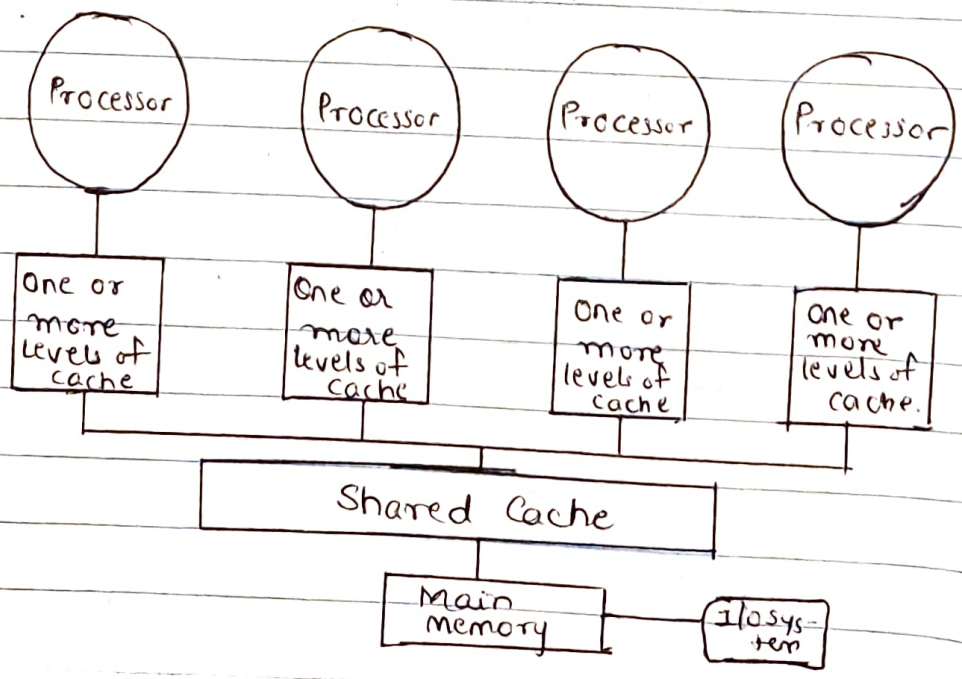
Rushikesh Khamkar

COA

Q1) Below is the difference among symmetric multi processors architecture, distributed shared memory multiprocessor architecture and distributed memory multicomputer architecture.

1) Symmetric multiprocessor Architecture -

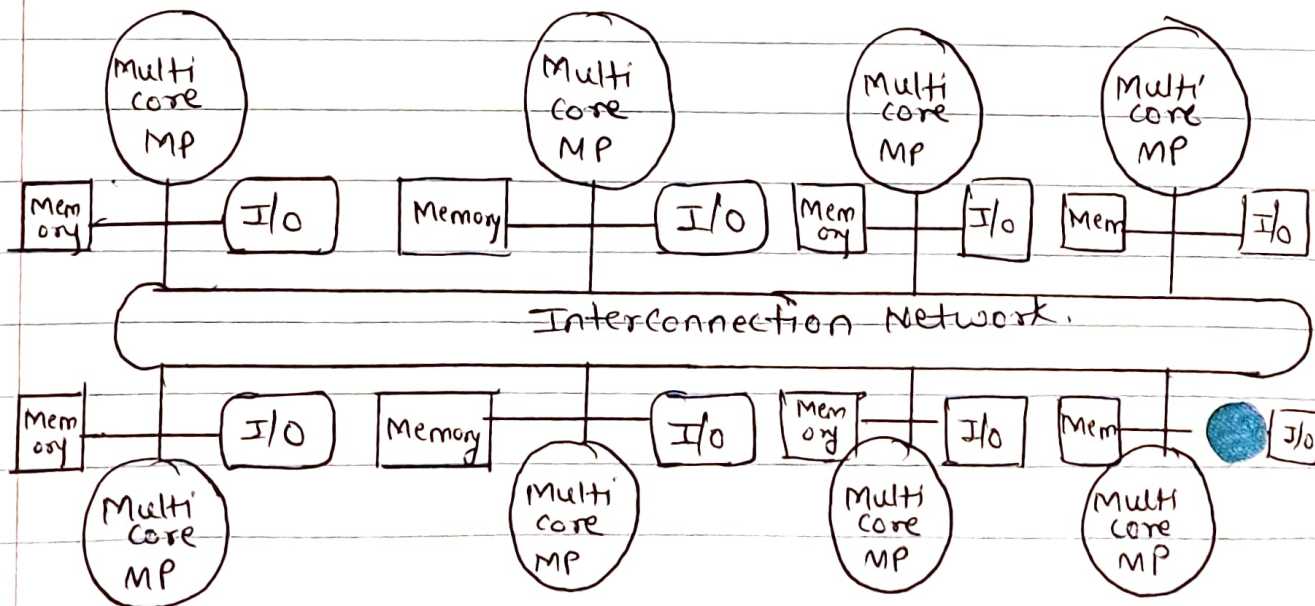
- it contains small number of core.
- it share single memory with uniform memory latency.
- it is also called a centralized shared memory multiprocessor.
- In a SMP system there is usually a single pool of memory which is equally fast for all processors.
- Below is the diagram of Symmetric multi processor architecture →



