Module 14: "Directory-based Cache Coherence"
Lecture 30: "SGI Origin 2000"

Directory-based Cache Coherence:

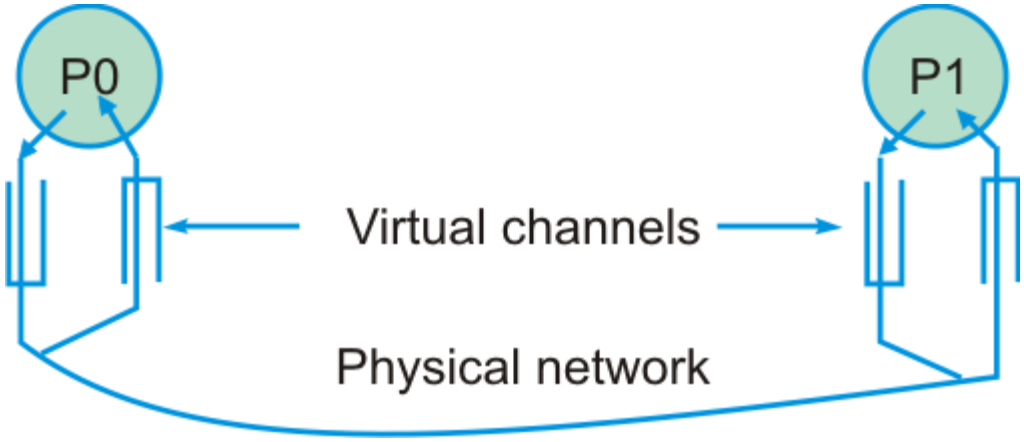
- Virtual networks
- Three-lane protocols
- Performance issues
- SGI Origin 2000
- Origin 2000 network
- Origin 2000 I/O
- Origin directory
- Cache and dir. states
- Handling a read miss
- Serializing requests
- Handling writebacks

[From Chapter 8 of Culler, Singh, Gupta]
[SGI Origin 2000 material taken from Laudon and Lenoski, ISCA 1997]
[GS320 material taken from Gharachorloo et al., ASPLOS 2000]

Previous Next

Virtual networks

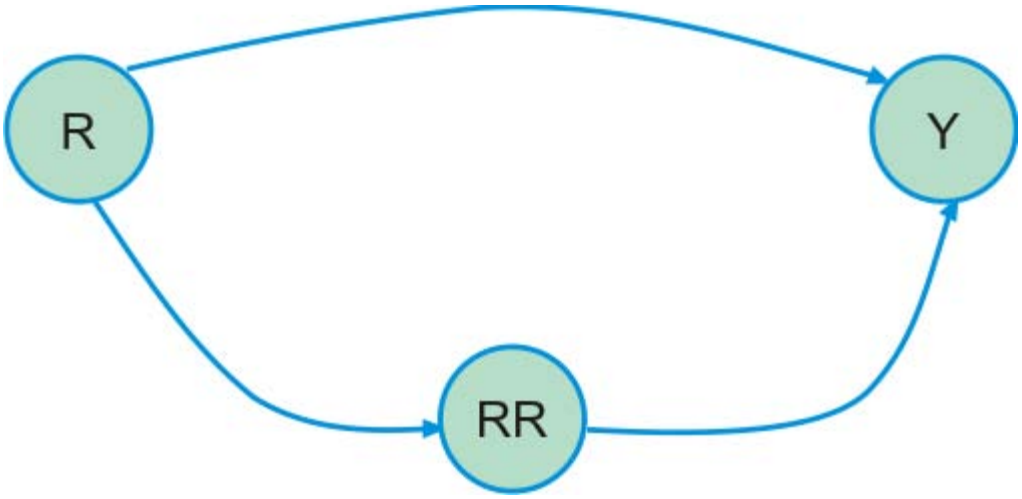
- Consider a two-node system with one incoming and one outgoing queue on each node



