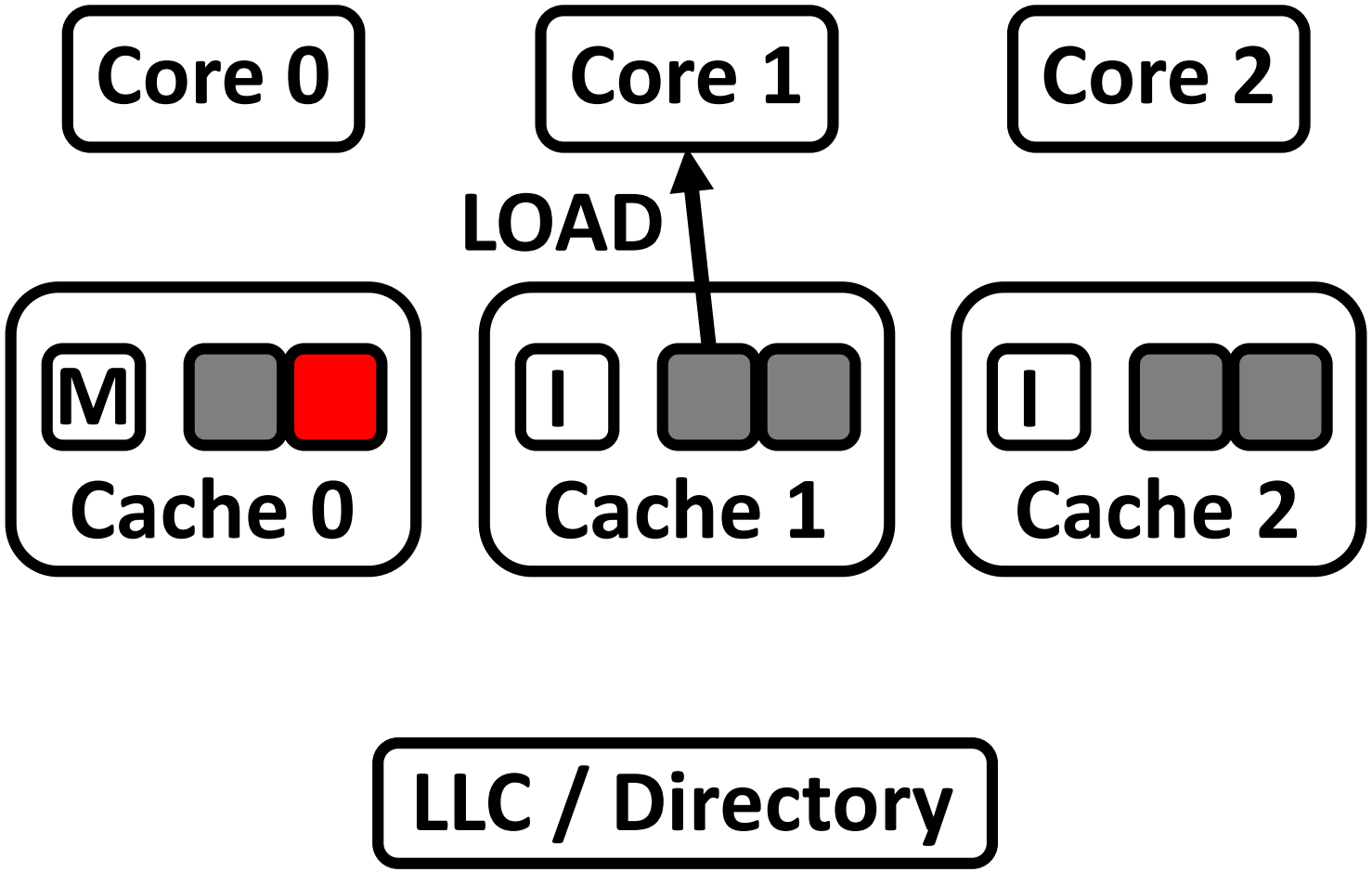
Example: Migratory Sharing



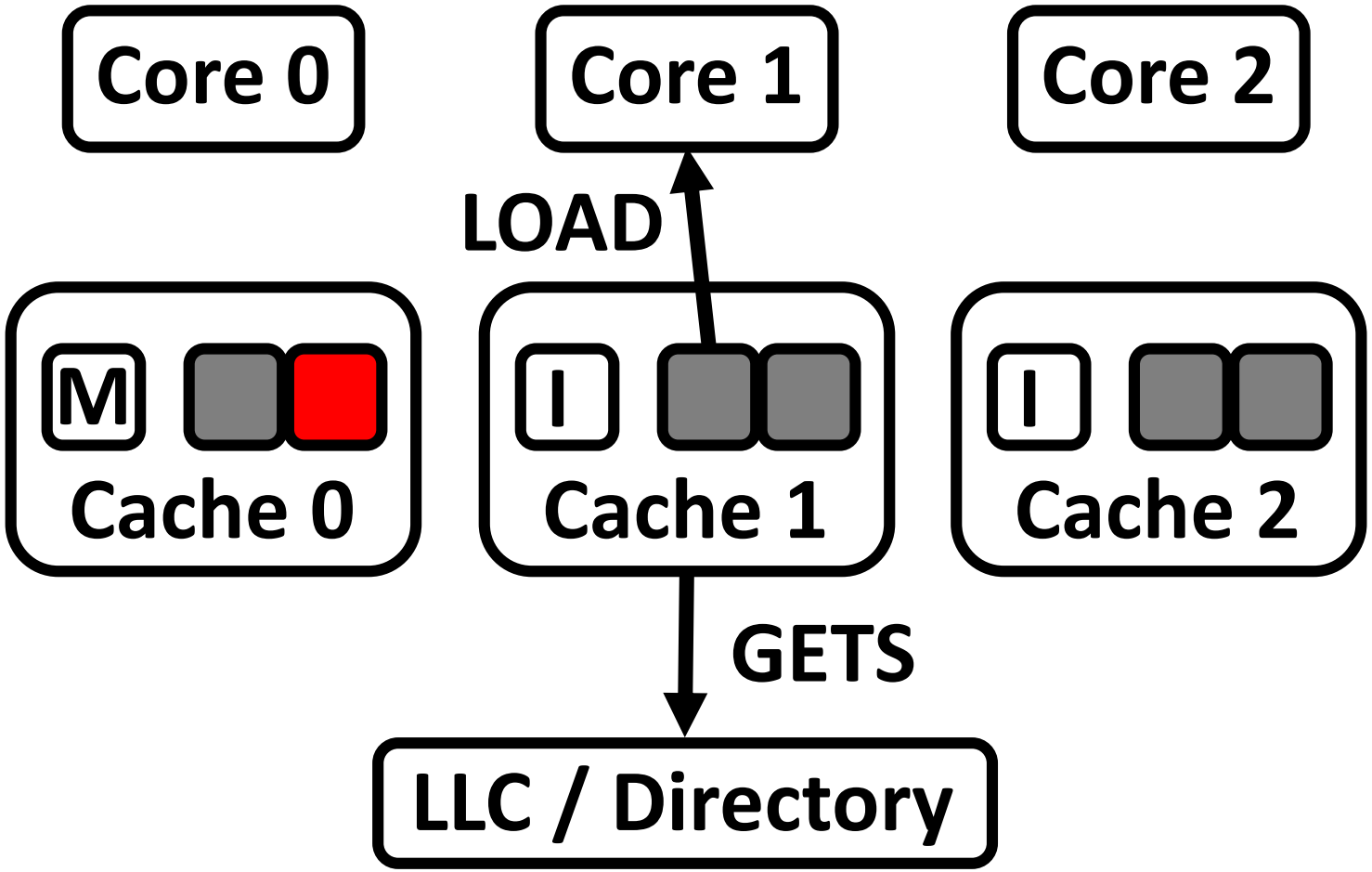
Example: Migratory Sharing



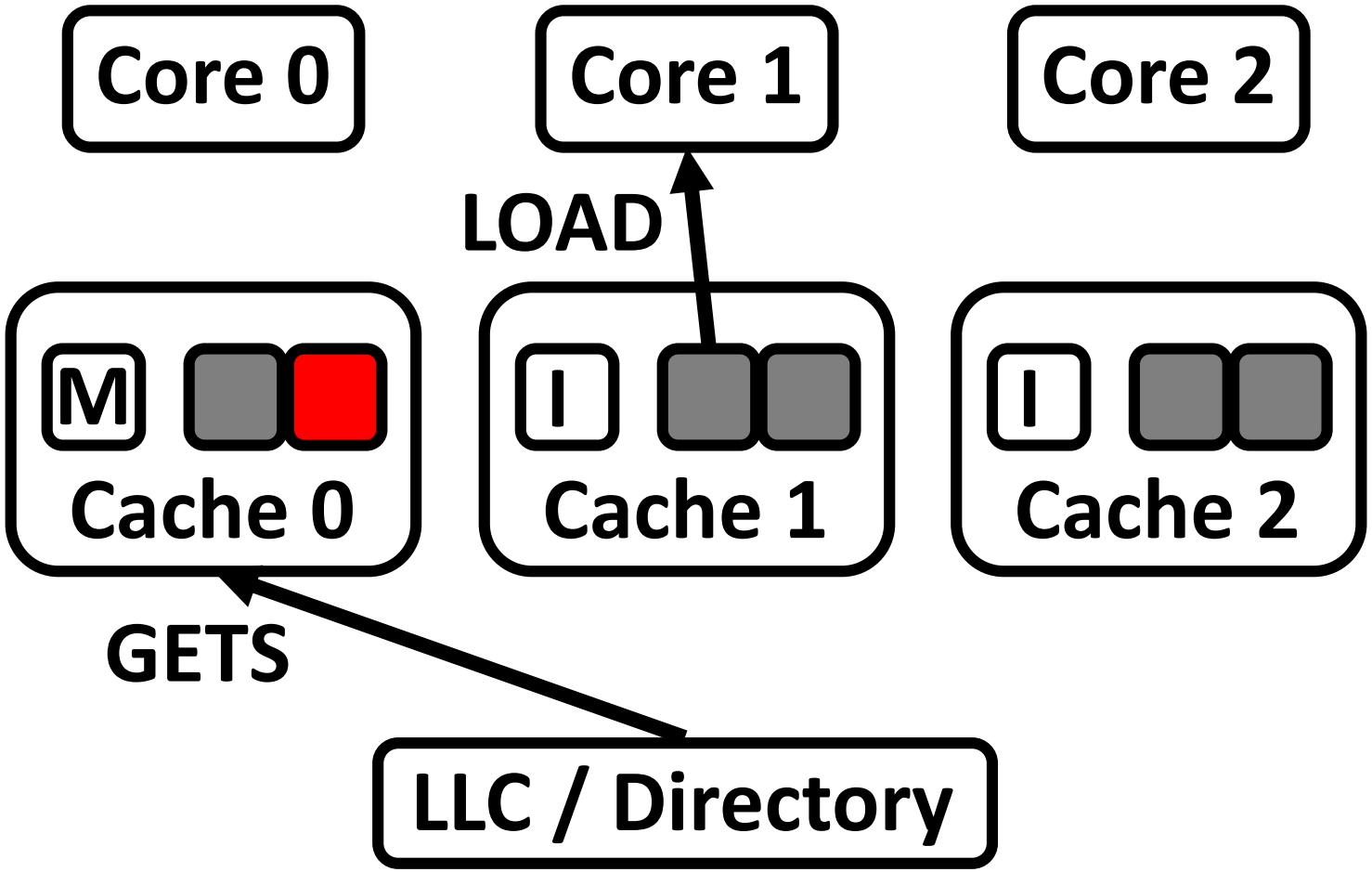
Example: Migratory Sharing



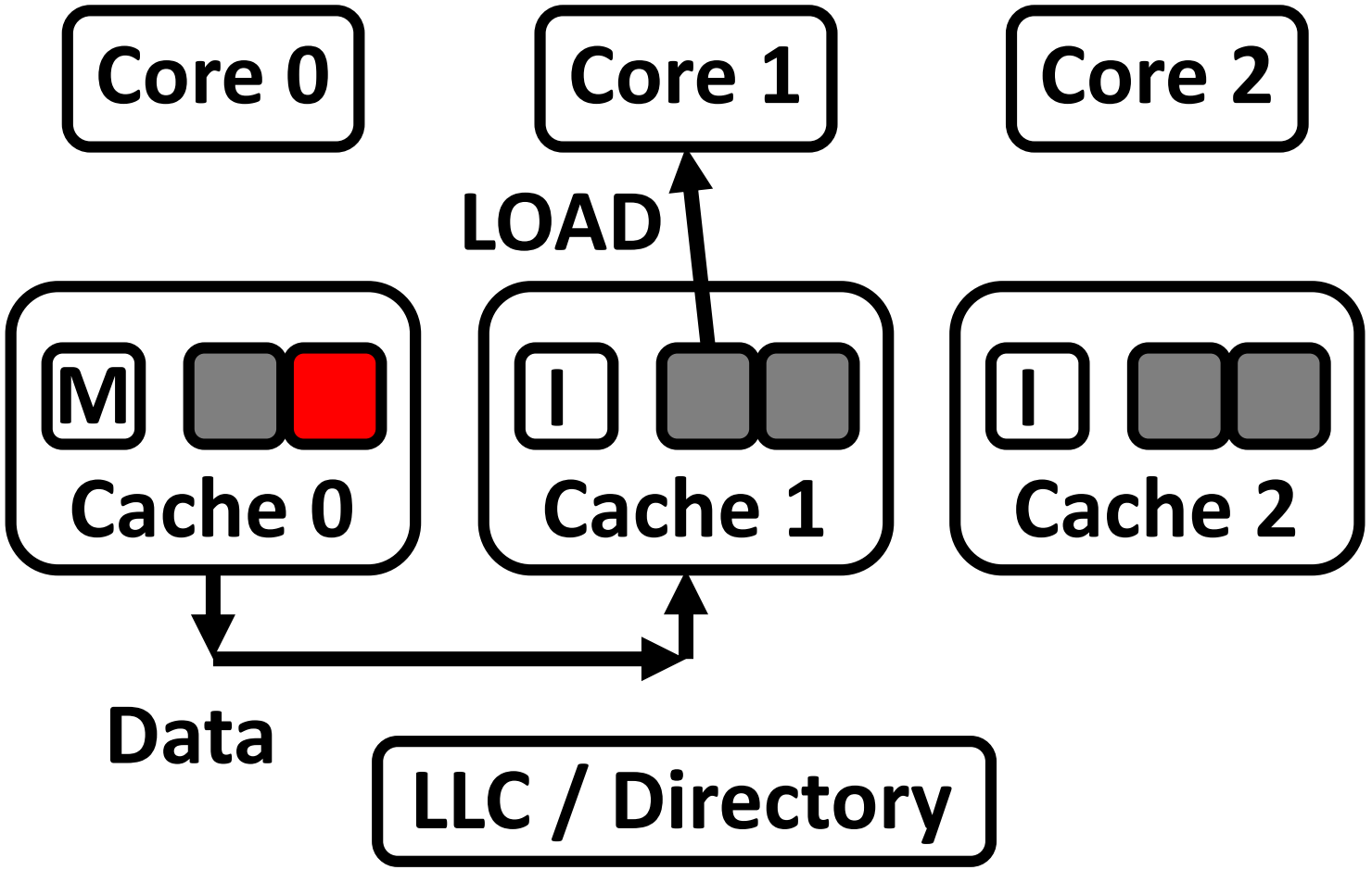
Example: Migratory Sharing



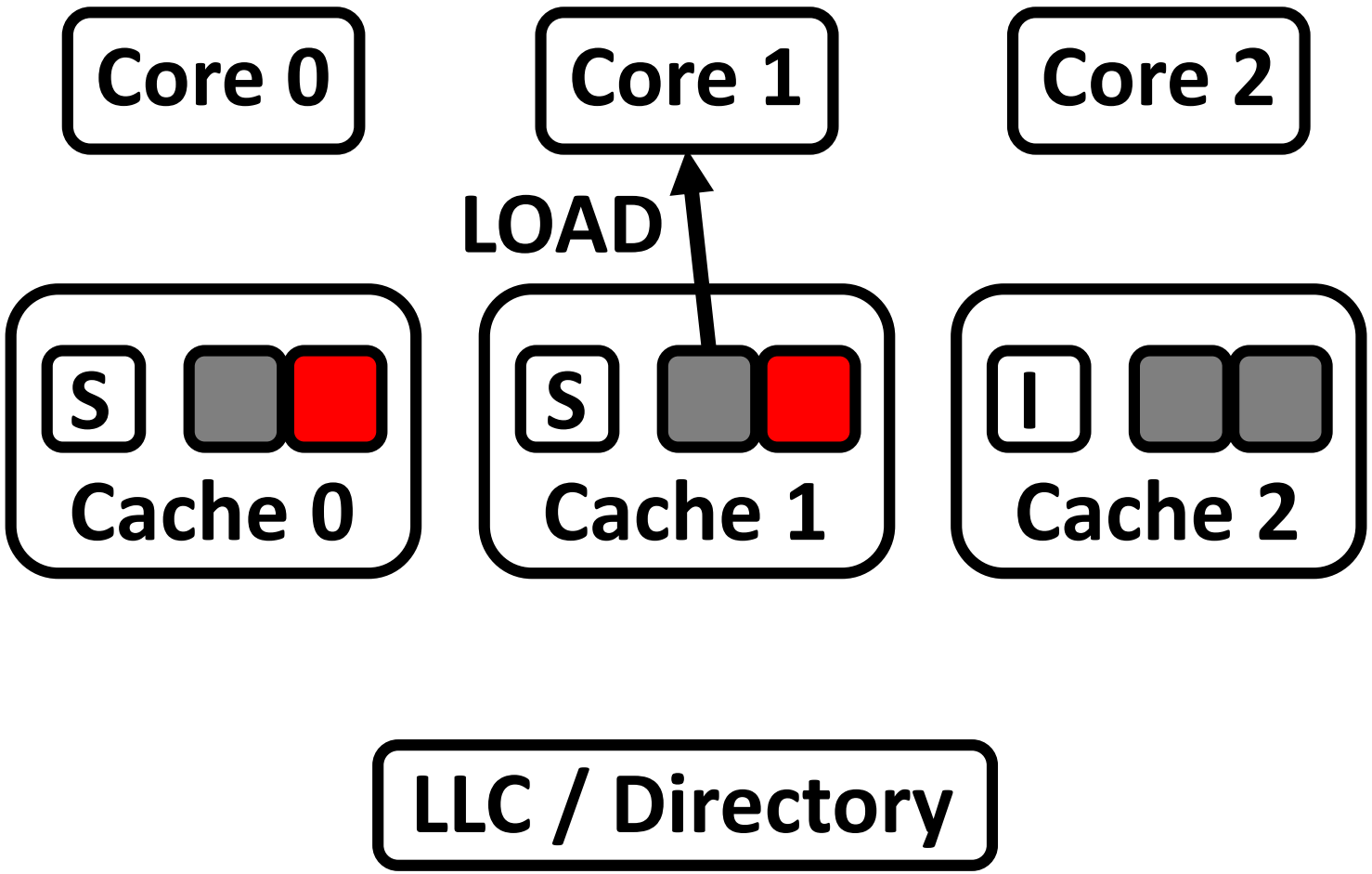
Example: Migratory Sharing



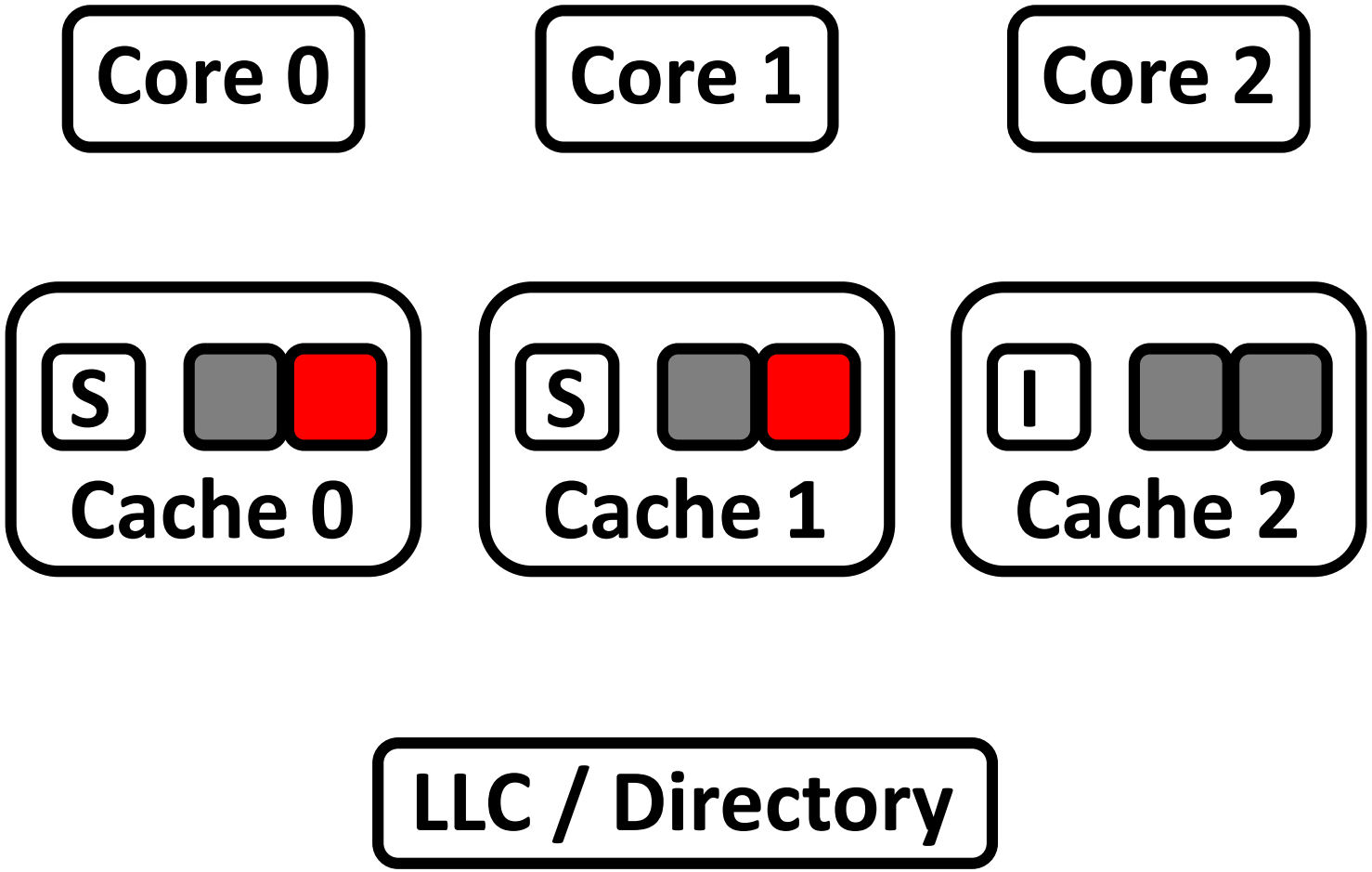
Example: Migratory Sharing



