

Implementazione lock

Uniprocessore

Per garantire mutua esclusione è sufficiente disabilitare gli interrupt (impedendo quindi il context switch), ma:

- tenerli disabilitati riduce la reattività del sistema, e
- codice utente buggato o malevolo può monopolizzare il processore.

```
acquire() {                                release() {  
    disable_interrupts();                  disable_interrupts();  
    if (value == BUSY) {                  if (!waiting.empty()) {  
        waiting.add(TCB);                t = waiting.remove();  
        suspend();                      sched.add_ready(t);  
    } else {                            } else {  
        value = BUSY;                  value = FREE;  
    }                                 }  
    enable_interrupts();                enable_interrupts();  
}                                         }
```

suspend: context switch, rimozione del thread dai ready, abilitazione interrupt e rilascio spinlock (sotto).

Multiprocessore

Spinlock con istruzioni RMW:

```
spinlock_acquire() {                      spinlock_release() {  
    while (test_and_set(spinlock));      spinlock = 0;  
}                                         WMB();  
}
```

usato per proteggere la sezione critica di acquire e release:

```
acquire(){                                release() {  
    spinlock_acquire();                  spinlock_acquire();  
    disable_interrupts();                disable_interrupts();  
    if (value == BUSY){                  if (!waiting.empty()) {  
        waiting.add(TCB);                t = waiting.remove();  
        suspend();                      sched.add_ready(t);  
    } else {                            } else {  
        value = BUSY;                  value = FREE;  
        spinlock_release();            }  
        enable_interrupts();          spinlock_release();  
    }                                 }  
}
```