Overview of computer organization and architecture

What is computer organization and architecture?

- In a computer system, hardware and many software components are fundamentally related
- Computer organization and architecture help us to understand how hardware and software interact with each other

Overview of computer organization and architecture

Computer organization

- It encompasses all physical aspects of computer systems.
  - How components are connected together?
  - How components interact with/talk to each other?
- It addresses issues such as
  - Control signals, signaling methods
  - Memory types, …
- It helps us to answer the question: **How does a computer work?**

Overview of computer organization and architecture

Computer architecture

- It focuses on the structure and behavior of the computer system
- It refers to the logical aspects of system implementation as seen by the programmer.
- It includes many elements such as
  - instruction sets and formats, data types, addressing modes, number and type of registers, …
- It helps us to answer the question: **How do I design a computer?**

Overview of computer organization and architecture

Computer organization VS architecture

- The distinction between computer organization and computer architecture is not clear-cut.
- These are strongly interrelated and interdependent.
- We can truly understand each of them only when we comprehend both of them.

The main components of a computer

From Software to Hardware

- Computer scientists design algorithms
- Computer scientists implement an algorithm by using high level programing language (Java, C, etc.)
- Another algorithm runs this algorithm and another one runs that algorithm and so on
- We finally get down to machine level
- Machine level can be thought of as an algorithm implemented as an electronic device

The main components of a computer

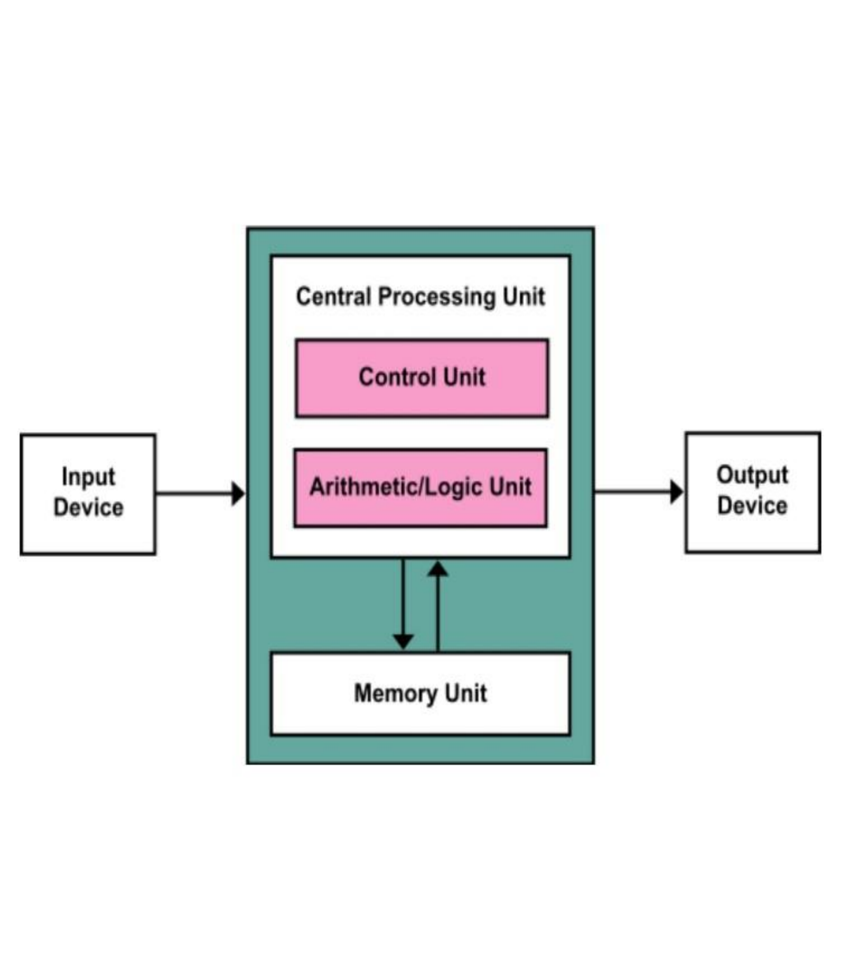
Principle of Equivalence of Hardware and Software

- *Anything that can be done with software can also be done with hardware, and anything that can be done with hardware can also be done with software*
- We implement an application in Hardware or Software level
- Our knowledge of computer organization and architecture will help us to make the best choice

Computer Hardware main components

At the most basic level, a computer is a device consisting of 3 pieces:

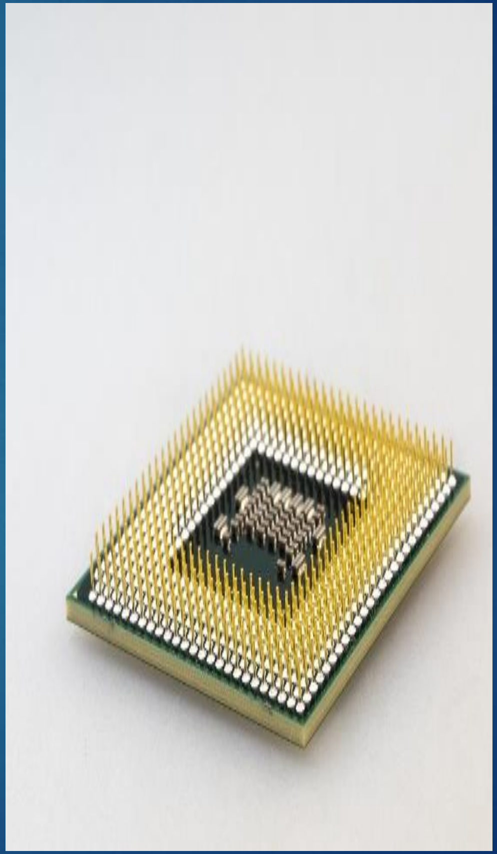
- A processor to interpret and execute programs
- A memory to store both data and programs
- A mechanism for transferring data to and from the outside world



Central Processing Unit (CPU)

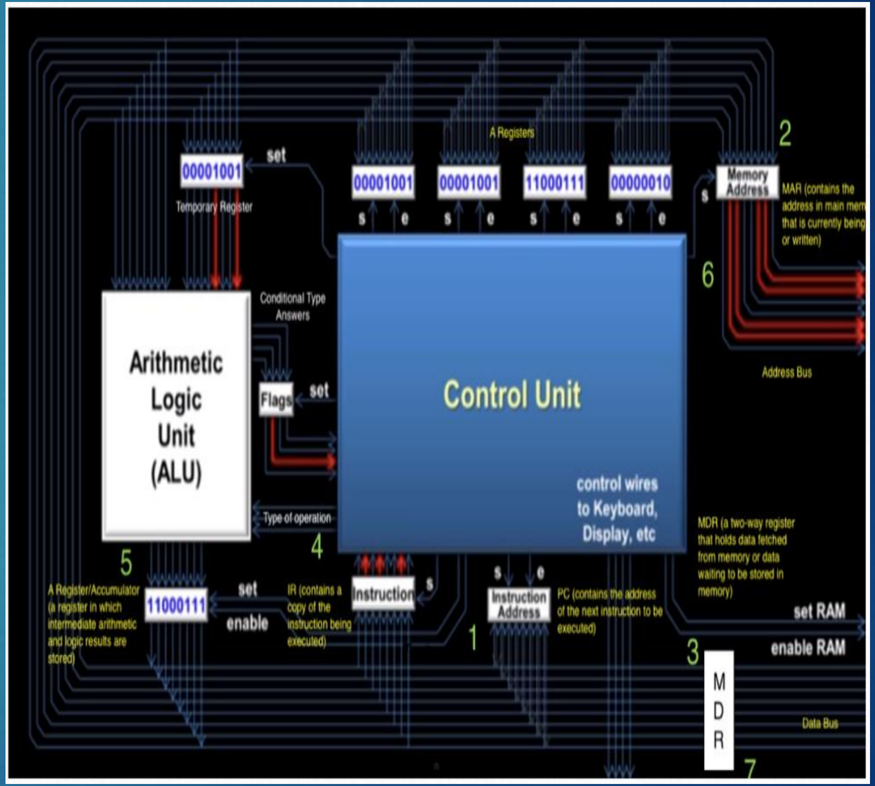
- The central processing unit (CPU) is responsible for fetching the program instructions, decoding each instruction that is fetched, and performing the indicated sequence of operations on the correct data.

- CPU acts like the "brain" of a computer system. It contains the circuitry to interpret and execute program instructions.



