+ Content

Chapter 17

Parallel Processing

- Multiple processor organizations
  - Types of parallel processor systems
  - Parallel organizations
- Symmetric multiprocessors
  - Organization
  - Multiprocessor operating system design considerations
- Clusters
  - Cluster configurations
  - Operating system design issues
  - Cluster computer architecture
  - Blade servers
  - Clusters compared to SMP

Multiple Processor Organization

- Single instruction, single data **(SISD)** stream
  - Single processor executes a single instruction stream to operate on data stored in a single memory
  - Uniprocessors fall into this category
- Single instruction, multiple data **(SIMD)** stream
  - A single machine instruction controls the simultaneous execution of a number of processing elements on a lockstep basis
  - Vector and array processors fall into this category
- 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
  - Not commercially implemented
- Multiple instruction, multiple data **(MIMD)** stream
  - A set of processors simultaneously execute different instruction sequences on different data sets
  - SMPs, clusters and NUMA systems fit this category

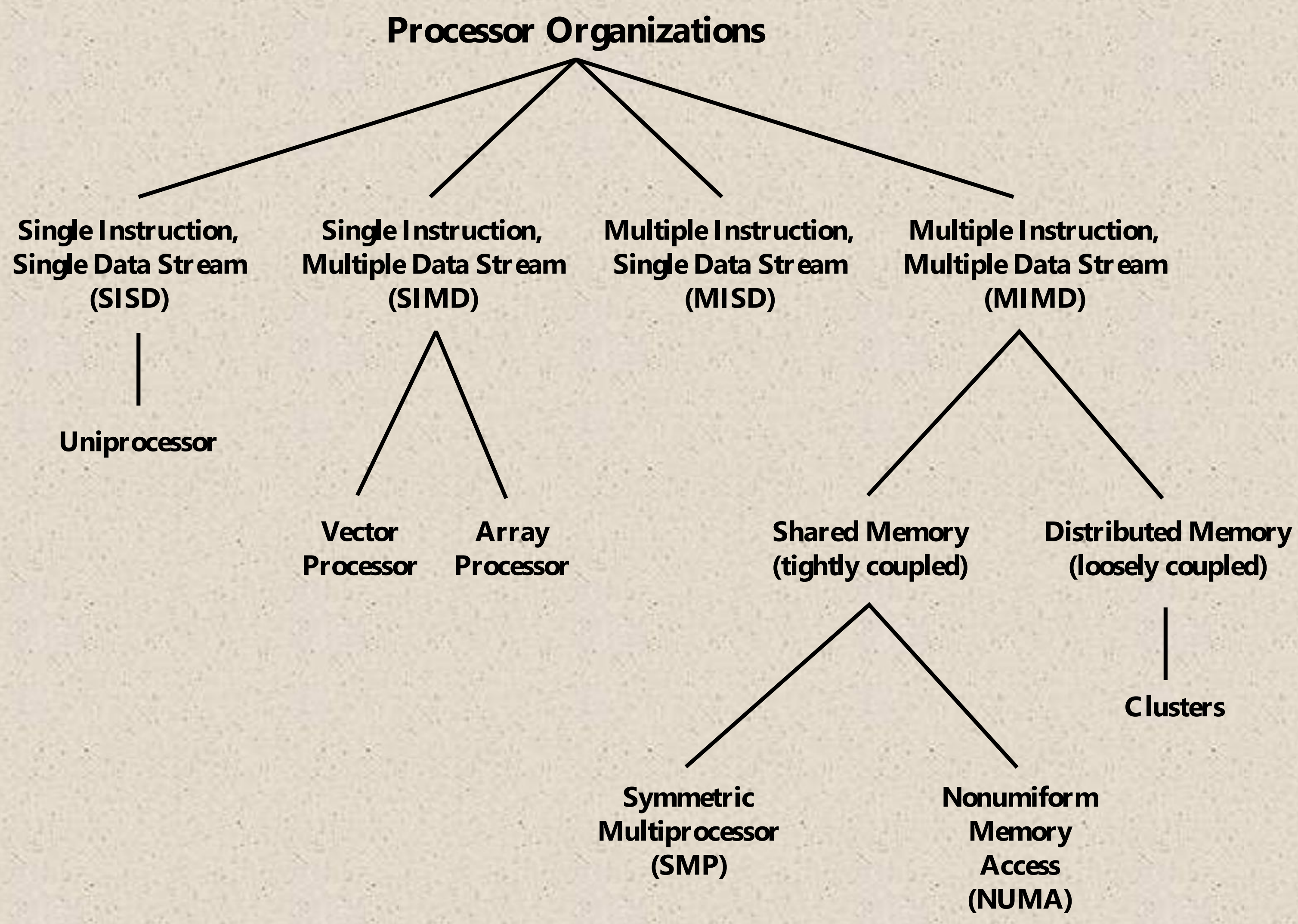
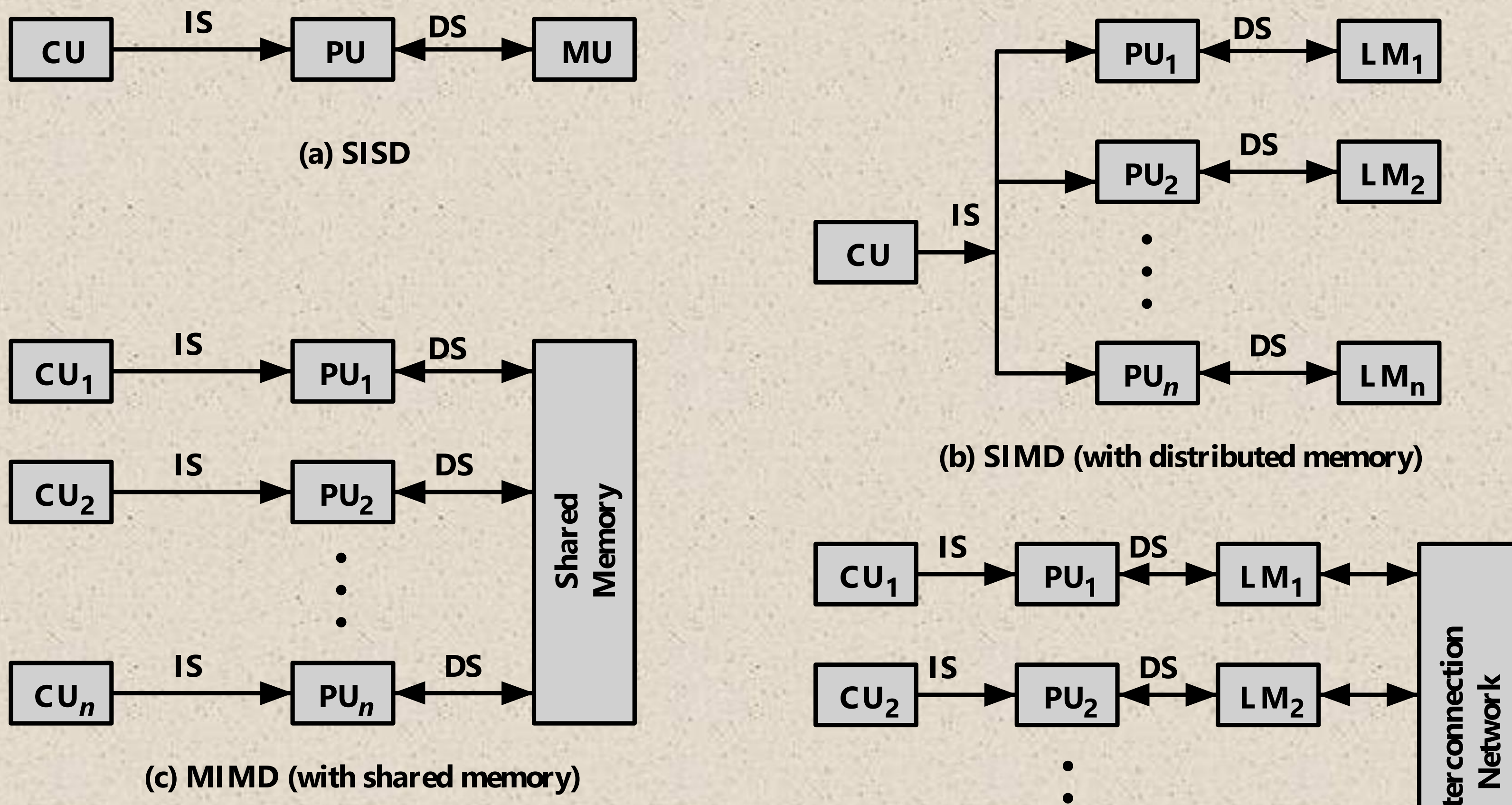


