Lecture 16: Networks & Interconnect (Internetworking, Protocols, Examples) and Intro to Multiprocessors

Professor David A. Patterson

Computer Science 252

Fall 1996

Review: Interconnect Issues

- **Performance Measures**
- **Interface Issues**
- **Network Media**
- **Connecting Multiple Computers**

Review: Interconnections

- **Media sets cost, distance**
- **Shared vs. Swicthed Media determines BW**
- **HW and SW Interface to computer affects overhead, latency, bandwidth**
- **Topologies: many to chose from, but (SW) overheads make them look alike; cost issues in topologies**
- **Routing issues: store and forward vs. cut through, congestion, ...**
- **Standardization key for LAN, WAN**

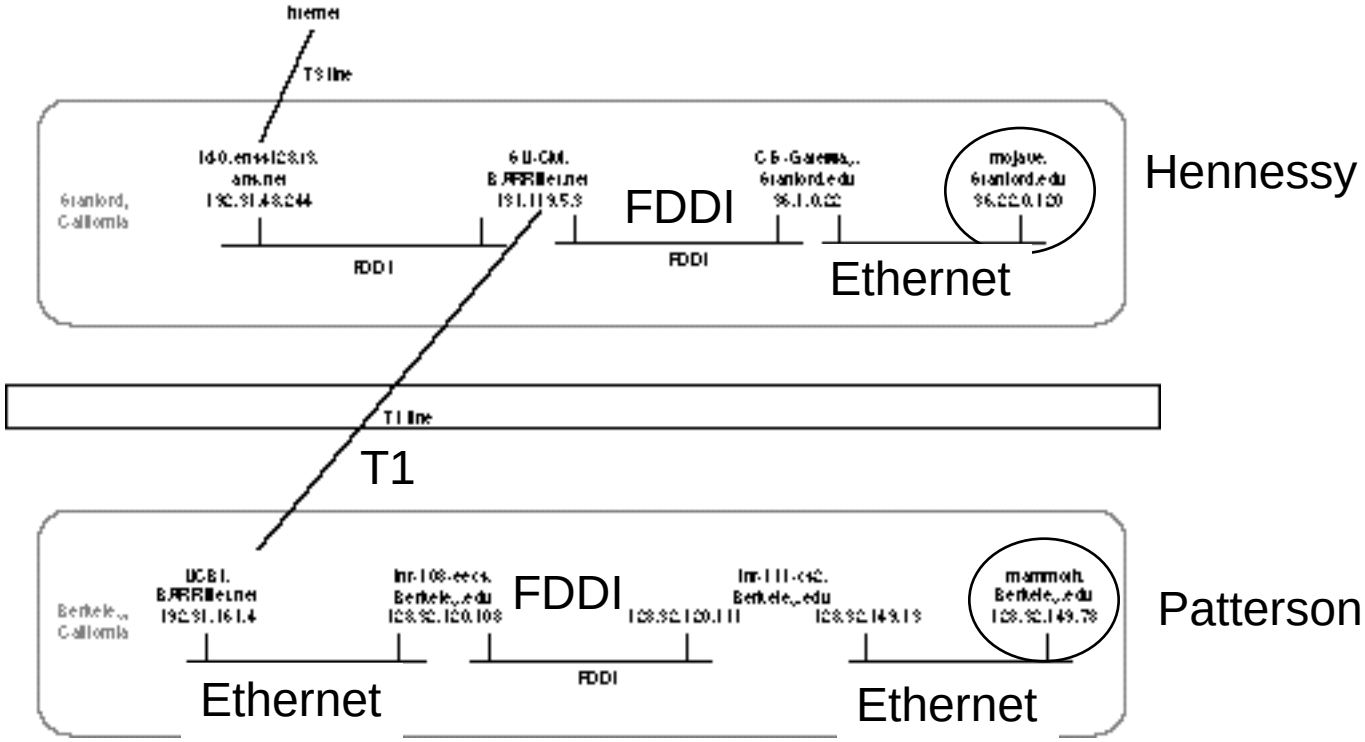
Cross-Cutting Issues for Networking

- **Efficient Interface to Memory Hierarchy vs. to Network**
  - **SPEC ratings => fast to memory hierarchy**
  - **Writes go via write buffer, reads via L1 and L2 caches**
- **Example: 40 MHz SS-2 vs 50 MHz SS-20, no L2$ vs 50 MHz SS-20 with L2$ I/O bus latency**
- **SS-2: combined memory, I/O bus => 200 ns**
- **SS-20, no L2$: 2 busses +300ns => 500ns**
- **SS-20, w L2$: cache miss+500ns => 1000ns**

Protocols: HW/SW Interface

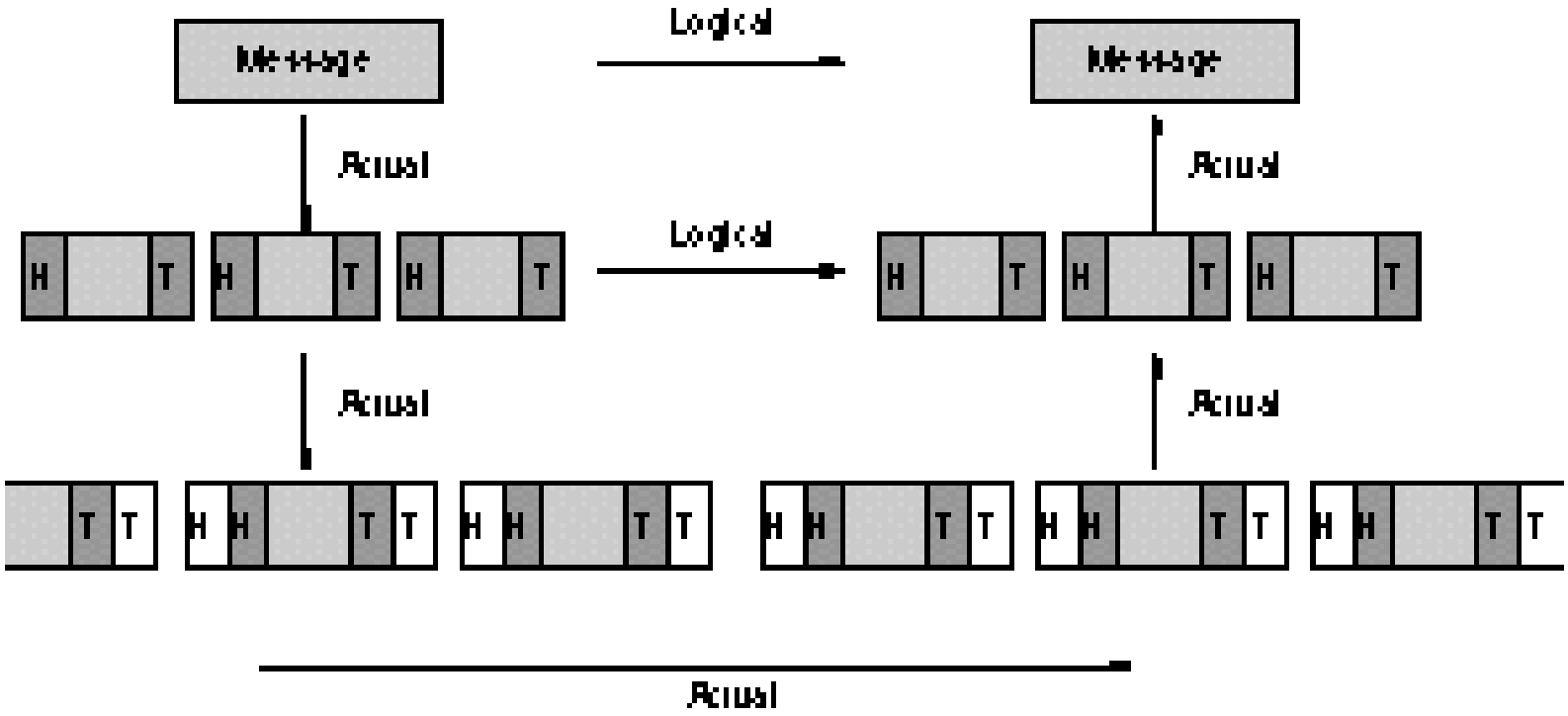
- **Internetworking: allows computers on independent and incompatible networks to communicate reliably and efficiently;**
  - **Enabling technologies: SW standards that allow reliable communications without reliable networks**
  - **Hierarchy of layers, giving each layer responsibility for portion of overall communications task, called protocol families or protocol suites**
- **Transmission Control Protocol/Internet Protocol (TCP/IP)**
  - **This protocol family is the basis of the Internet**
  - **IP makes best effort to deliver; TCP guarantees delivery**
  - **TCP/IP used even when communicating locally: NFS uses IP even though communicating across homogeneous LAN**

