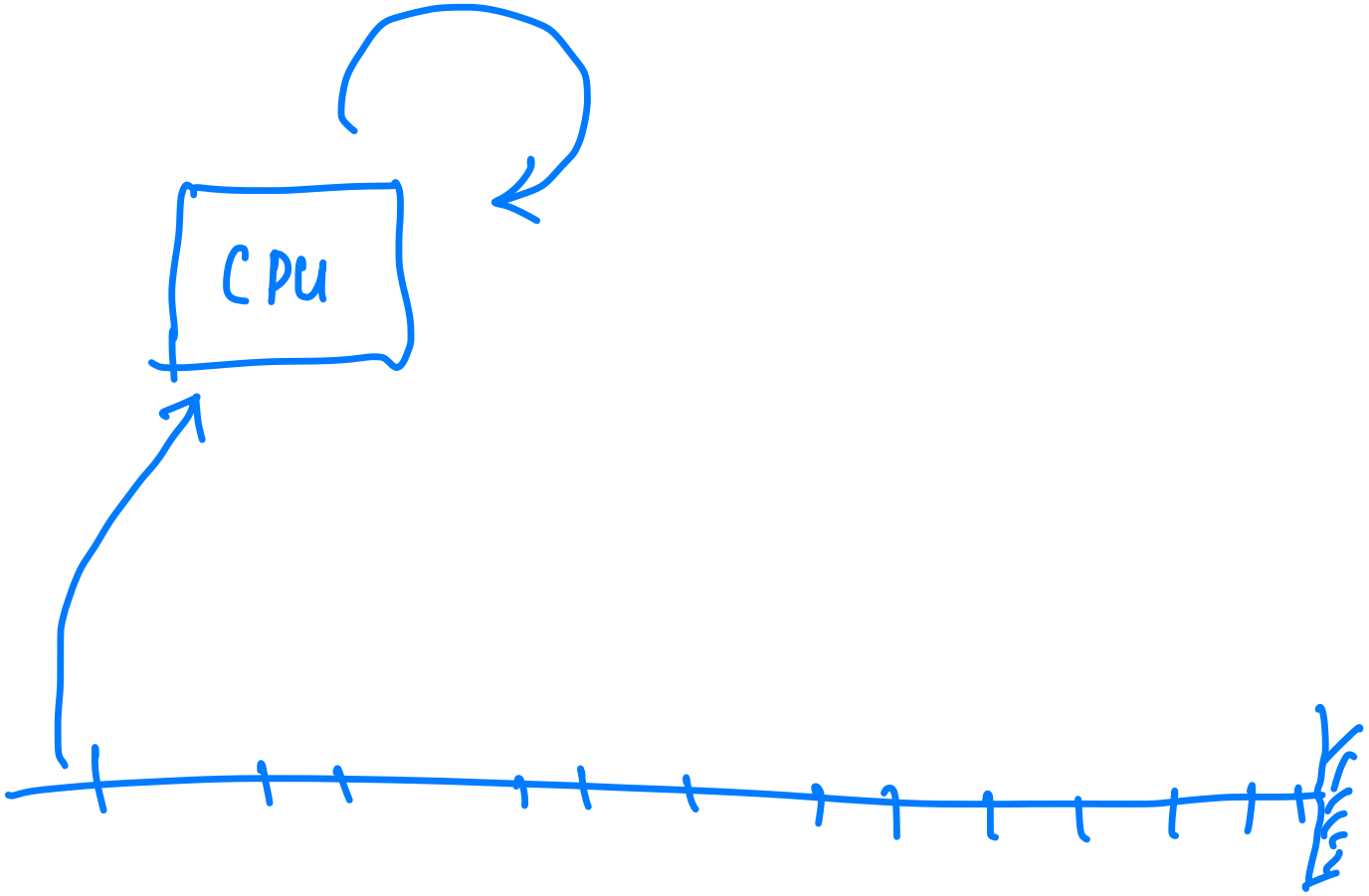
