Chapter 7
Multi-processing

Ref: Computer Organization and Architecture, 8th Edition, William Stalling

Content

- Types of Parallel Processor Systems
- SMP
- Cluster
- NUMA
- Multicore processors

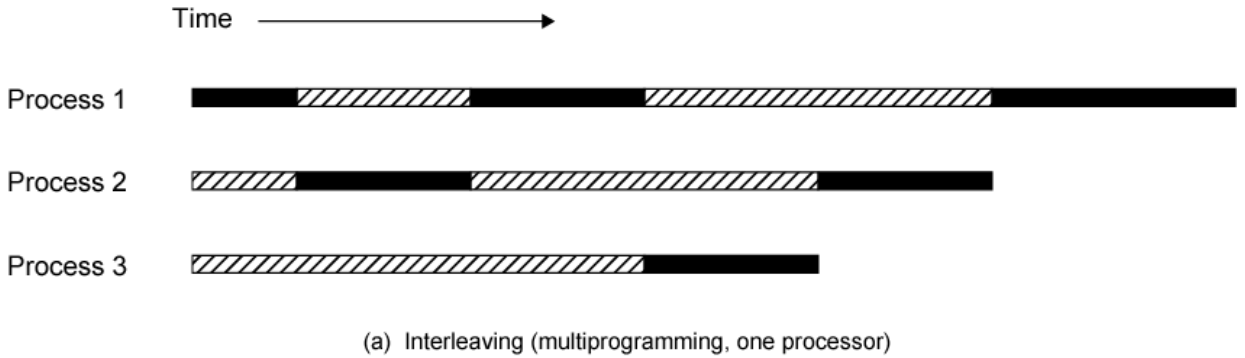
Introduction

- Overall goal: increasing performance
  - For a large software
  - For a large number of small individual software
  - With energy efficiency
- Approaches in previous chapters?
  - Pipeline
  - Super scaler
  - Multi-threaded

    → increase performance of a single CPU core

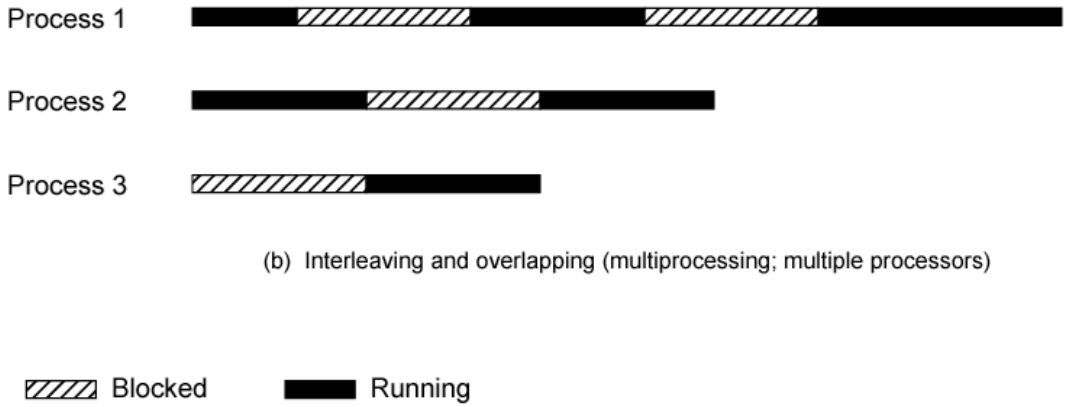
Multi-thread vs multi-processing

- Multi-thread



(a) Interleaving (multiprogramming, one processor)

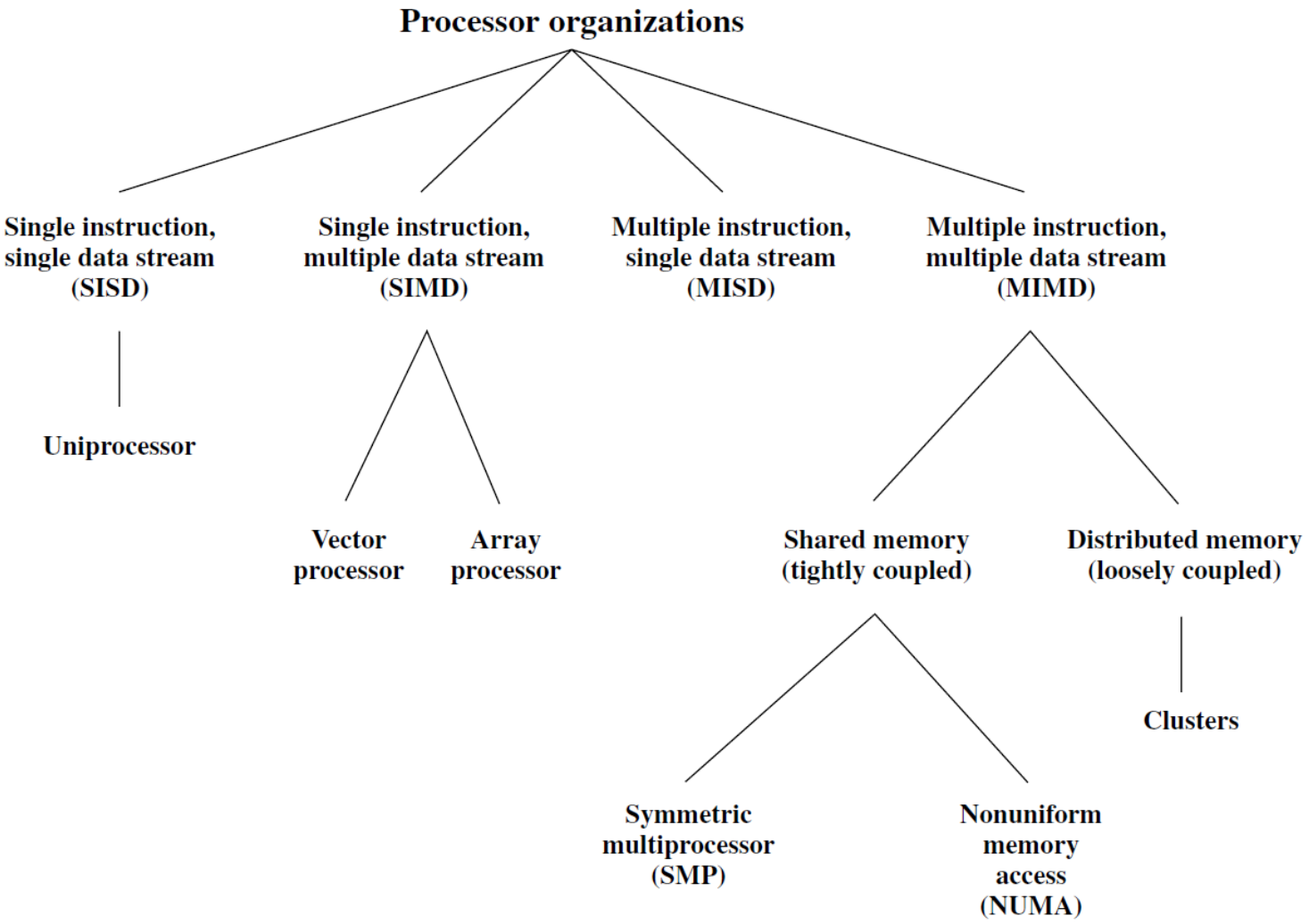
- Multi-processing



(b) Interleaving and overlapping (multiprocessing; multiple processors)

- Comparison?

