**操作系统课程设计实验报告**

实验题目： 实验二添加系统调用实现定时关机

姓 名： 王一凡

学 号： 19061536

专 业： 软件工程

班 级： 19052713

老师姓名： 刘真

日 期： 2021 年 4 月 21 日

目 录

[一 题目介绍 1](#_Toc515881940)

[二 实验思路 1](#_Toc515881941)

[三 遇到问题及解决方法 1](#_Toc515881942)

[四 核心代码及实验结果展示 1](#_Toc515881943)

[五 个人实验改进与总结 1](#_Toc515881944)

[5.1 个人实验改进 1](#_Toc515881945)

[5.2 个人实验总结 1](#_Toc515881946)

[六 参考文献 2](#_Toc515881947)

八 源代码详单 2

# 一 题目介绍

题目内容：设计一个带参数的模块，其参数为倒计时的秒数，模块功能是让系统倒计时指定秒数后关机。

考察的知识点：系统调用的编写方式，Linux内核的编译步骤、安装步骤，关机指令源码，延时函数源码。

问题的关键点：找到关机指令、延时函数的源码，引入相应的头文件调用

# 二 实验思路

1. 下载新的内核，解压
2. 添加系统调用号
3. 在syscall.h中声明系统调用服务例程
4. 在sys.c中编写延时、关机指令
5. 编译内核，下载缺少的各种包，编译模块，安装模块，安装内核，修改配置文件，换成新的内核

# 三 遇到问题及解决方法

1. 如何实现关机？halt，poweroff，shutdown等，选用哪一个指令对应的函数。通过查看源码发现他们底层都是reboot函数实现的，选出最合适的poweroff实现本次实验
2. 如何实现延时？起初百度搜索内核态下的延时函数，百度到的五花八门，看着很复杂。后面查看关机指令的源码时偶然间看到了delay.h，进去看到了延时ms，s的函数。

# 3、编译太慢？多分配点cpu和内存，还可以使用ccache加快内核编译速度。

# 四 核心代码及实验结果展示

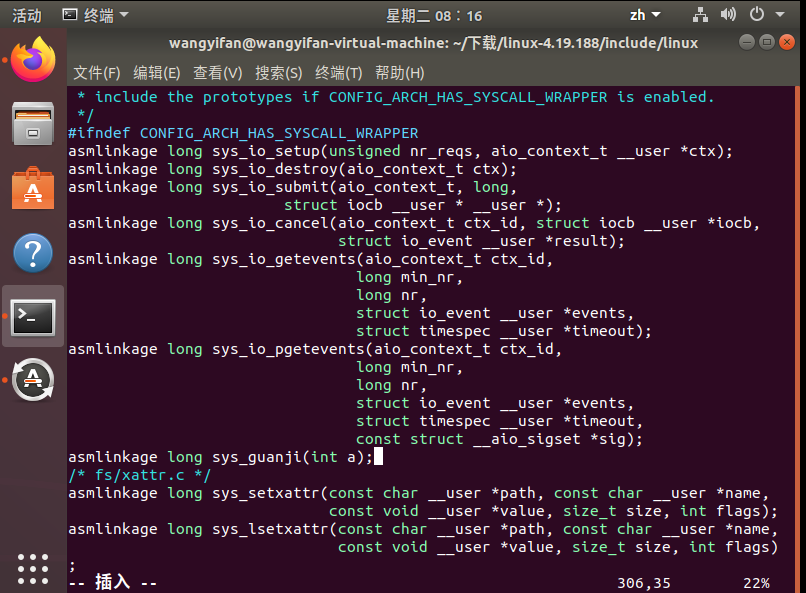
syscall\_define1(guanji,int,a){

ssleep(a);

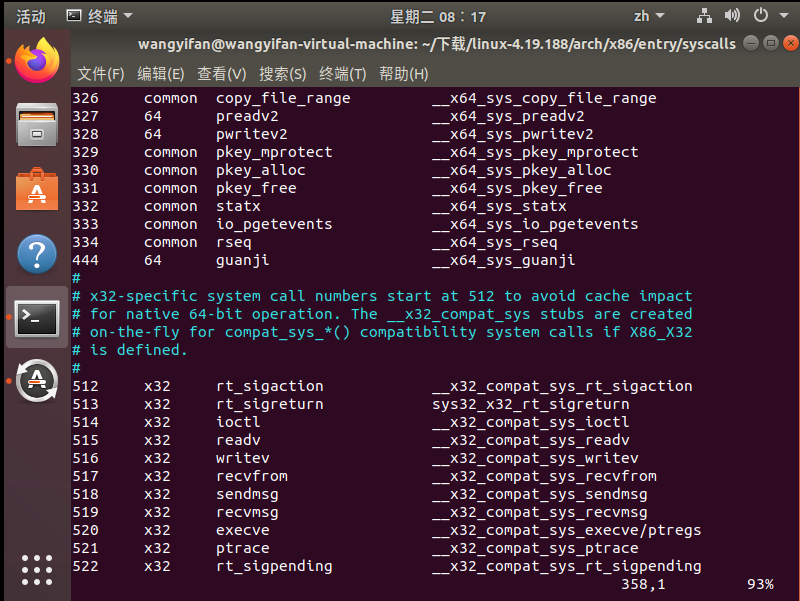
kernel\_power\_off();

return 0;

