Towards Thousand-Core RISC-V Shared Memory Systems

Quan Nguyen

Massachusetts Institute of Technology

30 November 2016

Outline

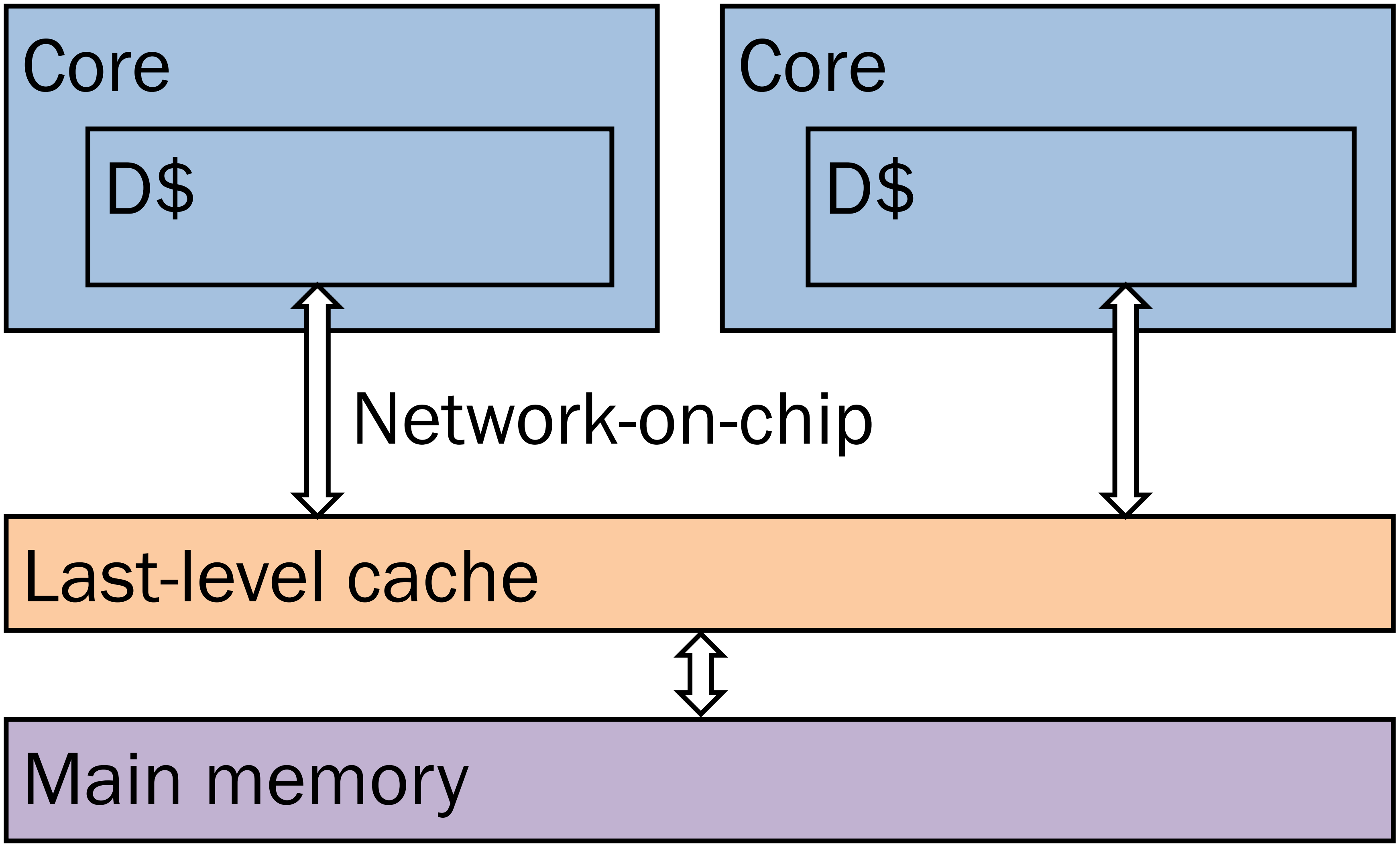
- **The Tardis cache coherence protocol**
  - Example
  - Scalability advantages
- Thousand-core prototype
- RISC-V and Tardis

Tardis

- A new cache coherence protocol
- Enforces consistency through timestamps
- Key idea: logical leases
  - Timestamps **wts** and **rts**: start and end of lease
  - Processors keep timestamp **pts** in register

Xiangyao Yu and Srinivas Devadas, "Tardis: Time Traveling Coherence Algorithm for Distributed Shared Memory", PACT 2015.

