Memory Architecture and Management in a NoC Platform

Axel Jantsch
Xiaowen Chen
Zhonghai Lu
Chaochao Feng
Abdul Nameed
Yuang Zhang
Ahmed Hemani

DATE 2011

Overview

- Motivation
- State of the Art
- Data Management Engine
  - Architecture
  - Virtual Address Space
  - *Cache Coherence*
  - *Memory Consistency*
  - *Synchronization*
  - *Message Passing based Communication*
- 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:

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

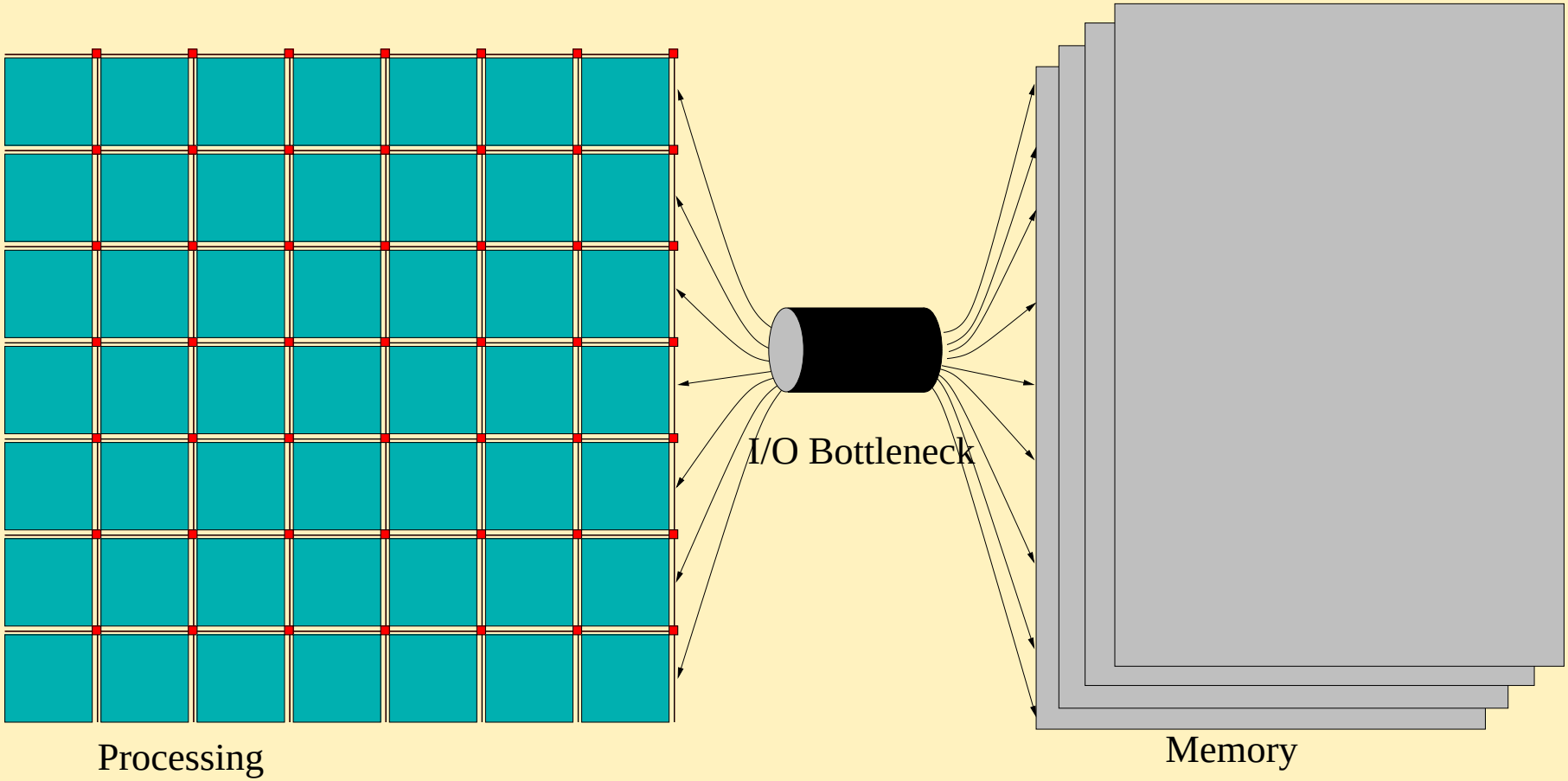
t_{avg}...average access latency for one data word
p ...probability of a cache hit
t_c ...access time to the cache
t_m ...access time to main memory

Today we navigate at the edge of this wall.

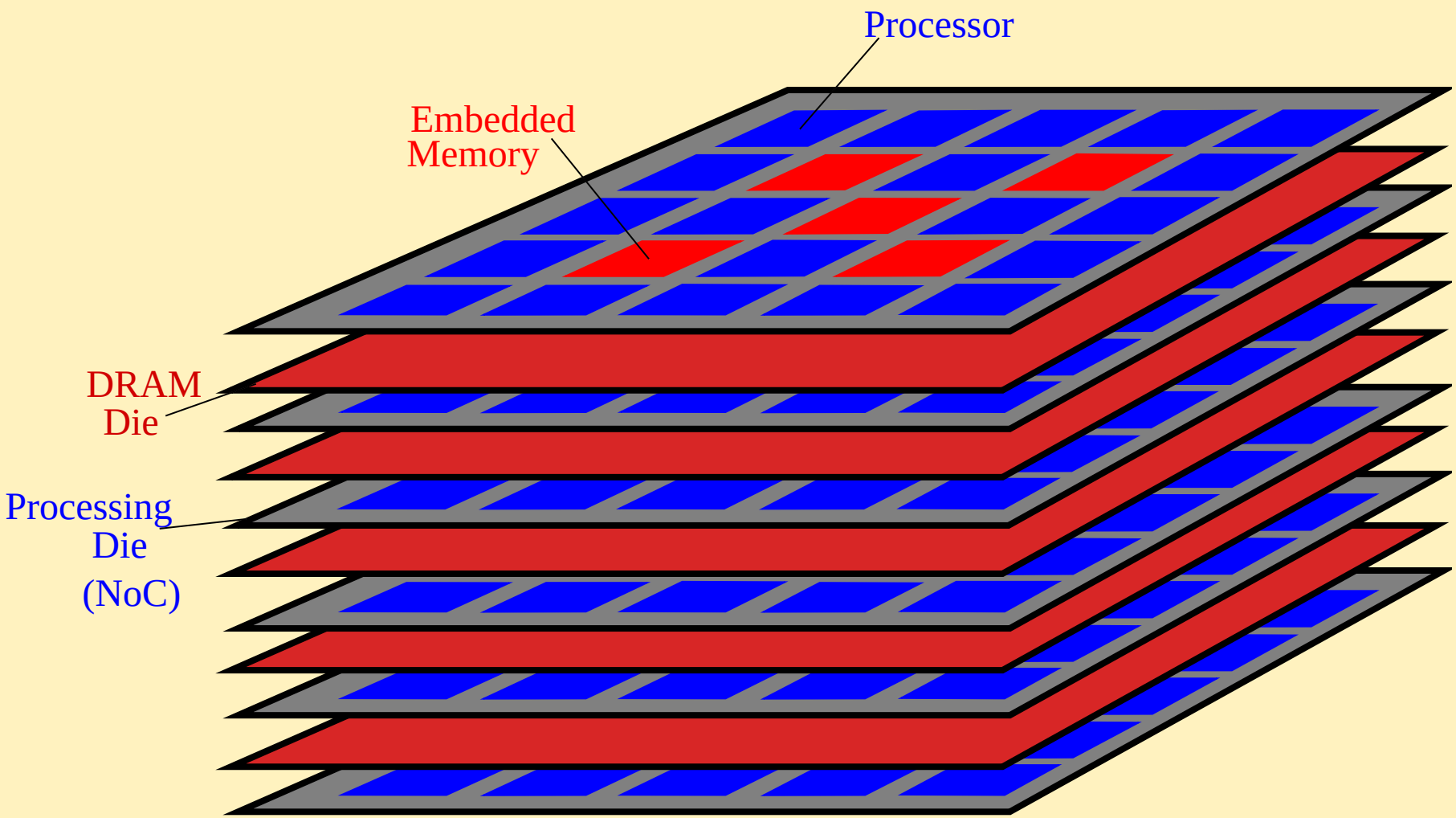
For example the Tilera 64 core chip:

- 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

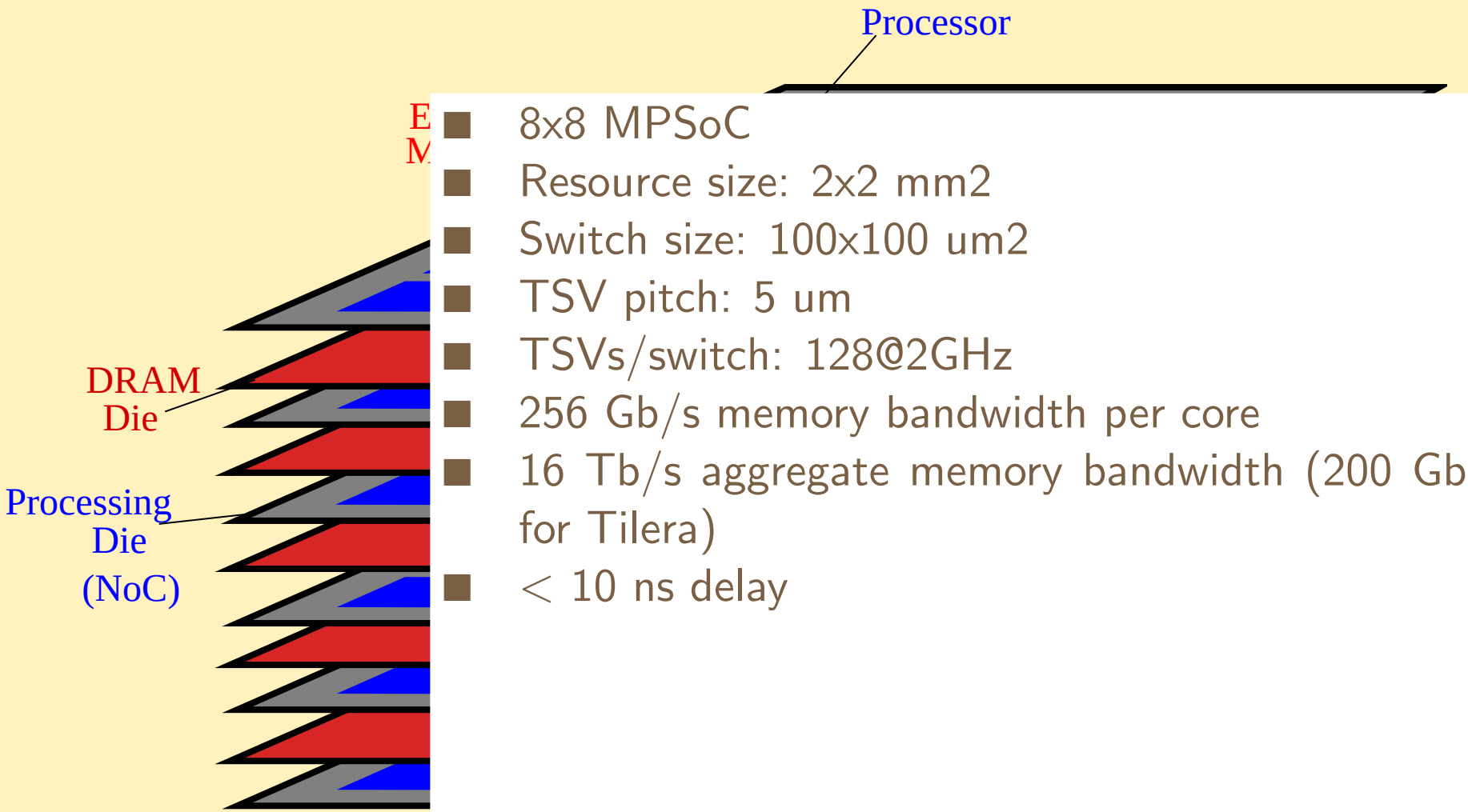
The Memory Access Bottleneck



Memory Bandwidth in 3D ICs

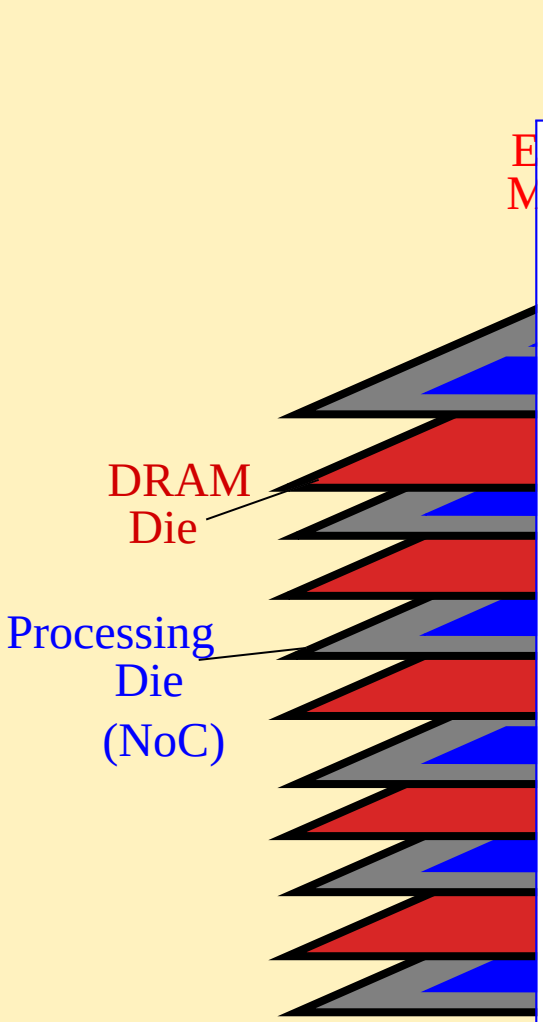


Memory Bandwidth in 3D ICs



- 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

Memory Access Protocols

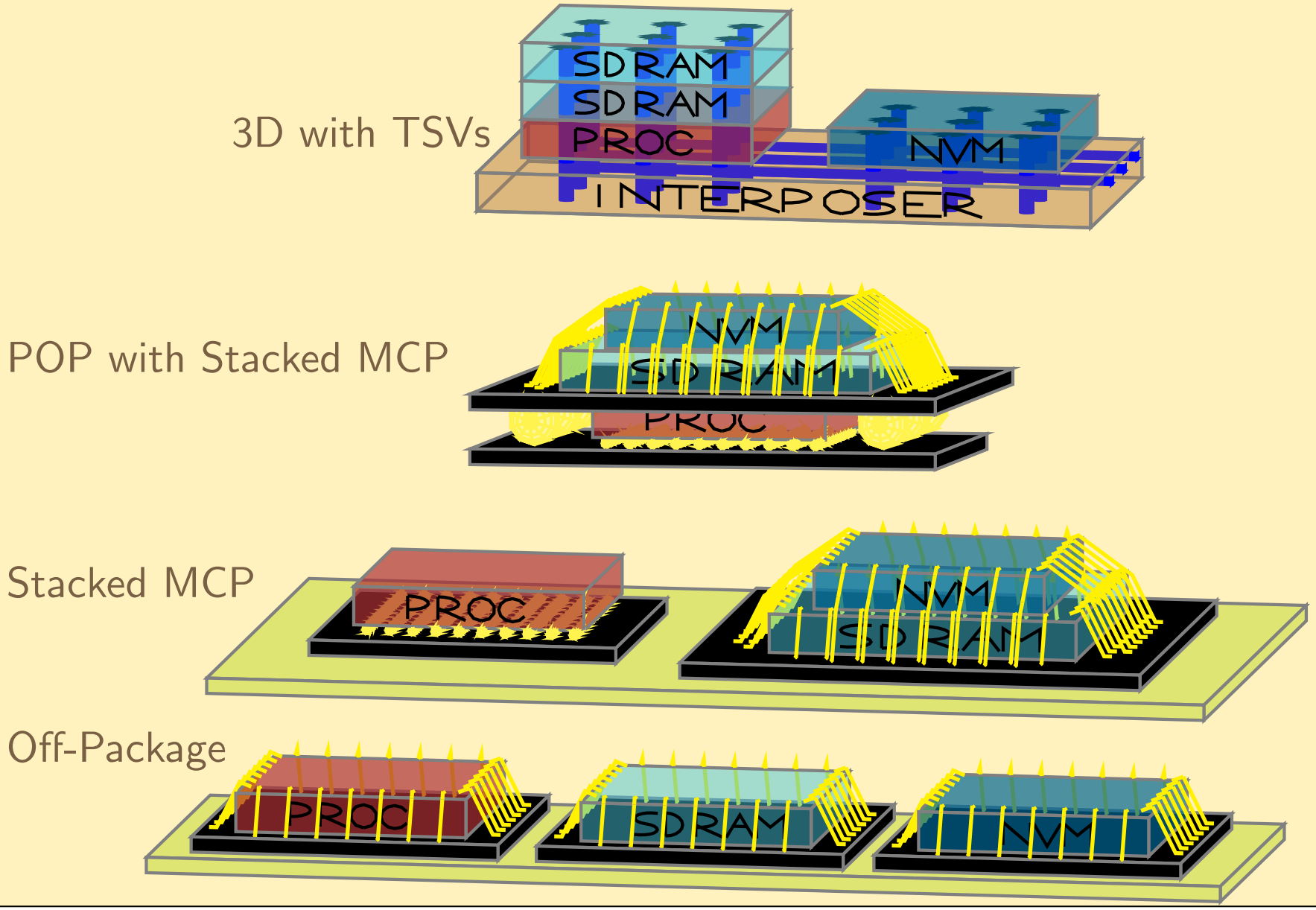
Protocol Characteristics

	Frequency	buswidth	Voltage	max bandiwtdh	Efficiency
DDR3	1066 MHz	32 bit	1.5 V	8.532 GB/s	4.17 GB/s/W
LPDDR2	533 MHz	32 bit	1.2 V	4.264 GB/s	6.25 GB/s/W
Wide I/O	200 MHz	512 bit	1.2 V	12.8 GB/s	25.0 GB/s/W

Cost for 1 TB/s

Protocol	No of ports	No of pads	Power
DDR3	120	3800	240 W
LPDDR2	240	7700	160 W
Wide I/O	80	41000	40 W

Package Architecture Roadmap



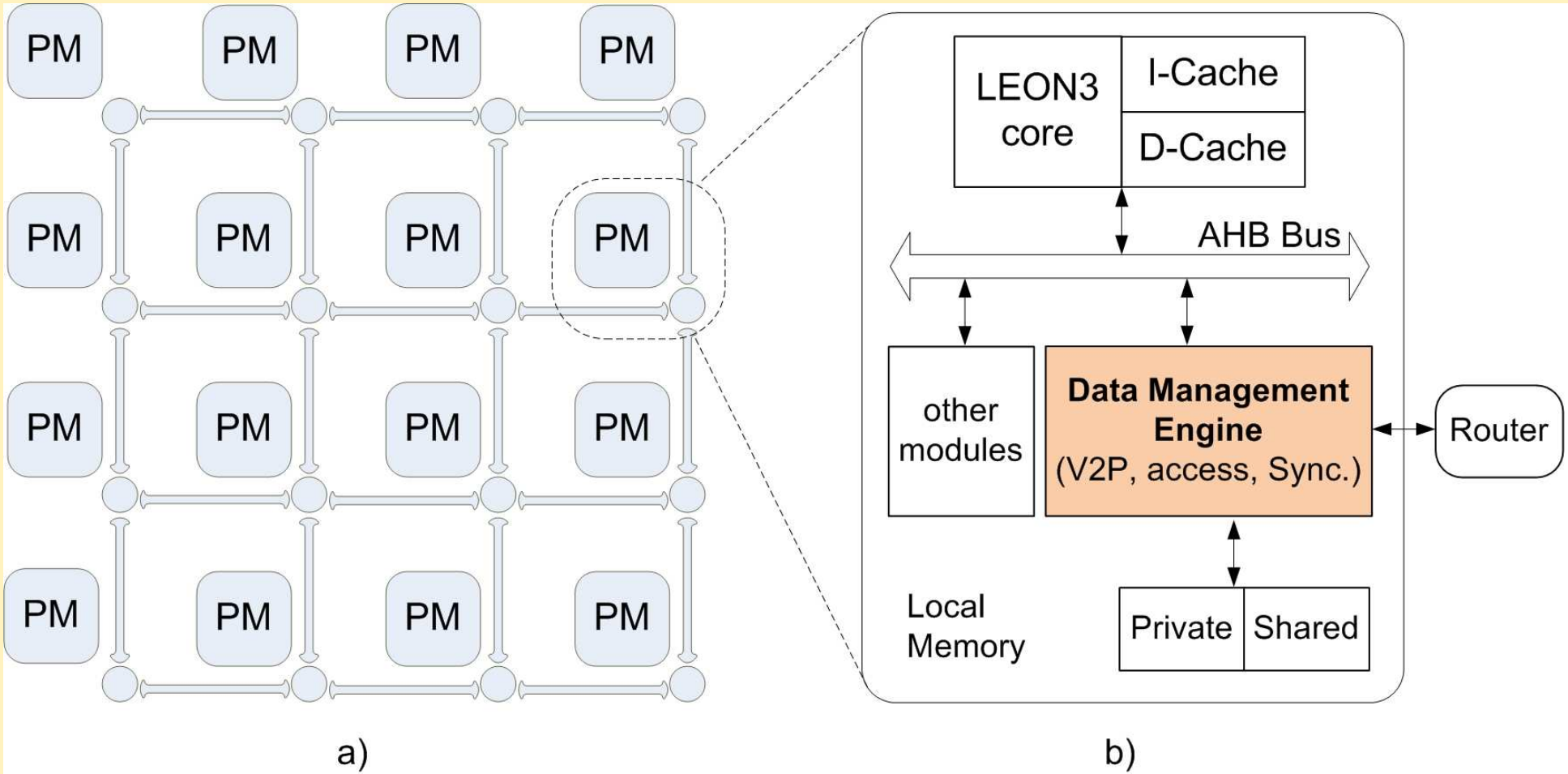
From: Denis Dutois and Ahmed Jerraya. "3D Integration Opportunities for Memory Interconnect in Mobile Computing Architectures". In: *Future Fab International* 34 (July 2010)

Memory Architecture
State of the Art

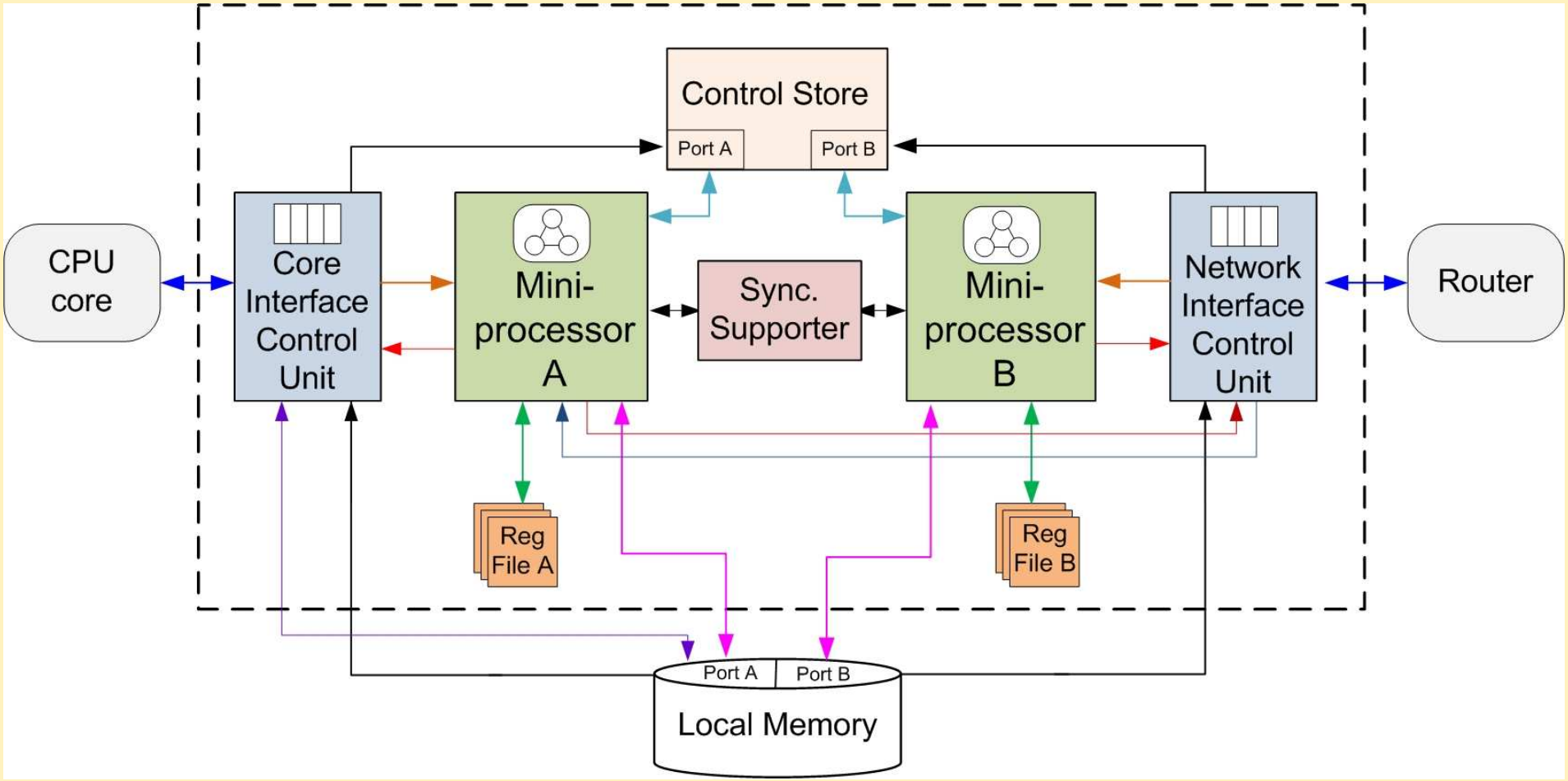
- Custom memory architectures in many SoCs and embedded systems with no shared memory space
- No HW support for shared memory space, cache coherence and consistency (e.g. Inte's 48 core SCC)
- Uniform shared memory space with snooping based cache coherence for small multi-core systems (ARM's Cortex A9)
- Non-Uniform Cache Architecture (NUCA) in regular many-core SoCs; introduced in 2002:
  C. Kim, D. Burger, and S. Keckler. "An Adaptive, Non-uniform Cache Structure for Wire-Delay Dominated On-Chip Caches". In: *Proceedings of the 20th International Conference on Architectural Support for Programming Languages and Operating Systems*. 2002