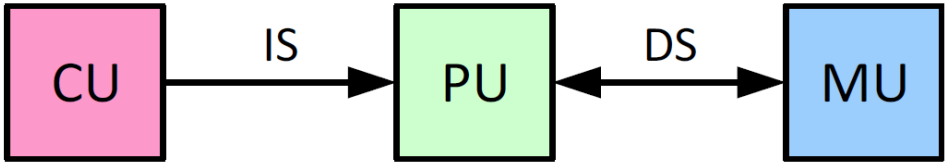
Types of Parallel Processor Systems



Types of Parallel Processor Systems

- Single instruction, single data (**SISD**) stream: A single processor executes a single instruction stream to operate on data stored in a single memory.
- Single instruction, multiple data (**SIMD**) stream: A single machine instruction controls the simultaneous execution of a number of processing elements
- Multiple instruction, single data (**MISD**) stream: not implemented
- Multiple instruction, multiple data (**MIMD**) stream: A set of processors simultaneously execute different instruction sequences on different data sets.

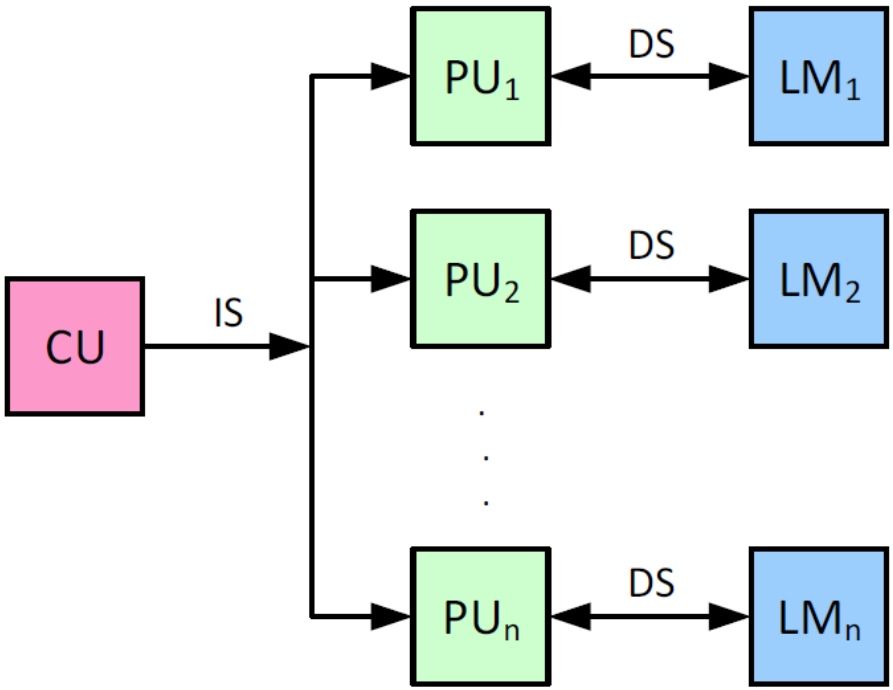
SISD



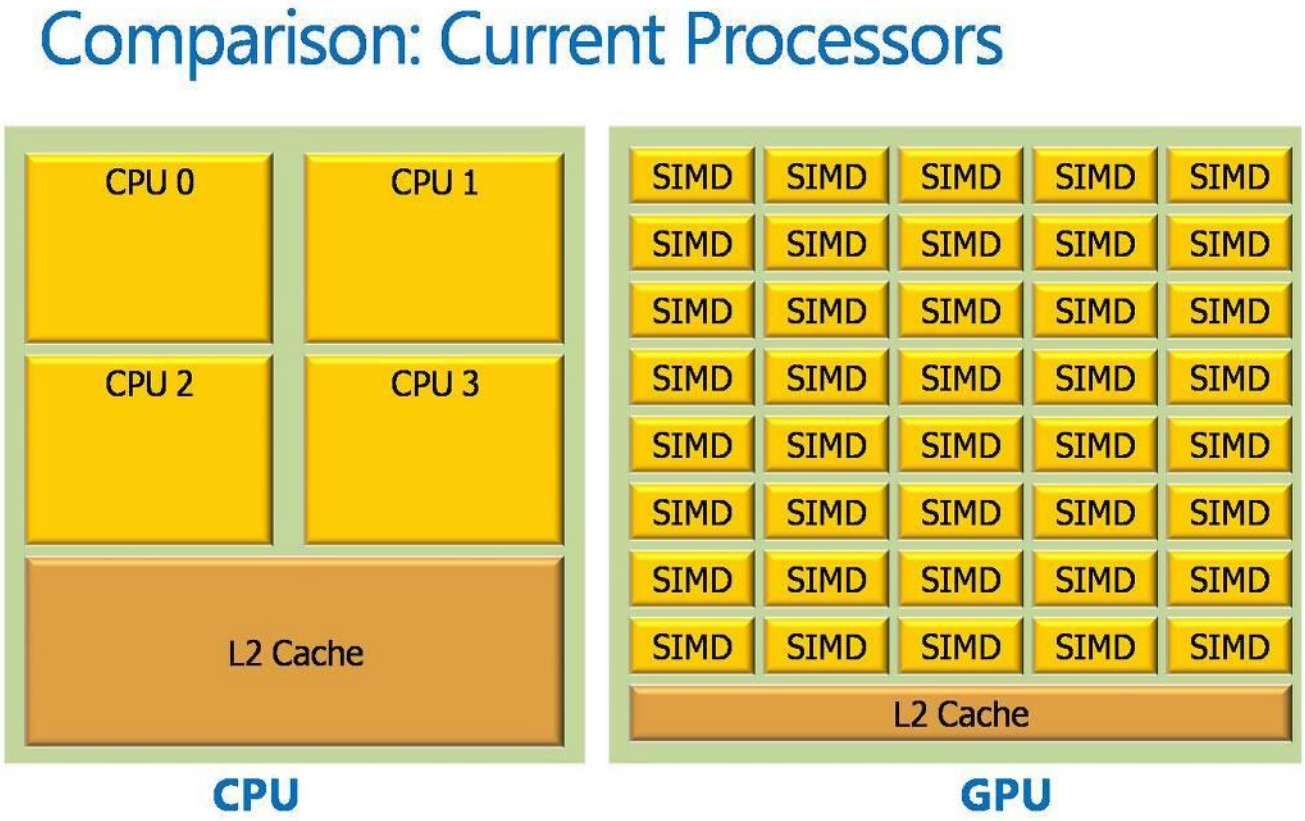
- CU: Control Unit
- PU: Processing Unit
- MU: Memory Unit
- Sequential execution
- Data stored in a single main memory
- ➔ Uniprocessor computer