Figure 17.1 A Taxonomy of Parallel Processor Architectures



CU = control unit
IS = instruction stream
PU = processing unit
DS = data stream
MU = memory unit
LM = local memory

SISD = single instruction, single data stream
SIMD = single instruction, multiple data stream
MIMD = multiple instruction, multiple data stream

Figure 17.2 Alternative Computer Organizations

Symmetric Multiprocessor (SMP)

A stand alone computer with the following characteristics:

- Two or more similar processors of comparable capacity
- Processors share same memory and I/O facilities
  - Processors are connected by a bus or other internal connection
  - Memory access time is approximately the same for each processor
- All processors share access to I/O devices
  - Either through same channels or different channels giving paths to same devices
- All processors can perform the same functions (hence "symmetric")
- System controlled by integrated operating system
  - Provides interaction between processors and their programs at job, task, file and data element levels

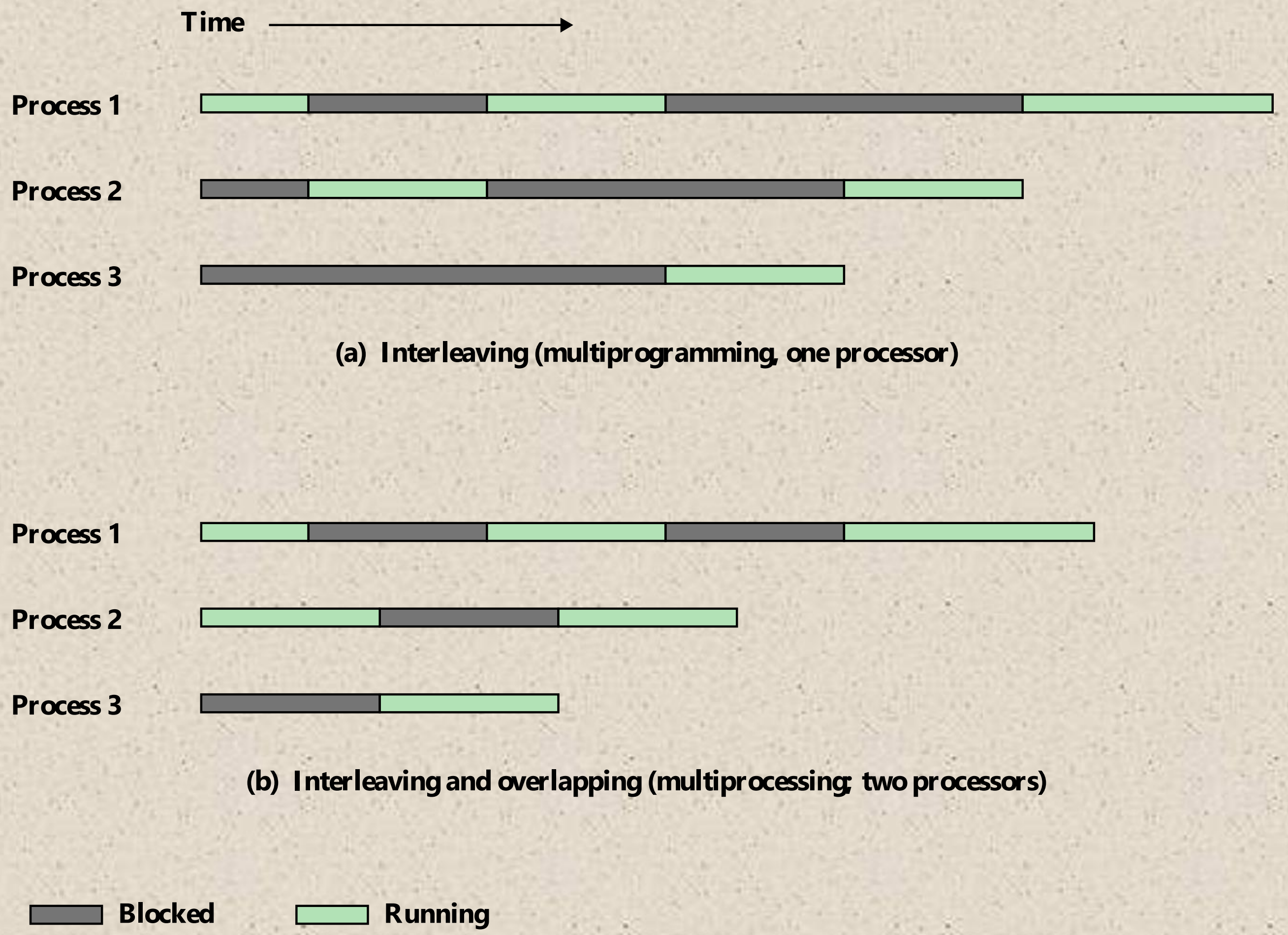


Figure 17.3 Multiprogramming and Multiprocessing

SMP Advantages

- Performance
  - If some work can be done in parallel
- Availability
  - Since all processors can perform the same functions, failure of a single processor does not halt the system
- Incremental growth
  - User can enhance performance by adding additional processors
- Scaling
  - Vendors can offer range of products based on number of processors