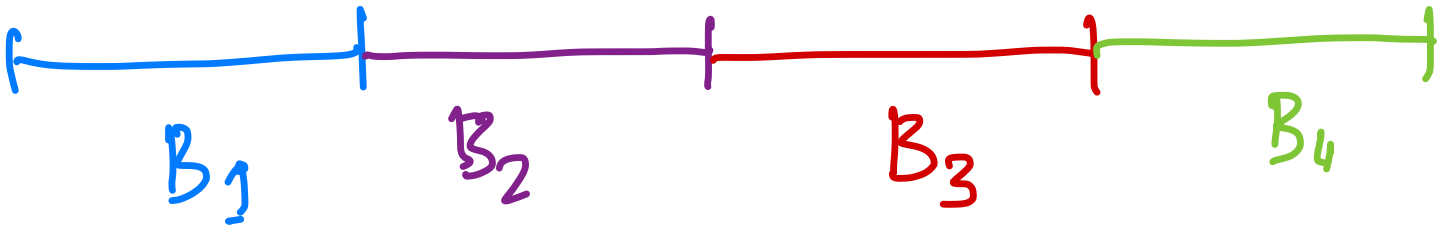
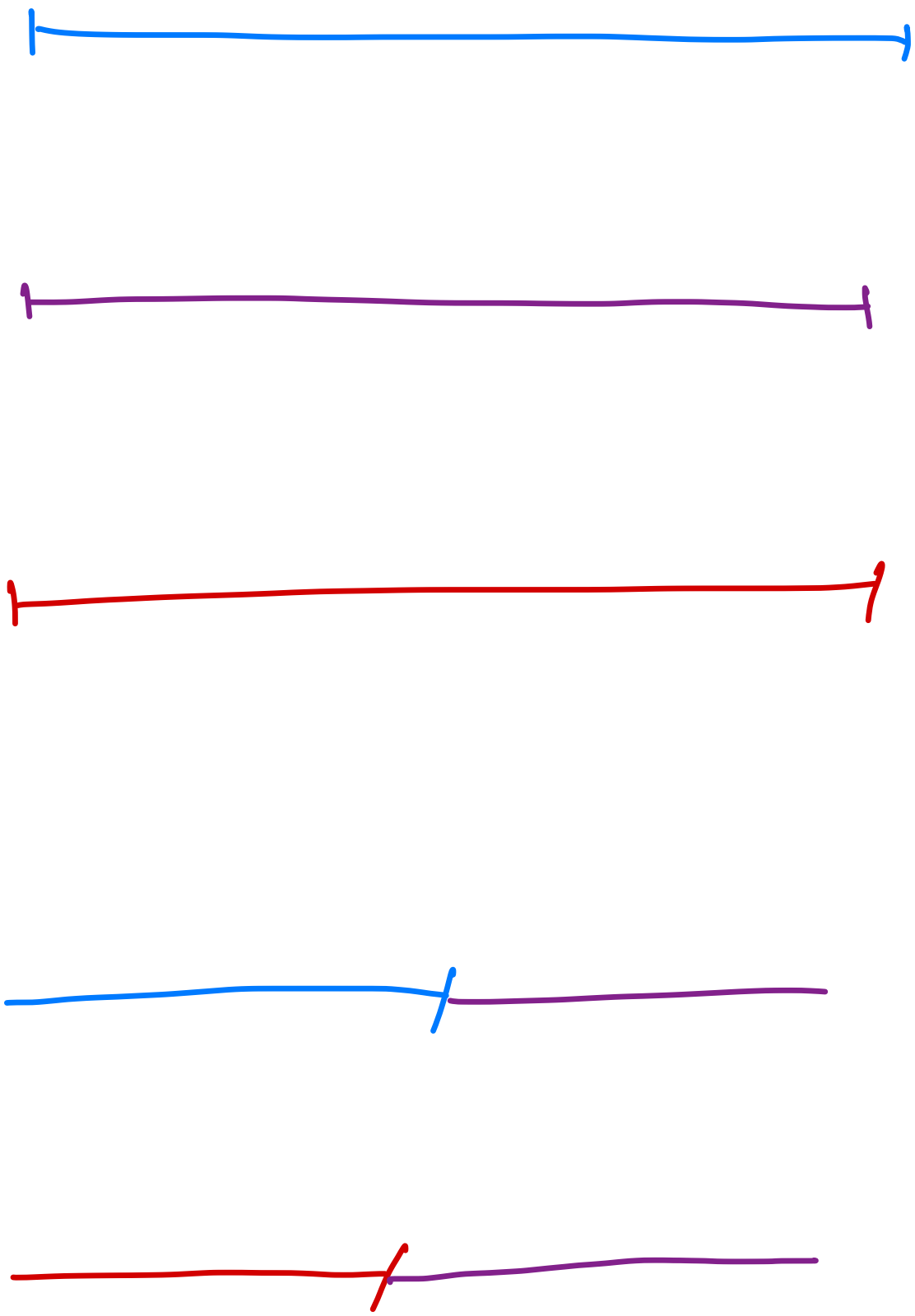
