Parallel Distributed & Computing

Presented By:

Course Instructor

Snoopy-Bus Cache Coherence Protocol

Parallel distributed and computing

Table of Contents

Points to discuss:

- Cache Coherence
- Cache coherence protocol
- Snoopy Bus Protocol
  - Write - invalidate protocol
 - Write - Update (Write- Broadcast)

Cache Coherence

For higher performance in a multiprocessor system, each processor usually has its own cache.

In a multiprocessor system, data inconsistency may occur among adjacent levels or within the same level of the memory hierarchy.

As multiple processors operate in parallel, and independently multiple caches may possess different copies of the same memory block, this creates a cache coherence problem.

Cache coherence **refers to the problem of keeping the data in these caches** consistent.

The main problem is dealing with writing by writing by a processor.
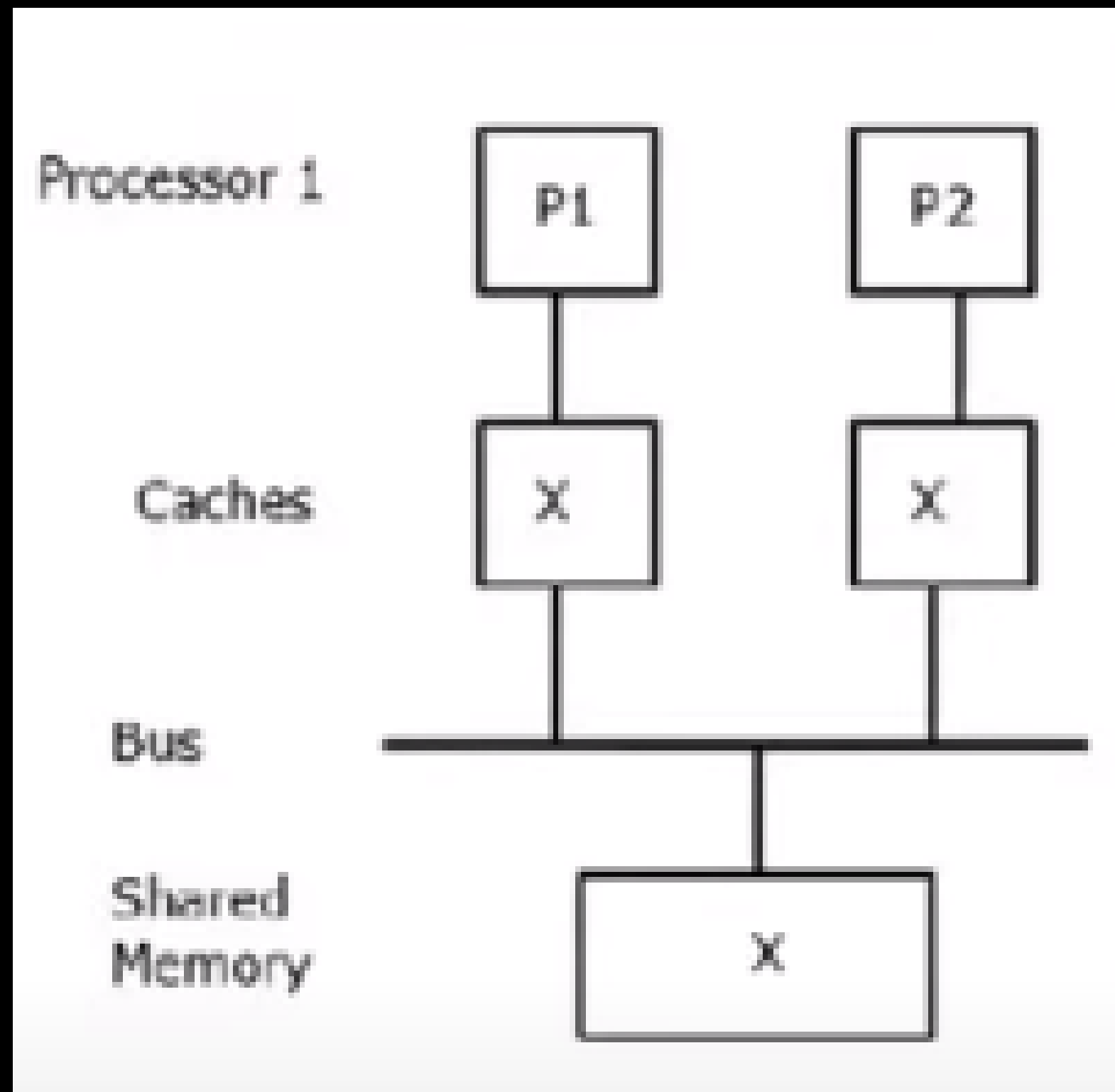
Cache Coherence Protocols
- **Snoopy-Bus Protocol**
- **Directory Based Protocol**

Snoopy-Bus Protocol

Used for bus-based multiprocessor systems (UMA Machines)

Transactions on the bus are visible to all processors.

