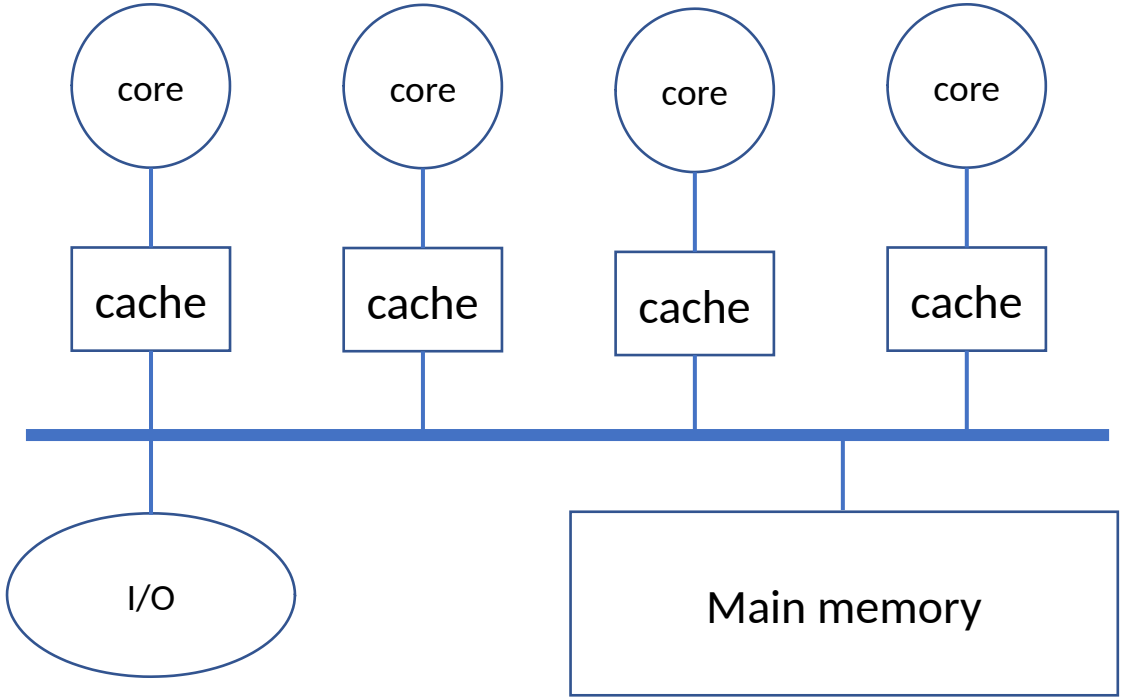
Videolesson 5 slides

Centralized Shared Memory

- Note: System with several cores, each with its private cache memory, accessing to centralized main memory and I/O through a common bus

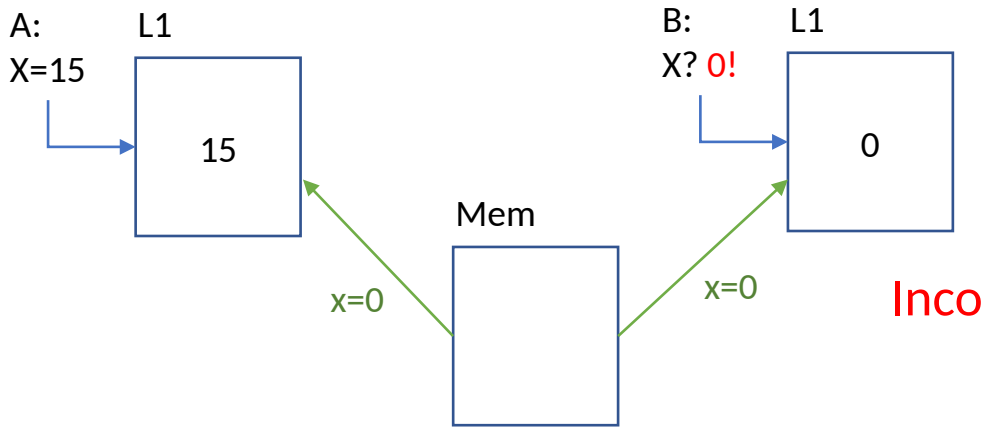


- Multi-core
- UMA: Uniform Memory Access (Time)
- SMP: Symmetric MultiProcessor

Cache coherence problem

- Programmer: shared memory
  - Core A writes x=15 , core B reads x, sees 15
- Hardware: each core has its own cache
  - One large L1 cache -> Too slow, not enough throughput
  - Private (per core) L1 caches

Note1: Diagram showing two cores A and B, with private cache L1, and main memory Mem.
Note 2: Listen to explanation in video for the specific example



Cache coherence: makes this behave as a single memory

Incoherent!