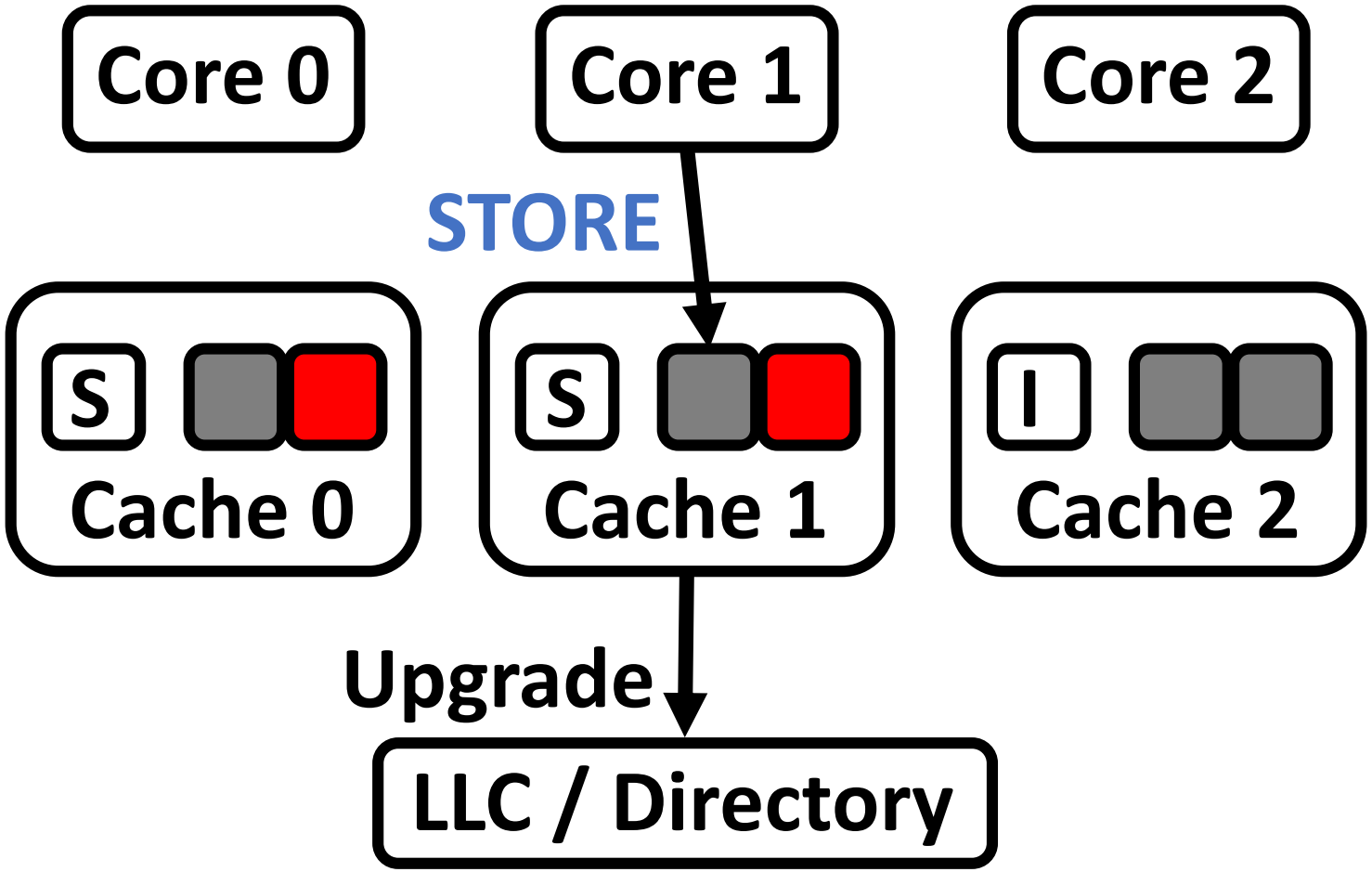
Example: Migratory Sharing



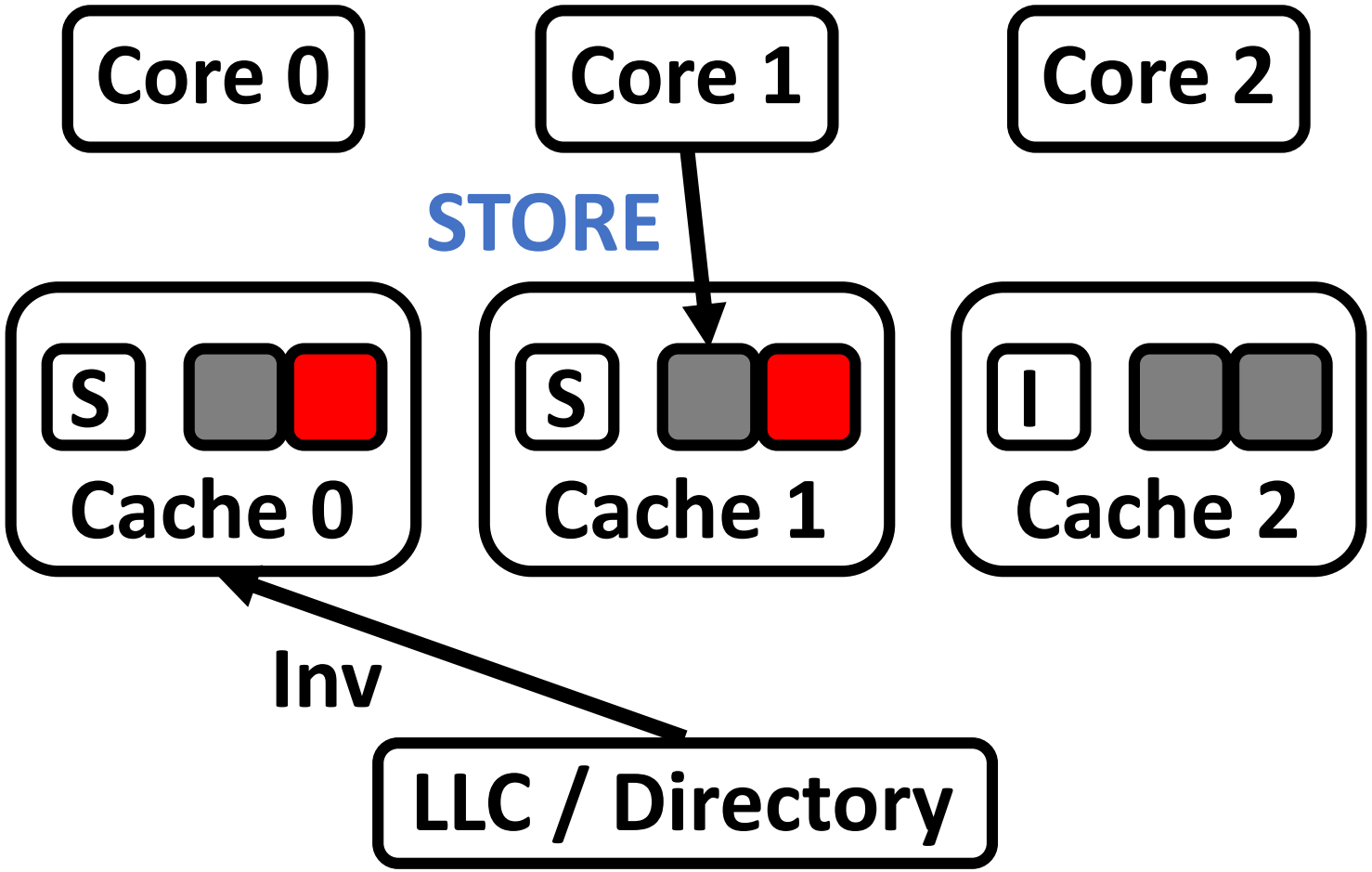
Example: Migratory Sharing



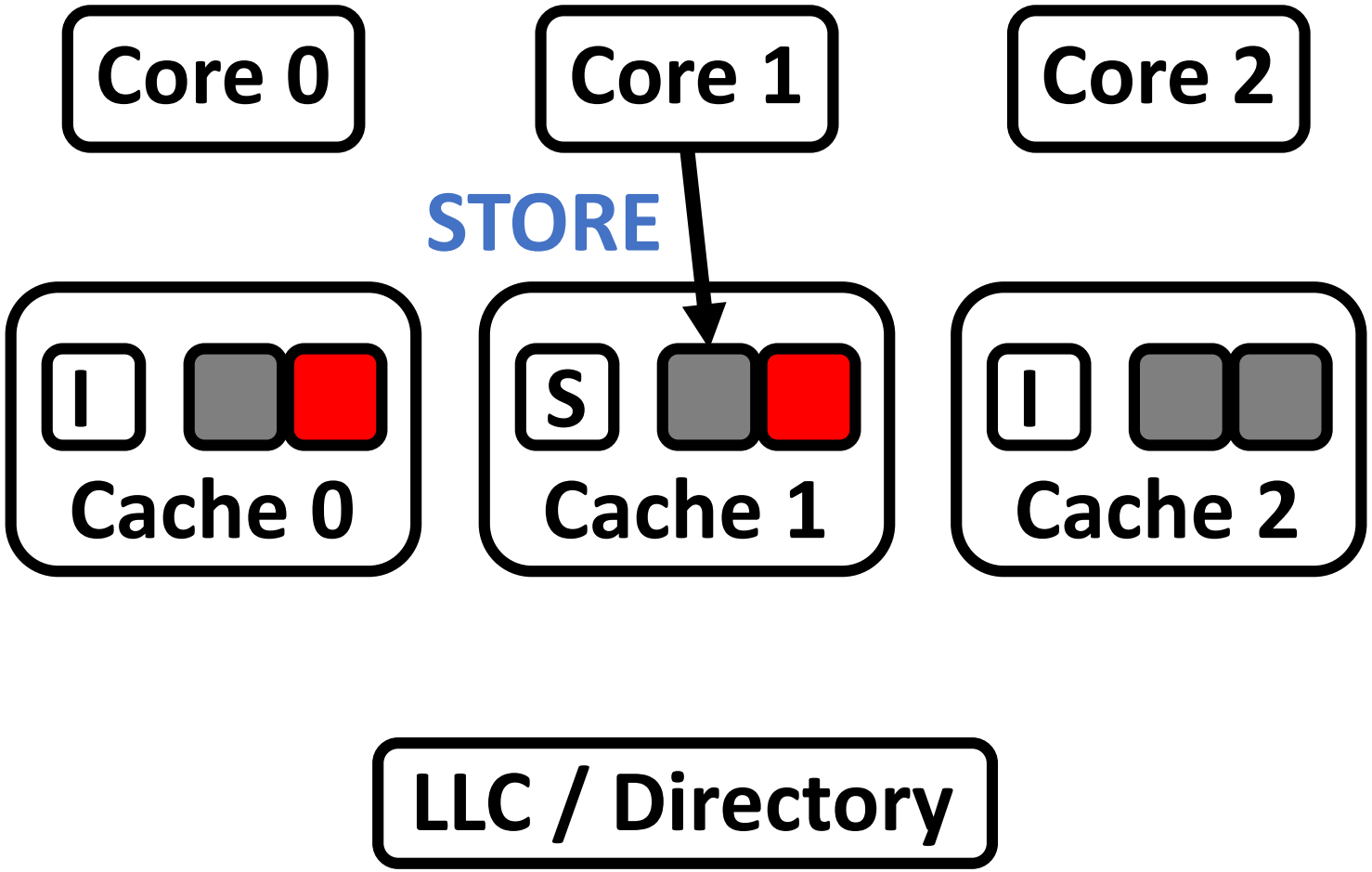
Example: Migratory Sharing



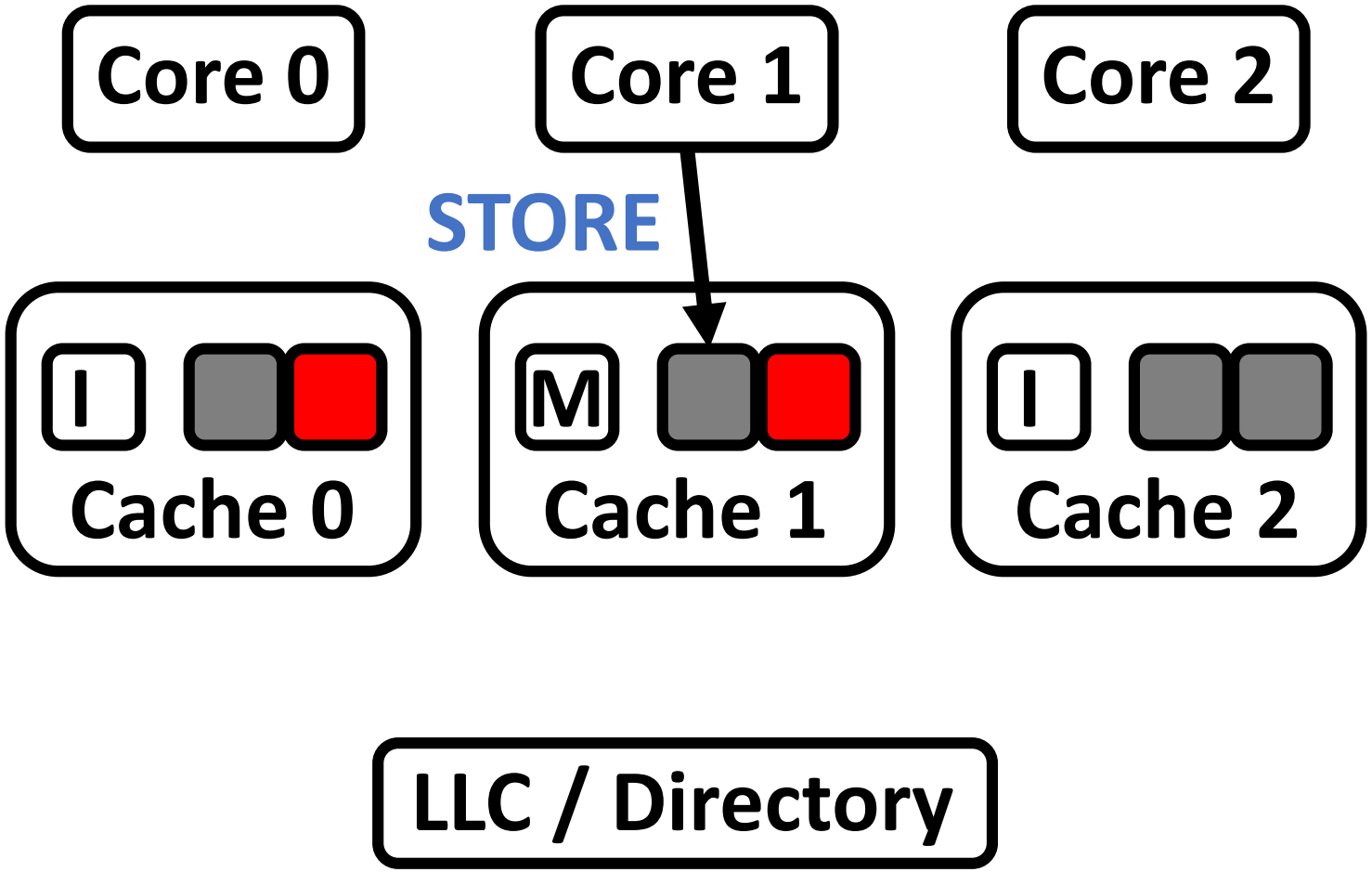
Example: Migratory Sharing



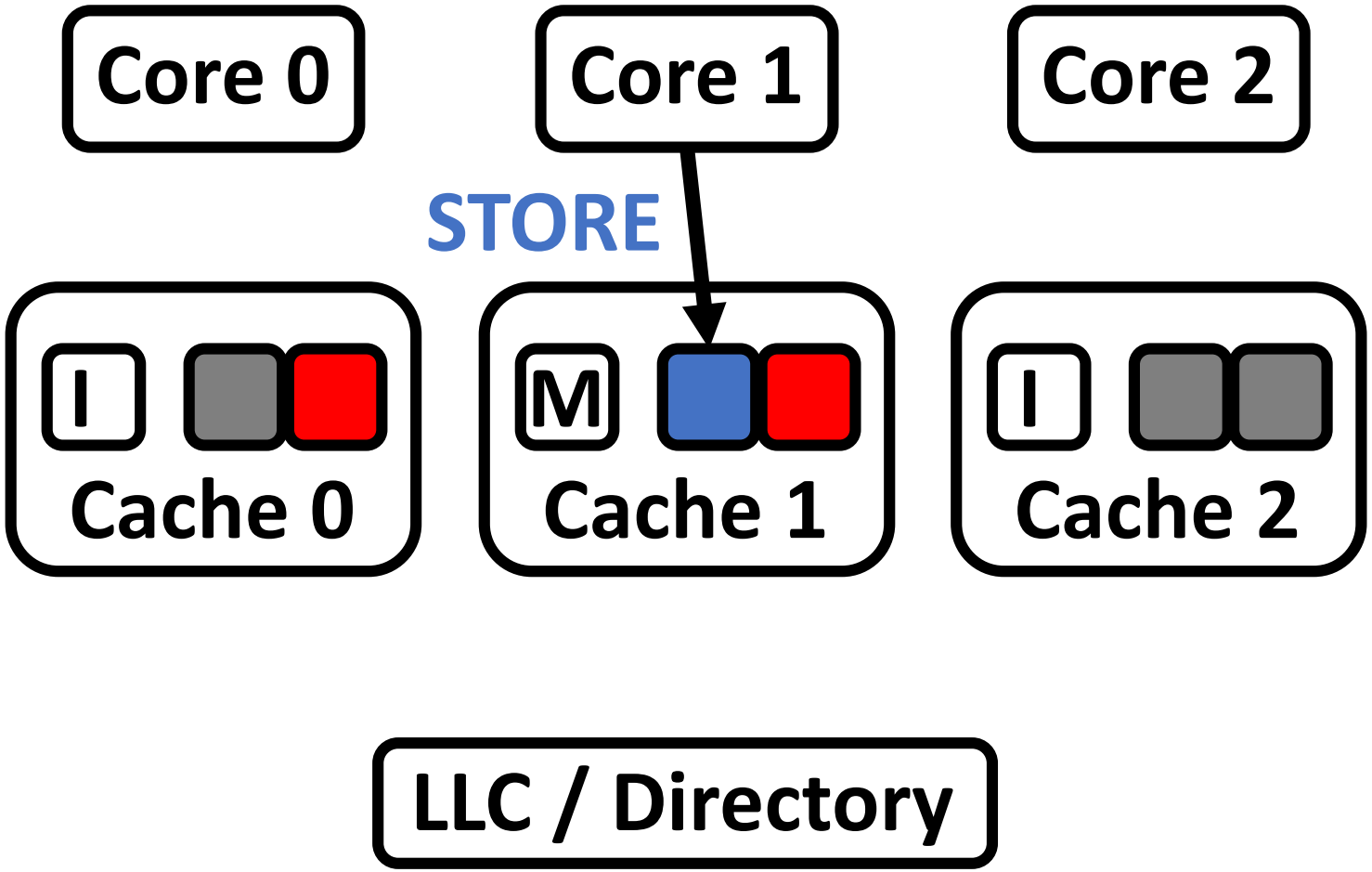
Example: Migratory Sharing



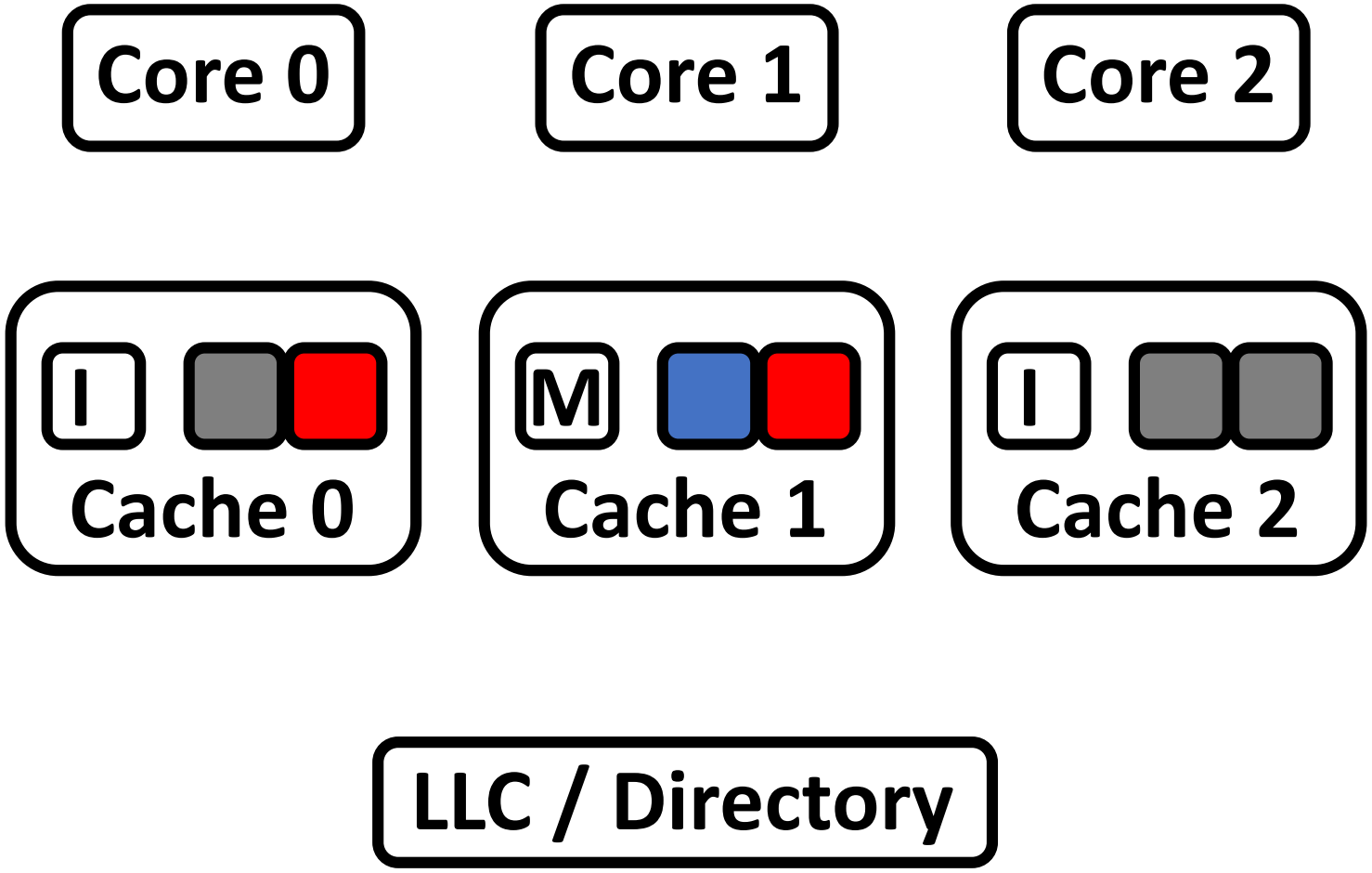
Example: Migratory Sharing



