Operating Systems

Curated computer science study track (not a degree, no licensure claim). Original explanations and practice only, following the reviewed course handout.

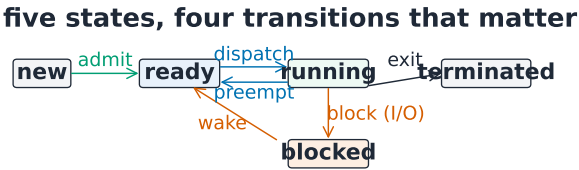
One processor, many machines

- 7.1-7.3 the CPU illusions (processes, turns, safe sharing); 7.4-7.5 memory and storage; 7.6 the wall.
- Every table, **interleaving** and page walk in this deck is deterministic and suite-computed.

Processes and threads: five states, one PCB

1. **States** New, **ready**, **running**, **blocked**, terminated - dispatch and preempt between ready and running, block on I/O, wake on completion.
2. **Switch** The PCB saves registers, PC, page-table pointer, open files: 2000 bytes at 2 GB/s is 1 us, plus cache pollution.
3. **Threads** A thread per connection: about 50 us create-and-switch against 500 us as processes - sharing is the point and the danger.

Process state diagram



blocked waits on a device; ready waits only for the CPU — the PCB carries everything a suspend must save

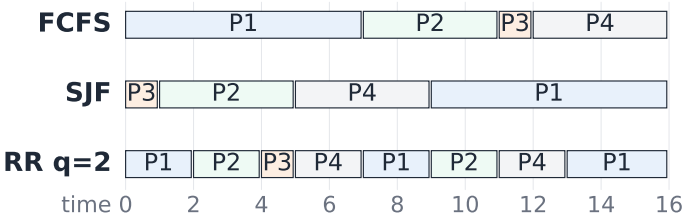
Five **states**, four transitions: dispatch, preempt, block, wake.

Scheduling: the average is a policy choice

1. **Set** Bursts 7, 4, 1, 4 arriving at 0: **FCFS** finishes 7, 11, 12, 16 (average 11.5); **SJF** finishes 1, 5, 9, 16 (7.75).
2. **RR** Quantum 2 runs 9 segments: finishes 5, 11, 13, 16 - average 11.25, worst wait far lower than FCFS 11 for the 1-burst job.
3. **Quantum** 1 us switch in a 2 ms quantum wastes 0.05 percent; in a 20 us quantum, 5 percent - switches are work, not progress.

Scheduling Gantt timelines

one arrival set (7, 4, 1, 4), three policies



average turnaround: FCFS 11.5 · SJF 7.75 · RR 11.25 — the average is a policy choice

One arrival set (7, 4, 1, 4), three policies, three timelines.

Synchronisation: the lost update

1. **Race** Both threads read 5, both write 6 - one increment lost; with four each, every final value 1 to 8 is reachable.
2. **Rules Mutual exclusion**, progress, bounded waiting; Peterson and test-and-set both **spin** - mutex queues sleep instead.
3. **Deadlock** Mutual exclusion + hold-and-wait + no preemption + circular wait: break one, and a fixed **lock** order is the cheapest.

Race interleaving and lock

read-modify-write, with and without the lock

unlocked — a lost update

A reads counter = 5
B reads counter = 5
A writes counter = 6
B writes counter = 6

final counter: 6, not 7

locked — the second thread waits

A: acquire lock, reads 5, writes 6
B: blocks on the lock
A: releases
B: acquires, reads 6, writes 7

final counter: 7, always

four increments each, unlocked: any final value from 2 to 8; the lock pins it at 8
deadlock needs all four: mutual exclusion, hold and wait, no preemption, a cycle

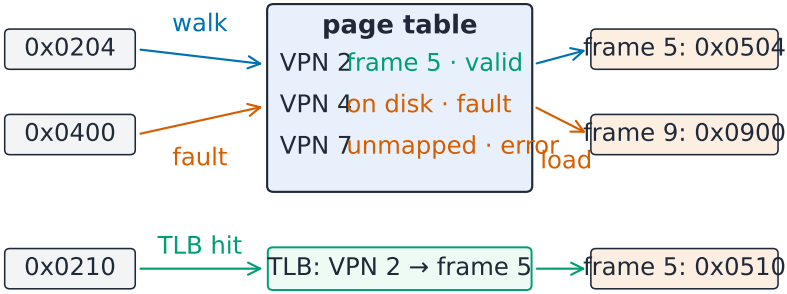
Unlocked: both read 5, both write 6. Locked: the second **thread** waits.