- No general, flexible, and scalable solution for the many-core era

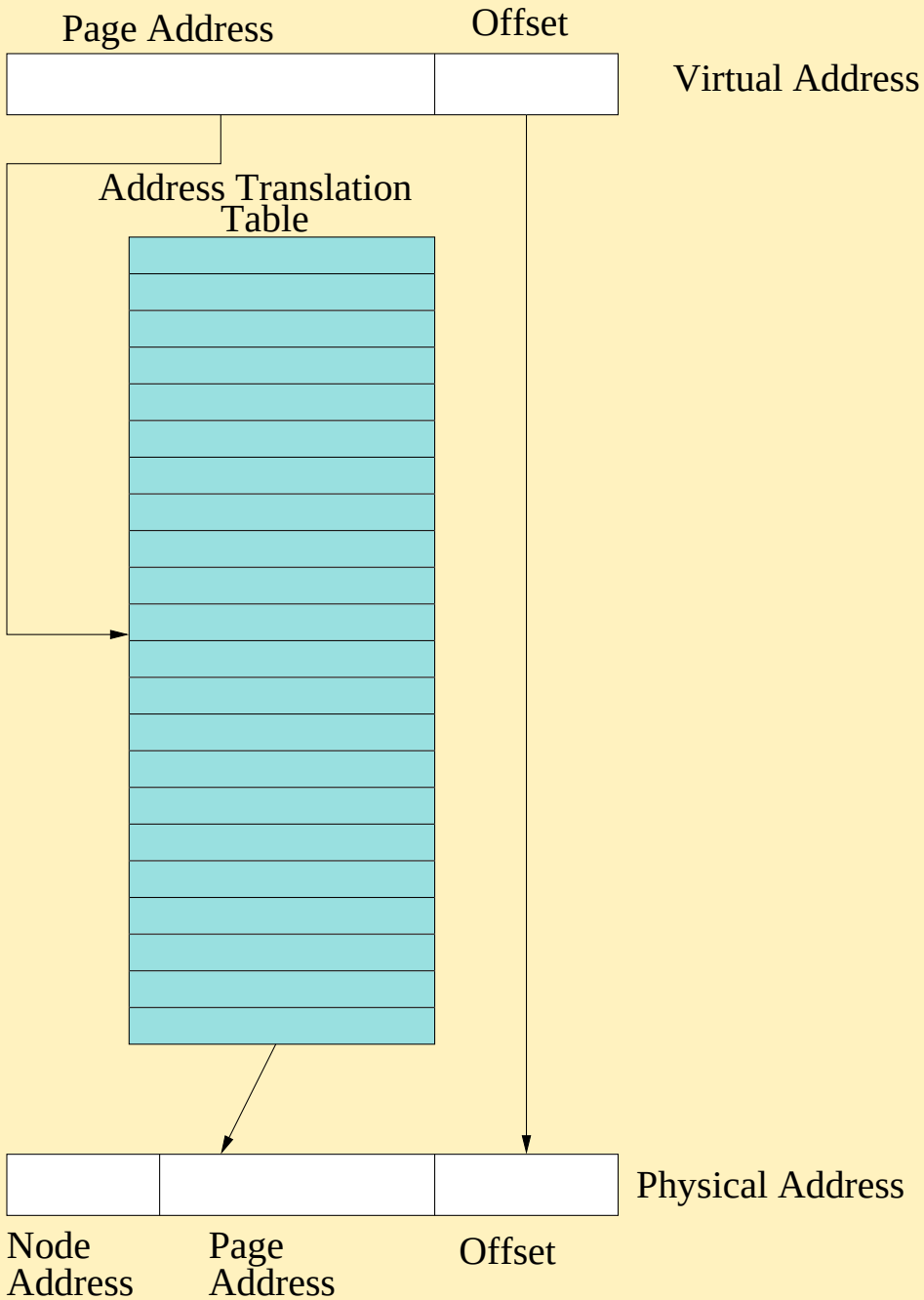
Platform Overview



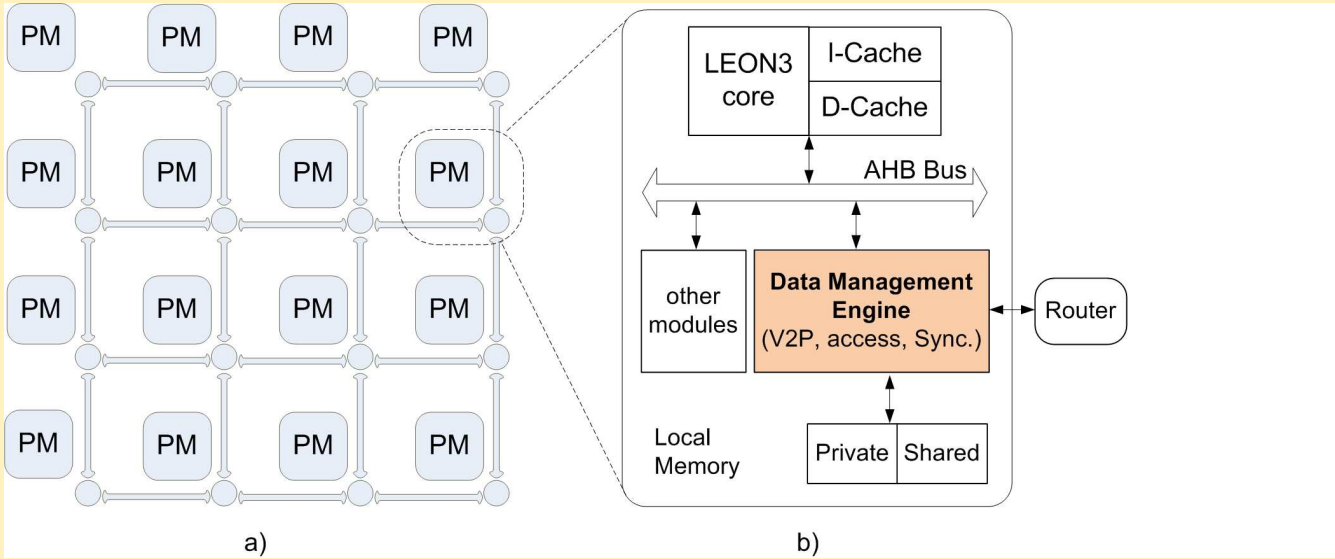
Data Management Engine Architecture



Virtual Address Space Translation



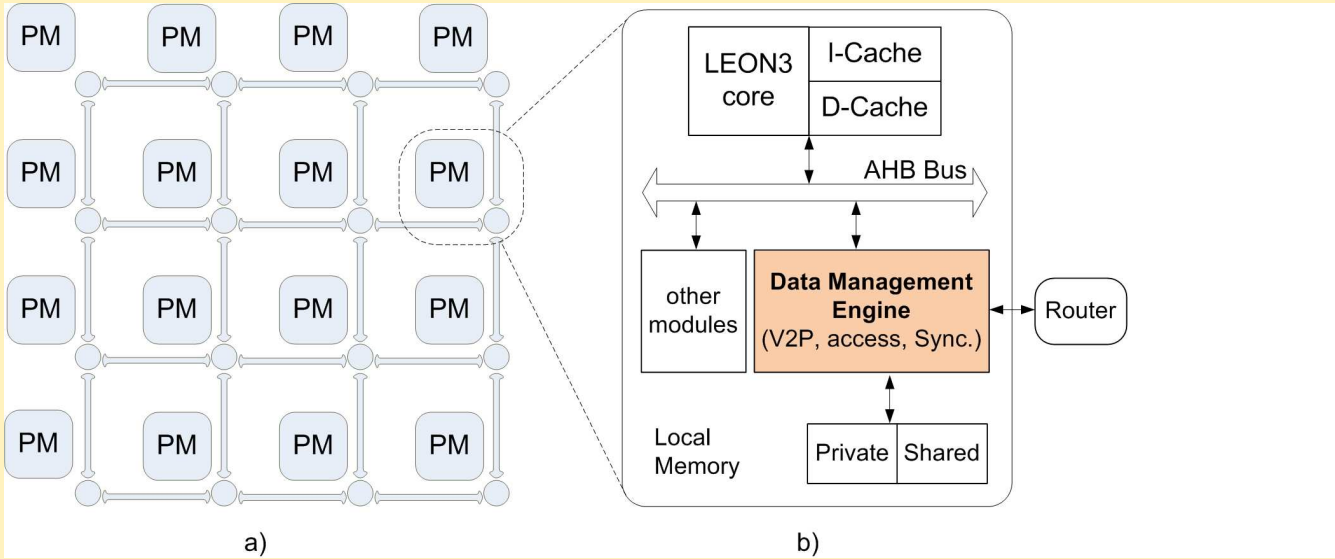
Virtual Address Space Translation



$$S_T = (\log_2 N + \log_2 M_N - \log_2 P_S) \cdot (M_T / P_S)$$

S_T ... Size of translation table per node
N ... Number of nodes
M_N ... Memory per node
P_S ... Page size
M_T ... Total memory size

Virtual Address Space Translation

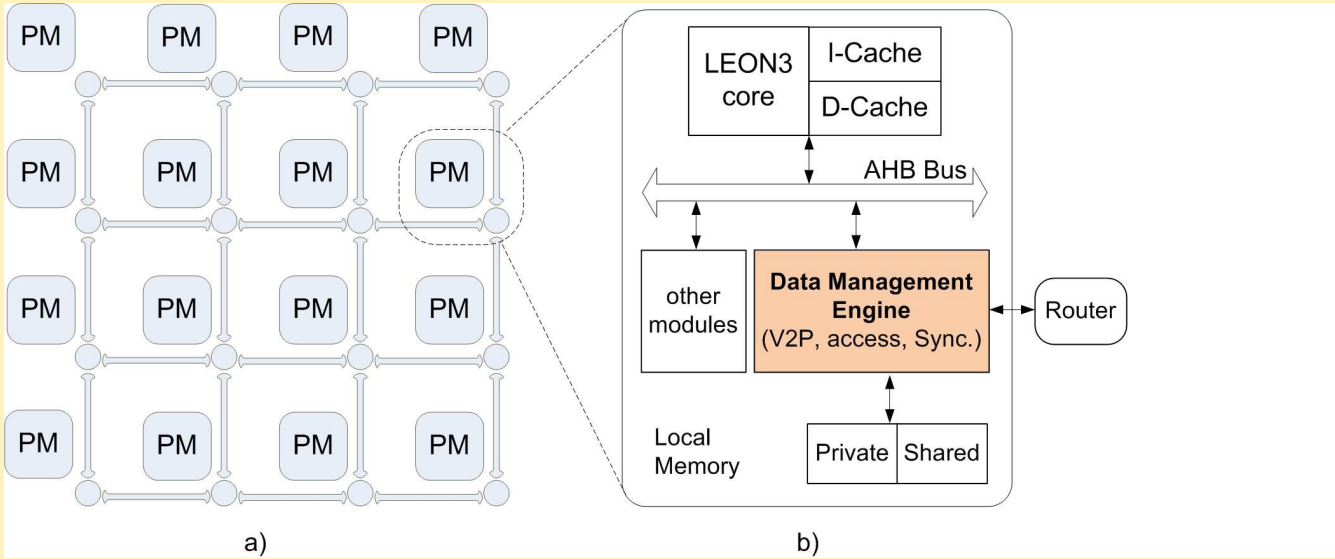


$$S_T = (\log_2 N + \log_2 M_N - \log_2 P_S) \cdot (M_T / P_S)$$

- S_T ... Size of translation table per node
- N ... Number of nodes
- M_N ... Memory per node
- P_S ... Page size
- M_T ... Total memory size

- 64 nodes - 2^6

Virtual Address Space Translation

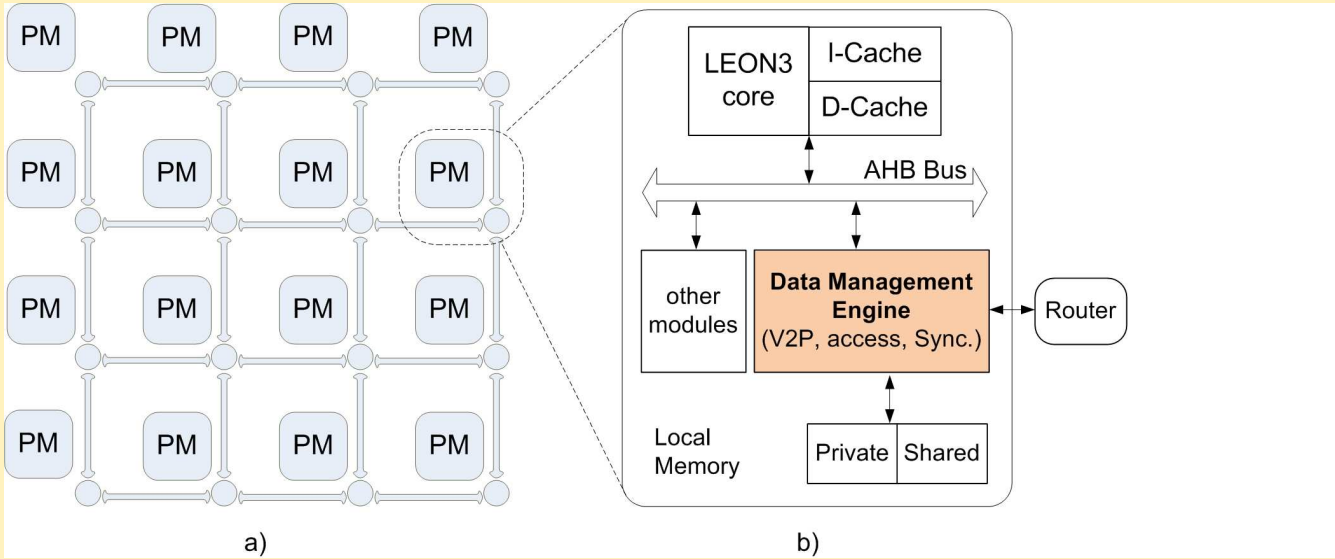


$$S_T = (\log_2 N + \log_2 M_N - \log_2 P_S) \cdot (M_T / P_S)$$

- S_T ... Size of translation table per node
- N ... Number of nodes
- M_N ... Memory per node
- P_S ... Page size
- M_T ... Total memory size