- Single queue is not enough to avoid deadlock
  - Single queue forms a single virtual network
- Similar deadlock issues as multi-level caches
  - An incoming message may generate another message e.g., request generates reply, ReadX generates reply and invalidation requests, request may generate intervention request
  - Memory controller refuses to schedule a message if the outgoing queue is full
  - Same situation may happen on all nodes: deadlock
  - One incoming and one outgoing queue is not enough
  - What if we have two in each direction?: one for request and one for reply
  - Replies can usually sink
  - Requests generating requests?
- What is the length of the longest transaction in terms of number of messages?
  - This decides the number of queues needed in each direction (Origin 2000 uses a different scheme)
  - One type of message is usually assigned to a queue
  - One queue type connected across the system forms a virtual network of that type e.g. request network, reply network, third party request (invalidations and interventions) network
  - Virtual networks are multiplexed over a physical network
- Sink message type must get scheduled eventually
  - Resources should be sized properly so that scheduling of these messages does not depend on anything
  - Avoid buffer shortage (and deadlock) by keeping reserved buffer for the sink queue

Three-lane protocols

- Quite popular due to its simplicity
  - Let the request network be R, reply network Y, intervention/invalidation network be RR
  - Network dependence (aka lane dependence) graph looks something like this



Previous Next

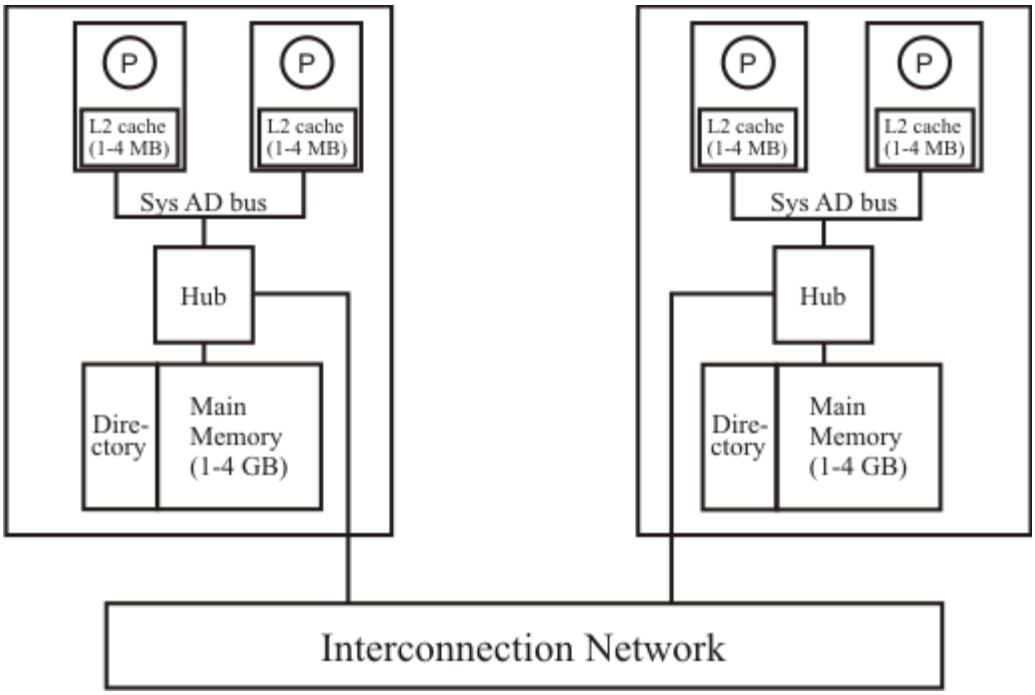
Module 14: "Directory-based Cache Coherence"
Lecture 30: "SGI Origin 2000"