Block diagram



Metadata per cache line

- Client:

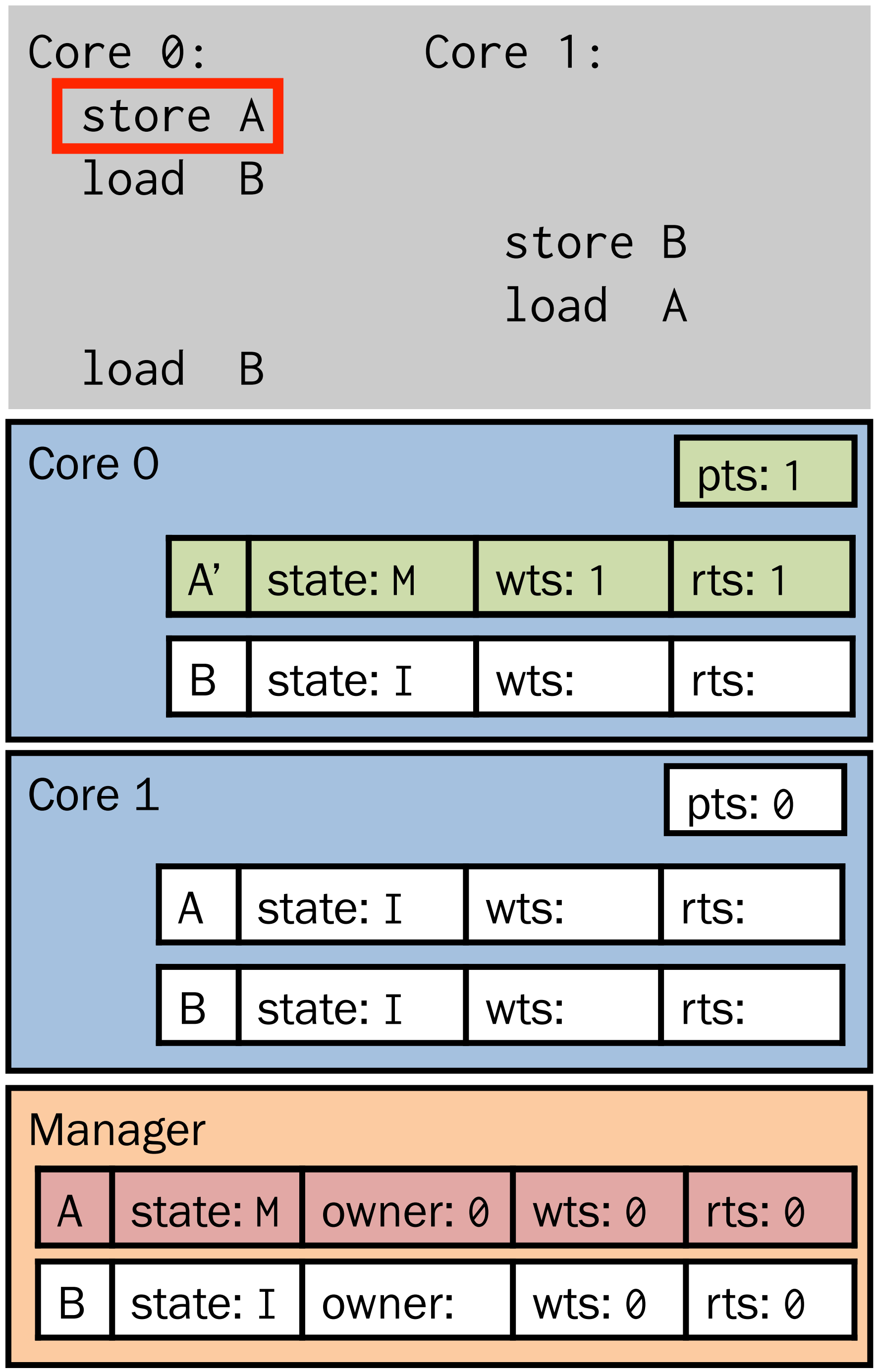
A	state: M	wts: 1	rts: 1

  – Use ordinary MESI states

- Manager:

A	state: M	owner: 0	wts: 0	rts: 0

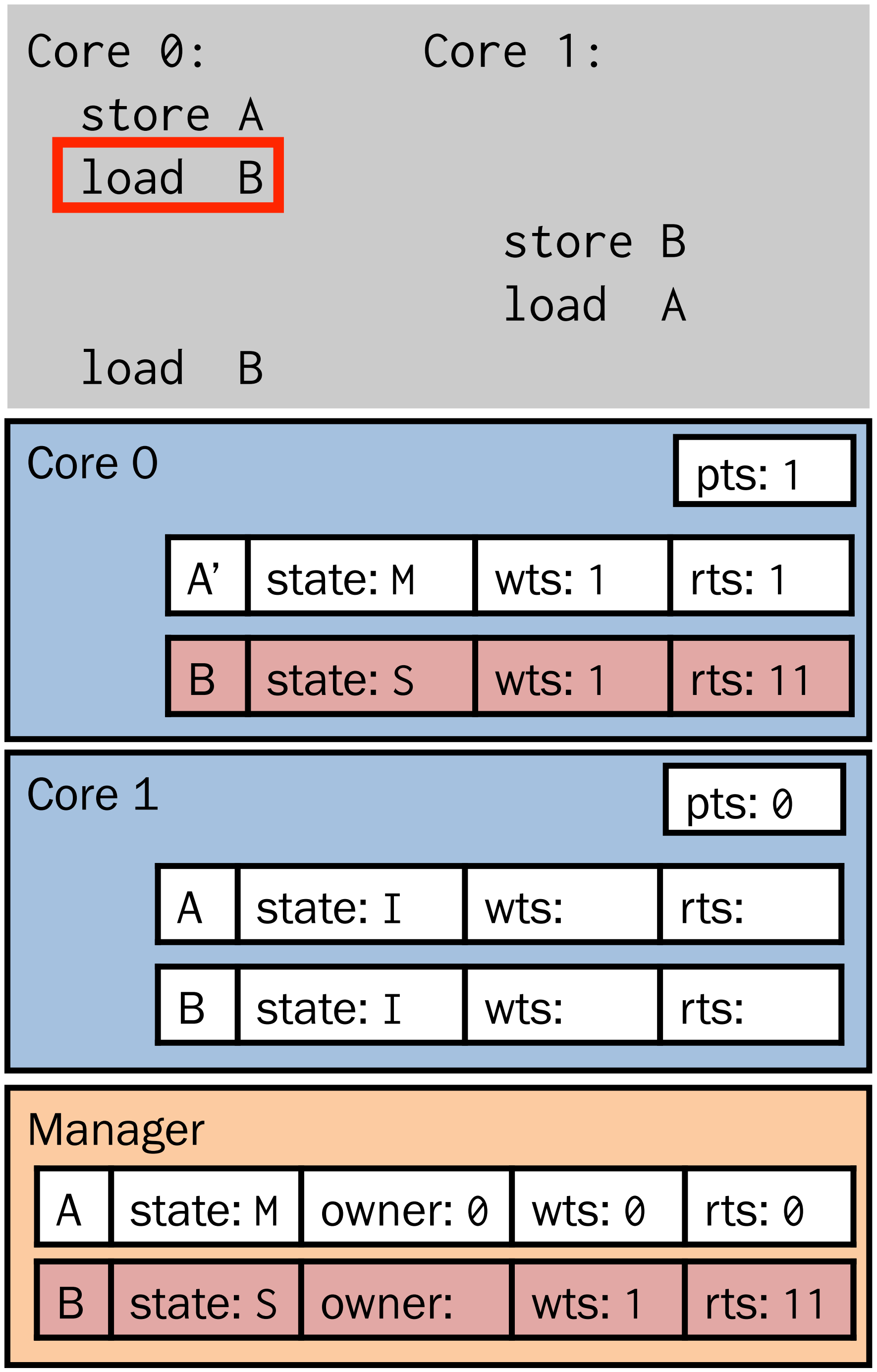
  – Tracks only the exclusive owner
  – Tracks most recent lease


```
Core 0:         Core 1:
  store A
  load  B
                   store B
                   load  A
  load  B
```

