

# Parallel Architecture

Sathish Vadhiyar

Department of Computational and Data Sciences  
Supercomputer Education and Research Centre  
Indian Institute of Science, Bangalore, India



# Motivations of Parallel Computing

- Faster execution times
  - From days or months to hours or seconds
  - E.g., climate modelling, bioinformatics
- Large amount of data dictate parallelism
- Parallelism more natural for certain kinds of problems, e.g., climate modelling
- Due to computer architecture trends
  - CPU speeds have saturated
  - Slow memory bandwidths

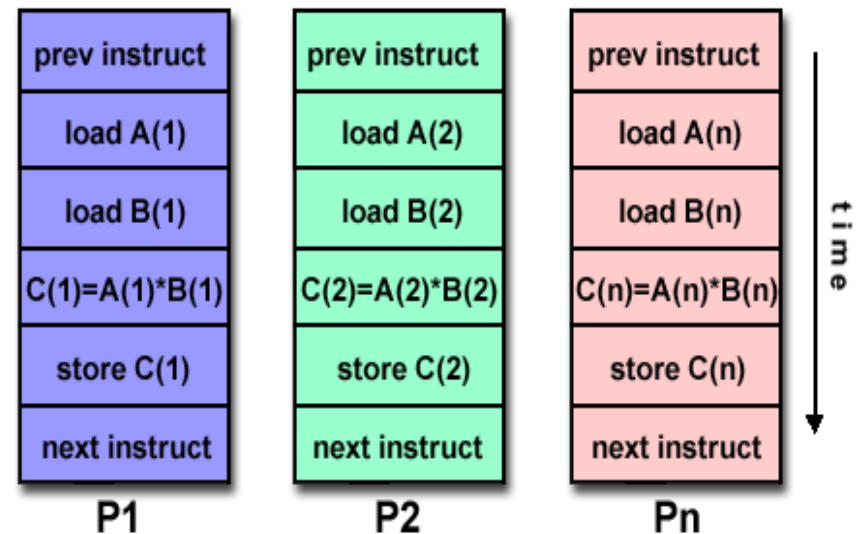


# PARALLEL ARCHITECTURES

# Classification of Architectures – Flynn's classification

In terms of parallelism in instruction and data stream

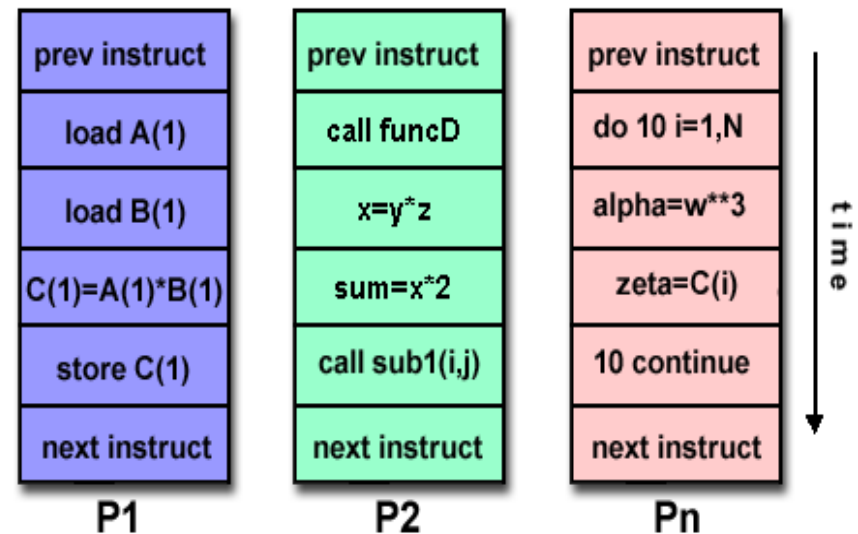
- Single Instruction Single Data (SISD): Serial Computers
- Single Instruction Multiple Data (SIMD)
  - Vector processors and processor arrays
  - Examples: CM-2, Cray-90, Cray YMP, Hitachi 3600



Courtesy: [http://www.llnl.gov/computing/tutorials/parallel\\_comp/](http://www.llnl.gov/computing/tutorials/parallel_comp/)

# Classification of Architectures – Flynn's classification

- Multiple Instruction Single Data (MISD): Not popular
- Multiple Instruction Multiple Data (MIMD)
  - Most popular
  - IBM SP and most other supercomputers, clusters, computational Grids etc.

