parallel processing

- A parallel processing system can carry out simultaneous data-processing to achieve faster execution time. For instance, while an instruction is being processed in the ALU component of the CPU, the next instruction can be read from memory.
- The primary purpose of parallel processing is to enhance the computer processing capability and increase its throughput, i.e. the amount of processing that can be accomplished during a given interval of time.

- Parallel computing is the use of two or more processors (cores, computers) in combination to solve a single problem. It is a type of computing architecture in which several processors execute or process an application or computation simultaneously. Parallel computing helps in performing large computations by dividing the workload between more than one processor, all of which work through the computation at the same time. Most supercomputers employ parallel computing principles to operate.

Advantages

- Parallel computing saves time, allowing the execution of applications in a shorter wall-clock time.
- Solve Larger Problems in a short point of time.
- Compared to serial computing, parallel computing is much better suited for modeling, simulating and understanding complex, real-world phenomena.
- Throwing more resources at a task will shorten its time to completion, with potential cost savings. Parallel computers can be built from cheap, commodity components.
- Many problems are so large and/or complex that it is impractical or impossible to solve them on a single computer, especially given limited computer memory.
- You can do many things simultaneously by using multiple computing resources.
- Can using computer resources on the Wide Area Network(WAN) or even on the internet.
- It can help keep you organized. If you have Internet, then communication and social networking can be made easier.
- It has massive data storage and quick data computations.

Disadvantages

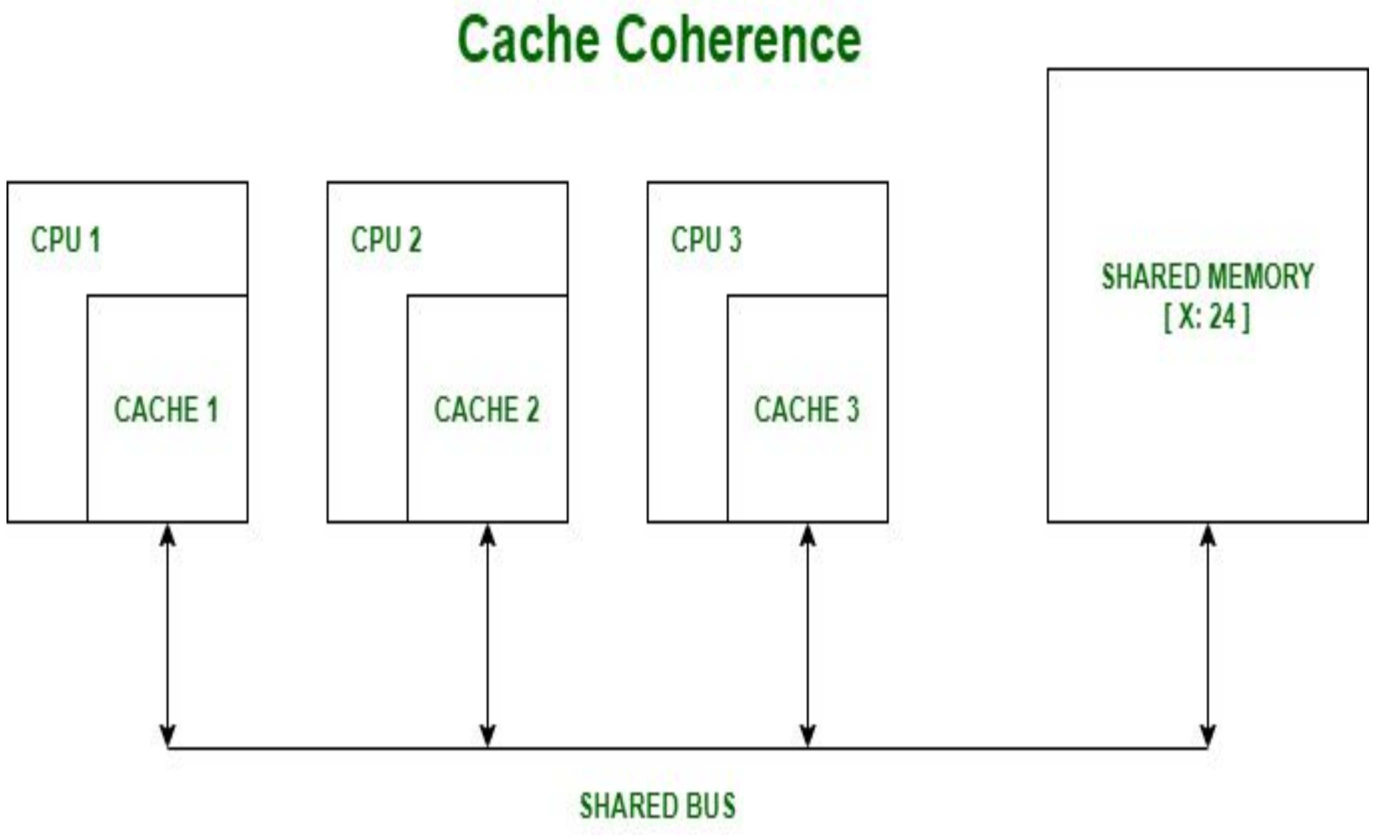
- Programming to target Parallel architecture is a bit difficult but with proper understanding and practice, you are good to go.
- The use of parallel computing lets you solve computationally and data-intensive problems using multicore processors, but, sometimes this effect on some of our control algorithm and does not give good results and this can also affect the convergence of the system due to the parallel option.
- The extra cost (i.e. increased execution time) incurred are due to data transfers, synchronization, communication, thread creation/destruction, etc. These costs can sometimes be quite large, and may actually exceed the gains due to parallelization.
- Various code tweaking has to be performed for different target architectures for improved performance.
- Better cooling technologies are required in case of clusters.
- Power consumption is huge by the multi-core architectures.
- Parallel solutions are harder to implement, they're harder to debug or prove correct, and they often perform worse than their serial counterparts due to communication and coordination overhead.

