Cache Coherence Design: A Report

Debleena Das

March 11, 2024

Abstract

1 Introduction

With a view to design PMSI, we are designing a write invalidate snoopy based system with write back cache.Here in this Document we are designing a system that is aim to have multi core connected via interconnection network. In a multi core system, single core with L1 private cache are connected by a shared bus.

2 component of the Multi core

2.1 Single core

2.1.1 processor: PICORV32

2.1.2 cache: L1

The single core is consisted of PICORV-32 processor, connected with l1 cache which comprises of 2-way set associative cache.

2.2 Interconnection Network:

Interconnection is the point of serialization of ordering requests from cache controller. All cache controller must be able to see the messages from all of the cache controller.This is the responsibility of the interconnect to maintain the ordering of the request. —Primer 20202.

The interconnection Network must be able to broadcast the requests to all of the participating coherence controller. The broadcast network must ensure the total order of coherence requests to guarantee correct updating of the cache-block's state. In this design we have used a bus as the broadcast medium.

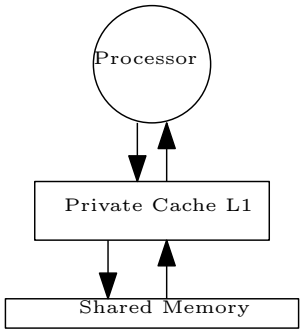

Figure 1: single core with shared memory .

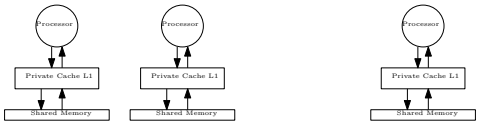

Figure 2: n no. of single core

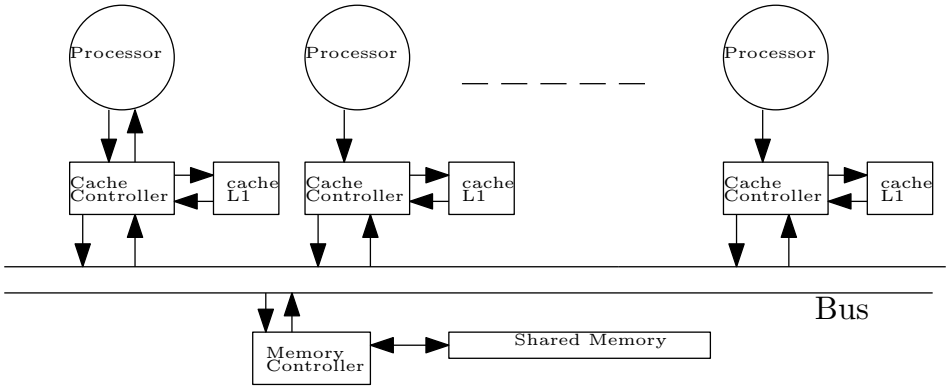

Figure 3: Multi core architecture

2.3 Shared Memory:

Here Main Memory is the shared Memory. In this design, we have considered 512 lines of data , each of 32 bits. One clock cycle is required to read or write the data in memory.

2.4 multi core

In multi core design, N no. of single core $c_0, c_1, ...c_n$ are connected to a shared interconnect with shared memory.

2.5 Multi core operations

In a Multi core architecture when a cache gets a read/write Miss, it search for the data in other core to ensure that no one is having a modified data. If any one of the core has the data in Modified state, it is ensured by the coherence controller of that core to write it in memory before moving into further communication. The following table illustrates the cache operation, MSI state change and the corresponding coherence requests:

From the 1, A core can put 3 types of requests to the bus:

- Write_invalidate : when a core gets a write hit in its local cache(L1) on shared block, then it broadcast a write_invalidate requests, to inform other cores that it has updated the block, other core must invalidate their copy. If the core has the data in Modified state, then it can update the data, due to Single Write Multiple Read principle, only One core can have data in Modified state.

cache operation	pr_st current	pr_state next	bs_st current	bs_st next	bus request
Read hit	Shared	Shared	shared	shared	No Request
Read hit	Modified	Modified	Invalid	Invalid	No Request
Write hit	Shared	Modified	shared	Invalid	write_Invalidate
Write hit	Modified	Modified	Invalid	Invalid	No requests
Read Miss	Invalid	Shared	shared	shared	Bus_Read
Read Miss	Invalid	Shared	Modified	shared	Bus_Read
Write Miss	Invalid	Modified	shared	Invalid	Bus_Write
Write Miss	Invalid	Modified	Modified	Invalid	Bus_Write

Table 1: MSI state change .

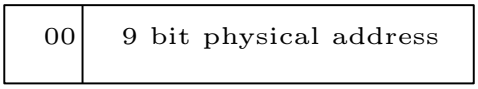

Figure 4: request message for write_invalidate .

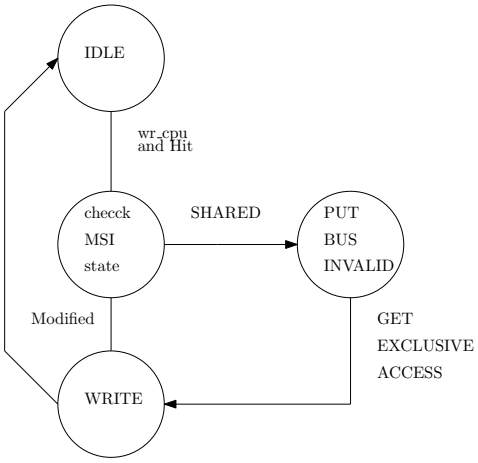

Figure 5: FSM of Write Hit opeartion .

