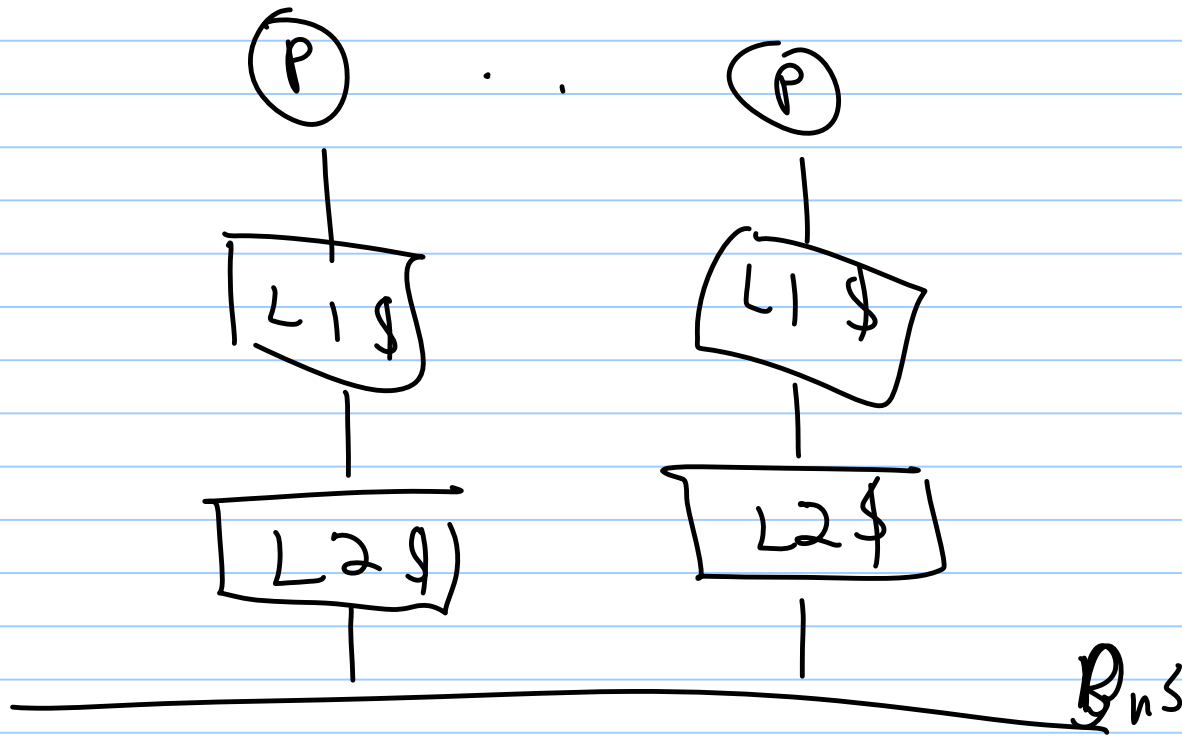


# Real Caches



snoop multi-level cache

- independent snooping @ each level
- maintain cache inclusion

- data in higher-level cache is subset of data in lower-level cache

- modified in higher-level cache  $\Rightarrow$  marked modified in lower-level cache

only need to snoop lower-level cache

Violations of inclusion - how?

