校徽

**江西师范大学**

**嵌入式系统开发技术**

**毕业实习设计**

专 业： 电子信息工程

班 级： 14级电子信息工程（1）班

学 号： 1408071026 1408071021

姓 名： 胡玉杰 何弘

设计题目： 按键和蜂鸣器驱动程序设计

**2018年1月**

目 录

# 一、概要

目前嵌入式系统已发展成为以高速处理器和嵌入式操作系统作为核心的软硬件综合系统，Linux作为一种开源、跨平台的操作系统，受到了越来越多开发者的青睐。随着物联网和人工智能的发展，Linux将更多地应用在嵌入式设备中，这对Linux内核中的驱动设计和实现也提出了更高的要求。

驱动程序在 Linux 内核里扮演着特殊的角色。它们完全隐藏了设备工作的细节，用户的活动通过 一套标准化的调用来进行, 这些调用与特别的驱动是独立的；设备驱动的角色就是将这些调用映射 到作用于实际硬件的和设备相关的操作上。驱动可以与内核的其他部分分开建立, 并在需要的时候 在运行时"插入"。这种模块化使得 Linux 的驱动程序非常的灵活。

Linux是一套可以免费使用和自由传播的类UNIX操作系统，其实际上只是一个操作系统的内核，主要用于Intelx86系列CPU的计算机上，谈及Linux的起源，起灵感来自于UNIX。UNIX操作系统于1969年由Bell实验室设计开发，之后Linus Torvalds设计了Linux，该系统在发展初期就得到了广大程序员的帮助，逐步发展成为现今这样一个拥有自己版权的完整的系统。Linux具有很多特点，如支持多种体系结构，支持大量的外围设备，具有完善的网络功能，开放源代码，软件资源丰富， 内核稳定等，可以总结为一下几点：

（1）强大的编程能力。由于Linux源自于世界各地成千上万的程序员和黑客，使得Linux就犹如加入到了一个感受如云的编程组织中，同时由于GPL的存在，Linux开放源代码，吸引更多专业人士的加入，在这种需求的刺激下，Linux提供的开发工具功能也越来越完善，越来越强大。

（2）完善的组网能力。Linux具有强大的组网能力，它对当前的TCP/IP协议提供了完全的支持，同时也支持下一代Internet协议Ipv6.在安全性方面，Linux内核中包括了IP防火墙代码、IP防伪及IP服务质量控制等特性。

（3）Linux是自由开放的。Linux和运行在其上的自由软件，允许成千上万的人检查软件，修改软件，最终可以按照用户自己的意愿来定义自由软件，可以定制自己的Linux。