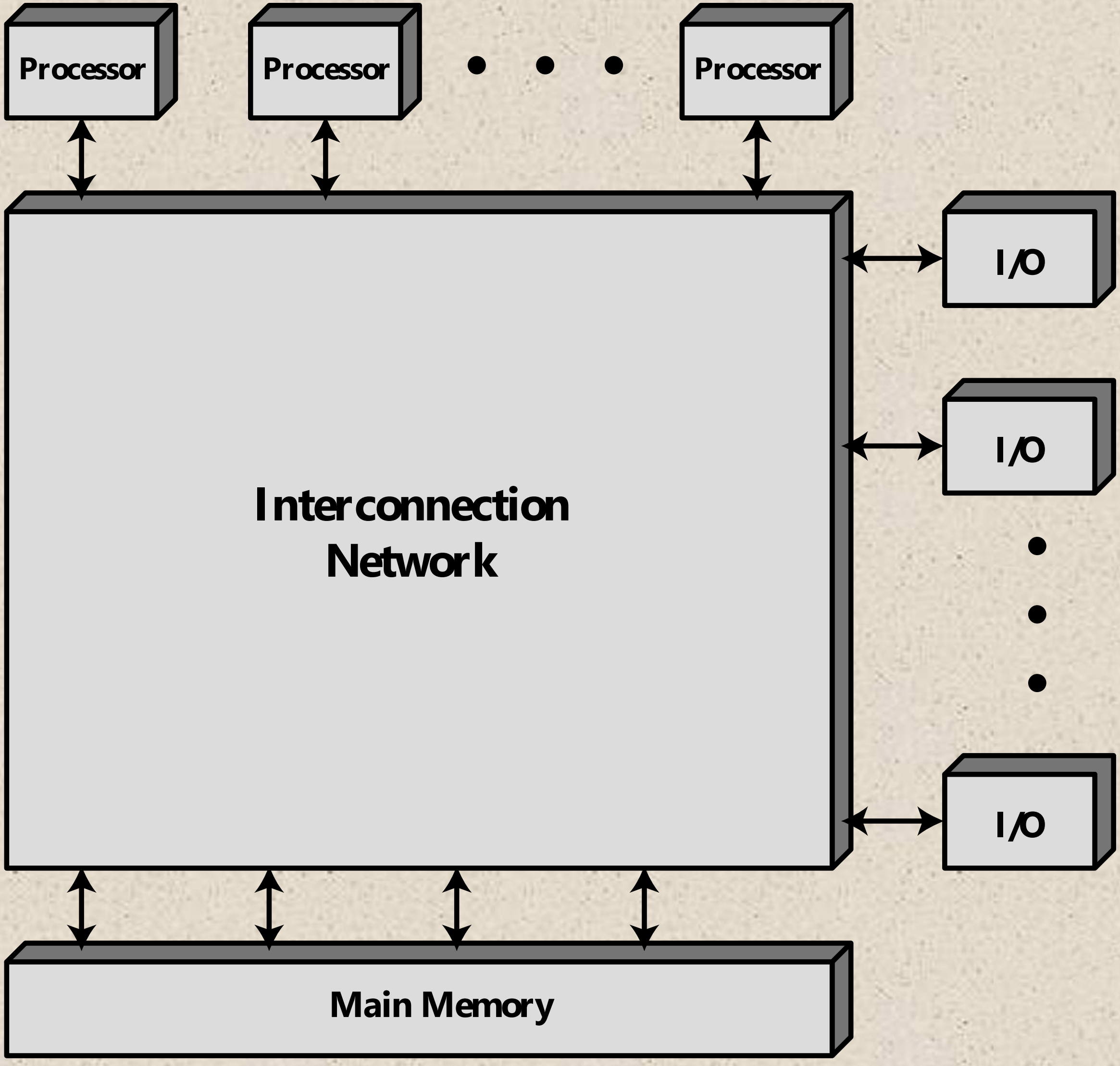


Figure 17.4 Generic Block Diagram of a Tightly Coupled Multiprocessor

+

Time Shared Bus

- Simplest form
- Structure and interface similar to single processor system
- Following features provided
  - Addressing - distinguish modules on bus
