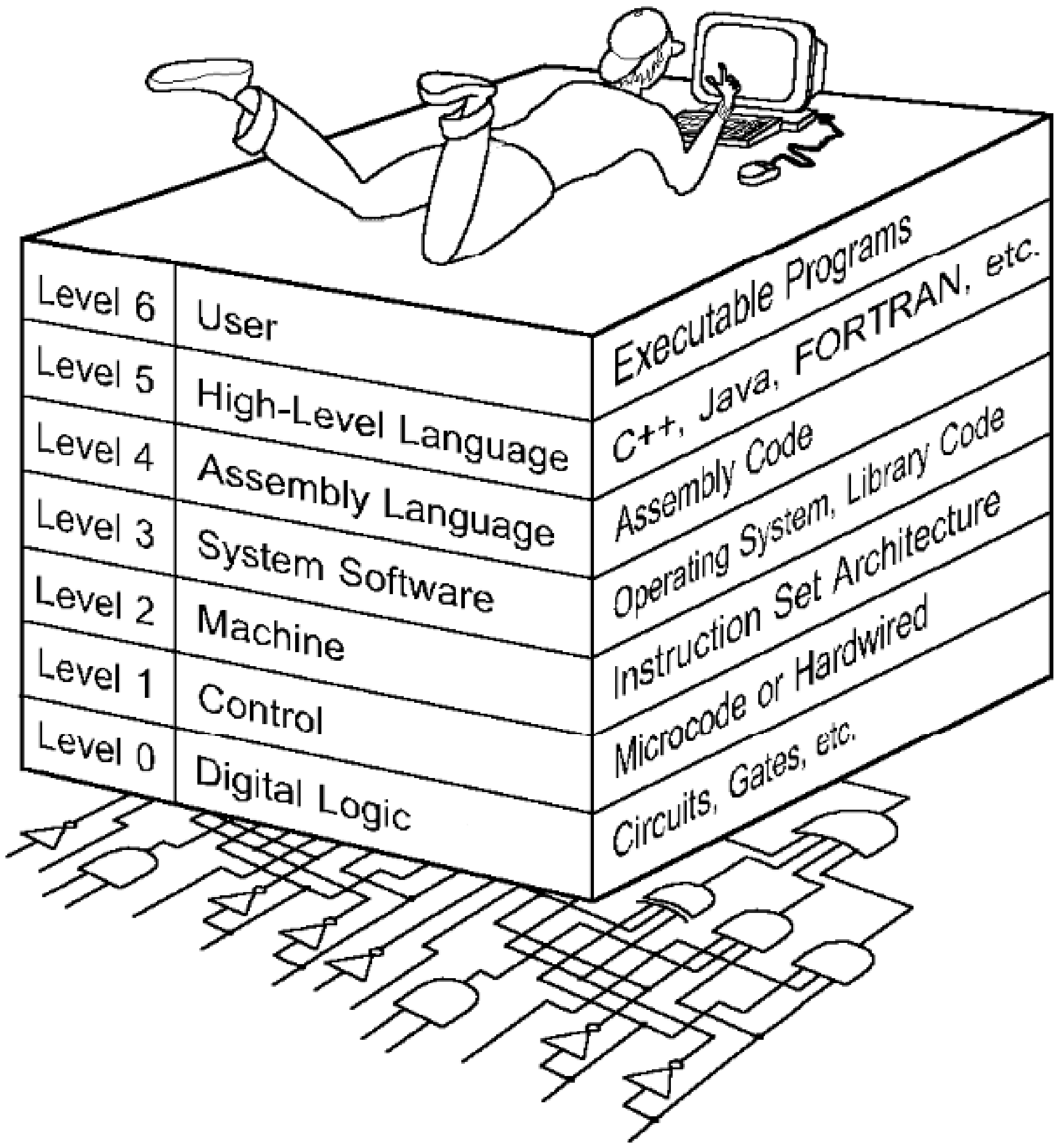
# Computer Organization
# +
# DIGITAL DESIGN

**SUKHENDU DAS**

www.cse.iitm.ac.in/~sdas

sdas@iitm.ac.in

# Computer Level Hierarchy



## Program Execution

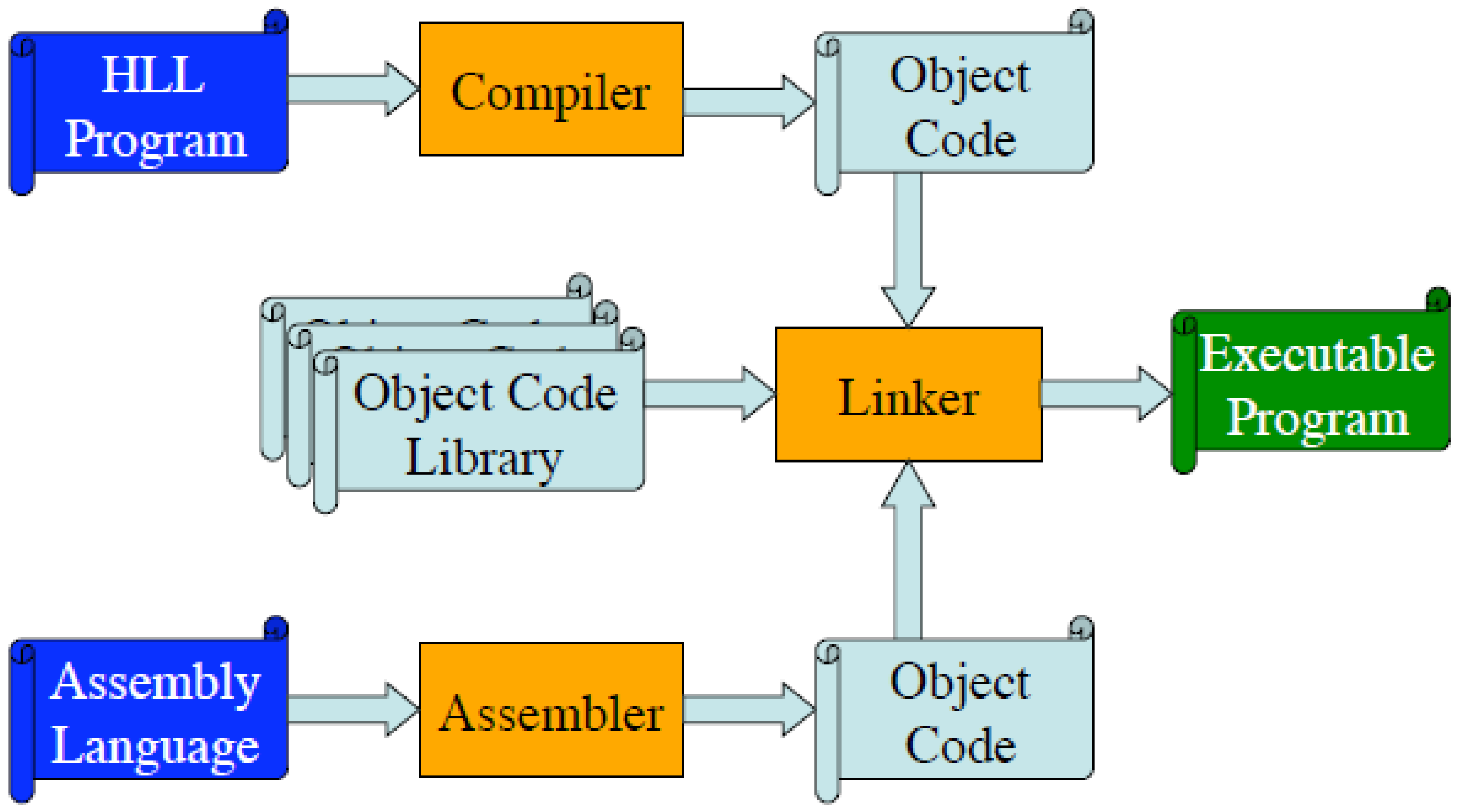
**Translation:** The entire high level program is translated into an equivalent machine language program. Then the machine language program is executed.

**Interpretation**: Another program reads the high level program instructions one-by-one and executes a equivalent series of machine language instructions.

