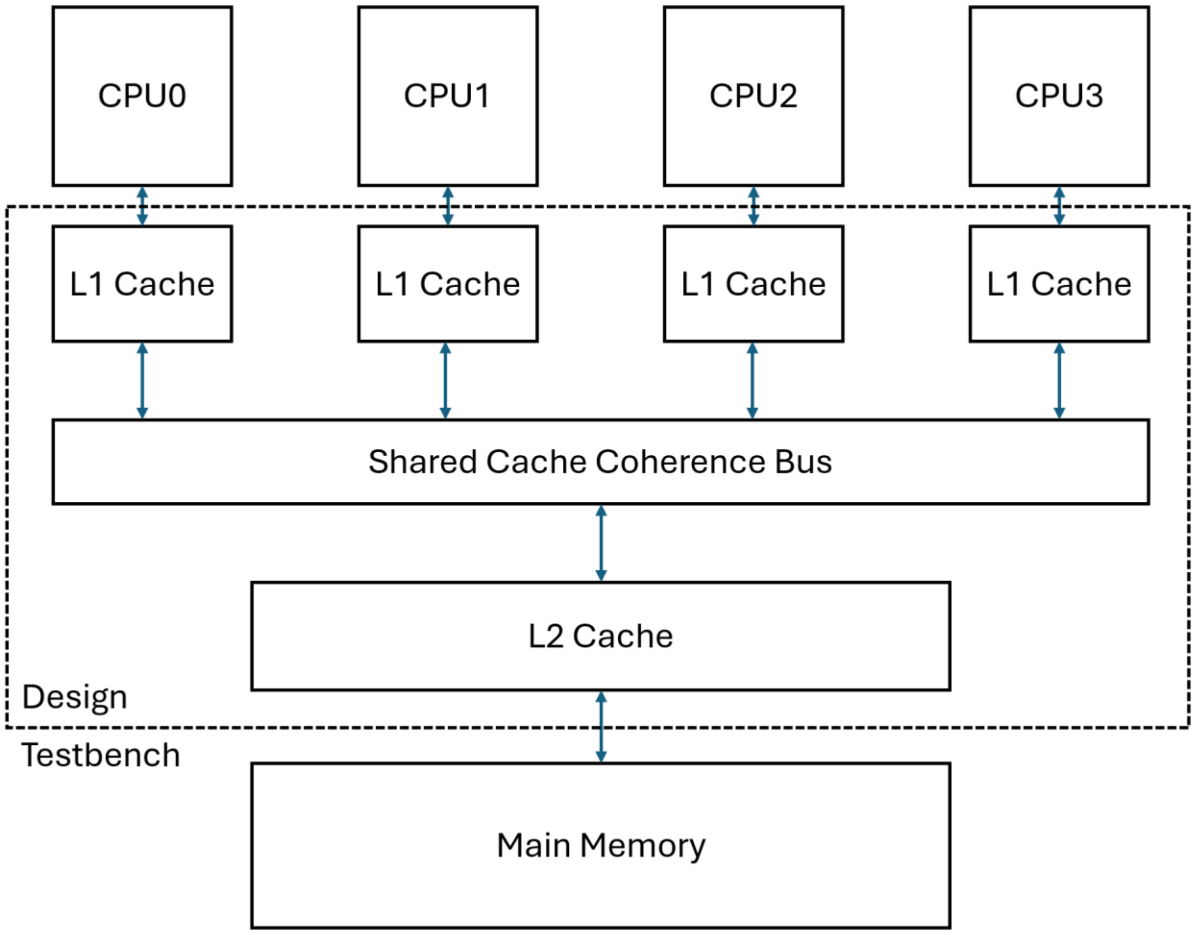
Project Title: Multicore Cache System

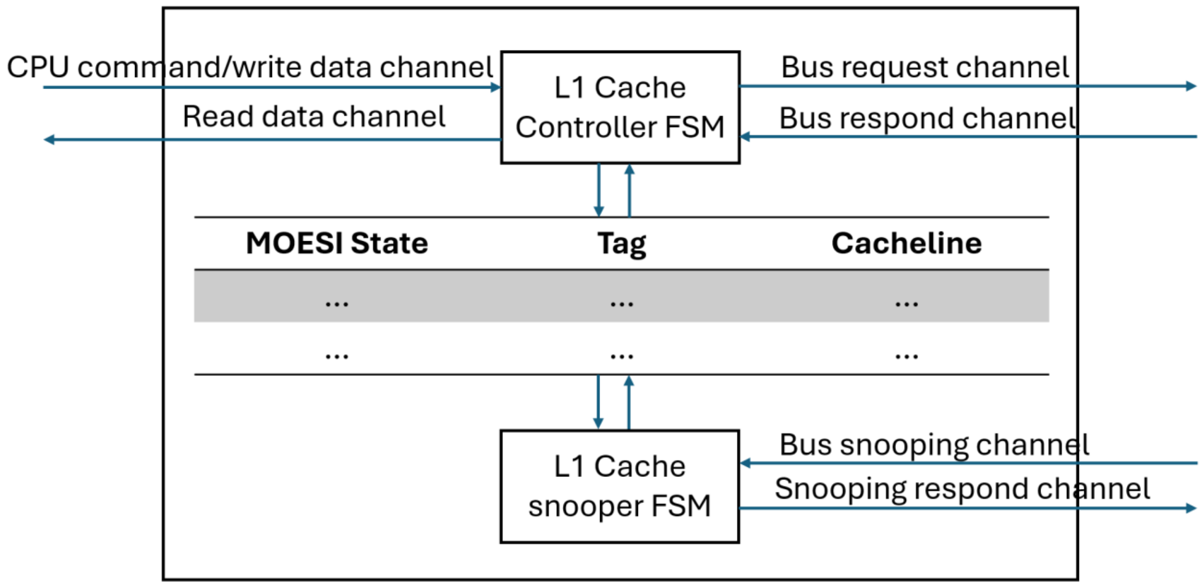
Implementing a MOESI cache coherence mechanism to manage shared memory in a multicore system.

