Memory Architecture and Management in a NoC Platform

Axel Jantsch, Xiaowen Chen, Abdul Naeem,
Yuang Zhang, Sandro Penolazzi, Zhonghai Lu

December 1, 2010

Overview

- Motivation
- Data Management Engine Architecture
- DME Programming
- Synchronization
- Cache coherence
- Memory consistency
- Experiments

The Memory Wall

Wulf and McKee predicted in 1995 the impact into the memory wall:

$$t_{\text{avg}} = p \times t_c + (1 - p) \times t_m$$

t_{avg}...average access latency for one data word
p ...probability of a cache hit
t_c ...access time to the cache
t_m ...access time to main memory

The Memory Wall

Wulf and McKee predicted in 1995 the impact into the memory wall:

$$t_{\text{avg}} = p \times t_c + (1 - p) \times t_m$$

t_{avg}...average access latency for one data word
p ...probability of a cache hit
t_c ...access time to the cache
t_m ...access time to main memory

Today we navigate at the edge of this wall.

The Memory Wall

Wulf and McKee predicted in 1995 the impact into the memory wall:

$$t_{\text{avg}} = p \times t_c + (1 - p) \times t_m$$

t_{avg}...average access latency for one data word
p ...probability of a cache hit
t_c ...access time to the cache
t_m ...access time to main memory

Today we navigate at the edge of this wall.

For example the Tilera 64 core chip:

- Raw processor performance: 443 Gops

The Memory Wall

Wulf and McKee predicted in 1995 the impact into the memory wall:

$$t_{\text{avg}} = p \times t_c + (1 - p) \times t_m$$

t_{avg}...average access latency for one data word
p ...probability of a cache hit
t_c ...access time to the cache
t_m ...access time to main memory

Today we navigate at the edge of this wall.

For example the Tilera 64 core chip:

- Raw processor performance: 443 Gops
- Required memory bandwidth with two cache levels and 95% hit rate: 7.2Gb/s
- Available memory bandwidth: 50 Gb/s

The Memory Wall

Wulf and McKee predicted in 1995 the impact into the memory wall:

$$t_{\text{avg}} = p \times t_c + (1 - p) \times t_m$$

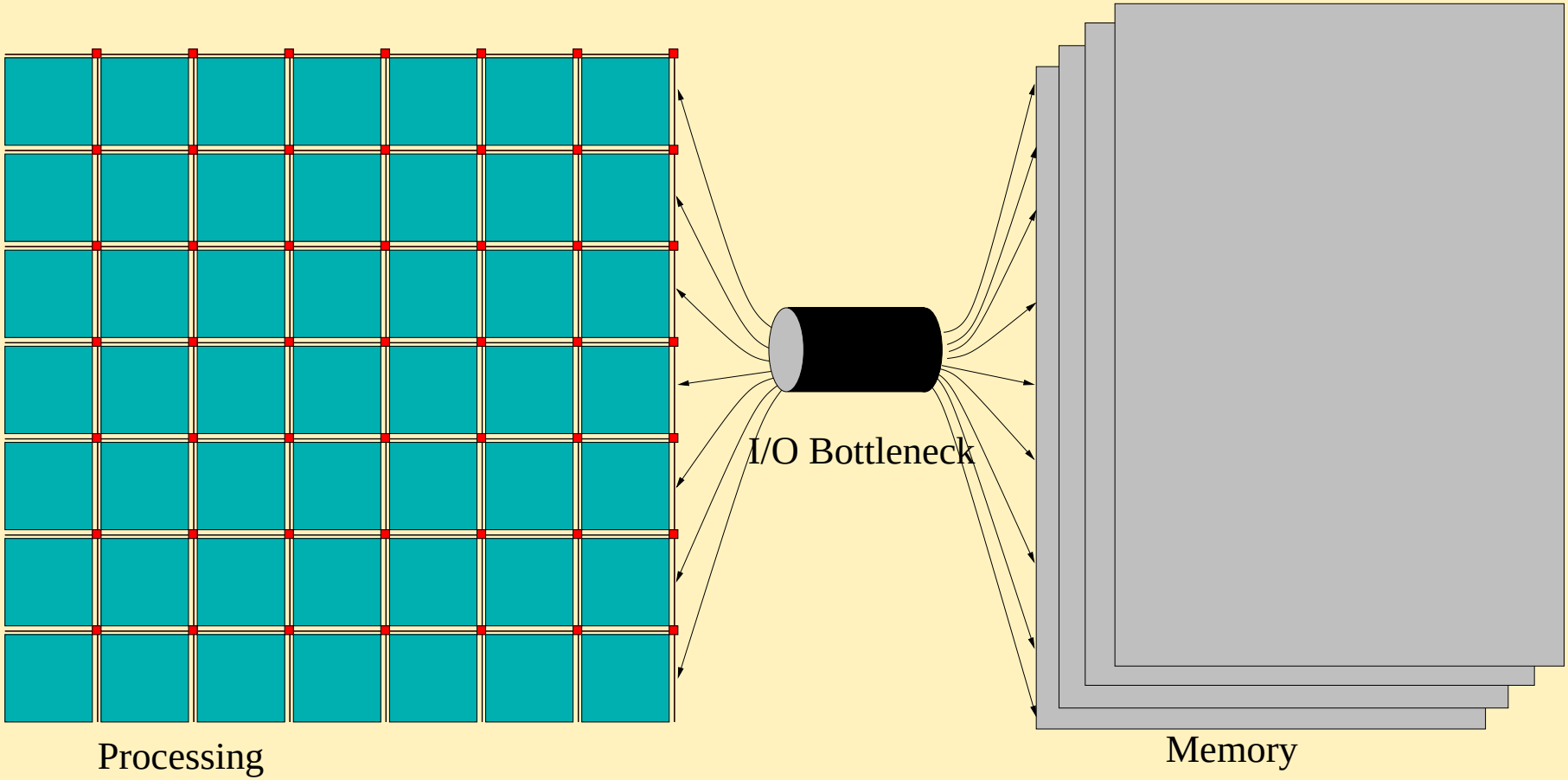
t_{avg}...average access latency for one data word
p ...probability of a cache hit
t_c ...access time to the cache
t_m ...access time to main memory

Today we navigate at the edge of this wall.

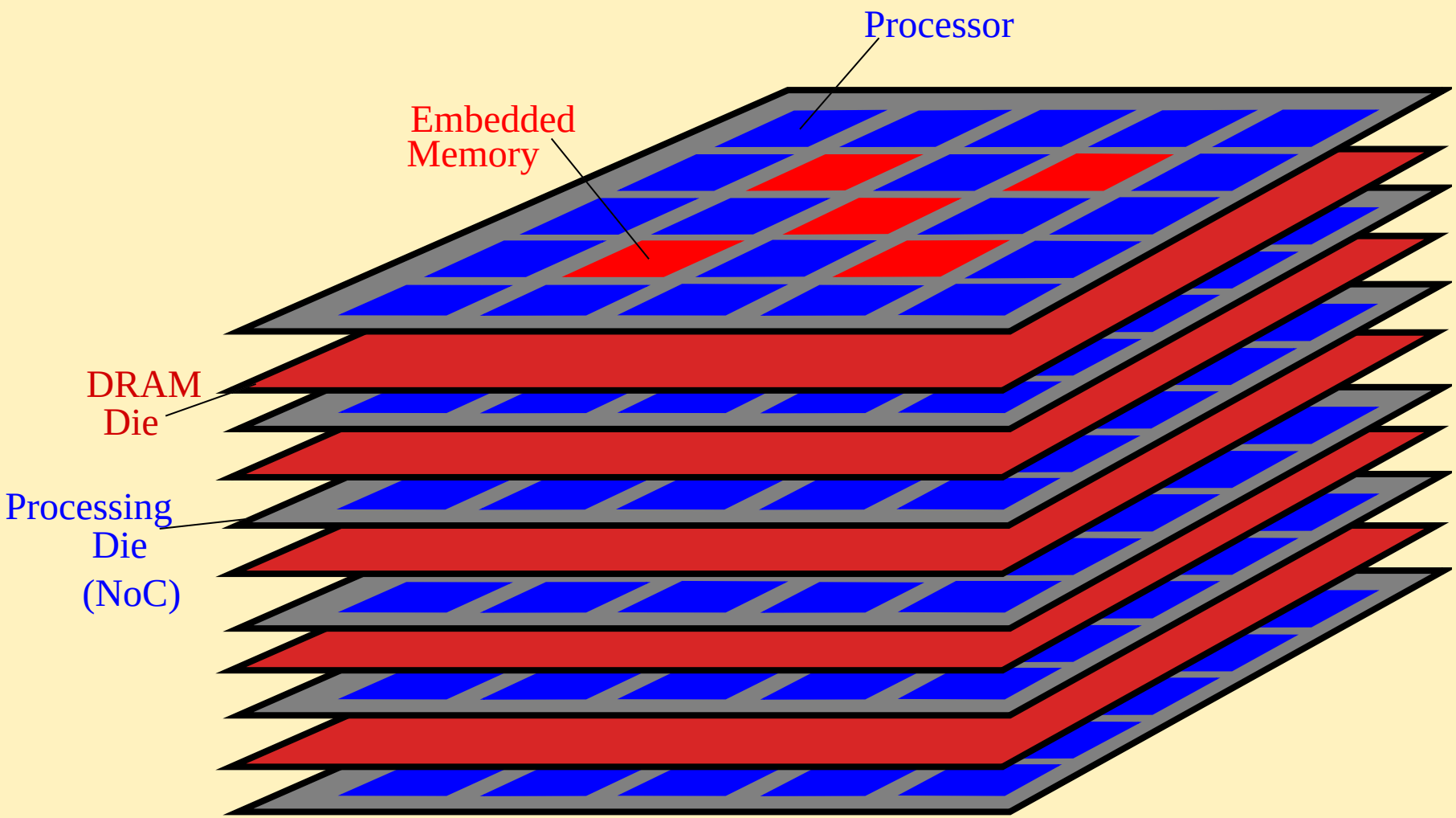
For example the Tilera 64 core chip:

- Raw processor performance: 443 Gops
- Required memory bandwidth with two cache levels and 95% hit rate: 7.2Gb/s
- Available memory bandwidth: 50 Gb/s
- Required memory access latency: 0.625 cycles
- Available average access time: 1.475 cycles