SIMD

- 1 instruction stream
- Multiple processing units
- Each PC processes data from a separate memory
- All PUs execute the same instruction stream from CU
- Example: GPU

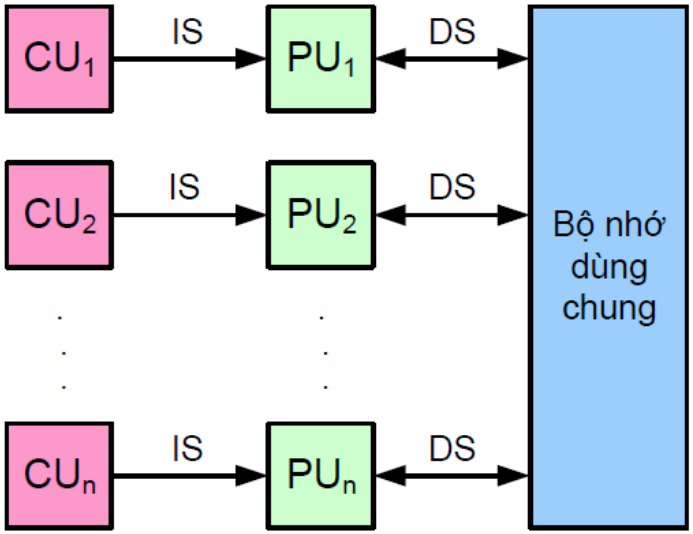


CPU vs GPU

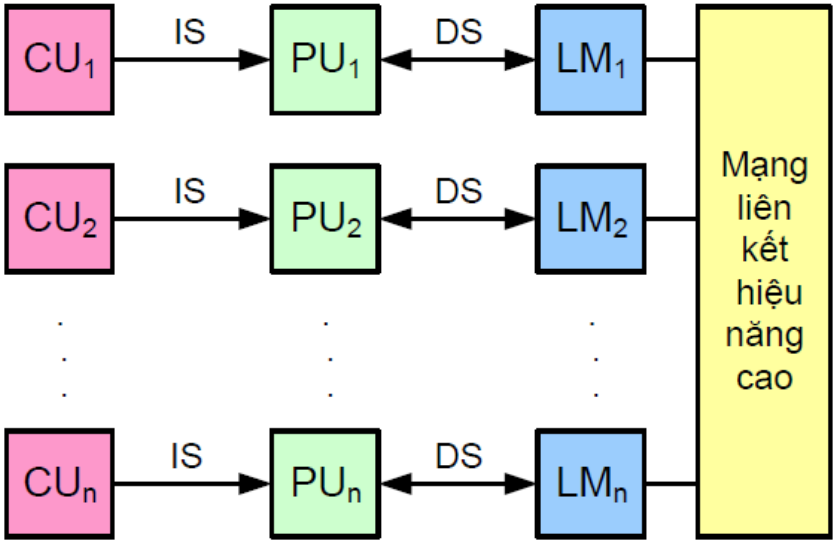

Comparison: Current Processors
CPU 0
CPU 1
CPU 2
CPU 3
L2 Cache
CPU
SIMD
L2 Cache
GPU

MIMD

- Multiple instruction, multiple data
- Require multiple CUs and PUs
- Shared or distributed memory



