MIPS R4000 Caches and Coherency

Paul Ries

R4000 Primary Caches

- 8KB to 32KB
- Direct-mapped
- Virtual index, Physical Tag
- Pipelined
- Write-back
Direct-mapped Primary Caches

Direct-mapped load delay = 2 cycles

Set-associative load delay = 3 cycles or slower cycle time

EX - ALU
DF - D-cache 1st
DS - D-cache 2nd
TC - Tag Check

Virtual Index, Physical Tag

Physical Tag
- No cache flush on map change
- Simple sharing of same physical location mapped multiple times

Virtual Index
- TLB in parallel with cache
- No extra pipe stages required
- Alias problem for same physical location mapped multiple times
R4000 Secondary Cache

- External, standard asynchronous SRAMs
- 256KB to 4MB
- Direct-mapped
- Physical index, Physical Tag
- 128b Wide
- ECC data protection

R4000 Cache Coherency

Multiprocessor Interconnect Network

almost anything user ASIC

snoopy bus write-invalidate
cross-bar switch write-update
butterfly switch

low-level, generic interface

Read Non-coherent
Read Coherent
Read Coherent Exclusive
Write
Invalidate
Update

Read Data
Read Data, Shared
Snoop Request
Intervention Request
State Change
Invalidate
Update
R4000 Cache Coherency States

Primary caches are forced to be a subset of Secondary cache

- s-cache misses check and invalidate the primaries if necessary

- Received Snoop, Interventions, Invalidates, and Updates cause R4000 to check S-cache in parallel without interrupting program execution
  - s-cache is physical index, physical tag, so snoops are trivial

- Misses in the S-cache require no further action
  - no duplicate tags required, because s-cache serves as a filter

- Hits may require access to the primary caches to complete the transaction, but the primary caches are virtually indexed, so we store the virtual index in the s-cache tag
R4000 Cache Coherency Options

- Page Table Entries select one of several cache algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Load Miss</th>
<th>Store Miss</th>
<th>Store Hit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncached</td>
<td>Word Read</td>
<td>Word Write</td>
<td>-</td>
</tr>
<tr>
<td>Non-coherent exclusive</td>
<td>Read Non-coherent</td>
<td>Read Non-coherent</td>
<td>-</td>
</tr>
<tr>
<td>Coherent exclusive</td>
<td>Read Exclusive</td>
<td>Read Exclusive</td>
<td>-</td>
</tr>
<tr>
<td>Coherent write exclusive</td>
<td>Read</td>
<td>Read Exclusive</td>
<td>-</td>
</tr>
<tr>
<td>Coherent write update</td>
<td>Read</td>
<td>Read / Update</td>
<td>Update</td>
</tr>
</tbody>
</table>

- Protocol Interface ASIC can control some state transitions, thus effecting a subset protocol

R4000 Synchronization

LL/SC instructions provide synchronization between processes based only on cache coherency

Example (fetch-and-add):

Loop:

```
LL T0, 0(T1)     load counter
ADDU T1, T0, 1   increment
SC T1, 0(T1)     store back if unchanged
BEQ T1, 0, Loop  retry if store failed
```

SC fails if the location has been invalidated or updated since preceding LL

Can implement semaphores, bit-locks, fetch-and-add, etc. with the LL and SC primitives