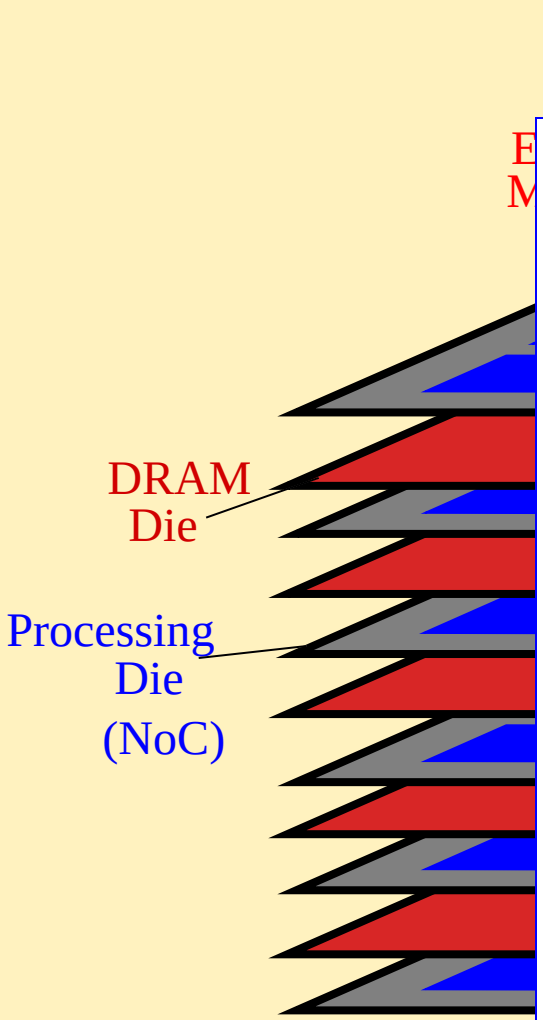
The Memory Access Bottleneck



Memory Bandwidth in 3D ICs

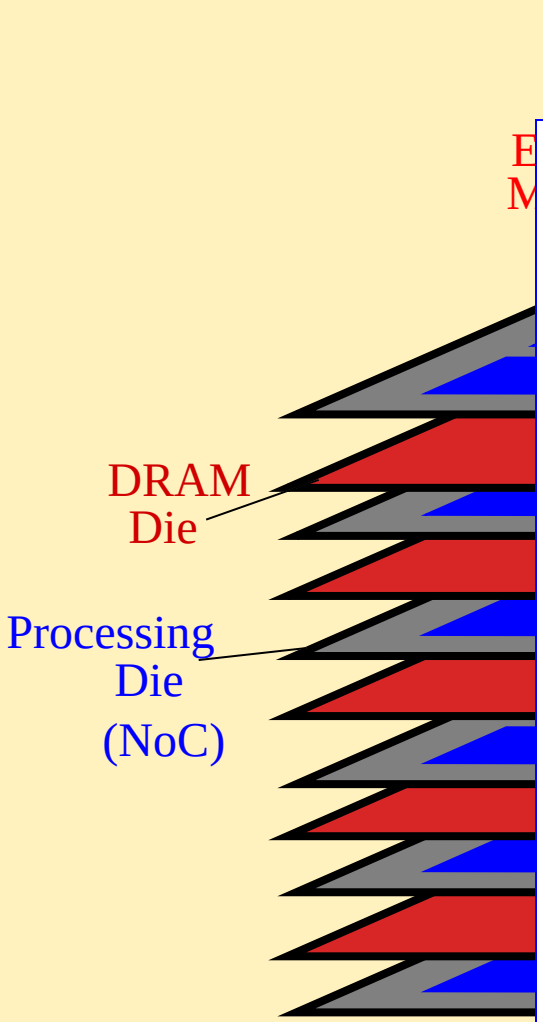


Memory Bandwidth in 3D ICs



- 8x8 MPSoC
- Resource size: 2x2 mm2
- Switch size: 100x100 um2
- TSV pitch: 5 um
- TSVs/switch: 128@2GHz
- 256 Gb/s memory bandwidth per core
- 16 Tb/s aggregate memory bandwidth (200 Gb/s for Tilera)
- $<$ 10 ns delay

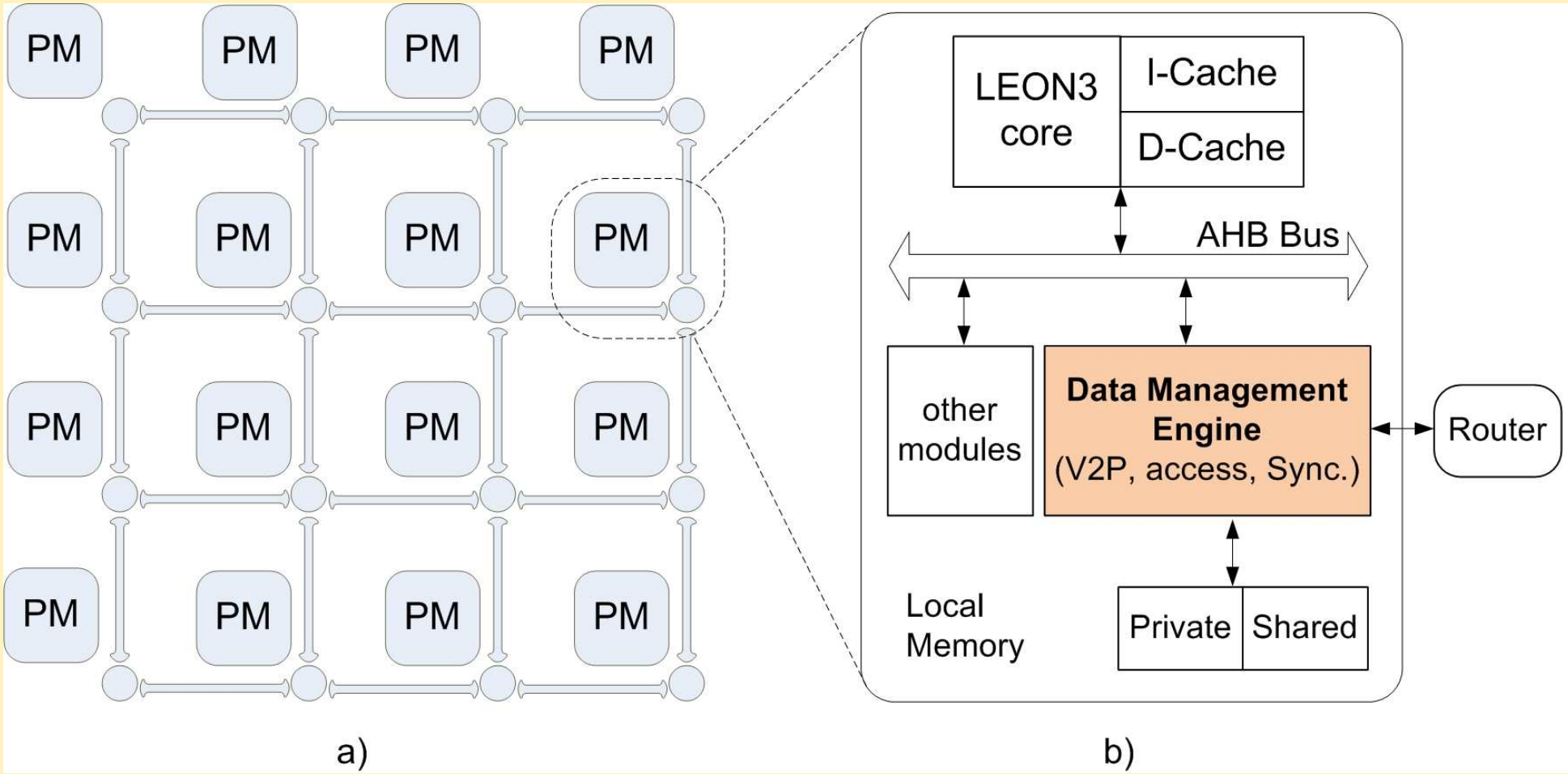
Memory Bandwidth in 3D ICs



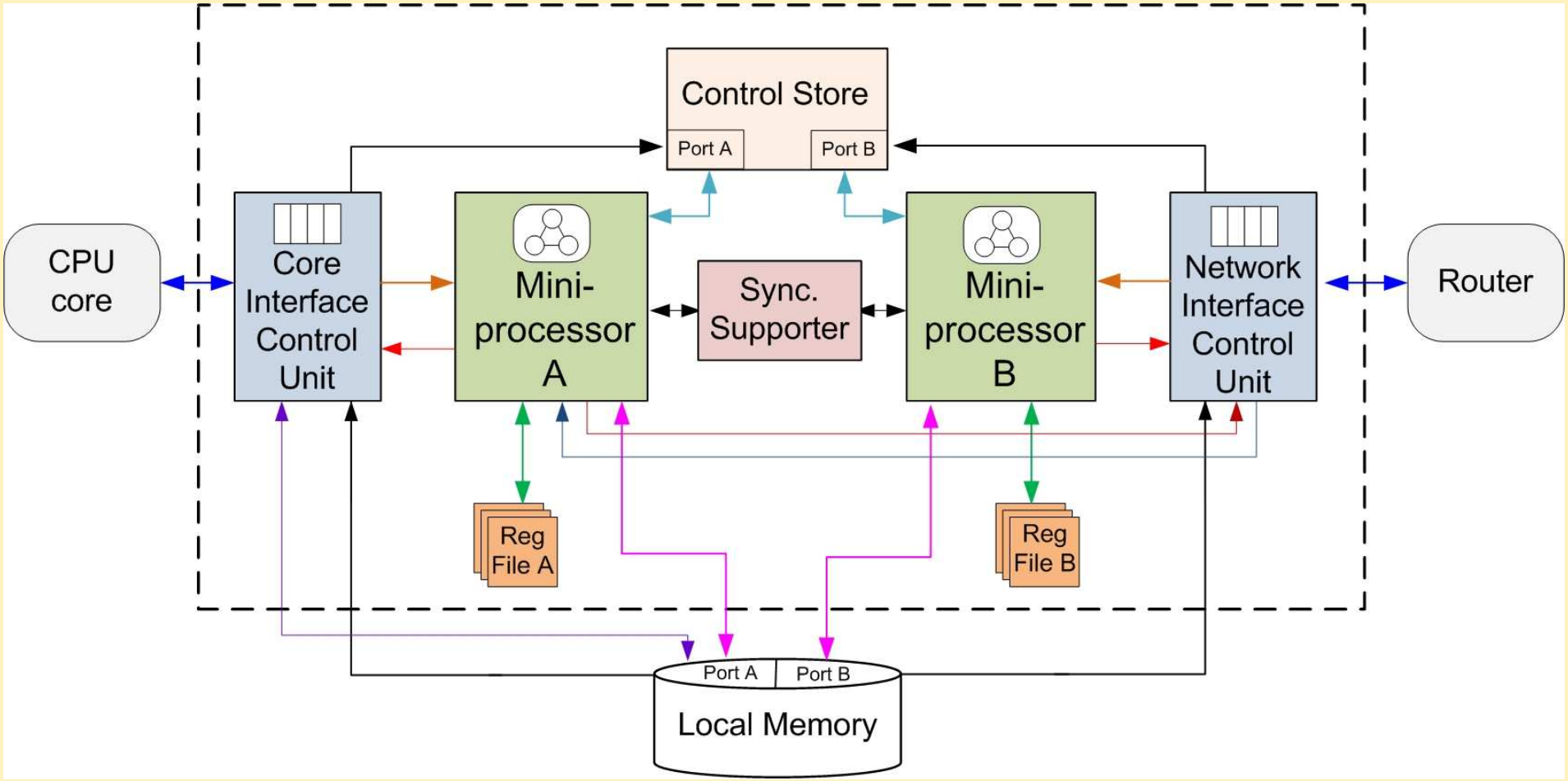
- 8x8 MPSoC
- Resource size: 2x2 mm2
- Switch size: 100x100 um2
- TSV pitch: 5 um
- TSVs/switch: 128@2GHz
- 256 Gb/s memory bandwidth per core
- 16 Tb/s aggregate memory bandwidth (200 Gb/s for Tilera)
- $<$ 10 ns delay

