Badji-Mokhtar University. Annaba
Faculty of Technology
Department of Computer Science

Chapter 1: General introduction

Dr. Khelifi Hakima

S1- 2025/2026

Lessons Objectives

- Architecture of a computer system
- Von Neumann architecture (ALU, buses, registers, RAM memory, ROM, access time, cache memory….)
- Machine language (binary code)
- Evolution of computers

Architecture of a computer system

Architecture of a computer system

- Computer System
  - Software
    - instructions
    - Systems Software i.g. Windows
    - Application Software i.g. Internet Explore
  - Hardware
    - Central Processing Unit (CPU) → Receives and executes instructions
    - Main Memory → Holds instructions that are about to be processed
    - Secondary Storage → Hard drive, DVD, etc
    - Input/Output Devices → Mouse, monitor, etc

Von Neumann Architecture

Von Neumann Architecture

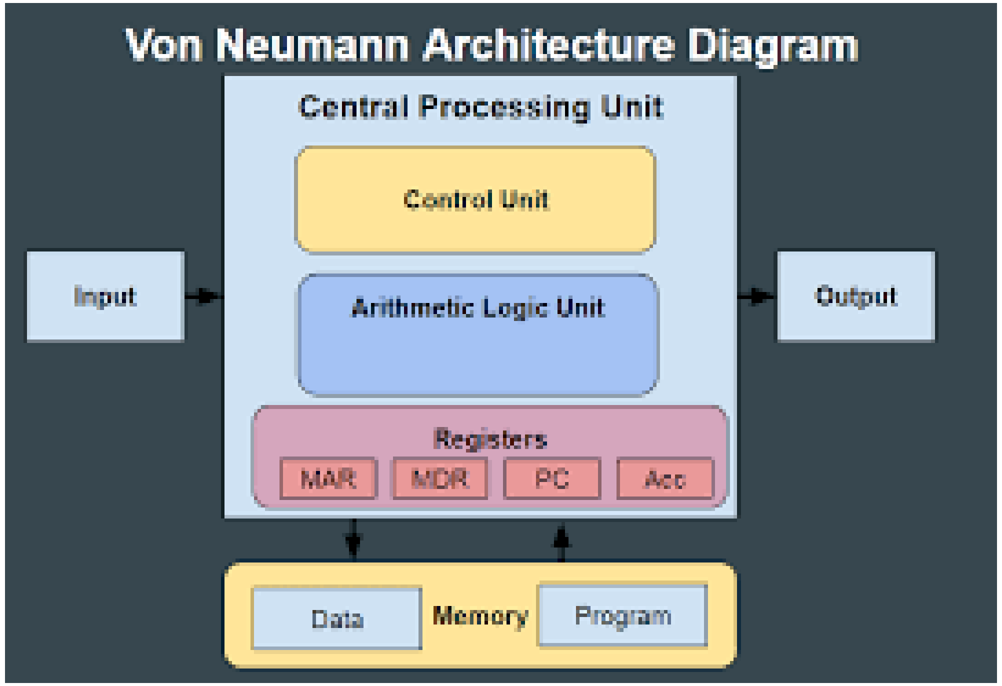
The early days:

- programs were not stored in the memory alongside data.
- Physically rewriting connections

ENIAC computer

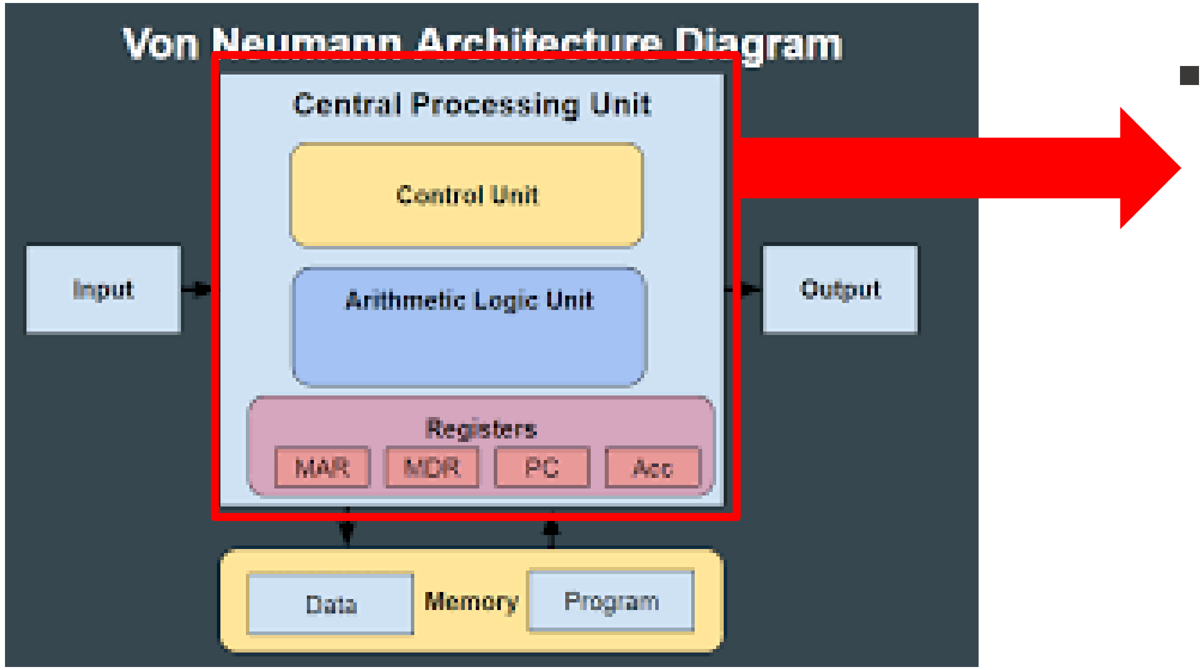
Von Neumann Architecture

- John von Neumann
- In 1945 he described a computer architecture in which the data and the program are both stored in the same memory.
- This is the fundamental design concept behind all modern computer systems



Central processing Unit

Central processing Unit

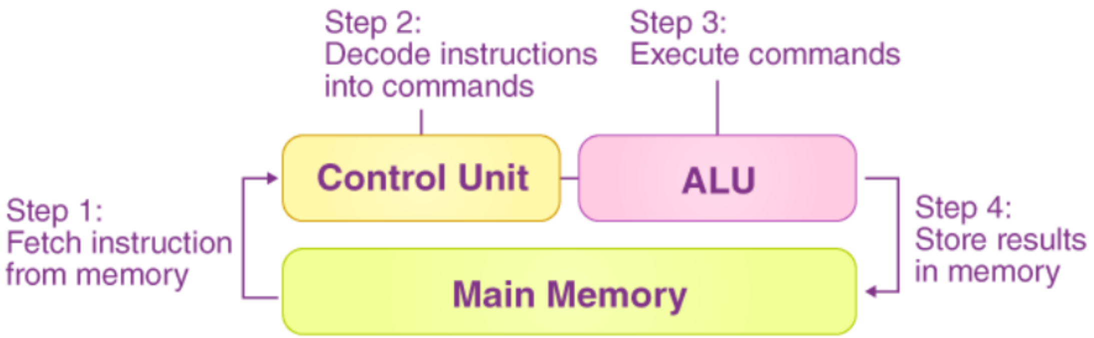


- **CPU (Central Processing Unit)**: receives and executes instructions from memory, these instructions are provided by a computer program.

Central processing Unit

Control Unit (CU)

- Generates the necessary control signals to coordinate the activities of CPU.
- Manages the flow of data between the CPU, memory, and input/output devices.
- Manages the execution cycle (fetch-decode-Execute cycle) → Store result back into memory.
- Ensures that all operations happen in a well-timed (clock signal).



Central processing Unit

Arithmetic and Logic Unit (ALU)

- Performs Arithmetic and logical operations.

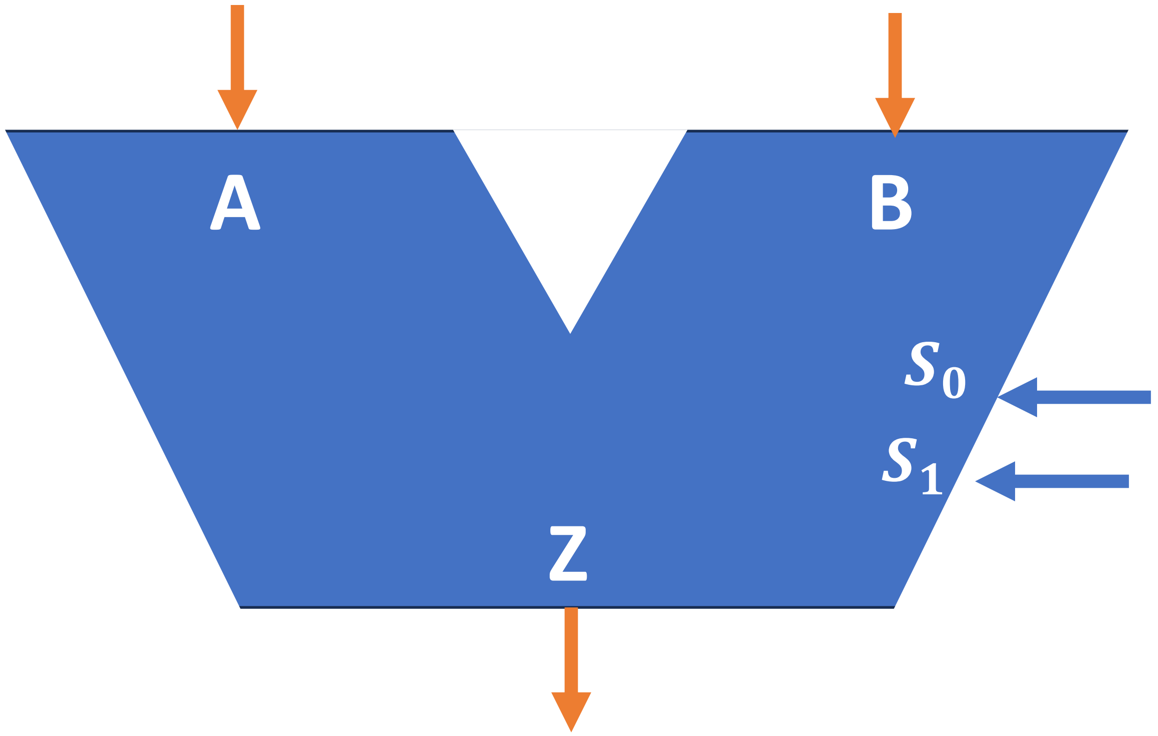
- Handles all mathematical calculations and decision- making tasks.