Core 0
pts: 1
A’ | state: M | wts: 1 | rts: 1
B | state: I | wts: | rts:
Core 1
pts: 0
A | state: I | wts: | rts:
B | state: I | wts: | rts:
Manager
A | state: M | owner: 0 | wts: 0 | rts: 0
B | state: I | owner: | wts: 0 | rts: 0

Example

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



```
Core 0:         Core 1:
  store A
  load  B
                   store B
                   load  A
  load  B
```

Core 0 — pts: 1

A’	state: M	wts: 1	rts: 1
B	state: S	wts: 1	rts: 11

Core 1 — pts: 0

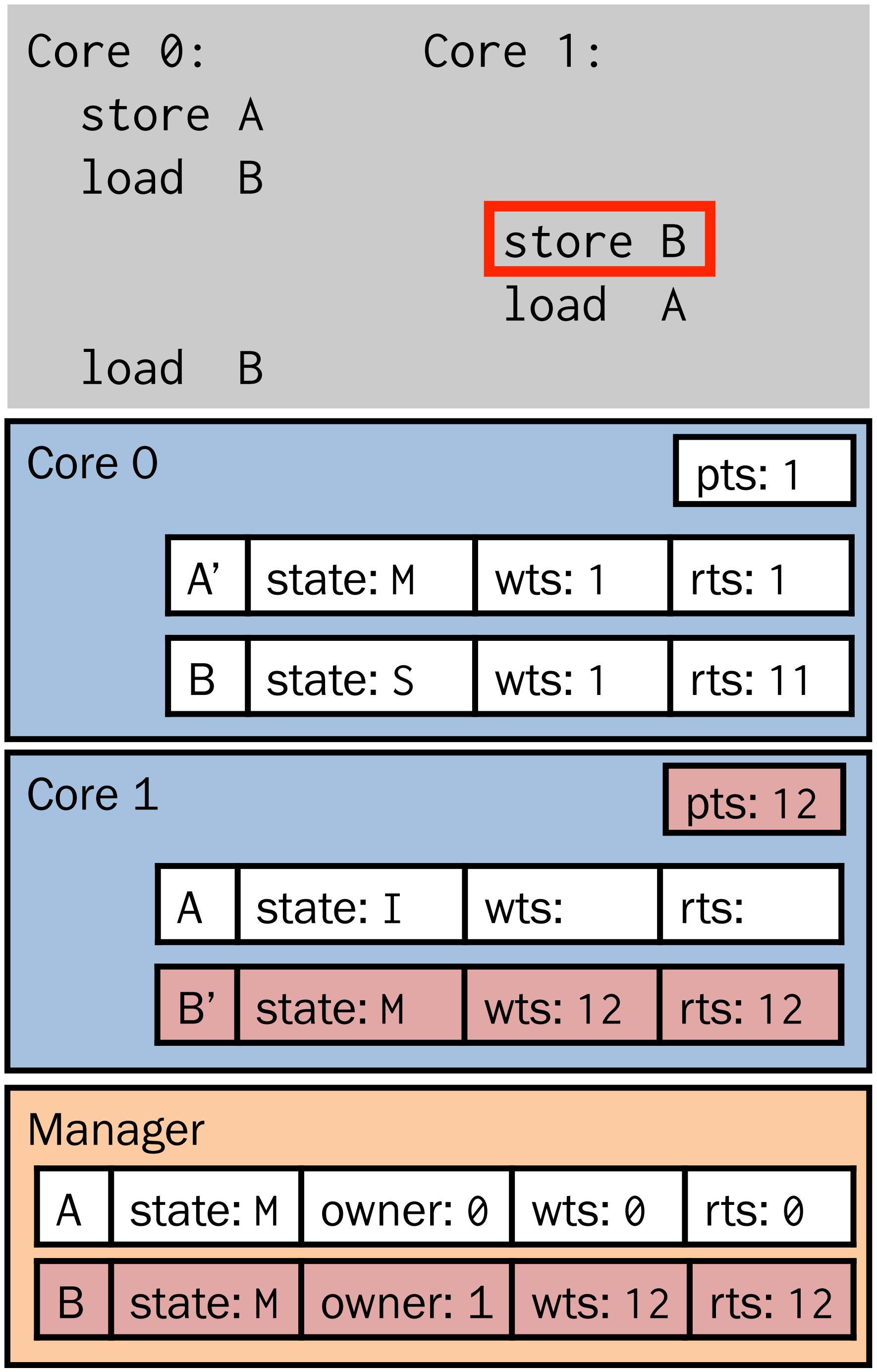
A	state: I	wts:	rts:
B	state: I	wts:	rts:

Manager

A	state: M	owner: 0	wts: 0	rts: 0
B	state: S	owner:	wts: 1	rts: 11

Example

- Core 0 loads B
  - Sends pts to manager
- Manager leases B
  - Sets lease based on pts
  - [wts, rts] = [1, 11]
- Core 0 reads B at pts 1



```
Core 0:          Core 1:
  store A
  load  B
                   store B
                   load  A
  load  B
```

Core 0 pts: 1

A’	state: M	wts: 1	rts: 1
B	state: S	wts: 1	rts: 11

Core 1 pts: 12

A	state: I	wts:	rts:
B’	state: M	wts: 12	rts: 12

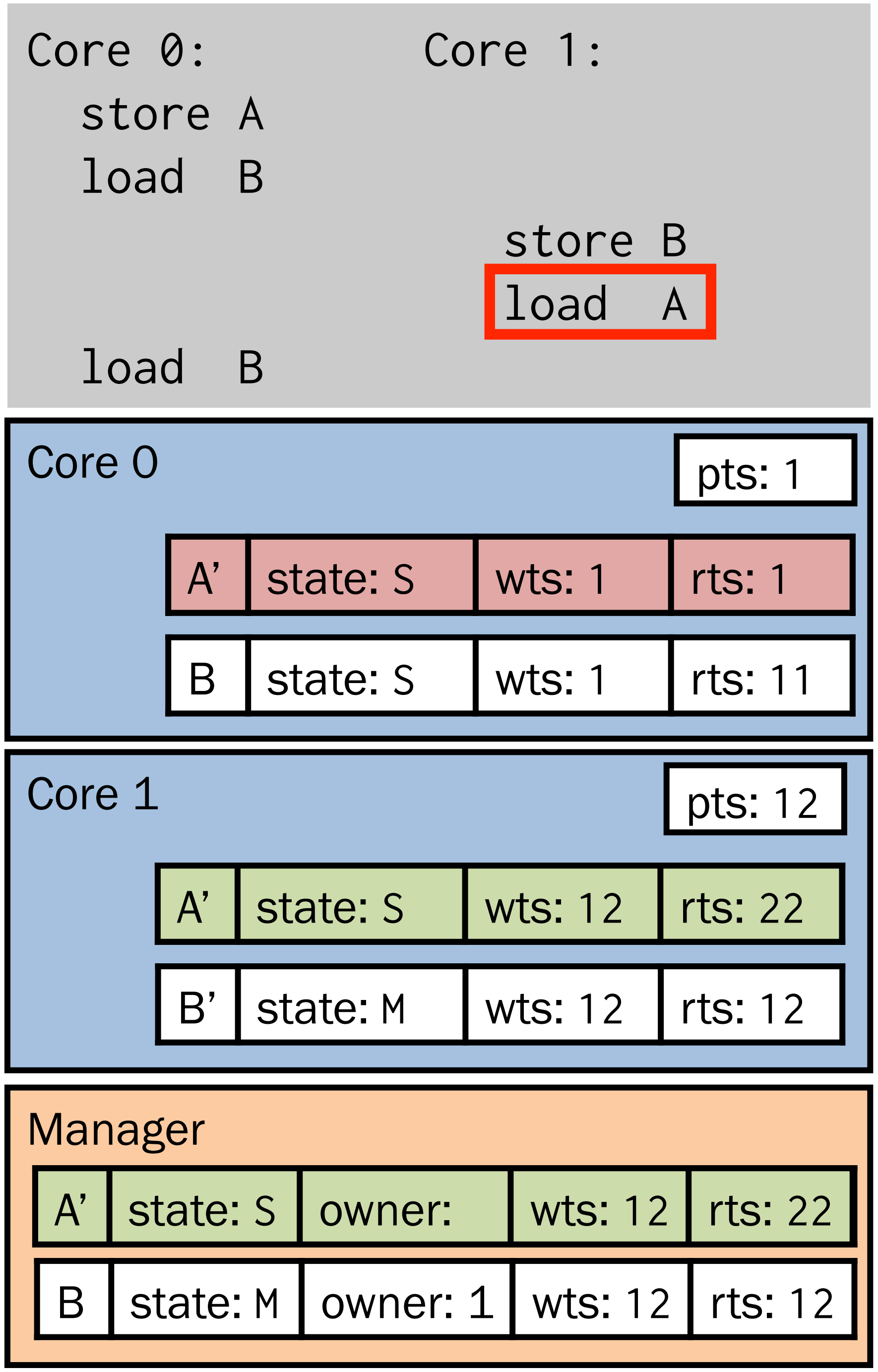
Manager

A	state: M	owner: 0	wts: 0	rts: 0
B	state: M	owner: 1	wts: 12	rts: 12

Example

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

Example



Core 0:
store A
load B
load B
Core 1:
store B
load A
Core 0
pts: 1
A' | state: S | wts: 1 | rts: 1
B | state: S | wts: 1 | rts: 11
Core 1
pts: 12
A' | state: S | wts: 12 | rts: 22
B' | state: M | wts: 12 | rts: 12
Manager
A' | state: S | owner: | wts: 12 | rts: 22