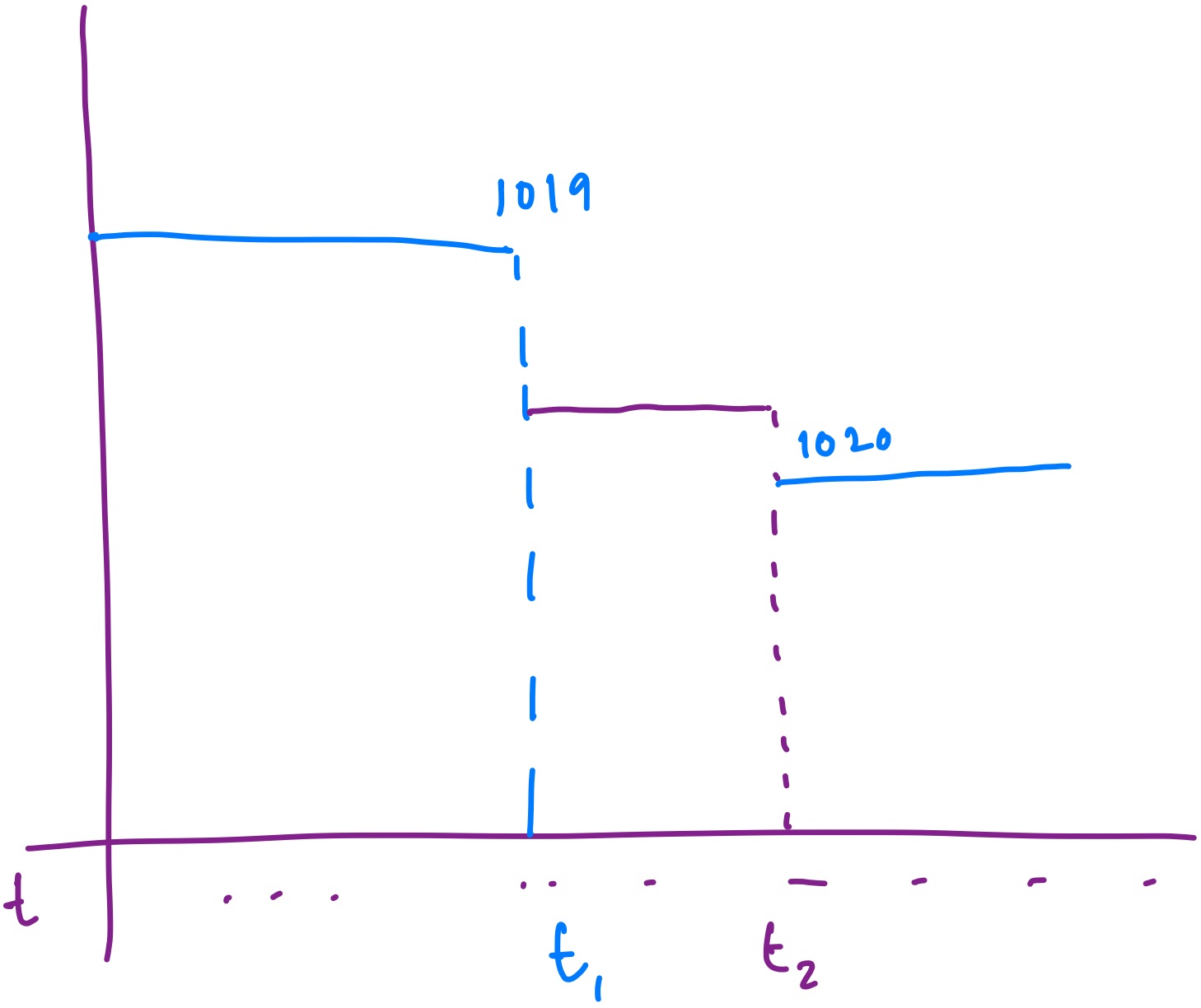