- 64 nodes - 2^6
- 16 MB/node - 2^{24}

Virtual Address Space Translation

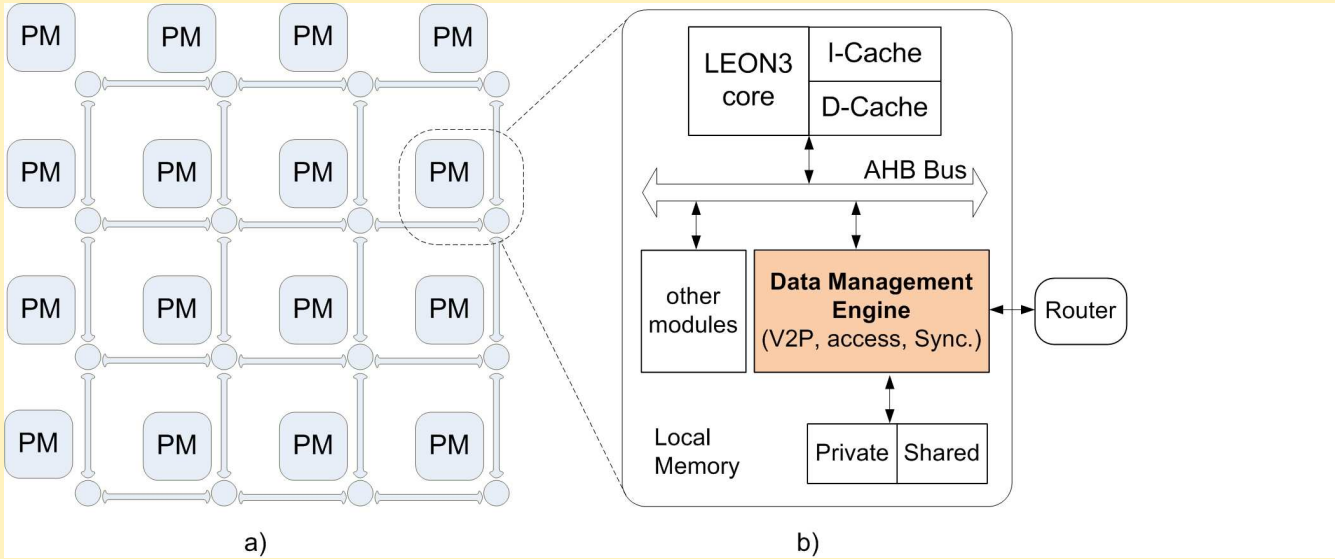


$$S_T = (\log_2 N + \log_2 M_N - \log_2 P_S) \cdot (M_T / P_S)$$

- S_T ... Size of translation table per node
- N ... Number of nodes
- M_N ... Memory per node
- P_S ... Page size
- M_T ... Total memory size

- 64 nodes - 2^6
- 16 MB/node - 2^{24}
- 1 GB total memory - 2^{30}

Virtual Address Space Translation

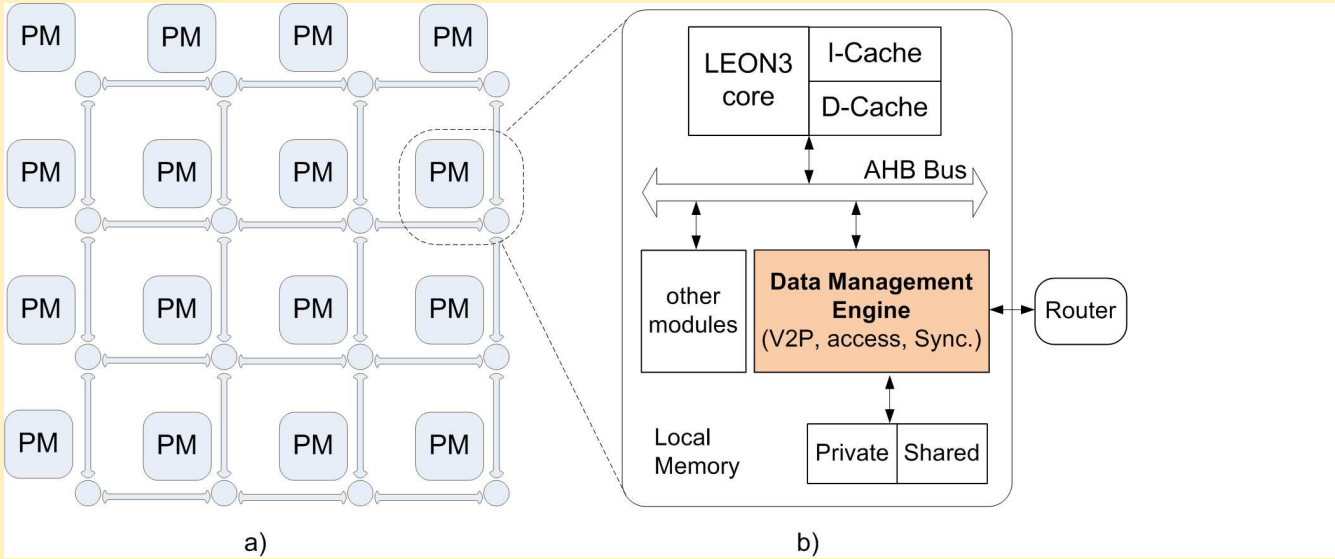


$$S_T = (\log_2 N + \log_2 M_N - \log_2 P_S) \cdot (M_T / P_S)$$

- S_T ... Size of translation table per node
- N ... Number of nodes
- M_N ... Memory per node
- P_S ... Page size
- M_T ... Total memory size

- 64 nodes - 2^6
- 16 MB/node - 2^{24}
- 1 GB total memory - 2^{30}
- Page size 1KB - 2^{10}

Virtual Address Space Translation

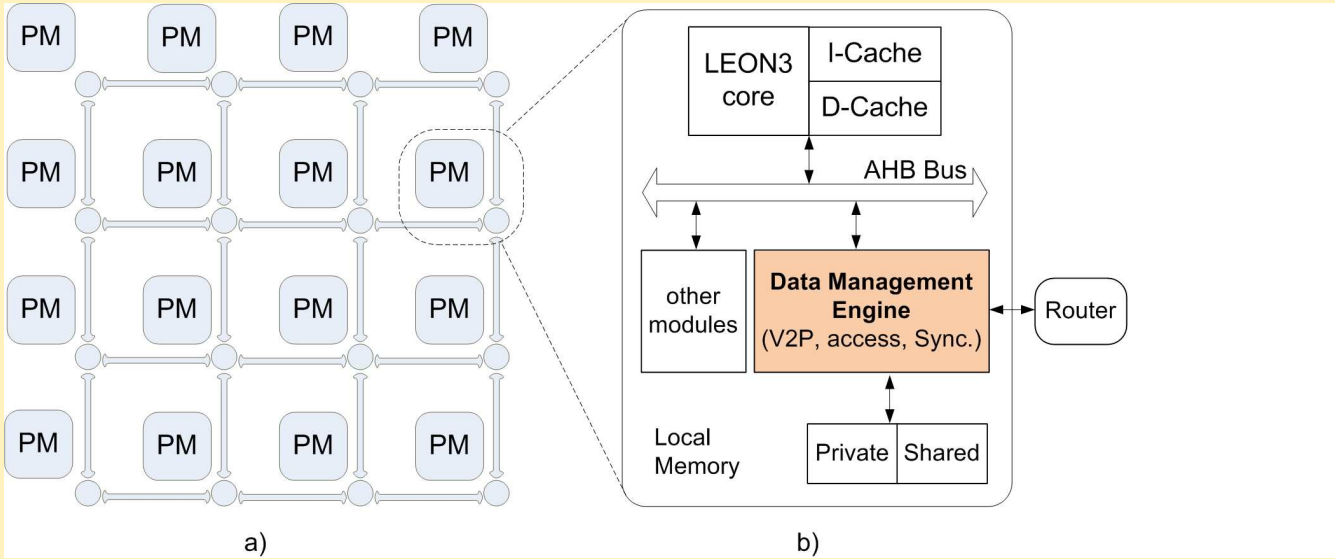


$$S_T = (\log_2 N + \log_2 M_N - \log_2 P_S) \cdot (M_T / P_S)$$

- S_T ... Size of translation table per node
- N ... Number of nodes
- M_N ... Memory per node
- P_S ... Page size
- M_T ... Total memory size

- 64 nodes - 2^6
- 16 MB/node - 2^{24}
- 1 GB total memory - 2^{30}
- Page size 1KB - 2^{10}
- Translation table: 1 M entries - 2^{20}

Virtual Address Space Translation

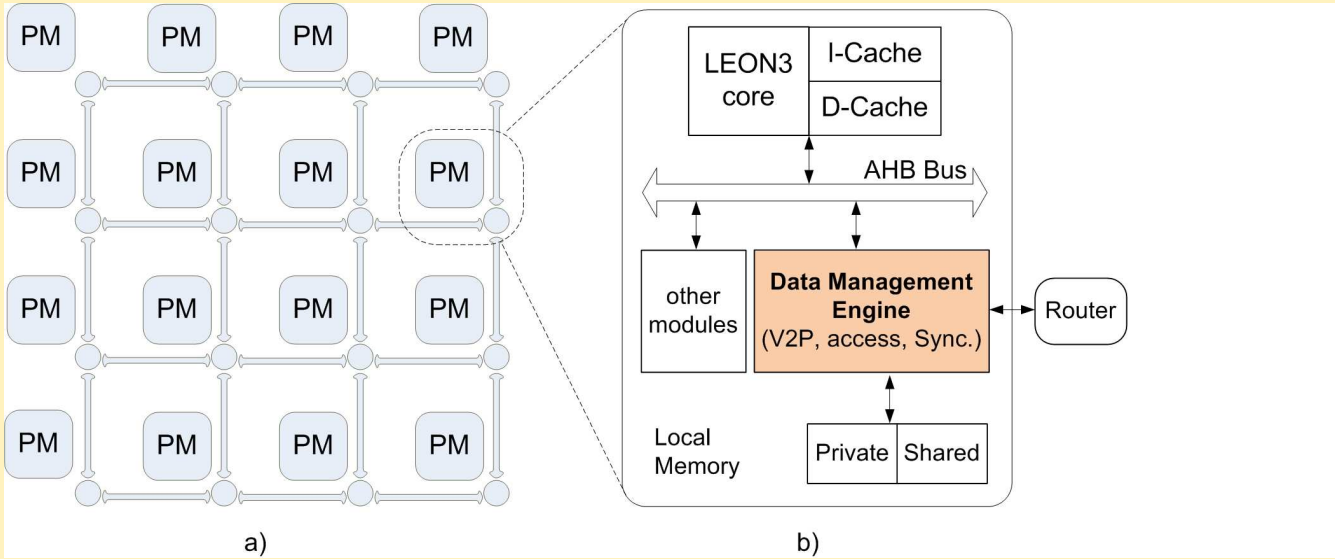


$$S_T = (\log_2 N + \log_2 M_N - \log_2 P_S) \cdot (M_T / P_S)$$

S_T ... Size of translation table per node
N ... Number of nodes
M_N ... Memory per node
P_S ... Page size
M_T ... Total memory size

- 64 nodes - 2^6
- 16 MB/node - 2^{24}
- 1 GB total memory - 2^{30}
- Page size 1KB - 2^{10}
- Translation table: 1 M entries - 2^{20}
- 1 entry: 6+14=20 bit $\rightarrow$ 3 Byte

Virtual Address Space Translation

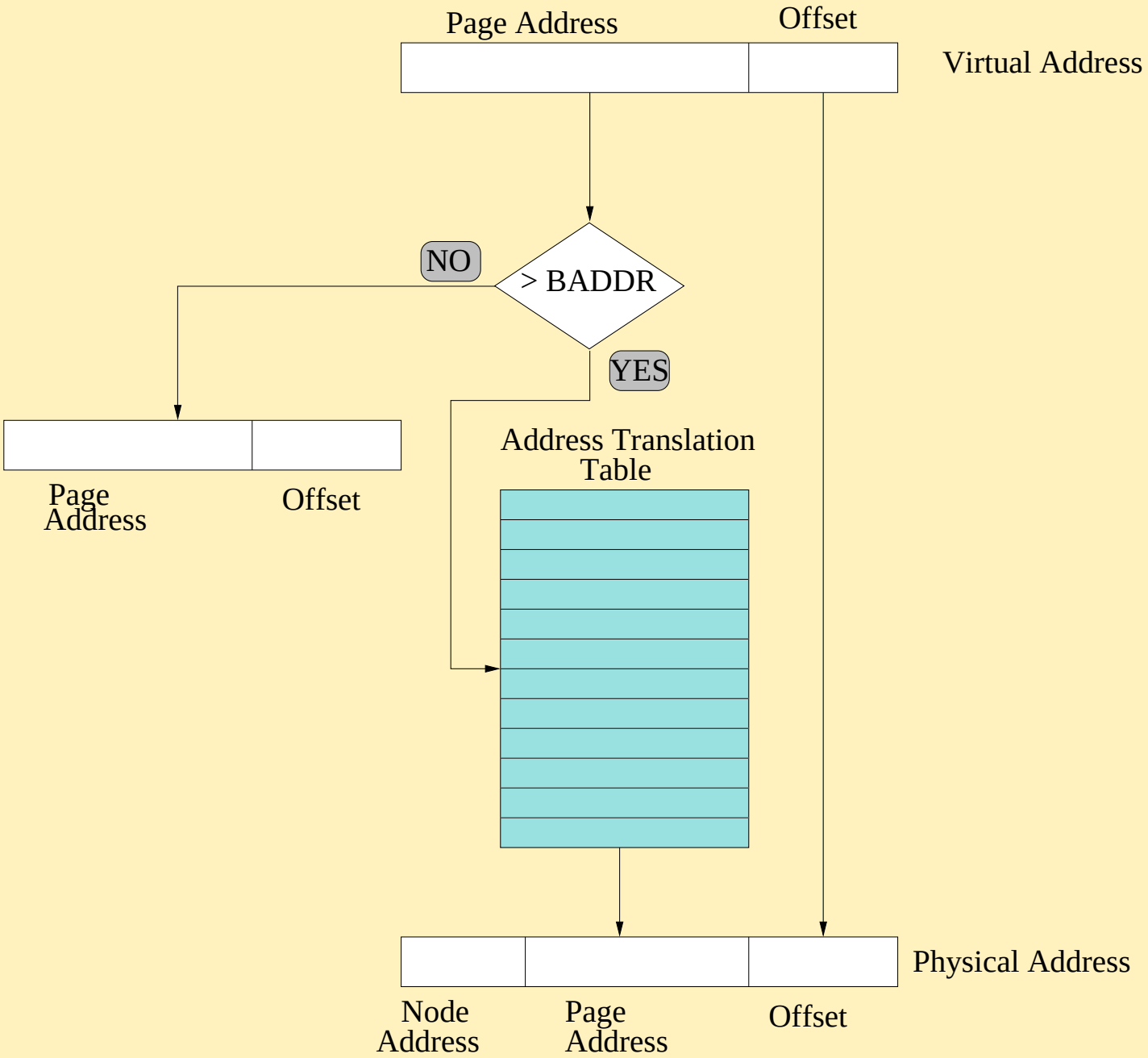


$$S_T = (\log_2 N + \log_2 M_N - \log_2 P_S) \cdot (M_T / P_S)$$

S_T ... Size of translation table per node
N ... Number of nodes
M_N ... Memory per node
P_S ... Page size
M_T ... Total memory size

