Assignment-2

For each memory access it put whole 4096 B i.e. 64 set each with block of size 64 (2^6: block size & set: 2^9/2^3 = 2^6)

MSI Protocol:

Instruction	Expected	Actual
R 0x0 0	Read miss, busRq, Memory access ,Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
R 0x40 0	Read MIss, memory access, invalid to share	Read Hit, busRq
R 0x1000 0	Read miss, busRq, Memory access, Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
failed	Hit:0, miss:3, busrq:3, share:0 mem:3	Hit:1, miss:2, busrq:2, share:0 mem:2

Note:

- 0x0: belongs to set 0 and 0x40 belongs to set 1, conceptually, it should be miss but it show hit for different set address too, even up to **"0xfff"** (4096)
- That mean it treating 8set as a block, since set * size of block = 4096 (2^8 * 64);
- This bugs found in all three protocols.

Invalid State to shared State:

Instruction	Expected	Actual
R 0x1 0	Read miss, busRq, Memory access ,Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
R 0x1 2	Read miss, busRq, Memory access, Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
failed/correct	Hit:1, miss:1, busrq:2, share:1 mem:1	Hit:0, miss:2, busrq:2, share:0 mem:2

Note:

- Conceptually, In initial case cache have no address, so for each different cache transaction there should be memory access, in order to get address into cache

- After address get into cache it will perform according to what we have studied in class.
- So for initial case for above translation, it is correct according to me, unlike rest student saying it as bug, but it think its correct.

Instruction	Expected	Actual
R 0x1 0	Read miss, busRq, Memory access ,Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
R 0x1 2	Read miss, busRq, Memory access, Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
failed	Hit:1, miss:1, busrq:2, share:1 mem:1	Hit:0, miss:2, busrq:2, share:0 mem:2

Note:

- for second instruction, since cache of core-2 have no address it will get address from memory.
- Despite of both cache has clean data still cache 0 share data with cache 2

Invalid to Modified State:

Instruction	Expected	Actual
W 0x1 0	Write miss, busRq, Memory access ,Invalid to Modified	Write miss, busRq, Memory access ,Invalid to Modified
W 0x1 2	Write miss, busRq, memory access and flush other, Invalid to Modified	Write miss, busRq, memory access and flush other, Invalid to Modified
Passed	Hit:0, miss:2, flush:1 busrq:2, mem:3	Hit:0, miss:2, flush:1 busrq:2, mem:3

Shared to shared :

(Hit)

Instruction	Expected	Actual
R 0x1 0	Read miss, busRq, Memory access ,Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
R 0xfff 0	Read hit, shared to shared	Read hit, shared to shared

Passed	Hit:1, miss:1, flush:1 busrq:1, mem:1	Hit:1, miss:1, flush:1 busrq:1, mem:1

Note: if we pass 0x1001 as address it will miss again, since it access same block but different address.

(MIss)

Instruction	Expected	Actual
R 0x1 0	Read miss, busRq, Memory access ,Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
R 0x1001 0	Read Miss, shared to shared	Read Miss, shared to shared
Passed	Hit:0, miss:2, flush:0 busrq:2, mem:2	Hit:0, miss:2, flush:0 busrq:2, mem:2

Shared to Invalid(c0) & Modified(c1):

Instruction	Expected	Actual
R 0x1 0	Read miss, busRq, Memory access ,Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
R 0x1 1	Read miss, busRq, Memory access, Invalid to shared	Read hit, shared to shared
W 0x1 0	Write Hit, shared to Modified	Write MIss, busrq, Memory access
Failed	Hit:1, miss:2, flush:0 busrq:2, mem:2	Hit:0, miss:3, flush:0 busrq:3, mem:3

Modified to Modified:

Instruction	Expected	Actual
W 0x1 0	Write miss, busrq, Memory access, invalid to modified	Write miss, busrq, Memory access, invalid to modified
W 0xc6 1	Write miss, busRq, Memory access, Invalid to modified	Write miss, busRq, Memory access, Invalid to modified
W 0x1 0	Write Hit, modified to Modified	Write Hit, modified to Modified
Passed	Hit:1, miss:2, flush:0 busrq:2, mem:2	Hit:1, miss:2, flush:0 busrq:2, mem:2