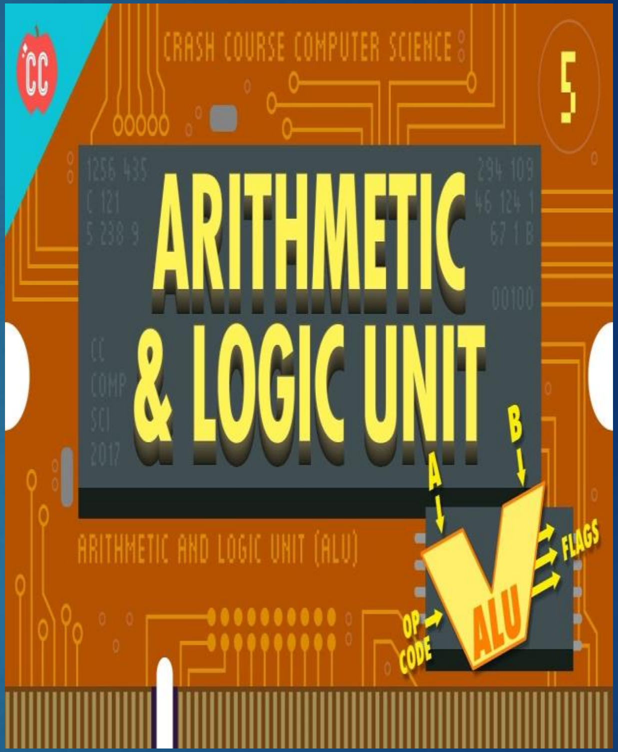
All computers have a CPU that can be divided into two pieces.

- The first is the datapath, which is a network of storage units (registers) and arithmetic and logic units (for performing various operations on data) connected by buses (capable of moving data from place to place) where the timing is controlled by clocks.
- The second CPU component is the control unit, a module responsible for sequencing operations and making sure the correct data are where they need to be at the correct time.


A Registers
00001001
set
Temporary Register
00001001
00001001
11000111
00000010
s
e
2
Memory Address
MAR (contains the address in main mem
that is currently being or written)
6
Conditional Type Answers
Arithmetic Logic Unit (ALU)
Flags
set
Control Unit
Address Bus
control wires to Keyboard, Display, etc
Type of operation
4
5
MDR (a two-way register that holds data fetched from memory or data waiting to be stored in memory)
A Register/Accumulator (a register in which intermediate arithmetic and logic results are stored)
11000111
set
enable
IR (contains a copy of the instruction being executed)
Instruction
Instruction Address
PC (contains the address of the next instruction to be executed)
1
set RAM
enable RAM
3
MDR
Data Bus
7

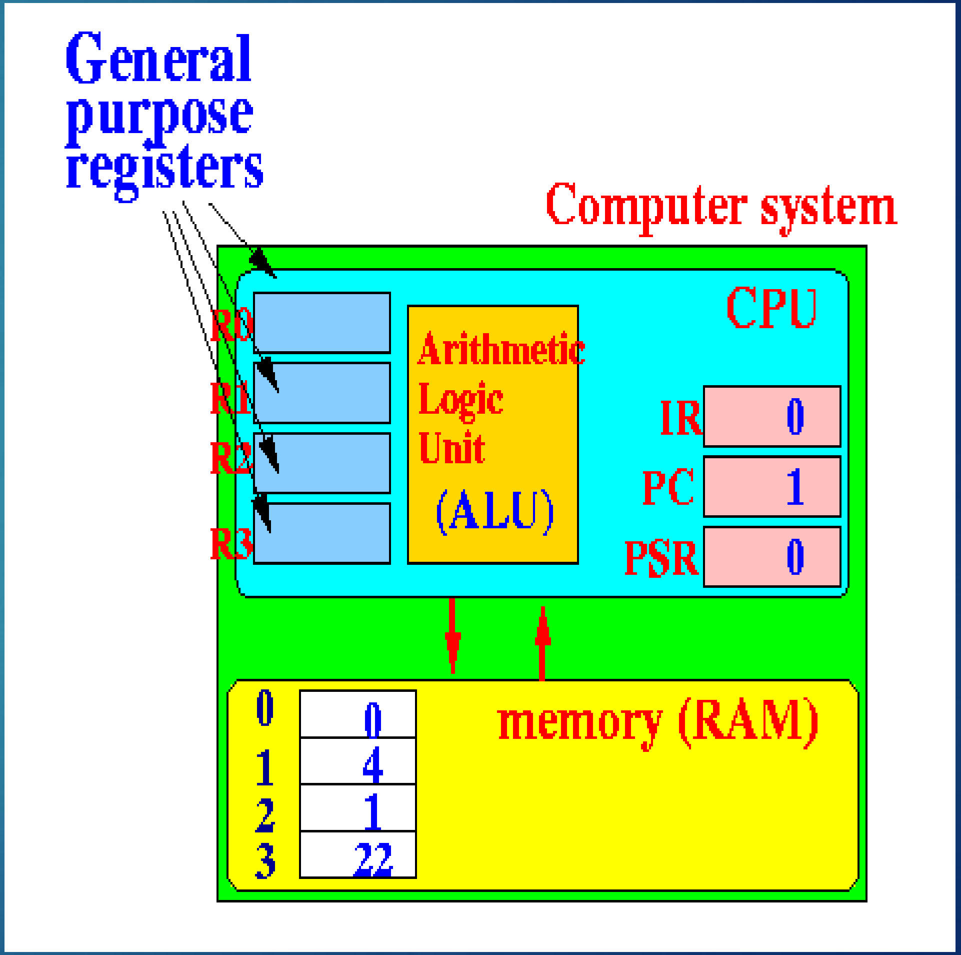
Arithmetic/Logic Unit (ALU)

- ALU does all mathematical operations (arithmetic) (+/-/compare) and logical (AND/OR) calculations.
- Some computers have two ALUs to process two calculations simultaneously. That is called **dual core** technology.
- ALUs contain several special storage units called registers.


CRASH COURSE COMPUTER SCIENCE
5
ARITHMETIC
& LOGIC UNIT
ARITHMETIC AND LOGIC UNIT (ALU)
A
B
FLAGS
OP CODE
ALU

Register

- Register is a small, very fast storage area inside CPU.
- It is used to store intermediate values from calculations or instructions that is needed again immediately.
- For instance, when a ALU is commanded to calculate A*(B+C), ALU needs to calculate B+C first, then ALU need to store the result for a moment and use the result to multiply A.
- It is faster for ALU to access register than store the data in memory units.

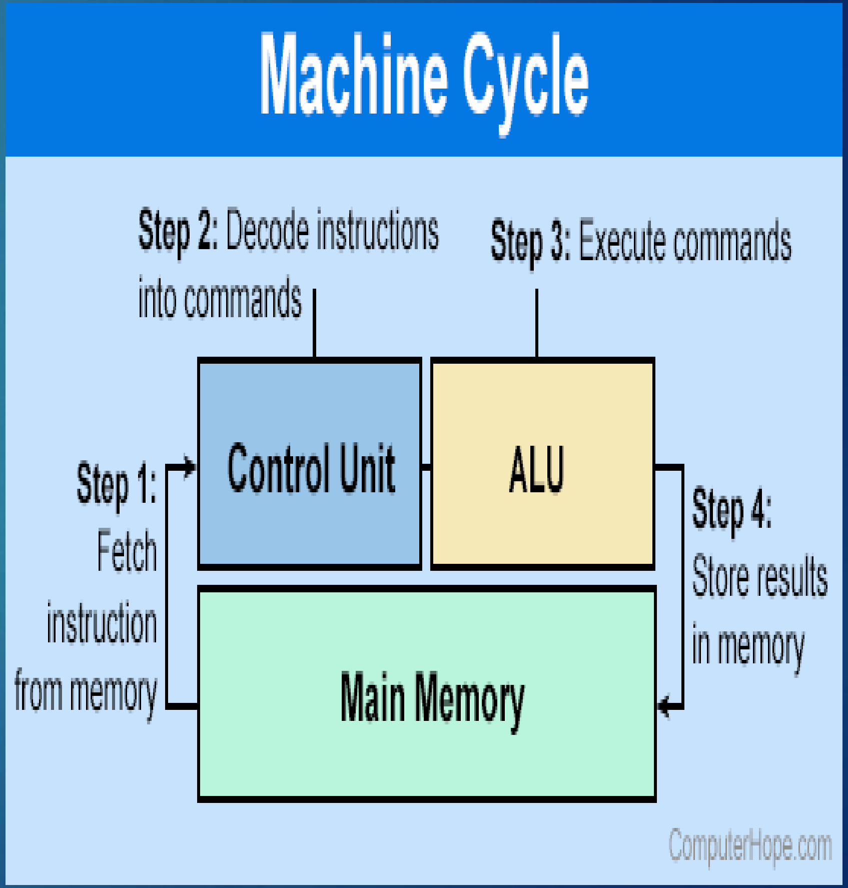


Four types of Register

- MAR (memory address register): store memory that currently be accessed inside RAM
- MDR/MBR (memory data/buffer register): store data from memory
- Accumulator (A register): a register stores **immediate** result from ALU
- Program Counter (PC): store the address of the next instruction

Control Unit (CU)

- CU acts like a manager in a computer.
- CU receives orders from RAM in the form of instruction and decode (break) that instruction down into specific commands for other components inside computer system.
- It directs the data flow and the operation of the ALU.
- Together, these components perform the tasks of the CPU: fetching instructions, decoding them, and finally performing the indicated sequence of operations.
- The performance of a machine is directly affected by the design of the datapath and the control unit.