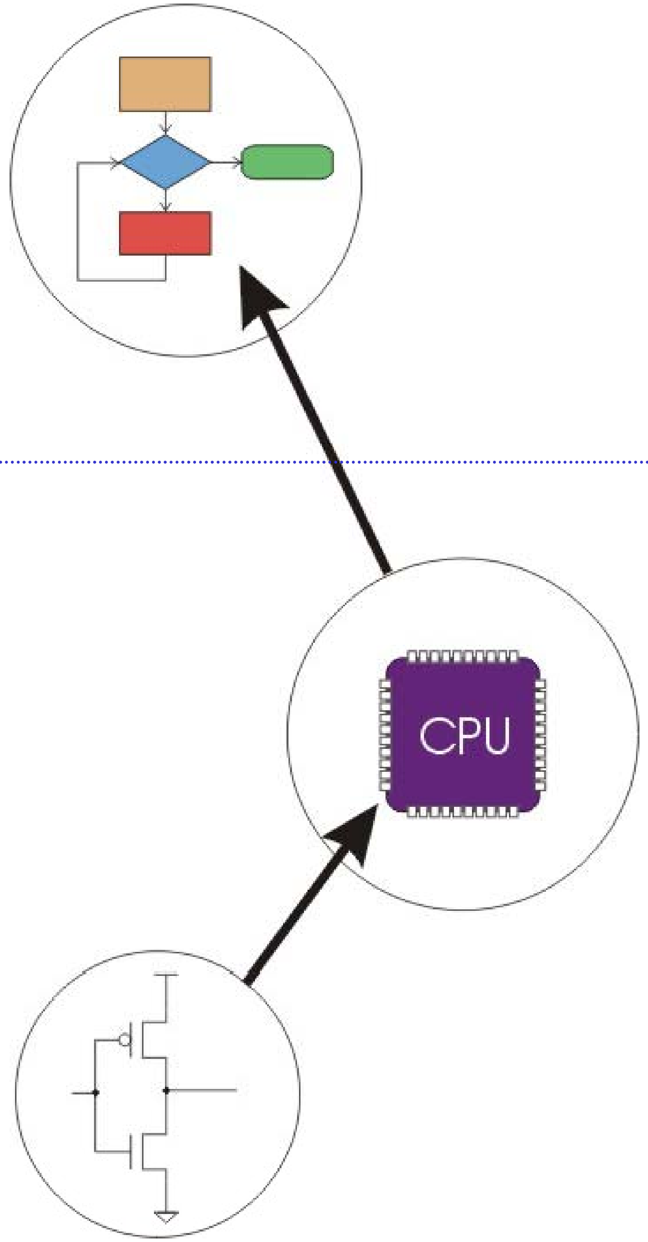
Program translation uses a collection of tools to perform the translation:

- **Compiler:** Translates high level language programs into a lower level language often called object code.
- **Assembler:** Translates assembly language instructions into object code.
- **Linker:** Combines collections of object code into a single executable machine language program.

# Program Translation



# Computer System: Layers of Abstraction



**Application Program**

**Algorithms**

**Language**

*Software*

*Hardware*

**Instruction Set Architecture (and I/O Interfaces)**

**Microarchitecture**

**Circuits**

**Devices**

## From Theory to Practice

**In theory, computer can *compute* anything that's possible to compute**

- **given enough *memory* and *time***

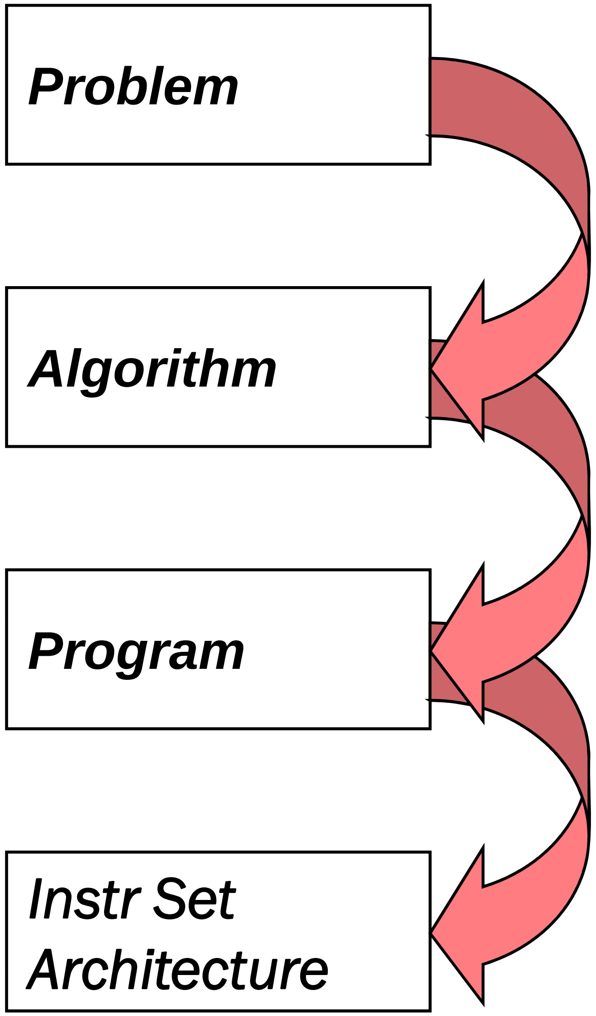
**In practice, *solving problems* involves computing under constraints.**

- **time**
  - ➢ **weather forecast, next frame of animation, ...**
- **cost**
  - ➢ **cell phone, automotive engine controller, ...**
- **power**
  - ➢ **cell phone, handheld video game, ...**

## Transformations Between Layers

**How do we solve a problem using a computer?**

**A systematic sequence of transformations between layers of abstraction.**

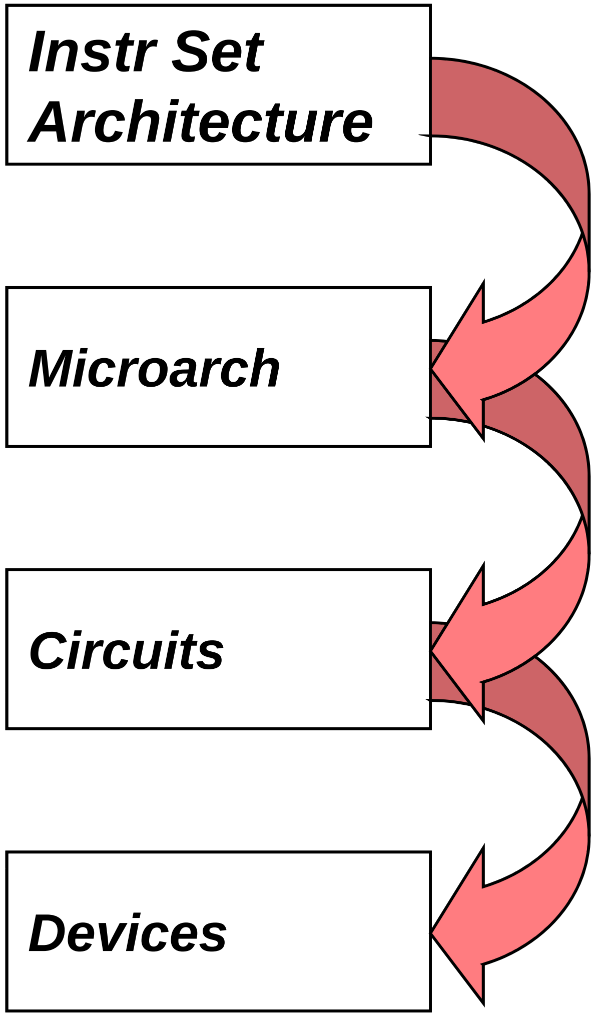


***Software Design:***
choose algorithms and data structures

***Programming:***
use language to express design

***Compiling/Interpreting:***
convert language to
machine instructions

## Deeper and Deeper…



***Processor Design:***
choose structures to implement ISA

***Logic/Circuit Design:***
gates and low-level circuits to implement components

***Process Engineering & Fabrication:***
develop and manufacture lowest-level components

## Descriptions of Each Level

### Problem Statement

- stated using "natural language"
- may be ambiguous, imprecise

### Algorithm

- step-by-step procedure, guaranteed to finish
- definiteness, effective computability, finiteness

### Program

- express the algorithm using a computer language
- high-level language, low-level language

### Instruction Set Architecture (ISA)

- specifies the set of instructions the computer can perform
- data types, addressing mode

## Descriptions of Each Level (cont.)

### Microarchitecture

- detailed organization of a processor implementation
- different implementations of a single ISA

### Logic Circuits

- combine basic operations to realize microarchitecture
- many different ways to implement a single function (e.g., addition)

### Devices

- properties of materials, manufacturability

## Structure and Function of a COMPUTER SYSTEM:

A computer is a complex system; For analysis, understanding and design - Identify the ***hierarchical nature*** of most complex system.

A hierarchical system is a set of interrelated subsystems, each in turn, hierarchical in structure; until at the lowest level we have elementary subsystems.

The hierarchical nature of complex systems is essential to both their design and their description. The designer need only deal with a particular level of the system at a time.

At each level, the system consists of a set of ***components and their interrelationships***.

The behavior at each level depends only on a simplified, abstracted characterization of the system at the next lower level.

At each level, the designer is concerned with structure and function:

**Structure**: The way in which the components are interrelated.

**Function:** The operation of each individual component as part of the structure.

## The von Neumann Model

- The invention of stored program computers has been ascribed to a mathematician, John von Neumann, who was a contemporary of Mauchley and Eckert.

- Stored-program computers have become known as **von Neumann Architecture** systems.

