(good) implementation of a spin lock

```
struct lock {
    int held = 0;
}

void acquire (lock) {
    while test-and-set(&lock->held);
}

void release (lock) {
    lock->held = 0;
}
```

Busy wait (a.k.a spin)

- Waste of CPU time
- Unfair access to lock

(good) implementation of a spin lock

```
struct lock {
     int held = 0;
}

void acquire (lock) {
     while test-and-set(&lock->held);
}

void release (lock) {
     lock->held = 0;
}
```

Busy wait (a.k.a spin)

- Waste of CPU time
- Unfair access to lock

(bad) implementation of a sleeping lock

```
struct lock {
}

void acquire (lock) {
    disable_interrupts();
}

void release (lock) {
    enable_interrupts();
}
```

➡ Disabling interrupts blocks notification of external events that could trigger a context switch

◉ Can miss or delay important events

◉ The thread is no longer preemptive