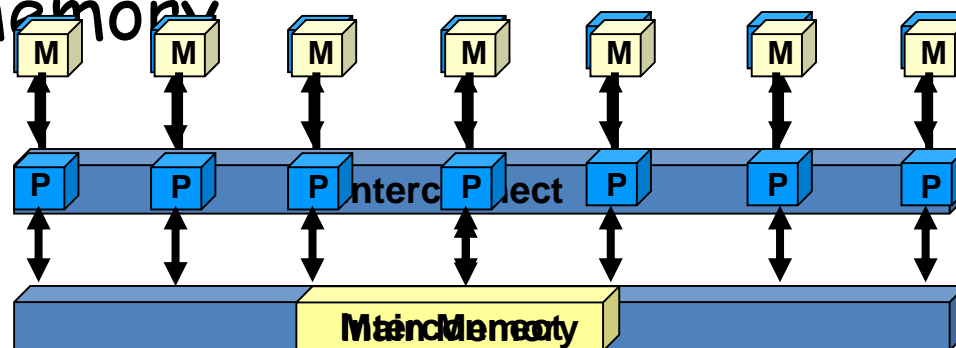


Courtesy: [http://www.llnl.gov/computing/tutorials/parallel\\_comp/](http://www.llnl.gov/computing/tutorials/parallel_comp/)

# Classification 2:

## Shared Memory vs Message Passing

- **Shared memory machine:** The  $n$  processors share physical address space
  - Communication can be done through this shared memory



- The alternative is sometimes referred to as a **message passing machine** or a **distributed memory machine**

# Shared Memory Machines

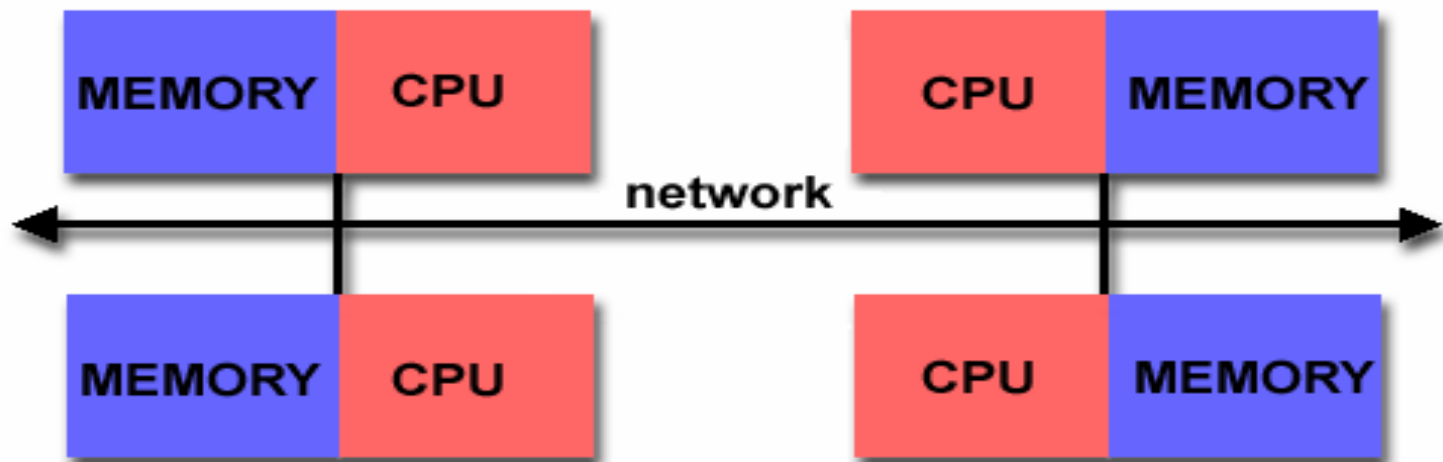
The shared memory could itself be distributed among the processor nodes

- Each processor might have some portion of the shared physical address space that is physically close to it and therefore accessible in less time
- Terms: NUMA vs UMA architecture
  - Non-Uniform Memory Access
  - Uniform Memory Access



# Classification of Architectures – Based on Memory

- Distributed memory



Courtesy: [http://www.llnl.gov/computing/tutorials/parallel\\_comp/](http://www.llnl.gov/computing/tutorials/parallel_comp/)