* I'm assuming our design won't get taped out, so I'll use more inputs and outputs as well as more gate budget.

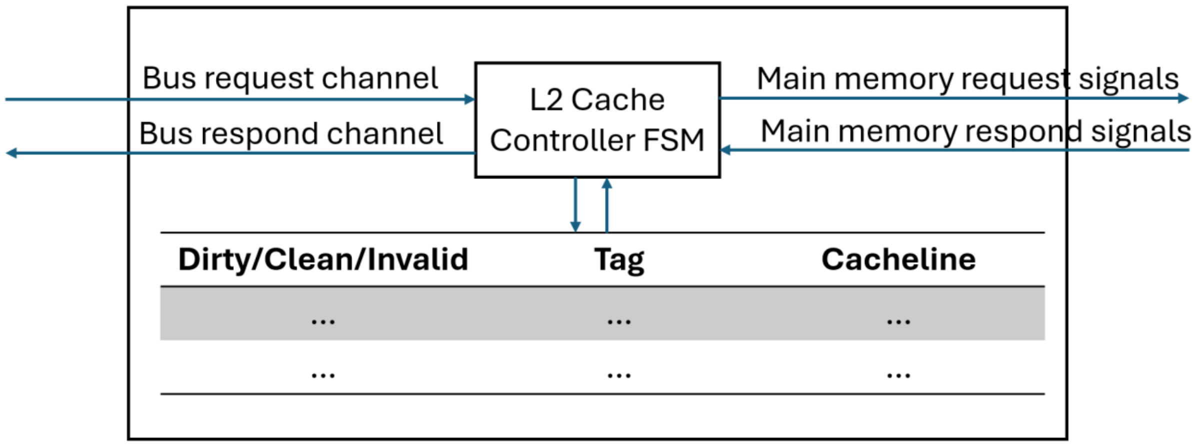
Datapath Schematic:



Overall Schematic

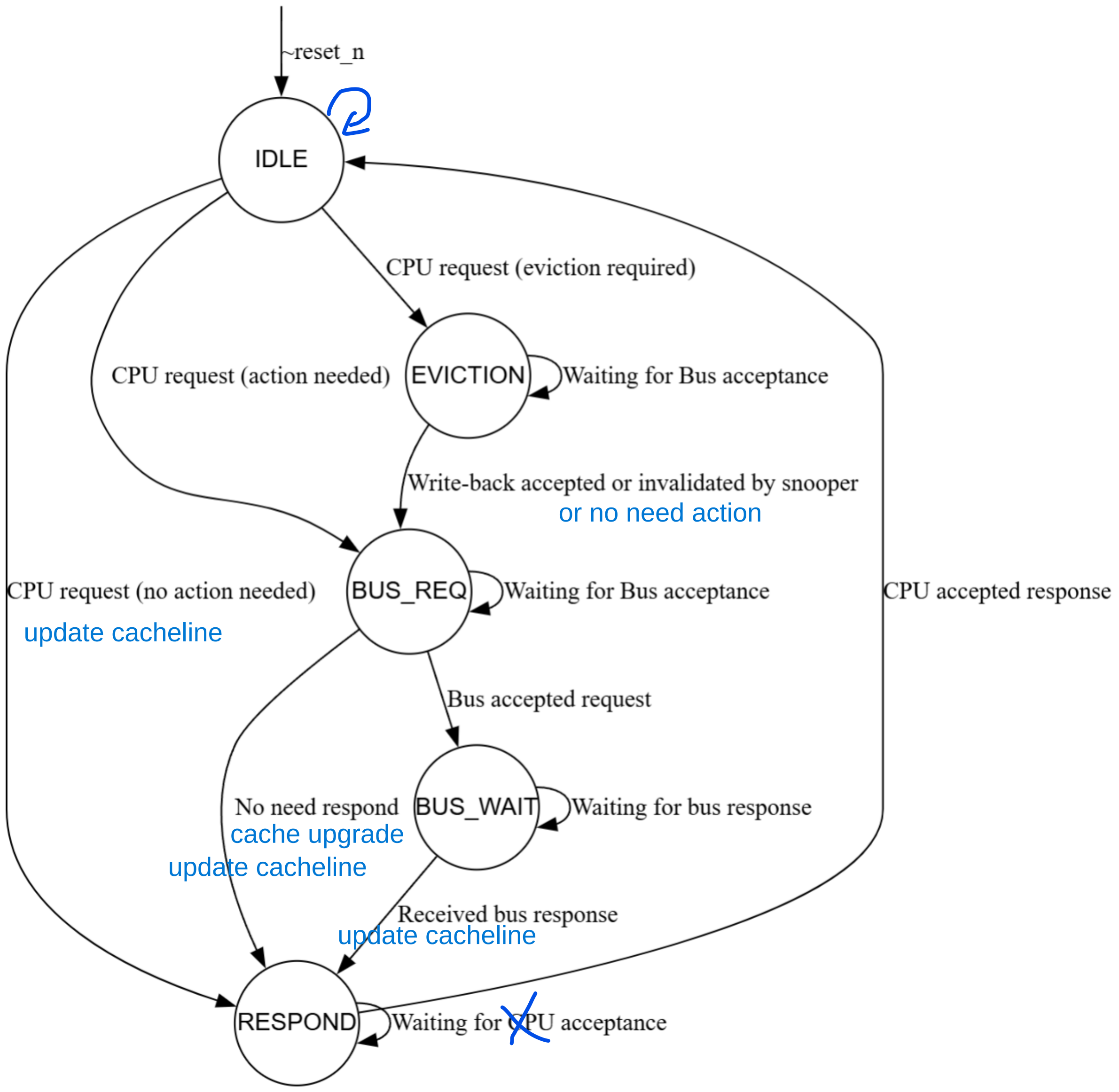


L1 Cache Schematic

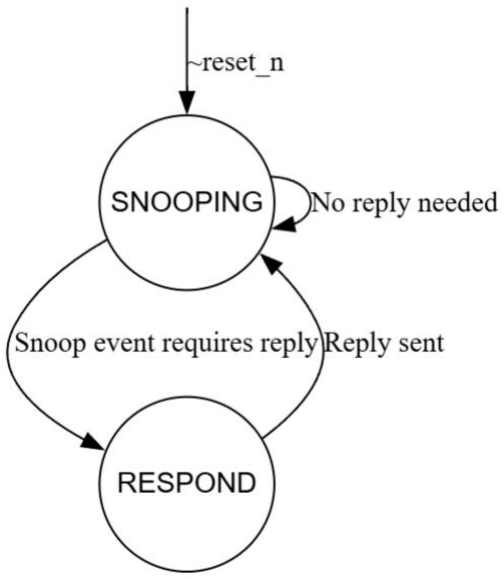


