Towards Thousand-Core RISC-V Shared Memory Systems

Quan Nguyen
Massachusetts Institute of Technology
30 November 2016
Outline

• The Tardis cache coherence protocol
  – Example
  – Scalability advantages

• Thousand-core prototype

• RISC-V and Tardis
Tardis

• A new cache coherence protocol
• Enforces consistency through timestamps
• Key idea: logical leases
  – Timestamps $\text{wts}$ and $\text{rts}$: start and end of lease
  – Processors keep timestamp $\text{pts}$ in register

Xiangyao Yu and Srinivas Devadas, “Tardis: Time Traveling Coherence Algorithm for Distributed Shared Memory”, PACT 2015.
Metadata per cache line

- **Client:**
  - Use ordinary MESI states

- **Manager:**
  - Tracks only the exclusive owner
  - Tracks most recent lease
Example

- Core 0 stores A
- Manager leases A
  - Sets owner, wts, rts
- Core 0:
  - Gets A with \([\text{wts}, \text{rts}] = [0, 0]\)
  - Writes A’, creates new version at [1, 1]
  - Updates its pts to 1
Example

- Core 0 loads B
  - Sends pts to manager
- Manager leases B
  - Sets lease based on pts
  - \([\text{wts}, \text{rts}] = [1, 11]\)
- Core 0 reads B at pts 1
Core 0:          Core 1:
  store A      store B
  load  B      load  A
  load  B

Example

- Core 1 stores B
  - Sends pts to manager
- Manager advances past Core 0’s lease
  - \([\text{wts}, \text{rts}] = [12, 12]\)
- Instantly grant Core 0 exclusive ownership
- Core 1 writes B’ at pts 12
- Different versions of B coexist!
Example

- Core 1 loads A
- Manager sends Core 0 writeback request
- Core 0 downgrades
- Core 1 receives new lease based on its pts – [wts, rts] = [12, 22]
- Core 1 reads A’ at pts 12
Example

- Core 0 loads B
- Cache hit; simply loads B from data cache
- Sequential order ≠ physical order
A case for scalability

• Track only one node: $O(\log N)$ storage
• No broadcast invalidations
• Timestamps not tied to core count
  – No need for synchronized real-time clocks
  – Can be compressed
Outline

• The Tardis cache coherence protocol
• Thousand-core prototype
• RISC-V and Tardis
Thousand-core shared memory systems

- Fit as many cores will fit on a ZC706
- Connect in a 3D mesh
  - Aurora links, six connectors per board
- Demonstrate shared memory at scale
- Name: T-1000
Tardis and RISC-V

- RISC-V: clean, extensible, orthogonal, free
- Tardis supports RISC-V’s release consistency
- Prototype Tardis on rocket-chip
- Chisel simplifies extending hardware
Outline

• The Tardis cache coherence protocol
• Thousand-core prototype
• RISC-V and Tardis
  – Release consistency
  – Atomic instructions
  – Synchronization (see S.M.)

## Consistency model comparison

<table>
<thead>
<tr>
<th>Type</th>
<th>Ordering rule</th>
<th>Tardis rule</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SC</strong></td>
<td>$X &lt;_p Y \Rightarrow X &lt;_s Y$</td>
<td>$X &lt;<em>p Y \Rightarrow X &lt;</em>{ts} Y$</td>
</tr>
<tr>
<td><strong>RC: acquires</strong></td>
<td>$\text{acq} &lt;_p X \Rightarrow \text{acq} &lt;_s X$</td>
<td>$\text{acq} &lt;<em>p X \Rightarrow \text{acq} &lt;</em>{ts} X$</td>
</tr>
<tr>
<td><strong>RC: releases</strong></td>
<td>$X &lt;_p \text{rel} \Rightarrow X &lt;_s \text{rel}$</td>
<td>$X &lt;<em>p \text{rel} \Rightarrow X &lt;</em>{ts} \text{rel}$</td>
</tr>
<tr>
<td><strong>RC: sync</strong></td>
<td>$S_X &lt;_p S_Y \Rightarrow S_X &lt;_s S_Y$</td>
<td>$S_X &lt;<em>p S_Y \Rightarrow S_X &lt;</em>{ts} S_Y$</td>
</tr>
</tbody>
</table>

$<_p$: program order  $<_s$: global memory order  $<_{ts}$: timestamp order
Release consistency and Tardis

- Loosen constraint of pts with $\text{ts}_{\text{min}}$
- Track most recent operation with $\text{ts}_{\text{max}}$
- Fences: $\text{ts}_{\text{min}} \leftarrow \text{ts}_{\text{max}}$
- Track acquires/releases with $\text{ts}_{\text{rel}}$
  - Release: $\text{ts}_{\text{rel}} \leftarrow \text{ts}_{\text{max}}$
  - Acquire: $\text{ts}_{\text{min}} \leftarrow \text{ts}_{\text{rel}}$
Load-reserved and store-conditional

• Tardis gives neat solution to LR/SC livelock
• wts tracks cache line version
• SC success condition: $wts_{lr} \equiv wts_{\text{before sc}}$
LR/SC example

- Core 0 performs lr on C
  - exclusive ownership
  - $wts_{lr} = 0$
- Core 1 performs lr on C
  - core 0 downgraded
- Core 0 performs sc
  - core 1 downgraded
  - succeeds; $wts_{lr} == wts$
  - writes C' at pts 1
Block diagram

Rocket Core

HellaCache D$

TileLink NoC

Last-level cache

Main memory

pts

metadata, hit/miss logic

message timestamps

new coherence logic
Thanks!

• Special thanks to Xiangyao Yu and Srini Devadas for their extensive advice and input