- 64 nodes - 2^6
- 16 MB/node - 2^{24}
- 1 GB total memory - 2^{30}
- Page size 1KB - 2^{10}
- Translation table: 1 M entries - 2^{20}
- 1 entry: 6+14=20 bit $\rightarrow$ 3 Byte
- $\Rightarrow$ 3MB/node (18.75%) for translation table

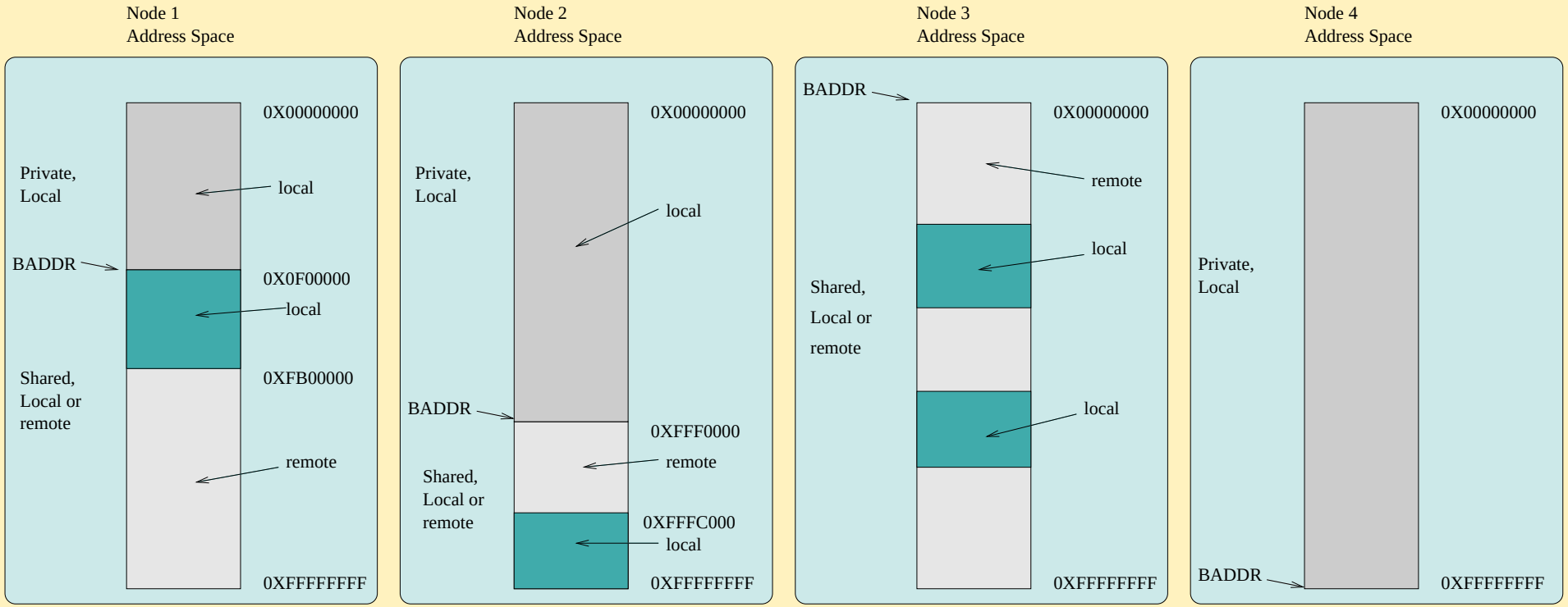
DME Virtual Address Translation



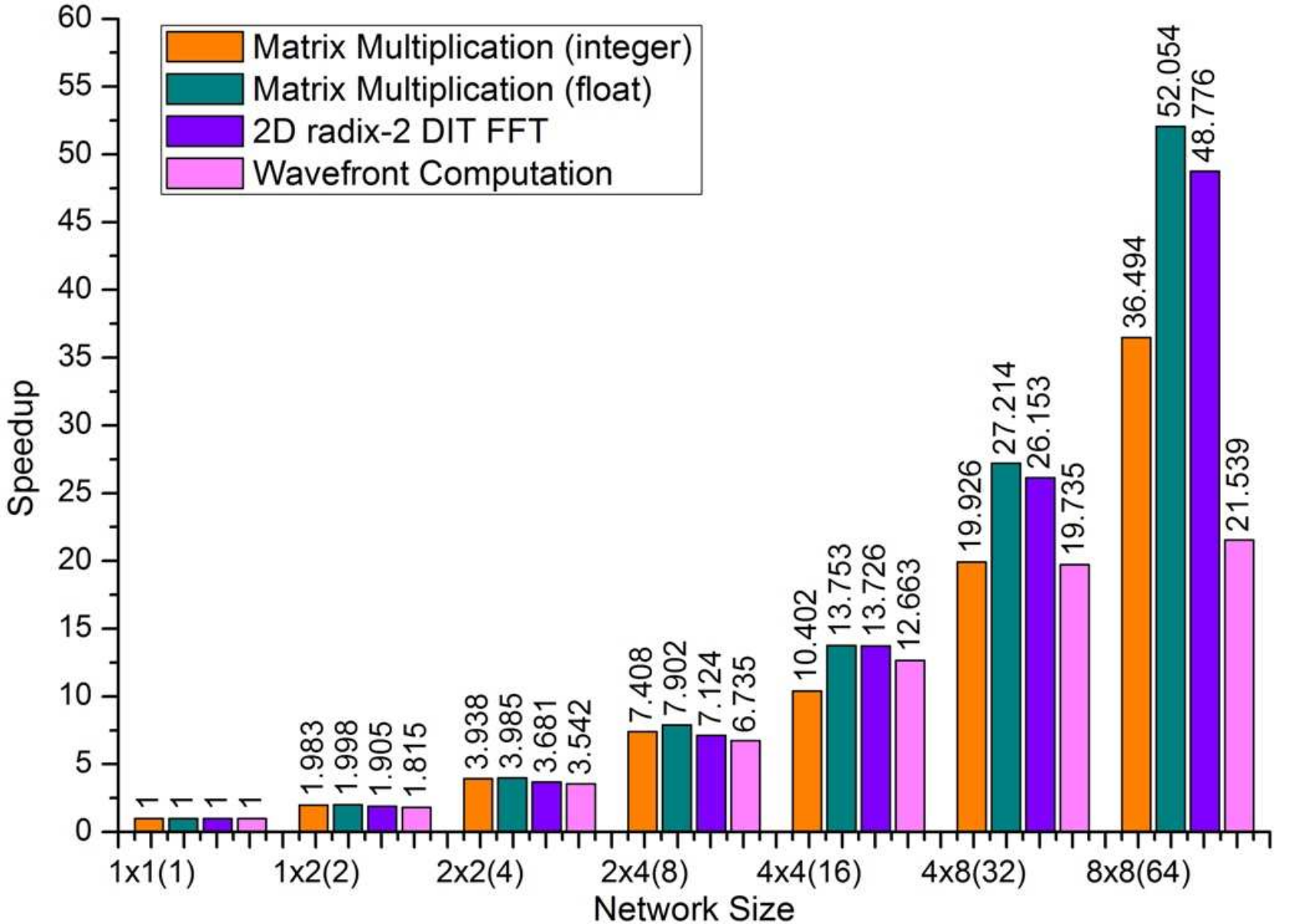
Supported Memory Partitions

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

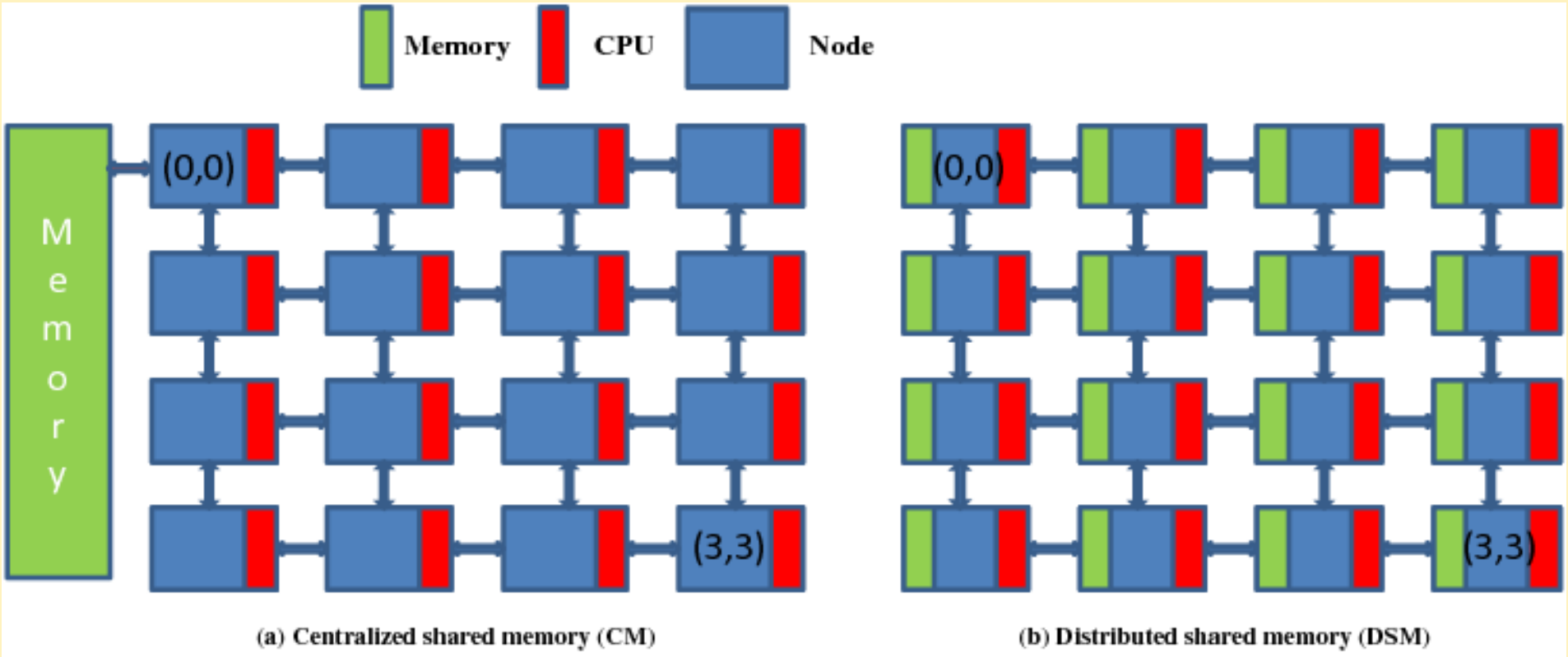
Address Space Management



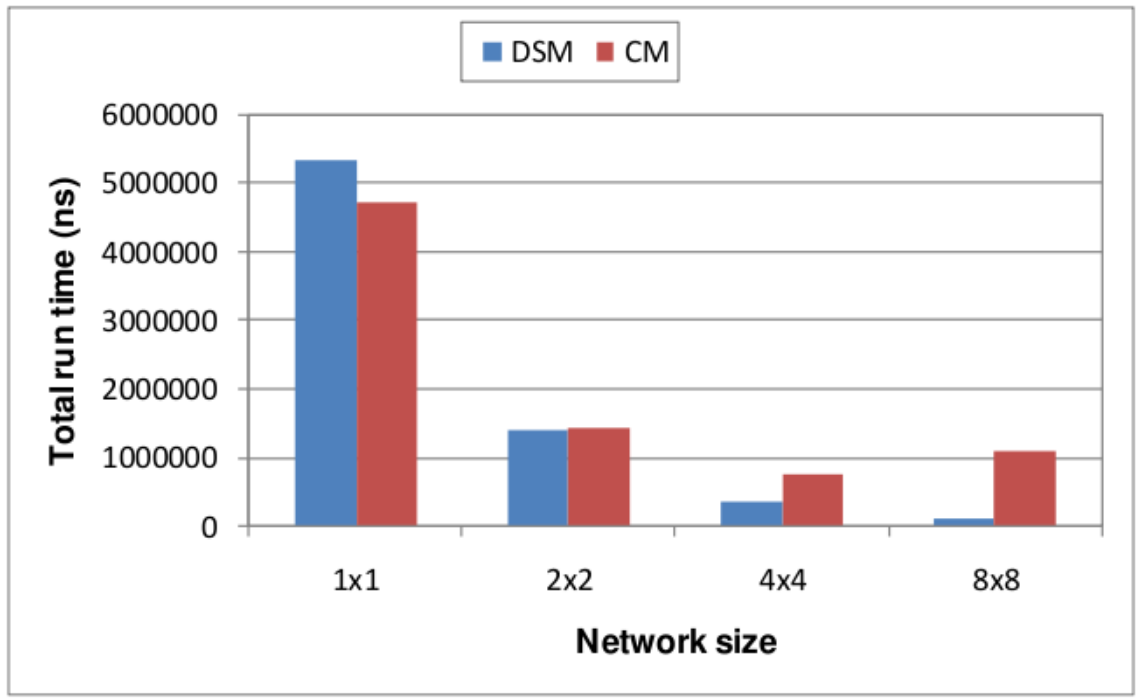
Multi-core Speedup

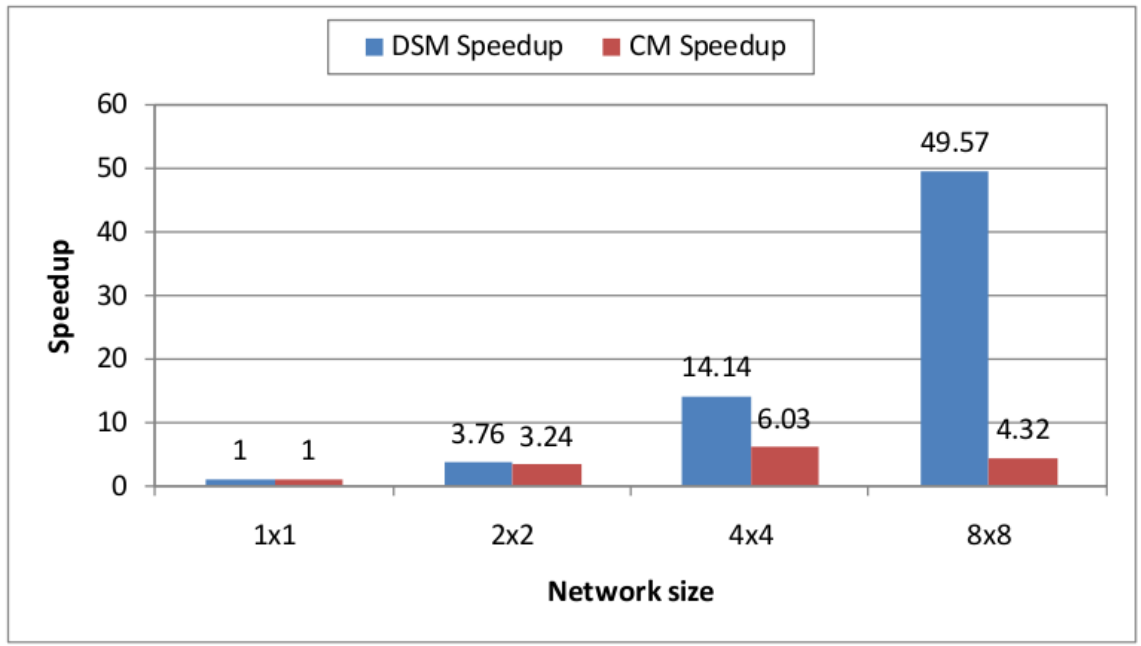


Experiment: Central vs. Distributed Shared Memory

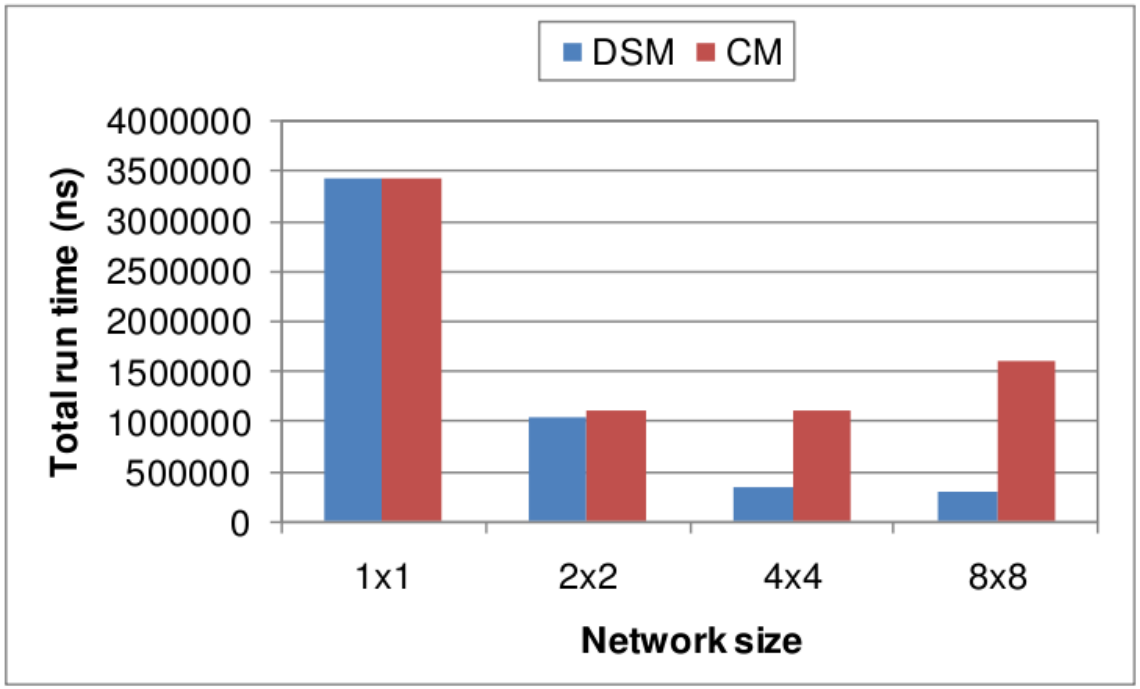


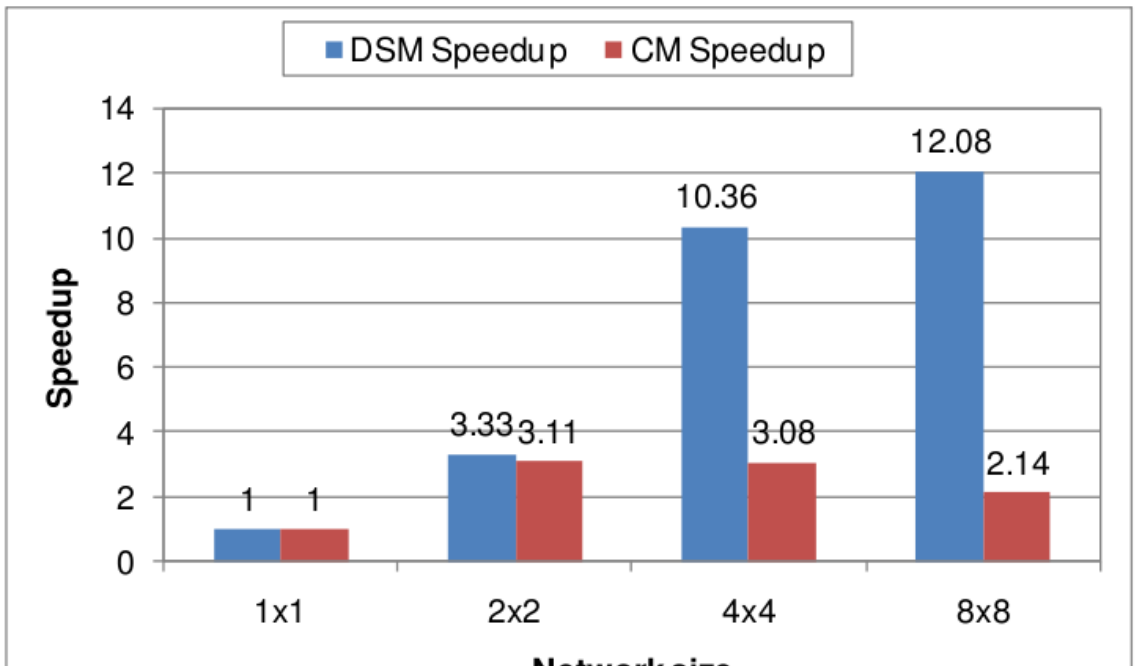
Central vs. Distributed Shared Memory: Matrix Multiplication





Central vs. Distributed Shared Memory: FFT





