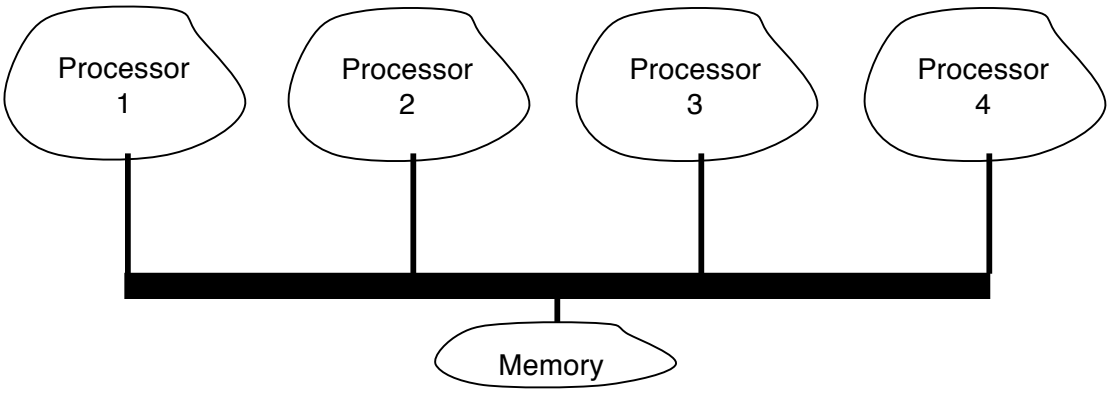
Lecture 25: Board Example: Cache Coherency

Assume that we have a centralized-shared-memory architecture with 4 processors sharing a bus; each processor has a local, direct-mapped cache:



You know the following:

- Each processor's cache is initially empty
- There are 4 states that a cache entry could be in:
    - (M) Modified, (E) Exclusive, (S) Shared, (I) Invalid
        - (This is the MESI protocol discussed in class.)
- The caches at the processing nodes snoop on the bus
- A write back, write invalidate policy is used.

The following events happen in SEQUENTIAL order:

Event #	Processor #	Event	Comment
1	2	Writes to address with tag A	
2	2	Writes to address with tag K	
3	2	Reads from address with tag Z	Address with tag Z maps to address with tag A
4	1	Writes to address with tag A	
5	4	Reads from address with tag Z	
6	3	Writes to address with tag I	Address with tag I maps to address with tag K
7	2	Reads from address with tag Q	Address with tag Q maps to address with tag Z
8	2	Reads from address with tag B	
9	2	Writes to address with tag B	
10	1	Reads from address with tag B	
11	1	Writes to address with tag K	
12	3	Reads from address with tag N	
13	3	Reads from address with tag K	

Question A:
True or False? After Event 1, A is cached in the Exclusive (E) state.

False. Because we are doing a write, A is cached in the modified state.

Question B:
After Event 3, what state is Z cached in?

Z is cached in the Exclusive (E) state as we are just doing a read.

Question C:
True or False? After Event 4, address Z's data on node 2 must be written back to main memory.

False. Even though A maps to Z, A and Z are cached on different nodes.

Question D:
After Event 5, what state is Z in on node 4?

Z is in the shared state as there are now 2 copies of Z are cached.

Question E:
True or False? Even though A maps to Z, we do not have to invalidate A on Node 1 after Event 5.

True. They are on different nodes.

Question F:
True or False? After Event 7, how many entries (blocks) on node 2's cache are invalidated?

0 or 1 both accepted.

Question F:
After Event 7, what state is Q in?

The Exclusive (E) state.

Question G:
After Event 10, what state is B in *on node 1?*

The Shared (S) state as there is another copy on node 2 as well.

Question H:
After Event 13, how many valid entries are there in node 2's cache?

2.