FTP From Stanford to Berkeley



- **BARRNet is WAN for Bay Area**
- **T1 is 1.5 mbps leased line; T3 is 45 mbps; FDDI is 100 mbps LAN**
- **IP sets up connection, TCP sends file**

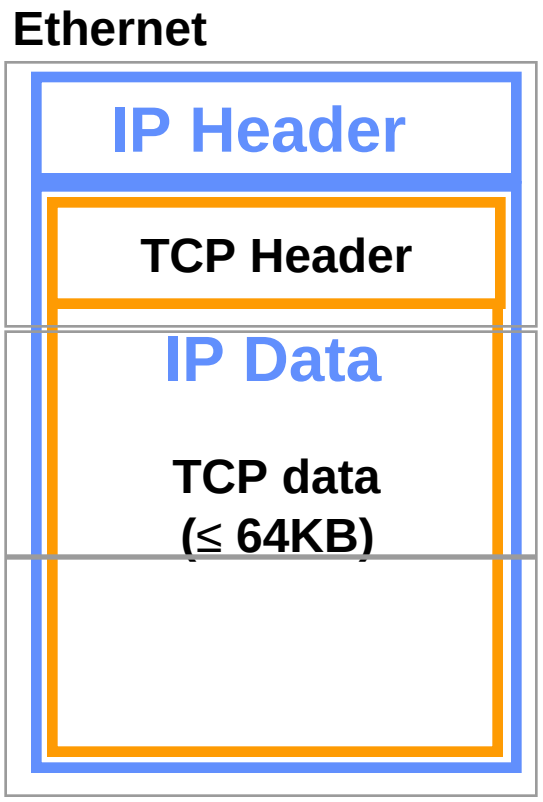
Protocol



- **Key to protocol families is that communication occurs logically at the same level of the protocol, called peer-to-peer, but is implemented via services at the lower level**
- **Danger is each level increases latency**

TCP/IP packet

- Application sends message
- TCP breaks into 64KB segements, adds 20B header
- IP adds 20B header, sends to network
- If Ethernet, broken into 1500B packets with headers, trailers



Example Networks

- **Ethernet: shared media 10 MB/s proposed in 1978, carrier sensing with expotential backoff on collision detect**
- **15 years with no improvement; higher BW?**
- **Multiple Ethernets with devices to allow Ehternets to operate in parallel!**
- **10 Mbit Ethernet successors?**
  - **FDDI: shared media**
  - **100 Mbit Ethernet (Fast Ethernet)**
  - **Switched Ethernet**
  - **ATM**

Connecting Networks

- **Bridges: connect LANs together, passing traffic from one side to another depending on the addresses in the packet.**
  - **operate at the Ethernet protocol level,**
  - **usually simpler and cheaper than routers.**
- **Routers or Gateways: these devices connect LANs to WANs or WANs to WANs and resolve incompatible addressing.**
  - **Generally slower than bridges, they operate at the internetworking protocol level.**
  - **Routers divide the interconnect into separate smaller subnets, which simplifies manageability and improves security.**

CS 252 Administrivia

- **Homework on Chapter 7 due Monday 11/4 at 5 PM in 252 box, done in pairs:**
  - **Exercises 7.1, 7.3, 7.10**
- **Wednesday Oct 30: surprise guest lecture by Internet Demi-God, Phil Karn of Qualcomm**
  - **Lecture Thursday Oct 31, 4:30-5:30 in Sibley Auditorium**
- **Next reading is Chapter 8 of CA:AQA 2/e and Chapter 1 of upcoming book by Culler, Singh, and Gupta (postscript available)**

