# SCORPIO: 36-Core Shared Memory Processor

## Demonstrating Snoopy Coherence on a Mesh Interconnect

**Chia-Hsin Owen Chen**

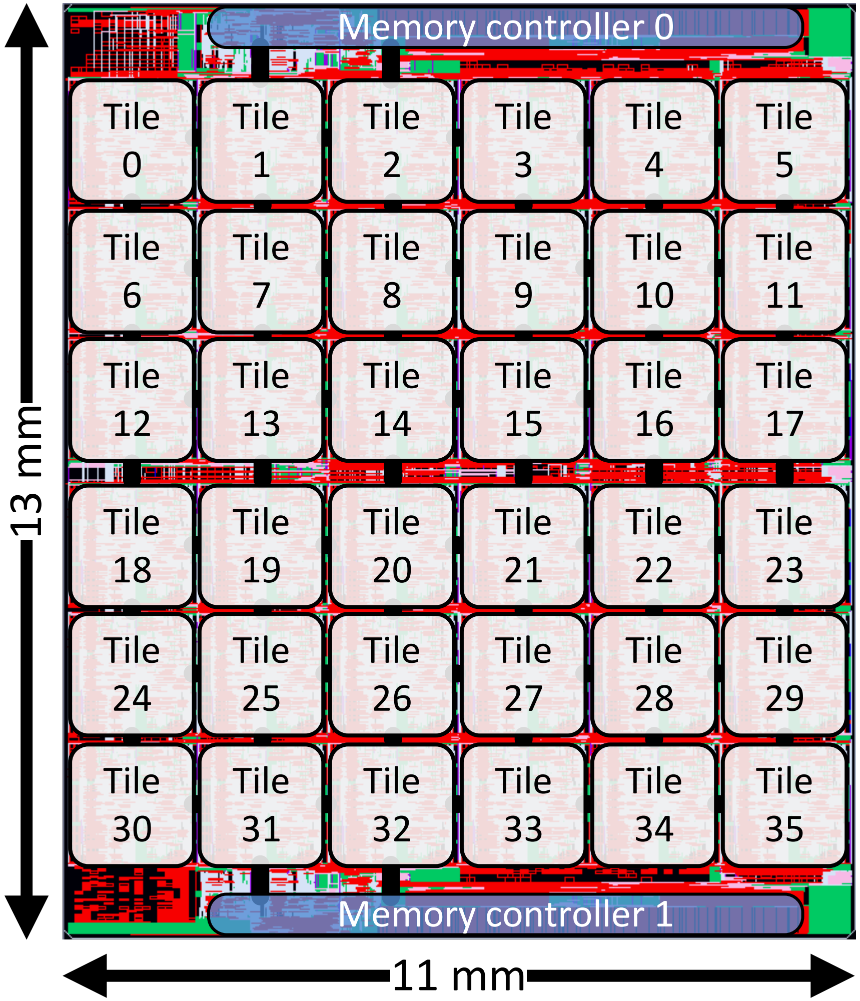
Collaborators: Sunghyun Park, Suvinay Subramanian, Tushar Krishna,
Bhavya Daya, Woo Cheol Kwon, Brett Wilkerson, John Arends,
Anantha Chandrakasan, Li-Shiuan Peh

Contributions:

Core integration (Bhavya and Owen),
Cache coherence protocol design (Bhavya and Woo Cheol)
L2 cache controller implementation (Bhavya)
Memory interface controller implementation (Owen)
High-level idea of notification network (Woo-Cheol)
Network architecture (Woo-Cheol, Bhavya, Owen, Tushar, Suvinay)
Network implementation (Suvinay)

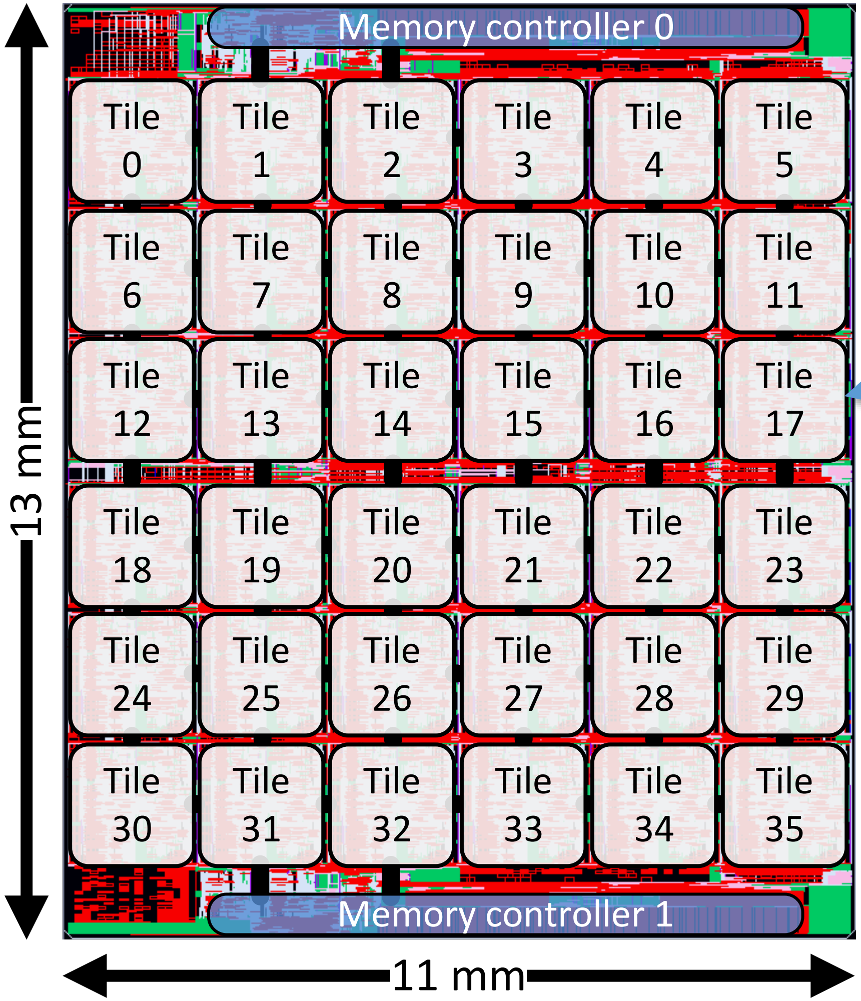
DDR2 and PHY integration (Sunghyun and Owen)
Backend of entire chip (Owen)
FPGA interfaces, on-chip testers and scan chains (Tushar)
RTL functional simulations (Bhavya, Owen, Suvinay)
Full-system GEMS simulations (Woo-Cheol)
Board Design (Sunghyun)
Software Stack (Bhavya and Owen)
Package Design (Freescale)

# SCORPIO Overview



**IBM 45nm SOI, 143mm$^2$**
**600M transistors**

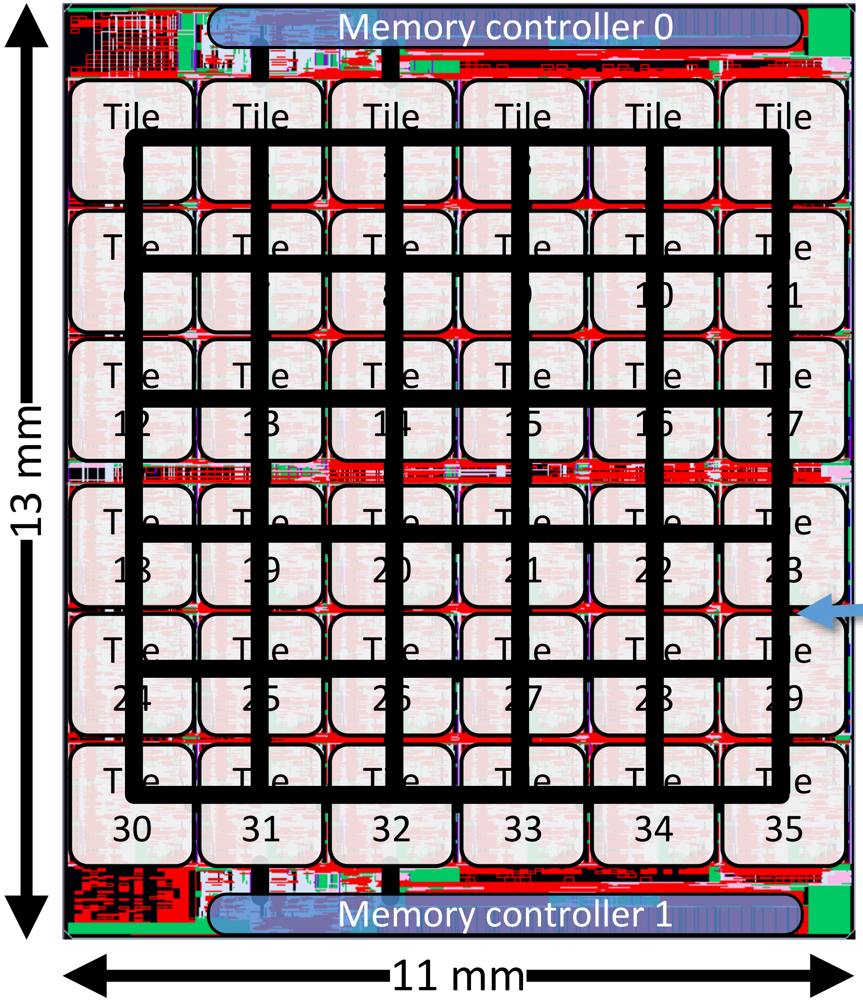
# SCORPIO Overview



**IBM 45nm SOI, 143mm$^2$**
**600M transistors**

**36 cores with total 4.5MB L2**

# SCORPIO Overview

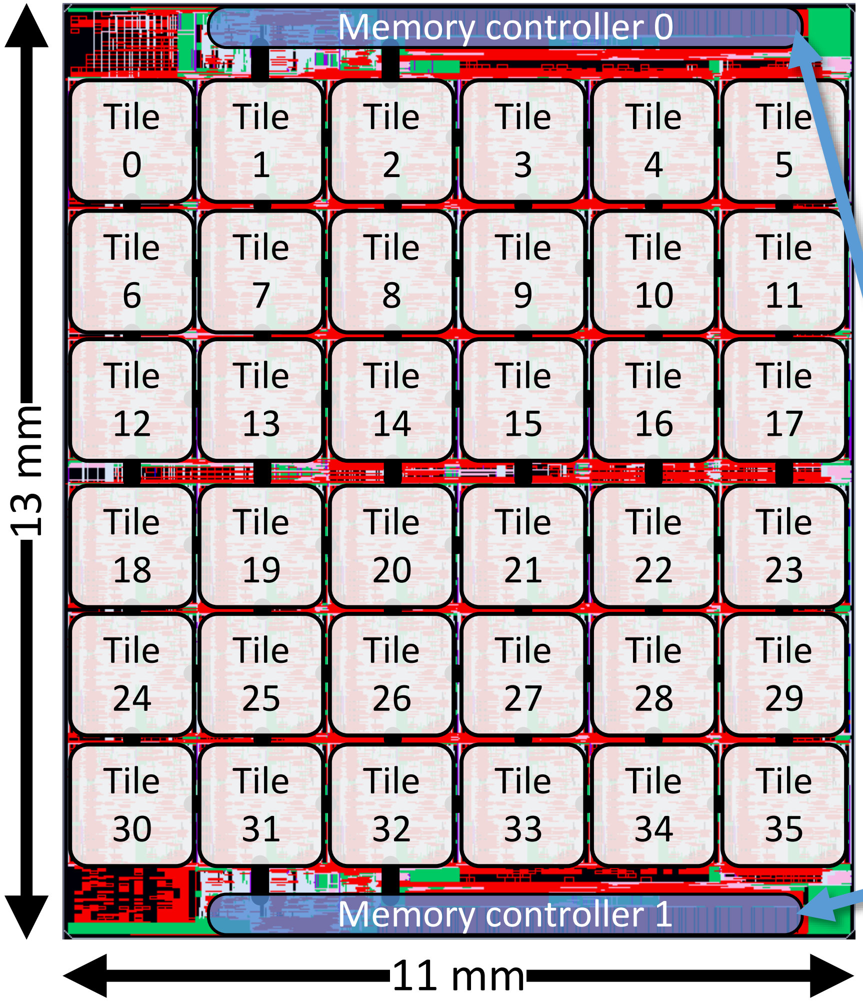


IBM 45nm SOI, 143mm$^2$
600M transistors

36 cores with total 4.5MB L2

6×6 mesh on-chip network supporting snoopy coherence

# SCORPIO Overview



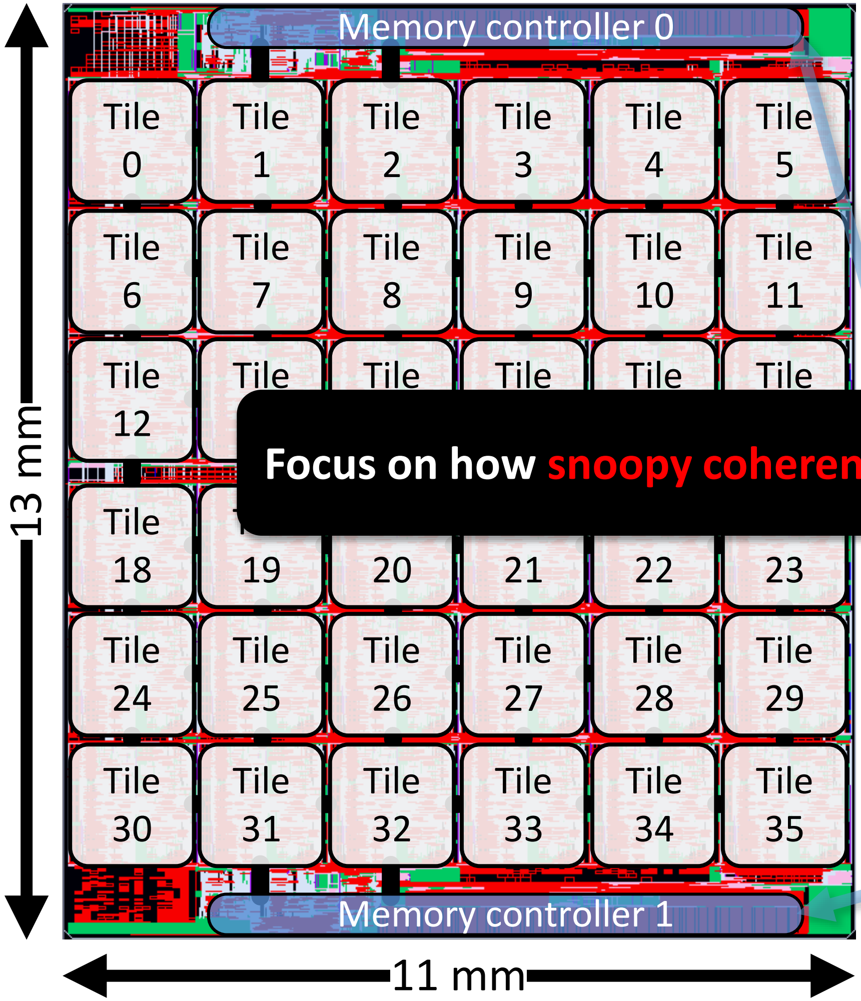
**IBM 45nm SOI, 143mm$^2$**
**600M transistors**