Summary

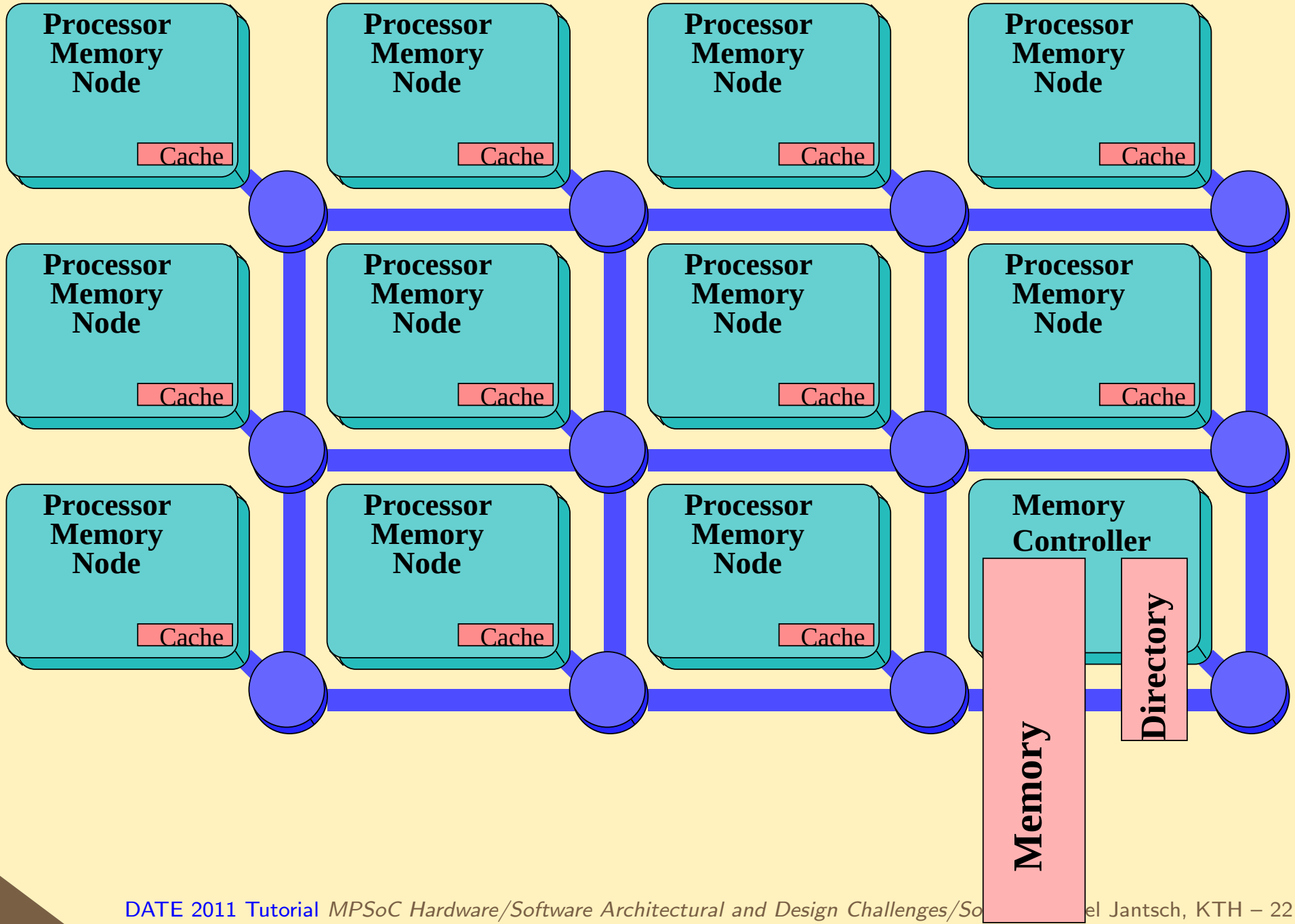
- After computation (multi-core) and communication (NoC), memory access must be parallalized
- DME parallizes memory handling
- DME supports
  - Central/distributed memory
  - Private/shared
  - Physical/virtual address space
- DME features
  - Synchronization support
  - Cache coherence protocols
  - Memory consistency support
  - Message passing communication
  - Dynamic memory allocator
- Future Work
  - Improve cache coherency
  - Improved high level support for programming models
  - Adapt DME to integrate heterogeneous SoC subsystems

Data Management Engine Implementation

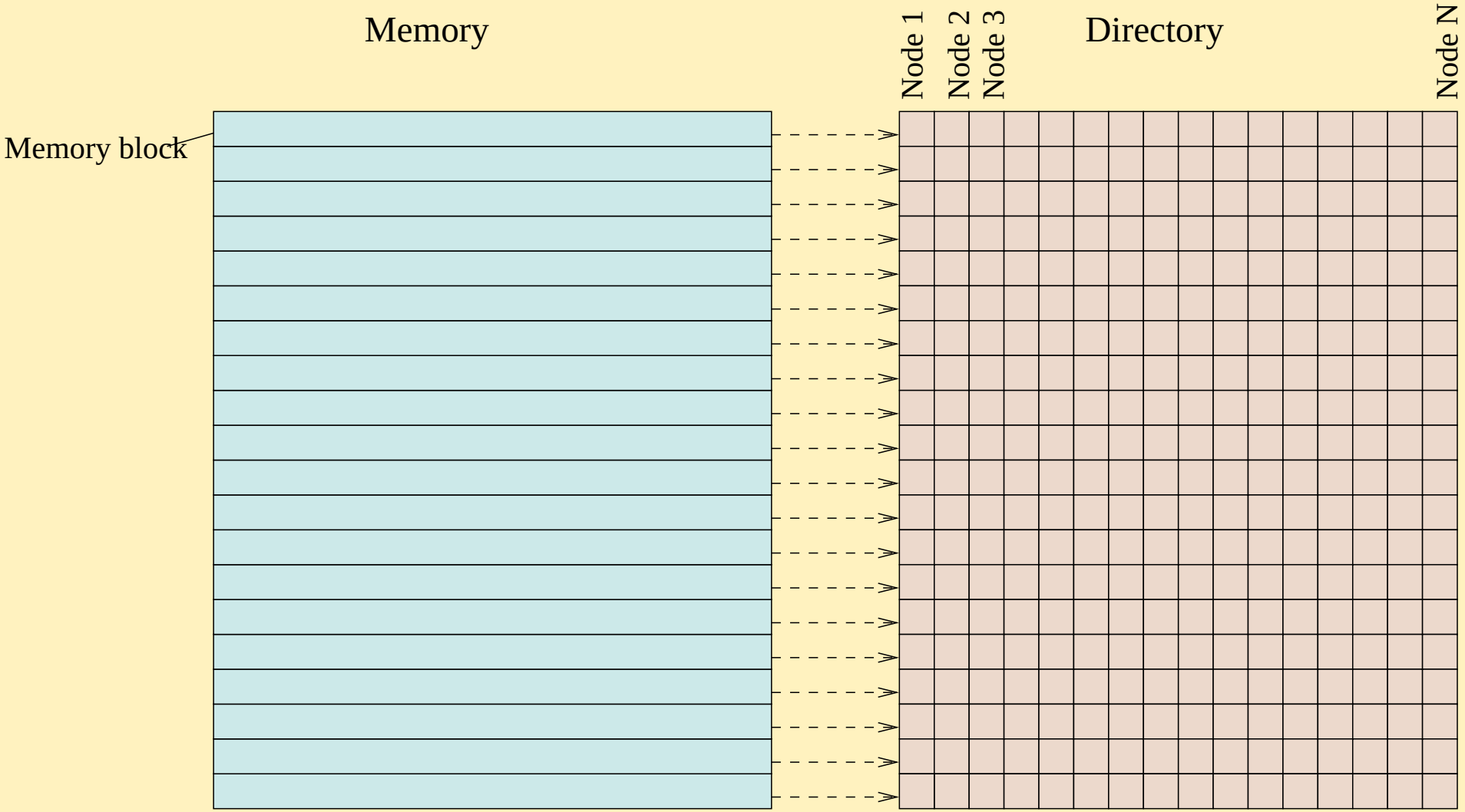
	Optimized for area	Optimized for speed
Frequency	444 MHz (2.25 ns)	455 MHz (2.2 ns)
Area (Logic)	44k NAND gates	51k NAND gates
Area (Control Store)	300k NAND gates	

