

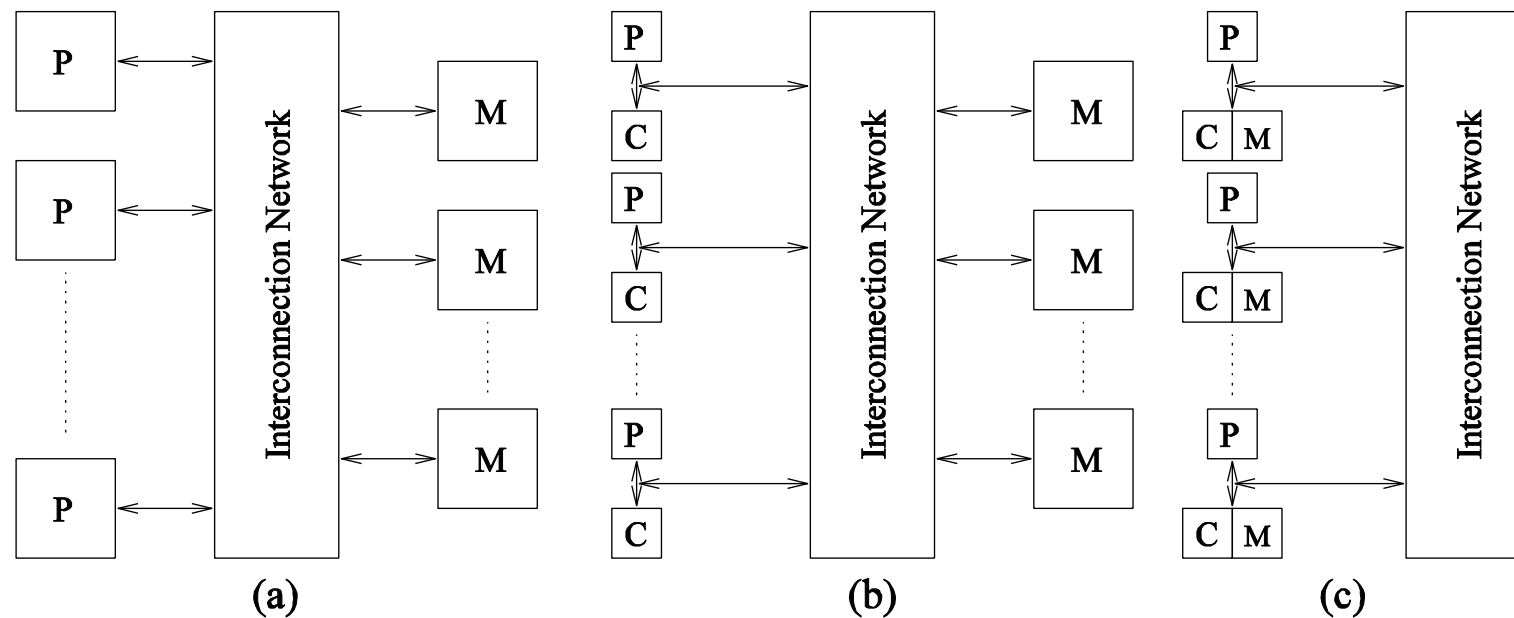


# Memory Architecture and Coherency

## Ozcan Ozturk – Sabancı University

# NUMA and UMA Shared-Address-Space Platforms

- Uniform-memory access shared-address-space computer
- Uniform-memory-access shared-address-space computer with caches and memories
- Non-uniform-memory-access shared-address-space computer with local memory only

