xv6©-rev10
(Copyright Frans Kaashoek, Robert Morris, and Russ Cox.)

sleep syscall

Carmi Merimovich

Tel-Aviv Academic College

February 20, 2017

- The `sleep` system call suspends execution of the process for the number of ticks supplied by the argument.
- There is an argument!!
- It must be checked carefully!!
- The `sys_sleep` implementation is very simple:
  - It is assumed each clock tick declares an event with id `&ticks`.
  - `sys_sleep` waits for the `&ticks` event.
  - When sys_sleep resumes execution, it checks if it was suspended for long enough.
  - If not it returns to the event waiting.

Variables in `sys_sleep`

- `n`: Number of ticks to wait.
- `ticks`: Global variable containing the number of ticks from boot.
- `tickslock`: A spinlock protecting `ticks`.

sys_sleep

```
3815  sys_sleep(void) {
       int n;
       uint ticks0;

       if (argint(0, &n) < 0)  return -1;
       acquire(&tickslock);
       ticks0 = ticks;
       while (ticks - ticks0 < n) {
        if (myproc()->killed) {
         release(&tickslock);
         return -1;
        }
        sleep(&ticks, &tickslock);
       }
       release(&tickslock);
       return 0;
      }
```

ticks?!

Somehwere in the code the following should happen:

- ticks increments.
- Event &ticks is declared.

In trap():

```
  case T_IRQ0+IRQ_TIMER:
    if (cpuid() == 0) {
      acquire(&tickslock);
      ticks++;
      wakeup(&ticks);
      release(&tickslock);
    }
    lapiceoi();
    break;
```