

SNOOPING PROTOCOL

- 3 states: Exclusive, Invalid, shared.
- Main idea: only 1 processor should modify a specific piece of data. what would happen otherwise?
- Concep of "block ownership"
- Modes Operandi:
 - 1: Processor owns the sole copy of a cache block if block is to be modified/or owned then:
 - change state from shared to exclusive
 - send invalidation signal to other processors.
 - 2: IF another processor requests this block, state must be shared again.
 - 3: Snooping also sees misses.
 - knows when exclusive cache block has been requested by another processor and state should be made shared.

* Protocol implemented with a FSM @ each node

- controller responds to requests from:
 - processor
 - bus
- controller changes the state of the selected block
- controller uses bus to access data or invalidate it.

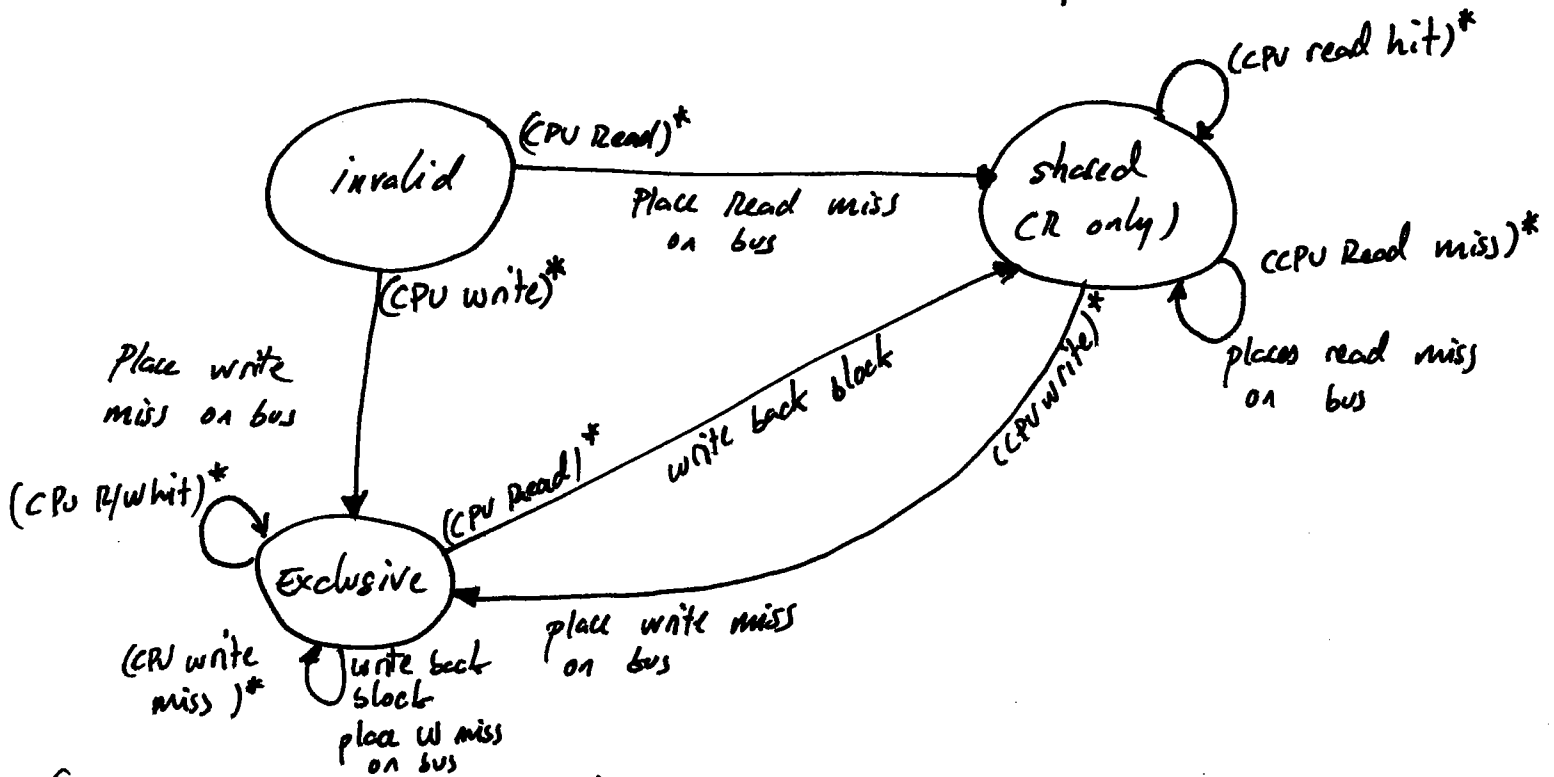
* Some operations involved in the protocol.

