Summer 2025-CSE332

Computer Organization and Architecture

Textbooks

- Computer Organization and Design MIPS Edition, Fifth Edition: The Hardware/Software Interface, by David A. Patterson and John L. Hennessy, Publisher: Morgan Kaufmann; 5 edition

- Computer Organization and Architecture (10th Edition) by William Stallings, Pearson Publisher: 10 edition, January 22, 2015.

Tests and Evaluation

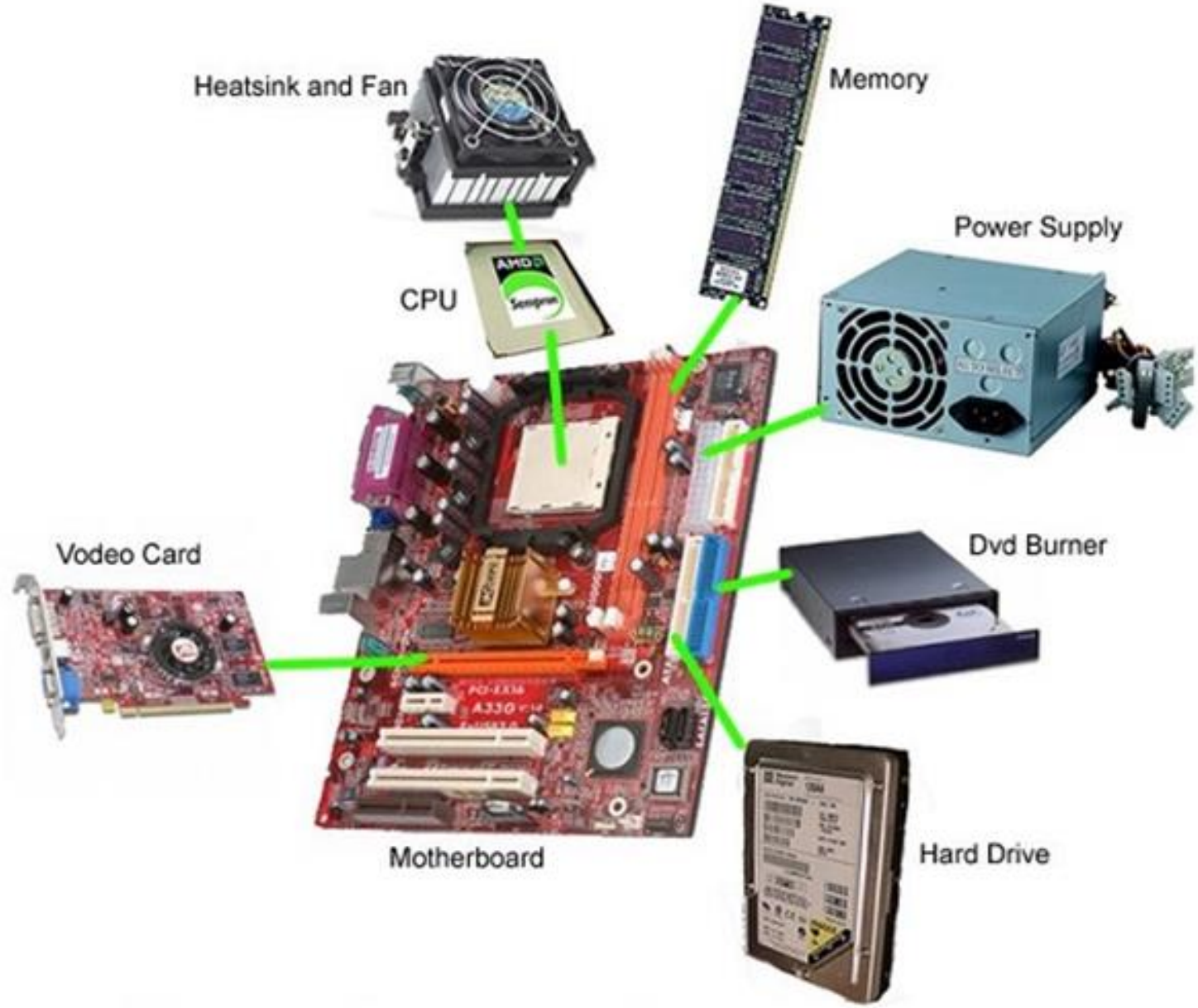
- Attendance and Attention: 5%
- Quiz: 15% (best 3 quizzes will be counted)
- Assignment: 5% (average of all)
- MCQ: 10%
- Mid Term-1: 15%
- Mid Term-2: 15%
- Lab: 10%
- Term Final: 25%
- **Grading:** NSU standard

Students' Comments: (please read carefully)

- gives excess pressure which might put a bad impact and student might lose interest teaches a lot of things that we can't memorize at a time.
- takes classes and quizzes on government holidays. Giving continuous pressures from day one is not good for our mental situation.
- Exam questions are different and difficult than class lectures. exam questions are very difficult a great understanding is required to do good
- too many exams. Almost everyday or every other day we had exam for this course which made it extremely painful and difficult for us to balance good grades in other courses.
- the course focuses too much on slides and because slides are meant to be just the gist of information they were never adequate in creating the proper understanding in my case
- teach so many thing on class but it's not present in the slide or you just add a single picture on the slide and then discuss too many stuff on the class. Now it's almost impossible for someone who miss the class to understand what he or she needs to study.
- took countless class on problem solving at night. It is not possible for everyone to join as we have other courses too.
- always gives slides or info before the exam night; it became quite hectic to get all the info; it's really too much to finish before the exam.
- Worst. Not student friendly. If we have exam in tomorrow. His syllebus will be till today. That's very disappointing for us.
- should state properly what he is lecturing about and what to do with it. He should make his lectures more lively. He teaches continuously and most of the students are clueless what they are doing and why they are doing. If the lectures were a bit more cheerful and the problems were stated perfectly this would have been a great course.
- The teaching process is hectic. Unnecessary lengthy lectures. No specific syllabus before the exam. Students keep wondering if should they study this topic or not. This made the course very difficult to understand.
- too much questions slides confused us what to read or not.
- If sir were a character in the Marvel Cinematic Universe, his role might be- Night Owl: Runtime of a Lifetime. Takes too many classes at night, making it difficult for students with tuition to attend.
- Sometime we don't understand what he is actually teaching.
- He always takes an extra 5 to 10 minutes to finish class
- Our teacher starts the class before the scheduled time and finishes 8-10 minutes late, which causes us to be late for our next class. Additionally, his teaching style is not engaging, as he gives nonstop lectures, making it difficult for us to concentrate.
- Bad /worst course

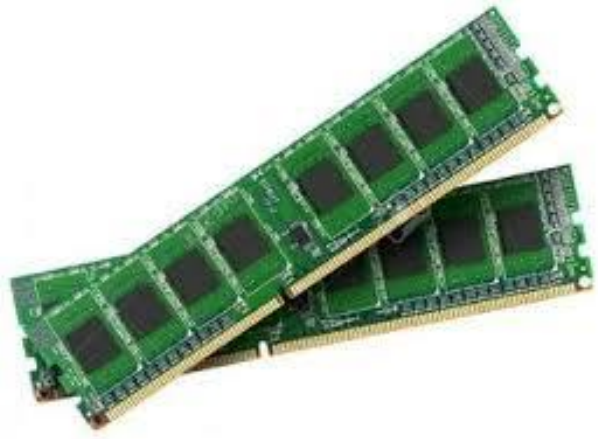
What we will discuss…

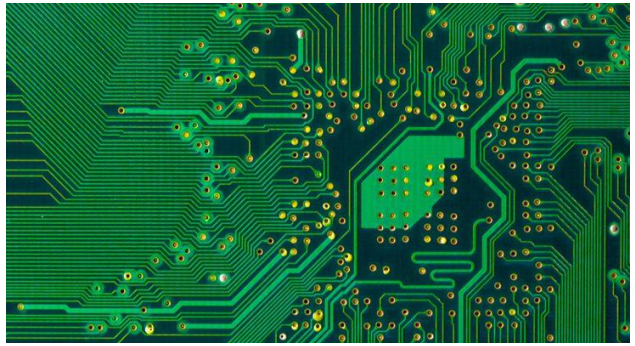
- *Functional units*
- *Block and functional diagram*
- *Computer Architecture*
- *Computer Organization*

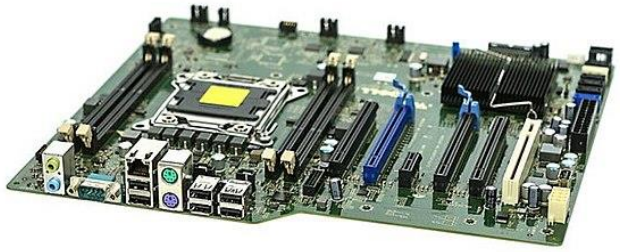

Heatsink and Fan
Memory
Power Supply
CPU
Vodeo Card
Dvd Burner
Motherboard
Hard Drive

Technology of main functional unit:

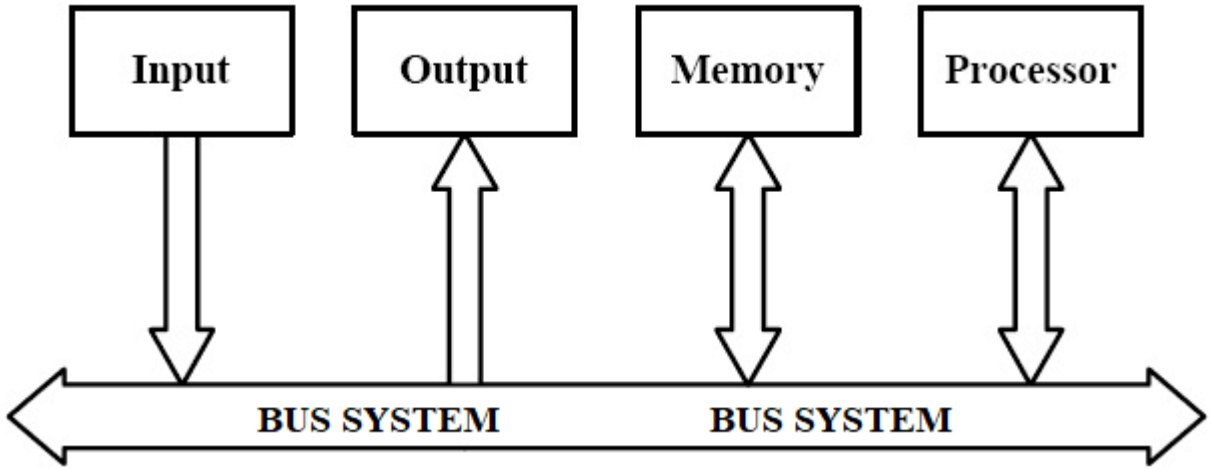
- Semiconductor Technology
- Transistor as basic element in CPU Design
- Signal level: 2 levels (On-OFF)
- High – Low (Binary- Digital)
- Arithmetic Operations in Binary
- Data Representation in Binary
- Program Representation in Binary
- Digital computer

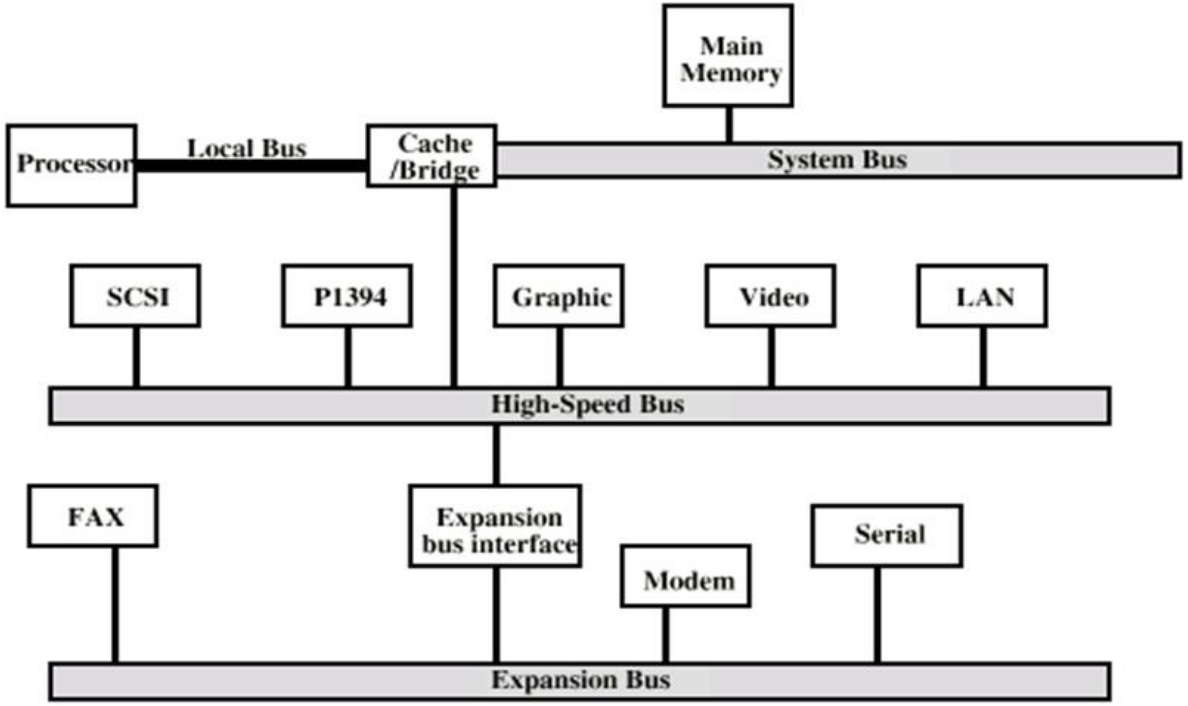


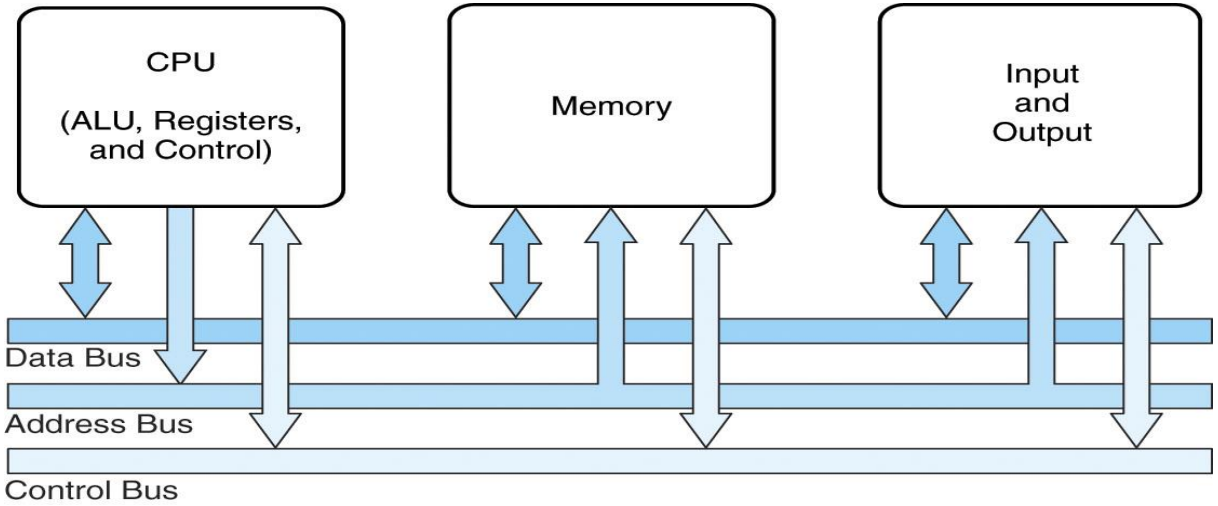




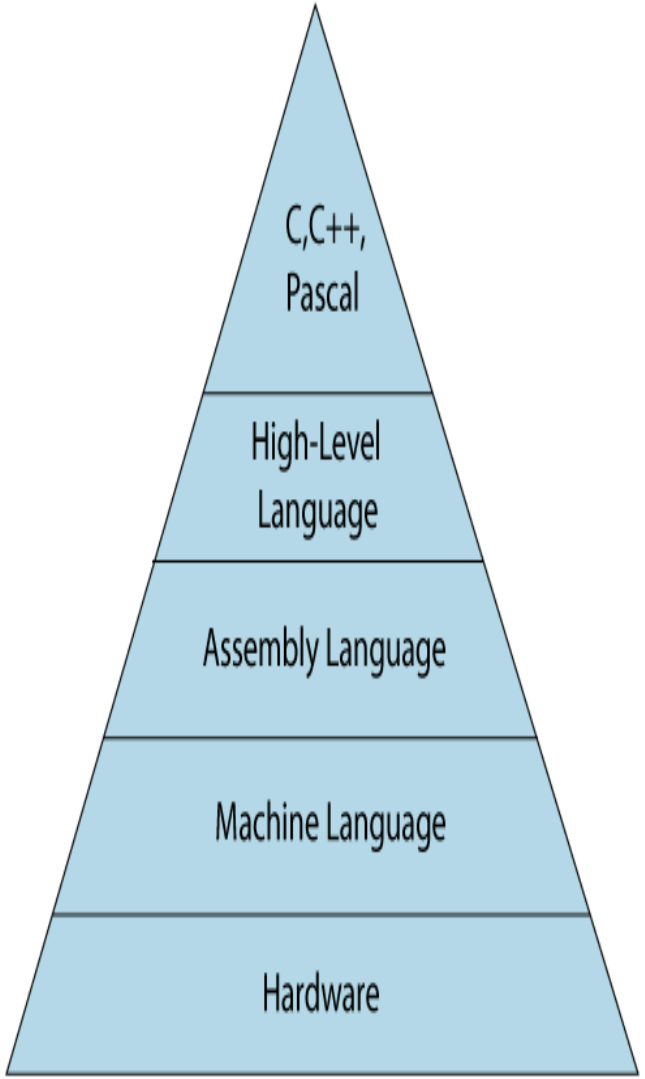
Representing computer Architecture:
schematic diagram: easy to explain interconnection and operation







Levels of Program code



High-level program

```
class Triangle {
  ...
  float surface()
    return b*h/2;
  }
```

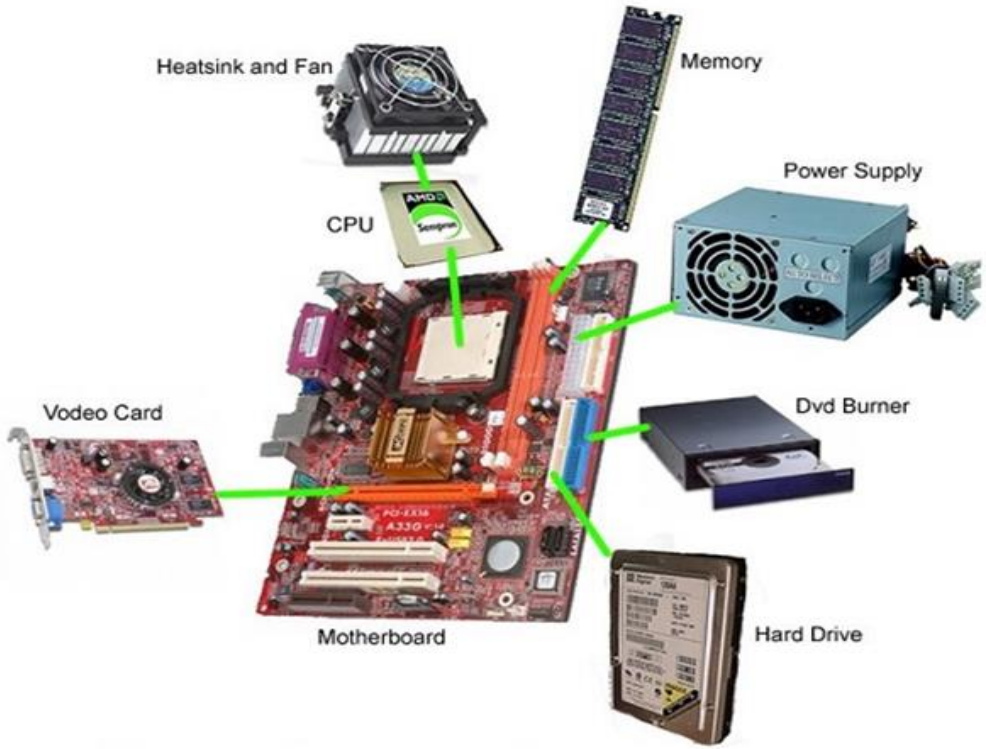
Low-level program

```
LOAD r1,b
LOAD r2,h
MUL r1,r2
DIV r1,#2
RET
```

Machine code

```
000100100100010 1
0010010011101100
10101101001...
```

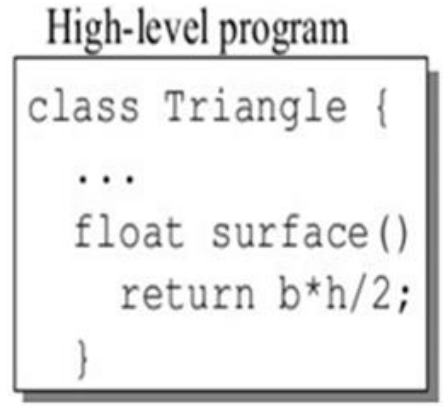
- We write programs in high-level languages without thinking about on what machine our program will run.
- Electronics of the CPU and computer recognize binary representations of instructions. That means each CPU has its own machine language that it understands.



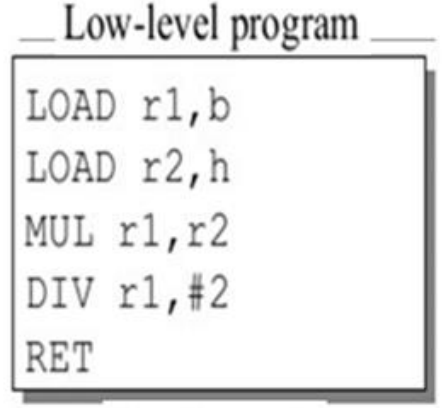
- Programs are typically written in higher-level languages and then translated into machine language (executable code).

- A **compiler** is a program that translates code written in one language into another language.

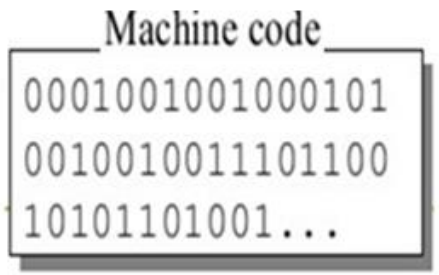
- An **interpreter** translates the instructions one line at a time into something that can be executed by the computer's hardware.



COMPILER

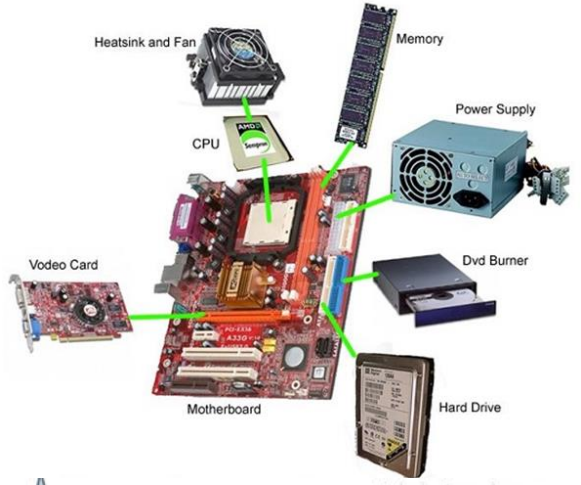


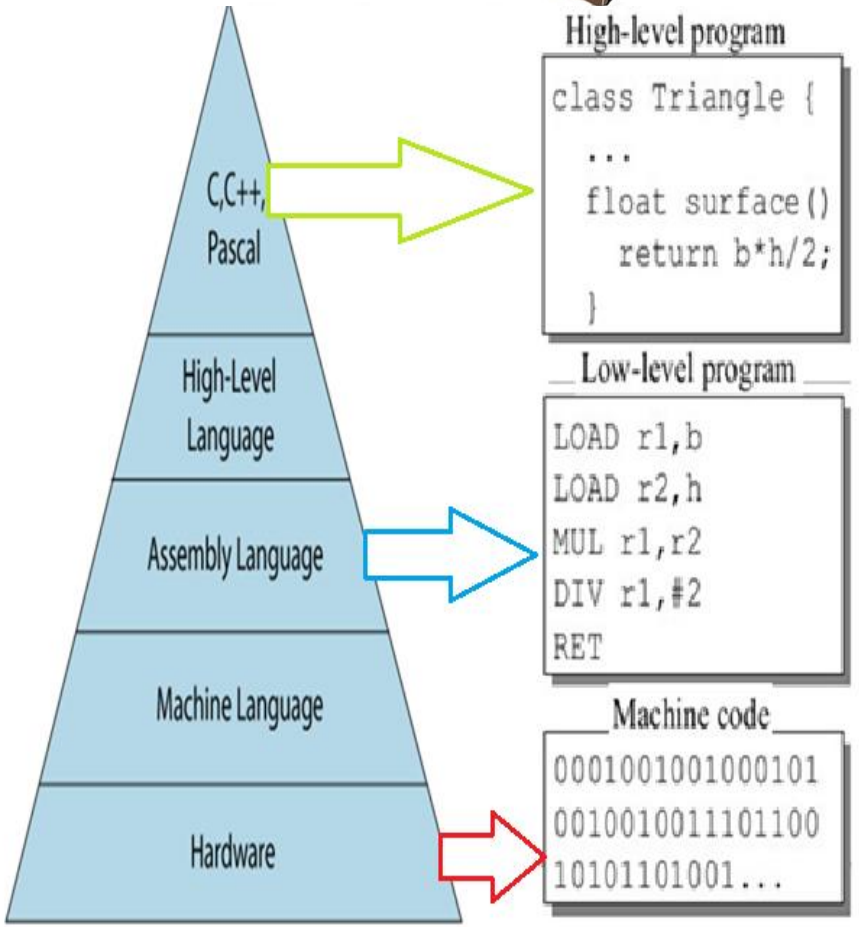
INTERPRETER



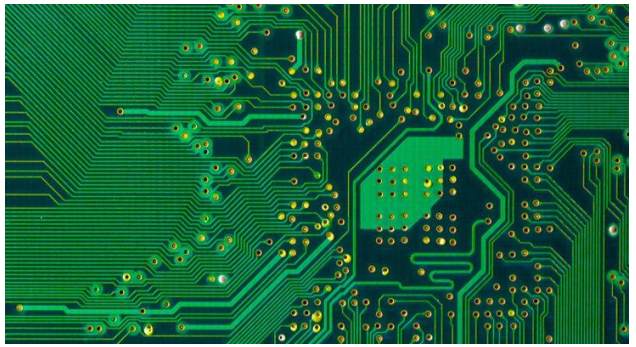
Computer, software, program and Instruction

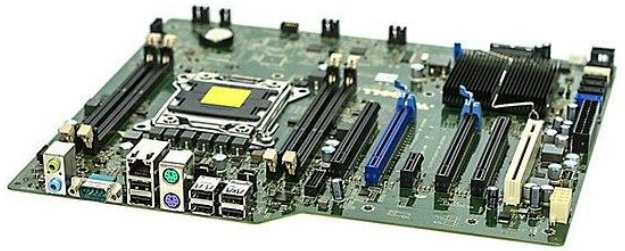
- CPU and Hardware is designed to perform a fixed number of very simple and elementary operations (arithmetic/logical/data transfer operation), such as addition of two number, compare two numbers, shift bits left or right, complement etc.
- To do a particular simple and elementary operation, a command is formed in binary and given to CPU and hardware, called Instruction.
- Command/instruction in binary is called Machine code. A machine code specifies operation, data or whereabouts data and where to store result.
- A **program** is a sequence of instructions arranged to solve a mathematical, scientific computation of data processing of different types.
- **Software** is a set of computer programs





Digital Computers





- Program
- Data
- Represented, stored, processed, transferred in Binary:

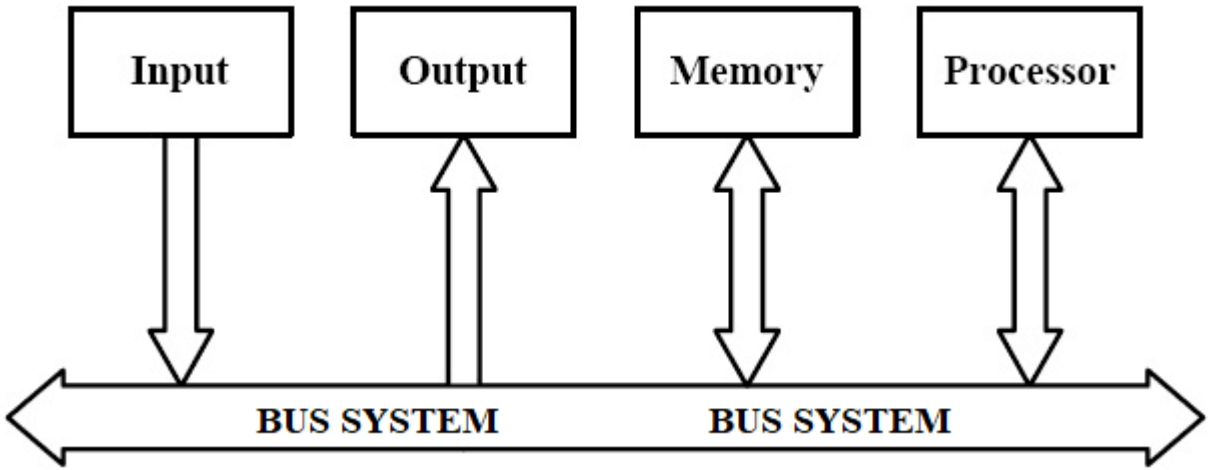
0 1 0 1 0 0 1 1 (logical)

+3V / +5V

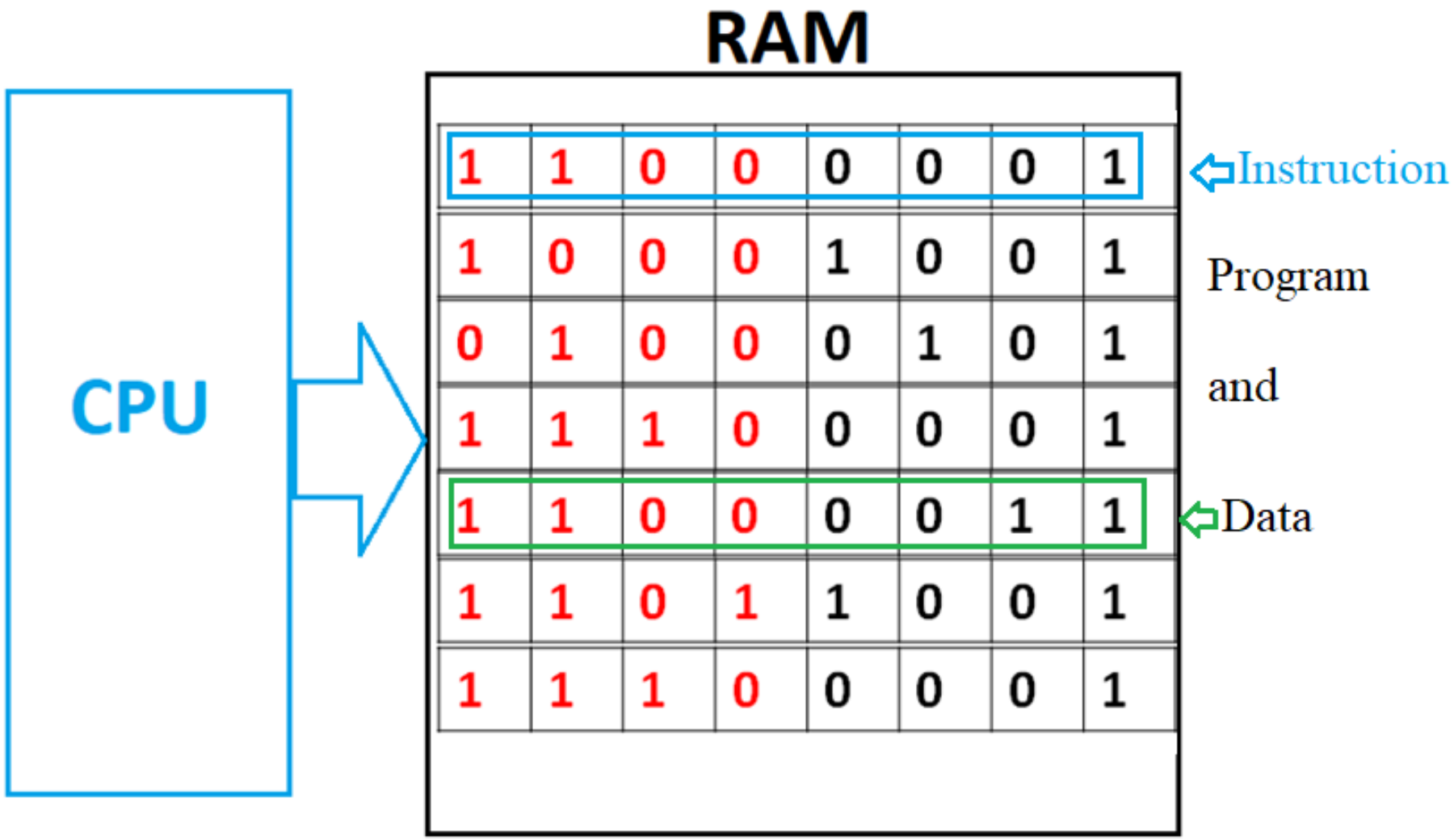
- 3V / -5V /0V

On / Off switches (Transistors)

Charges (+Q or No Charges: in RAM)



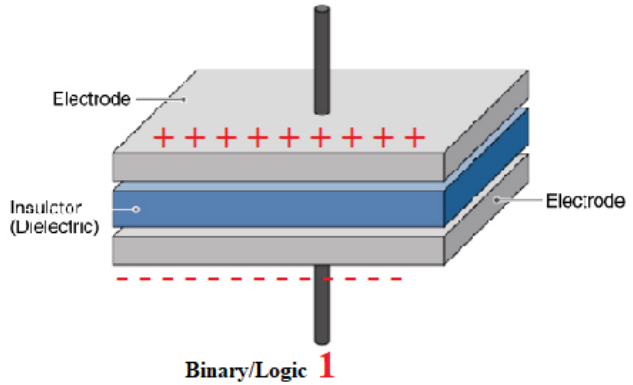
Program and Data are temporarily stored in RAM as binary strings of 1's and 0's

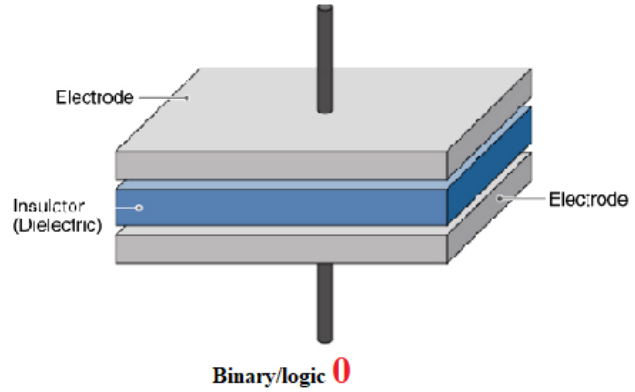


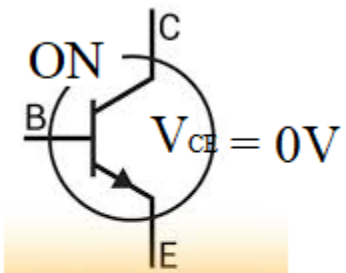
RAM Technology

RAM

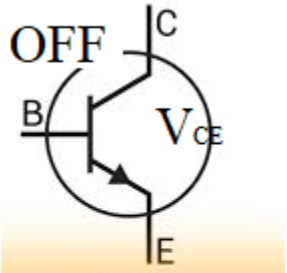
1	1	0	0	0	0	0	1	⇦Instruction
1	0	0	0	1	0	0	1	Program
0	1	0	0	0	1	0	1	and
1	1	1	0	0	0	0	1	
1	1	0	0	0	0	1	1	⇦Data
1	1	0	1	1	0	0	1	
1	1	1	0	0	0	0	1	







Binary/logic 0



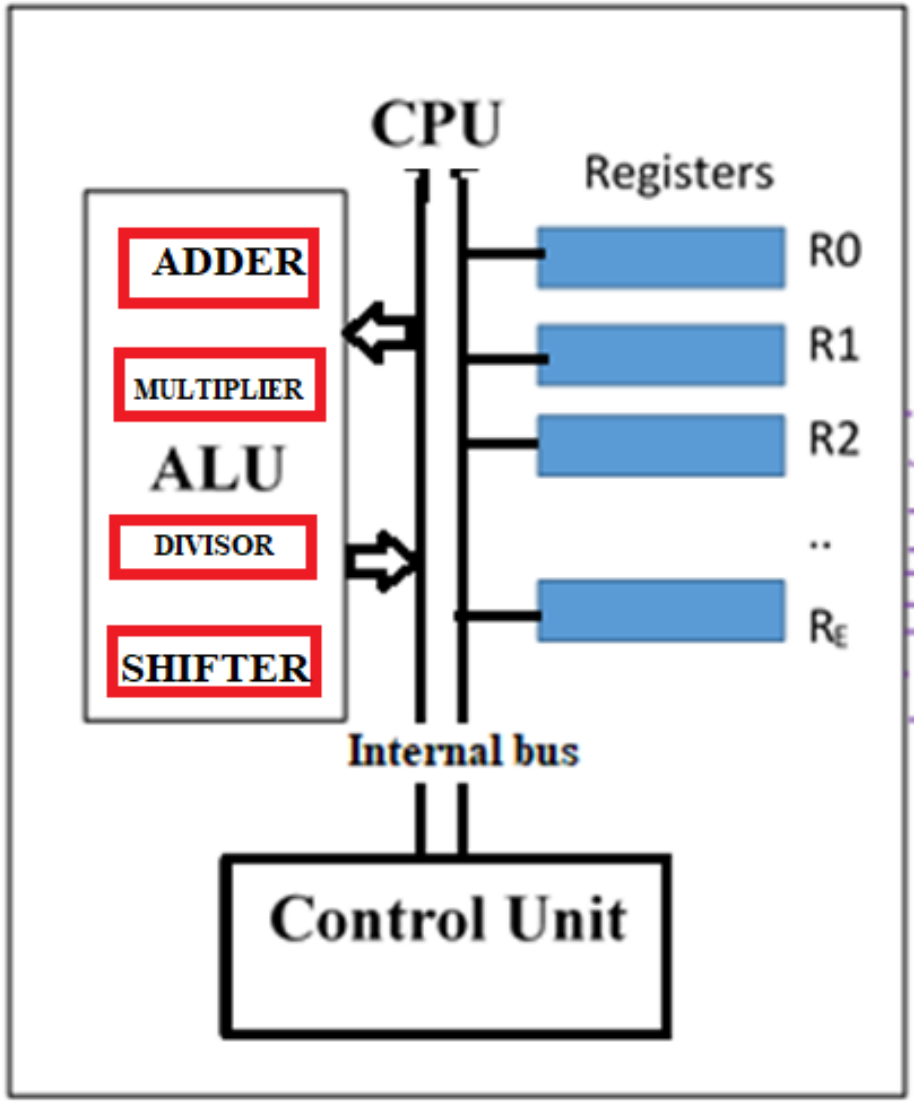
Binary/logic 1

CPU: Central Processing Unit

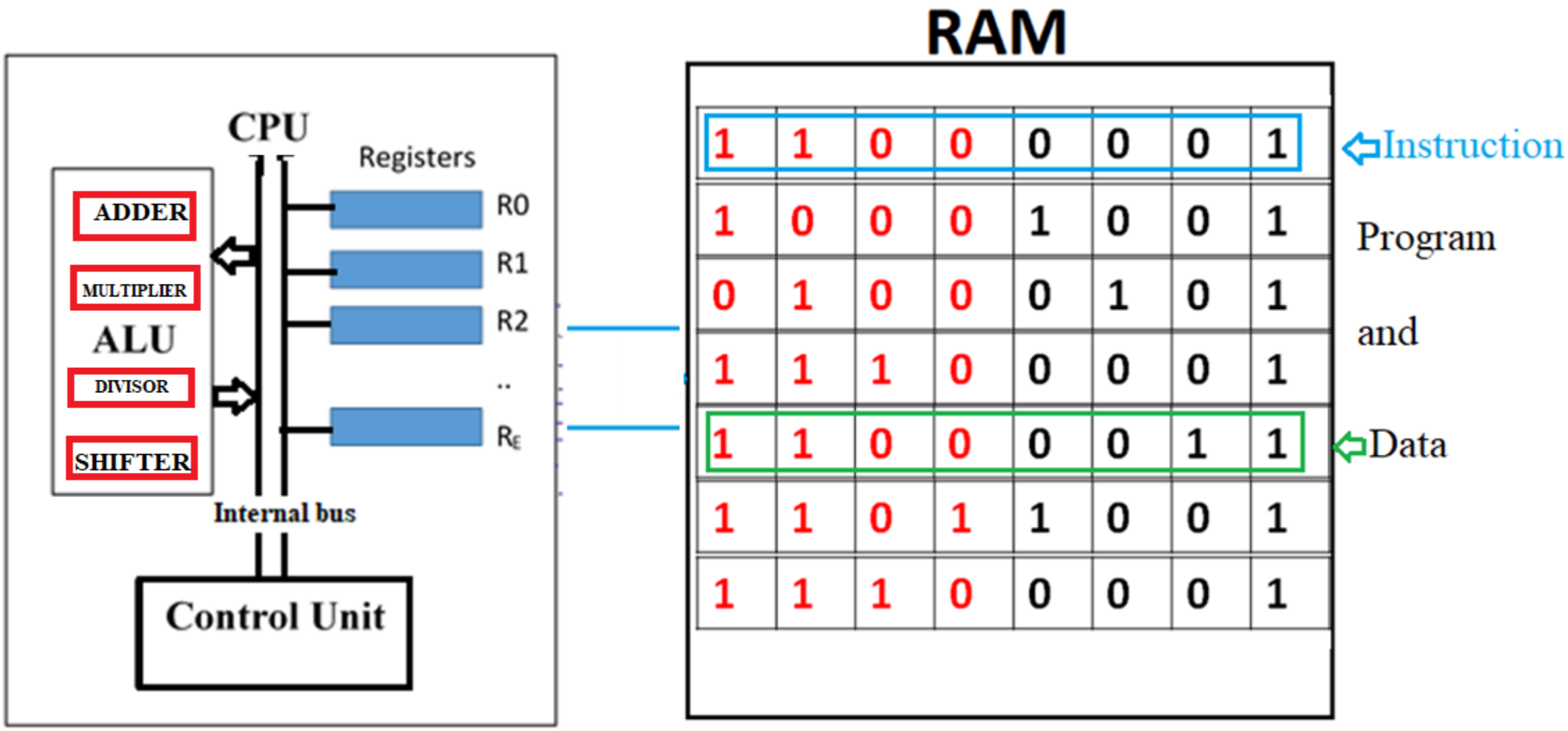
- **Registers** – high speed electronic storage to hold binary data currently being processed or to be processed next

- **Arithmetic Logic Unit (ALU)** – Basic Arithmetic and Logical operations: only two/one data at a time. Addition, subtraction, AND, OR, Rotate bits etc

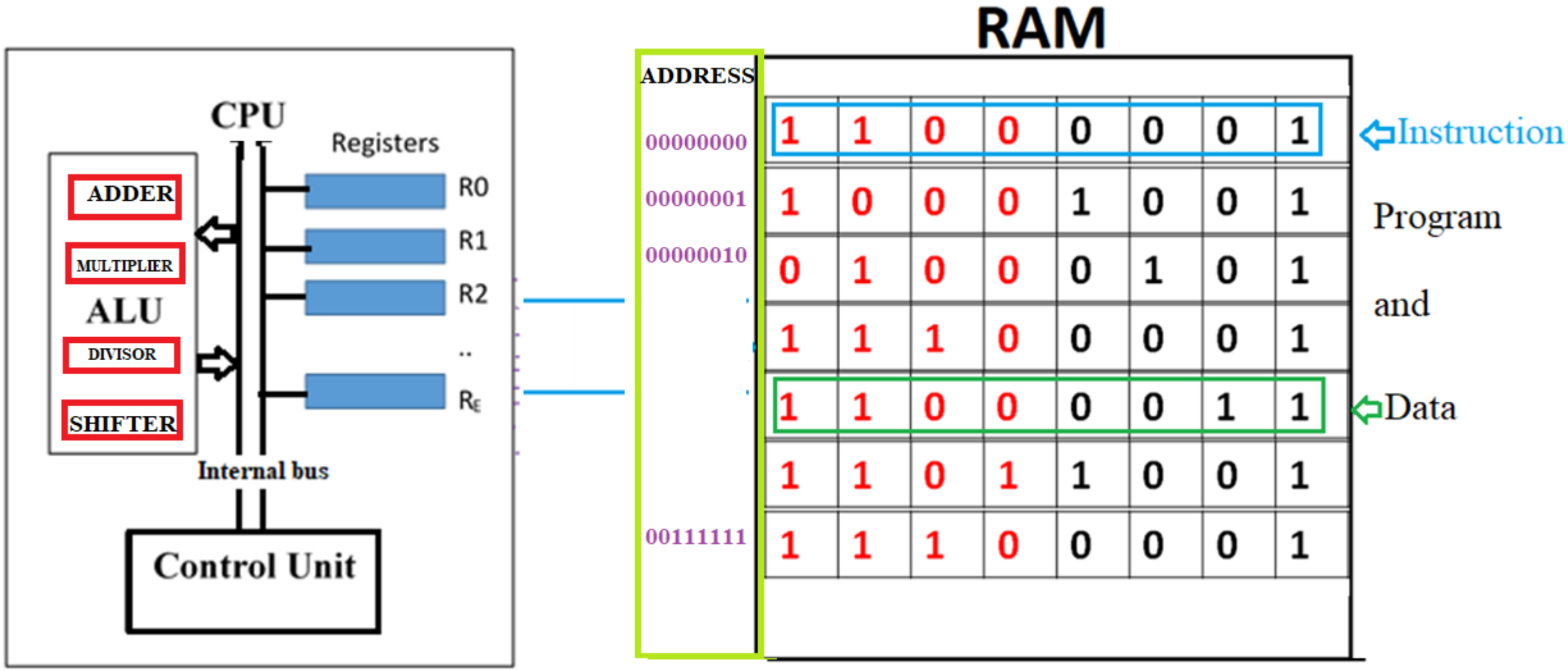
- **Control Unit** – This contains the logic circuitry controlling the other parts of the CPU.



