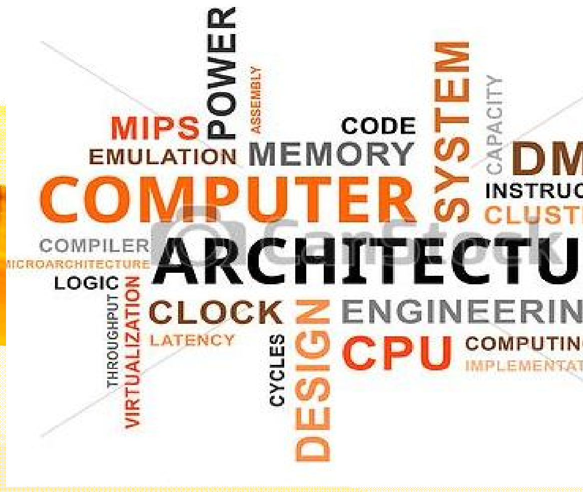
# UNIT V
# I/O ORGANIZATION AND PARALLELISM

Accessing I/O devices – Interrupts – Direct Memory Access – Buses–Interface circuits – Standard I/O Interfaces (PCI, SCSI, USB) –Instruction Level Parallelism : Concepts and Challenges – **Introduction to multicore processor** – Graphics Processing Unit



# Recap the previous Class

# Introduction

- **Multi-Core Processor:**
  - A processing system composed of two or more independent cores or CPUs.
  - The cores are typically integrated onto a single integrated circuit die, or they may be integrated on multiple dies in a single-chip package.
- **Cores share memory:**
  - In modern multi-core systems, typically the L1 and L2 cache are private to each core, while the L3 cache is shared among the cores.
- In symmetric multi-core systems, all the cores are identical.
  - Example: multi-core processors used in computer systems.
- In asymmetric multi-core systems, the cores may have different functionalities.

# Why Multi-core?

- It is difficult to sustain Moore's law and at the same time meet performance demands of various applications.
  - Difficult to increase clock frequency, mainly due to power consumption issues.
- Possible solution:
  - Replicate hardware and run them at a lower clock rate to reduce power consumption.
  - 1 core running at 3 GHz has the same performance as 2 cores running at 1.5 GHz, with lower power consumption.