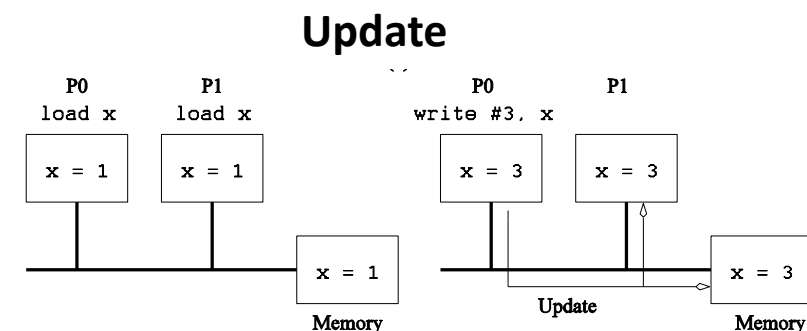
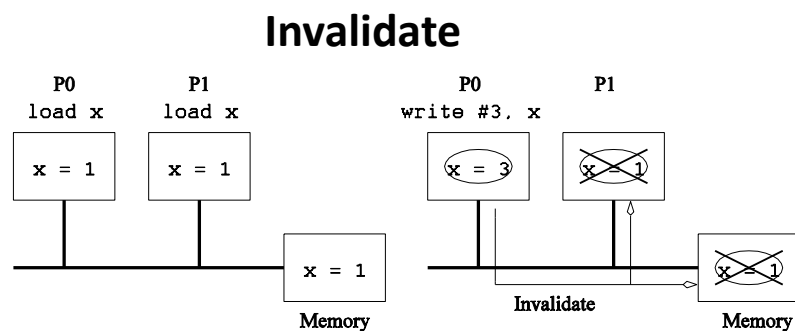


# NUMA and UMA Shared-Address-Space Platforms

- Difference between the terms shared address space and shared memory.
  - The former as a programming abstraction
  - The latter as a physical machine attribute
- It is possible to provide a shared address space using a physically distributed memory

# Cache Coherence in Multiprocessor Systems

- Additional hardware is required to coordinate access to data that might have multiple copies in the network
  - The underlying technique must provide some guarantees on the semantics.
  - This guarantee is generally one of **serializability**
    - There exists some serial order of instruction execution that corresponds to the parallel schedule.
- When the value of a variable is changes, all its copies must either be invalidated or updated.