`http://http.cs.berkeley.edu/~culler/book-alpha.html`

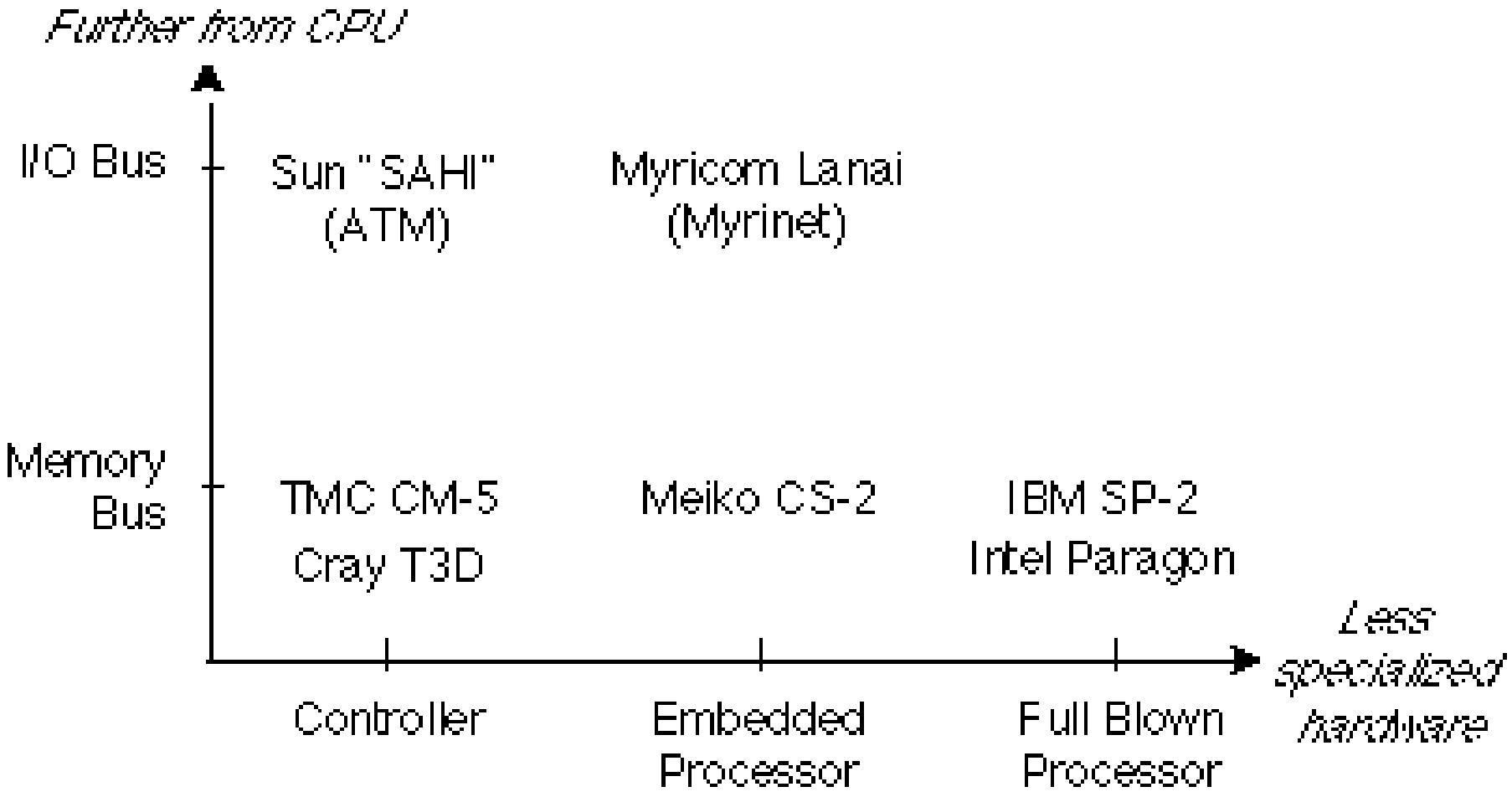
Example Networks

	MPP	LAN	WAN
	IBM SP-2	**100 Mb Ethernet**	**ATM**
Length (meters)	**10**	**200**	**100/1000**
Number data lines	**8**	**1**	**1**
Clock Rate	**40 MHz**	**100 MHz**	**155/622…**
Switch?	**Yes**	**No**	**Yes**
Nodes (N)	**≤512**	**≤254**	**≈10000**
Material	**copper**	**copper**	**copper/fiber**
Bisection BW (Mbit/s)	**320xNodes**	**100**	**155xNodes**
Peak Link BW (Mbits/s)	**320**	**100**	**155**
Measured Link BW	**284**	**--**	**80**

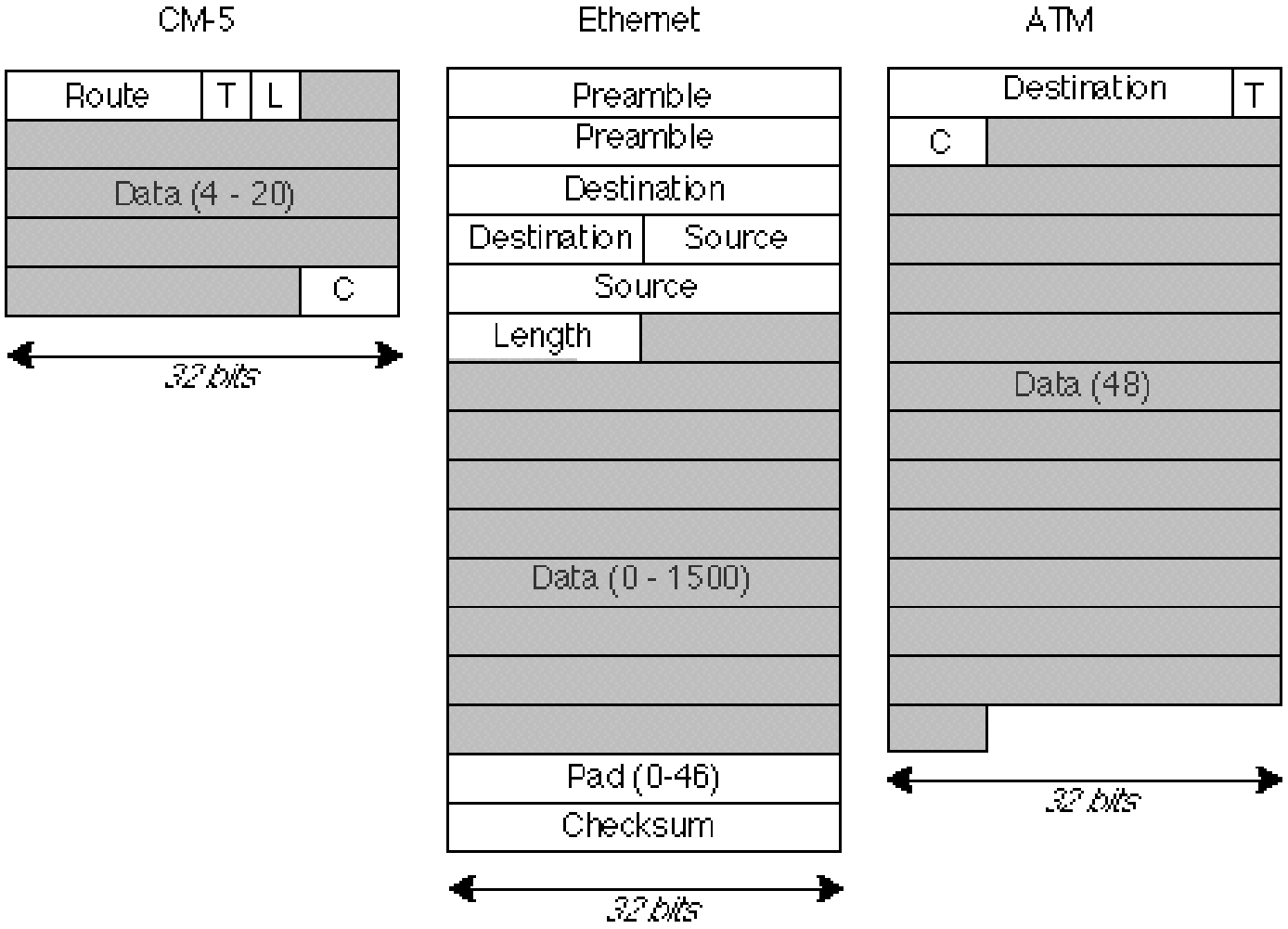
Example Networks (cont'd)

	MPP	LAN	WAN
	IBM SP-2	**100 Mb Ethernet**	**ATM**
Latency (μsecs)	**1**	**1.5**	**≈50**
Send+Receive Ovhd (μsecs)	**39**	**440**	**630**
Topology	**Fat tree**	**Line**	**Star**
Connectionless?	**Yes**	**Yes**	**No**
Store & Forward?	**No**	**No**	**Yes**
Congestion Control	**Back-pressure**	**Carrier Sense**	**Choke packets**
Standard	**No**	**Yes**	**Yes**
Fault Tolerance	**Yes**	**Yes**	**Yes**

Examples: Interface to Processor



Packet Formats



- **Fields: Destination, Checksum(C), Length(L), Type(T)**
- **Data/Header Sizes in bytes: (4 to 20)4, (0 to 1500)/26, 48/5**

Example Switched LAN Performance

Network Interface	*Switch*	*Link BW*
AMD Lance Ethernet	Baynetworks EtherCell 28115	10 Mb/s
Fore SBA-200 ATM	Fore ASX-200	155 MB/s
Myricom Myrinet	Myricom Myrinet	640 MB/s

- On SPARCstation-20 running Solaris 2.4 OS
- Myrinet is example of “System Area Network”: networks for a single room or floor: 25m limit
  - shorter => wider faster, less need for optical
  - short distance => source-based routing => simpler switches
  - Tandem also sponsoring SAN

Example Switched LAN Performance

Switch	*Switch Latency*
Baynetworks EtherCell 28115	52.0 µsecs
Fore ASX-200 ATM	13.0 µsecs
Myricom Myrinet	0.5 µsecs

- Measurements taken from “LogP Quantyified: The Case for Low-Overhead Local Area Networks”, K. Keeton, T. Anderson, D. Patterson, Hot Interconnects III, Stanford California, August 1995.