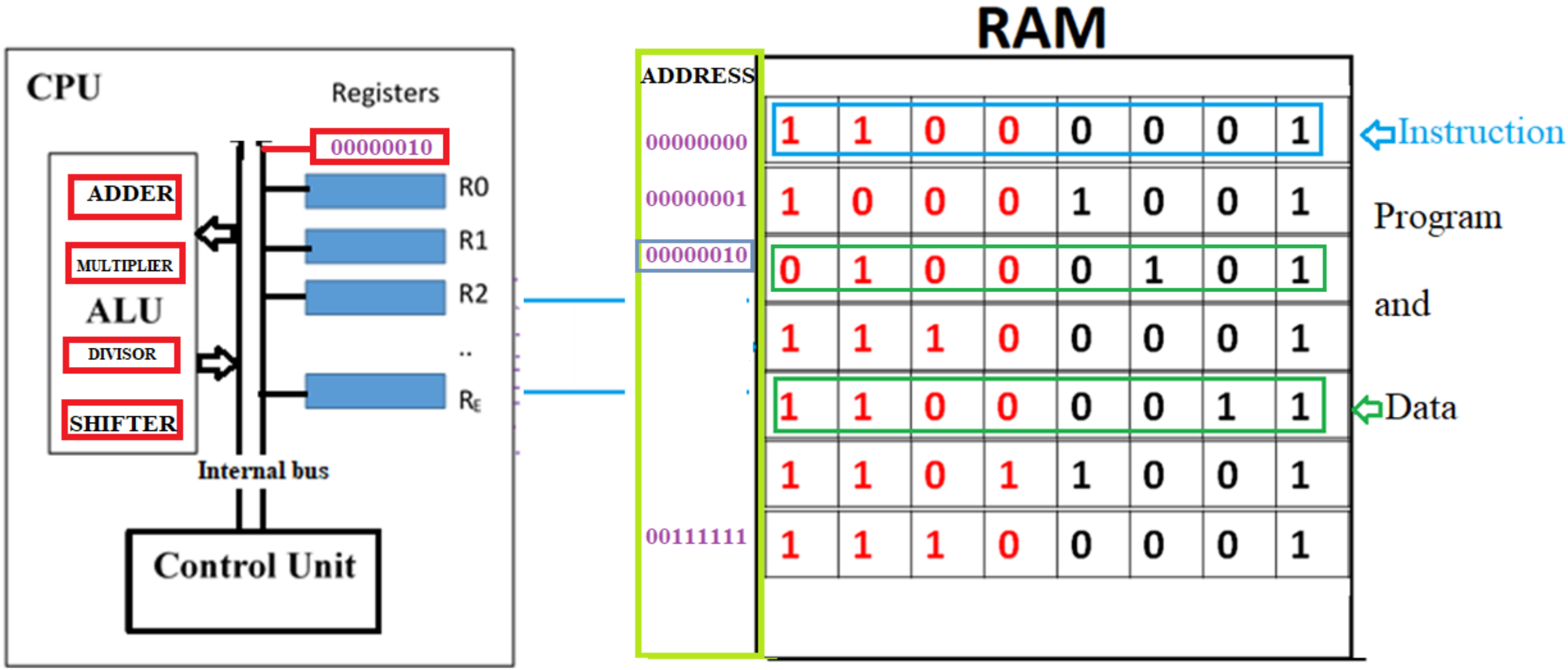
Basic Computer Function



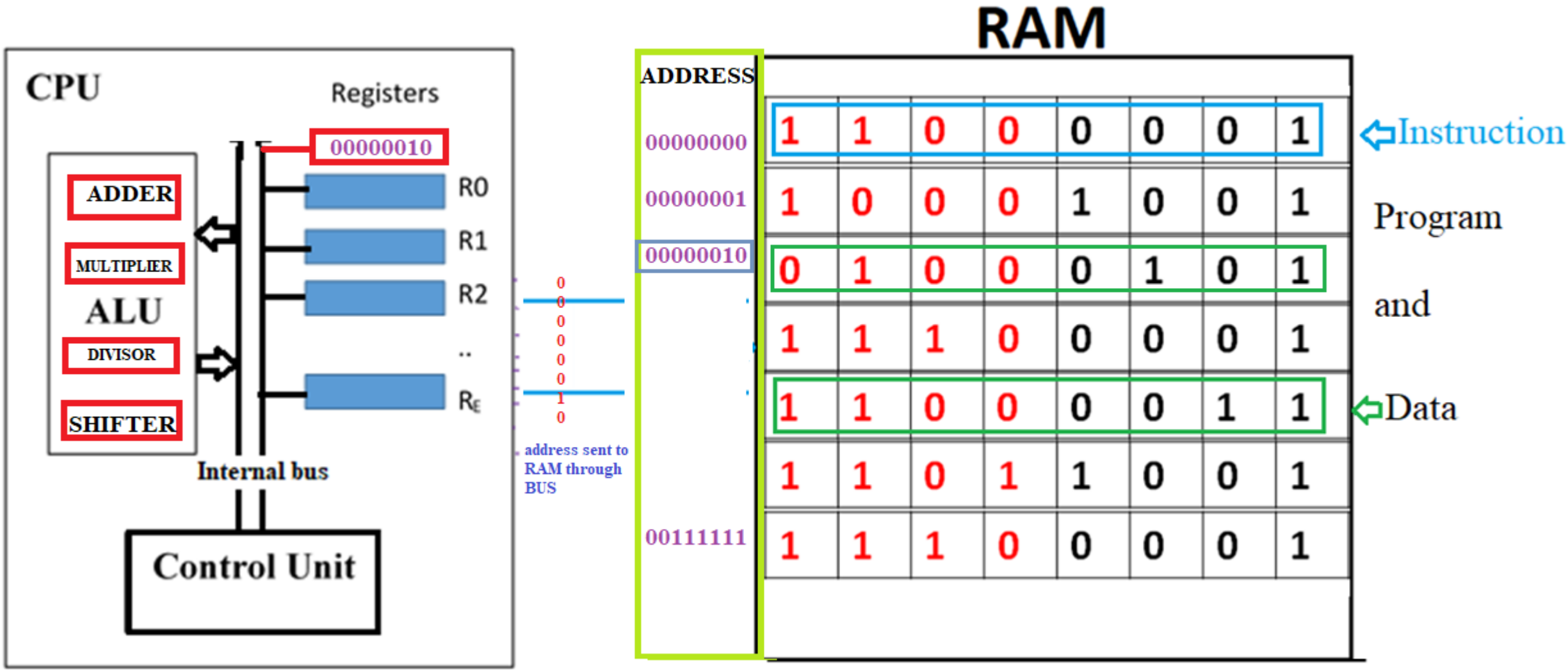
Basic Computer Function



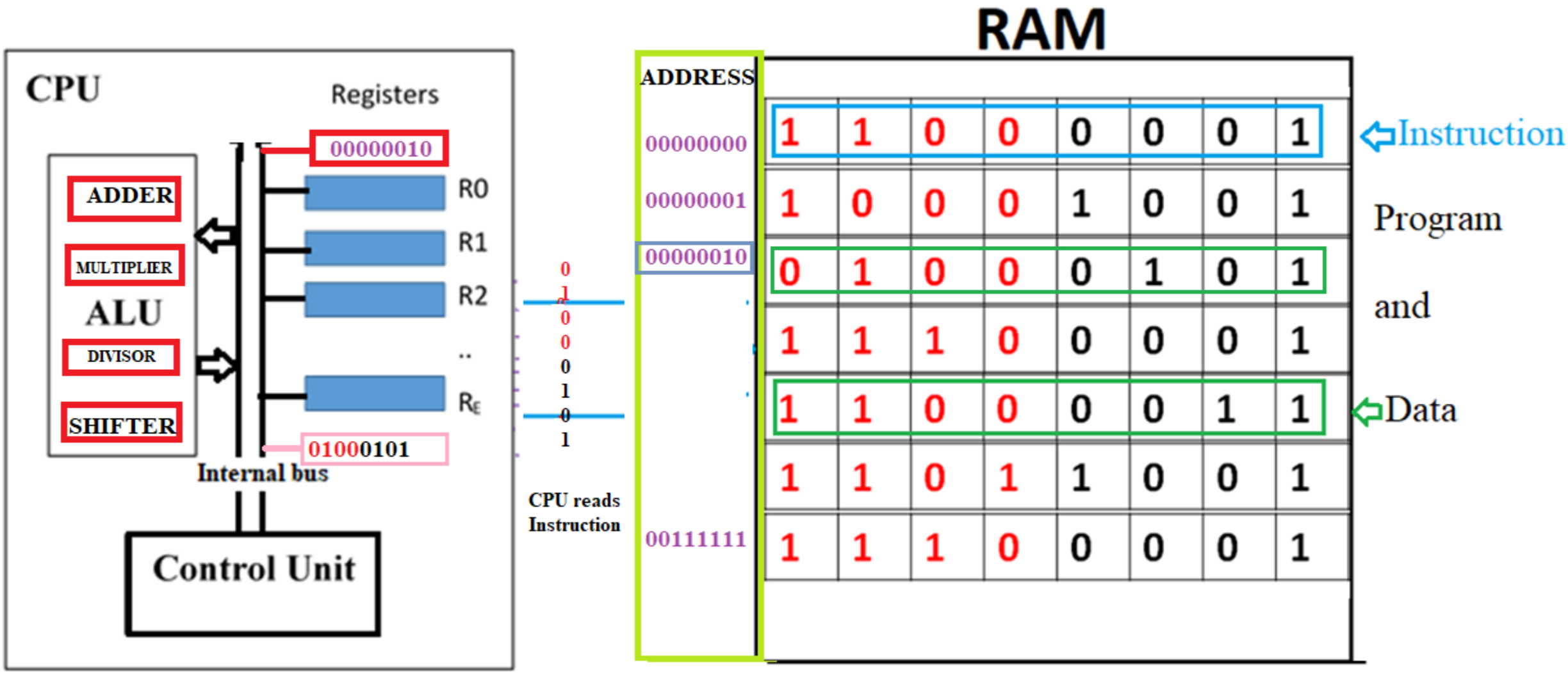
CPU: points instruction in RAM



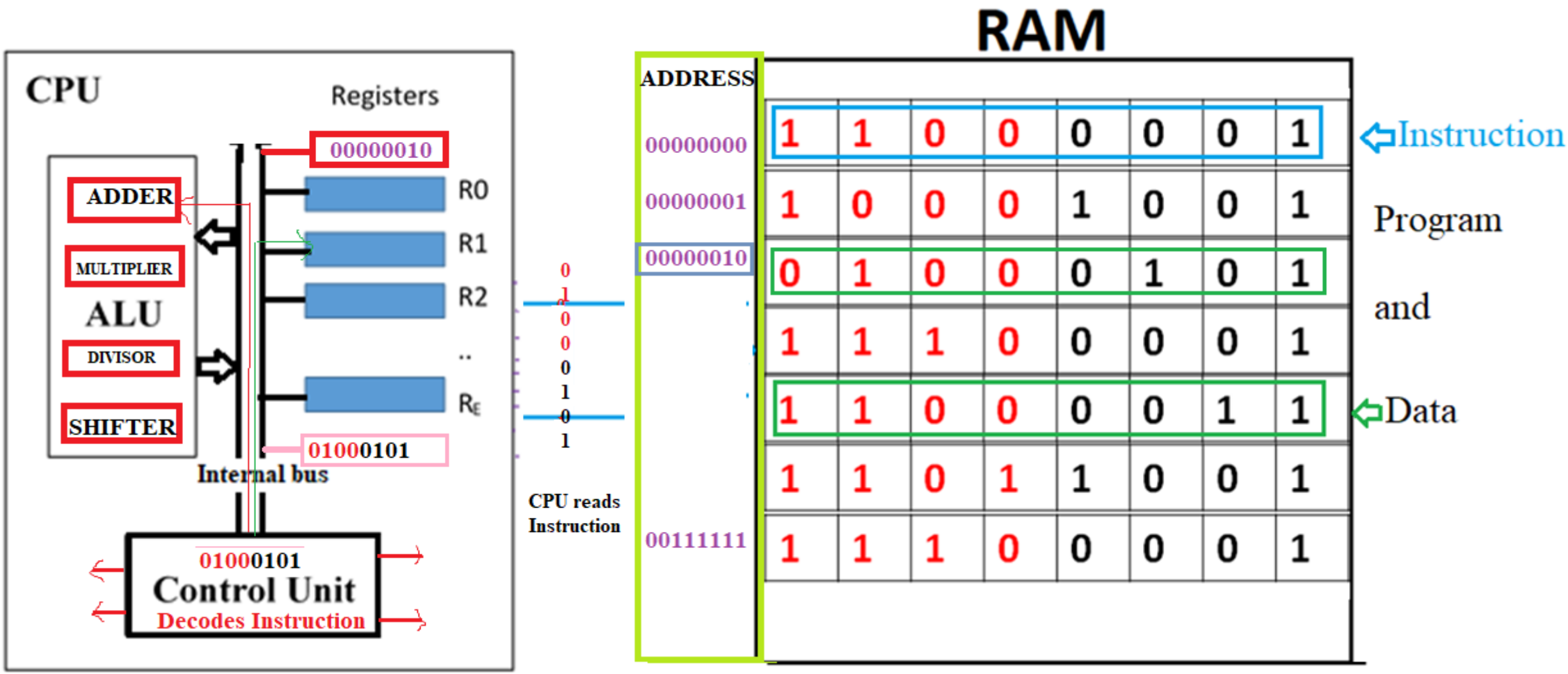
CPU sends address of instruction to RAM



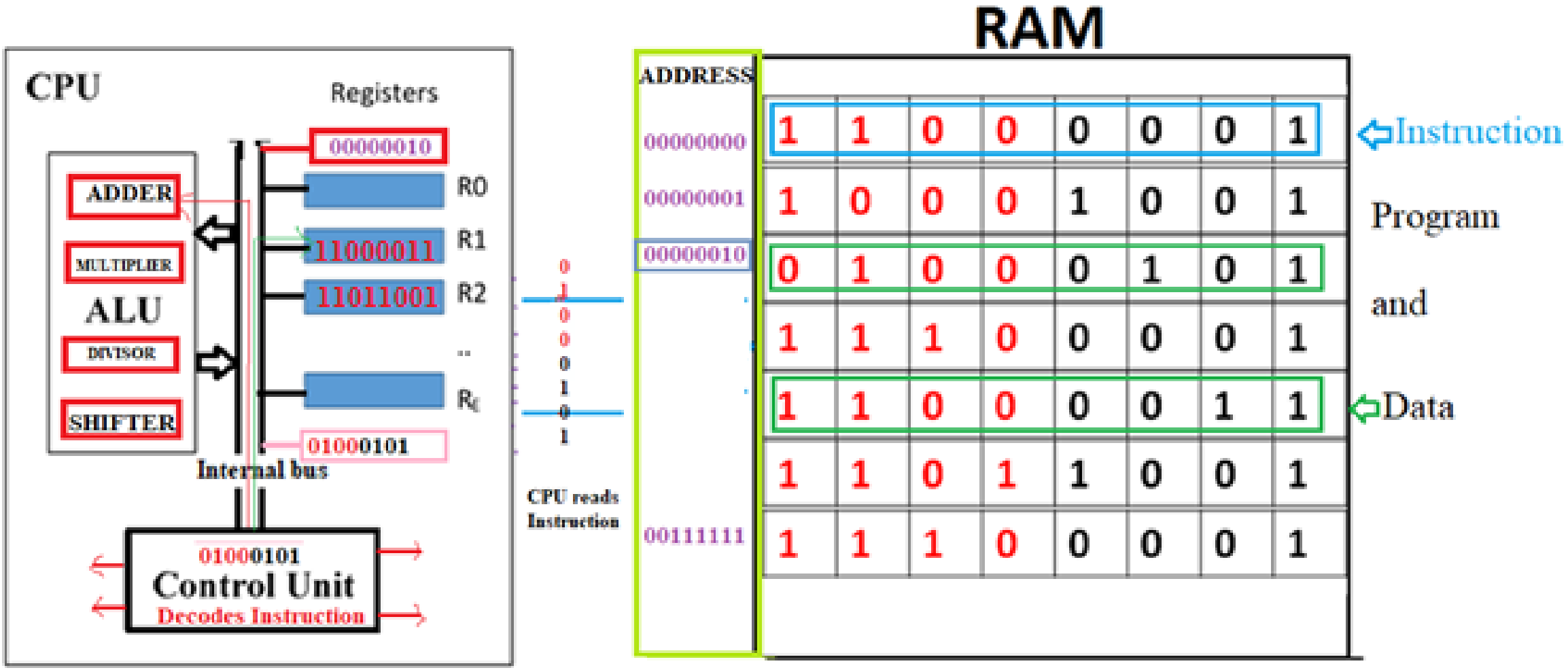
CPU reads Instruction



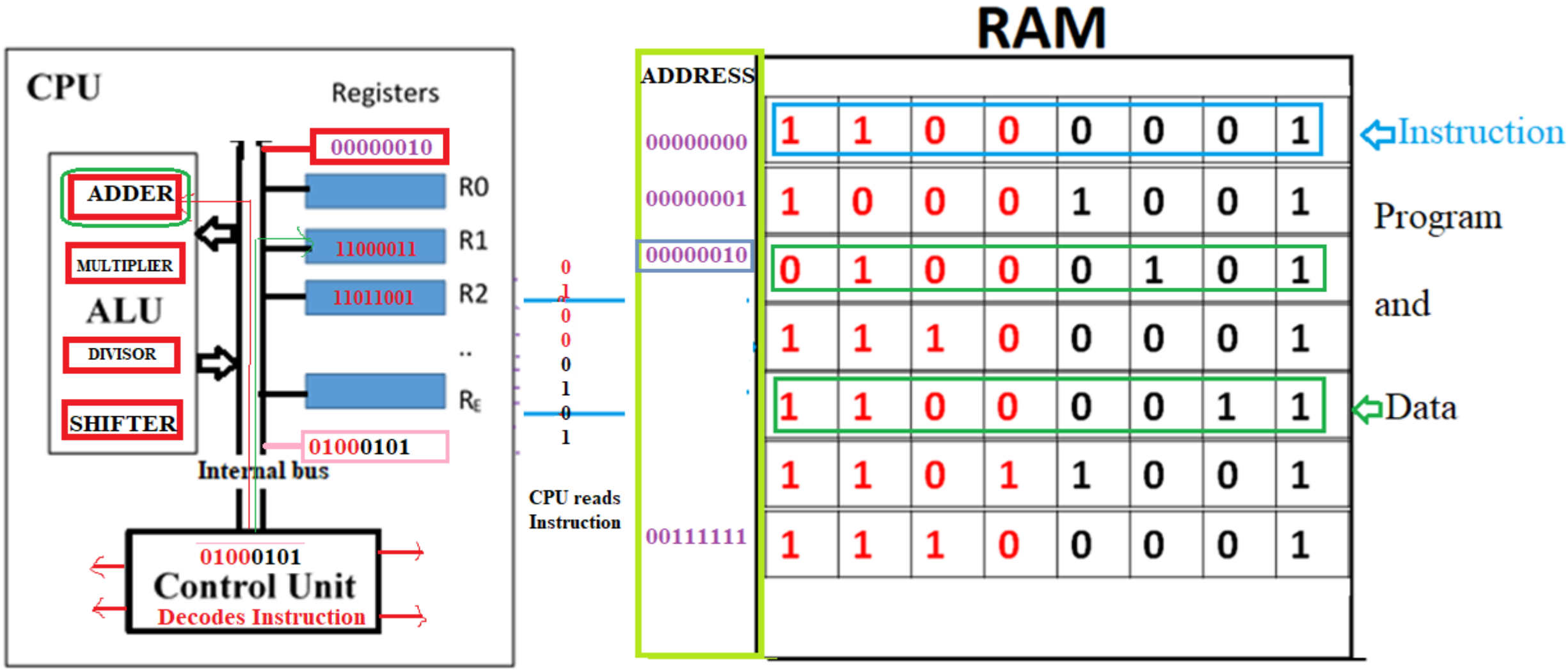
CPU decodes Instruction



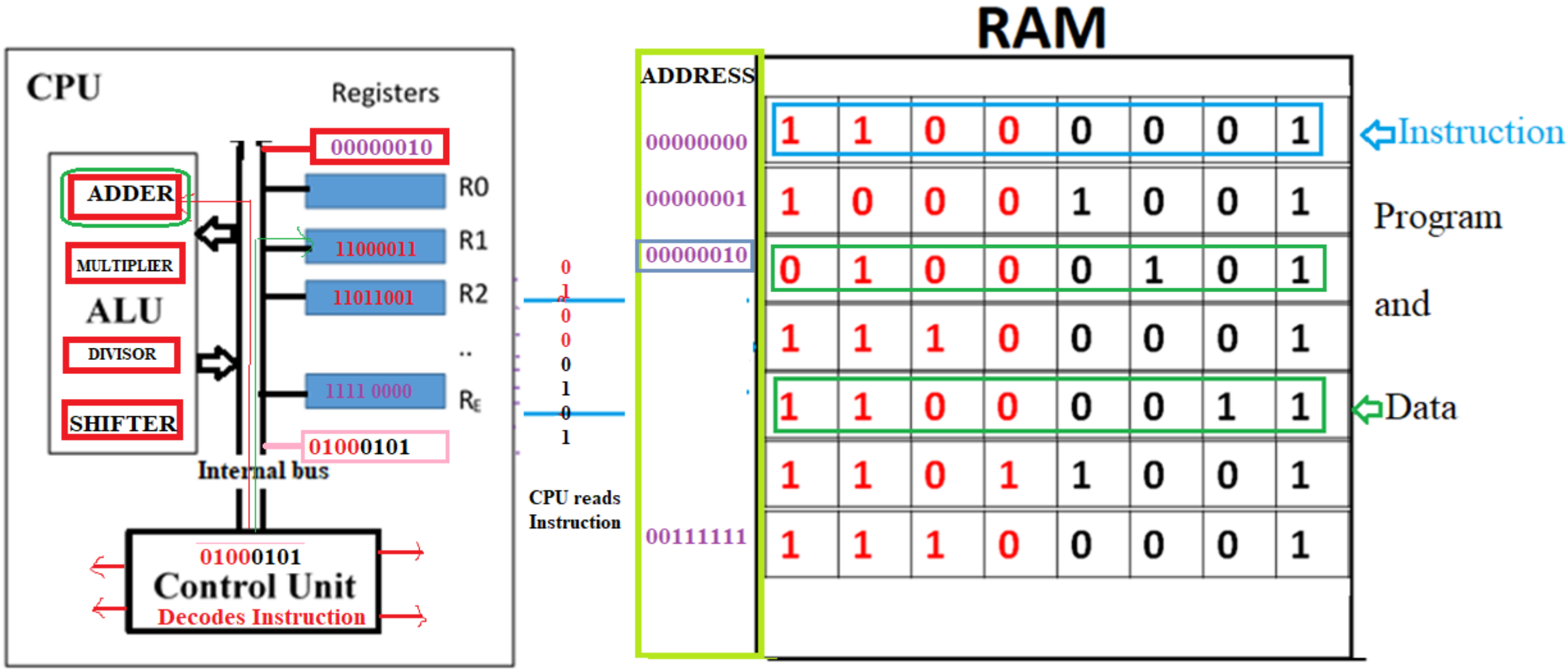
CPU reads data

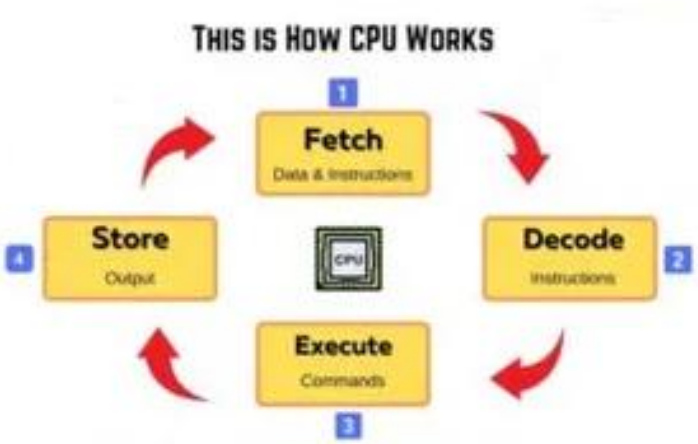


CPU performs Arithmetic operation



CPU stores result



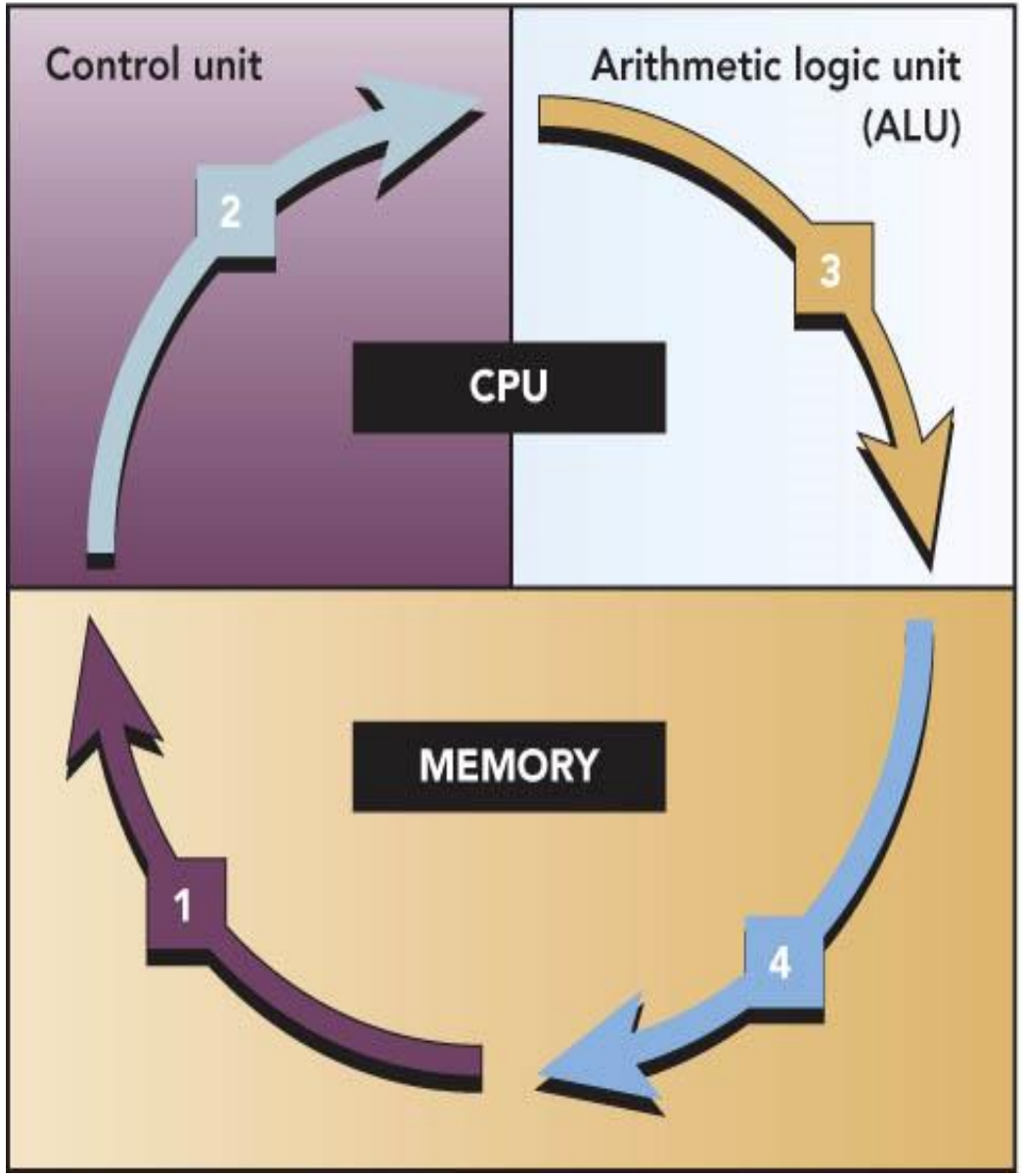


FETCH
instructions from memory

DECODE
into binary instructions

EXECUTE
action and move to next step

STORE
write output to memory



INSTRUCTION CYCLE

1 Fetch

Retrieves the next program instruction from memory

2 Decode

Determines what the program is telling the computer to do

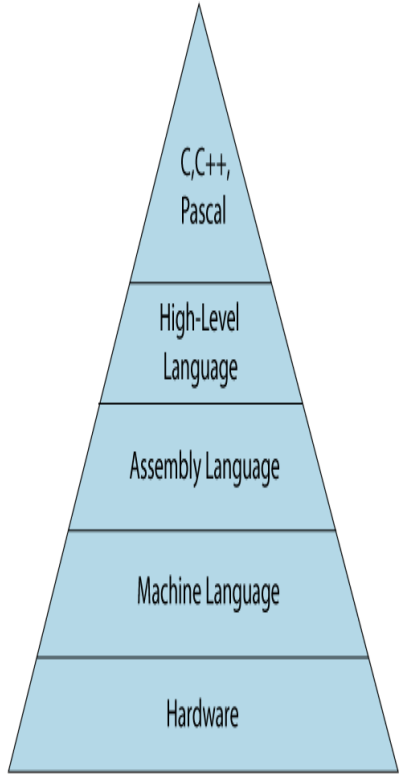
EXECUTION CYCLE

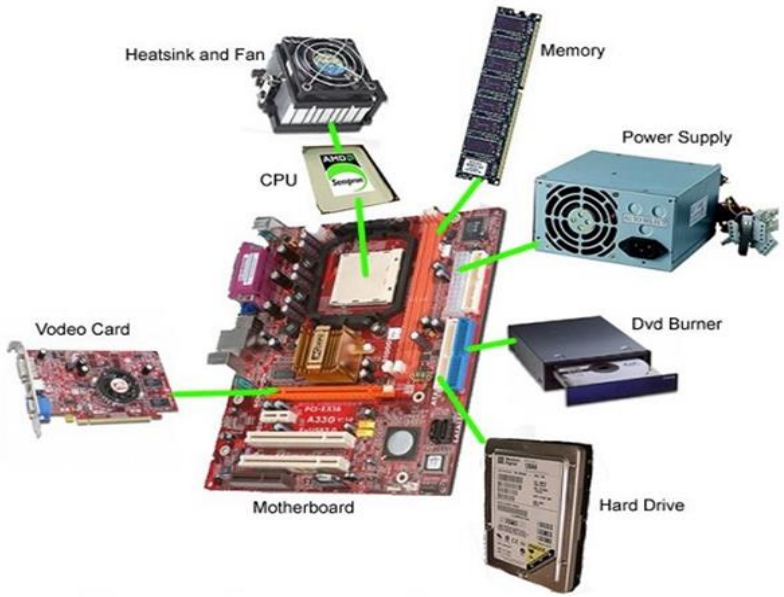
3 Execute

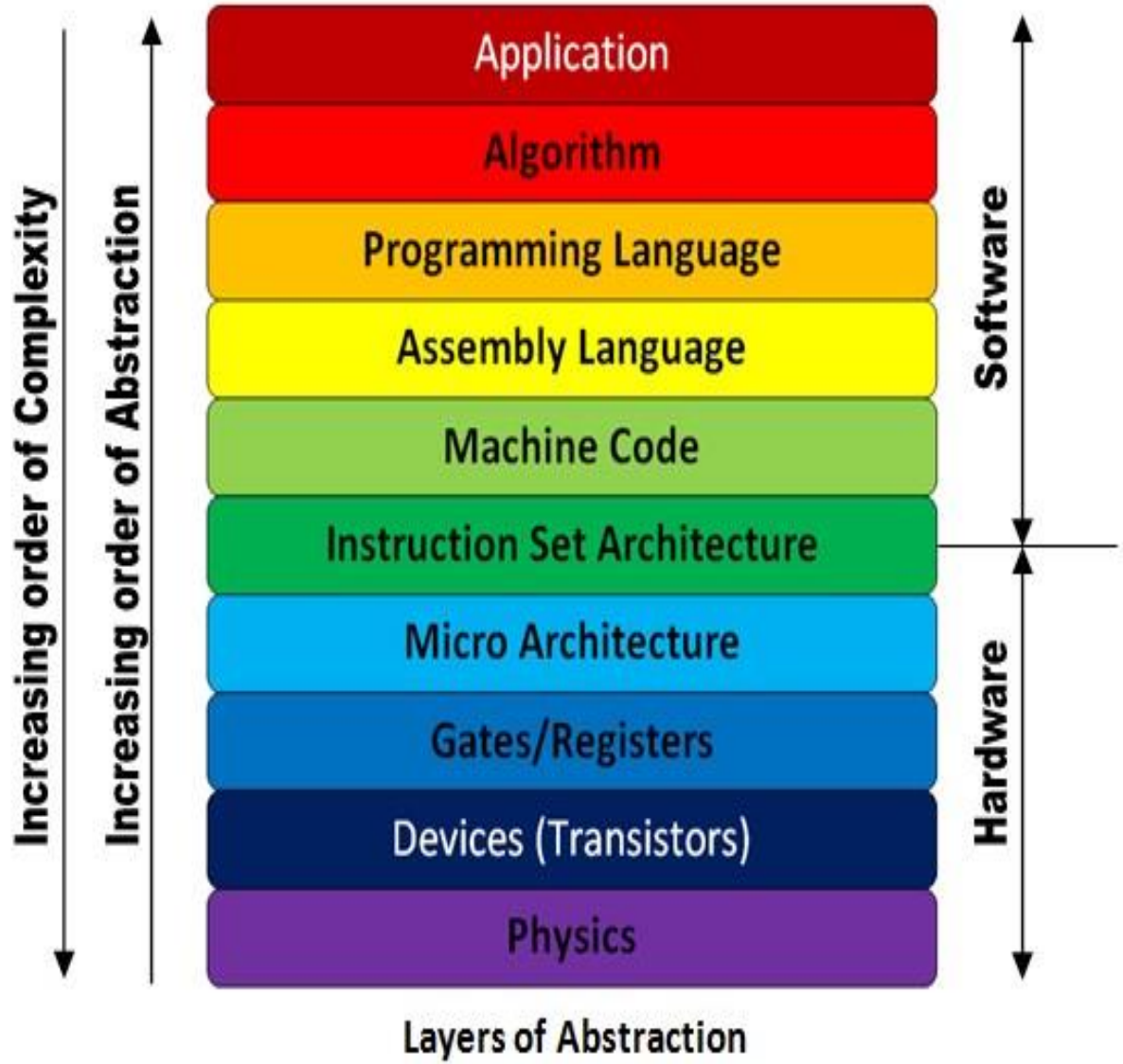
Performs the requested instruction

4 Store

Stores the results to an internal register (a temporary storage location) or to memory

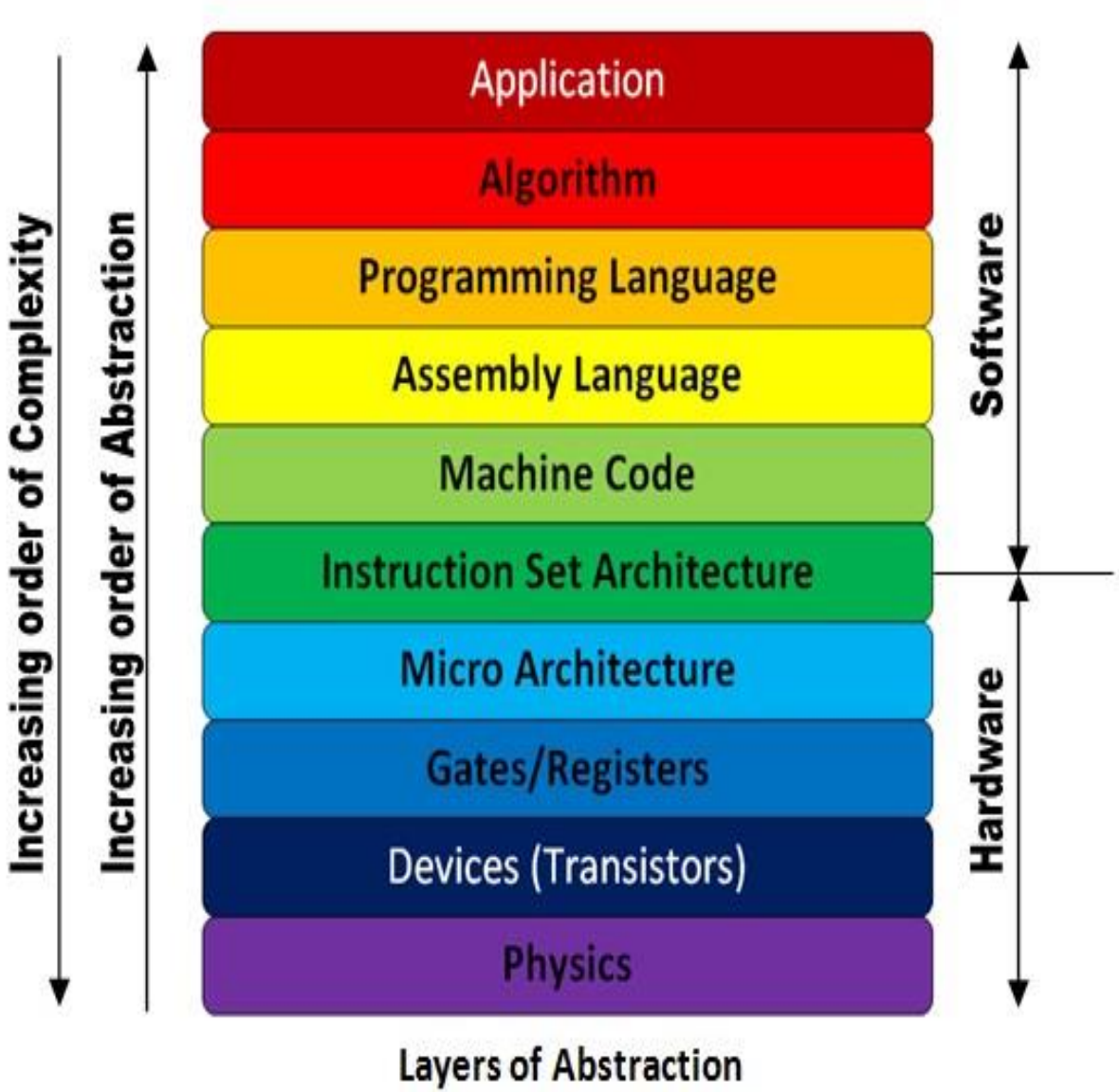

C,C++,
Pascal
High-Level
Language
Assembly Language
Machine Language
Hardware


Heatsink and Fan
Memory
Power Supply
CPU
Vodeo Card
Dvd Burner
Motherboard
Hard Drive


Increasing order of Complexity
Increasing order of Abstraction
Application
Algorithm
Programming Language
Assembly Language
Machine Code
Instruction Set Architecture
Micro Architecture
Gates/Registers
Devices (Transistors)
Physics
Software
Hardware
Layers of Abstraction

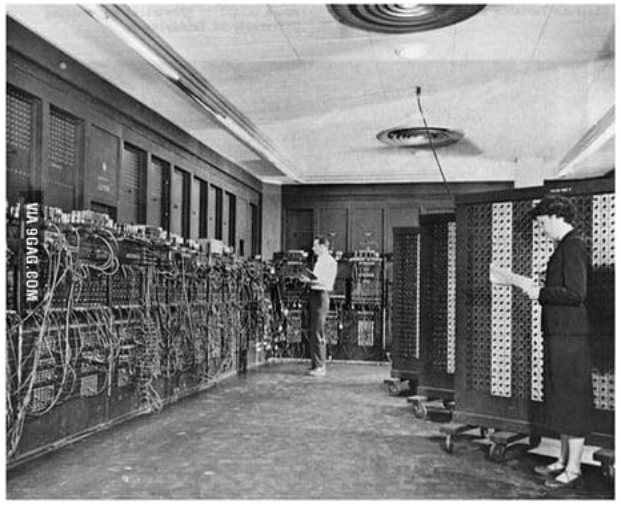
Representing computer Architecture: Layers of abstraction

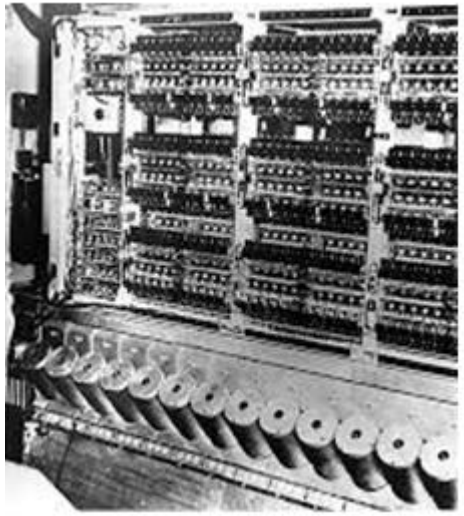
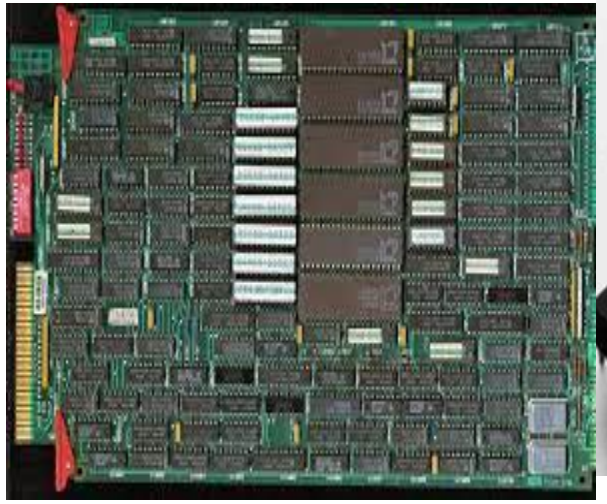
- Use layers of abstraction to hide details of the computer design.
- We can work in any layer, not needing to know how the lower layers work or how the current layer fits into the larger system.
  - transistors
  - gates
  - circuits (adders, multiplexors …)
  - central processing units (ALU, registers …)
  - computer
- A component at a higher abstraction layer uses components from a lower abstraction layer without having to know the details of how it is built.
- It only needs to know what it does.



