ECE552 Lab Assignment 5: Coherence Report
Student: Daokun Chen 998699813 Ruyu Fan 999586363

Section 5 Questions

Q1. How does the FSM know it has received the last Inv Ack reply for a TBE entry?
Ans: Because the directory controller has the information of all the cores sharing a specific block, it knows 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.
Ans: Initially, Core 1 is the owner of a Modified block A. So, A is in state M on Core 1 cache and Directory. The following events occur over time:

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?
Ans: Because if there are no more copy of a block in cache, the directory controller can invalidate the block. In other word, 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.
Ans: Initially, 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.
Ans: Since the M state cache block has exactly one owner and no sharers, there will never be a case that the directory receives a GETM request, but Inv is sent. The 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.
Ans: Because when the directory receives a GetS request for a block in M state, it will issue a Fwd_GetS request to the owner of that block. However, if the block is in SI_A state, it cannot be the owner. 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)?
Ans: The verification test we have done was not exhaustive. It's very difficult to test all possible transition cases. What we have done is just increase the number of instructions, increase the number of cores and varying the cache sizes. This definitely increases the variety of transition cases that occur in the testing. However, it still cannot guarantee that all possible transition cases will occur.

PreLab Questions

Q1. Why are transient states necessary?
Ans: Transient states are intermediate states between stable states (MSI). In reality, state changes will not happen atomically, thus the transient states are introduced to handle the intermediate states. For instance, the need to wait for data forwarding/acknowledge, the block must wait in transient states.

Q2. Why does a coherence protocol use stalls?
Ans: To enforce the rules of coherence protocol, stalls are the actions taken to handle certain events. For example, one core is writing to a block, while another core is trying to access the same block. It has to stall until the first core finished. This is the place where coherence protocol ensure data invariant and SWMR invariant.

Q3. What is deadlock and how can we avoid it?
Ans: Deadlock is the case where two inter-dependent events holds against each other, none of them can make progress. It can be avoided by introducing virtual networks. The virtual networks manage messages by distinguish their types so that different types of message communication won't affect each other.

Q4. What is the functionality of Put-Ack (i.e., WB-Ack) messages? They are used as a response to which message types?
Ans: The Put_Ack message is used by the directory to inform the core that a data writeback was successful. The core can safely invalidate the block. This is the response message of Put_S/Put_M.

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

Ans: The block states determines who is the sender of data. If it is in M state, the owner will be the sender, if it is in S state, the cache has the data will be the sender, else if it is in I state, the directory will be the sender. Thus, the directory or one of the caches would be possible.

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

Ans: As described in Q4, Put_Ack message indicates a writeback have just happened, and the data block are invalidated. However, Inv_Ack message are used to inform other caches that the data block are invalidated in my local cache.

Modifications On Table 8.1

State	Trigger Event	Modification
I	Inv	Send Inv_Ack

Modification On Table 8.2

State	Trigger Event	Modification
I	GETM/GETS	Add new transient state IM_D/IS_D
IS_D, IM_D, MS_D, MS_A, MI_A	GETM/GETS	Stall
IM_D, IS_D	Mem_Data	Send data to requestor, copy data to director
M	GETS	Add new transient state MS_D
MS_D	Cache_Data	Add new transient state MS_A
MS_A	Mem_Ack	Clear Owner, Transient to S safely
MS_D, MS_A	PUTM+data from owner	Remove from sharer list, Send PUT_Ack back
MS_D, MS_A	PUTS_Last	Stall
M	PUTM+data from owner	Add new transient state MI_A
MI_A	Mem_Ack	Inform requestor WB is done,Transient to I safely

Work Breakdown:

Daokun Chen	**Ruyu Fan**
Directory transitions + New states design	Cache transitions
Report Writing	Report Writing
Debugging	Debugging