ECE 259 / CPS 221
Advanced Computer Architecture II
(Parallel Computer Architecture)

Shared Memory MPs - Directories

Duke University

Slides are derived from work by Sarita Adve (Illinois), Babak Falsafi (CMU), Mark Hill (Wisconsin), Alvy Lebeck (Duke), Steve Reinhardt (Michigan), and J. P. Singh (Princeton). Thanks!

Outline

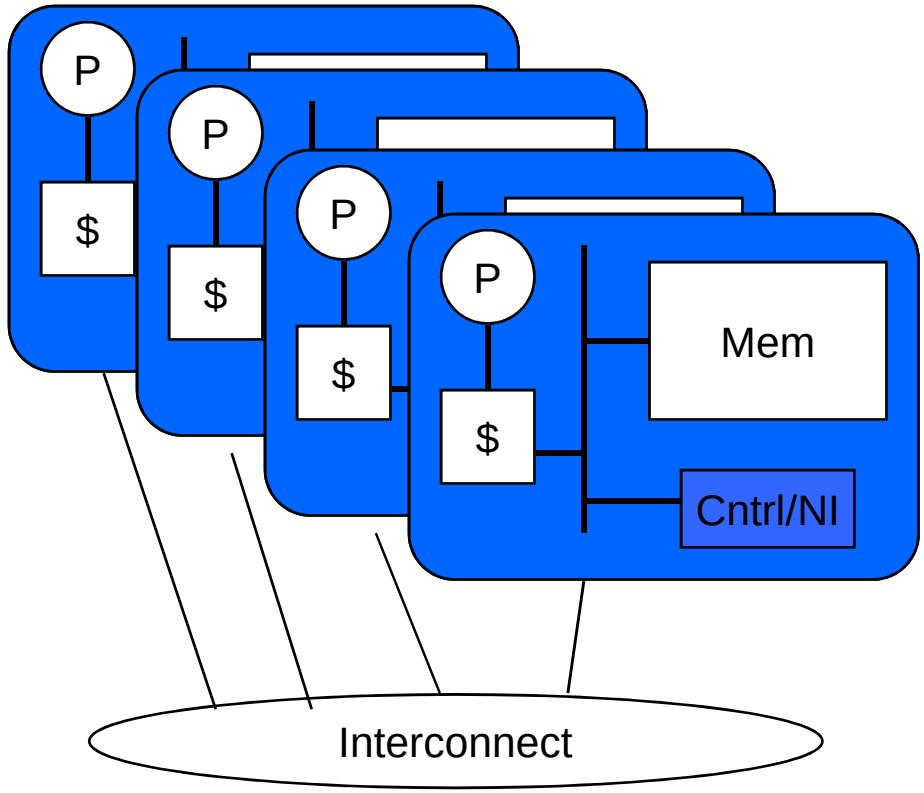
- **Directory-Based Cache Coherence**
  - **Motivation**
  - **Basic Idea**
  - **Some Variations**
- **Stanford DASH Case Study**
- **SGI Origin Case Study**
- **Advanced Directory Systems**

Why Snooping Doesn't Scale

- **Limitations of snooping**
  - **Broadcasting uses lots of "bus" bandwidth**
  - **Snooping every transaction uses lots of controller bandwidth**
- **Snooping is great for small-medium size machines**
  - **Largest current snooping system has 128 processors**
- **Snooping hits bottleneck for large machines**
  - **Even with tricks like Multicast Snooping**
- **So how do we overcome this bottleneck?**
  - **Get rid of protocol that requires:**
    - » **Broadcasting**
    - » **Logical bus**

Large Scale Shared Memory Multiprocessors

- **100s to 1000s of nodes (processors) with single shared physical address space**
- **Use general purpose interconnection network**
  - **Still have cache coherence protocol**
  - **Use messages instead of bus transactions**
  - **No hardware broadcast**

