Write Invalidate Protocol

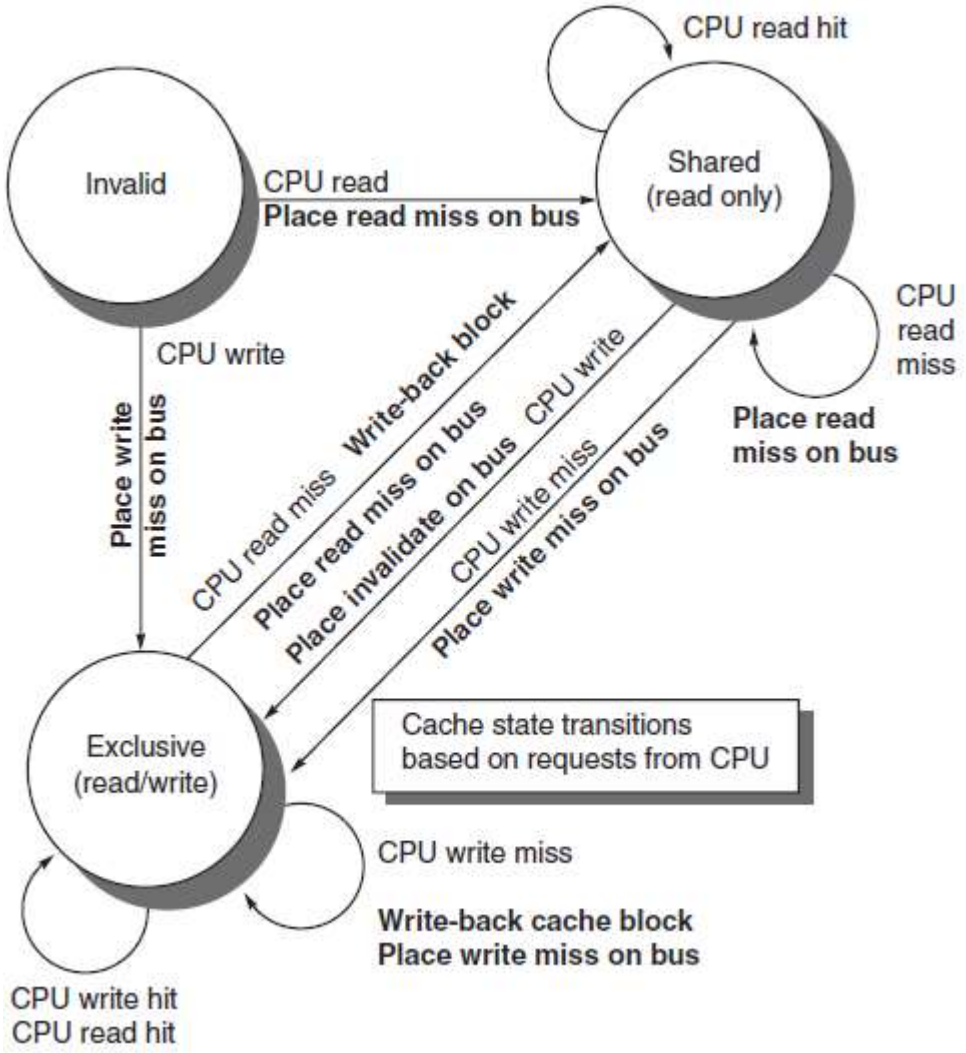
Processor activity	Bus activity	Processor A cache	Processor B cache	Memory location X
				0
Processor A reads X	Cache miss for X	0		0
Processor B reads X	Cache miss for X	0	0	0
Processor A writes 1 to X	Invalidation for X	1		0
Processor B reads X	Cache miss for X	1	1	1

- Alternative
  - Write update (or write broadcast)
  - Offers lower cache miss rate, but consumes considerably more bandwidth