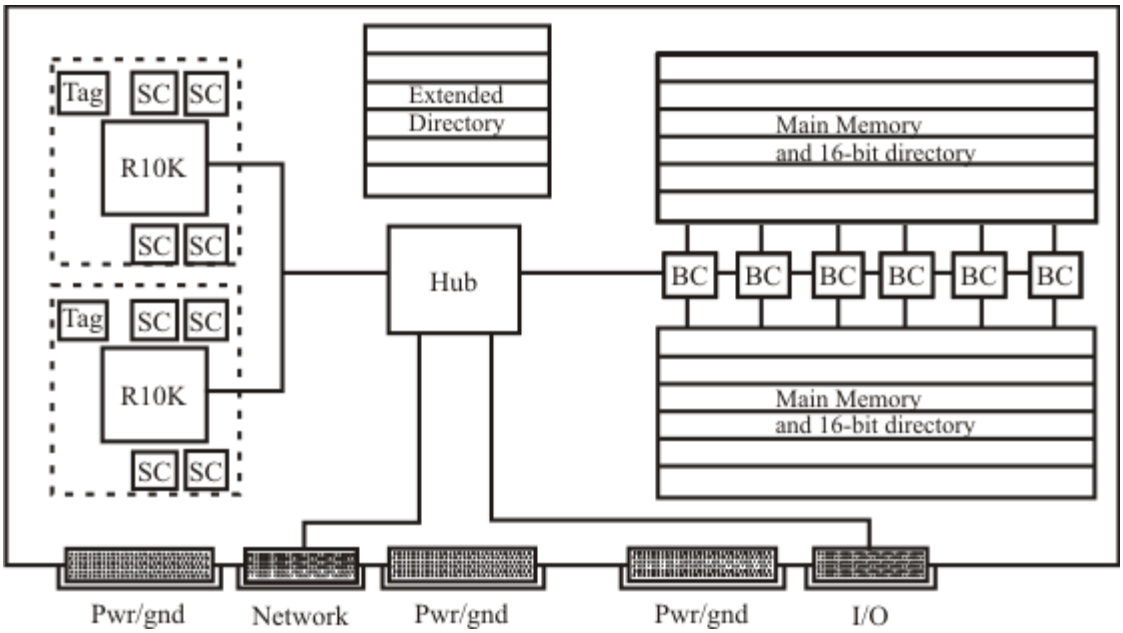
Performance issues

- Latency optimizations
  - Reduce transactions on critical path: 3-hop vs. 4-hop
  - Overlap activities: protocol processing and data access, invalidations, invalidation acknowledgments
  - Make critical path fast: directory cache, integrated memory controller, smart protocol
  - Reduce occupancy of protocol engine
- Throughput optimizations
  - Pipeline the protocol processing
  - Multiple coherence engines
  - Protocol decisions: where to collect invalidation acknowledgments, existence of clean replacement hints