Modified to shared:

Instruction	Expected	Actual
W 0x1 0	Write miss, busrq, Memory access, invalid to modified	Write miss, busrq, Memory access, invalid to modified
W 0x1 1	Write miss, invalid to modified	Write miss, invalid to modified
R 0x1 0	Read Hit, cache 1 modified to Shared	Read Hit, cache 1 , modified to Shared
Failed	Hit:0, miss:3, share 0, flush2 busrq:3, mem:3	Hit:0, miss:3, flush2, share 0, busrq:3, mem:5

Modified to invalid:

Instruction	Expected	Actual
W 0x1 0	Write miss, busrq, Memory access, invalid to modified	Write miss, busrq, Memory access, invalid to modified
W 0x10 1	Write miss, busrq, memory access flush cache0 (mem) invalidate c 0 & invalid to modified	Write miss, busrq, memory access flush cache0 (mem) invalidate c 0 & invalid to modified
Passed	Hit:0, miss:2, flush:1 busrq:2, mem:3	Hit:0, miss:2, flush:1 busrq:2, mem:3

MESI Protocol:

Improvement Over MSI;

Instructions	MSI	MESI
R 0x1 0	# of miss : 6	# of miss : 3
W 0x1 0	# of hit : 6	# of hit : 3
R 0x1001 0	# bus req : 6	# bus req : 3
W 0x1001 0	# mem : 6	# mem : 6

R 0x2001 0	# share : 0	# share : 0
W 0x2001 0	# flush : 0	# flush : 0

Observe that: # of bus request msi is 6, while in mesi is 3 (avoid invalid unnecessary), also Since Exclusive write data on cache silently its # miss only 3 unlike msi has 6.
But we can see that there is no improvement in # of mem access.
So, mesi is befinicial in those application which have frequent memory access each for different sets.

Invalid to Exclusive:

Instruction	Expected	Actual
R 0x1 0	Read miss, busrq, Invalid to shared	Read miss, busrq, Invalid to shared
R 0x40 1	Read miss, busrq, Invalid to shared	Read miss, busrq, Invalid to shared
Failed	Hit:0, miss:2, flush:0 busrq:2, mem:2	Hit:0, miss:2, flush:0 busrq:2, mem:4

Note: if we read in same set i.e.(0x0 - ox3f) it gives correct # memory access (mem 2). (Change in tag bit cause 4 mem access)

Instruction	Expected	Actual
R 0x1 0	Read miss, busRq, Memory access ,Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
R 0x3f 2	Read miss, busRq, Memory access, Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
failed	Hit:1, miss:1, busrq:2, share:1 mem:1	Hit:0, miss:2, busrq:2, share:0 mem:2

Note: for 0x3f (same set) it uses 2 mem while for 0x40 (diff set) it uses 4 mem , but conceptually for earlier case it should needed 2 mem and later it should 1 mem since sharing.

Exclusive to shared:

Instruction	Expected	Actual
R 0x1 0	Read miss, busrq, Invalid to shared	Read miss, busrq, Invalid to shared
R 0x34 1	Read miss, busrq, Invalid to shared & Cache 0 exclusive to shared	Read miss, busrq, Invalid to shared & Cache 0 exclusive to shared
Failed	Hit:1, miss:1, share:1 flush:0 busrq:2, mem:1	Hit:0, miss:2, share:1 flush:0 busrq:2, mem:4

Exclusive to Modified:

Instruction	Expected	Actual
R 0x1 0	Read miss, busrq, Invalid to shared	Read miss, busrq, Invalid to shared
R 0x3e 0	Write,hit, busrq, Cache 0 exclusive to modified	Write,hit, busrq, Cache 0 exclusive to modified
Passed	Hit:1, miss:1, share:0 flush:0 busrq:2, mem:1	Hit:1, miss:1, share:0 flush:0 busrq:2, mem:1