MSI Protocol States

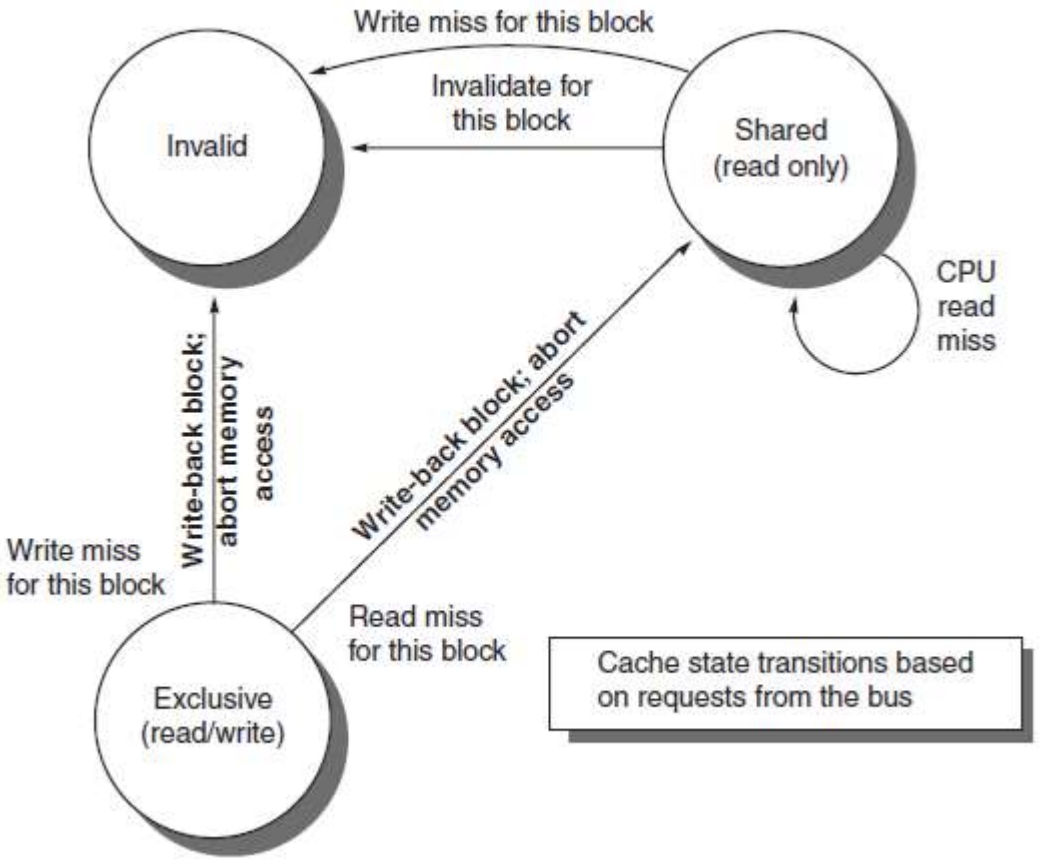
- Modified (M)
  - Block has been updated in this cache and is inconsistent with memory.
  - This cache has exclusive ownership and must supply data on a read miss.
- Shared (S)
  - Block is potentially shared with other caches.
  - The copy is clean (matches memory).
  - A write requires gaining exclusive access first.
- Invalid (I)
  - Block is not present in the cache or has been invalidated.
  - A read or write will cause a cache miss.

MSI Transitions Requested by CPU



Request	State	Cache action	Explanation
Read hit	S or M	Normal hit	Read data in local cache.
Read miss	I	Normal miss	Place read miss on bus.
Read miss	S	Replacement	Address conflict miss: place read miss on bus.
Read miss	M	Replacement	Address conflict miss: place read miss on bus.
Write hit	S	Coherence	Place invalidate on bus.
Write miss	I	Normal miss	Place write miss on bus.
Write miss	S	Replacement	Address conflict miss: place write miss on bus.
Write miss	M	Replacement	Address conflict miss: write-back block, then place write miss on bus.

MSI Transitions Requested by the Bus



Request	State	Cache action	Explanation
Read miss	S	No action	Allow shared cache or memory to service read miss.
Read miss	M	Coherence	Attempt to share data: place cache block on bus and change state to shared.
Invalidate	S	Coherence	Attempt to write shared block; invalidate the block.
Write miss	S	Coherence	Attempt to write shared block; invalidate the block.
Write miss	M	Coherence	Attempt to write block that is exclusive elsewhere; write-back the cache block and make its state invalid in the local cache.

Extensions

- MESI (additional Exclusive state)
  - If a block is in the E state, it can be written without generating any invalidates, which optimizes the case where a block is read by a single cache before being written by that same cache.
- MOESI (additional Owner state)
  - In MSI and MESI protocols, when there is an attempt to share a block in the Modified state, the state is changed to Shared (in both the original and newly sharing cache), and the block must be written back to memory.
  - In a MOESI protocol, the block can be changed from the Modified to Owned state in the original cache without writing it to memory.

Limitations of Snooping Protocols

Bandwidth Bottleneck

Shared bus becomes a bottleneck as processor count or speed increases

Limited Scalability

Practical limit of 4-8 high-performance cores with symmetric access

Broadcast Overhead

Every cache must process every coherence request

Directory Solution

Moving to directory-based protocols eliminates broadcast requirement

These limitations have led designers to adopt directory protocols for larger systems and hybrid approaches for intermediate-sized systems.