Module 1: How Computers Work

Prof. Dr. Florian Haselbeck
Memoona Shakeel
Weihenstephan-Triesdorf University of Applied Sciences

Learning outcomes

Understand the basic architecture and components of a computer and how programs run on it.

Key Topics

- Hardware Vs Software
- CPU & RAM, Storage
- Input/Output
- Program & Process
- File Systems
- Embedded System

Introduction — Why learn how computers work?

Everything in digital farming runs on a computer

- Sensors, drones, irrigation controllers, and yield monitors have a processor, memory, and storage inside it.
- When a sensor script crashes, a log file disappears, or a model runs out of memory, you need to know why.
- **This module gives you the vocabulary to read error messages, understand hardware specs, and debug real problems.**

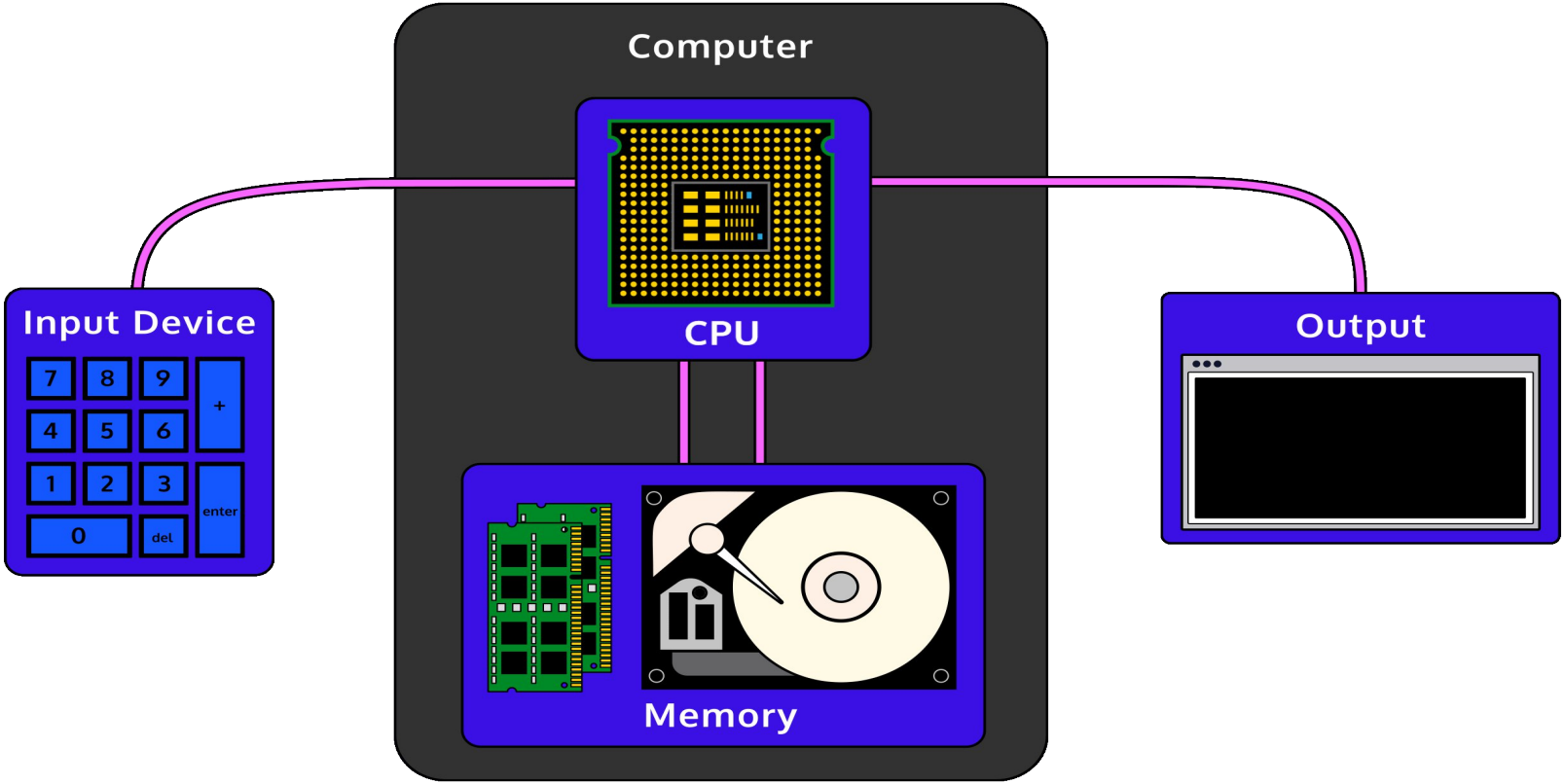
Name the components	Trace program execution	Diagnose problems
CPU, RAM, storage, and I/O devices, what each one does, and why all three must exist together	Understand step-by-step how a Python script goes from a file on disk to running in memory	Tell apart hardware failures, software bugs, and file-system errors.