**36 cores with total 4.5MB L2**

**6×6 mesh on-chip network supporting snoopy coherence**

**Dual channel DDR2 memory controller**

# SCORPIO Overview



IBM 45nm SOI, 143mm$^2$
600M transistors

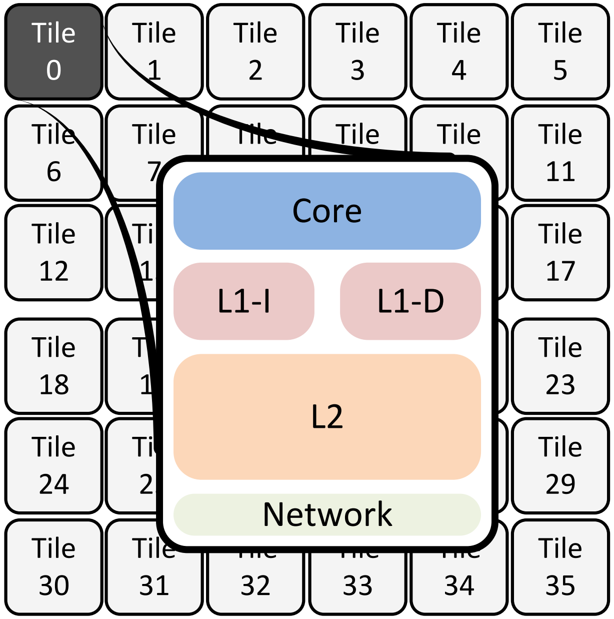
36 cores with total 4.5MB L2

6×6 mesh on-chip network supporting snoopy coherence

Dual channel DDR2 memory controller

**Focus on how snoopy coherence is enabled on a mesh interconnect**

# Tile Architecture



## Core

- Freescale e200 z760n3
- In-order
- Dual-issue

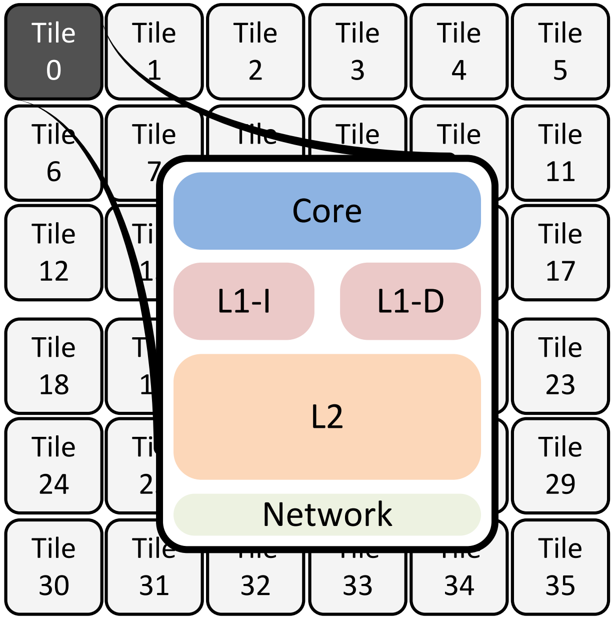
## Private L1 cache

- Split 16KB for Inst / Data
- 4-way set associative

## Private L2 cache

- 128KB
- 4-way set associative
- Inclusive

# Tile Architecture



## Core

- Freescale e200 z760n3
- In-order
- Dual-issue

## Private L1 cache

- Split 16KB for Inst / Data
- 4-way set associative

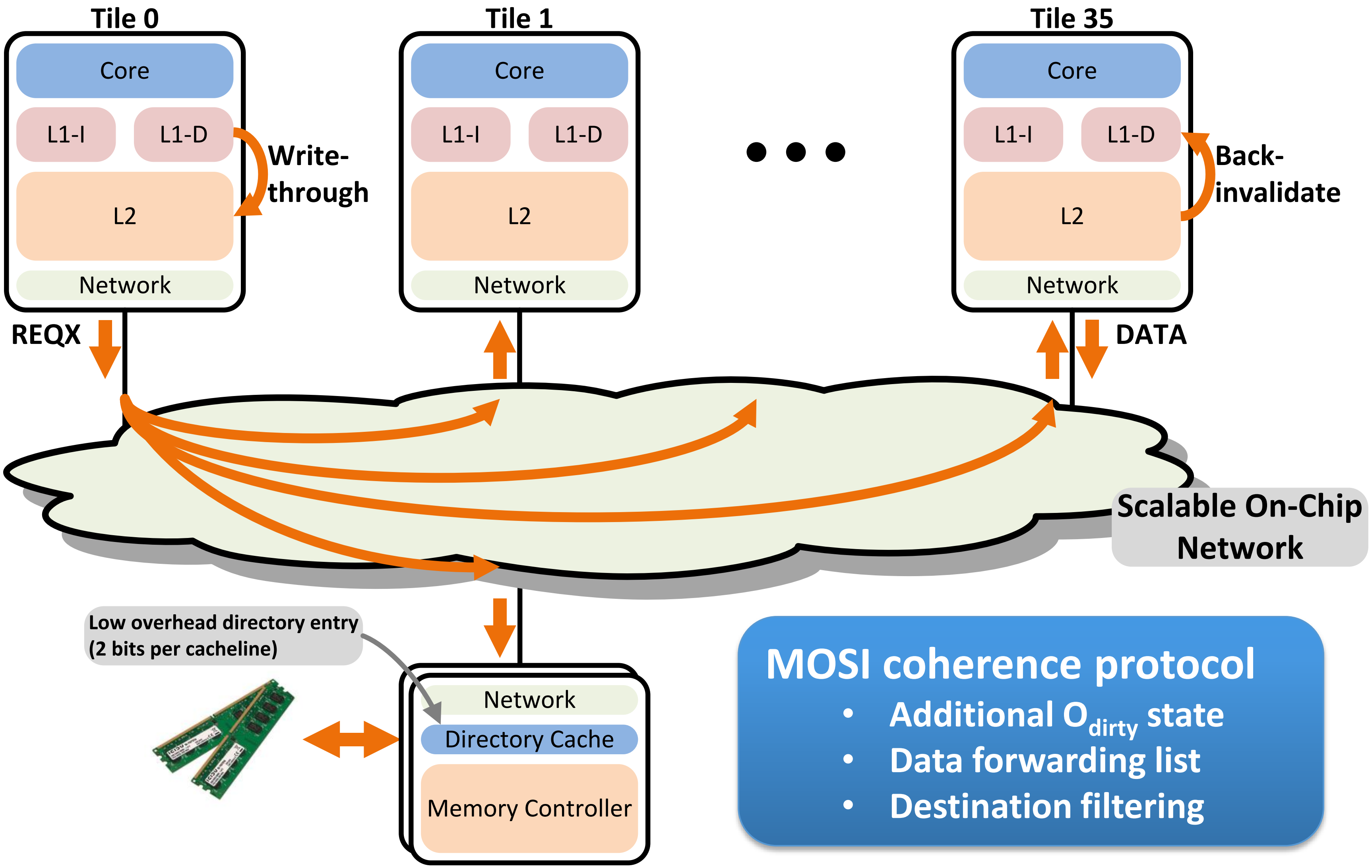
Write-through

Back-invalidate

## Private L2 cache

- 128KB
- 4-way set associative
- Inclusive

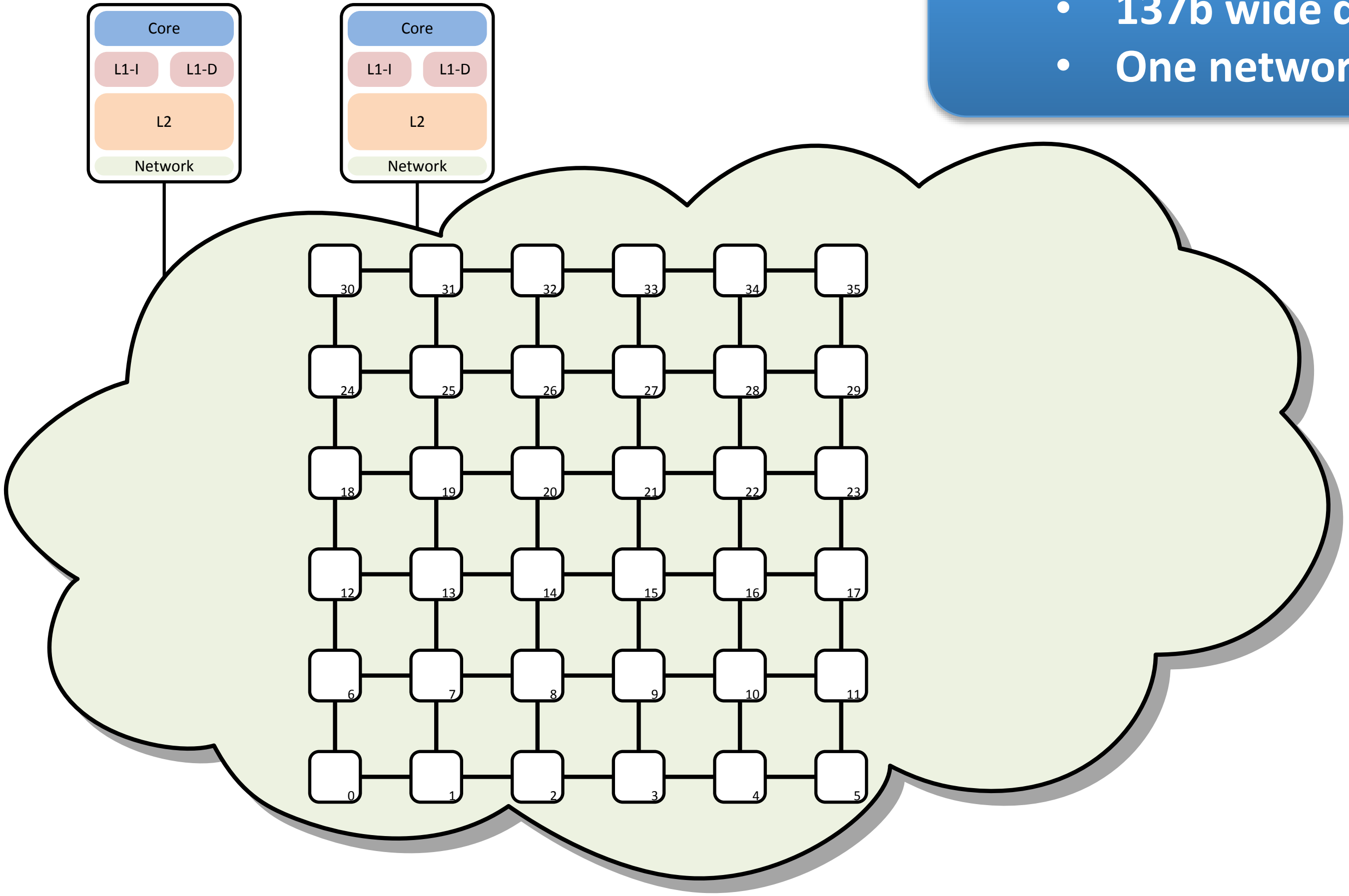
# Snoopy Coherence



## MOSI coherence protocol

- Additional $O_{dirty}$ state
- Data forwarding list
- Destination filtering

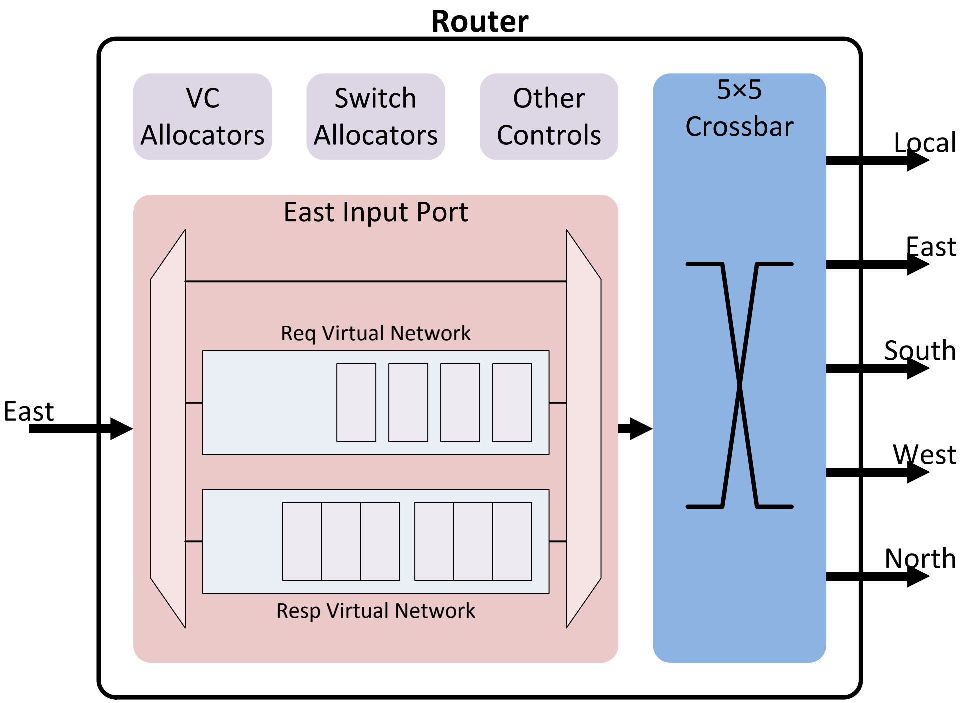
# Scalable On-Chip Network



6×6 mesh interconnect

- 137b wide data-path
- One network node / tile

# Scalable On-Chip Network

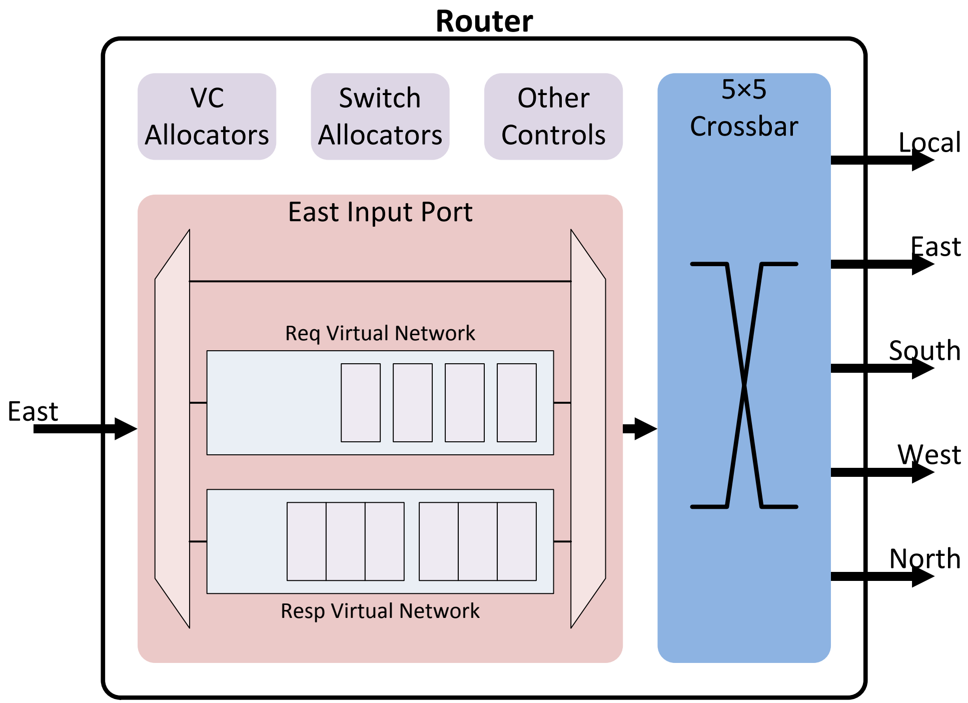


| Regular Pipeline Stages | Buffer Write (BW)<br>Switch Arbitration Inport (SA-I) | Buffer Read (BR)<br>Switch Allocation Outport (SA-O)<br>VC Allocation (VA)<br>Lookahead/Header Generation | Switch Traversal (ST) |
|---|---|---|---|

## 6×6 mesh interconnect

- 137b wide data-path
- One network node / tile

# Scalable On-Chip Network



| | | | |
|---|---|---|---|
| Regular Pipeline Stages | Buffer Write (BW)<br>Switch Arbitration Inport (SA-I) | Buffer Read (BR)<br>Switch Allocation Outport (SA-O)<br>VC Allocation (VA)<br>Lookahead/Header Generation | Switch Traversal (ST) |