- 80 x bandwidth
- 1/5 latency
- 1/10 power
- No area overhead

Platform Overview



Data Management Engine Architecture

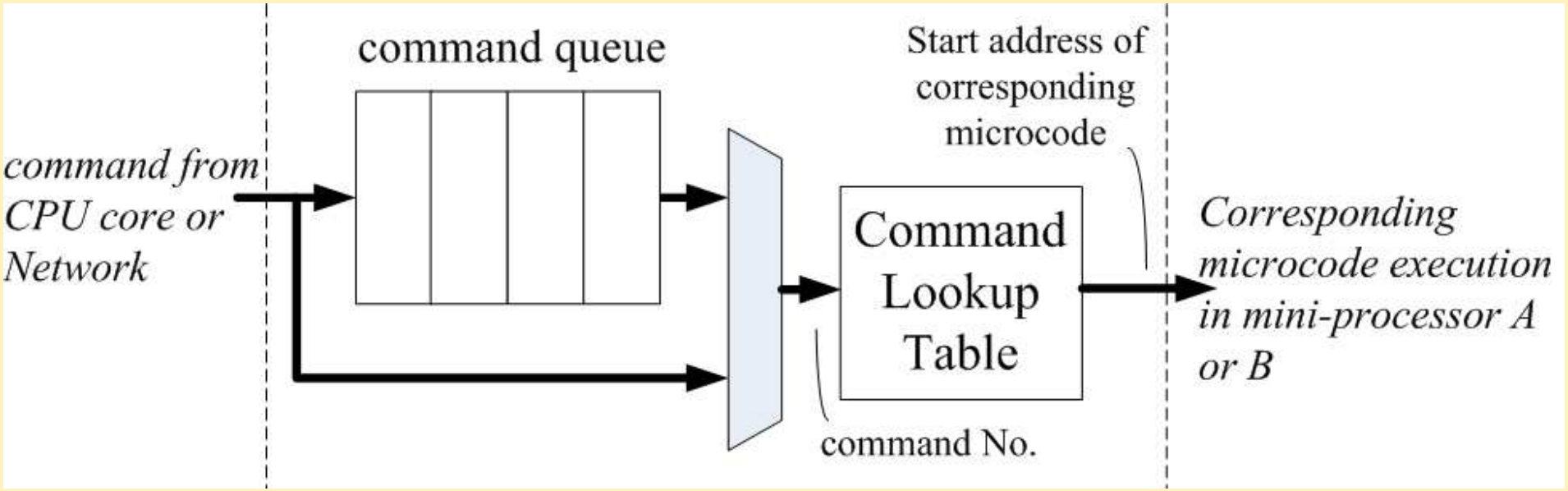


Data Management Engine Implementation

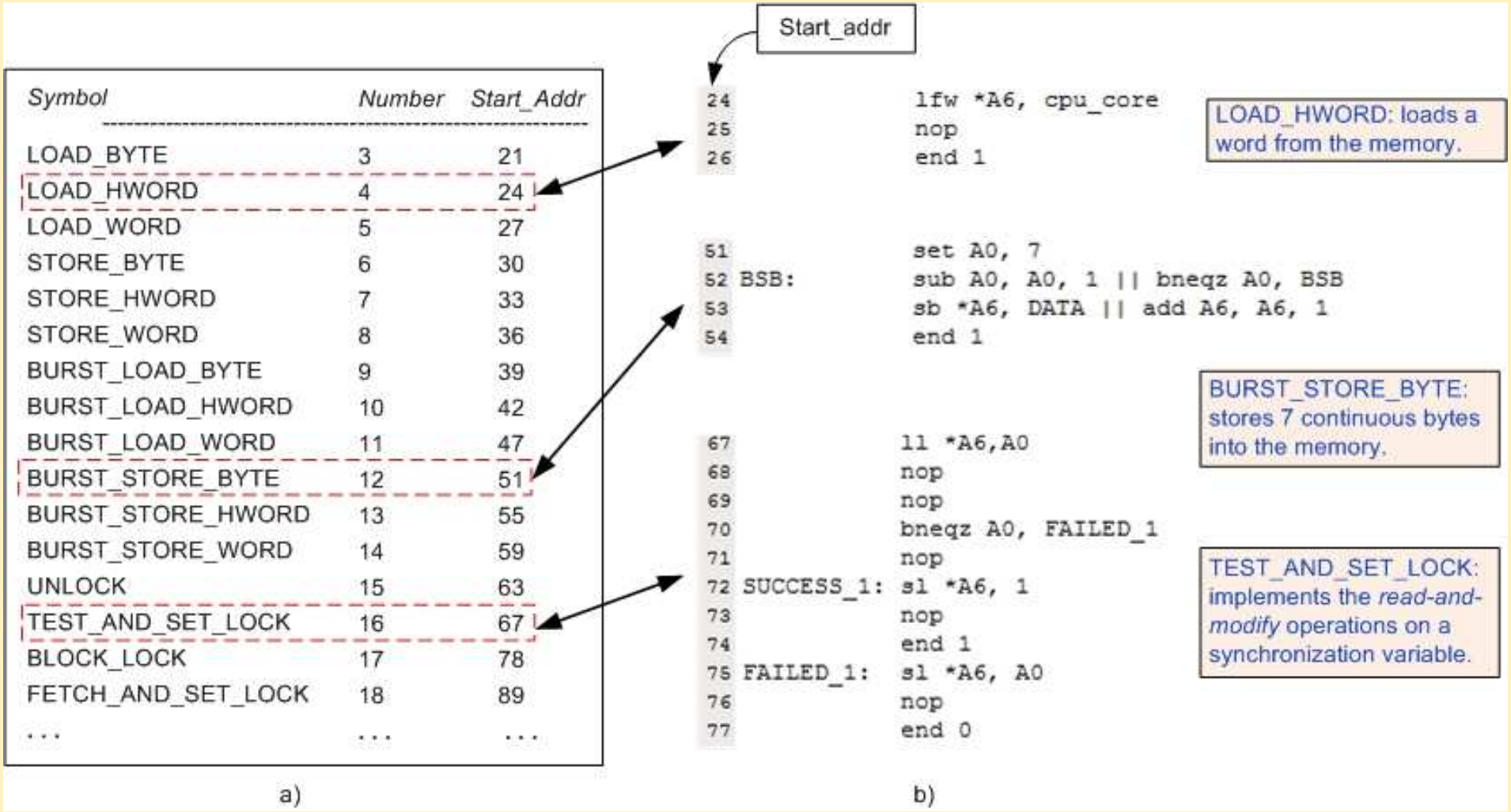
	Optimized for area	Optimized for speed
Frequency	444 MHz (2.25 ns)	455 MHz (2.2 ns)
Area (Logic)	44k NAND gates	51k NAND gates
Area (Control Store)	300k NAND gates	

	power consumption [mW]
Mini A	6.9
Mini B	7.0
NICU	2.3
CICU	5.2
Synchronizer	0.2
DME total	21.6