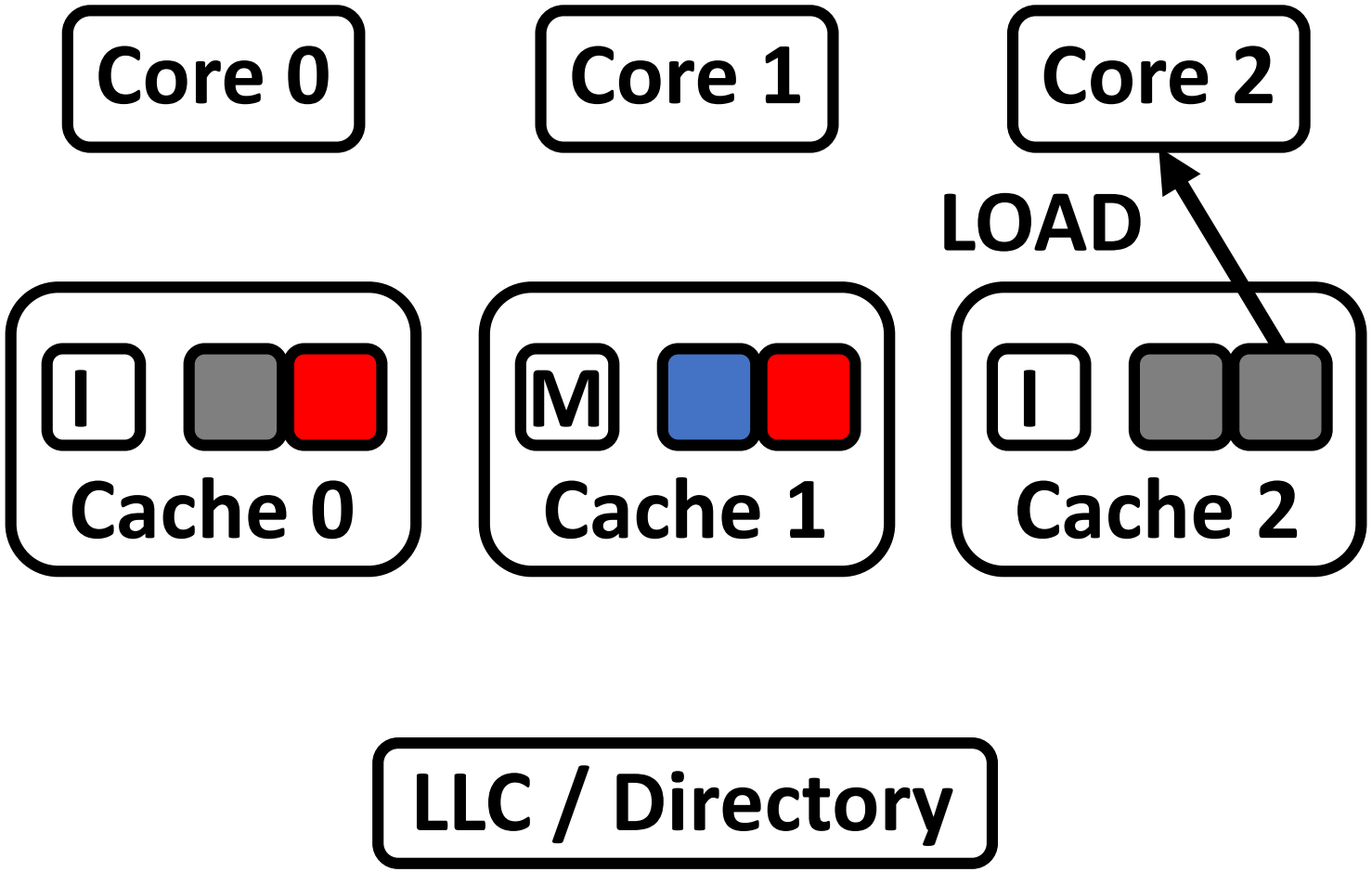
Example: Migratory Sharing



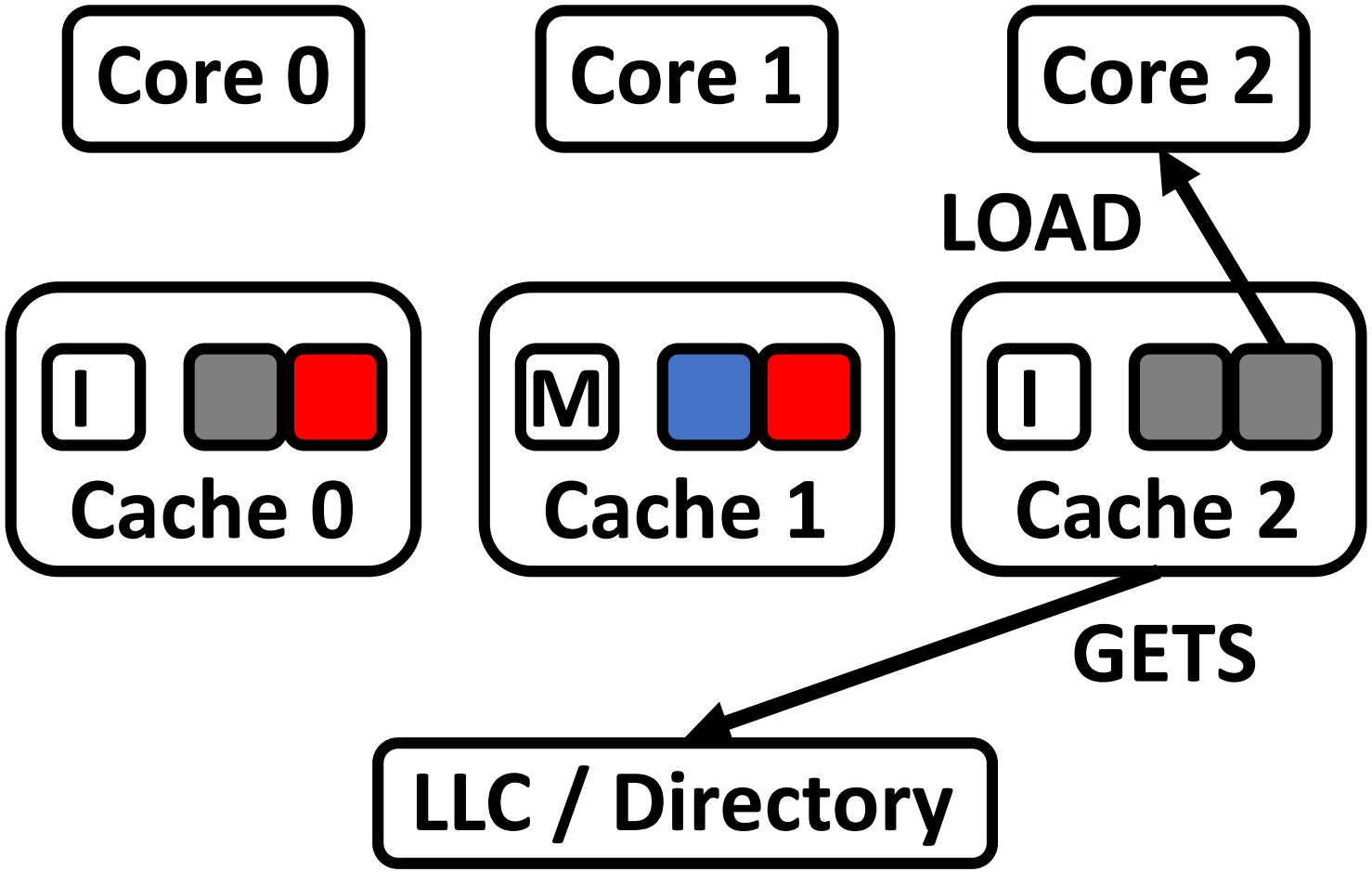
Example: Migratory Sharing



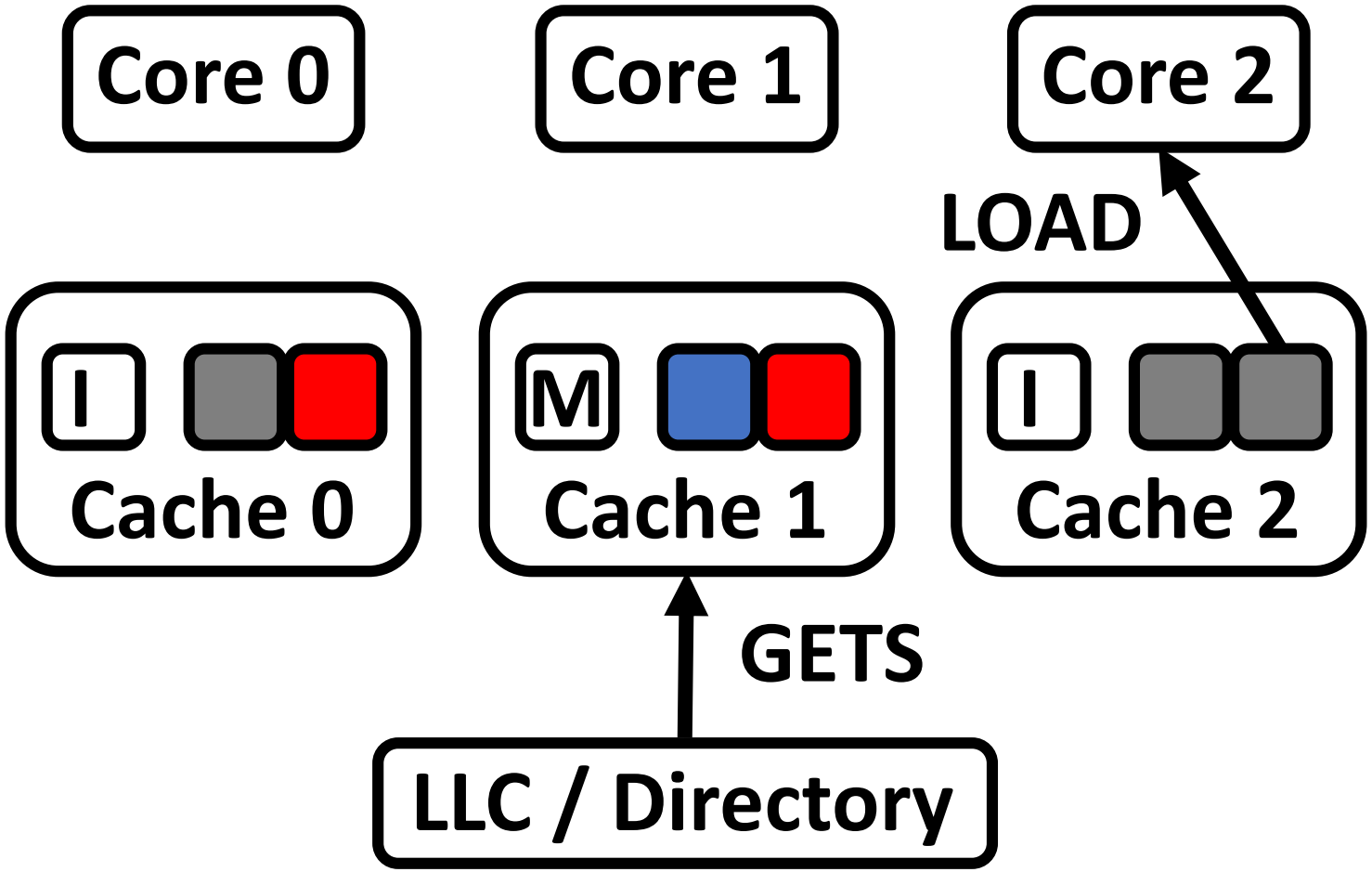
Example: Migratory Sharing



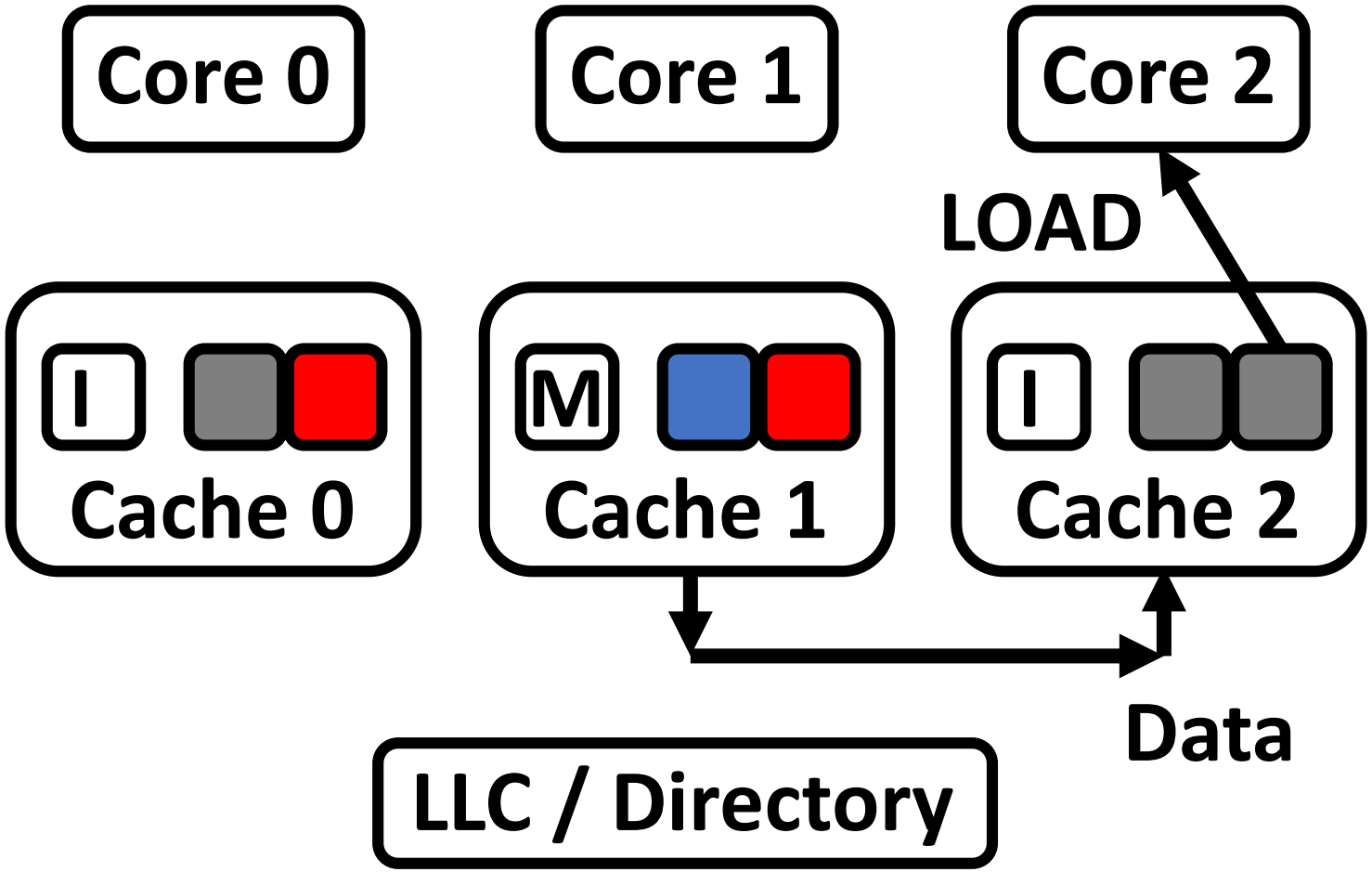
Example: Migratory Sharing



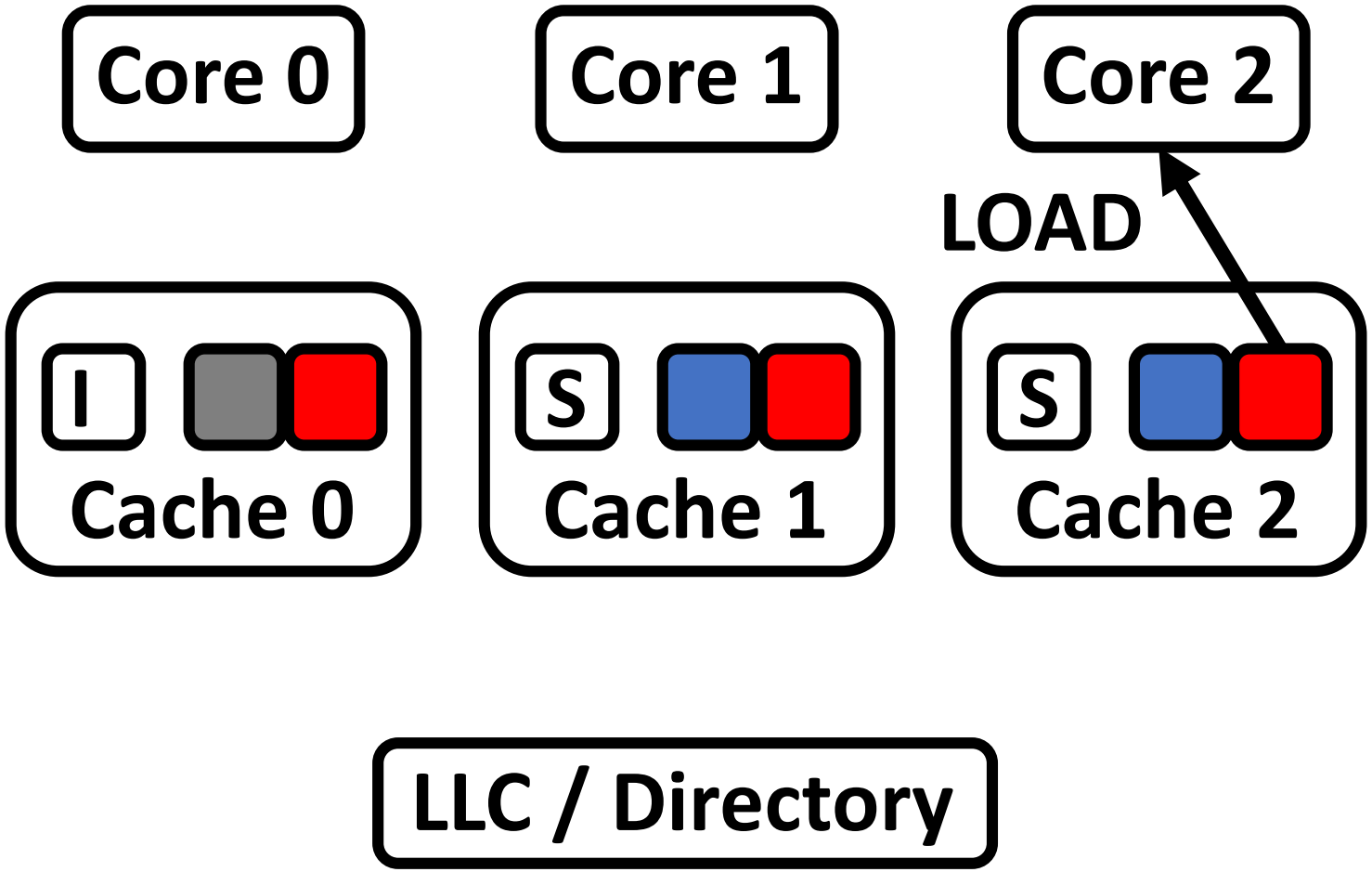
Example: Migratory Sharing



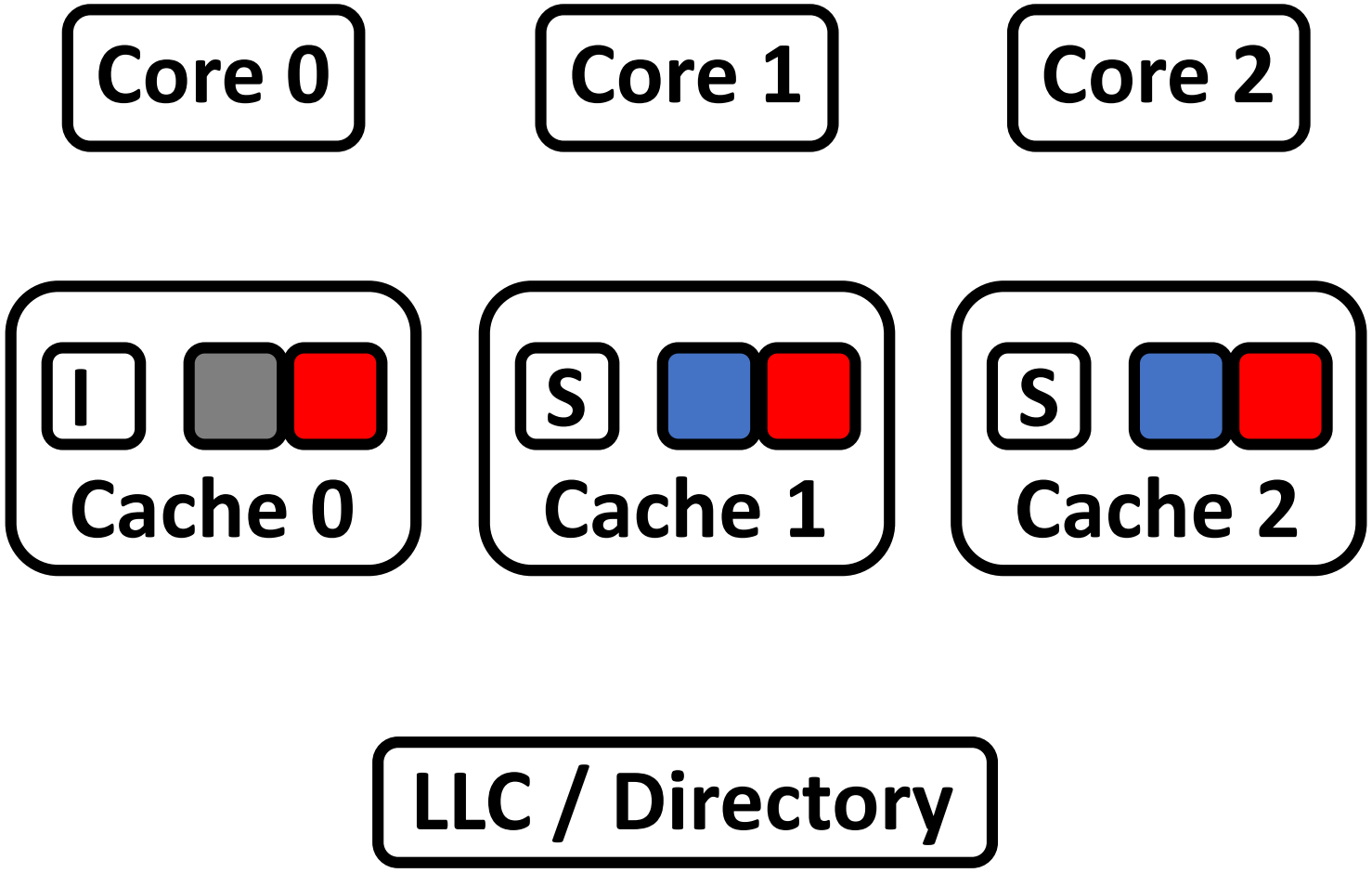
Example: Migratory Sharing