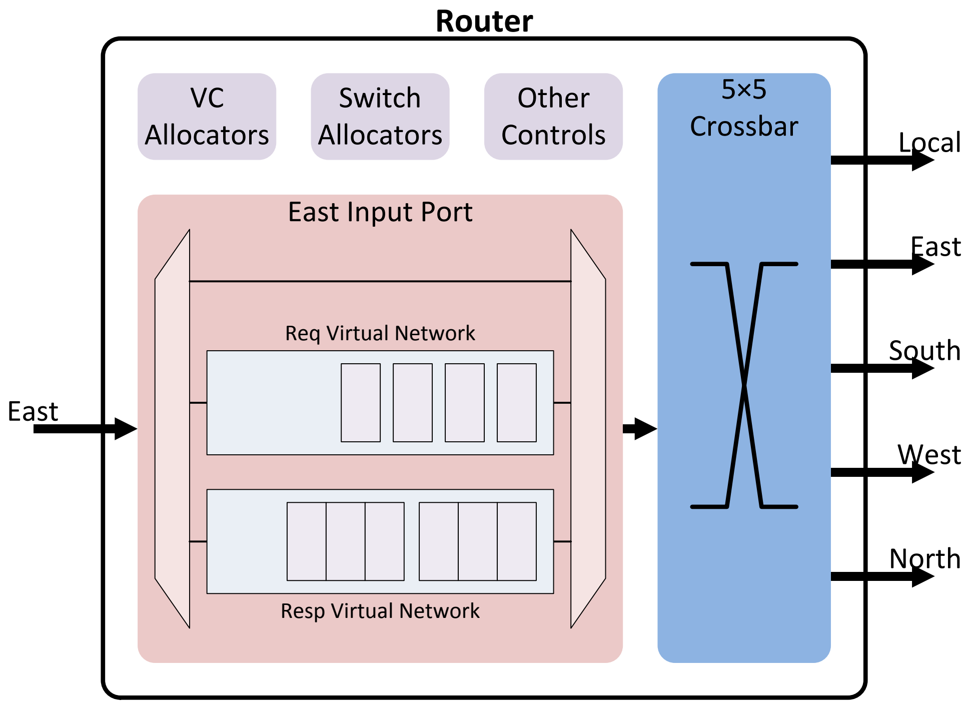
## 6×6 mesh interconnect

- 137b wide data-path
- One network node / tile

## Deadlock avoidance

- Two virtual networks
- Dimensional X-Y routing

# Scalable On-Chip Network



| | | | |
|---|---|---|---|
| Regular Pipeline Stages | Buffer Write (BW)<br>Switch Arbitration Inport (SA-I) | Buffer Read (BR)<br>Switch Allocation Outport (SA-O)<br>VC Allocation (VA)<br>Lookahead/Header Generation | Switch Traversal (ST) |

## 6×6 mesh interconnect

- 137b wide data-path
- One network node / tile

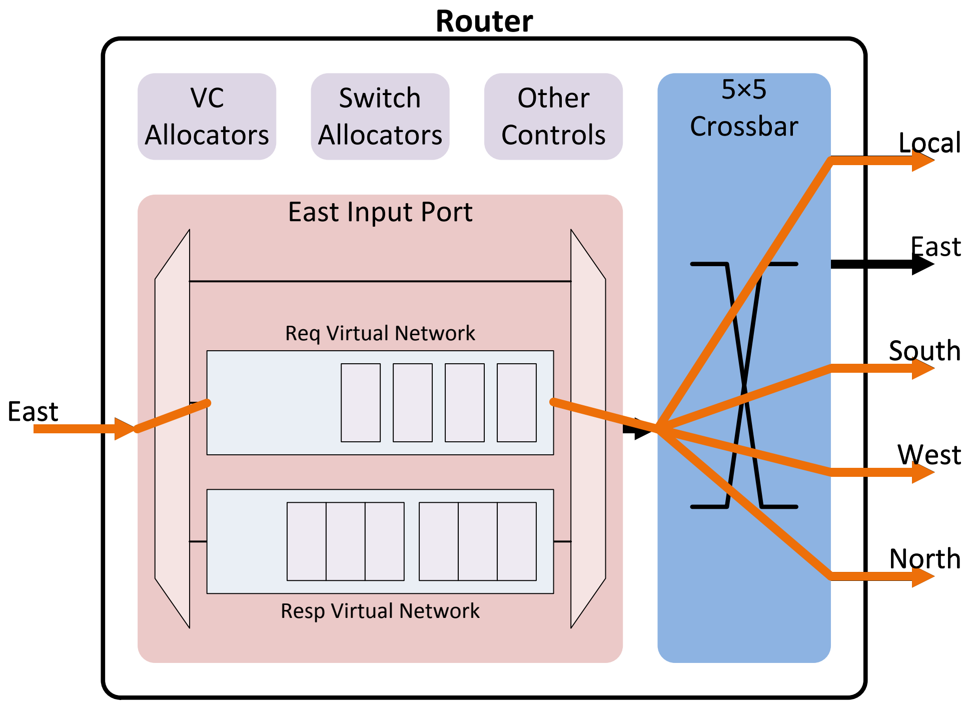
## Deadlock avoidance

- Two virtual networks
- Dimensional X-Y routing

## Optimizations

- Multiple virtual channels / VN

# Scalable On-Chip Network



| | | | |
|---|---|---|---|
| Regular Pipeline Stages | Buffer Write (BW)<br>Switch Arbitration Inport (SA-I) | Buffer Read (BR)<br>Switch Allocation Outport (SA-O)<br>VC Allocation (VA)<br>Lookahead/Header Generation | Switch Traversal (ST) |

## 6×6 mesh interconnect

- 137b wide data-path
- One network node / tile

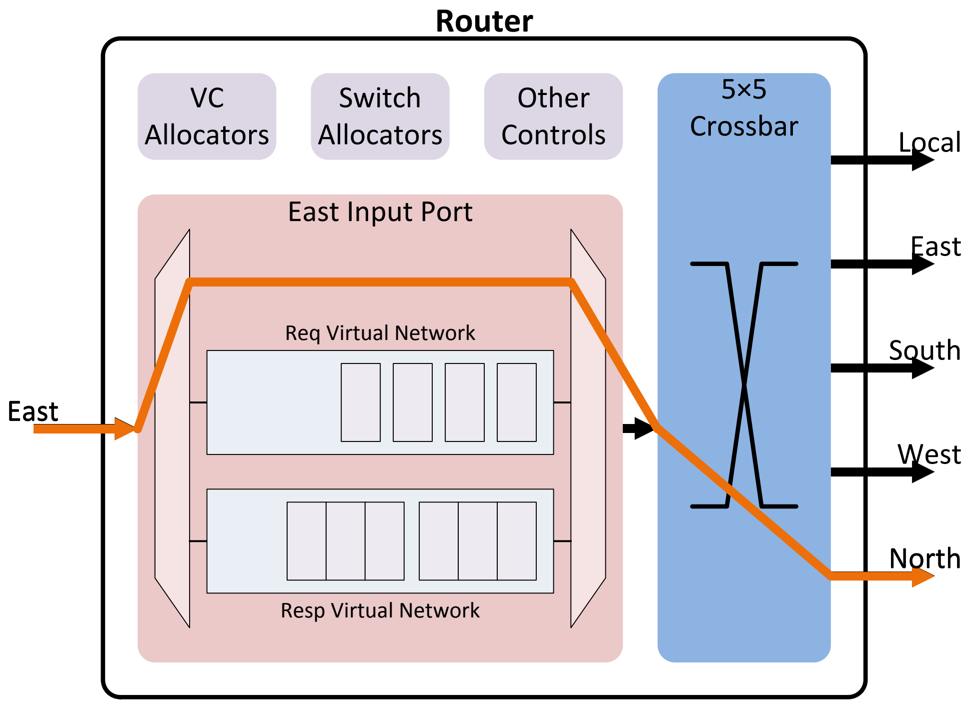
## Deadlock avoidance

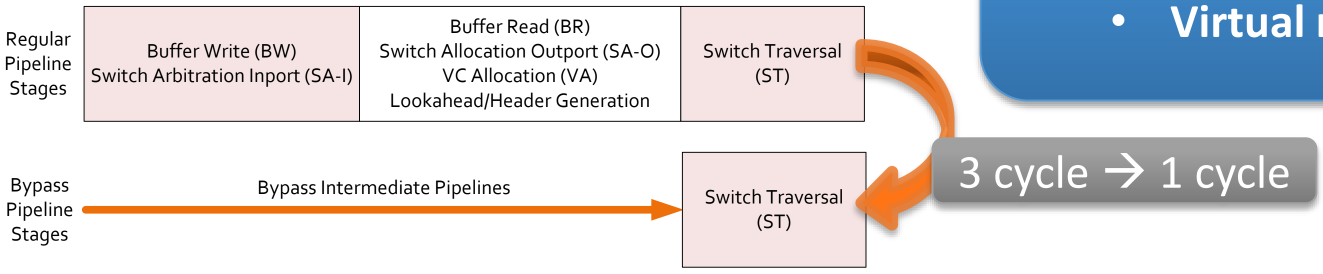
- Two virtual networks
- Dimensional X-Y routing

## Optimizations

- Multiple virtual channels / VN
- In-network broadcast support

# Scalable On-Chip Network





## 6×6 mesh interconnect

- 137b wide data-path
- One network node / tile

## Deadlock avoidance

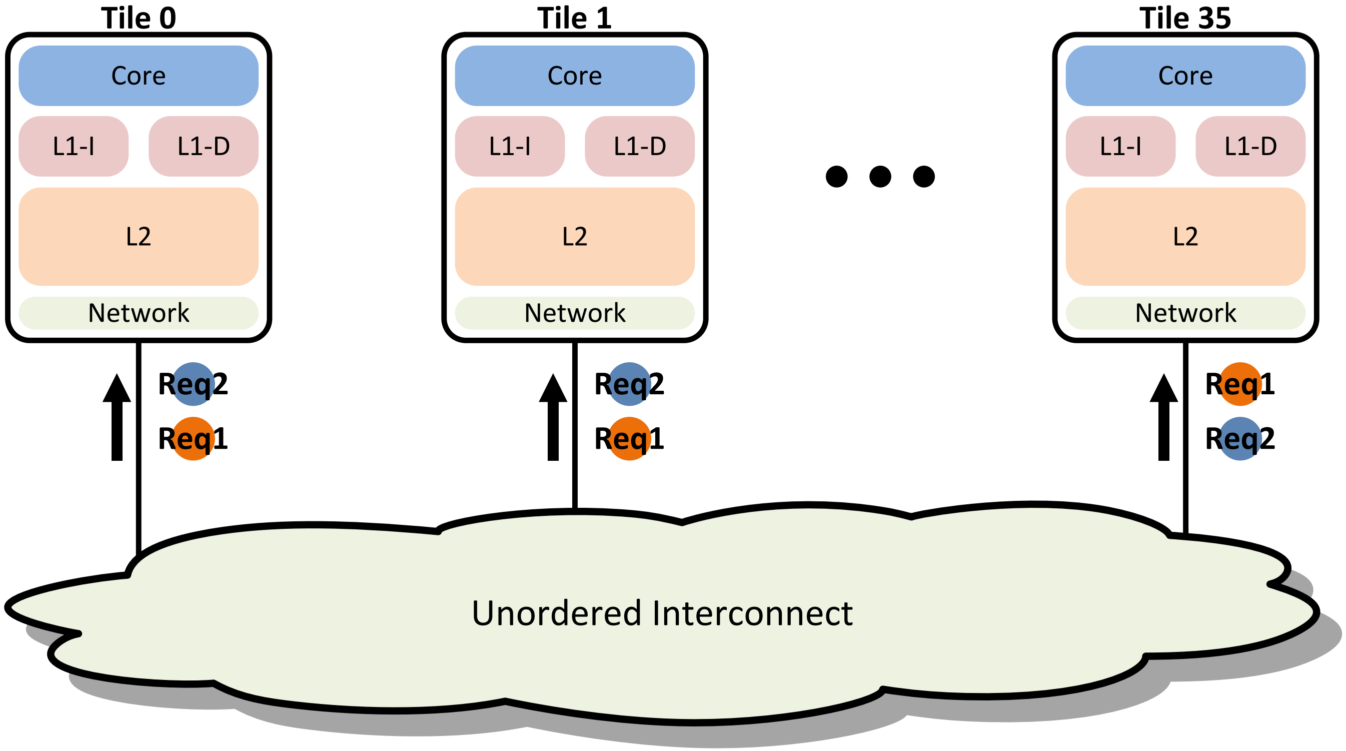
- Two virtual networks
- Dimensional X-Y routing

## Optimizations

- Multiple virtual channels / VN
- In-network broadcast support
- Virtual router pipeline bypass

# Globally-Ordered Virtual Network

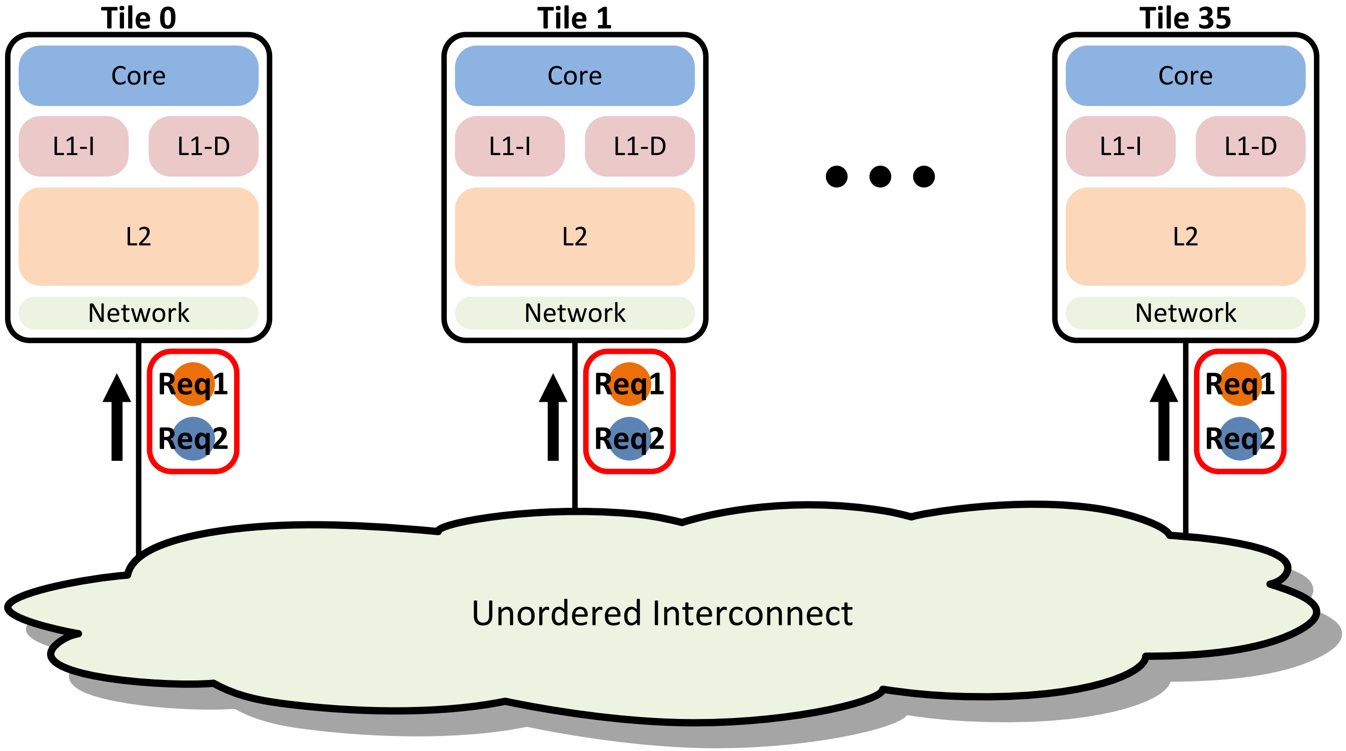
**Problem:** Broadcast Messages delivered to different nodes in different orders on unordered networks