（4）系统稳定。Linux提供了完全的内存保护机制，每个进程都运行在各自的虚拟地址空间中，不会损坏进程或内核使用的地址空间。

# 二、设计内容

本次设计是分别实现蜂鸣器驱动和按键驱动，以及将蜂鸣器驱动和按键驱动进行组合联调，基本熟悉Linux字符设备开发流程，达到毕业实习预期的目的。

# 三、准备工作

## 1、开发环境的搭建

### 1)、安装VMware Workstation和Red Hat Enterprise Linux 5

安装并打开VMware Workstation，如图1所示。

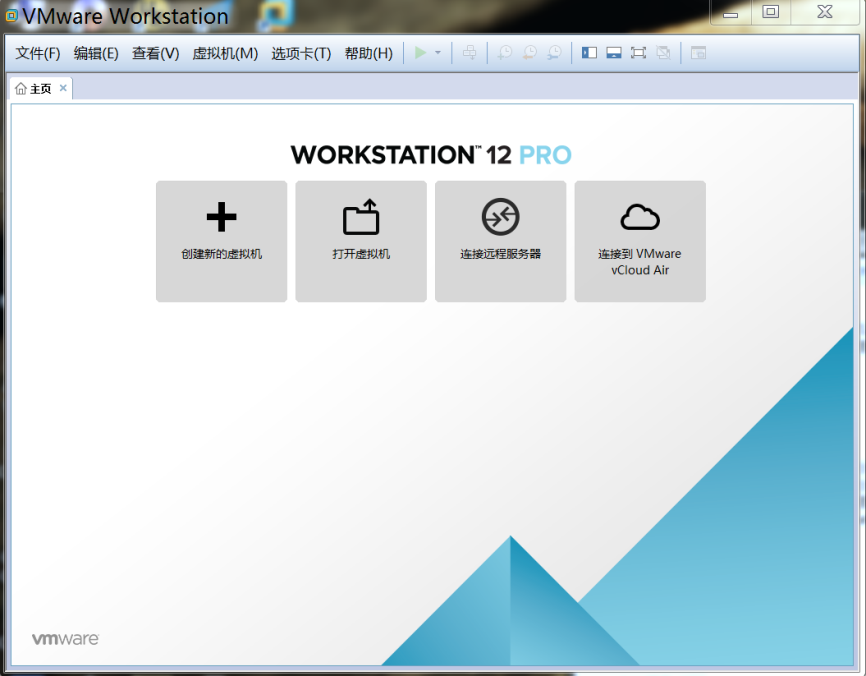