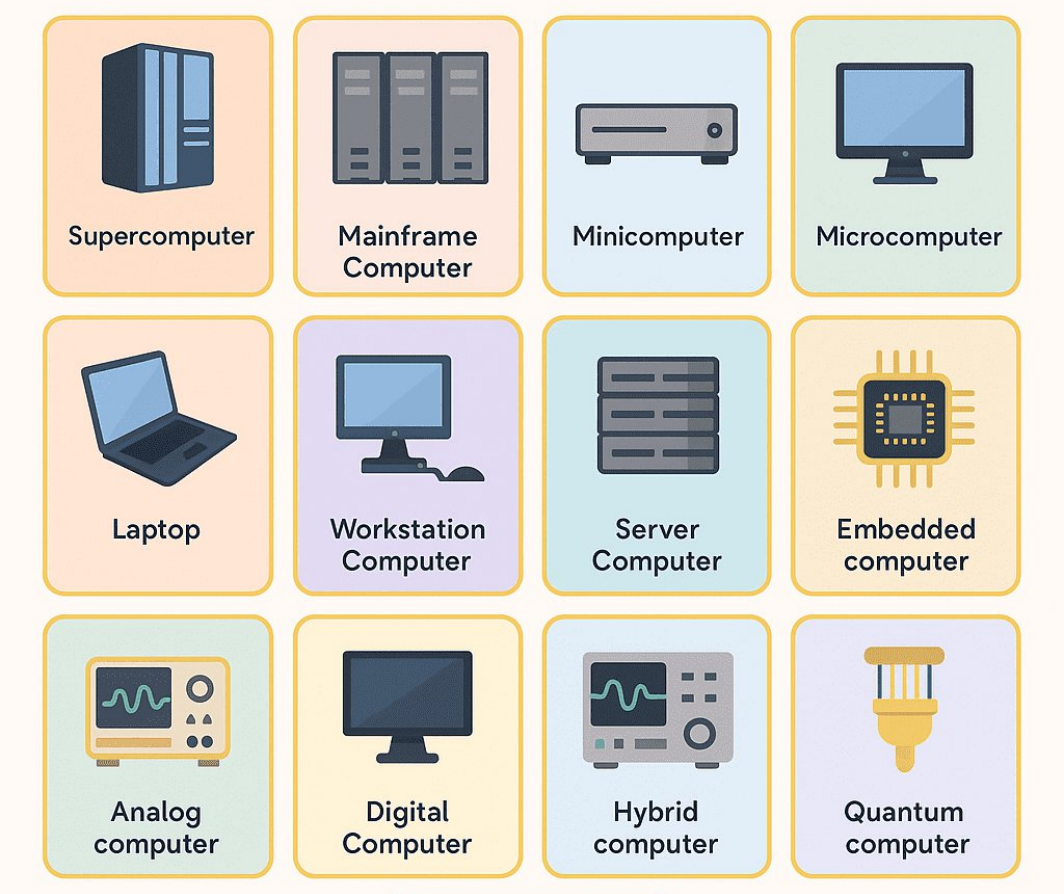
What is a computer?

A computer is a programmable electronic device that processes data according to a stored set of instructions (a program).