Arithmetic operations
Z = A+B
Z= A-B
Z=A*B
Z=A-1
Z= A+1

Logical operations
Z=NOT A
Z=A AND B
Z= A OR B
Z=A
Z=0

Central processing Unit

Arithmetic and Logic Unit (ALU)



0	0	0	0	1	0	1	0	**A**

[redacted] **B**

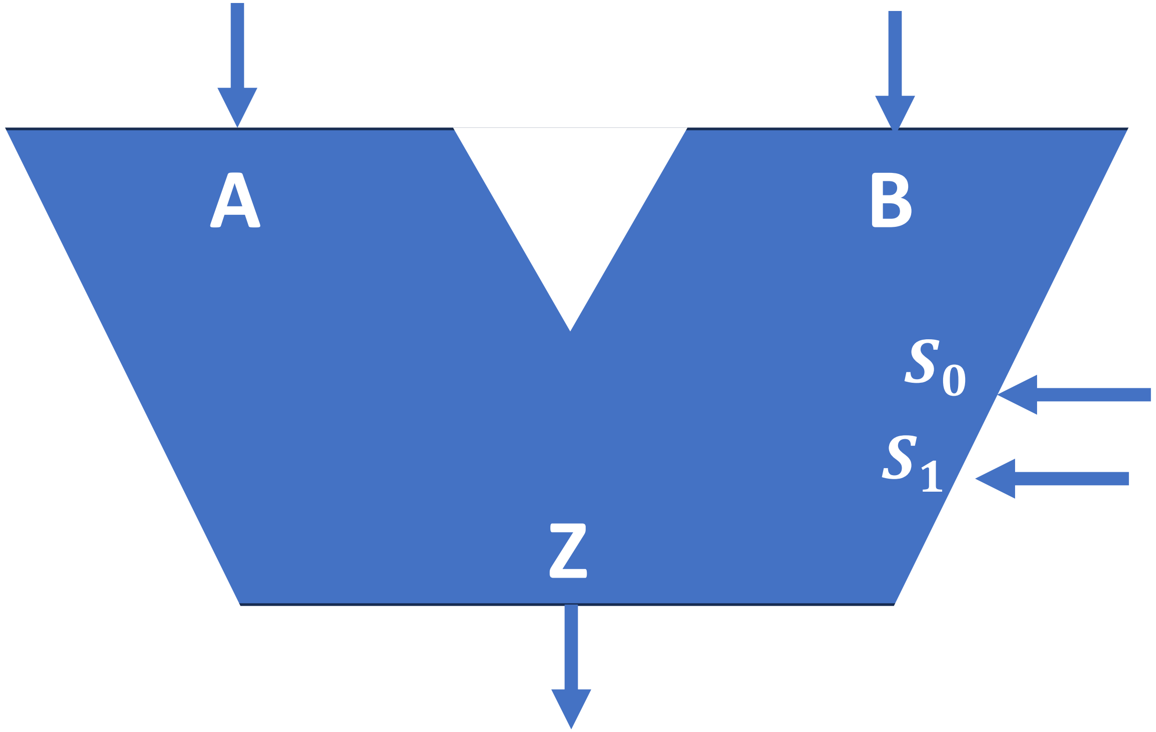
S_0	S_1
0	1

0	0	0	0	1	0	1	1	**Z**

S_0 S_1	operation
0 0	Z = A+B
0 1	Z= A+1
1 0	Z=NOT A
1 1	Z=0

Central processing Unit

Arithmetic and Logic Unit (ALU)



0	0	0	0	1	0	1	0	**A**

[redacted] **B**

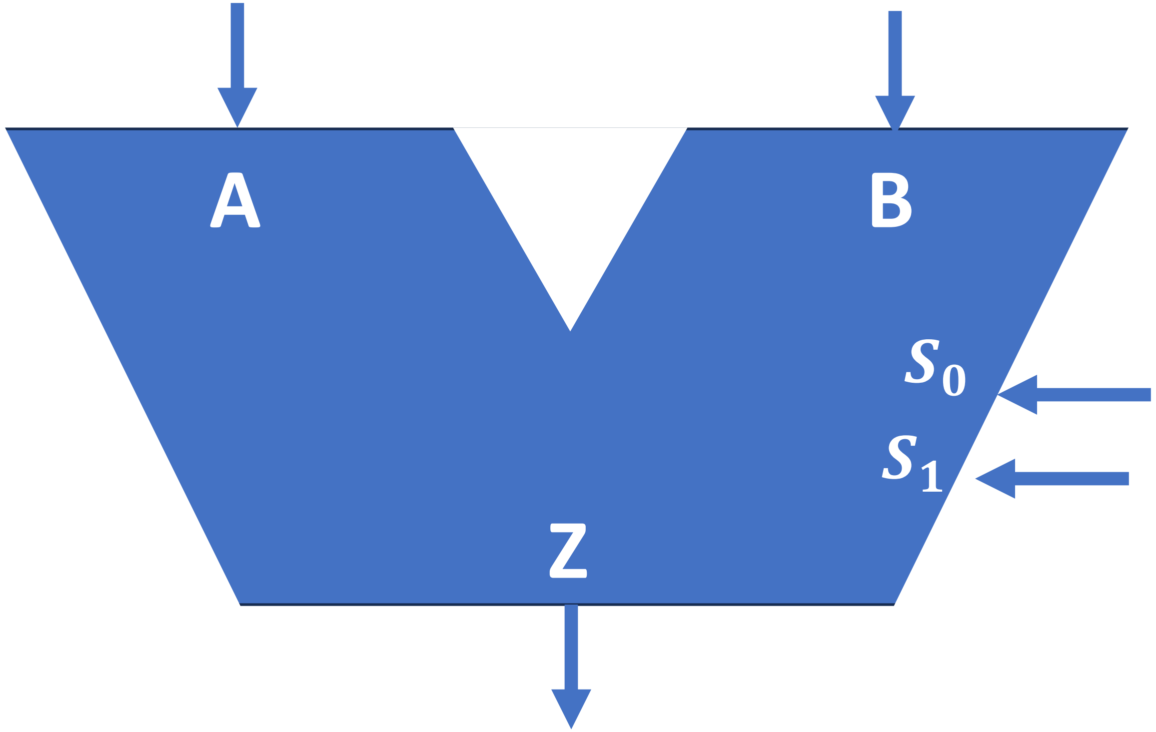
1	1	1	1	0	1	0	1	**Z**

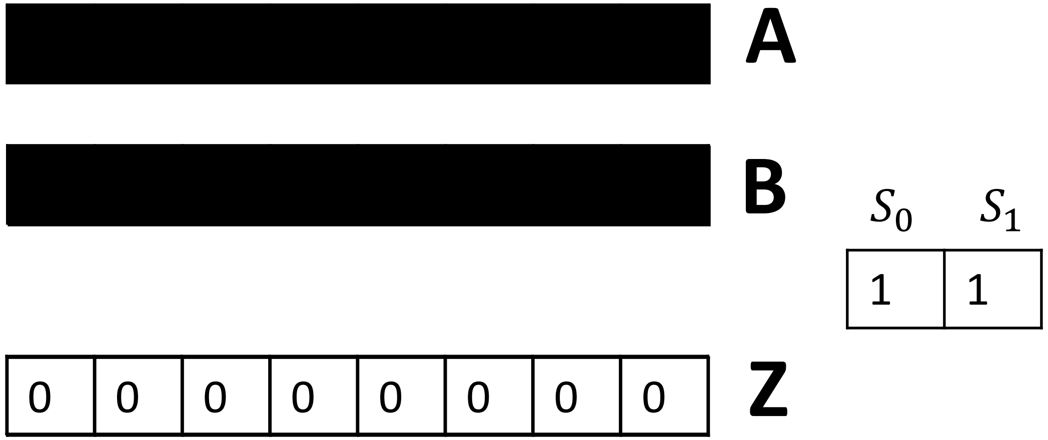
S_0	S_1
1	0

S_0 S_1	operation
0 0	Z = A+B
0 1	Z= A+1
1 0	Z=NOT A
1 1	Z=0

Central processing Unit

Arithmetic and Logic Unit (ALU)



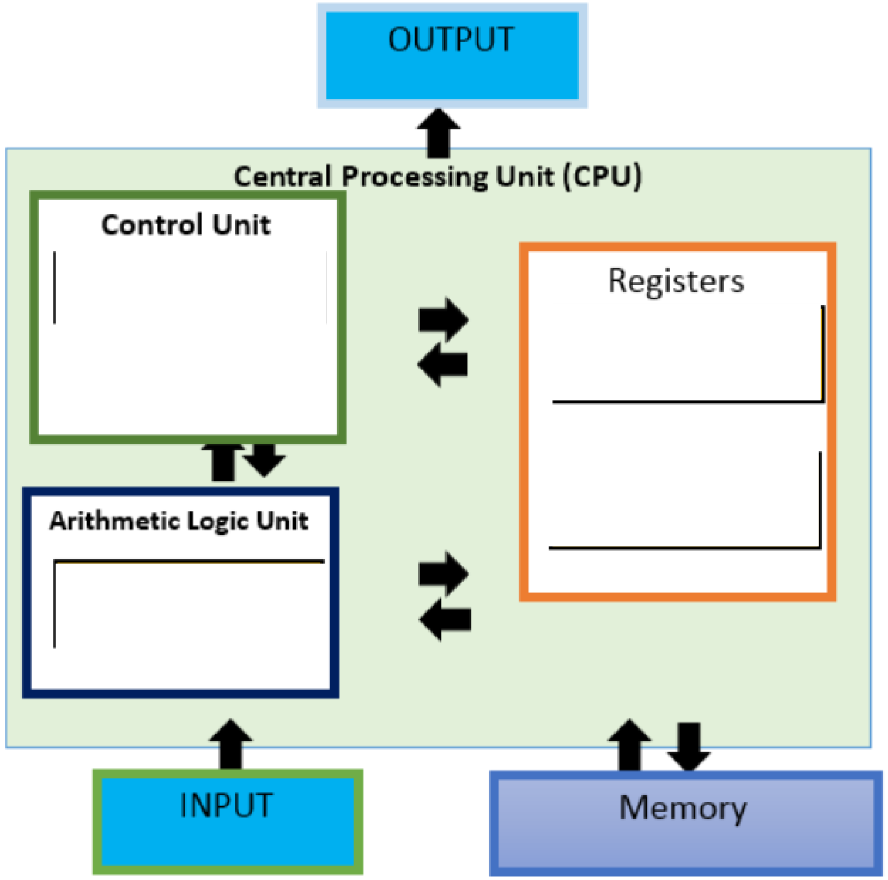


S_0 S_1	operation
0 0	Z = A+B
0 1	Z= A+1
1 0	Z=NOT A
1 1	Z=0

Central processing Unit

- **Registers**: are small, fast memory locations within the CPU that can be accessed very quickly for specific purposes. It holds the instructions and data the computer is using right now.

