任务三：线程间互斥锁

中国科学院大学 操作系统研讨课

2017.09.27

1. 介绍

本任务实现线程间互斥锁

lock.c 中的三个控制锁的函数：lock_init、lock_acquire、lock_release，分完成互斥锁的初始化、获取和释放。

schedule.c 中线程 block 与 unblock 操作：block、unblock，分别执行线程阻塞和唤醒。

1.1. 需要了解的部分

- 非抢占式调度和context switch
- 互斥锁(Mutex Lock)的原理

2. 初始代码

2.1. 文件介绍

- Makefile: 编译文件。
- bootblock.s: 内核启动程序，使用任务一中实现的代码。
- Createimage.c: 生成内核镜像的Linux工具，使用任务一中实现的代码。
- entry_mips.S: 代码中*task* context switch时需要进行的操作，在本次任务中需要完成scheduler_entry和save_pcb。本文件也提供了获取时间函数。

- kernel.c: 内核最先执行的文件，放在内核的起始处。在本次任务中需要完成多tasks的初始化
- lock.c: 实现一个互斥锁，在本次任务中实现。
- scheduler.c: 调度器，实现*task*的调度，本次任务中需要完成多个tasks轮流运行，以及yield和exit操作。
- syslib.S: 系统调用函数，进程通过系统调用进入内核，本次任务需要了解。
- queue.c: 队列操作函数，在本次任务中，如果非抢占调度器用队列实现，可以直接使用该文件内函数。
- task.c: 将本次任务中所有*task*通过一个task_info的结构体引用。初始化*task*时就可以通过该文件提供的task数组，做本次任务时task数组的值不需要改变。完成不同任务时，请给task数组赋不同的task值，用于不同的测试目的。文件中的注释部分表明每一个任务需要测试的进程或者线程（将注释部分去掉就可以直接使用）。
- process1/2/3.c：定义了测试用例进程，可以分别用于不同的任务。
- th1/2/3/4.c：定义了测试用例内核线程，可以分别用于不同的任务。
- util.c: 提供了一些print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。
- *.h: 相应.c/.S文件的头文件。

2.2. 获取：

课程网站。

2.3. 运行

createimage 为提供的可执行文件，当 createimage.c 实现完成后，将 Makefile 中的 createimage 项去掉注释。

make 命令编译文件

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

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. 设计问题考虑

- 怎样设计互斥锁
- 怎样设计block和unblock函数

4. 开发

4.1. 注意事项

- 已经写好的内核线程程序th4.c调用了对锁的初始化、获取和释放，lock.h中也已经声明好了要实现的函数。所有要实现的函数都在lock.c中。之前的任务中使用的是已提供的一个自旋锁的实现。在本次任务中，需要换成自己写的互斥锁。需要把lock.c中的SPIN改成FALSE或0。
- 进程获得锁失败后需阻塞(block)，需要调用schedule.c：block函数，以及获得锁后需唤醒(unblock)，需要调用schedule.c:unblock，同样需要保存pcb等操作。

4.2. 代码实现中的问题

本次作业中有以下几个地方可能会因为代码实现方式或者编译环境发生改变，如遇到请自即调整：

- bootblock.S中kernel_main的值为内核开始执行第一条指令的地址，请确保此处一致。
- syslib.S中的宏定义SYSCALL跳转的地址为kernel_entry函数的地址，请确保一致。
- 本系统输出为串口输出，可能会因为minicom窗口大小不同显示结果不同，这里可以自行调整或者自己设计
- do_yield()和block()函数都需要首先执行save_pcb函数,此时需要检查之前实现的save_pcb()是否同样满足block()函数的条件。

5. 测试

本任务相当于把已有代码提供的自旋锁换成互斥锁，因此测试时只需要把 lock.c 中的 SPIN 改成 0 或 FALSE 即可，task.c 的测试样例换做该任务的测试样例。下图是 start_code 提供的自旋锁的运行结果，不做为最后评测互斥锁的标准：