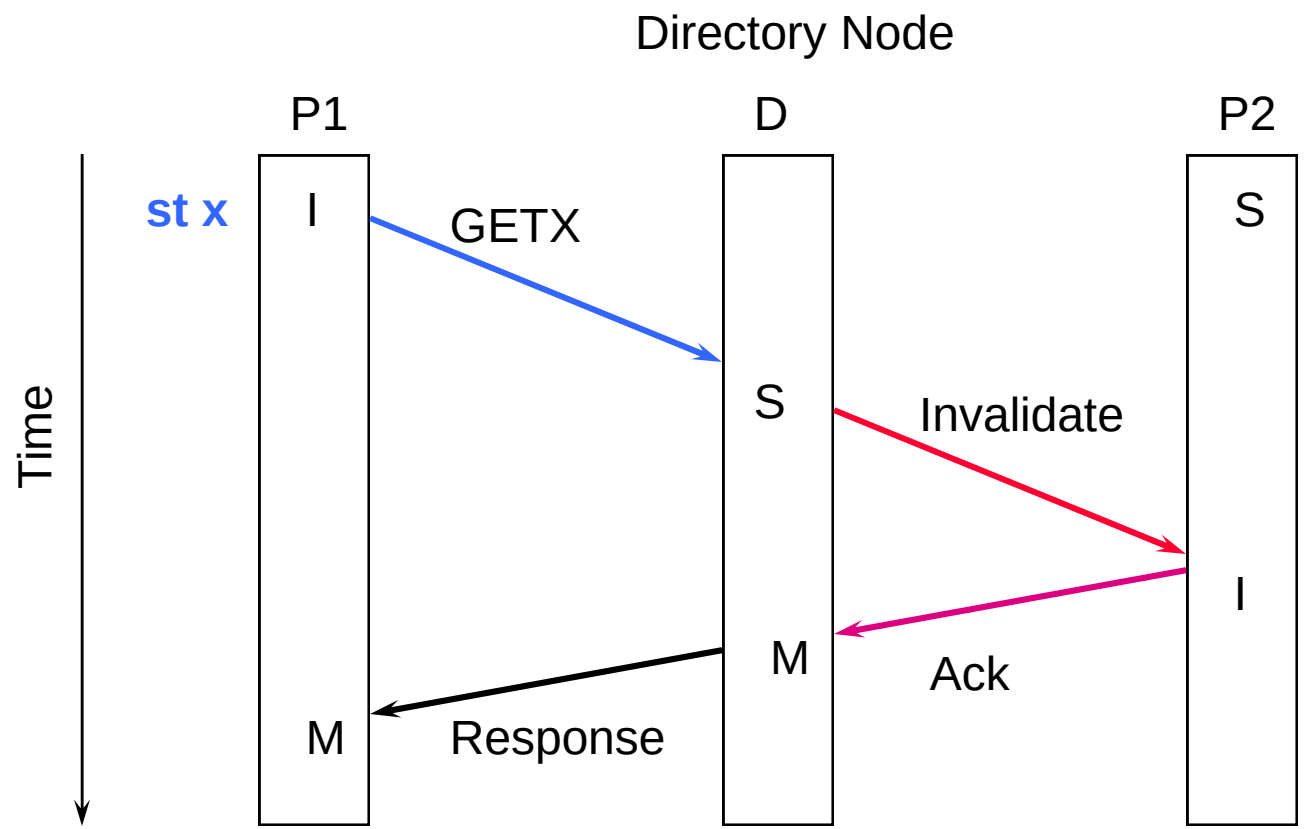


Directory Based Cache Coherence

- **Avoid broadcast request to all nodes on a miss**
- **Maintain directory of which nodes have cached copies of the block (directory controller + directory state)**
- **On a miss, cache controller sends message to directory**
- **Directory determines what (if any) protocol action is required**
  - **E.g., invalidations of Shared nodes**
- **Directory waits for protocol actions to finish and then responds to the original request**

Directory is new serialization point (instead of bus)