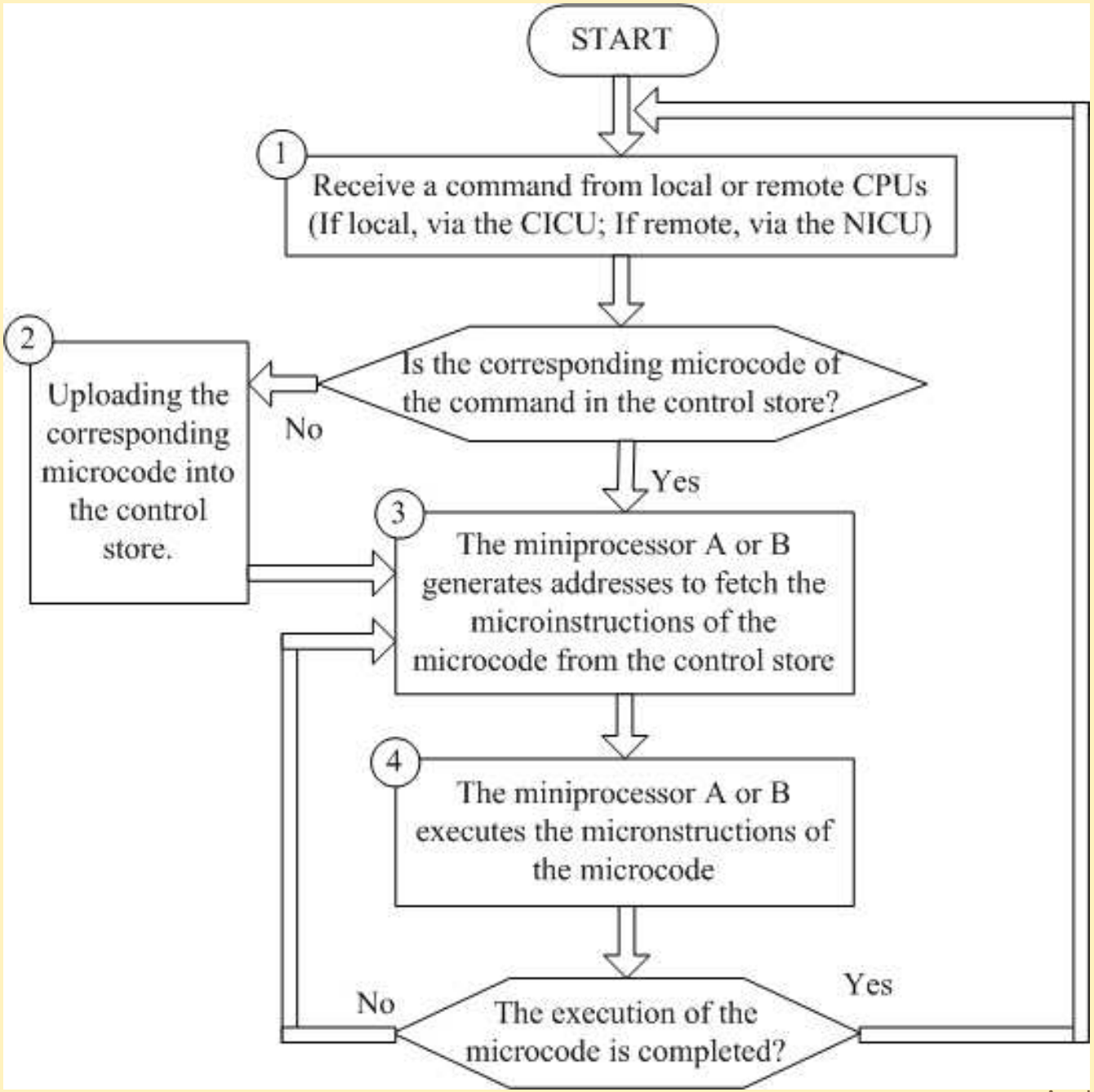
Command Triggered Microcode Execution



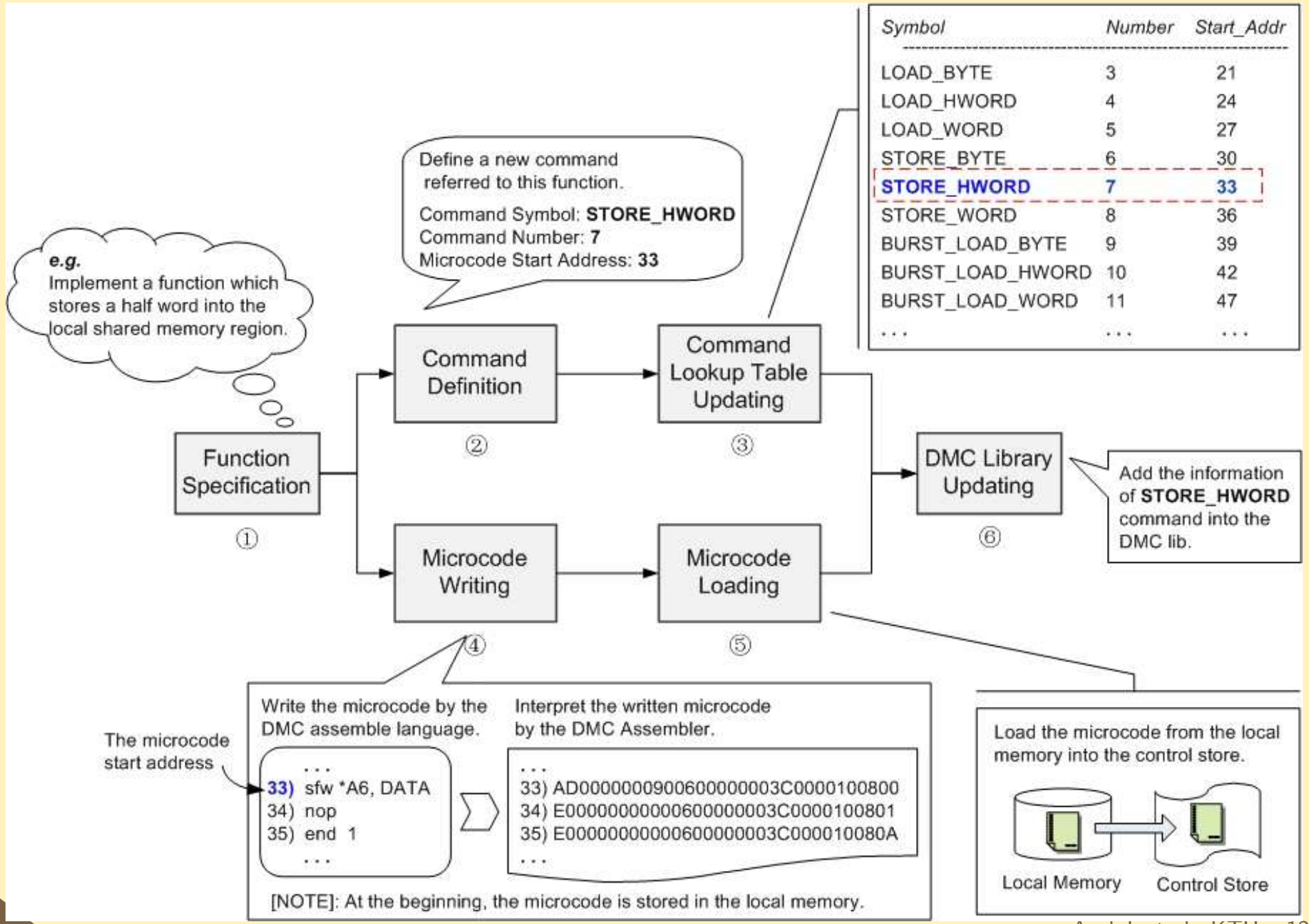
Command Lookup Table (CLT)