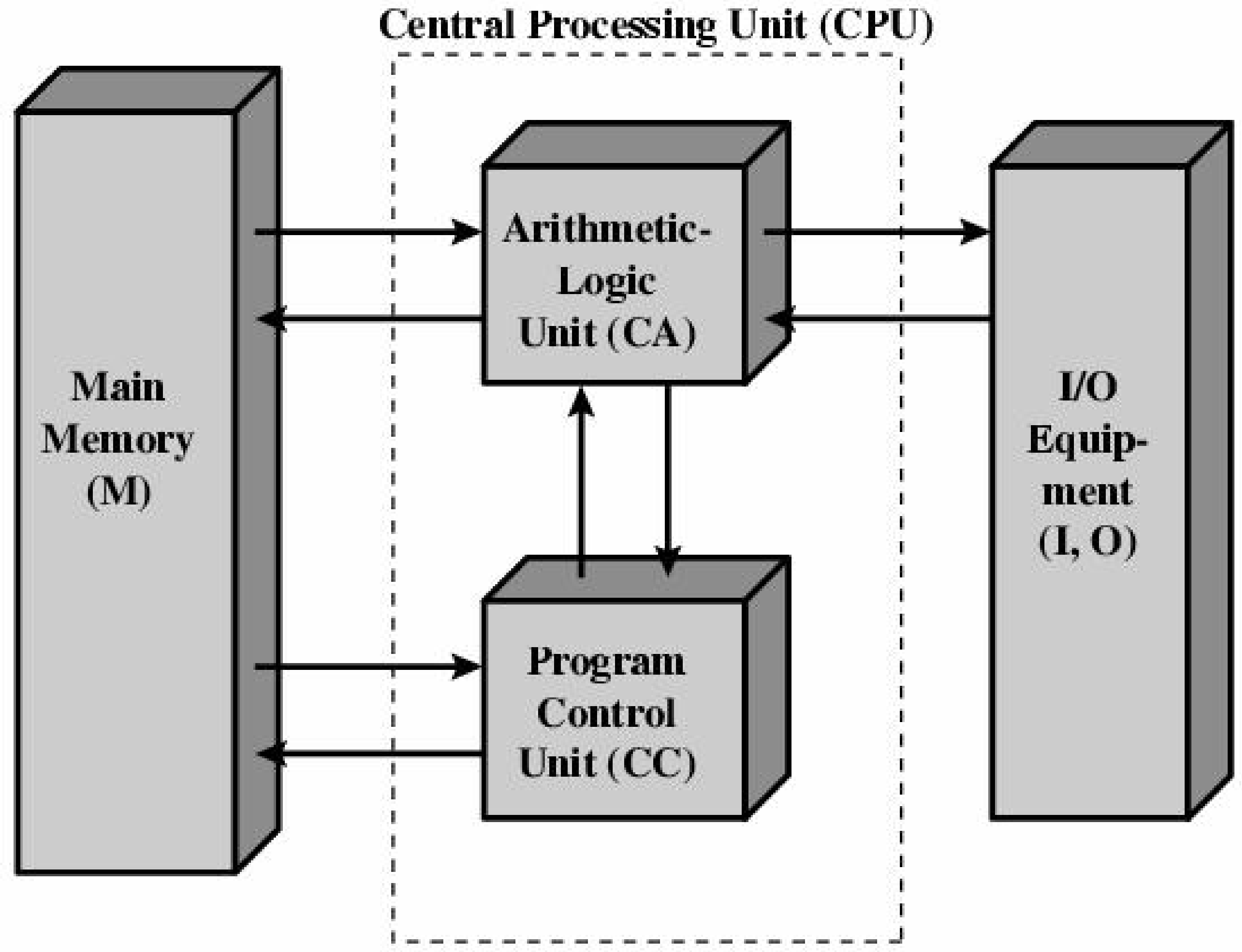
## The von Neumann Model

- **Today's stored-program computers have the following characteristics:**
  - **Three hardware systems:**
    - **A central processing unit (CPU)**
    - **A main memory system**
    - **An I/O system**

**The capacity to carry out sequential instruction processing.**

**A single data path between the CPU and main memory.**

**This single path is known as the *von Neumann bottleneck*.**


Central Processing Unit (CPU)
Main Memory (M)
Arithmetic-Logic Unit (CA)
Program Control Unit (CC)
I/O Equip-ment (I, O)

**IAS (Princeton) computer model by Von Neumann’s group.**

**IAS computer consists of:**

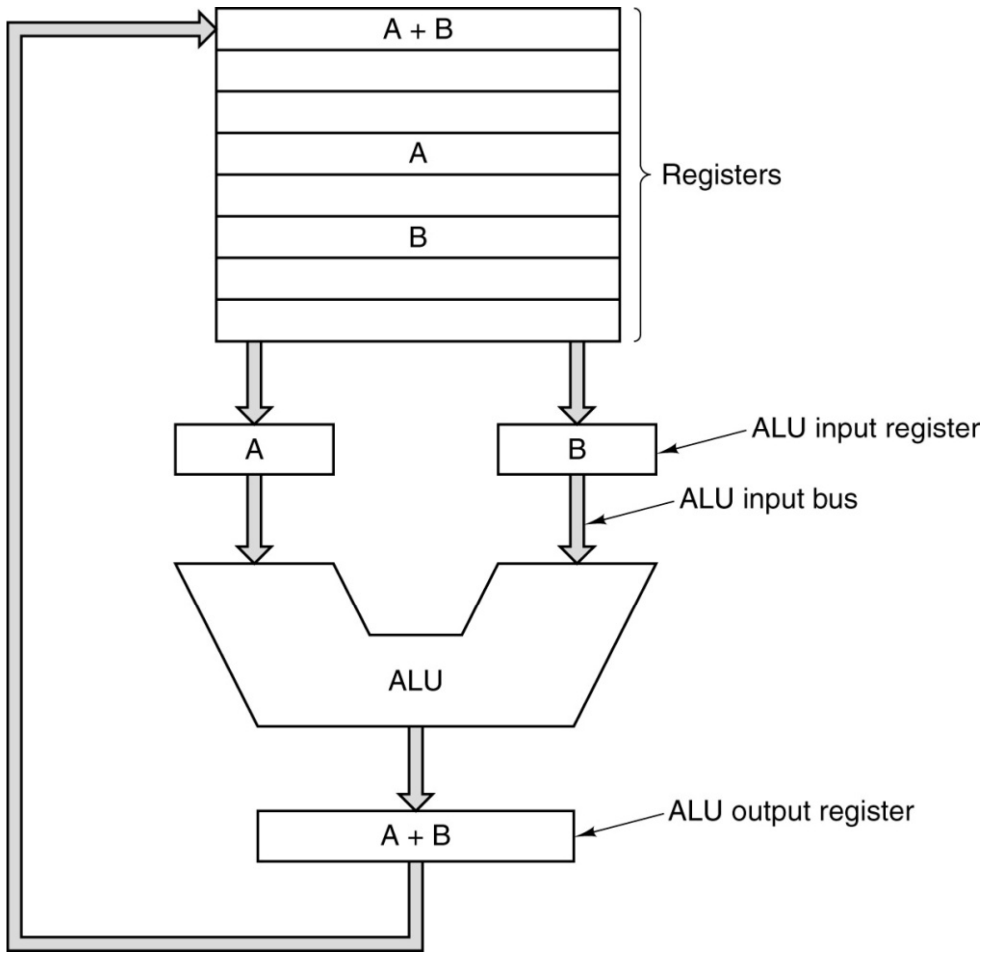
**- A main memory, which stores both data and instructions.**

**- An arithmetic-logical unit (ALU) capable of operating on binary data.**

**- A control unit, which interprets the instructions in memory and causes them to be executed.**

**- Input and output (I/O) equipment operated by the control unit.**

# CPU Organization

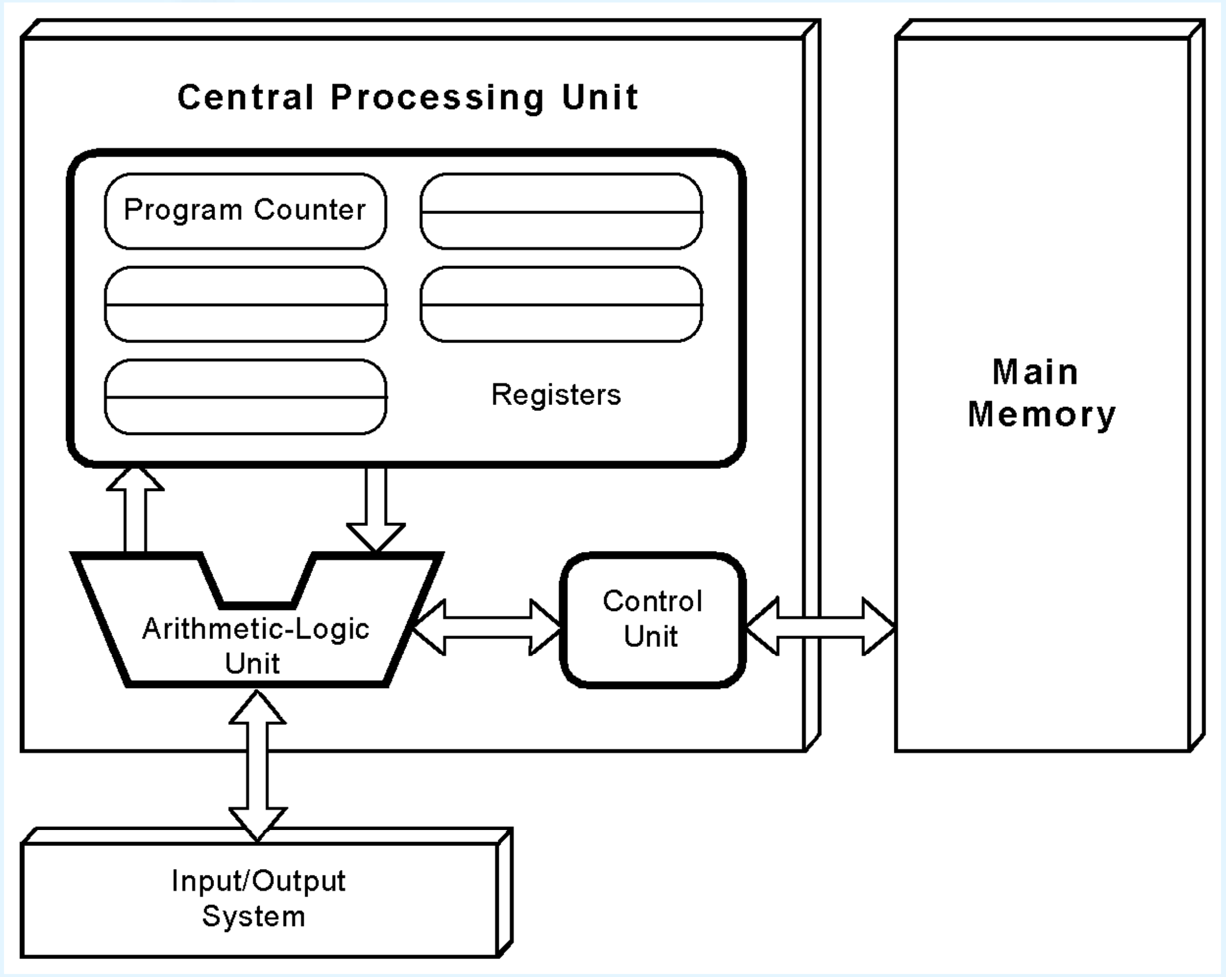


The data path of a typical Von Neumann machine.

