


CS 415
OS Practicum

Project 2: This time it's interruptible


Project 1 wrap up

- Scheduling: Should start() append or prepend?
- Why is using locks/semaphores a bad idea in yield()?
  - Why is using locks/semaphores a bad idea in fork()?
- Why should your idle thread NOT yield()?


Project 2:Goals

- Learn about interrupts
- Add new functionality
  - Alarms
  - Sleep
- More advanced scheduling


Meet the clock

- Interrupts
  - Like a function call... without the call
- Clock Interrupts
  - Called every PERIOD μs
  - Initialized with minithread_clock_init()
  - More on the interrupt handler later...


Alarms

- An alarm is a scheduled function call
- alarm.c/h
  - Register an alarm and return an id
  - Deregister an alarm with an id
  - You may also wish to create a trigger function


Alarms (cont.)

- Where do alarms trigger?
- How do you store alarms?
  - You want O(1) check times
- How would you design sleep with this?


Scheduler

- What are Priorities?
- Should priorities be dynamic?
- Can we detect which processes should have a high priority?
- Let's build that in...


Multilevel Queues

- Just an array of queues
- dequeue(queue) -> dequeue(queue, id)
- Each queue level represents a priority level


Scheduling Mentality

What is a well behaved thread?

How much time should a well behaved thread get?

How can we codify this into a set of rules?


Scheduler Rules

- Only threads at the highest priority level can run. Choose between these in a round-robin manner. If none are runnable at this level, repeat the process at the next level.
    - Threads at priority level 't' get to run for 2^(3-t) interrupts before they are forced to yield().
- Threads start at the highest priority (level 3).
- A thread that is replaced (via yield) drops to the next lowest priority level (down to the lowest level 0).
- A thread that has not run for 10 interrupts ascends to the next highest priority level. (This is on a per-thread basis)
- A thread that stops() and is restarted is placed at the highest priority.


Back to Interrupts

What does the interrupt handler do?


Back to Interrupts

- What does the interrupt handler do?
  - Check for alarms (and run if needed)
  - Maintain the global ‘ticks’ variable
  - Promote ignored threads
  - yield()

fin