SGI Origin 2000

- Similar to Stanford DASH
- Flat memory-based directory organization





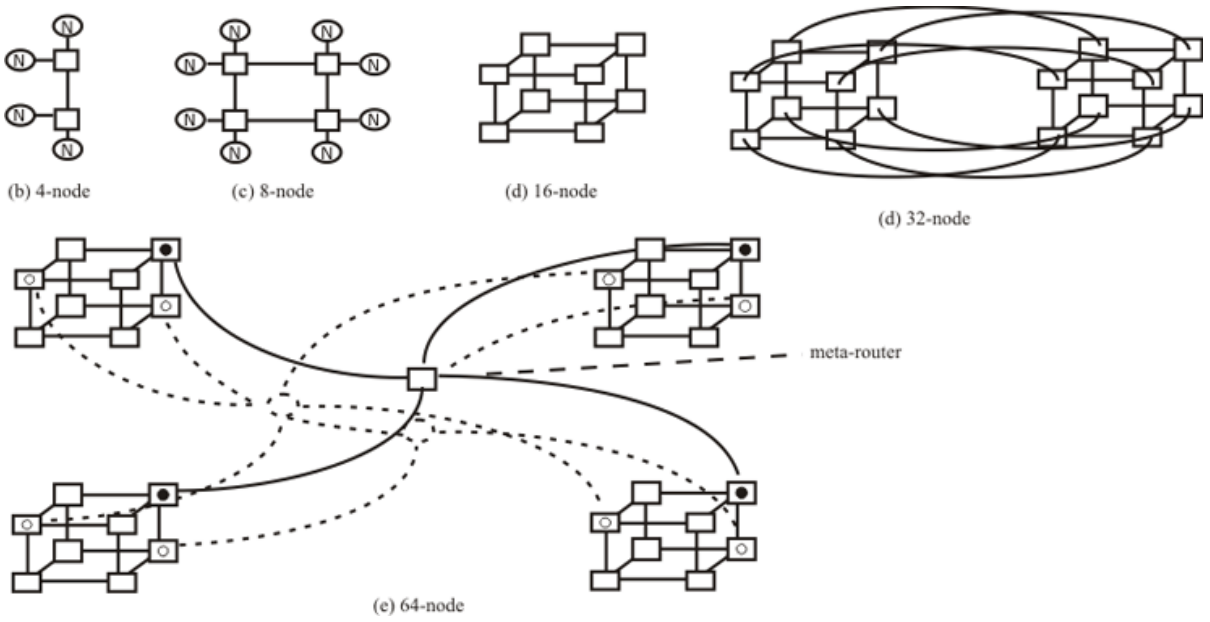
Connections to Backplane

- Directory state in separate DRAMs, accessed in parallel with data
- Up to 512 nodes (1024 processors)
- 195 MHz MIPS R10k (peak 390 MFLOPS and 780 MIPS per processor)
- Peak SysADBus (64 bits) bandwidth is 780 MB/s; same for hub-memory
- Hub to router and Xbow (I/O processor) is 1.56 GB/s
- Hub is 500 K gates in 0.5 micron CMOS
- Outstanding transaction buffer (aka CRB): 4 per processor
- Two processors per node are not snoop-coherent

Previous Next

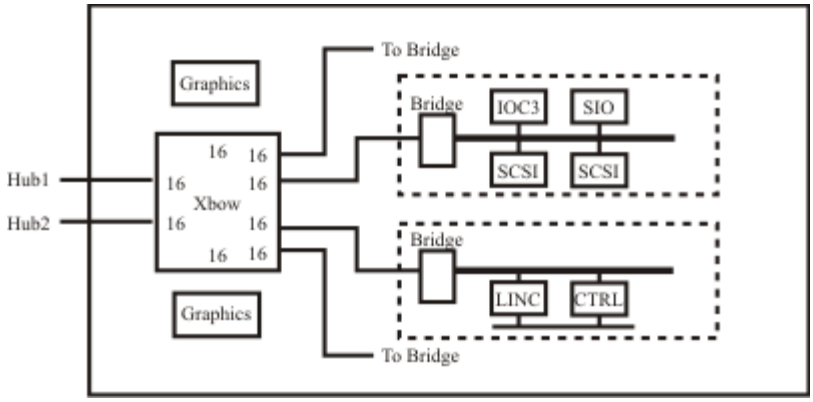
Module 14: "Directory-based Cache Coherence"
Lecture 30: "SGI Origin 2000"

Origin 2000 network



- Each router has six pairs of 1.56 GB/s unidirectional links; two to nodes (bristled), four to other routers
- 41 ns pin to pin latency
- Four virtual networks: request, reply, priority, I/O

Origin 2000 I/O



- Any processor can access I/O device either through uncached ops or through coherent DMA
- Any I/O device can access any data through router/hub

Origin directory

- Directory formats
  - If exclusive in a cache, entry contains processor number (not node number)
  - If shared, entry is a bitvector of sharers where each corresponds to a node (not a processor)

- Invalidations sent to a node is broadcast to both processors by hub

- Two sizes
  - 16-bit format (up to 32 processors), kept in DRAM
  - 64-bit format (up to 128 processors), kept in extension DRAM
  - For machine sizes larger than 128 processors the protocol is coarse-vector (each bit is for 8 nodes)
  - Machine can switch between BV and CV dynamically

Previous Next

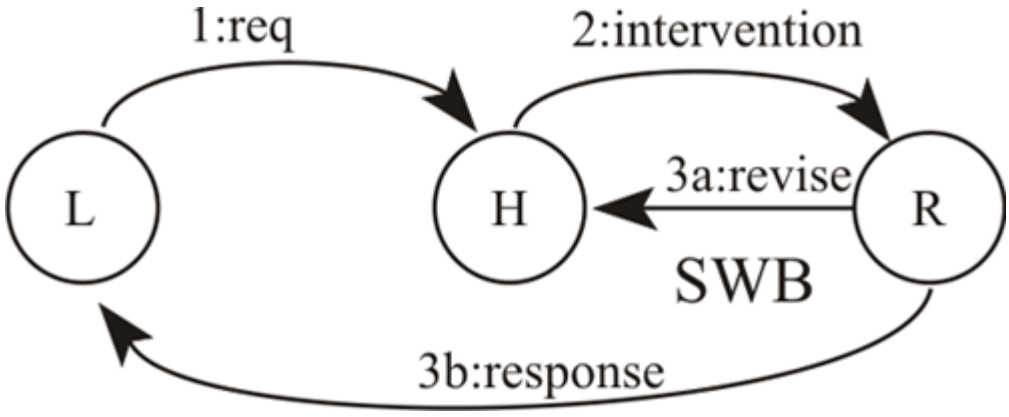
Module 14: "Directory-based Cache Coherence"
Lecture 30: "SGI Origin 2000"