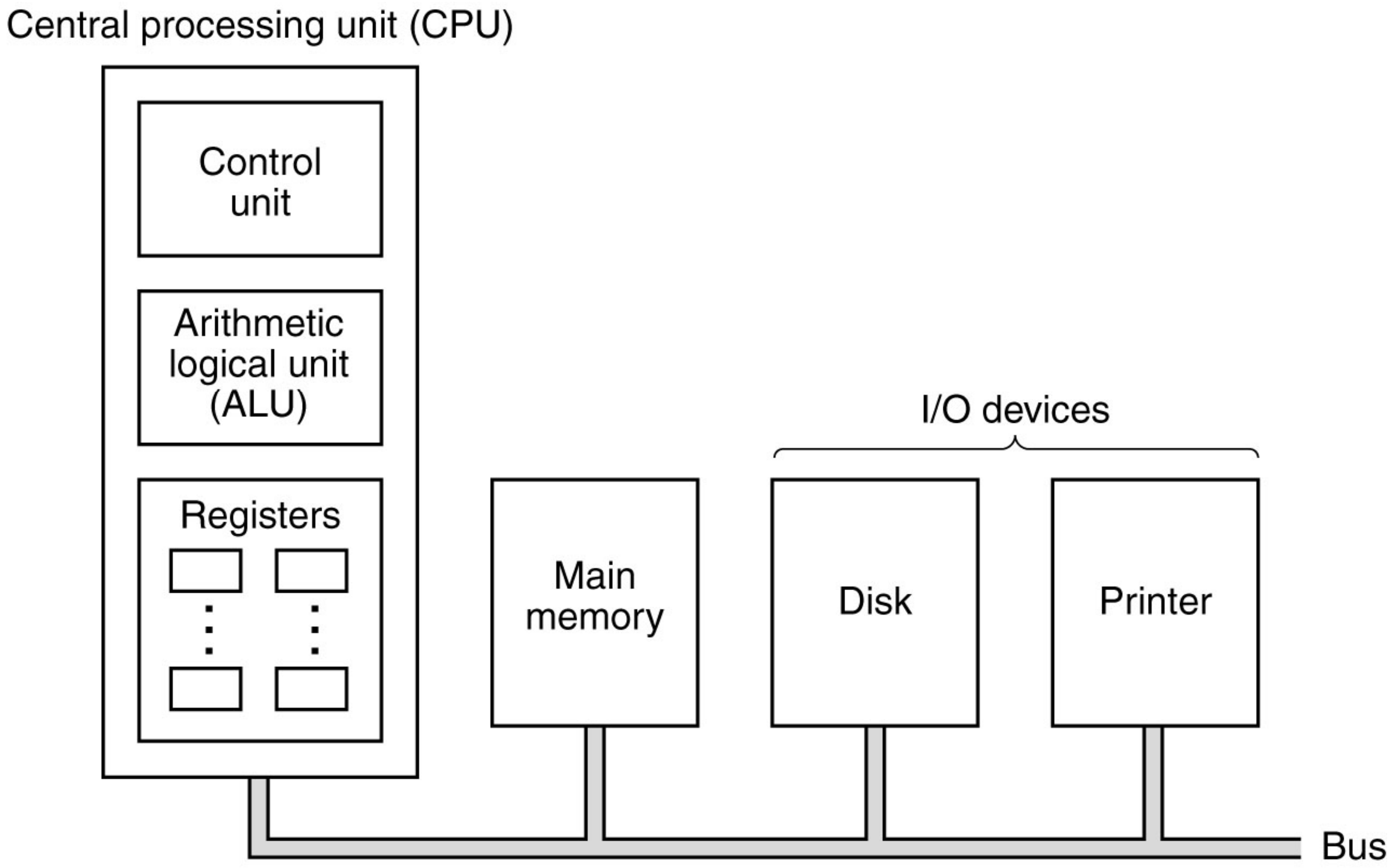
## The von Neumann Model

- This is a general depiction of a von Neumann system:

- These computers employ a fetch-decode-execute cycle to run programs as follows . . .



# Central Processing Unit (CPU) based CO



The organization of a simple computer with one CPU and two I/O devices

## There are four main functions of a computer:

- Data processing
- Data storage
- Data movement
- Control

MAIN STRUCTURAL BLOCKS / PARTS:

Central Processing Unit (CPU): Controls the operation of the computer and performs its data processing functions. Often simply referred to as processor.

Main Memory: Stores data.

I / O: Moves data between the computer and its external environment.

System Interconnection: e.g. BUS for communication among CPU, main memory, and I / O.

The major structural components of a CPU are:

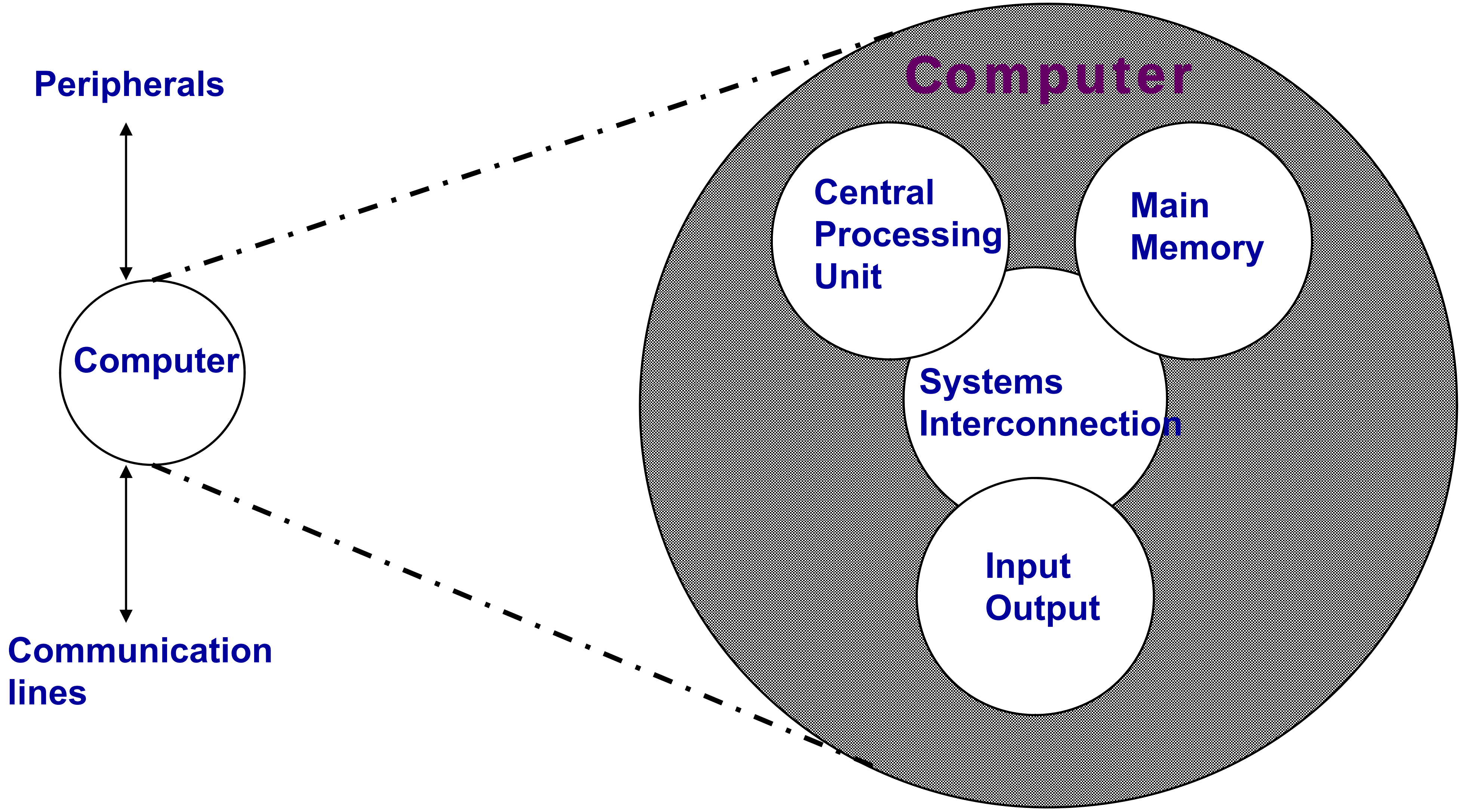
**Control Unit (CU):** Controls the operation of the CPU and hence the computer.