UDP/IP performance

Network	*UDP/IP roundtrip, N=8*	*Formula*
Bay. EtherCell	**1009 µsecs**	**+2.18*N**
Fore ASX-200 ATM	**1285 µsecs**	**+0.32*N**
Myricom Myrinet	**1443 µsecs**	**+0.36*N**

- **Formula from simple linear regression for tests from N = 8B to N = 8192B**
- **Software overhead not tuned for Fore, Myrinet; EtherCell using standard driver for Ethernet**

NFS performance

Network	*Avg. NFS response*	*LinkBW/E.*	*UDP/E.*
Bay. EtherCell	**14.5 ms**	**1**	**1.00**
Fore ASX-200 ATM	**11.8 ms**	**15**	**1.36**
Myricom Myrinet	**13.3 ms**	**64**	**1.43**

- **Last 2 columns show ratios of link bandwidth and UDP roundtrip times for 8B message to Ethernet**

Database performance

Network	*Avg. TPS*	*LinkBW/E.*	*TCP/E.*
Bay. EtherCell	**77 tps**	**1**	**1.00**
Fore ASX-200 ATM	**67 tps**	**15**	**1.47**
Myricom Myrinet	**66 tps**	**64**	**1.46**

- **Number of Transactions per Second (TPS) for DebitCredit Benchmark; front end to server with entire database in main memory (256 MB)**
  - **Each transaction => 4 messages via TCP/IP**
  - **DebitCredit Message sizes < 200 bytes**
- **Last 2 columns show ratios of link bandwidth and TCP/IP roundtrip times for 8B message to Ethernet**

Networking Summary

- **Protocols allow hetereogeneous networking**
- **Protocols allow operation in the presense of failures**
- **Internetworking protocols used as LAN protocols => large overhead for LAN**
- **Integrated circuit revolutionizing networks as well as processors**
- **Switch is a specialized computer**
- **Faster networks and slow overheads violate of Amdahl's Law**

Multiprocessors

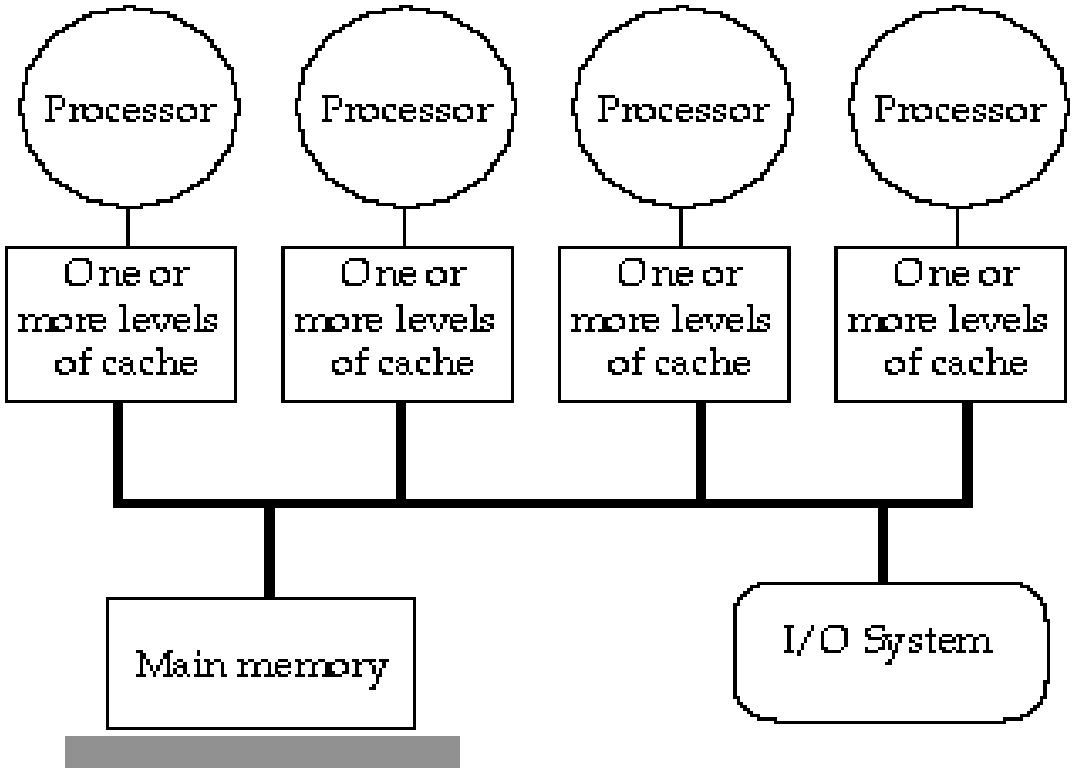
- The dream of computer architects for 30 years: replicate processors to add performance vs. design a faster processor
- Borders religious fervor at times: you must believe!
  - e.g., uniprocessors must stop getting faster due to limit of speed of light: 1972,…, 1989
- Fervor damped some when companies went out of business: Thinking Machines, Kendall Square, ...
- Multiprocessor success stories:
  - File servers, Database servers

Flynn Categories

- **SISD (Single Instruction Single Data)**
  - **Uniprocessors**
- **MISD (Multiple Instruction Single Data)**
  - **???**
- **SIMD (Single Instruction Multiple Data)**
  - **Examples: Illiac-IV, CM-2**
    - » **Simple programming model**
    - » **Low overhead**
    - » **Flexibility**
    - » **All custom**
- **MIMD (Multiple Instruction Multiple Data)**
  - **Examples: SPARCCenter, T3D**
    - » **Flexible**
    - » ***Use off-the-shelf micros***