Exclusive to Exclusive:

Instruction	Expected	Actual
R 0x1 0	Read miss, busrq, Invalid to shared	Read miss, busrq, Invalid to shared
R 0xfff 0	Read hit, Cache 0 exclusive to exclusive	Write,hit, busrq, Cache 0 exclusive to modified
Passed	Hit:1, miss:1, share:1 flush:0 busrq:1, mem:1	Hit:1, miss:1, share:1 flush:0 busrq:2, mem:2

Note: Read hit should be only 0x0 to 0x3f only in same block but this implantation, consider upto 0xfff as one block.

Exclusive to Invalid:

Instruction	Expected	Actual
R 0x1 0	Read miss, busrq, Invalid to shared	Read miss, busrq, Invalid to shared

W 0x0f 1	Write miss, busrq, Cache 0 exclusive to Invalid & cache 0 of core 1 invalid to modified	Write miss, busrq, Cache 0 exclusive to Invalid & cache 0 of core 1 invalid to modified
Passed	Hit:0, miss:2, share:0 flush:0 busrq:2, mem:2	Hit:0, miss:2, share:1 flush:0 busrq:2, mem:2

Invalid to Modified:

Same as in MSI protocol

Shared to shared

Instruction	Expected	Actual
R 0x1 0	Read miss, busRq, Memory access ,Invalid to shared	Read miss, busRq, Memory access ,Invalid to shared
R 0xfff 0	Read hit, shared to shared	Read hit, shared to shared
failed	Hit:1, miss:1, flush:1 busrq:1, mem:1	Hit:1, miss:1, flush:1 busrq:1, mem:2

Note: unlike MSI (pass) it has 2 memory access, but should be 1 only

Shared to Invalid(c0) & Modified(c1):

Same as in MSI protocol

Modified to Modified:

Same as in MSI protocol

Modified to shared:

Instruction	Expected	Actual
W 0x1 0	Write miss, busrq, Memory access, invalid to modified	Write miss, busrq, Memory access, invalid to modified
W 0x1 1	Write miss, invalid to modified	Write miss, invalid to modified

R 0x1 0	Read Hit, cache 1 modified to Shared	Read Hit, cache 1 , modified to Shared
failed	Hit:0, miss:3, share 2, flush2 busrq:3, mem:3	Hit:0, miss:3, flush2, share 2, busrq:3, mem:5

Note: Unlike MSI 2 share but change no in mem access

Modified to invalid:
Same as in MSI but with share:1

MOESI Protocol:

Improvement Over MSI & MESI:

Instructions	MSI	MESI	MOESI
R 0x1 0	# of miss : 6	# of miss : 3	# of miss : 3
W 0x1 0	# of hit : 6	# of hit : 3	# of hit : 3
R 0x1001 0	# bus req : 6	# bus req : 3	# bus req : 3
W 0x1001 0	# mem : 6	# mem : 6	# mem : 3
R 0x2001 0	# share : 0	# share : 0	# share : 0
W 0x2001 0	# flush : 0	# flush : 0	# flush : 0

Observe that: # of mem access in msi & moesi both have 6, while in moesi it is 3, just by allowing owned state, it avoid unnecessary memory access,
So, Moesi protocol, in suitable in those application in with data change more frequently in cache i.e. PrWr.

Modified to owned state:

Instruction	Expected	Actual
W 0x1 1	write miss, busRq, Memory access ,Invalid to modified	write miss, busRq, Memory access ,Invalid to modified

R 0x1 0	Read miss, busRq, shared data from cache 1 & cache 1 in owned state	Read miss, busRq, shared data from cache 1 & cache 1 in owned state
pass	Hit:0, miss:2, flush:0 busrq:2, mem:1	Hit:0, miss:2, flush:0 busrq:2, mem:1