L1, L2 may choose to replace different blocks

- diff reference history  
set-assoc. L1 w/ LRU replacement

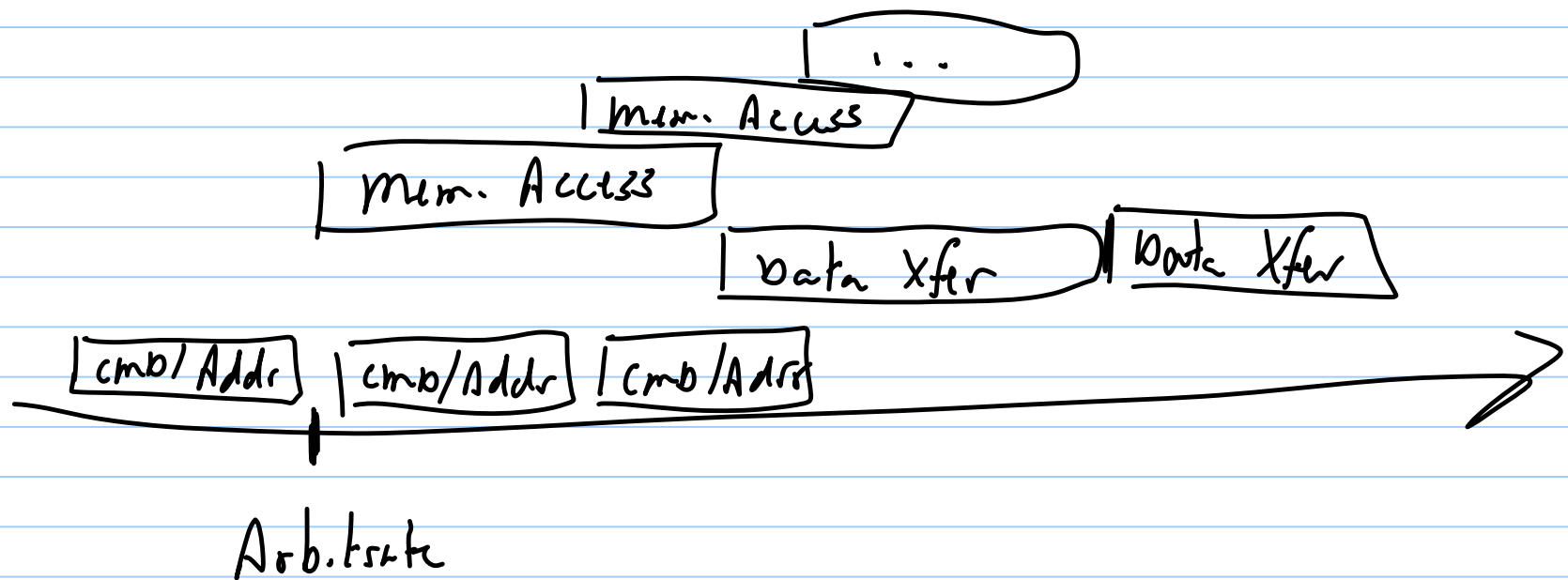
- split I-D caches @ L1  
conflict in L2

Need to explicitly preserve inclusion