CPU / Microprocessor / Processor

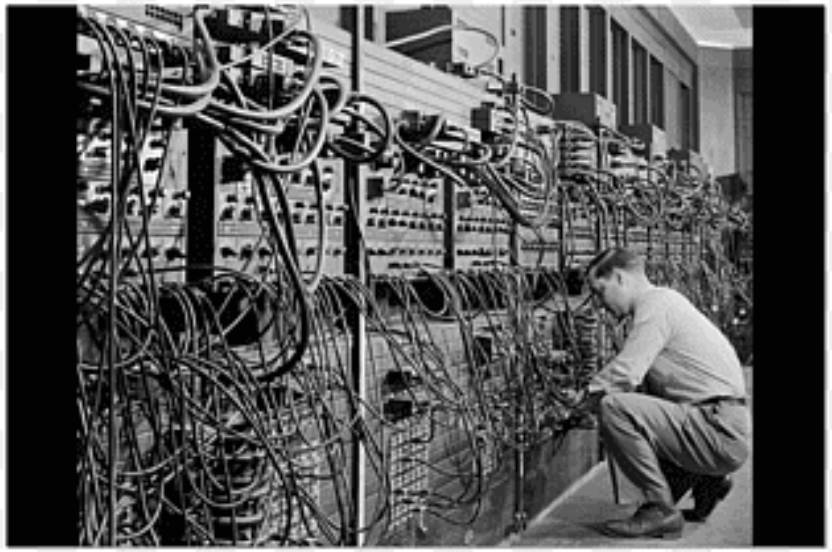
- Back in the old days, a CPU consisted of many different separate electronic circuits or chips connected together. So for example, the CPU registers might be on one (or more) chips, the Arithmetic Logic Unit might be another chip.
- Microprocessor is an Integrated Circuit (IC) that contains all functional units of a CPU. Nowadays all CPUs are Microprocessors.




ENIAC (Electronic Numerical Integrator and Computer)

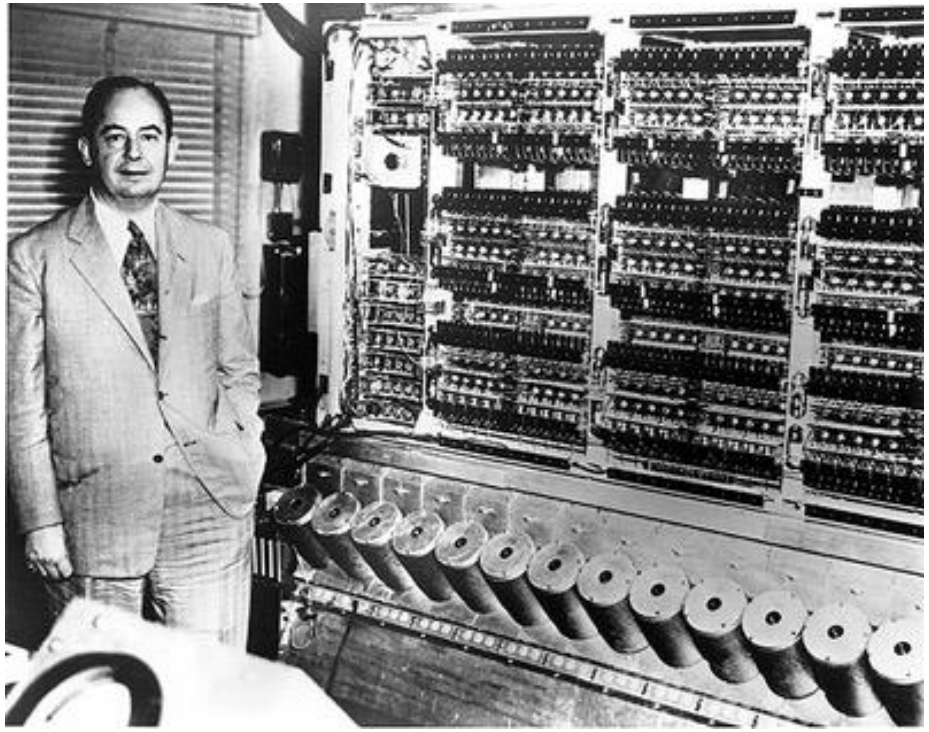
- **F**irst programmable, electronic, general-purpose digital computer.
- Task of taking a problem and feed it onto the machine was complex, and usually took weeks.
- After the program was figured out on paper, the process of getting the program into ENIAC by manipulating its switches and cables could take days.
- Although ENIAC was designed and primarily used to calculate artillery firing tables for the United States Army's Ballistic Research Laboratory (which later became a part of the Army Research Laboratory),its first program was a study of the feasibility of the thermonuclear weapon.



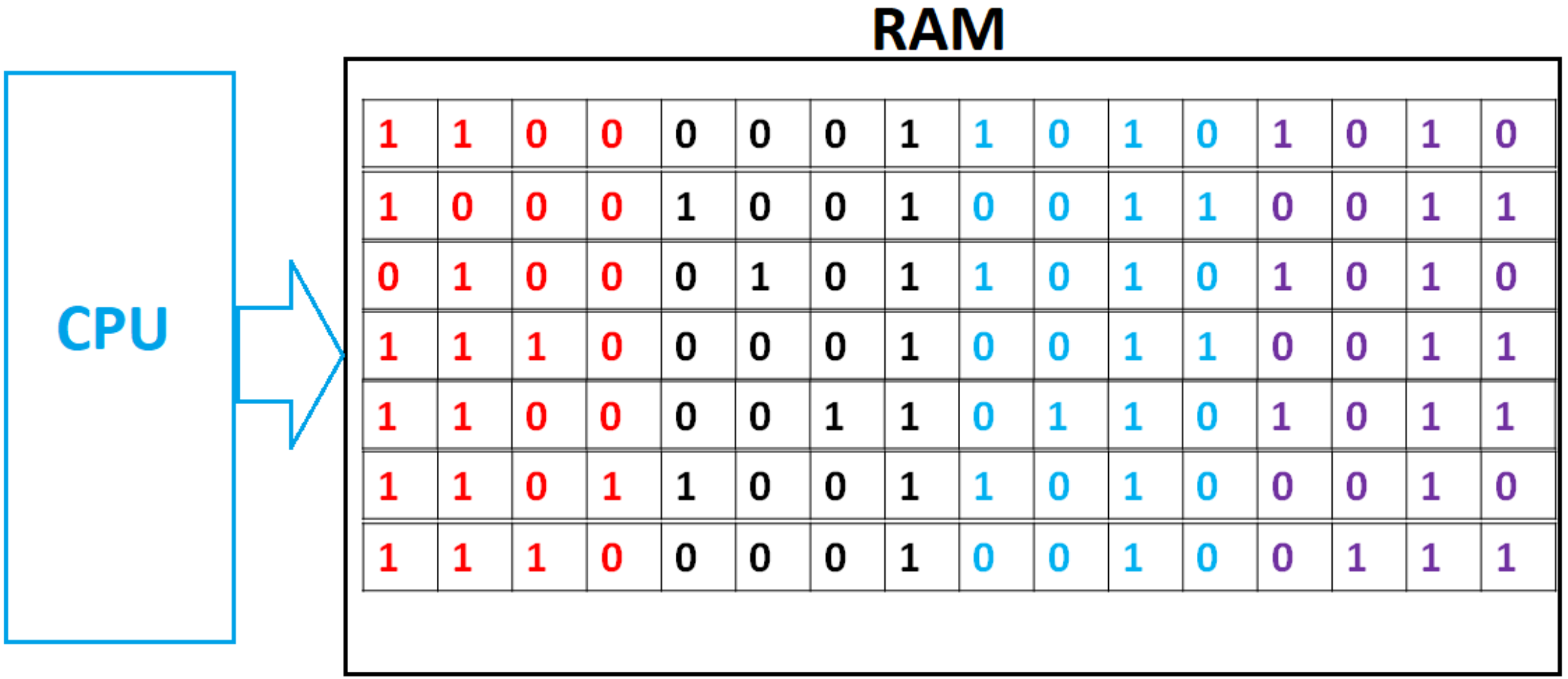
- ENIAC was completed in 1945 and first put to work for practical purposes on December 10, 1945

EDVAC (Electronic Discrete Variable Automatic Computer)

- It was binary rather than decimal, and was designed to be a stored-program computer.
- Functionally, EDVAC was a binary serial computer with automatic addition, subtraction, multiplication, programmed division and automatic checking with an ultrasonic serial memory capacity of 1,000 words.
- EDVAC's average addition time was 864 microseconds and its average multiplication time was 2,900 microseconds.
- EDVAC was designed to receive its instructions electronically; moreover, the program, coded in zeros and ones, would be kept in the same place that held the numbers the computer would be processing. This approach—letting a program treat its own instructions as data—offered huge advantages.



- **Stored program computer:** This was the storing of the program in the same memory as the data. It has been followed by the great majority of all computers ever made since.
- Primary memory is fast, but ***volatile*** – it loses all information stored in it when the power goes off. Primary memory is primarily RAM (Random Access Memory).

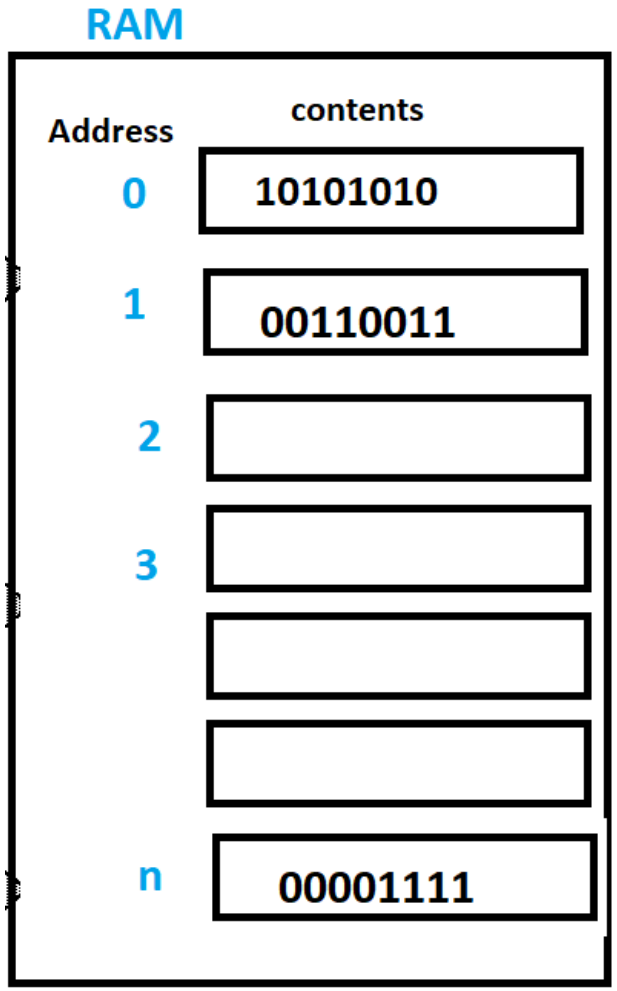


Memory Addresses

- The memory location is the smallest addressable unit
- Each memory location has a unique address, consecutive locations have addresses differing by 1
- For byte-sized memory locations

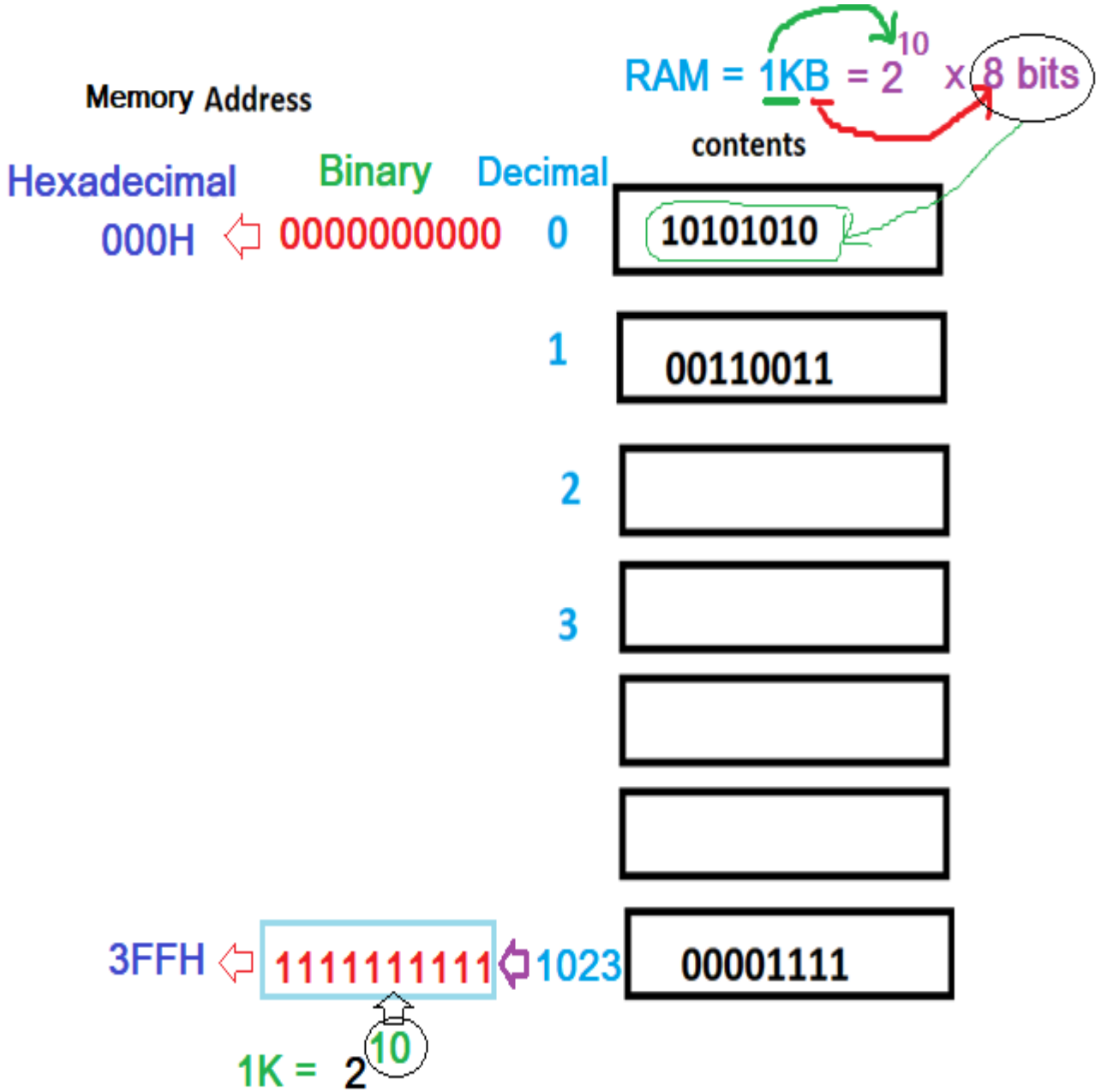
0	00001111
1	10000001
2	00000000
3	00000000

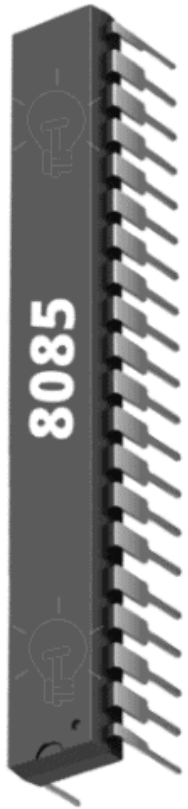
← 8 bits →



Memory Address

- Unique codes used by the programmer to identify locations fo data read and write from/to RAM i low-level language
- Unique codes used by the CPU to identify locations for data read and write from/to RAM
- For 1KB RAM
- No of locations = 1024
- Contents of each location = 8 bits
- 8 bits = 1 Byte
- 1KB is a byte addressable Memory





CPU

address bits send

1 0 0 1 1 1 0 0 1 1

ADDRESS BUS

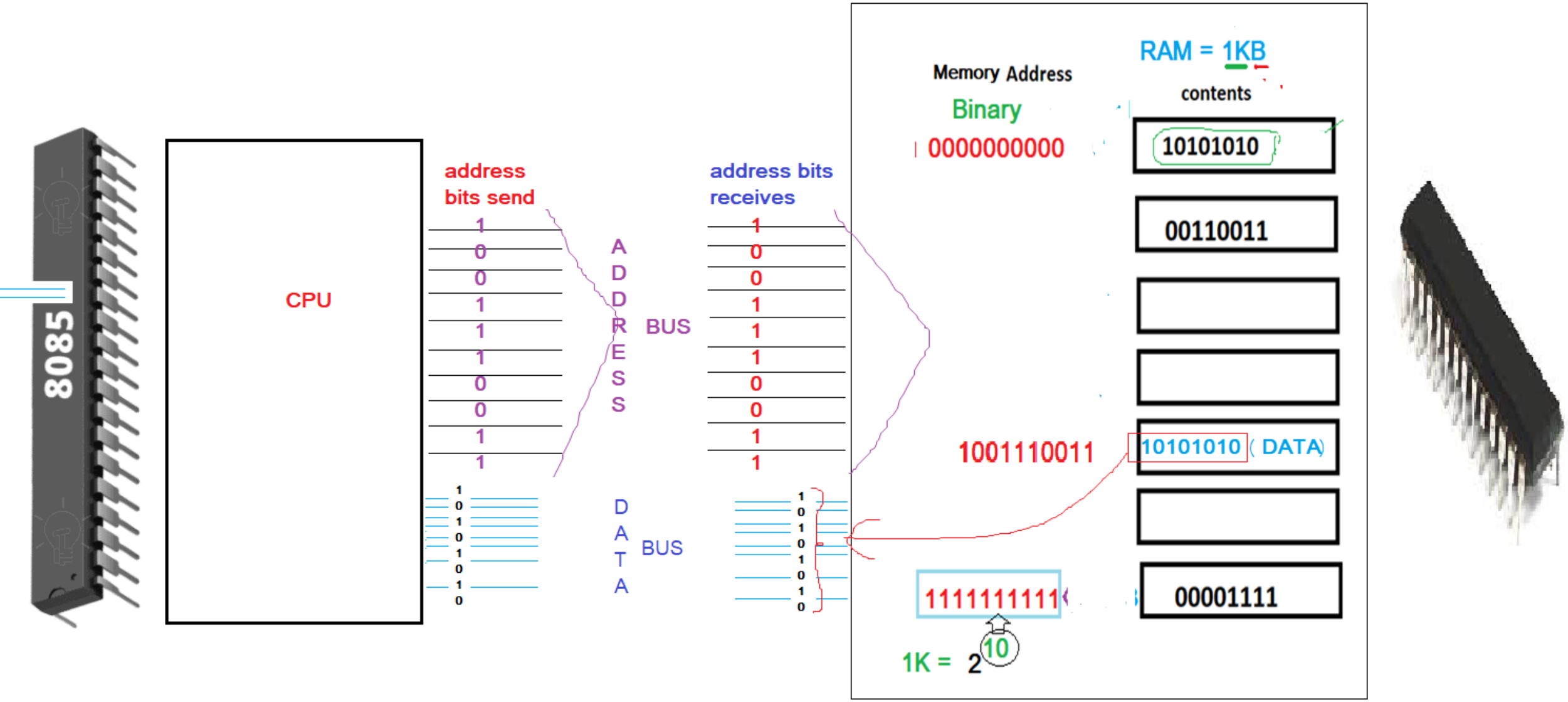
address bits receives

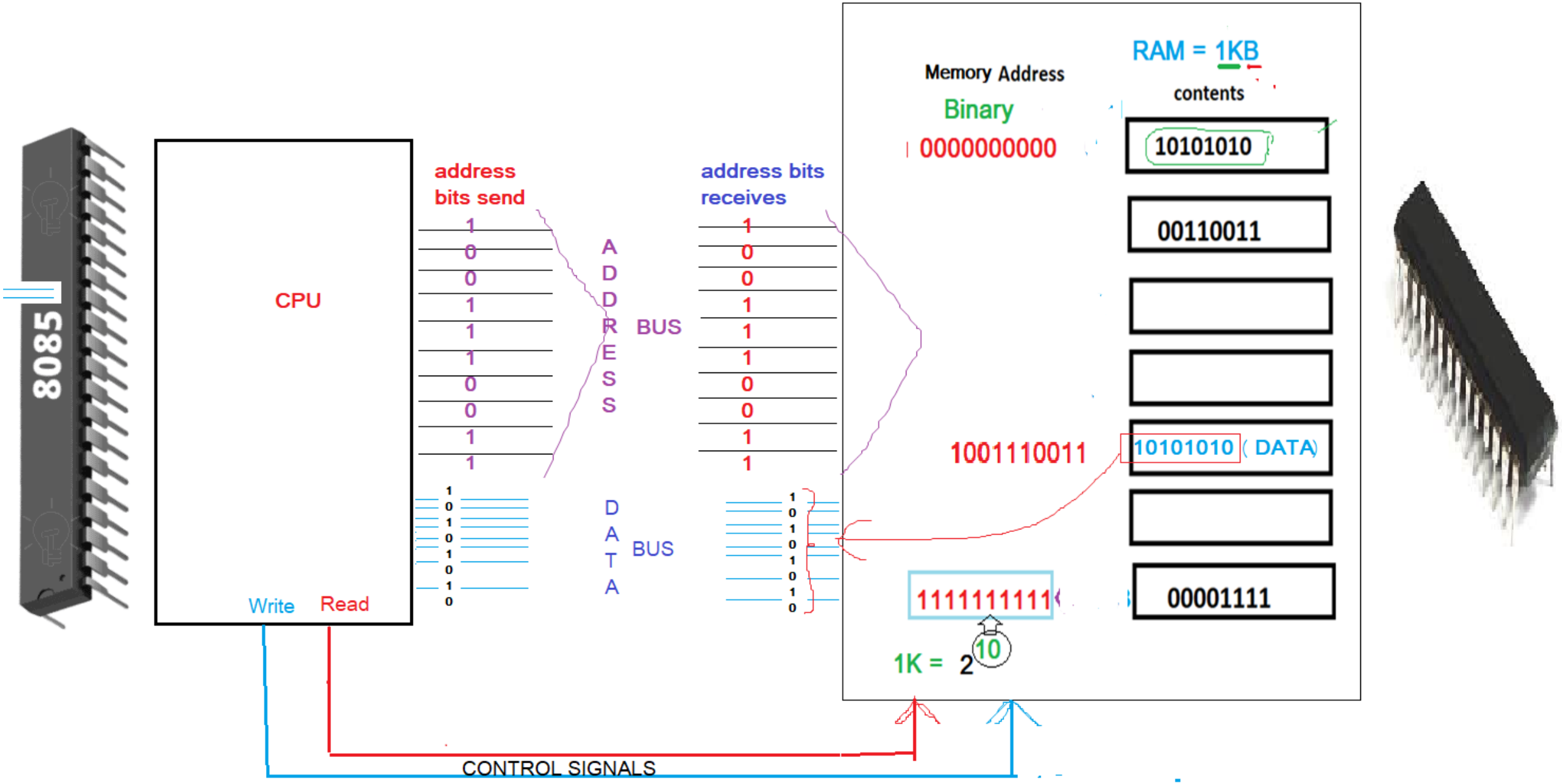
1 0 0 1 1 1 0 0 1 1

RAM = 1KB

Memory Address (Binary)	contents
0000000000	10101010
	00110011
1001110011	10101010 (DATA)
1111111111	00001111

1K = 2^{10}


8085
CPU
address
bits send
1
0
0
1
1
1
0
0
1
1
A
D
D
R
E
S
S
BUS
address bits
receives
1
0
0
1
1
1
0
0
1
1
D
A
T
A
BUS
1
0
1
0
1
0
1
0
RAM = 1KB
Memory Address
Binary
contents
0000000000
10101010
00110011
1001110011
10101010 (DATA)
1111111111
00001111
1K = 2^10


8085
CPU
Write
Read
address
bits send
address bits
receives
ADDRESS BUS
DATA BUS
CONTROL SIGNALS
RAM = 1KB
Memory Address
Binary
contents
0000000000
10101010
00110011
1001110011
10101010 (DATA)
1111111111
00001111
1K = 2^10

