

thread aur process mei sbhse bda difference hai:- threads same memory space of process mei kaam krti hai aur alag alag process ki alag alag memory space hogi

process ke paas PCB hota hai
aur thread ke paas TCB

LEC-15: Introduction to Concurrency

process control block

1. **Concurrency** is the execution of the multiple instruction sequences at the same time. It happens in the operating system when there are several process threads running in parallel.
2. **Thread:**
 - Single sequence stream within a process.
 - An independent path of execution in a process.
 - Light-weight process.
 - Used to achieve parallelism by dividing a process's tasks which are independent path of execution.
 - E.g., Multiple tabs in a browser, text editor (When you are typing in an editor, spell checking, formatting of text and saving the text are done concurrently by multiple threads.)
3. **Thread Scheduling:** Threads are scheduled for execution based on their priority. Even though threads are executing within the runtime, all threads are assigned processor time slices by the operating system.
4. **Threads context switching**
 - OS saves current state of thread & switches to another thread of same process.
 - Doesn't includes switching of memory address space. (But Program counter, registers & stack are included.)
 - Fast switching as compared to process switching
 - CPU's cache state is preserved.
5. **How each thread get access to the CPU?**
 - Each thread has its own program counter.
 - Depending upon the thread scheduling algorithm, OS schedule these threads.
 - OS will fetch instructions corresponding to PC of that thread and execute instruction.
6. **I/O or TQ, based context switching is done here as well**
 - We have TCB (Thread control block) like PCB for state storage management while performing context switching.
7. **Will single CPU system would gain by multi-threading technique?**
 - Never.
 - As two threads have to context switch for that single CPU.
 - This won't give any gain.
8. **Benefits of Multi-threading.**
 - Responsiveness
 - Resource sharing: Efficient resource sharing.
 - Economy: It is more economical to create and context switch threads.
 1. Also, allocating memory and resources for process creation is costly, so better to divide tasks into threads of same process.
 - Threads allow utilization of multiprocessor architectures to a greater scale and efficiency.