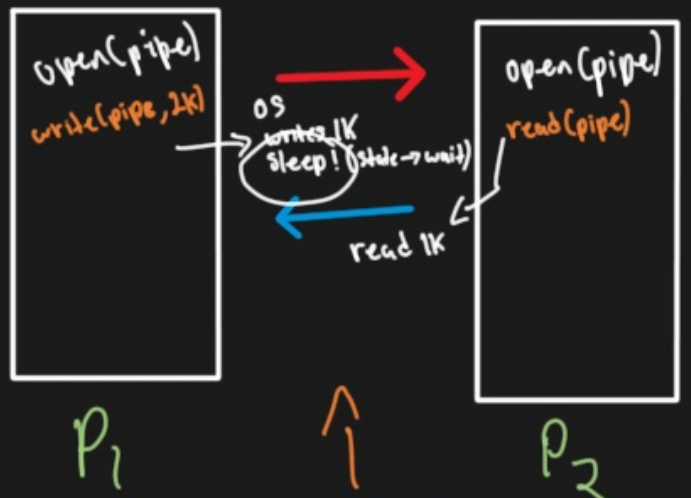
Today

- IPC (inter-process communication)
  - named pipes
  - signals
  - shared memory
- Scheduling
  - 0) First-Come, First Serve
  - 1) Round-Robin
  - 2) Priority + Decay
- System Calls
  - calling conventions
  - software interrupts
  - exception / privilege levels
- The first process
  - bootstrapping + more processes
- Concurrency
  - threads
  - multiprocessor / multicore

IPC



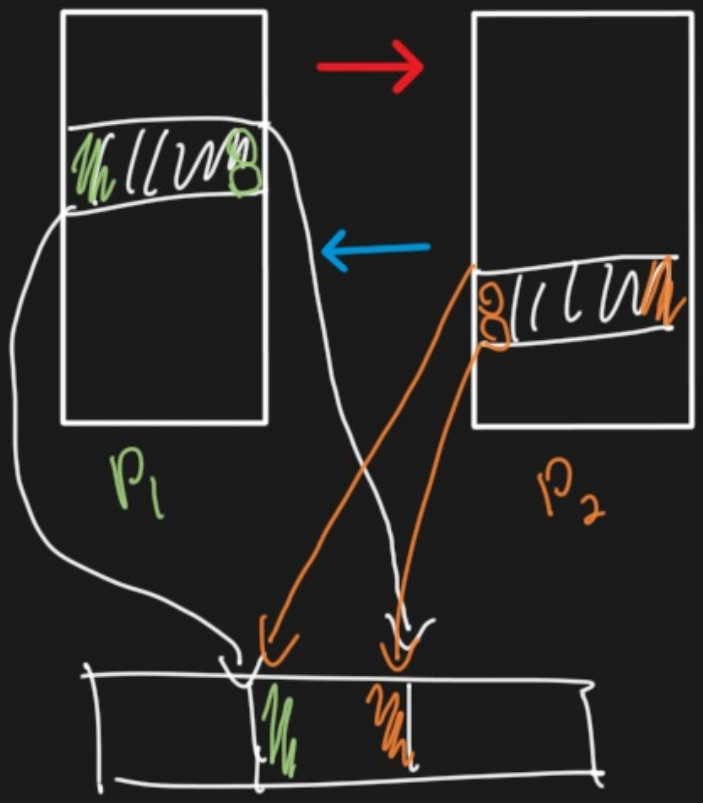
1) named pipes
- file FIFO

(un) - bounded?
- blocking?

vs. async non-blocking

Bounded 1k pipe

IPC



2) Shared Memory

+ larger shared region

+ performance

  + no copying
  + no switching ↓ kernel

mmap()

Scheduling

Which process ("thing", task) to run next?

Where - .... to run next?