Directory Example



Centralized Directory

- **Single directory that contains a copy of all nodes' cache tags**

Disadvantages

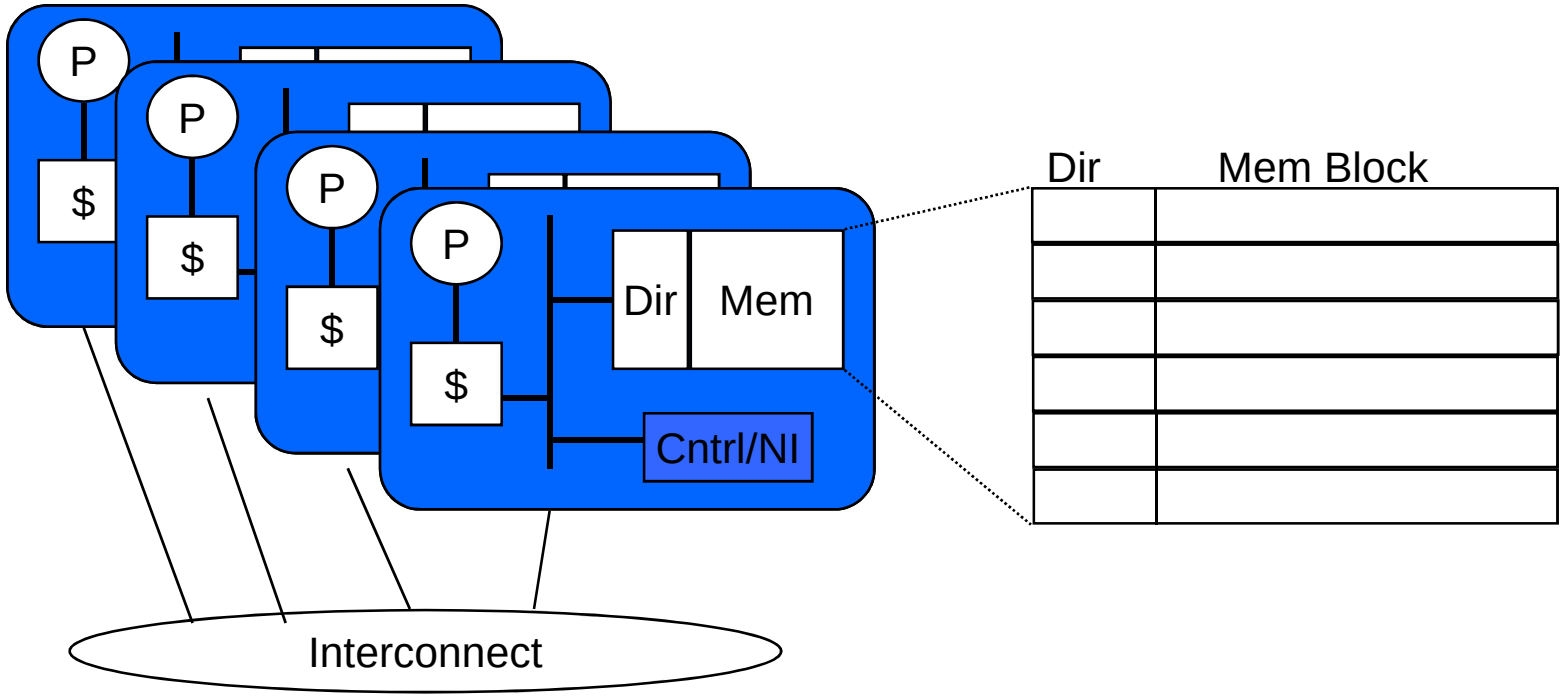
- **Bottleneck (1000s of processors…)**
- **Directory structure changes with number of nodes**

Advantages

- **Only send invalidates/updates to those nodes that have copy of block**

Flat, Memory-Based Distributed Directory

- **Distribute directory among memory modules**
- **Maintain directory for each memory block**
  - **Block's home node = node with directory info for that block**



Directory Nomenclature

- **Dir_iX -- 2 variables (i and X)**

- **Directory of *i* pointers (*i* <= total number of nodes)**
- **X specifies what to do on Shared to Modified transition**
  - **B = Broadcast**
  - **NB = No Broadcast**
  - **SW = Software**