Machine Cycle
Step 2: Decode instructions into commands
Step 3: Execute commands
Step 1: Fetch instruction from memory
Control Unit
ALU
Step 4: Store results in memory
Main Memory
ComputerHope.com

How a CPU works?

1. CU turn on enable wire of **Instruction Address Register(Program Counter)** to allow the next Instruction address to flows to RAM.
2. Instruction address first goes to **Memory Address Register** which only tells RAM what address the CPU wants next. CU turns on the set wire of MAR, then memory address goes to RAM. CU turns on the enable RAM wire.
3. Then data goes to **Memory Data Register(buffer)** from RAM via data bus. Then data (instruction) goes to Instruction Register.
4. CU turns on the set **Instruction Register(store instruction)** wire. Instruction goes to CU. CU decodes instruction, tells ALU the type of operation ALU need to perform and stores the input in Temporary registers by turning on the set wire.

How a CPU works?

5. ALU operates the instruction. CU turns on the set wire of Accumulator and instructs ALU to store the immediate result in **Accumulator (A Register)**. CU turns on the enable wire of Accumulator to instruct Accumulator to output the result to the system bus.
6. **CU** send address to **MAR** again and points out an address in RAM.
7. The result stored temporarily in A Register will go to the RAM via data bus.
8. Data will be stored in the specified address in RAM.
9. CPU is ready for the next instruction. Repeat steps 1-6.

How a CPU works?

You may watch this video for a simple explanation of how a cpu works.

https://www.youtube.com/watch?v=RIrQwd_srRw

Memory

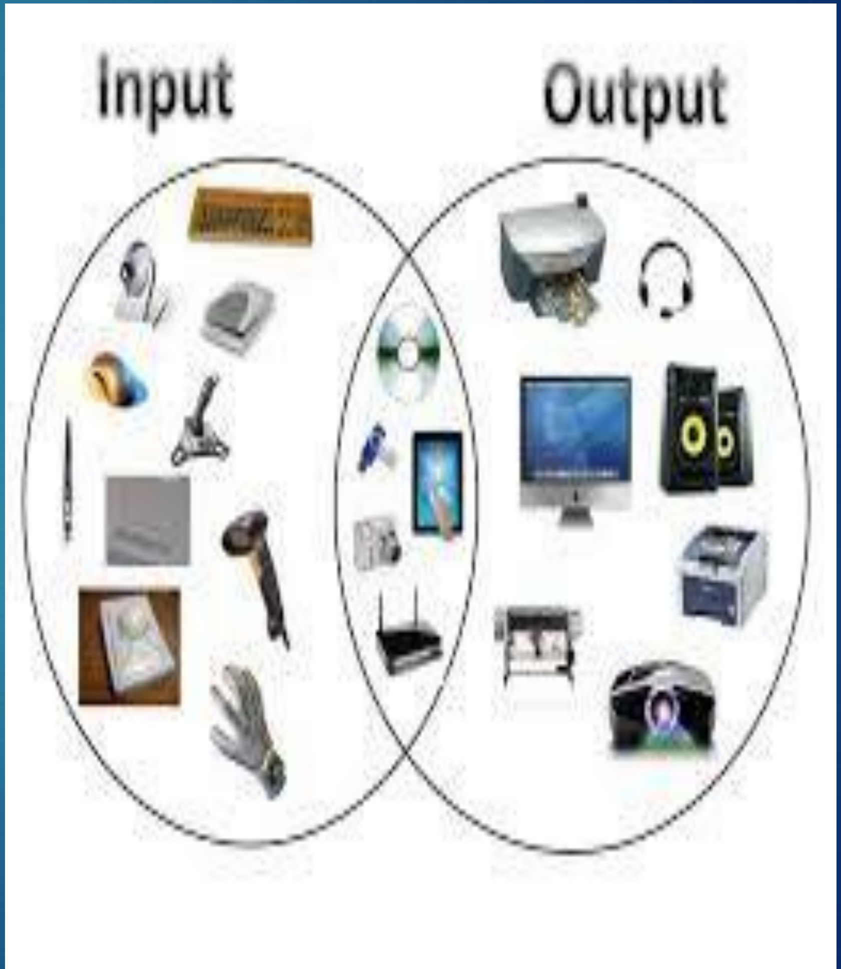
- A memory is just like a human brain.
- It is used to store data and instructions.
- Is the storage space in the computer, where data is to be processed and instructions required for processing are stored.
- The memory is divided into large number of small parts called cells. Each location or cell has a unique address, which varies from zero to memory size minus one. For example, if the computer has 64k words, then this memory unit has 64 * 1024 = 65536 memory locations. The address of these locations varies from 0 to 65535.

Memory

- Primary memory holds only those data and instructions on which the computer is currently working.
- It has a limited capacity and data is lost when power is switched off.
- It is generally made up of semiconductor device.
- Are not as fast as registers.

Input / Output Devices

- Input and output devices allow the computer system to interact with the outside world by moving data into and out of the system.
- Input/output devices are usually called I/O devices.
- They are directly connected to an electronic module inside the systems unit called a device controller.
- For example, the speakers of a multimedia computer system are directly connected to a device controller called an audio card (such as a SoundBlaster), which in turn is connected to the rest of the system.



Input device

An **input device** is used to bring data into the system.

Some input devices are:

- Keyboard
- Mouse
- Microphone
- Bar code reader
- Graphics tablet
- Web Camera
- Scanner
- Joystick

Output device

An **output device** is used to send data out of the system.

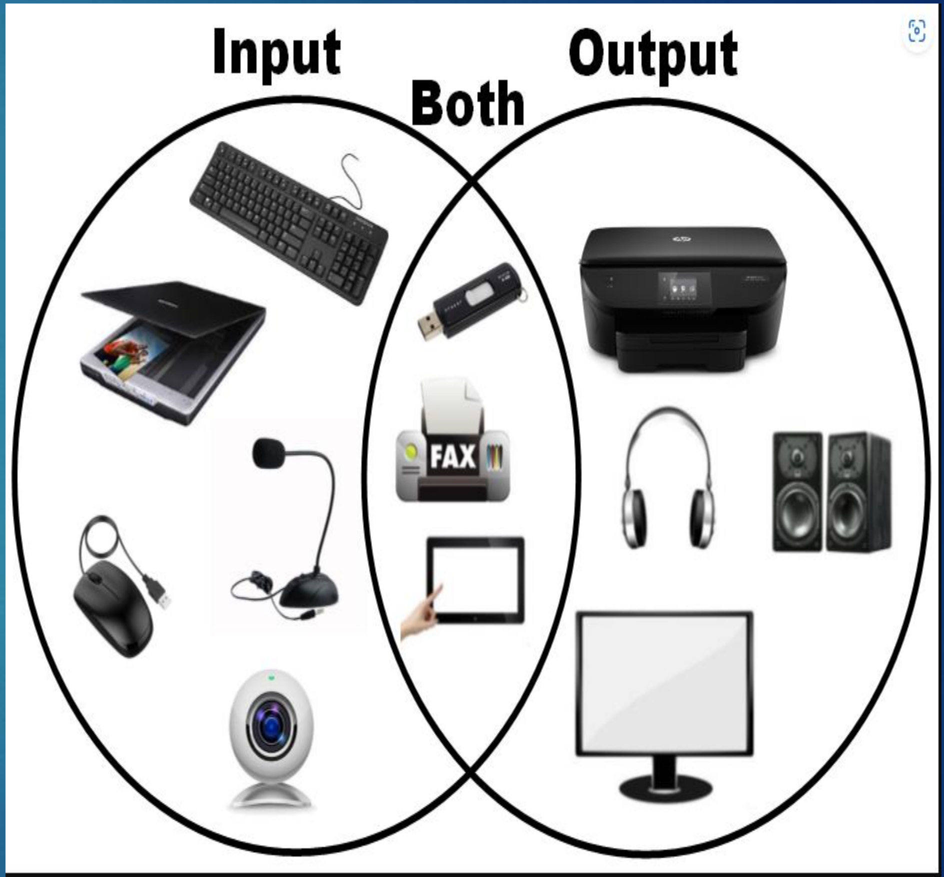
Some output devices are:

- Monitor
- Printer
- Speaker
- Projector

OUTPUT DEVICES
MONITOR
PRINTER
SPEAKER
HEADPHONE
PROJECTOR

Combined Input/Output Devices

- **Touchscreen**: Acts as both an input device (when you touch it) and an output device (displaying information).
- **External Hard Drive**: Can receive data from a computer (input) and send data to a computer (output).
- **USB Flash Drive**: Similar to an external hard drive, it can both receive and send data.