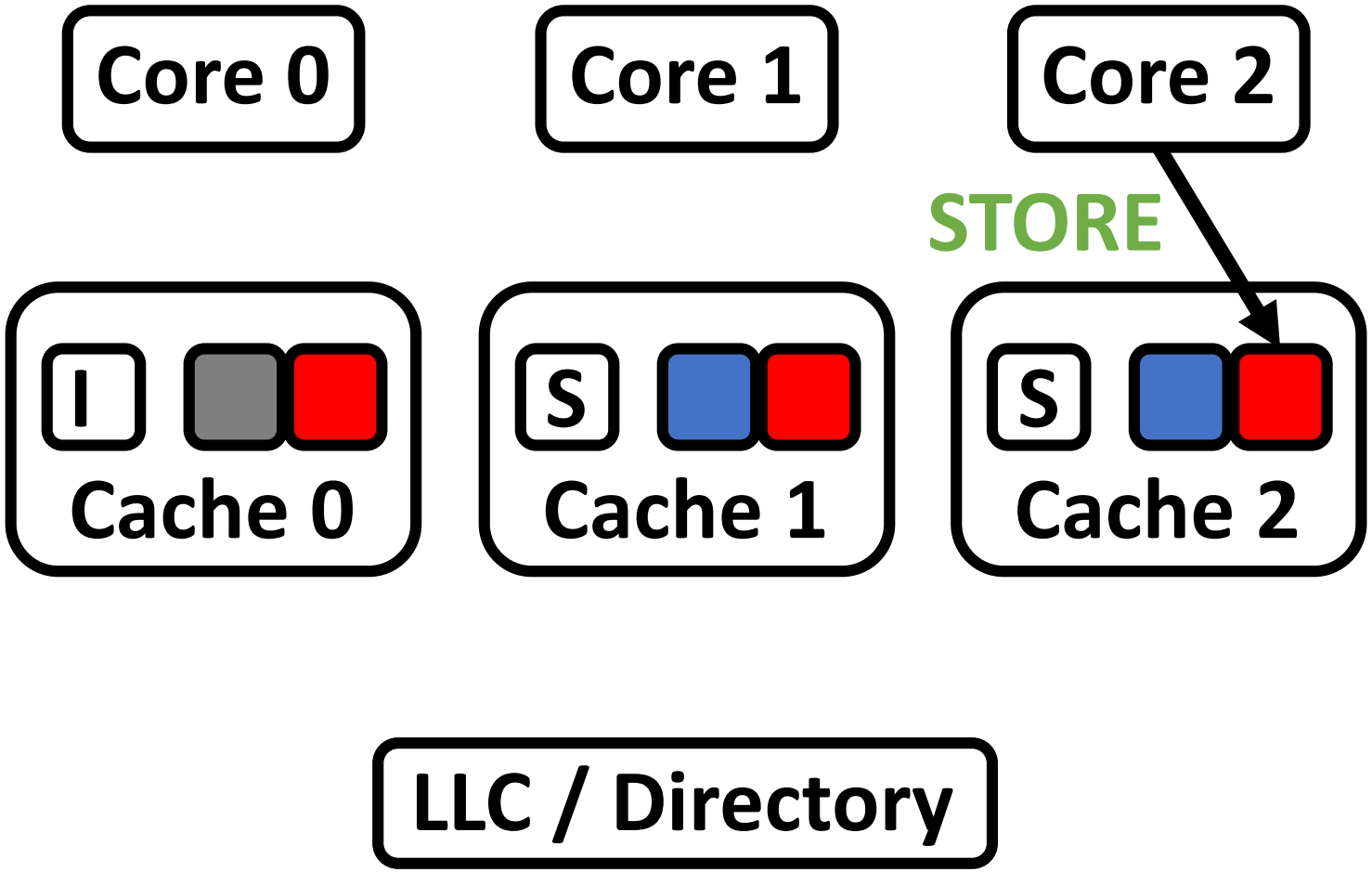
Example: Migratory Sharing



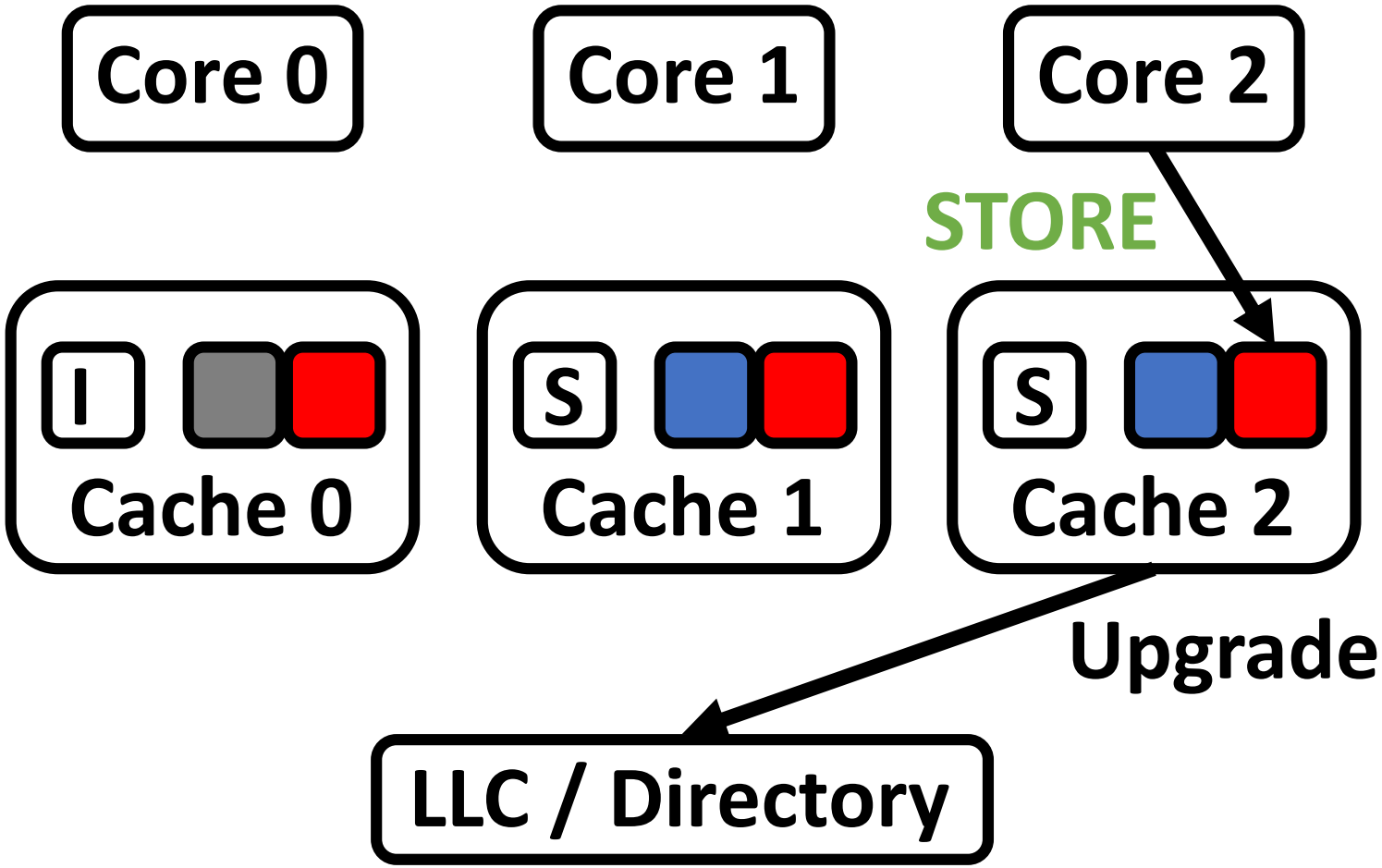
Example: Migratory Sharing



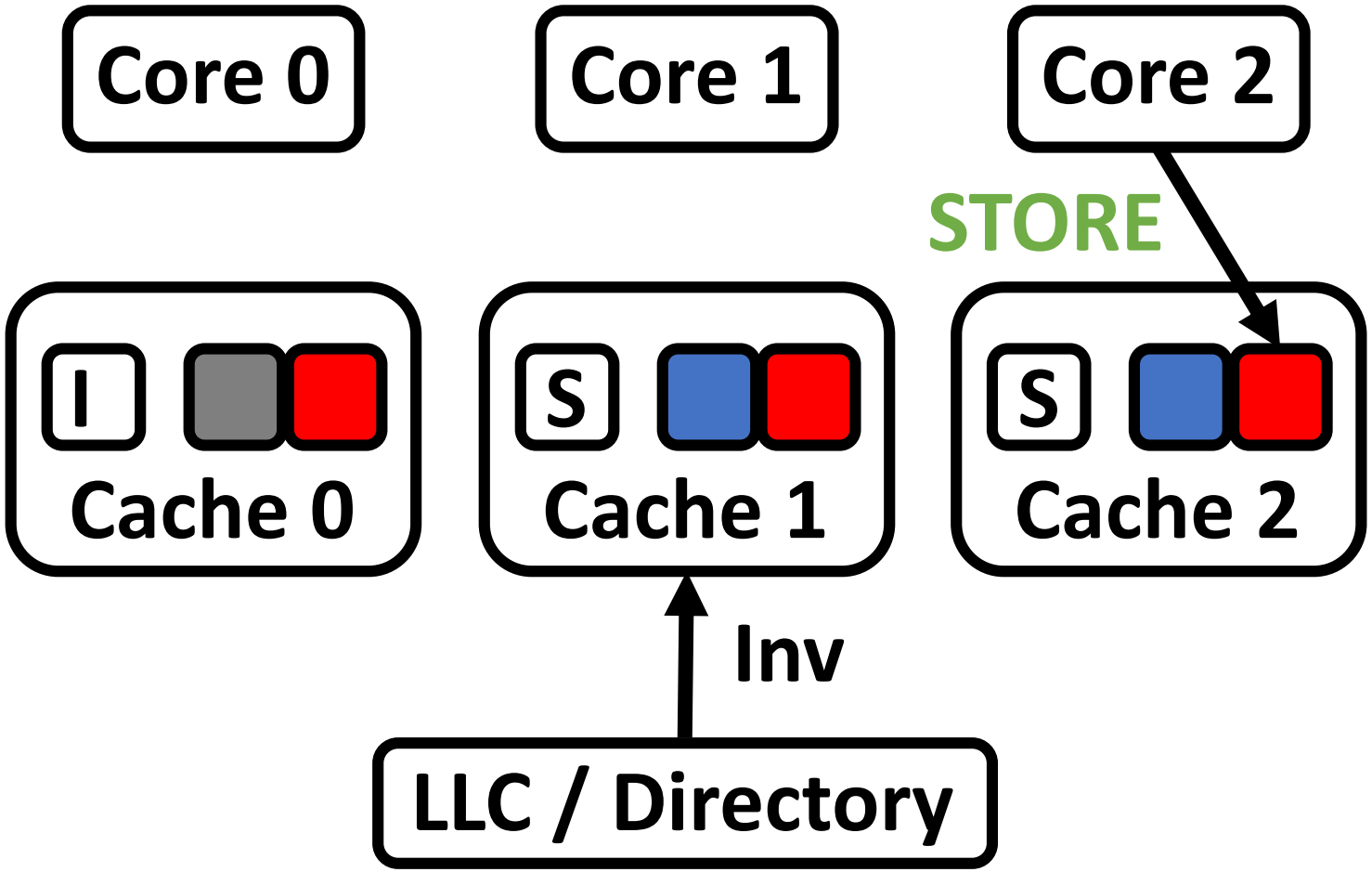
Example: Migratory Sharing



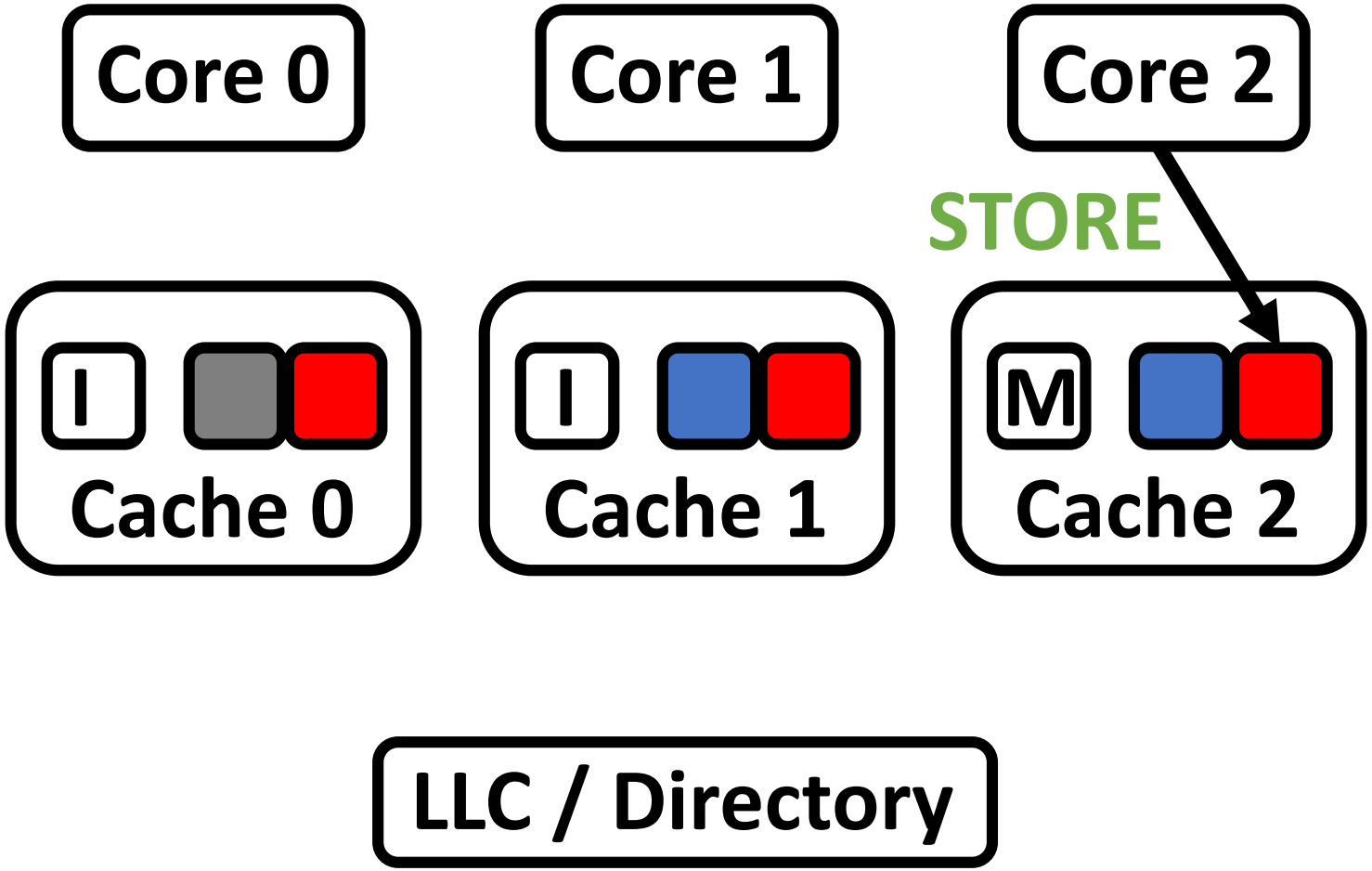
Example: Migratory Sharing



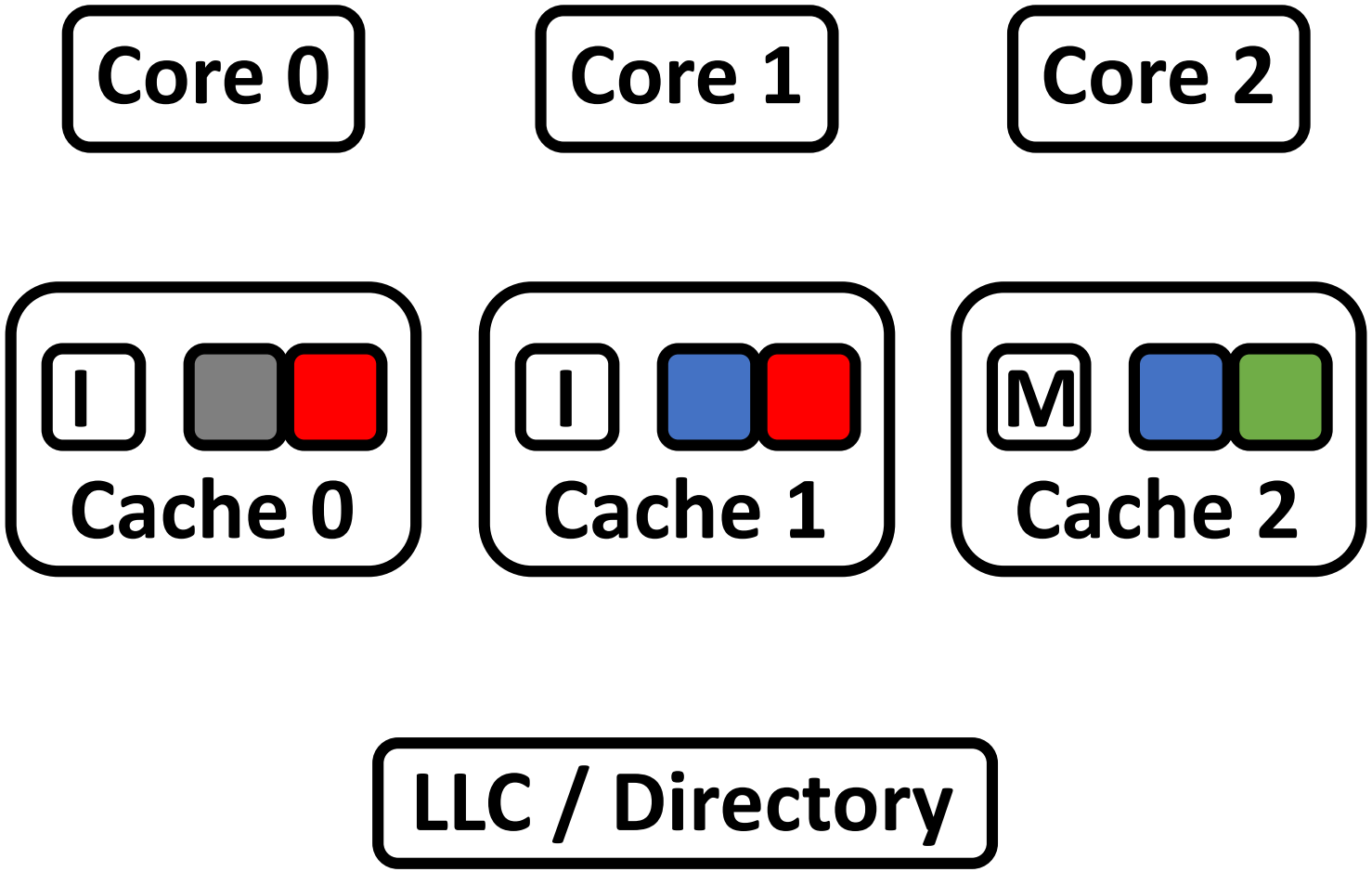
Example: Migratory Sharing



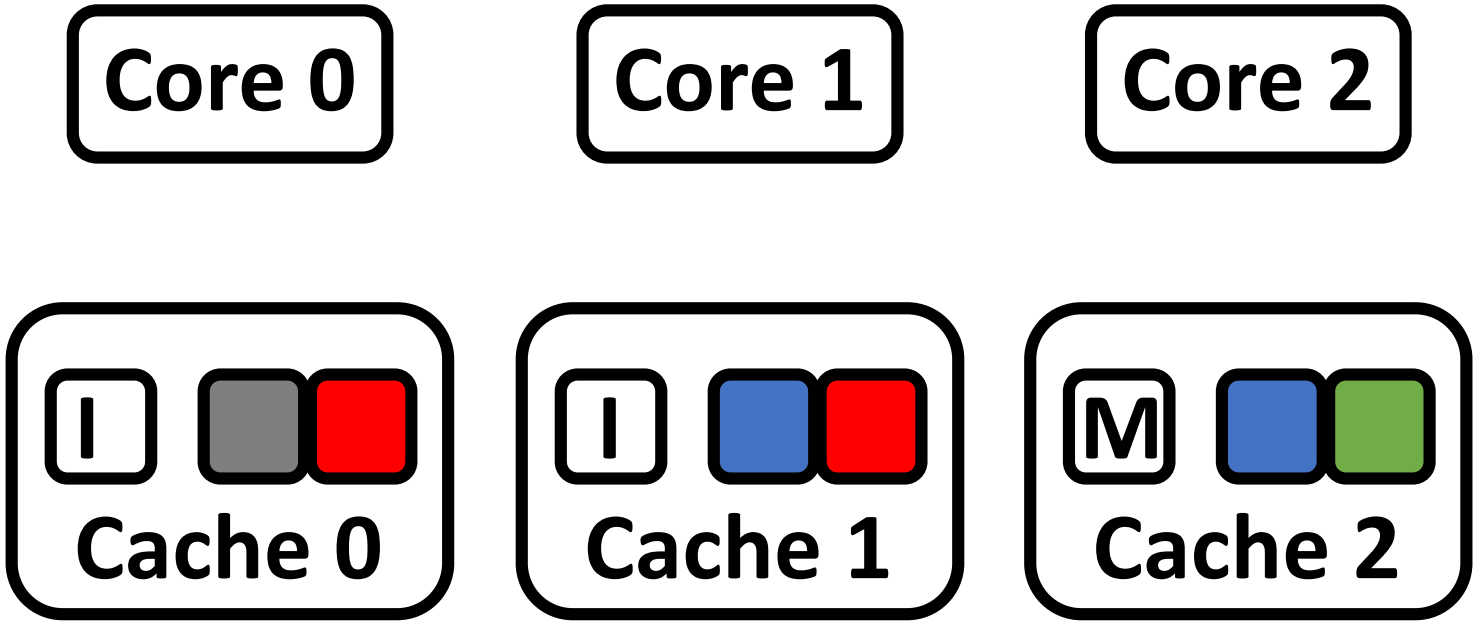
Example: Migratory Sharing



Example: Migratory Sharing