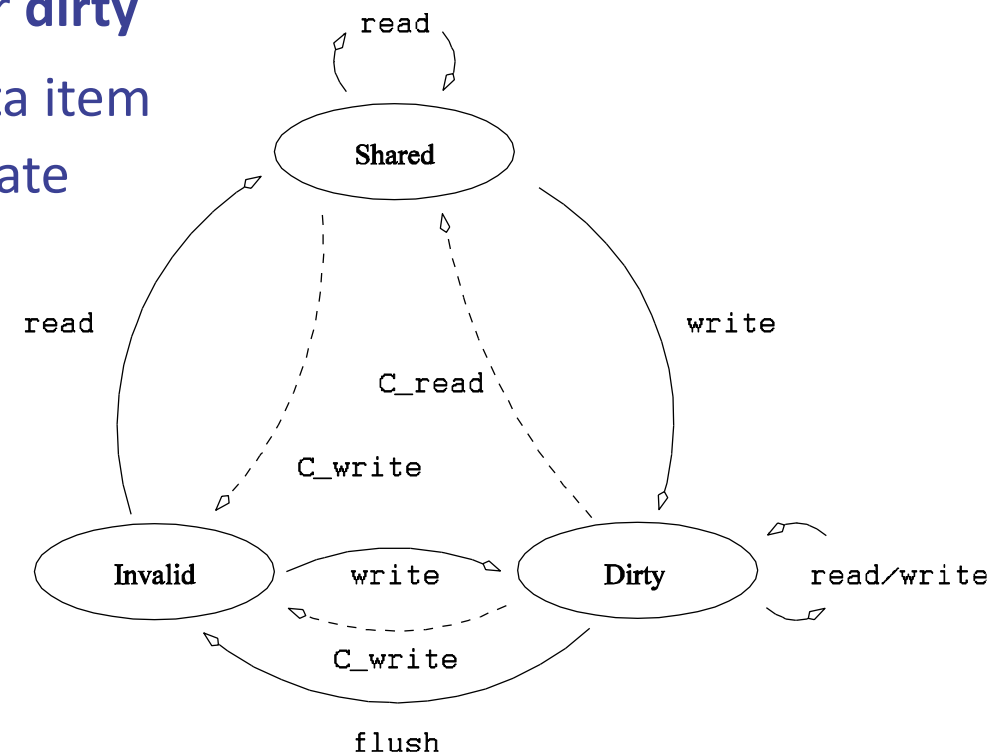


# Cache Coherence: Update and Invalidate Protocols

- If a processor just reads a value once and does not need it again, an update protocol may generate significant overhead.
- If two processors make interleaved test and updates to a variable, an update protocol is better.
- Both protocols suffer from false sharing overheads (two words that are not shared, however, they lie on the same cache line).
- Most current machines use invalidate protocols.

# Maintaining Coherence Using Invalidate Protocols

- Each copy of a data item is associated with a state
- One example of such a set of states is, **shared**, **invalid**, or **dirty**
- In shared state, there are multiple valid copies of the data item
  - An invalidate would have to be generated on an update
- In dirty state, only one copy exists
  - No invalidates needed to be generated
- In invalid state, the data copy is invalid,
  - A read generates a data request



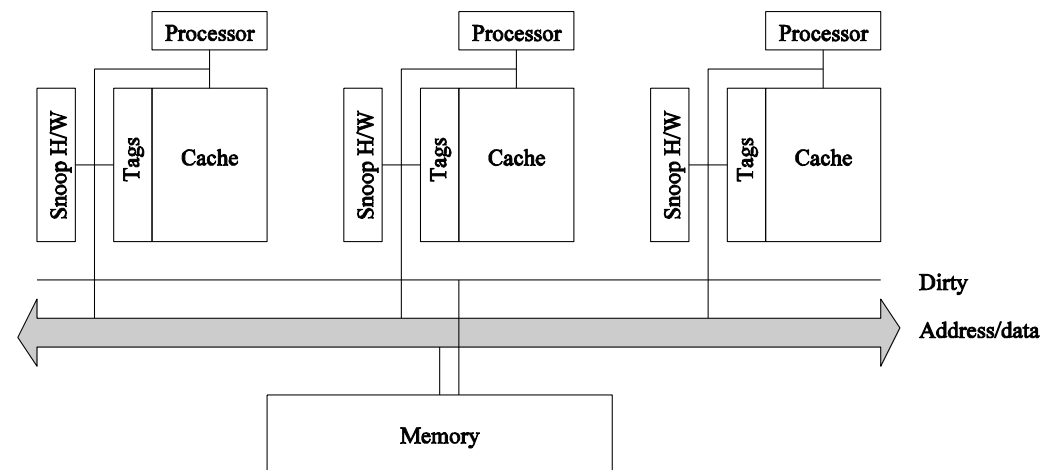
# Maintaining Coherence Using Invalidate Protocols

- Example of parallel program execution

| Time<br>↓ | Instruction at<br>Processor 0 | Instruction at<br>Processor 1 | Variables and<br>their states at<br>Processor 0 | Variables and<br>their states at<br>Processor 1 | Variables and<br>their states in<br>Global mem. |
|-----------|-------------------------------|-------------------------------|-------------------------------------------------|-------------------------------------------------|-------------------------------------------------|
|           |                               |                               |                                                 |                                                 | x = 5, D<br>y = 12, D                           |
|           | read x                        | read y                        | x = 5, S                                        | y = 12, S                                       | x = 5, S<br>y = 12, S                           |
|           | x = x + 1                     | y = y + 1                     | x = 6, D                                        | y = 13, D                                       | x = 5, I<br>y = 12, I                           |
|           | read y                        | read x                        | y = 13, S<br>x = 6, S                           | y = 13, S<br>x = 6, S                           | y = 13, S<br>x = 6, S                           |
|           | x = x + y                     | y = x + y                     | x = 19, D<br>y = 13, I                          | x = 6, I<br>y = 19, D                           | x = 6, I<br>y = 13, I                           |
|           | x = x + 1                     | y = y + 1                     | x = 20, D                                       | y = 20, D                                       | x = 6, I<br>y = 13, I                           |

# Snoopy Cache Systems

- How are invalidates sent to the right processors?
- In snoopy caches, there is a broadcast media that listens to all invalidates and read requests and performs appropriate coherence operations locally.