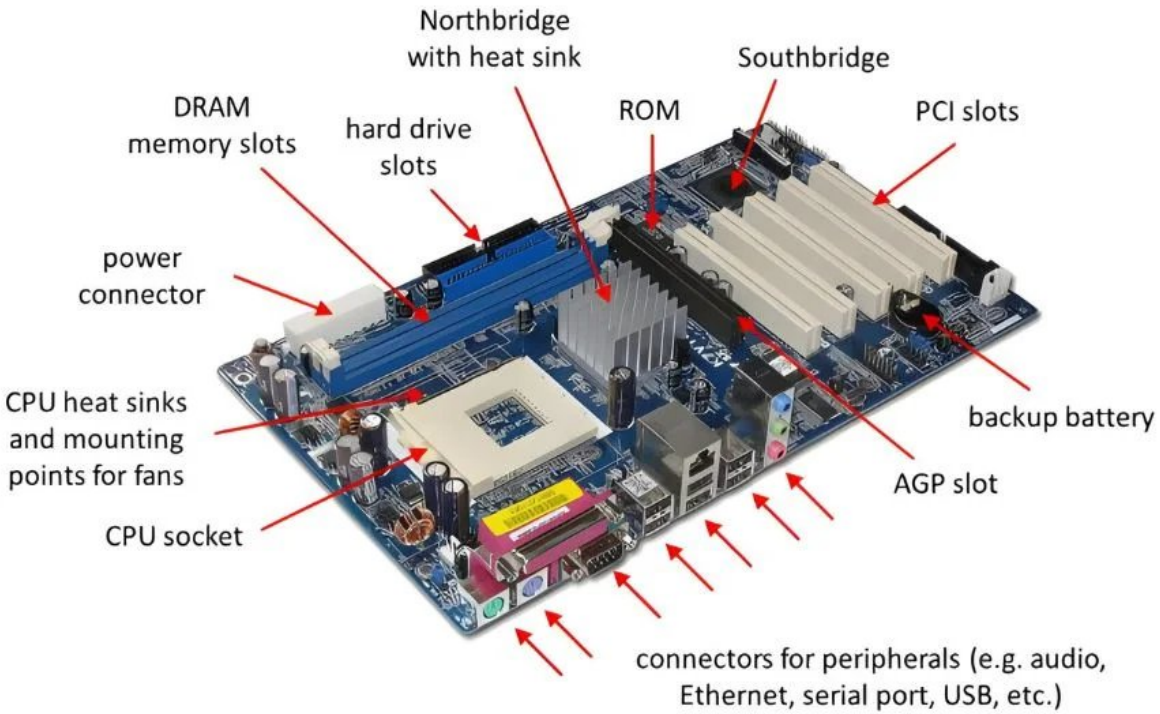
INPUT	PROCESS	OUTPUT
Data entering the computer from the outside world	The CPU runs the program's instructions on that data	Processed data leaving the computer to affect the outside world
How: Via hardware interfaces: USB, GPIO pins, network socket, serial port	How: Arithmetic (add, subtract), logic (AND, OR), control flow, memory access	How: Via hardware interfaces: GPIO, display controller, network, serial
Examples: Keyboard press, mouse click, sensor voltage on a GPIO pin, data packet arriving over Wi-Fi	**Examples:** Comparing two integer values, calling a function, running a loop iteration, updating a variable	**Examples:** Pixel written to screen, voltage on GPIO pin, byte sent over UART, packet sent over Ethernet

Every component inside a computer, CPU, RAM, storage, and OS, exists solely to move data through this Input → Process → Output cycle faster and more reliably.


Computer
CPU
Input Device
7
8
9
+
4
5
6
1
2
3
enter
0
del
Memory
Output


Supercomputer
Mainframe Computer
Minicomputer
Microcomputer
Laptop
Workstation Computer
Server Computer
Embedded computer
Analog computer
Digital Computer
Hybrid computer
Quantum computer

