**What Is a Context Switch?**

A **context switch** happens when the **CPU stops working on one process** and **starts working on another**.

Since the CPU can handle only one process at a time (per core), it must **pause** the current one and **switch** to the next.

**🔷 What Happens During a Context Switch?**

1. **Save the current state** of the running process (like register values, program counter, etc.)
2. Store this state in the **Process Control Block (PCB)**.
3. **Load the saved state** of the new process from its PCB.
4. **Resume execution** of the new process from where it left off.

📦 The **context** is everything the CPU needs to remember to continue a process later.

**🔷 Why Is Context Switch Important?**

It allows **multitasking** — switching between different processes so they appear to run in parallel.

But…

**🔶 Context Switch = Overhead**

* While switching, the CPU is **not doing any actual work** (like calculations or I/O).
* It’s busy saving and loading states → this is **overhead**.
* The more frequently context switches happen, the more time is lost doing non-useful work.

**🔷 Factors Affecting Context Switch Time**

1. **Complexity of OS and PCB**
   * A complex OS might have more data to save/load → takes longer.
2. **Hardware Support**
   * Some CPUs have **multiple sets of registers** (like backup storage).
   * This allows **faster switching** because you can just **swap contexts instantly** without saving to memory.

**🔷 Real-Life Analogy**

Imagine you're writing an essay (Process A), then someone asks you to help with math homework (Process B):

* You **bookmark** your essay, close it.
* Then **open your math notes**, continue where you left off.
* Later, you **come back to your essay** by reopening and checking where you stopped.

Each switch between tasks is like a **context switch**. The more you switch, the more time you waste on just switching.