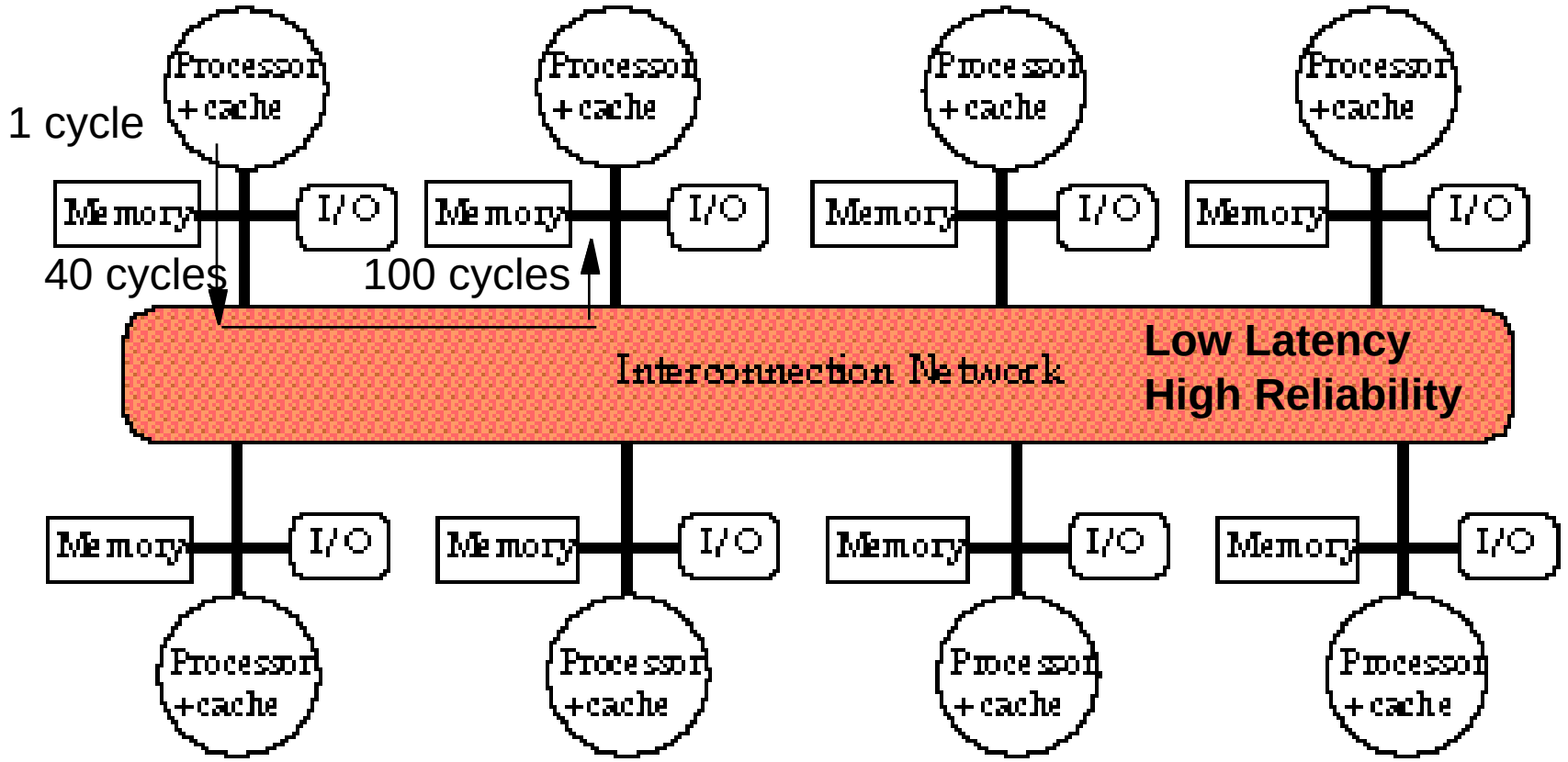
Small-Scale MIMD Designs

- **Memory: centralized with uniform access time ("uma") and bus interconnect**
- **Examples: SPARCCenter, Challenge, SystemPro**



Large-Scale MIMD Designs

- **Memory: distributed with nonuniform access time ("numa") and scalable interconnect (distributed memory)**
- **Examples: T3D, HP Exemplar, SGI Origin, CM-5**



Communication Models

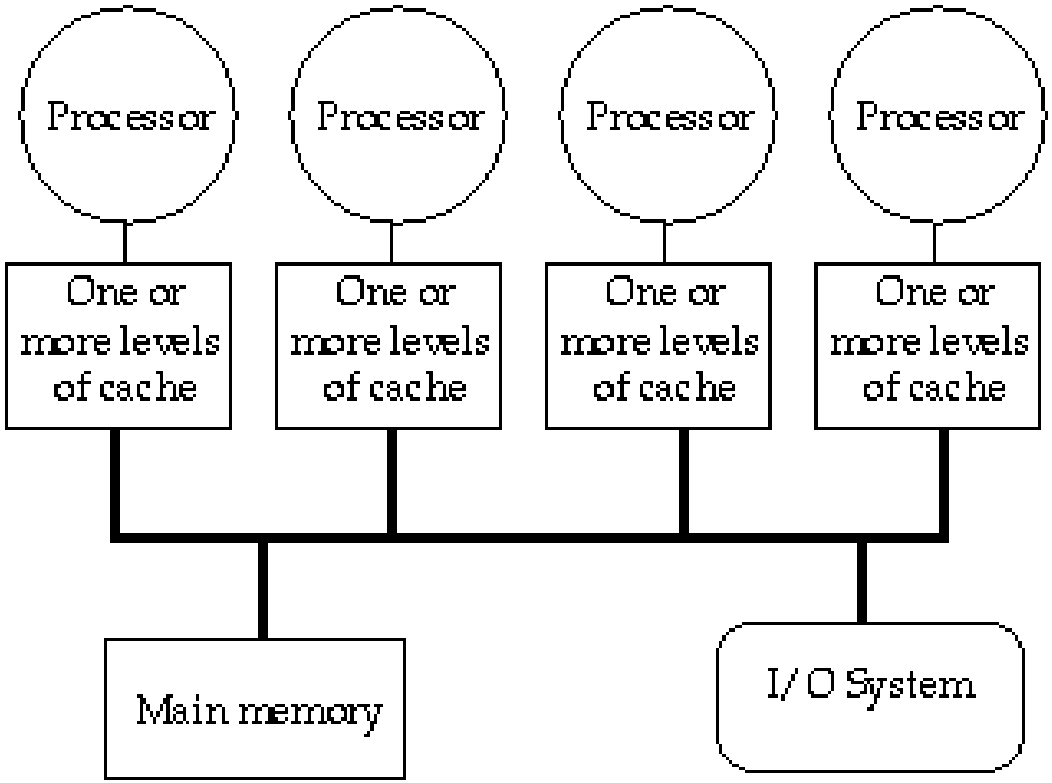
- **Shared Memory**
  - **Processors communicate with shared address space**
  - **Easy on small-scale machines**
  - **Advantages:**
    - » **Model of choice for uniprocessors, small-scale MPs**
    - » **Ease of programming**
    - » **Lower latency**
    - » **Easier to use hardware controlled caching**
- **Message passing**
  - **Processors have private memories, communicate via messages**
  - **Advantages:**
    - » **Less hardware, easier to design**
    - » **Focuses attention on costly non-local operations**
- **Can support either model on either HW base**