B | state: M | owner: 1 | wts: 12 | rts: 12

- 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

```
Core 0:        Core 1:
  store A
  load  B
                  store B
                  load  A
  load  B
```

Core 0 pts: 1

A'	state: S	wts: 1	rts: 1
B	state: S	wts: 1	rts: 11

Core 1 pts: 12

A'	state: S	wts: 12	rts: 22
B'	state: M	wts: 12	rts: 12

Manager

A'	state: S	owner:	wts: 12	rts: 22
B	state: M	owner: 1	wts: 12	rts: 12

Example

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

A case for scalability

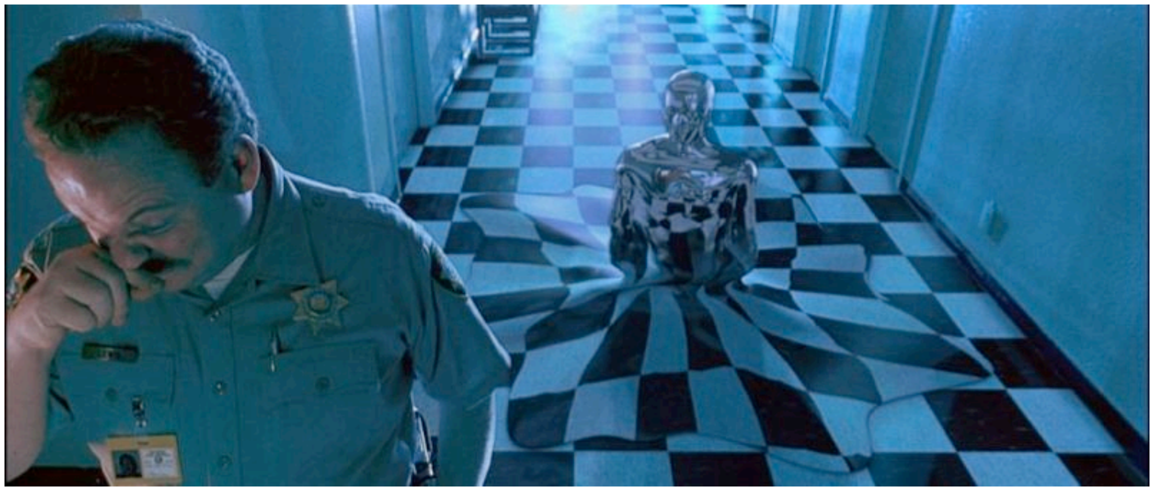
- Track only one node: O(log N) storage
- No broadcast invalidations
- Timestamps not tied to core count
  - No need for synchronized real-time clocks
  - Can be compressed

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


Terminator 2: Judgment Day (1991)
Carolco Pictures, Lightstorm Entertainment, Le Studio Canal+, and TriStar Pictures

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.)

