Threads and concurrency

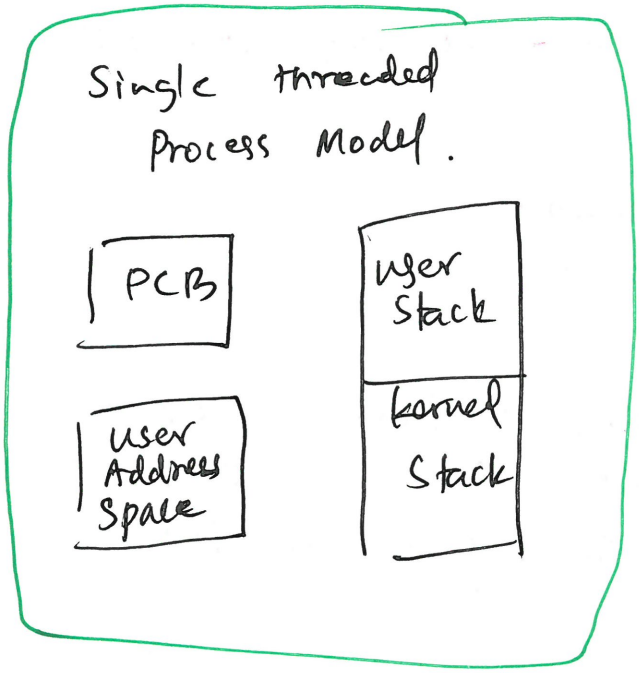
We redefine a process and a thread in this lecture.

- A process is unit of resource ownership
- A thread is unit of dispatch.

A process can contain multiple threads.

Each thread has

- Execution state.
- A saved thread context when not running
- Execution Stack
- per thread static storage



Why threads?

- Easly and fast to create
- less time to terminate
- switching between threads is easier.
- Threads enhance efficiency in communication between programs.

concurrency

- Ability to run multiple processes or tasks simultaneously or in an overlapping manner.

Parallelism

- Processes are truely executed simultaneously

concurrency and parallelism are related concepts but not the same.

parallelism is a type of concurrency.

Race condition

- Occurs when multiple processes or threads read & write dat ~~e~~ items to a shared resource at the same time. The final result will depend on the order of the execution.

When concurrent processes compete for resources following problems must be faced.

- Need for mutual Exclusion
- Deadlocks
- Starvation.

How to ensure mutual Exclusion?

Hardware Support

- Interrupt Disabling
- Atomic Instructions.

OS & programming language support.

- Mutex
- Semaphores
- Monitors.