# Globally-Ordered Virtual Network

**Problem:** Broadcast Messages delivered to different nodes in different orders on unordered networks

**We want:** Every node to see all messages in the same global order

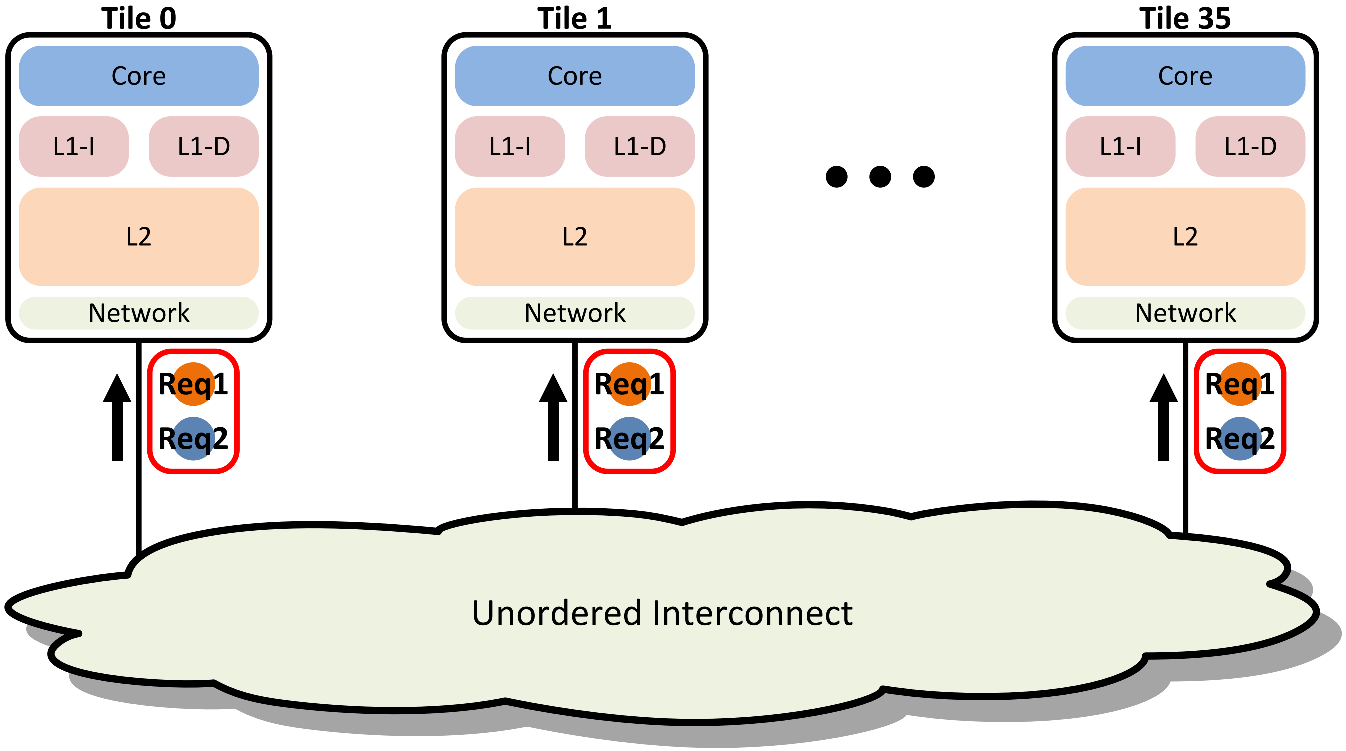


# Globally-Ordered Virtual Network

**Problem:** Broadcast Messages delivered to different nodes in

**Solution: Decouple message delivery from ordering**

**We want:** Every node to see all messages in the same global order

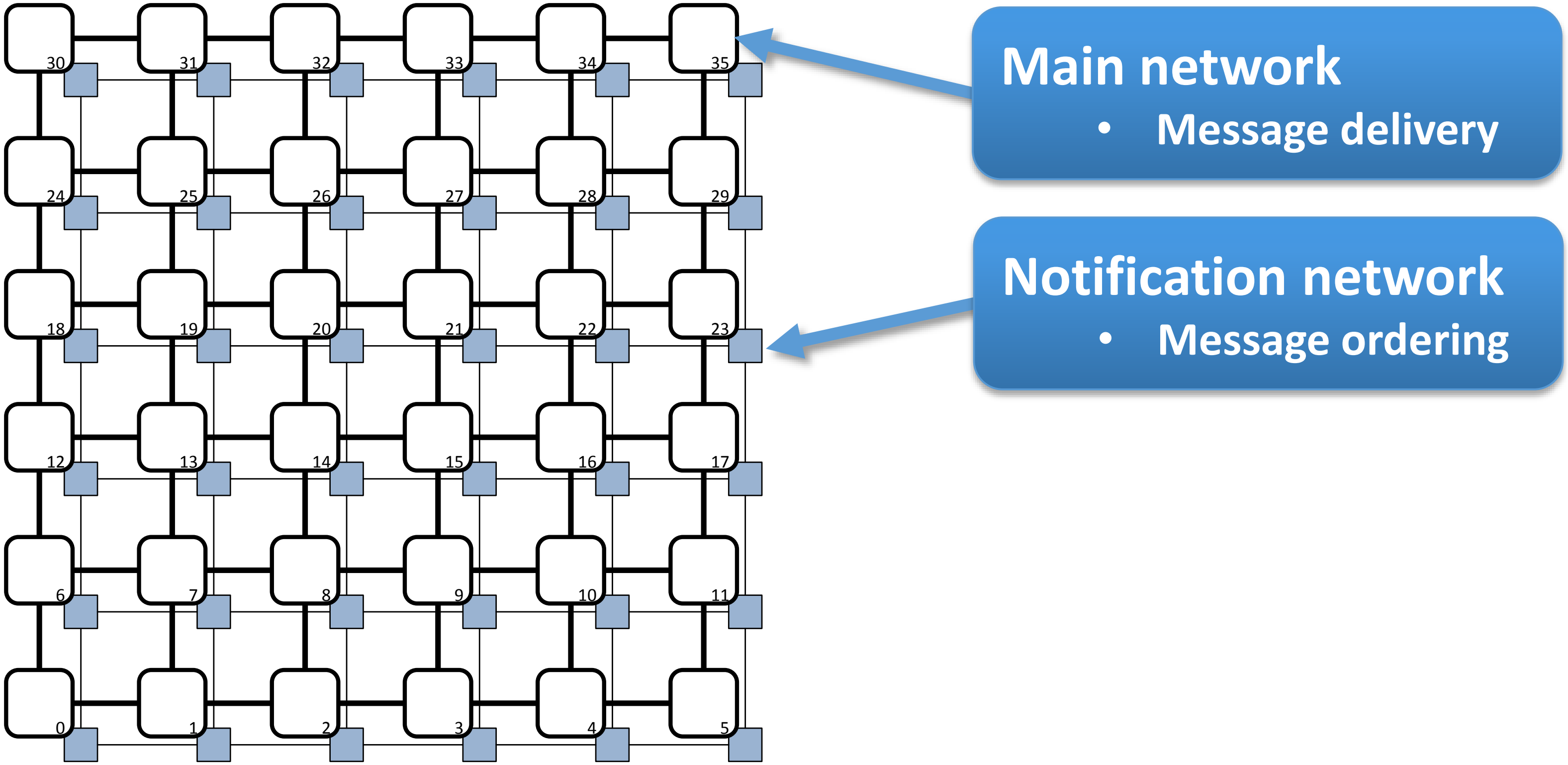


# Globally-Ordered Virtual Network

**Problem:** Broadcast Messages delivered to different nodes in

**Solution: Decouple message delivery from ordering**

**We want:** Every node to see all messages in the same global order



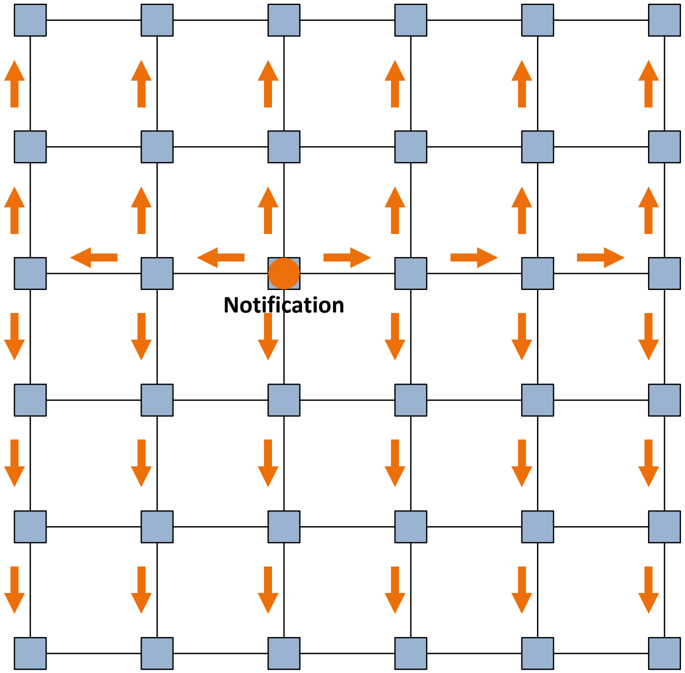
**Main network**

- Message delivery

**Notification network**

- Message ordering

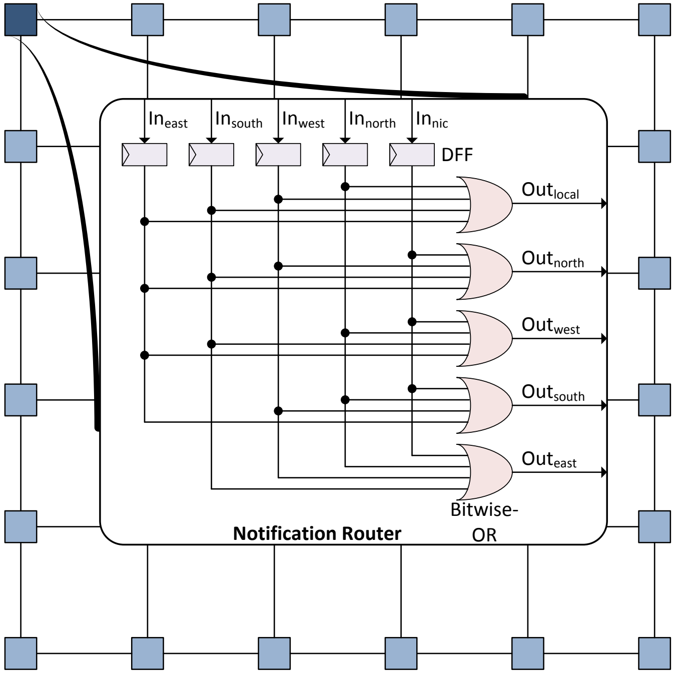
# Notification Network



## Bounded latency ( ≤ 12 cycle )

- Non-blocking
- 1 cycle / hop broadcast mesh
- Dedicated 1 bit / tile

# Notification Network



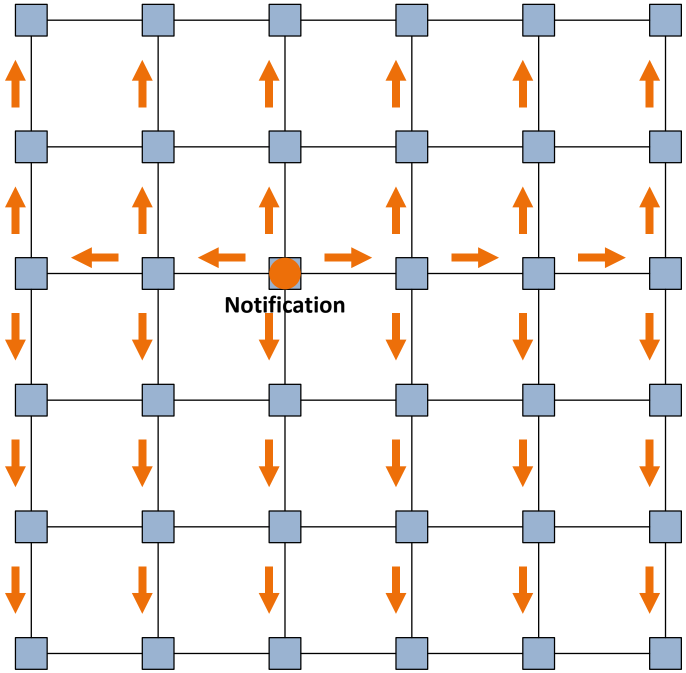
## Bounded latency ( ≤ 12 cycle )