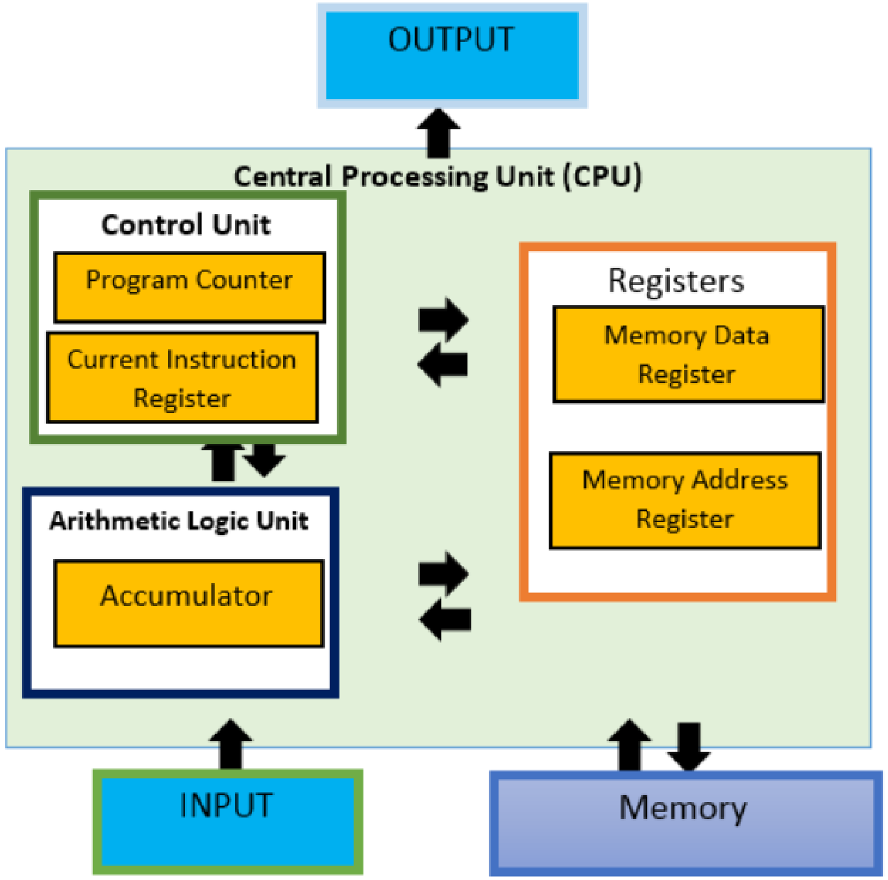
- **Types of register** :
  - ✓ Accumulator
  - ✓ Program counter(PC)
  - ✓ Memory Address Register (MAR)
  - ✓ Memory Data Register (MDR)
  - ✓ Current Instruction Register (CIR)



Central processing Unit

- **Registers**: are small, fast memory locations within the CPU that can be accessed very quickly for specific purposes. It holds the instructions and data the computer is using right now.

- **Types of register** :
  - ✓ Accumulator
  - ✓ Program counter(PC)
  - ✓ Memory Address Register (MAR)
  - ✓ Memory Data Register (MDR)
  - ✓ Current Instruction Register (CIR)



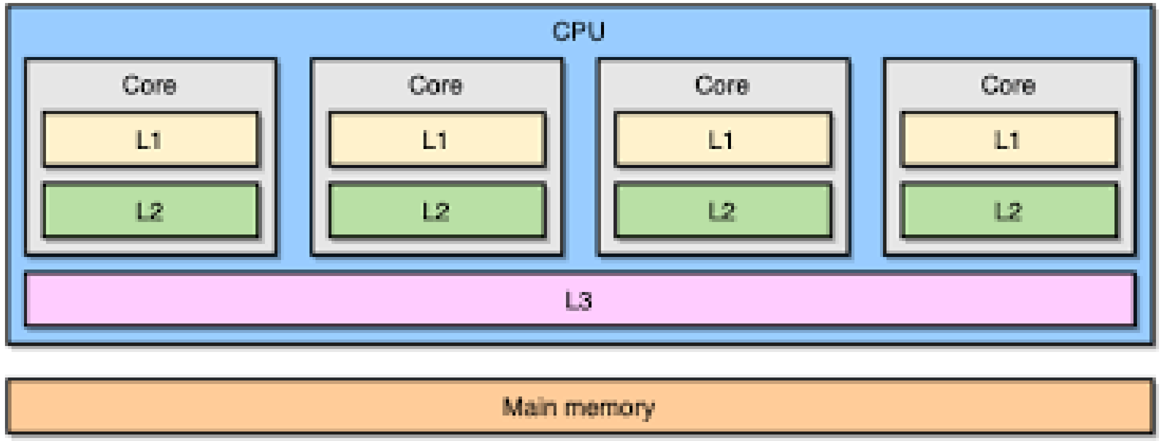
Central processing Unit

- **Bus**: enable data and control signals to move around a CPU, memory, and input/output devices.

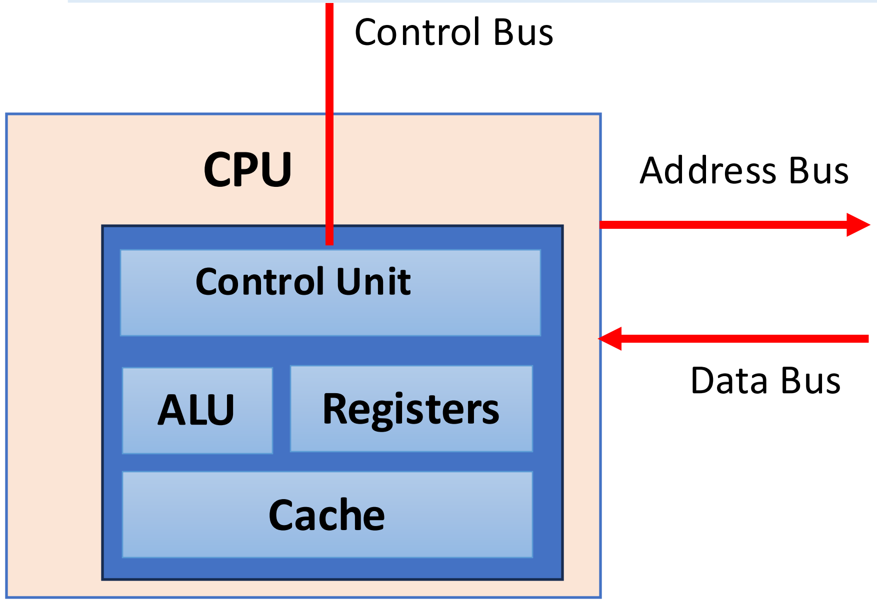
- ✓ **Control bus**: indicates whether the operation is read or written and ensures that the operation happens at the right time (clock signals).
- ✓ **Address bus**: carries addresses of memory for the memory location to be read from or written to.
- ✓ **Data bus**: carries data between the CPU, memory, and other peripherals.

Central processing Unit

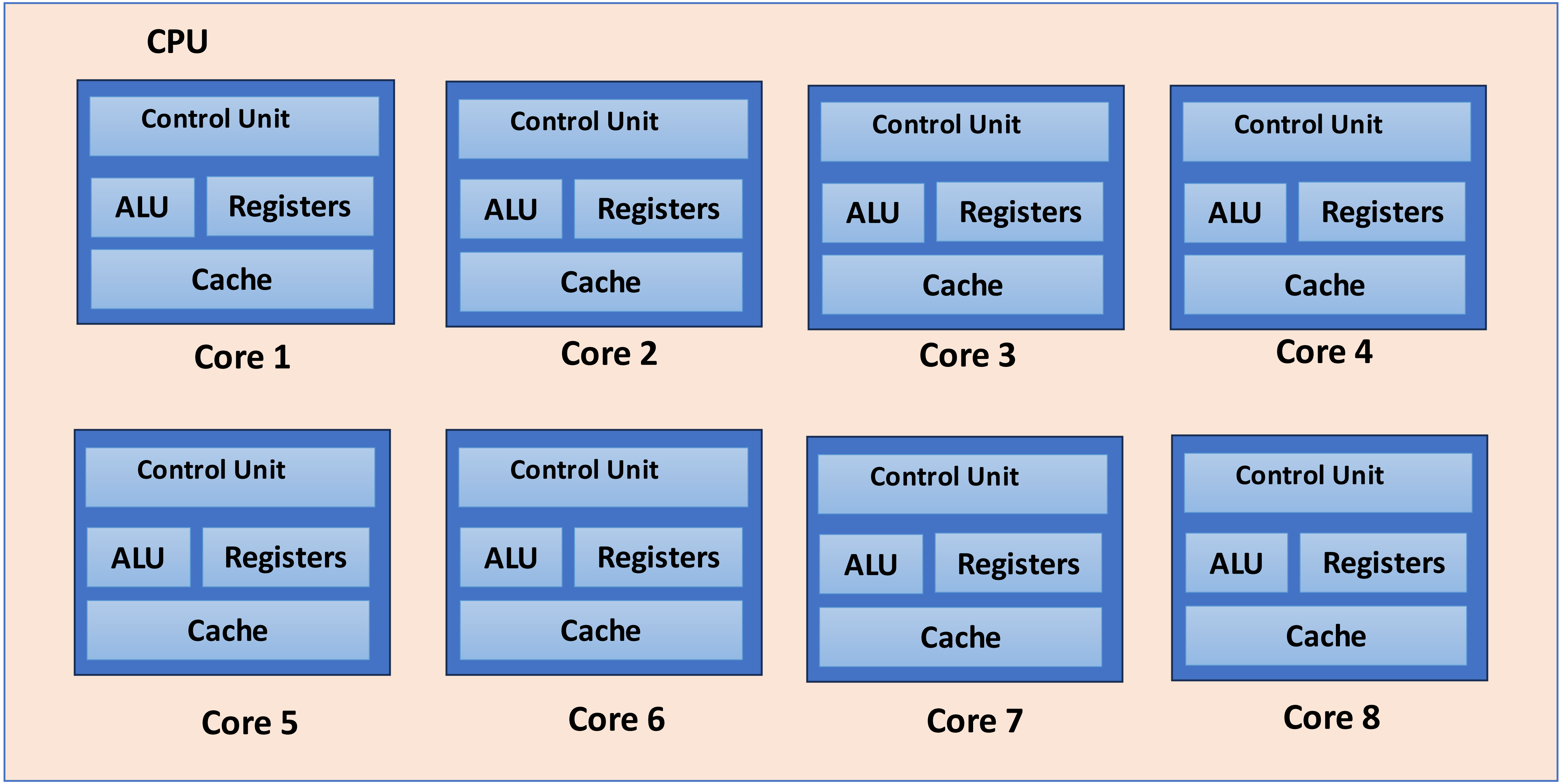
- **Cache memory**: built into CPU to store instructions or data that are frequently used or about to be used.
- Types:
  - ✓ L1 (Level 1): 32KB-256KB per core
  - ✓ L2 (Level 2): 256KB – 2MB per core
  - ✓ L3 (Level 3) 2MB- 50 MB



Central processing Unit



Central processing Unit



Central processing Unit

CPU performance:

1. **Clock speed**: each time it ticks, the CPU can process one instruction. It is measured in hertz (Hz).