MIMD with shared memory



MIMD with distributed memory

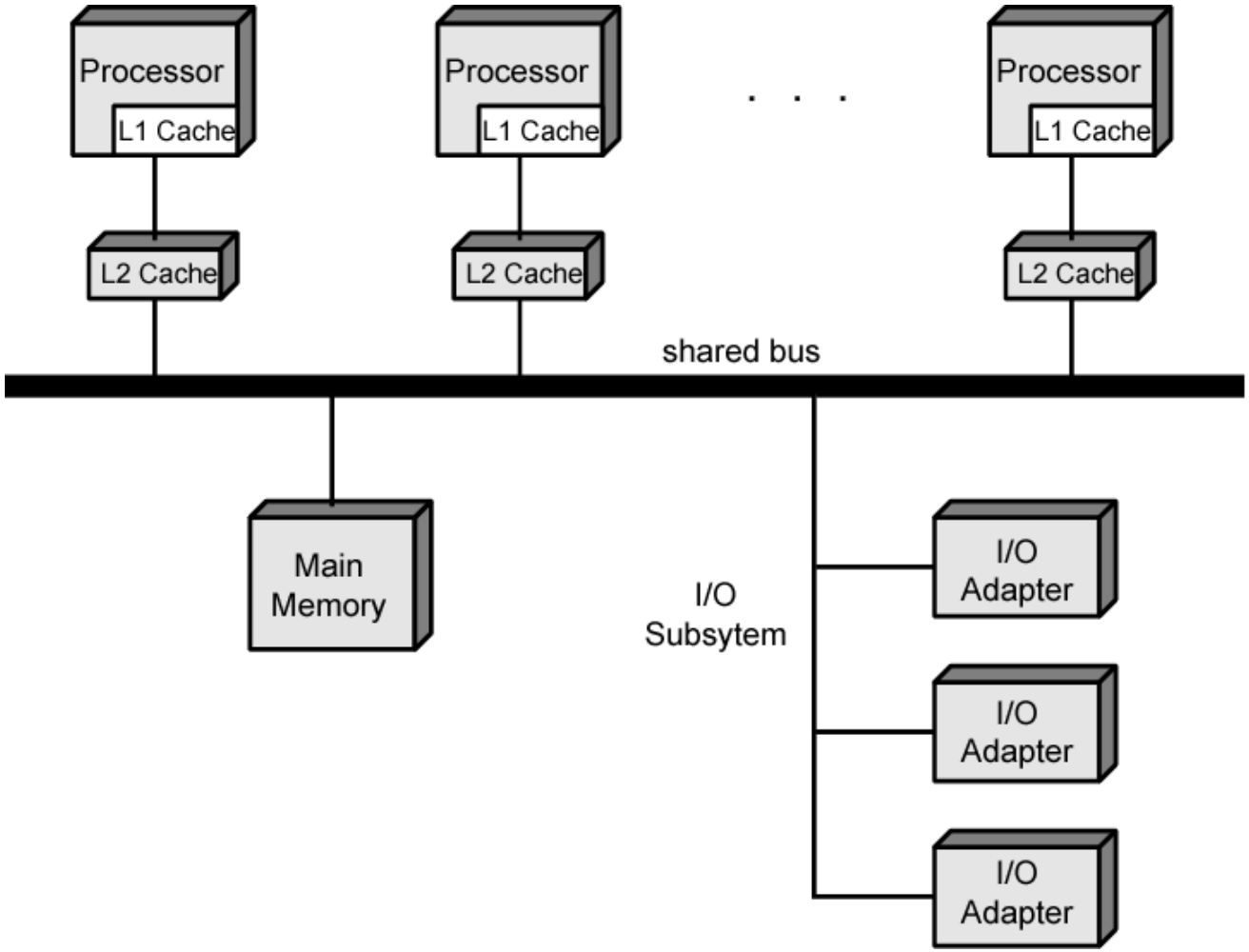
Types of MIMD

- Tightly coupled: standalone CPUs connect to memory and IOs by system buses.
  - Symmetric Multiprocessor: e.g. multi-core CPU.
  - Non-uniform Memory Access.
- Loosely coupled: CPUs are connected via high speed network connections

Symmetric Multiprocessor (SMP)

- Two or more similar processors of comparable capability
- These processors share the same main memory and I/O facilities
- All processors share access to I/O devices
- All processors can perform the same functions (*symmetric*)
- The system is controlled by an integrated operating system
  - Provides interaction between processors and system resources

Symmetric Multiprocessor Organization



SMP Design Considerations (1)

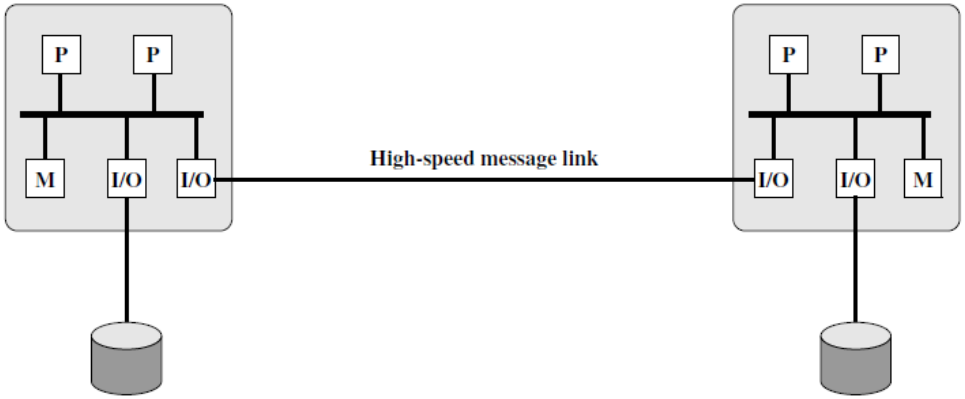
- Hardware
  - ***Cache coherence***: Each processor has a separate cache. A single data on main memory can be mapped to multiple cache on different CPUs.
- Software (OS):
  - Simultaneous concurrent processes.
  - Multi-processer scheduling.
  - Synchronization.
  - Memory management
  - Reliability and fault tolerance

Cluster

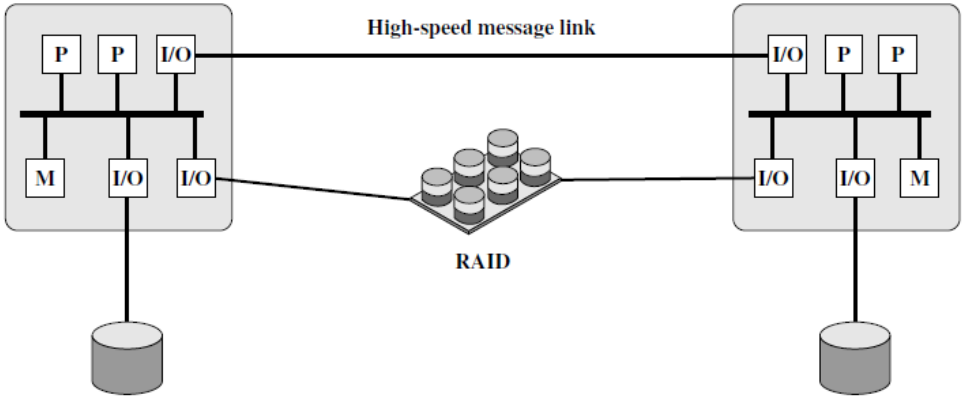
- Group of interconnected, whole computers working together as a unified computing resource. Each computer in a cluster is typically referred to as a node.

- Absolute scalability
- Incremental scalability
- High availability
- Superior price/performance