# Taxonomy of Parallel Architectures

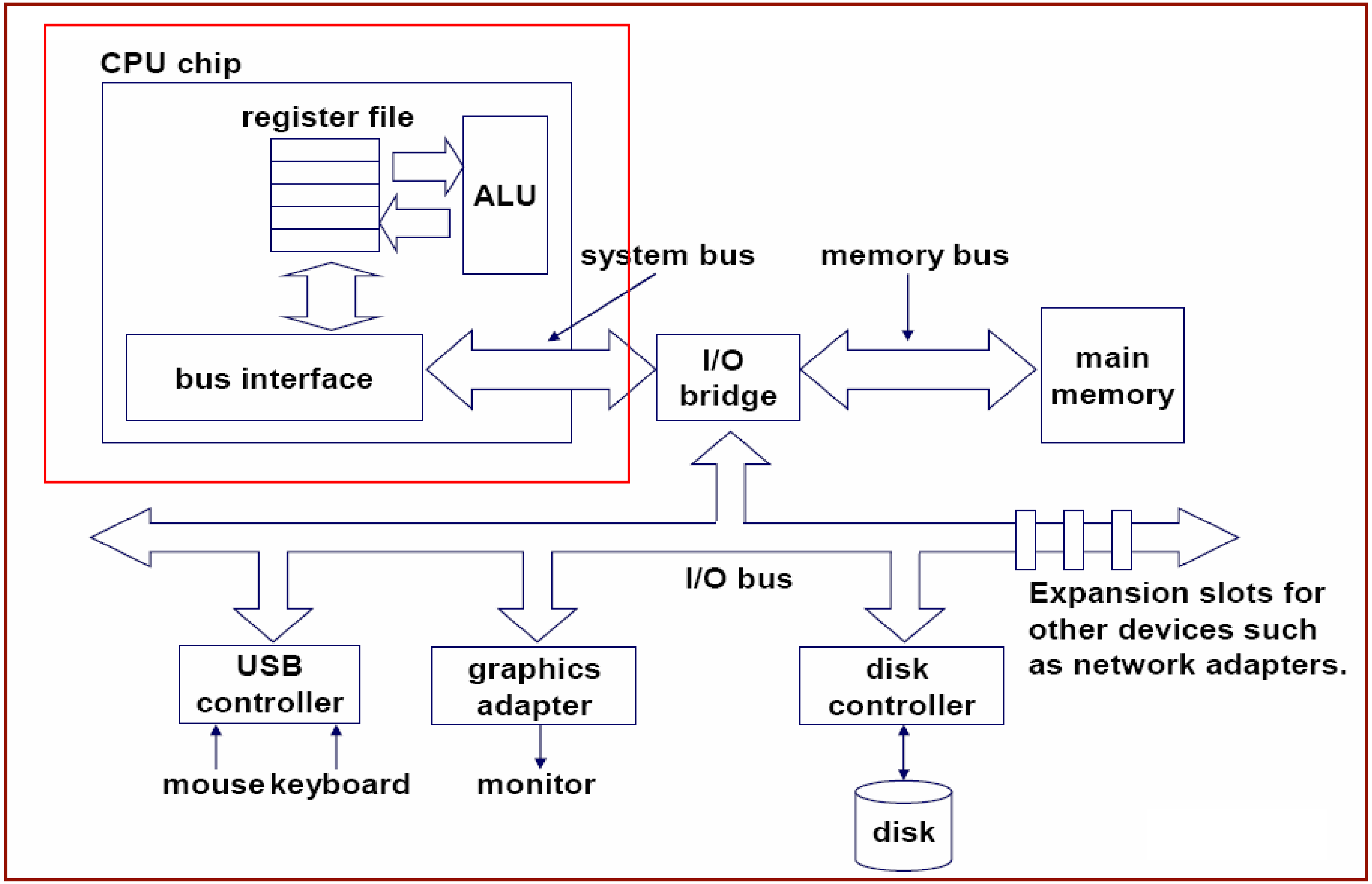
- Single instruction-stream single data-stream (SISD)
  - Traditional uniprocessor systems.
- Multiple instruction-stream single data-stream (MISD)
  - No commercial implementation exists.
  - Pipelining can be argued as a type of MISD processing.
- Single instruction-stream multiple data-stream (SIMD)
  - Array and vector processors.
- Multiple instruction-stream multiple data-stream (MIMD)
  - Multiprocessor systems (various architectures exist).