图 1 打开VMware Workstation

解压拷贝到的已安装好的Red Hat Enterprise Linux 5镜像，然后在VMware Workstation中将其打开。

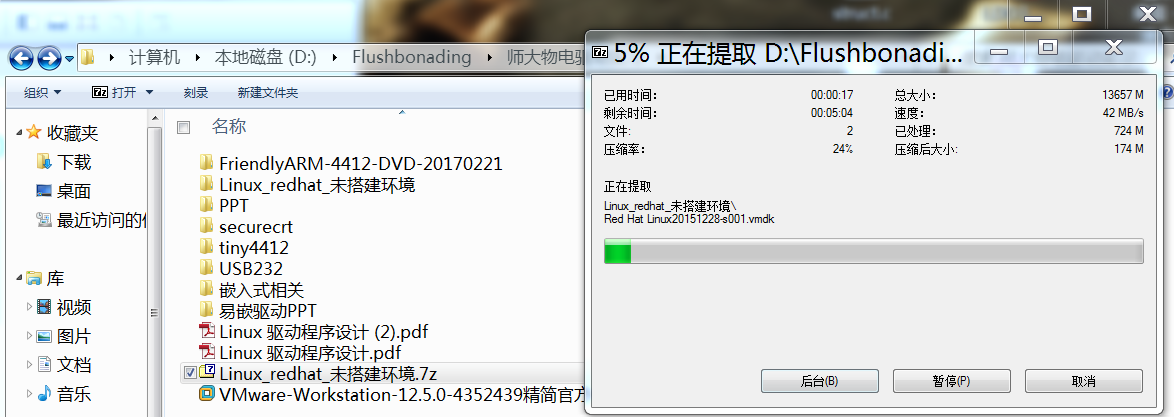


图2

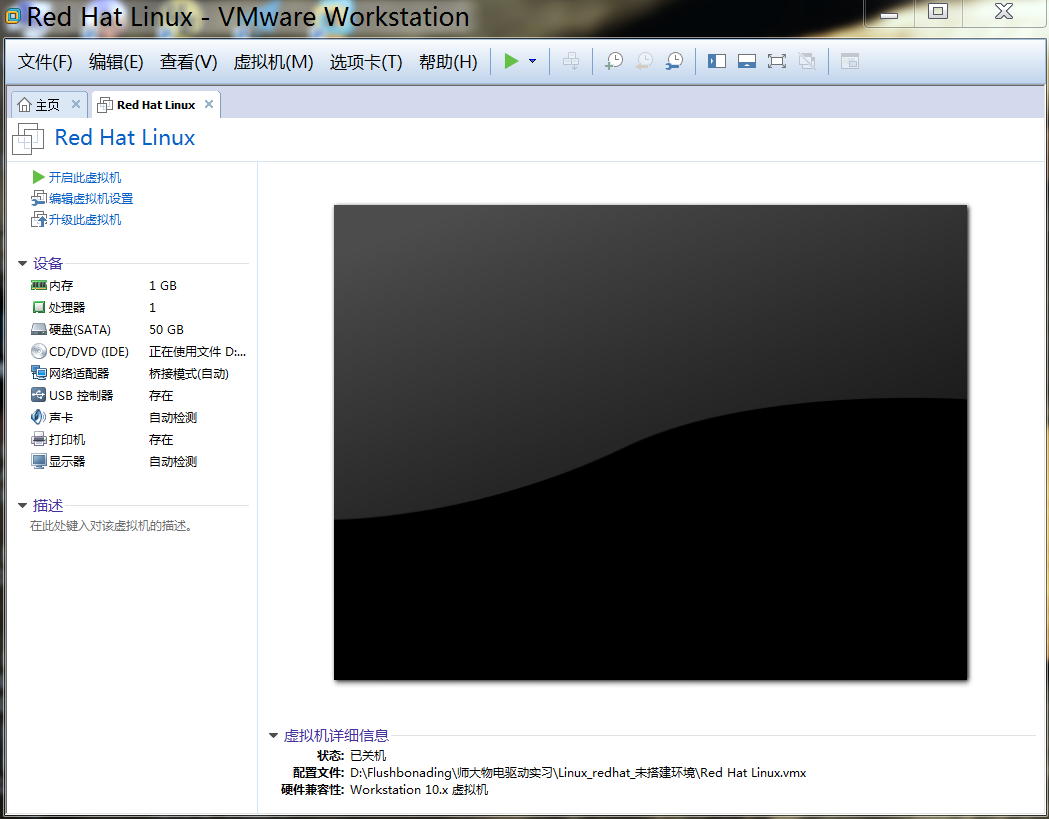


图 3

配置共享文件夹，以便于Red Hat Enterprise Linux 5虚拟机和宿主机交换文件。本次设置的共享文件夹目录为D:\Flushbonading\师大物电驱动实习\Linux\_redhat\_未搭建环境\share。