In the Snoopy-Bus protocol, each processor's cache controller monitors or snoops on the bus for memory transactions and take proper action to invalidate or update the local cache content if needed.

Shortcoming: not scalable
More scalable solution: 'directory-based' coherence schemes
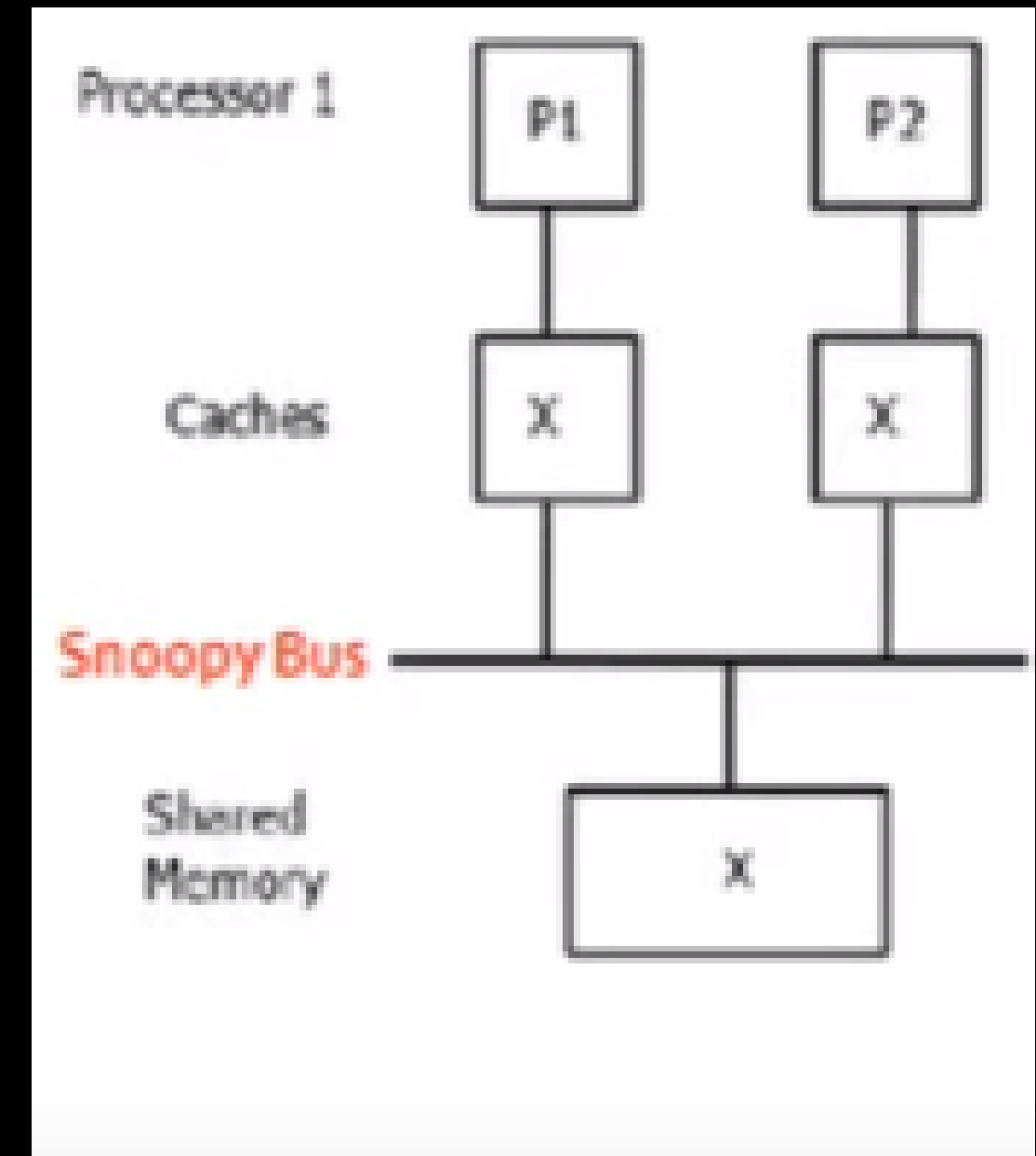
Two Basic Protocol

Using private caches associated with processors tied to a common bus, two approaches are used to maintain cache consistency.