Batch processing



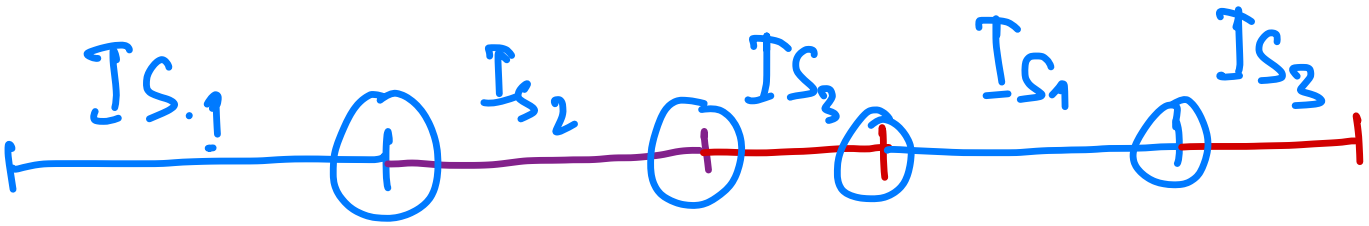

Sequential

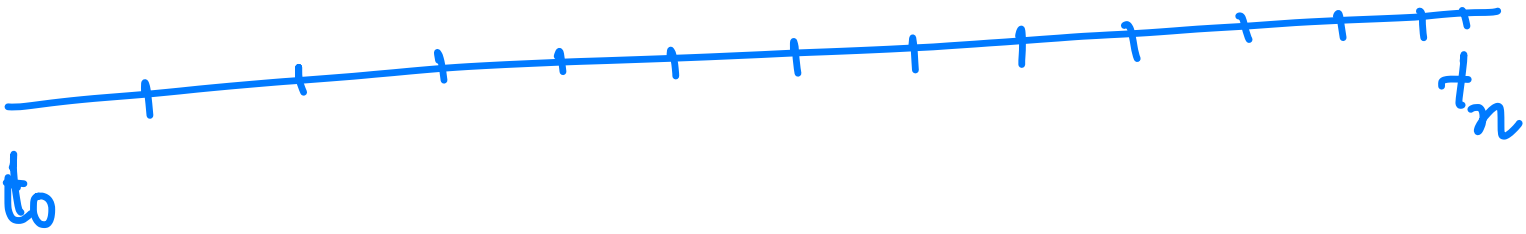




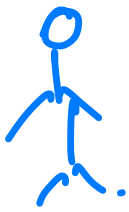

O.S.