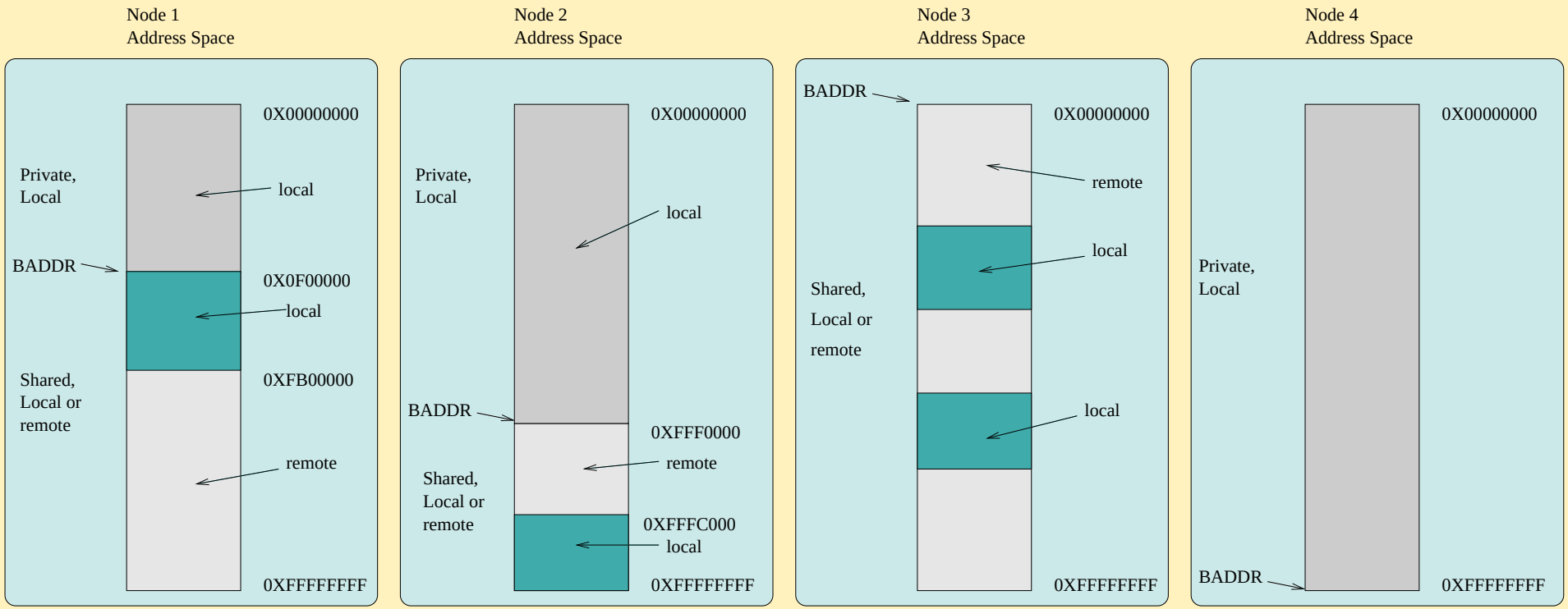
DME Execution Flow



Microprogram Development Flow



Address Space Management

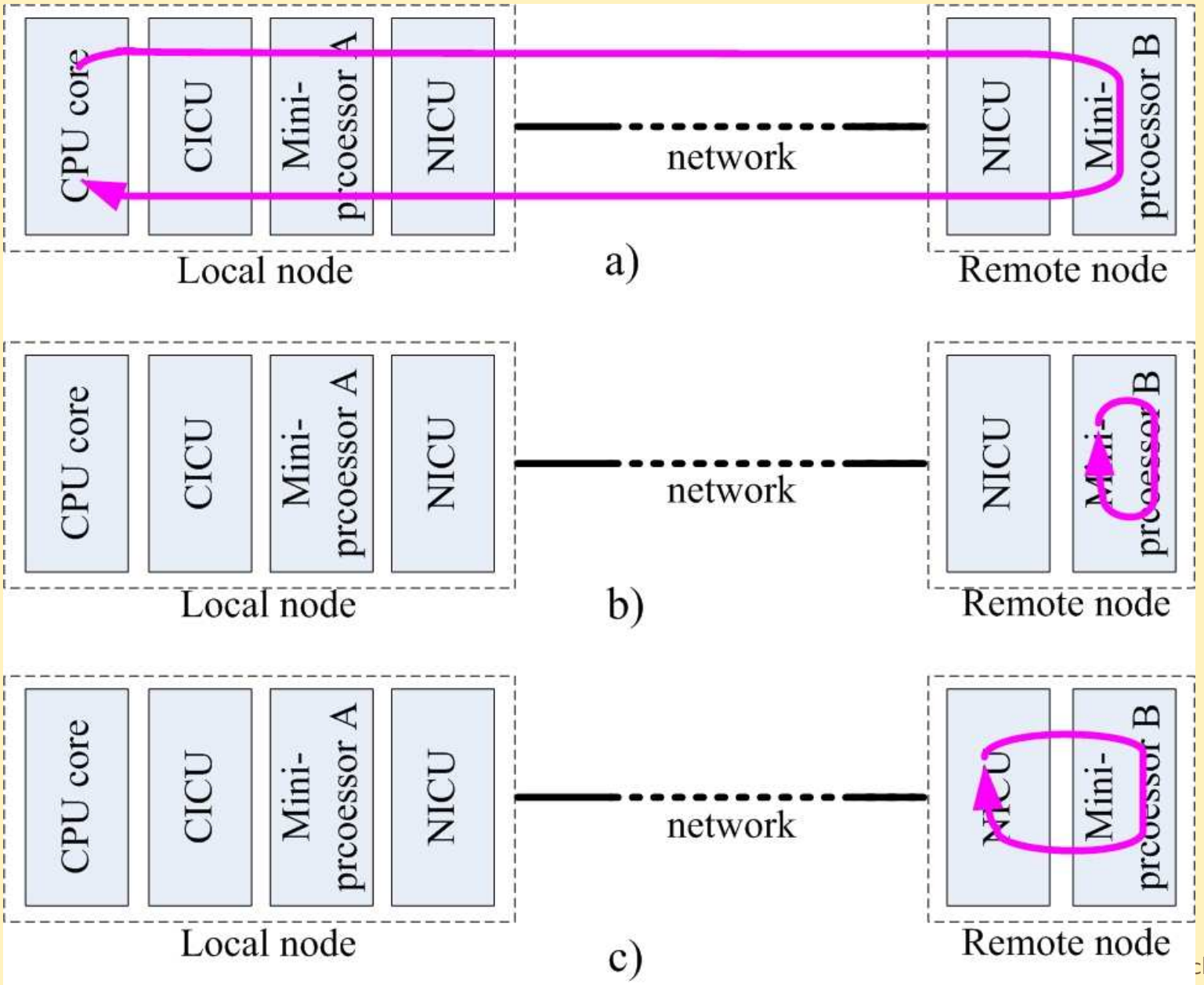


Supported Memory Partitions

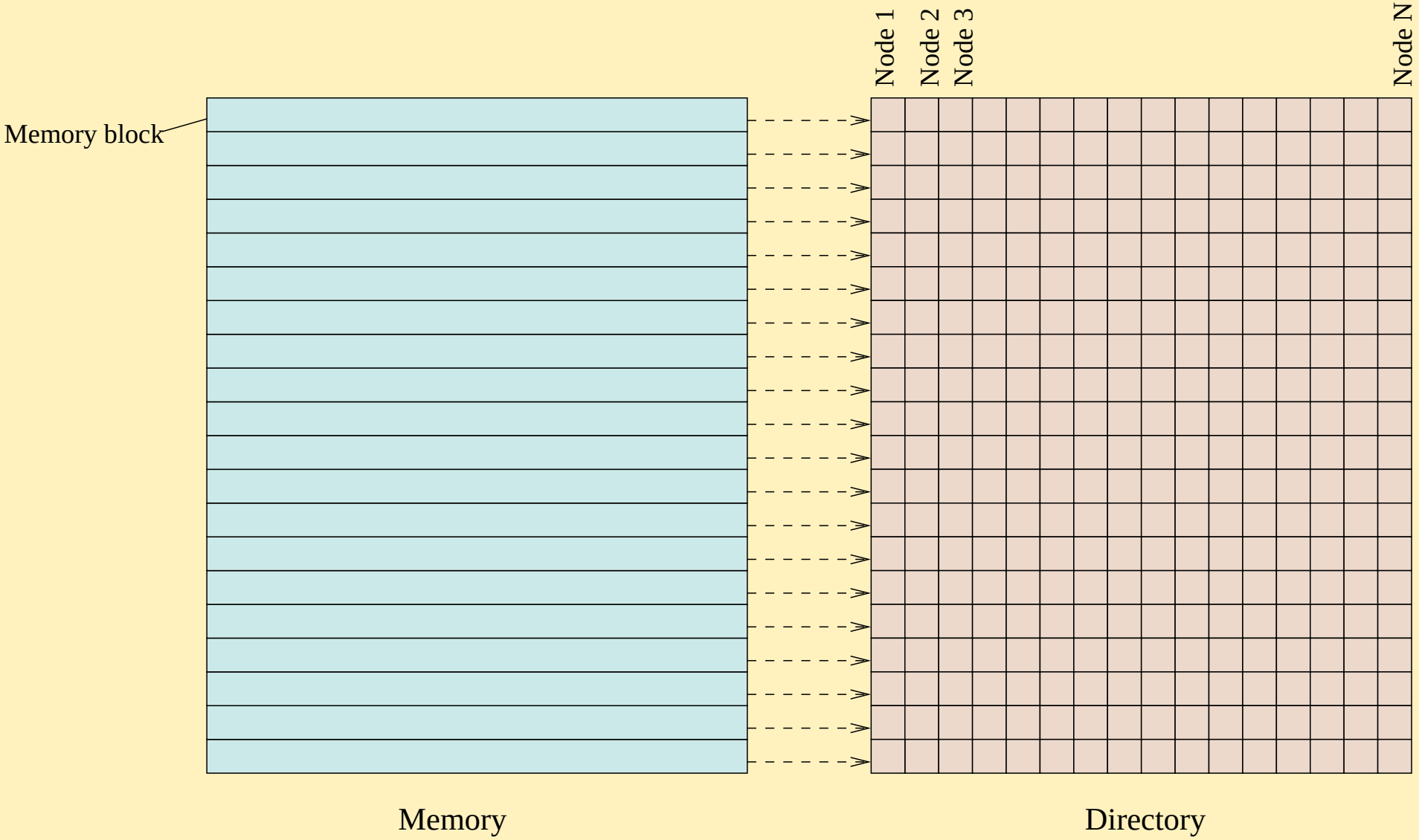
local	private	physical	Supported
local	private	virtual	-
local	shared	physical	-
local	shared	virtual	Supported
remote	private	physical	-
remote	private	virtual	-
remote	shared	physical	-
remote	shared	virtual	Supported

Synchronization Support

Test&Set() and Test&SetBlocking()