1. Write-invalidate:
When a local cache copy is modified, Write-invalidate policy it invalidates all remote copies of cache (invalidated items are sometimes called "dirty")

2. Write-update (Write-broadcast):
When a local cache copy is updated, the Write-update policy broadcast a modified value of a data object to all other caches at the time of modifications

Snoopy protocols achieve data consistency among the caches and shared memory through a bus-watching mechanism. Two snoopy bus protocols create different results.

• **Let's see.** the **write-invalidate** and **write-update** coherence protocols write-through caches

Consider three processors (PI, P2, and P3) maintaining consistent copies of **block X** in their local caches and in the shared-memory module
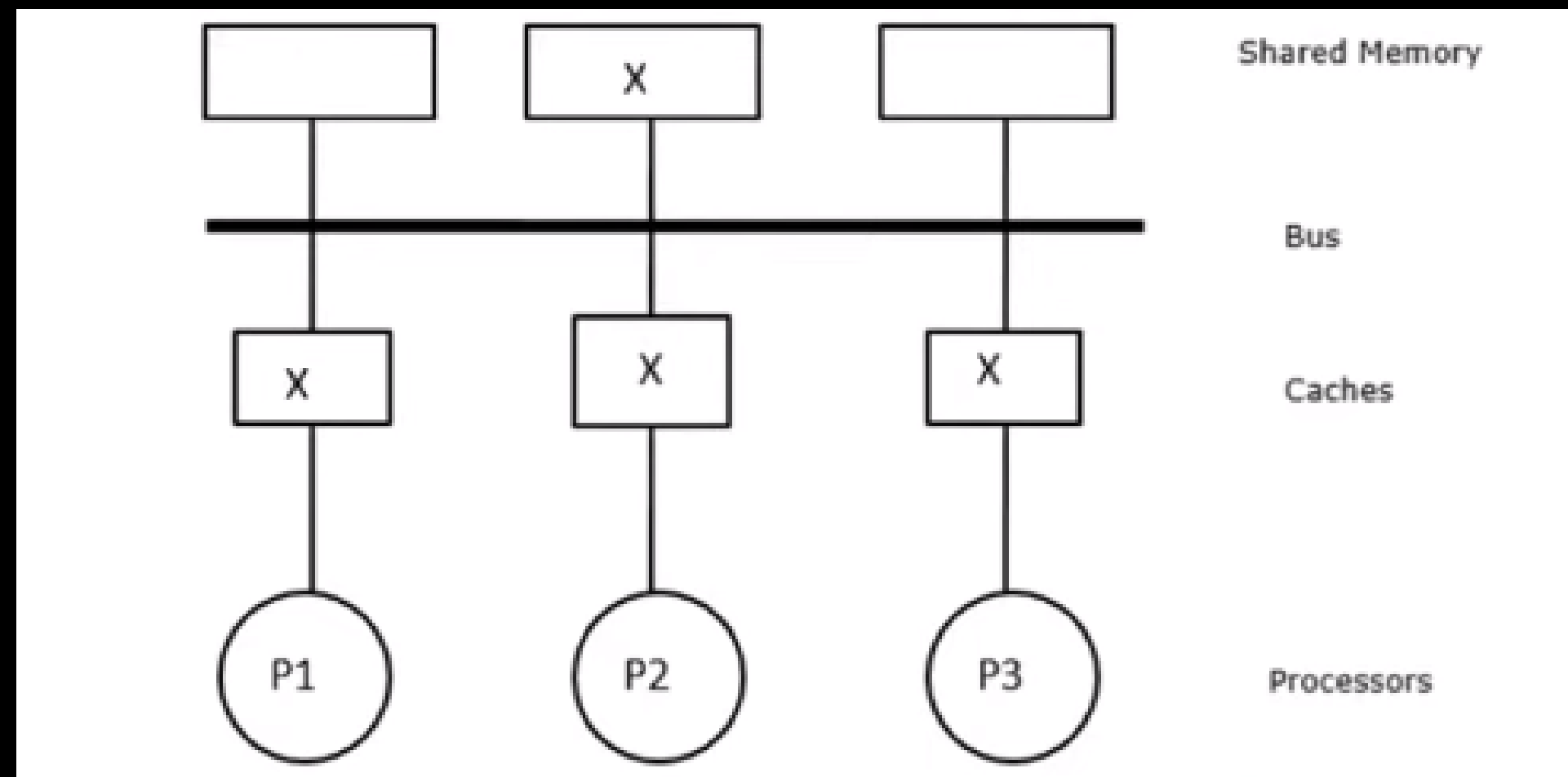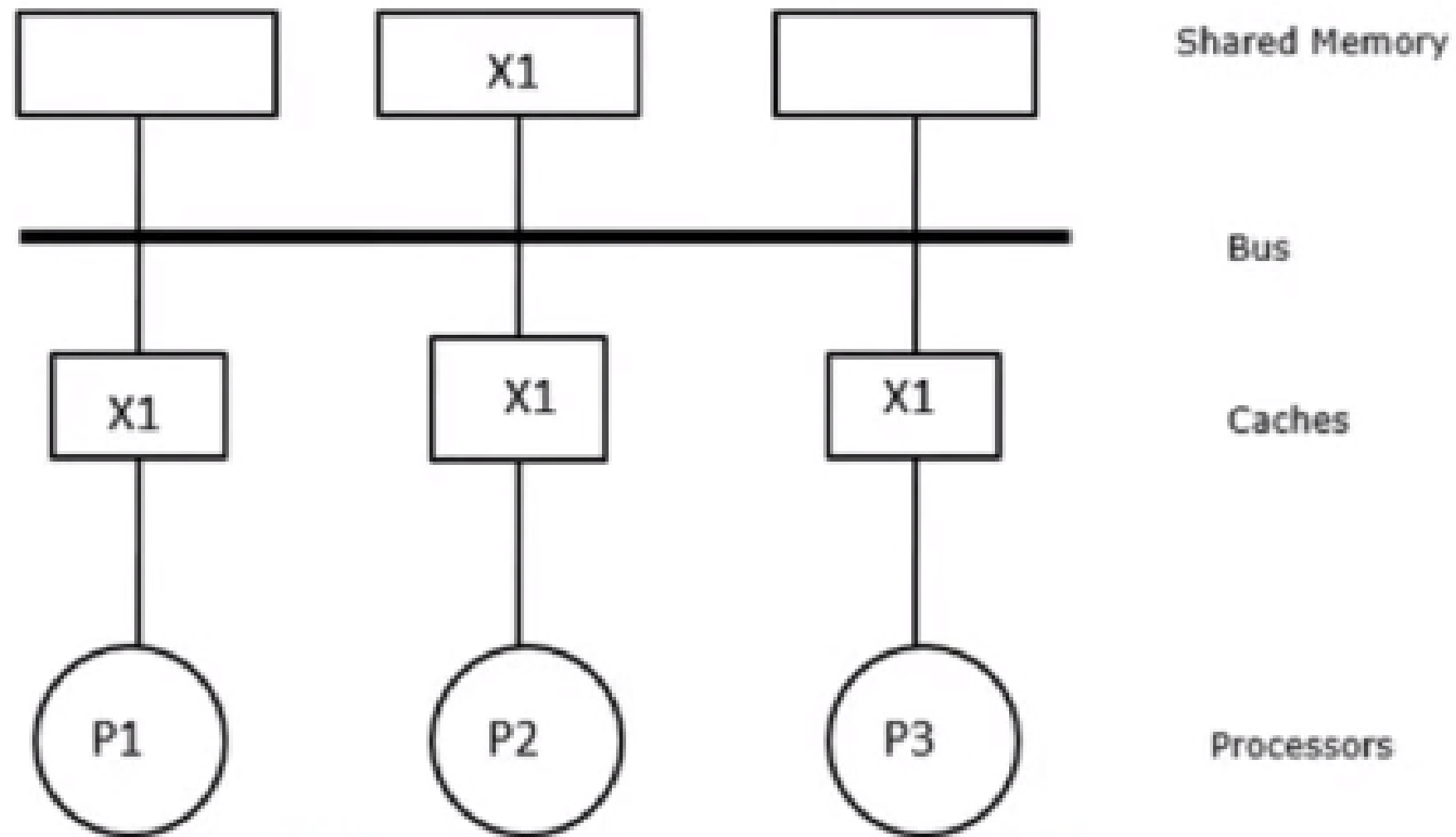
