מבנה מחשבים ספרתיים
234267

MESA Protocol: 5 מס"ל תרגול
A memory system is **coherent** if

1. If P1 writes to address X, and later on P2 reads X, and there are no other writes to X in between
   ⇒ P2’s read returns the value written by P1’s write

2. Writes to the same location are serialized:
   two writes to location X are seen in the same order by all processors
Mesi Protocol

- Each cache line can be in one of 4 states

  - Invalid – Line’s data is not valid

  - Shared – Line is valid and not dirty, copies may exist in other processors

  - Exclusive – Line is valid and not dirty, other processors do not have the line in their local caches

  - Modified – Line is valid and dirty, other processors do not have the line in their local caches
## MESI Protocol States

<table>
<thead>
<tr>
<th>State</th>
<th>Valid</th>
<th>Modified</th>
<th>Copies may exist in other processors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid</td>
<td>No</td>
<td>N.A.</td>
<td>N.A</td>
</tr>
<tr>
<td>Shared</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Exclusive</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Modified</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

- **To modify a line it mustn’t exist in other processors**
  - Otherwise, another processor which has the line will be using stale data
  - Therefore, before modifying a line, a processor must request ownership of the line
Multi-processor System: Example

- P1 reads 1000
- P1 writes 1000
Multi-processor System: Example

- P1 reads 1000
- P1 writes 1000
- P2 reads 1000
- L2 snoops 1000
- P1 writes back 1000
- P2 gets 1000
Multi-processor System: Example

- P1 reads 1000
- P1 writes 1000
- P2 reads 1000
- L2 snoops 1000
- P1 writes back 1000
- P2 gets 1000
- P2 requests for ownership with write intent
Core Valid Bits and Inclusion

- L2 keeps track of the presence of each line in each of the cores’ L1 caches
  - Determine if it needs to send a snoop to a processor
  - Determine in what state to provide a requested line (S,E)
  - Maintain Core Valid Bits (CVB) per cache line
  - Need to guarantee that the L1 caches in each core are contained in the L2 cache

- When L2 evicts a line
  - L2 sends a snoop invalidate to all processors that have it
  - If the line is modified in the L1 cache of one of the processors (in which case it exists only in that processor)
    - The processor responds by sending the updated value to L2
    - When the line is evicted from L2, the updated value gets written to memory
A four-processor shared-memory system implements MESI protocol.

For the following sequence of memory references, show the state of the line containing the variable X in each processor’s cache after each reference is resolved.

All processors start out with the line containing X invalid in their cache.

<table>
<thead>
<tr>
<th></th>
<th>P0’s state</th>
<th>P1’s state</th>
<th>P2’s state</th>
<th>P3’s state</th>
<th>CVBs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial State</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>0000</td>
</tr>
<tr>
<td>P0 reads X</td>
<td>E</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>1000</td>
</tr>
<tr>
<td>P1 reads X</td>
<td>S</td>
<td>S</td>
<td>I</td>
<td>I</td>
<td>1100</td>
</tr>
<tr>
<td>P2 reads X</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>I</td>
<td>1110</td>
</tr>
<tr>
<td>P3 writes X</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>M</td>
<td>0001</td>
</tr>
<tr>
<td>P0 reads X</td>
<td>S</td>
<td>I</td>
<td>I</td>
<td>S</td>
<td>1001</td>
</tr>
</tbody>
</table>