**多线程通讯机制之信号量**

1. **简介**

**OK,我们简单的学习了我们的线程之后，其实有的同学已经有所发现了。由于我们的线程可以共享进程的地址空间，因此，对于一些数据。比如，全局变量，我们的多个子线程都可以访问到它。这样，既有优点，也有缺点，优点就是我们**

**我们可以通过全局变量来实现数据的共享。缺点就是，假如A线程此时正在修改共享的数据，而我们的B线程却认为我们的共享数据没有更改过。这样我们的到的信息，就是有问题的。为了解决这样的问题，我们的引入了线程之间的通讯机制。**

1. **线程之间的通讯机制**

**由于线程之间共享进程的资源和地址空间，对这些资源进行操作的时候，必须考虑线程之间同步与互斥的问题。我们线程之间的通讯机制有以下三种：**

**<1>信号量**

**<2>互斥锁**

**<3>条件变量**

**OK，正式了解之前，大家需要先了解一下一些基本术语**

**=🡺详见《linux基本术语了解》**

**三.线程间的同步-------信号量[可用于多个线程之间]**

**<1>基本概念了解**

**说我们的信号量之前，我们来说说我们的是同步。**

**思考：什么叫做同步？**

**首先，有很多人有个误区，认为同步就是平时说的两件事情同时进行。这个是错误的说法。这里的“同”字意思为协同步调的意思。同步的意思就是按照预定的先后次序依次进行的依次。可理解为：相互之间配合，按照一定的顺序，完成任务。这个过程就是我们的同步。**

**OK，我们再来举个通俗的例子：**

**假如，你刚刚追到手了一个女孩，你现在去它的楼下等她，等到她之后。等到她之后，你们一起去吃饭。吃完饭之后，你们去篮球场上，她静静的看着你打篮球，望着你那帅气的姿势，她笑了。**

**以上例子中，按照一定的顺序执行相应的操作，这个就是同步。**

**<2>信号量**

**我们线程之间的同步使用的是信号量。**

1. **什么是信号量？它用来干嘛？**

**信号量是多线程环境下使用的一种通讯机制。它被用来保护共享资源，用来协调两个或者两个以上的多个线程，以它们能够正确、合理的使用公共资源。使得资源在一个时刻只有一个线程所拥有。**

1. **信号量的分类有哪些呢？**

**我们的信号量用户态来划分的话，我们分为有名信号量和无名信号量。**

**无名信号量对应我们的POSIX标准而言，常用于线程之间的通信。**

**有名信号量是对我们的SytemV标准而言的，常用于进程之间的通信。**

1. **信号量的使用方法**

**信号量是对资源的抽象。资源为0的时候，当前线程会阻塞。通常信号量是使用一个整形数。线程只能对其进行两个实施两个操作：P(prolagen[减少])操作和V(verhogen[增加])操作.这里我们理解为P为申请资源，V为释放资源。P操作可以是我们信号量的值减1，而V操作回事我们的信号量的值加1。以我们的读写竞争资源来为例。我们肯定是希望线程先写入数据，写入数据的过程中，不被读线程打扰。可以先让读线程阻塞。当我们的写线程完成之后，我们就可以唤醒线程，此时写线程阻塞。只使用读线程。这个想法就可以用我们的P-V操实现。**

**信号量的操作：**

**注意:资源个数为0的时候，当前线程会阻塞。**

**申请资源[P操作：资源个数减1]**

**释放资源[V操作：资源的个数加1]**

1. **初始化资源------- sem\_write = 1,sem\_read = 0 ;(写先操作)**
2. **创建两个线程,read\_thread ,write\_thread.**

