

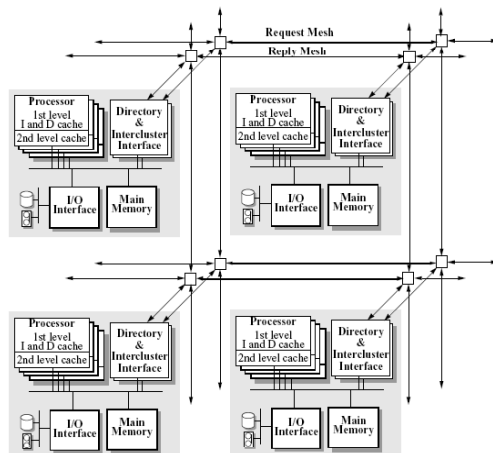
# The Directory-Based Cache Coherence Protocol for the DASH Multiprocessor

Lenoski, Laudon, Gharachorloo,  
Gupta, Hennessy  
ISCA 1990

## DASH Priorities

- Correctness (issues?)
- Performance
  - Latency
  - Bandwidth
- Distributed Control and Complexity

## Overview



## Overview

- Distributed directory
- Illinois protocol???
- Off-the-shelf SMP
- Split transaction
  - Def?
  - How support?

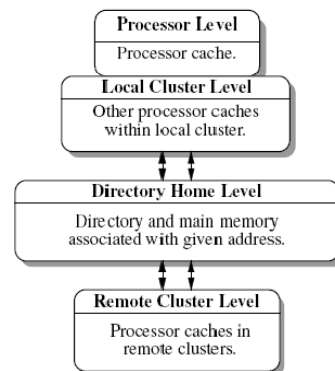
## Directory Controller Board

## Memory Consistency

- Stay tuned...
- Release Consistency – one of several proposed weak consistency models.
- Requires explicit *fence* operations.

## DASH Coherence

- Local, home, remote nodes
- Invalidation-based
- Nodes are clusters, not processors.
- Memory states
  - Uncached
  - Shared-remote
  - Dirty-remote

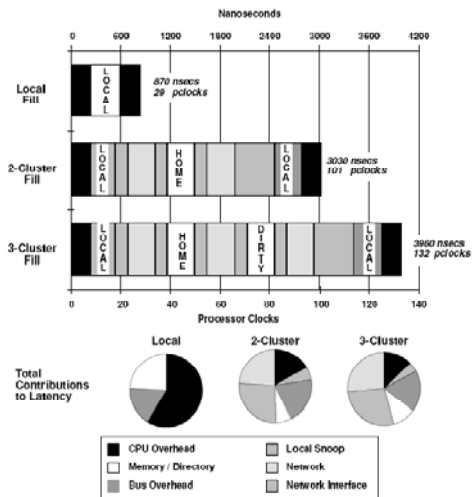


## DASH Coherence

- Processor Cache states
  - D, S, I
- Read
  - Local L2, cache-cache, RAC, home node, ...
  - At home node, causes read or forward to remote for read
- RAC – looks like another processor cache on the local bus
- Read of dirty line held in local cluster????
- Three hop protocol
  - Implications?
  - What happens with two reads close together (of remotely held data?)



## DASH Read Latency



## Read-Exclusive

- Local L2, cache-cache, RAC, home node
- Home node sends invalidations
- May need to forward



## Other protocol features

- Queued locks
- Read prefetch, read-excl prefetch
- Explicit write-update

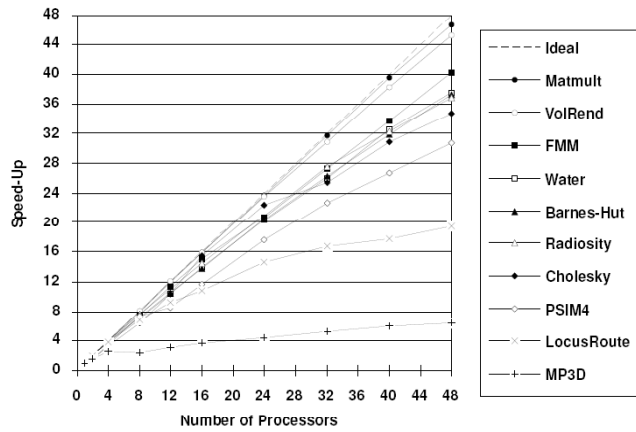


## Bit-vector directory

- Not scalable?
- Other Solutions?



