**Parallel Processing**

Traditionally, the computer has been viewed as a sequential machine. Most computer programming languages require the programmer to specify algorithms as sequences of instructions. Processors execute programs by executing machine instructions in a sequence and one at a time. Each instruction is executed in a sequence of operations (fetch instruction, fetch operands, perform operation, store results).

This view of the computer has never been entirely true. At the micro-operation level, multiple control signals are generated at the same time. Instruction pipelining, at least to the extent of overlapping fetch and execute operations, has been around for a long time. Both of these are examples of performing functions in parallel. This approach is taken further with superscalar organization, which exploits instruction-level parallelism. With a superscalar machine, there are multiple execution units within a single processor, and these may execute multiple instructions from the same program in parallel.

For the purpose of increasing the computational speed of computer system, the term ‘parallel-processing employed to give simultaneous data-processing operations is used to represent a large class. In addition, a parallel processing system is capable of concurrent data processing to achieve faster execution times.

***Types of Parallel Processer Systems***

A taxonomy first introduced by Flynn is still the most common way of categorizing systems with parallel capability. Flynn proposed the following categories of computer systems:

1. **Single instruction, single data (SISD) stream:** A single processor executes a single instruction stream to operate on data stored in a single memory. Uniprocessors fall into this category.
2. **Single instruction, multiple data (SIMD) stream:** A single machine instruction controls the simultaneous execution of a number of processing elements on a lockstep basis. Each processing element has an associated data memory, so that each instruction is executed on a different set of data by the different processors. Vector and array processors fall into this category.
3. **Multiple instruction, single data (MISD) stream:** A sequence of data is transmitted to a set of processors, each of which executes a different instruction sequence. This structure is not commercially implemented.
4. **Multiple instruction, multiple data (MIMD) stream:** A sequence of processors simultaneously execute different instruction sequences on different data sets. Symmetric multiprocessors (SMPs), clusters, and non-uniform memory access(NUMA) systems fit into this category.

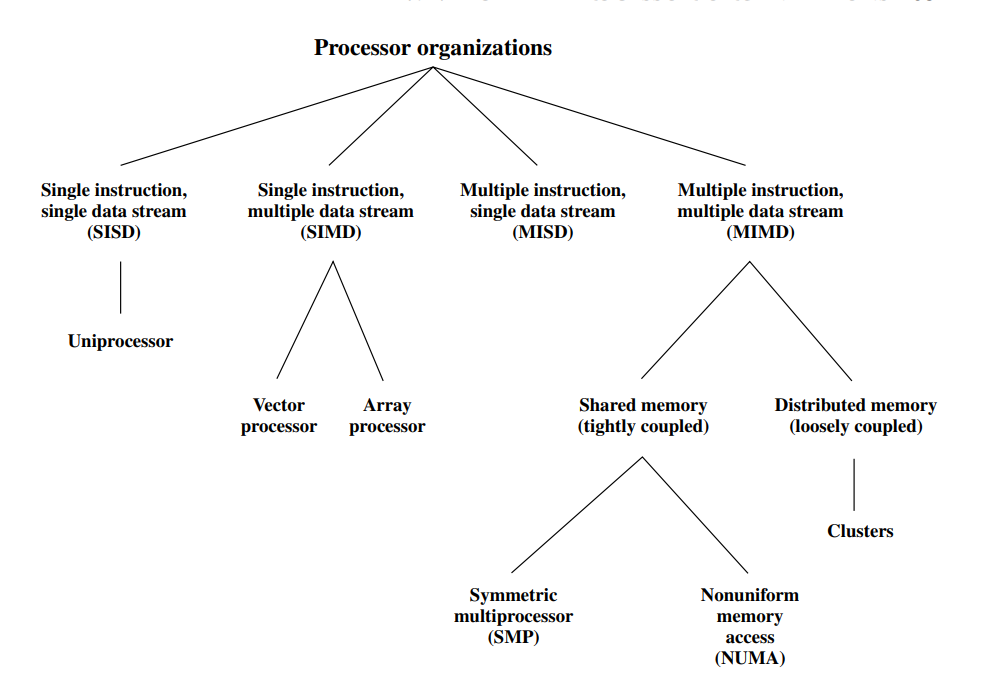


Figure: A Taxonomy of Parallel Processor Architectures

With the MIMD organization, the processors are general purpose; each is able to process all of the instructions necessary to perform the appropriate data transformation. MIMDs can further subdivided by the means in which the processors communicate in the figure above.

***Symmetric Multiprocessors (SMPs)***

SMP refers to a computer hardware architecture and also to the operating system behavior that reflects that architecture. m SMP refers to a computer hardware architecture and also to the operating system behavior that reflects that architecture. SMP can be defined as a standalone computer system with the following characteristics:

1. There are two or more similar processors of comparable capability.
2. These processors share the same main memory and I/O facilities and are interconnected by a bus or other internal connection scheme, such that memory access time is approximately the same for each processor.
3. All processors share access to I/O devices, either through the same channels or through different channels that provide paths to the same device.
4. All processors can perform the same functions (hence the term symmetric).
5. The system is controlled by an integrated operating system that provides interaction between processors and their programs at the job, task, file, and data element levels.

