6.823 Computer System Architecture

Directory-based Cache Coherence Protocol

Last Updated: 11/23/2005 3:33 PM

Note: The protocol described in this handout is supposed to be consistent with that in Lecture 18 (2005). If you find any discrepancy or typo, please report to the course staff immediately.

Before introducing a directory-based cache coherence protocol, we make the following assumptions about the interconnection network:

- Message passing is reliable, and free from deadlock, livelock and starvation. In other words, the transfer latency of any protocol message is finite.
- Message passing is FIFO. That is, protocol messages with the same source and destination sites are always received in the same order as that in which they were issued.

Cache states: For each cache line, there are 4 possible states:

- C-invalid (= `Nothing`): The accessed data is not resident in the cache.
- C-shared (= `Sh`): The accessed data is resident in the cache, and possibly also cached at other sites. The data in memory is valid.
- C-modified (= `Ex`): The accessed data is exclusively resident in this cache, and has been modified. Memory does not have the most up-to-date data.
- C-transient (= `Pending`): The accessed data is in a *transient* state (for example, the site has just issued a protocol request, but has not received the corresponding protocol reply).

Home directory states: For each memory block, there are 4 possible states:

- R(*dir*): The memory block is shared by the sites specified in *dir* (*dir* is a set of sites). The data in memory is valid in this state. If *dir* is empty (i.e., *dir* = ε), the memory block is not cached by any site.
- W(*id*): The memory block is exclusively cached at site *id*, and has been modified at that site. Memory does not have the most up-to-date data.
- T_R(*dir*): The memory block is in a transient state waiting for the acknowledgements to the invalidation requests that the home site has issued.
- T_W(*id*): The memory block is in a transient state waiting for a block exclusively cached at site *id* (i.e., in C-modified state) to make the memory block at the home site up-to-date.

Protocol messages: There are 10 different protocol messages, which are summarized in the following table (their meaning will become clear later).

Category	Messages
Cache to Memory Requests	`ShReq, ExReq`
Memory to Cache Requests	`WbReq, InvReq, FlushReq`
Cache to Memory Responses	`WbRep(v), InvRep, FlushRep(v)`
Memory to Cache Responses	`ShRep(v), ExRep(v)`

No.	Current State	Handling Message	Next State	Dequeue Message?	Action
1	C-nothing	Load	C-pending	No	ShReq(id,Home,a)
2	C-nothing	Store	C-pending	No	ExReq(id,Home,a)
3	C-nothing	WbReq(a)	C-nothing	Yes	None
4	C-nothing	FlushReq(a)	C-nothing	Yes	None
5	C-nothing	InvReq(a)	C-nothing	Yes	None
6	C-nothing	ShRep (a)	C-shared	Yes	updates cache with prefetch data
7	C-nothing	ExRep (a)	C-exclusive	Yes	updates cache with data
8	C-shared	Load	C-shared	Yes	Reads cache
9	C-shared	WbReq(a)	C-shared	Yes	None
10	C-shared	FlushReq(a)	C-nothing	Yes	InvRep(id, Home, a)
11	C-shared	InvReq(a)	C-nothing	Yes	InvRep(id, Home, a)
12	C-shared	ExRep(a)	C-exclusive	Yes	None
13	C-shared	(Voluntary Invalidate)	C-nothing	N/A	InvRep(id, Home, a)
14	C-exclusive	Load	C-exclusive	Yes	reads cache
15	C-exclusive	Store	C-exclusive	Yes	writes cache
16	C-exclusive	WbReq(a)	C-shared	Yes	WbRep(id, Home, data(a))
17	C-exclusive	FlushReq(a)	C-nothing	Yes	FlushRep(id, Home, data(a))
18	C-exclusive	(Voluntary Writeback)	C-shared	N/A	WbRep(id, Home, data(a))
19	C-exclusive	(Voluntary Flush)	C-nothing	N/A	FlushRep(id, Home, data(a))
20	C-pending	WbReq(a)	C-pending	Yes	None
21	C-pending	FlushReq(a)	C-pending	Yes	None
22	C-pending	InvReq(a)	C-pending	Yes	None
23	C-pending	ShRep(a)	C-shared	Yes	updates cache with data
24	C-pending	ExRep(a)	C-exclusive	Yes	update cache with data

Table H12-1: Cache State Transitions

No.	Current State	Message Received	Next State	Dequeue Message?	Action
1	R(dir) & (dir = ε)	ShReq(a)	R({id})	Yes	ShRep(Home, id, data(a))
2	R(dir) & (dir = ε)	ExReq(a)	W(id)	Yes	ExRep(Home, id, data(a))
3	R(dir) & (dir = ε)	(Voluntary Prefetch)	R({id})	N/A	ShRep(Home, id, data(a))
4	R(dir) & (id ∉ dir) & (dir ≠ ε)	ShReq(a)	R(dir + {id})	Yes	ShRep(Home, id, data(a))
5	R(dir) & (id ∉ dir) & (dir ≠ ε)	ExReq(a)	Tr(dir)	No	InvReq(Home, dir, a)
6	R(dir) & (id ∉ dir) & (dir ≠ ε)	(Voluntary Prefetch)	R(dir + {id})	N/A	ShRep(Home, id, data(a))
7	R(dir) & (dir = {id})	ShReq(a)	R(dir)	Yes	None
8	R(dir) & (dir = {id})	ExReq(a)	W(id)	Yes	ExRep(Home, id, data(a))
9	R(dir) & (dir = {id})	InvRep(a)	R(ε)	Yes	None
10	R(dir) & (id ∈ dir) & (dir ≠ {id})	ShReq(a)	R(dir)	Yes	None
11	R(dir) & (id ∈ dir) & (dir ≠ {id})	ExReq(a)	Tr(dir-{id})	No	InvReq(Home, dir - {id}, a)
12	R(dir) & (id ∈ dir) & (dir ≠ {id})	InvRep(a)	R(dir - {id})	Yes	None
13	W(id')	ShReq(a)	Tw(id')	No	WbReq(Home, id', a)
14	W(id')	ExReq(a)	Tw(id')	No	FlushReq(Home, id', a)
15	W(id)	ExReq(a)	W(id)	Yes	None
16	W(id)	WbRep(a)	R({id})	Yes	data -> memory
17	W(id)	FlushRep(a)	R(ε)	Yes	data -> memory
18	Tr(dir) & (id ∈ dir)	InvRep(a)	Tr(dir - {id})	Yes	None
19	Tr(dir) & (id ∉ dir)	InvRep(a)	Tr(dir)	Yes	None
20	Tw(id)	WbRep(a)	R({id})	Yes	data-> memory
21	Tw(id)	FlushRep(a)	R(ε)	Yes	data-> memory

Table H12-2: Home Directory State Transitions, Messages sent from site **id**