10.1 信号量

信号量像是一种上锁机制，代码必须获得对应的钥匙才能继续执行，一旦获得了钥匙，也就意味着该任务具有进入被锁部分代码的权限。一旦执行至被锁代码段，则任务一直等待，直到对应被锁部分代码的钥匙被再次释放才能继续执行。

信号量分为两种：二进制信号量与计数型信号量，二进制信号量只能取 0 和 1 两个值，计数型信号量不止可以取 2 个值，在共享资源中只有任何可以使用信号量，中断服务程序则不能使用。

1、二进制信号量

某一资源对应的信号量为 1 的时候，那么就可以使用这一资源，如果对应资源的信号量为 0，那么等待该信号量的任务就会被放进等待信号量的任务表中。在等待信号量的时候也可以设置超时，如果超过设定的时间任务没有等到信号量的话那么该任务就会进入就绪态。任务以“发信号”的方式操作信号量。可以看出如果一个信号量为二进制信号量的话，一次只能一个任务使用共享资源。

2、计数型信号量

有时候我们需要可以同时有多个任务访问共享资源，这个时候二进制信号量就不能使用了，计数型信号量就是用来解决这个问题的。比如某一个信号量初始化值为 10，那么只有前 10 个请求该信号量的任务可以使用共享资源，以后的任务需要等待前 10 个任务释放掉信号量。每当有任务请求信号量的时候，信号量的值就会减 1，直到减为 0。当有任务释放掉信号量的时候信号量的值就会加 1。

有关信号量的 API 函数如表 10.1.1 所示。

函数	描述
OSSemCreate()	创建一个信号量
OSSemDel()	删除一个信号量
OSSemPend()	等待一个信号量
OSSemPendAbort()	取消等待
OSSemPost()	释放一个信号量
OSSemSet()	强制设置一个信号量的值

表 10.1.1 信号量 API 函数

10.1.1 创建信号量

要想使用信号量，肯定需要先创建一个信号量，我们使用函数 OSSemCreate()来创建信号量，函数原型如下：

```
void   OSSemCreate ( OS_SEM        *p_sem,
                     CPU_CHAR      *p_name,
                     OS_SEM_CTR    cnt,
                     OS_ERR        *p_err)
```

p_sem： 指向信号量控制块，我们需要按照如下所示方式定义一个全局信号量，并将这个信号量的指针传递给函数 OSSemCreate()。

```
OS_SEM      TestSem;
```

p_name: 指向信号量的名字。

cnt： 设置信号量的初始值，如果此值为 1，代表此信号量为二进制信号量，如果大于 1 的话就代表此信号量为计数型信号量。

p_err： 保存调用此函数后的返回的错误码。

10.1.2 请求信号量

当一个任务需要独占式的访问某个特定的系统资源时，需要与其他任务或中断服务程序同步，或者需要等待某个事件的发生，应该调用函数 OSSemPend()，函数原型如下：

```
OS_SEM_CTR  OSSemPend ( OS_SEM    *p_sem,
                        OS_TICK   timeout,
                        OS_OPT    opt,
                        CPU_TS    *p_ts,
                        OS_ERR    *p_err)
```

p_sem： 指向一个信号量的指针。

timeout： 指定等待信号量的超时时间(时钟节拍数)，如果在指定时间内没有等到信号量则允许任务恢复执行。如果指定时间为 0 的话任务就会一直等待下去，直到等到信号量。

opt： 用于设置是否使用阻塞模式，有下面两个选项。

OS_OPT_PEND_BLOCKING 指定信号量无效时，任务挂起以等待信号量。

OS_OPT_PEND_NON_BLOCKING 信号量无效时，任务直接返回。

p_ts： 指向一个时间戳，用来记录接收到信号量的时刻，如果给这个参数赋值 NULL，则说明用户没有要求时间戳。

p_err： 保存调用本函数后返回的错误码。

10.1.3 发送信号量

任务获得信号量以后就可以访问共享资源了，在任务访问完共享资源以后必须释放信号量，释放信号量也叫发送信号量，使用函数 OSSemPost()发送信号量。如果没有任务在等待该信号量的话则 OSSemPost()函数只是简单的将信号量加 1，然后返回到调用该函数的任务中继续运行。如果有一个或者多个任务在等待这个信号量，则优先级最高的任务将获得这个信号量，然后由调度器来判定刚获得信号量的任务是否为系统中优先级最高的就绪任务，如果是，则系统将进行任务切换，运行这个就绪任务，OSSemPost()函数原型如下：

```
OS_SEM_CTR  OSSemPost ( OS_SEM   *p_sem,
                        OS_OPT    opt,
                        OS_ERR   *p_err)
```

p_sem： 指向一个信号量的指针

opt： 用来选择信号量发送的方式。

OS_OPT_POST_1 仅向等待该信号量的优先级最高的任务发送信号量。

OS_OPT_POST_ALL 向等待该信号量的所有任务发送信号量。

OS_OPT_POST_NO_SCHED 该选项禁止在本函数内执行任务调度操作。即使该函数使得更高优先级的任务结束挂起进入就绪状态，也不会执行任务调度，而是会在其他后续函数中完成任务调度。

p_err： 用来保存调用此函数后返回的错误码