- prop. L2 replacement to L1

- prop mod. state from L1 to L2 on writes

# Split Transaction Bus



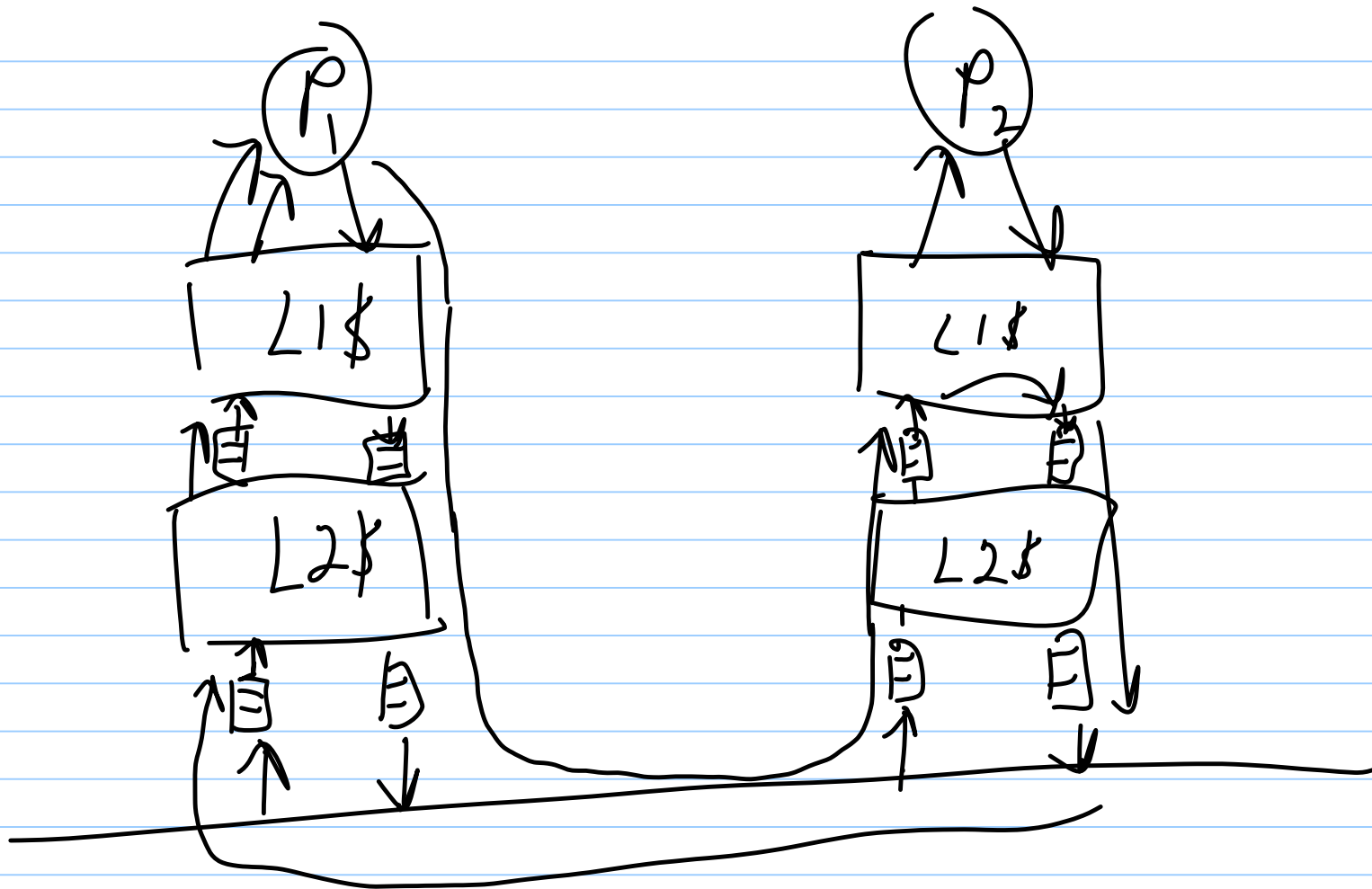
New request can appear on bus before previous one is serviced, even before snop results are known