Standards Organizations

There are several key standard organizations that play a crucial role in developing and maintaining standards for computers and related technologies.

Aim

- Establish common guide lines for a particular type of equipment to ensure a “worldwide”[1] interoperability (compatibility)

Some international standards organizations

- **ISO (International Organization for Standardization)**
  - Develops and publishes international standards across various industries, including computing and technology.

- **IEC (International Electrotechnical Commission)**
  - Focuses on international standards for electrical, electronic, and related technologies.

Some international standards organizations

IEEE (Institute of Electrical and Electronic Engineers)

- sets standards for various computer components, signaling protocols, and data representation

ITU (International Telecommunications Union)

- Sets standards for telecommunications systems, including telephone, telegraph, and data communication systems

Some international standards organizations

- ANSI (American National Standards Institute)
  - Oversees the development of voluntary consensus standards for products, services, processes, and systems in the U.S.
- ETSI (European Telecommunications Standards Institute)
  - Produces globally applicable standards for information and communications technologies, including fixed, mobile, radio, and internet technologies.

Some international standards organizations

CEN (European Committee for Standardization)

- Develops standards in various sectors, including information technology

Historical Development of Computers

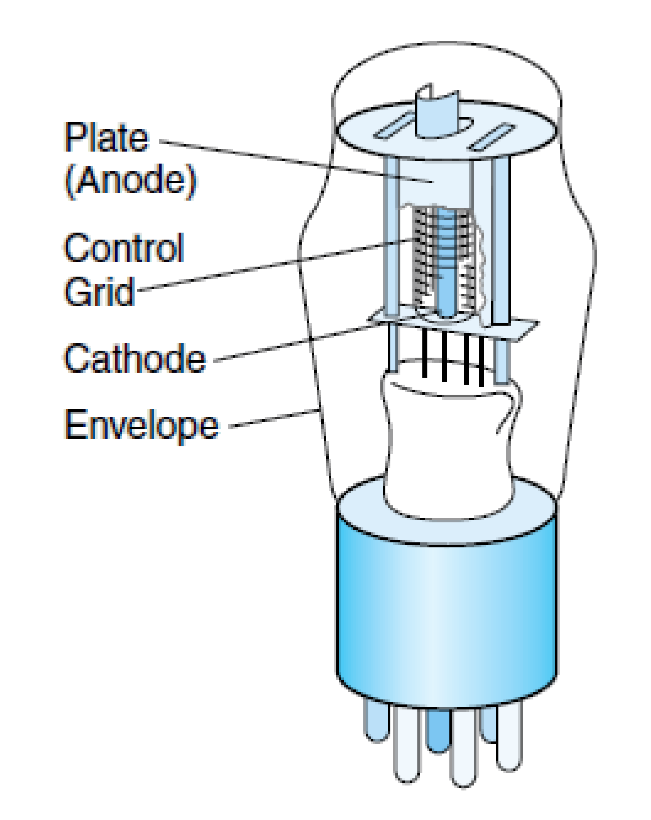
Generation Zero: Mechanical Calculating Machines (1642 – 1945)

- Use of mechanical technology to do calculations
- Suggest the use of binary number system rather than the decimal number system

Historical Development of Computers

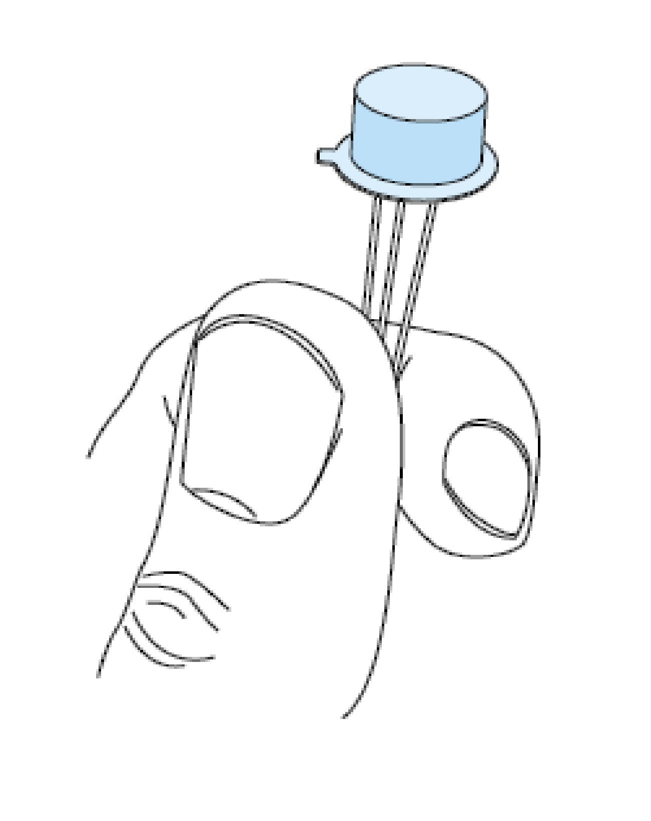
First Generation: Vacuum tube computers (1945–1953)

- Use of electrical/electronic technology (much faster than mechanical technology)
- Invent binary machines built from vacuum tubes
- Invent vaccum tubes diodes and triodes
- Disadvantages: Bulky systems, Power consumption and heat dissipation.



Historical Development of Computers

The Second Generation: Transistorized Computers (1954–1965)

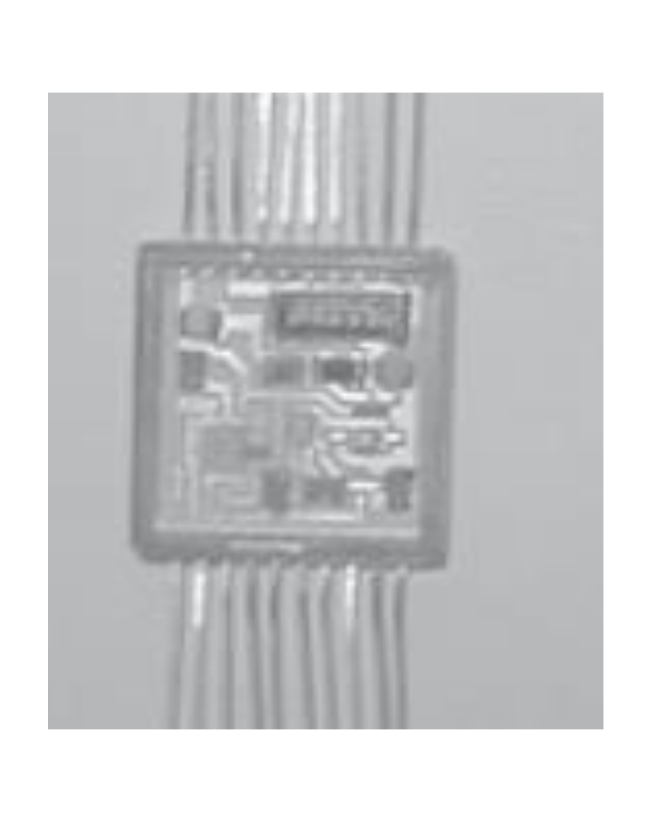


- ➢ Transistors revolutionize computers
  - Transistors consume less power than vacuum tubes, are smaller, and work more reliably,
  - the circuitry in computers became smaller and more reliable.

Historical Development of Computers

The Third Generation: Integrated Circuit Computers (1965–1980)

- ➢ Integrating multiple transistors in a single silicon/germanium chip
- ➢ Explosion in computer use
- ➢ Computers became faster, smaller, and cheaper, bringing huge gains in processing power