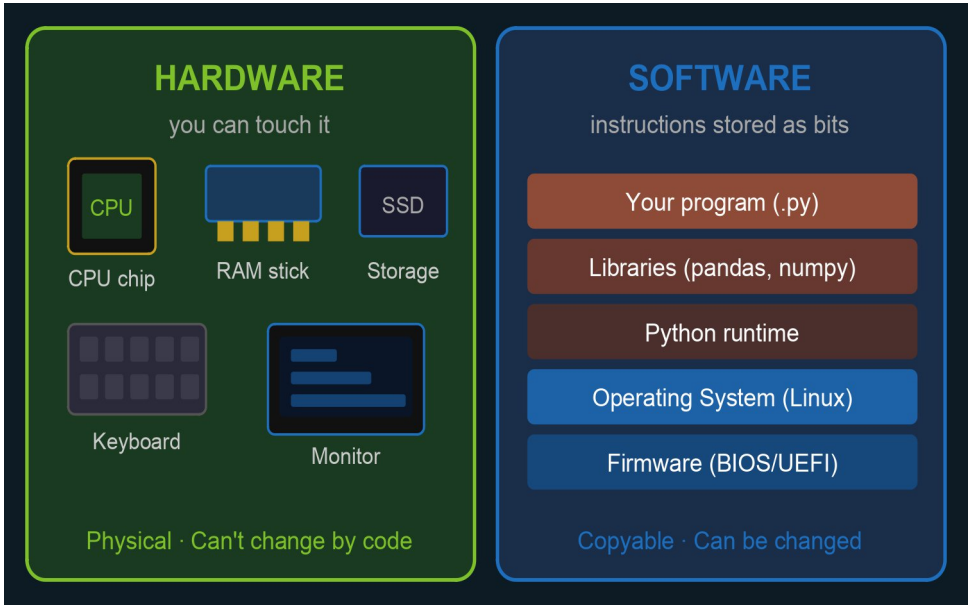
Motherboard



Hardware Vs Software

Hardware is the **physical**, **tangible** equipment you can touch, like a processor or screen, that performs the actual work.
- Physical parts: chips, wires, screens
- Fixed — cannot change by writing code
- Breaks? → replace the physical part

Software is the **intangible** set of digital instructions and programs that tell the hardware exactly what to do. Essentially, hardware is the body of the machine, while software is the mind that operates it.
- Instructions stored as bits (0s and 1s)
- Can be copied, changed, or deleted
- Breaks? → reinstall or fix the code



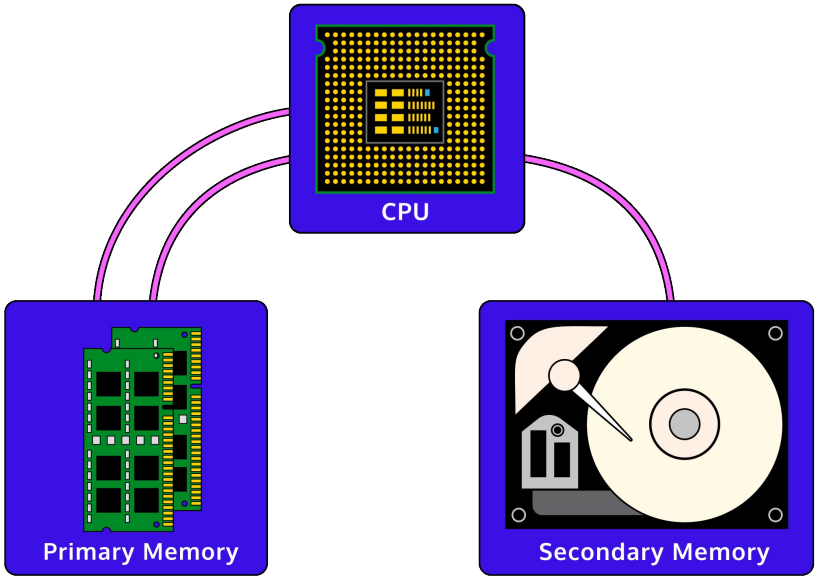
Key rule: hardware defines what operations are physically possible; software specifies which of those operations to perform and in what order. The same CPU can run Windows, Linux, or macOS; the hardware did not change.

The three core components

CPU is the brain. It reads instructions and executes them billions of times per second, like a processor or screen, that performs the actual work.

Primary Memory or RAM (Random Access Memory) is your computer's short-term memory. When you run a program, it gets loaded here so the CPU can access it fast. The key thing: RAM is volatile; everything in it disappears the moment you switch the computer off.

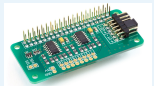
Secondary Memory is a **hard drive** (HDD or SSD). This is long-term storage, where your files, programs, and operating system reside permanently. It survives a power-off. It is much slower than RAM but holds far more data.



Key rule: The CPU can only work with data that is currently on RAM — if it's on the SSD or HDD, it must be loaded into RAM.