**Arithmetic and Logic Unit (ALU):** Performs computer’s data processing functions.

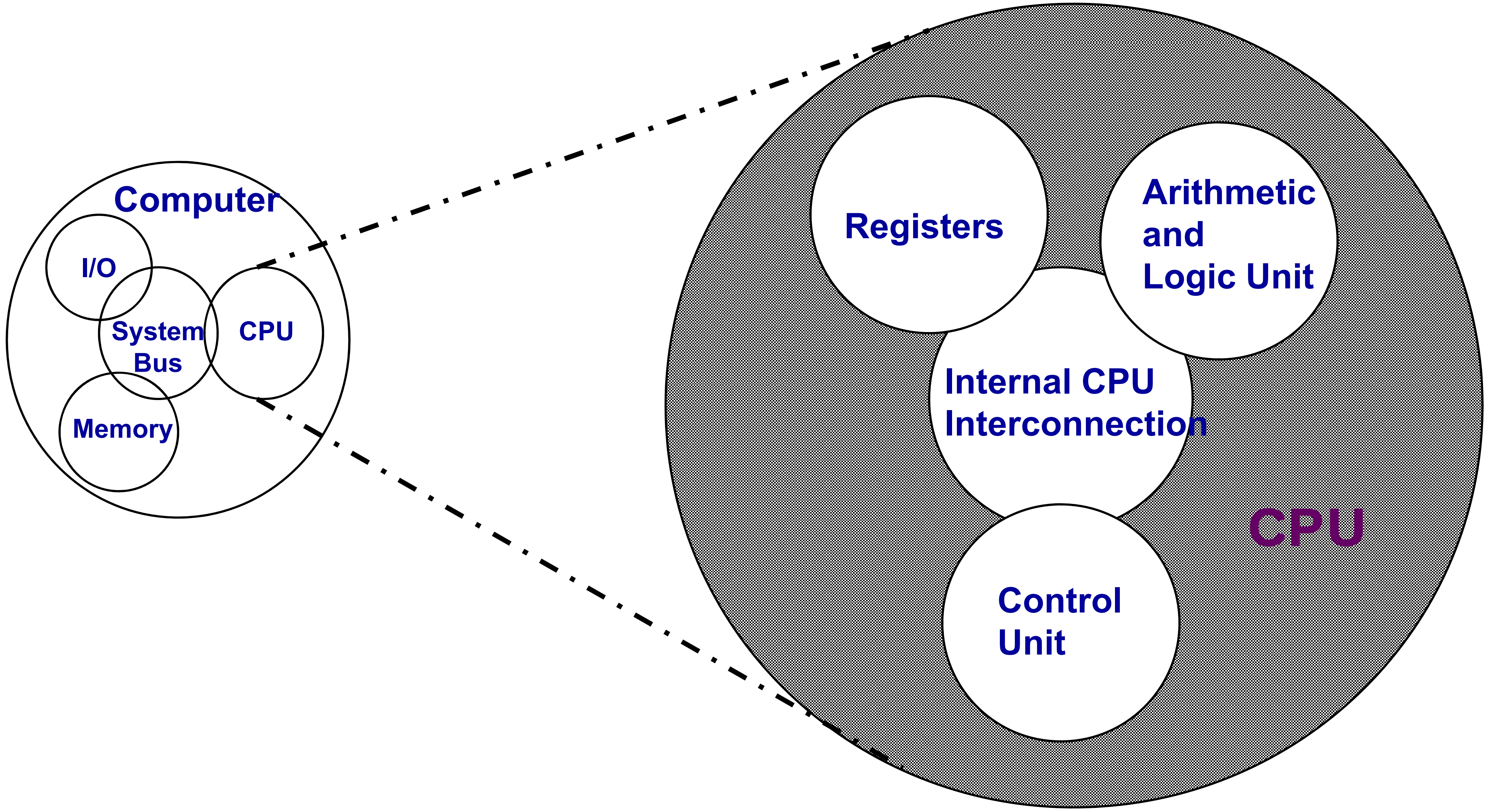
**Register:** Provides storage internal to the CPU.

**CPU Interconnection:** communication among the control unit, ALU, and register.

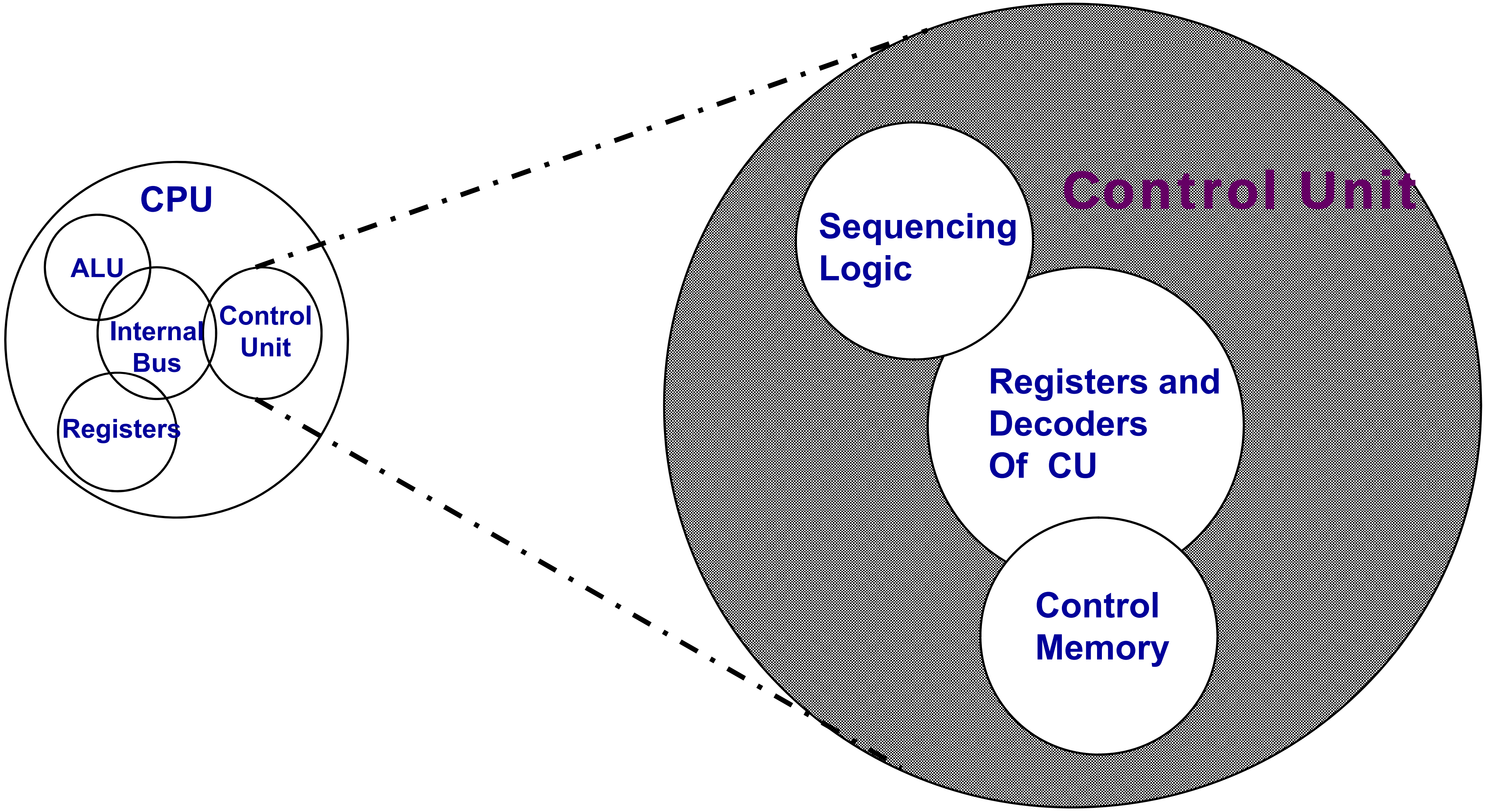
# Structure - Top Level



# Structure - The CPU



# Structure - The Control Unit



## Computer Architecture

Logical aspects of system implementation as seen by the programmer; such as, instruction sets (ISA) and formats, opcode, data types, addressing modes and I/O.

Instruction set architecture (ISA) is different from “microarchitecture”, which consist of various processor design techniques used to implement the instruction set.

Computers with different microarchitectures can share a common instruction set.

For example, the Intel Pentium and the AMD Athlon implement nearly identical versions of the x86 instruction set, but have radically different internal designs.

Computer architecture comes before computer organization.