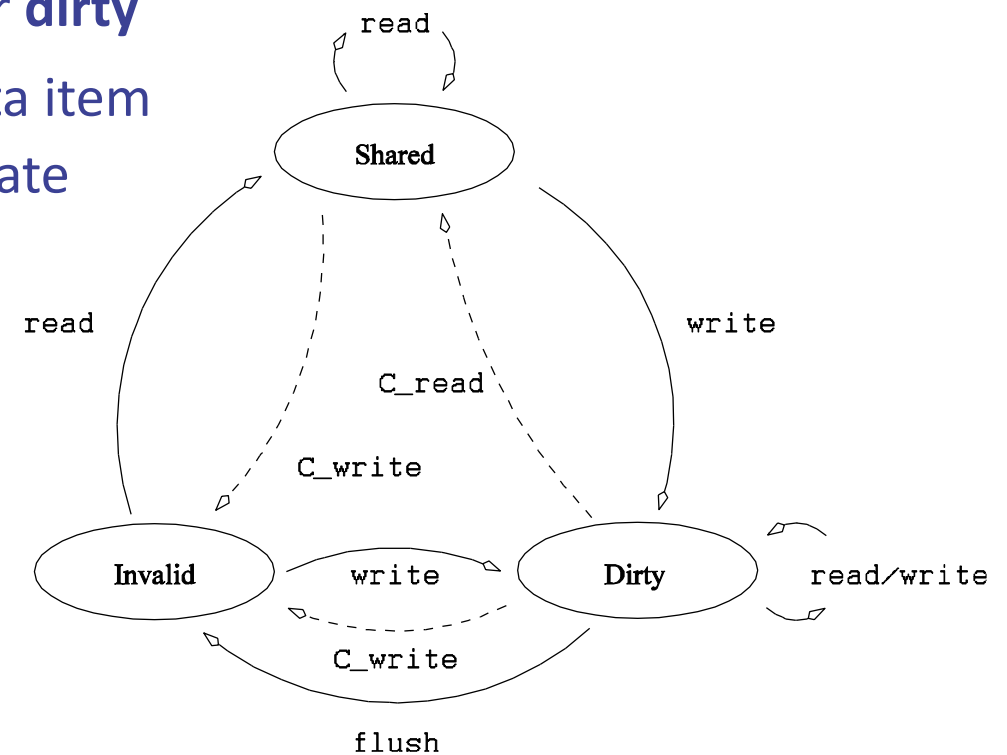
A simple snoopy bus based cache coherence system.

# Performance of Snoopy Caches

- Once copies of data are tagged dirty, all subsequent operations can be performed locally on the cache without generating external traffic.
- If a data item is read by a number of processors, it transitions to the shared state in the cache and all subsequent read operations become local.
- If processors read and update data at the same time, they generate coherence requests on the bus - which is ultimately bandwidth limited.