Instruction	Expected	Actual
W 0x1 0	write miss, busRq, Memory access ,Invalid to modified	write miss, busRq, Memory access ,Invalid to modified
R 0x1 1	Read miss, busRq, Memory access, invalid to exclusive	Read miss, busRq, shared data from cache 1 & cache 1 in owned state
R 0x1 0	Read hit, modified stat	Read hit, modified stat
W 0x1 1	Write hit, shared to modified	Write miss, busrq, cache 1 write on bus busrqX and invalid itself & cache 0 invalid & cache 1 in modified state
failed	Hit:2, miss:2, flush:0 busrq:2, mem:1	Hit:1, miss:3, flush:0 busrq:3, mem:1

Owned to invalid state

Instruction	Expected	Actual
W 0x1 1	write miss, busRq, Memory access ,Invalid to modified	write miss, busRq, Memory access ,Invalid to modified
R 0x1 0	Read miss, busRq, memory access shared data from cache 1 & cache 1 in owned state	Read miss, busRq, shared data from cache 1 & cache 1 in owned state
W 0x1 2	Write miss, busrq, cache 1 write on bus busrqX and invalid itself & cache 0 invalid & cache in owned	Write miss, busrq, cache 1 write on bus busrqX and invalid itself & cache 0 invalid & cache in owned
pass/failed	Hit:0, miss:3, flush:0 busrq:3, mem:3	Hit:0, miss:3, flush:0 busrq:3, mem:1

Instruction	Expected	Actual
W 0x1 0	write miss, busRq, Memory access ,Invalid to modified	write miss, busRq, Memory access ,Invalid to modified

R 0x1 1	Read miss, busRq, memory access	Read miss, busRq, shared data from cache 1 & cache 1 in owned state
W 0x1 1	Write Hit, cache write back(mm) and invalidate	Write miss, busrq, cache 1 write on bus busrqX and invalid itself & cache 0 invalid & cache in owned
failed	Hit:0, miss:3, flush:0 busrq:3, mem:2	Hit:0, miss:3, flush:0 busrq:3, mem:1

Owned to modified:
(write hit)

Instruction	Expected	Actual
W 0x1 1	write miss, busRq, Memory access ,Invalid to modified	write miss, busRq, Memory access ,Invalid to modified
R 0x1 0	Read miss, busRq, shared data from cache 1 & cache 1 in owned state	Read miss, busRq, shared data from cache 1 & cache 1 in owned state
W 0x1 1	Write hit & modified state	Write hit & modified state
pass	Hit:1, miss:2, flush:0 busrq:3, mem:2	Hit:1, miss:2, flush:0 busrq:3, mem:2

Owned to owned:
(Read hit)

Instruction	Expected	Actual
W 0x1 1	write miss, busRq, Memory access ,Invalid to modified	write miss, busRq, Memory access ,Invalid to modified
R 0x1 0	Read miss, busRq, shared data from cache 1 & cache 1 in owned state	Read miss, busRq, shared data from cache 1 & cache 1 in owned state
R 0x1 1	Read hit & owned state	Read hit & owned state
pass	Hit:1, miss:2, flush:0 busrq:2, mem:2	Hit:1, miss:2, flush:0 busrq:2, mem:2

Eviction:

To Evict any block in k-way associative, we need k+1 transactions to replace any one block.

In this simulator it uses 8-way associative, hence 9 transactions will be needed.

Property of eviction:
All block must be in modified/exclusive state except last block of same cache.

Transaction	MSI	MESI	MOESI
W 0x0000 0 (block 0)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
W 0x1000 0 (block 1)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
W 0x2000 0 (block 2)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
W 0x3000 0 (block 3)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
W 0x4000 0 (block 4)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
W 0x5000 0 (block 5)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
W 0x6000 0 (block 6)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
W 0x7000 0 (block 7)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
W 0x8000 0 (block 8)	Write miss , Modified	Write miss , Exclusive	Write miss , Exclusive
Eviction	Miss 9, flush 1, busRq 9, mem 9	Miss 9, flush 1, busRq 9, mem 9	Miss 9, flush 1, busRq 9, mem 9

Eviction is independent of protocol, so i had shown eviction separately, rather than in each each protocol for simplicity.