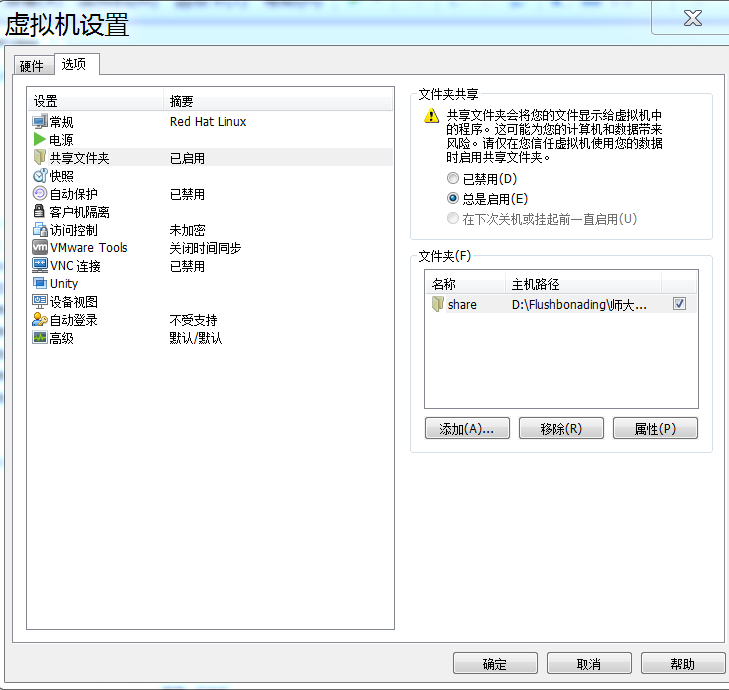


图 4

打开虚拟机，并以root账户登录，密码为123456

### 2)、安装交叉编译工具arm-linux-gcc

将arm-linux-gcc-4.5.1-v6-vfp-20120301.tgz和linux-3.5-20170221.tgz拷贝到共享文件夹。

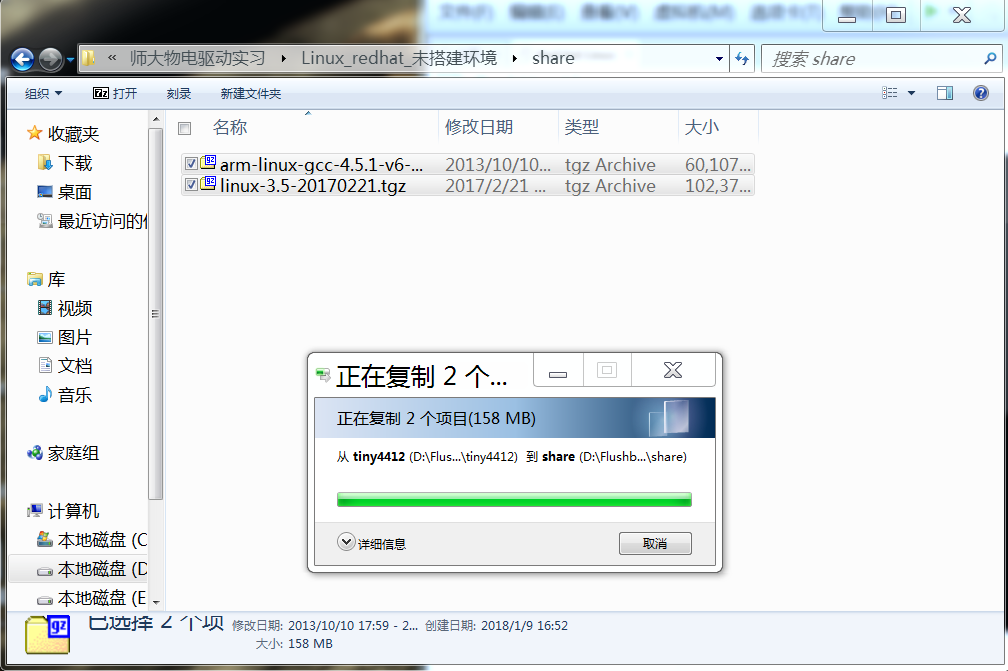


图 5

在虚拟机中打开终端，进入到共享文件夹/mnt/hgfs/share

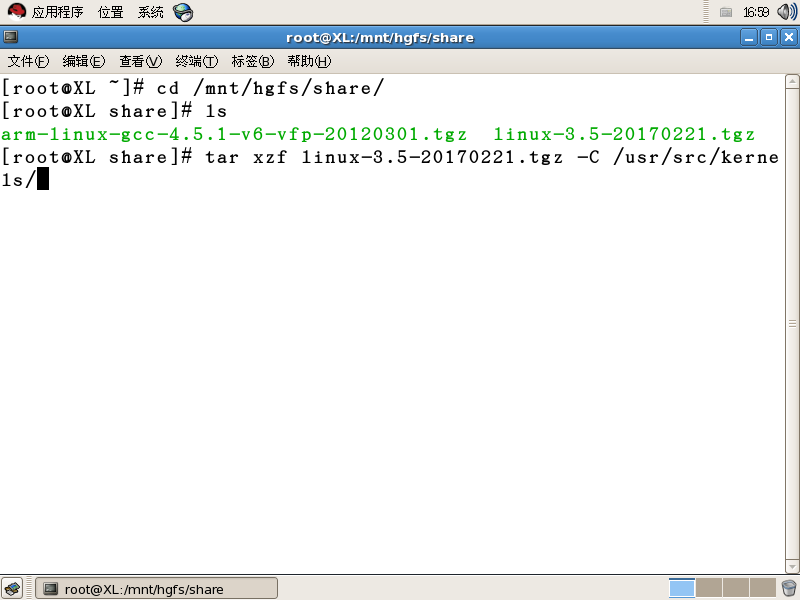


图 6

将arm-linux-gcc-4.5.1-v6-vfp-20120301.tgz tar使用tar命令解压到/opt目录下。将linux-3.5-20170221.tgz使用tar命令解压到/usr/src/kernels目录下。复制解压的交叉编译工具arm-linux-gcc的执行文件所在的目录绝对路径，并将其加入到环境变量中，且用source命令使环境变量生效。