  - Arbitration - any module can be temporary master
  - Time sharing - if one module has the bus, others must wait and may have to suspend
- Now have multiple processors as well as multiple I/O modules

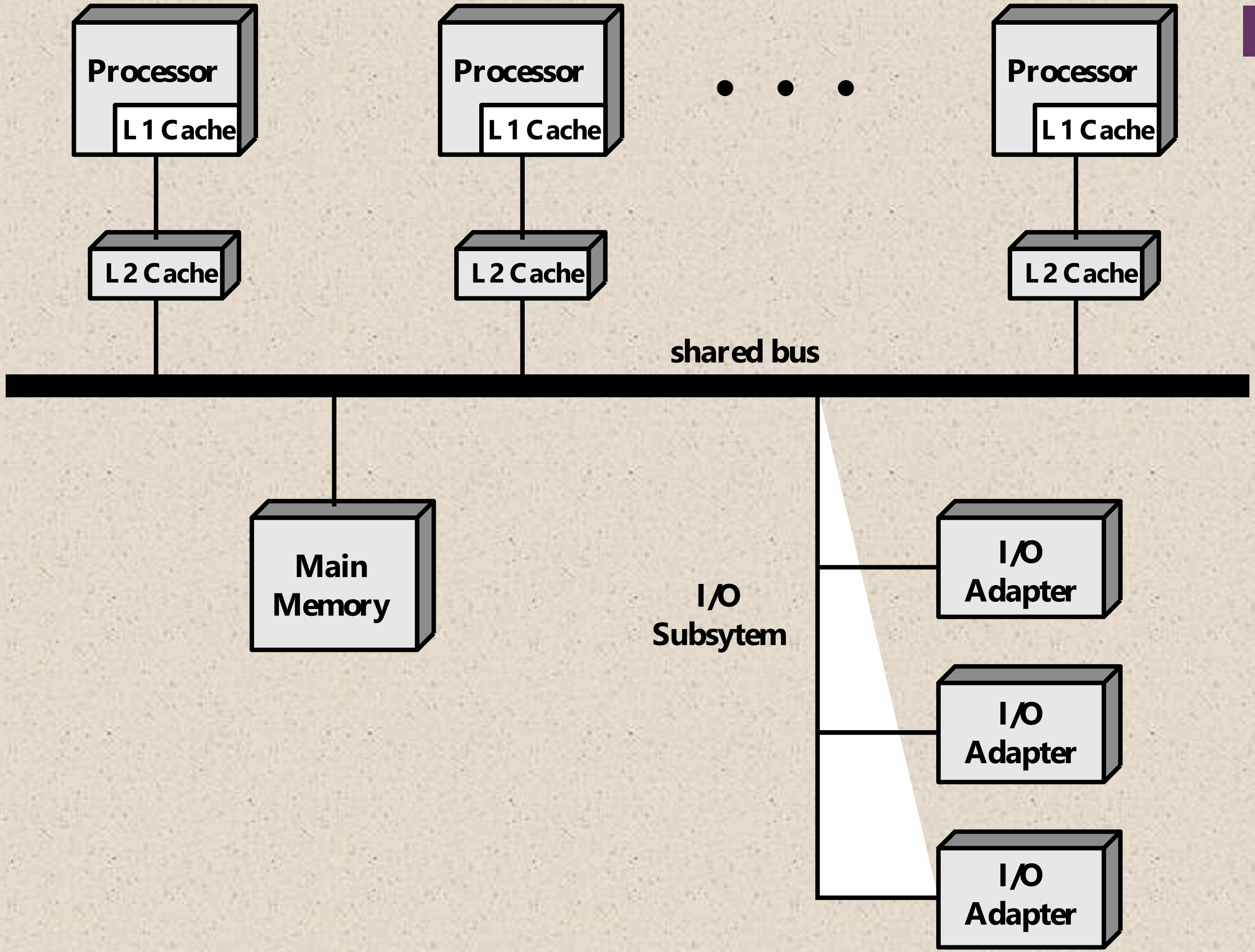
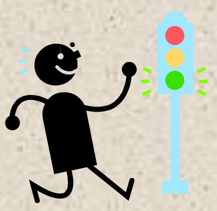