- Multi-cores and Many-cores



# INTERCONNECTION NETWORKS

# Interconnects

- Used in both shared memory and distributed memory architectures
- In shared memory: Used to connect processors to memory
- In distributed memory: Used to connect different processors
- Components
  - *Interface* (PCI or PCI-e): for connecting processor to network link
  - *Network link* connected to a communication network (network of connections)



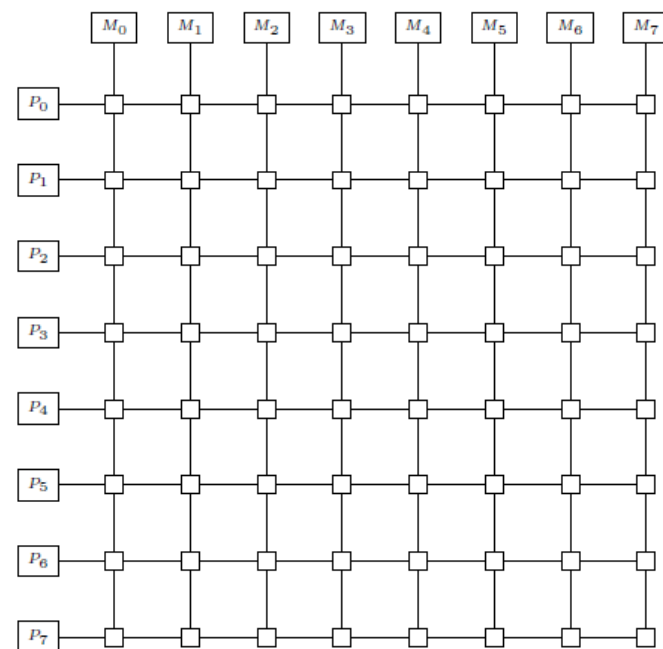
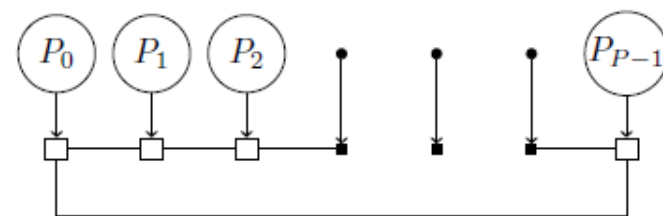
# Communication network

- Consists of *switching elements* to which processors are connected through *ports*
- *Switch*: network of switching elements
- Switching elements connected with each other using a pattern of connections
- Pattern defines the *network topology*
- In shared memory systems, memory units are also connected to communication network



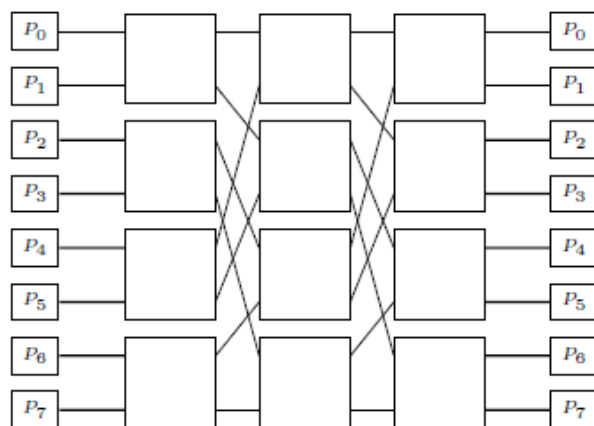
# Network Topologies

- Bus, ring – used in small-scale shared memory systems
- Crossbar switch – used in some small-scale shared memory machines, small or medium-scale distributed memory machines



# Multistage network – Omega network

- To reduce switching complexity
- Omega network – consisting of  $\log P$  stages, each consisting of  $P/2$  switching elements



