CS-307 Project

Lab 5: Designing an MSI coherence protocol implementation

Goal: design a hardware that implements an MSI coherence protocol implementation.

Task

You will create cache controller state machine specifications for a NIOS2 bus-based multiprocessor implementing an MSI coherence protocol. You will start from a baseline non-coherent design. You will also specify a new state machine, the snoop state machine that performs snoops on the bus, invalidating and writing blocks back to memory when necessary. The baseline state-machines can be found in the same folder of this document, in a file named “`state-machine.pdf`”.

Differently from the previous design phases, this time you will not design the datapath, and the state machine description can be less precise than before. Please refer to the baseline state machine for an example.

MSI Coherence protocol

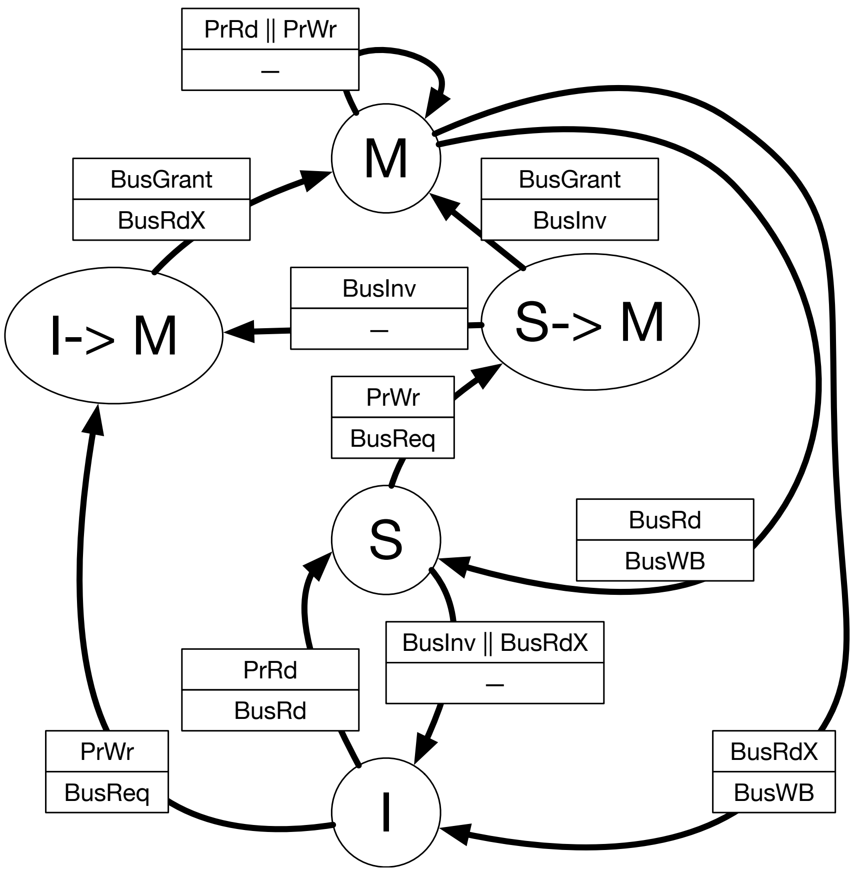

Figure 1 MSI Protocol with intermediate states

Figure 1 shows the MSI protocol with its intermediate states. You must observe the following details when implementing it:

- The baseline implements a write-through, write-back hybrid cache. In the previous design, we removed the write-back behavior, and now we are going keep the write-back behavior, removing the write-through behavior. Thus, the cacheWrite operations that miss will perform a busRdX operation, and mark the received block as (M)odified upon completion of the busRdX.
- Additionally, the write state machine will exhibit the same behavior as the read state machine, namely, it must write back dirty blocks that are evicted to make place for incoming blocks in the M state.
- In the previous task, you implemented a VI protocol, and you had to observe cases when invalidations arrived concurrently with processor operations. This time, you must observe those cases, and also the case where invalidations arrive while the cache controller is waiting for the bus. These situations were studied in class as "transient states". You may observe them in Figure 1, in the `I->M` and `S->M` states. You can implement a transient state by changing the behavior of the state that waits for the bus grant in the state machines ("wait bus grant" states). Additionally, you must mind that, whenever the cache controller state machine needs to perform a write back, it must coordinate with the snoop state machine so that they don't both write-back the same block twice.
- Finally, you must implement write-backs triggered by bus operations. This feature is different than what we did before because it is a bus operation triggered by the snoop state machine. The snoop state machine will write blocks back whenever there is a busRdX to a resident block in the M state. Furthermore, make sure you show how the snoop and the cache controller state machine interact, when both need to use the bus. For simplicity, you may assume that the tag array has one port for each state machine.

Deliverables

You will hand in the specifications for the cache controller state machine, that may be broken down into a write and a read state machine. You will also hand in the specifications for the snoop state machine. There is no need to hand in the datapath specification or the specification of the bus controller.