任务三：进程间通信 mbox 实现

中国科学院大学 操作系统研讨课

2017.11.15

1. 介绍

本次作业主要实现一个 FIFO 的 mailbox 处理。

1.1. 需要了解的部分

mailbox 的实现以及应用场景

2. 初始代码

2.1. 文件介绍

- Makefile: 编译文件。
- bootblock.s: 内核启动程序，**请使用作业一中自己写的代码**。
- Createimage.c: 生成内核镜像的Linux工具，**请使用作业一中自己写的代码**。
- Entry.S: 时钟中断处理函数。
- kernel.c: 内核最先执行的文件，放在内核的起始处。
- scheduler.c: 调度器，实现*task*的调度。
- syslib.S: 系统调用函数
- syslib.c: 系统调用接口
- interrupt.c: 系统调用和中断处理相关的函数。
- queue.c: 队列处理函数，提供了队列操作的一些接口。

- print*.c：提供一些输出函数，可以用于调试以及显示信息。请在本任务开始前了解该文件。
- sync.c: 一些同步操作。
- mbox.c: 邮箱的操作，本次任务需要实现。
- util.c: 提供了一些输出函数，可以用于调试以及显示信息。请在本任务开始前了解该文件。
- file.c: 将所有任务放在File的机构体中，供do_spawn查找使用
- ramdisk.c：提供操作File的一些接口
- settest: 设置需要测试的样例。
- test_*文件夹：针对不同任务提供不同的测试，本次任务需要测试test_sanguo。
- *.h: 相应.c/.S文件的头文件。

2.2. 获取：

课程网站。

2.3. 运行

Makefile 文件提供编译功能。

./settest test_XXX 设置测试对象以及编译

make 编译命令

make clean 对编译产生的文件进行清除

sudo dd if=image of=/dev/sdb 将产生的 image 写进 SD 卡中

在 minicom 中执行 *loadboot* 运行程序

2.4. 注意

无

3. 任务

3.1. 设计和评审

帮助学生发现设计的错误，及时完成任务。学生需要对这次的作业进行全面考虑，在实现代码之前有清晰的思路。学生讲解设计思路时可以用不同的形式，如伪代码、流程图等，建议使用 PPT。

3.1.1. 设计介绍

- mailbox的结构是怎么设计的?采用了什么机制解决producer-consumer问题?

3.2. 开发

要求

mailbox 支持 4 个操作:

1. do_mbox_open(name)创造一个名字为name的IPC的队列(mailbox)并返回(返回一个指向该mailbox的整数)。如果该队列已经存在,则队列的使用计数加1。即允许多个task访问同一mailbox。其中: a) 系统在同一时间最多打开MAX_MBOXEN个mailbox。 b) 每一个mailbox的name不应该超过MBOX_NAME_LENGTH个字节长。 c) 每一个mailbox里面的信息不能超过MAX_MBOX_LENGTH。
2. do_mbox_close(mbox):减少指定mailbox的引用计数。当没有task使用该mailbox时,则对其进行回收。
3. do_mbox_send(mbox,buf,size):向mbox指定的mailbox发送buf指定的长度为size的信息。如果该mailbox已经满了,则该操作阻塞。每次发送的信息长度不超过MAX_MESSAGE_LENGTH。
4. do_mbox_recv(mbox,buf,size):向指定的mailbox里面接收size大小的数据到buf中。
5. do_mbox_is_full(mbox):如果指定的mailbox为满的,则返回1,否则返回 0。

加分项

实现一个支持锁的 do_kill 系统调用，即当一个程序获得锁然后被 kill 掉时，它获得的锁也应该被释放掉。具体实现需满足如下：

- 实现一个测试用例，最少为两个进程task。测试用例需要实现程序获得锁，被kill掉（在释放锁之前），接着其他task运行时获得锁。即需要检测出程序被kill掉后其锁是否被释放。
- 测试用例为进程，即在用户态，start code中的实现的lock是内核态实现的，所以需要自行实现系统调用接口，即lock_init,lock_acquire,lock_release等系统调用接口（可以酌情增加其他接口）。
- Sync.h中的lock_t结构体是在内核态中的，实现的测试用例不能直接使用，所以需要实现新的接口来定义锁（可以使用int来传递变量）。

3.3. 注意事项

mailbox 在操作之前进行初始化,否则一切操作不能保证正确性。

mailbox 为一个先进先出的队列,即发送的消息只能加到队尾,接收消息从队首取得。

4. 测试

本次任务需要使用 test_sanguo 文件夹里的程序测试 mailbox 和 kill，wait 的实现，通过运行 test_sanguo 文件夹中的程序,测试 LiuBei 和 SunQuan 能否互相 wait, 并会被 CaoCao 随机 kill 掉后，又会由对方通过 spawn 运行。

加分项测试样例需要自己设计，作业检查时请讲解测试样例的设计。