Coordinator




Instruction Set →
processes



Change-over

Context-switching

OS → to efficiently manage the CPU time between different processes

Efficient management of CPU time

1) Equal importance to all processes (fairness)

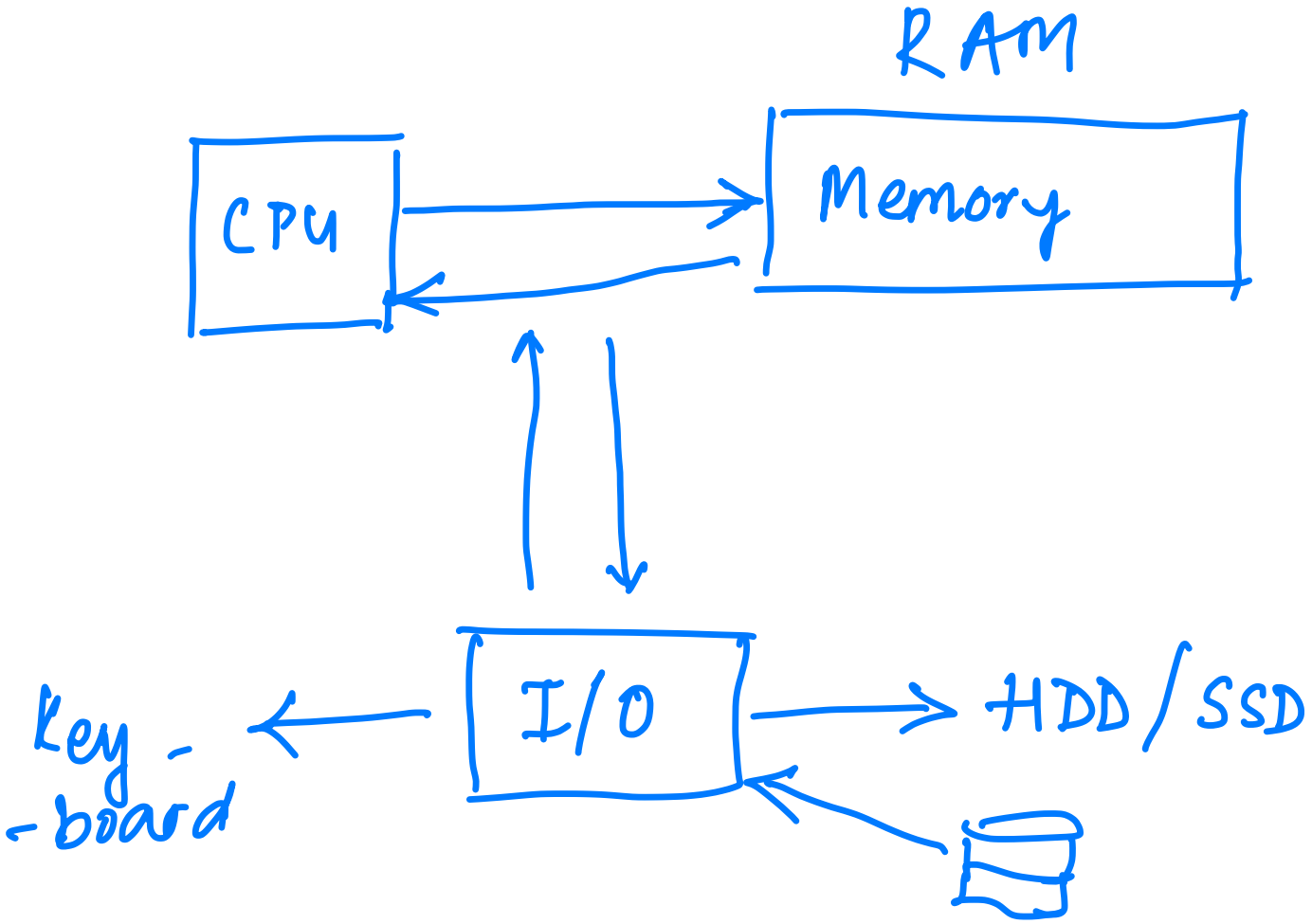
2) provide the isolation ~~for~~ the processes and yet provide the way for communication.