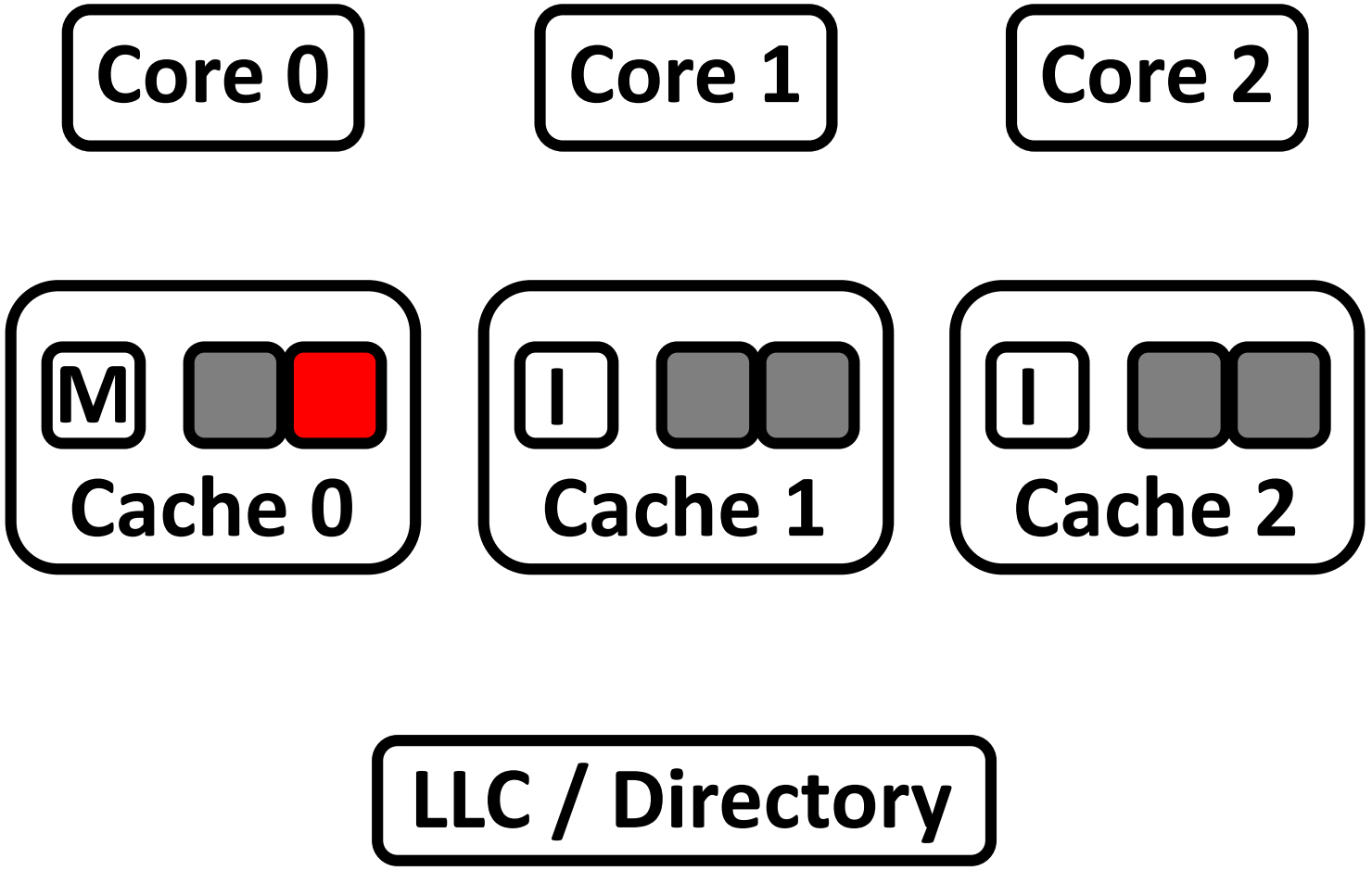


Example: Migratory Sharing

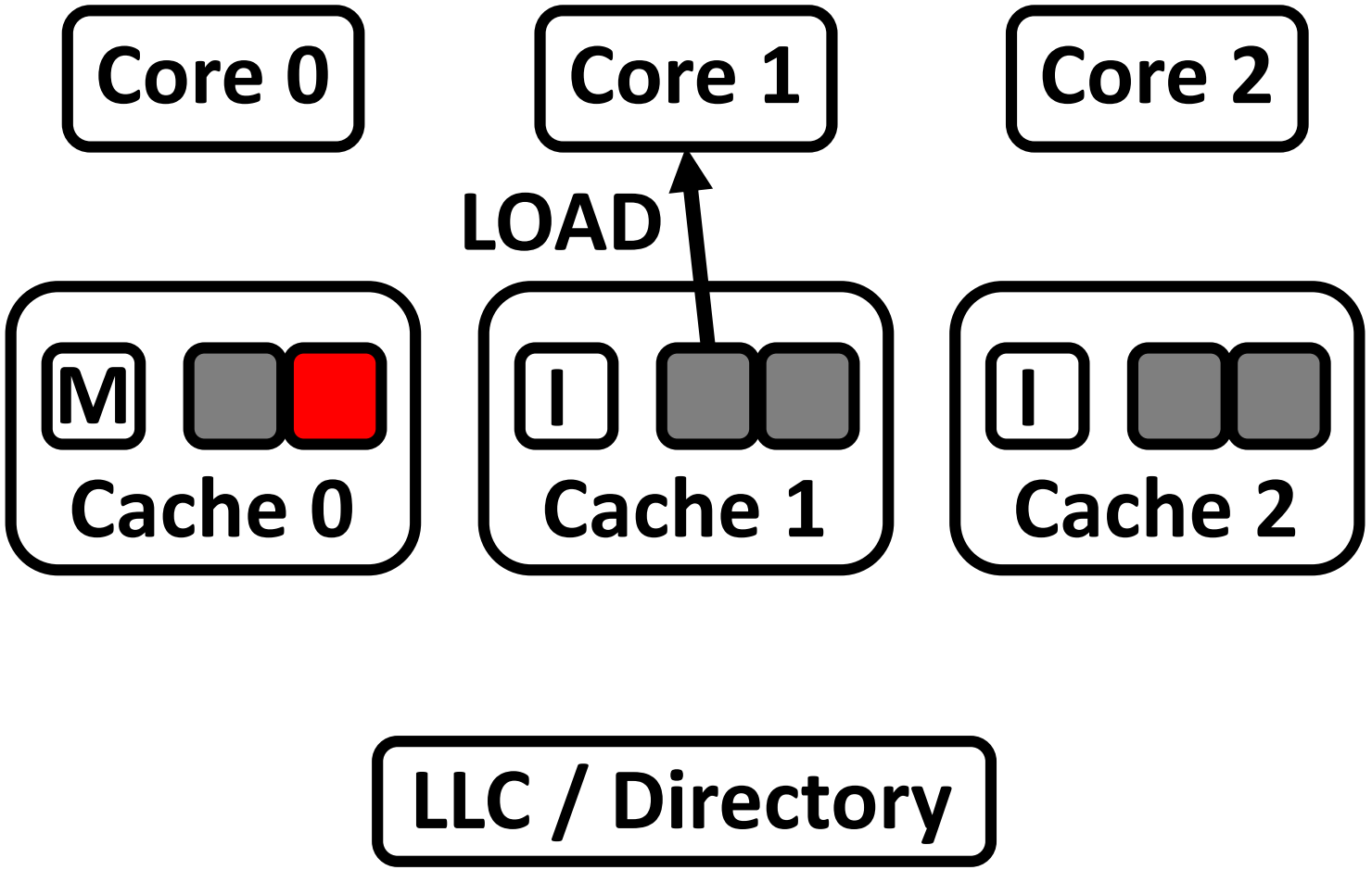


All loads miss on Invalid, All stores miss on Shared

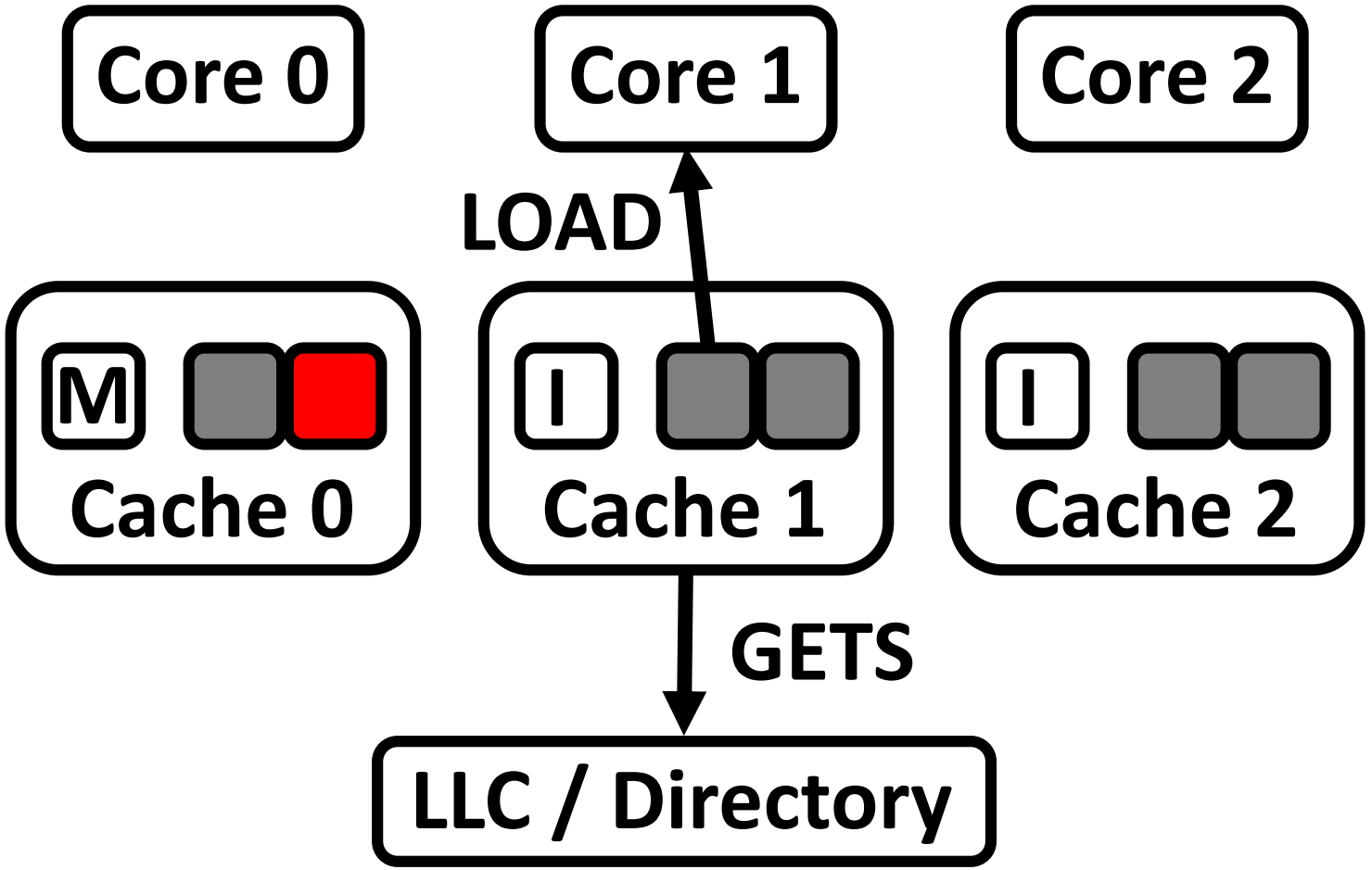
Example: Migratory Sharing (Ghostwriter)



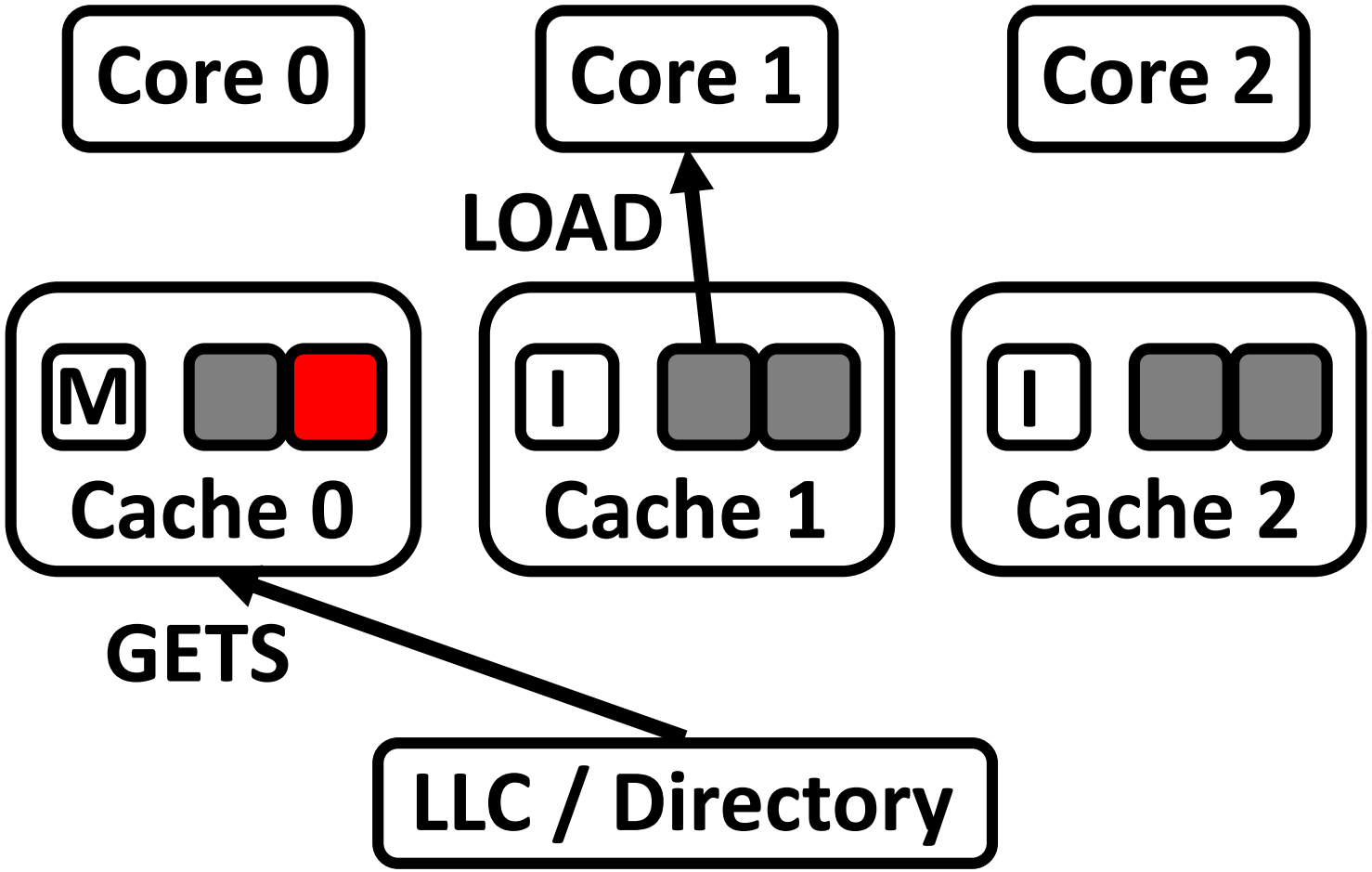
Example: Migratory Sharing (Ghostwriter)



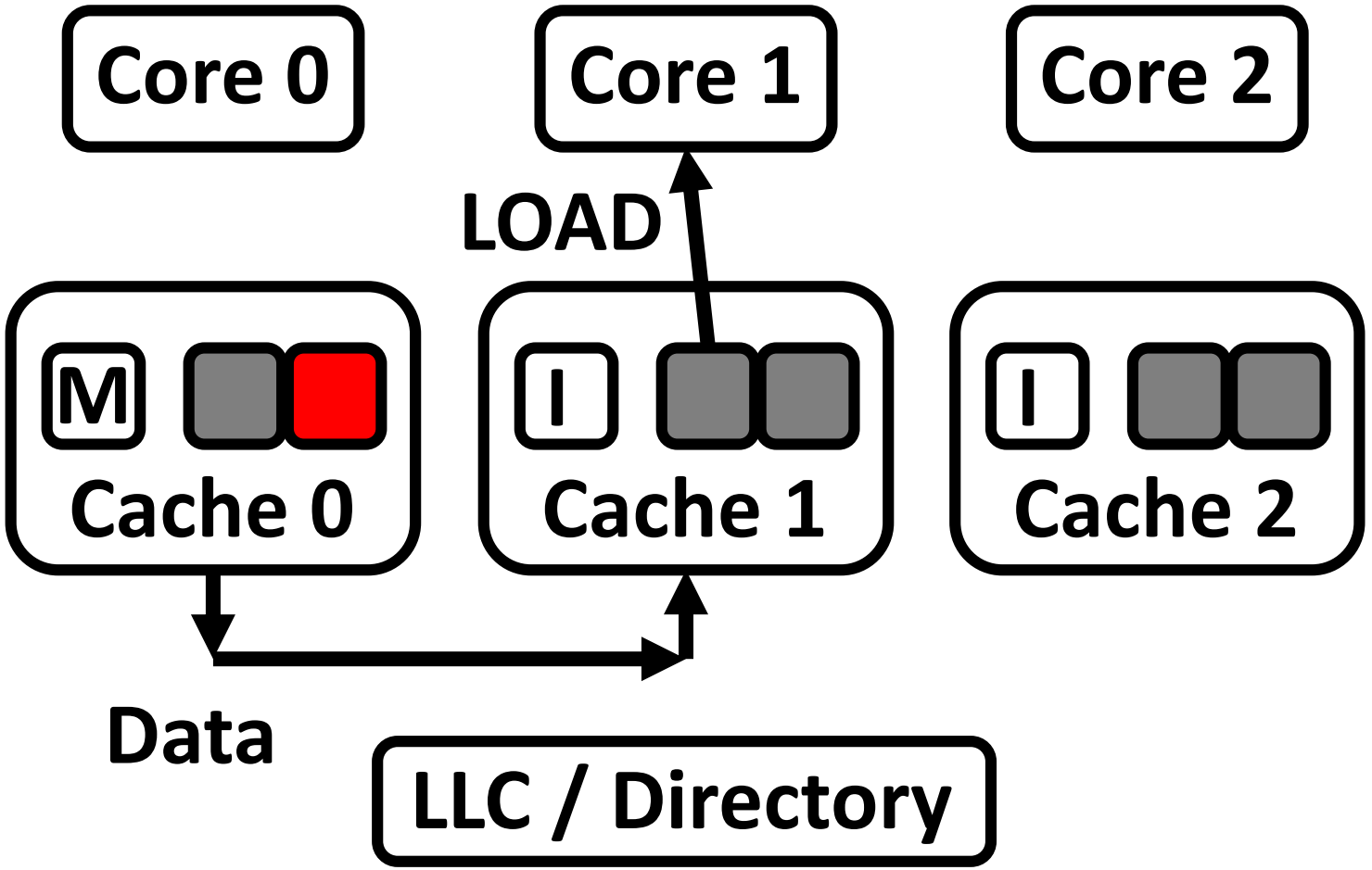
Example: Migratory Sharing (Ghostwriter)