- **SISD (*Single Instruction, Single Data*)** refers to the traditional von Neumann architecture where a single sequential processing element (PE) operates on a single stream of data. SISD represents a computer organization with a control unit, a processing unit, and a memory unit. SISD is like the serial computer in use. SISD executes instructions sequentially and they may or may not have parallel processing capabilities.
- •**SIMD (*Single Instruction, Multiple Data*)** performs the same operation on multiple data items simultaneously. SIMD organization includes multiple processing elements. All these elements are below the administration of a common control unit. All processors get identical instruction from the control unit but work on multiple data items.
- •**MIMD (*Multiple Instruction, Multiple Data*)** uses multiple PEs to execute different instructions on different data streams.
- •**MISD (*Multiple Instruction, Single Data*)** employs multiple PEs to execute different instructions on a single stream of data. This type of parallelism is not so common but can be found in pipelined architectures such as systolic arrays

Cache coherence :

- In a multiprocessor system, data inconsistency may occur among adjacent levels or within the same level of the memory hierarchy.
- In a shared memory multiprocessor with a separate cache memory for each processor, it is possible to have many copies of any one instruction operand: one copy in the main memory and one in each cache memory. When one copy of an operand is changed, the other copies of the operand must be changed also.

Contd…


Cache Coherence
CPU 1
CACHE 1
CPU 2
CACHE 2
CPU 3
CACHE 3
SHARED MEMORY
[X: 24]
SHARED BUS

Example::

- Suppose there are three processors, each having cache. Suppose the following scenario:-
- **Processor 1 read X :** obtains 24 from the memory and caches it.
- **Processor 2 read X :** obtains 24 from memory and caches it.
- **Again, processor 1 writes as X :** 64, Its locally cached copy is updated. Now, processor 3 reads X, what value should it get?
- Memory and processor 2 thinks it is 24 and processor 1 thinks it is 64.

Cache coherence

- Cache coherence is the discipline that ensures that changes in the values of shared operands are propagated throughout the system in a timely fashion.
- There are three distinct level of cache coherence :-
- Every write operation appears to occur instantaneously.
- All processors see exactly the same sequence of changes of values for each separate operand.
- Different processors may see an operation and assume different sequences of values; this is known as non-coherent behavior.

- There are various Cache Coherence Protocols in multiprocessor system. These are :-
- MSI protocol (Modified, Shared, Invalid)
- MOSI protocol (Modified, Owned, Shared, Invalid)
- MESI protocol (Modified, Exclusive, Shared, Invalid)
- MOESI protocol (Modified, Owned, Exclusive, Shared, Invalid)
- These important terms are discussed as follows:
- **Modified –**
  It means that the the value in the cache is dirty, that is the value in current cache is different from the main memory.
- **Exclusive –**
  It means that the value present in the cache is same as that present in the main memory, that is the value is clean.
- **Shared –**
  It means that the cache value holds the most recent data copy and that is what shared among all the cache and main memory as well.
- **Owned –**
  It means that the current cache holds the block and is now the owner of that block, that is having all rights on that particular blocks.
- **Invalid –**
  This states that the current cache block itself is invalid and is required to be fetched from other cache or main memory.

Coherency mechanisms

There are three types of coherence :

- **Directory-based –**
  In a directory-based system, the data being shared is placed in a common directory that maintains the coherence between caches. The directory acts as a filter through which the processor must ask permission to load an entry from the primary memory to its cache. When an entry is changed, the directory either updates or invalidates the other caches with that entry.
- **Snooping (taak chaak karna)–**
  First introduced in 1983, snooping is a process where the individual caches monitor address lines for accesses to memory locations that they have cached. It is called a write invalidate protocol. When a write operation is observed to a location that a cache has a copy of and the cache controller invalidates its own copy of the snooped memory location.
- **Snarfing(soomghna) –**
  It is a mechanism where a cache controller watches both address and data in an attempt to update its own copy of a memory location when a second master modifies a location in main memory. When a write operation is observed to a location that a cache has a copy of the cache controller updates its own copy of the snarfed memory location with the new data.