# Performance



# Details

TABLE 6. MP3D memory access characteristics.

Execution Attribute	1 PE	2 PEs	4 PEs	8 PEs	12 PEs	16 PEs	24 PEs	32 PEs	40 PEs	48 PEs
Speedup	1.00	1.60	2.48	2.33	3.12	3.74	4.42	5.27	5.99	6.42
Busy Pelks between Proc. Stalls	82.5	46.1	43.6	44.2	38.6	49.7	56.0	58.9	61.5	84.5
Est. Processor Utilization (%)	73.8	67.5	62.0	34.6	24.8	27.2	26.5	26.2	26.4	32.1
Cache Read (%)	98.3	72.9	67.5	57.7	56.9	54.0	51.8	52.2	53.4	51.6
Cache Read Exclusive (%)	0.9	26.9	32.2	42.0	42.7	45.5	47.3	46.6	45.2	46.3
Cache Lock (%)	0.3	0.1	0.0	0.0	0.2	0.3	0.5	0.8	1.0	1.5
Cache Unlock (%)	0.3	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
References to Local Memory (%)	100.0	100.0	100.0	46.5	29.3	23.0	16.0	11.6	8.9	8.4
Local Ref's satisfied Locally (%)	99.8	99.9	100.0	79.3	74.7	67.9	62.3	63.3	66.1	59.3
Remote Ref's satisfied Locally (%)	0.0	0.0	0.0	30.8	18.9	17.1	12.9	10.0	8.1	10.4
Remote Ref's satisfied in Home (%)	100.0	100.0	100.0	99.3	89.0	80.5	71.6	71.0	73.3	63.1
Measured Local Cache Fill (Pelks)	29.4	30.8	37.0	37.2	38.1	36.6	36.0	35.7	35.2	34.1
Unloaded Remote Cache Fill (Pelks)	101.0	101.0	101.0	101.3	108.6	114.7	120.3	119.8	117.5	124.6
Measured Remote Cache Fill (Pelks)	109.5	117.5	130.4	138.7	160.9	172.2	186.8	191.3	192.7	201.6
Bus Utilization (%)	16.6	42.9	78.6	77.3	78.6	73.6	71.0	69.6	67.6	62.2
Req. Net Bisection Util. (%)	0.5	0.6	0.9	10.7	15.0	30.6	32.5	33.1	32.5	30.6
Reply Net Bisection Util. (%)	0.4	0.5	0.9	13.7	15.6	30.1	29.6	29.3	28.7	26.3



# Token Coherence: Decoupling Performance and Correctness

Martin, Hill, Wood  
ISCA 2003



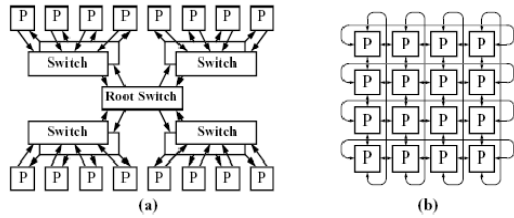
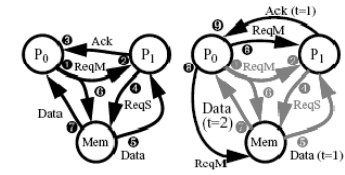


Figure 1. (a) 16-processor two-level tree interconnect and (b) 16-processor (4x4) bi-directional torus interconnect. The boxes marked "P" represent highly-integrated nodes that include a processor, caches, memory controller, and coherence controllers. The indirect broadcast tree uses discrete switches, while the torus is a directly connected interconnect. In this example, the torus has lower latency (two vs. four chip crossings on average) and does not require any glue chips; however, unlike the indirect tree, the torus provides no request total order, making it unsuitable for traditional snooping.



