

**实验报告**



**题目： 键盘驱动程序的分析与修改**

**班 级： 2023211321**

**学 号： 2023212872**

**姓 名： 计子毅**

**学 院： 计算机学院**

**2024年 12 月 9 日**

一、实验目的

**1、理解 I/O 系统调用函数和 C 标准 I/O 函数的概念和区别；**

**2、建立内核空间 I/O 软件层次结构概念，即与设备无关的操作系统软件、设备驱动程序和中**

**断服务程序；**

**3、了解 Linux-0.11 字符设备驱动程序及功能，初步理解控制台终端程序的工作原理；**

**4、通过阅读源代码，进一步提高 C 语言和汇编程序的编程技巧以及源代码分析能力；**

**5、锻炼和提高对复杂工程问题进行分析的能力，并根据需求进行设计和实现的能力。**

1. 实验环境

**1、硬件：学生个人电脑（x86-64）**

**2、软件：Windows 10，VMware Workstation 15 Player，32 位 Linux-Ubuntu 16.04.1**

**3、gcc-3.4 编译环境**

**4、GDB 调试工具**

三、实验内容

**解压lab4.tar.gz 文件，解压后进入 lab4 目录得到如下文件和目录：**

**\*\*\*\***

**安装gcc编译器：**

**\*\*\*\***

**实验常用执行命令如下：**

** 执行./run ，可启动 bochs 模拟器，进而加载执行 Linux-0.11 目录下的 Image 文件启动linux-0.11 操作系统**

** 进入 lab4/linux-0.11 目录，执行 make 编译生成 Image 文件，每次重新编译（make）前需先执行 make clean**

** 如果对 linux-0.11 目录下的某些源文件进行了修改，执行./run init 可把修改文件回复初始状态**

**本实验包含 2 关，要求如下：**

** Phase 1**

**键入 F12，激活\*功能，键入学生本人的姓名拼音，首尾字母等显示\***

**比如：zhangsan，显示为：\*ha\*gsa\***

** Phase 2**

**键入“学生本人的学号” ：激活\*功能,键入学生本人的姓名拼音,首尾字母等显示\***

**比如：zhangsan，显示为：\*ha\*gsa\*，**

**键入“学生本人的学号-” ：取消显示\*功能**

**提示：完成本实验需要对 lab4/linux-0.11/kernel/chr\_drv/目录下的 keyboard.s、console.c 和 tty\_io.c 源文件进行分析，理解按下按键到回显到显示频上程序的执行过程，然后对涉及到的数据结构进行分析，完成对前两个源程序的修改。修改方案有两种：**

** 在 C 语言源程序层面进行修改**

** 在汇编语言源程序层面进行修改**

**其他说明见 实验四.ppt 。linux 内核完全注释(高清版).pdf 一书中对源代码有详细的说明和注释。**

四、实验步骤及实验分析

**准备工作：**

* + - 1. 从官网下载VMware workstation 15，用于虚拟机的安装
      2. 从官网下载并在虚拟机上安装32位Linux—Ubuntu 16.04.01；
      3. 从课堂派下载lab4.tar.gz并解压，进入lab4文件夹，将linux-11.0.tar.gz解压

图形用户界面

低可信度描述已自动生成

图表, 漏斗图

描述已自动生成

* + - 1. 通过ppt的连接在虚拟机上安装并配置gcc3.4

wget http://old-releases.ubuntu.com/ubuntu/pool/universe/g/gcc-3.4/gcc-3.4-base\_3.4.6-6ubuntu3\_i386.deb

sudo dpkg -i gcc-3.4-base\_3.4.6-6ubuntu3\_i386.deb

wget http://old-releases.ubuntu.com/ubuntu/pool/universe/g/gcc-3.4/cpp-3.4\_3.4.6-6ubuntu3\_i386.deb

sudo dpkg -i cpp-3.4\_3.4.6-6ubuntu3\_i386.deb

wget http://old-releases.ubuntu.com/ubuntu/pool/universe/g/gcc-3.4/gcc-3.4\_3.4.6-6ubuntu3\_i386.deb

sudo dpkg -i gcc-3.4\_3.4.6-6ubuntu3\_i386.deb

wget http://old-releases.ubuntu.com/ubuntu/pool/universe/g/gcc-3.4/libstdc++6-dev\_3.4.6-6ubuntu3\_i386.deb

sudo dpkg --force-depends -i libstdc++6-dev\_3.4.6-6ubuntu3\_i386.deb

wget http://old-releases.ubuntu.com/ubuntu/pool/universe/g/gcc-3.4/g++-3.4\_3.4.6-6ubuntu3\_i386.deb