Cluster Configurations

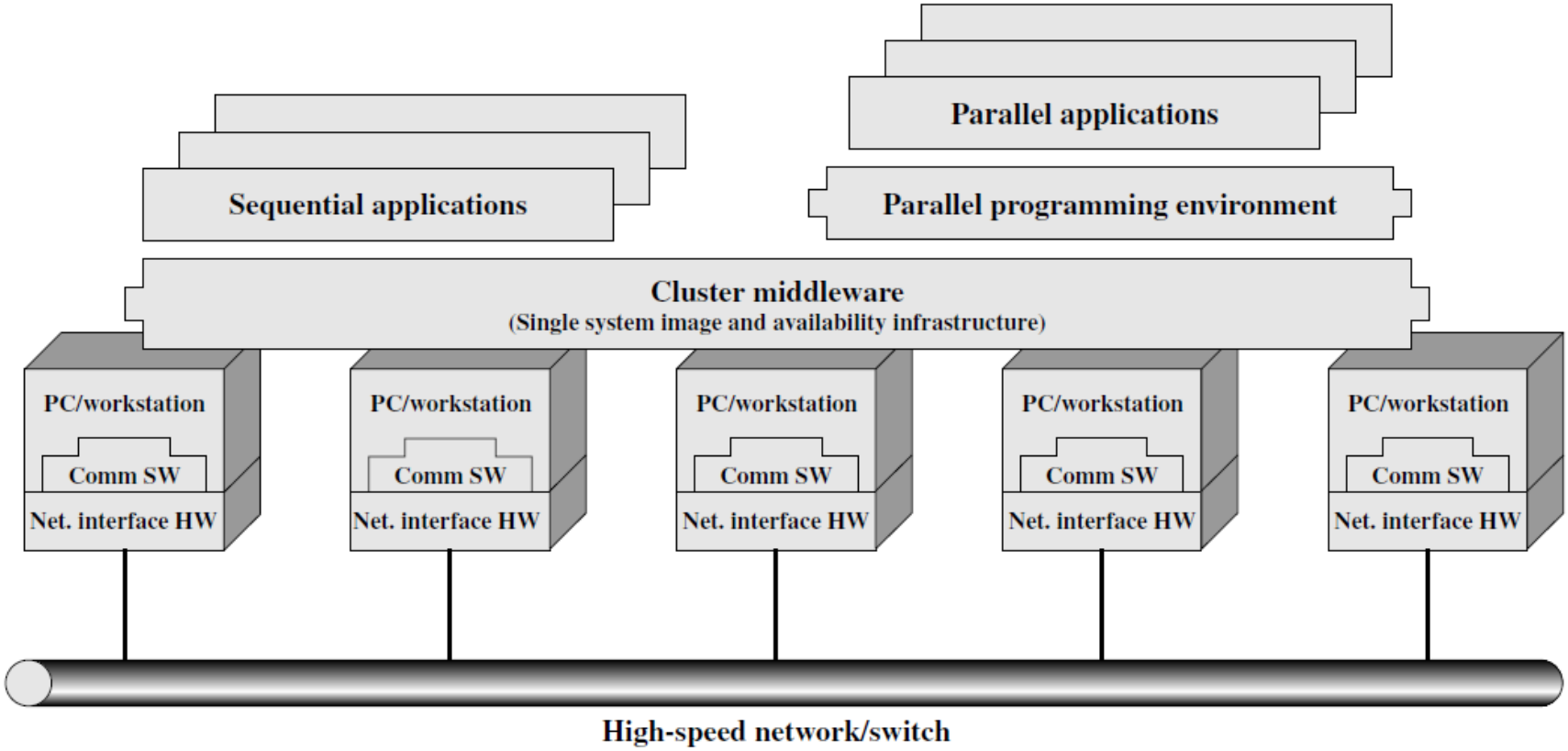


(a) Standby server with no shared disk

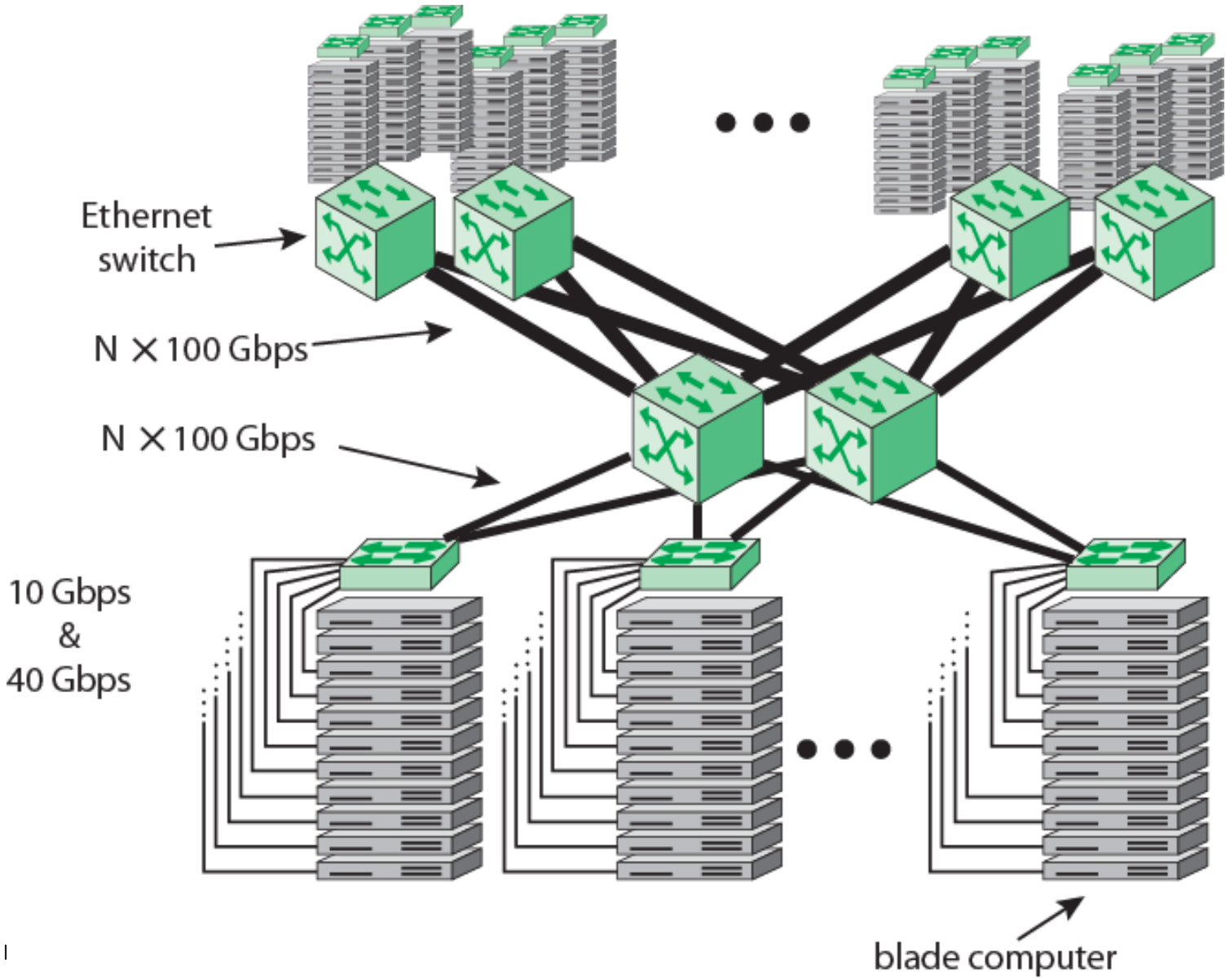


(b) Shared Disk

Cluster Computer Architecture



Example: blade server



SMP vs cluster

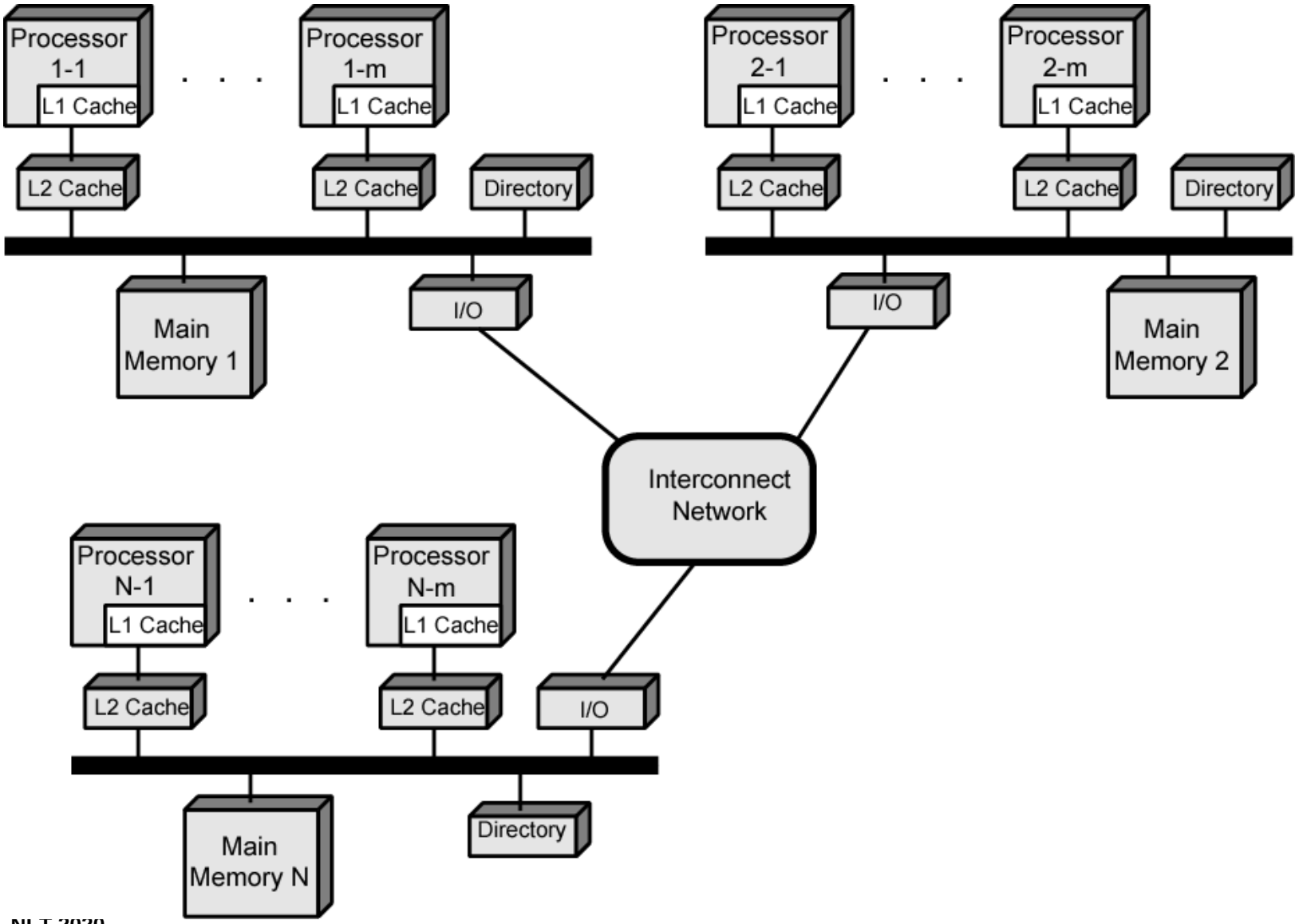
- Both are high performance computer architecture
- SMP
  - Easy to use and maintenance
  - Closer to uniprocessor system
  - Small size and low power consumption
- Cluster
  - High computing capability
  - Scalability
  - Hight dependability and availability

SMP vs cluster

- SMP
  - Limited capability.
- Cluster
  - Separated memory on each node
  - Complicated software

➔ combining advantages of SMP and cluster: NUMA

Cache-coherence NUMA

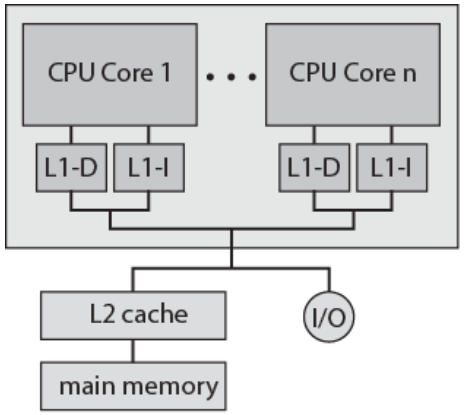


Multicore Processors

- Conventional performance improvement
  - Pipelining
  - Superscalar
  - Multithreading
  - ➔ increasingly difficult engineering challenge in CPU design
- Pollack:
  - “Performance is roughly proportional to square root of increase in complexity”
  - double the logic in a processor core, then it delivers only 40% more performance
- The use of multiple cores has the potential to provide near-linear performance improvement with the increase in the number of cores