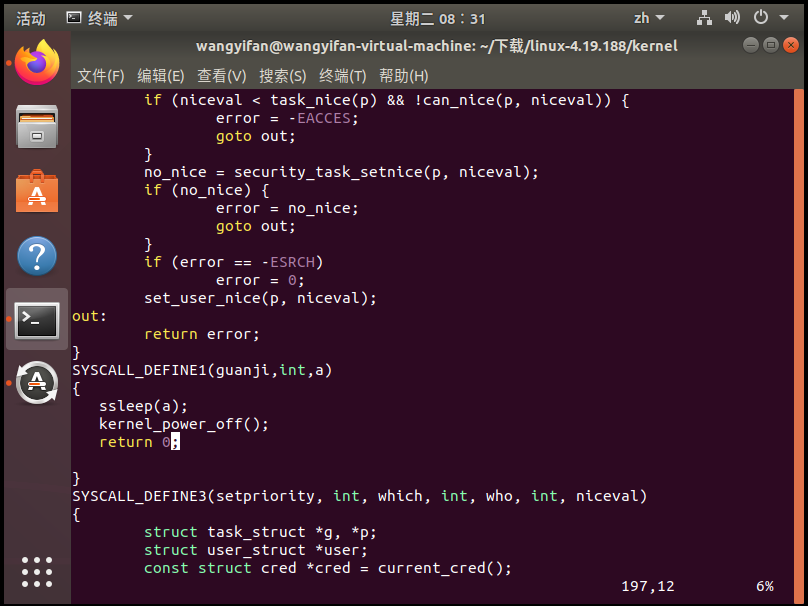
}



图表 1添加系统调用声明



图表 2添加系统调用号



图表 3系统调用核心代码



图表 4成果展示

编译完内核后调用系统调用：  
#include<unistd.h>

#include<sys/syscall.h>

Int main(){

syscall(444,10);}

10s之后系统成功关机



# 五 个人实验改进与总结

## 5.1 个人实验改进

本次实验纯靠看源码实现，起初先百度搜索资料，百度到的都是一些指令，比如shutdown、halt等，后面看到system（“halt”）用C语言实现对电脑的关机，但是显而易见这是用户态下才可以用的，果断放弃。之后开始查询shutdown的源码，发现他们都是一个sys\_reboot的系统调用实现的，因此我先阅读了一边reboot的源码，最后恍然大悟，还有意外的收获，发现了delay.h的定义，看到了延时函数的用法。因此我先在内核态下加装了一个模块，调用halt，实现延时关机，最后发现halt只是对cpu禁用了，电脑什么都操作不了，但是没有完全的关机，因此查阅资料在添加系统调用时采用了kernel\_power\_off这个函数，成功的完成了延时关机，一次成功。

## 5.2 个人实验总结

通过本次实验，我掌握了增加系统调用的基本方法，对整个的流程有了一定的了解。还掌握了Linux内核编译的方法，掌握了Linux内核的编译流程，感受到了Linux的魅力，以后就可以定制自己想要的模块、系统调用了。

最重要的是本次实验网上没有任何的答案，起初做实验几天摸不着头脑，对流程很快的就可以了解，但是对代码的编写一头雾水。最后发现了Linux源码的那个官方网站，通过阅读源码掌握了很多知识。早就知道阅读源码是很好的学习方式，比如jdk，springboot，python库的源码，但是一直胆怯，懒得去读源码，只会百度，Google别人现成的内容。通过这次实验，阅读Linux源码对reboot、delay的了解，掌握了源码的魅力，与解决问题的高效。说不定还有意外的收获，比如这次偶然看到delay.h的文件。

# 六 参考文献

# <https://www.cnblogs.com/harrypotterjackson/p/11846222.html>

操作系统课本

www.kernel.org

https://blog.csdn.net/weixin\_48299611/article/details/115563624?utm\_source=app&app\_version=4.5.8

# 七 源代码详单

Cd /arch/x86/entry/syscalls

vim syscall\_64.tbl

333 444 guanji sys\_guanji

cd /arch/x86/include/asm/

vim syscalls.h

asmlinkage long sys\_guanji(int a);

cd /kernel/sys.c

syscall\_define1(guanji,int,a){

ssleep(a);

kernel\_power\_off();

return 0;

}

### 编译内核

先分两步解压

xz -d linux-4.16.3.tar.xz

tar -xvf linux-4.16.3.tar

清除残留的文件

sudo apt install libncurses5-dev

make mrproper

配置内核，save退出即可，选择默认的

make menuconfig

编译内核

sudo apt install openssl

sudo apt install libssl-dev

sudo apt install bison

sudo apt install flex

make -j4

编译模块

make modules

安装内核

make modules\_install

make install

配置grub引导程序

低版本向高版本编译：update-grub2

高版本向低版本编译：sudo update-initramfs -c -k 5.3.10（版本号）

sudo vi /etc/default/grub

sudo update-grub

重启reboot之后选择advanced，进入新的内核即可