# Single-core Computer

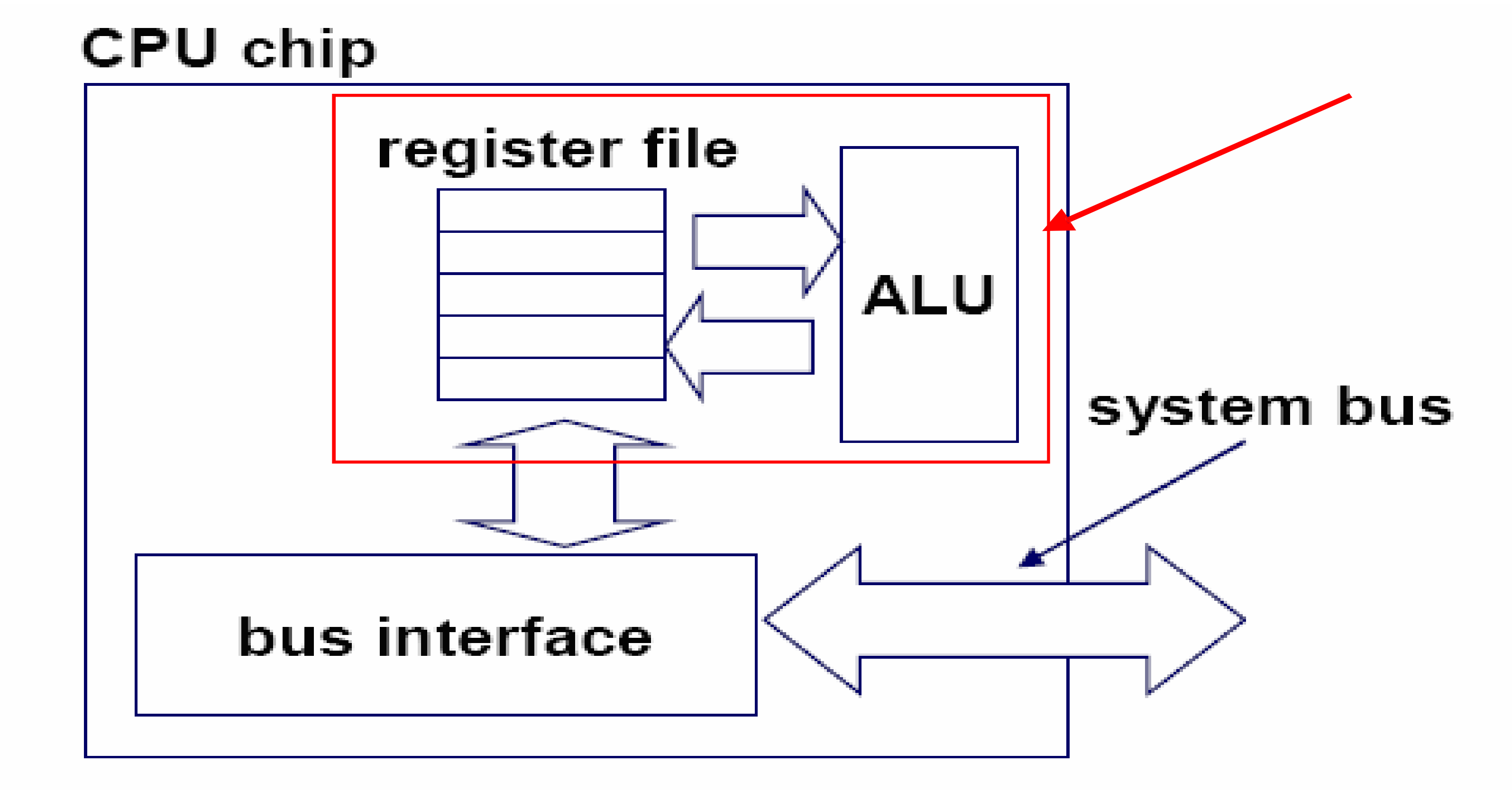
- Falls under SISD category.
- Typically two buses:

  a) A high-speed CPU-memory bus, that also connects to I/O bridge.

  b) A lower-speed I/O bus, connecting various peripherals.