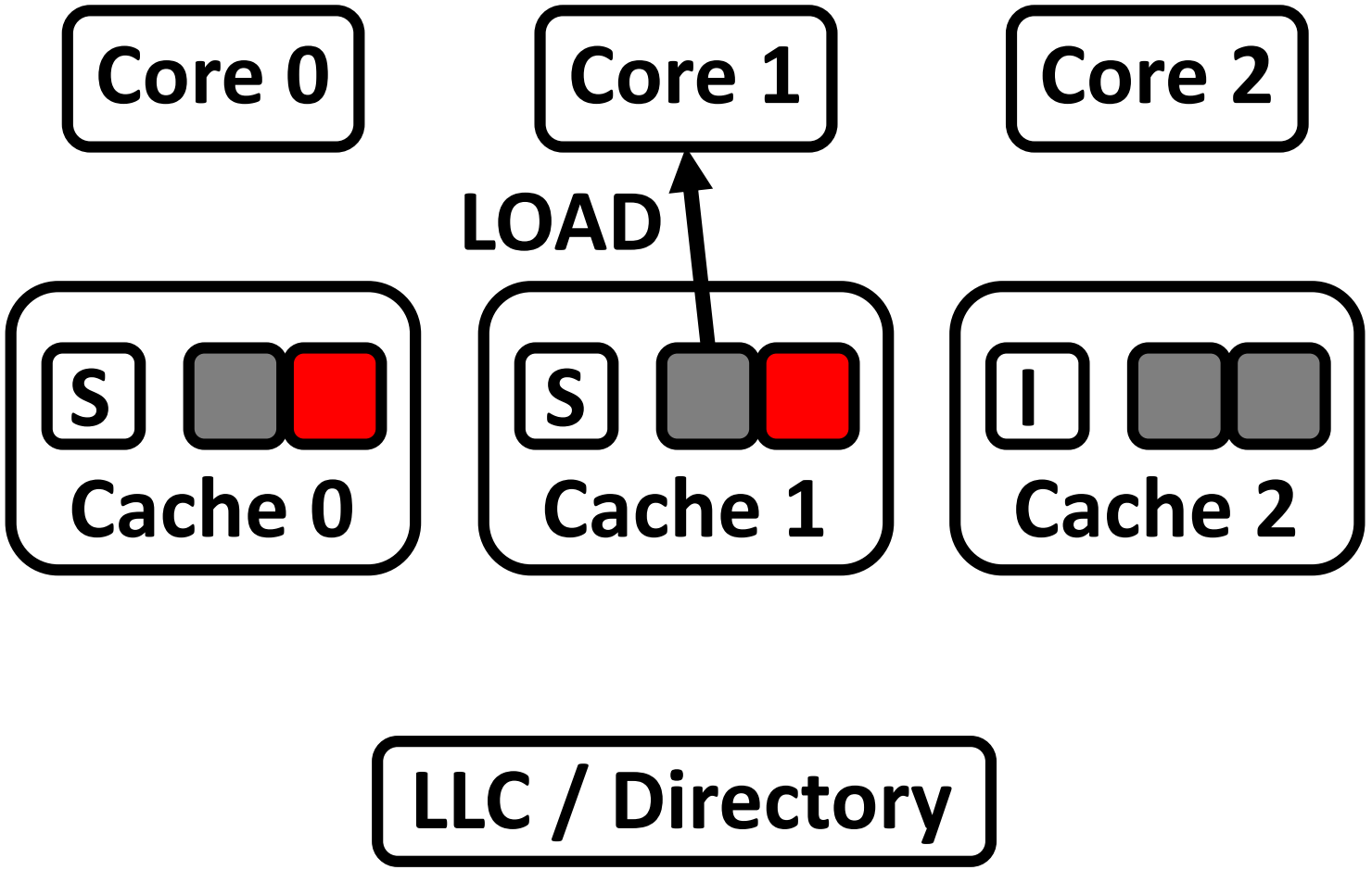
Example: Migratory Sharing (Ghostwriter)



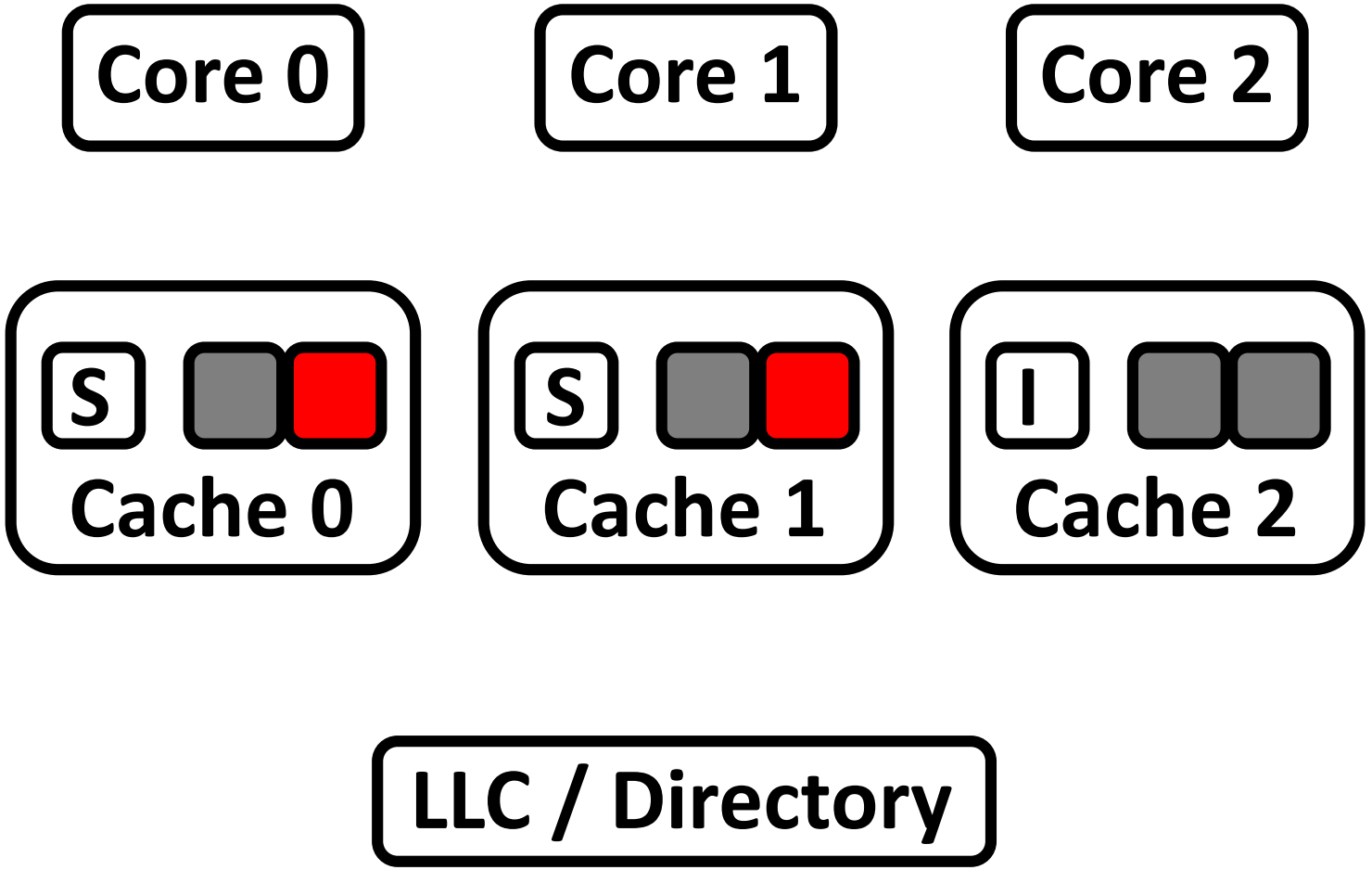
Example: Migratory Sharing (Ghostwriter)



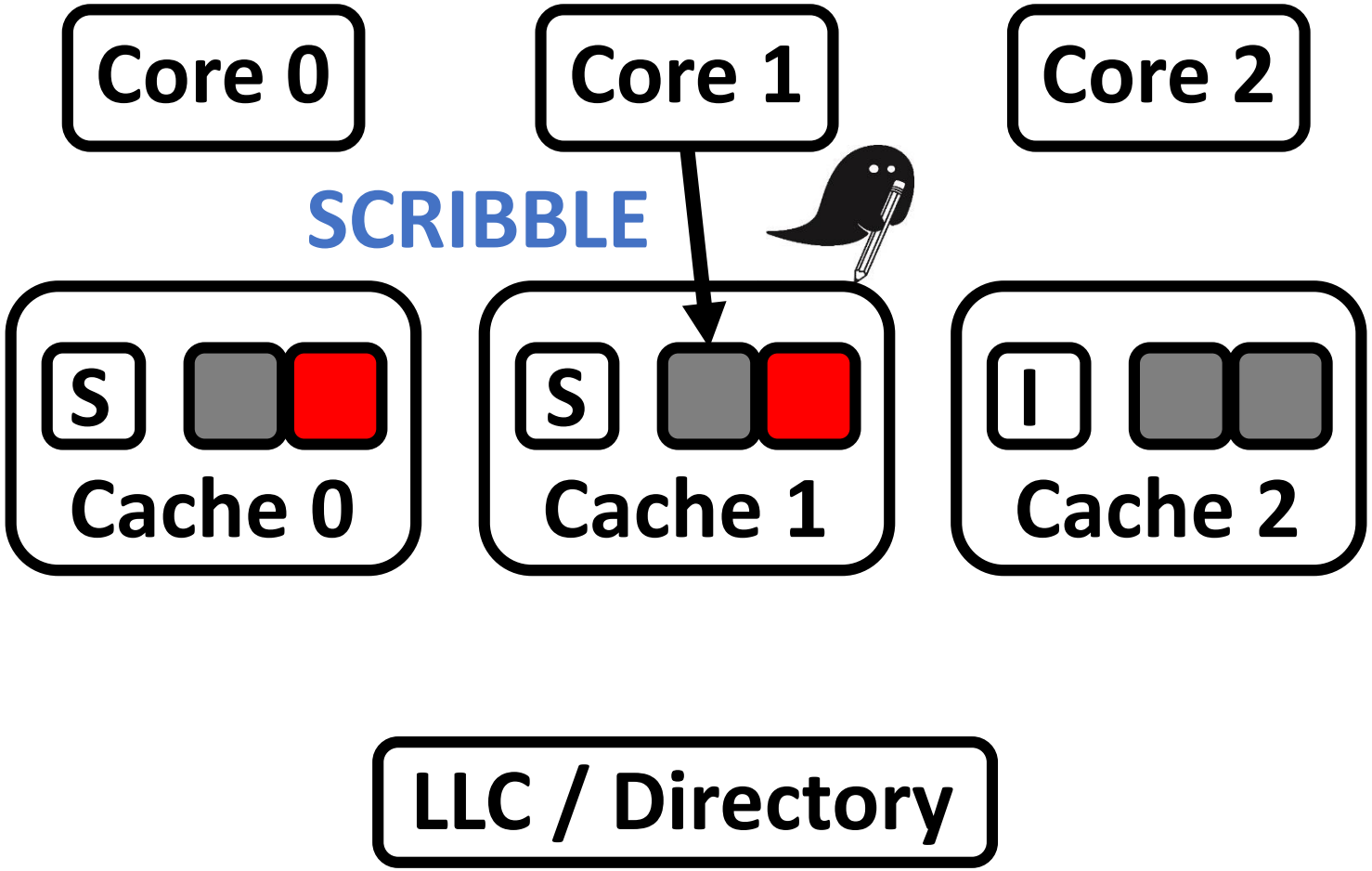
Example: Migratory Sharing (Ghostwriter)



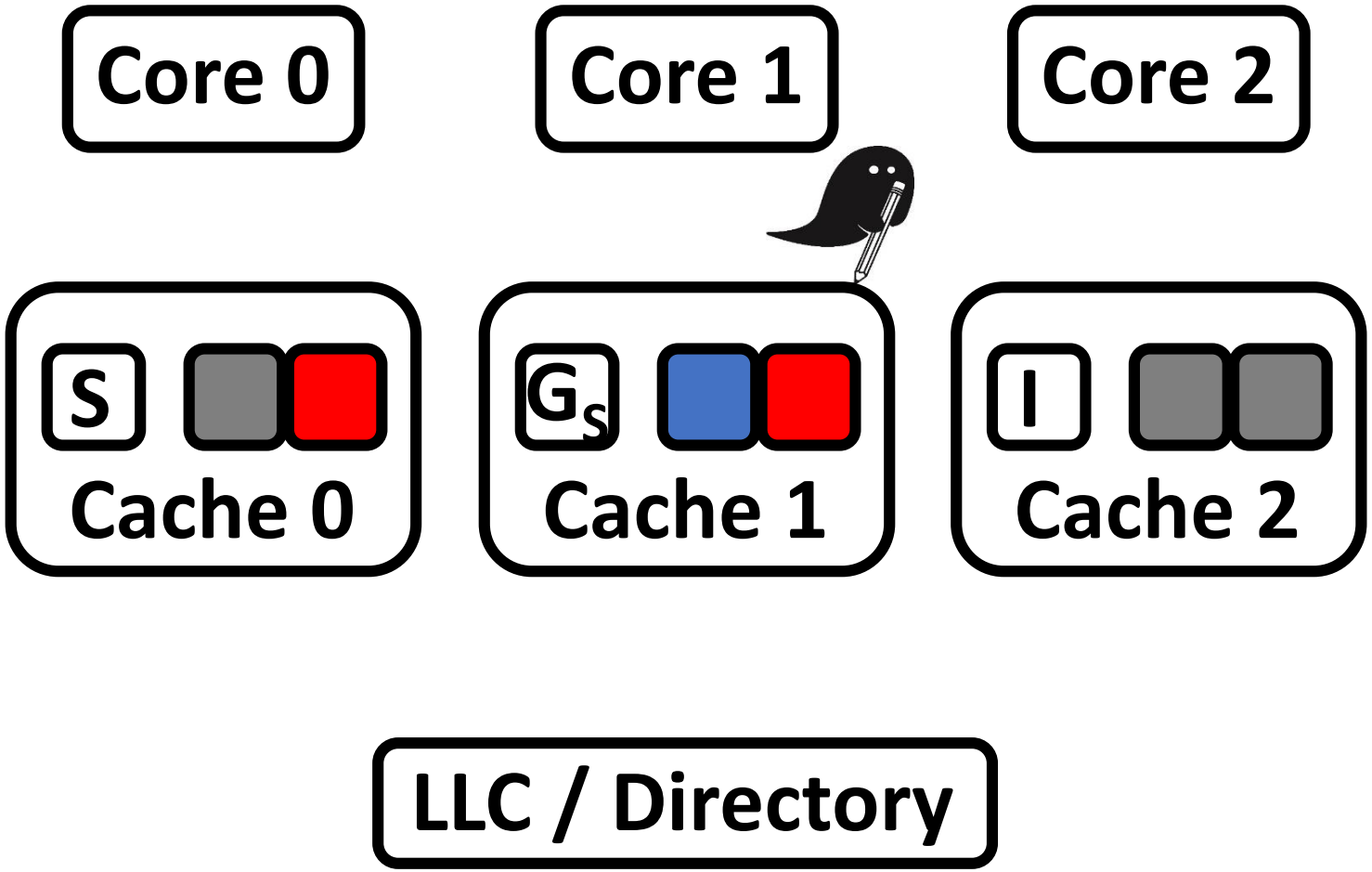
Example: Migratory Sharing (Ghostwriter)



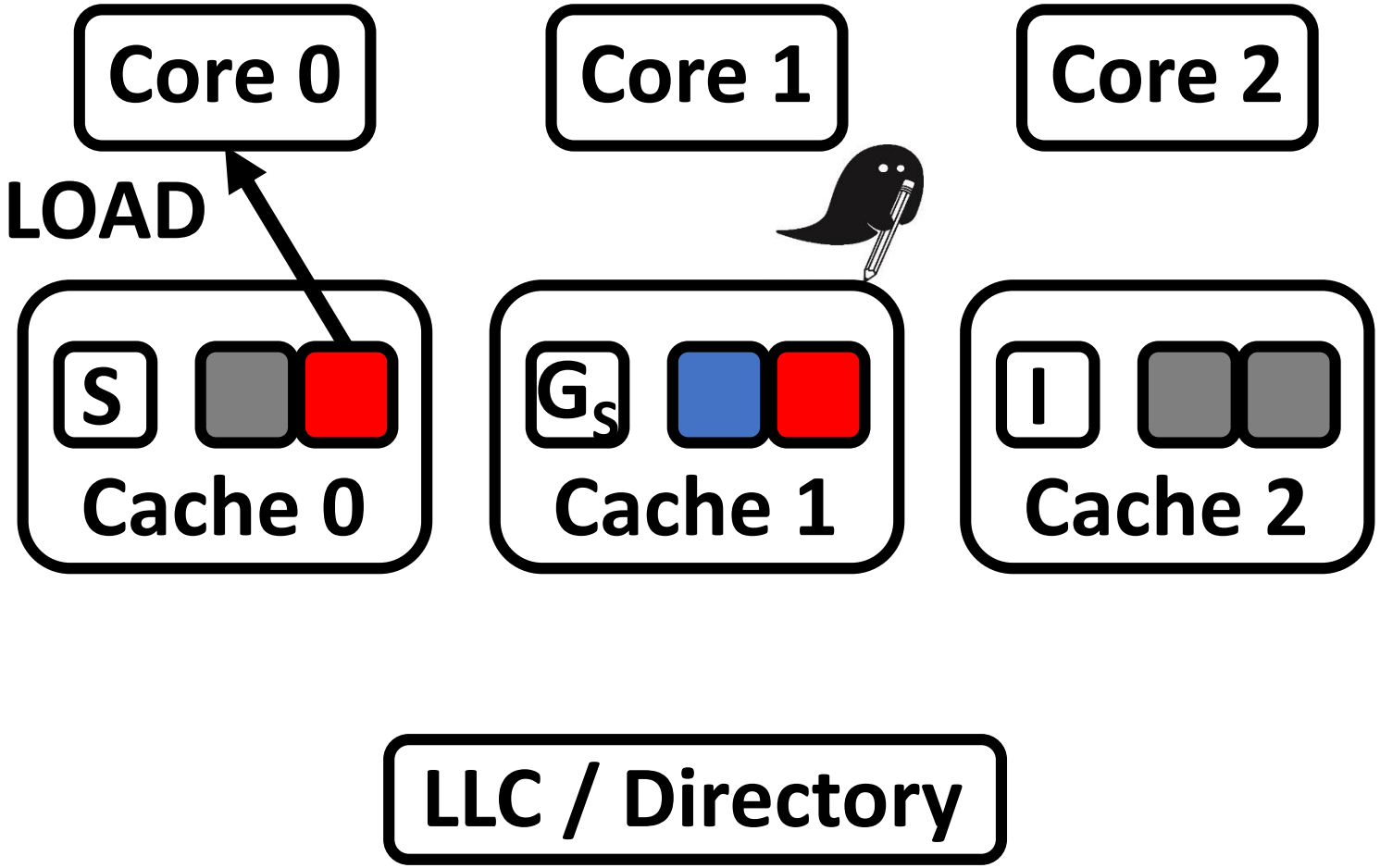
Example: Migratory Sharing (Ghostwriter)



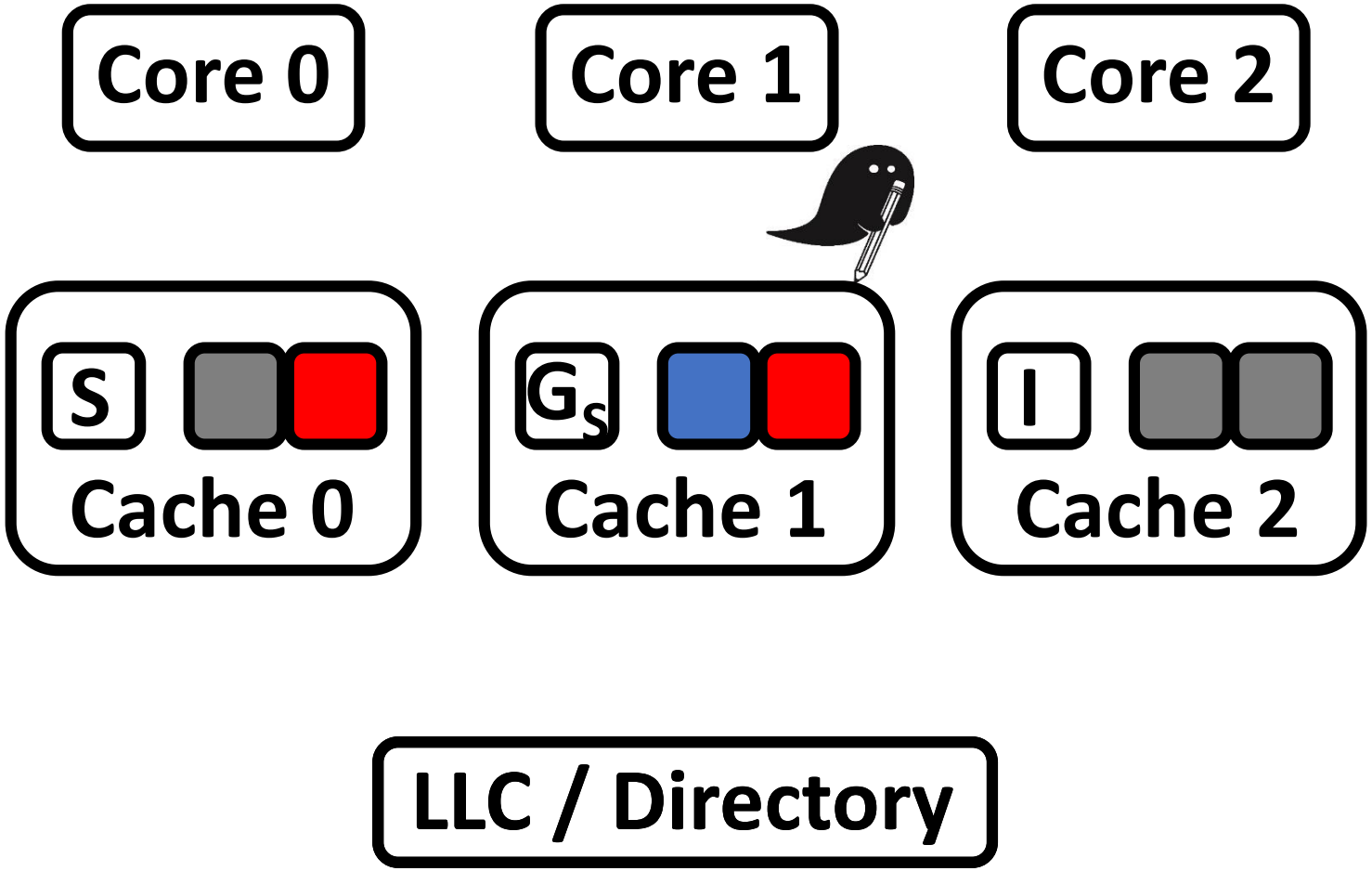
Example: Migratory Sharing (Ghostwriter)