So, if the current core has the data in Modified state, no request should be put for invalidation. Request Message: 4

If A core does not have the block in its local(L1) cache, miss occurs and the coherence controller check the presence of the block in other cores in Modified state connected to that interconnection network by putting either bus_rd or bus_wr requests. As in this design we are not considering cache to cache transfer, so we only look for Modified data in other core, and wait for write back the data in memory and then will access the data from memory.

- bus_rd: when a core gets a read miss in L1, it asks the other cores by putting the bus_rd request in the interconnection network. Request Message: 6
- bus_wr: when a core gets a read miss in L1, it asks the other cores by putting the bus_wr request in the interconnection network. Request Message: 8

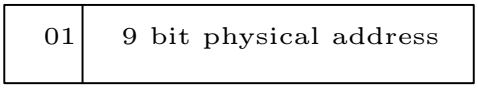

Figure 6: request message for read miss .

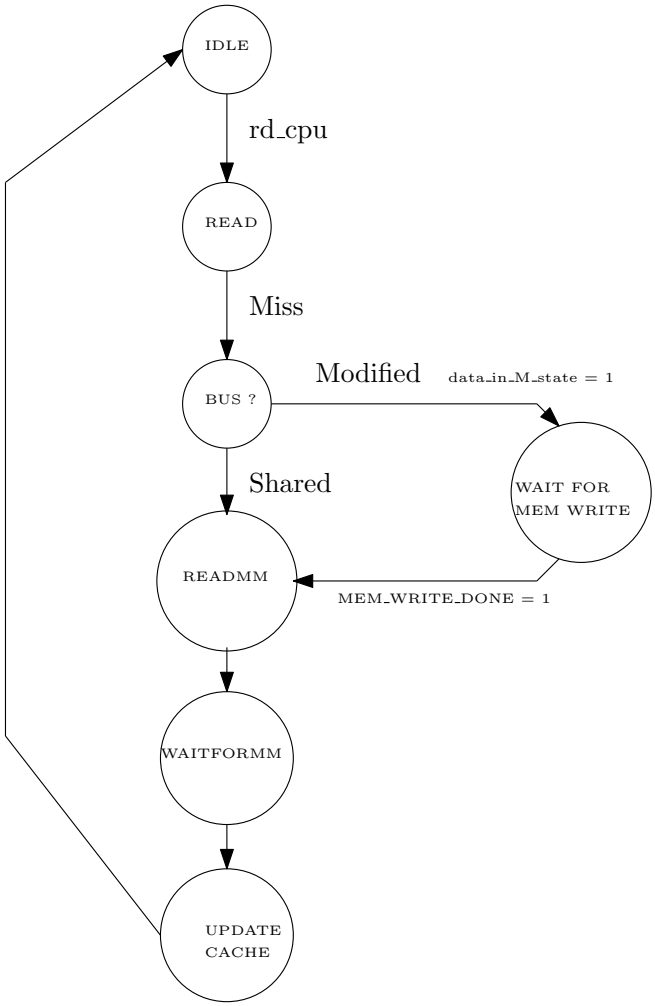

Figure 7: FSM of Read opeartion .

10	9 bit physical address

Figure 8: request message for write miss .

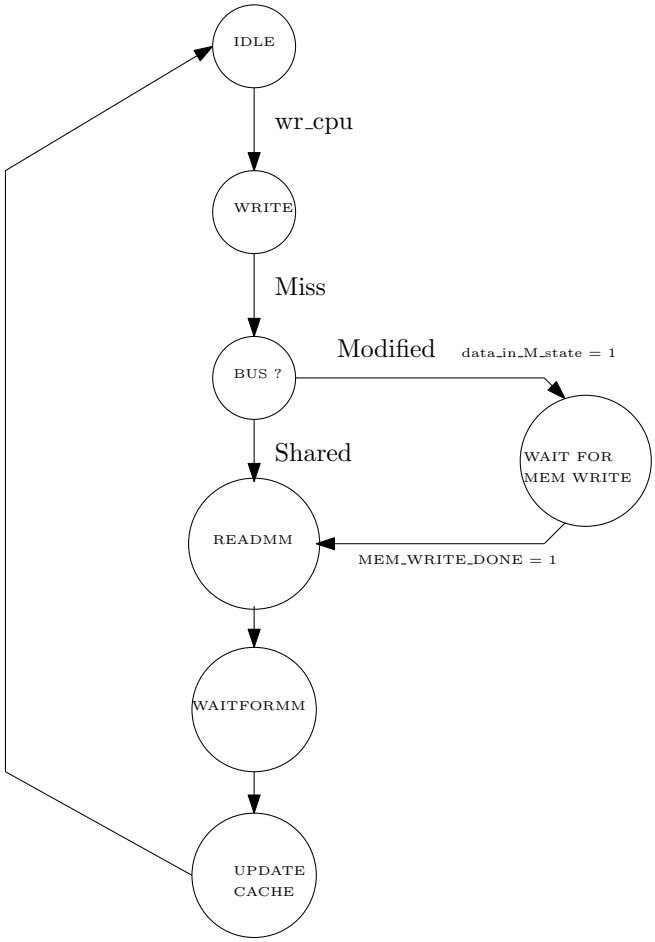

Figure 9: FSM of Write opeartion .