Zexi Pan 1000138246
Xiaoyang Guo 999578513

Lab 5 Report

Prelab

Q1) Why are transient states necessary?

Transient states are necessary because state changes do not happen atomically. This is because there are delays that can occur as a state is changing. While this request is delayed, another request can be issued and completed. Therefore transient states are needed as an intermediate stage as data is being forwarded and acknowledged and ensure atomicity.

Q2) Why does a coherence protocol use stalls?

A coherence protocol uses stalls to enforce the two cache coherence invariants. Only a single writer is only allowed at one time, therefore a stall would be to stall a second writer until the first finished. Furthermore to ensure actions are completed atomically, stalls are needed to delay other requests.

Q3) What is deadlock and how can we avoid it?

A deadlock are cyclic dependencies where two events are waiting for the other to complete in order to progress. This can be caused by using one resource for different message types. The commonly used solution is a virtual network (i.e. separating networks into incoming/outgoing queues and buffers).

Q4) What is the functionality of Put-Ack (i.e., WB-Ack) messages? They are used as a response to which message types?

The functionality of Put-Ack is to inform the core that a writeback was successful. The core can then invalidate that block. They are used as a response message to the message type Put_S and Put_M.

Q5) What determines who is the sender of a data reply to the requesting core? Which are the possible options?

The sender of a data reply is determined by the state of the requested block.

- M: Sender is the cache
- S: Sender is the cache
- I: Sender is the directory

The possible options are the cache and the directory.

Zexi Pan 1000138246
Xiaoyang Guo 999578513

Q6) What is the difference between a Put-Ack and an Inv-Ack message?

Put-Ack indicates to the core that a writeback has just occurred successfully. Inv-Ack is used to inform other caches that the local block has been invalidated.

Section 5

Q1) How does the FSM know it has received the last Inv Ack reply for a TBE entry?
The directory controller has the information of which core is sharing which block. It tracks the total number of sharers and how many Inv Ack messages to expect.

Q2) How is it possible to receive a PUTM request from a NonOwner core? Please provide a set of events that will lead to this scenario.

Initial condition: Core 1 is the owner of a Modified block A which is in state M on Core 1 cache and Directory.

1: Core 2 issues a GETM request on block A to Directory

2: Core 1 replaces the block A and issues PUTM to Directory

3: Core 2's GETM request received, the directory sends Fwd_GETM to A's owner

Core1 and updates Core 2 as the new owner of A

4: Core 1's PUTM request received, so the Directory sees a PUTM request from a

NonOwner core

Q3) Why do we need to differentiate between a PUTS and a PUTS_Last request?

The directory controller must keep the block until it receives PUTS_Last.

Q4) How is it possible to receive a PUTS_Last request for a block in modified state in the directory? Please provide a set of events that will make this possible.

Initial condition: block A is in S state on Core 1. (Last sharer)

1: Core 1 issues a PUTS_Last request to Directory to evict block A

2: Core 2 issues a GETM request to Directory for accessing block A

3: Core 2's GETM request reaches Directory first, the Directory sends Inv messages to each Core that has A (including Core 1) and changes A from S to M state.

4: Finally, Core 1's PUTS_Last request arrived, so block A in directory received a PUTS_Last request in M state

Q5) Why is it not possible to get an Invalidation request for a cache block in a Modified state? Please explain.

Since the M state cache block has only one owner, it does not make sense if the directory receives a GETM request, and send invalidation request. Such request will be forward to the owner.

Q6) Why is it not possible for the cache controller to get a Fwd-GetS request for a block in SI_A state? Please explain.

Because when the directory receives a GetS request for a block in M state, it will issue a Fwd_GetS request to the owner. It cannot be the owner is the block was in SI_A state. The block was in S state.

Q7) Was your verification testing exhaustive? How can you ensure that the random tester has exercised all possible transitions (i.e., all {State, Event} pairs)?

It was not exhaustive. It is unlikely to test with all cases. We tried different combinations of core number and cache size with a large amount of instructions. This can cover almost all the cases but there will be corner cases untested.

Zexi Pan 1000138246
Xiaoyang Guo 999578513

Modification to Table 8.1

State	Trigger Event	Modification
I	Inv	Send Inv_Ack

Modification to Table 8.2

State	Trigger Event	Modification
I	GETS, GETM	New transient state: IS_D, IM_D
MI_A	Mem_Ack	Inform requestor that WB is done
M	PUTM	New transient state: MI_A
MS_D	Cache_Data	New transient state: MS_A
MS_A	Mem_Ack	Clear owner, go to S
MS_D, MS_A	PUTS_LAST	stall
IM_D, IS_D, MS_D, MS_A, MI_A	GETM, GETS	stall
IM_D, IS_D	Mem_Data	Cody data to directory, forward data to requestor
MS_D, MS_A	PUTM	Send back Put_Ack, remove from sharer list.
M	GETS	New transient state: MS_D

Workload Breakdown:
Directory states: Xiaoyang Guo
Cache states: Zexi Pan
Testing: Zexi Pan, Xiaoyang Guo
Lap Report: Zexi Pan, Xiaoyang Guo