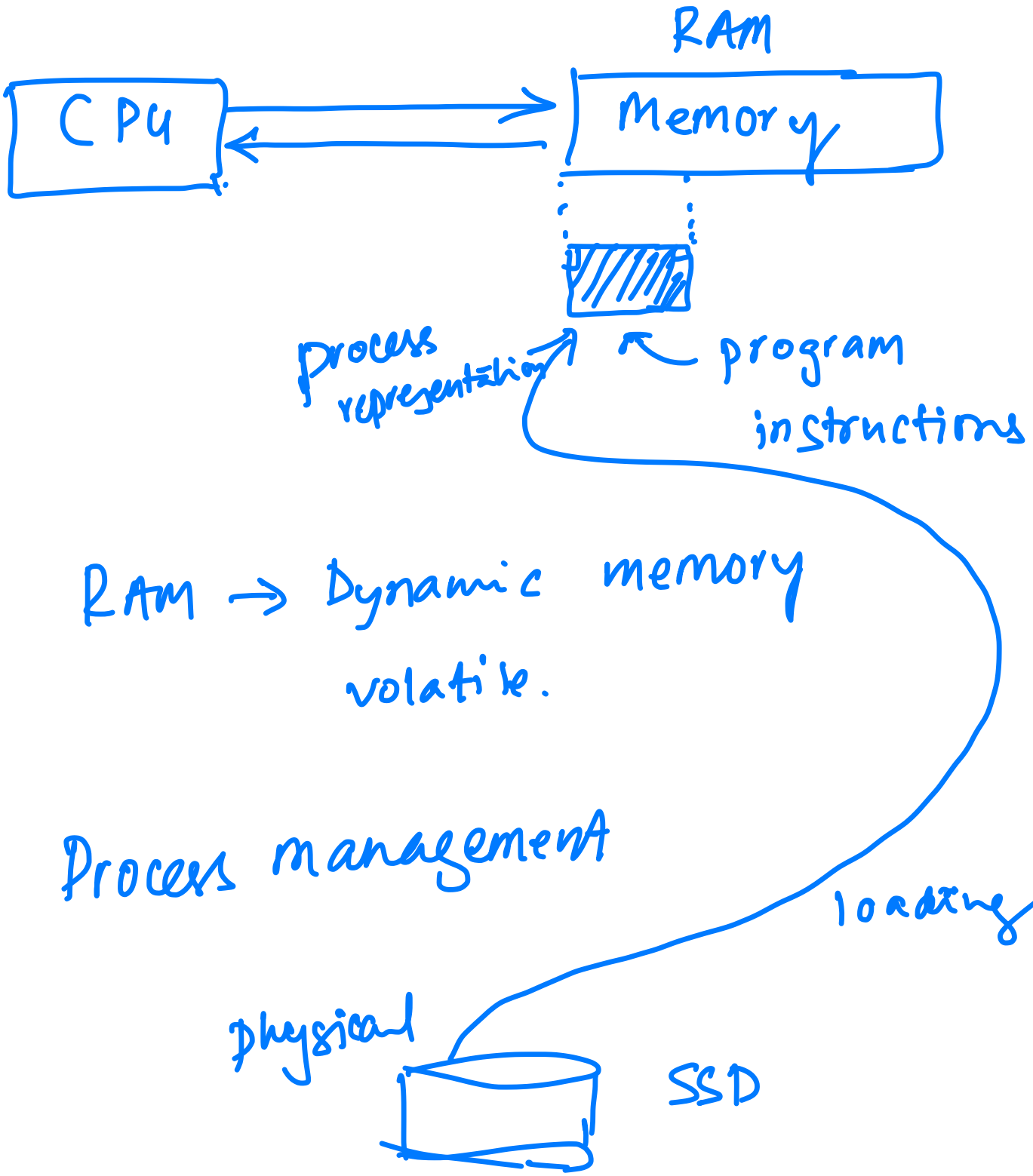
3) Make sure a process doesn't hug the CPU beyond the limits

Process → set of instructions for the CPU.