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

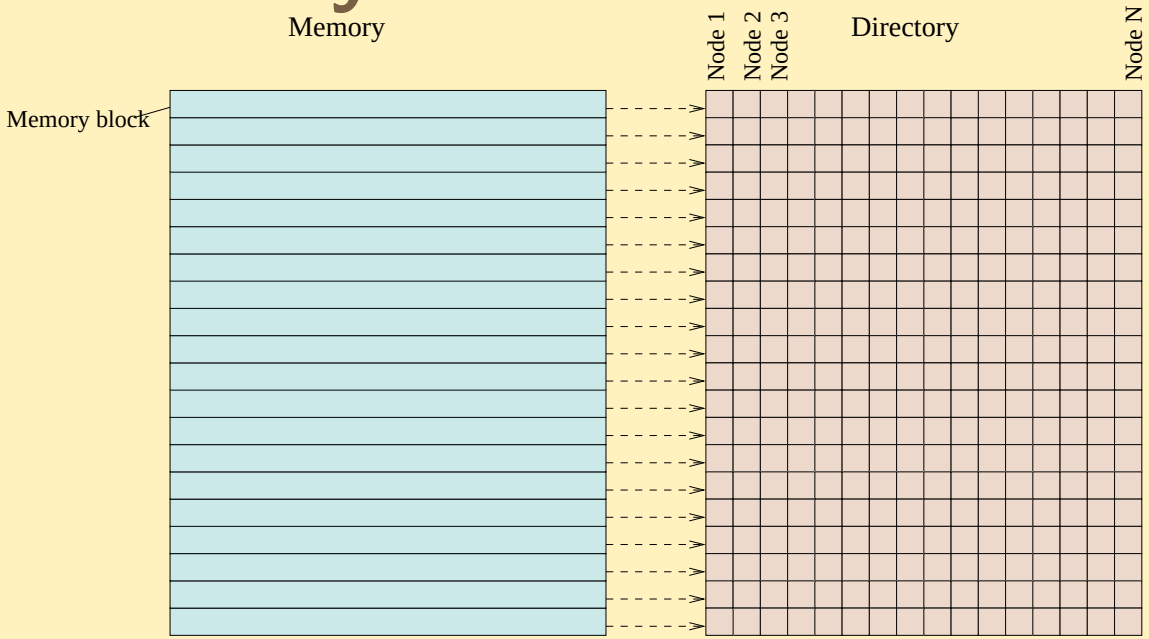
Directory Based Cache Coherence



Directory Structure



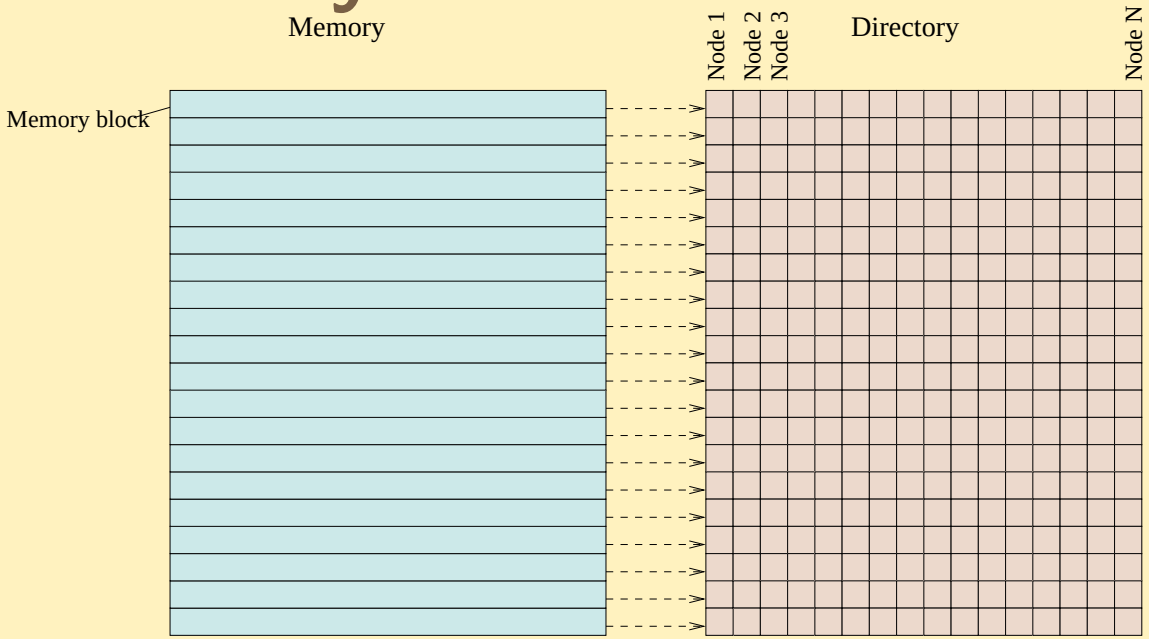
Directory Size
1 entry / memory block



$$S_D = S_T / S_B \times (N + 1)$$

S_D ... Size of directory
S_T ... Total memory
N ... No of nodes
C_B ... blocks/cache
S_B ... Block size
S_C ... Cache size

Directory Size
1 entry / memory block

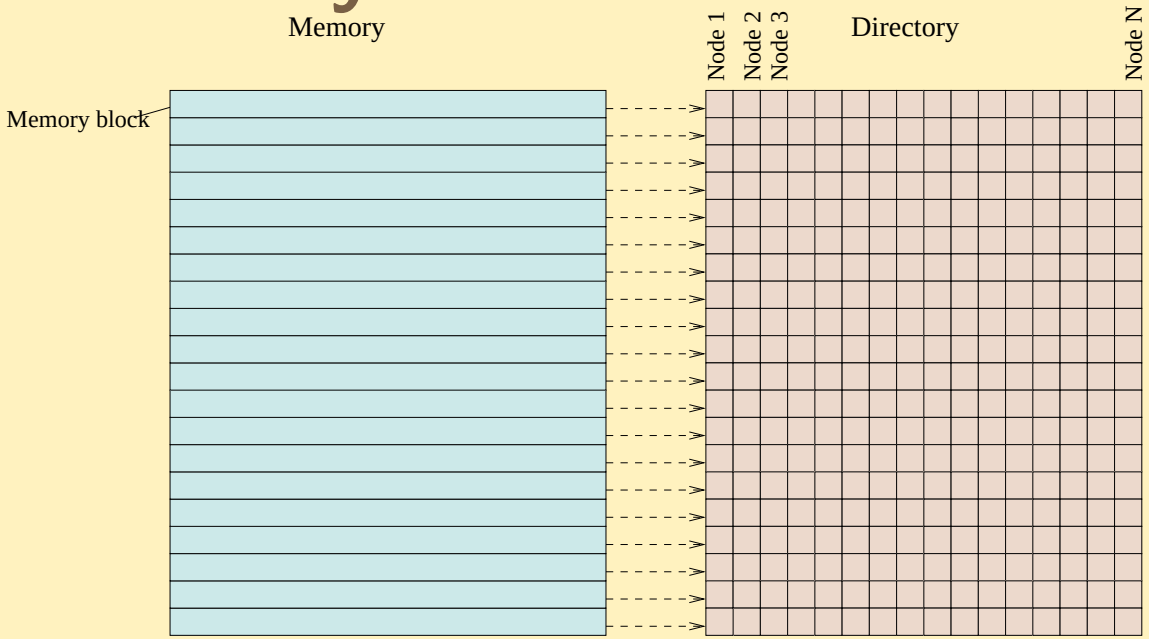


$$S_D = S_T / S_B \times (N + 1)$$

- S_D ... Size of directory
- S_T ... Total memory
- N ... No of nodes
- C_B ... blocks/cache
- S_B ... Block size
- S_C ... Cache size

- 64 nodes - 2^6

Directory Size
1 entry / memory block

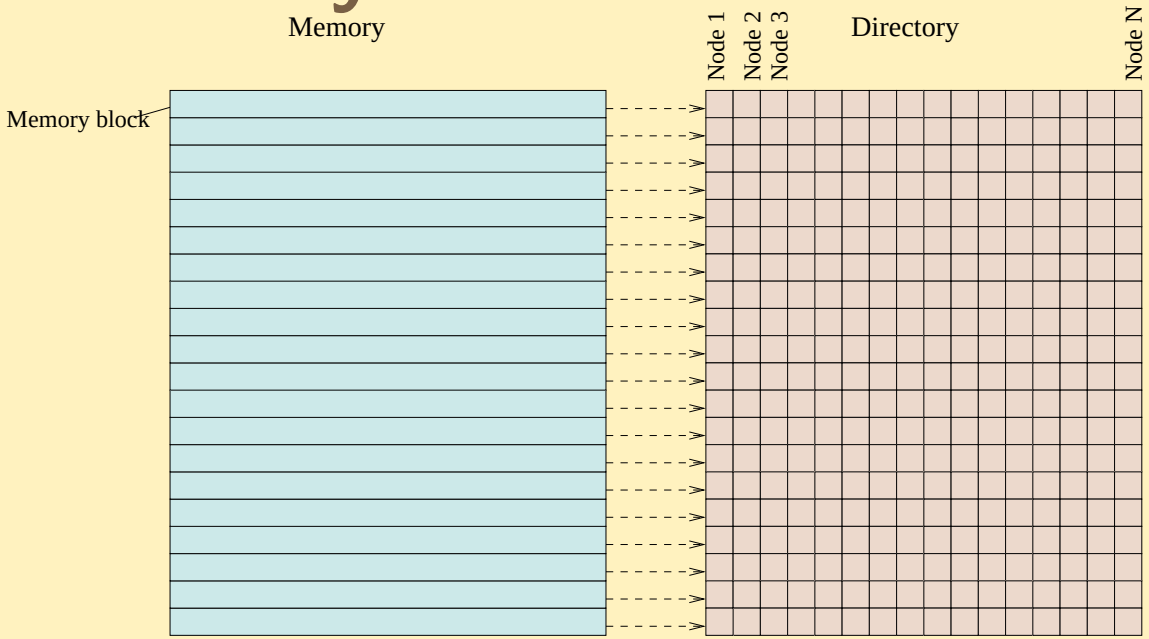


$$S_D = S_T / S_B \times (N + 1)$$

- S_D ... Size of directory
- S_T ... Total memory
- N ... No of nodes
- C_B ... blocks/cache
- S_B ... Block size
- S_C ... Cache size

- 64 nodes - 2^6
- 4 GB total memory - 2^{32}

Directory Size
1 entry / memory block

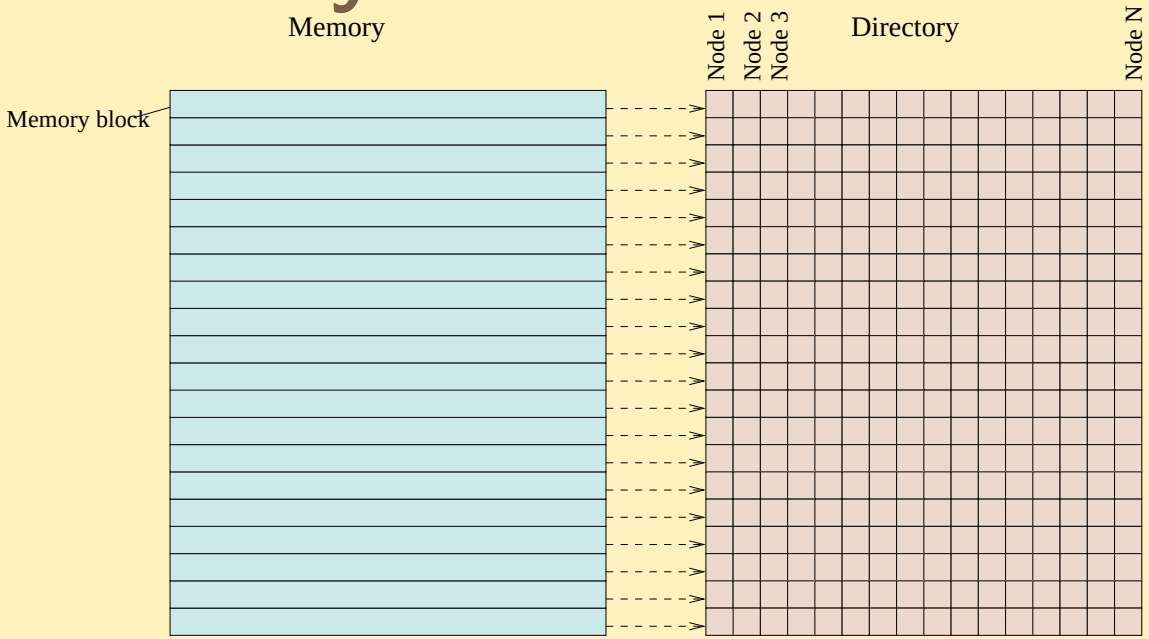


$$S_D = S_T / S_B \times (N + 1)$$

- S_D ... Size of directory
- S_T ... Total memory
- N ... No of nodes
- C_B ... blocks/cache
- S_B ... Block size
- S_C ... Cache size