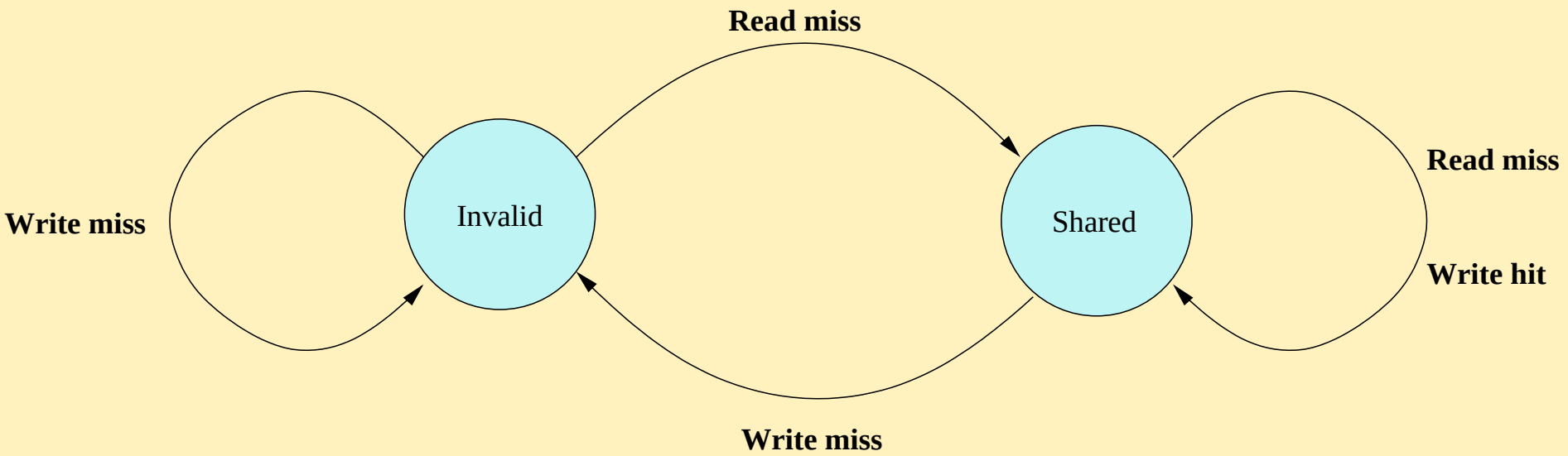


Directory Based Cache Coherence

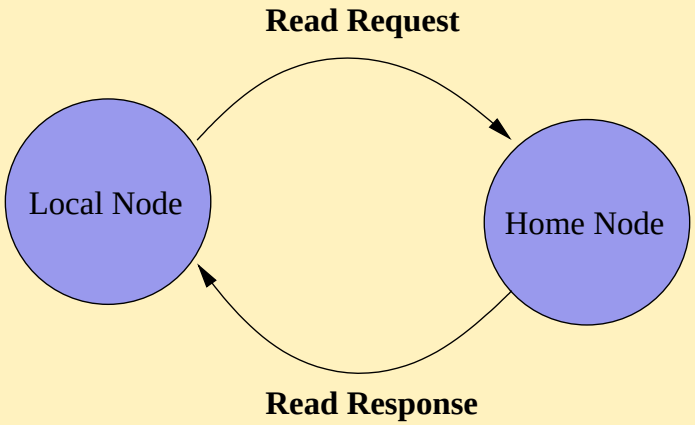


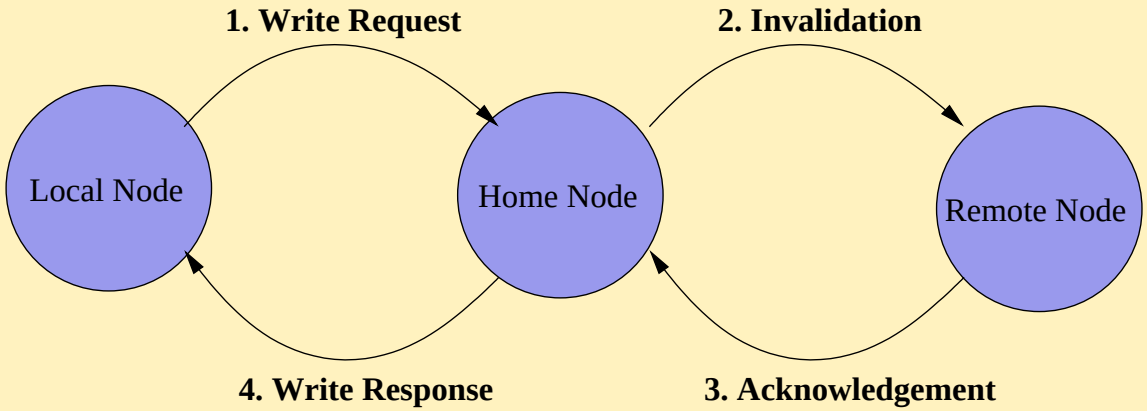
SI Cache Coherence Protocol

Write-trhough No-allocate Cache Policy

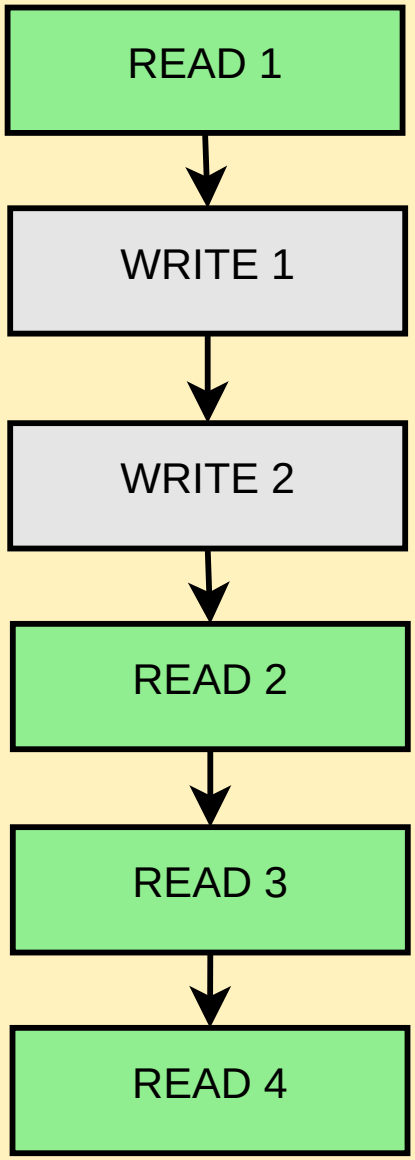


Coherent Read and Write

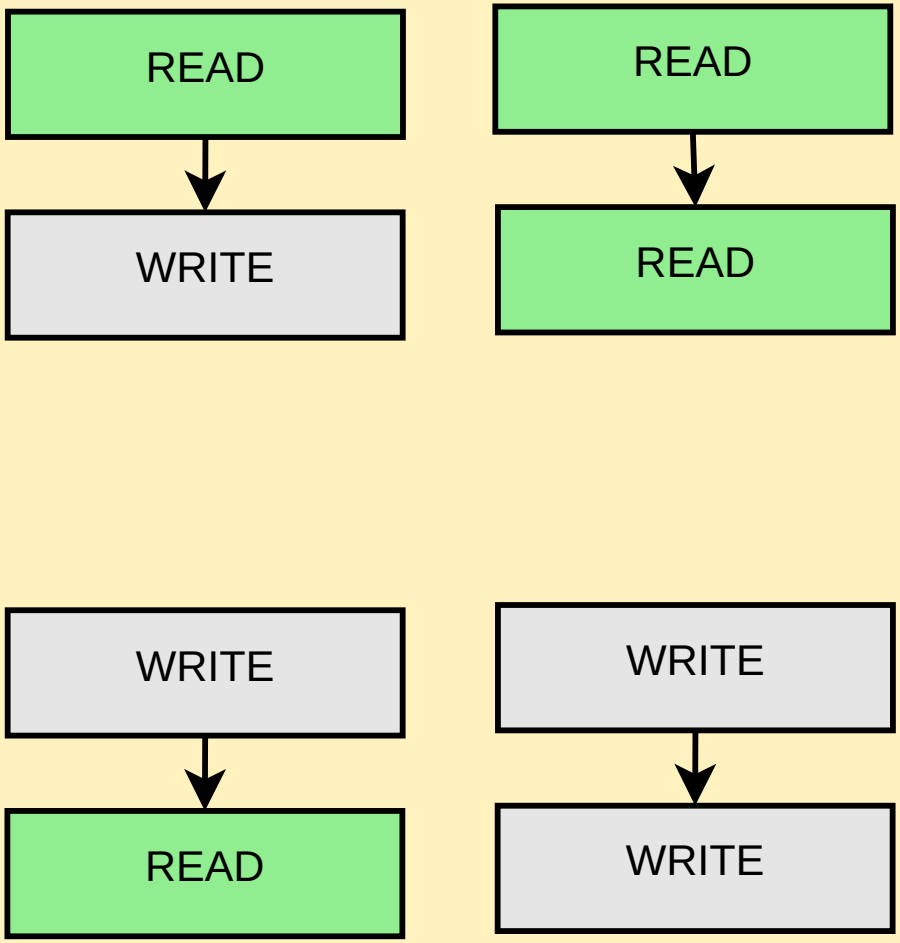




Memory Consistency and Transaction Ordering - Sequential Consistency

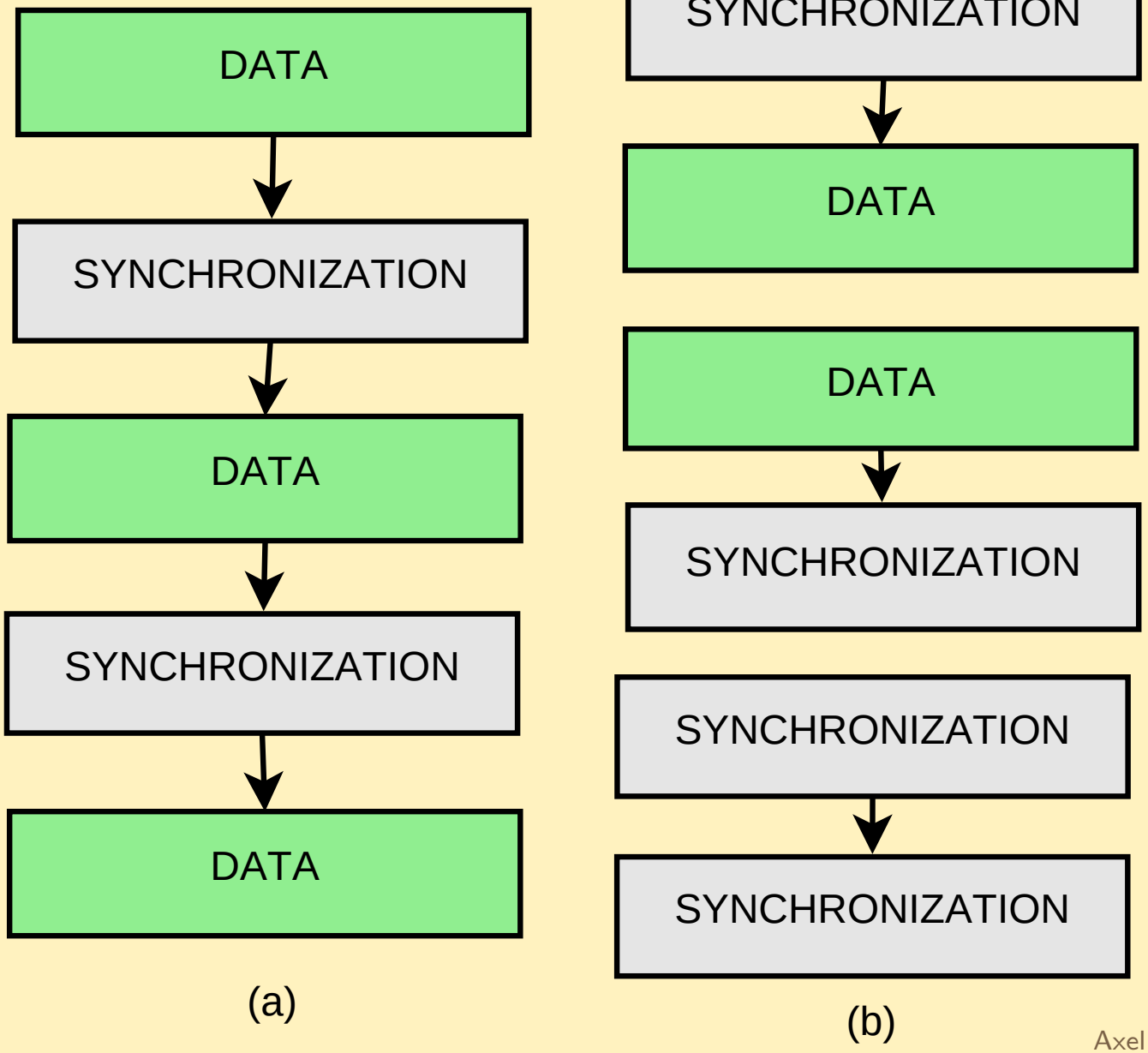


(a)