Important Communication Properties

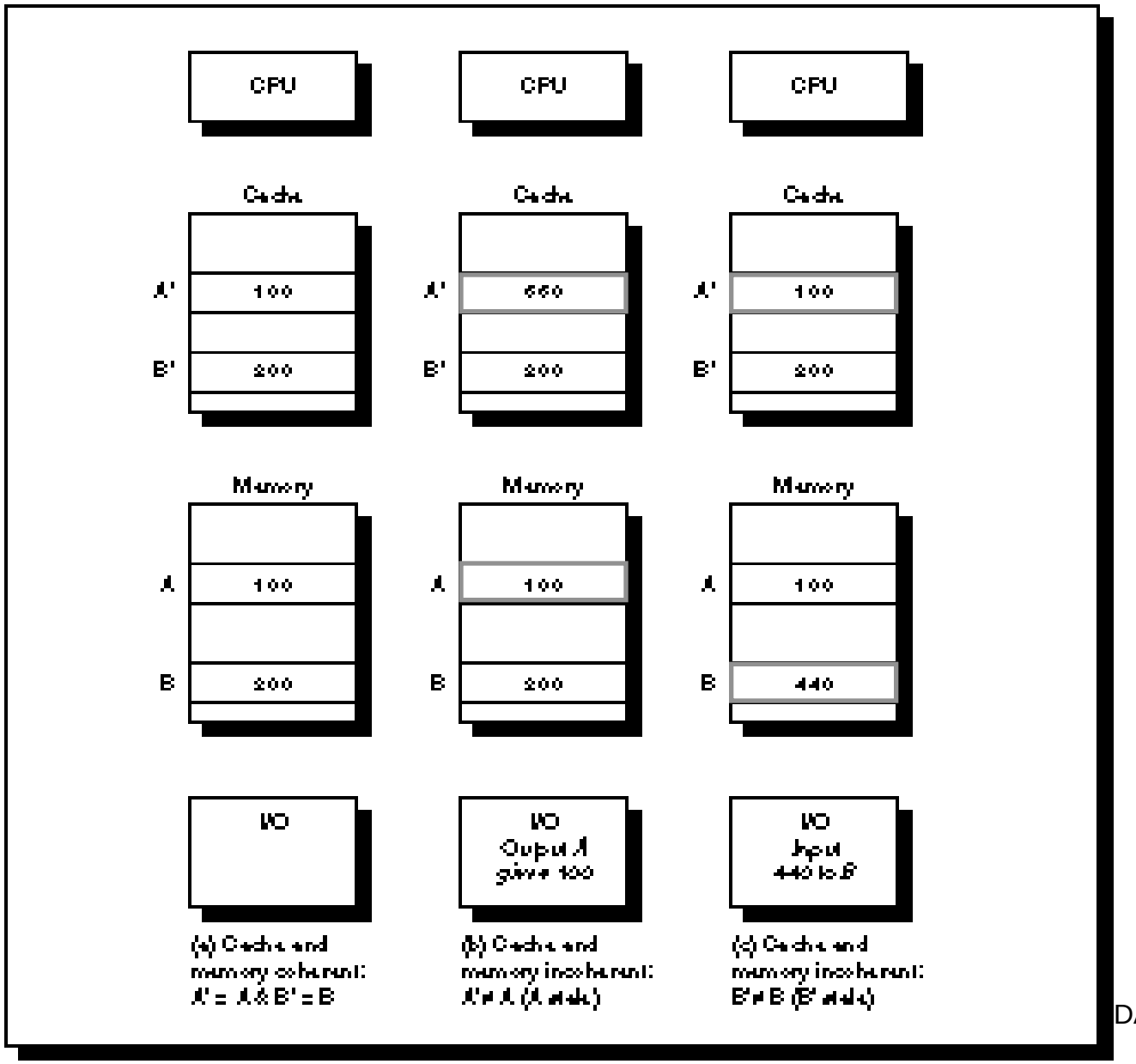
- **Bandwidth**
  - **Need high bandwidth in communication**
  - **Cannot scale, but stay close**
  - **Make limits in network, memory, and processor**
  - **Overhead to communicate is a problem in many machines**
- **Latency**
  - **Affects performance, since processor may have to wait**
  - **Affects ease of programming, since requires more thought to overlap communication and computation**
- **Latency Hiding**
  - **How can a mechanism help hide latency?**
  - **Examples: overlap message send with computation, prefetch**

Small-Scale—Shared Memory

- **Caches serve to:**
  - **Increase bandwidth versus bus/memory**
  - **Reduce latency of access**
  - **Valuable for both private data and shared data**
- **What about cache consistency?**



The Problem of Cache Coherency



What Does Coherency Mean?

- **Informally:**
  - **Any read must return the most recent write**
  - **Too strict and very difficult to implement**
- **Better:**
  - **Any write must eventually be seen by a read**
  - **All writes are seen in order ("serialization")**
- **Two rules to ensure this:**
  - **If P writes x and P1 reads it, P's write will be seen if the read and write are sufficiently far apart**
  - **Writes to a single location are serialized: seen in one order**
    - » **Latest write will be seen**
    - » **Otherwise could see writes in illogical order (could see older value after a newer value)**

Potential Solutions

- **Snooping Solution (Snoopy Bus):**
  - **Send all requests for data to all processors**
  - **Processors snoop to see if they have a copy and respond accordingly**
  - **Requires broadcast, since caching information is at processors**
  - **Works well with bus (natural broadcast medium)**
  - **Dominates for small scale machines (most of the market)**
- **Directory-Based Schemes**
  - **Keep track of what is being shared in one centralized place**
  - **Distributed memory => distributed directory (avoids bottlenecks)**
  - **Send point-to-point requests to processors**
  - **Scales better than Snoop**
  - **Actually existed BEFORE Snoop-based schemes**

Basic Snoopy Protocols

- **Write Invalidate Protocol:**
  - **Multiple readers, single writer**
  - **Write to shared data: an invalidate is sent to all caches which snoop and *invalidate* any copies**
  - **Read Miss:**
    - » **Write-through: memory is always up-to-date**
    - » **Write-back: snoop in caches to find most recent copy**
- **Write Broadcast Protocol:**
  - **Write to shared data: broadcast on bus, processors snoop, and *update* copies**
  - **Read miss: memory is always up-to-date**
- **Write serialization: bus serializes requests**
  - **Bus is single point of arbitration**

Basic Snoopy Protocols

- **Write Invalidate versus Broadcast:**
  - **Invalidate requires one transaction per write-run**
  - **Invalidate uses spatial locality: one transaction per block**
  - **Broadcast has lower latency between write and read**
  - **Broadcast: BW (increased) vs. latency (decreased)**