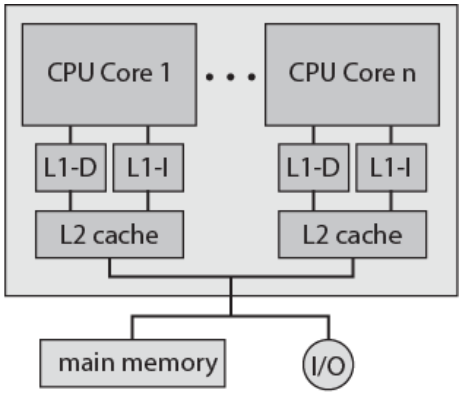
Multicore Organization Alternatives

ARM11 MPCore



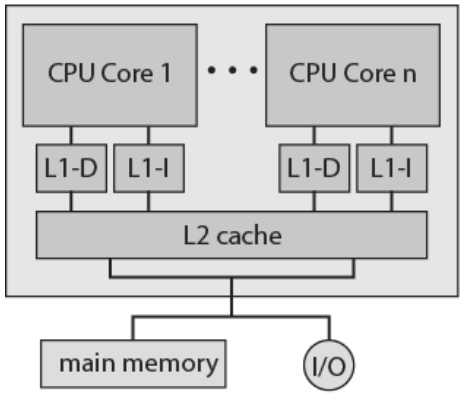
(a) Dedicated L1 cache

AMD Opteron



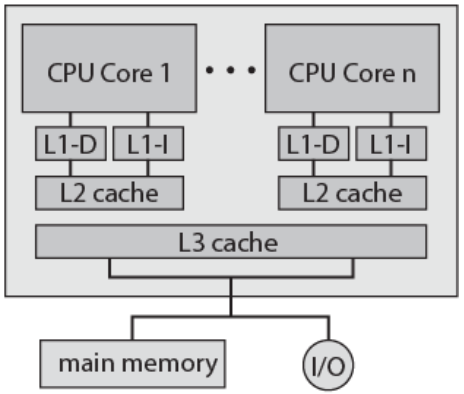
(b) Dedicated L2 cache

Intel Core Duo



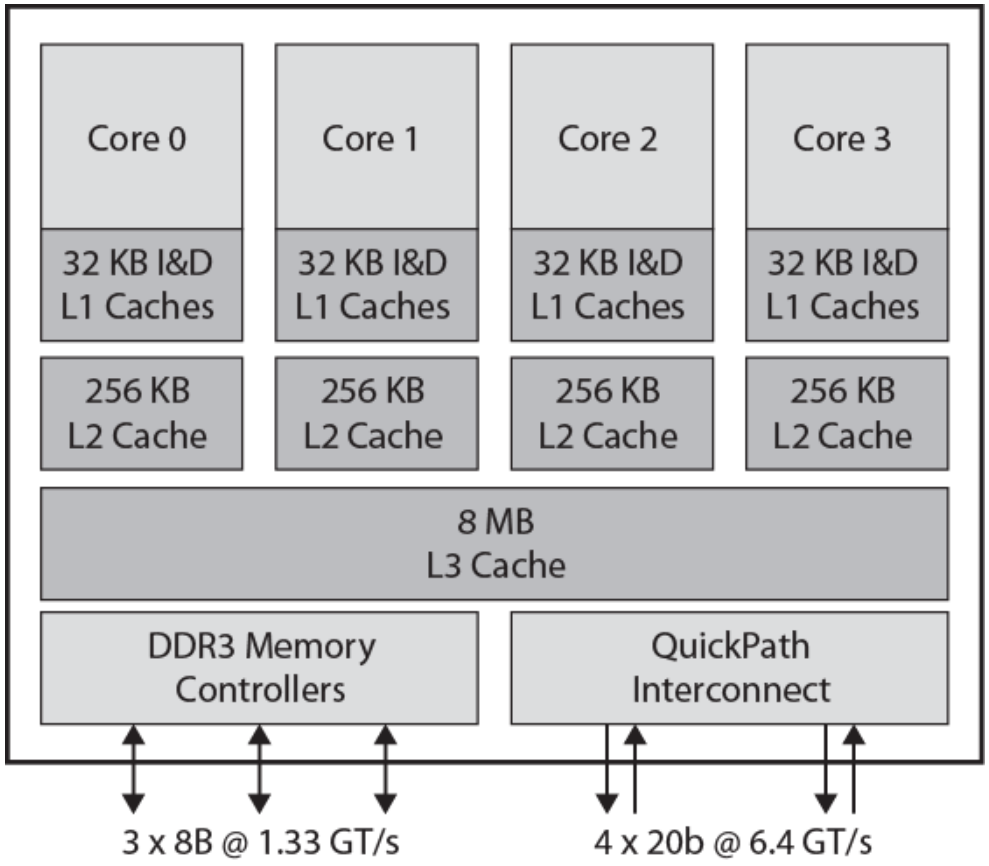
(c) Shared L2 cache

Intel Core i7

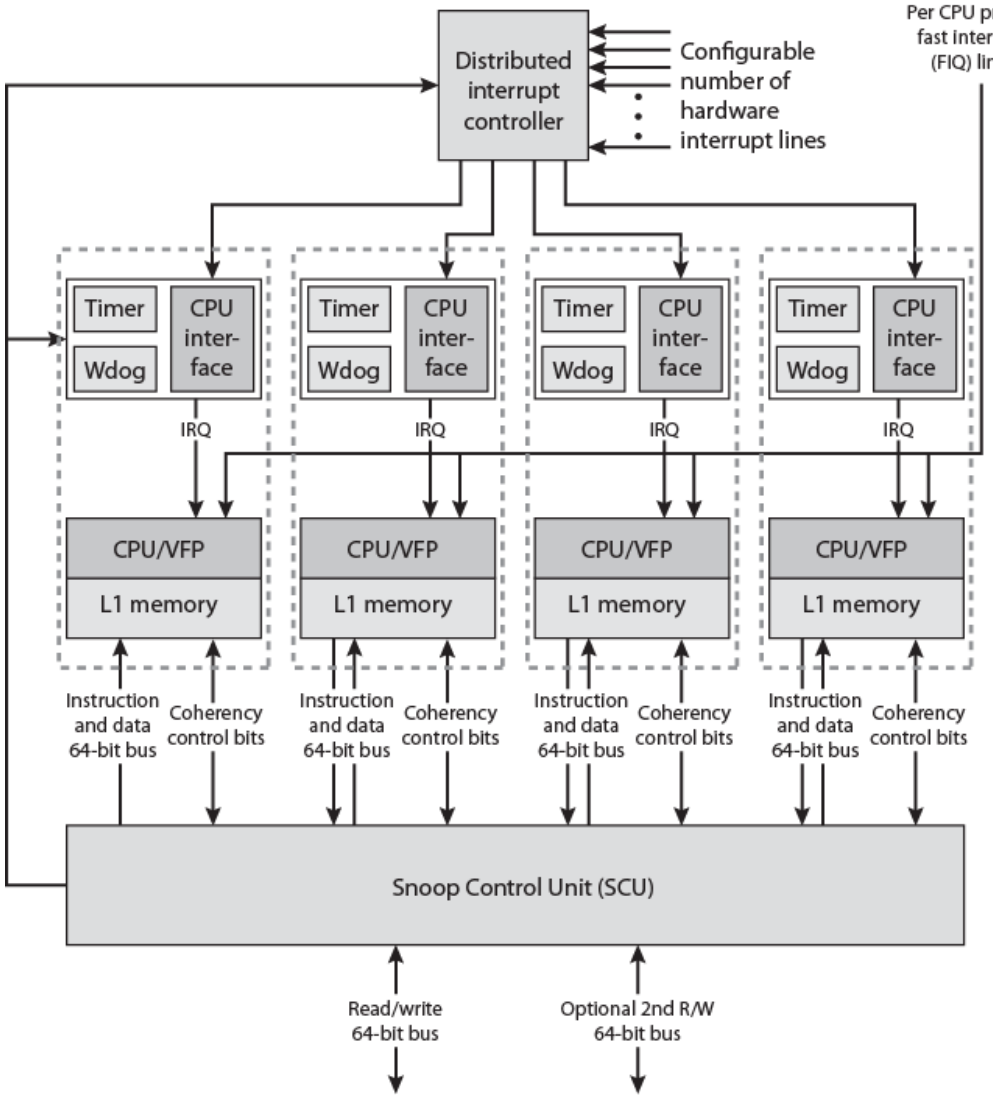


(d) Shared L3 cache

Intel Core i7

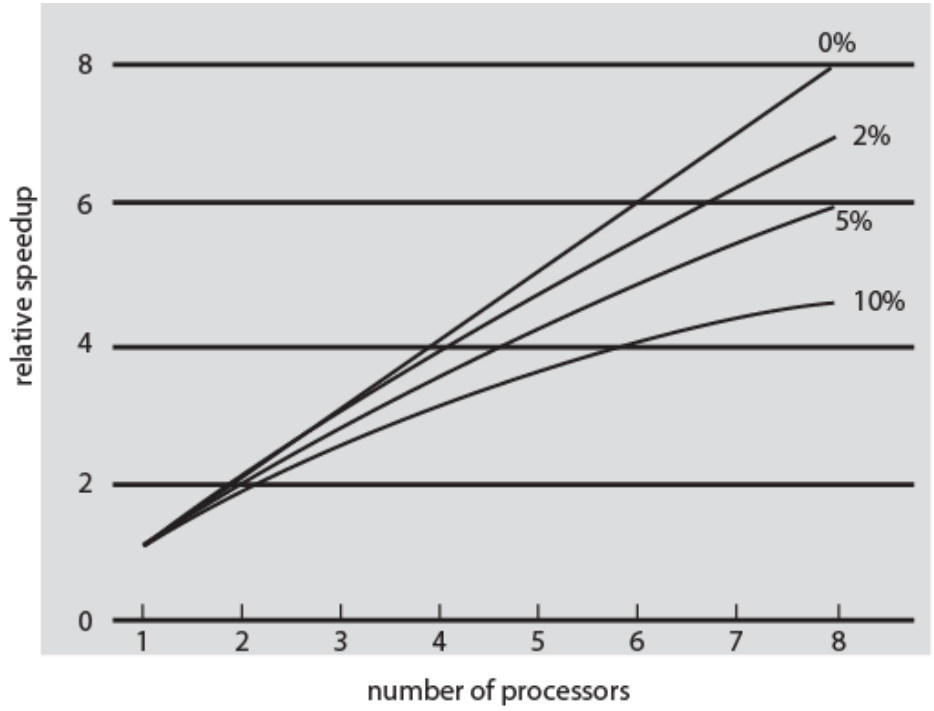


ARM11 MPCore



How many core is enough?

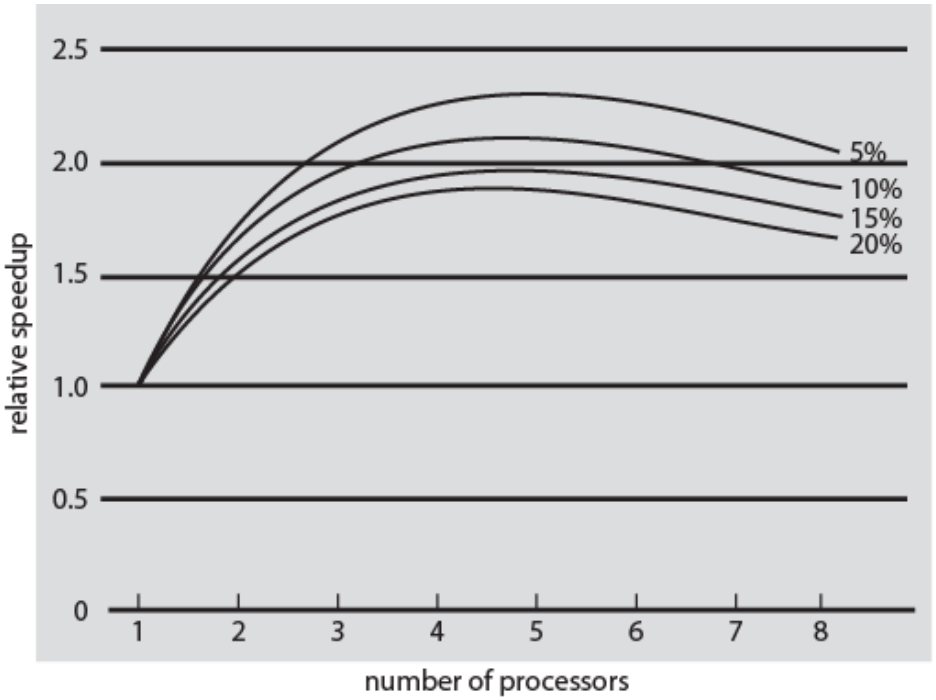
