Lesson 09:
Cache Coherence Problem and Cache synchronization solutions— Part 1
Objective

- To understand cache coherence problem
- To learn the methods used to solve it
- To understand the synchronization mechanisms for maintaining cache coherence
- Cache-coherence problem protocol MESI
Disadvantage of Bus shared systems
Disadvantage of Bus shared systems

- Only some of the data in the memory directly accessible by each processor, since a processor can only read and write its local memory system.
- Accessing data in another processor’s memory requires communication through the network.
Possibility that two or more copies of a given datum

- Two or more copies could exist in different processors’ memories
- The copies may result because of data sharing, because of process migration from one processor to another, or because of an I/O operation
Cache coherence problem
Cache coherence problem

- When two or more copies of a given datum exist in different processors’ memories, it may lead to different processors having different values for the same variable.
- Major source of complexity in shared-memory systems.
Cache coherence problem

• Problem of inconsistency between a cached copy and the shared memory or between cached copies themselves due to the existence of multiple cached copies of data
Cache Coherence Protocols— snooping bus protocols
Maintaining cache coherence using cache snooping

- Cache snooping easy in a bus-based multiprocessor
- Each processor in the system can observe the state of the memory bus, called cache snooping
- Cache snooping—allows each processor to see any requests that other processors make to the main memory
Cache coherence protocol of a shared memory multiprocessor

- Defines how the data may be shared and replicated across processors
Memory-coupling (consistency) model

- Defines when the programs running on the processors will see operations executed on other processors.
Cache coherence protocol

- Defines the specific set of actions that are executed to keep each processor’s view of the memory system consistent.
- Operates on cache lines of data at a time, communicating an entire line between processors when necessary, rather than just sending a single word.
Two categories of Cache-coherence protocols

- Snoopy bus protocols for shared buses
- Directory based protocols for the multistage networks
Two categories of snooping bus protocols

- Invalidation-based protocols
- Update-based protocols
Invalidation based cache protocol

<table>
<thead>
<tr>
<th>Events</th>
<th>Processor $P_1$</th>
<th>Processor $P_2$</th>
<th>Processor $P_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$ reads line</td>
<td>No Copy</td>
<td>No Copy</td>
<td>No Copy</td>
</tr>
<tr>
<td>$P_2$ reads line</td>
<td>Read-Only Copy</td>
<td>No Copy</td>
<td>No Copy</td>
</tr>
<tr>
<td>$P_3$ writes line (other copies invalidated)</td>
<td>Read-Only Copy</td>
<td>Read-Only Copy</td>
<td>No Copy</td>
</tr>
<tr>
<td>$P_2$ reads line</td>
<td>No Copy</td>
<td>Read-Only Copy</td>
<td>Writable copy</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Invalidation based protocols— MESI Protocol
MESI Protocol

- Commonly used invalidation-based cache-coherence protocol
- Each line in a processor’s cache assigned one of four states to track which caches have copies of the line
Four possible states of a cache line in MESI Protocol

- Cache line state—M, E or S or I (Modified, Exclusive, Shared, or Invalid)
Invalidation based cache protocol
State I

- The invalid state means that the processor does not have a copy of the line.
- Any access to the line will require that the shared-memory system send a request message to the memory that contains the line to get a copy of the line.
State S

- The shared state means that the processor has a copy of the line, and that one or more other processors also have copies.
- The processor may read from the line, but any attempt to write the line requires that the other copies of the line be invalidated.
State E

- If a line is in the exclusive state, the processor is the only one that has a copy of the line, but it has not written the line since it acquired the copy.
Advantage of State E

- The definition of the exclusive unmodified state of data entirely avoids the transition to invalidations on write operations to unmodified non-shared blocks
State M

- The modified state means that the processor is the only one with a copy of the line, and it has written the line since it acquired the copy.
- In both the exclusive and modified states, the processor may read and write the line freely.
Different snoopy bus protocols
## The states in different snoopy bus protocols

<table>
<thead>
<tr>
<th>Protocol</th>
<th>MESI</th>
<th>Write Once</th>
<th>Berkeley</th>
<th>Synapse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modified</td>
<td>Write Once#</td>
<td>Dirty (data is valid, shared with no other cache, and not consistent with main memory -- write-back needed)</td>
<td>Shared Dirty</td>
<td>Dirty (not consistent with main memory)*</td>
</tr>
<tr>
<td>Exclusive</td>
<td>Reserved</td>
<td>Reserved</td>
<td>Exclusive Dirty</td>
<td>Dirty (data valid but shared with no other cache)*</td>
</tr>
<tr>
<td>Shared</td>
<td>Valid but may be shared with other cache</td>
<td>Valid</td>
<td>Valid</td>
<td>Valid and may be shared with other caches</td>
</tr>
<tr>
<td>Invalid</td>
<td>Invalid Writable copy</td>
<td>Invalid Dirty (block is the owner of that block.)*</td>
<td>Invalid Writable copy</td>
<td></td>
</tr>
</tbody>
</table>
The meaning of terms—dirty and invalid

# A block is reserved if it has been modified exactly once. It becomes dirty if it is modified more than once.
$If a block is not owned by any cache, memory is the owner—in any case, the owner supplies the block on a miss.
Summary
We Learnt

- Cache coherence problem
- Maintenance of identical copies when a copy is written and two or more copies already exist
- Snoopy bus protocols
- Invalidation based snoopy bus protocols
End of Lesson 04 on
Cache Coherence Problem and Cache
synchronization solutions— Part 1