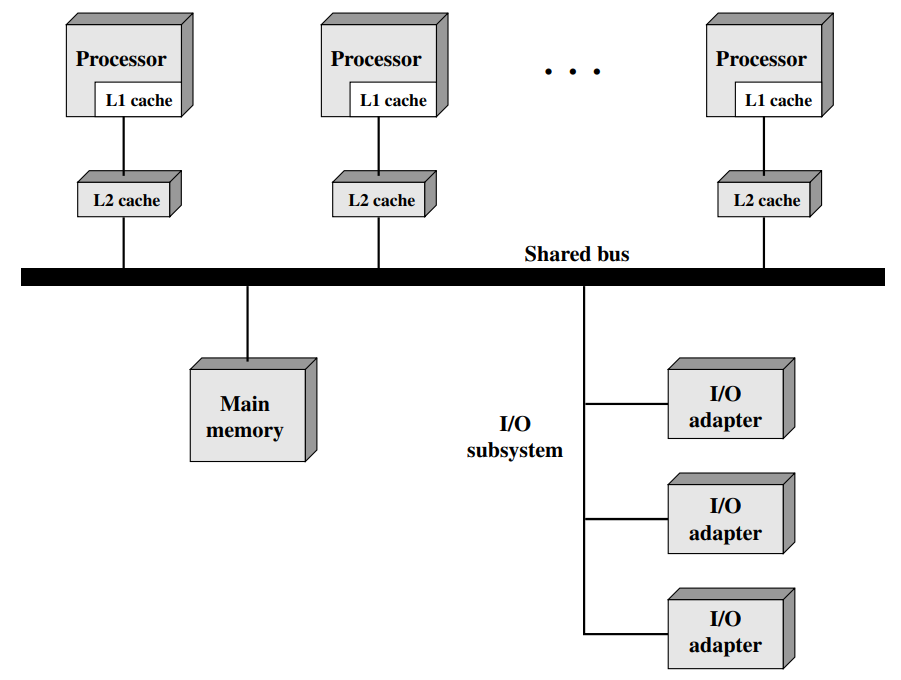
The operating system of an SMP schedules processes or threads across all of the processors. An SMP organization has a number of potential advantages over a uniprocessor organization, including the following:

* Performance: If the work to be done by a computer can be organized so that some portions of the work can be done in parallel, then a system with multiple processors will yield greater performance than one with a single processor of the same type (
* Availability: In a symmetric multiprocessor, because all processors can perform the same functions, the failure of a single processor does not halt the machine. Instead, the system can continue to function at reduced performance.
* Incremental growth: A user can enhance the performance of a system by adding an additional processor.
* Scaling: Vendors can offer a range of products with different price and performance characteristics based on the number of processors configured in the system.

***Organization of Symmetric Multiprocessors (SMPs)***

There are two or more processors. Each processor is self-contained, including a control unit, ALU, registers, and, typically, one or more levels of cache. Each processor has access to a shared main memory and the I/O devices through some from of interconnection mechanism. The processors can communicate with each other through memory (messages and status information left in common data areas). It may also be possible for processors to exchange signals directly. The memory is often organized so that multiple simultaneous accesses to separate blocks of memory are possible. In some configurations, each processor may also have its own private main memory and I/O channels in addition to the shared resources.

The most common organization for personal computers, workstations, and servers is the time-shared bus. The time-shared bus is the simplest mechanism for constructing a multiprocessor system as shown in the figure below. The structure and interfaces are basically the same as for a single-processor system that uses a bus interconnection. The bus consists of control, address, and data lines.



The bus organization has several attractive features:

* Simplicity: This is the simplest approach to multiprocessor organization. The physical interface and the addressing, arbitration, and time-sharing logic of each processor remain the same as in a single-processor system.
* Flexibility: It is generally easy to expand the system by attaching more processors to the bus.
* Reliability: The bus is essentially a passive medium, and the failure of any attached device should not cause failure of the whole system.

The main drawback to the bus organization is performance. All memory references pass through the common bus. Thus, the bus cycle time limits the speed of the system. To improve performance, it is desirable to equip each processor with a cache memory. This should reduce the number of bus accesses dramatically. Typically, workstation and PC SMPs have two levels of cache, with the L1 cache internal (same chip as the processor) and the L2 cache either internal or external. Some processors now employ a L3 cache as well. The use of caches introduces some new design considerations. Because each local cache contains an image of a portion of memory, if a word is altered in one cache, it could conceivably invalidate a word in another cache. To prevent this, the other processors must be alerted that an update has taken place. This problem is known as the cache coherence problem and is typically addressed in hardware rather than by the operating system.

***Multiprocessor Operating System Design Considerations***

In both the SMP and uniprocessor cases, multiple jobs or processes may be active at one time, and it is the responsibility of the operating system to schedule their execution and to allocate resources. A user may construct applications that use multiple processes or multiple threads within processes without regard to whether a single processor or multiple processors will be available. Thus, a multiprocessor operating system must provide all the functionality of a multiprogramming system plus additional features to accommodate multiple processors. Among the key design issues:

* **Simultaneous concurrent processes:** OS routines need to be reentrant to allow several processors to execute the same IS code simultaneously. With multiple processors executing the same or different parts of the OS, OS tables and management structures must be managed properly to avoid deadlock or invalid operations.
* **Scheduling:** Any processor may perform scheduling, so conflicts must be avoided. The scheduler must assign ready processes to available processors.
* **Synchronization:** With multiple active processes having potential access to shared address spaces or shared I/O resources, care must be taken to provide effective synchronization. Synchronization is a facility that enforces mutual exclusion and event ordering.
* **Memory management:** Memory management on a multiprocessor must deal with all of the issues found on uniprocessor machines. In addition, the operating system needs to exploit the available hardware parallelism, such as multi-ported memories, to achieve the best performance. The paging mechanisms on different processors must be coordinated to enforce consistency when several processors share a page or segment and to decide on page replacement.
* **Reliability and fault tolerance:** The operating system should provide graceful degradation in the face of processor failure. The scheduler and other portions of the operating system must recognize the loss of a processor and restructure management tables accordingly.