Request	Source	Function
- Read hit	processor	Read data in cache
- Write hit	processor	write data in cache
- Read miss	Bus	Request Data from cache or memory
- Write miss	Bus	Request Data from cache or memory perform any needed invalidates

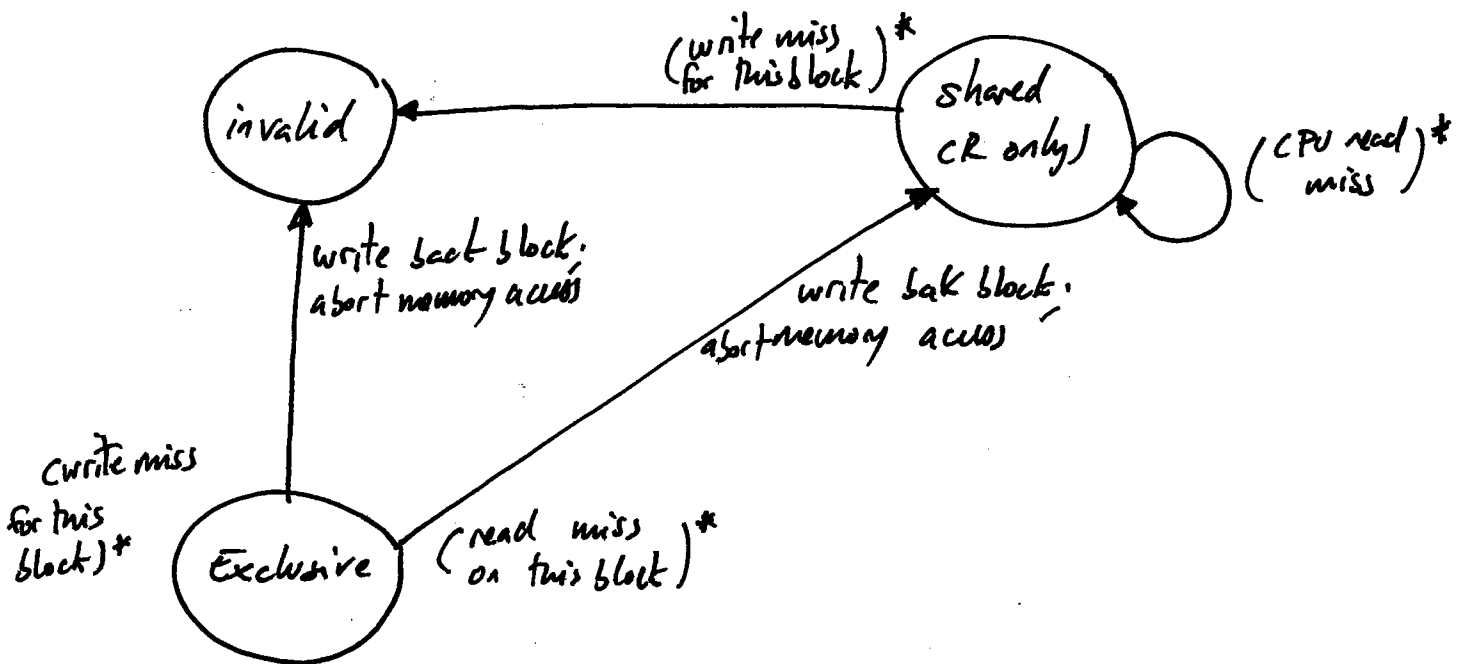
STATE TRANSITIONS:

(a) Based on CPU requests:

*: conditions on which state transition depends.



(b) Based on Bus requests:



Why does this work?

- Valid slots are in the following states:
 - shared: in multiple caches
or
 - Exclusive: in exactly 1 cache.
- If a read miss occurs on the bus to a slot in exclusive state:
 - owning cache makes it shared
 - Forces subsequent write to require exclusive ownership
- States map to: invalid, valid (and clean), dirty states in a uniprocessor cache.