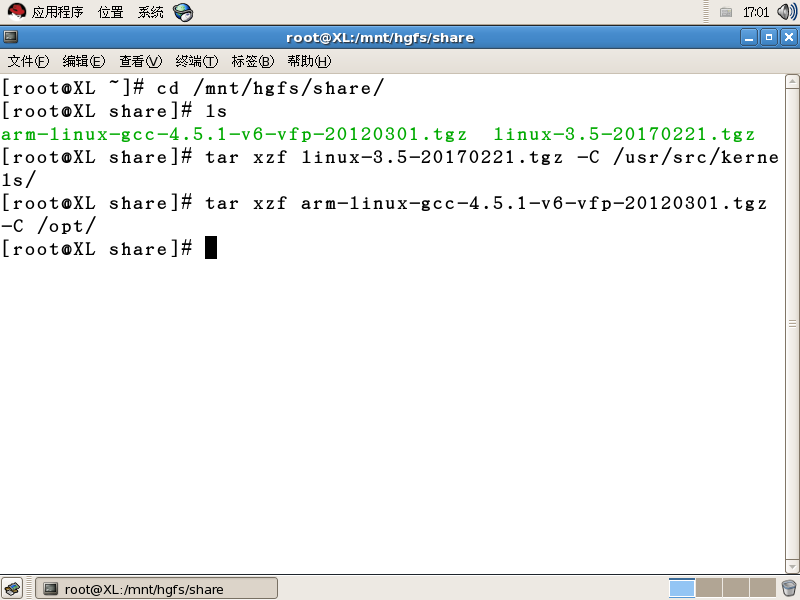


图 7

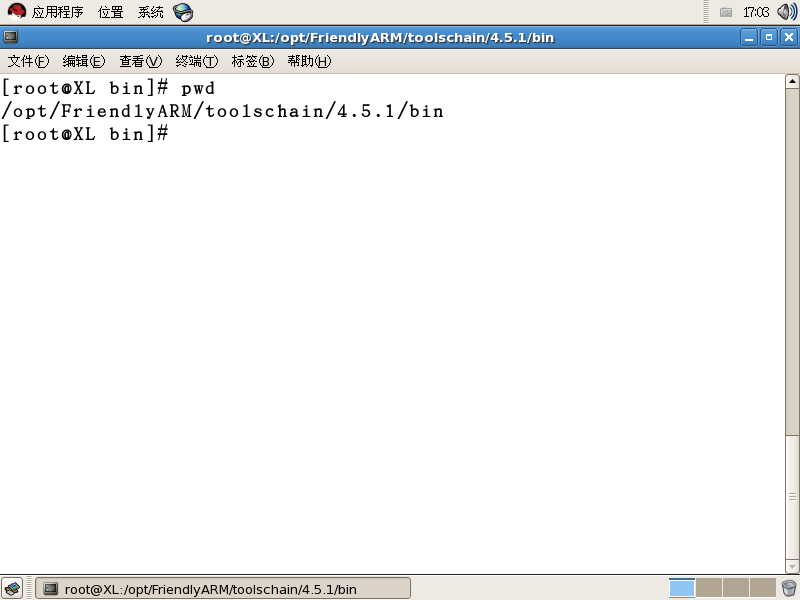


图 8

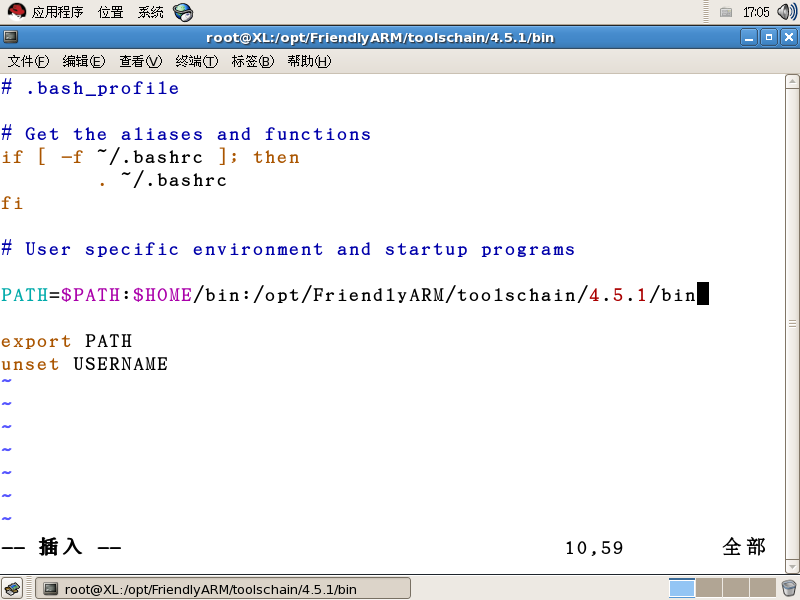


图 9

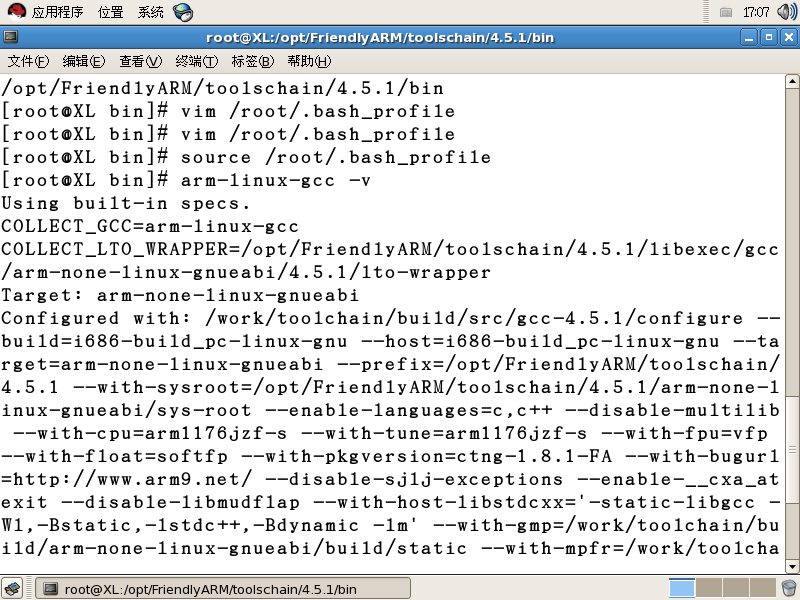


图 10

执行arm-linux-gcc –v出现如上所示的版本信息，则交叉编译工具已搭建好。

### 3)、构建开发板内核树

进入到前面解压的开发板内核源码文件夹(/usr/src/kernels/linux-3.5)。

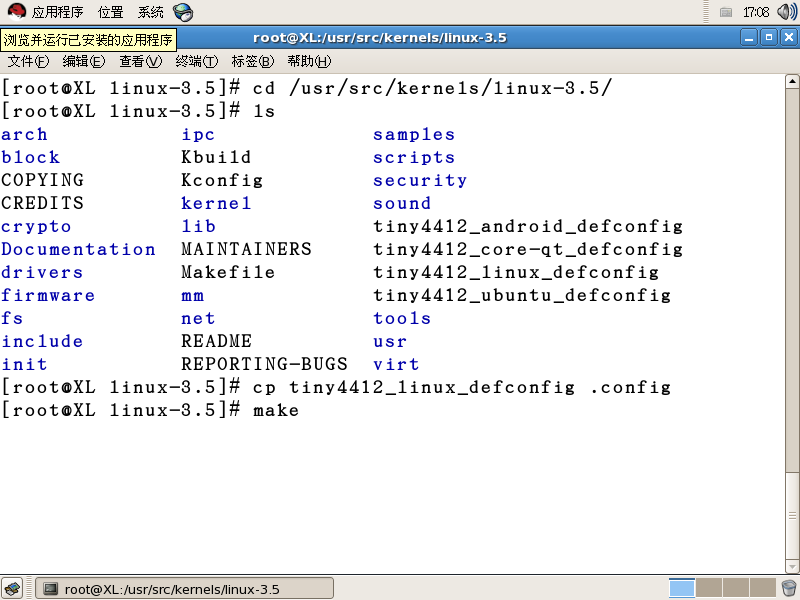


图 11

由于Linux源码支持非常多的硬件架构，编译内核的架构是仅仅由配置参数决定，所以执行cp tiny4412\_linux\_defconfig .config ，将友善之臂配置的Linux配置文件作为当前构建内核树的配置文件，然后执行make命令，等待期编译完。至此，交叉编译环境已经搭建完毕了。

## 2、C语言知识复习

### 1)、变量存储位置

栈区stack：由编译器自动分配释放，存放函数的参数值，函数的返回地址，局部变量等，#栈区容量有限，不能溢出。

堆区heap：一般由程序员分配释放。