- 64 nodes - 2^6
- 4 GB total memory - 2^{32}
- Block size 64B - 2^6

Directory Size
1 entry / memory block

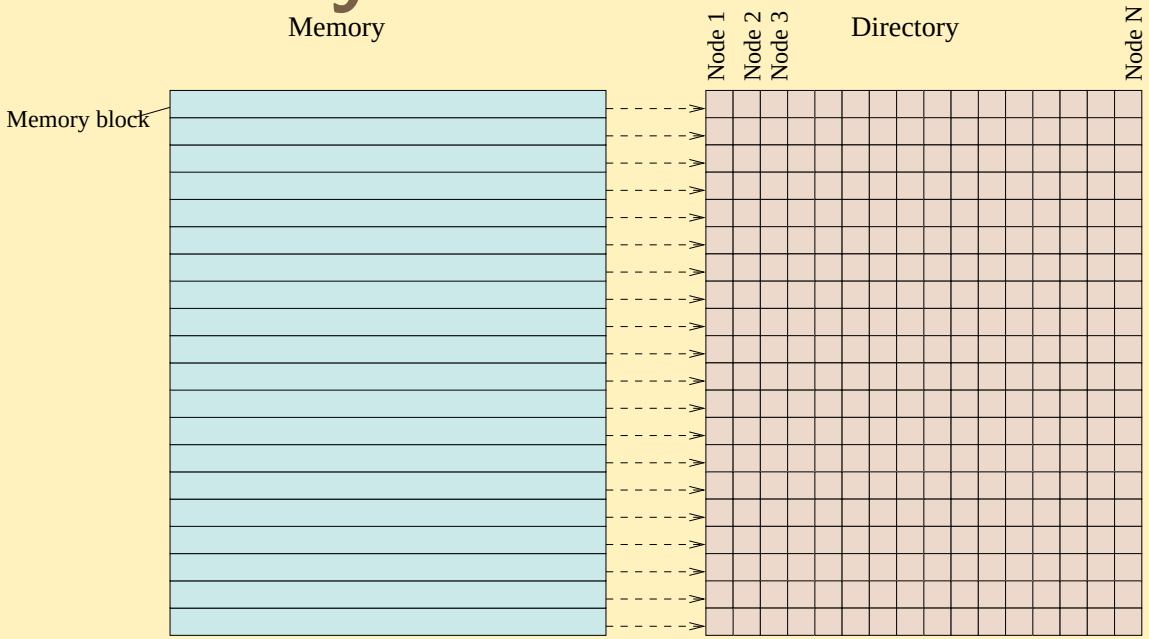


$$S_D = S_T / S_B \times (N + 1)$$

- S_D ... Size of directory
- S_T ... Total memory
- N ... No of nodes
- C_B ... blocks/cache
- S_B ... Block size
- S_C ... Cache size

- 64 nodes - 2^6
- 4 GB total memory - 2^{32}
- Block size 64B - 2^6
- Cache size 8MB - 2^{23}

Directory Size
1 entry / memory block

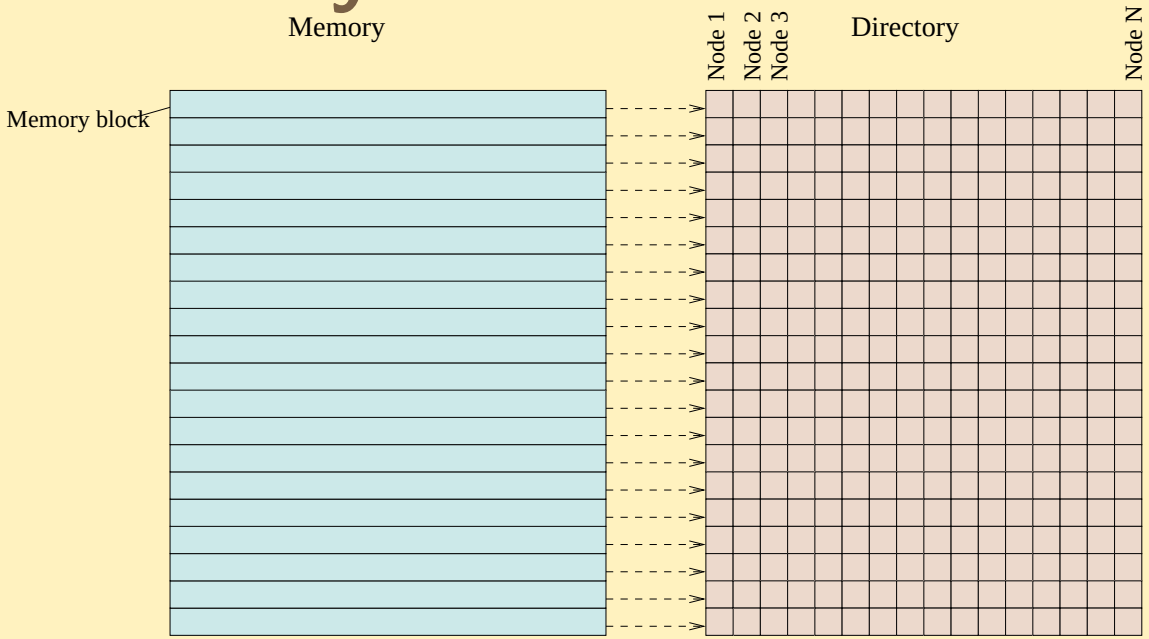


$$S_D = S_T / S_B \times (N + 1)$$

- S_D ... Size of directory
- S_T ... Total memory
- N ... No of nodes
- C_B ... blocks/cache
- S_B ... Block size
- S_C ... Cache size

- 64 nodes - 2^6
- 4 GB total memory - 2^{32}
- Block size 64B - 2^6
- Cache size 8MB - 2^{23}
- No of blocks/cache: 128K - 2^{17}

Directory Size
1 entry / memory block

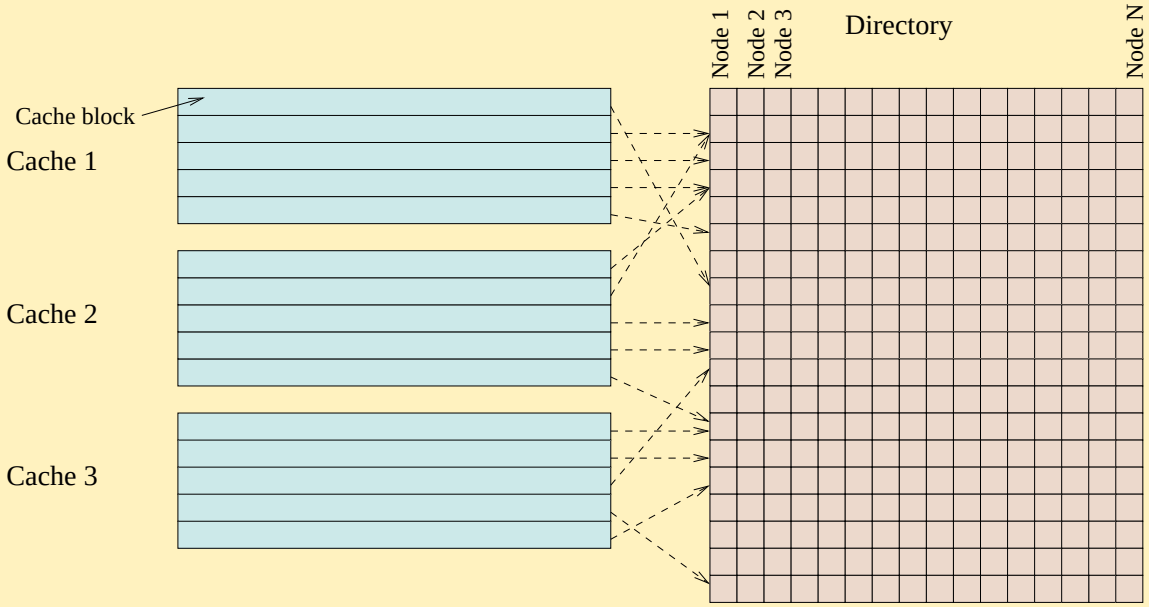


$$S_D = S_T / S_B \times (N + 1)$$

S_D ... Size of directory
S_T ... Total memory
N ... No of nodes
C_B ... blocks/cache
S_B ... Block size
S_C ... Cache size

- 64 nodes - 2^6
- 4 GB total memory - 2^{32}
- Block size 64B - 2^6
- Cache size 8MB - 2^{23}
- No of blocks/cache: 128K - 2^{17}
- $S_D = 520$MB
  13% of total memory
  102% of total cache size

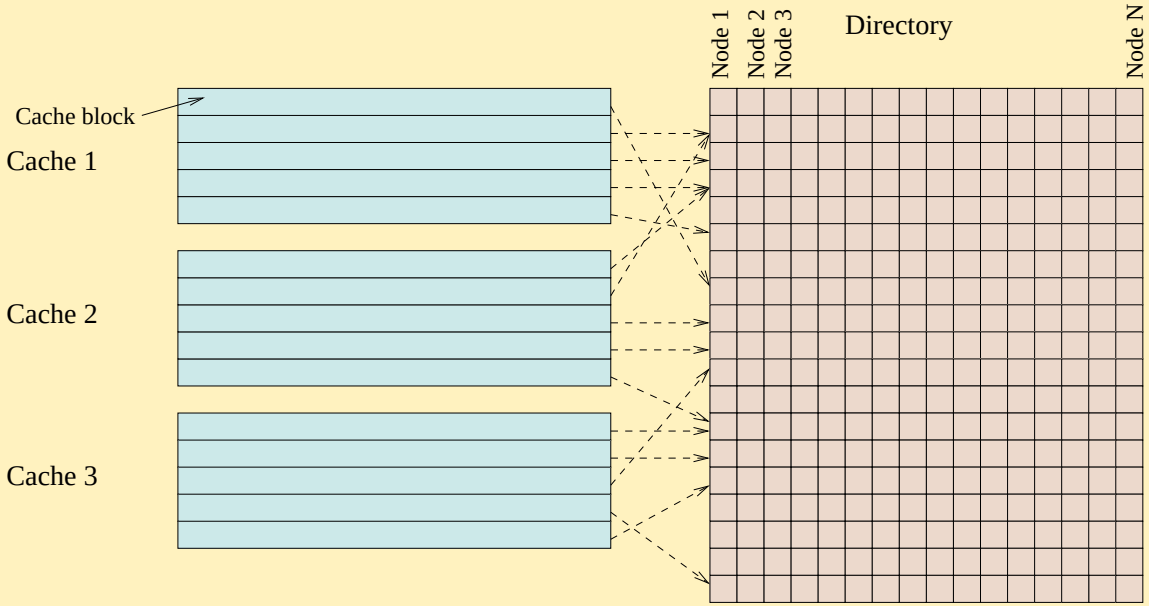
Directory Size
1 entry / cache block



$$S_D = N \times C_B \times (N + 1)$$

S_D ... Size of directory
S_T ... Total memory
N ... No of nodes
C_B ... blocks/cache
S_B ... Block size
S_C ... Cache size

Directory Size
1 entry / cache block

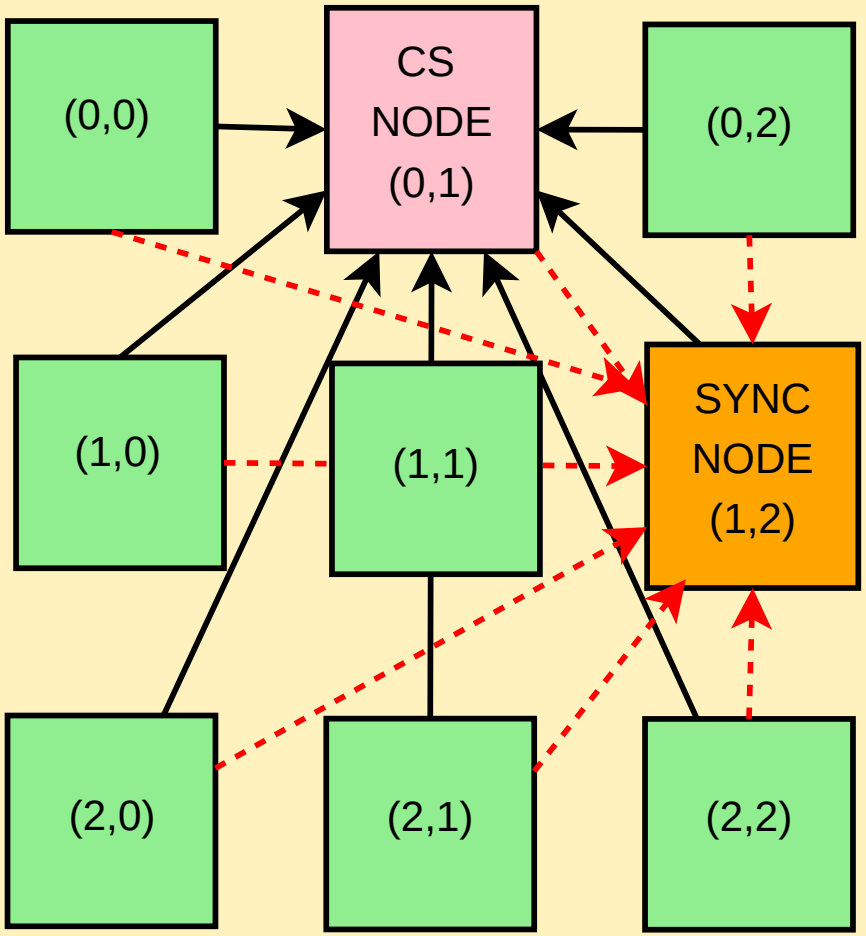


$$S_D = N \times C_B \times (N + 1)$$

S_D ... Size of directory
S_T ... Total memory
N ... No of nodes
C_B ... blocks/cache
S_B ... Block size
S_C ... Cache size

- 64 nodes - 2^6
- 4 GB total memory - 2^{32}
- Block size 64B - 2^6
- Cache size 8MB - 2^{23}
- No of blocks/cache: 128K - 2^{17}
- $S_D = 65$MB
  1.5% of total memory
  12.6% of total cache size

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