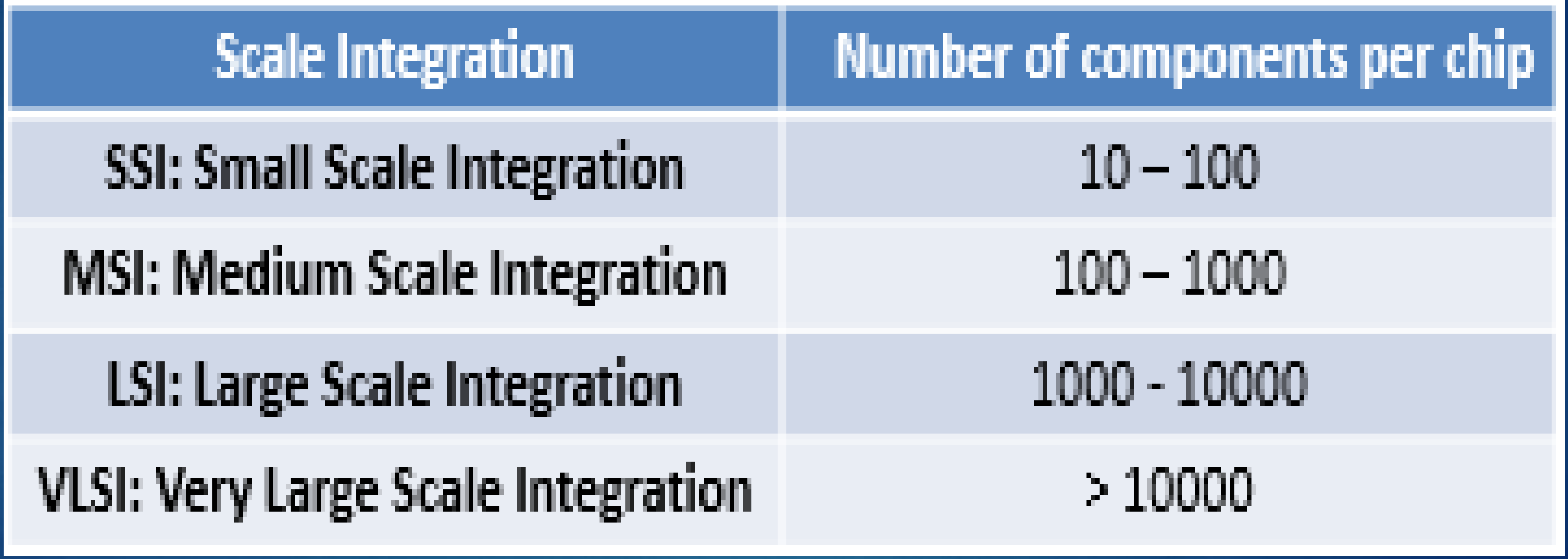
Historical Development of Computers

The Fourth Generation: VLSI Computers (>1980)

- More integration, more transistors on a single silicon chip
- Computers became smaller: Appearance of micro-computers
- Increasing the processing power of all computers types (also supercomputers and main frame computers)

Historical Development of Computers

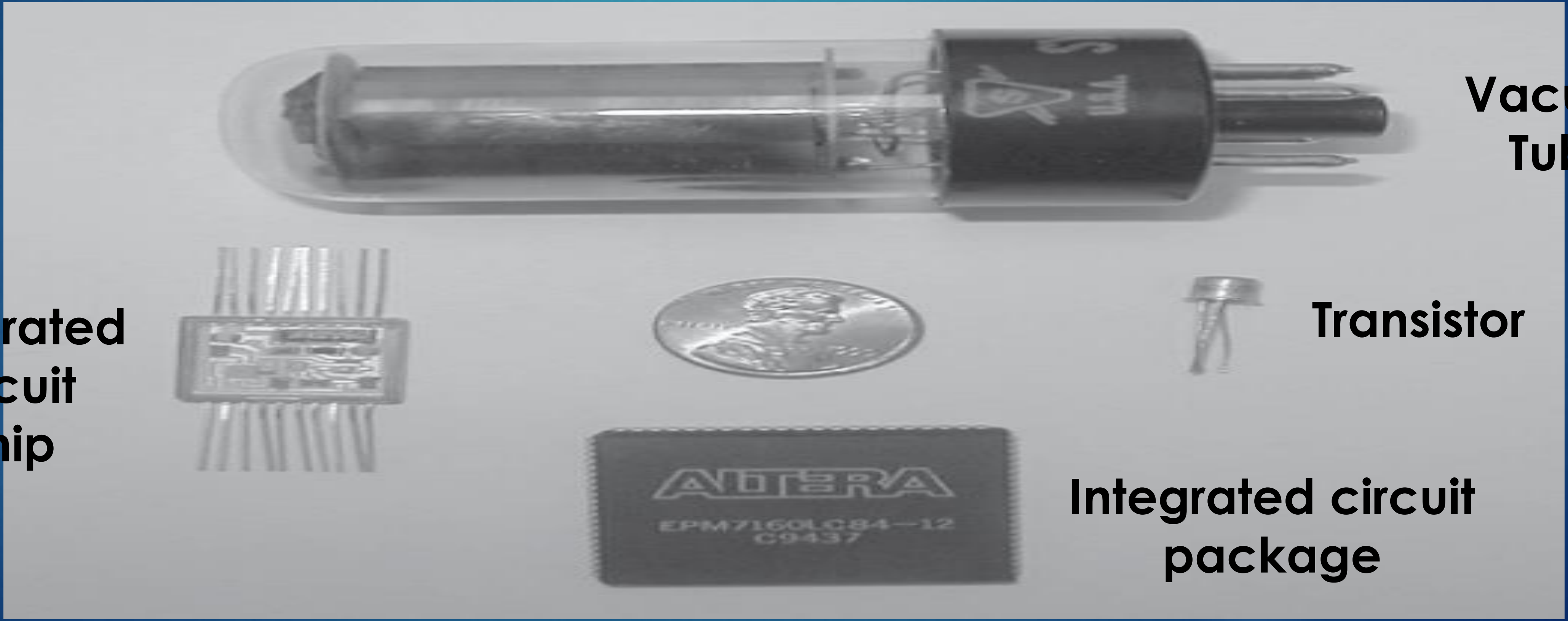
The Fourth Generation: VLSI Computers (>1980)



Scale Integration	Number of components per chip
SSI: Small Scale Integration	10 – 100
MSI: Medium Scale Integration	100 – 1000
LSI: Large Scale Integration	1000 - 10000
VLSI: Very Large Scale Integration	> 10000

Historical Development of Computers

Size comparison



Historical Development of Computers

Moore’s Law (Gordon Moore – Intel Founder, 1965)

- The density of transistors in an integrated circuit doubles every 18 months (1.5 year).

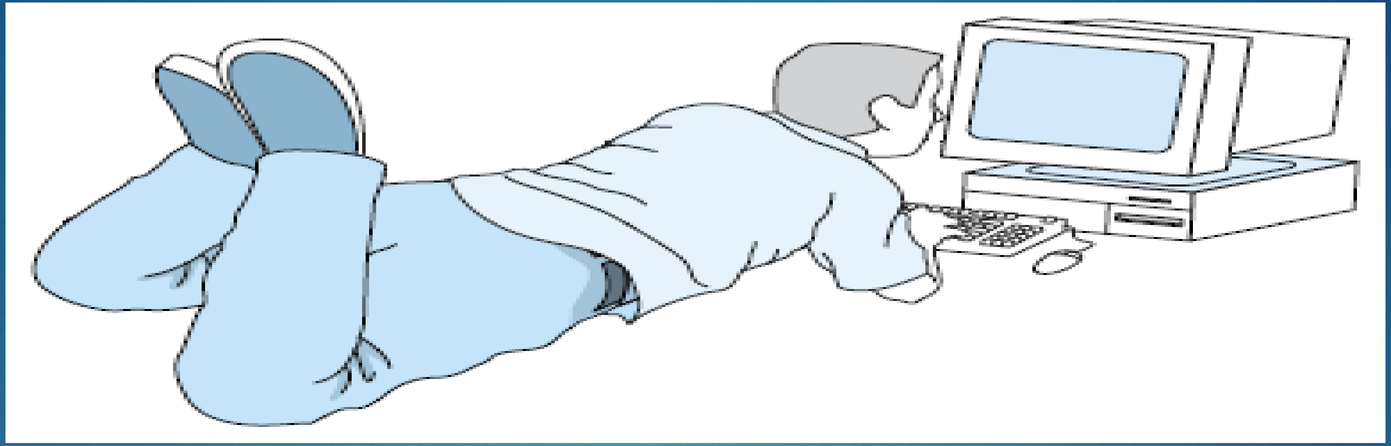
Rock’s Law (early Intel capitalist Arthur Rock)

- The cost of capital equipment to build semiconductors will double every four years

Historical Development of Computers

- If Moore's Law is to hold, Rock's Law must fall or computers must shift to a radically different technology
- Research into new computing paradigms has been proceeding in earnest during the last half decade