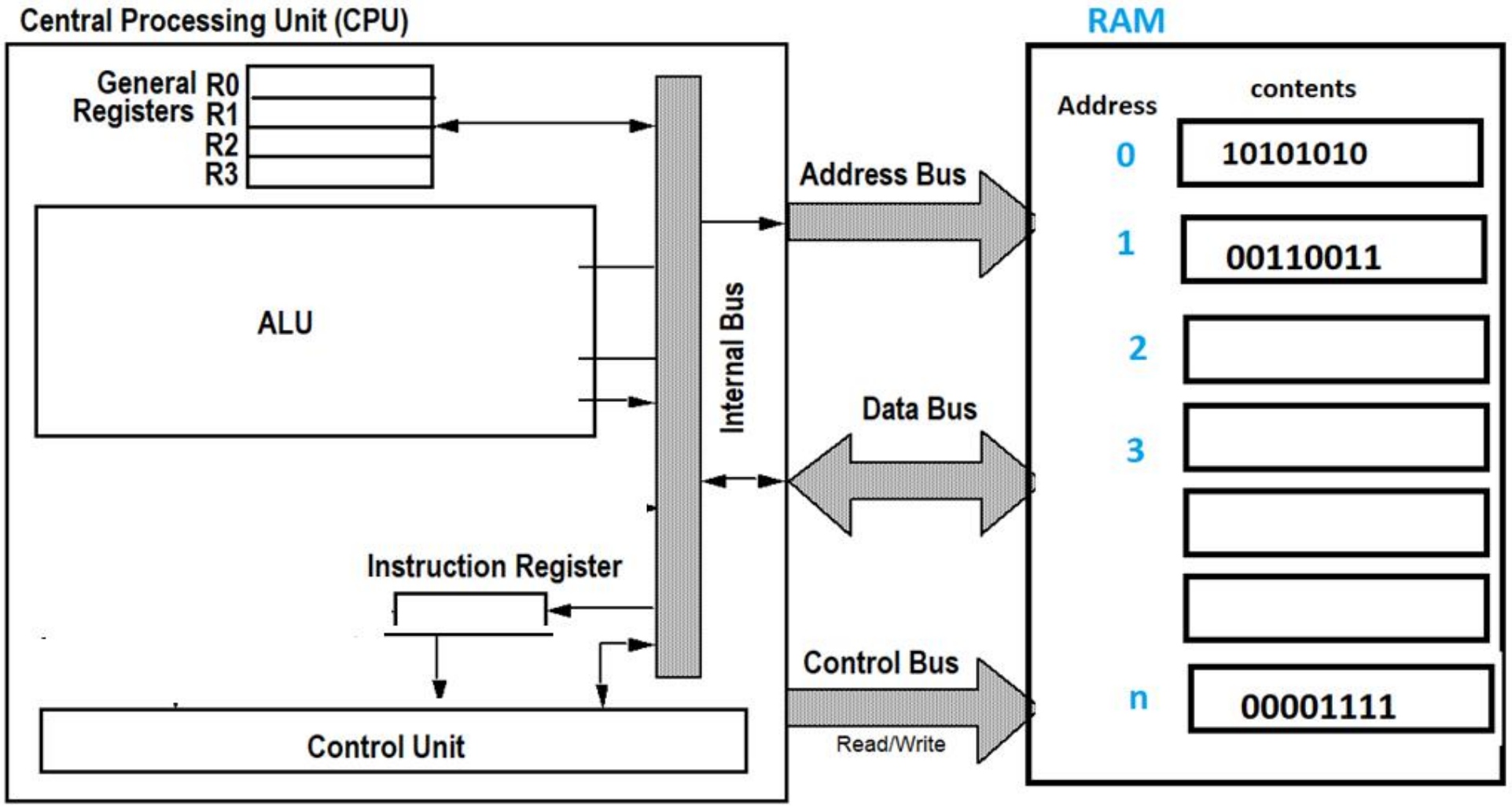
Memory Structure

- The basic unit of memory is the binary digit, called a bit 0 1
- Number of bits per memory location is computer-dependant
  - IBM 370: 8 bits per cell
  - Dec PDP-8: 12 bits per cell
  - Honeywell 6180: 36 bits per cell
  - CDC Cyber: 60 bits per cell
- A memory location that is 8 bits wide is referred to as a byte

  00000000 11111111 00101011
- A memory location that is 16 bits wide is referred to as a word

  0000000000000000 1111010100001111

- Memory: part of the computer where programs and data are stored
- Memory consists of a number of sequential storage locations
- Each memory location is capable of storing one data element or one computer instruction

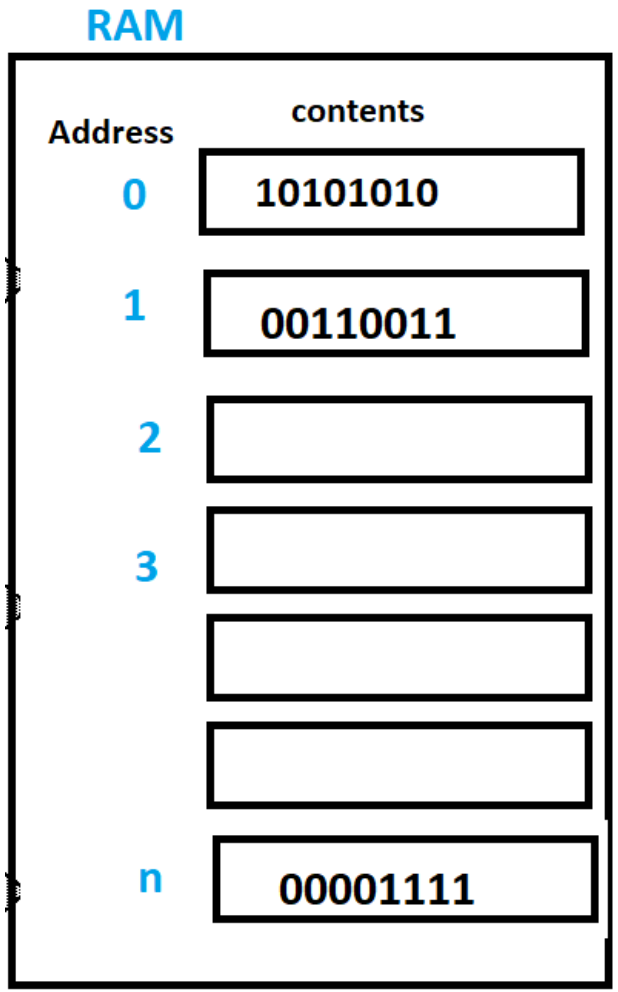


Memory Types

- Most computer systems have two types of memory
  - Main memory, which contains two kinds of memory
    - RAM - Random Access Memory
      - Used for reading/writing
      - Contents are lost when power is removed
    - ROM - Read Only Memory
      - Contents of ROM not lost when computer is turned off
      - Useful for storing "boot" program
      - Can contain firmware - software permanently stored in hardware
  - Secondary storage
    - Used to save programs and data normally stored in RAM
    - Contents not lost when unit is turned off
    - E.g., Floppy disk drive

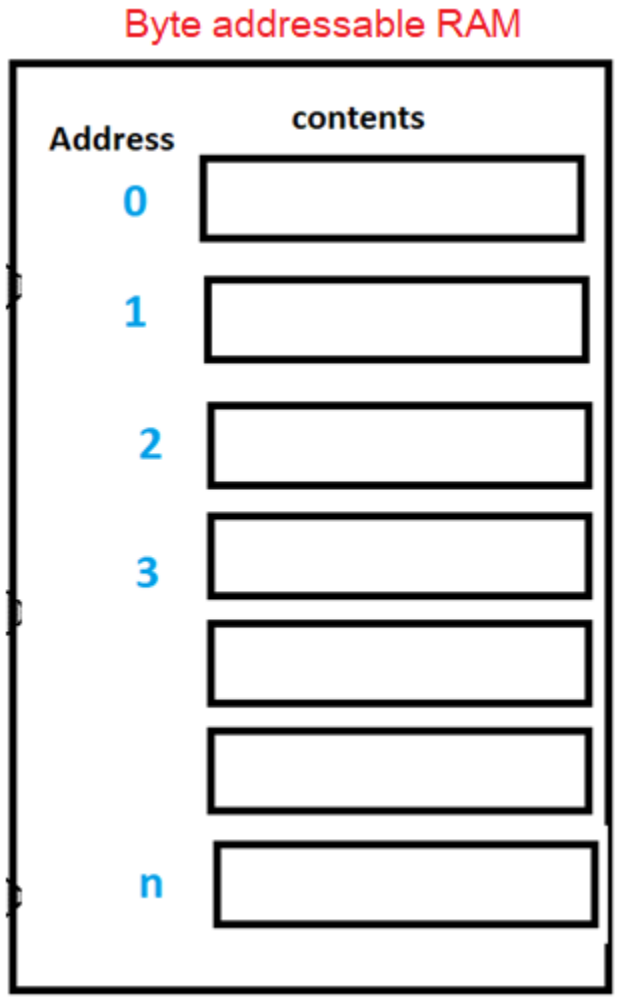
What is a Byte addressable Memory?

- Content of each location is 8 bit
- Content of each addressable location is 8 bit

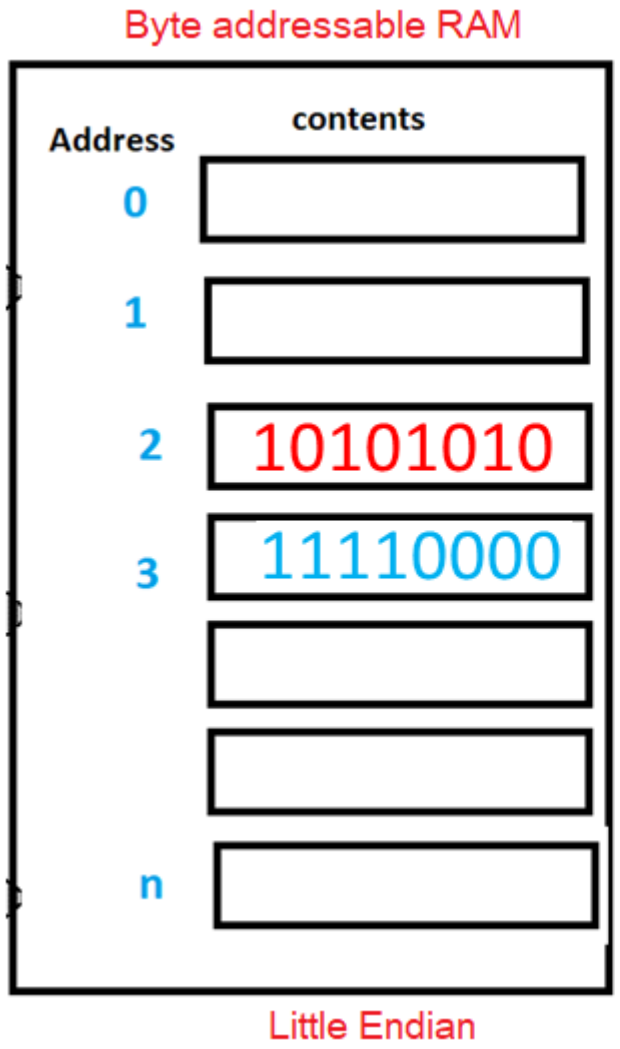


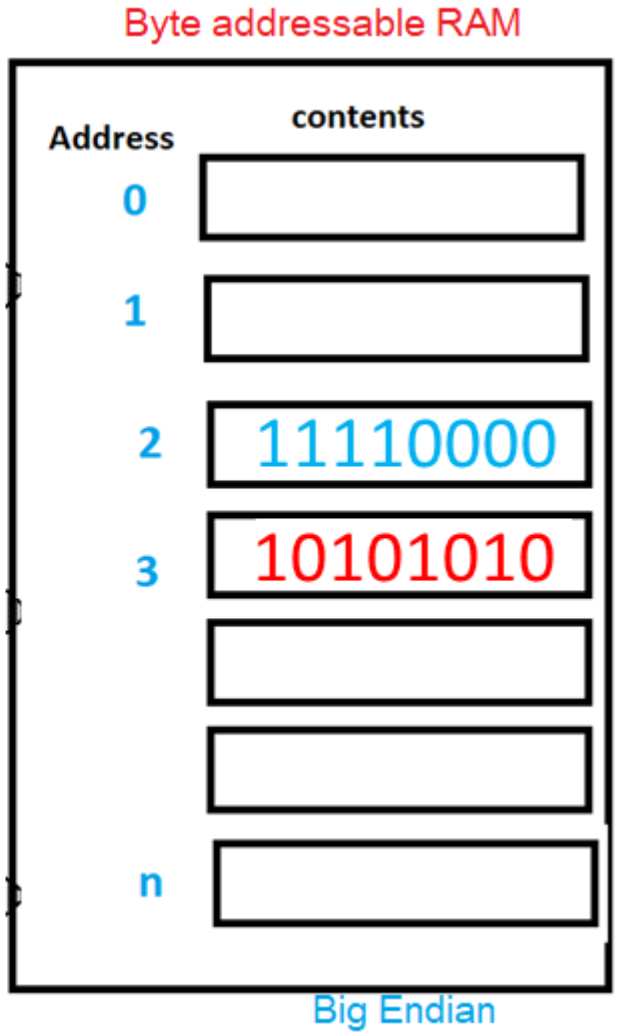
How 16 bits data are stored in a Byte addressable RAM?

- 11110000 10101010



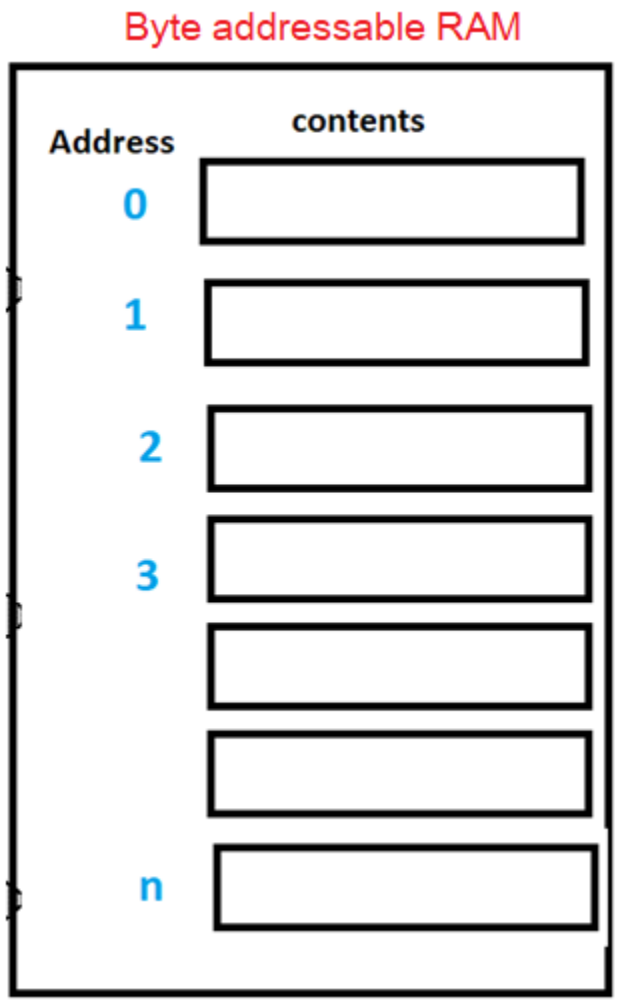
How 16 bits data are stored in a Byte addressable RAM? Example: 11110000 10101010

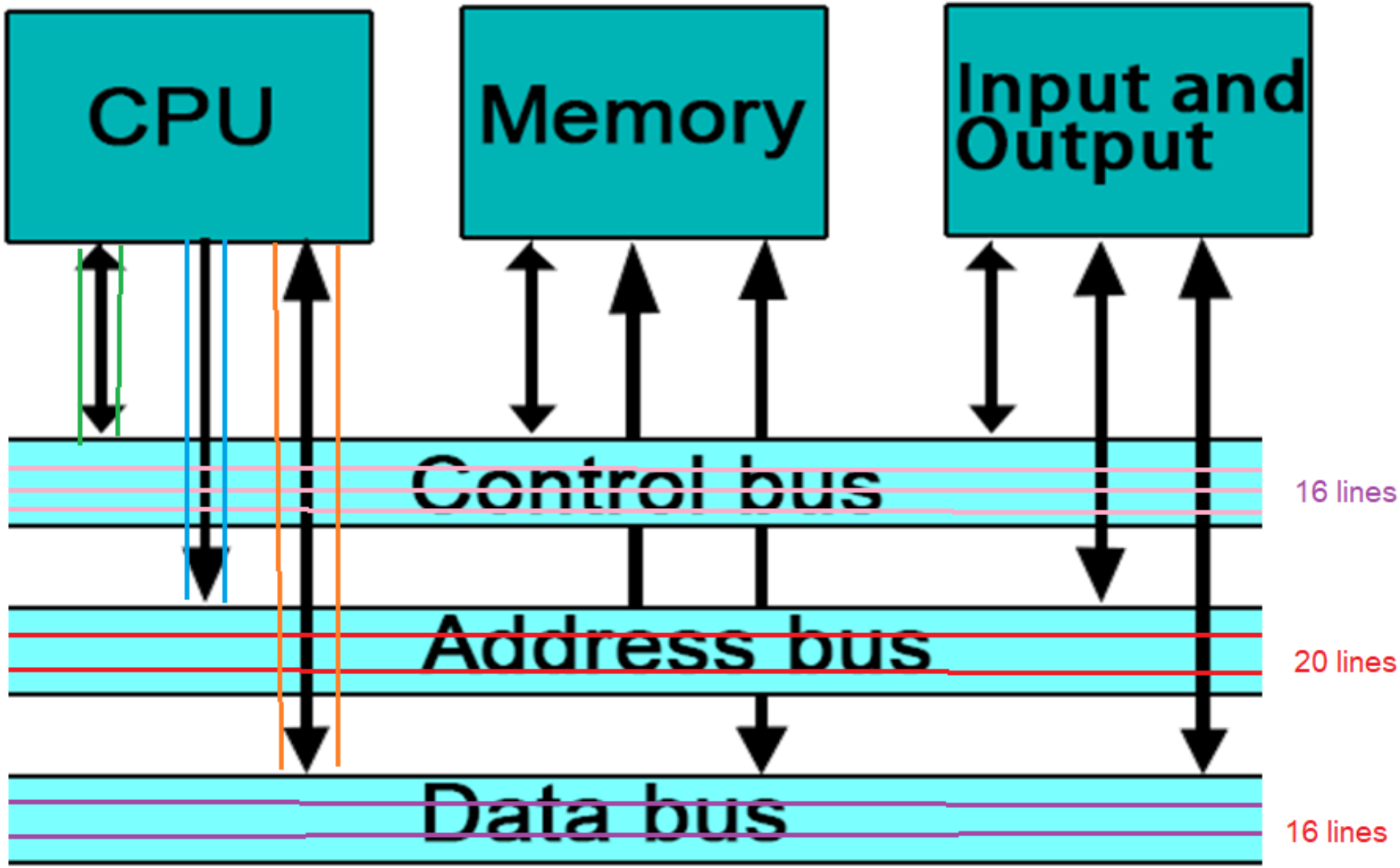




How 32 bits data are stored in a Byte addressable RAM?

Example: 00001111 10101010 11001100 01010101




CPU
Memory
Input and Output
Control bus
Address bus
Data bus
16 lines
20 lines
16 lines

Bus system

- Data bus

  Size: No of lines: decides no of bits transferred at a time (CPU to/from RAM)
  Which one is better: 8 bit data bus or 16 bit data bus or 32 bit data bus
  Signal flow is bidirectional
  RAM to CPU (during read operation): Input with respect to CPU
  and
  CPU to RAM (during write operation): Output with respect to CPU

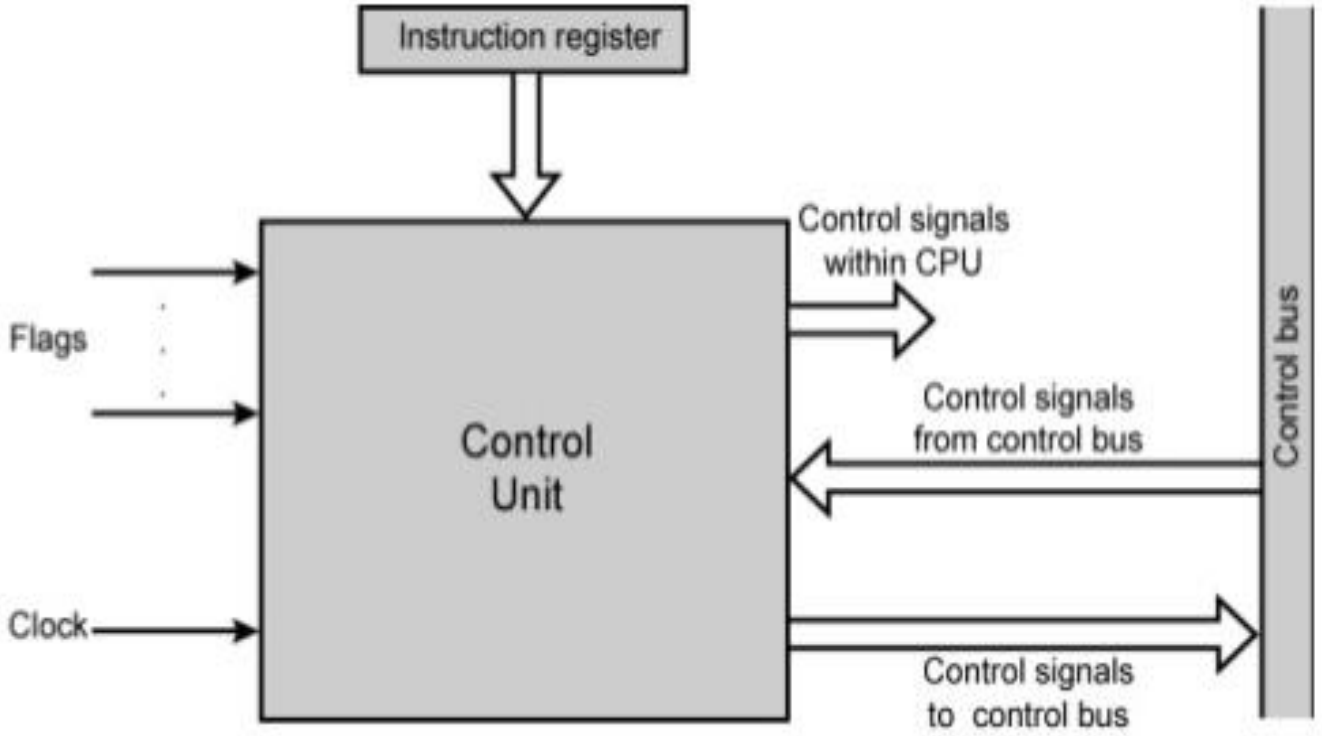
- Address bus
  - Size: No of lines: defines the capacity of RAM
  - If address bus is 20 bits, what is the capacity of RAM to be used with CPU
  - Signal flow: always from CPU to RAM: Output with respect to CPU
- Control bus
  - Size: No of functions: defines the functions a CPU can perform
  - Examples of control signals: READ, WRITE, RESET, INTERRUPT
  - Some control signals are Input to CPU
  - Some control signals are output with respect to CPU
  - Some are bidirectional

Control Unit

The **control unit** (CU) is a component of a computer's central processing **unit** (**CPU**) that directs the operation of the **processor**. It tells the computer's memory, arithmetic logic **unit** and input and output devices how to respond to the instructions that have been sent to the **processor**.

The **control unit** of the central processing **unit** regulates and integrates the operations of the **computer**. It selects and retrieves instructions from the main memory in proper sequence and interprets them so as to activate the other functional elements of the system at the appropriate moment.

A control Unit (programs are decoded into electronic signals and electronic control and implementation of all other sections

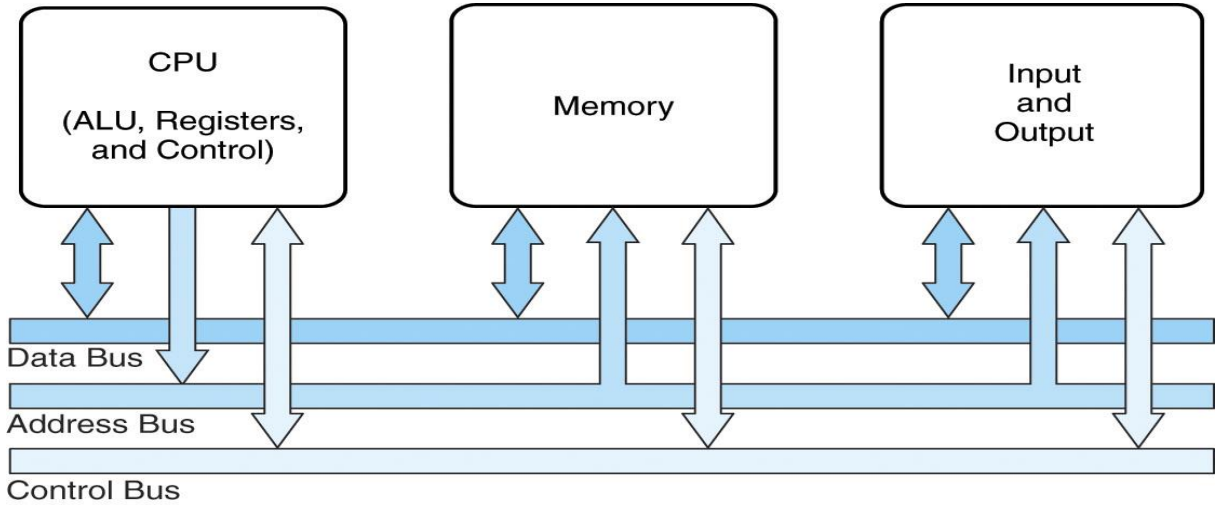


Input/Output Devices

- Computer I/O devices are often referred to as Peripherals
- Examples: disk drives, video monitors, printers, keyboards
- Two general classes of I/O interface
  - Serial interface port
    - Transmits/Receives data one bit at a time
    - Can transmit over long distances
  - Parallel interface port
    - High speed data transfer
    - Limited by cable length

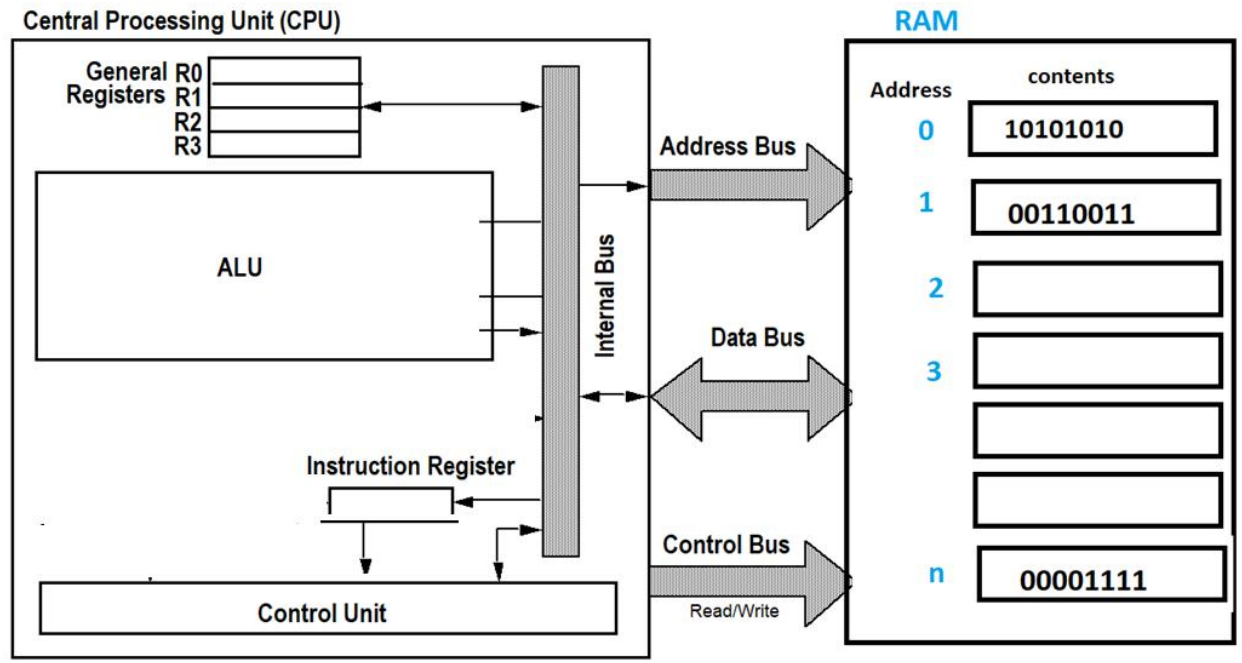
Bus Architecture

- A bus is a collection of electronic signal lines all dedicated to a particular task
- The CPU, memory, and I/O are separate electronic modules that are interconnected by buses
- In a computer, there typically are three buses
  - Address Bus
  - Data Bus
  - Control Bus