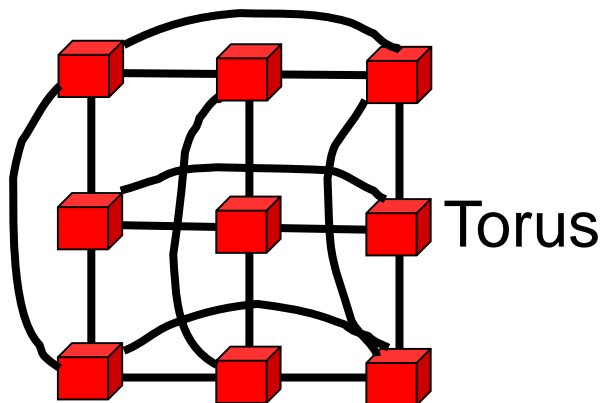
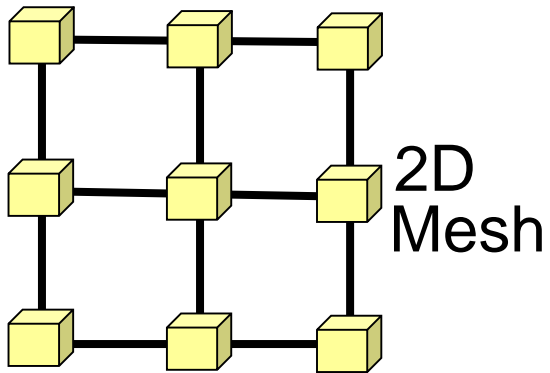
- Contention
  - In crossbar – nonblocking
  - In Omega – can occur during multiple communications to disjoint pairs

# Mesh, Torus, Hypercubes, Fat-tree

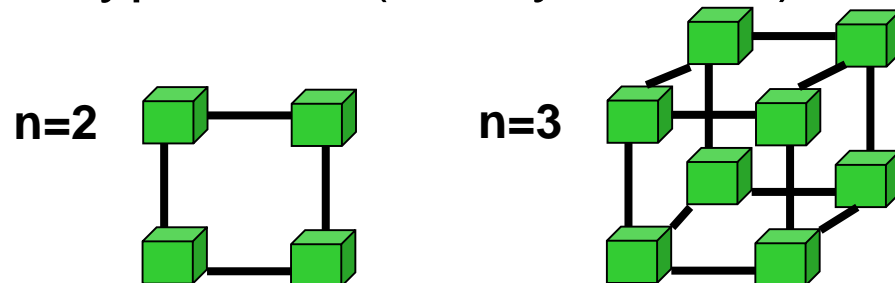
- Commonly used network topologies in distributed memory architectures
- Hypercubes are networks with dimensions



# Mesh, Torus, Hypercubes

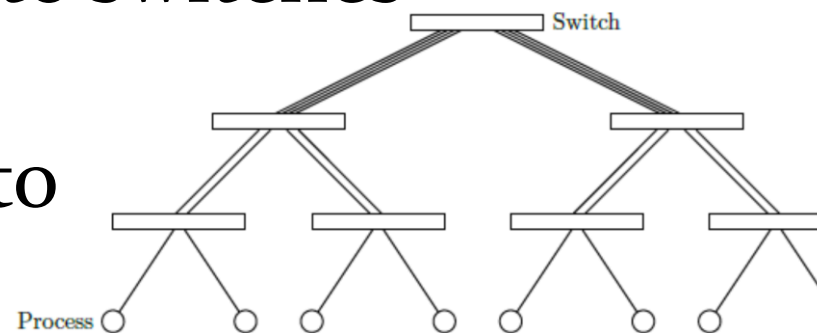


Hypercube (binary n-cube)



# Fat Tree Networks

- Binary tree
- Processors arranged in leaves
- Other nodes correspond to switches
- Fundamental property:  
No. of links from a node to a children = no. of links from the node to its parent
- Edges become fatter as we traverse up the tree





# Evaluating Interconnection topologies

- Diameter – maximum distance between any two processing nodes
  - Full-connected – 1
  - Star – 2
  - Ring –  $p/2$
  - Hypercube –  $\log P$
- Connectivity – multiplicity of paths between 2 nodes. Minimum number of arcs to be removed from network to break it into two disconnected networks
  - Linear-array – 1
  - Ring – 2
  - 2-d mesh – 2
  - 2-d mesh with wraparound – 4
  - D-dimension hypercubes –  $d$



# Evaluating Interconnection topologies

- bisection width – minimum number of links to be removed from network to partition it into 2 equal halves
  - Ring –  $\sqrt{P}$
  - P-node 2-D mesh –  $\sqrt{P}$
  - Tree –  $\log_2 P$
  - Star –  $P/2$
  - Completely connected –  $P(P-1)/2$
  - Hypercubes –  $\log_2 P$



