

# OSlab1 实验报告

161220117 唐诗美

## 一、实验简述

本次实验实现了一个简单的操作系统内核

## 二、实验心得

本次实验的实验心得如下：

1. 首先是把之前写的 `malloc.c` 里面的东西移植到 `pmm.c` 中，一开始移植过去发现居然 `unistd.h` 也不能用，于是就仿照 PA 写了一个 `sbrk` 和 `brk`，当时 PA 在这里就很懵，但是现在就比较好的理解了这两个东西的实现
2. 接着就是对 `kmt.c` 里面的东西就很懵了，一开始都不知道应该写什么，接着通过不断的上网查资料大概写了雏形，但是 `create` 一开始不是很清楚栈要怎么加载，后来通过理解以及连蒙带猜地大概实现好了线程的栈的分配。
3. 接着差不多写了一些之后就开始编译，然后就编译错（觉得没有程序设计语言课程总是面向编译错写代码…）后来发现是结构没有加入到头文件里面出现了 `incomplete struct`。接着又通过不断 debug 发现了一开始的初始化链表的时候没有分配内存，各种错误（虽然后面没有用链表实现…）；还有在给 `sem->name` 赋值的时候算一个不存在的长度再用这个长度去赋值；有些链表的指针也忘记设置了（虽然最后没有用链表实现觉得这些 bug 真的比较浪费时间了…）最后面向 bug 一点一点地改对了一些出于手残的 bug
4. 后面又发现在 `sem->queue` 里面的时候可能一开始实现不对，然后数组就爆掉了，思考了一下没找到根本原因就把 `queue` 改成了链表实现。（把一开始链表实现的改成数组，一开始数组实现的改成链表…）
5. **最后就是最根本的一个 bug，理解上出现了偏差！**在 `os_interrupt()` 里面就很懵，一开始是直接返回的是由 `schedule()` 返回的线程的寄存器现场，**没有把 `regs` 赋值给当前线程的寄存器**，就相当于每次执行都是从每个函数的最开始那条语句开始执行。而且这个错误最坑的是是要执行一段时间才会发现错误。后来发现错误了之后，又懵了好久，为什么寄存器的现场不变啊？？接着就开始怀疑是不是链表的实现内存是不是没有分配好，于是就把链表改成数组（这个才不是主要原因啊…），又打印寄存器的值，但是没有变化！！然后就很懵地编译了很多次想找出 bug，然而并没有什么用。突然就觉得 `os_interrupt()` 里面传的 `regs` 的那个参数都没用啊，应该不会有没用的参数传过来吧，于是又去看了一下讲义，努力理解了一番，发现似乎应该要把 `regs` 设置为当前线程的寄存器现场。于是按照这个思路就去改。由于开头的中断现场不应该保存在线程的寄存器现场，因为那个不是线程的，但改了一番之后并没有发现这个问题，于是就开始运行了。接着就触发了如下图的 bug…把终端都给卡住了。然后又 de 了几次 bug 还会出现这种报错。

```

WARNING: Image format was not specified for 'build/os.img' and probing
Automatically detecting the format is dangerous for raw image.
Specify the 'raw' format explicitly to remove the restriction.
qemu: fatal: Trying to execute code outside RAM or ROM at 0x6602e6c1

EAX=00000000 EBX=00100013 ECX=00000000 EDX=00000008
ESI=0020432c EDI=0010008f EBP=66fff100 ESP=000fff9f
EIP=6602e6c1 EFL=00000086 [--S--P-] CPL=0 II=0 A20=1 SMM=0 HLT=0
ES =0010 00000000 ffffffff 00cf9300 DPL=0 DS [-WA]
CS =0008 00000000 ffffffff 00cf9a00 DPL=0 CS32 [-R-]
SS =0010 00000000 ffffffff 00cf9300 DPL=0 DS [-WA]
DS =0010 00000000 ffffffff 00cf9300 DPL=0 DS [-WA]
FS =0000 00000000 0000ffff 00009300 DPL=0 DS16 [-WA]
GS =0000 00000000 0000ffff 00009300 DPL=0 DS16 [-WA]
LDT=0000 00000000 0000ffff 00008200 DPL=0 LDT
TR =0028 0010b960 00000063 00408900 DPL=0 TSS32-avl
GDT= 00108720 0000002f
IDT= 00104fa0 000007ff
CR0=00000011 CR2=00000000 CR3=00000000 CR4=00000000
DR0=00000000 DR1=00000000 DR2=00000000 DR3=00000000
DR6=ffff0ff0 DR7=00000400
CCS=0000001f CCD=fee00000 CCO=LOGICL
EFER=0000000000000000
FCW=037f FSW=0000 [ST=0] FTW=00 MXCSR=00001f80
FPR0=0000000000000000 0000 FPR1=0000000000000000 0000
FPR2=0000000000000000 0000 FPR3=0000000000000000 0000
FPR4=0000000000000000 0000 FPR5=0000000000000000 0000
FPR6=0000000000000000 0000 FPR7=0000000000000000 0000

```

虚拟机也开始在崩溃边缘徘徊(虽然觉得触发这个东西和虚拟机不好应该没有太必然的联系吧…不是明白虚拟机的不稳定)。后面就发现了 `os_interrupt()` 第一次进去的时候的寄存器现场应该不要保存，是返回 `schedule()` 中的那个线程的寄存器。后来改了之后就基本正确的实现了

6. 在改好 `os_interrupt()` 后又有一个微妙的地方。现在也还不是很懂。我实现信号量是用 `while()` 在一个队列中阻塞进程的接下来的执行。但在那个 `while()` 循环里面必须要添加一句 `printf("")` 才能够正确执行??? 我觉得是很懵了…希望能在以后的学习中找到这个问题的答案。

本次实验觉得讲义一开始讲的有些地方非常的不清楚，靠上网各种查资料然后连蒙带猜地实现，还有 `os_interrupt()` 的实现，不知道每个中断处理函数是什么样的，于是只是简单地打印一下这个是这个中断处理函数。觉得可能并不应该是这么处理，但似乎不是很影响。比如页错误的那个，都不知道分页的机制在哪里实现，就觉得比较懵了。如果是在以后的实验中有实现的话可以先提一下吧。

以及小小吐个槽：os 实验最近感觉比较频繁，再加上其他有些课程比较困难，感觉有时候时间不是很有用了。