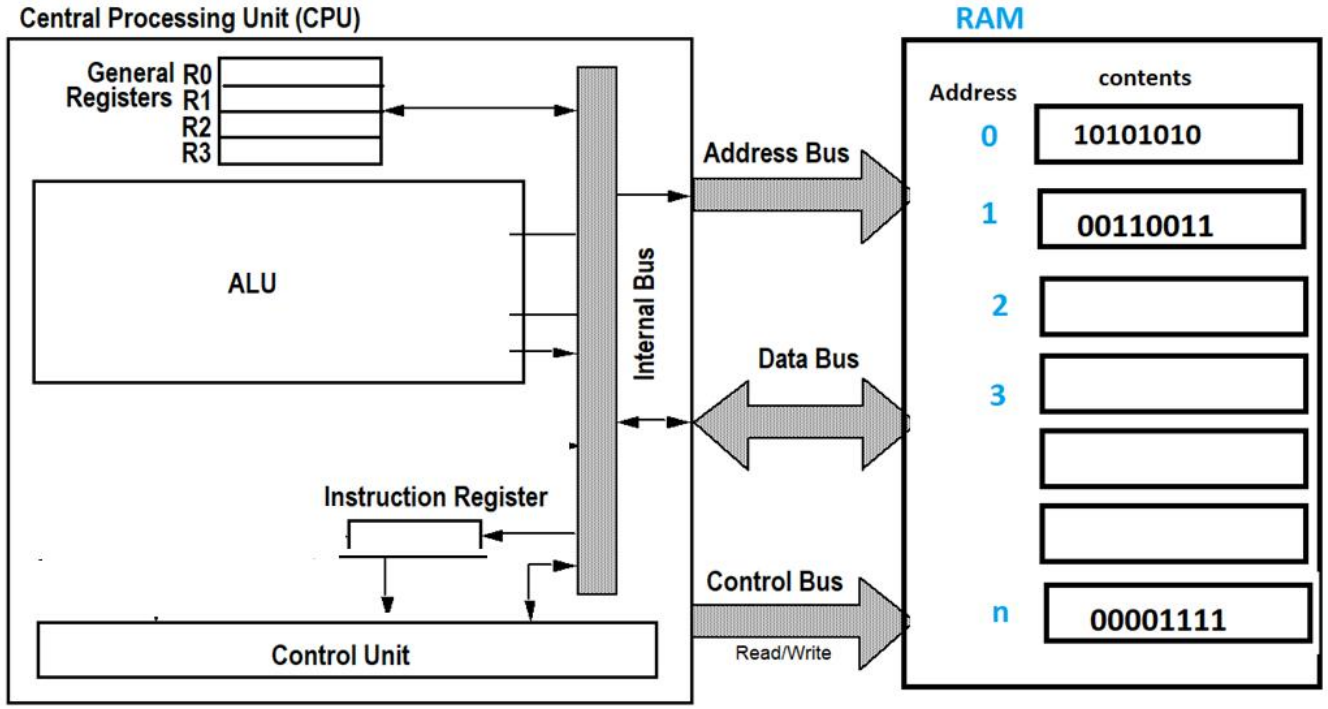
Data Bus

- The width of the data bus in bits is used to classify the processor
  - 8-bit processor has an 8-bit data bus
  - 16-bit processor has a 16-bit data bus (Intel 8086 processor)
- The width of the data bus determines how much data the processor can read or write in one memory or I/O cycle
- If width of data exceeds capacity of data bus, reading/writing requires more than one read/write cycle -- less efficient
- The data bus is a bi-directional line



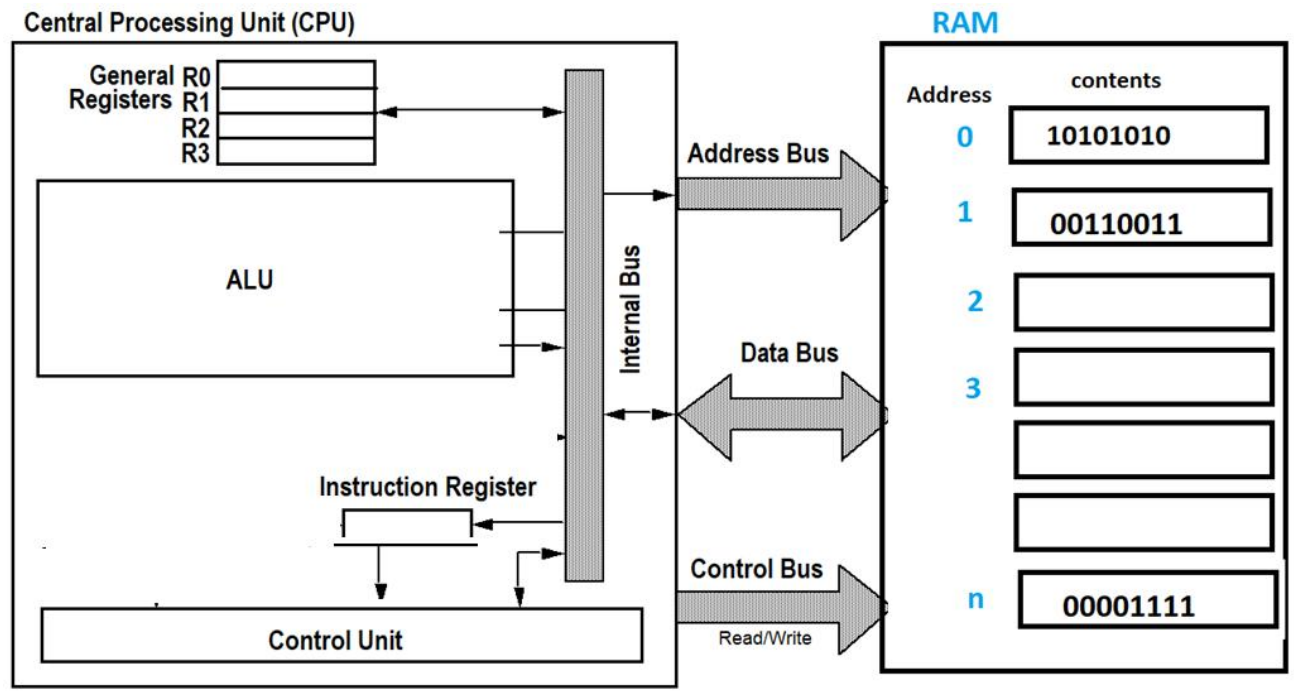
Address Bus

- The address bus is used to identify the location that the CPU will communicate with
- The address bus can identify a memory location or an I/O device, also called an I/O port
- For the 8086, the address bus is 20 bits wide, which allows for output to 2^{20} unique addresses
- The address bus is an output line

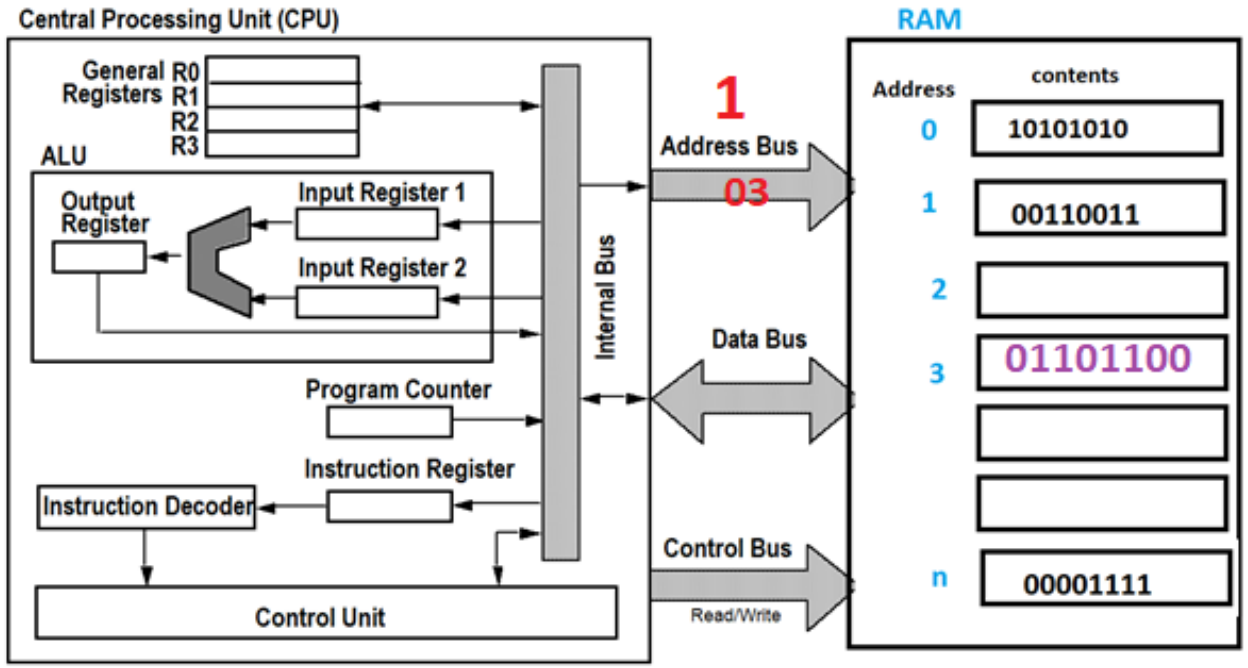


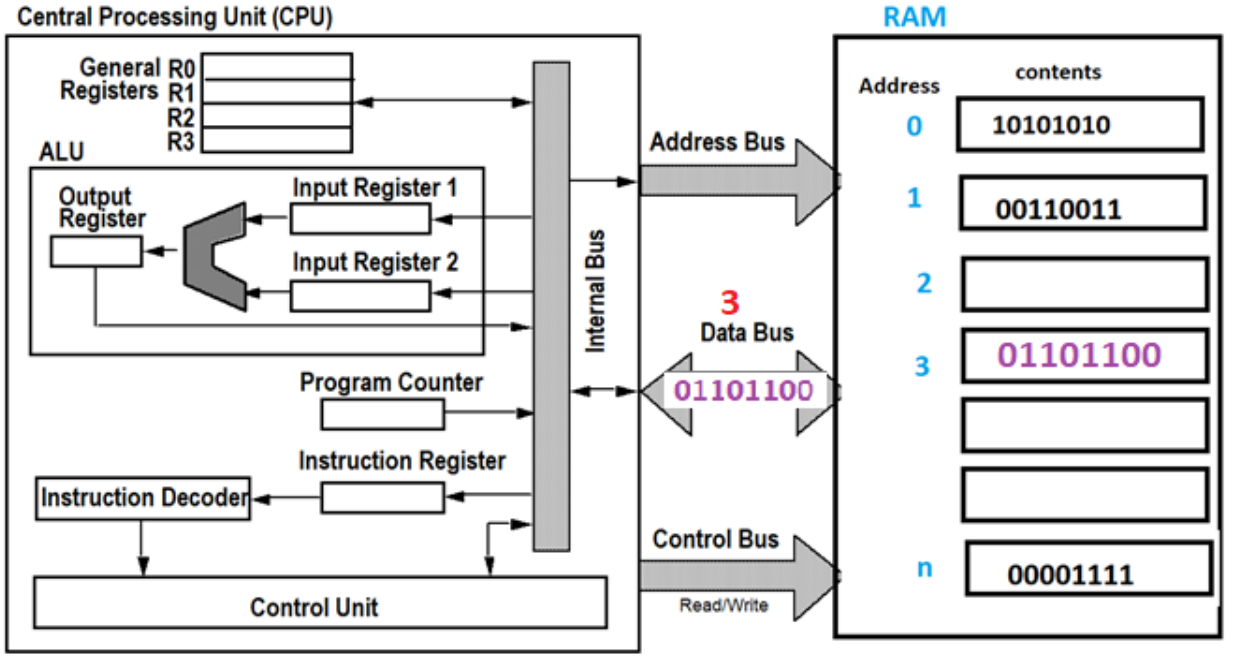
Control Bus

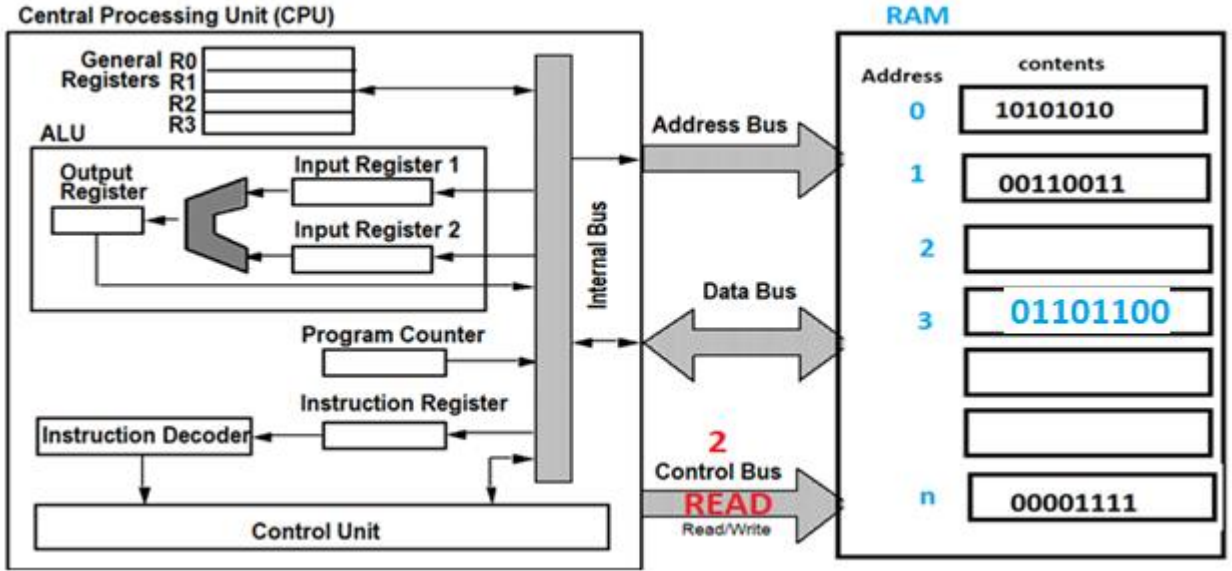
- The control bus identifies if the address on the address bus is for a memory location or for an I/O port
- The control bus identifies the direction of data flow on the data bus
- The CPU activates a control bus signal
  - MEMORY READ - read data from memory into the CPU
  - MEMORY WRITE - write data from the CPU to memory
  - I/O READ - read data from an input device into the CPU
  - I/O WRITE - write data from the CPU to an output device
- The control bus is an output line

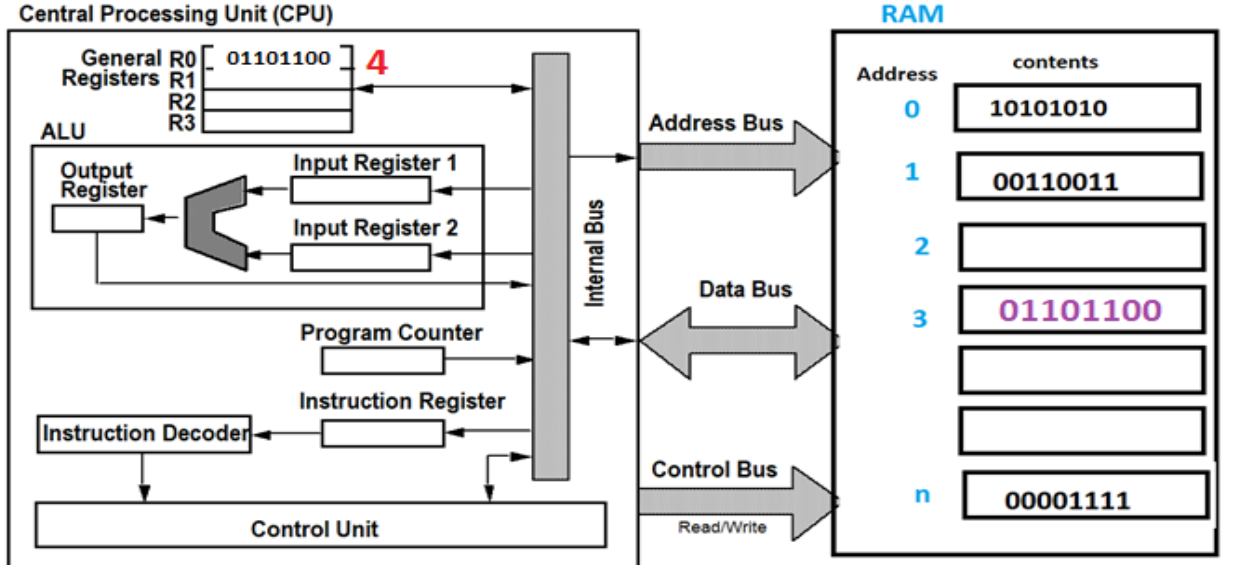


Read operation: Content of a RAM is copied in a Register

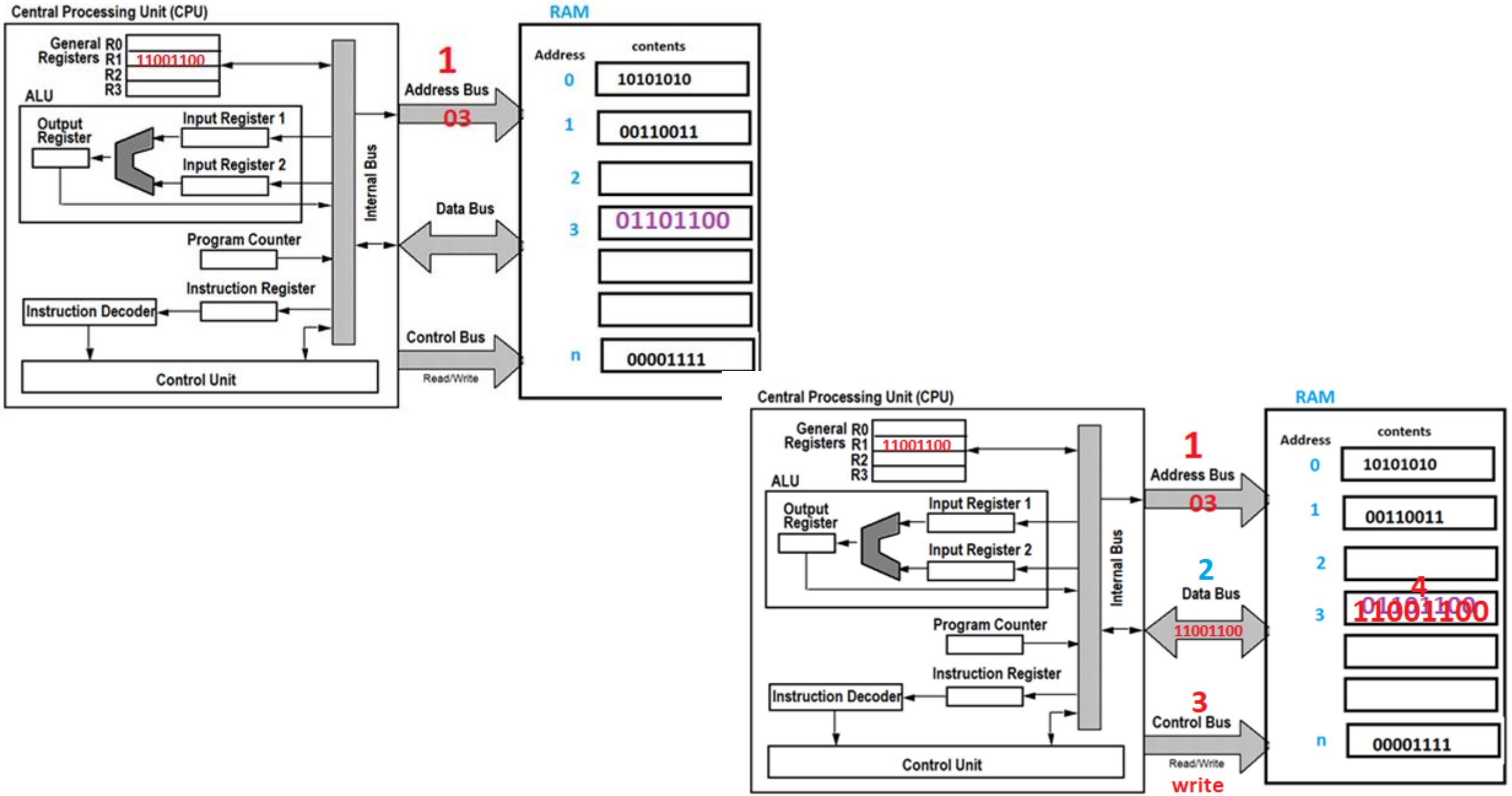








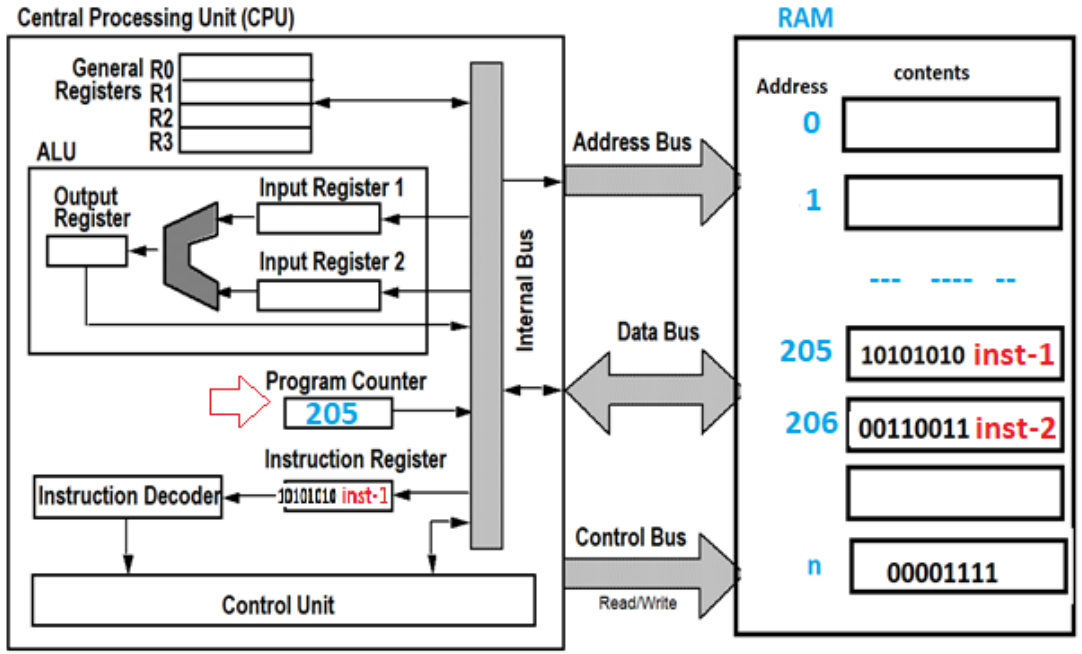
Write operation: Content of a Register is copied to a location of RAM

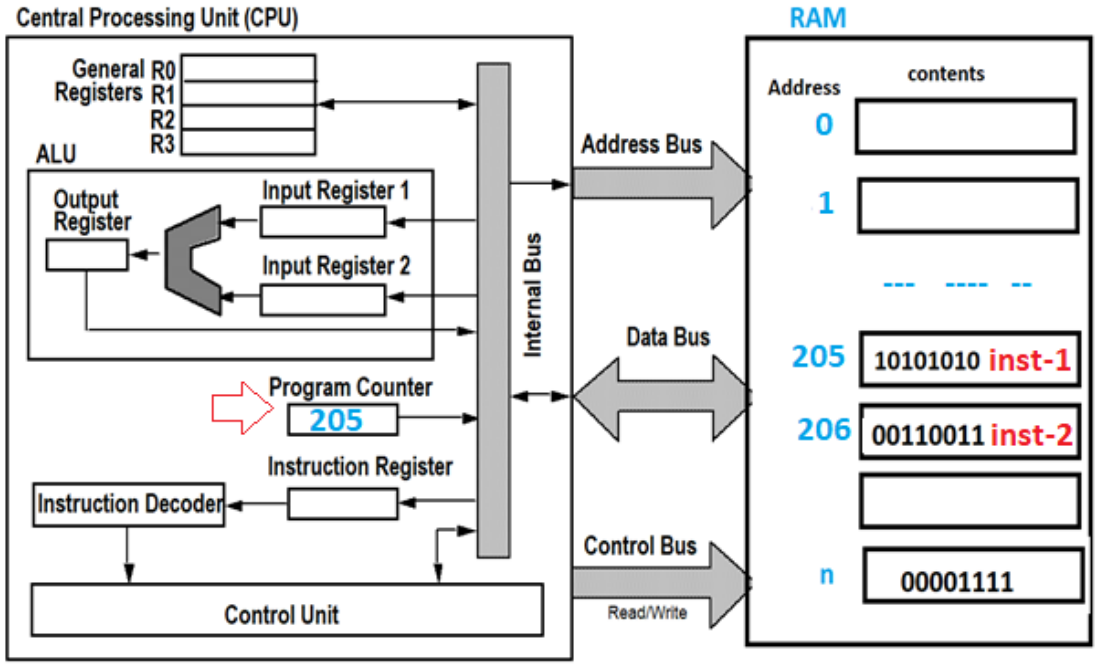


How does CPU run a program? How does CPU process a single Instruction?

The CPU executes an instructions in the following series of steps

- The **Program Counter (PC)** is a special register that holds the **address** of the 1st instruction to be fetched from Memory
- Read/Fetch 1st instruction from memory
- Move instruction into instruction register



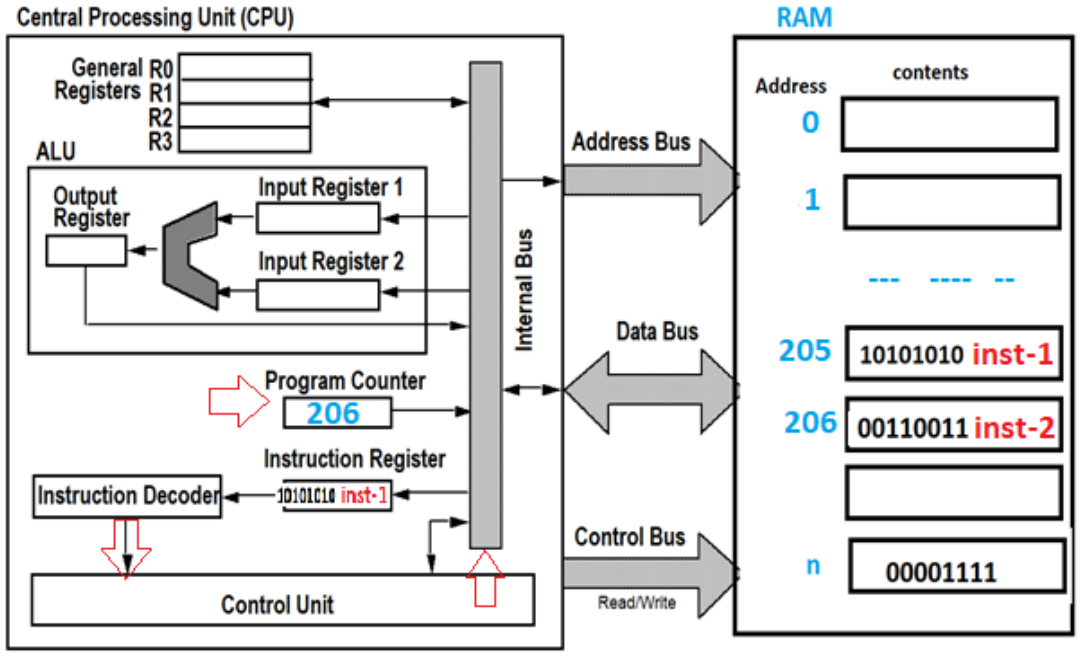


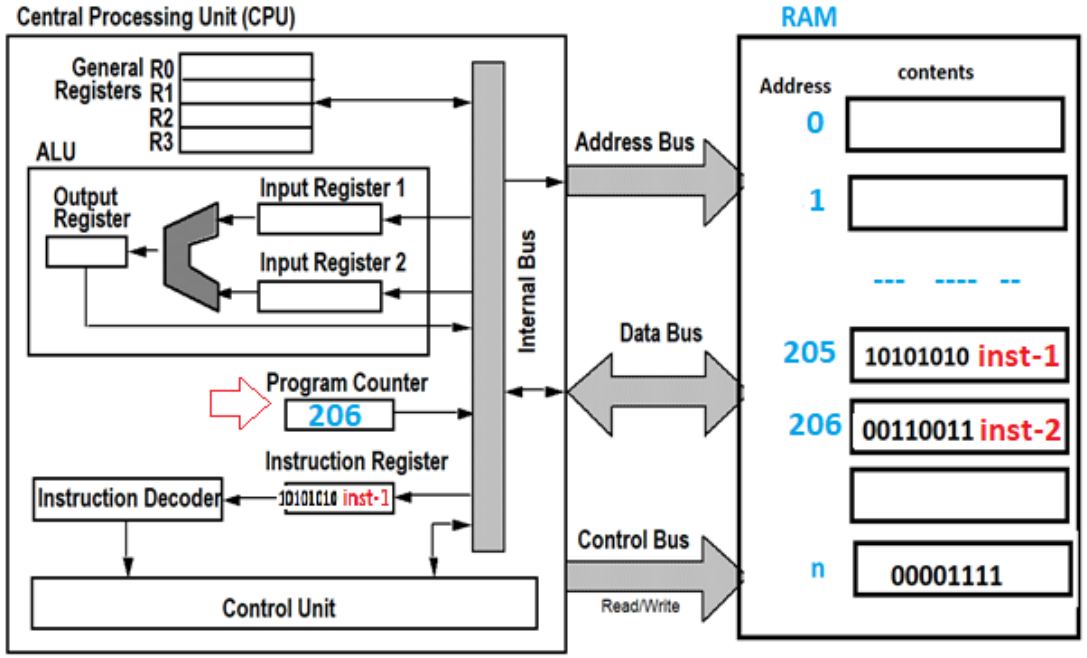
How does CPU run a program? How does CPU process a single Instruction?