Cache and dir. states

- Cache states: MESI
- Six directory states (may not be six bits)
  - Unowned (I): no cache has a copy, memory copy is valid
  - Shared (S): one or more caches have copies, memory copy is valid
  - Dirty exclusive (M or DEX): exactly one cache has block in M or E state
    - Directory cannot distinguish between M and E
  - Two pending or busy or transient states (PSH and PDEX): a transaction for the cache block is in progress; home cannot accept any new request
  - Poisoned state: used for efficient page migration

Handling a read miss

- Origin protocol does not assume anything about ordering of messages in the network
- At requesting hub
  - Address is decoded and home is located
  - Request forwarded to home if home is remote
- At home
  - Directory lookup and data lookup are initiated in parallel
  - Directory banks are designed to be slightly faster than other banks
  - The directory entry may reveal several possible states
    - Actions taken depends on this
- Directory state lookup
  - Unowned: mark directory to point to requester, state becomes M, send cache line
  - Shared: mark directory bit, send cache line
  - Busy: send NACK to requester
  - Modified: if owner is not home, forward to owner



  - 3-hop vs. 4-hop reply?
  - Origin has only two virtual networks available to protocol
    - How to handle interventions?
- Directory state M
  - Actions at home: set PSH state, set the vector with two sharers, NACK all subsequent requests until state is S
  - Actions at owner: if cache state is M send reply to requester (how to know who the requester is?) and send sharing writeback (SWB) to home; if cache state is E send completion

messages to requester and home (no data is sent); in all cases cache state becomes S
- Sharing writeback or completion message, on arrival at home, changes directory state to S
- If the owner state is E, how does the requester get the data?
    - The famous speculative reply of Origin 2000
    - Note how processor design (in this case MIPS R10k) influences protocol decisions

Previous Next

Handling a write miss

- Request opcode could be upgrade or read exclusive
  - State busy: send NACK
  - State unowned: if ReadX send cache block, change state to M, mark owner in vector; if upgrade what do you do?
  - State shared: send reply (upgrade ack or ReadX reply) with number of sharers, send invalidations to sharers, change state to M, mark owner in vector; sharers send invalidation acknowledgments to requester directly
    - What if outgoing request network queue fills up before all invalidations are sent?
  - State M: same as read miss except directory remains in PDEX state until completion message (no data) is received from owner; directory remains in M state, only the owner changes; how do you handle upgrades here?

Serializing requests

- The tricky situation is collection of invalidation acknowledgments
  - Note from previous slides that even before all acknowledgments are collected at the requester, the directory at home goes to M state with the new owner marked
  - A subsequent request will get forwarded to the new owner (at this point directory goes to PSH or PDEX state)
  - The owner is responsible for serializing the new request with the previous write
    - The write is not complete until all invalidation acknowledgments are collected
    - OTT (aka CRB) of the owner is equipped to block any incoming request until all the acknowledgments and the reply from home are collected (early interventions)
  - Note that there is no completion message back to home

Handling writebacks

- Valid directory states: M or busy; cannot be S or I
- State M
  - Just clear directory entry, write block to memory
  - Need to send writeback acknowledgment to the evicting processor (explanation coming up)
- State busy
  - How can this happen? (Late intervention race)
  - Can NACK writeback? What support needed for this?
  - Better solution: writeback forwarding
  - Any special consideration at the evicting node?
    - Drop intervention (how?)
  - How does the directory state change in this case?