全局区（静态区，数据区，静态数据区，数据段）static：全局变量和静态变量的存储是放在一块的，初始化的全局变量和静态变量在一块区域，未初始化的全局变量和静态变量存放在另一块相邻的区域，程序结束后由系统释放。

文字常量区：常量字符串，程序结束后由系统释放。

程序代码区：存放函数体的二进制代码。

### 2)、函数返回指针的几种方法

a、在函数内用malloc，由调用者释放

b、在函数内使用static 定义

c、由调用者分配空间，只是把指针传给函数，函数内部把数据拷贝到内存中。（推荐）

### 3)、其他杂项

函数指针

返回值类型 (\*指针变量名)([形参列表])。

## 3、Linux基本操作命令

### 1)、查看文件的几种方法

cat: 打印文件全部内容

more ： 分页显示（回车键），只能向下

less： 使用方向键向上或向下

head：head -10 xx 文件首10行

tail：tail -5 xx 文件尾部5行

### 2)、关于Linux用户管理的命令

su： 切换用户，默认的参数为root

用户组的信息构成：通过查看/etc/passwd文件可以看到如下信息，第一个500表示UID，即用户ID，第二个500表示GID，即组ID。

double:x:500:500::/home/double:/bin/bash

adduser： -g 加入组，默认建立主目录/home/xx

userdel： userdel xx [-rf] 删除用户xx[和其主目录]