- The more core the higher performance?
  - Not really, it depends on the sequential portion of code
- Amdahl's law



(a) Speedup with 0%, 2%, 5%, and 10% sequential portions

How many core is enough?

- Overhead

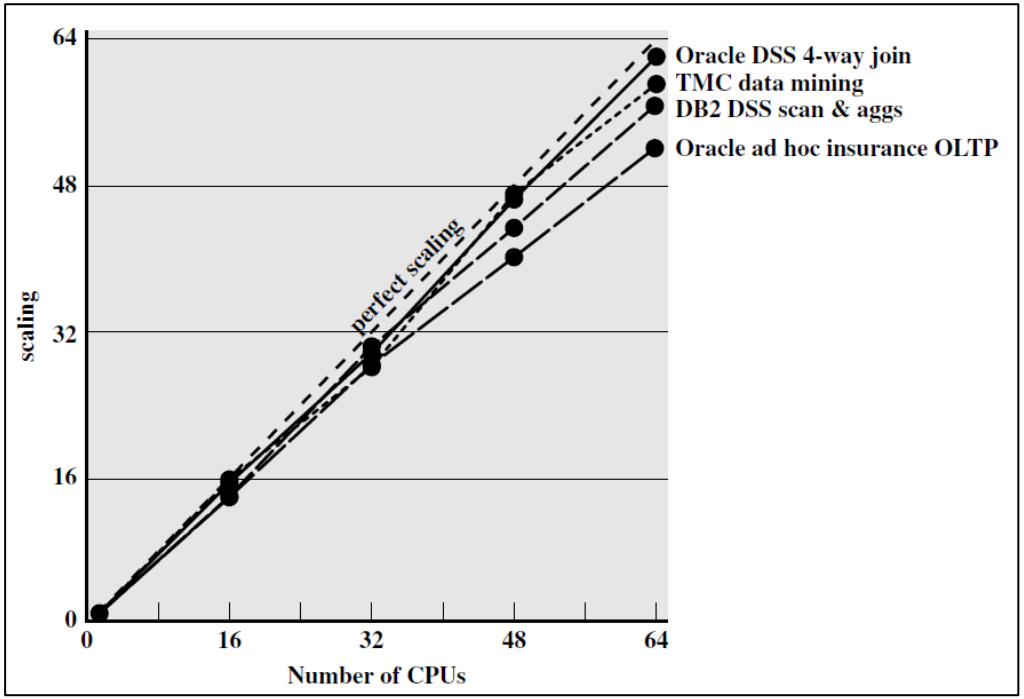


(b) Speedup with overheads

- How many core is best suited for end-user PC?

Khi nào cần nhiều core?

- Database server. Ex: SELECT * FROM …
- Multithreaded native applications
- Multiprocess applications
- Java applications



Scaling of Database Workloads on Multiple-Processor Hardware

The end!