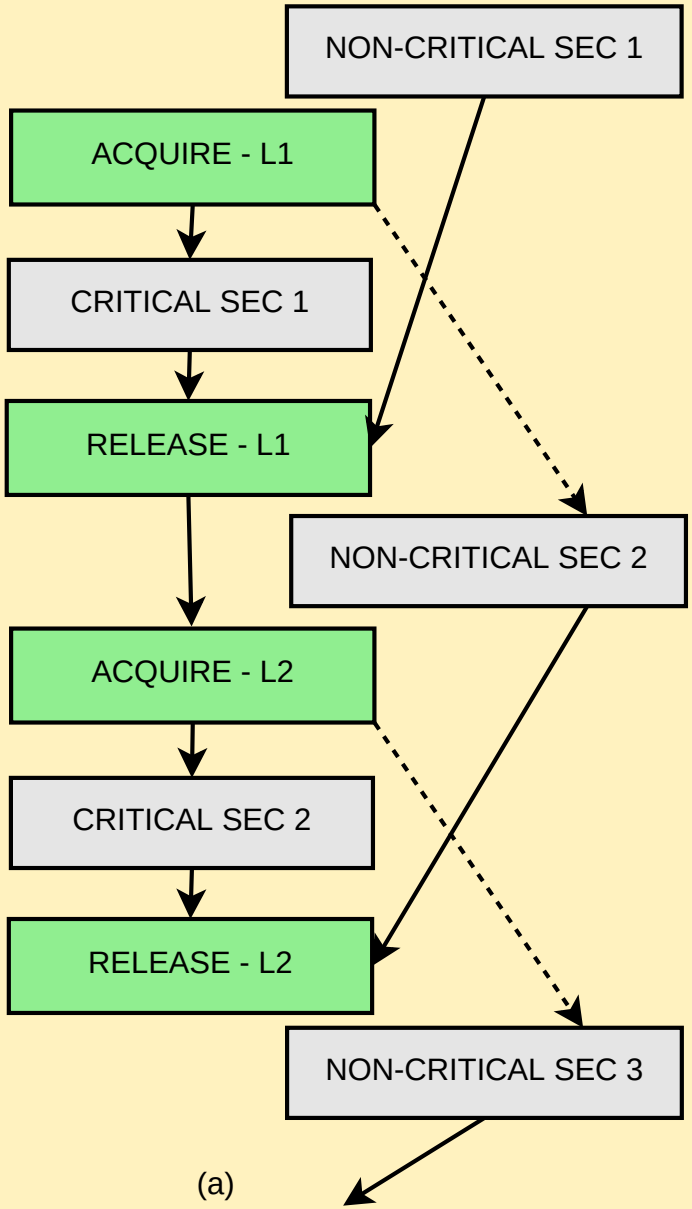


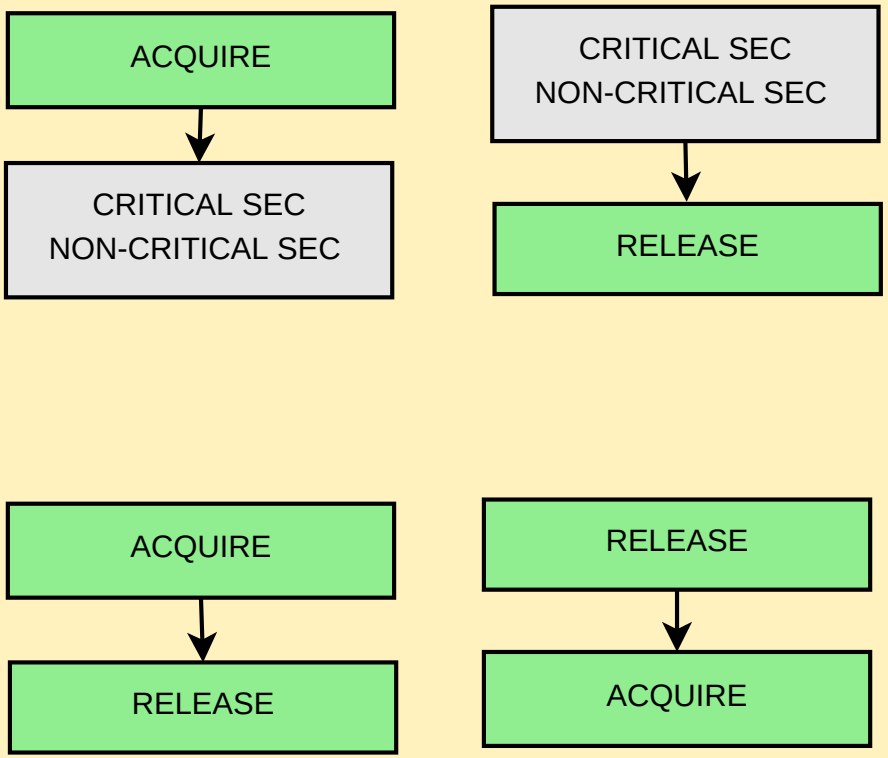
(b)