How to get coherence?

1) No caches (bad performance)
2) All cores share same L1 cache (bad performance)
3) Private write-through caches (Incoherent!)
4) Force read in one cache to see write made in another

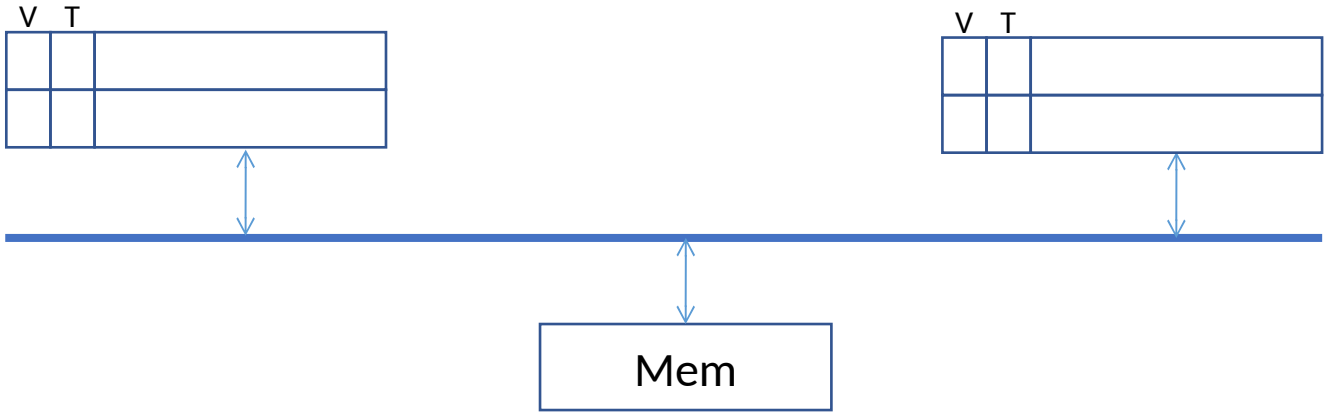
   on the writer side:

   a) Broadcast writes to update other caches (write update coherence)
   b) Writes prevent hits to other copies (write-invalidate coherence)

   on the reader side:

   c) Writes broadcasted on shared bus (snooping)
   d) Each block assigned an ordering point (directory)

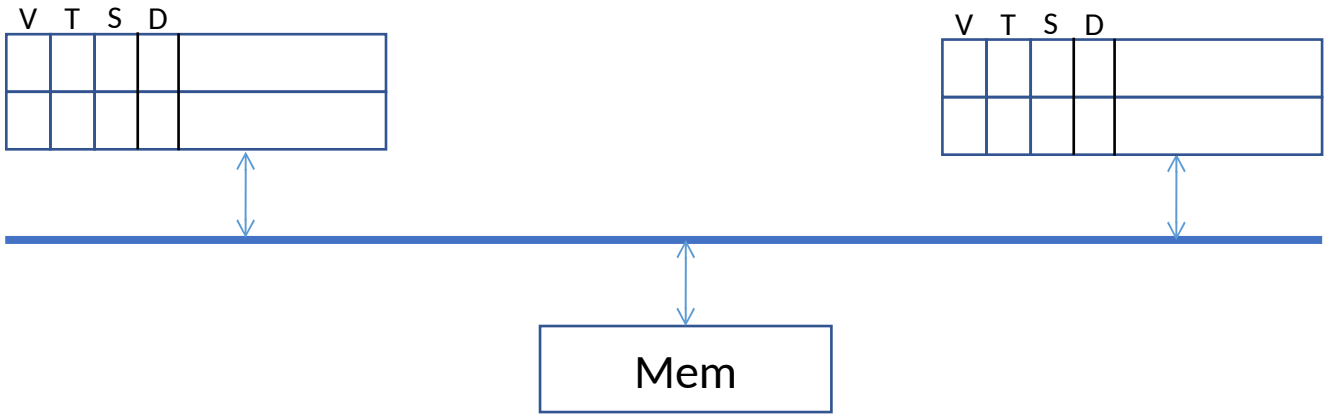
Write-update snooping coherence



Note 1: diagram showing the private caches of two processors and main memory. Each cache line has a valid bit (V), tag and data. Caches access the main memory through a bus.

Note 2: Listen to explanation in video for the specific example

Write-invalidate snooping coherence



Note 1: diagram showing the private caches of two processors and main memory. Each cache line has a valid bit (V), share bit (S) and dirty bit (D), tag and data. Caches access the main memory through a bus.

Note 2: Listen to explanation in video for the specific example