   - megahertz (MHz) = 1 million Hz

   - gigahertz (GHz) = 1 billion Hz

2. **Number of cores** (single/ dual/quad/hexa/octa) : more processing units allowsmore than one instruction to be executed at the same time.

   - Mutlitasking

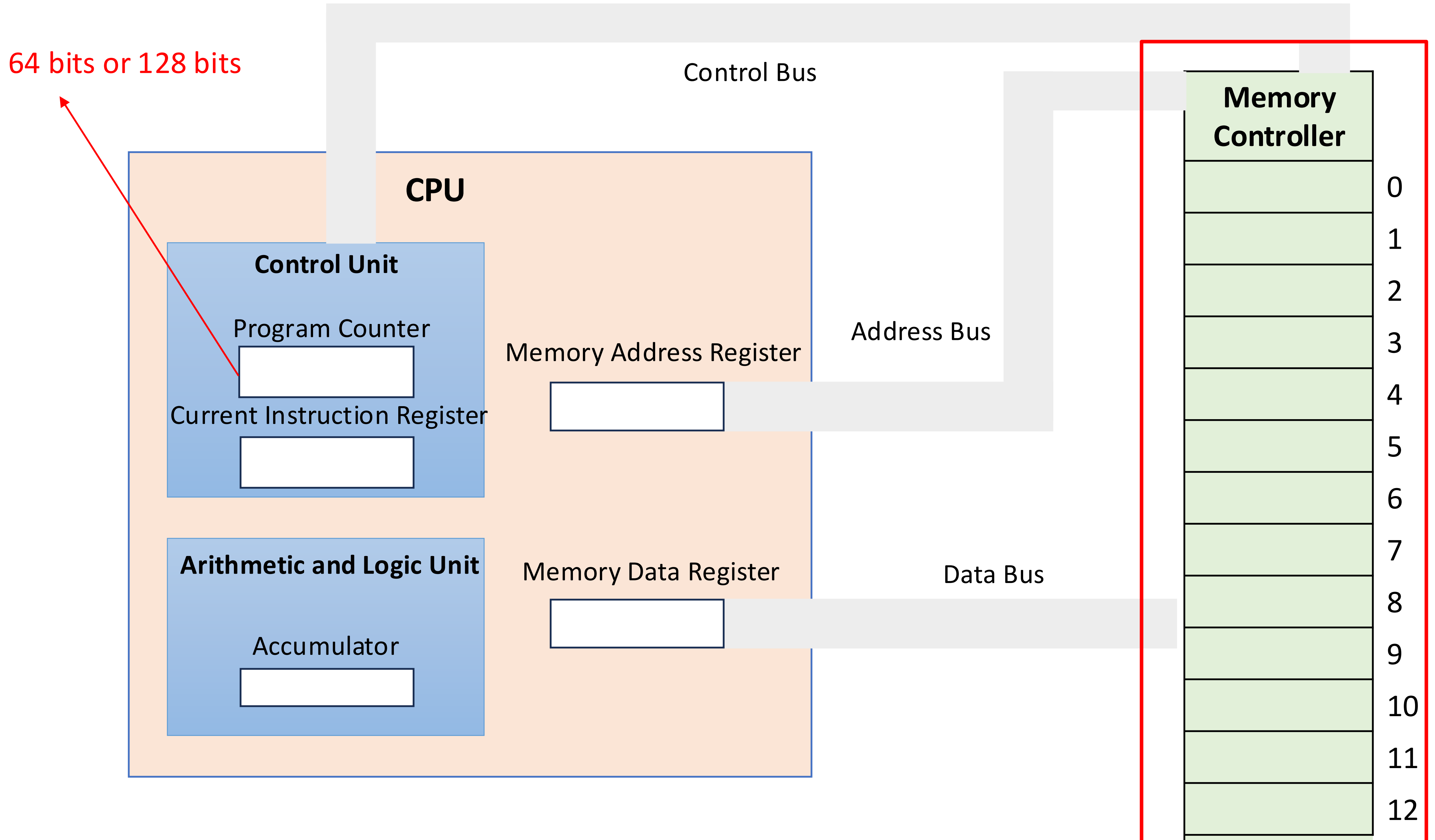
   - Parallel Processing

3**. Cache Memory**: a small amount of fast memory built into the CPU.

   - More cache = less time fetching data from memory

Fetch-Decode-Execute Cycle

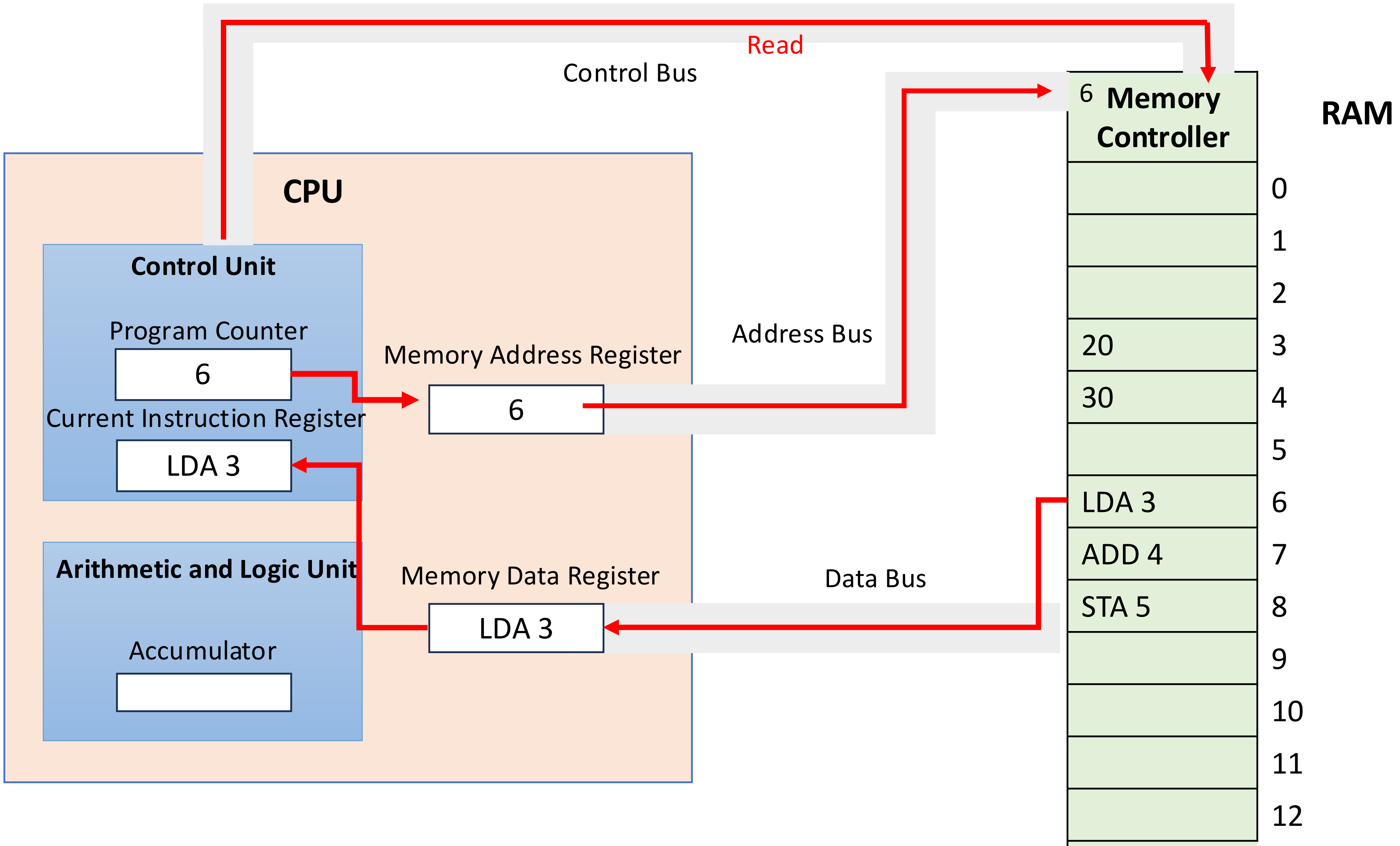
Fetch first instruction



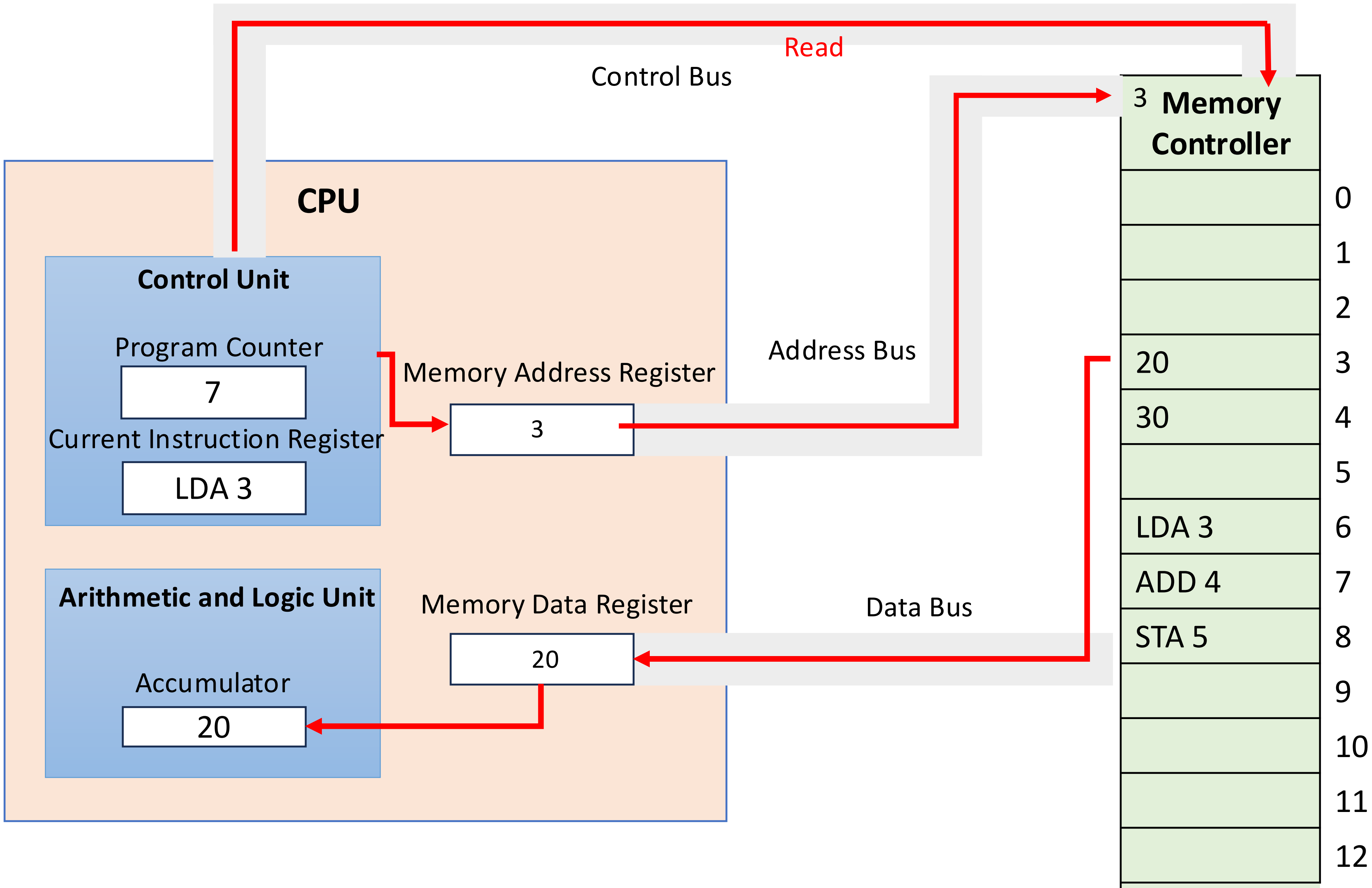
Central processing Unit

High level instruction		Low level code		Machine code
Z = x + y	→	LDA 3 ADD 4 STA 5	→	1001 0011 1100 0100 1110 0101

