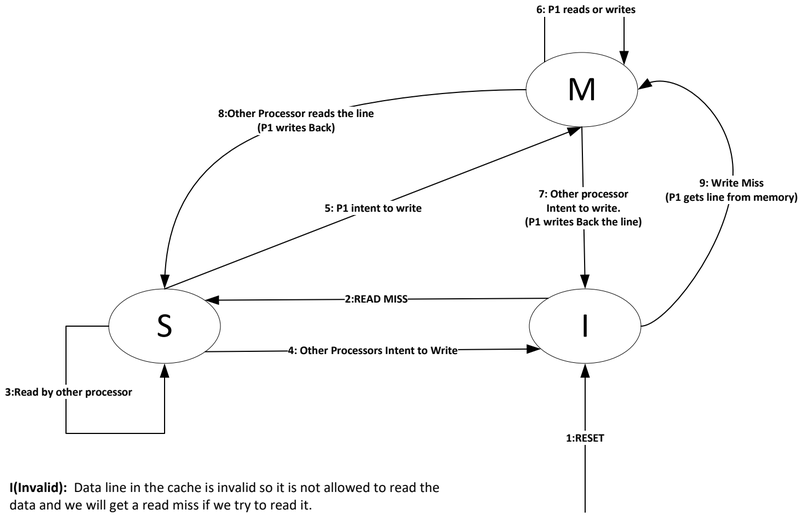
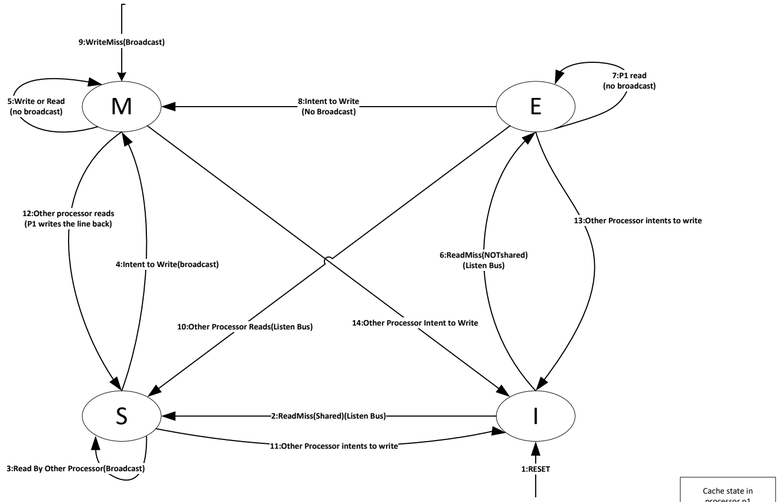


I(Invalid): Data line in the cache is invalid so it is not allowed to read the data and we will get a read miss if we try to read it.

M(Modified): Data is dirty. The data line in the cahce is modified after it was read from the main memory. It is the only copy. If a line is in the M state, no other cache can have the copy of the line. Before the line is moved to the Modified state, all the other copies are invalidated.

S(Shared): The line is a ready only copy. Some other processor may have a copy of the cache line, too.

2: P1 wants to read the data line and it is not in the cache. So P1 gets the line from the cache.
3: P1 has the cache line in the S state and some other processor also wants to read the cache line. No change is needed.
4: Another processor needs to write to a cache line that P1 has in shared state. P1 changes the state of the cache line to Invalid and doesn't need to write it back to cache(since it is not modified). It doesn't need to let anybody know that it had the line either. So if P1 wants to read that cache line again, it needs to go and read it again because the current copy is Invalid.
5: First P1 broadcast the intent to write to the cache so other processors know that P1 is about to write to that cache line. Some implementation strategies wait some time to let other processor invalidate their data, write data back to memory if they need to and so on. Some other implementations might wait for an ACK signal from all the other caches after the write intent is broadcasted. Or, if everyone has the read only copy of the data, they need to transition from Shared state to Invalid state. After the broadcast, all other caches need to invalidate this line in their cache line.
6: No other processor has the copy of the cache line so P1 can read and write to the cache line without communication with the other processors on the bus.
7: Some other processor broadcasts on the bus that it is going to write to a cache line and P1 has the same cache line in the Modified state(Most updated copy of the line).
8: P1 has the most up-to-date copy of the line so the other processors should get this copy. So P1 write the line back to the main memory first and then transition to the shared state. Then other processors can read the line from the main memory and get the most up-to-date copy of the line. So the cache in both processors will be in the shared state.
9: The cache line is in the Invalid state. First, P1 broadcasts the write intent and all the other processors invalidate(write back to memory if necessary) and P1 waits until other processors are done. Then, P1 reads the line from main memory and takes the line to the Modified state.



MESI(Enhanced MSI Protocol)
Difference from MSI:
M(Modified Exclusive):
E(Exclusive but unmodified): P1 has the only readable copy. No need to broadcast when it writes.
S(Shared):
I(Invalid):

1: RESET
2: First the P1 processor broadcasts that it is going to read the line. If any other processor responds and says that it has the data line, it goes to the shared state..
3: If any other processor wants to read the shared line, stay in the shared state and broadcast that P1 has the data. P1 broadcasts so the other processor know whether to go to the Exclusive state or Shared state.
4: P1 wants to write to the line, it broadcasts its intent to write so other processors invalidate their copy.
5: In modified state, P1 can do read and write without broadcasting or changing state.
6: P1 listens to the bus and checks if any other processor has the copy, if it doesn't it goes to the E state.
7: E state means that P1 has the only readable copy of the line so it can read and not broadcast.
8: E state means that P1 has the only readable copy so it doesn't need to broadcast to go to the Modified state.
9: Broadcast so if any other processor has the copy, wait other processors to write back their copy and go to the Modified state.
10: P1 sees that another processor wants to read the line so it goes to the shared state.