Figure 17.5 Symmetric Multiprocessor Organization

+

The bus organization has several attractive features:



- Simplicity
  - Simplest approach to multiprocessor organization
- Flexibility
  - 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

Disadvantages of the bus organization:

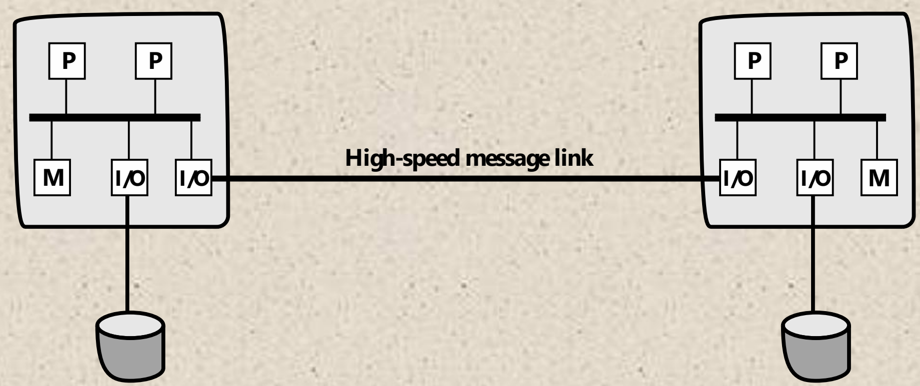
- Main drawback is performance
  - All memory references pass through the common bus
  - Performance is limited by bus cycle time
- Each processor should have cache memory
  - Reduces the number of bus accesses
- Leads to problems with *cache coherence*
  - If a word is altered in one cache it could conceivably invalidate a word in another cache
    - To prevent this the other processors must be alerted that an update has taken place
  - Typically addressed in hardware rather than the operating system

