ryan123 L49 2015-06-24 08:29 Linux 计算机专业课

实验代码

“操作系统原理与实践”实验报告

信号量的实现和应用

1,有，消费者消费的数据将不在按照递增顺序输出，并且会出现read error。因为没有了p(empty)操作，所以当生产者已经写满 了缓冲区仍然要进入生产时，就会覆盖掉还没有被消费者取走的数据，这样当消费者进入缓冲区取出数据时，就不是正常的递增顺序了，假设生产者在缓冲区中写了400，401，402，403，404，405，406，407，408，409共10个数据，那么还继续写，就会把400变为410，这样当消费者进入取数时，就会出现399，410，401，这样的输出顺序，同样的，因为没有了p(full)操作,当缓冲区的数据都被取走了，而消费者依然进入取数，读出的数据就是无效的。而没有p(mutex)v(mutex)这个加锁和解锁的操作，使得临界区代码可能出现多进程并发访问的情况，因此会有read error的情况出现。 2,不可行。会有死锁情况的发生，当mutex = 1，并且生产者要进入生产一个产品，假设此时empty.value为0，那么会有mutex.value = 0,p(empty)后得到的value小于0，生产者进程进入等待在信号量empty的等待队列上面，并调用schedule （）,可是此时并未解锁，即mutex.value值仍然为0。它们都等待在信号量mutex上面。同理，消费者进程也是如此，若mutex.value = 1,full.value = 0,在执行完p(mutex)p(full)之后，mutex.value = 0,并且将消费者进程放入等待在信号量full的等待队列上面，而此时也并未释放mutex,因此消费者和生产者进程都等待在mutex信号量上面。