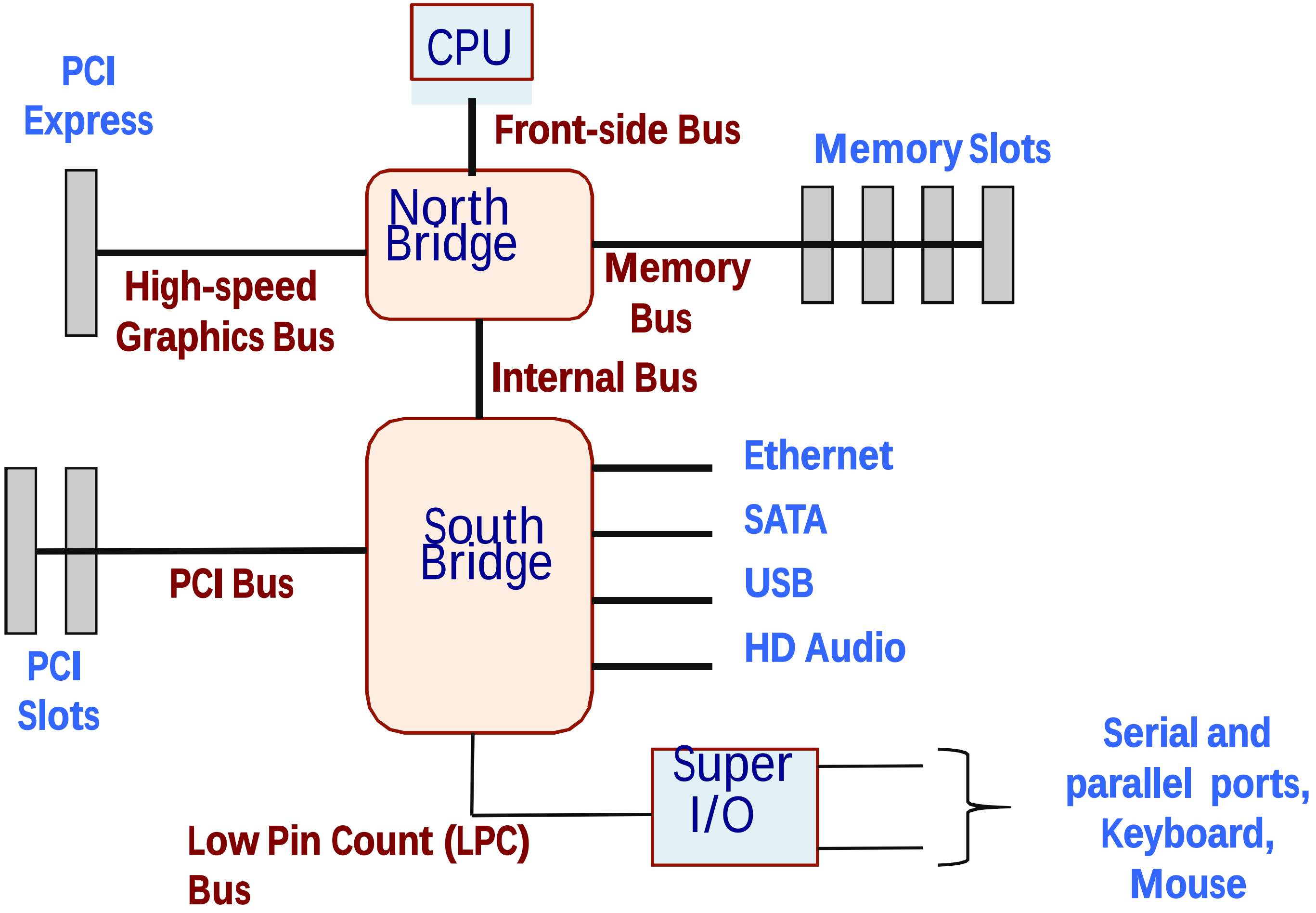


# Single-core Processor



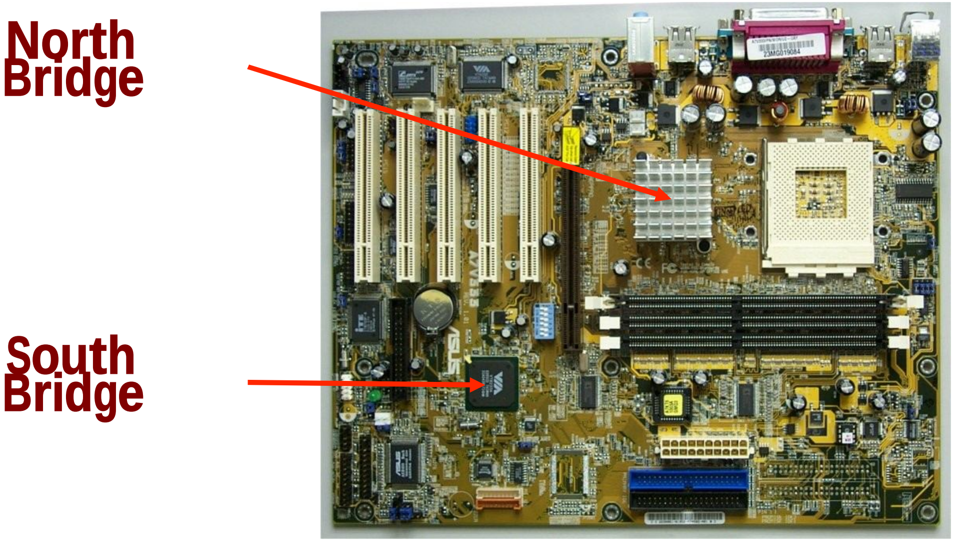
Typical mother board architecture:

- Chipset consisting of north bridge and south bridge



## Locating North Bridge and South Bridge Chipset on Motherboard

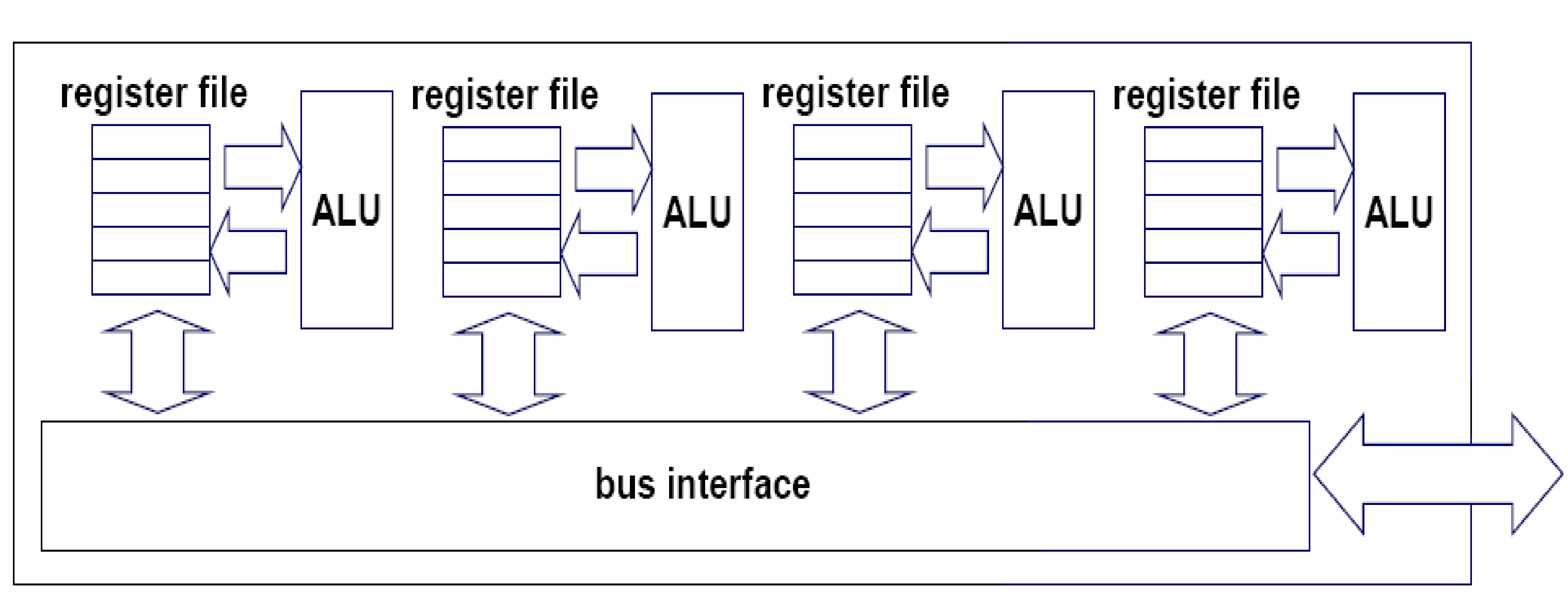
- Bus speeds and other capabilities depend upon the chipset.