Write Invalidate Protocol
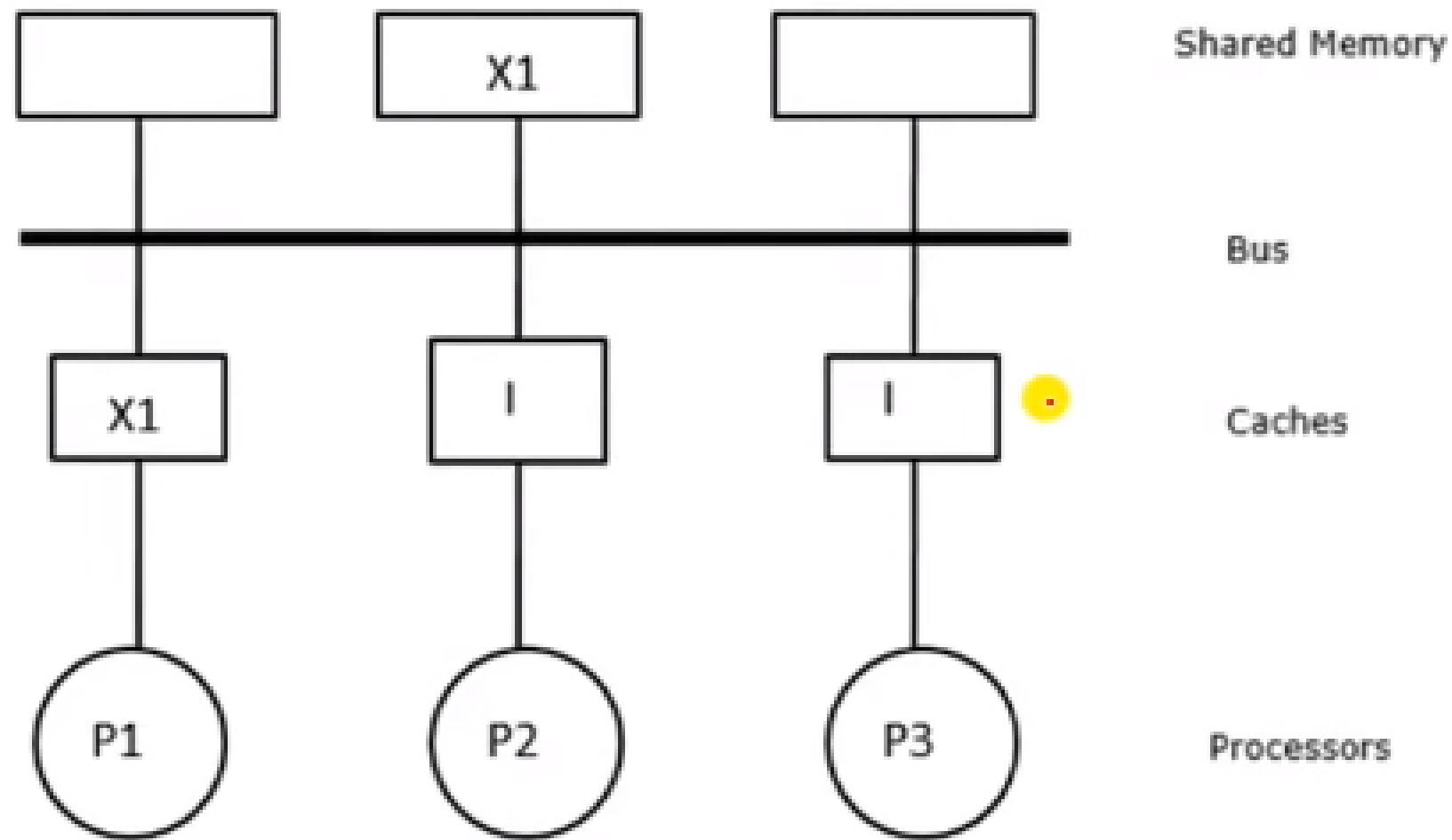
If processor **P1** modifies (writes) its cache from **X** to **X1**, then all other copies are invalidated via the bus. Invalidated blocks are sometimes called **dirty**, which means they should not be used.

Write Update Protocol

The new modified content X1 be broadcast (updated) to all cache copies via the bus.

THANK YOU