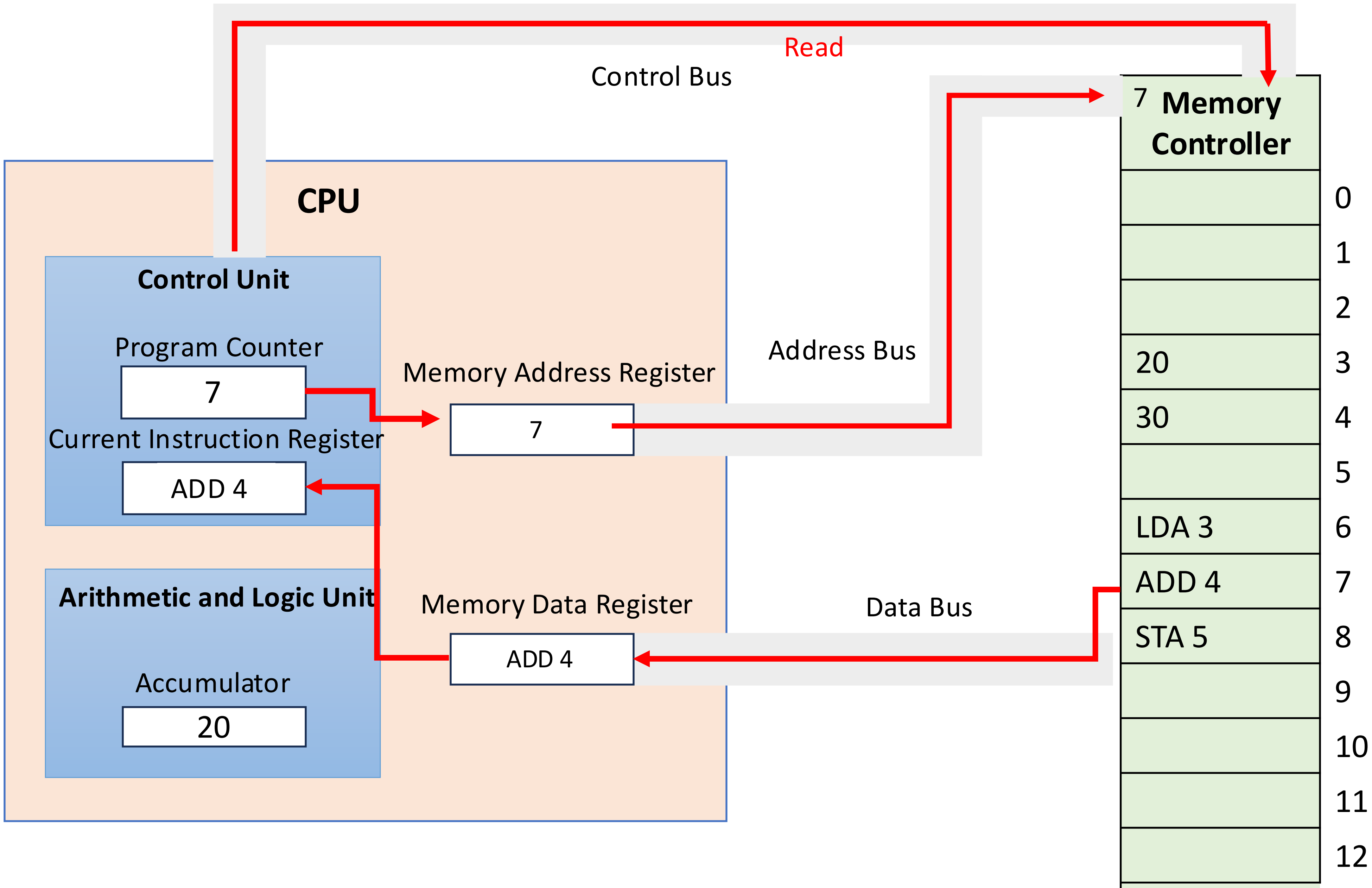
Fetch first instruction



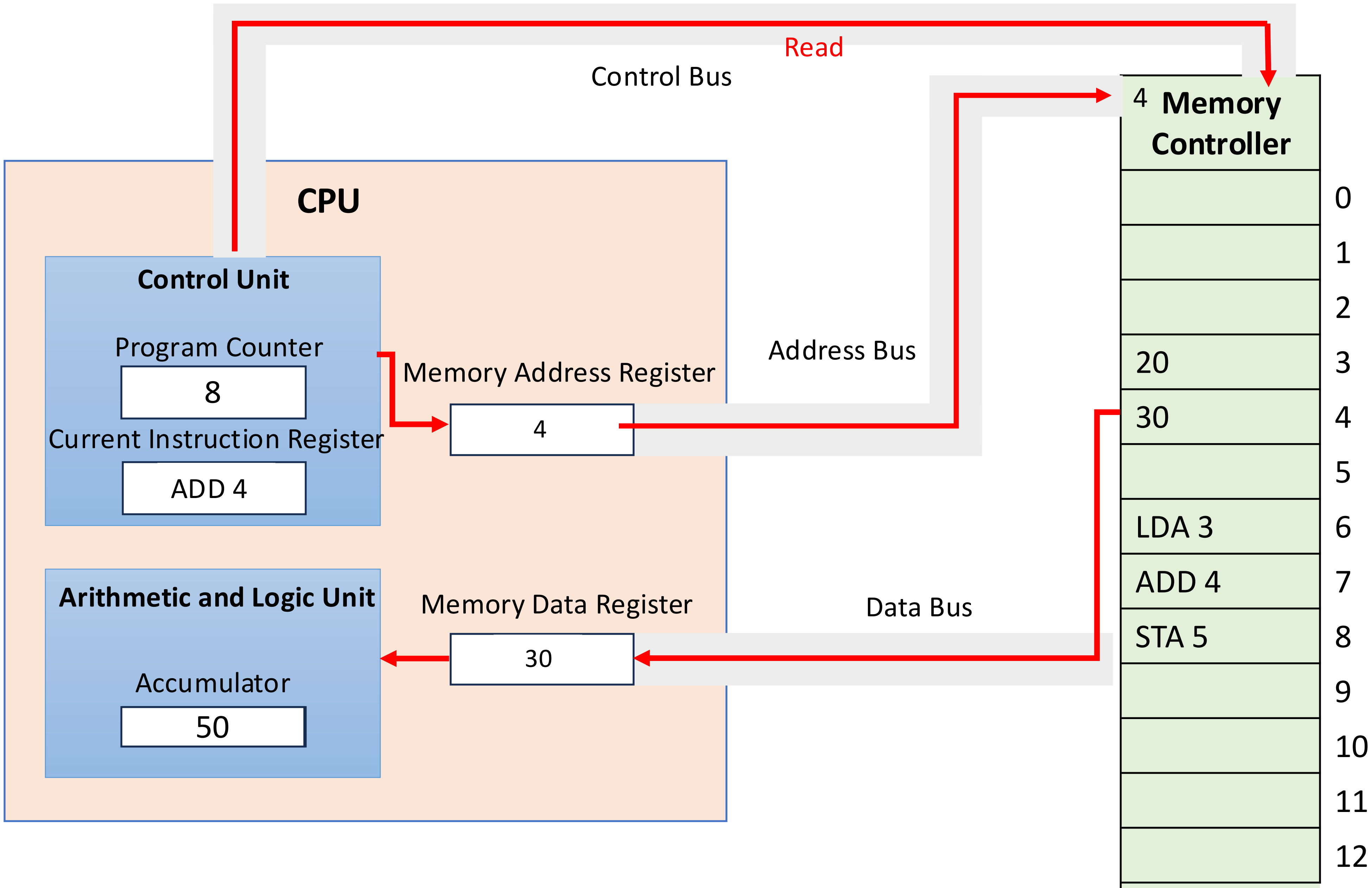
Execute first instruction



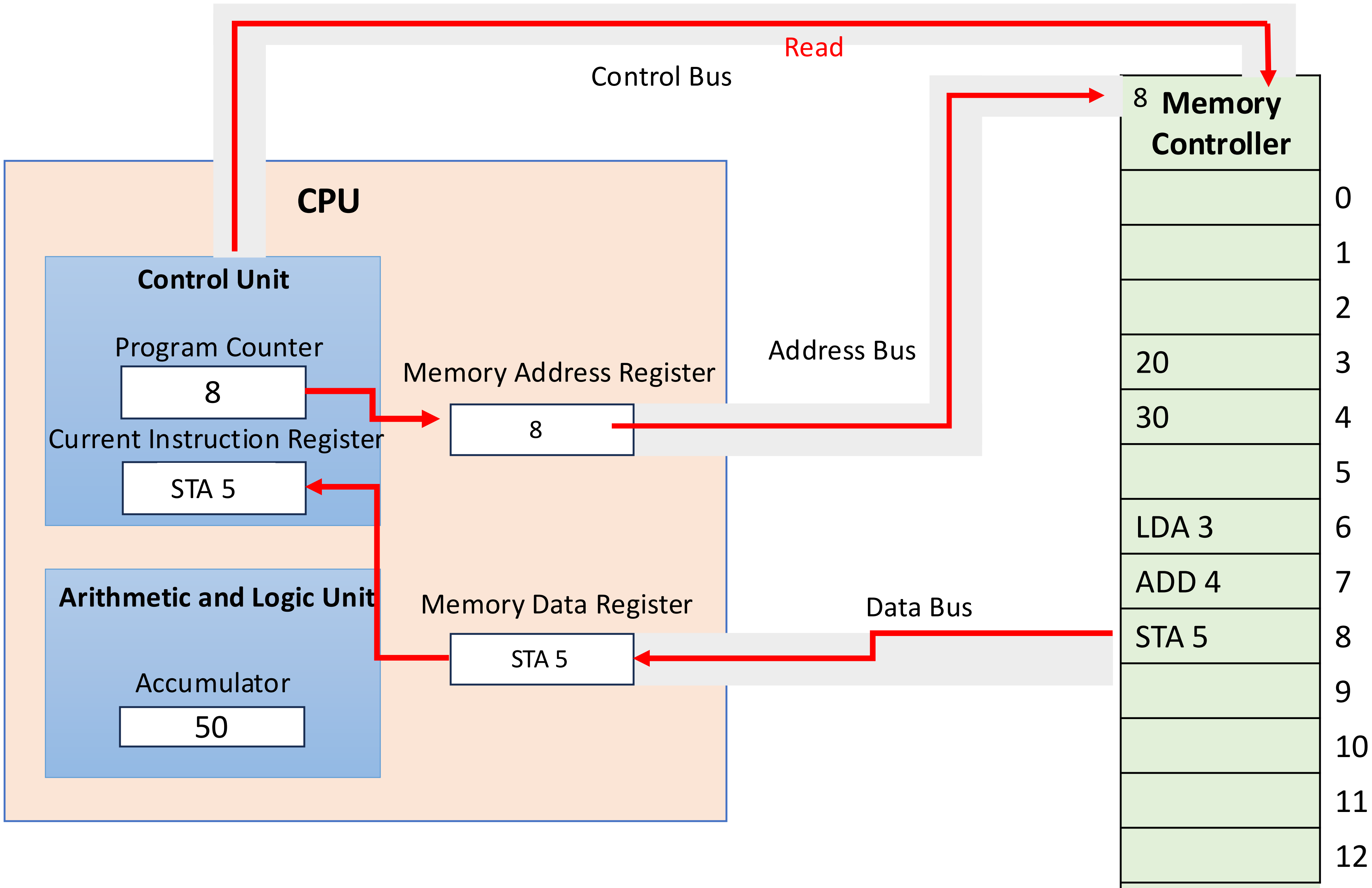
Fetch second instruction



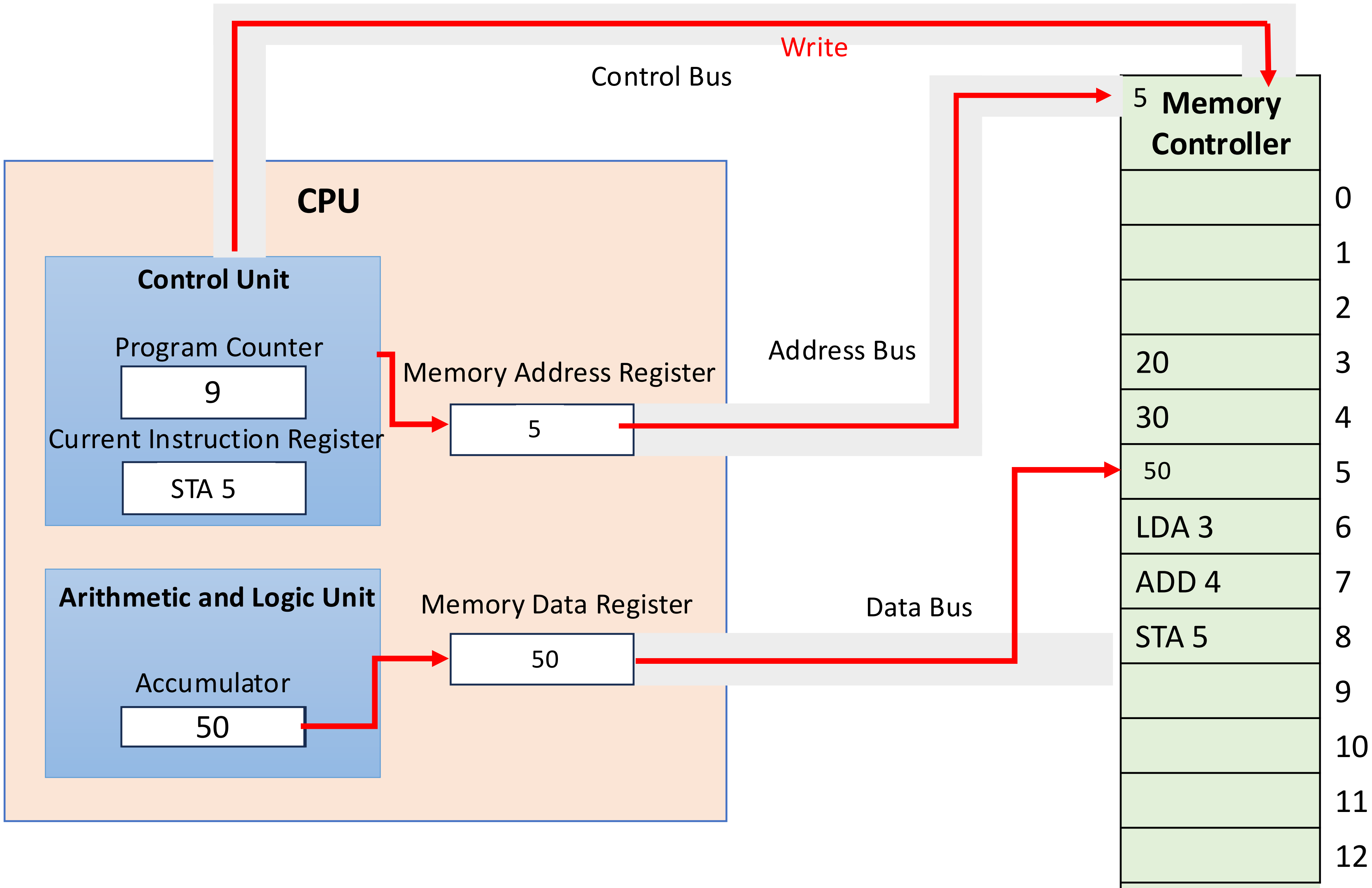
Execute second instruction



Fetch third instruction



Execute third instruction



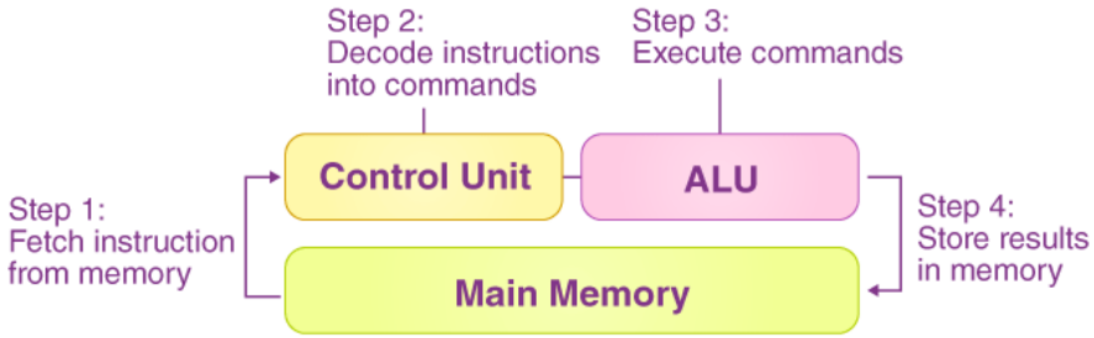
Fetch-Decode-Execute Cycle

Step 1 – **Fetch**: The CPU fetches **instruction** and **data** from RAM.

Step 2 – **Decode**: The control unit decodes the **instruction**.

Step 3 – **Execute**: The ALU executes **instruction** and operates on **data**.

Step 4 – **Store**: The processed **data** is stored in the RAM.