2) Distributed Shared Memory (DSM) -

- memory is distributed among processors in this DSM architecture
- DMA has non-uniform memory access / latency.
- Processor are connected via direct and non direct interconnection networks in DMA architecture.
- In DSM, there are multiple pools of memory and the latency to access memory depends on the relative position of the processor & memory. DSM is different from distributed memory multicomputers where memory is not shared and some form of message passing to solve a problem.

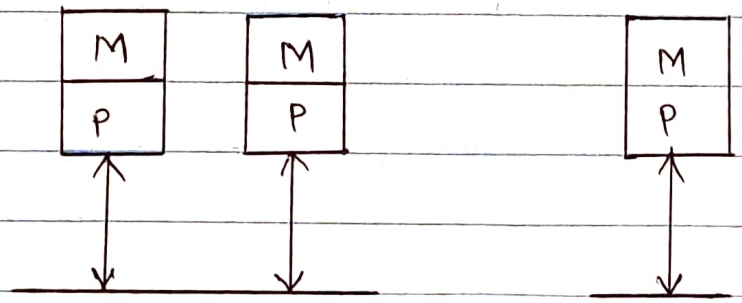
- Below diagram represents DSM architecture -



3) Distributed memory architecture -

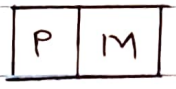
- This system includes multiple computers known as nodes and are related by message passing network.
- Each node is an independent computer including a processor, local memory & sometimes connected disks or I/O peripherals.
- Message passing n/w supports point to point static connections among node.
- All local memories provides and are applicable only by local processors.

- For this method, traditional multicomputer has been referred to as non remote memory access machines.



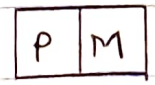
M

Message-Passing Interconnection Network (Mesh, ring, torus, hypercube, cube-connected cycle, etc.)



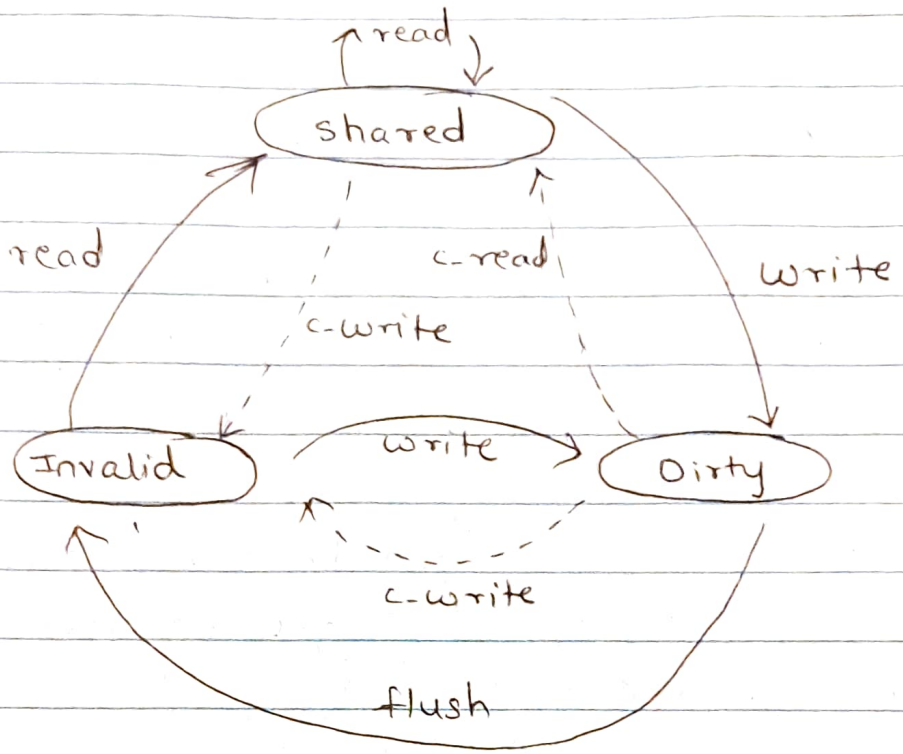
M

Generic Model of a Message-Passing multicomputer.

