操作系统Lab2——系统调用

151220045 蒋雨霖

一. 实验目的:

本实验通过实现一个简单的应用程序，并在其中调用一个自定义实现的系统调用。实现系统调用库函数printf，完善printf的格式化输出。

二. 实验思路

1.首先仿照LAB1系统引导修改start.S文件,从实模式进入保护模式,与LAB1不同的是不用初始化GS寄存器，以及删掉GDT表中的视频段。

2.之后修改bootloadr中的boot.c，仿照PA中loader利用调用elf实现加载内核至内存。跳转至内核kernel继续执行。

3.内核初始化IDT（Interrupt Descriptor Table，中断描述符表）：即中断号0xd和0x80 其余中断号未加(不加可过)。初始化GDT。然后在kvm.c中初始化tss。

4.内核加载用户程序至内存:实现函数loadUmain，代码格式类似于实现boot.c的方法，从201扇区开始加载。需注意的是需要判断ProgramHeader *ph, ph->type==1才进行memcpy和memset操作。对内核堆栈进行设置。

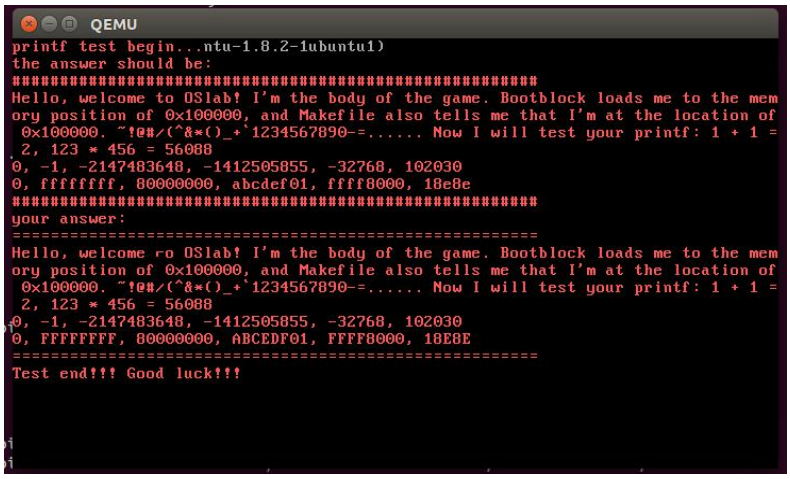
之后在enterUserSpace函数中通过iret切换至用户空间:需push eflags、push cs，最后push 入口地址，执行用户程序。

5.最后实现syscall.c中的syscall函数和printf函数格式化。

Syscall函数中利用内嵌汇编保存6个参数至通用寄存器。

而实现prtinf格式化利用到了头文件stdarg.h,通过对format的分析并利用switch case语句，将参数d，s，x，c情况下需要打印的内容存入一个总的数组当中，最后调用syscall实现打印内容到显存上。输出数组当中内容。注意由于输出屏幕大小的限制，需加入’\n’的判断，判断何时换行，以保证打印在正确的位置。

三. 实验效果截图:



四. 实验心得及建议：

这次实验相比第一次复杂了许多，心得：多参考 PA 中的内容，来实现 lab2，深刻理解整个系统调用的过程。

建议：应该适当在在线网页及课程中多给出些提示，以及相对于 LAB1 的三周，LAB2 两周时间显得过短。