Coherence, Locking and Consistency- Part I

Dr. A. Sahu
CSE, IIT Guwahati

Multi Processor System : Basic Knowledge

- One should read in Primary Class
  - Basic notions: Coherence, lock, consistency
- Coherence
  - Shared data at all placed should be same
- Critical Sections
  - Lock and Barrier Design
- Consistency
  - Order should be similar to serial (ROB)

Shared Memory with Caches

Multiple copies of data may exist
⇒ Problem of cache coherence
Cache coherence protocols
- What action is taken?
- Which processors/caches communicate?
- Status of each block?

What action is taken?

- Invalidate other caches and/or memory
  - Send a signal/message immediately, copy information only when unavoidable
  - Similar to write back policy
- Update other caches and/or memory
  - Write simultaneously at all places (send modifications immediately)
  - Similar to write through policy

Which proc/caches communicate?

- Snoopy protocol
  - Broadcast invalidate or update messages
  - All processors snoop on the bus
- Directory based protocol
  - Maintain directory - list of copies
  - Communicate selectively
  - Directory - centralized (memory) or distributed (caches)
**Status of each cache block?**
valid/invalid private/shared clean/dirty

- Simplest protocol (3 states)
  Invalid, (shared) clean, private dirty
- Berkeley protocol (4 states)
  Invalid, (shared) clean, private dirty, shared dirty
- Illinois protocols (4 states)
  Invalid (I), shared clean(S), private clean (E), private dirty (M) : MESI
- MOESI : Owner responsible for copy back to memory

**Simplest invalidation protocol**
Use 3 states : Invalid, shared clean, private dirty

**Read Miss in 3 State Protocol**

**Write Hit in 3 State Protocol**

**Write Miss in 3 State Protocol**
**Simplest invalidation protocol**

*Use 3 states: Invalid, shared clean, private dirty*

- **Invalid**: WR miss, INV
- **Shared clean**: RD miss
- **Private dirty**: WR miss, INV

**CPU event**

**BUS event**

**Are 3 states sufficient? No**

- **All writes in clean state cause invalidation traffic** (whether shared or not)
  - Introduce private clean state (e.g. Illinois protocol) - reduce invalidation traffic
- **Clean copies are always supplied by memory**
  - Introduce shared dirty state (e.g. Berkeley protocol) - permit copies to be supplied by caches which own the required blocks
  - Illinois protocol also permits a cache to supply the copy, but if it is coming from a dirty state, memory is also updated

**4-state Berkeley protocol**

*Use 4 states: Invalid, (shared?) clean, private dirty, shared dirty*

**CPU event**

**BUS event**

**Read Miss in Berkeley Protocol**

*Same as 3-state protocol*

- **Any**: C, C, C, C

Preceded by writing back a block if previous state was "dirty (private or shared)"

**If some other cache has the latest copy**

Permitting copies to be supplied by caches which own the required blocks
Write Hit in Berkeley Protocol

- If some other cache also has a copy.

Write Miss in Berkeley Protocol

- If no other cache has the latest copy.

Any

preceded by writing back a block if previous state was "dirty (private or shared)"

Write Miss in Berkeley Protocol

- If some other cache has the latest copy.

Any

preceded by writing back a block if previous state was "dirty (private or shared)"