North Bridge

South Bridge

# Multi-core Architecture



# Traditional Multiprocessor Architectures

- Can be broadly classified into two types:

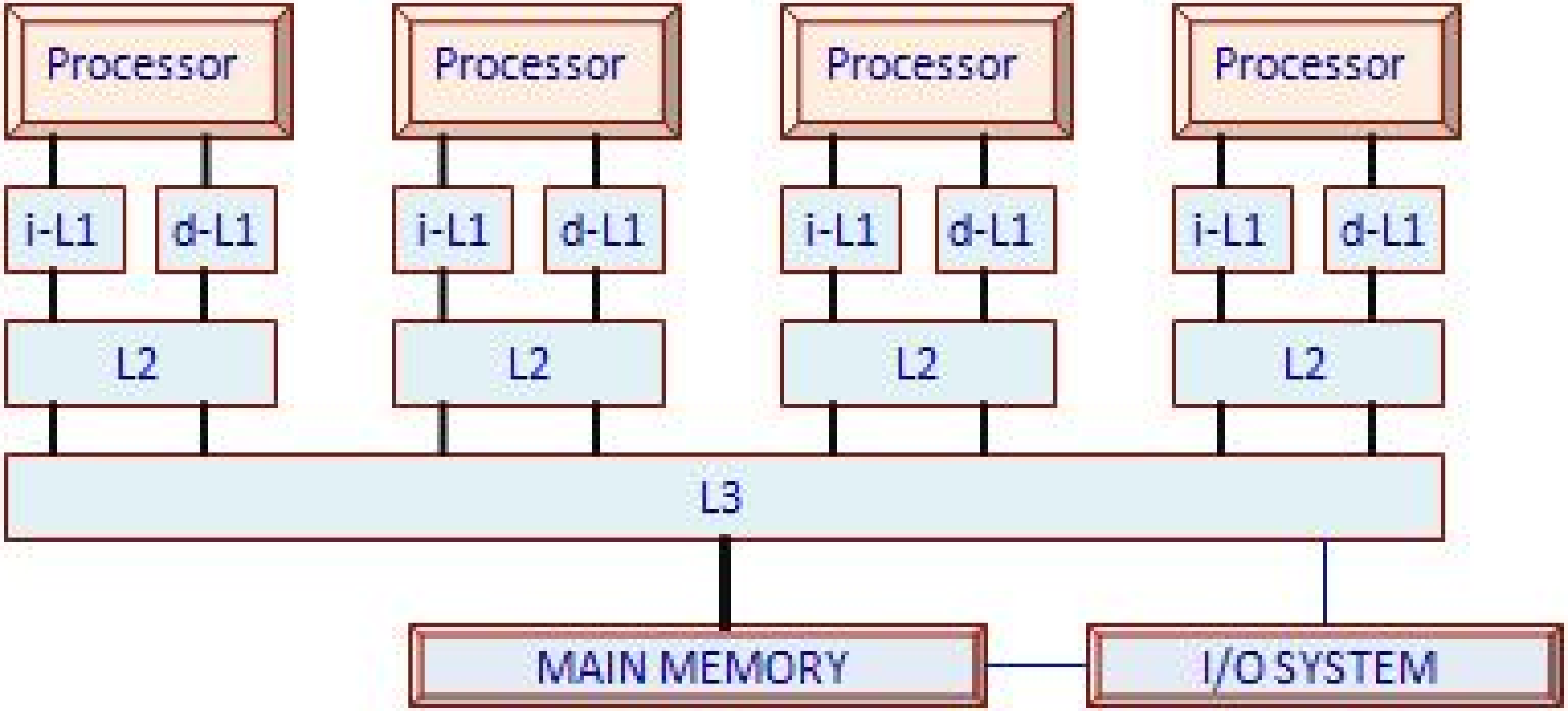
**a)Tightly coupled multiprocessors**

- The processors access common shared memory.
- Inter-processor communication takes place through shared memory.
- Multi-core architectures fall under this category.