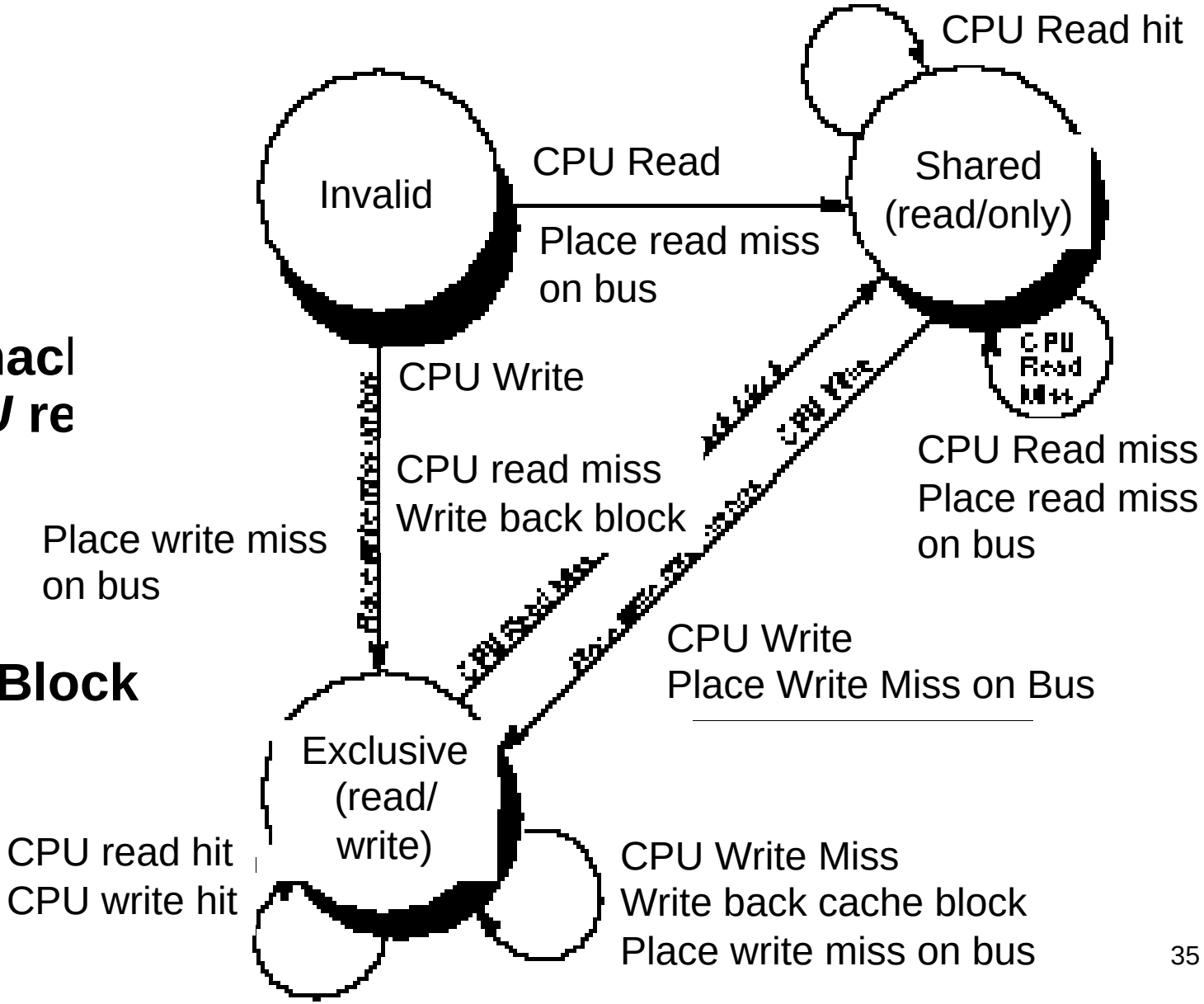
Name	Protocol Type	Memory-write policy	Machines using
Write Once	Write invalidate	Write back after first write	First snoopy protocol.
Synapse N+1	Write invalidate	Write back	1st cache-coherent MPs
Berkeley	Write invalidate	Write back	Berkeley SPUR
Illinois	Write invalidate	Write back	SGI Power and Challenge
"Firefly"	Write broadcast	Write back private, Write through shared	SPARCCenter 2000

An Example Snoopy Protocol

- Invalidation protocol, write-back cache
- Each block of memory is in one state:
  - Clean in all caches and up-to-date in memory
  - OR Dirty in exactly one cache
  - OR Not in any caches
- Each cache block is in one state:
  - Shared: block can be read
  - OR Exclusive: cache has only copy, its writeable, and dirty
  - OR Invalid: block contains no data
- Read misses: cause all caches to snoop
- Writes to clean line are treated as misses