The Computer Level Hierarchy

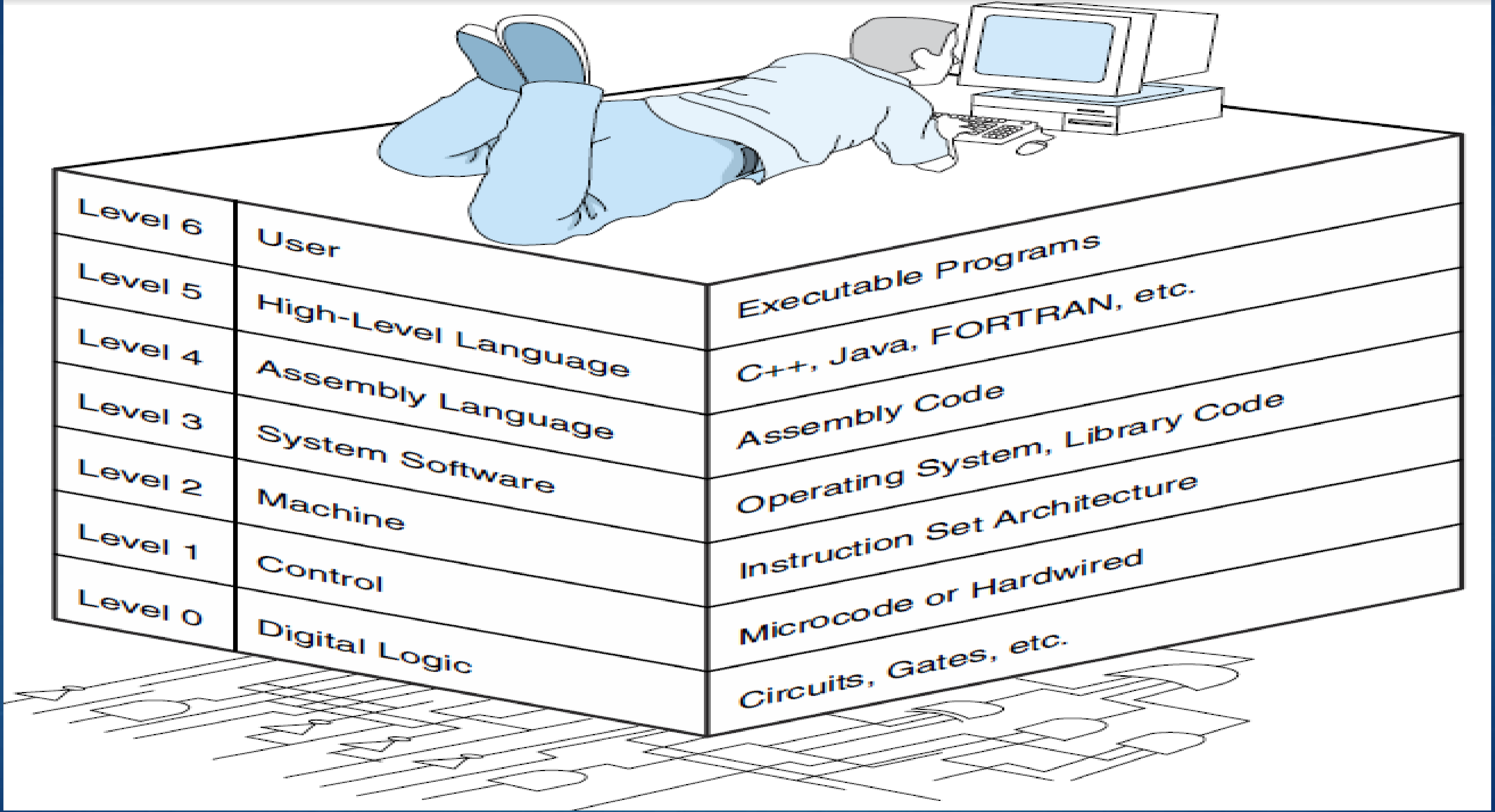


- The user executes programs on a PC (Paint, word files, games, etc.)
- The user is outside the computer! He uses input and output devices to communicate with the computer.
- Now, what happens INSIDE the computer?
- To understand, we will use a “divide and conquer” approach.

The Computer Level Hierarchy

- Imagine the machine (computer) as a hierarchy of levels, in which each level has a specific function.
- The highest level – Level 6 – is the "user's level"
  - Level 6 is composed of applications
  - User runs programs such as word processors, graphics packages, or games.
- The lower levels are unseen by the user those can be considered as "virtual machines".
- Let us discover these "virtual machines".

The Computer Level Hierarchy



Level 6	User	Executable Programs
Level 5	High-Level Language	C++, Java, FORTRAN, etc.
Level 4	Assembly Language	Assembly Code
Level 3	System Software	Operating System, Library Code
Level 2	Machine	Instruction Set Architecture
Level 1	Control	Microcode or Hardwired
Level 0	Digital Logic	Circuits, Gates, etc.

The Computer Level Hierarchy

Level 5: High-Level Language Level

- Consists of languages such as C, C++, FORTRAN, Lisp, Pascal, and Prolog.
- Programmers write programs at this level.
- Compilers translates these languages to a language the machine can understand (that lower levels could understand): Assembly then machine languages.

The Computer Level Hierarchy

Level 4: Assembly Language Level

- More "machine dependent" language.
- Assembly language is then one to one translated to machine language (one assembly language instruction is translated to exactly one machine language instruction).

The Computer Level Hierarchy

Level 3: System Software Level

- Deals with operating system instructions (multiprogramming, protecting memory, synchronizing processes, and various other important functions)
- Instructions translated from assembly language to machine language are passed through this level unmodified

The Computer Level Hierarchy

Level 2: Instruction Set Architecture (ISA), or Machine Level

- Machine language recognized by the particular architecture of the computer system
- Programs written in machine language can be executed directly by the electronic circuits without any interpreters, translators, or compilers.

The Computer Level Hierarchy

Level 1: The Control Level

- Is where a *control unit do its job*
  - Receives machine instructions from the level above
  - decodes and executes those instructions properly
  - Moves data to where and when it should be
- The control unit interprets the machine instructions

The Computer Level Hierarchy

Level 0: The Digital Logic Level

- is where we find the physical components of the computer system: the gates and wires

The Von Neumann Model

- A computer architecture model published by a famous Hungarian mathematician named John von Neumann
- The idea is to store programs' instructions inside a main memory in order to avoid rewiring the system each time it had a new problem to solve, or an old one to debug.
- All stored-program computers have come to be known as *von Neumann systems* using the *von Neumann architecture*

The Von Neumann Model

The Von Neumann architecture

It satisfies at least the following characteristics

- ➢ Consists of three hardware systems
- ➢ A *central processing unit* (CPU) with a control unit, an *arithmetic logic unit* (ALU), *registers* (small storage areas), and a program counter;

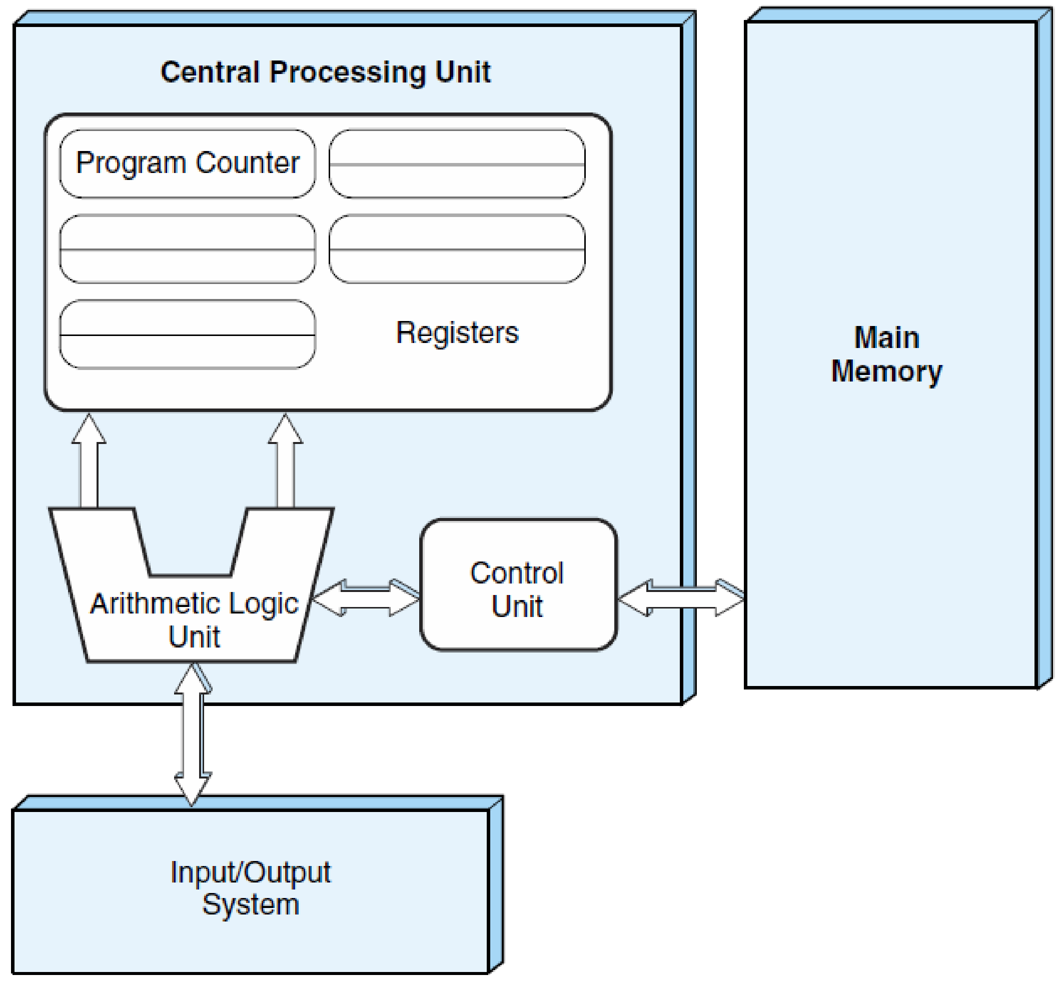


FIGURE 1.4 The von Neumann Architecture

The Von Neumann Model

The Von Neumann architecture

It satisfies at least the following characteristics

- a *main-memory system,* which holds programs that control the computer's operation;
- and an *I/O system.*