Input and Output devices

Input devices send data into the computer. Output devices receive data from the computer.

INPUT devices — send data to the CPU	
Keyboard	
Mouse / touchscreen	 
GPIO sensor (ADC)	
Network card (Ethernet)	
Camera (MIPI / USB)	

OUTPUT devices — receive data from the CPU	
Monitor (HDMI/DisplayPort)	  
Printer	
Speaker	
Projector	
Headphone / Microphone	 

Programs and Processes

PROGRAM — a file on storage

- A file containing instructions (code)
- Exists on disk, doing nothing, using no CPU or RAM
- You can copy it, delete it, or send it by email
- Examples: frost_check.py, python3.exe, notepad.exe
- One program file can be started multiple times

PROCESS — a running program

- What you get when you run a program
- The OS loads the file into RAM and the CPU starts executing it
- Has its own memory space, a process ID (PID), and state
- View running processes: Task Manager (Windows) / (Linux)
- When the program finishes, the process ends and RAM is freed

What happens when you type: `python3 frost_check.py`

1. OS finds frost_check.py on storage
2. Copies the code into RAM (now a process)
3. CPU begins executing line 1, 2, 3 …
4. Program finishes OS frees RAM Process is gone

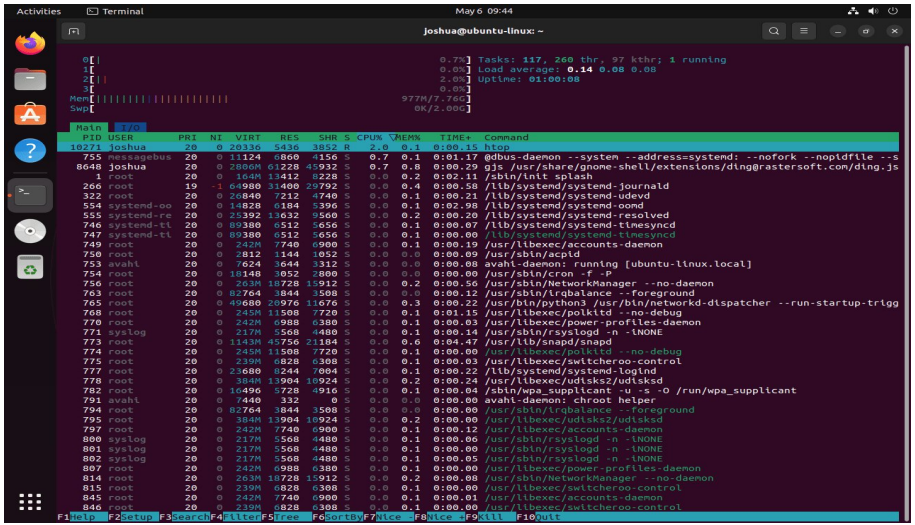
Monitoring Processes on Different Platforms

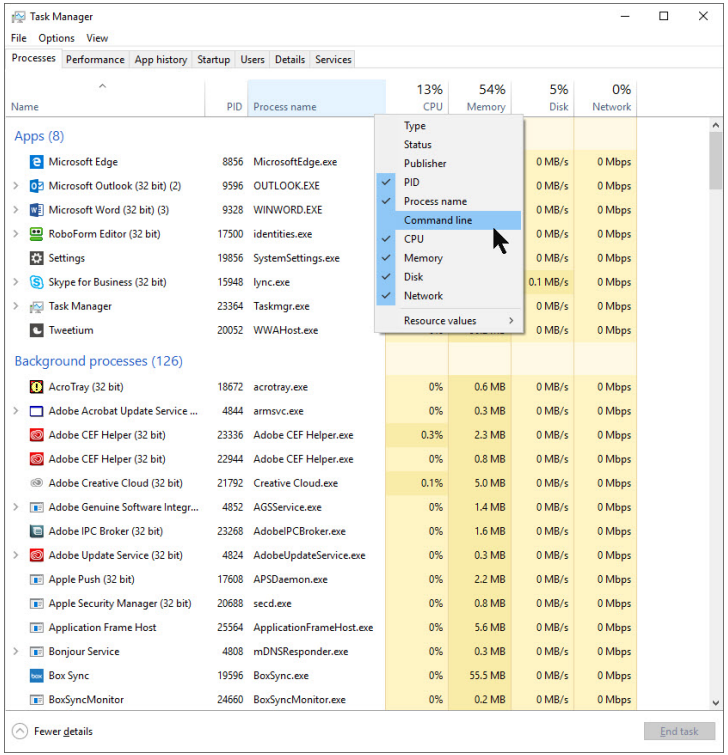
Windows

- **Task Manager:** Shows running apps and background processes. Displays CPU, memory, disk, and network usage