sudo dpkg -i g++-3.4\_3.4.6-6ubuntu3\_i386.deb

建议：前4个包在root命令模式下进行解包，第5个包在用户命令模式下进行解包

* + - 1. 安装as86

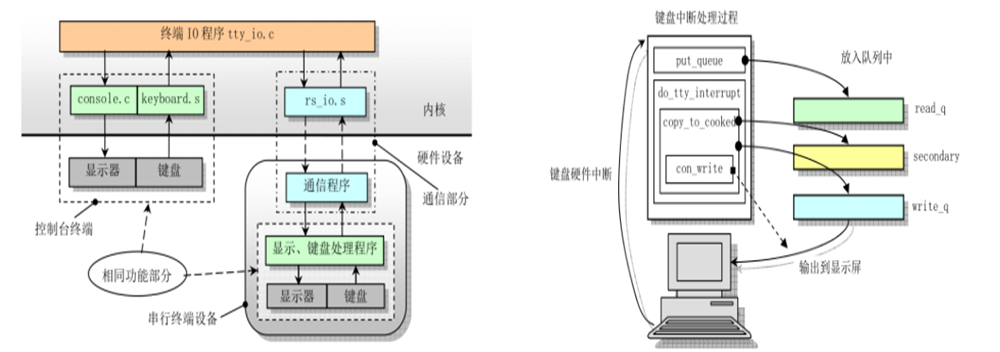
**Phase1：输入姓名拼音，首位字母显示‘\*’**

* + - 1. 在文件夹lab4下./run运行程序，发现按下F开头的键会有4行显示，并且会显示某个对应的大写英文字母
      2. 打开/lab4/linux-0.11/kernal/chr\_drv/keyboard.S文件，找到call show\_stat命令行，注释掉，方便后续实验。

文本

描述已自动生成

* + - 1. 对任务解读则为修改驱动程序，将键盘上的F12键作为开关，开启是将首字母与尾字母改为\*，则jiziyi需要改变的为j和i字符。
      2. 通过查阅资料Linux-0.11系统中可以使用两类终端，一类为主机上的控制终端，另一类是串行硬件终端设备。控制台终端是由‘keyboard.s’和显示控制程序‘console.c’进行管理。所以这两个程序为模拟显示器与键盘的程序。查阅资料得在系统中‘keyboard.s’程序处理用户输入的字符，并放入缓冲弄队列，而‘console.c’实现了将终端收到代码输出。控制台终端对应有一个tty\_struct数据结构，主要用来保存终端设备当前的参数、字符IO缓冲队列等信息，其中有三个缓冲队列，分别是read\_q，write\_q和secondary，分别保存从键盘输入的原始字符序列、写到控制台显示屏的数据、从read\_q取出的经过行规则程序处理的数据。tty\_io.c中包含字符设备的上层接口函数，包括终端读函数tty\_read和终端写函数tty\_write，当文件系统中操作字符设备文件时会调用这两个函数。



* + - 1. 故在‘console.c’中设全局变量ff为0，当F12被按下时，ff值取反，修改当前输出状态。再对‘keyboard.S’查看跳转表，在func函数中发现调用F12按键仅会跳转到func函数

图片包含 表格

描述已自动生成

* + - 1. 在keyboard.S中找到func对应部分，在最后一行添加call指令跳转到我们即将编写的函数changed中

图形用户界面, 文本

描述已自动生成

* + - 1. 打开console.c，添加全局变量f,初始化为0，并添加函数changed使得每按下一次F12使得f的值反转，再在con\_write中添加相应判断，根据f的值可以调整输入的字符c是否显示为‘\*’

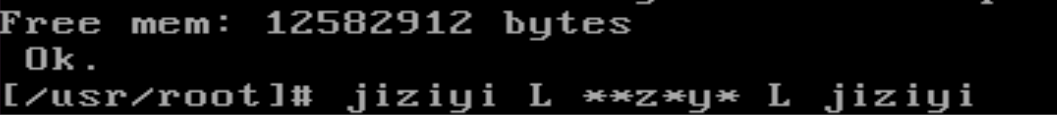
文本

中度可信度描述已自动生成

文本

描述已自动生成

* + - 1. 返回linux-0.11文件夹，使用make clean命令，再使用make命令重新编译整个程序。返回lab4文件夹，再次./run运行bochs,发现按下F12，输入的英文字母j和i都变为‘\*’,再次按下F12，输入的英文字母j和i会被显示



**Phase2：键入“学生本人的学号” ：激活\*功能,键入学生本人的姓名拼音,首尾字母等显示\*；键入“学生本人的学号” ：激活\*功能,键入学生本人的姓名拼音,首尾字母等显示\***

1. 本关需要输入规定字符串，激活\*功能（同阶段一），键入学生本人的姓名拼音，首尾字母等显示\* 比如：jiziyi，显示为：\*\*z\*y\*
2. c. 学号为2023212872编写开启功能，2023212872-关闭功能的C代码如下，数字之间为相互分离的函数，只需要根据shift\_flag值的转变成功调用函数即可

int shift\_flag=0;

void get\_0\_shift(void)

