Multiprocessors

Instructor: Dmitri A. Gusev

Fall 2007

CS 502: Computers and Communications Technology

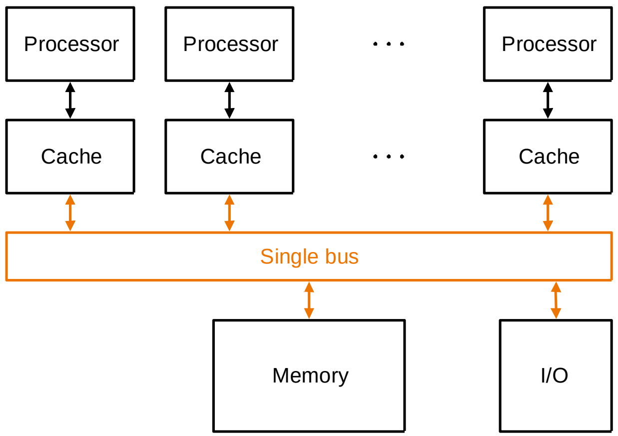
Lecture 13, October 17, 2007

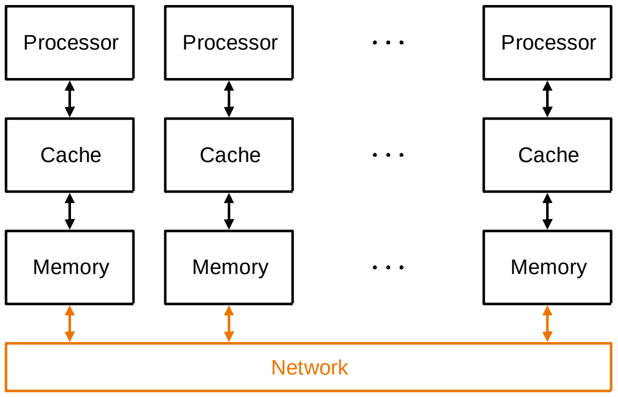
Multiprocessors

- Idea: create powerful computers by connecting many smaller ones

good news: works for timesharing (better than supercomputer)

bad news: its really hard to write good concurrent programs
many commercial failures





Questions

- How do parallel processors share data?
  — single address space (SMP vs. NUMA)
  — message passing

- How do parallel processors coordinate?
  — synchronization (locks, semaphores)
  — built into send / receive primitives
  — operating system protocols

- How are they implemented?
  — connected by a single bus
  — connected by a network

Programming multiprocessors

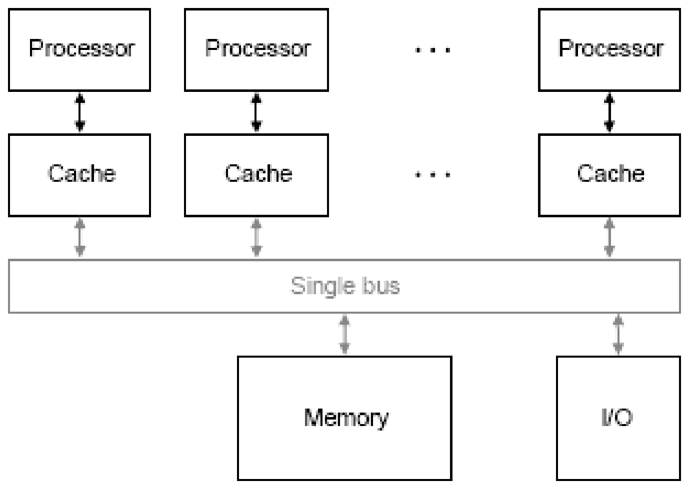
- Multiprogramming is difficult:

  - Communication problems

  - Requires knowledge about the hardware

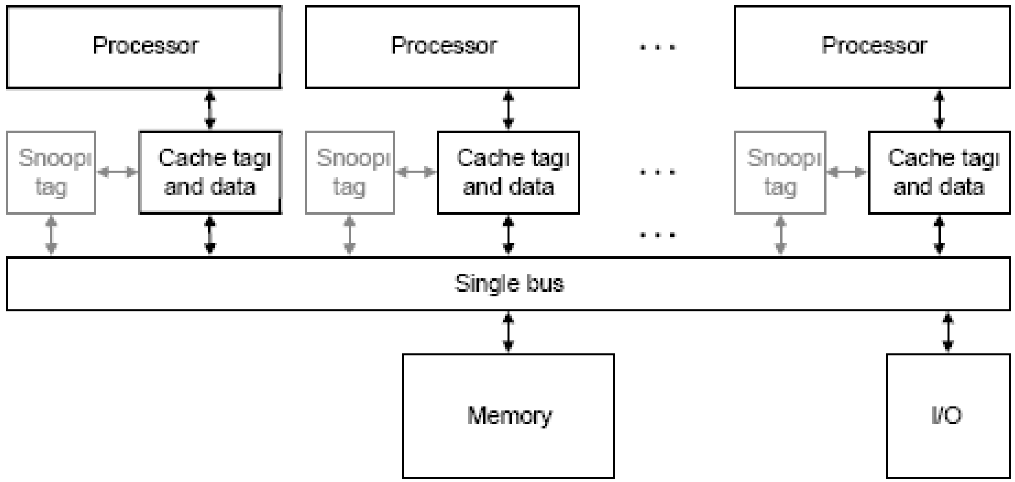
  - All parts of the program should be parallelized

Multiprocessors connected by a single bus



- Each processor is smaller than a multichip processor
- The use of caches can reduce the bus traffic
- There exists mechanisms to keep caches and memory consistent

Multiprocessor cache coherency



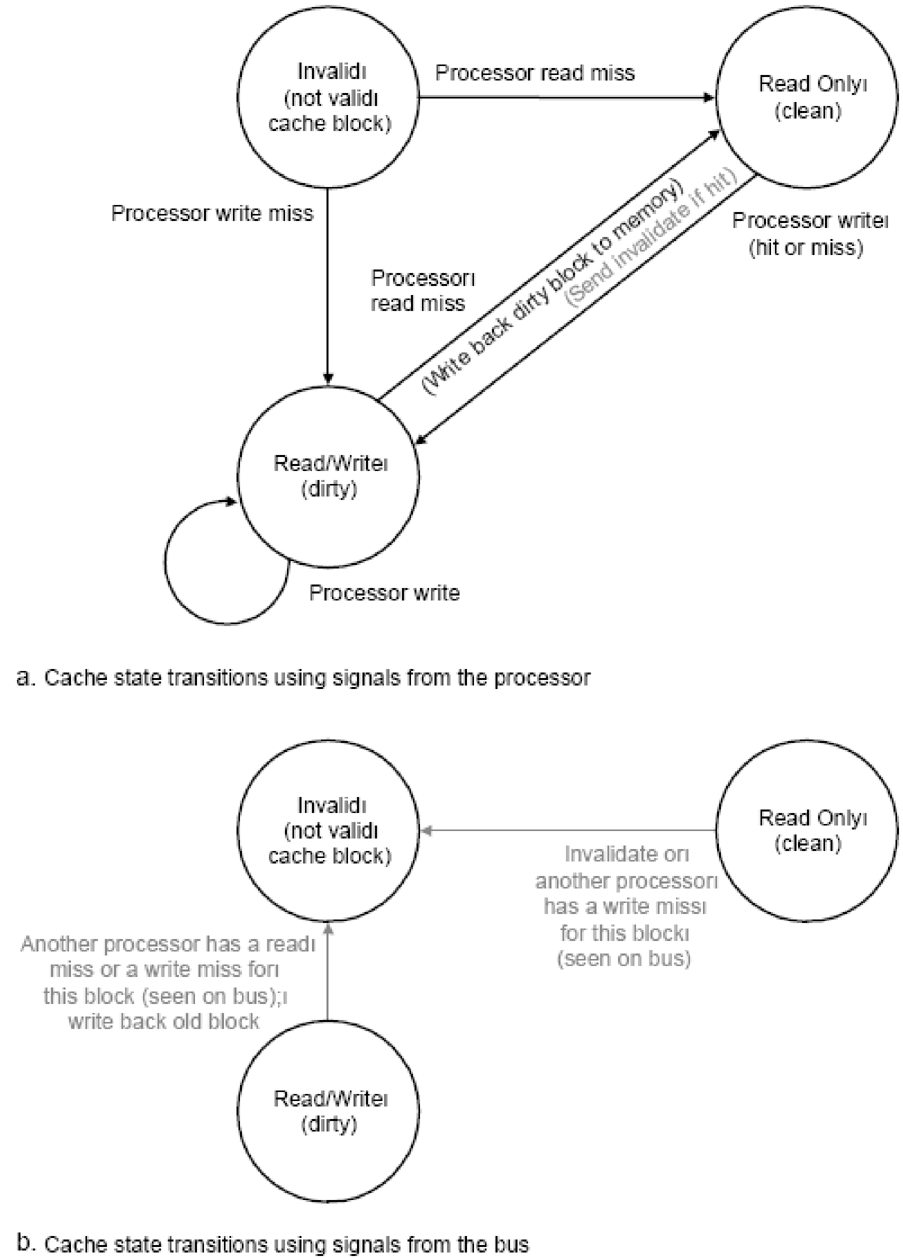
Snooping (monitoring) protocols: locate all caches that share a block to be written. Then:

- *Write-invalidate*: The writing processor causes all copies in other caches to be invalidated before changing its local copy. Similar to write-back.
- *Write-update (broadcast)*: The write processor sends the new data (the word) over the bus. Similar to write-through.
- The role of the size of the block (broadcasting only a word, false sharing).

Write-invalidate cache coherency protocol based on a write-back policy

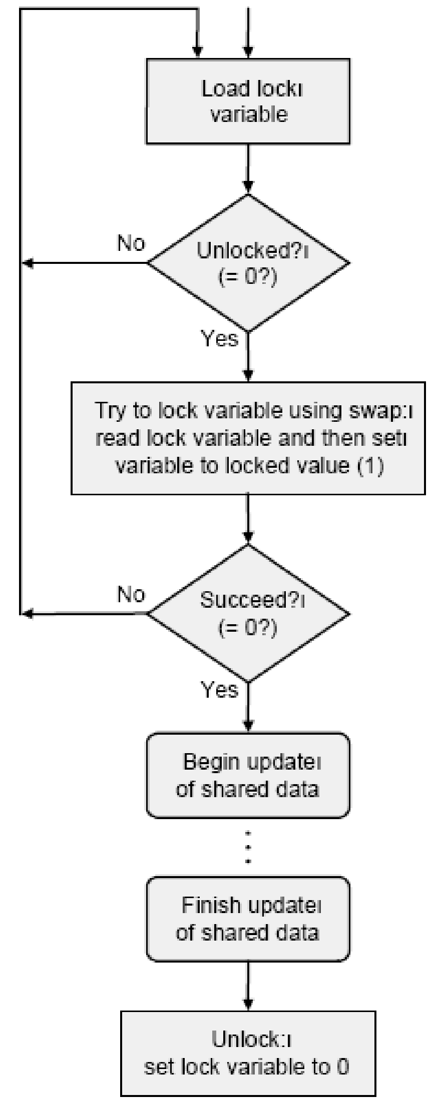
Each cache block is in one of the following states:

- *Read only*: the block is not wtitten and may be shared
- *Read/Write*: the block is written (dirty) and may not be shared
- *Invalid*: the block does not have valid data



