MOSI protocol

For those of you who have any doubts about the implementation of MOSI protocol, MOSI protocol can just be derived as an up-gradation from MSI and/or without the E state in MOESI protocol. All the states in MOSI are exactly the same as in MOESI.

Processor Side Transactions:

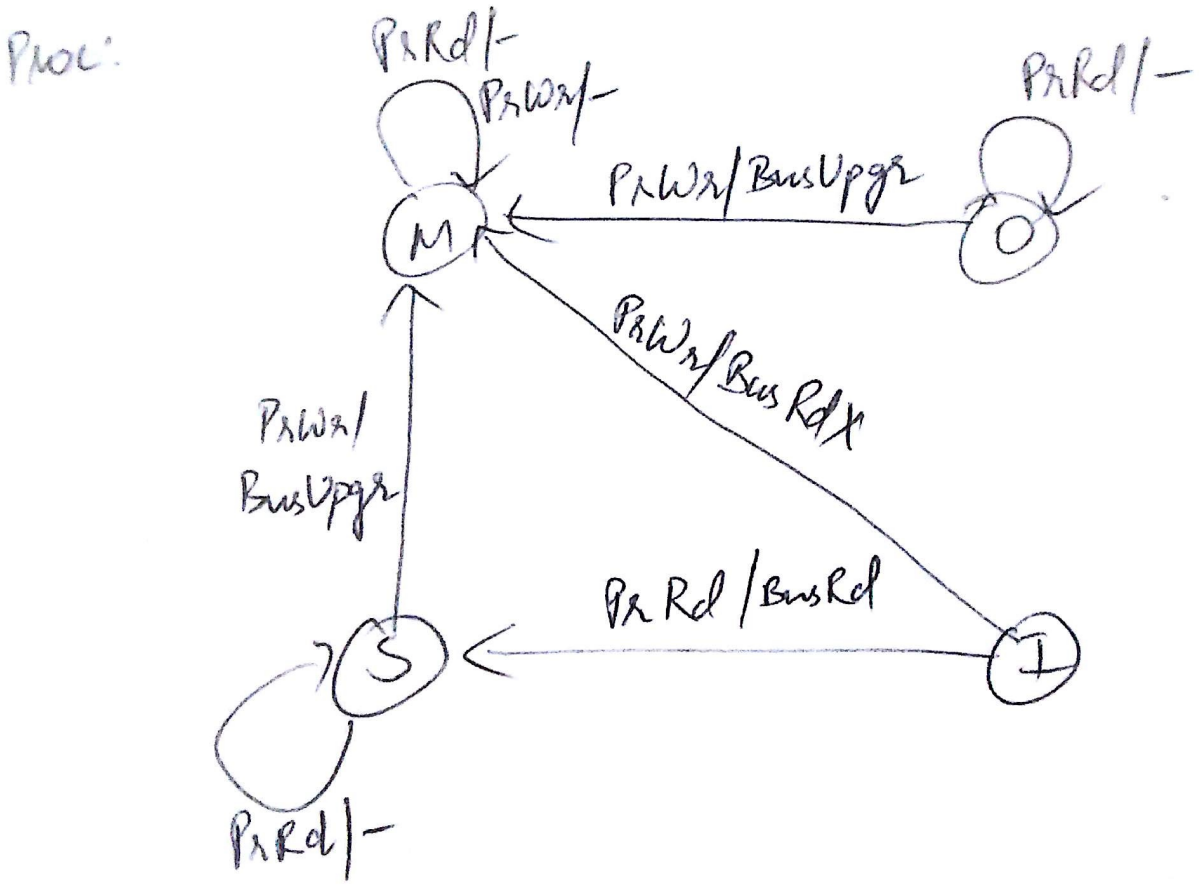
1. When there is no block in the cache in the beginning, since we don't have an E state, on a PrRd, we move from I to S always and send a BusRd signal and move from I to M on PrWr and send a BusRdX signal.
2. If the processor is in S state, on a PrRd request, the clean block is cached. So, you do nothing on the bus; but if PrWr request is received, we move from S to M and send a BusUpgr signal so that other processors can invalidate their soon to be stale cached blocks.

3. If the processor is in O state, on a PrRd request, the latest dirty block is cached within the processor itself. So, we generate no bus transactions; but if PrWr request is received, we move from O to M and send a BusUpgr signal so that other processors can invalidate their soon to be stale cached blocks. Note that the cache block is not getting evicted here, we are just going to write to an already existing dirty block. So, we dont worry about writing back to the main memory.
4. If the processors is in the M state, there is no need to generate any bus transaction on PrRd/PrWr requests.