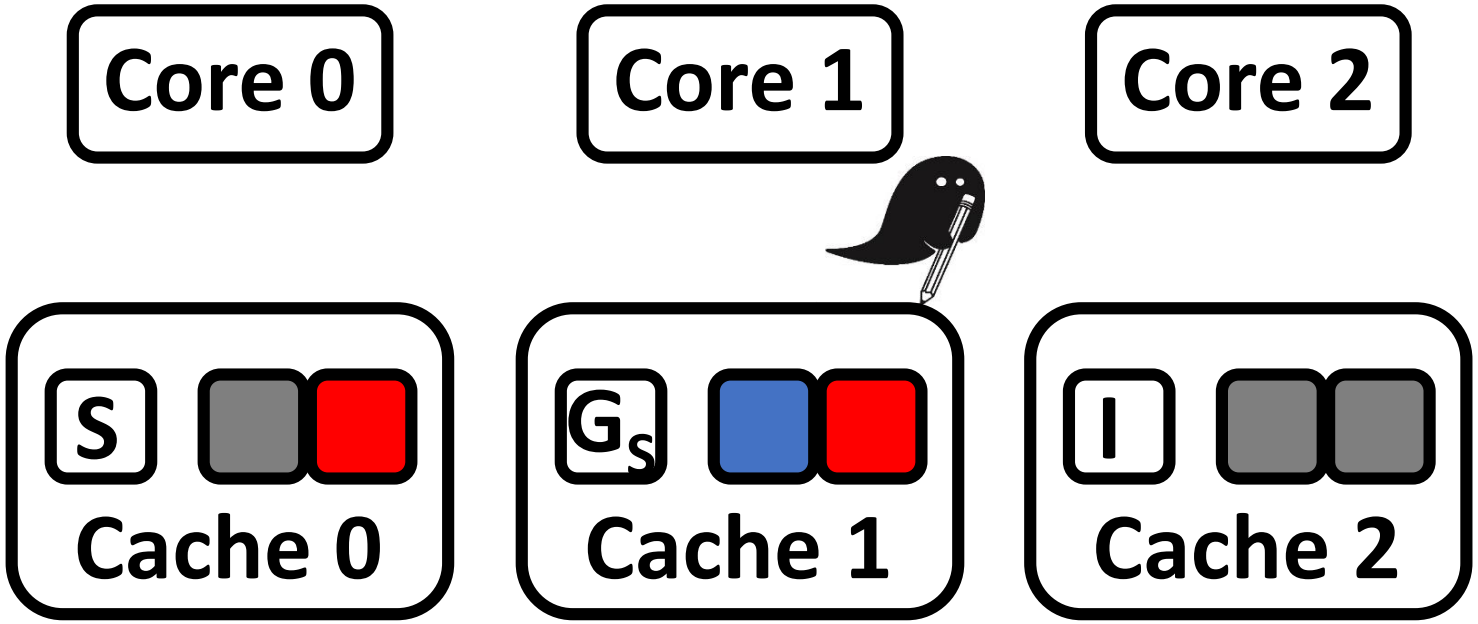
Example: Migratory Sharing (Ghostwriter)



Example: Migratory Sharing (Ghostwriter)

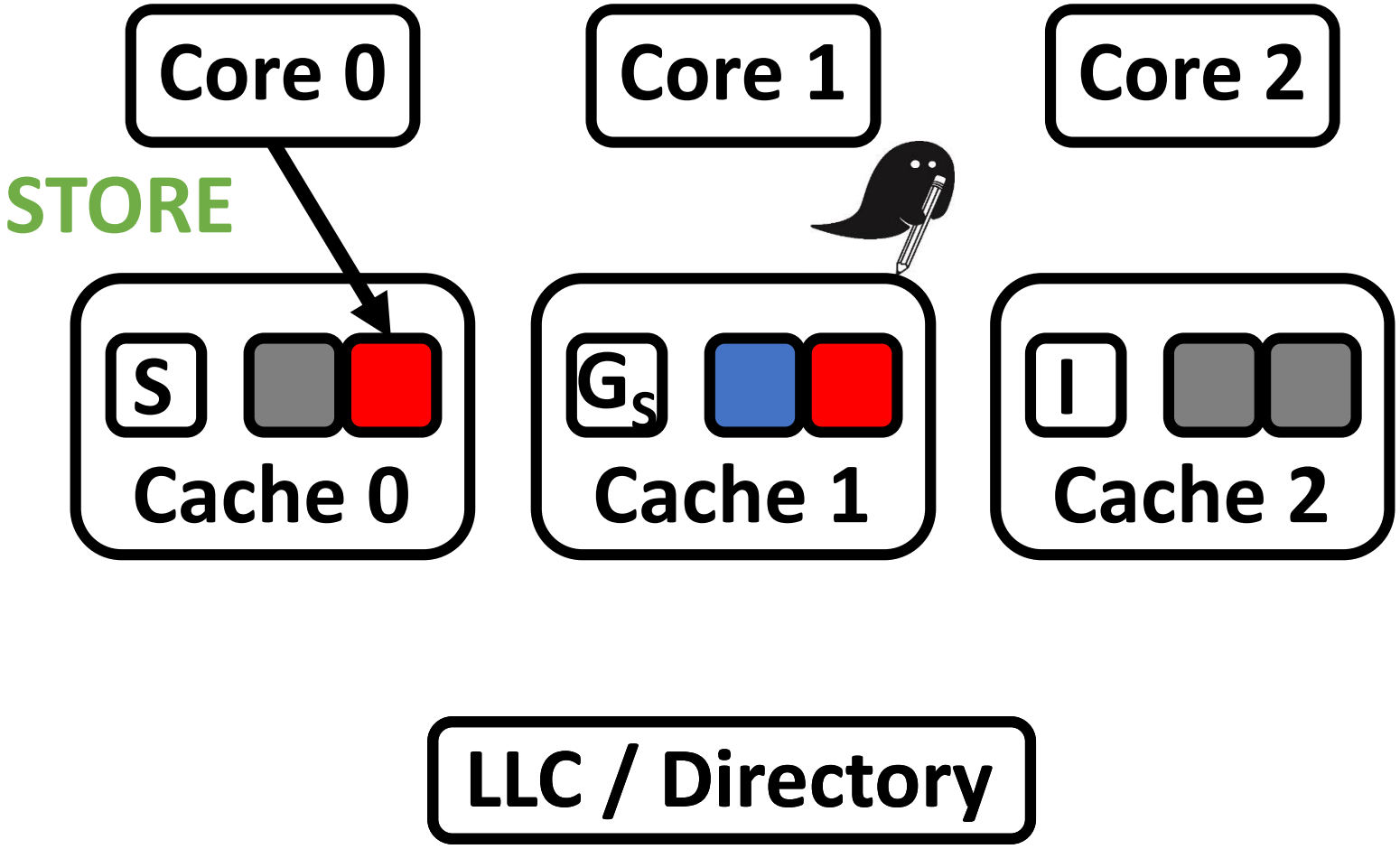


Example: Migratory Sharing (Ghostwriter)

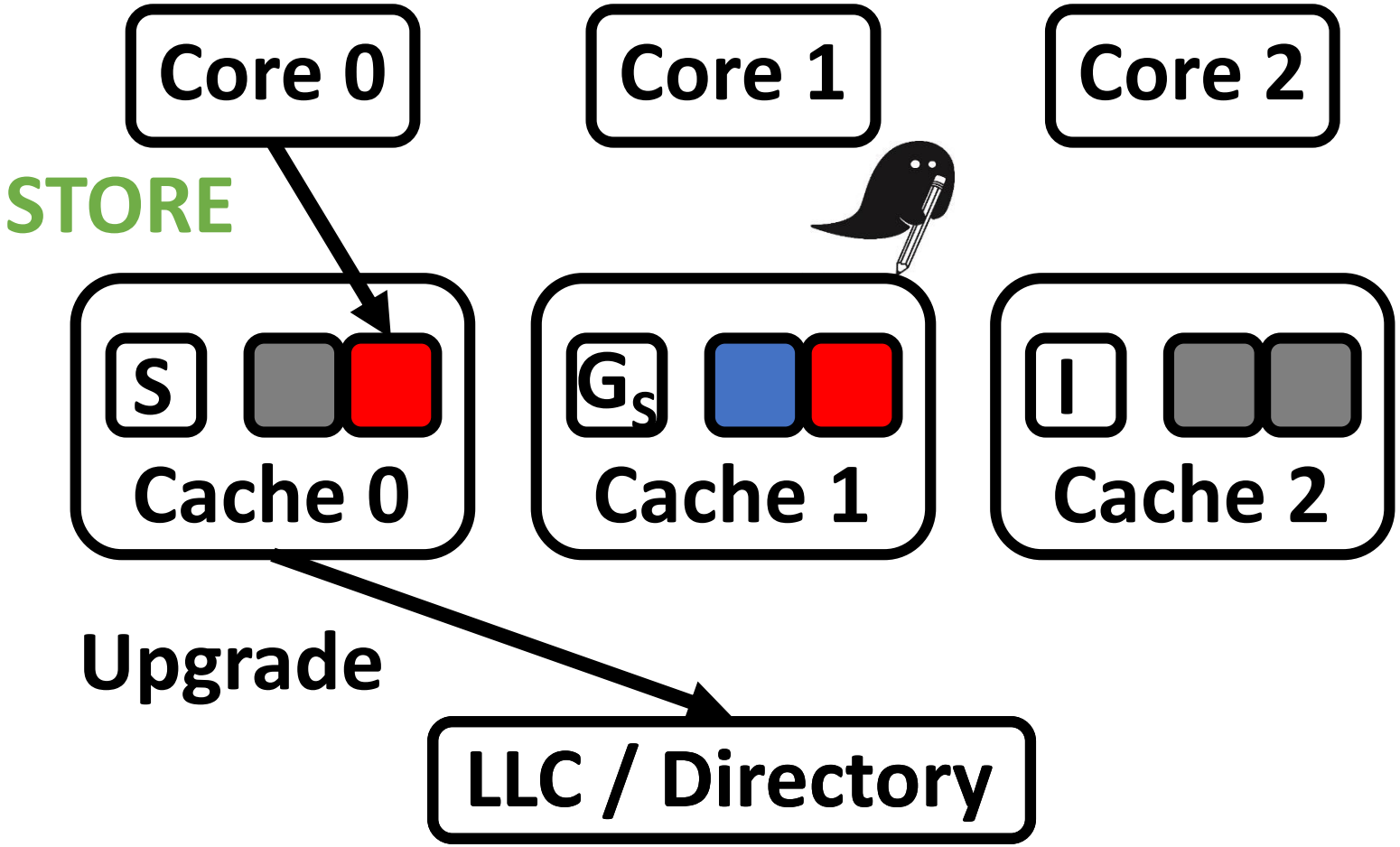


Saves coherence misses if data is approximate

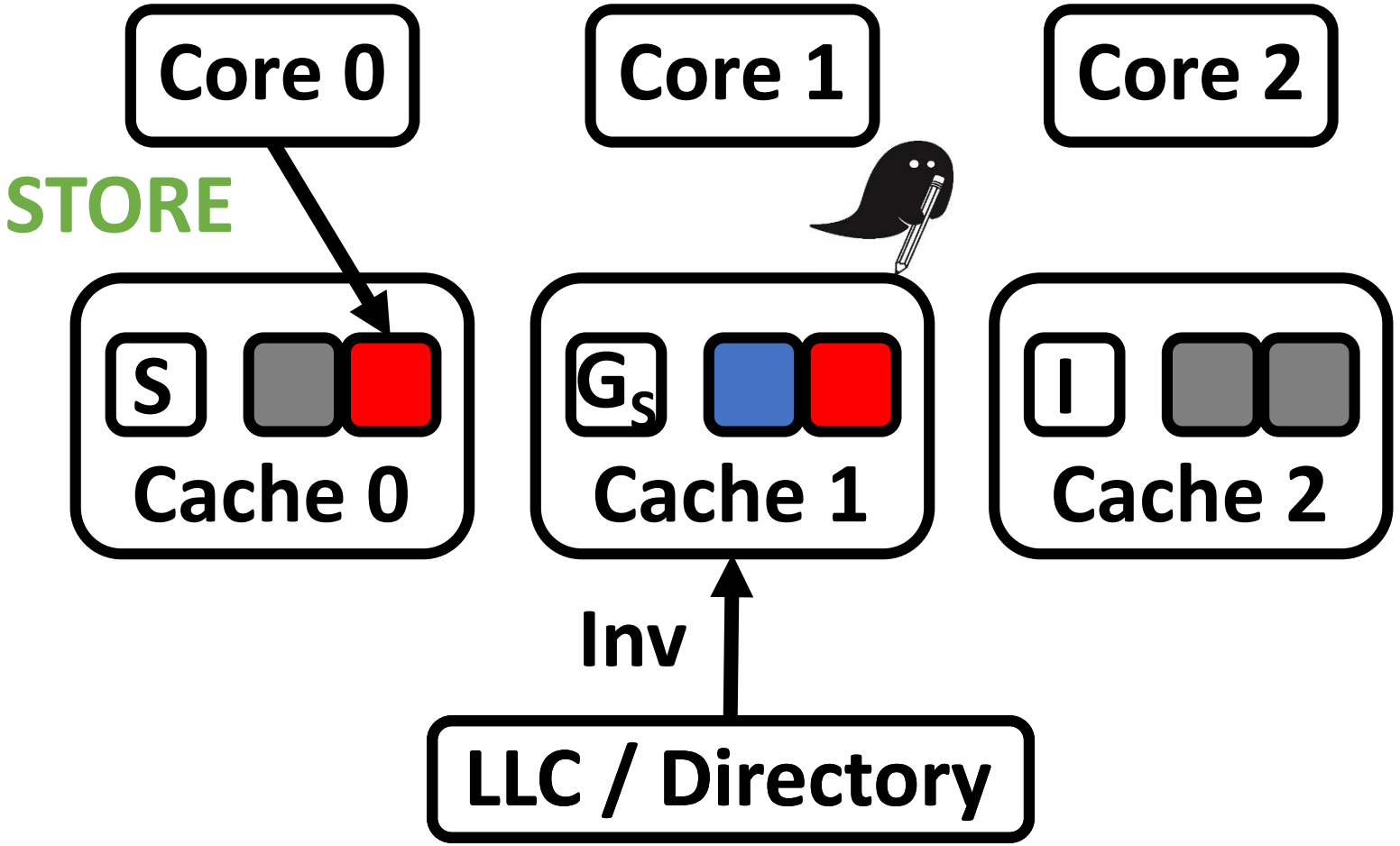
Example: Migratory Sharing (Ghostwriter)



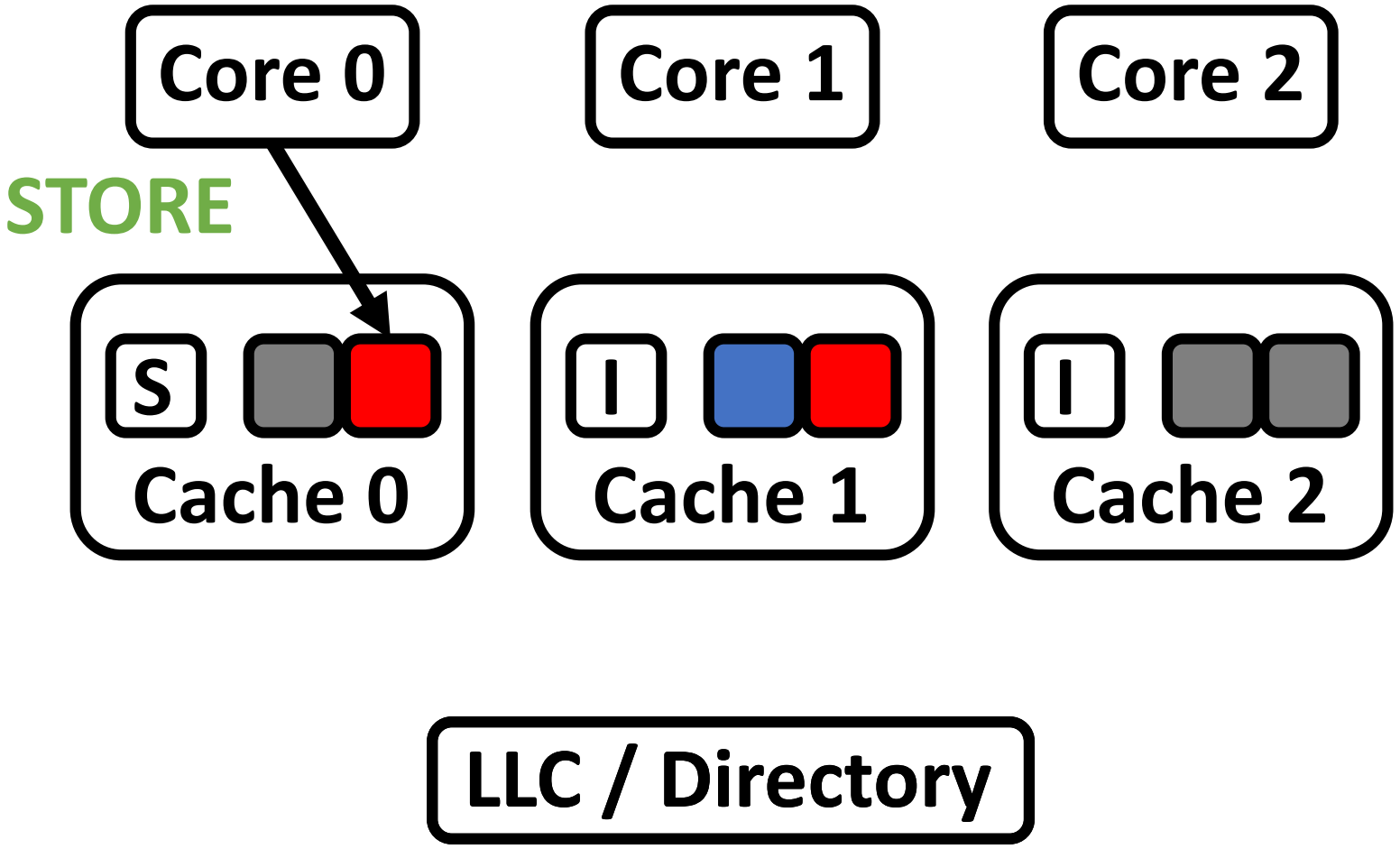
Example: Migratory Sharing (Ghostwriter)



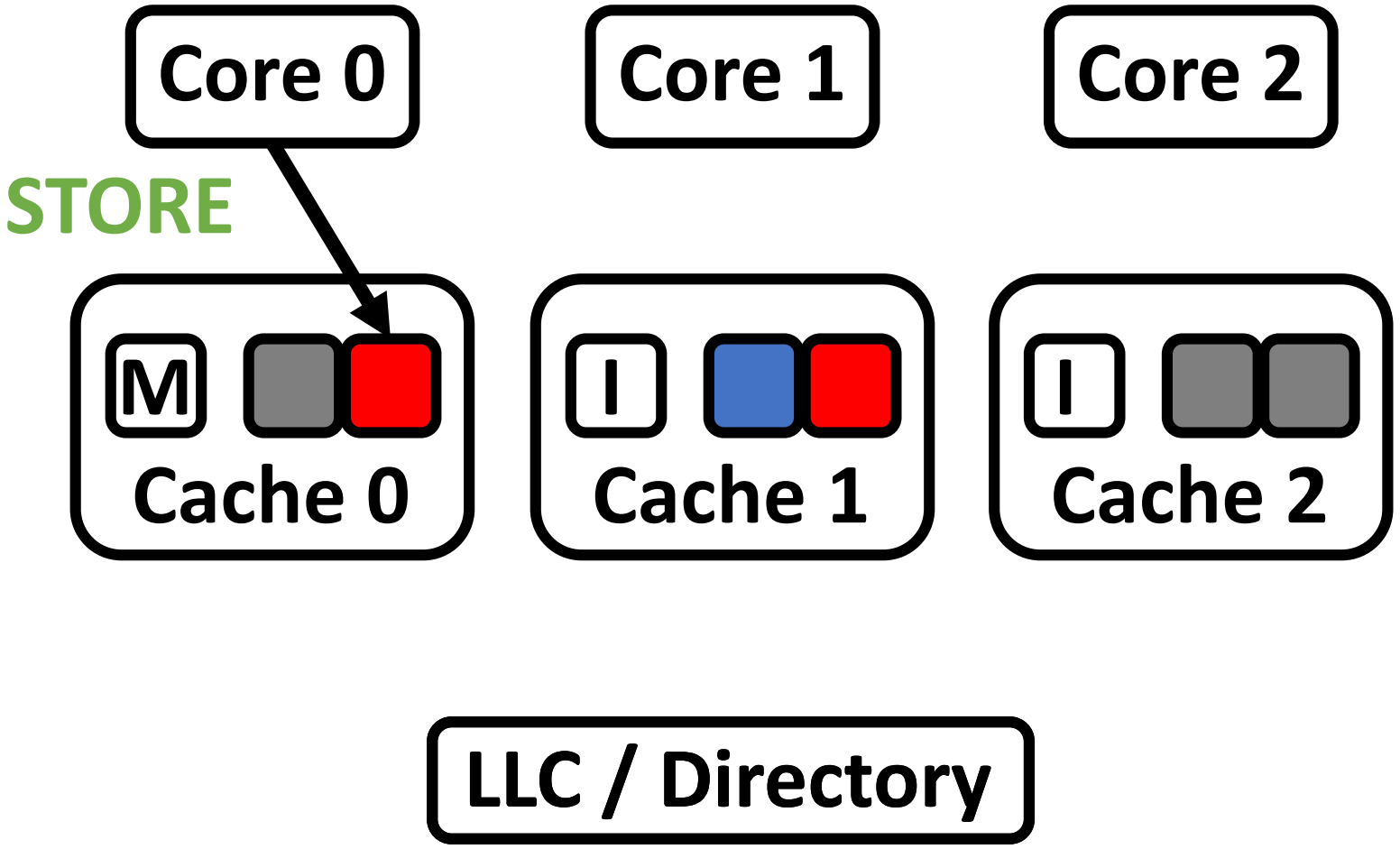
Example: Migratory Sharing (Ghostwriter)