Synchronization using coherency

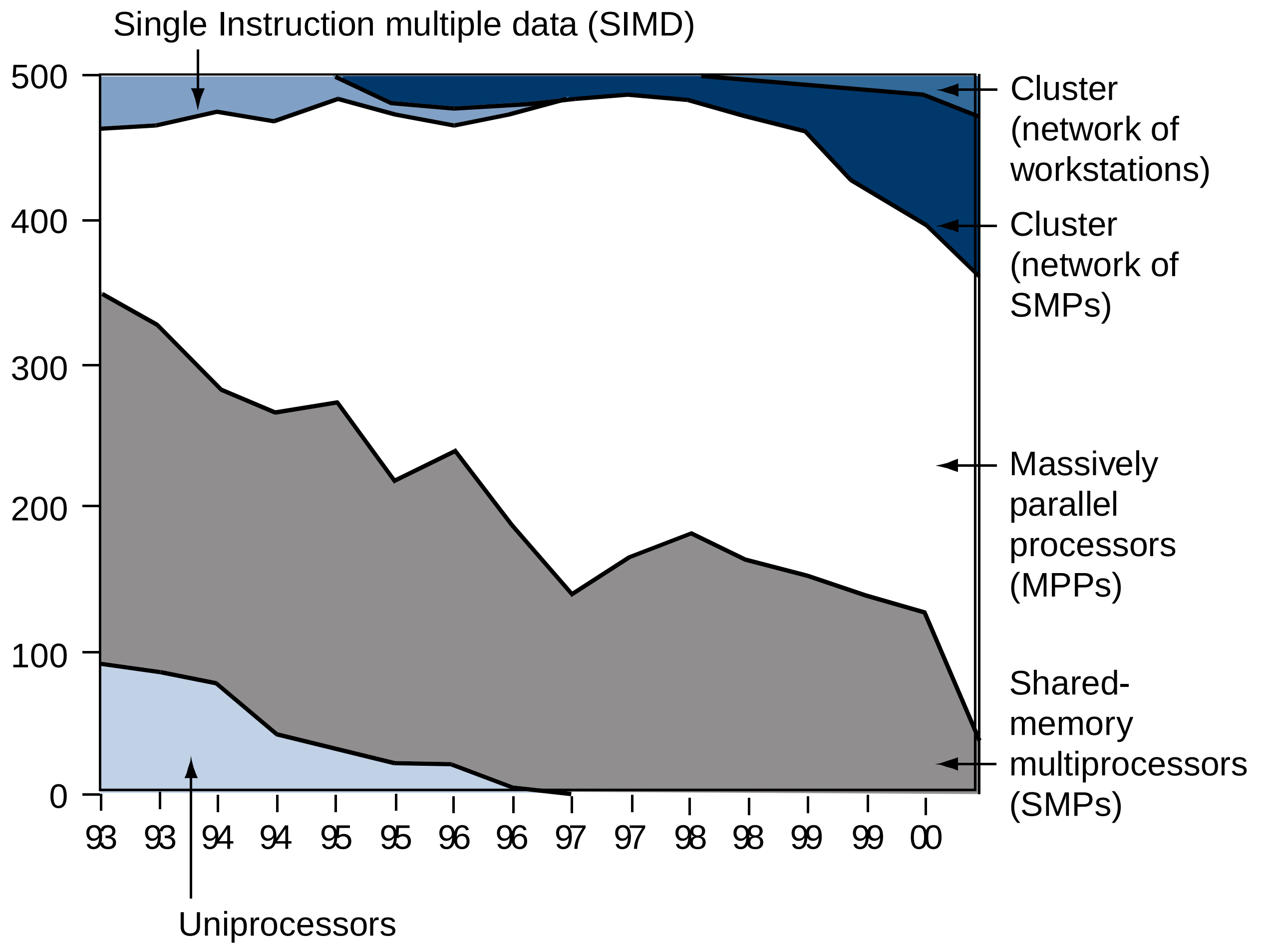
- Using locks (semaphores)
- Atomic swap operation



Step	P1	P2	P3	Bus activity
1	Has lock	Spins, testing lock=0	Spins, testing lock=0	none
2	Sets lock to 0 and 0 sent over bus	Spins, testing lock=0	Spins, testing lock=0	Write-invalidate of lock variable sent from P0
3		Cache miss	Cache miss	Bus decides to service P2 cache miss
4		Waits (bus busy)	Lock=0	Cache miss for P2 satisfied
5		Lock =0	Swap: reads locks and sets to 1	Cache miss for P1 satisfied
6		Swap: reads locks and sets to 1	Value from swap=0 and 1 sent over bus	Write-invalidate of lock variable sent from P2
7		Value from swap=1 and 1 sent over bus	Owns the locks and updates the shared data	Write-invalidate of lock variable sent from P1
8		Spins, testing lock=0		None

Supercomputers

Plot of top 500 supercomputer sites over a decade:



Using multiple processors an old idea

- Some SIMD designs:

Institution	Name	Maximum no. of proc.	Bits/ proc.	Proc. clock rate (MHz)	Number of FPUs	Maximum memory size/system (MB)	Communi-cations BW/system (MB/sec)	Year
U. Illinois	Illiac IV	64	64	13	64	1	2,560	1972
ICL	DAP	4,096	1	5	0	2	2,560	1980
Goodyear	MPP	16,384	1	10	0	2	20,480	1982
Thinking Machines	CM-2	65,536	1	7	2048 (optional)	512	16,384	1987
Maspar	MP-1216	16,384	4	25	0	256 or 1024	23,000	1989

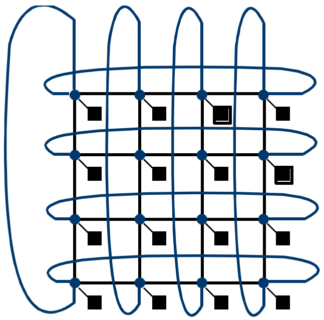
FIGURE 9.11.1 Characteristics of five SIMD computers. Number of FPUs means number of floating-point units.