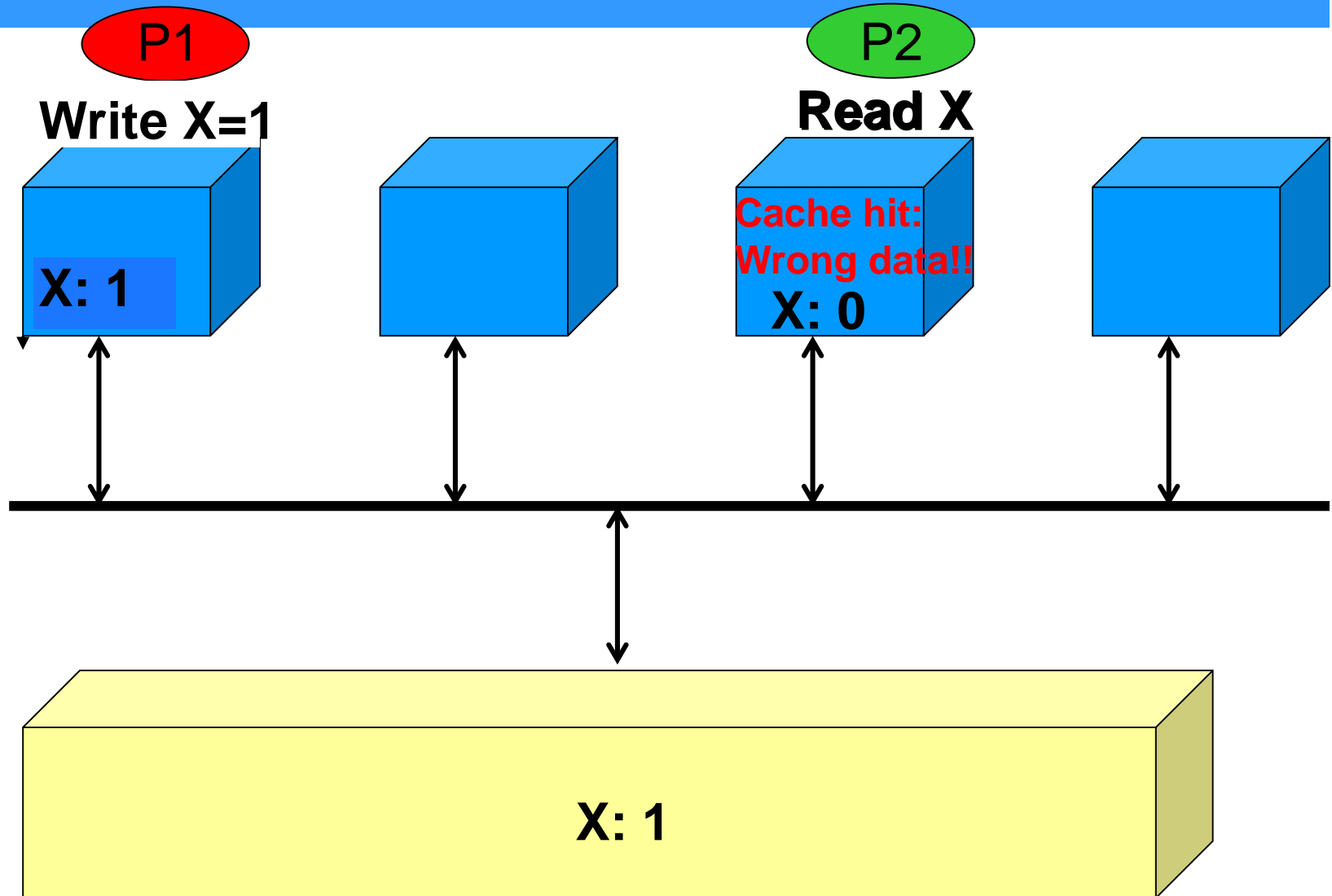
# Evaluating Interconnection topologies

- channel width - number of bits that can be simultaneously communicated over a link, i.e. number of physical wires between 2 nodes
- channel rate - performance of a single physical wire
- channel bandwidth - channel rate times channel width
- bisection bandwidth - maximum volume of communication between two halves of network, i.e. bisection width times channel bandwidth



# SHARED MEMORY AND CACHES

# Shared Memory Architecture: Caches



# Cache Coherence Problem

- If each processor in a shared memory multiple processor machine has a data cache
  - Potential data consistency problem: the cache coherence problem
  - Shared variable modification, private cache
- Objective: processes shouldn't read 'stale' data
- Solutions
  - Hardware: cache coherence mechanisms