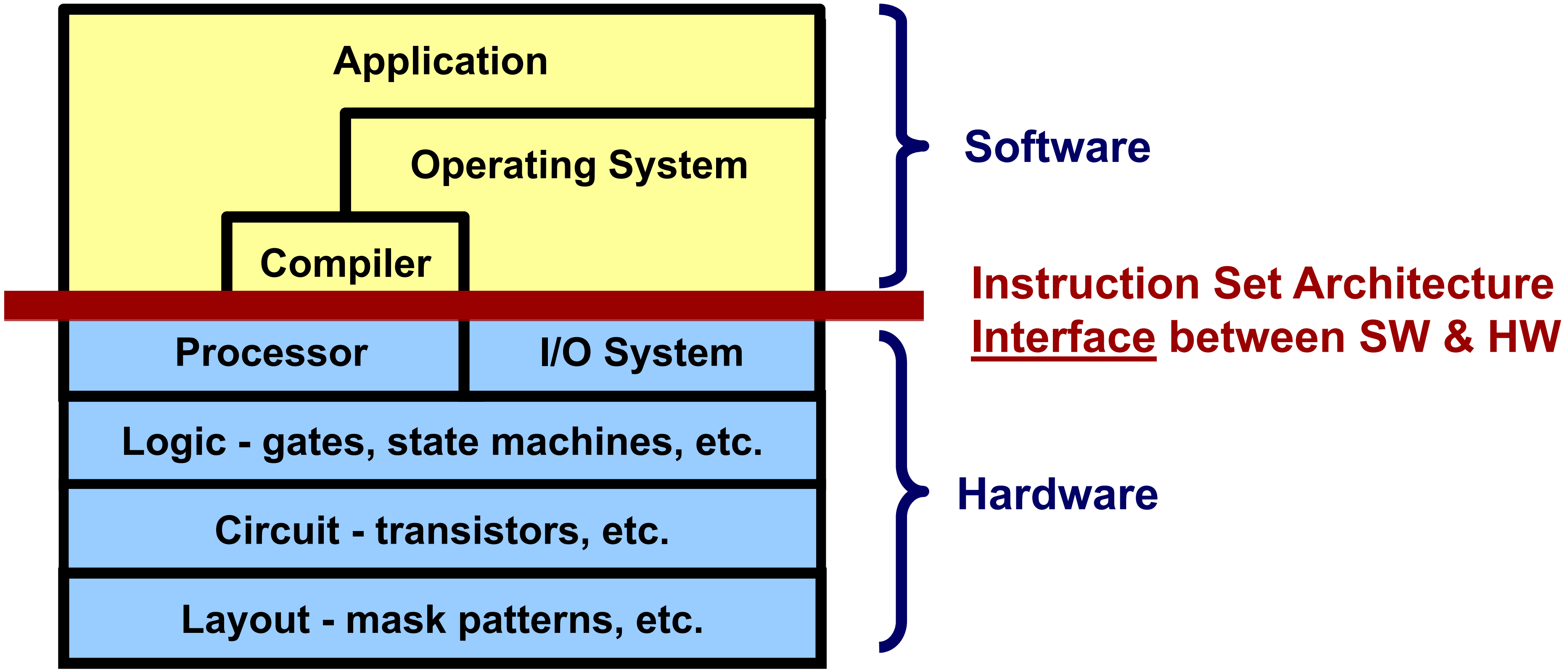
Computer organization (CO) is how operational attributes are linked together and contribute to realise the architectural specifications.

CO encompasses all physical aspects of computer systems

e.g. Circuit design, control signals, memory types.

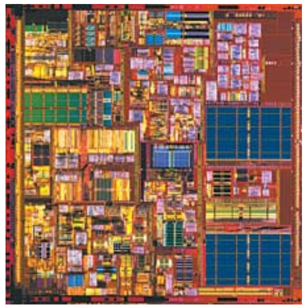
## Instruction Set Architecture (ISA) - The Hardware-Software Interface

- The **most important** abstraction of computer design


Application
Operating System
Compiler
Processor
I/O System
Logic - gates, state machines, etc.
Circuit - transistors, etc.
Layout - mask patterns, etc.
Software
Instruction Set Architecture
Interface between SW & HW
Hardware

## Important Building Blocks

- Microprocessor

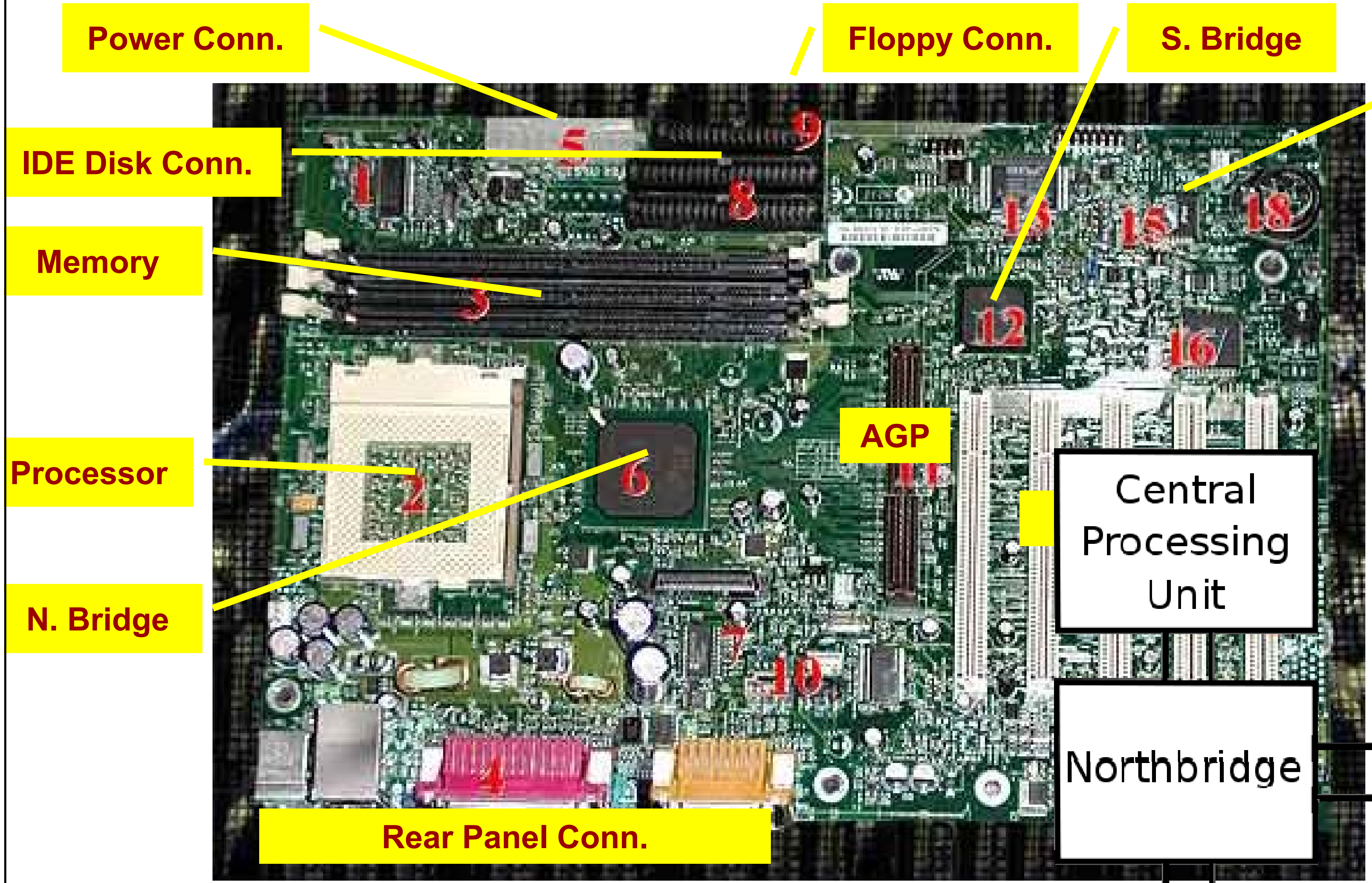


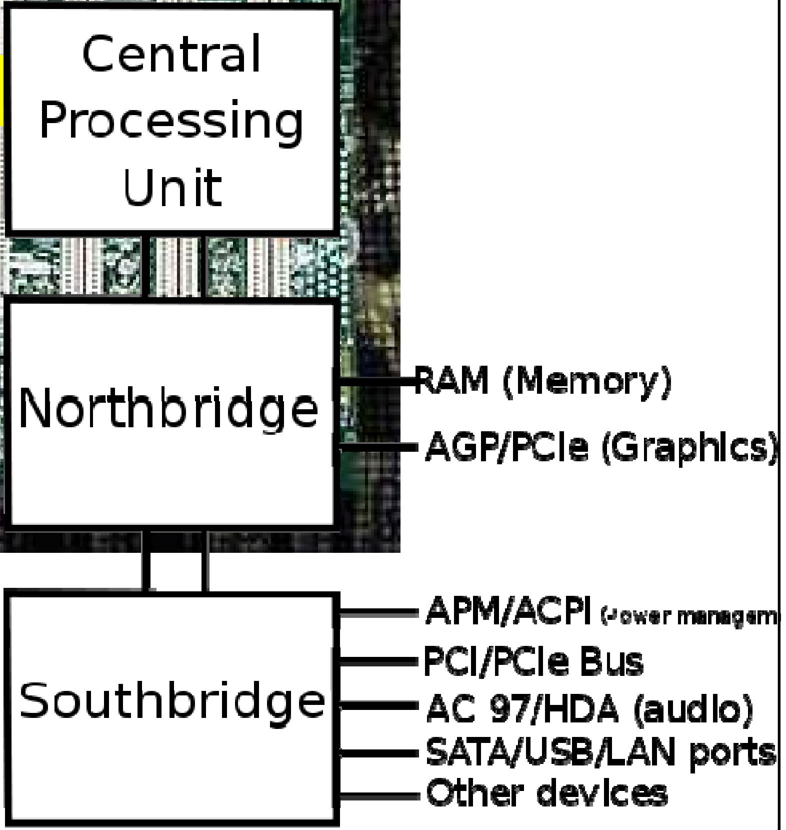
- Memory

- Mass Storage (Disk)