Usually store these instructions in binary in a file on the file system.

von Neumann Architecture.






RAM → Dynamic memory

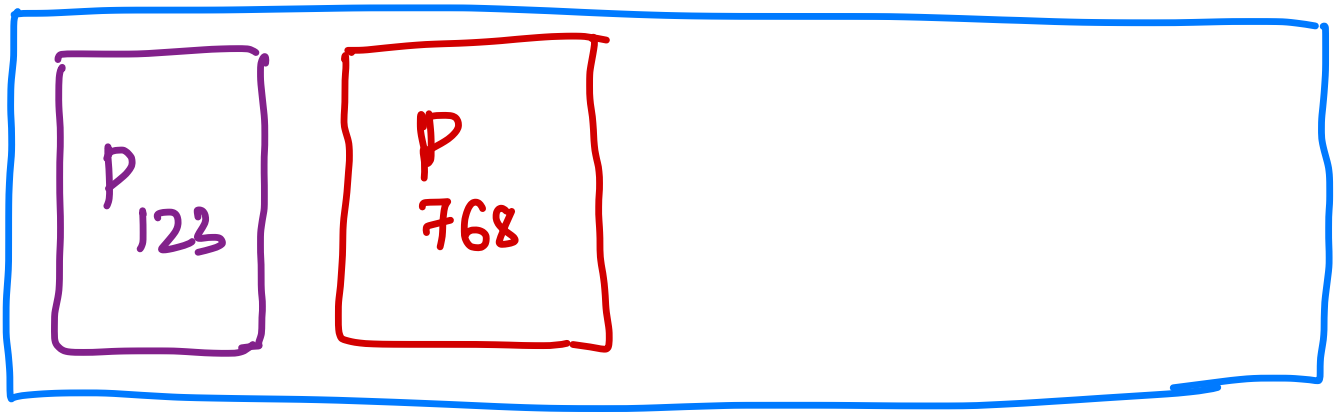
volatile.

Process management

Loading happens manually (by user) & automatically.

---

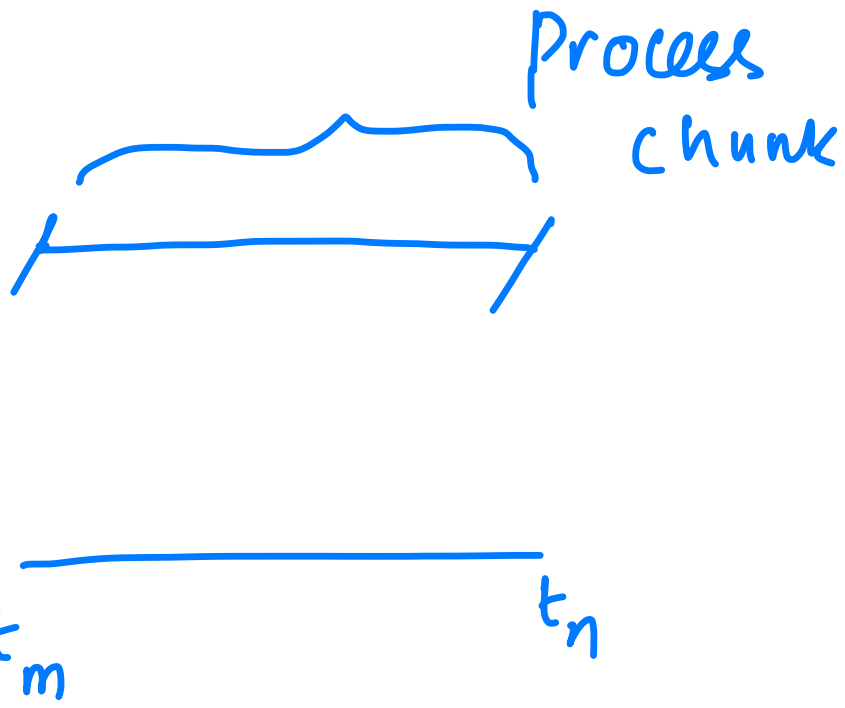
When a process is loaded, the OS provides an abstraction of the slice of memory to this process. This slice is an isolated box. (Sandboxing)