Main Memory

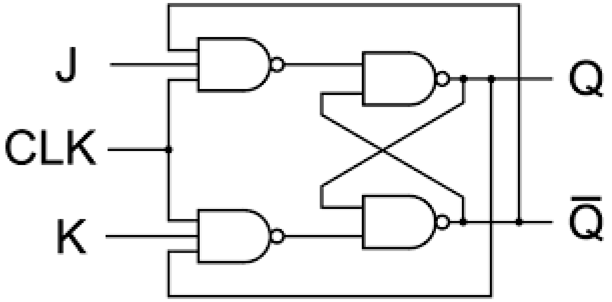
Main Memory

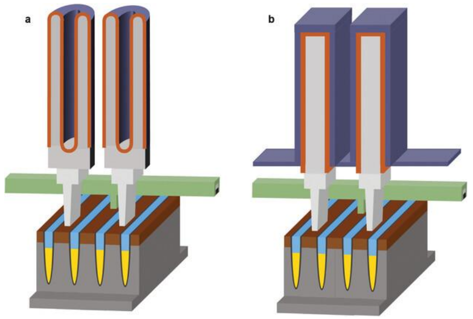
Main Memory or Primary Memory:

- Stores data and instructions that the CPU needs for quick access.
- volatile (stored information lost when the power is turned off).
- Types:
  - **RAM (Random Access Memory)**
  - **ROM (Read Only Memory)**:

Main Memory

- **Main Memory or Primary Memory**:
  - **RAM (Random Access Memory)**: accessed directly by the CPU, where it is used for temporary storage. The RAM holds billions of storage locations, each has its own memory address.
    - SRAM: not need refreshing, faster, expensive, less power
    - DRAM: need refreshing, slower, cheaper, more power