- Non-blocking
- 1 cycle / hop broadcast mesh
- Dedicated 1 bit / tile

## Low cost

- Only DFF + ORs

# Notification Network



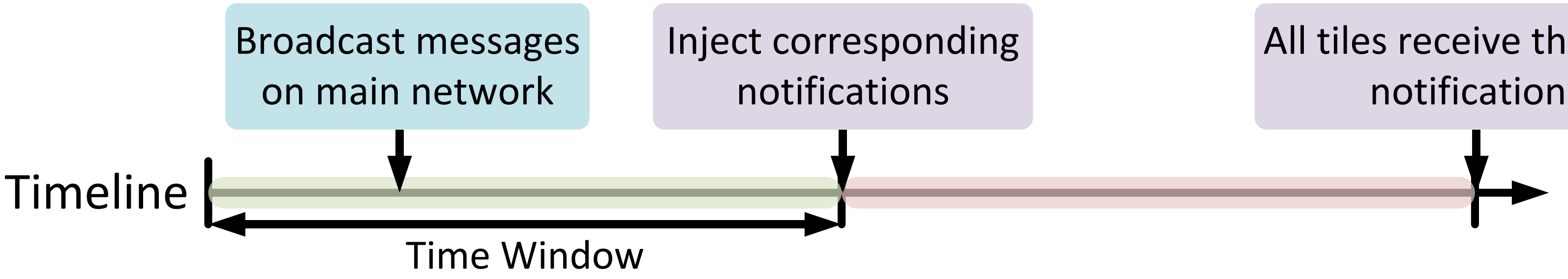
**Bounded latency ( ≤ 12 cycle )**

- Non-blocking
- 1 cycle / hop broadcast mesh
- Dedicated 1 bit / tile

**Low cost**

- Only DFF + ORs

**All tiles determine the global order locally**


Broadcast messages on main network

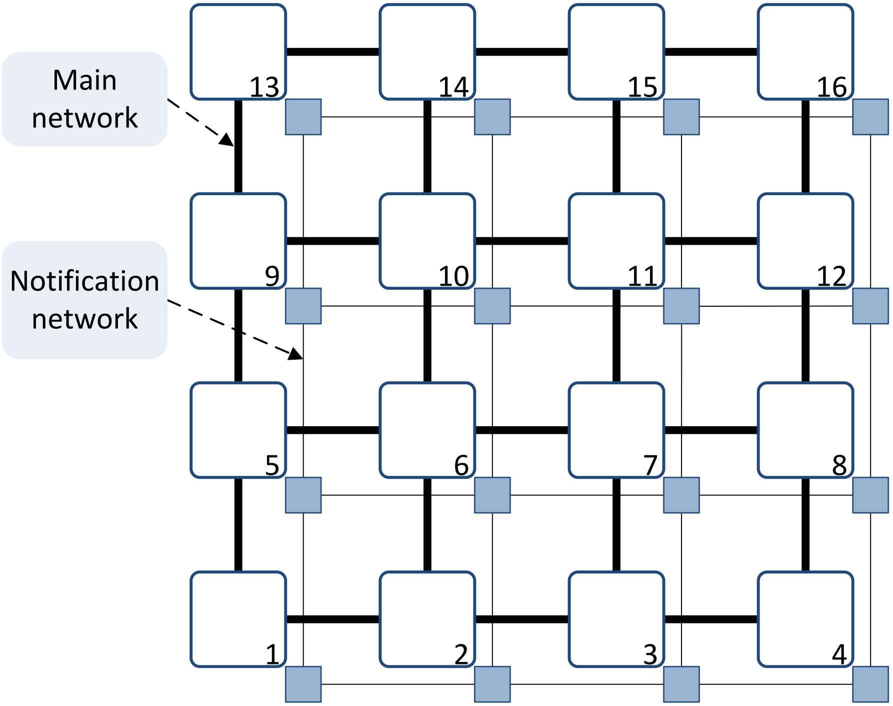
Inject corresponding notifications

All tiles receive the same notifications

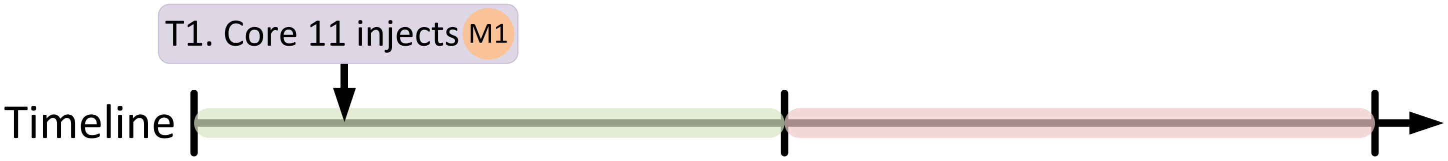
Timeline

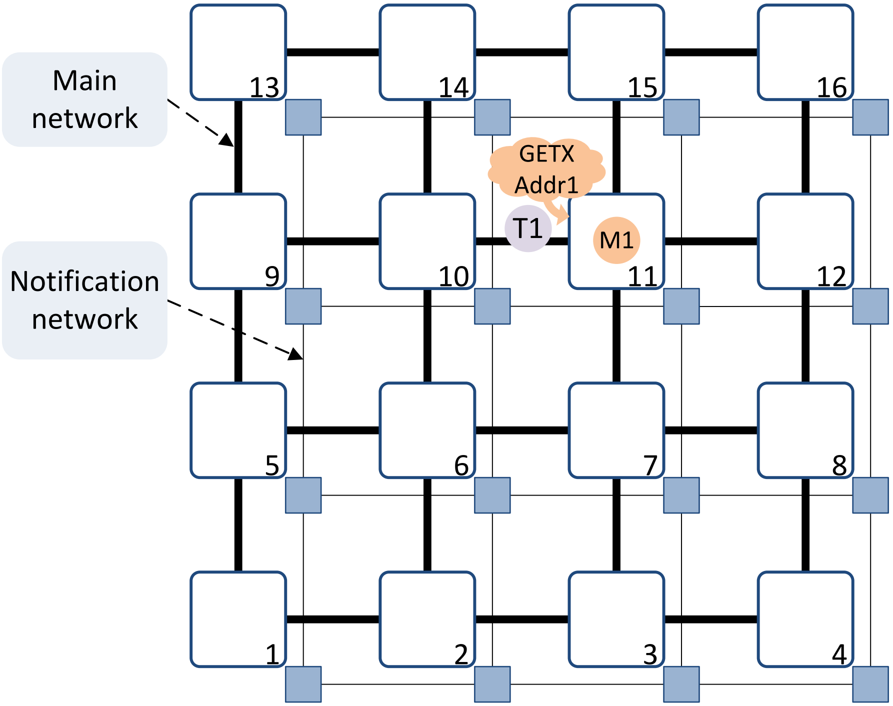
Time Window

# Walkthrough

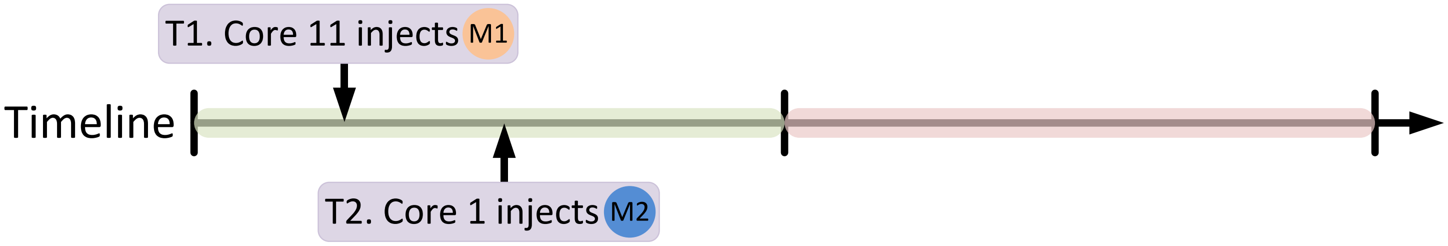


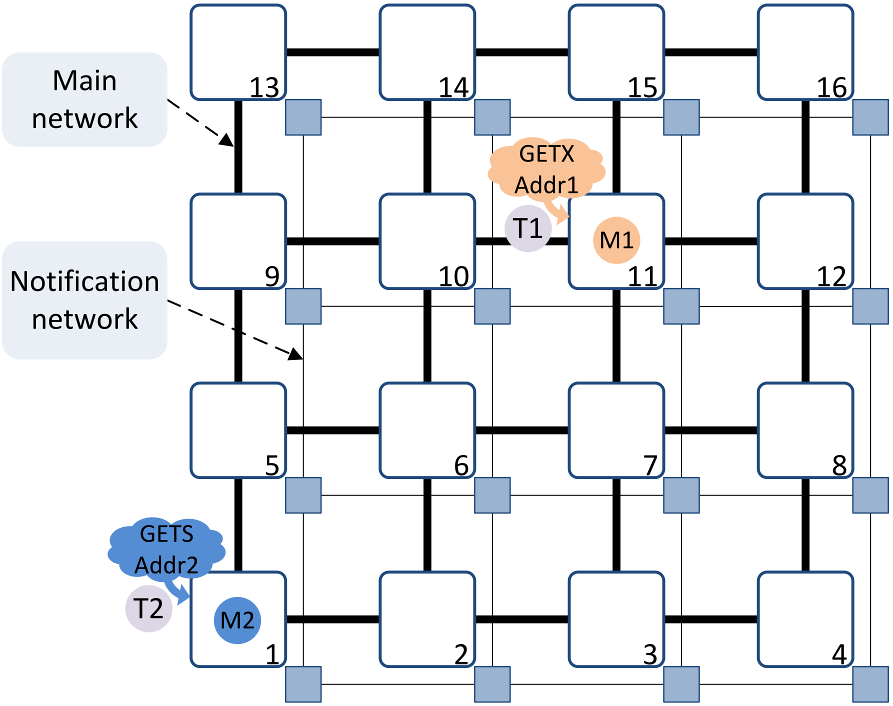
# Walkthrough



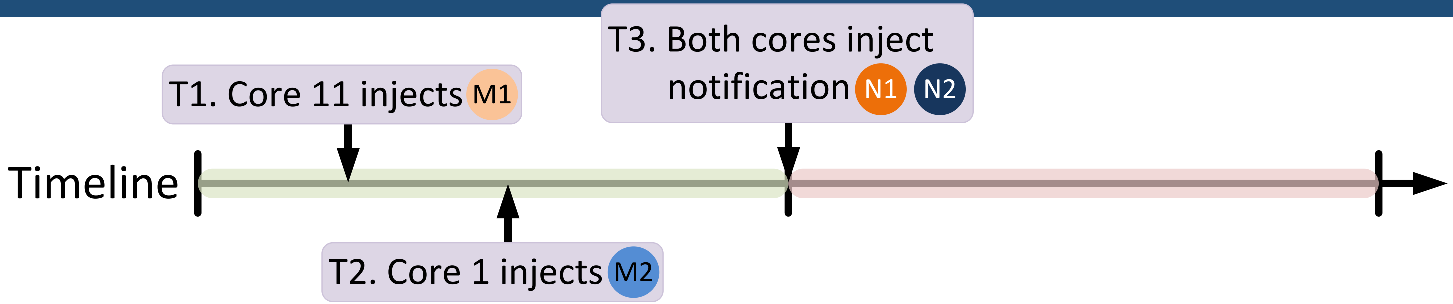


# Walkthrough





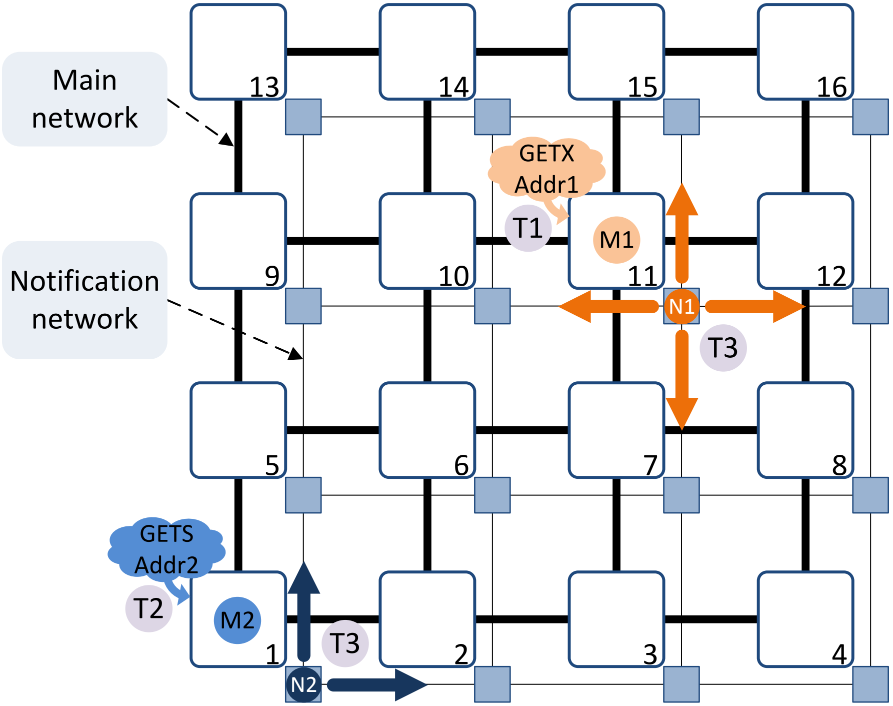
# Walkthrough



T1. Core 11 injects M1

T2. Core 1 injects M2

T3. Both cores inject notification N1 N2



Main network

Notification network

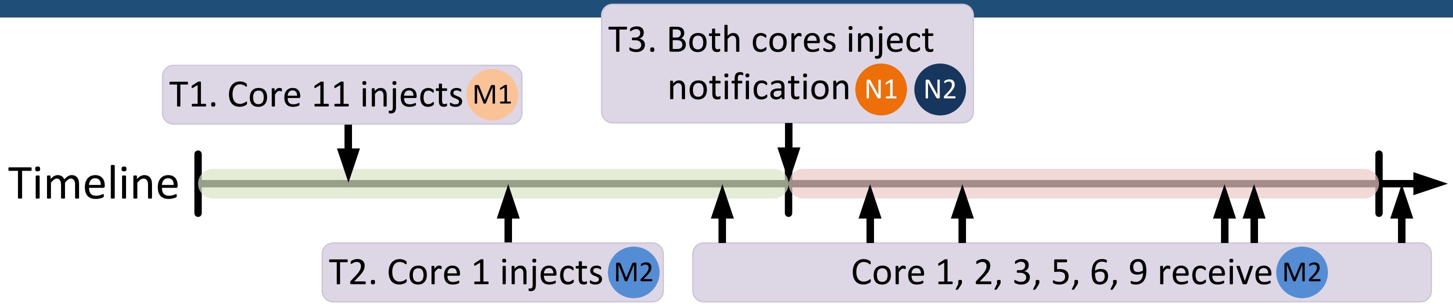
N1 Broadcast notification for M1

| 1 | 2 | 3 | ••• | 11 | ••• | 15 | 16 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | ••• | 1 | ••• | 0 | 0 |

N2 Broadcast notification for M2

| 1 | 2 | 3 | ••• | 11 | ••• | 15 | 16 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | ••• | 0 | ••• | 0 | 0 |

# Walkthrough



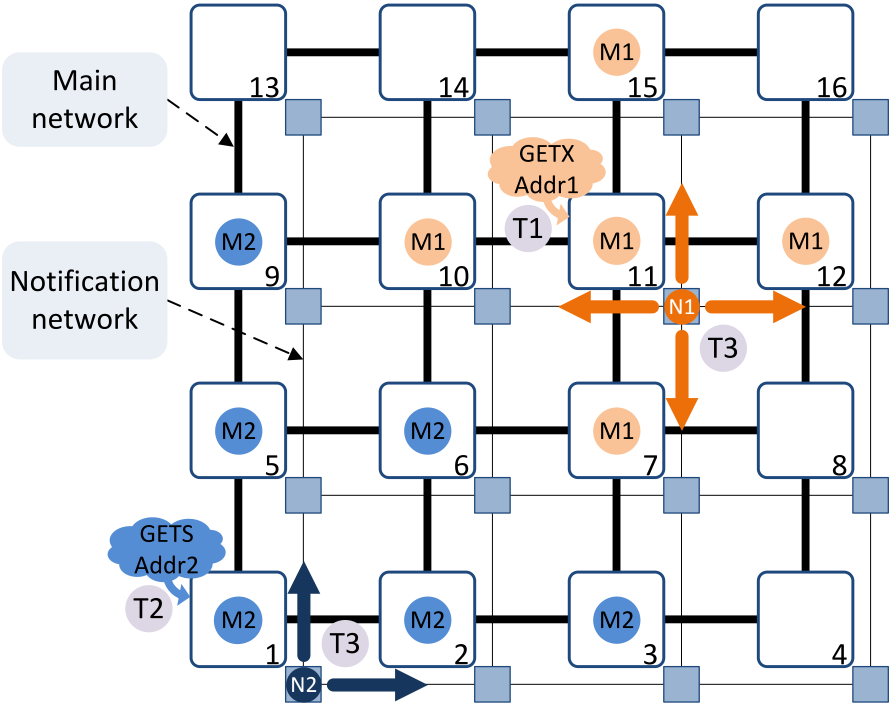
T1. Core 11 injects M1

T3. Both cores inject notification N1 N2

Timeline

T2. Core 1 injects M2

Core 1, 2, 3, 5, 6, 9 receive M2



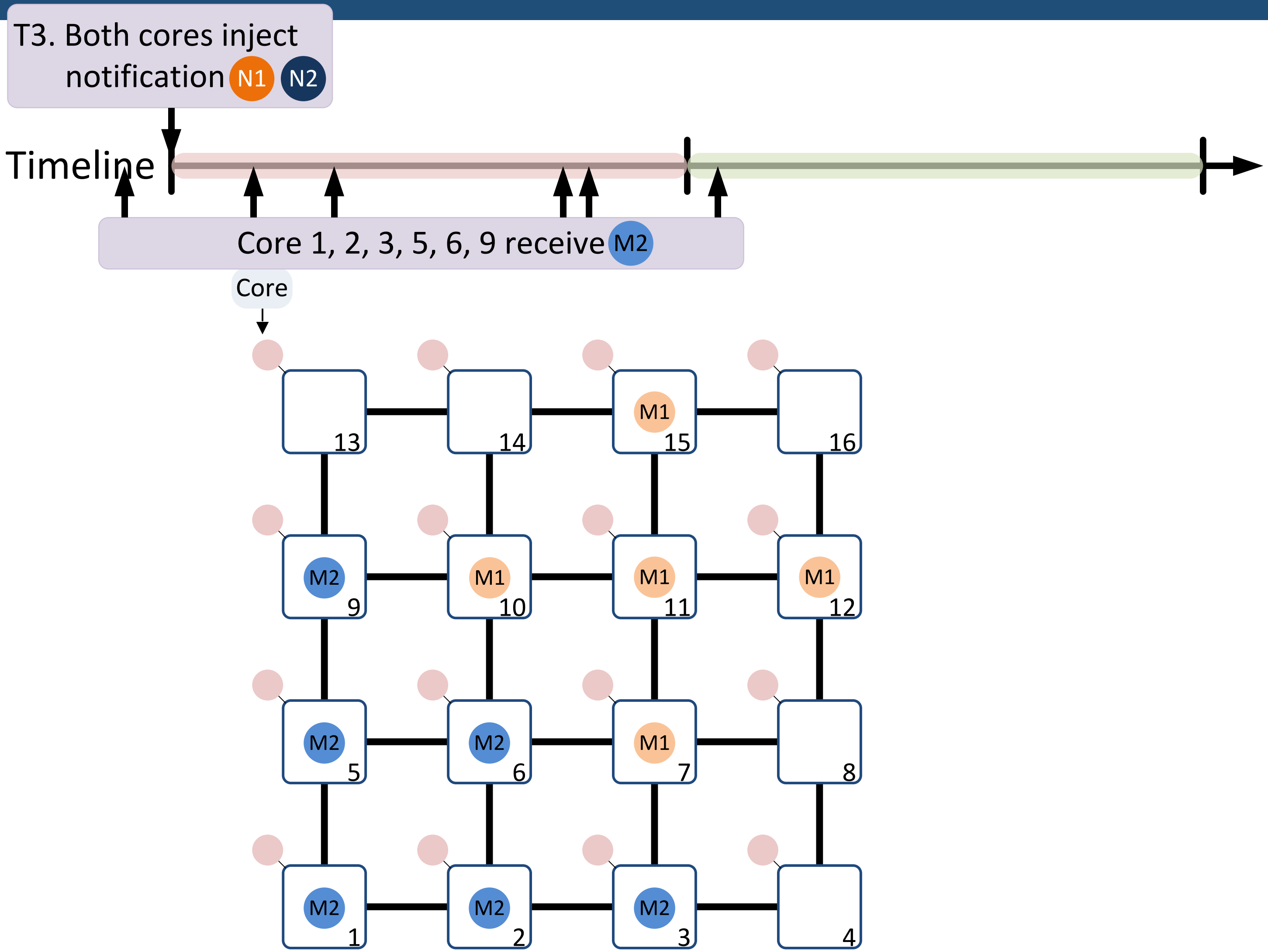
N1 Broadcast notification for M1

| 1 | 2 | 3 | ••• | 11 | ••• | 15 | 16 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | ••• | 1 | ••• | 0 | 0 |