Multiprocessor Operating System Design Considerations

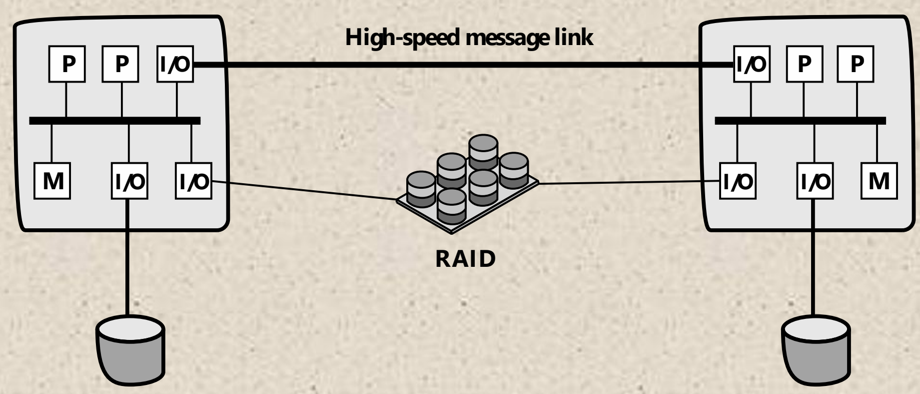
- **Simultaneous concurrent processes**
  - OS routines need to be reentrant to allow several processors to execute the same IS code simultaneously
  - OS tables and management structures must be managed properly to avoid deadlock or invalid operations
- **Scheduling**
  - Any processor may perform scheduling so conflicts must be avoided
  - Scheduler must assign ready processes to available processors
- **Synchronization**
  - With multiple active processes having potential access to shared address spaces or I/O resources, care must be taken to provide effective synchronization
  - Synchronization is a facility that enforces mutual exclusion and event ordering
- **Memory management**
  - In addition to dealing with all of the issues found on uniprocessor machines, the OS needs to exploit the available hardware parallelism to achieve the best performance
  - 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**
  - OS should provide graceful degradation in the face of processor failure
  - Scheduler and other portions of the operating system must recognize the loss of a processor and restructure accordingly

Clusters

- Alternative to SMP as an approach to providing high performance and high availability
- Particularly attractive for server applications
- Defined as:
  - A group of interconnected whole computers working together as a unified computing resource that can create the illusion of being one machine
  - (The term *whole computer* means a system that can run on its own, apart from the cluster)
- Each computer in a cluster is called a node
- Benefits:
  - Absolute scalability
  - Incremental scalability
  - High availability
  - Superior price/performance



(a) Standby server with no shared disk



(b) Shared disk

Figure 17.8 Cluster Configurations

Table 17.2
Clustering Methods: Benefits and Limitations

Clustering Method	Description	Benefits	Limitations
Passive Standby	A secondary server takes over in case of primary server failure.	Easy to implement.	High cost because the secondary server is unavailable for other processing tasks.
Active Secondary:	The secondary server is also used for processing tasks.	Reduced cost because secondary servers can be used for processing.	Increased complexity.
Separate Servers	Separate servers have their own disks. Data is continuously copied from primary to secondary server.	High availability.	High network and server overhead due to copying operations.
Servers Connected to Disks	Servers are cabled to the same disks, but each server owns its disks. If one server fails, its disks are taken over by the other server.	Reduced network and server overhead due to elimination of copying operations.	Usually requires disk mirroring or RAID technology to compensate for risk of disk failure.
Servers Share Disks	Multiple servers simultaneously share access to disks.	Low network and server overhead. Reduced risk of downtime caused by disk failure.	Requires lock manager software. Usually used with disk mirroring or RAID technology.

Operating System Design Issues

- How failures are managed depends on the clustering method used
- Two approaches:
  - Highly available clusters
  - Fault tolerant clusters
- Failover
  - The function of switching applications and data resources over from a failed system to an alternative system in the cluster
- Failback
  - Restoration of applications and data resources to the original system once it has been fixed
- Load balancing
  - Incremental scalability
  - Automatically include new computers in scheduling
  - Middleware needs to recognize that processes may switch between machines

Parallelizing Computation

Effective use of a cluster requires executing software from a single application in parallel

Three approaches are:

Parallelizing complier

- Determines at compile time which parts of an application can be executed in parallel
- These are then split off to be assigned to different computers in the cluster

Parallelized application

- Application written from the outset to run on a cluster and uses message passing to move data between cluster nodes

Parametric computing

- Can be used if the essence of the application is an algorithm or program that must be executed a large number of times, each time with a different set of starting conditions or parameters