Event #1: P2 writes to address with tag A

	Event	Address	State
Node 1			
Node 2			
(1)	Write	A	M
Node 3			
Node 4			

Event #2: P2 writes to address with tag K

	Event	Address	State
Node 1			
Node 2			
(1)	Write	A	M
(2)	Write	K	M
Node 3			
Node 4			

Event #3: P2 reads from address with tag Z

Address with tag Z maps to same block as address with tag A

	Event	Address	State
Node 1			
Node 2			
(3)	Read	Z	E
(2)	Write	K	M
Node 3			
Node 4			

Data associated with A kicked out of cache due to conflict miss; would be written back to memory because of modification

Event #4: P1 writes to address with tag A

	Event	Address	State
Node 1			
(4)	Write	A	M
Node 2			
(3)	Read	Z	E
(2)	Write	K	M
Node 3			
Node 4			

Event #5: P4 reads from address with tag Z

	Event	Address	State
Node 1			
(4)	Write	A	M
Node 2			
(3)	Read	Z	S
(2)	Write	K	M
Node 3			
Node 4			
(5)	Read	Z	S

Note that cache block with tag Z on Node 2 moves from E state to S state as there are now 2 cached copies in system
(also, note typo in handout → entries (3), (2) should be with Node 2)

Event #6: P3 writes to address with tag I

Address with tag I maps to same block as address with tag K

	Event	Address	State
Node 1			
(4)	Write	A	M
Node 2			
(3)	Read	Z	S
(2)	Write	K	M
Node 3			
(6)	Write	I	M
Node 4			
(5)	Read	Z	S

I and K are on different nodes, so there is no problem.

Event #7: P2 reads from address with tag Q

Address with tag Q maps to same block as address with tag Z

	Event	Address	State
Node 1			
(4)	W	A	M
Node 2			
~~(3)~~	~~R~~	~~Z~~	~~S~~
(2)	W	K	M
(7)	R	Q	E
Node 3			
(6)	W	I	M
Node 4			
(5)	R	Z	S

Now we have a conflict miss on Node 2; data in block with tag Z is kicked out; no write back needed because Z is just cached in shared state

Event #8: P2 reads from address with tag B

	Event	Address	State
Node 1			
(4)	W	A	M
Node 2			
(2)	W	K	M
(7)	R	Q	E
(8)	R	B	E
Node 3			
(6)	W	I	M
Node 4			
(5)	R	Z	S

Event #9: P2 writes to address with tag B

	Event	Address	State
Node 1			
(4)	W	A	M
Node 2			
(2)	W	K	M
(7)	R	Q	E
(9)	W	B	M
Node 3			
(6)	W	I	M
Node 4			
(5)	R	Z	S

Block with tag B moves to the M state.

Event #10: P1 reads from address with tag B

	Event	Address	State
Node 1			
(4)	W	A	M
(10)	R	B	S
Node 2			
(2)	W	K	M
(7)	R	Q	E
(9)	W	B	S
Node 3			
(6)	W	I	M
Node 4			
(5)	R	Z	S

Entry in Node 2 must move to S state, write data back

Event #11: P1 writes to address with tag K

	Event	Address	State
Node 1			
(4)	W	A	M
(10)	R	B	S
(11)	W	K	M
Node 2			
~~(2)~~	~~W~~	~~K~~	~~M~~
(7)	R	Q	E
(9)	W	B	S
Node 3			
(6)	W	I	M
Node 4			
(5)	R	Z	S

Entry in Node 2 must write data back and then invalidate itself

Event #12: P3 reads from address with tag N

	Event	Address	State
Node 1			
(4)	W	A	M
(10)	R	B	S
(11)	W	K	M
Node 2			
(7)	R	Q	E
(9)	W	B	S
Node 3			
(6)	W	I	M
(12)	R	N	E
Node 4			
(5)	R	Z	S

Event #13: P3 reads from address tag K

	Event	Address	State
Node 1			
(4)	W	A	M
(10)	R	B	S
(11)	W	K	S
Node 2			
(7)	R	Q	E
(9)	W	B	S
Node 3			
(6)	W	I	M
(12)	R	N	E
(13)	R	K	S
Node 4			
(5)	R	Z	S

Entry in Node 1 must write data back and then move to the shared state – as we are just doing a read on P3.