

Scheduling

Topics

- 1) How does the OS decide which task to run next?
(Why do we care with RT systems?)
- 2) What is priority inversion?
(Why do we care?)

Scheduling

Task Priority

- Each task has a priority
 - High priority: should be run sooner (RT tasks)
 - Normal/Low priority: run later, and preemptible
- Linux Priorities (PRI#)
 - In user level we see PRI# 0:139
 - Non Real-time: 100-139
 - Real-time: 0-99
 - "Lower" PRI# means higher priority
 - Often called..
The nicer you are, the higher your PRI#
- Ex: Run at a higher priority
(byai)\$ nice -n -10 ./lightSampler

Scheduling

- OS decides which of the ready tasks..
- Linux normally uses: Completely Fair Scheduler (CFS)
 - Aims to maximize CPU usage and minimize delays for interactive programs
 - Next scheduled task:
..
 - Max length time = $\frac{\text{The time the process has waited to run}}{\text{Total number of processes}}$
 - Processes which sleep a lot get scheduled often
 - Used for SCHED_NORMAL priority
- Why not great for RT?

Linux “RT” Task Scheduling

- RR: Round Robin
 - Starting with the highest priority level, ..
 - Then, move to the next lower priority.
- FIFO: First In, First Out
 - .. run till completion
 - Great for highest priority, time-critical tasks
- What could be a problem with FIFO?

Linux Scheduling

- Linux Scheduler Sequence
Each time Linux wants to schedule a task, it does:
 - First do all FIFO tasks
 - Then do all RR tasks
 - Then do all CFS tasks
- Challenge for system designer:
 - ..
to allow for the correct operation of the device

Earliest Deadline First

- Linux also supports Earliest Deadline First
 - Tasks declare..
- Scheduler tests new task for schedulability:
 - Given current set of tasks and their deadlines, can the OS..
 - Yes?
Accepts it (guaranteeing it will have the CPU budget to meet its deadline)
 - No?
Rejects it
- Tasks are scheduled based on EDF

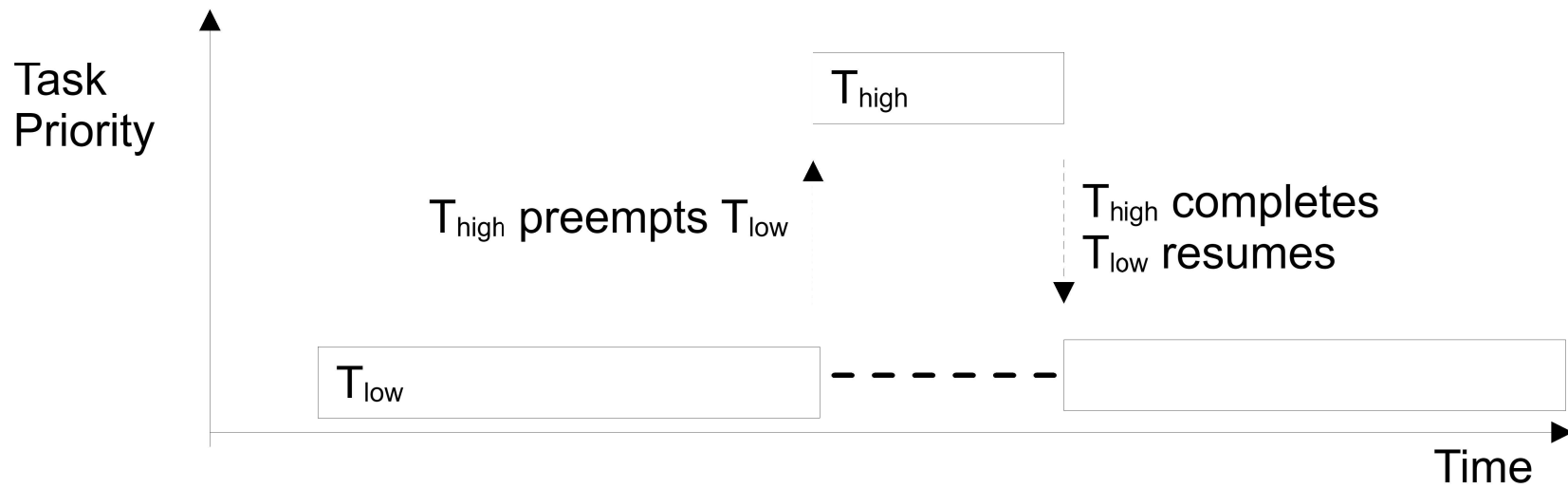
EDF vs FIFO/RR

- RR and FIFO
 - Developer must set a task priority to meet deadlines
 - ..
- EDF
 - ..
 - a task which misses its deadline does not take down other tasks