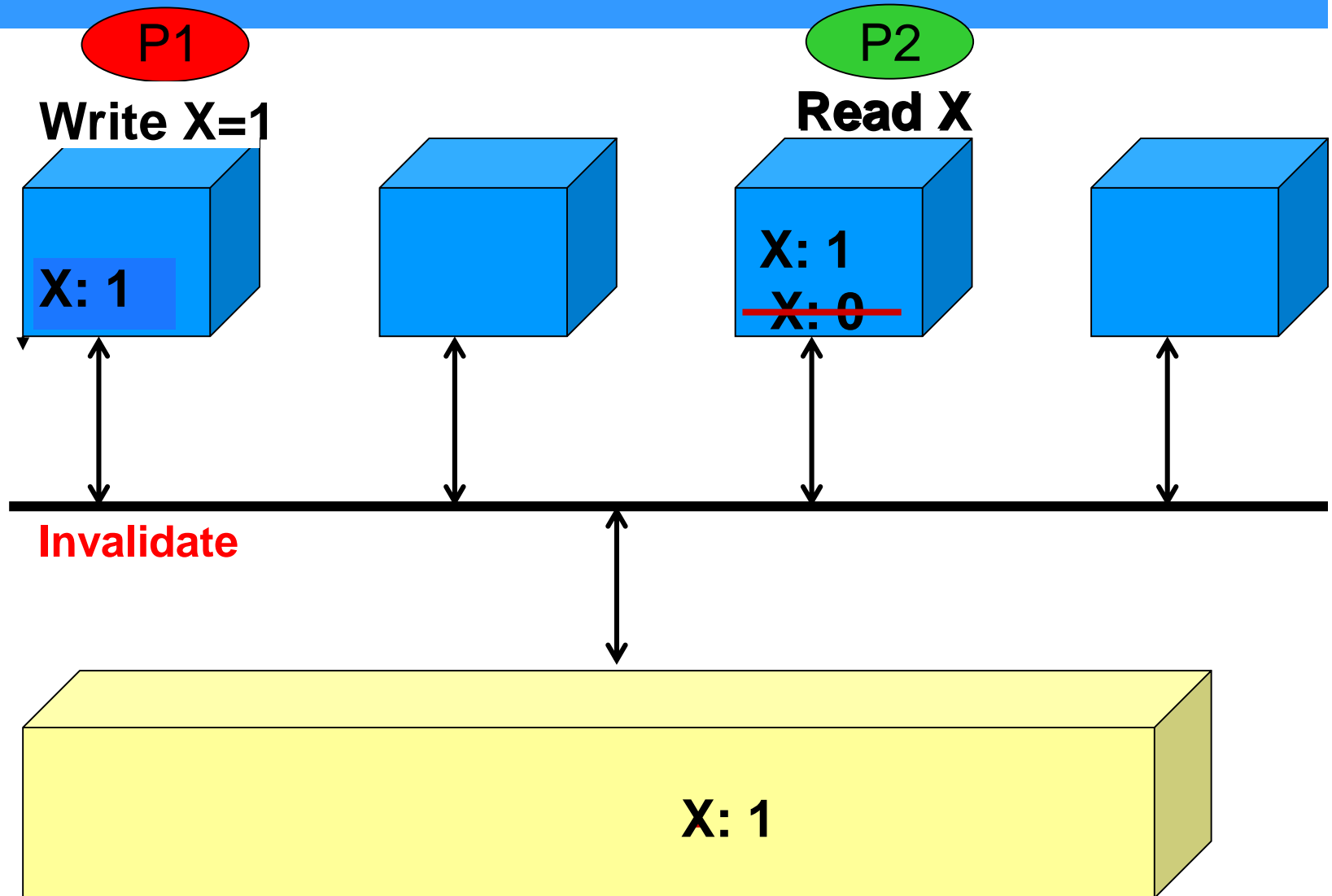


# Cache Coherence Protocols

- Write update - propagate cache line to other processors on every write to a processor
- Write invalidate - each processor gets the updated cache line whenever it reads stale data



# Invalidation Based Cache Coherence





# Cache Coherence using invalidate protocols

- 3 states associated with data items
  - Shared - a variable shared by 2 caches
  - Invalid - another processor (say P0) has updated the data item
  - Dirty - state of the data item in P0



# Thank You

