**第二章5 实现生产者和消费者的同步问题：**

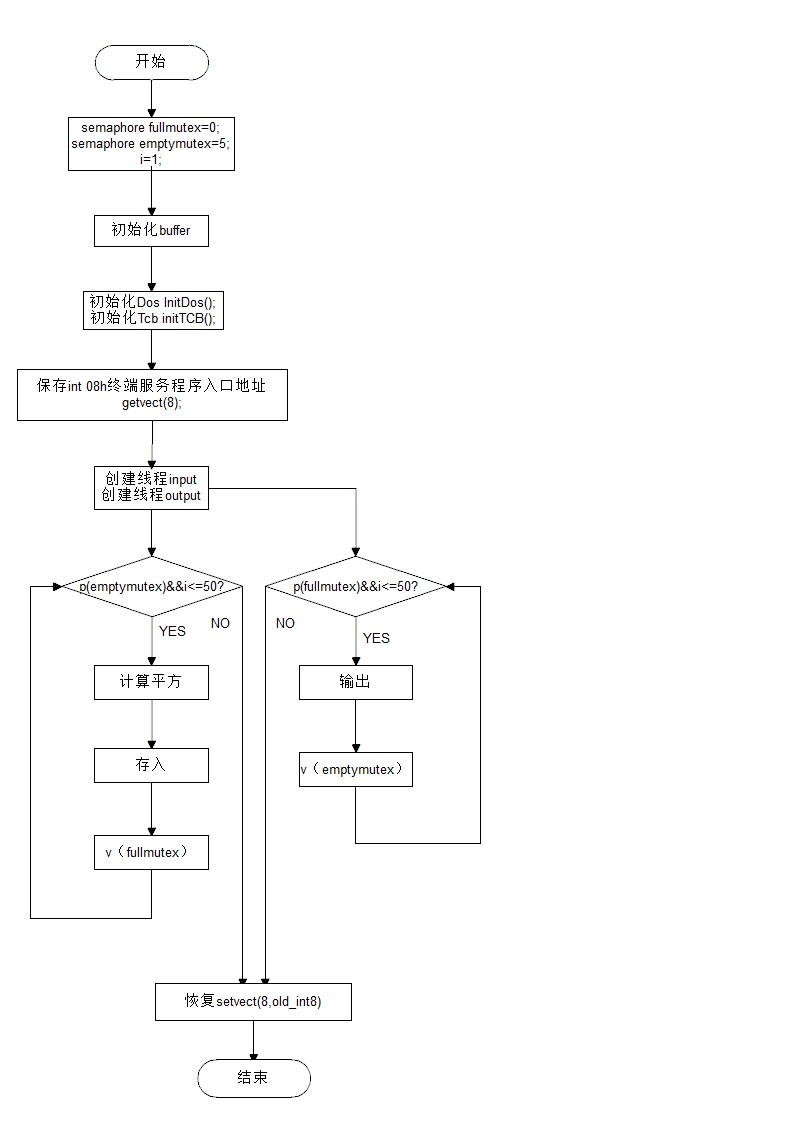
**总体设计思路**：

设置两个信号量，一个表示空的数量emptymutex，初始值为5，一个表示满的信号量fullmutex，初始值为0。

输入函数input在计算的之前先申请emptymutex，若申请到，则说明有空余的位置，emptymutex减一，计算后给fullmutex减一，表示占用了一个。

同理函数output在输出前线申请fullmutex，若申请到，则表示到有值待输出，fullmutex减一，然后输出它，给emptymutex加一，表示输出了一个。

**程序流程图：（与设计的时候有一些改正）**

****

**设计及调试过程中遇到的问题及解决办法**

主要问题是前期并没有考虑仔细，而且对互斥和同步的概念不是很明确，只涉及了一个mutex，结果发现难以实现，后来就改成了两个，然后就能实现了。

**后续改进思路**

输出的时候没有延时睡眠，然后导致演示结果一闪而过，后来加上了sleep（），就可以较慢地看到整个过程了。

**第二章6 实现消息缓冲通信**

**总体设计思路**：

Sender函数首先通过接收者的标识符找到它的tcb id，然后申请一个空闲缓冲区，再申请空闲缓冲区的互斥信号量，获取空闲缓冲区，把它从空闲缓冲区删除，并将信息填入获取到的空闲缓冲区内，最后申请接受者的缓冲队列的互斥信号量，插入信息，释放互斥信号量，并给它的数量加1.