N2 Broadcast notification for M2

| 1 | 2 | 3 | ••• | 11 | ••• | 15 | 16 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | ••• | 0 | ••• | 0 | 0 |

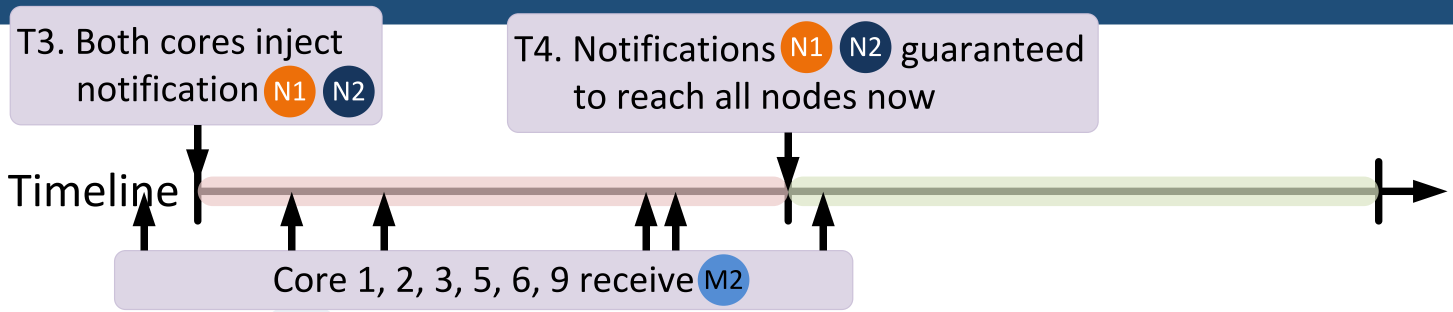
# Walkthrough



T3. Both cores inject notification N1 N2

Core 1, 2, 3, 5, 6, 9 receive M2

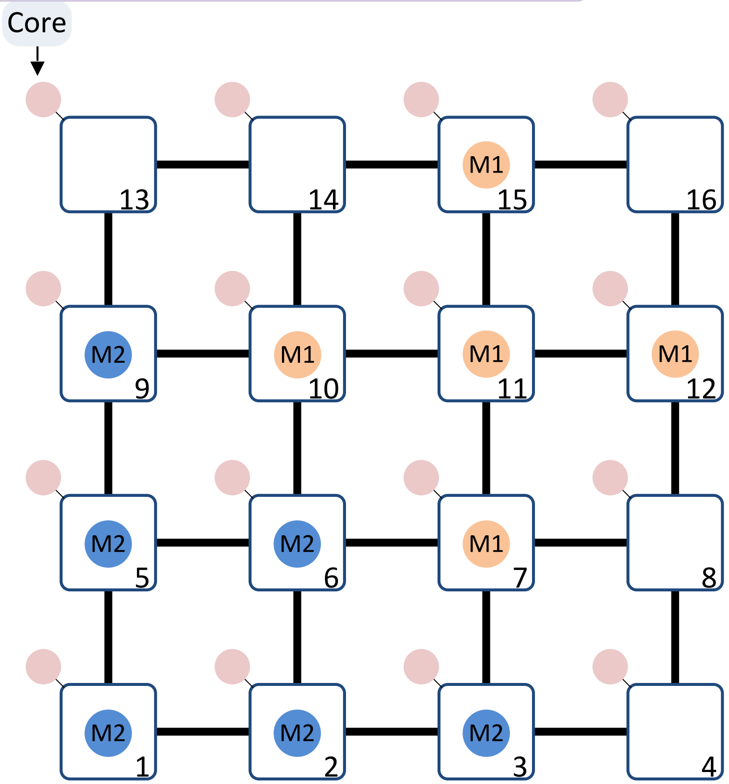
# Walkthrough



T3. Both cores inject notification N1 N2

T4. Notifications N1 N2 guaranteed to reach all nodes now

Core 1, 2, 3, 5, 6, 9 receive M2

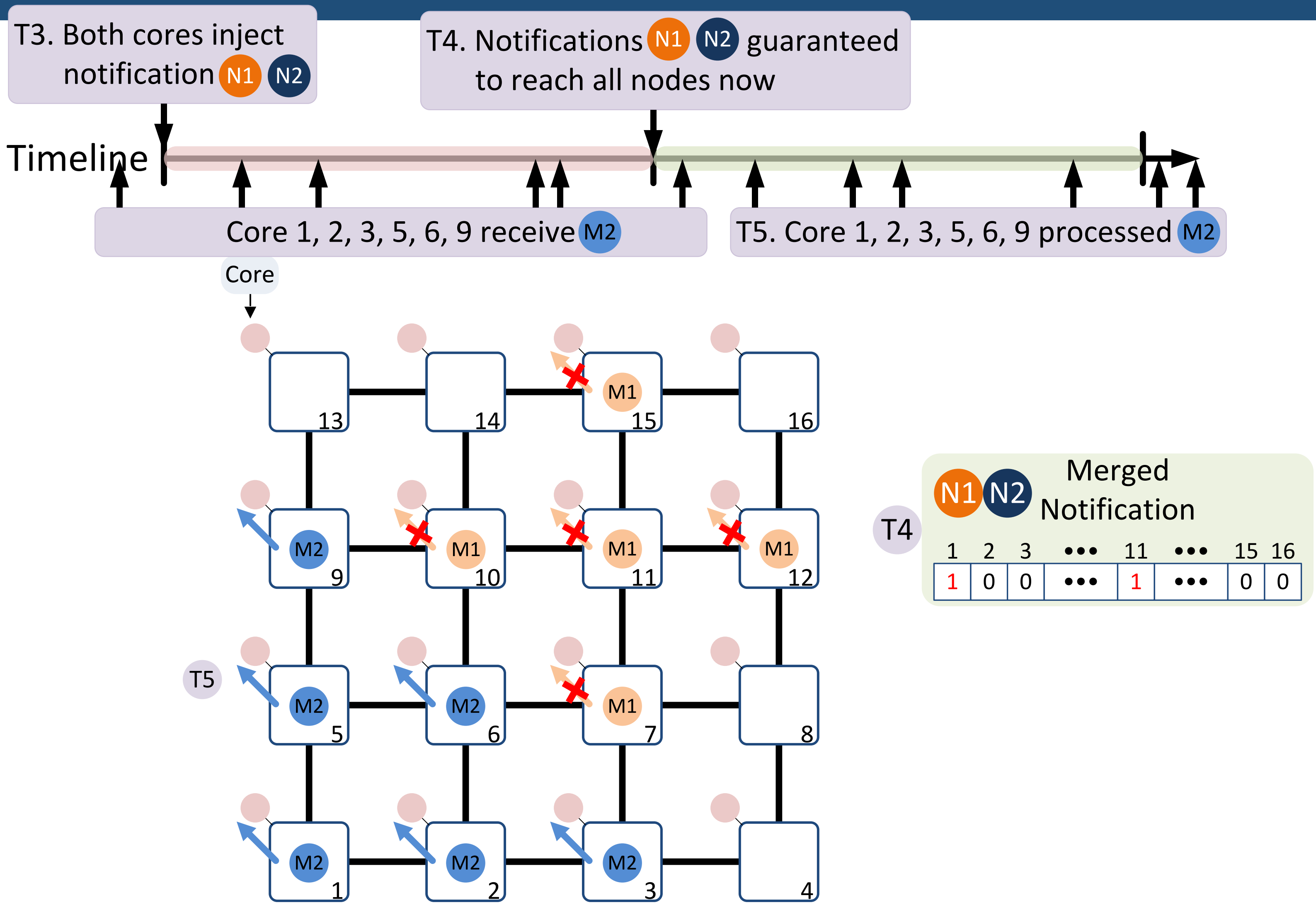


T4

N1 N2 Merged Notification

| 1 | 2 | 3 | ••• | 11 | ••• | 15 | 16 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | ••• | 1 | ••• | 0 | 0 |

# Walkthrough



T3. Both cores inject notification N1 N2

T4. Notifications N1 N2 guaranteed to reach all nodes now

Timeline

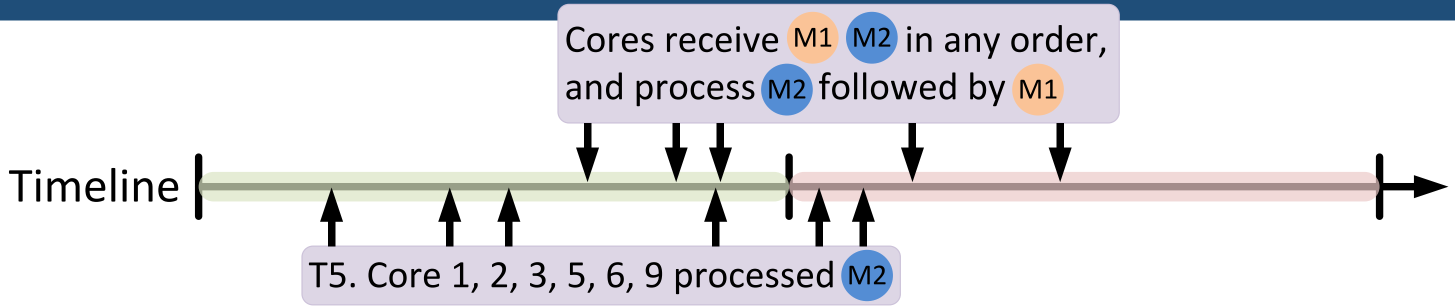
Core 1, 2, 3, 5, 6, 9 receive M2

T5. Core 1, 2, 3, 5, 6, 9 processed M2

N1 N2 Merged Notification

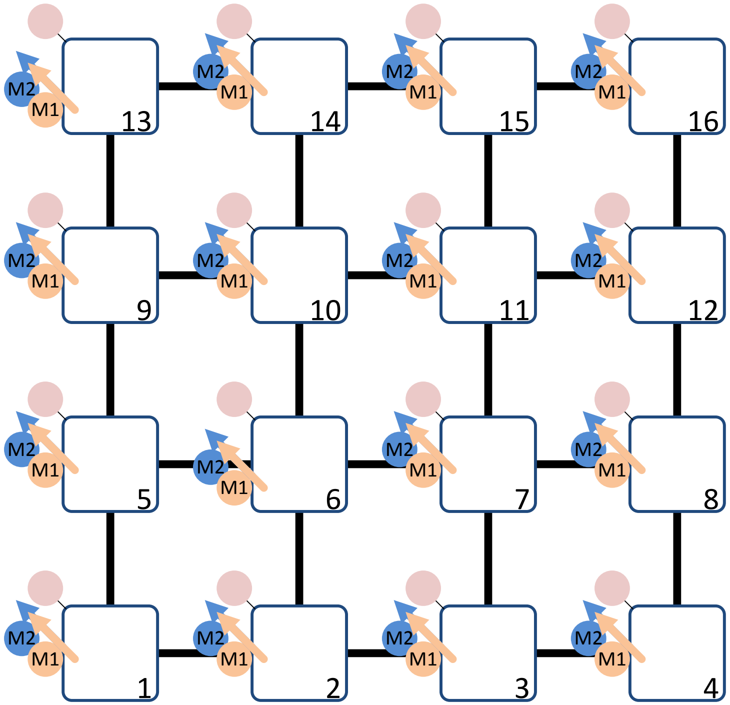
| 1 | 2 | 3 | ••• | 11 | ••• | 15 | 16 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | ••• | 1 | ••• | 0 | 0 |

# Walkthrough



Cores receive M1 M2 in any order, and process M2 followed by M1

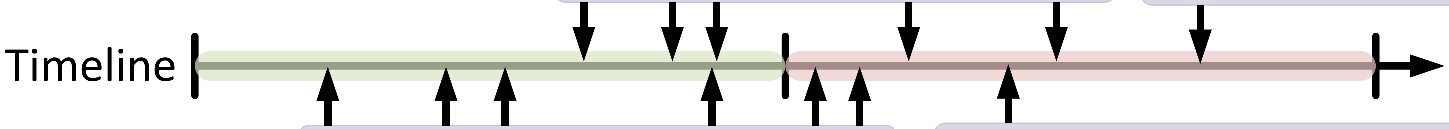
T5. Core 1, 2, 3, 5, 6, 9 processed M2



# Walkthrough

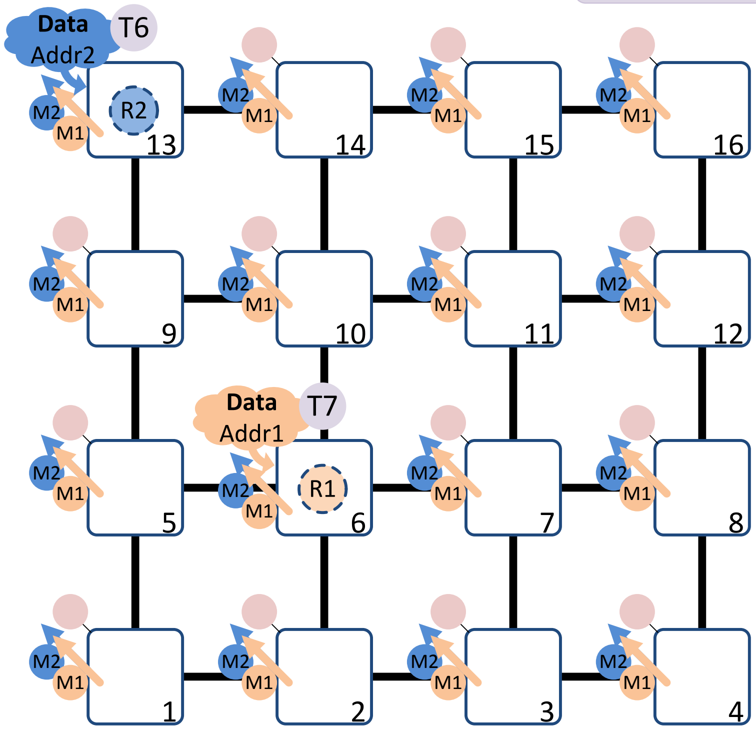
Cores receive M1 M2 in any order, and process M2 followed by M1