Quan Nguyen, “Synchronization in Timestamp-Based Cache Coherence Protocols”, S.M. thesis, MIT, 2016.

Consistency model comparison

Type	Ordering rule	Tardis rule
SC	$X <_p Y \Longrightarrow X <_s Y$	$X <_p Y \Longrightarrow X \boxed{<_{ts}} Y$
RC: acquires	$acq <_p X \Longrightarrow acq <_s X$	$acq <_p X \Longrightarrow acq <_{ts} X$
RC: releases	$X <_p rel \Longrightarrow X <_s rel$	$X <_p rel \Longrightarrow X <_{ts} rel$
RC: sync $S \in \{acq, rel\}$	$S_X <_p S_Y \Longrightarrow S_X <_s S_Y$	$S_X <_p S_Y \Longrightarrow S_X <_{ts} S_Y$

$<_p$: program order $<_s$: global memory order $<_{ts}$: timestamp order

Release consistency and Tardis

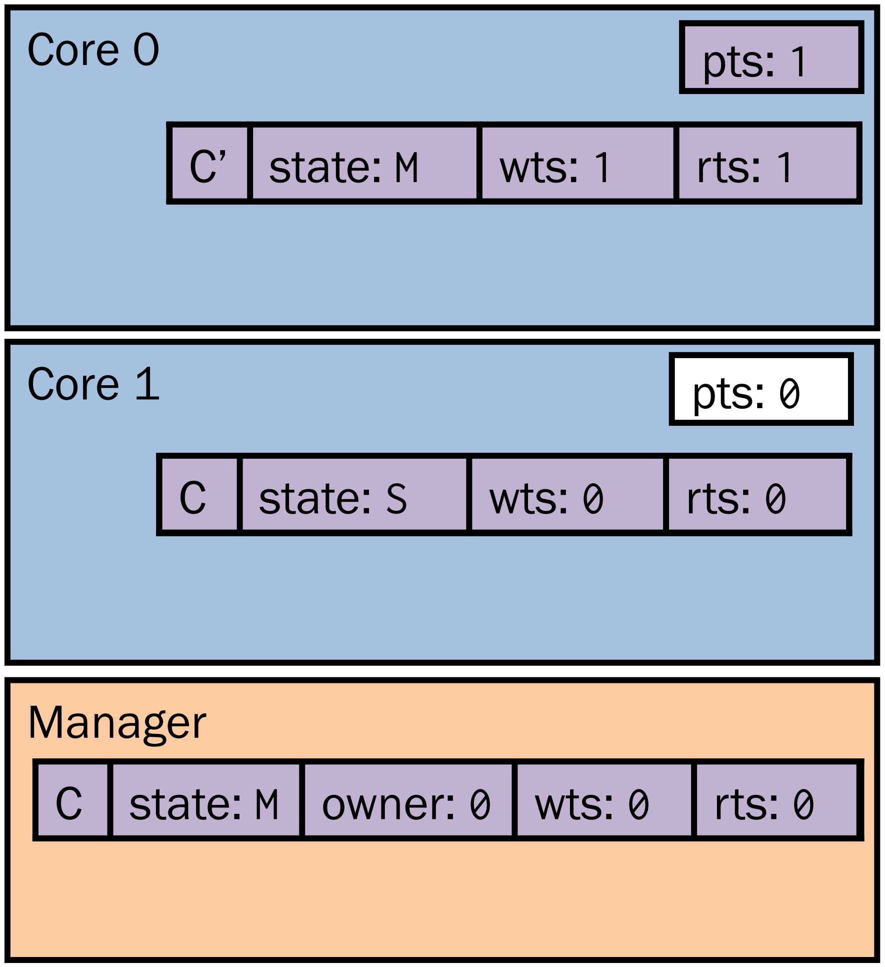
- Loosen constraint of pts with $\mathbf{ts_{min}}$
- Track most recent operation with $\mathbf{ts_{max}}$
- Fences: $ts_{min} \leftarrow ts_{max}$
- Track acquires/releases with $\mathbf{ts_{rel}}$
  - Release: $ts_{rel} \leftarrow ts_{max}$
  - Acquire: $ts_{min} \leftarrow ts_{rel}$