**write\_thread**

**{**

**//此时读资源为0，写资源为1.先申请写资源。写资源申请后**

**//会为0，程序向下执行。下次再碰到P若是资源为0，会阻塞。**

**P(sem\_write); //申请获得写资源。**

**Write(); //写的一些操作。**

**V(sem\_read); //释放读资源。sem\_read = sem\_read + 1;**

**}**

**read\_thread**

**{**

**//刚开始的时候sem\_read线程的资源为0，线程阻塞，当sem\_read线程被释放之后，sem\_read的值为1，此时获得资源。进行操作。**

**P(sem\_read); //申请获得读资源。**

**read(); //读的一些操作。**

**V(sem\_write); //释放读资源。sem\_write = sem\_write + 1;**

**}**

**四.信号量常用函数**

**常用操作：**

1. **初始化资源**

**注意:**

**(1)信号量是一个受保护变量，不能直接修改它的值**

**(2)线程用sem\_t 描述一个信号量**

**int sem\_init(sem\_t \*sem, int pshared, unsigned int value);**

**参数:**

**@sem 信号量地址**

**@pshared 0:线程间使用 非0:进程间使用**

**@value 初始化的值**

**返回值:**

**成功返回0,失败返回-1**

1. **申请资源[P操作]**

**int sem\_wait(sem\_t \*sem)**

**功能:**

**申请资源，如果资源为0则阻塞调用者**

**参数:**

**@sem 信号量地址**

**返回值:**

**成功返回0,失败返回-1**

1. **释放资源[V操作]**

**int sem\_post(sem\_t \*sem);**

**功能:**

**释放资源，如果有线程等待资源则唤醒等待的线程**

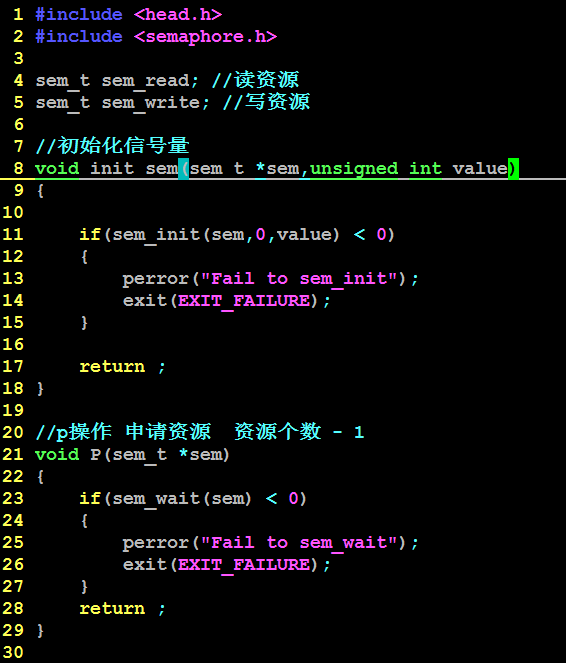
**参数:**

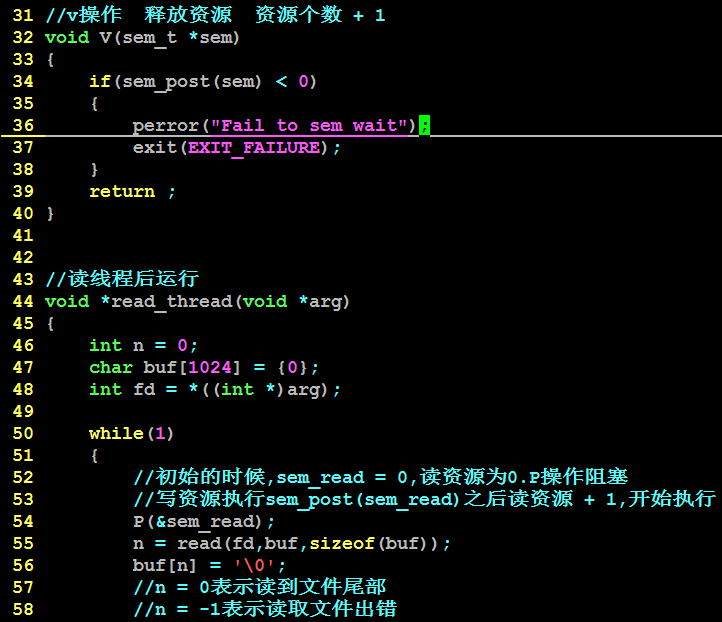
**@sem 信号量地址**

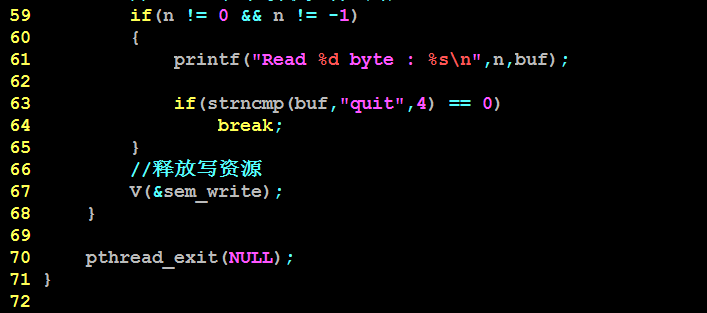
**返回值:**

**成功返回0,失败返回-1**

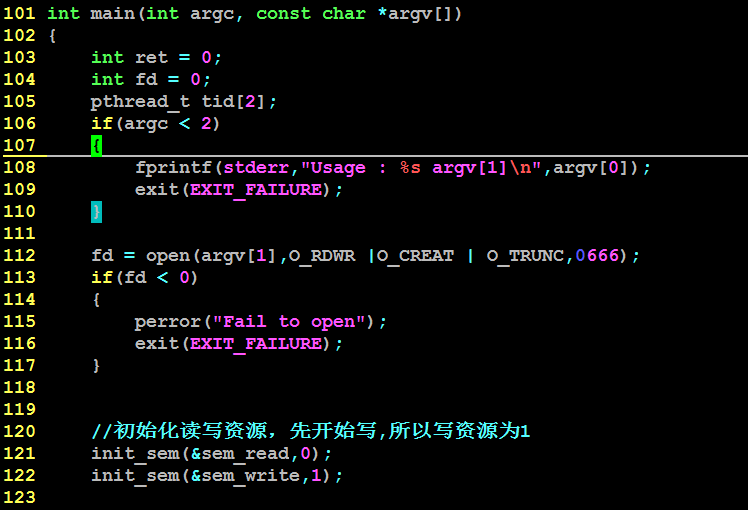
**pthread\_rw\_sem.c**

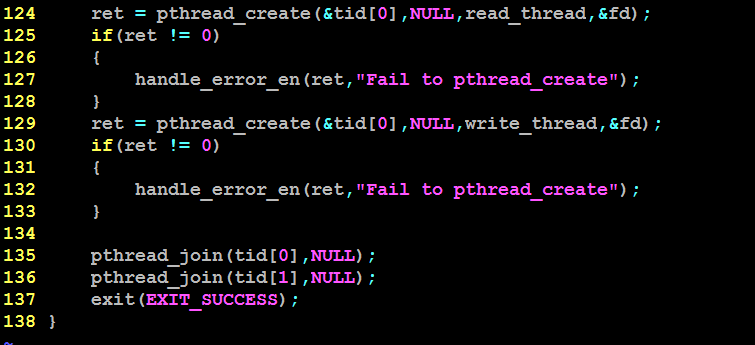












**运行结果：**