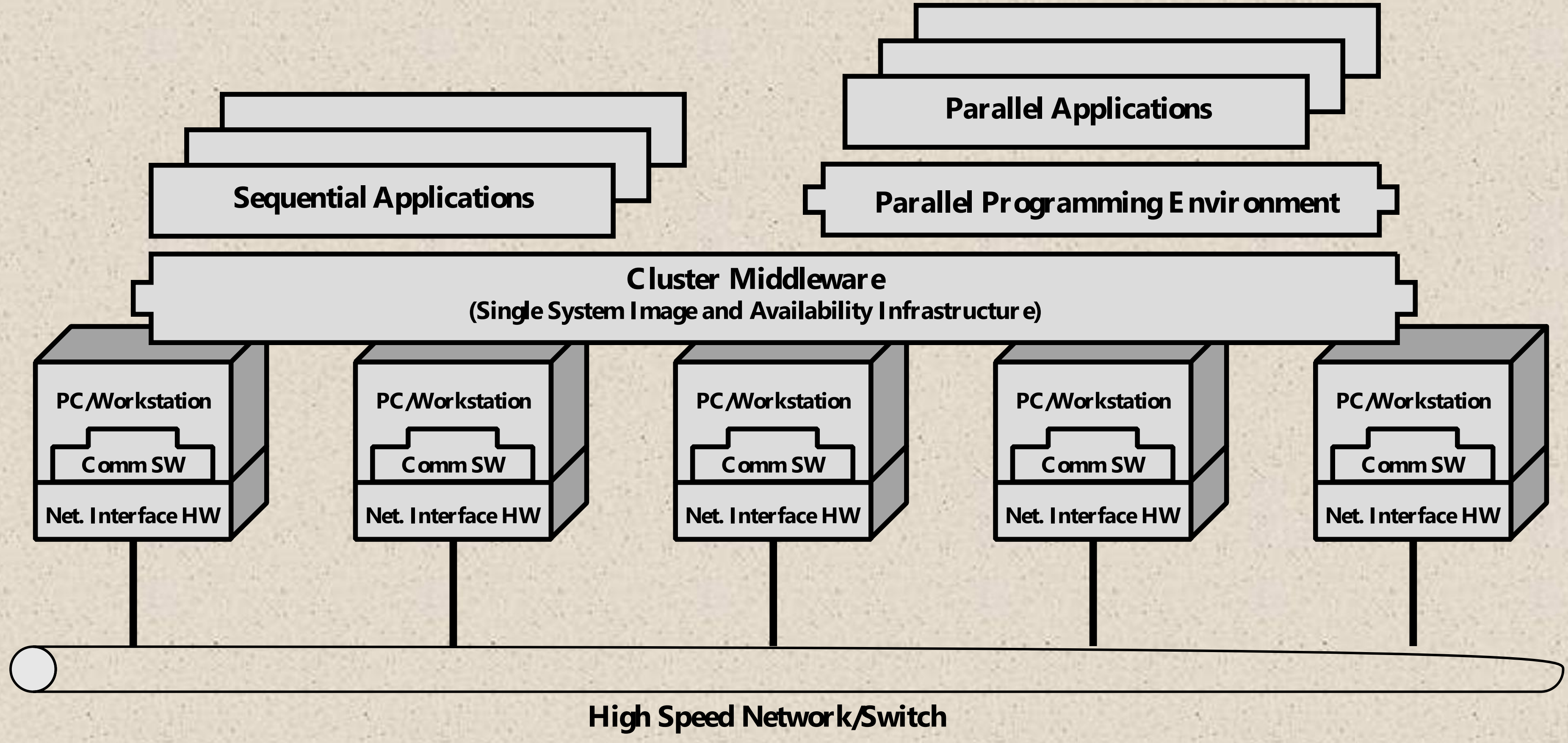


Figure 17.9 Cluster Computer Architecture

Clusters Compared to SMP

- Both provide a configuration with multiple processors to support high demand applications
- Both solutions are available commercially

SMP	Clustering
Easier to manage and configure	Far superior in terms of incremental and absolute scalability
Much closer to the original single processor model for which nearly all applications are written	Superior in terms of availability
Less physical space and lower power consumption	All components of the system can readily be made highly redundant
Well established and stable	

Thank You….!!!

All The Best