The CPU executes an instructions in the following series of steps

- Increment/Change program counter to point to next instruction

- Decode instruction at Control Unit

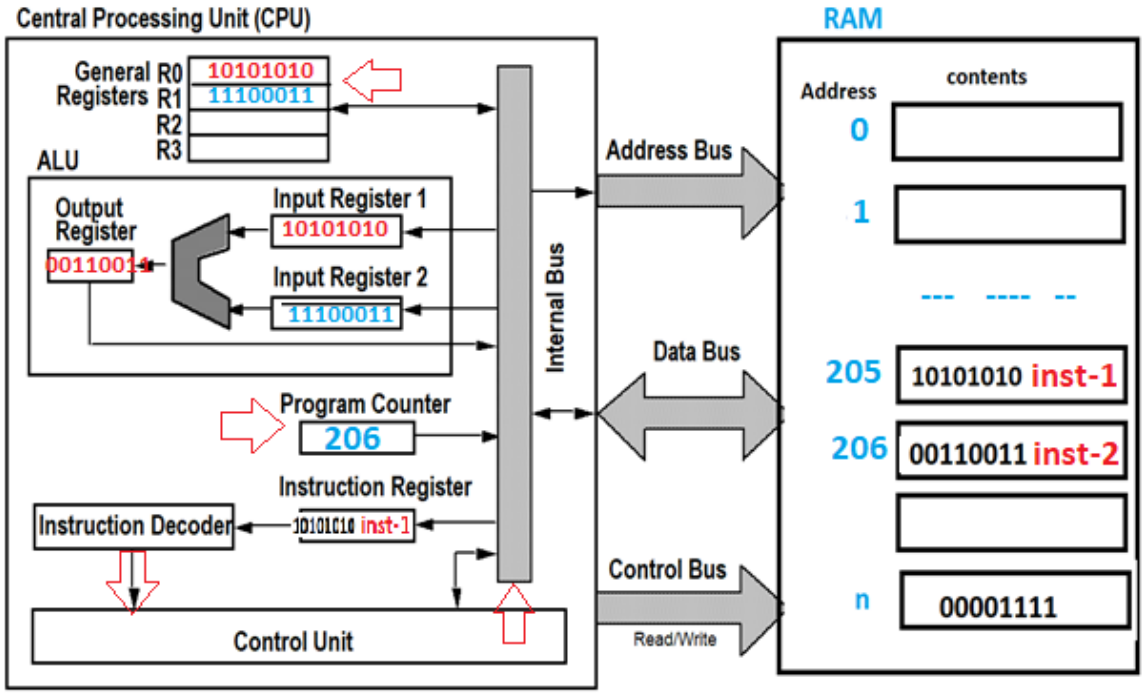


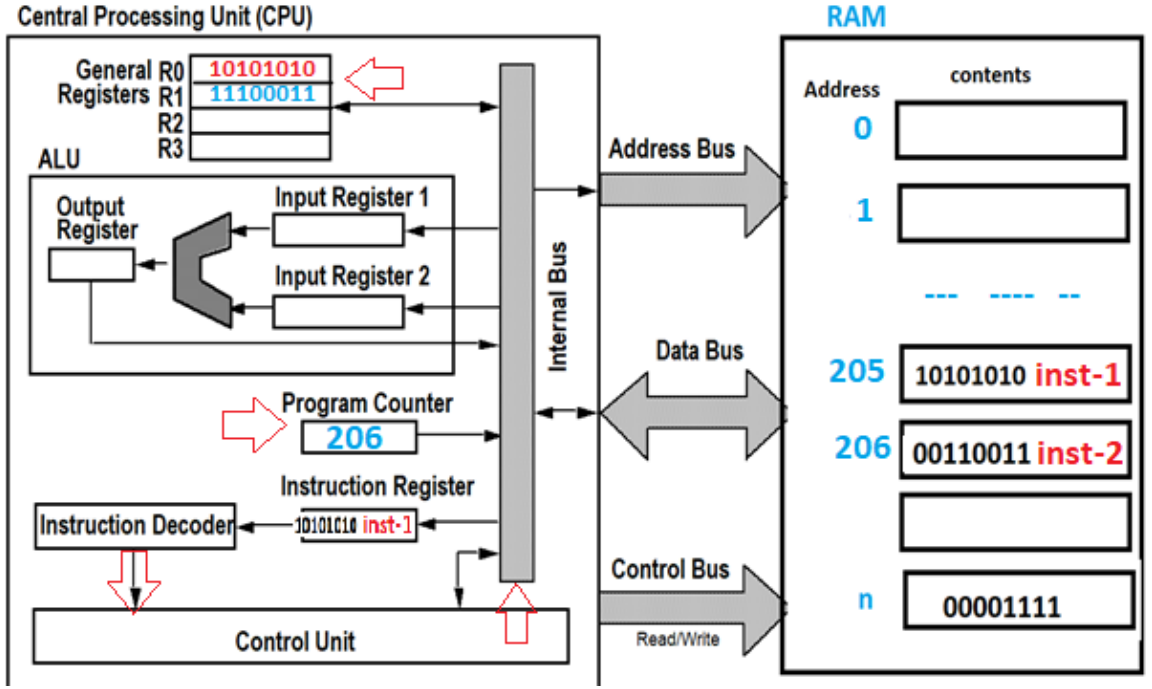


How does CPU run a program? How does CPU process a single Instruction?

The CPU executes an instructions in the following series of steps

- If required, read data from RAM and store into internal CPU registers
- Perform ALU operation at ALU
- Store result to a register
- Repeat steps…..for next instruction





Registers

- Program Counter
  - Holds the address of the memory cell to be fetched
  - "Points" to memory location of next instruction to be executed
  - After Fetch operation, program counter is incremented to point to the next instruction
- Instruction Register
  - Holds the instruction currently being executed
- Accumulator Register
  - Stores result of functions performed by arithmetic logic unit

Timing

- Basic timing is controlled by a clock generator circuit
- Clock signal is used to synchronize all activities within the computer
- Clock determines how fast instructions can be fetched from memory and executed
  - 16 MHz
  - 64 MHz
  - 800 MHz - Pentium 3
  - …

Computer Organization & Computer Architecture

Computer Organization

Design of the components and functional blocks using which computer systems are built.

– *Analogy*: civil engineer's task during building construction (cement, bricks, iron rods, and other building materials).

Computer Architecture

How to integrate the components to build a computer system to achieve a desired level of performance.

– *Analogy*: architect's task during the planning of a building (overall layout, floorplan, etc.).

Computer Architecture

- Data types
- Program (Instruction set)
- Input/Output mechanisms
- Techniques for addressing memory.

Computer Organization

- Hardware details
- Interfaces between the computer and peripherals
- Memory technology

Computer Architecture

Architecture describes what the computer does.

Computer Architecture deals with the functional behavior of computer systems.

In the above figure, it's clear that it deals with high-level design issues.

Architecture indicates its hardware.

As a programmer, you can view architecture as a series of instructions, addressing modes, and registers.

For designing a computer, its architecture is fixed first.

Computer Architecture is also called Instruction Set Architecture (ISA).

Computer Architecture comprises logical functions such as instruction sets, registers, data types, and addressing modes.

The different architectural categories found in our computer systems are as follows:

I. Von-Neumann Architecture
II. Harvard Architecture
III. Instruction Set Architecture
IV. Micro-architecture
V. System Design

It makes the computer's hardware visible.

Architecture coordinates the hardware and software of the system.

The software developer is aware of it.

Examples- Intel and AMD created the x86 processor. Sun Microsystems and others created the SPARC processor. Apple, IBM, and Motorola created the PowerPC.

Computer Organization

The Organization describes how it does it.

Computer Organization deals with a structural relationship.

In the above figure, it's also clear that it deals with low-level design issues.

Where Organization indicates its performance.

The implementation of the architecture is called organization.

For designing a computer, an organization is decided after its architecture.

Computer Organization is frequently called microarchitecture.

Computer Organization consists of physical units like circuit designs, peripherals, and adders.

CPU organization is classified into three categories based on the number of address fields:

I. Organization of a single Accumulator.
II. Organization of general registers
III. Stack organization

It offers details on how well the computer performs.

Computer Organization handles the segments of the network in a system.

It escapes the software programmer's detection.

Organizational qualities include hardware elements that are invisible to the programmer, such as interfacing of computer and peripherals, memory technologies, and control signals.

Evolution of Computers

about 2500 BC

Abacus

about 200 BC

Antithykera Mechanism

1840s

Difference Engine

early 1940s

War time analog computers

late 1940s

ENIAC Zeus & Colussus

early 1950s

Mainframe

late 1950s

1st Gen Minicomputers

1960s

2nd Gen Minicomputers

early 1970s

1st Gen Microcomputers

mid 1970s

2nd Gen Microcomputers

late 1970s

3rd Gen Microcomputers

early 1980s

1st Gen PC

mid 1980s

4th Gen Microcomputers

late 1980s

5th Gen Microcomputers

1990s

2nd Gen PC

present day

True mobile computing

Five generation of computer diagram

Experts say that there are five generations of computers which means there have been five periods during which the computer has taken a large jump in its technological development.

First Generation (1940 to 1956)	Second Generation (1956 to 1964)	Third Generation (1964 to 1971)	Fourth Generation (1971 to present)	Fifth Generation (Present & beyond)
				

Generation in computer terminology is **a change in technology a computer is/was being used**. Initially, the generation term was used to distinguish between varying hardware technologies. Nowadays, generation includes both hardware and software, which together make up an entire computer system.

First

(1940-1956) Everything began with vacuum tubes

Second

(1956-1963) Consisted of two types of devices, magnetic core, and transistors.

Third

(1964-1971) Silicon transistors were replaced by germanium transistors

Fourth

(1971-2010) Very-large-scale integration; single chip

Fifth

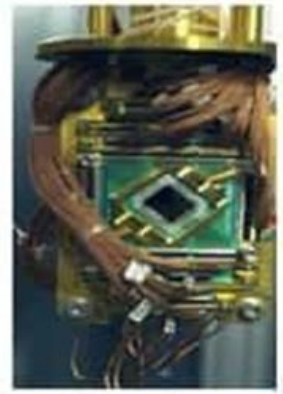
2010-current. AI. Very Large Scale Integration (VLSI) technology ; Ultra Large Scale Integration

Vacuum Tube

Transistors

Integrated Circuit

Microprocessor



Quantum Computer

1st Generation Computer

2nd Generation Computer

3rd Generation Computer

4th Generation Computer

5th Generation Computer

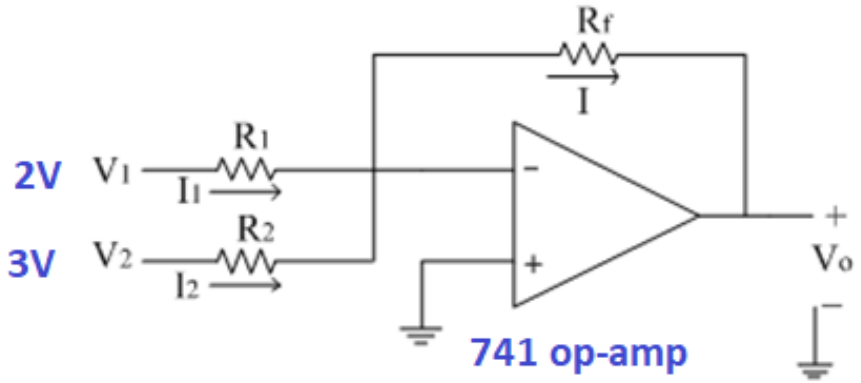
Computer Generations

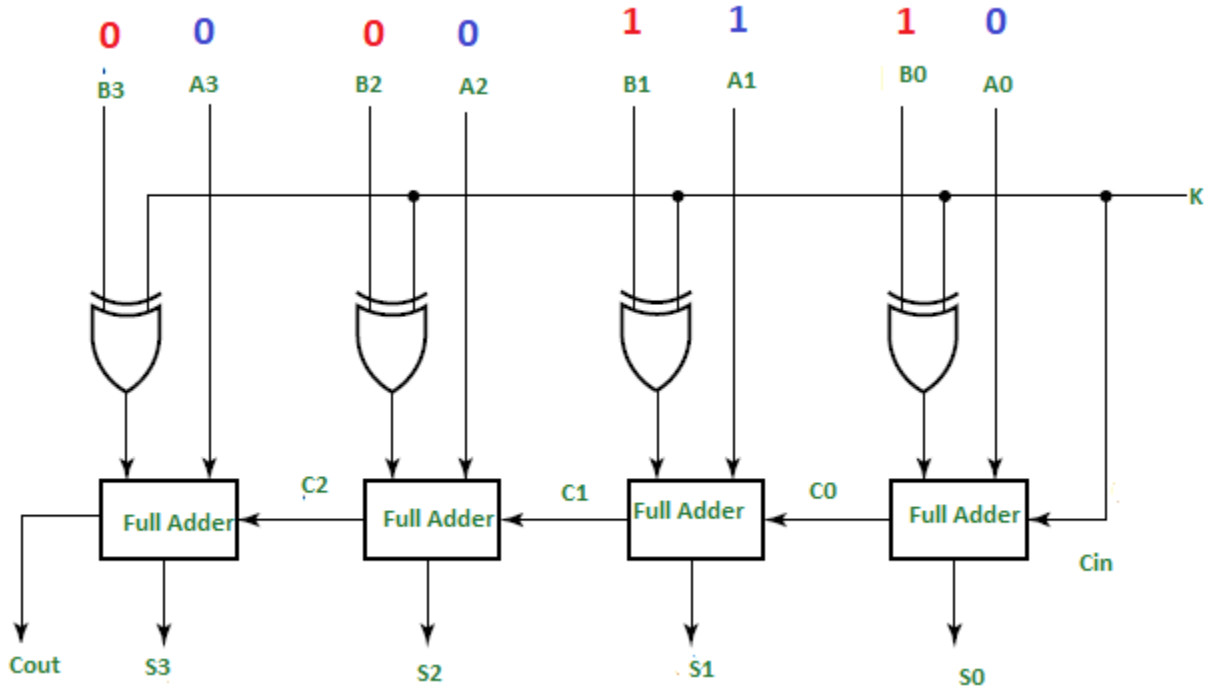
Generation	Hardware Components		Characteristics	Computers
First Generation (1942-1959)		⊙ Vaccum Tubes	⊙ Machine Language ⊙ Huge Size ⊙ Highly Expensive ⊙ High Consumption of Electricity	⊙ ENIAC ⊙ UNIVAC ⊙ EDVAC ⊙ EDSAC ⊙ IBM-701
Second Generation (1959-1965)		⊙ Transistors ⊙ Magnetic Tapes	⊙ Batch processing, Multiprogramming OS ⊙ Expensive ⊙ FORTRAN, COBOL	⊙ IBM 7000 ⊙ CDC 1604 ⊙ ATLAS ⊙ NCR 304 ⊙ Honeywell 400
Third Generation (1965-1975)		⊙ Integrated Circuits	⊙ Remote processing, time-sharing, Multiprogramming OS ⊙ Faster, Compact & Cheaper ⊙ PASCAL PL/1, BASIC, ALGOL-68	⊙ IBM 360/370 ⊙ PDP 8/11 ⊙ CDC 6600
Fourth Generation (1975-1988)		⊙ VLSI Microprocessor circuits	⊙ Time-sharing, real-time networks, distributed, GUI OS ⊙ Faster, Compact & Affordable ⊙ C, C++, DBASE	⊙ DEC 10 ⊙ STAR 1000 ⊙ CRAY-1/II ⊙ Apple II ⊙ VAX 9000
Fifth Generation (1988-Present)		⊙ ULSI Microprocessor circuits	⊙ Parallel Processing & Artificial Intelligence technology ⊙ C and C++, Java, .Net	⊙ IBM ⊙ Pentium ⊙ Param

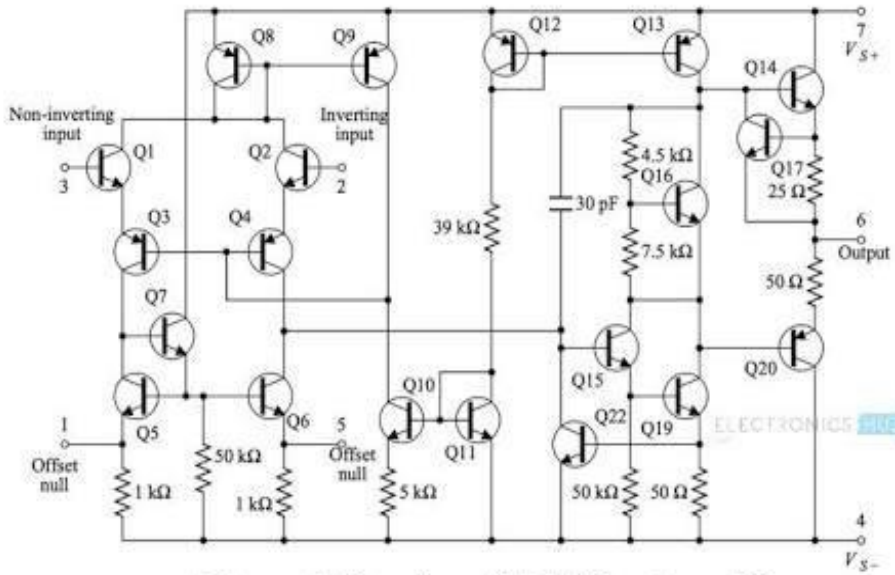
Analog vs Digital

2V = 0 0 1 0

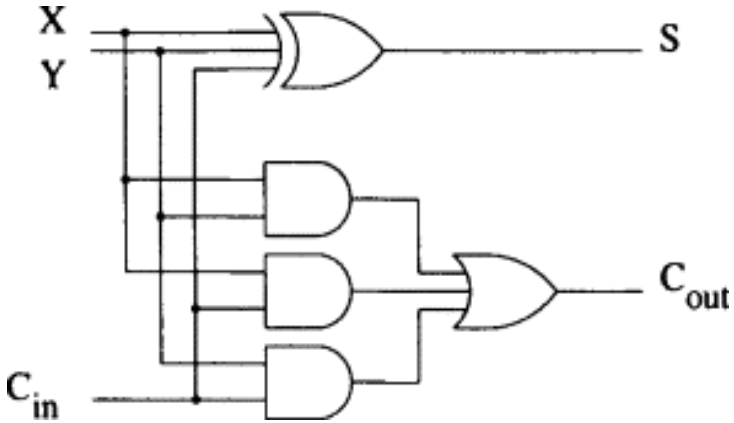
3V = 0 0 1 1







Internal Circuitry of 741 Op-Amp IC

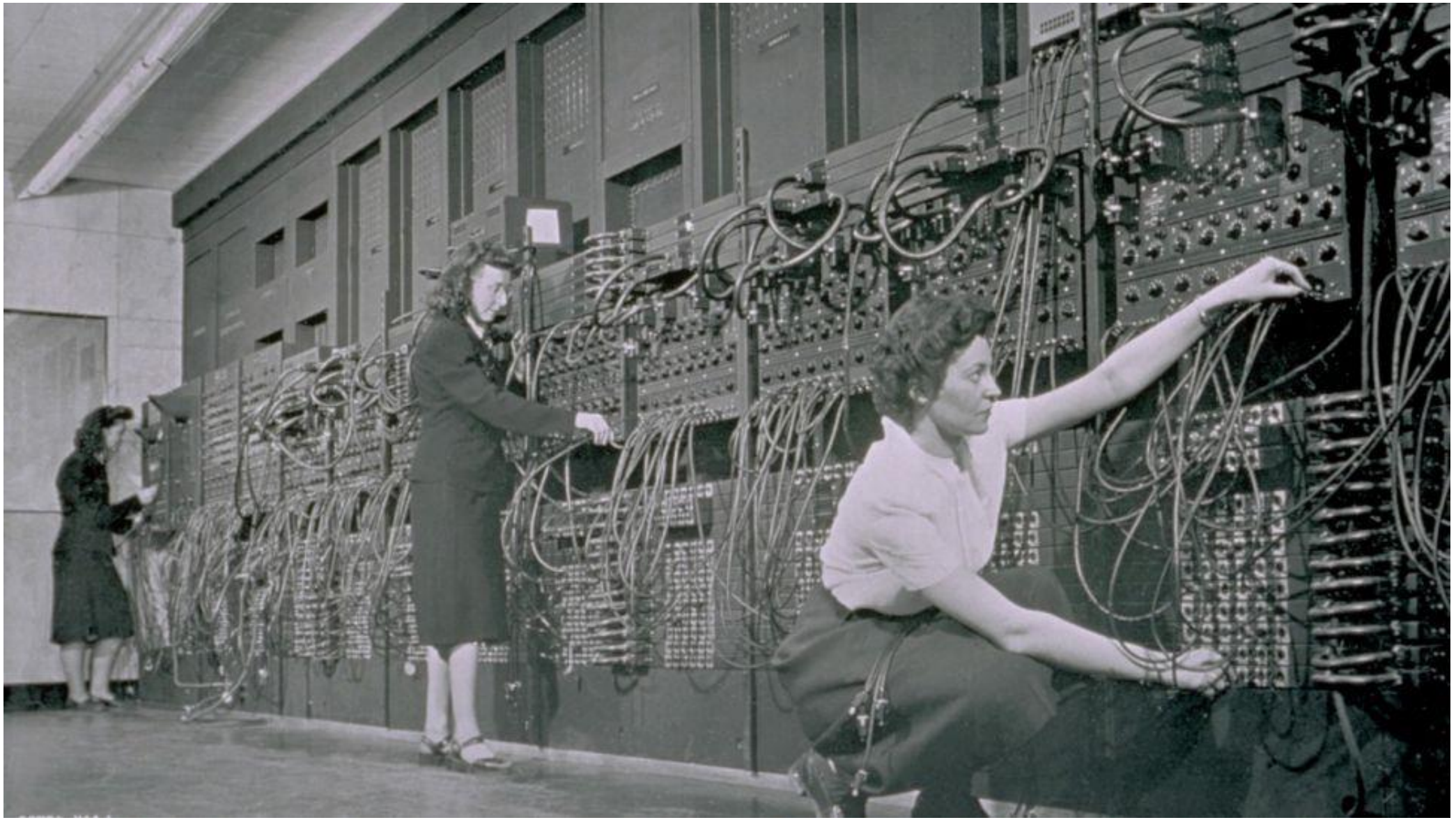


An **analog computer** or **analogue computer** is a type of computer that uses the continuous variation aspect of physical phenomena such as electrical, mechanical, or hydraulic quantities (*analog signals*) to model the problem being solved. In contrast, digital computers represent varying quantities symbolically and by discrete values of both time and amplitude (digital signals).

Analog computers were widely used in scientific and industrial applications even after the advent of digital computers, because at the time they were typically much faster, but they started to become obsolete as early as the 1950s and 1960s, although they remained in use in some specific applications, such as aircraft flight simulators, the flight computer in aircraft, and for teaching control systems in universities.

A 1960 Newmark analogue computer, made up of five units. This computer was used to solve differential equations and is currently housed at the Cambridge Museum of Technology.

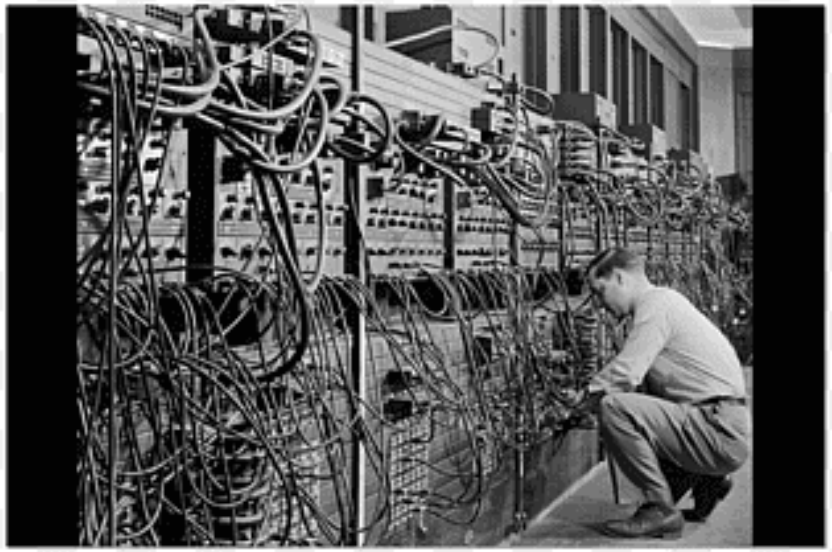
ENIAC: Two professors at the University of Pennsylvania, John Mauchly and J. Presper Eckert, design and build the Electronic Numerical Integrator and Calculator (ENIAC). The machine is the first "automatic, general-purpose, electronic, decimal, digital computer



1981: "Acorn," IBM's first personal computer, is released onto the market at a price point of $1,565, according to IBM. Acorn uses the MS-DOS operating system from Windows. Optional features include a display, printer, two diskette drives, extra memory, a game adapter and more.

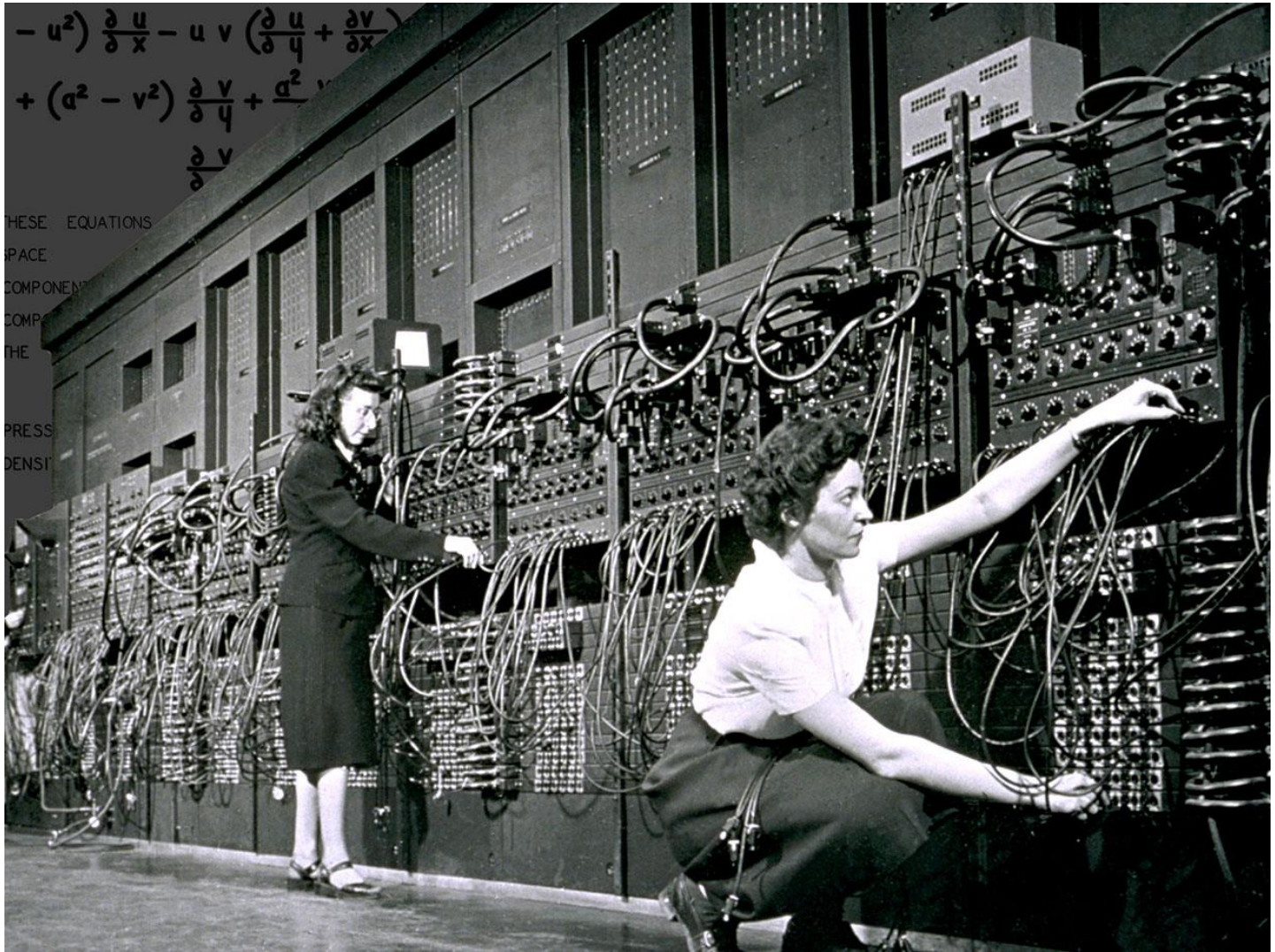
ENIAC (Electronic Numerical Integrator and Computer)

- **F**irst programmable, electronic, general-purpose digital computer.
- Task of taking a problem and feed it onto the machine was complex, and usually took weeks.
- After the program was figured out on paper, the process of getting the program into ENIAC by manipulating its switches and cables could take days.
- Although ENIAC was designed and primarily used to calculate artillery firing tables for the United States Army's Ballistic Research Laboratory (which later became a part of the Army Research Laboratory),its first program was a study of the feasibility of the thermonuclear weapon.