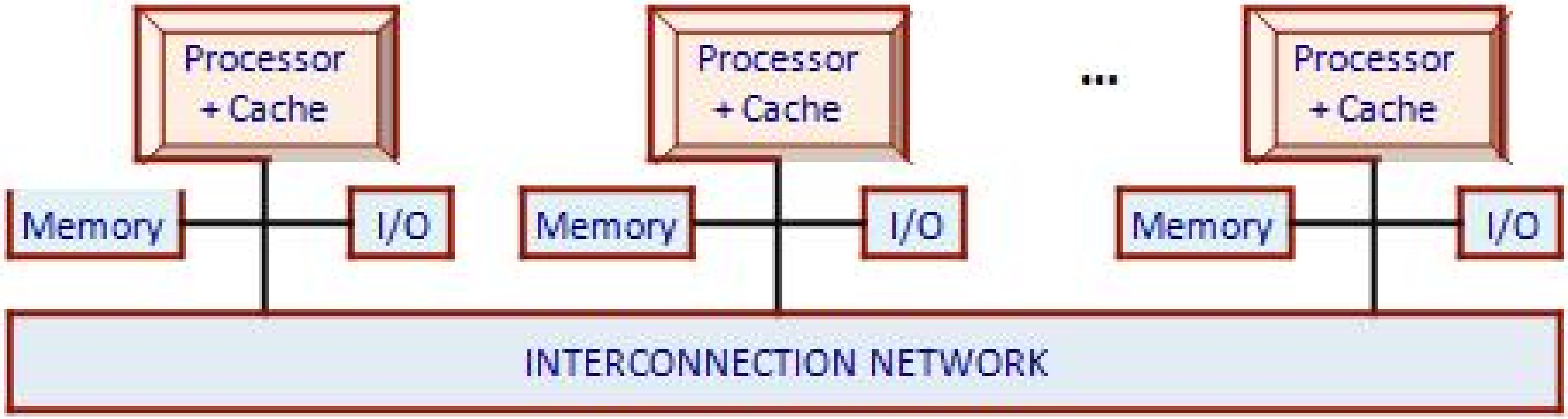
**b)Loosely coupled multiprocessors**

- Memory is distributed among the processors.
- Processors typically communicate through a high-speed interconnection network.

# (a) Tightly Coupled Multiprocessors



## (b) Loosely Coupled Multiprocessors



- **Some features:**
  - –Cost-effective way to scale memory bandwidth.
  - –Communicating data between processors is complex and has higher latency.
  - –Memory access time depends on the location of data.
    - Called *Non Uniform Memory Access – NUMA*.

# Cache Coherency Problem in Multiprocessors

- Maintaining coherence between data loaded in processor caches is an issue in multiprocessor systems.
  - Same memory block is loaded into two processor caches.
  - One of the processors updates the data in its local cache.
  - Data in the other processor cache and also memory becomes inconsistent.
- Broadly two classes of techniques are used to solve this problem:

a)Snoopy protocols

b)Directory-based protocols

- Some features:
  - – Difficult to extend it to large number of processors.
  - – Memory bandwidth requirements increase with the number of processors.
  - – Memory access time for all processors is uniform.
    - Called *Uniform Memory Access – UMA*.

# TEXT BOOK

Carl Hamacher, Zvonko Vranesic and Safwat Zaky, “Computer Organization”, McGraw-Hill, 6th Edition 2012.

# REFERENCES

1. David A. Patterson and John L. Hennessey, “Computer organization and design”, MorganKauffman ,Elsevier, 5th edition, 2014.
2. William Stallings, “Computer Organization and Architecture designing for Performance”, Pearson Education 8th Edition, 2010
3. John P.Hayes, “Computer Architecture and Organization”, McGraw Hill, 3rd Edition, 2002
4. M. Morris R. Mano “Computer System Architecture” 3rd Edition 2007
5. David A. Patterson “Computer Architecture: A Quantitative Approach”, Morgan Kaufmann; 5th edition 2011

# THANK YOU