Lecture 31 (Directory Protocol)

knowing English and following Indian accent are two different things

1 Directory Protocol

1. When we do not have a bus
2. Have a dedicated structure called directory
3. It co-ordinates the actions of coherence protocol
4. Sends and receives messages to/from caches and lower levels

1.1 Directory Entry

1. State
2. Block address
3. List of shareres

1.2 Design of Directory

1. RdX - locate a sharer and fetch the block
2. WrX and WrX.u - ask all sharers to invalidate their lines and give exclusive rights to write requestor
3. Evict - delete the sharere from entry

1.3 Issues

1. Need a very large directory
2. Directory may also become a point of contention

1.4 Resolution Ideas

1. Distributed directories - split the physical address space, resolves contention issues
2. Directory as cache - if entry is evicted from directory, invalidate all sharers

1.5 How to Maintain List of Shareres

1. Fully mapped scheme - inefficient
2. Maintain a bit for a set of caches - snoopy protocol inside this set
3. Partially mapped scheme - store id of only k sharers, if more than k, broadcast

2 Memory Models

2.1 Write-to-Read Order

1. `rfi` is not global
2. Because of LSQ forwarding and write buffers

2.2 Non-Atomic Writes

1. `rfe` is not global
2. Because of local tiles of caches

2.3 Write-to-Write Order

1. This is not allowed even if writes are atomic
2. But is violated in case of non-blocking caches

2.4 Read-to-Read Order

1. Violated because of OOO loads in LSQ

2.5 Read-to-Write Order

1. Is maintained in OOO machines in general
2. Violated when we have speculative writes

2.6 Can `rfi` be relaxed in SC?

1. Answer is yes
2. Proof is in book