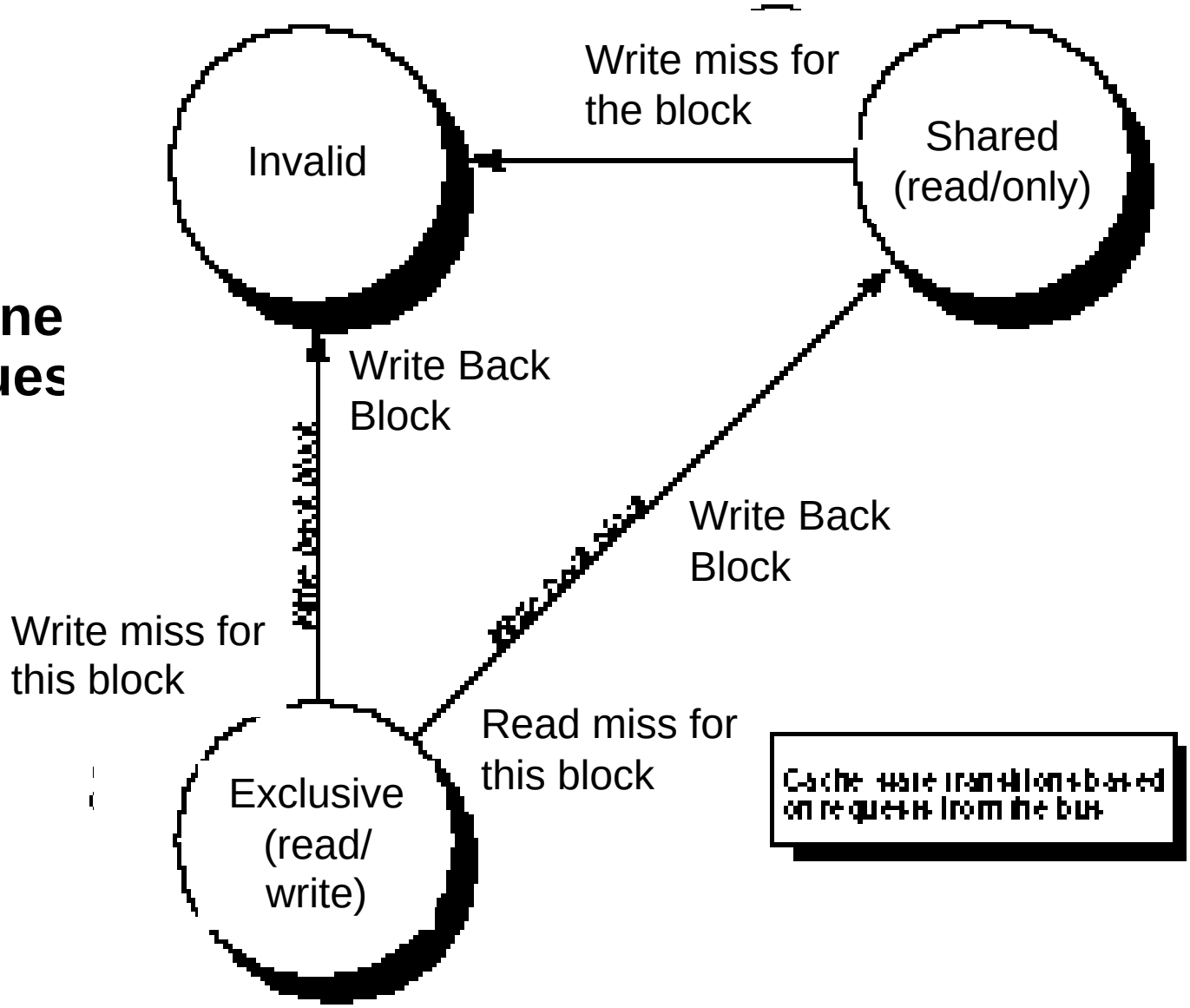
Snoopy-Cache State Machine-I

- **State mac**
  for *CPU* re

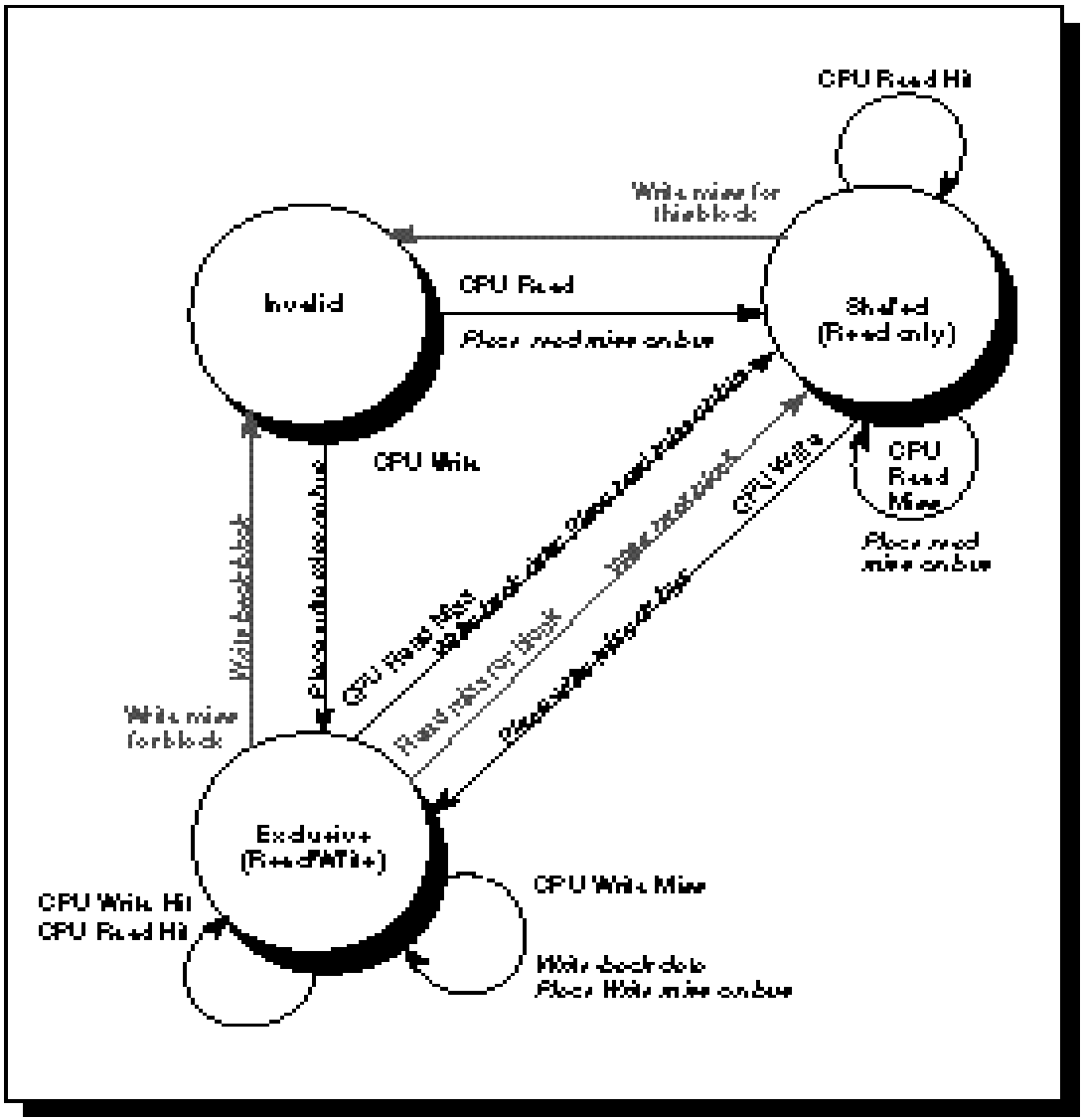


Snoopy-Cache State Machine-II

- **State machine for *bus* reques**



Snoop Cache: State Machine



Extensions:

- Fourth State: Ownership
- Clean-> dirty, need invalidate only (upgrade request) Berkeley Protocol
- Clean exclusive state (no miss for private data on write) Illinois Protocol

Example

	P1			P2			Bus				Memory	
step	State	Addr	Value	State	Addr	Value	Action	Proc.	Addr	Value	Addr	Value
P1: Write 10 to A1												
P1: Read A1												
P2: Read A1												
P2: Write 20 to A1												
P2: Write 40 to A2												

Assumes A1 and A2 map to same cache block

Example

	P1			P2			Bus				Memory	
step	State	Addr	Value	State	Addr	Value	Action	Proc.	Addr	Value	Addr	Value
P1: Write 10 to A1	Excl.	A1	10				WrMs	P1	A1			
P1: Read A1												
P2: Read A1												
P2: Write 20 to A1												
P2: Write 40 to A2												

Assumes A1 and A2 map to same cache block

Example

	P1			P2			Bus				Memory	
step	State	Addr	Value	State	Addr	Value	Action	Proc.	Addr	Value	Addr	Value
P1: Write 10 to A1	Excl.	A1	10				WrMs	P1	A1			
P1: Read A1	Excl.	A1	10									
P2: Read A1												
P2: Write 20 to A1												
P2: Write 40 to A2												

Assumes A1 and A2 map to same cache block

Example

	P1			P2			Bus				Memory	
step	State	Addr	Value	State	Addr	Value	Action	Proc.	Addr	Value	Addr	Value
P1: Write 10 to A1	Excl.	A1	10				WrMs	P1	A1			
P1: Read A1	Excl.	A1	10									
P2: Read A1				Shar.	A1		RdMs	P2	A1			
	Shar.	A1	10				WrBk	P1	A1	10		10
				Shar.	A1	10	RdDa	P2	A1	10		10
P2: Write 20 to A1												10
P2: Write 40 to A2												10
												10

Assumes A1 and A2 map to same cache block

Example

	P1			P2			Bus				Memory	
step	State	Addr	Value	State	Addr	Value	Action	Proc.	Addr	Value	Addr	Value
P1: Write 10 to A1	Excl.	A1	10				WrMs	P1	A1			
P1: Read A1	Excl.	A1	10									
P2: Read A1				Shar.	A1		RdMs	P2	A1			
	Shar.	A1	10				WrBk	P1	A1	10		10
				Shar.	A1	10	RdDa	P2	A1	10		10
P2: Write 20 to A1	Inv.			Excl.	A1	20	WrMs	P2	A1			10
P2: Write 40 to A2												10
												10

Assumes A1 and A2 map to same cache block

Example

	P1			P2			Bus				Memory	
step	State	Addr	Value	State	Addr	Value	Action	Proc.	Addr	Value	Addr	Value
P1: Write 10 to A1	Excl.	A1	10				WrMs	P1	A1			
P1: Read A1	Excl.	A1	10									
P2: Read A1				Shar.	A1		RdMs	P2	A1			
	Shar.	A1	10				WrBk	P1	A1	10		10
				Shar.	A1	10	RdDa	P2	A1	10		10
P2: Write 20 to A1	Inv.			Excl.	A1	20	WrMs	P2	A1			10
P2: Write 40 to A2							WrMs	P2	A2			10
				Excl.	A2	40	WrBk	P2	A1	20		20

Assumes A1 and A2 map to same cache block

Implementation Complications

- **Write Races:**
  - **Cannot update cache until bus is obtained**
    - » **Otherwise, another processor may get bus first, and write the same cache block**
  - **Two step process:**
    - » **Arbitrate for bus**
    - » **Place miss on bus and complete operation**
  - **If miss occurs to block while waiting for bus, handle miss (invalidate may be needed) and then restart.**
  - **Split transaction bus:**
    - » **Bus transaction is not atomic: can have multiple outstanding transactions for a block**
    - » **Multiple misses can interleave, allowing two caches to grab block in the Exclusive state**
    - » **Must track and prevent multiple misses for one block**
- **Must support interventions and invalidations**