Weak Memory Consistency

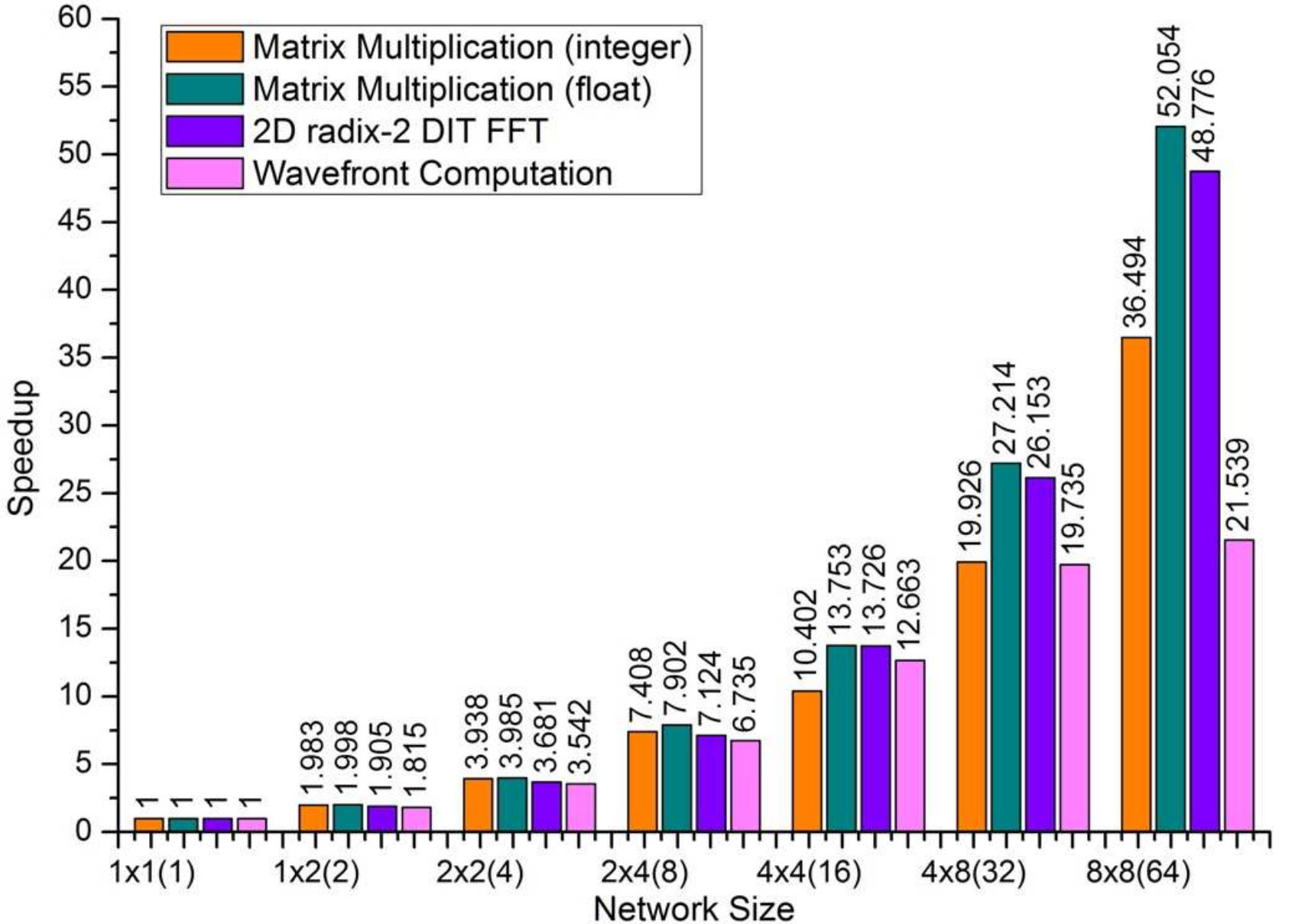


Release Consistency

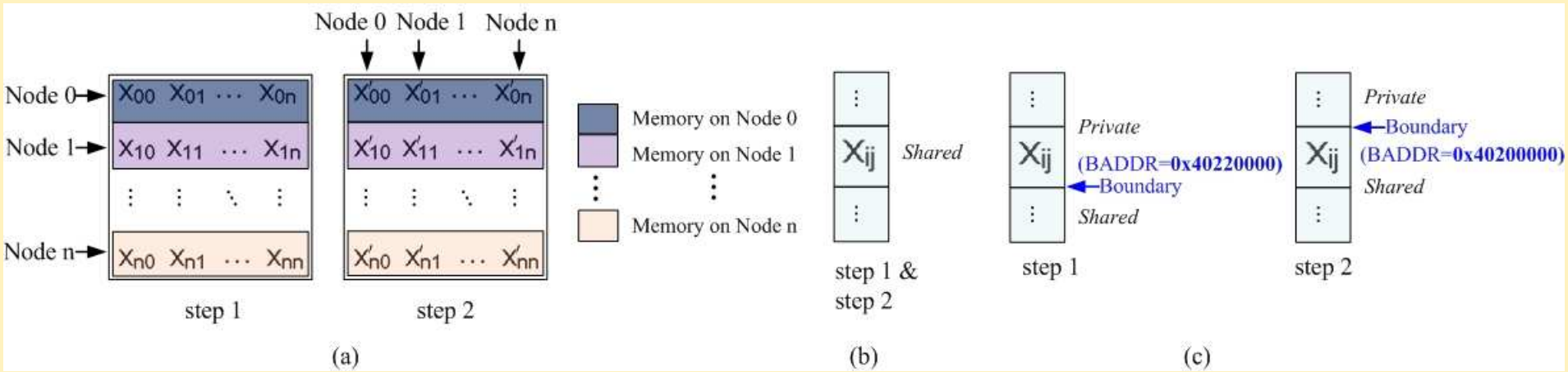




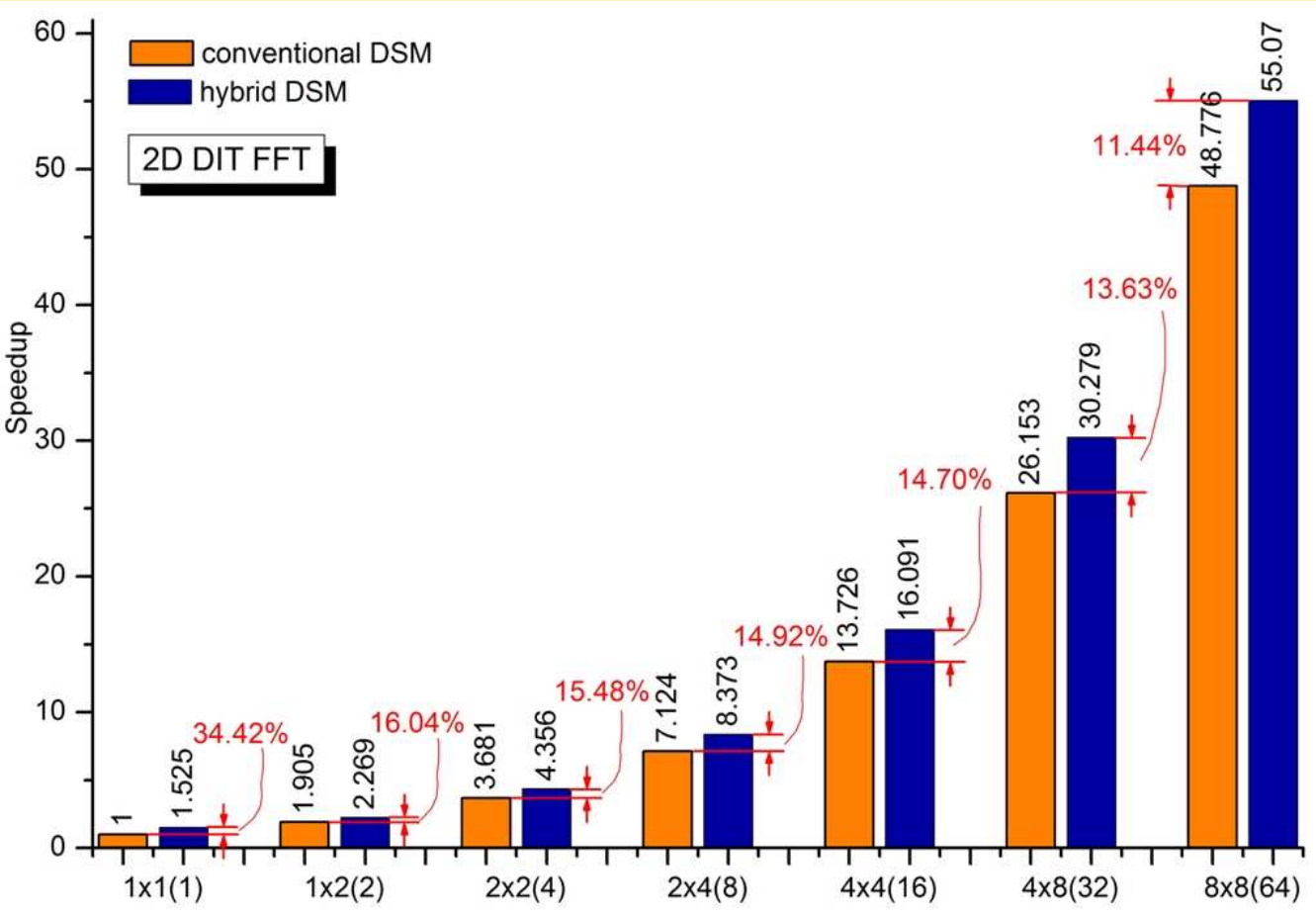
Multi-core Speedup



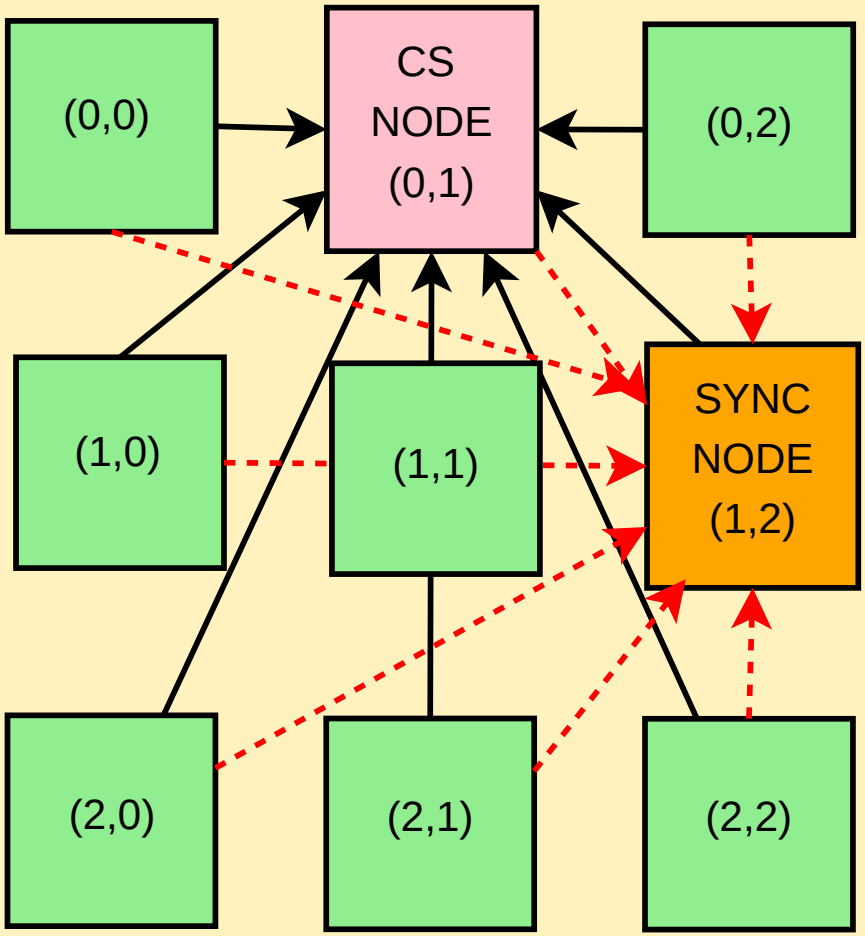
Run-Time Memory Space Re-Partitioning



Hybrid DSM Speedup



Memory Consistency Experiment

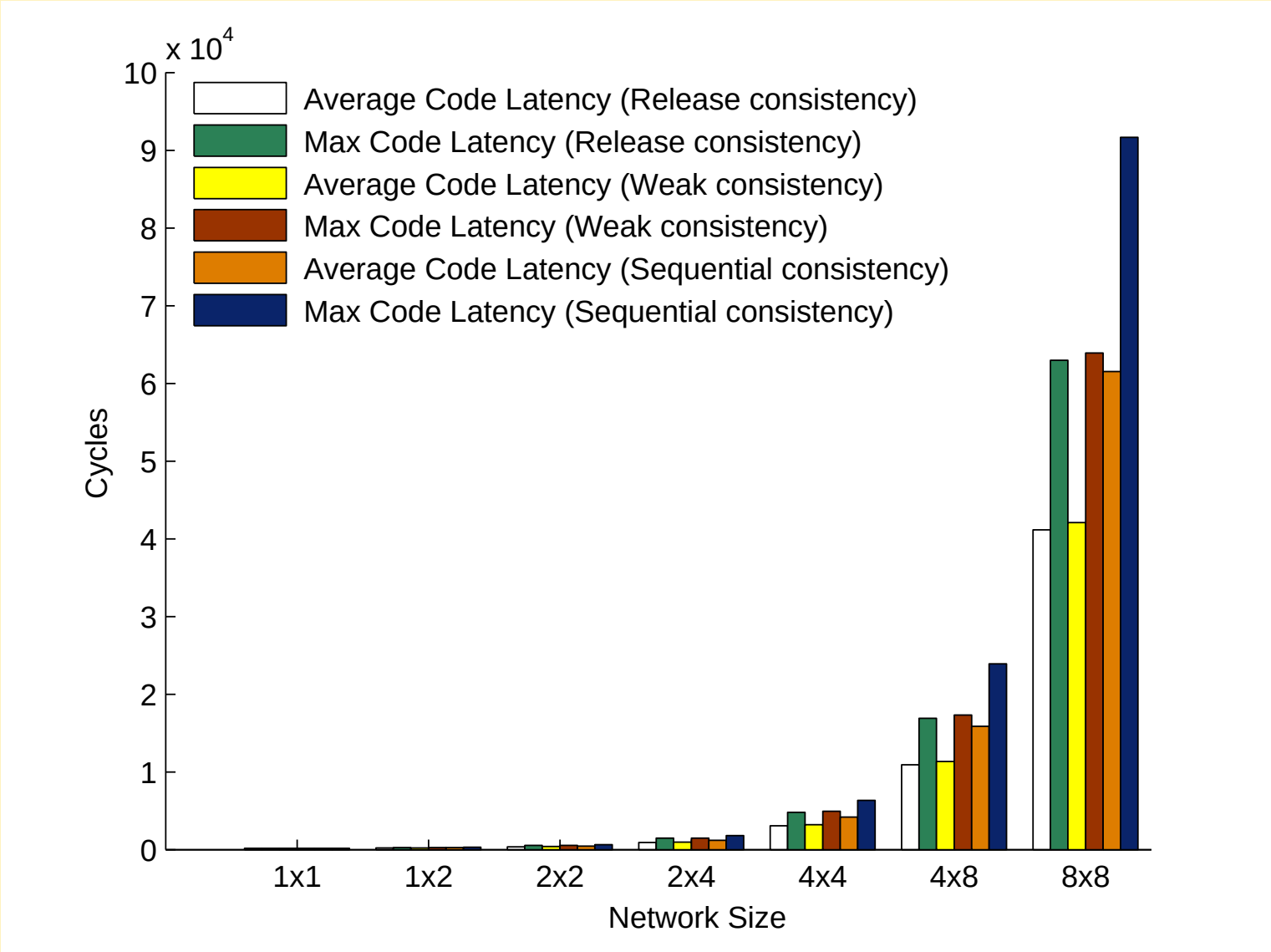


(a)

```
Initially, int x, y,z = 0;

// Non-critical section1
x = data1;
Reg1 = x;

// Lock Acquire
Acquire (L);

// Critical Section
y = data2;
Reg2 = y;

// Lock Release
Release(L)

// Non-critical Section2
z = data3;
Reg3 = z;
```

(b)

Memory Consistency Execution Time



Summary

- After computation (multi-core) and communication (NoC), memory access must be parallalized
- DME parallizes memory handling
- DME supports
  - Central/distributed memory
  - Private/shared
  - Physical/virtual address space
- DME features
  - Synchronization support
  - Cache coherence protocols
  - Memory consistency support
  - Dynamic memory allocator