➢ Capacity to carry out sequential instruction processing

➢ Contains a single path, between the main memory system and the control unit of the CPU

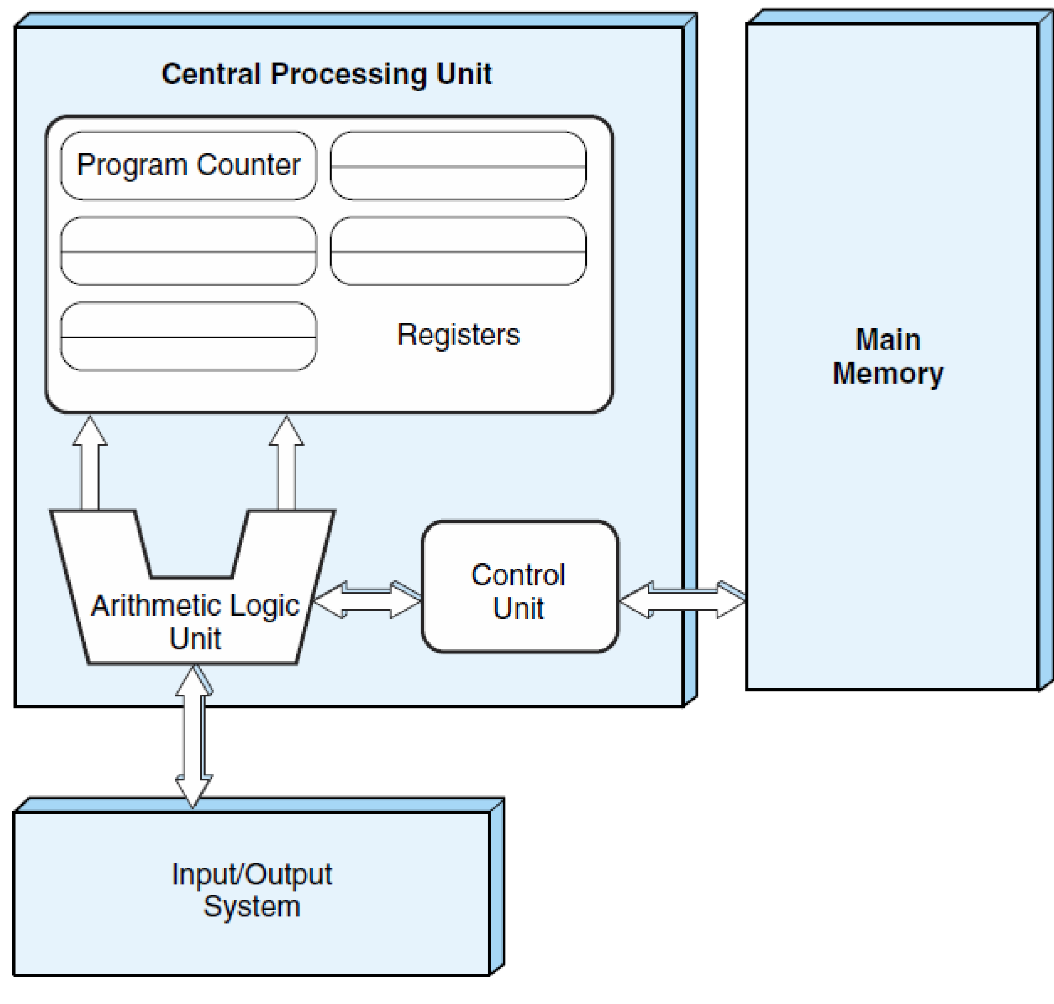


FIGURE 1.4 The von Neumann Architecture

The Von Neumann Model

- Program instructions are stored inside the main memory
- The machine runs the programs sequentially (instruction per instruction – machine instruction)
- Each machine instruction is fetched, decoded and executed during one cycle known as the *von Neumann execution cycle* (also called the *fetch-decode-execute cycle*)

The Von Neumann Model

One iteration of the cycle is as follows:

- **1.** The control unit fetches the next program instruction from the memory, using the program counter to determine where the instruction is located.
- **2.** The instruction is decoded into a language the ALU can understand.
- **3.** Any data operands required to execute the instruction are fetched from memory and placed into registers within the CPU.
- **4.** The ALU executes the instruction and places the results in registers or memory.

The Von Neumann Model

1. Fetch
   - PC indicates the iteration number
   - CU fill the instruction register
2. Decode
   - what ALU should do (add, multiply, ...)?
   - Fill registers with needed data

