第 0076 讲 6 自旋锁机制

一、自旋锁基础部分

自旋锁是专门为防止多处理器并发而引入的一种锁，它在 Linux 内核中大量应用于中断处理等部分，它是实现保护共享资源而提出一种锁的机制，自旋锁（spinlock）是 Linux 内核最常见的锁机制。自旋锁特性如下：

- 忙等待的锁机制；
- 同一时刻只能有一个内核代码路径可以获取该锁；
- 自旋锁可以在中断上下文中使用；
- 要求自旋锁持有者尽快完成临界区的执行任务。

include > linux > C spinlock_types.h > ...

```
20  typedef struct raw_spinlock {
21      arch_spinlock_t raw_lock;
22  #ifdef CONFIG_DEBUG_SPINLOCK
23      unsigned int magic, owner_cpu;
24      void *owner;
25  #endif
26  #ifdef CONFIG_DEBUG_LOCK_ALLOC
27      struct lockdep_map dep_map;
28  #endif
29  } raw_spinlock_t;
30
```

自旋锁基本形式：

```
spin_lock(&mr_lock);
// 中间部分临界区
spin_unlock(&mr_lock);
```

自旋锁用于处理器之间的互斥，适合保护非常短的临界区，并且不允许在临界区睡眠。申请自旋锁的时候，如果自旋锁被其他处理器占有，本处理器自旋等待（称忙等待）。进程、硬中断及软中断都可以使用自旋锁。

Linux 内核的自旋锁是排队自旋锁（FIFO）。

二、申请与释放自旋锁 API 函数如下：

include > linux > C spinlock.h > spin_lock_nested(lock, subclass)

```
327  static __always_inline void spin_lock(spinlock_t *lock)
328  {
329      raw_spin_lock(&lock->rlock);
330  }
331
```

include > linux > C spinlock.h > spin_unlock(spinlock_t *)

```
367  static __always_inline void spin_unlock(spinlock_t *lock)
368  {
369      raw_spin_unlock(&lock->rlock);
370  }
371
```

其余源码分析 spinlock.h 头文件设计即可。

特别备注：

1 只有在内核抢占或多处理器环境才需要它；

2 单 CPU 就不存在抢占，所以此操作为空。

存在缺陷：

死锁和占用 CPU 资源。