L2 Cache Schematic

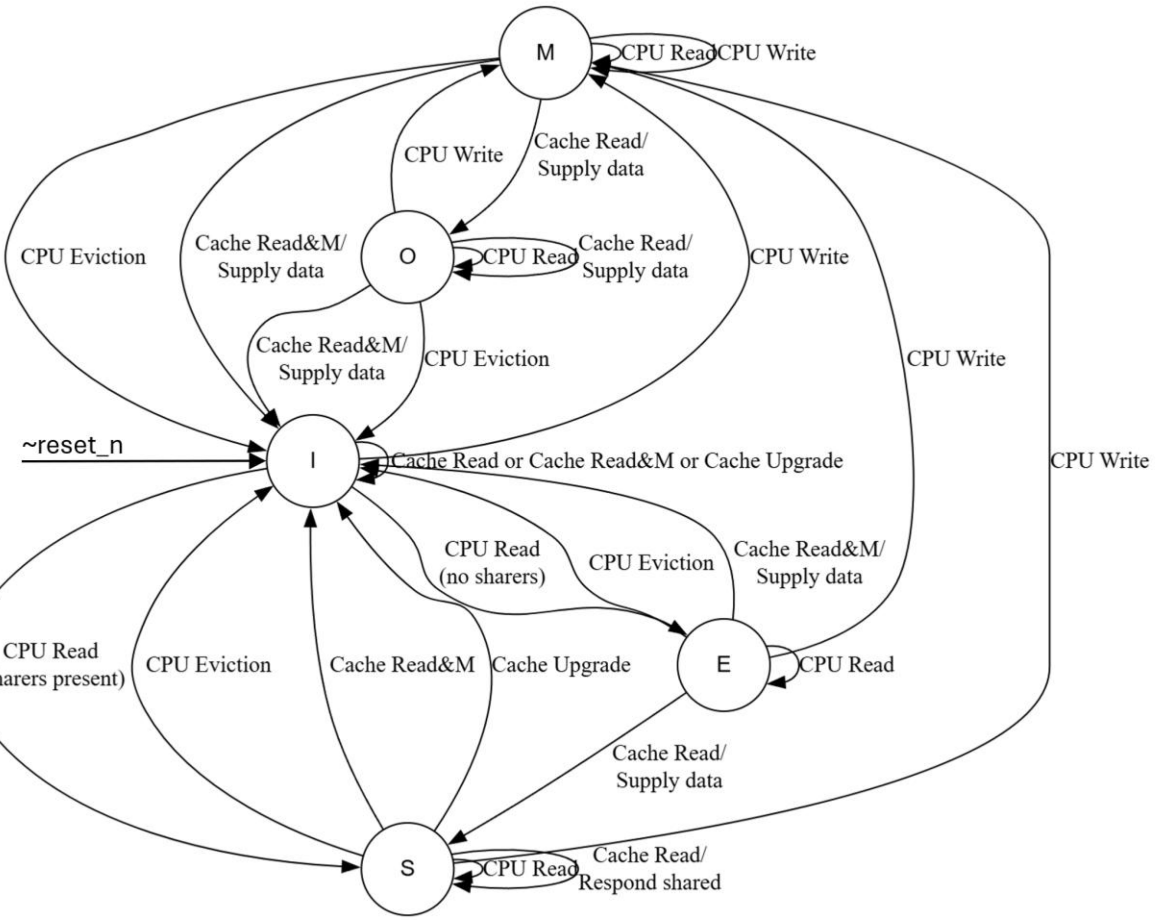
FSM State Transition Diagram:



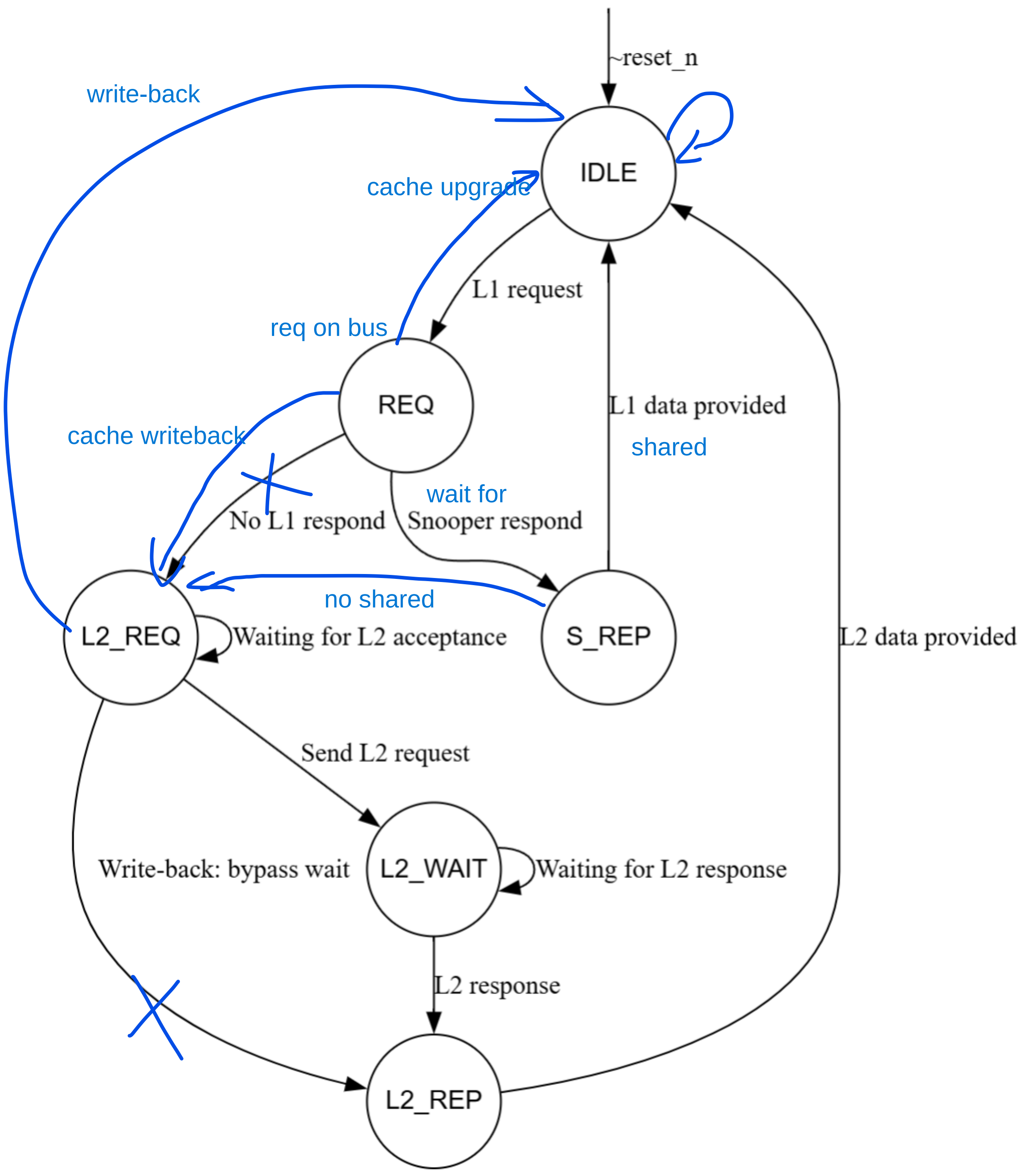
L1 Cache Controller FSM



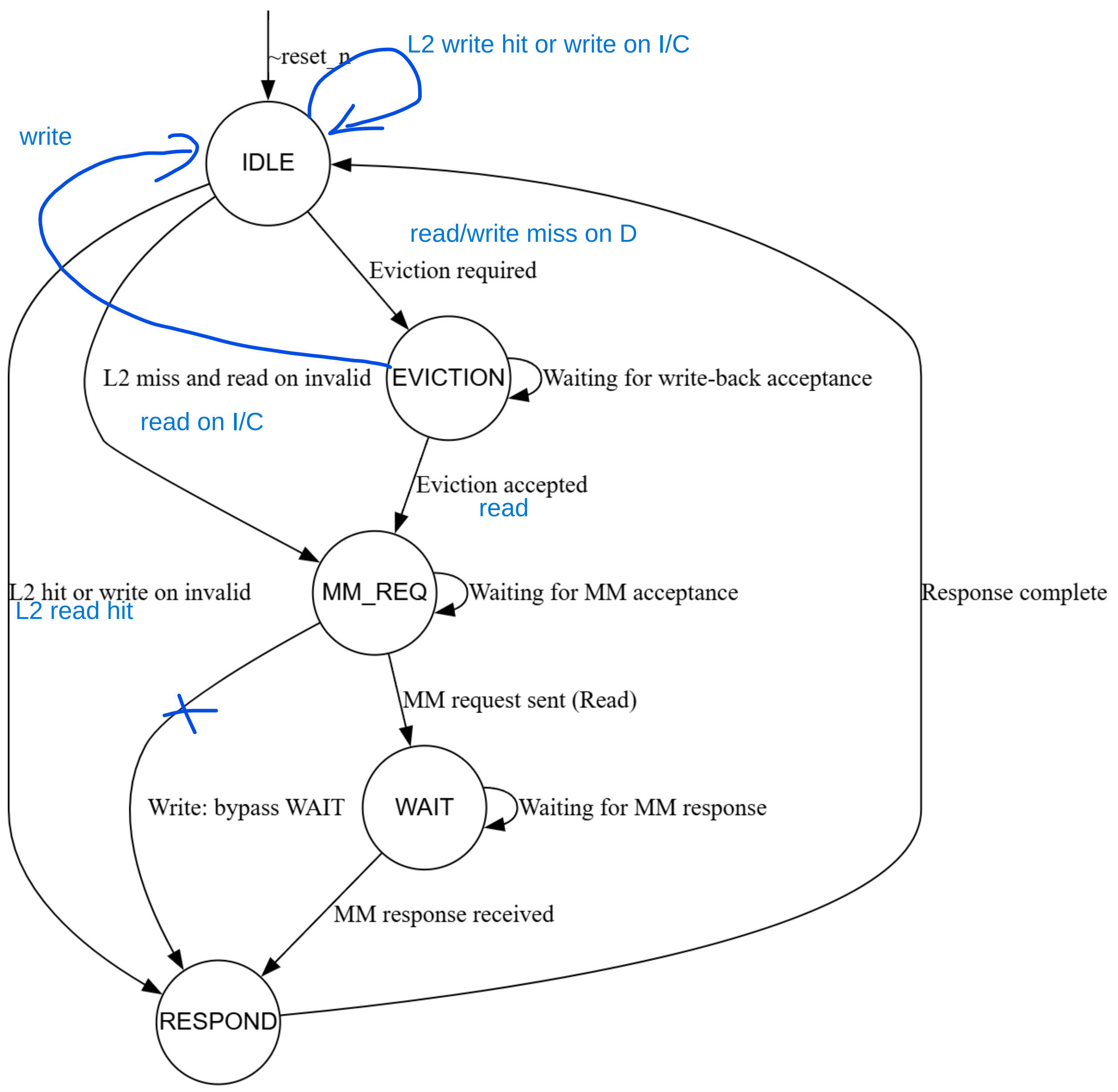
L1 Cache Snooper FSM



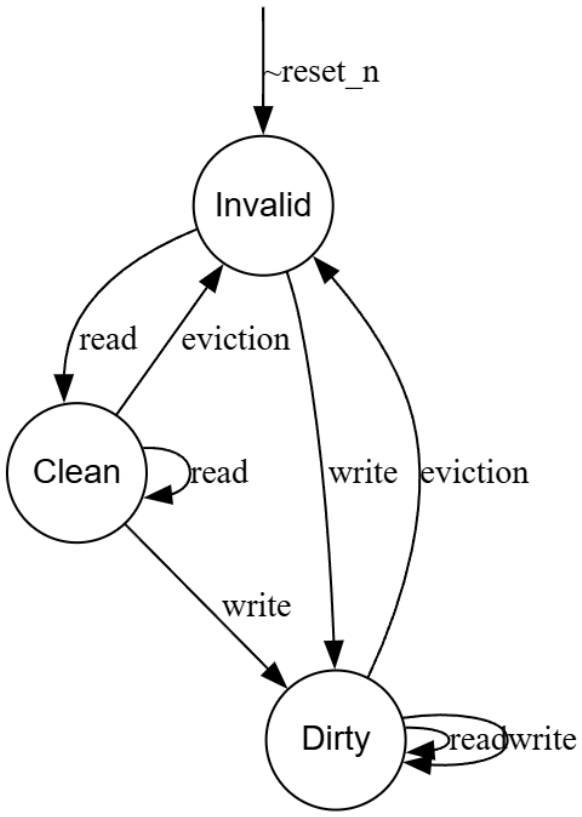
L1 Cache MOESI Protocol



Shared Cache Coherence Bus FSM



L2 Cache Controller FSM



L2 Cache States

Testbench Strategies:

In my testbench, I will instantiate four simulated CPUs. Each CPU will independently issue randomized sequences of load/store instructions to its own L1 cache. CPUs will generate both deterministic and random patterns to fully test various cache coherence scenarios. A simulated main memory will connect externally to the L2 cache, with a fixed latency (e.g., 10 cycles per access) to emulate real DRAM delays.

To verify correctness, the testbench will continuously monitor outputs to ensure each CPU sees results identical to those from a memory system without caches. To achieve that, I'll compare my cache design with a reference system that directly accesses main memory for correctness check. Additionally, the testbench will track the total number of simulation cycles, providing insight into the performance of my cache hierarchy.