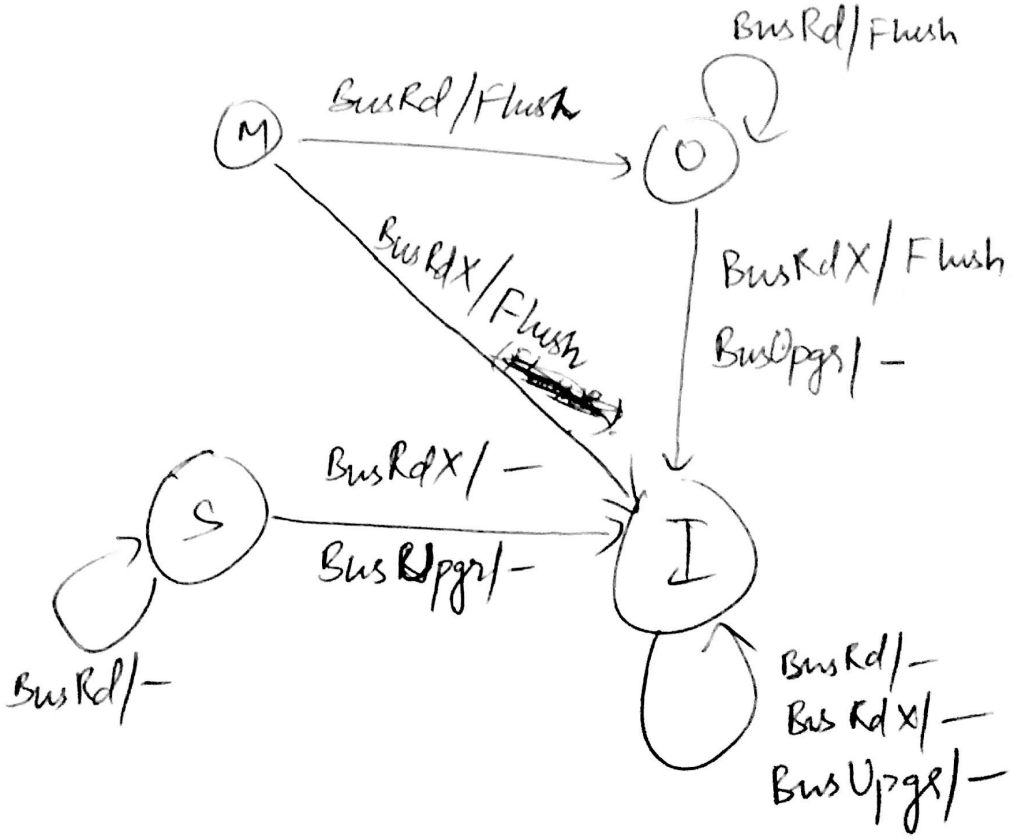
Bus Side Transactions:

1. If the processor is in I state, there is nothing to do on snooping any of the bus transactions.
2. If in M state, on a BusRd, processor moves from M to O and flushes the dirty data for the requesting processor. But still there is no eviction of its own data yet. It keeps the dirty status with itself so that it acts as a sole supplier of this cache block to other caches on any further BusRds. On eviction of the cache block from this state, it needs to write back to the main memory. On sensing a BusRdX on the other hand, it will flush the data and move to I state because a different processor will write to this block and have exclusive ownership. You will never sense a BusUpgr request from other processors as they will not have the block.
3. When in O state, on sensing a BusRd, the processor needs to flush the dirty block for other processors to read it. On a BusRdX, it has to write this block back to the main memory and also supply to the requesting processor and transition from O to I.On a busUpgr, it just moves from O to I.
4. When is S state, on sensing a BusRd, this processor does nothing as a clean block will either come from main memory or other processors may have a dirty block (in M or O state) and they will act as a supplier of the block. On sensing a BusUpgr/BusRdX, the processor simply moves from S to I.

State Diagram:



Snapes:



Side note on c2c_supplier function in main.cc :
If you notice the state diagrams of MOSI and MOESI, you can see that when the processors are not in S or I state, they need to share the cache block from one processor to the other.

project2

Updated 6 days ago by Mahita Nagabhiru

followup discussions *for lingering questions and comments*

Resolved Unresolved

Anonymous 2 days ago
Professor,

Could this note be given to the class as a handout. It would help in quicker reference.