- Costs for the the Illiac IV escalated from \$8 million in 1966 to \$32 million in 1972 despite completion of only ¼ of the machine. It took three more years before it was operational!

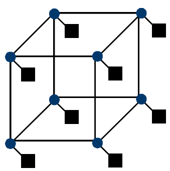
“For better or worse, computer architects are not easily discouraged”

Lots of interesting designs and ideas, lots of failures, few successes

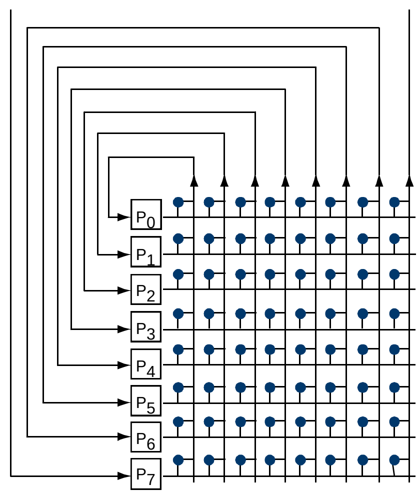
Topologies



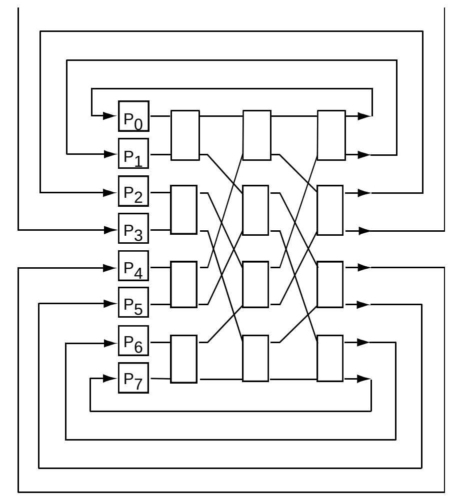
a. 2-D grid or mesh of 16 nodes



b. n-cube tree of 8 nodes (8 = 2^3 so n = 3)



a. Crossbar



b. Omega network

Clusters

- Constructed from whole computers
- Independent, scalable networks
- Strengths:
  - Many applications amenable to loosely coupled machines
  - Exploit local area networks
  - Cost effective / Easy to expand
- Weaknesses:
  - Administration costs not necessarily lower
  - Connected using I/O bus
- Highly available due to separation of memories
- In theory, we should be able to do better

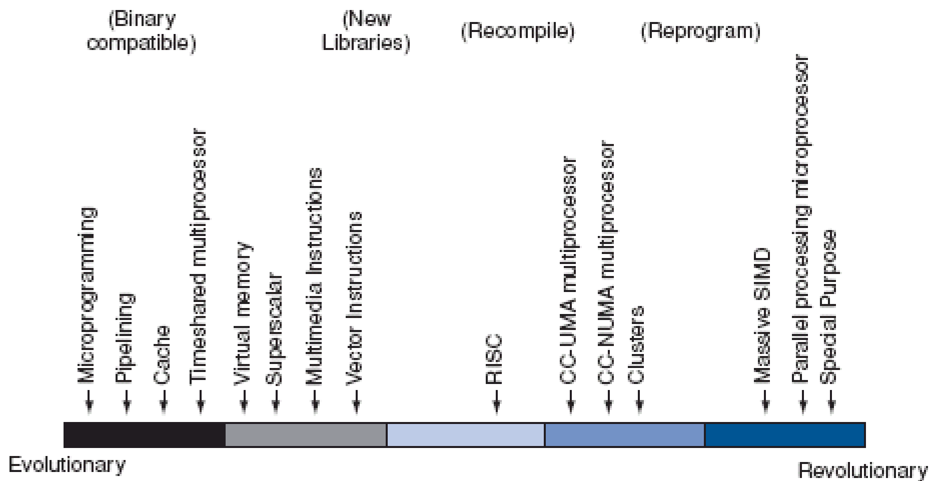
Google

- Google uses thousands of processors and disks to handle thousands of queries per second

Concluding Remarks

- Evolution vs. Revolution

“More often the expense of innovation comes from being too disruptive to computer users”



“Acceptance of hardware ideas requires acceptance by software people; therefore hardware people should learn about software. And if software people want good machines, they must learn more about hardware to be able to communicate with and thereby influence hardware engineers.”