OS-lab4

2112750011 李润泽
email：2112750011@smail.nju.edu.cn
实验进度：我完成了所有内容

1. 实现格式化输入函数(irqHandle.c)

- 完成keyboardHandle()函数：将按键的扫描码存储在键盘缓冲区中，并处理阻塞在标准输入上的进程，使其变为可运行状态。
- 完成syscallReadStdIn()函数：首先判断STD_IN对应的信号量的进程队列中是否有进程阻塞（value？0），如果没有，则将当前进程阻塞在dev[STD_IN]上（这里sleepTime设为-1表示无限期阻塞），然后唤醒进程，将键盘缓冲区的内容读入到用户缓冲区，读取数据；如果已有进程阻塞在标准输入上，则返回-1

2. 实现信号量（irqHandle.c）

- 完成syscallSemInit()函数：初始化信号量，首先寻找是否有可用的空闲信号量，若没有，直接返回-1，若有，将对应信号量state设为1占用，value设为传入值，返回相应信号量的索引
- 完成syscallSemWait()函数：P操作，首先考虑传入的信号量索引值是否合法（0~MAX_SEM_NUM），然后再检查对应信号量的state是否为1，为1说明对应信号量存在，value自减，然后检查当前value是否小于0，如果小于，则将当前进程置为阻塞态，加入对应信号量的进程队列，最后进行进程切换
- 完成syscallSemPost()函数：V操作，首先考虑传入的信号量索引值是否合法（0~MAX_SEM_NUM），然后再检查对应信号量的state是否为1，为1说明对应信号量存在，value自增，然后检查当前value是否小于等于0，如果是，唤醒信号量进程队列中的一个进程，将其置为就绪态
- 完成syscallSemDestroy()函数：首先检查信号量是否存在（state==1），若存在才可以销毁信号量，将state置为0，并显式进程切换

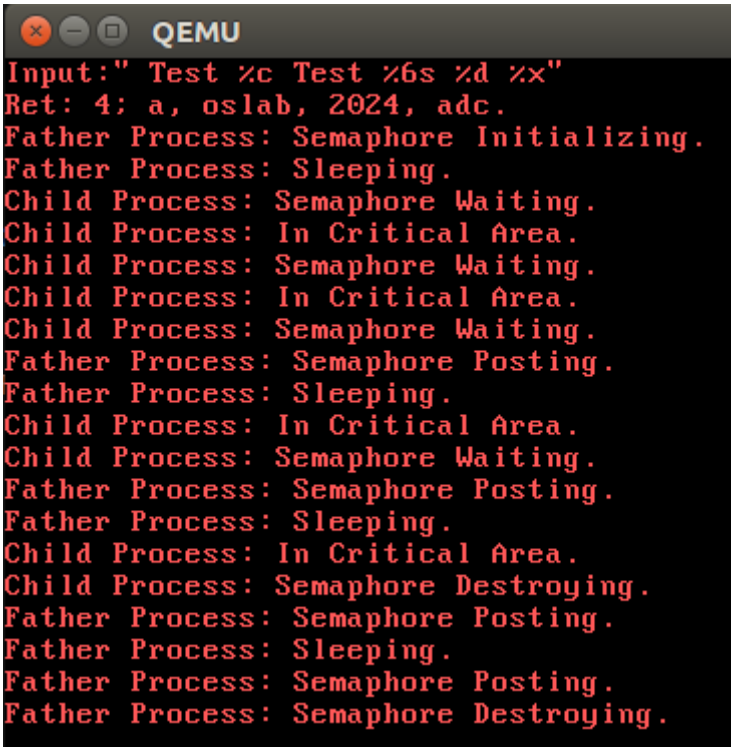
3. 解决进程同步问题（main.c）

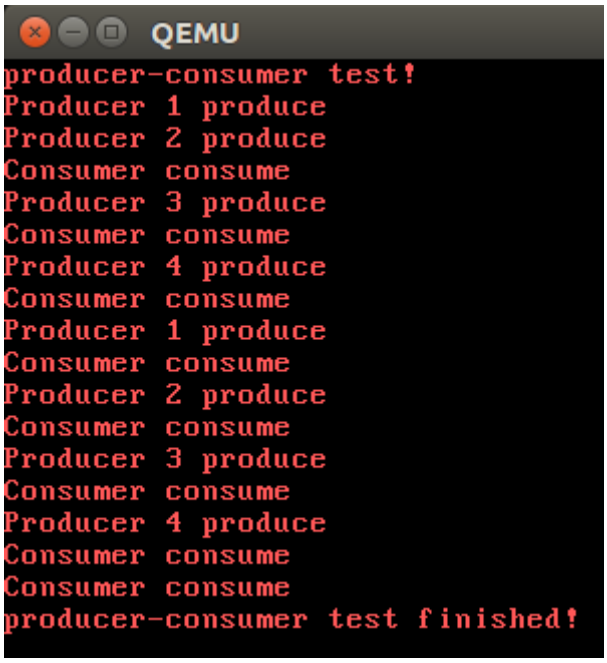
- 解决生产者消费者问题：通过创建四个生产者进程和一个消费者进程，利用信号量 empty、full 和 mutex 来协调它们之间的操作，验证了生产者-消费者问题的解决方案，确保生产者在缓冲区有空槽时生产，消费者在有产品时消费，并通过互斥信号量避免竞态条件，实现正确同步。具体的，设置缓冲区大小为2，每个生产者生产两个产品就停止。

4. 实验结果

在lab4文件夹下执行如下命令：

```
chmod +x utils/genBoot.pl # 需要首先将genBoot.pl文件提权
chmod +x utils/genKernel.pl # 需要首先将genKernel.pl文件提权
make
make play
```


QEMU
Input:" Test %c Test %6s %d %x"
Ret: 4; a, oslab, 2024, adc.
Father Process: Semaphore Initializing.
Father Process: Sleeping.
Child Process: Semaphore Waiting.
Child Process: In Critical Area.
Child Process: Semaphore Waiting.
Child Process: In Critical Area.
Child Process: Semaphore Waiting.
Father Process: Semaphore Posting.
Father Process: Sleeping.
Child Process: In Critical Area.
Child Process: Semaphore Waiting.
Father Process: Semaphore Posting.
Father Process: Sleeping.
Child Process: In Critical Area.
Child Process: Semaphore Destroying.
Father Process: Semaphore Posting.
Father Process: Sleeping.
Father Process: Semaphore Posting.
Father Process: Semaphore Destroying.


QEMU
producer-consumer test!
Producer 1 produce
Producer 2 produce
Consumer consume
Producer 3 produce
Consumer consume
Producer 4 produce
Consumer consume
Producer 1 produce
Consumer consume
Producer 2 produce
Consumer consume
Producer 3 produce
Consumer consume
Producer 4 produce
Consumer consume
Consumer consume
producer-consumer test finished!