Let's look @ SBI Challenge

- no conflicting requests for same block allowed on bus
- flow control via NACK
- = # outstanding requests allowed
- responses allowed to be out of order

- essentially 2 buses request / response
- out of order requires matching mechanism
  - 3-bit tag is sufficient

→ Serialization via request bus



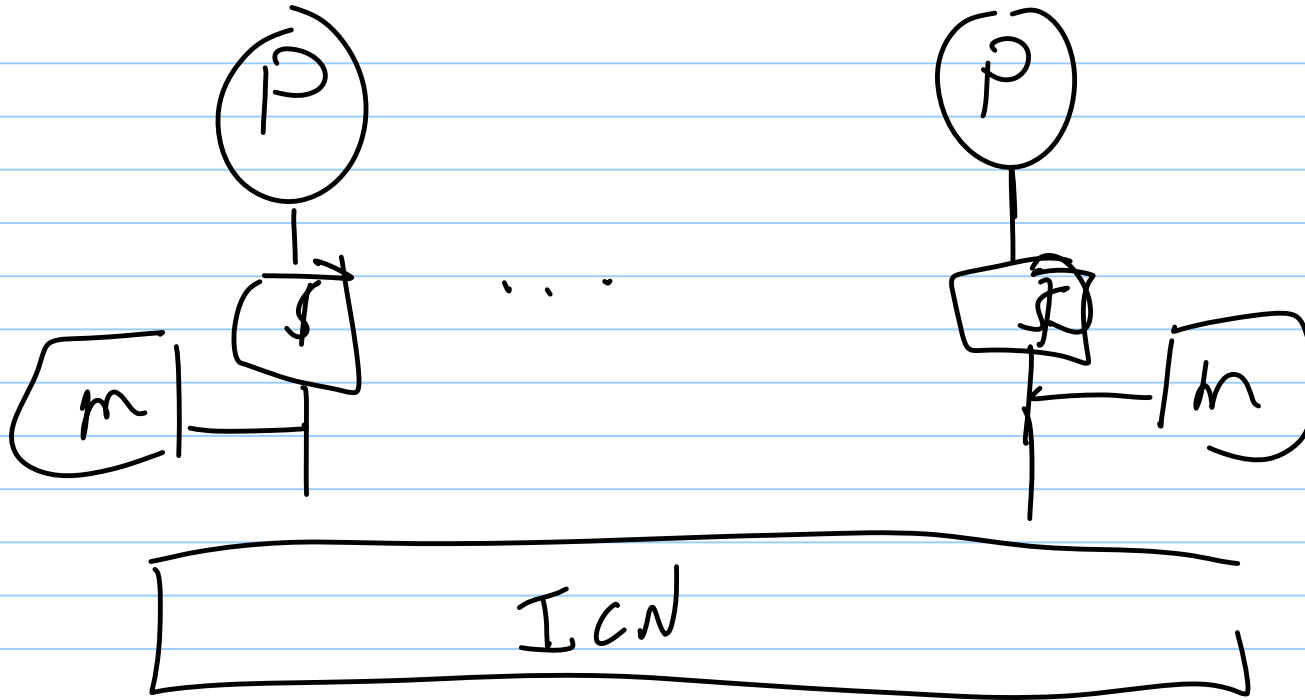
Fetch deadlock

need  $n+1$  buffer slots ( $n$  is # of proc)

Buffer deadlock

L1 to L2 full with read req. from proc.

L2 to L1 full with bus req.

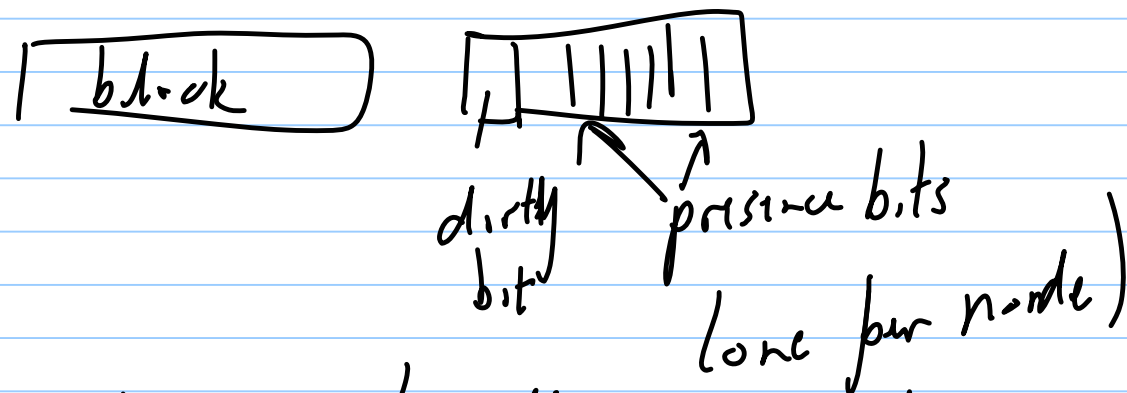


- unbounded latency due to contention
- variable delay for each  $(s, d)$  pair
- order preserving only for  $(s, d)$  pair, not global
- snooping doesn't scale