Receiver函数首先通过发送者的标识符找到它的tcb id，然后在自己的消息缓冲队列里面寻找是否存在发送者的消息，如果没有则返回，否则申请自己的消息缓冲队列的互斥信号量，然后把它移除，释放互斥信号量，给它的数量减一，把信息复制出来，申请空闲缓冲区的互斥信号量，然后插入这个缓冲区，给它的数量加1，释放空闲缓冲区的互斥信号量，

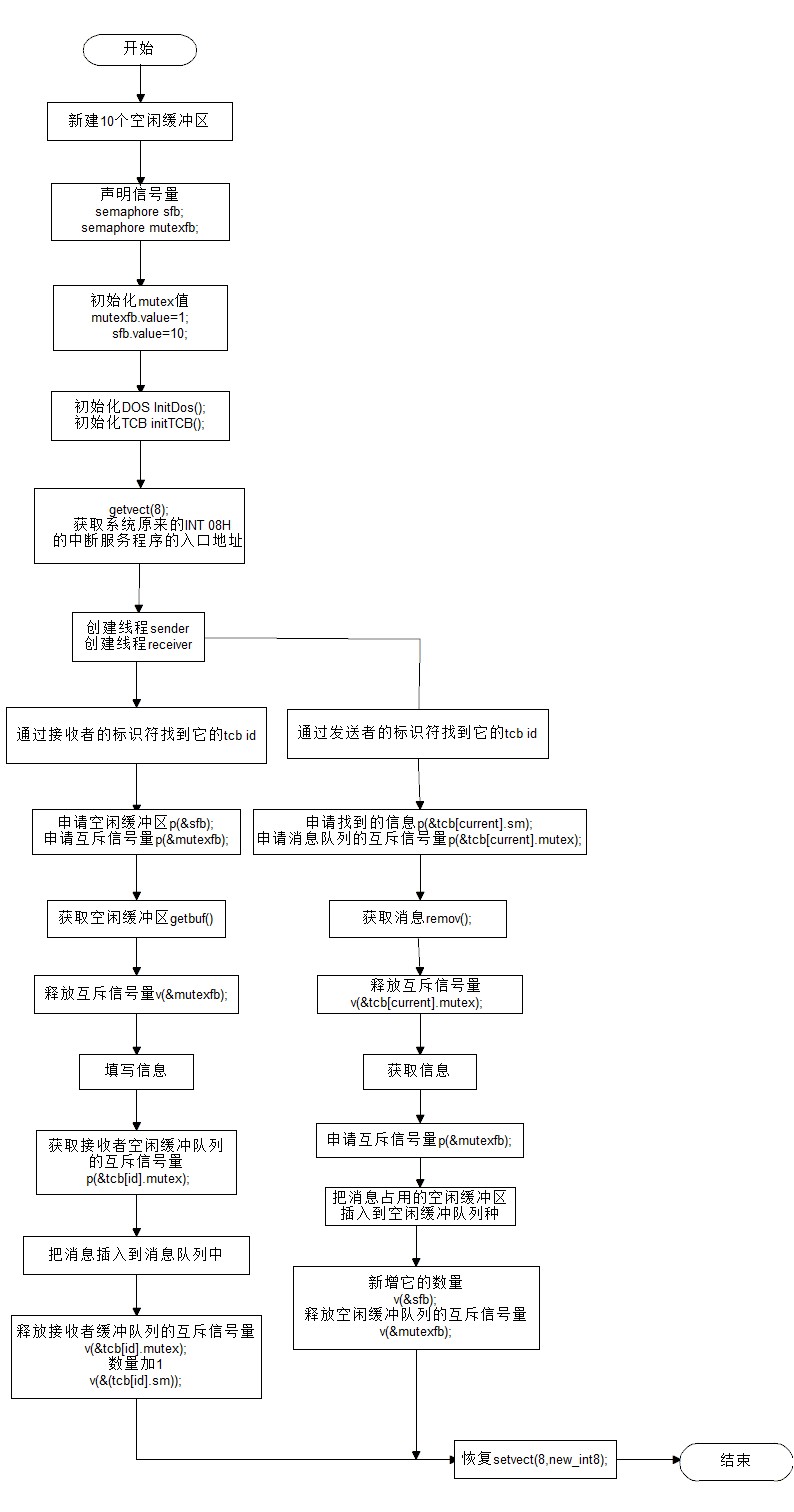
**设计及调试过程中遇到的问题及解决办法**

1. 主要问题是因为一时疏忽，在查找tcb id的时候忽略了第一个就是要找的id，单独扫描第一个就好了。
2. 信号量申请的顺序反了，偶尔会出现问题，斟酌一下，考虑全面就可以了。