T7. Core 6, owner of Addr1, responds R1 with data to Core 11



T5. Core 1, 2, 3, 5, 6, 9 processed M2

T6. Core 13, owner of Addr2, responds R2 with data to Core 1



# Synchronization Primitives

## lwarx, stwcx

- Link in L2 cacheline granularity
- Detect modifications after load-link using coherence protocol

## msync

- Broadcast sync requests
- Gather acks from all cores when they complete the sync request

# Evaluation Setup

| | |
|---|---|
| Simulator | GEMS + GARNET |
| Access times | L1 – 1 cycle; L2 – 10 cycles; DRAM 90 cycles |
| LPD | Limited Pointer Directory Coherence |
| HT | AMD HyperTransport Coherence |
| SCORPIO | Snoopy Coherence: MOSI |

# Evaluation Setup

| | |
|---|---|
| Simulator | GEMS + GARNET |
| Access times | L1 – 1 cycle; L2 – 10 cycles; DRAM 90 cycles |
| LPD | Limited Pointer Directory Coherence |
| HT | AMD HyperTransport Coherence |
| SCORPIO | Snoopy Coherence: MOSI |

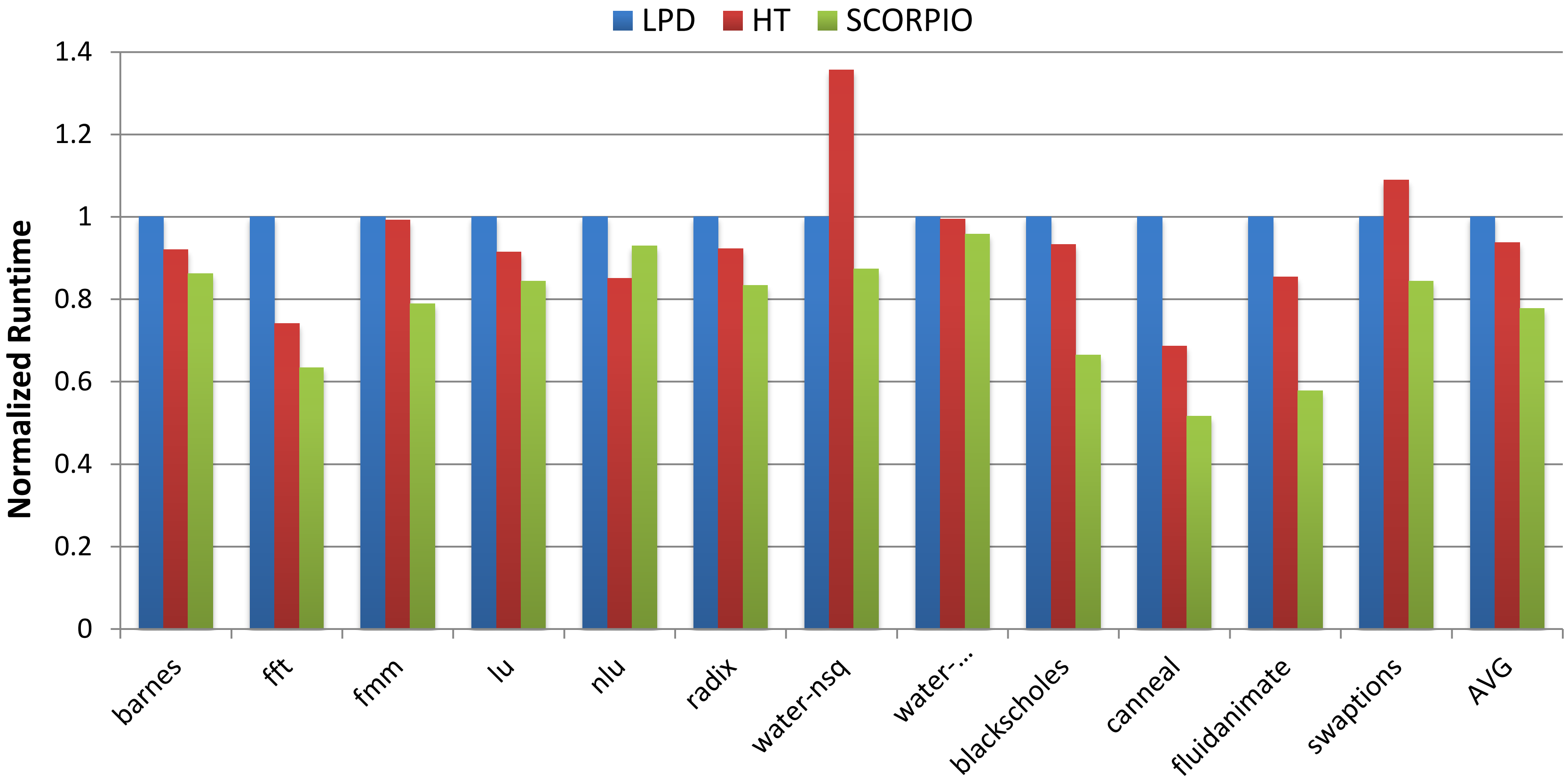
| | LPD | HT | SCORPIO |
|---|---|---|---|
| What is tracked? | Few sharers | Presence of owner | Presence of owner |
| Who orders requests? | Directory | Directory | Network |