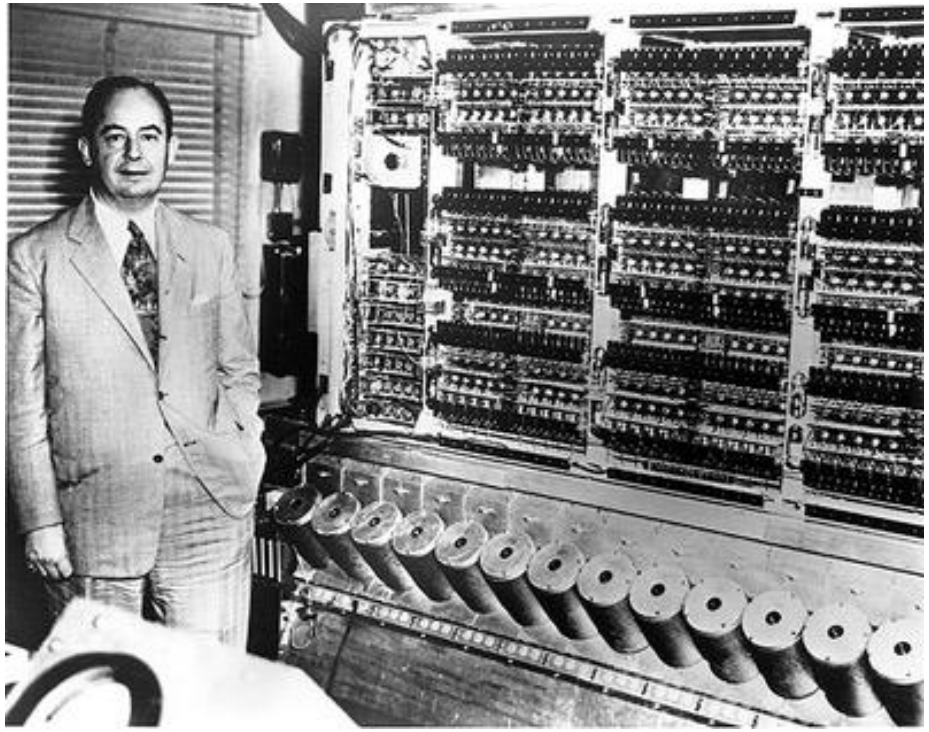
- ENIAC was completed in 1945 and first put to work for practical purposes on December 10, 1945

- ENIAC was just a large collection of arithmetic machines, which originally had programs set up into the machine by a combination of plugboard wiring and three portable function tables (containing 1200 ten-way switches each).



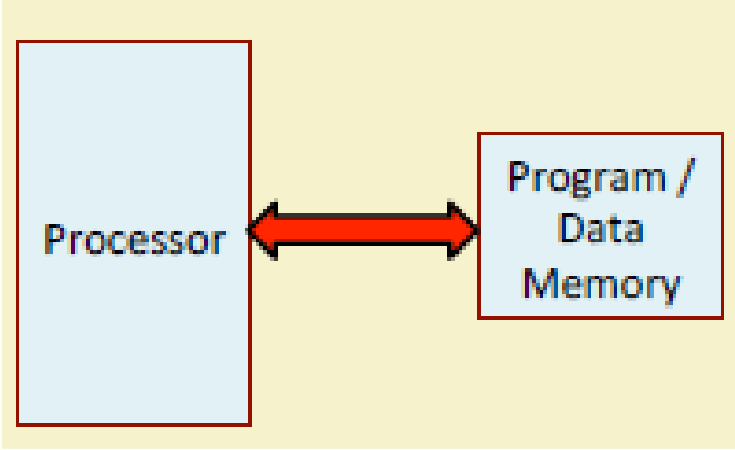
EDVAC (Electronic Discrete Variable Automatic Computer)

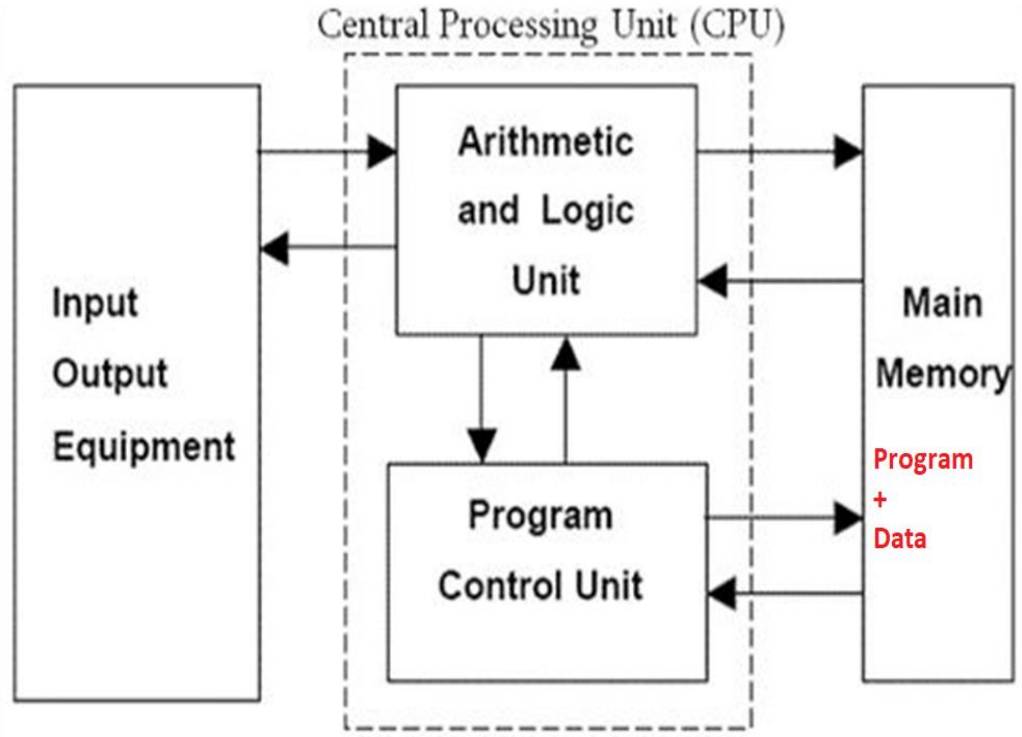
- It was binary rather than decimal, and was designed to be a stored-program computer.
- Functionally, EDVAC was a binary serial computer with automatic addition, subtraction, multiplication, programmed division and automatic checking with an ultrasonic serial memory capacity of 1,000 words.
- EDVAC's average addition time was 864 microseconds and its average multiplication time was 2,900 microseconds.
- EDVAC was designed to receive its instructions electronically; moreover, the program, coded in zeros and ones, would be kept in the same place that held the numbers the computer would be processing. This approach—letting a program treat its own instructions as data—offered huge advantages.

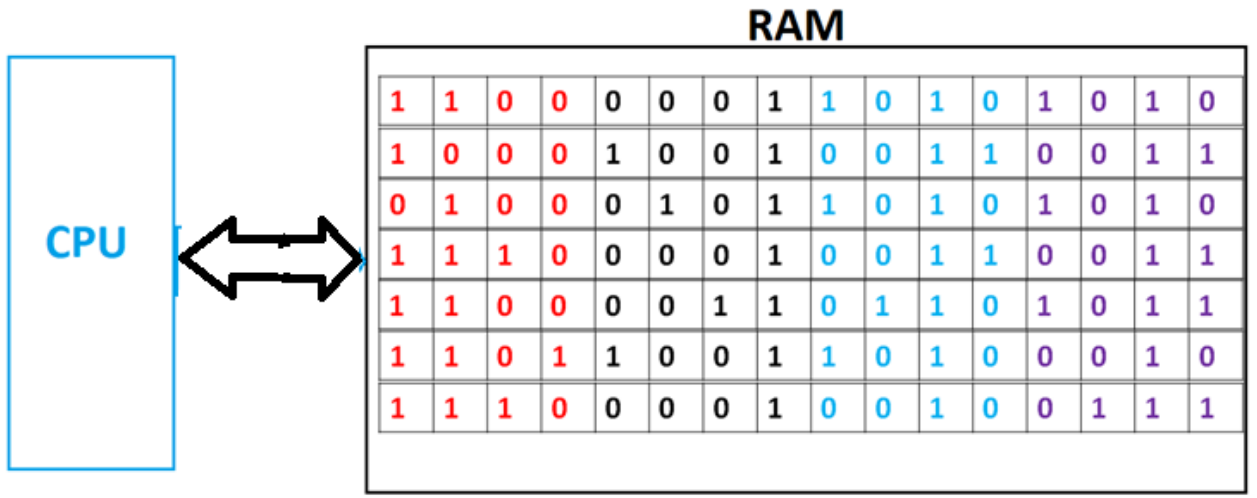


Stored program computer: Von Neumann architecture

- Both Instructions and Data are stored in RAM in binary
- Instructions (Program) are modified easily
- Single RAM holds both Instructions and Data
- Single BUS system was used to connect CPU to RAM



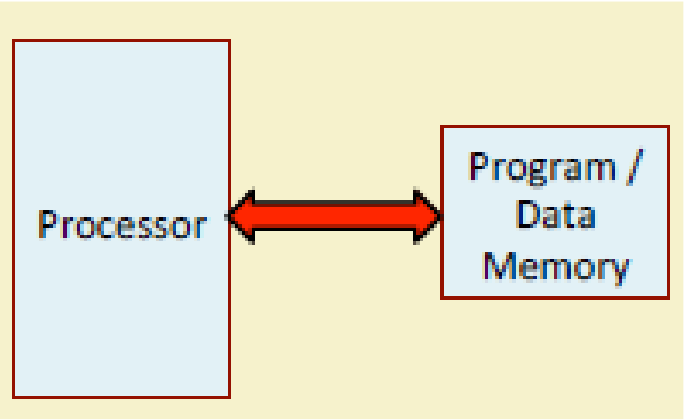




von Neumann Architecture

Key Features

- Data and instructions are both stored in a single primary storage
- Instructions are read from memory one at a time and in order (serially as it is stored/as it appears in a program).
- The processor decodes and executes one instruction at a time.
- Result is stored
- CPU reads next instruction from RAM
- This process continues till the end of the program.
- Parallel implementation of instructions is not allowed.
- Instructions can only be carried out one at a time and sequentially.
- CPU cannot read Instruction and data simultaneously due to a single bus system

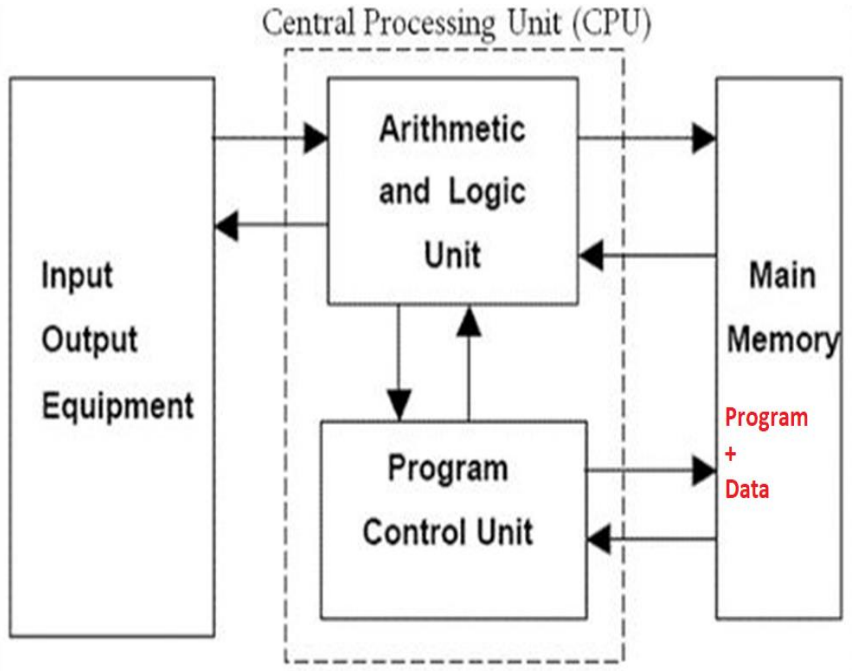


Limitations: Von Neumann Bottleneck

An instruction read/fetch and a data operation cannot occur at the same time because they share a common bus.

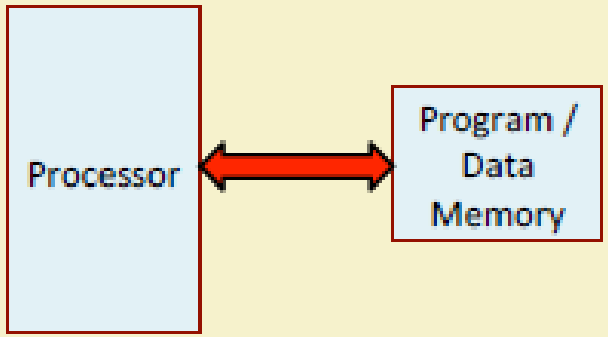
CPU has to wait and remains idle for a certain amount of time while low speed memory is being accessed.

Von Neumann bottleneck limits the performance of the system.



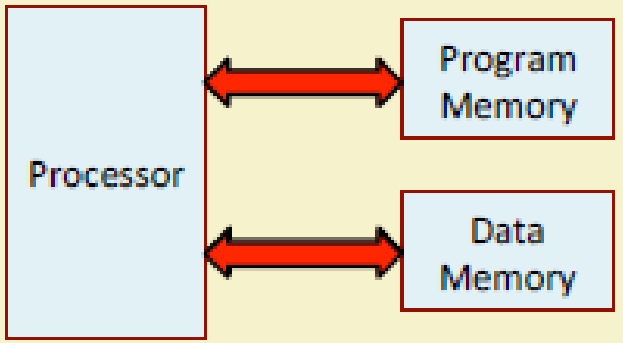
von-Neumann Architecture

- Instructions and data are stored in the same memory module.
- More flexible and easier to implement.
- Suitable for most of the general purpose processors.
- General Disadvantage:
  - The processor-memory bus acts as the bottleneck.
  - All instructions and data are moved back and forth through the pipe.

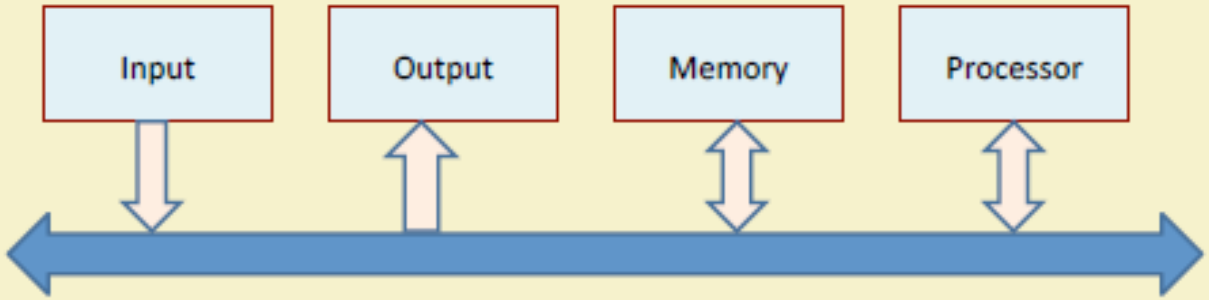


Harvard Architecture

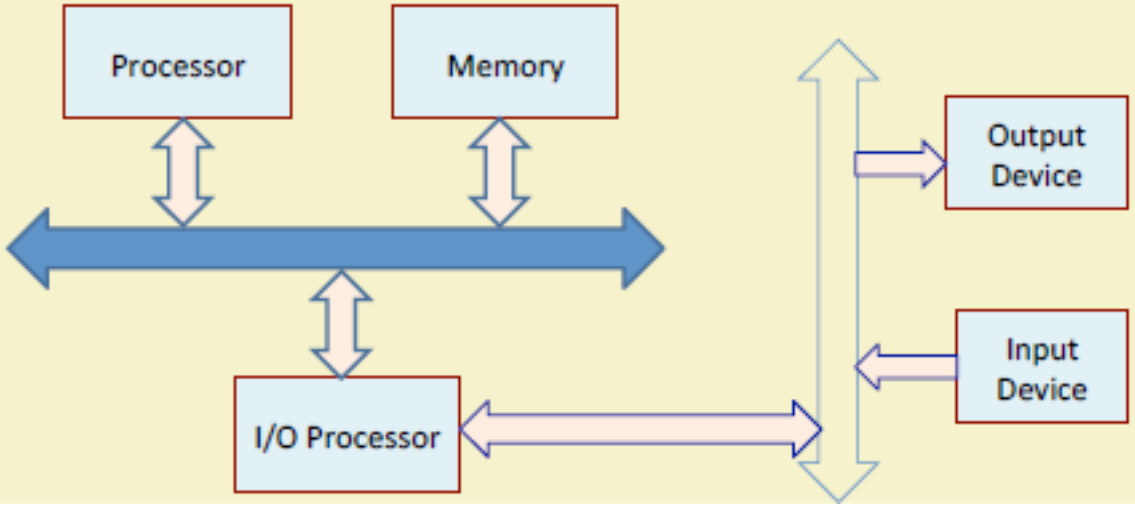
- Separate memory for program and data.
  - Instructions are stored in program memory and data are stored in data memory.
- Instruction and data accesses can be done in parallel.
- Some microcontrollers and pipelines with separate instruction and data caches follow this concept.
- The processor-memory bottleneck remains.



System-Level Single Bus Architecture



System-Level Two-Bus Architecture

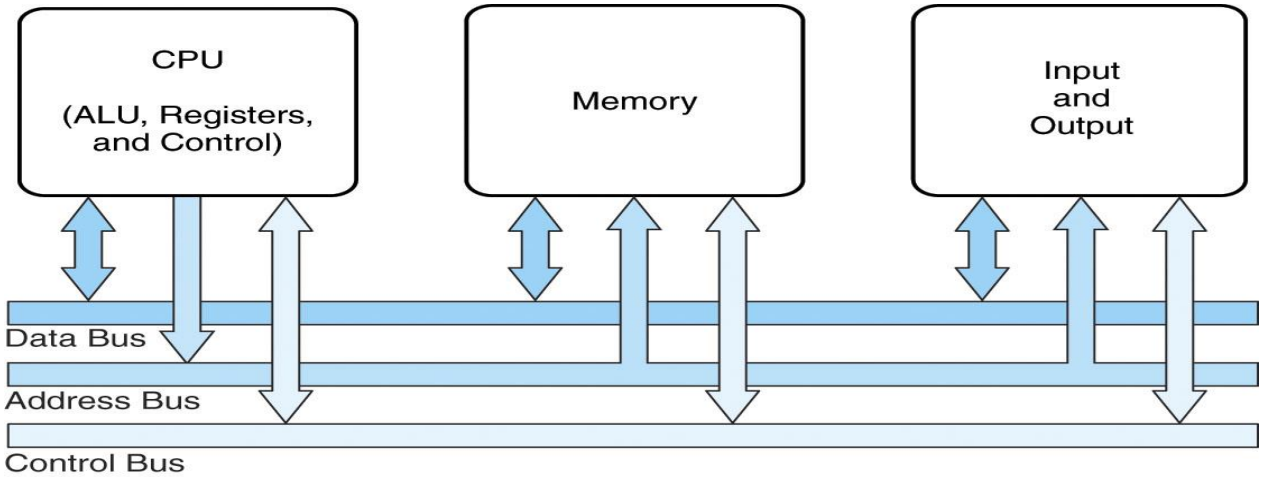


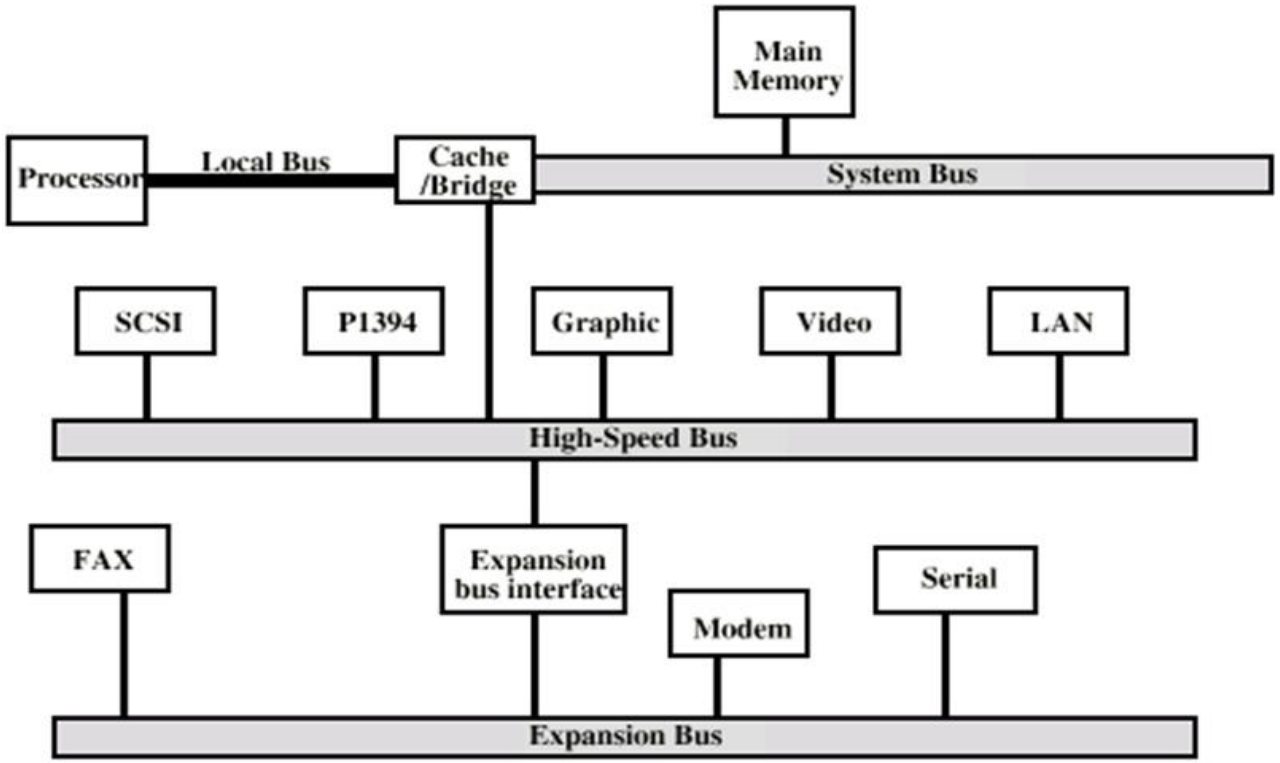
Bus Architecture

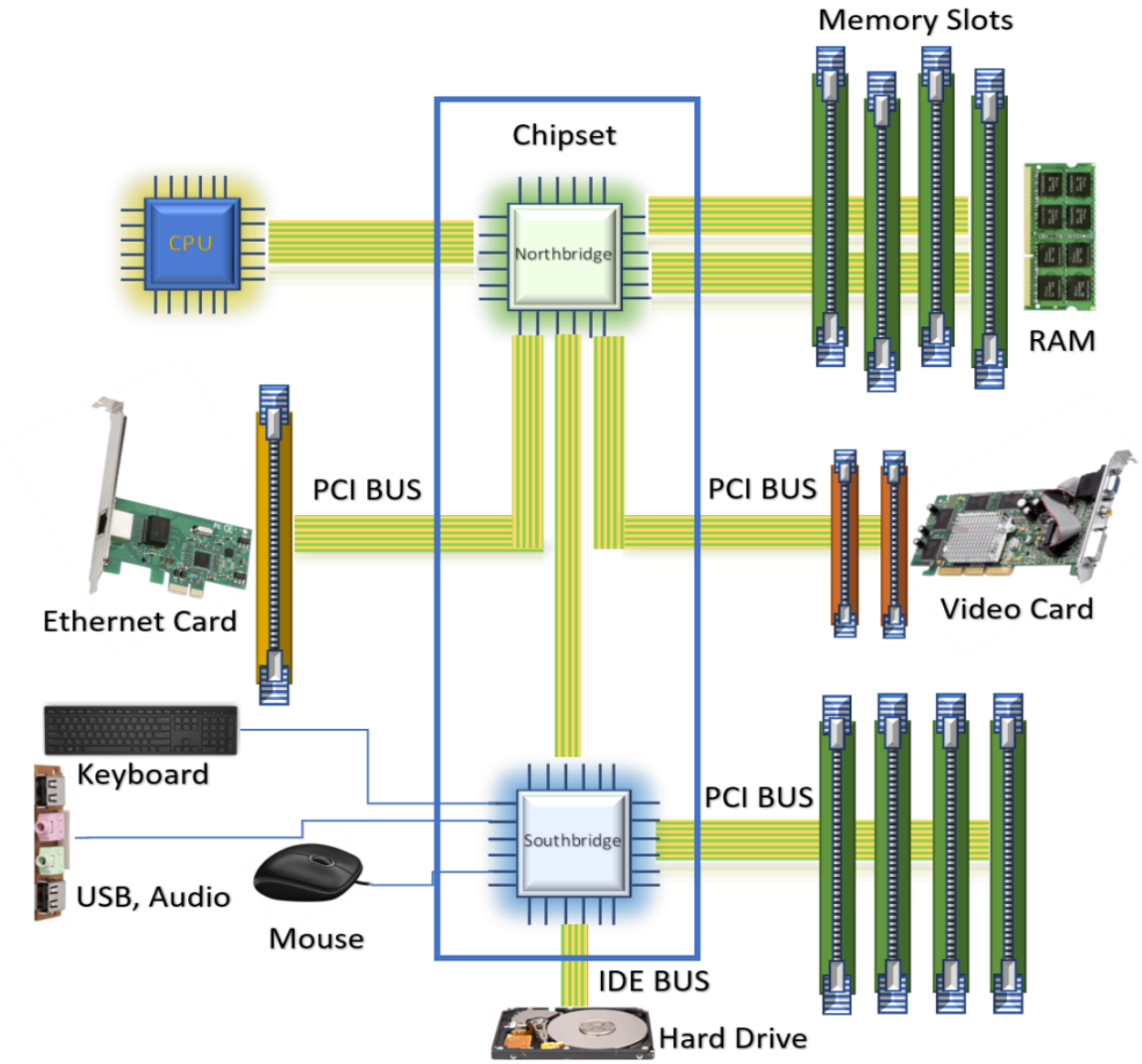
- The different functional modules must be connected in an organized manner to form an operational system.
- Bus refers to a group of lines that serves as a connecting path for several devices.
- The simplest way to connect the functional unit is to use the single bus architecture.
  - Only one data transfer allowed in one clock cycle.
  - For multi-bus architecture, parallelism in data transfer is allowed.

Multi-Bus Architectures

- Modern processors have multiple buses that connect the registers and other functional units.
  - Allows multiple data transfer micro-operations to be executed in the same clock cycle.
  - Results in overall faster instruction execution.
- Also advantageous to have multiple shorter buses rather than a single long bus.
  - Smaller parasitic capacitance, and hence smaller delay.


CPU
(ALU, Registers, and Control)
Memory
Input and Output
Data Bus
Address Bus
Control Bus


Main Memory
Processor
Local Bus
Cache /Bridge
System Bus
SCSI
P1394
Graphic
Video
LAN
High-Speed Bus
FAX
Expansion bus interface
Modem
Serial
Expansion Bus


Memory Slots
Chipset
CPU
Northbridge
RAM
PCI BUS
PCI BUS
Ethernet Card
Video Card
Keyboard
Southbridge
PCI BUS
USB, Audio
Mouse
IDE BUS
Hard Drive

- **Local bus** is a **computer bus** that connects **CPU** directly to one or more slots. The significance of direct connection to the **CPU** is avoiding the bottleneck created by the expansion **bus**, thus providing fast throughput.
- **Expansion Bus Types**
- These are some of the common expansion bus types that have been used in computers:
- **ISA** - Industry Standard Architecture
- **EISA** - Extended Industry Standard Architecture
- **MCA** - Micro Channel Architecture
- **VESA** - Video Electronics Standards Association
- **PCI** - Peripheral Component Interconnect
- PCI Express (PCI-X)
- **PCMCIA** - Personal Computer Memory Card Industry Association (Also called PC bus)
- **AGP** - Accelerated Graphics Port
- **SCSI** - Small Computer Systems Interface