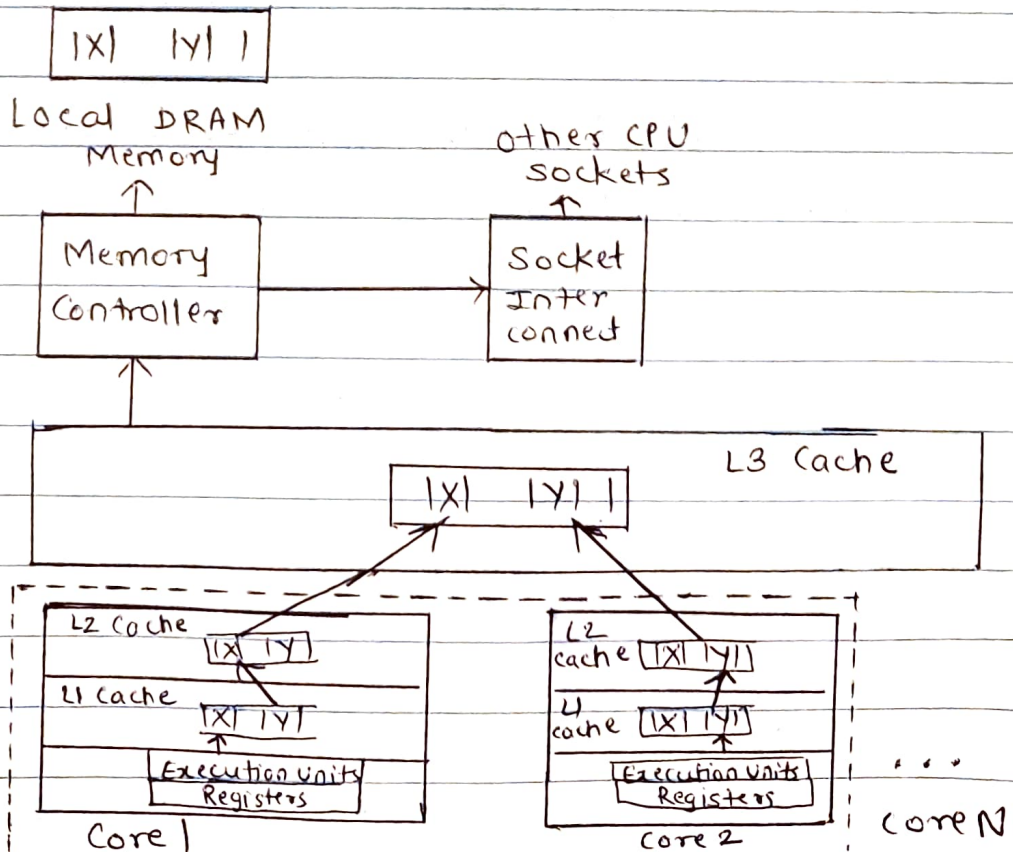


Q.2) - False sharing occurs when two words are not shared but lie on the same cache line.
- It occurs when processors that are accessing logically distinct data nevertheless conflict because the locations they are accesing lie on the same cache line.
- Scenarios when false sharing occurs -
① If processor just reads a value once and does not need it again, an update protocal may generate significant overhead.
② If two processor make interleaved test and updates to variable, update protocal is better.

- Both the above protocol suffer from false sharing overheads.
- Most current machine use invalidate protocols. Each copy of data item is associated with a state such as shared, invalid or directly / executive
- In shared state there are multiple valid copies of the data item.
- In dirty state, only one copy exists and therefore no invalidates need to be generated.
- In invalidate state, the data copy is invalid.



consider the below diagram to illustrate false sharing -



The above figure illustrate the issue of false sharing. A thread running on core 1 wants to update variable x while a thread on core 2 wants to update variable y. But these two variable reside on the same cache line. Each thread will race for ownership of this cache line. so that they can update it.

If core1 gets ownership then cache sub system will need to invalidate the corresponding cache line for core2.

When core 2 gets ownership & performs its update then core1 will be told to invalidate copy of the cache line.

They will ping pong back and forth via L3 cache impacting performance.