- Network Interface

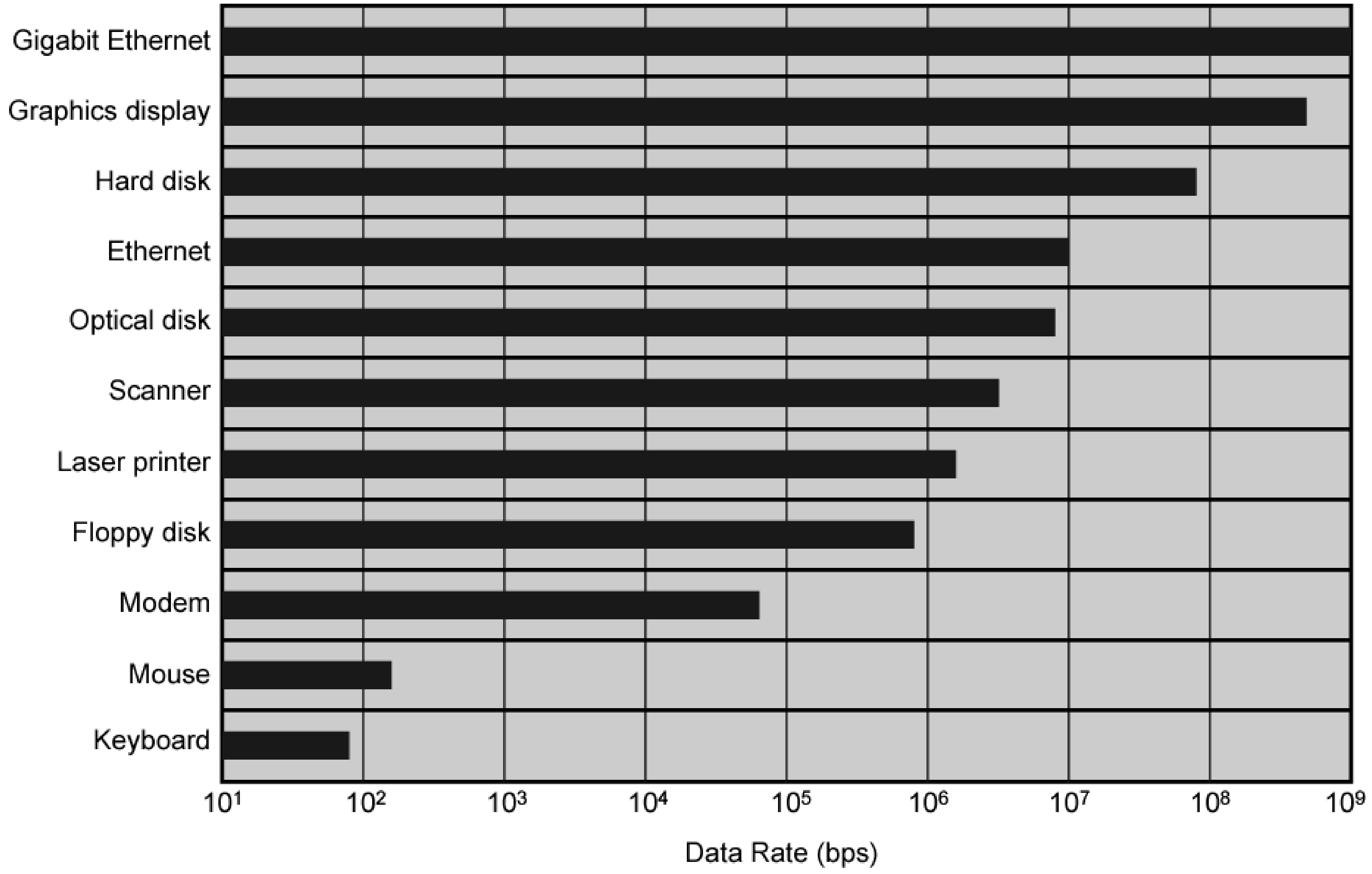
# Typical Motherboard (Pentium III)





**AGP -**
**PCI -**
**IDE –**
**BIOS -**

## Typical I/O Device Data Rates



From first to fifth/sixth generation systems, the following factors were also taken into consideration, to improve the performance.

- Reduced Power dissipation
- Reduced Space Area
- More increase in Speed and Registers (GPRs) for operation
- More memory size
- Use of Cache
- Set of cores on CPU
- pipelining and special MMX hardware.
-

*Key terminologies:*

- **Microcontroller**
- **CPU design**
- **Hardware description language**
- **Von-Neumann architecture**
- **Multi-core (computing)**
- **Datapath**
- **Dataflow architecture**
- **Stream processing**
- **Instruction-level parallelism (ILP)**
- **Vector processor**
- **Control Path**
- **ALU, FPU, GPU etc.**
- **Pipelining**
- **Cache**
- **Superscalar**
- **Out-of-order execution**
- **Register renaming**
- **multi-threading**
- **RISC, CISC**
- **Addressing Modes**
- **Instruction set**

- **SIMD, MIMD**
- **Flynn’s taxonomy**
- **MMX instructions**
-

## Boolean Algebra

## Introduction

- Basic mathematics needed for the study of logic design of digital systems

- All switching devices are two state devices, hence two-valued Boolean Algebra is called as switching algebra

- If X is a boolean (switching) variable then either X=0 or X=1

- Symbol 0, corresponds to a low voltage
  Symbol 1, corresponds to a high voltage

## Basic Operations

- AND
- OR
- Complement (Inverse)

### Complement

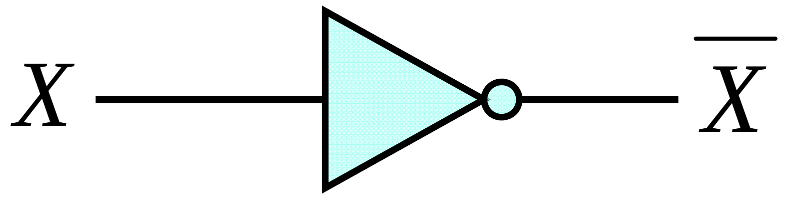
$$\overline{0} \text{ or } 0' = 1$$

$$\overline{1} \text{ or } 1' = 0$$

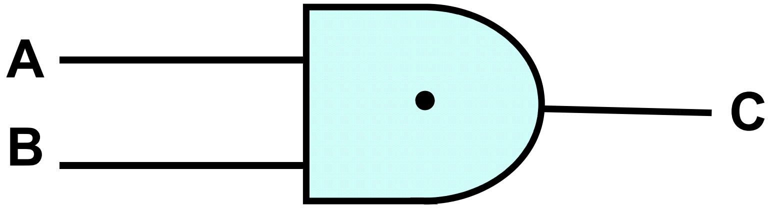
$$X' = 1, \quad if \quad X = 0$$

$$X' = 0, \quad if \quad X = 1$$

### Inverter



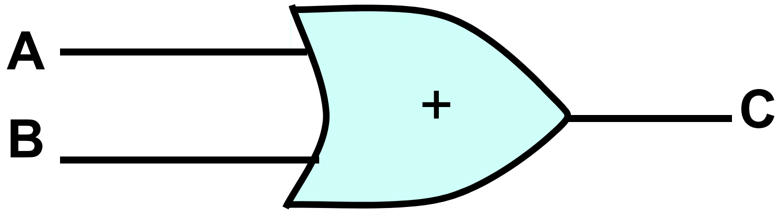
## AND



$$C = A \bullet B$$

| A | B | C |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

## OR

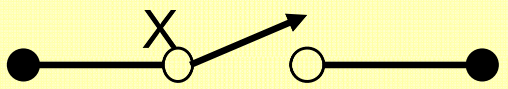


$$C = A + B$$

| A | B | C |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

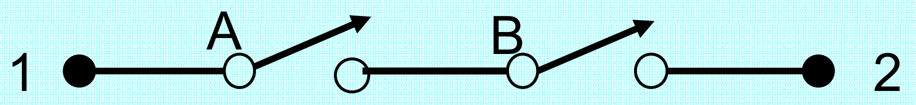
## Application of switching algebra to network with switches

**Switch**



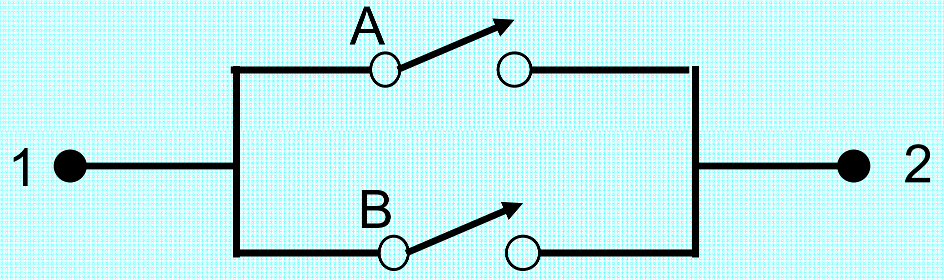
**X=0 → switch open**
**X=1 → switch closed**