When the CPU is executing $P_{123}$ it is only allowed to access the addresses within the silo of $P_{123}$

OS. Time slice. Process.

Thread. → execution path.







Within a process, we may group the instructions so that they constitute an execution path. This happens usually in the code.

Technically, a process may have more than one such execution paths.

Execution path is called a Thread

There is always at least one thread per process.

---

A process is basically instructions executing. These instructions are themselves (or part of) functions. or methods. The execution, then, is the chain of method calls, where every method has other instructions. A group of instructions, then, is the group of such function call chain. Thread is the method call chain.

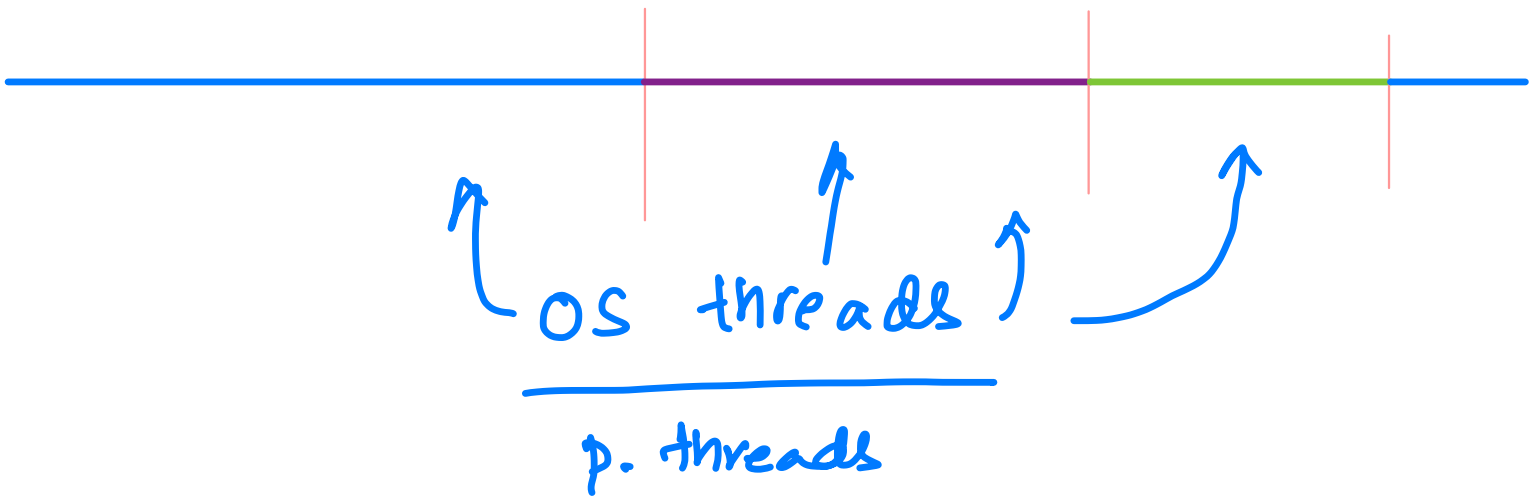
In a method, we create variables (local variables), we may get parameters passed, we may call other methods.

The programming abstraction of this is called a Stack

In a Stack, we have the ability to declare local variables, parameters, and call chain handling.

---

A thread gets its own Stack


OS threads
p. threads


t
hardware thread.