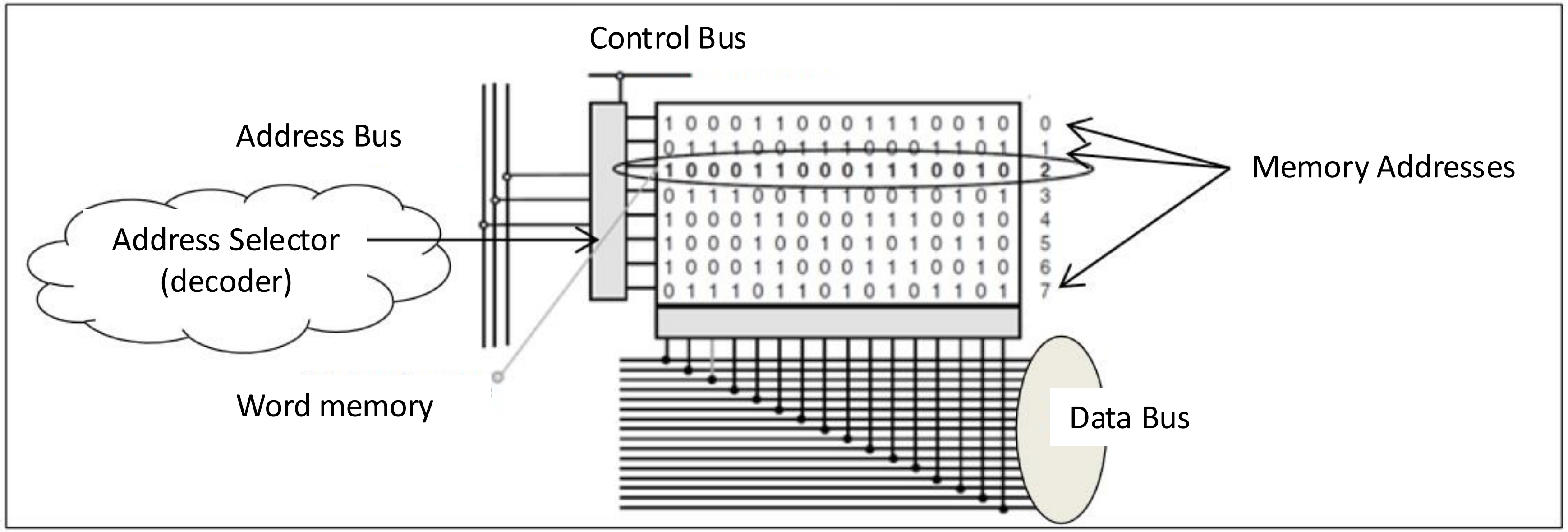


Main Memory

- **Main Memory or Primary Memory**:
  - **RAM (Random Access Memory)**: accessed directly by the CPU, where it is used for temporary storage. The RAM holds billions of storage locations, each has its own memory address.
    - SRAM: not need refreshing, faster, expensive, less power
    - DRAM: need refreshing, slower, cheaper, more power
  - **ROM (Read Only Memory):** Non-volatile, the data cannot be changed. Often used to store important data such as how to start/boot the computer.

Main Memory

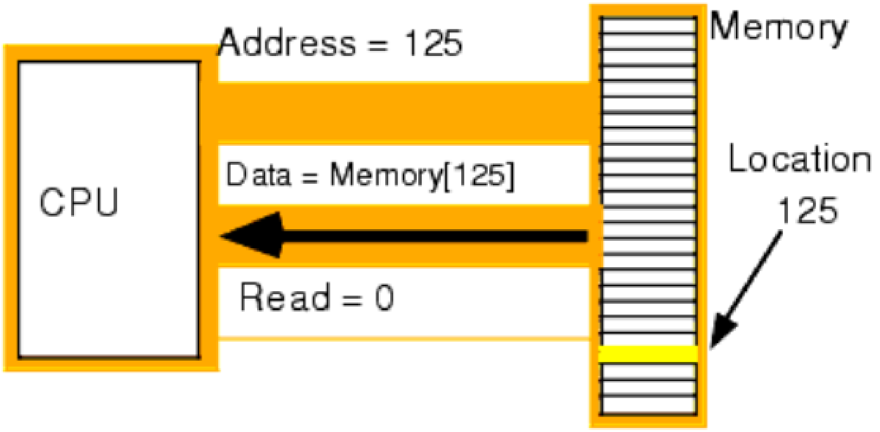
- The smallest unit that constitutes memory is the **bit**
- Two stable states coded in the form of a **0 or a 1**
- It is divided into memory cells called memory **words**
- Each word is identified by its address in the memory
- The size of the memory word: 4 bits (quartet) or even 8 bits (octet or byte), 16 bits or even 32 or 64 bits.

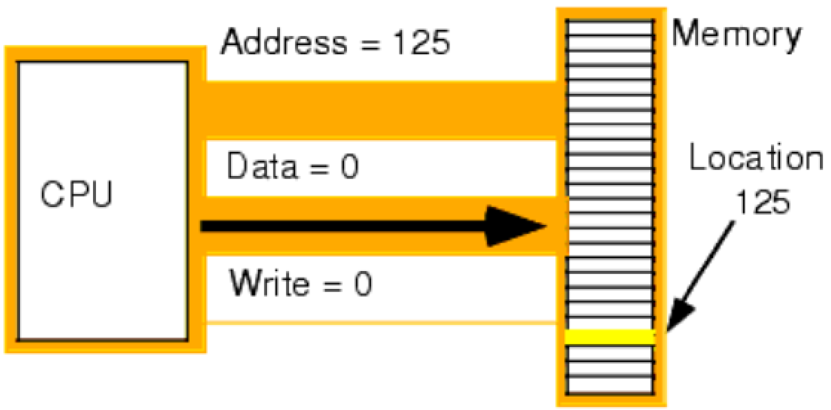


Main Memory

The possible operations on the memory are:

- **Reading**: A word reading operation consists of defining the address of the word and triggering a read command that brings the content of the word from the memory to the CPU.
- **Writing**: A write operation consists of defining the address of the word whose content we want to change and then triggering a write operation that transfers the information from the CPU to the memory word whose address is specified.





Secondary Storage

- Secondary Storage: Non volatile

  - Contents are kept even when the power is off.

  - Long term data storage.

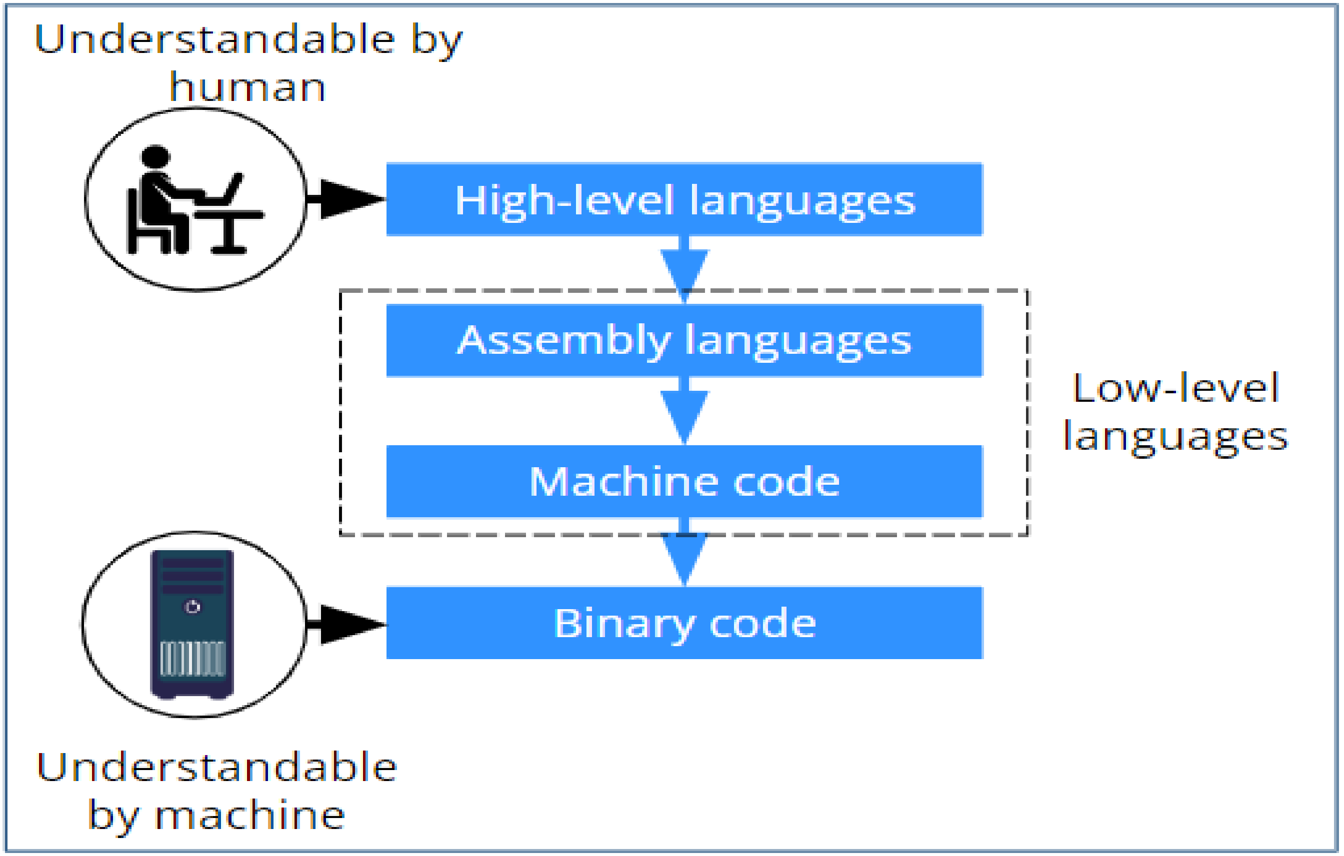
  - Types:

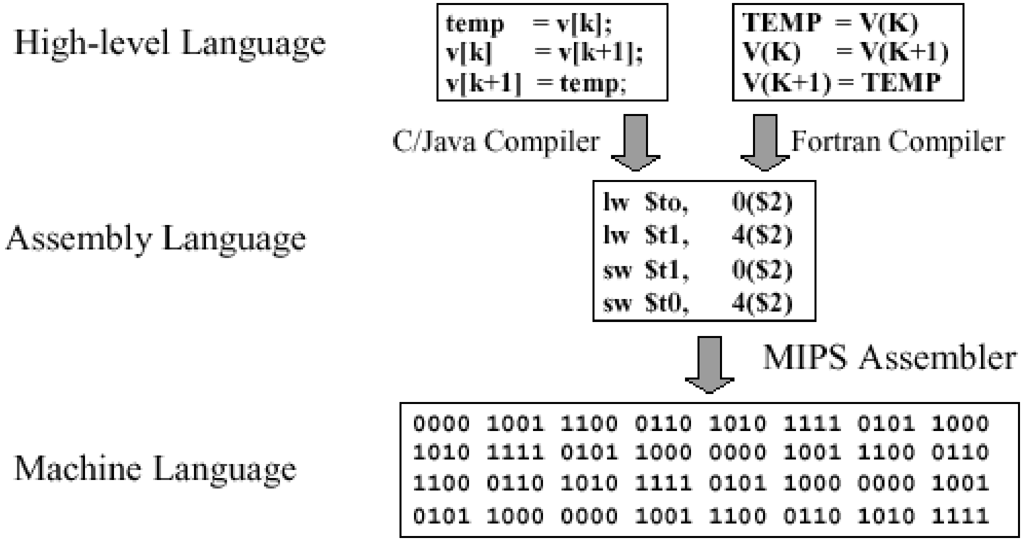
    - **Magnetic - hard disk drive (HDD)**
    - **Flash – solid state drive (SSD)**
    - **Optical**: CD, DVD, blu-ray
    - **Cloud storage**: onedrive, dropbox




Machine Language

Machine Language





Machine Language

Machine Language:

- Called machine code
- Low-level programming language
- Native to a processor: executed directly by hardware
- Instructions consist of binary code: 1s and 0s (instructions are written in binary e.g. 1010011).