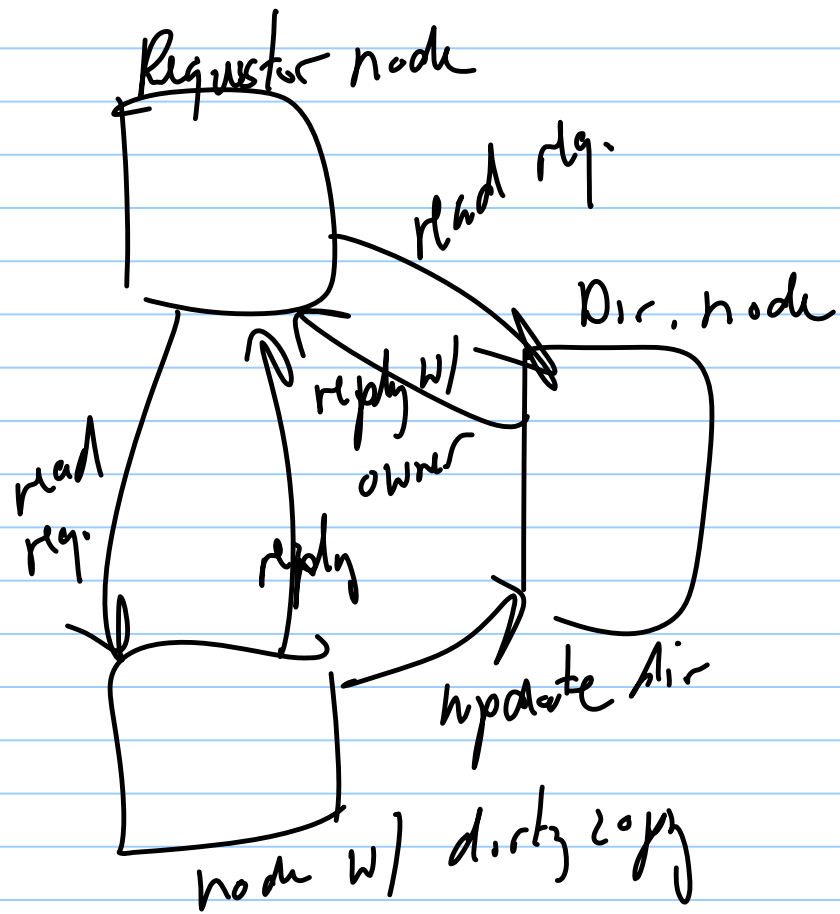
⇒ directory-based coherence

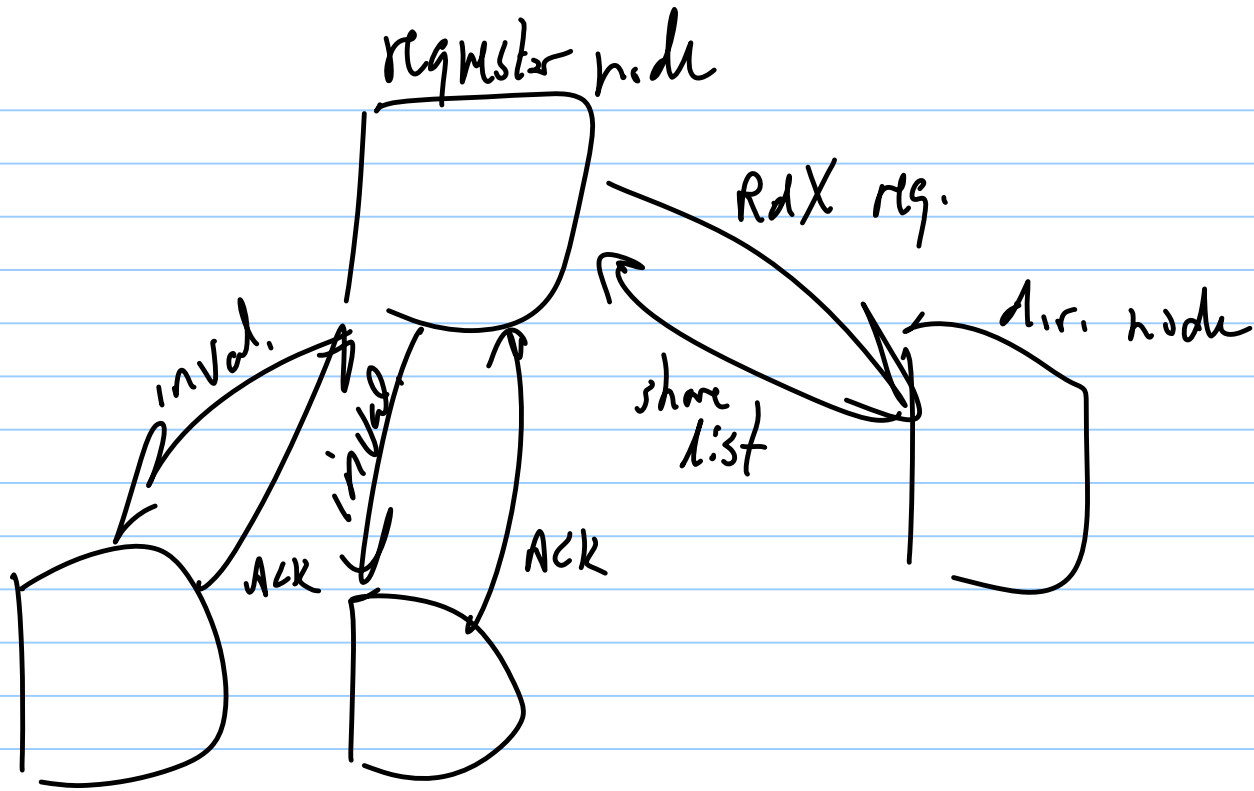
every memory block has assoc. dir. info.

keeps track of copies of cached blocks + their status



on miss, go to dir and only comm. w/  
procs that have copies





snoop  
with in  
node

Avr.  
across  
nodes