//when get 0

{

if(shift\_flag==1)

shift\_flag=2;

else if(shift\_flag==11)

shift\_flag=12;

}

void get\_1\_shift(void)

//when get 1

{

if(shift\_flag==5)

shift\_flag=6;

else if(shift\_flag==15)

shift\_flag=16;

}

void get\_2\_shift(void)

//when get 2

{

if(shift\_flag==0)

shift\_flag=1;

else if(shift\_flag==2)

shift\_flag=3;

else if(shift\_flag==4)

shift\_flag=5;

else if(shift\_flag==10)

shift\_flag=11;

else if(shift\_flag==12)

shift\_flag=13;

else if(shift\_flag==14)

shift\_flag=15;

else if(shift\_flag==6)

shift\_flag=7;

else if(shift\_flag==9)

shift\_flag=10;

else if(shift\_flag==16)

shift\_flag=17;

else if(shift\_flag==19)

shift\_flag=20;

}

void get\_3\_shift(void)

{

if(shift\_flag==3) shift\_flag=4;

else if(shift\_flag==13)

shift\_flag=14;

}

void get\_8\_shift(void)

//when get 8

{

if(shift\_flag==17)

shift\_flag=18;

else if(shift\_flag==7)

shift\_flag=8;

}

void get\_7\_shift(void)

//when get 7

{

if(shift\_flag==8)

shift\_flag=9;

else if(shift\_flag==18)

shift\_flag=19;

}

void get\_sign\_shift(void)

//when get -

{

if(shift\_flag==20)

shift\_flag=0;

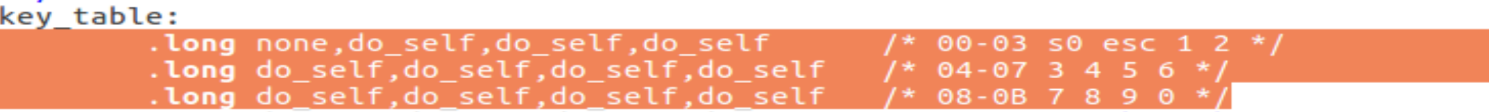
}

1. 调整ccon\_wright如下：

文本

描述已自动生成

4. 打开keyboard.S查看跳转表，发现只需要修改do\_self函数





1. 编辑跳转表如下：

图片包含 文本

描述已自动生成

1. 返回linux-11.0文件夹，make clean一下,再make重新编译。返回lab4文件夹，执行./run运行bochs。先输入jiziyi完全显示，输入2023212872，再输入jiziyi相关字母被隐藏，再输入2023212872-，再输入jiziyi完全显示

文本

描述已自动生成

五、总结体会

实验中关于系统，以及虚拟机和编译环境的安装锻炼了对操作系统的掌握，是实验中较为有难度的部分，因为环境不同bug也不一样，所以会查阅很多博客最后才装好虚拟机以及环境。在关卡方面首先是对Linux内核的基本原理的理解才能找到相对应文件，虽然只是改变一个按键的功能，但背后的原理需要花费大量时间理解，然后在make语句环节发现了先前配系统环境以及编译器的缺漏，在第一关做出来后，第二关在先前阶段一以及上次实验三的基础上较为容易，很容易想到开设记录状态转移的变量，但求得的解法只是满足的合法输入下的功能开启，而会有许多非法输入也能开启功能，尚且为存在的不足，状态转移作为判断的标准还远远不够到实际上手，若在规定字符中间插入其他字符又会有怎么样的结果。考虑到还有许多功能键以及其间相互影响，未去完善这一代码漏洞。

这次实验让我对Linux内核以及I/O设备驱动的原理有了更加清晰的理解，巩固之前所学的知识的基础上，提高了我对计算机的兴趣，使我受益良多。

六、诚信声明（不签扣10分）

需要填写如下声明，并在底部给出手写签名的电子版。

在完成本次实验过程中，我曾分别与以下各位同学就以下方面做过交流：

1、未与其余同学交流

2、

此外，我还参考了以下资料：

1. csdn中关于虚拟机安装的教程

在我提交的程序中，还在对应的位置以注释形式记录了具体的参考内容。

我独立完成了本次实验除以上方面之外的所有工作，包括分析、设计、编码、调试与测试。

我清楚地知道，从以上方面获得的信息在一定程度上降低了实验的难度，可能影响起评分。

我从未使用他人代码，不管是原封不动地复制，还是经过某些等价转换。

我未曾也不会向同一课程（包括此后各届）的同学复制或公开我这份程序的代码，我有义务妥善保管好它们。

我编写这个程序无意于破坏或妨碍任何计算机系统的正常运行。

我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的实验成绩将按照0分计。

（签名）图片包含 图示

描述已自动生成