- **Dir_N = full-map directory**
  - **Bit vector per memory block**
  - **Bit per node in system**
  - **No need to broadcast (unless <u>all</u> nodes are sharers)**

Limited Pointer Directory

- **Each directory entry contains < N pointers**
  - **Not bit vector, but instead pointers to nodes**
- **Less overhead than full-map directory**
- **What to do when run out of pointers depends on what we want to do on S→M transition**

Broadcast (Dir$_i$B)

- **Just give out another copy of block**
- **Modify state to indicate broadcast**
- **If S→M, then broadcast invalidation**

No Broadcast (Dir$_i$NB)

- **Never allow more than i Sharers**
  - **If another request for shared, then invalidate a current sharer**

Replacement Notification

- **Should the directory be notified for blocks that are replaced from Shared state?**

Reasons to do this

- **Can avoid broadcast, clear bit/pointer when notified**

Reasons not to do this

- **Read-only data that will never be invalidated**
- **Notifications may cause unnecessary traffic**

Coarse Vector and Sparse Directories

Coarse Vector

- Instead of full-map or broadcast, indicate a set of nodes that may have the block
- Reduces space requirements
- Many applications have near neighbor sharing

Sparse Directory (aka Directory Caching)

- Not all of memory is in processor caches
  - Size of memory >> sum of cache sizes
- Cache of directory entries at memory

Software Assistance

- **Trap to software if we run out of pointers**
- **Limitless Directory (MIT Alewife)**
- **Dir_1SW (Wisconsin Wind Tunnel Group)**

Why software assistance?

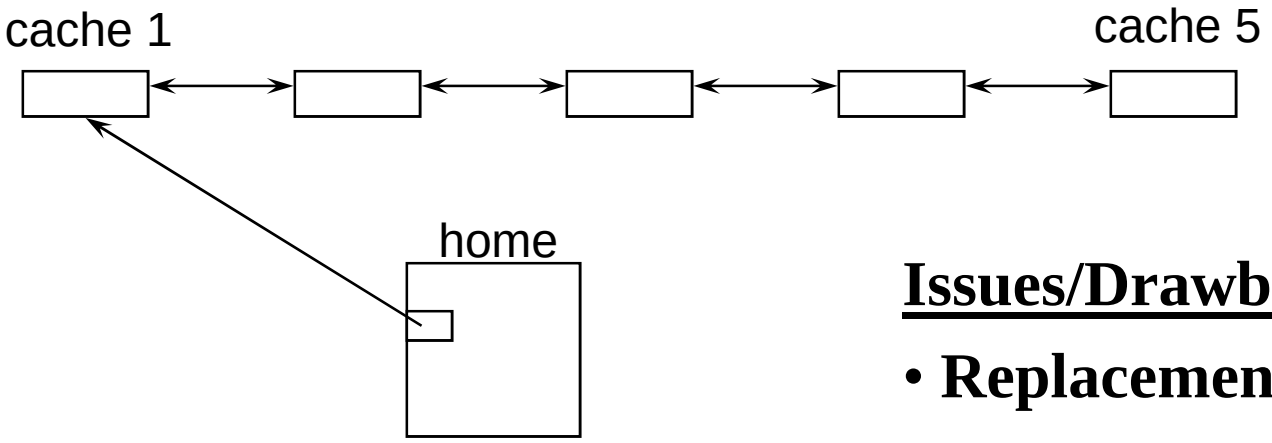
- **Cost (less hardware)**
- **Flexibility**

Actually, can do *everything* in software = Software DSM

- **Page-based DSM**
- **Blizzard-S**
- **Shasta**

Flat, Cache-Based Distributed Directory (SCI)

- **Also known as "chaining directory"**
- **Build linked list of nodes containing cache block**
- **Store pointers in cache with block of data**
- **Home node points to start of list**



Issues/Drawbacks

- **Replacements**
- **Time to send invalidations**