Virtual memory: walk, hit, fault

1. **Space** 16-bit virtual, 256-byte pages: 256 pages over 16 frames - each **process** sees 64 KiB while 4 KiB exists.
2. **Walk** 0x0204 is offset 4, VPN 2, frame 5 - physical 0x0504; 0x0210 hits the **TLB** at 0x0510, no walk.
3. **Fault** Invalid entry: block, evict (LRU, dirty first), read in, restart - 1 ns hit, 10 ns walk, 5 ms fault.

Page-table mapping

VPN through the table to a frame; the TLB skips the walk



16-bit virtual, 256-byte pages: 256 pages; 12-bit physical: 16 frames

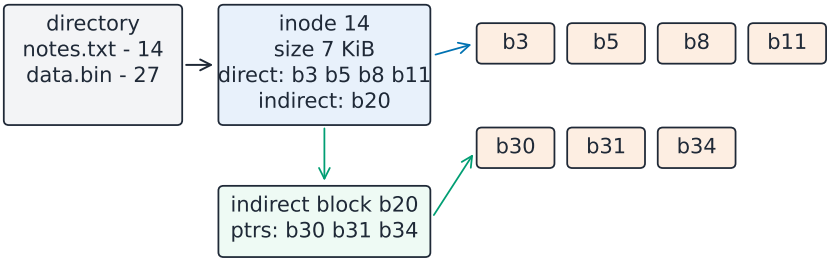
VPN through the table to a frame; the **TLB** skips the walk.

File systems: names to blocks

1. **Sizes** 1 KiB blocks, 4 direct + 256-pointer indirect: max file (4 + 256) x 1 KiB = 260 KiB; 4 KiB blocks give about 4 MiB.
2. **Price** Block 6 needs the indirect block first: two reads for one logical block; a 100-block file costs 101 reads.
3. **Crash** Data-then-inode orphans a block; inode-then-data lies about size - **journaling** writes the intent first and replays.

File system layout

name to inode to blocks — direct first, indirect for more



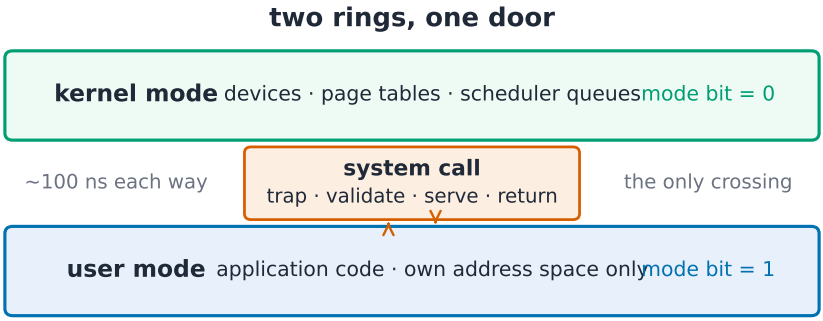
1 KiB blocks, 256 pointers per indirect block: max (4 + 256) × 1 KiB = 260 KiB

Directory entries to inodes to blocks, indirect for the big files.

Protection: two rings, one door

1. **Rings Kernel mode** touches devices and page tables; user mode crosses only by trap - two crossings, about 100 ns each.
2. **Batch** A million syscalls cost 0.2 s of crossings - the reason libraries buffer and batch.
3. **Deputy** The confused printer daemon obeys the user with its own authority - hand it capabilities, not file names.

Privilege boundary



User code crosses to the kernel only through the syscall door.

Check yourself

- Waiting on disk I/O is the... state? (**Blocked**.)
- Course set average turnaround under **SJF**...? (7.75.)
- 16-bit VA, 256-byte pages: virtual pages...? (256.)
- The only user-to-kernel crossing is the...? (**System call**.)