In Modified MOESI we have tweaked the transition to Owner state. Thereby, now there is a transition from Exclusive state to the Shared owner and not to the Owner state. Thus, Shared owner state is guaranteed to have clean data. The original transition in MOESI from Modified state to Owner state is retained as it is. Thus it is guaranteed that Owner state will have dirty data since it is obtained from Modified state. That means a particular processor will write and modify the data and when other processors’ cache ask for that same location then the processor’s cache which has the latest updated copy becomes the owner for that block and supplies it to requesting cache.

Thence there is a dedicated state for clean and dirty data. This is achieved by the introduction of Shared Owner state. Thus, in longer runs of instructions there would no need to keep a track of data whether its clean or dirty since they have dedicated states. The protocol state diagram becomes a bit complicated but it is OK to increase the complexity as long as it improves the performance.

All of this can be better handled if there is some instruction prediction mechanism in the system. It will depend on the application for which the system is used. It solely depends on the number of Loads and Stores in the instruction list.

The instruction format defined for this design is given below, a 32 bit address coming from the Processor is broken into 23 bit Tag, 5 bit Set and 2 bit Word & Byte. Cache designed for this project is 2-way set associative cache such that size of local cache for each processor is 1MB.

### Results

<table>
<thead>
<tr>
<th>Tag (23 bits)</th>
<th>Set (5 bits)</th>
<th>Word (2 bits)</th>
<th>Byte (2 bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>31 to 9</td>
<td>8 to 4</td>
<td>3 to 2</td>
<td>1 to 0</td>
</tr>
</tbody>
</table>

For the MOESI protocol, Owner state has data ambiguity as there are two entry point, one from Modified state (Dirty Data) and other from Exclusive state (Clean Data). To clear this ambiguity, a new state is added to MOESI protocol called SO - Shared Owner (Shared Clean) state and the state machine for the new protocol is represented below.

MOESI protocol has a state machine representation, as shown below, each state signifies the cache line’s status, which is derived from the instructions executed by the processors. The states in MOESI protocol are: M – Modified (Exclusive Dirty), O – Owner (Shared & Clean/Dirty), E – Exclusive (Exclusive Clean), S – Shared (Clean), I – Invalid.

Here we are demonstrating modified version of MOESI cache coherence protocol for a quad-core processor architecture. Each core has its own private L1 cache and all the cores have a shared L2 cache. Following Figure depicts the architecture of multi-processor system employed in this project.

Acknowledgements

I am grateful to all those who helped me through this project in terms of academic and moral support. First of all I would like to thank my project advisor, Prof. Morris Jones for his support, guidance and unfounded attention towards making this project a success. Without him, it would have not been possible to complete this project in timely manner as required. I am grateful to him for creating an UNIX account for my project on his server so that I wouldn’t face any host server performance issues during semester crunch time. I would also like to take this opportunity to thank San Jose State University, Electrical Engineering Department and the iECSAD Lab.