Priority Inversion

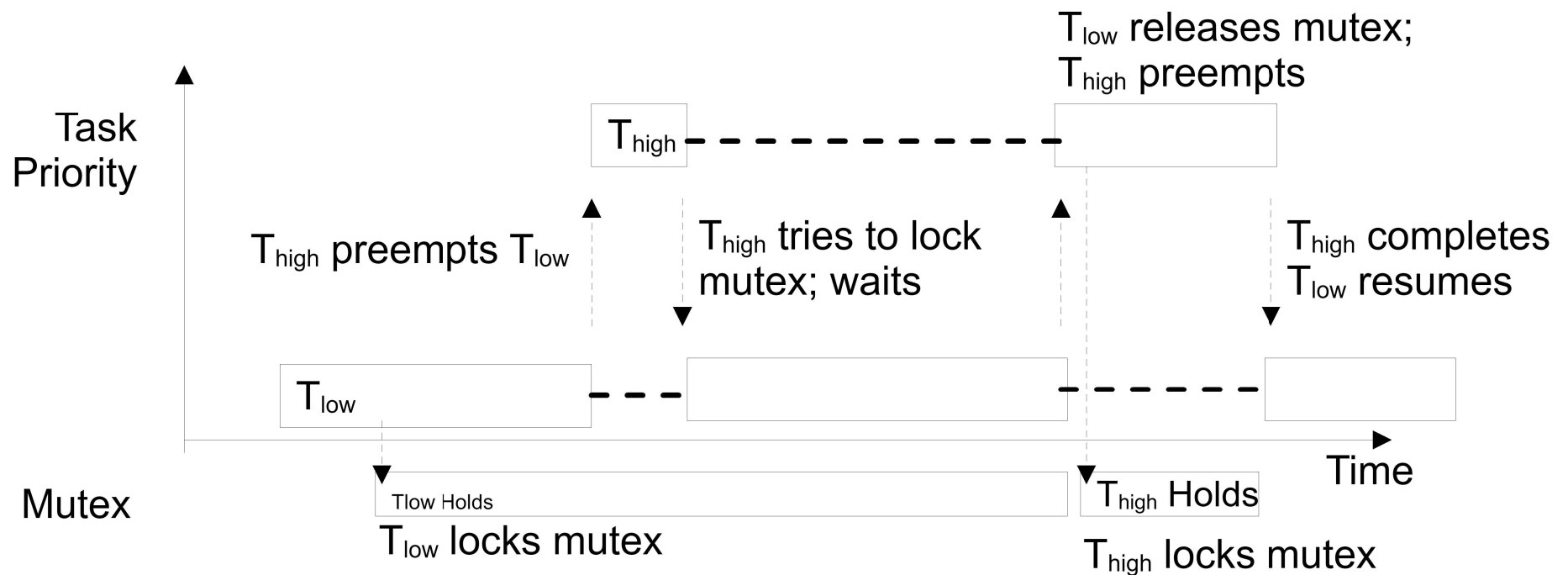
How Priority Works

- The simple case of preemption with priority



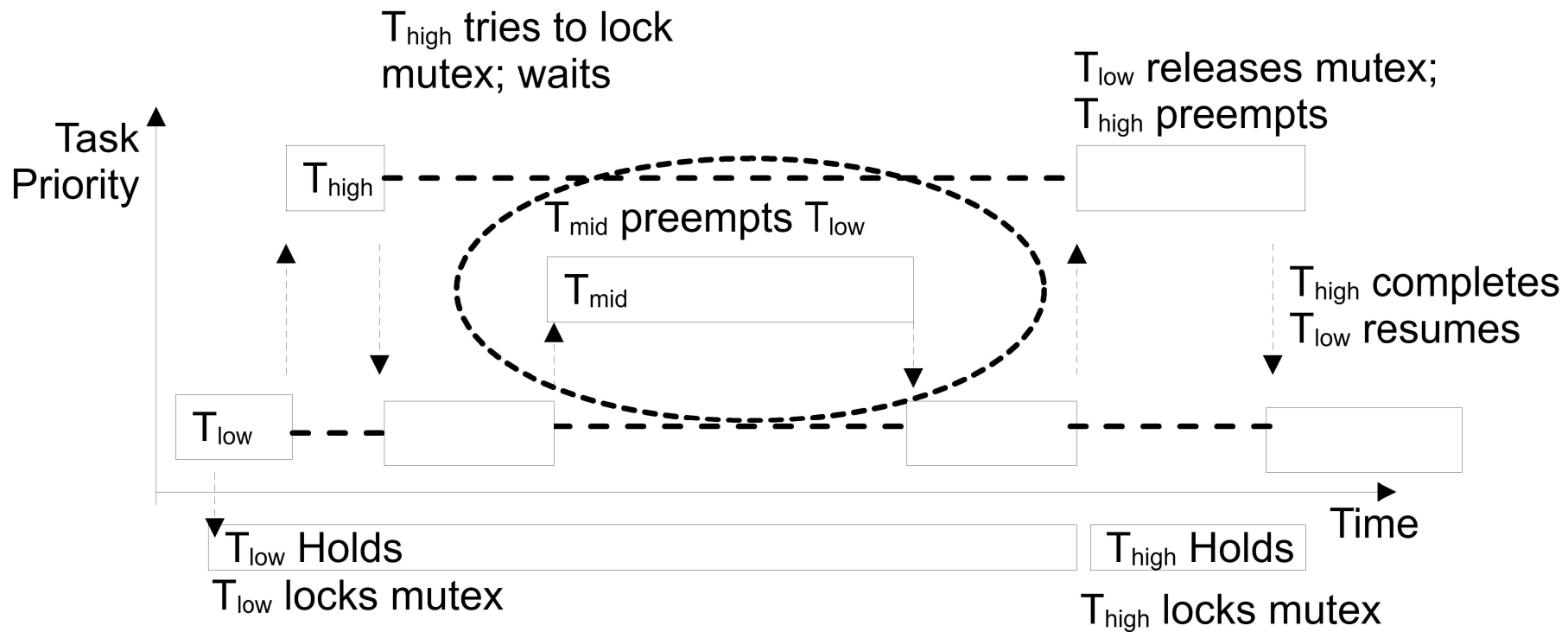
Priority with Mutexes

- What if processes of different priorities share a mutex?



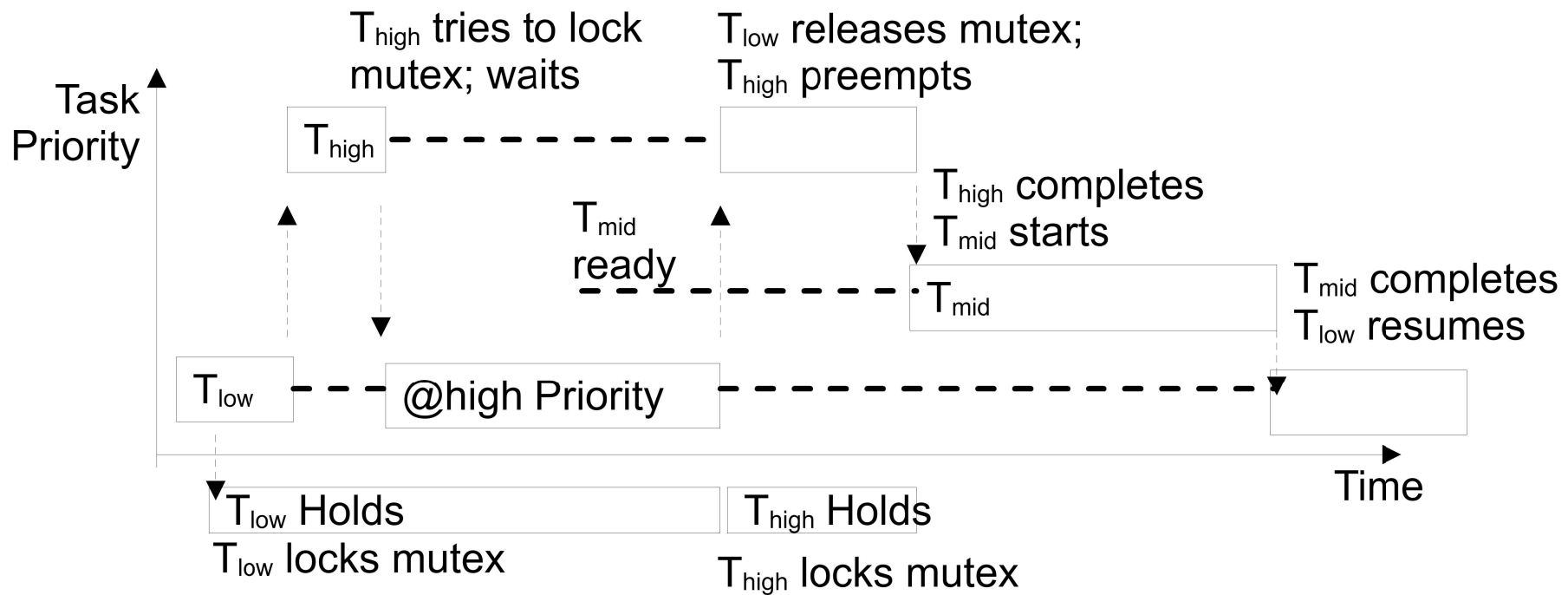
What could go wrong?

- Mutex held between tasks of two priorities with other tasks preempting low priority task..



Solution: Priority Inheritance

- When a higher priority process blocks on a mutex held by a lower priority process, then..



Summary

- OS Schedules tasks
 - Round Robin
 - First-in First-out
 - Completely Fair Schedule
 - Earliest Deadline First
- Priority Inversion
 - When a high priority task is waiting on an resource held by a low priority task, and that low priority task is preempted by a mid priority task.
- Priority Inheritance
 - When holding a mutex, a task's priority is promoted to the priority of other tasks waiting on that mutex.