Machine Language

Characteristics:

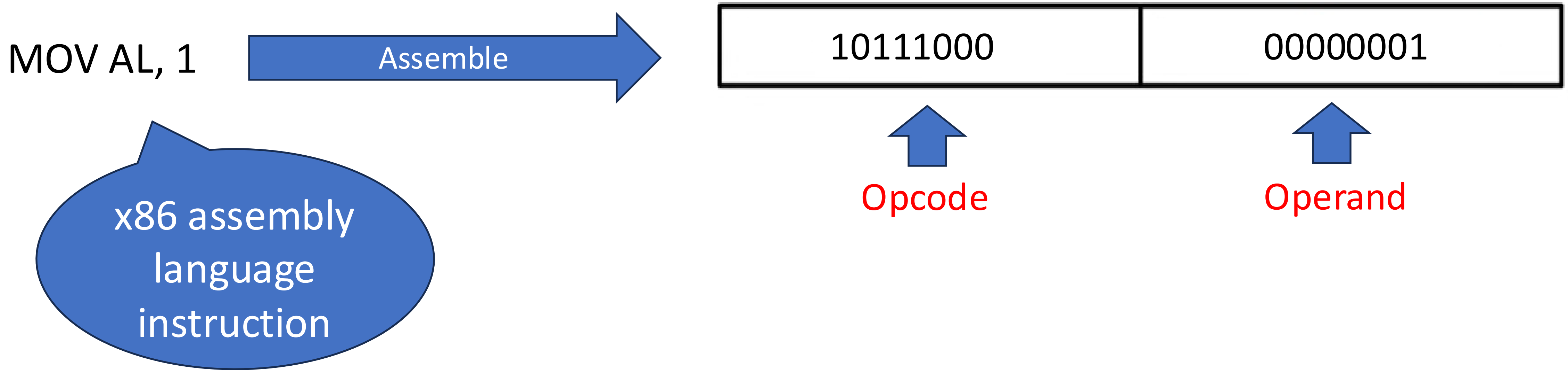
- **Binary Format**: instructions are written in binary (e.g. 10010101).
- **Processor specific**: code written for one processor won't run on another without modification (e.g. x86, ARM, MIPS).
- **Instruction Set Architecture (ISA)**: defines the set of operations the CPU can perform
- **No abstraction**: deals directly with memory addresses, CPU registers, and basic arithmetic and logical operations.

Machine Language

Machine language instructions: consists of operation code and operand.

Operation Code	Operand

Example:



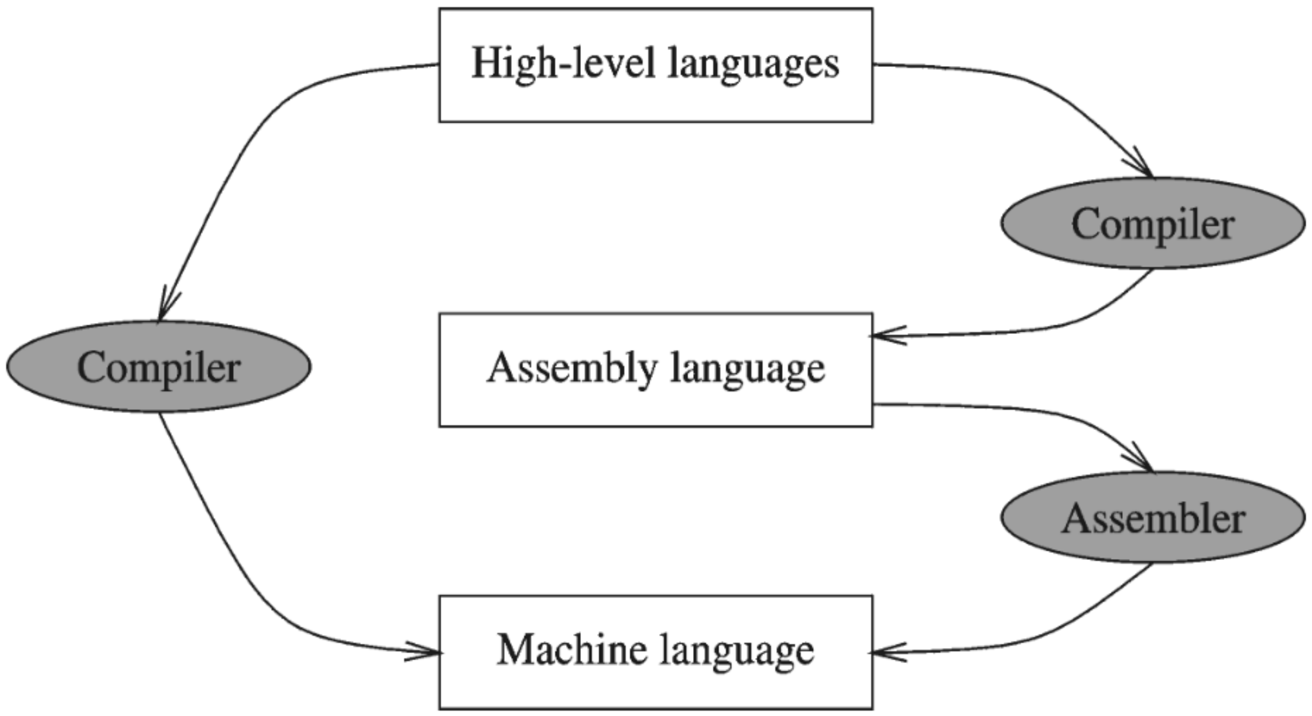
Machine Language

Assembly language:

- A programming language that uses symbolic names (e.g. mnemonics, ex: MOV AL, 1) to represent operations, registers, and memory locations.
- Slightly higher-level language
- Readability of instructions is better than machine language
- Assembly languages: ARM, MIPS, x86, Z80

Machine Language

- Compilers translate high-level programs to machine code (directly or indirectly via an assembler).
- Assemblers translate assembly to machine code.



Evolution of computers

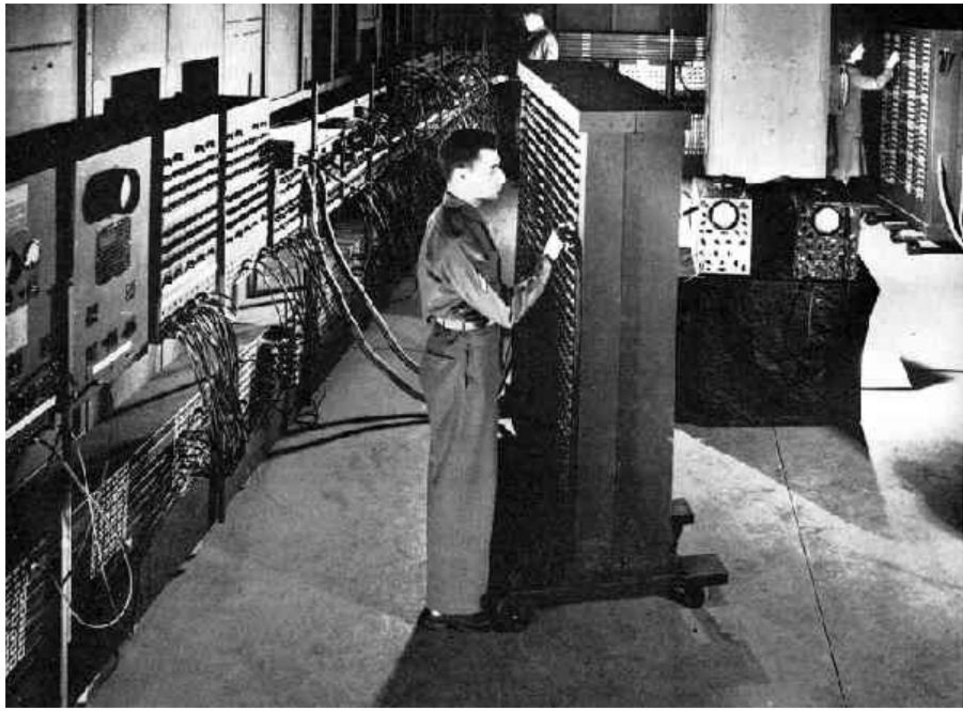
Evolution of computers

There are five generations of computers:

- **First Generation of Computers:** The first generation of computers used the basic programming language (e.g. machine language) but without operating systems. Where these computers were very heavy with big size. They used vacuum tubes as switches and amplifiers with magnetic drums for memory. These computers required a large room, and consumed a lot of electricity, with the complexity of programming.
- Some examples of the first-generation computers include:

Evolution of computers

- Some examples of the first-generation computers include:
  - ✓ ENIAC (Electronic Numerical Integrator and Computer)
  - ✓ EDVAC (Electronic Discrete Variable Automatic Computer).



Vacuum Tube

Evolution of computers

- **Second Generation of Computers:** In this generation, vacuum tubes were replaced with transistors, making computers smaller, faster, and more reliable. This generation also presented the use of CPU, memory, and input/output units, and used for programing FORTRAN (1956), ALGOL (1958), and COBOL (1959).

Transistor

Evolution of computers

- **Third Generation of Computers:** Third-generation computers started using integrated circuits also known as semiconductor chips instead of transistors, and contain thousands of tiny transistors. As a result, computers become faster, smaller in size, more reliable, and with less computational time. In this generation, keyboards and monitors were used instead of punch cards, also used remote processing, time-sharing, multiprogramming operating systems, and High-level languages like FORTRAN-II TO IV, COBOL, PASCAL PL/1, BASIC, ALGOL-68, etc.

Integrated Circuit

Evolution of computers

- **Fourth Generation of Computers:** Very Large Scale Integrated (VLSI) circuits were introduced based on a single chip, known as microprocessors. The main advantage of it is that arithmetic, logic, and control functions can be contained on a single chip, and using a single microprocessor. Computers have become smaller in size and more powerful, compact, and reliable. In this generation, multiprocessing, multiprogramming, time-sharing, distributed operating speed, virtual memory, and real-time networks were used.

Microprocessor

Evolution of computers

- **Fifth Generation of Computers:** The period of the fifth generation is 1980-till date. In the fifth generation, VLSI technology became ULSI (Ultra Large Scale Integration)technology, which has millions of transistors on a single microchip. The computers of this generation are the most efficient, reliable, less expensive, and compact in size. Several concepts of advanced computing are introduced in this generation such as AI (Artificial Intelligence), voice recognition, natural language processing (NLP), optical fiber, etc, also used All high-level languages like C and C++, Java, .Net, etc..

Evolution of computers

- Examples of Fifth Generation Computers include desktops, laptops, tablets, smartphones, and other similar devices.

The end