**后续改进思路**

复制信息的时候使用循环太过麻烦，所以直接换成了strcpy，方便了许多。

**流程图：（与设计的时候有一些改正）**

****

**第四章6 实现两个进程对共享内存的互斥使用**

**总体设计思路**

**Main函数初始化共享控件的互斥信号量sem\_id ，writer函数先申请一个固定名称的共享内存，然后将它与固定地址连接，writer申请信号量，得到以后请用户输入信息，接着释放信号量，于是进入循环读取缓冲区的状态，reader函数也先找到该名称的共享内存，然后将它与固定的地址连接，接着不断得循环申请信号量，申请到了以后，读取并输出信息，并将共享内存的内容置为”over”，等待writer读取，然后它断开共享缓存的连接，并退出。另一方面writer不断地循环读取缓存区，当读取到的内容是over时销毁信号量也断开连接，然后返回退出。**

**设计及调试过程中遇到的问题及解决办法**

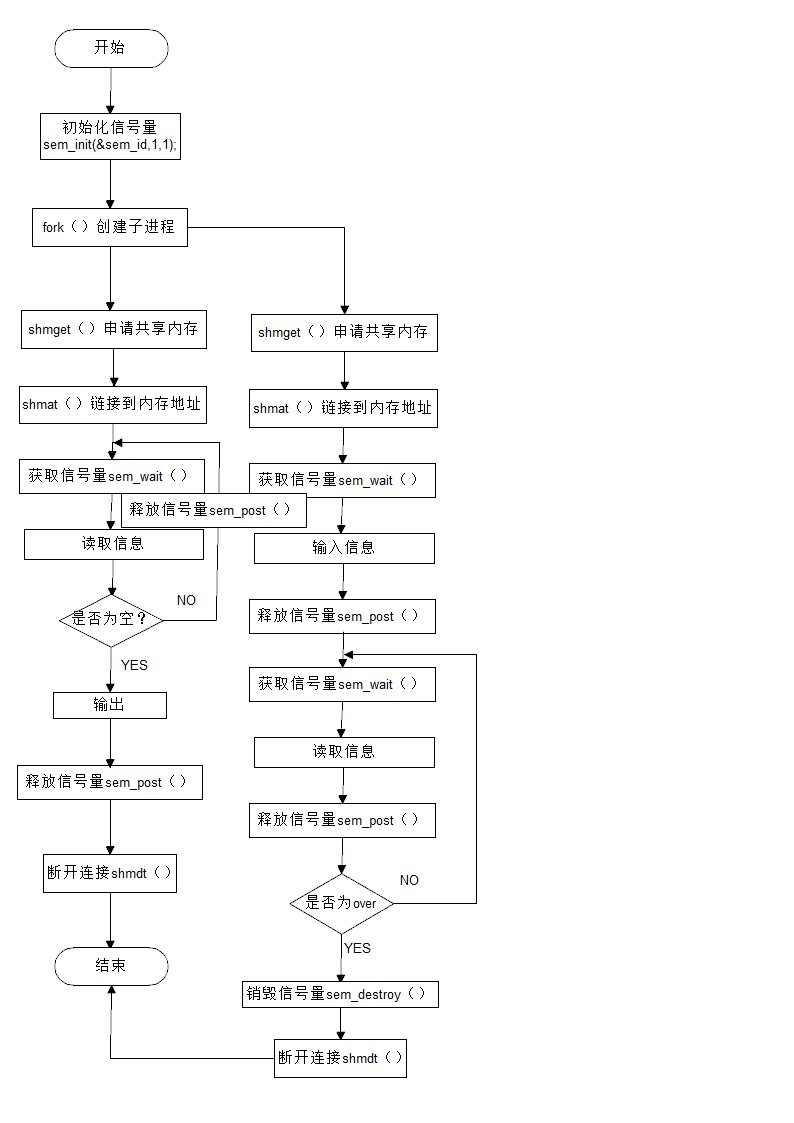
1. 对于fork的机制不是很了解，刚开始创建进程的时候运行完全乱套，之后彻底地学习了一下，fork（），顺便学习了vfork（），就顺利多了。
2. 因为么注意到，没有指定同一个名称，共享内存申请到的不是同一块区域，然后一直收不到消息

**后续改进思路**

暂无

**流程图**

**（与设计的时候有一些改正）**

****