Isolate

Storage overhead

Indirection latency

# Runtime Comparison

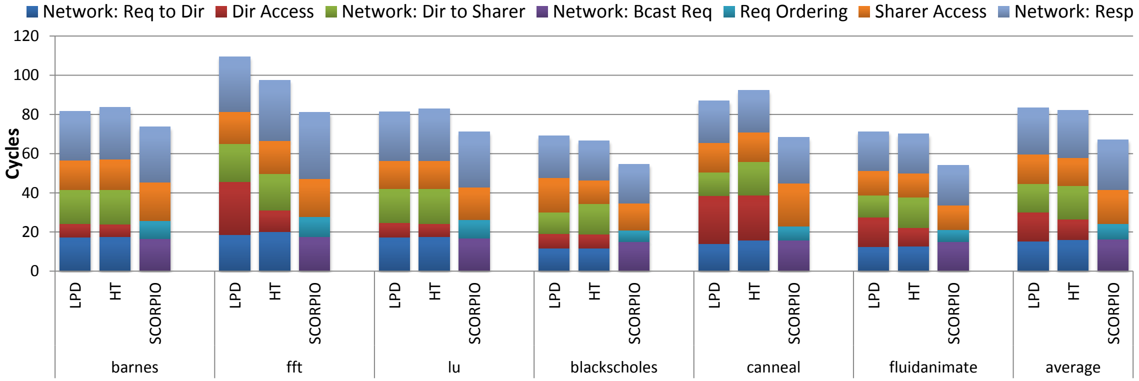


→ 24% better than Limited Pointer Directory

→ 13% better than Hyper-Transport

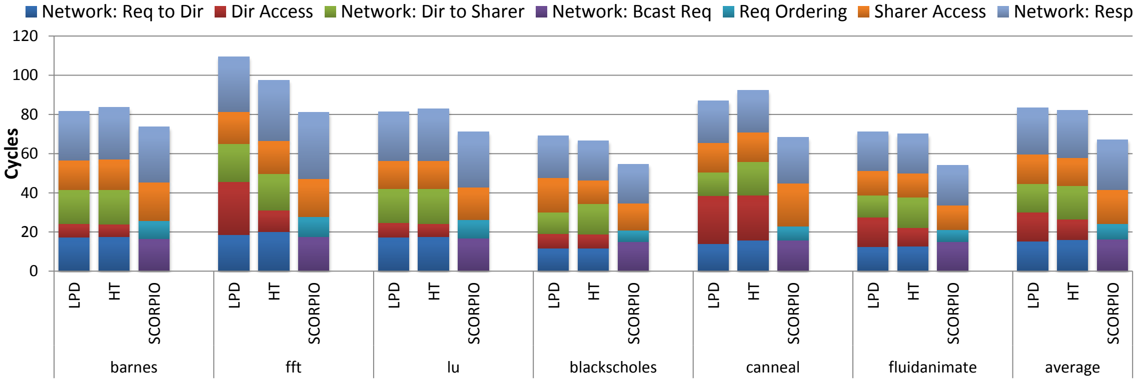
# L2 Service Latency

## Requests served by other caches



# L2 Service Latency

## Requests served by other caches

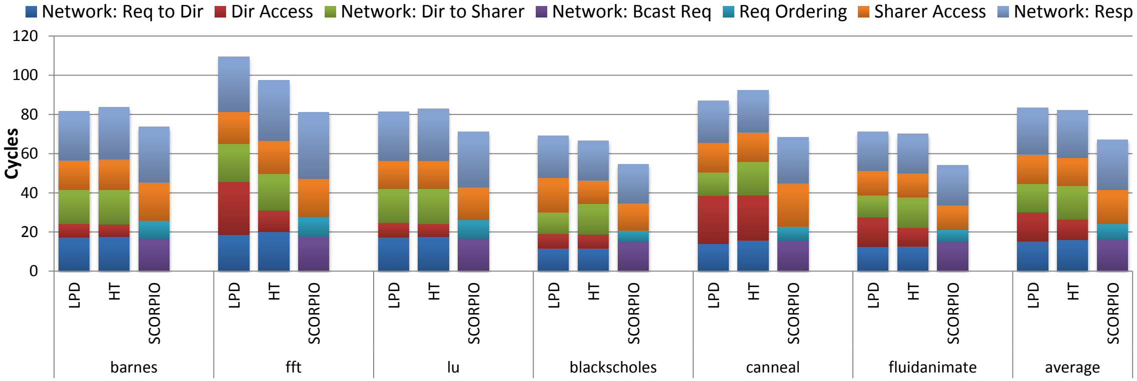


→ 19% lower than LPD

→ 18% lower than HT

# L2 Service Latency

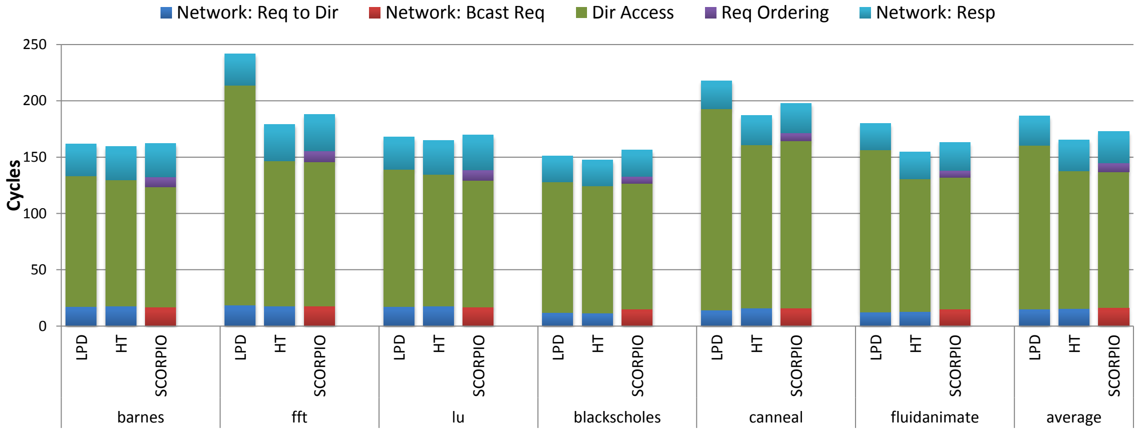
## Requests served by other caches



→19% lower than LPD

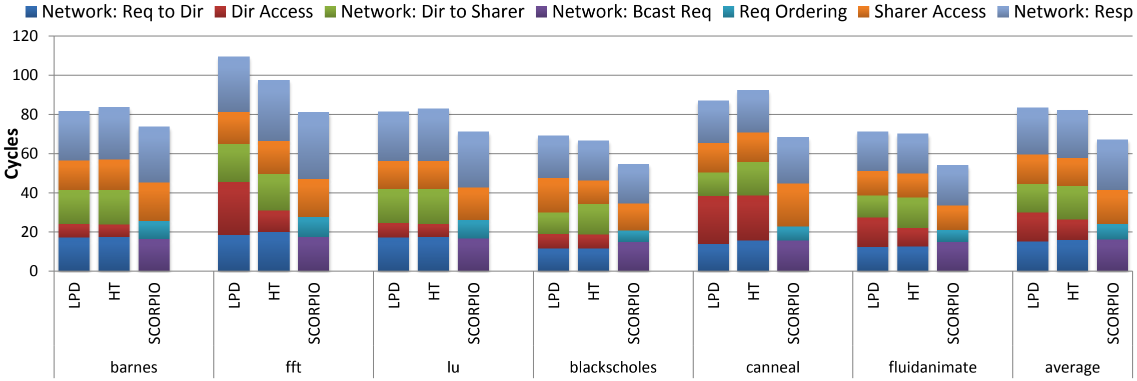
→18% lower than HT

## Requests served by directory -- MC



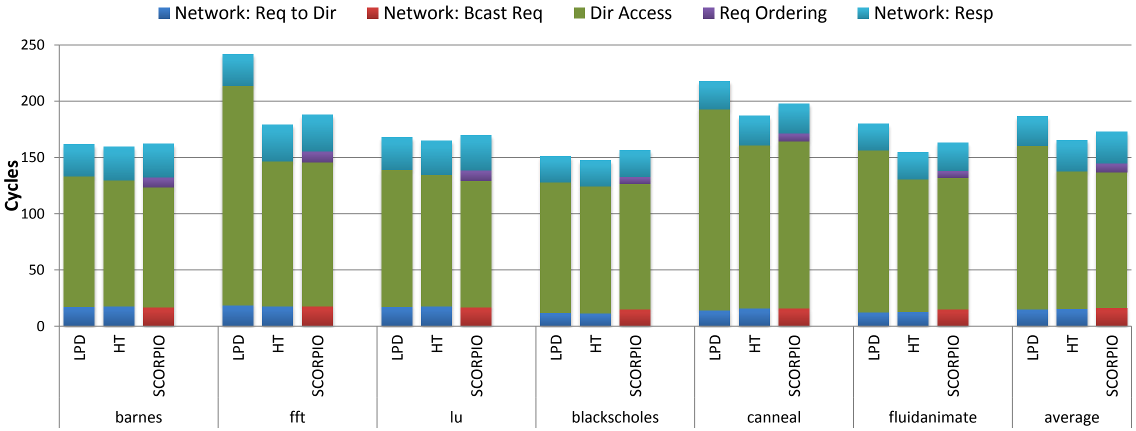
# L2 Service Latency

## Requests served by other caches



- ➔ 19% lower than LPD
- ➔ 18% lower than HT

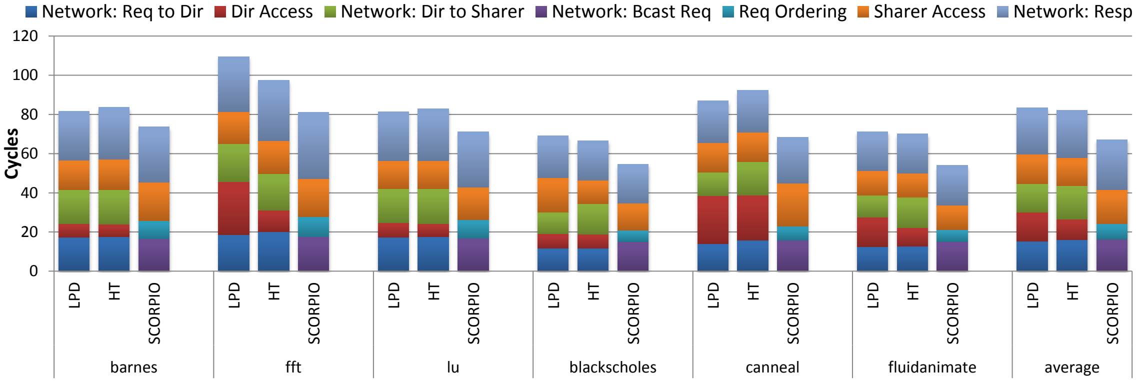
## Requests served by directory -- MC



- ➔ 7.5% lower than LPD
- ➔ 4.2% higher than HT

# L2 Service Latency

## Requests served by other caches

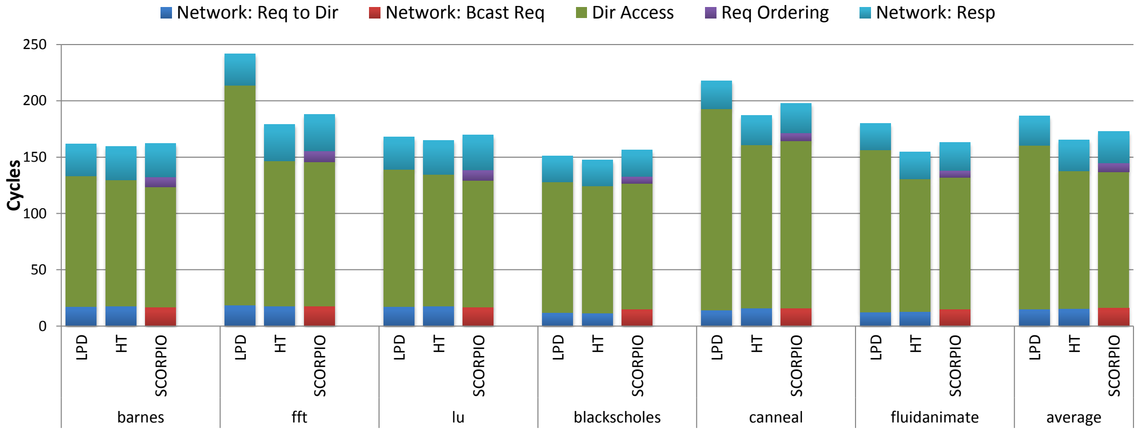


→19% lower than LPD

→18% lower than HT

90% requests served by other caches

## Requests served by directory -- MC

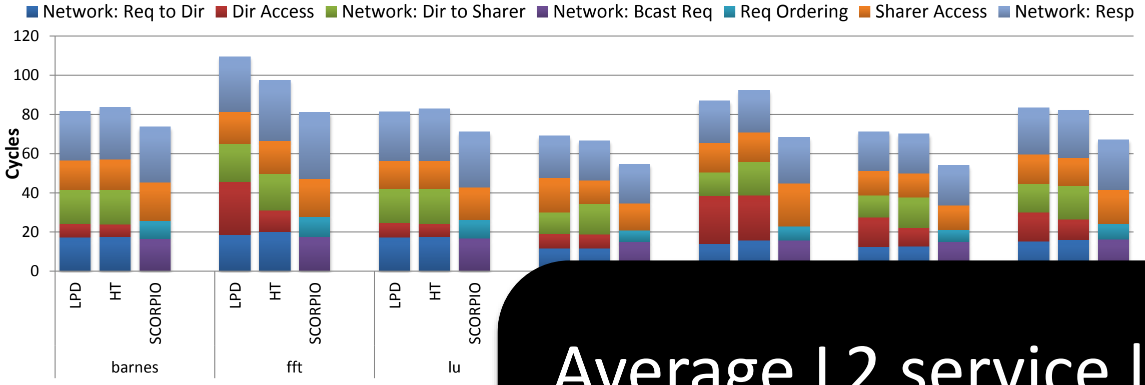


→7.5% lower than LPD

→4.2% higher than HT

# L2 Service Latency

## Requests served by other caches



→19% lower than LPD

→18% lower than HT

Average L2 service latency

→17% lower than LPD

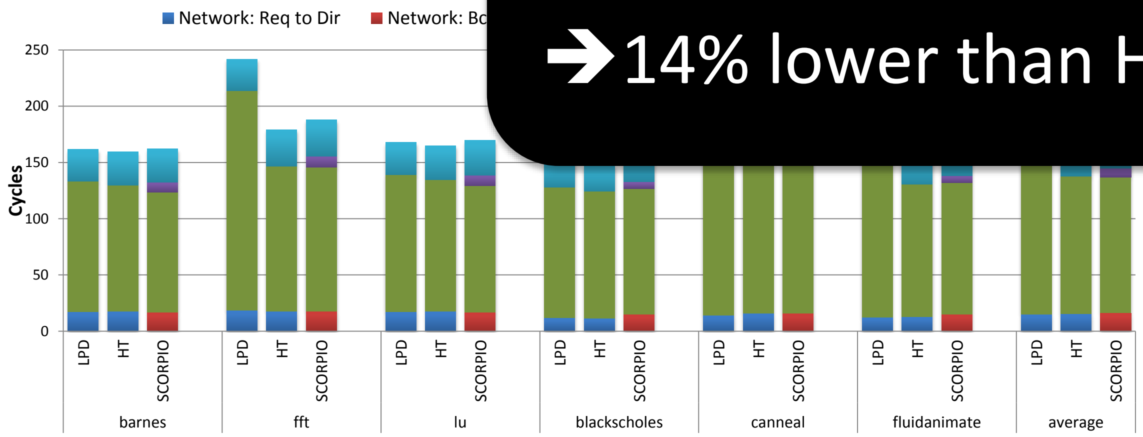
→14% lower than HT

equests

by other

ches

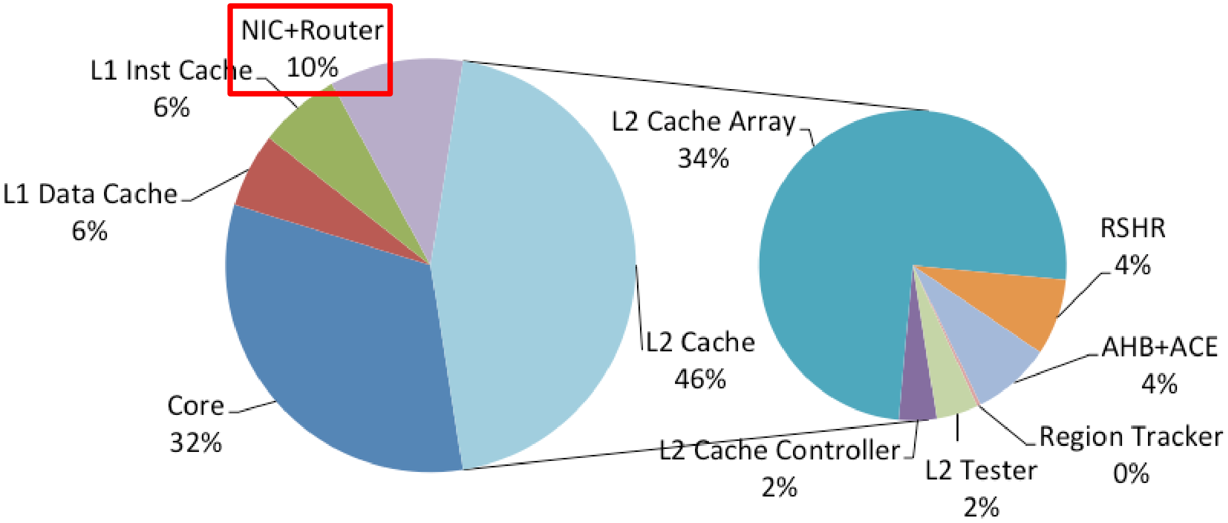
## Requests served by di



→7.5% lower than LPD

→4.2% higher than HT

# Network Cost

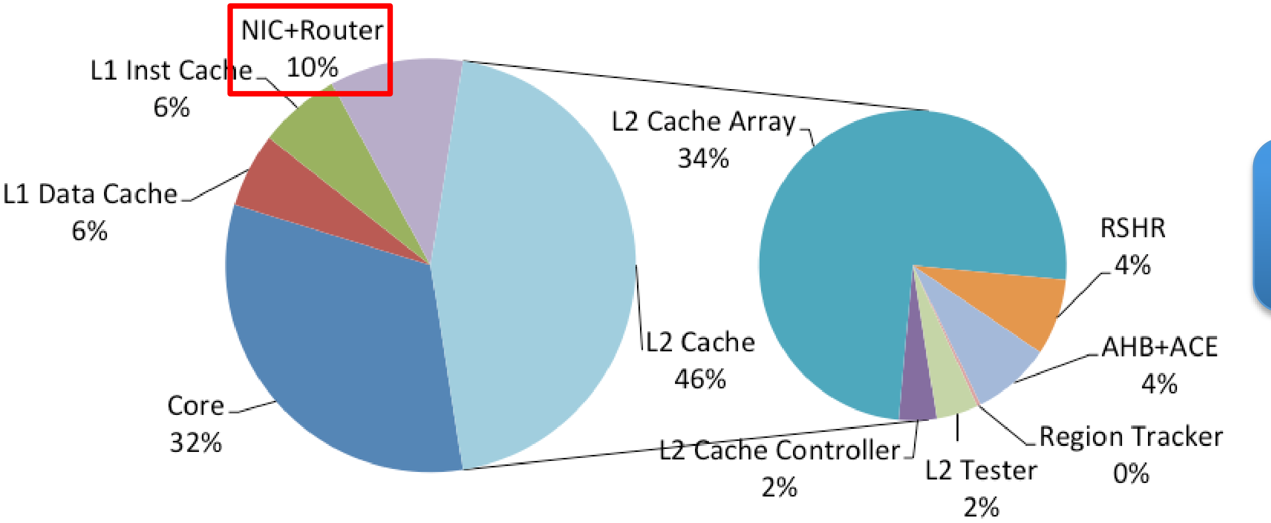


Network occupies only 10% of the area

**Area**

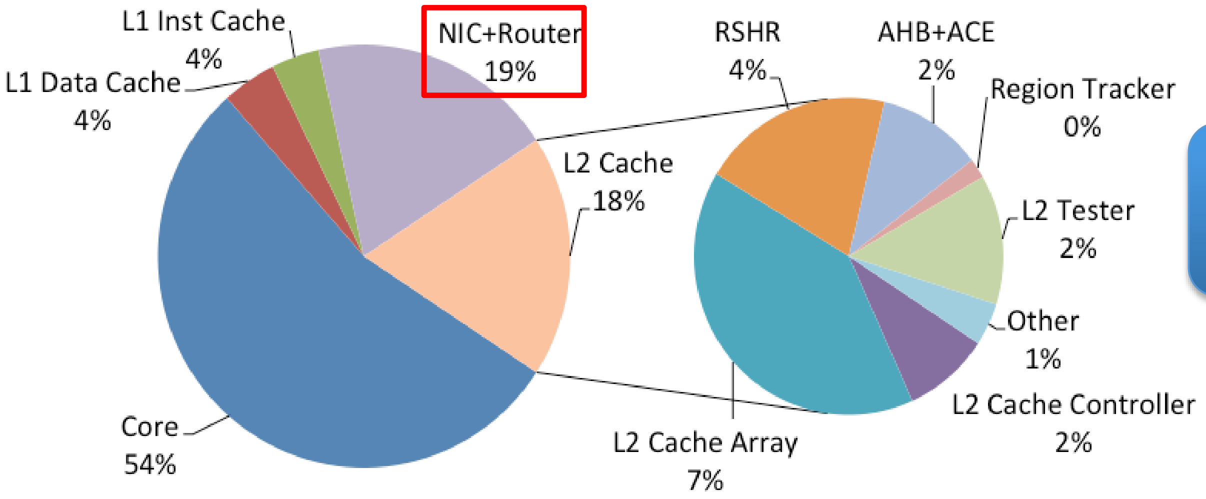
**Post-layout frequency: 833 MHz**

# Network Cost



Network occupies only 10% of the area

**Area**



Network consumes 20% of the power

**Power**

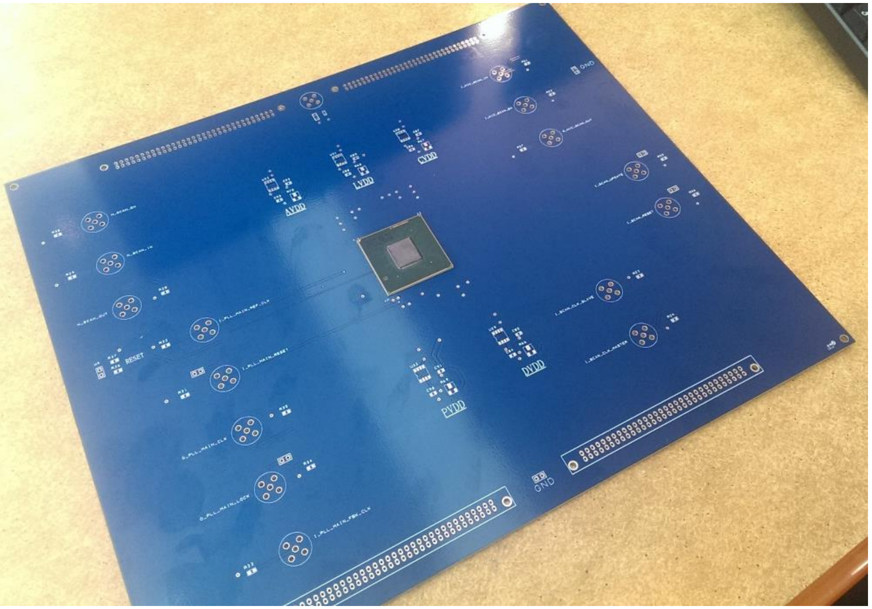
**Post-layout frequency: 833 MHz**

# Contributions

- **SCORPIO: A 36-core shared-memory processor**
  **Snoopy coherency on a mesh interconnect:**
  - Runtime: 24% better than LPD, 13% better than HT
  - Cost: 28.8W @ 833MHz

- **Novel network-on-chip for scalable snoopy coherence**
  **New ideas:**
  - Distributed in-network ordering mechanism
  - Decouple message delivery from message ordering

# Ongoing Work

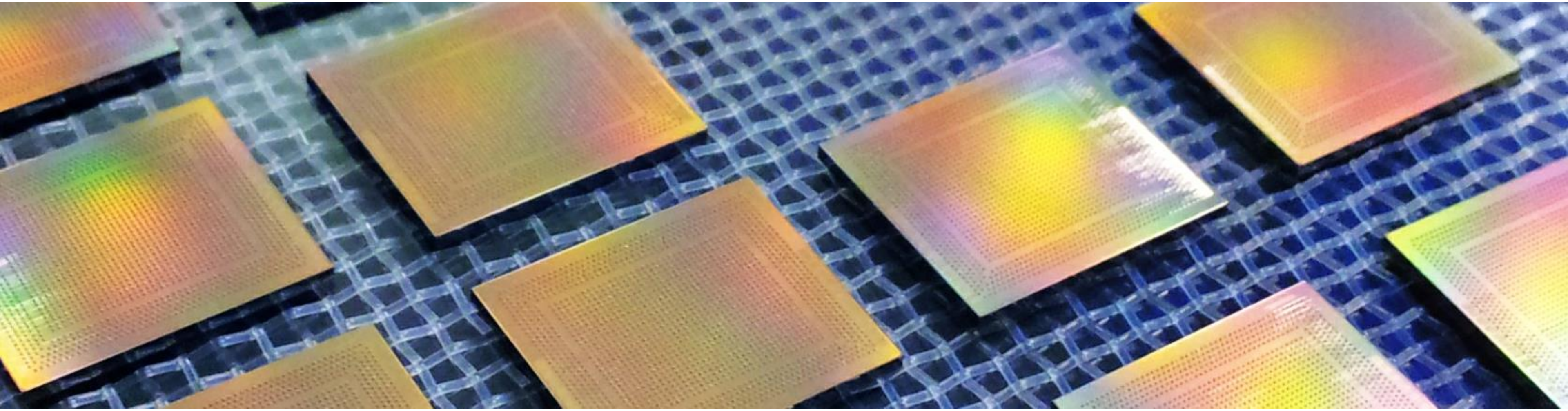


## Software stack development

- Boot Linux
- Run PARSEC, SPLASH, …, etc

## Chip measurement

- Power, timing
- Performance



# THANK YOU!