- Multicore (symmetric / asymmetric)
- GPU APU

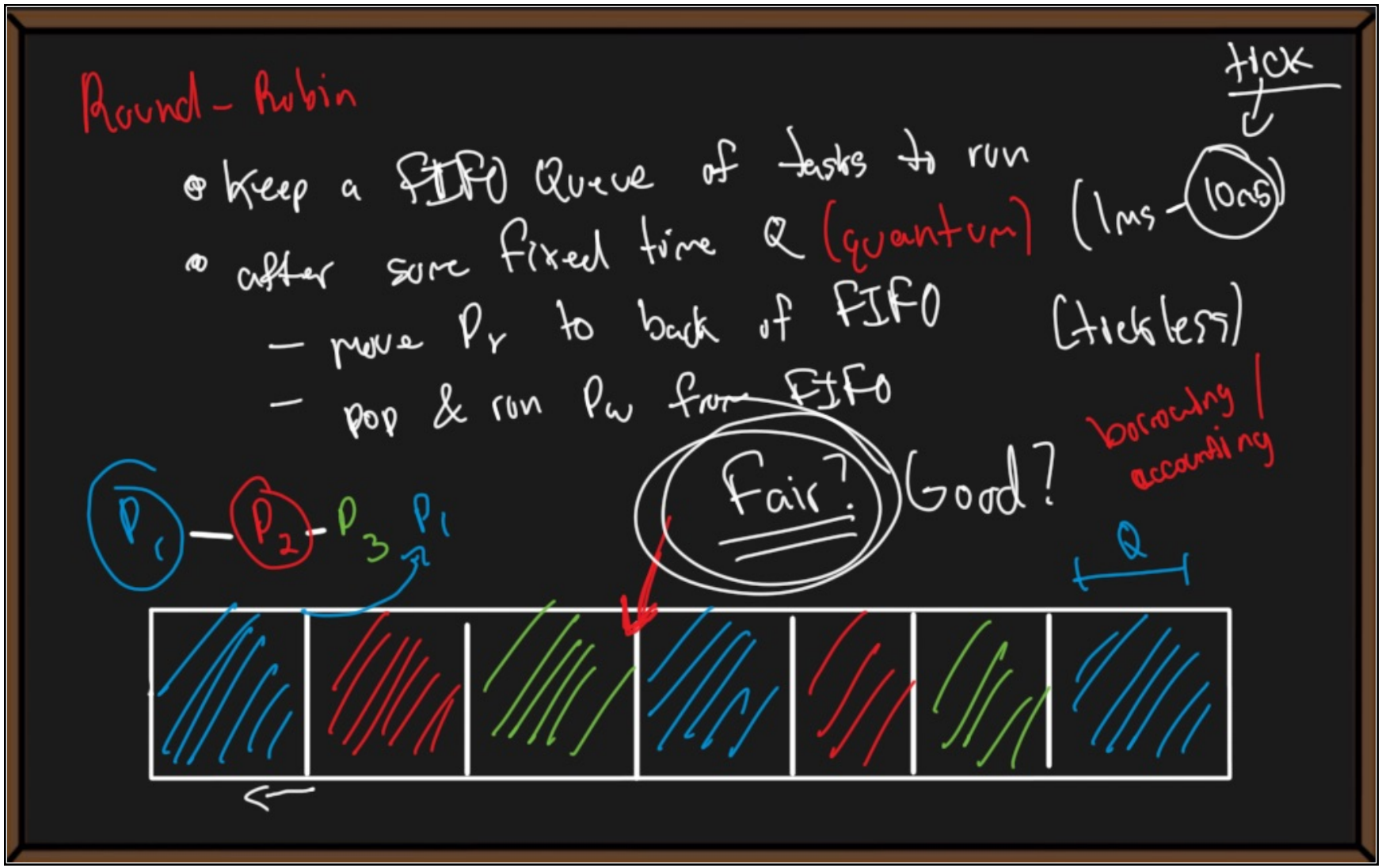
What's a Good scheduler?

↑ throughput

↓ latency
- time to finish
- "observed latency" req. → res

↑ cpu is doing something "useful"

↓ & " not " " "


Round-Robin
keep a FIFO Queue of tasks to run
after some fixed time Q (quantum)
— move P_r to back of FIFO
— pop & run P_w from FIFO
tick
(1ms - 10ms)
(tickless)
Fair?
Good?
borrowing / accounting
Q
P_1
P_2
P_3
P_1

Scheduling Algorithms: Round-Robin

2 jobs: each takes 10 s

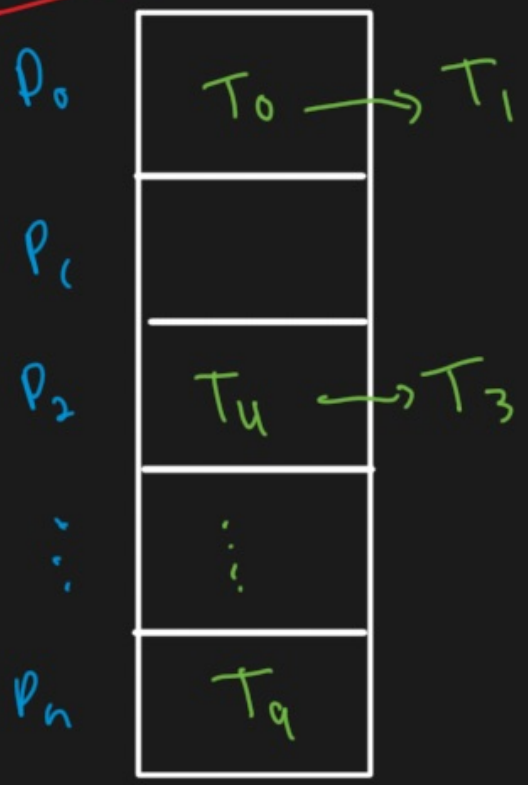
RR:

Avg. Completion Time? 9.5s! (assuming ctxt switch is free)

Running each to completion serially?

(10 + 20) / 2 = 15 s

Priorities



Each Process (task)
— associated priority

choose first from
highest priority queue

Starvation

Decay!