Q. 3).

(a) P0 : read 120 (address 120 hashed to block B0 for all processor)

cpu Read - a read miss : I → S

P0 : B0 : (S, 120, 00, 20)

(b) P0 : write 120 ← 80

CPU write - I → M

P0 . B0 : (M, 120, 00, 80)

Invalidate other caches containing 120

P15 B0 (I, 120, 00, 20).

(c) P15 write 120 ← 80

CPU write - S ← M

P15 . B0 - (M, 120, 00, 80)

(d) P1 - read 110

CPU Read - Read miss I → S.

P1 . B2 (S, 110, 00, 30) because P0 was the owner, it carries valid content.

M → S

P0 . B2 : (S, 110, 00, 30)

Also update memory now (Write Back)

M[110] = (00 30)

Result of record i.e. 00 03.

(e) P0 - write 108 ← 48

CPU write miss - S → M

P0. B1 : m (108, 00, 48)

Invalidate other cache - S → I

P15 · B1 : (1 108, 00, 08)

Q.4)

- In snoopy cache system, cache coherence operation is sent to to all processor which is an inherent limitation.

- Instead we can send coherence requests to only those processor that need to be notified.

- This is done using directory which maintains a presence vector for each data item along with its state.

- The advantage of directory based cache coherence system over snoopy cache system is that it is more scalable than snoopy or centralized directory system.

- It has one more advantage that directory based cache coherence system permit

simultenous coherence operations.

- Remains significant overhead of directory storages.

Q:5) Two main types of h/w threading-

① Temporal multi threading-

- multiple h/w threads share access to pipeline resource one thread at a time.
- Context switches involve two threads one being saved and one being resored.
- There are 2 types -
  ① course grained- performed by kernal.
  ② fine grained- performed using h/w driver

② Simultenous multithreading-

- The maximize number of threads that can simultenously execute in pipeline is more than one.
- context switch involve N threads, where {N > 1}
- GPU utilize SMP mechanism in order to maintain internal h/w concurreny.

Q.6) - Pipeline flush is also known as pipeline break or pipeline stalls.

- Pipeline flush is a procedure enacted by a CPU when it connect ensure that it will correctly process its instruction pipeline in the next clock cycle.

- Hzard condition may present itself it for instance, if two pending instruction are "data dependent". In such case, CPU may need to clear or flush the instruction pipeline to ensure calculation are not corrupted by the pipelining process.

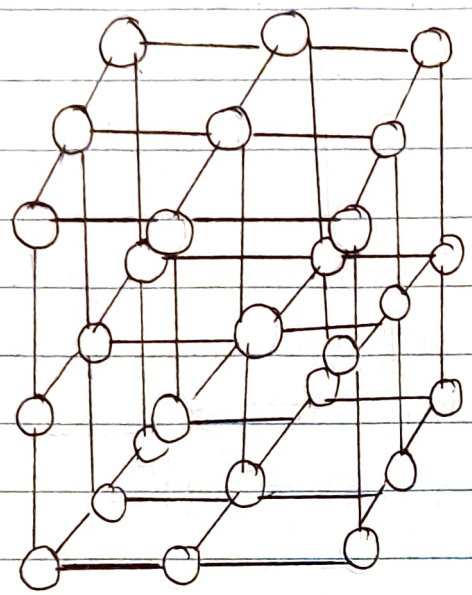
- Pipeline flush means rewind any state that hasn't been fully executed in order to ensure that pipeline is not poisoned for the next thread of execution.

- In h/w multithreading the goal is to maximize use of h/w execution pipeline.

- Steps for that are follow -
  - Trigger the context switch - This may involve one or more mechanism interruptions.

- Pause the pipeline - pipeline can no longer make forward progress.
- Save the architecture state.
- Perform pipeline flush.
- Restore state from another thread.
- Restart the pipeline.

Q.7) - 3-D torus interconnect topology with 27 nodes →



Q.8) - Dividing the mesh in two equal part each of $P/2$.

- Each processor would leave atleast $\sqrt{P}$ communication links betn the partitions.

- Hence bisection width of the mesh is also $\sqrt{P}$.

- The processor which are present at the the two execution of mesh need the largest number of communication links to communicate:

$$= 2\log\sqrt{P} + 2\log\sqrt{P}$$
$$= 2\log P$$

- A complete binary tree is imposed on each row & column of the mesh.

- There are $2\sqrt{P}$ such rows & columns. each such tree has $\sqrt{P}-1$ switches.

- By accumulating the above, the total number of switching nodes are-

$$= 2\sqrt{P} * \sqrt{P} - 1$$
$$= \underline{\underline{2P\sqrt{P}}}$$