(a) fast but incorrect (b) using token coherence  
Figure 2. Example Race. A request for shared (ReqS) racing with a request for modified (ReqM).

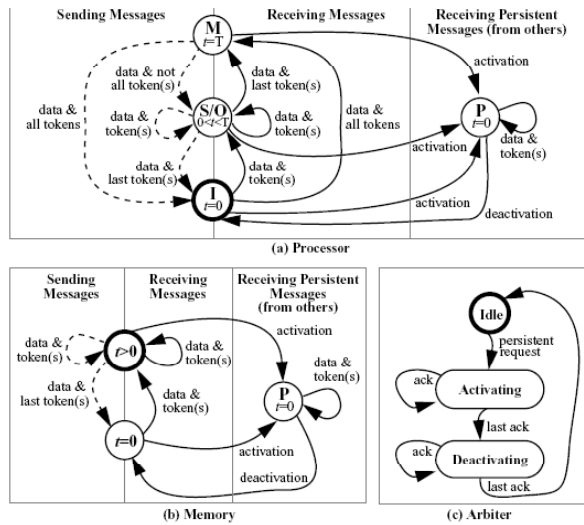
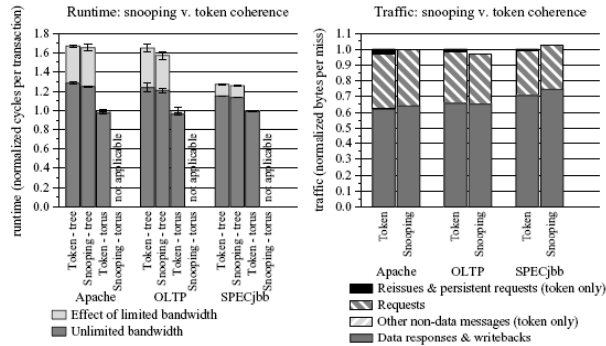


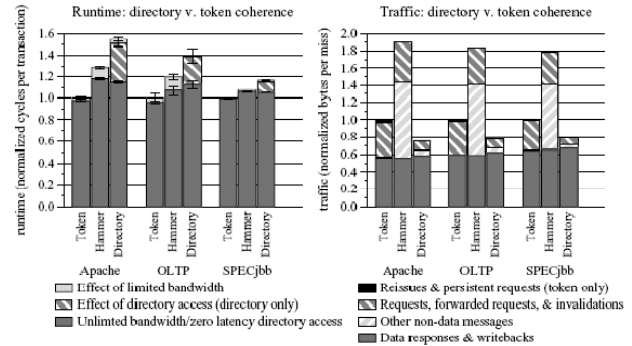
Figure 3. Correctness substrate state transitions for the (a) processor, (b) memory, and (c) persistent request arbiter. As a simplification, the figure shows only tokens sent with data. The symbol  $t$  represents the current token count, and  $T$  represents all the tokens. Solid arcs are transitions in response to incoming messages. Dashed arcs are transitions a performance protocol (Section 4) can invoke at any time (e.g., when receiving a transient request). The "P" states occur when a node receives another processor's persistent request from the arbiter. Each processor must also remember its own persistent request, not explicitly shown in this figure. The initial states are emphasized with thick borders.

Table 2. Overhead due to reissued requests

Workload	Percentage of Misses			
	Not Reissued	Reissued Once	Reissued > Once	Persistent Requests
Apache	95.75%	3.25%	0.71%	0.29%
OLTP	97.57%	1.79%	0.43%	0.21%
SPECjbb	97.60%	2.03%	0.30%	0.07%
Average	96.97%	2.36%	0.48%	0.19%



(a) (b)  
**Figure 4. Snooping v. TokenB: runtime and traffic**



(a) (b)  
**Figure 5. Directory and Hammer v. TokenB: runtime and traffic**