**Two switches in series**



$$T = A \bullet B$$

**Two switches in parallel**



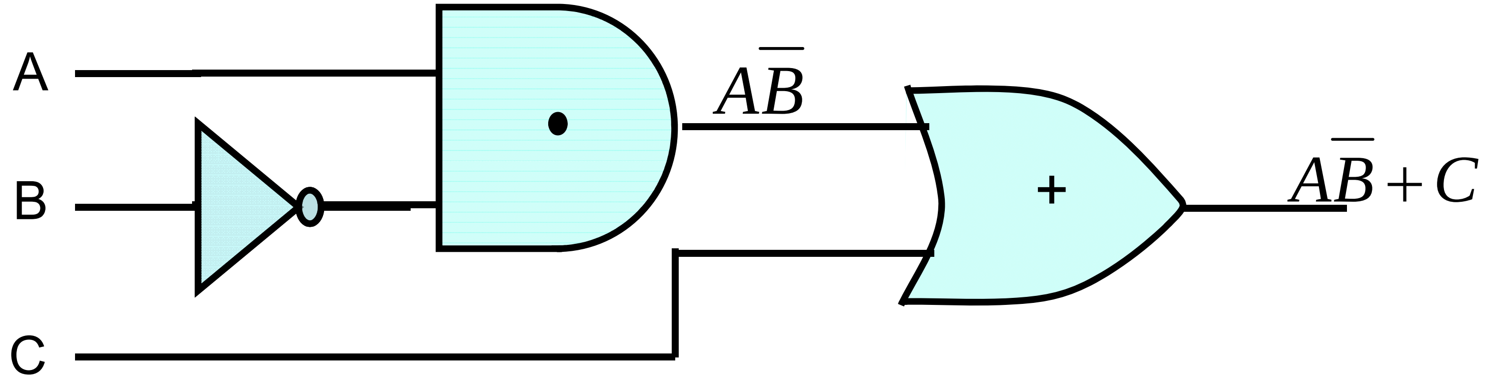
$$T = A + B$$

# Boolean Expressions and Truth tables

- Boolean expressions are formed by application of the basic operations to one or more variables or constants

- Simplest form $0\ ,X \quad or \quad Y'$

Example: $A\overline{B}+C$



- Each appearance of a variable or its complement in an expression will be referred to as a literal

$a\overline{b}c+\overline{a}b+\overline{a}b\overline{c}+\overline{b}\overline{c}$ has 6+4=10 literals

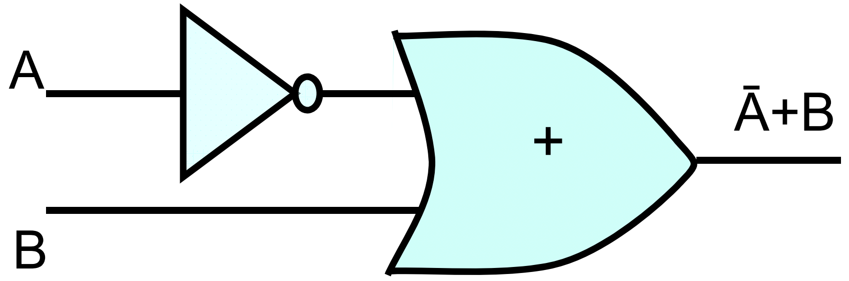
Each literal corresponds to a gate input

# Truth table

Table of combinations

➢ **Specifies the values of a Boolean expression for every possible combination of values of the variables in the expression**

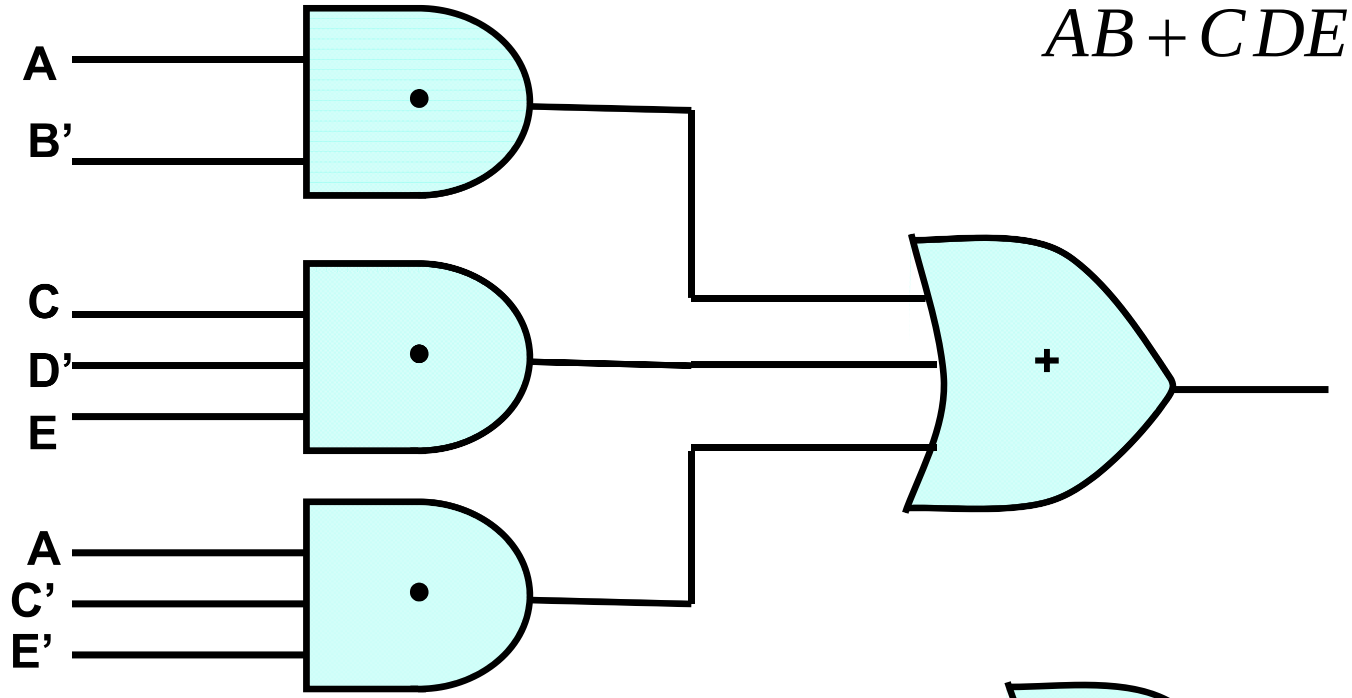
$$F = \overline{A} + B$$

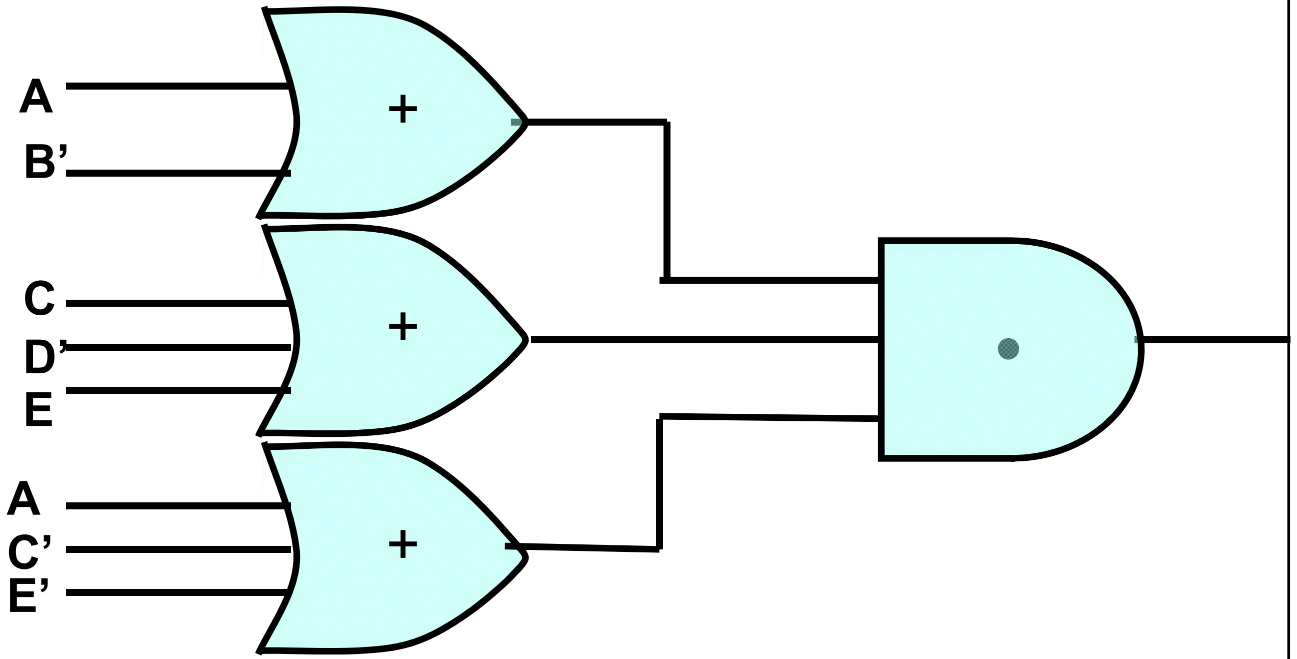


**Truth table**

| A | B | Ā | F= Ā+B |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 |

$$A\overline{B}+C\overline{D}E+A\overline{C}\,\overline{E}$$ **SOP form**





$$\left(A+\overline{B}\right)\left(C+\overline{D}+E\right)\left(A+\overline{C}+\overline{E}\right)$$ **POS Form**

***END of INTRO  to Hardware–***

***For calculations***
***on binary arithmetic***
***and***
***CPU Design***

***- wait for next set of courses***