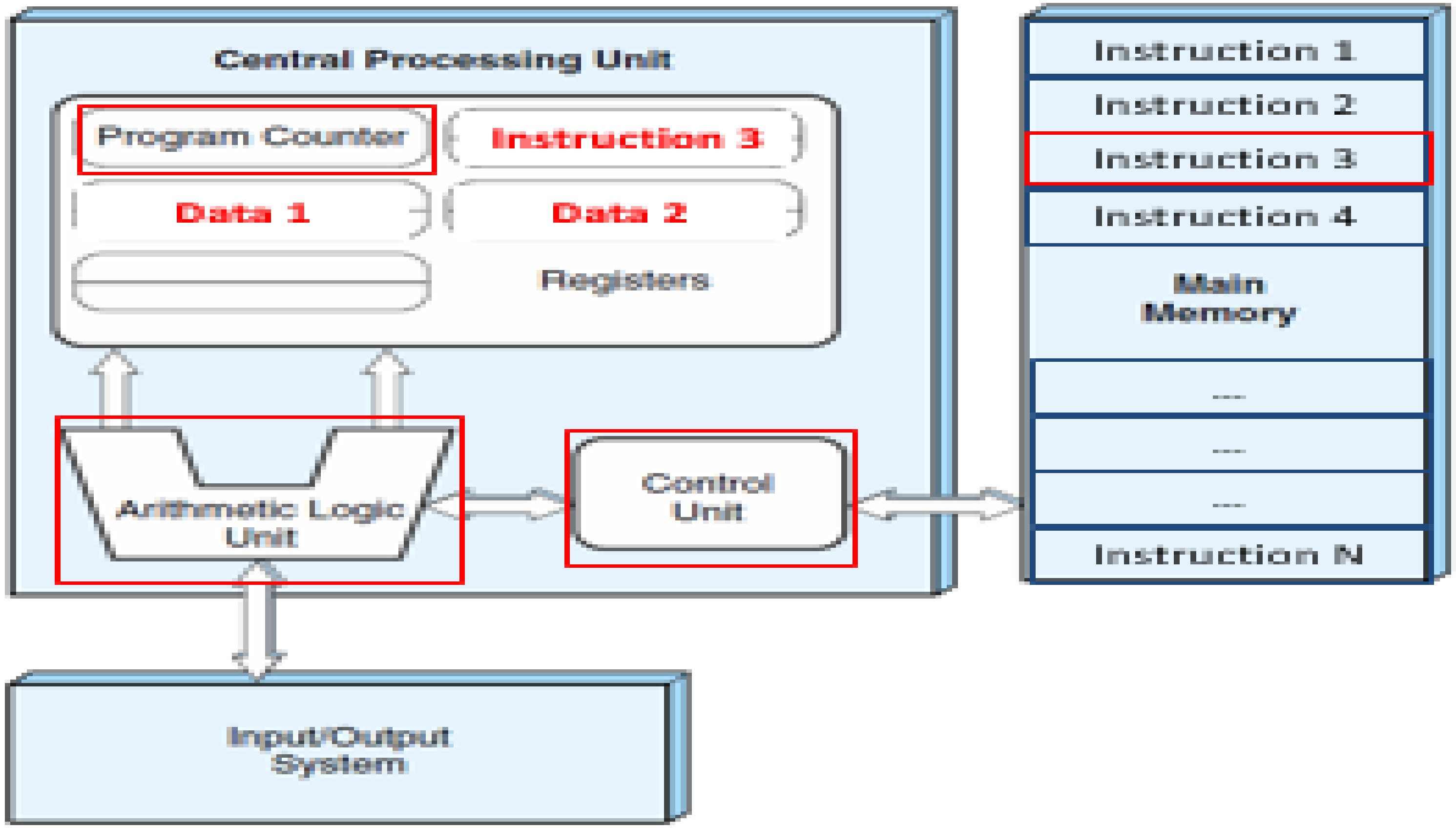


FIGURE 1.4 The von Neumann Architecture

The Von Neumann Model

3. Execute

- Execute the instruction
- Place the results in registers or memory

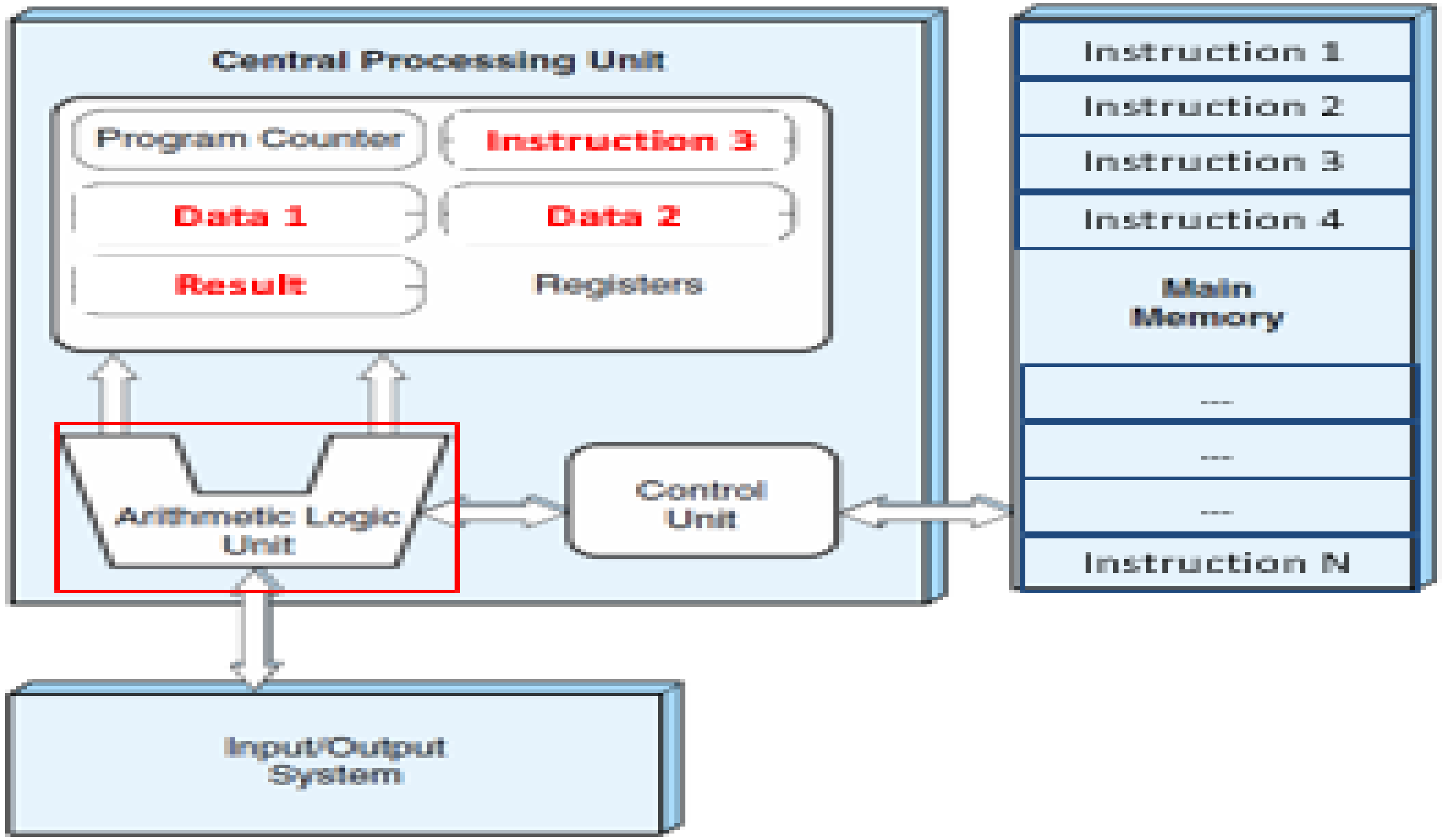
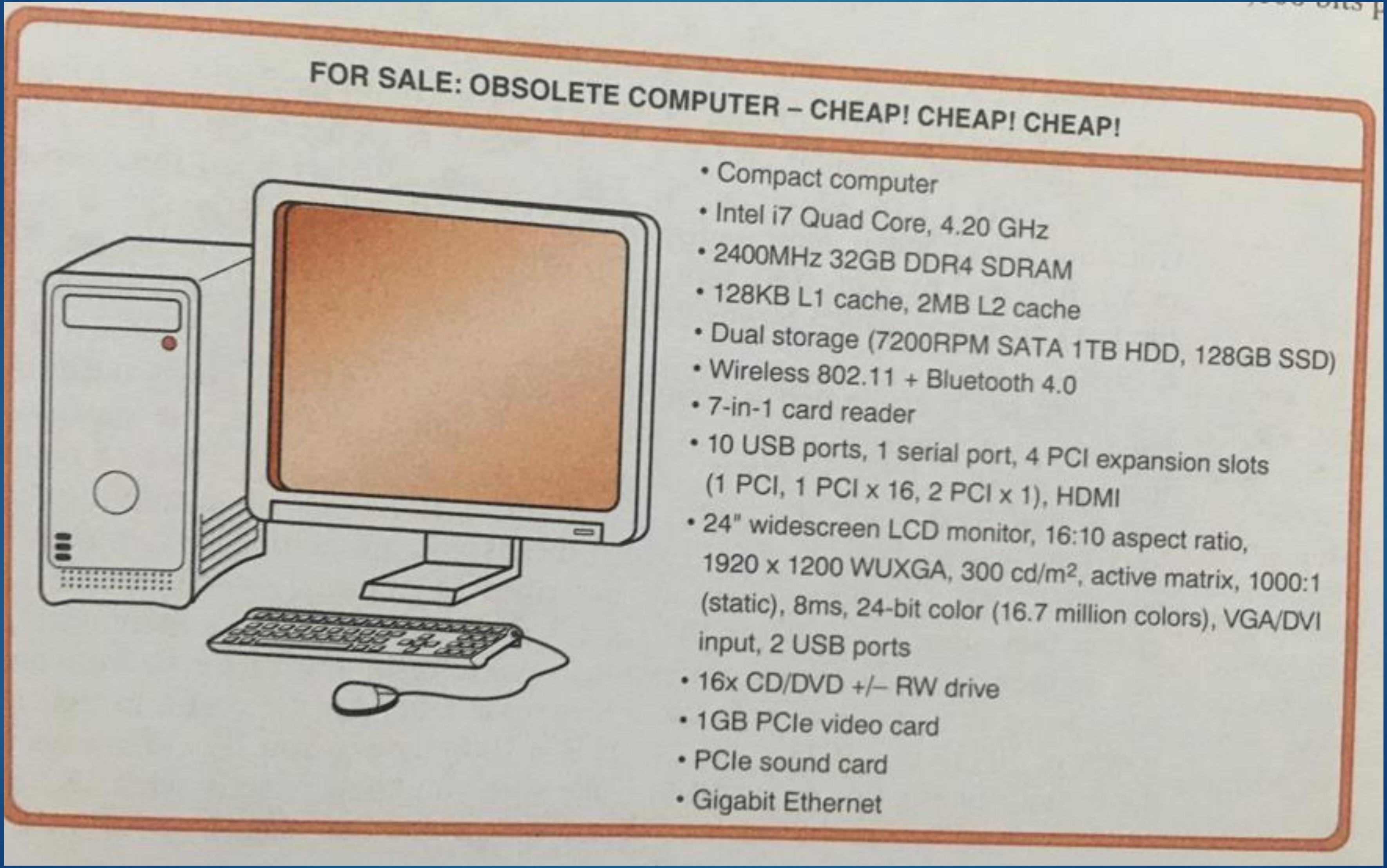


FIGURE 1.4 The von Neumann Architecture

An example system: wading through the jargon



FOR SALE: OBSOLETE COMPUTER – CHEAP! CHEAP! CHEAP!

- Compact computer
- Intel i7 Quad Core, 4.20 GHz
- 2400MHz 32GB DDR4 SDRAM
- 128KB L1 cache, 2MB L2 cache
- Dual storage (7200RPM SATA 1TB HDD, 128GB SSD)
- Wireless 802.11 + Bluetooth 4.0
- 7-in-1 card reader
- 10 USB ports, 1 serial port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1), HDMI
- 24" widescreen LCD monitor, 16:10 aspect ratio, 1920 x 1200 WUXGA, 300 cd/m^2, active matrix, 1000:1 (static), 8ms, 24-bit color (16.7 million colors), VGA/DVI input, 2 USB ports
- 16x CD/DVD +/– RW drive
- 1GB PCIe video card
- PCIe sound card
- Gigabit Ethernet