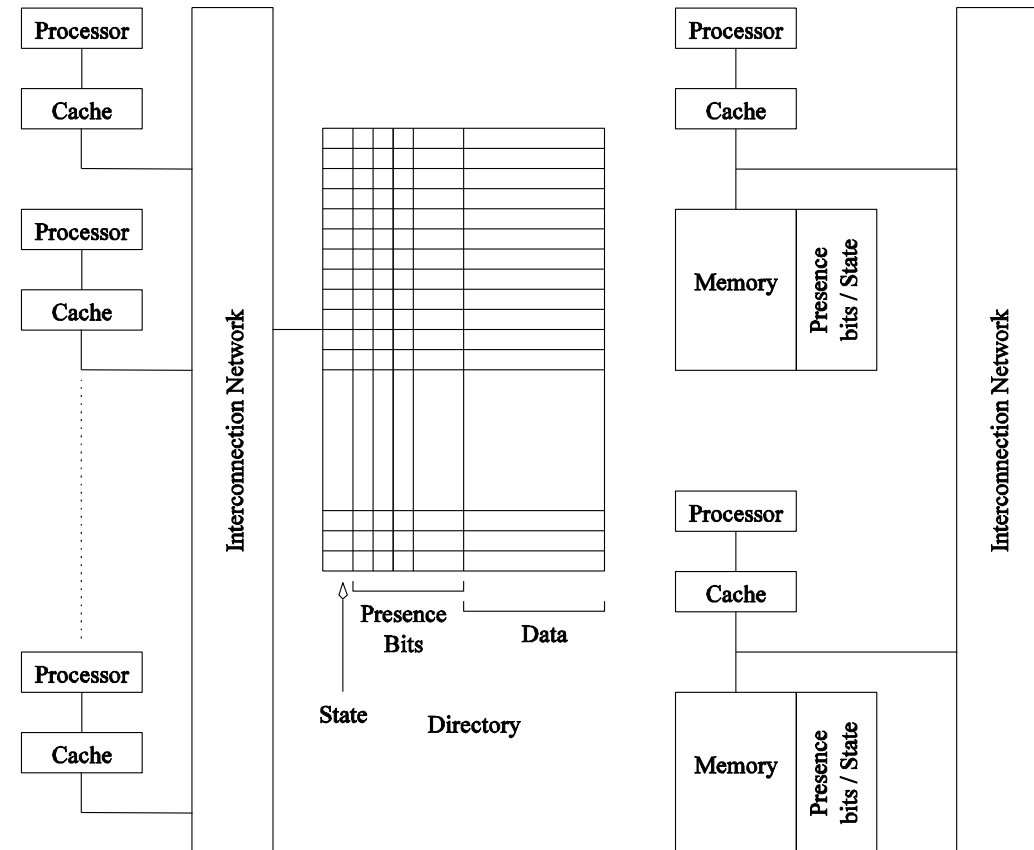
# Directory Based Systems

- Each copy of a data item is associated with a state
- One example of such a set of states is, **shared**, **invalid**, or **dirty**
- In shared state, there are multiple valid copies of the data item
  - An invalidate would have to be generated on an update
- In dirty state, only one copy exists
  - No invalidates needed to be generated
- In invalid state, the data copy is invalid,
  - A read generates a data request



# Maintaining Coherence Using Invalidate Protocols

- In snoopy caches, each coherence operation is sent to all processors. This is an inherent limitation.
- Why not send coherence requests to only those processors that need to be notified?
- This is done using a directory, which maintains a presence vector for each data item (cache line) along with its global state.



## Next Lecture

- Memory and Performance

# References

Grama, Ananth, Anshul Gupta, George Karypis, and Vipin Kumar. 2003. *Introduction to Parallel Computing*. 2nd ed. Boston: Addison-Wesley.

# Sabancı Universitesi Thank you!



**Co-funded by  
the European Union**



**EuroHPC**  
Joint Undertaking



EuroCC 3 has received funding from the [European High-Performance Computing Joint Undertaking \(JU\)](#) under Grant Agreement No. 101306701. The JU receives support from the [European Union](#)'s Digital Europe Programme and Germany, Albania, Austria, Belgium, Bosnia and Herzegovina, Bulgaria, Croatia, Cyprus, Czechia, Denmark, Estonia, Finland, France, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Montenegro, the Netherlands, North Macedonia, Norway, Poland, Portugal, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Türkiye, and Kosovo.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or EuroHPC Joint Undertaking. Neither the European Union nor the EuroHPC Joint Undertaking can be held responsible for them.