Load-reserved and store-conditional

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

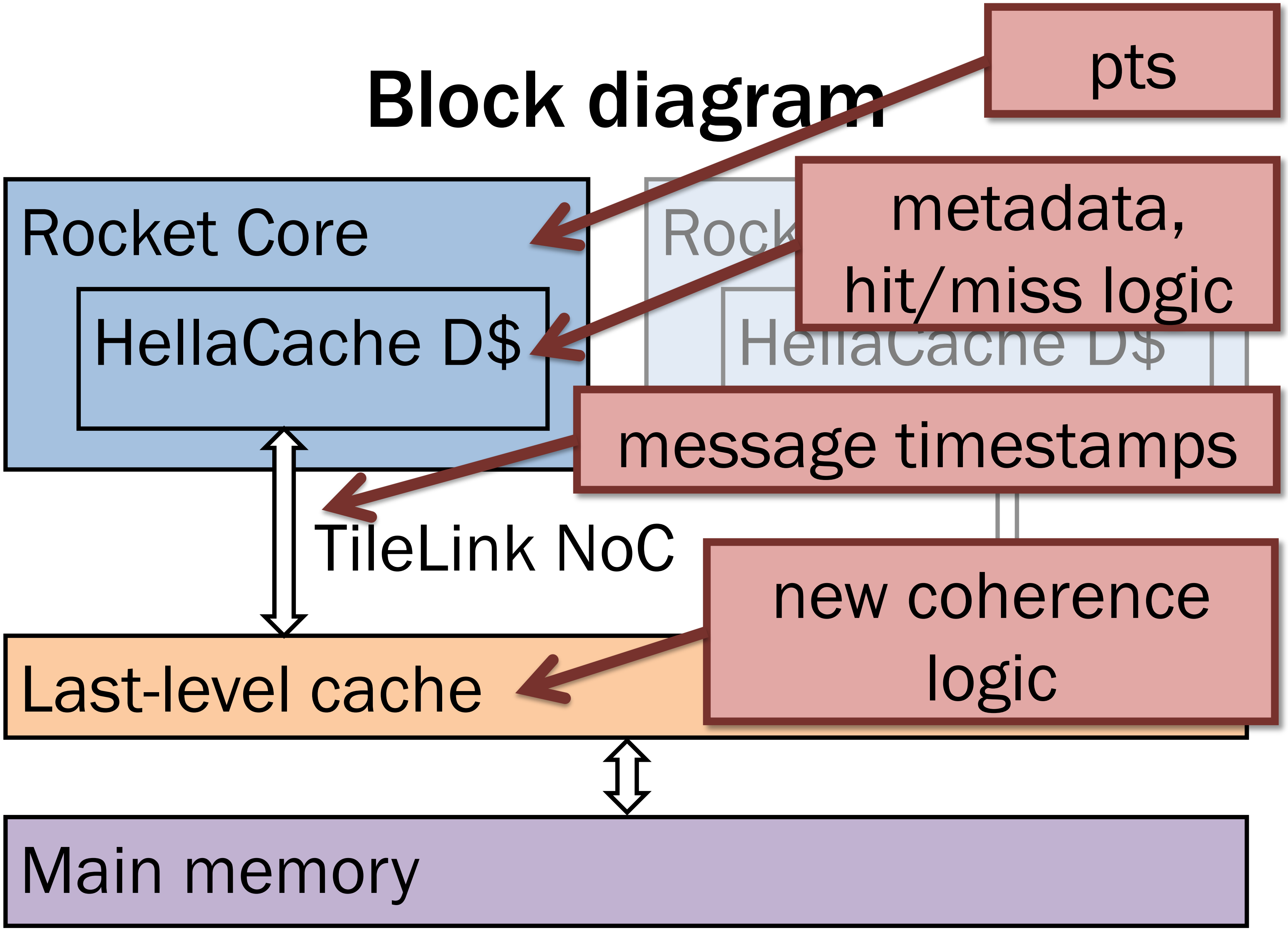
```
loop:
  lr.d      x1, 0(C)
  <do stuff to x1>
  sc.d  x2, x1, 0(C)
  bnez  x2, loop
```

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



Thanks!

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