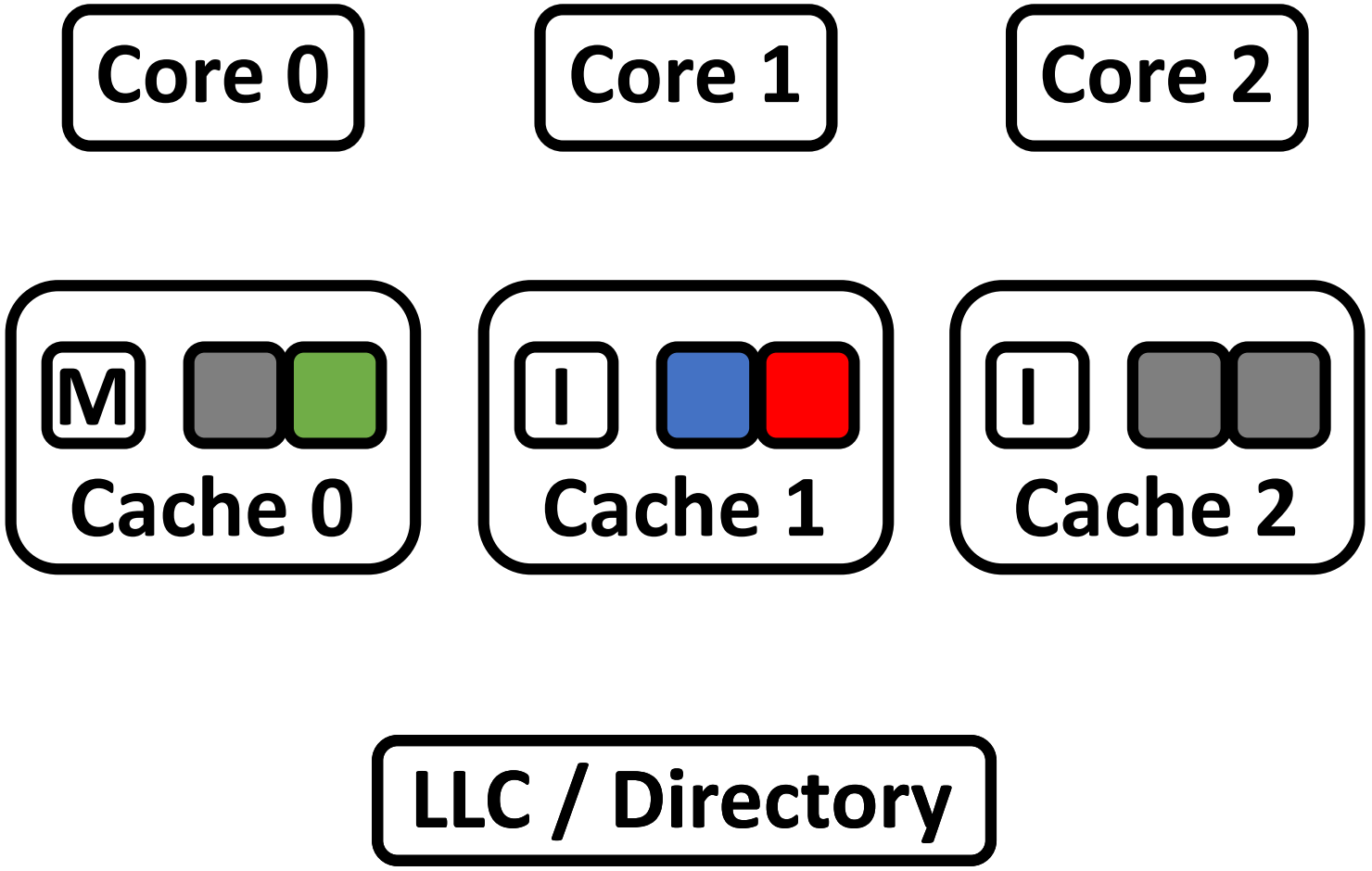
Example: Migratory Sharing (Ghostwriter)



Example: Migratory Sharing (Ghostwriter)

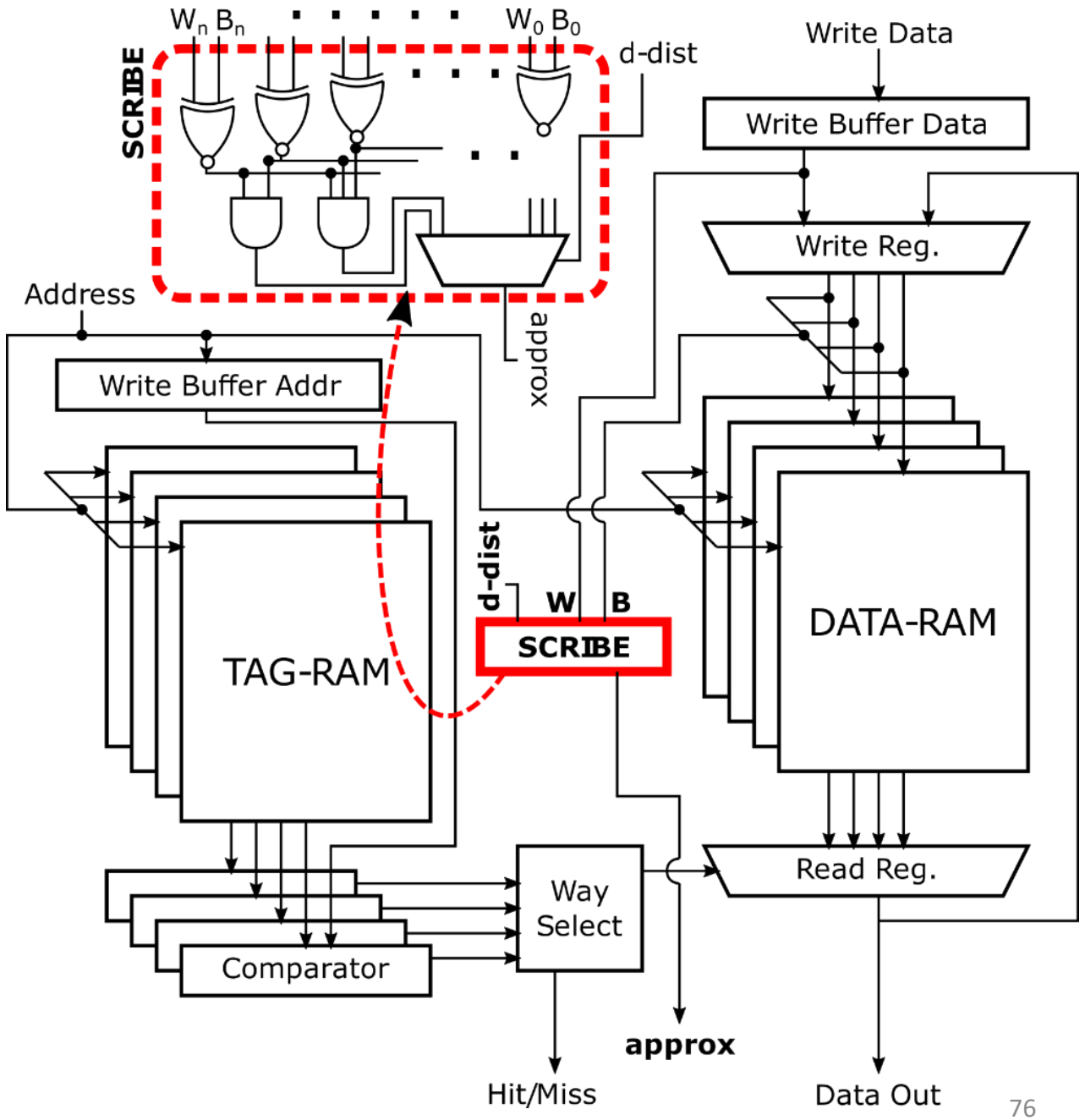


Example: Migratory Sharing (Ghostwriter)



Cache Controller

- Modification to cache controller **Scribe**
- MSb comparator to compare stores values with cache block values
- **Scribble** enabled if d-distance met



Ghostwriter

- Programmers annotate approximate memory addresses
- Approximate Store instructions determined by d-distance
- Approximate state transitions to G_I and G_S
- Ghostwriter adapts to different sharing patterns through approximation

Ghostwriter

Error

- Invalidation of G_S and Timeout of G_I
- Different cores update values in blocks in G_S/G_I state
- Approximated blocks that are replaced/evicted

Ghostwriter exploits value similarity on error-tolerant applications

Evaluation

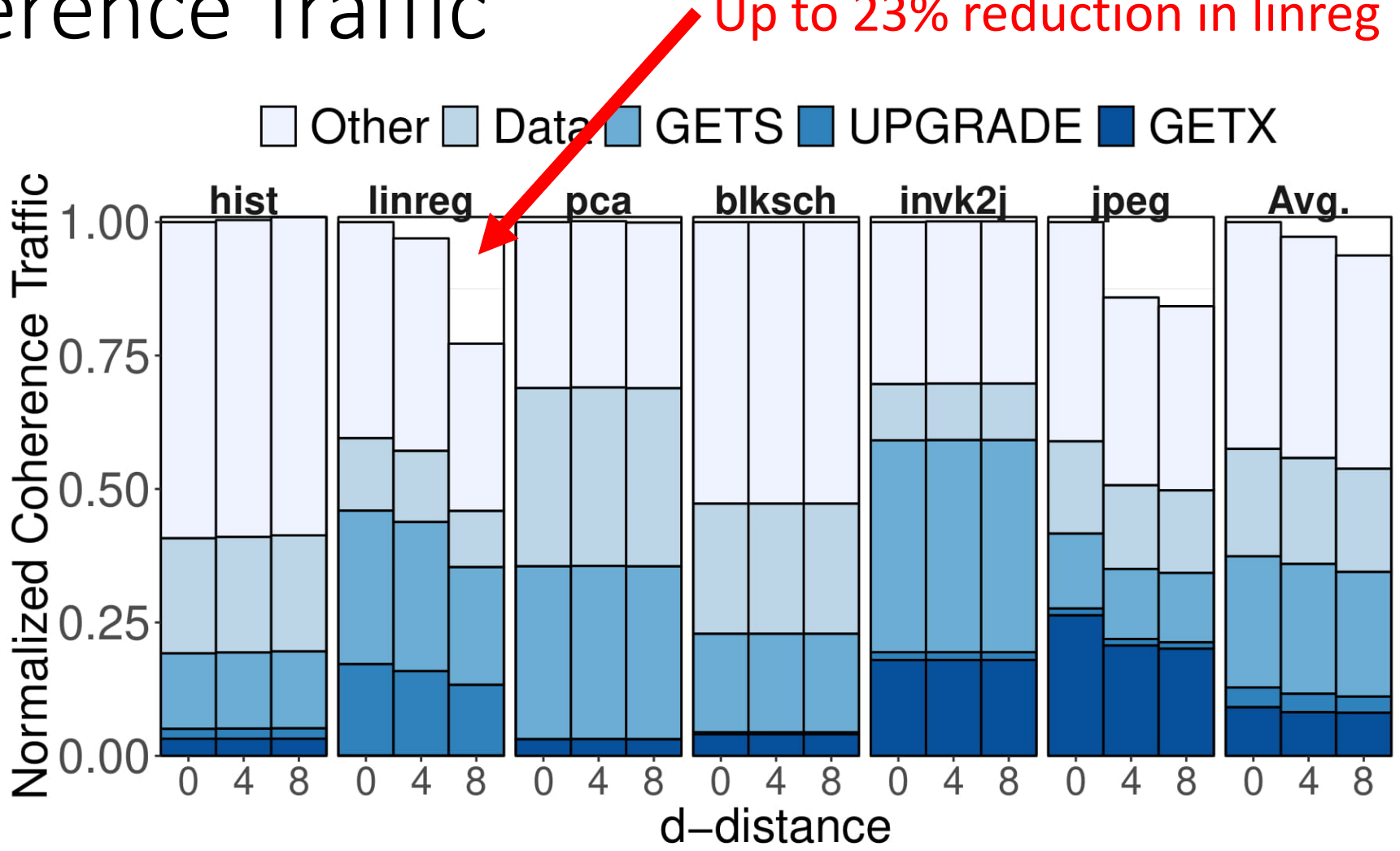
Simulator

- gem5 full-system
- Detailed memory and network models
- CACTI and DSENT for cache and network power

Machine

- 24 in-order cores
- Private 32kB I/D-caches, Shared L2 128kB per core.
- Ghostwriter Coherence (MESI baseline)
- d-distance {4,8}, G_I Timeout 1024 cycles

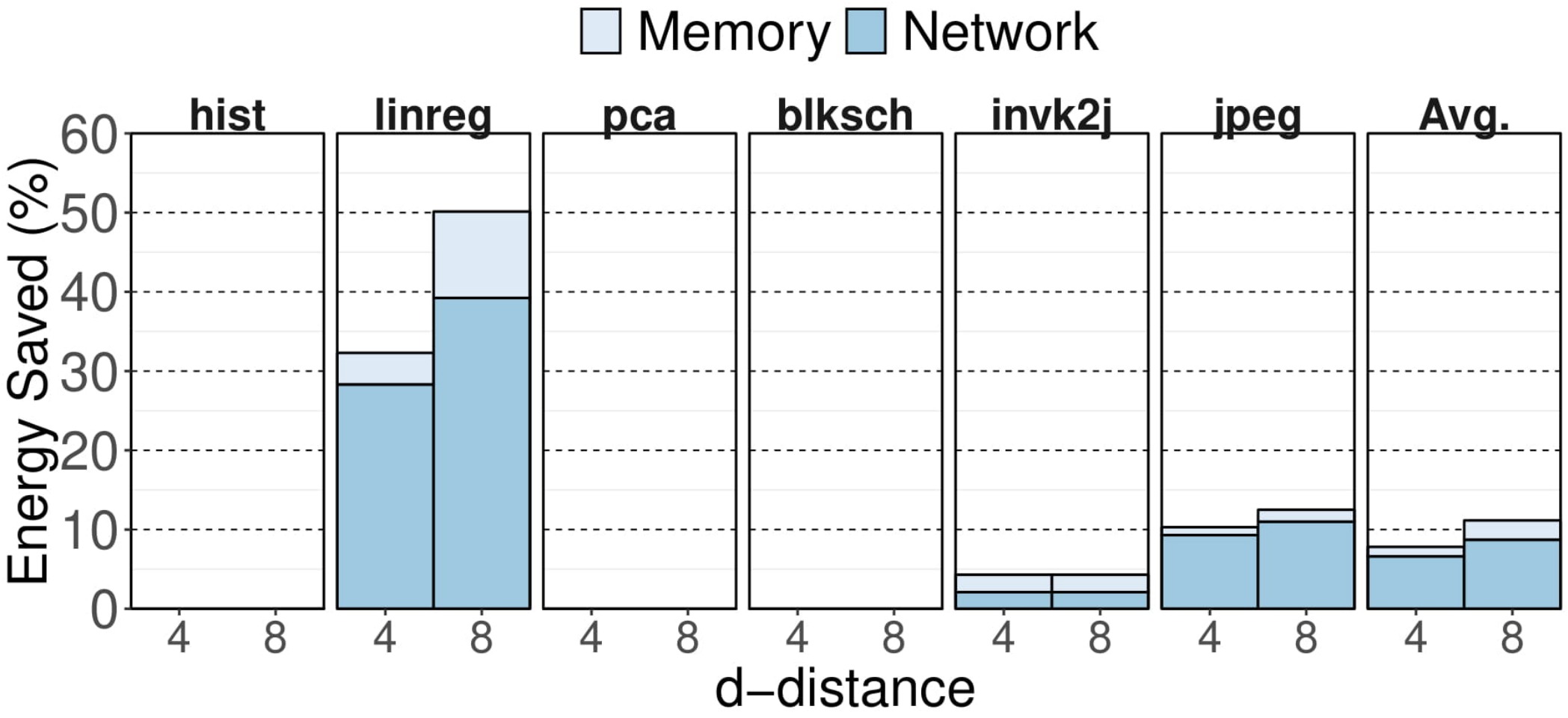
Coherence Traffic



Coherence traffic saved using Ghostwriter

Energy

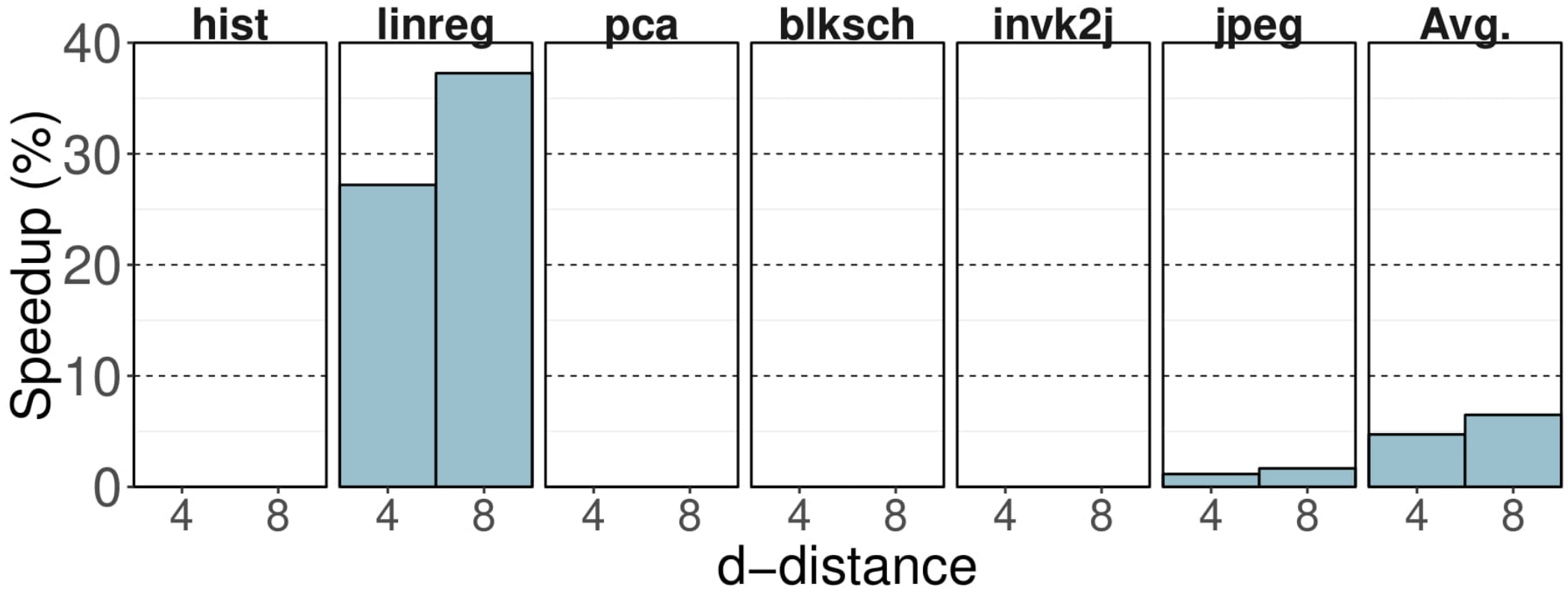
11% average dynamic energy saved



NoC and cache energy saved using Ghostwriter

Speedup

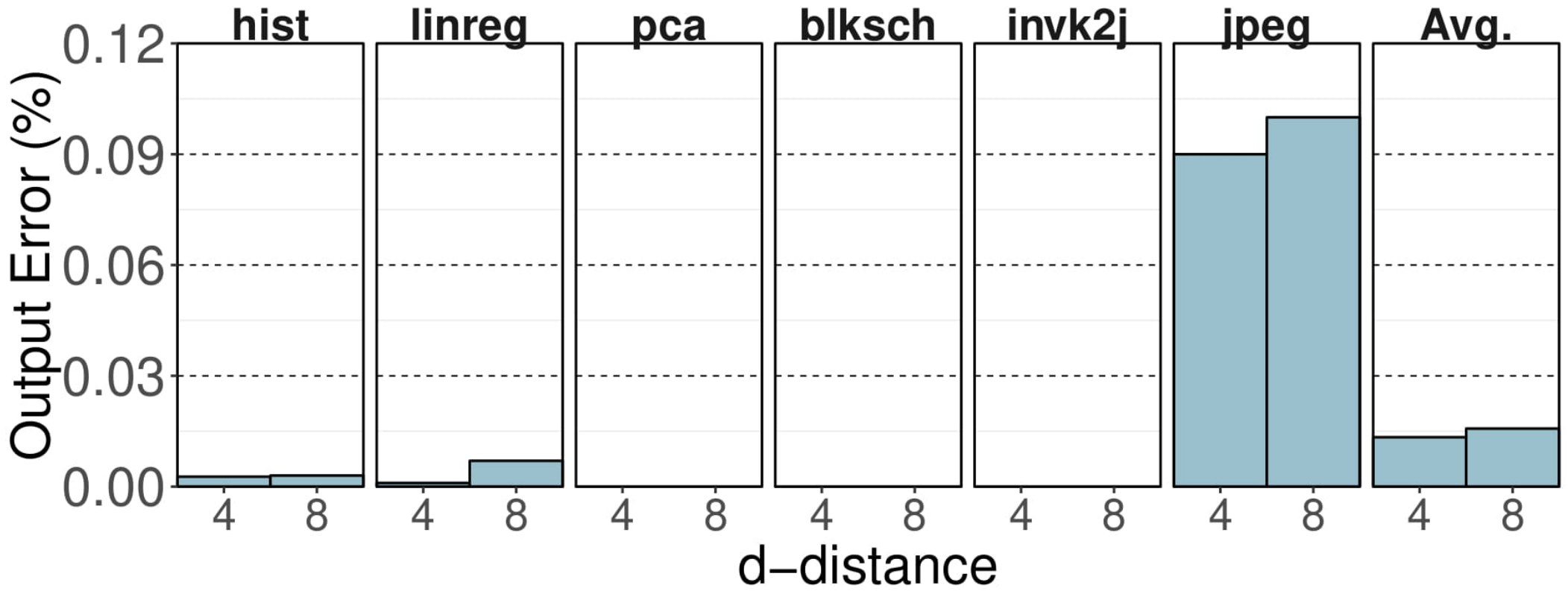
6.5% average speedup



Speedup gained using Ghostwriter

Output Error

Maximum of 0.10% error



Output error of applications using Ghostwriter

Summary

- Cache coherence is used to keep data coherent in multicores
- Communication of data is expensive within processors
- Limited power budget makes us look to new processor paradigms – Approximate Computing

Contribution: Ghostwriter Coherence Protocol

- Approximate stores instr. and approximate coherence states
- Up to 50.1% savings in dynamic energy
- Up to 37.3% speedup

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Ghostwriter: Cache Coherence for Error-Tolerant Applications

Henry Kao **Joshua San Miguel** **Natalie Enright Jerger**