Ubuntu

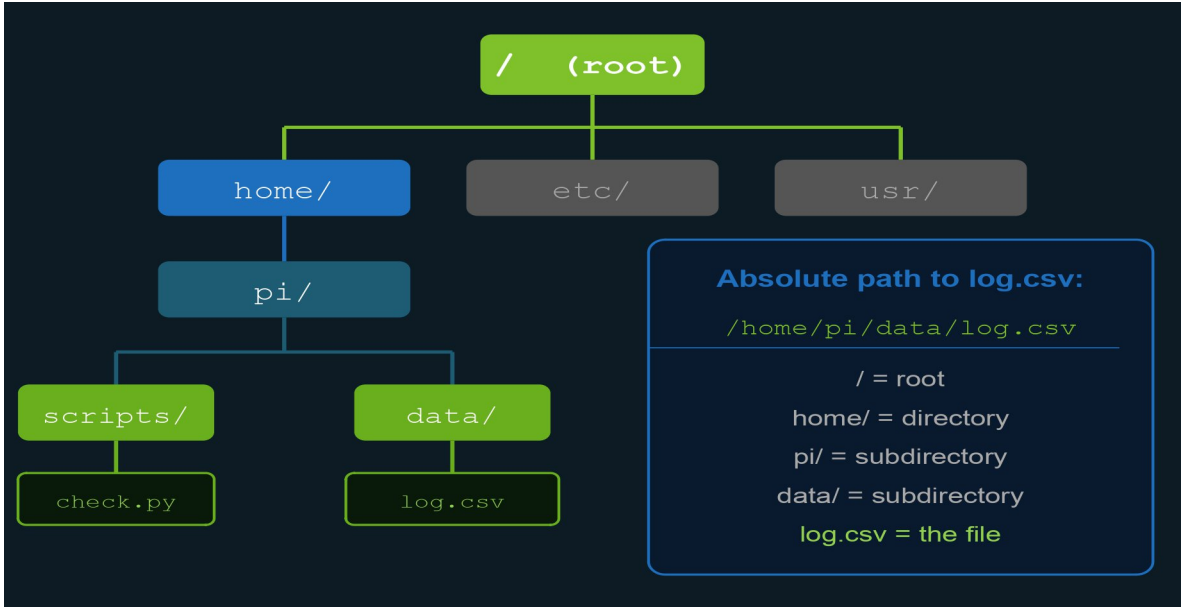
- **htop:** Shows real-time CPU, RAM, and processes





Files and File Systems

A file system organises how data is stored on a device, like a folder tree on your computer.



File

A named collection of bytes on disk

e.g. log.csv, check.py, photo.jpg

Directory

A container holding files and other folders e.g. /home/pi/ or C:\Users\

Path

The full address of a file

/home/pi/data/log.csv

Embedded System

Embedded system is a **computational system** that is developed based on an integration of both hardware and software in order to perform a given task. It can be said as a dedicated computer system has been developed for some particular reason.

Components of Embedded Systems

1. Hardware 2. Software 3. Firmware

Examples of Embedded Systems

- Digital watches, Washing Machine
- Televisions, Digital phones
- Laser Printer, Industrial machines
- Electronic Calculators
- Automobiles, Medical Equipment

Overview of key content

- **CPU** fetches, decodes and executes instructions; **RAM** holds active data; **storage** persists data
- **Hardware** defines capability; **software** defines behaviour
- A **program** is a file; a **process** is a running instance with its own PID and virtual memory
- A **file system** maps names to byte sequences on storage via inodes and directory entries

Thank you!

Contact Detail

Prof. Dr. Florian Haselbeck

florian.haselbeck@hswt.de

Room D1.230a

For any questions, please feel free to use the **Moodle-Course Forum**!