usermod： usermod xx -g root 更改用户所属组

### 3)、gcc 编译过程

a、预处理 -E xx.i

b、编译 语法检查 -S xx.S

c、汇编 -c xx.o

d、链接 -l(library) xx

# 四、字符设备驱动基本结构

## 1、内核核心功能

进程管理

内存管理

文件管理

网路管理

设备管理

## 2、驱动类型

字符设备

像字节流一样的存取设备

通过/dev下文件系统节点访问

至少要实现open,close,read,write等系统调用

只能顺序访问的数据通道，不能前后移动指针

块设备

通过/dev下文件系统节点访问

块设备有专门的接口，必须支持挂载mount文件系统

块设备和字符设备的区别仅仅在于内核内部管理数据的方式

网络设备

通过单独的网络接口调用

## 3、驱动程序加载到内核的方法

直接编译进内核

linux提供了 模块Module 的机制

insmod 加载模块

rmmod 卸载模块

## 4、开发步骤

基本步骤

1、确定主设备号和次设备号

2、实现字符驱动程序

实现file\_operations结构题

实现初始化函数，注册字符设备

实现销毁函数，释放字符设备

3、创建设备文件节点

主设备号是内核识别一个设备的标识

整数（占12bits），范围从0到4095，通常使用1-255

次设备号由内核使用，用于正确确定设备文件所指向的设备

整数（占20bits），范围从0到1048575，通常使用1-255

设备编号的内部表达

dev\_t类型(32位)

用来保存设备编号 高12位为主设备号，低20位为次设备号

从dev\_t获得主次设备号

MAJOR(dev\_t) 主

MINOR(dev\_t) 次

将主次设备号转换为dev\_t类型

MKDEV(int major,int minor);

分配主设备号

手动分配

#include <linux/fs.h>

int register\_chrdev\_region(dev\_t first,unsigned int count,char \*name)

动态分配

#include <linux/fs.h>

int alloc\_chrdev\_region(dev\_t \*dev,unsigned int firstminor,unsigned int count,char \*name);

释放设备号

void unregister\_chrdev\_region(dev\_t first,unsigned int count);

实现字符驱动

cdev结构体

struct cdev

{

struct kobject kobj; //内嵌的kobject对象

struct module \*owner; //所属模块

struct file\_operations \*ops; //文件操作结构体

struct list\_head list;

dev\_t dev; //设备号

unsigned int count;

}

操作cdev的函数

void cdev\_init( struct cdev \*, struc t file\_operations \*);

struct cdev \*cdev\_alloc(void);

int cdev\_add(struct cdev \*, dev\_t, unsigned);

void cdev\_del(struct cdev \*);

file\_operations

根据硬件设备的特点，实现某些函数

1、应用程序调用fopen打开设备时，内核会将fopen映射到file\_operations里面的open函数指针处

2、主要成员

owner

unlocked\_ioctl //传递控制参数

open

release

read

write 被应用程序调用写

3、用户空间和内核空间的数据拷贝过程

unsigned long copy\_from\_user(void \*to, const void \_\_user \*from,unsigned long count) ;用户到内核，在write内

unsigned long copy\_to\_user(void \_\_user \*to, const void \*from,unsigned long count );内核到用户，在read内

如果要复制的内存是简单类型，如char、int、long 等，

put\_user()和get\_user()

# 五、蜂鸣器驱动模块

# 六、按键驱动模块

# 七、蜂鸣器模块与按键模块组合

# 八、总体设计

（1）设计思路

（2）设计步骤

# 九、小组总结

本次实习我们小组在老师的带领下学习了Linux字符驱动程序的基本框架。主要完成了对开发环境的搭建、字符设备驱动、蜂鸣器驱动模块、按键驱动模块以及蜂鸣器模块与按键模块的组合的学习，让我们认识了Linux字符驱动程序的设计，感受到了Linux的强大以及发展前景。

本次实习由于时间有限，只是对嵌入式有了一个大概的了解。还有许多问题还需我们进行进一步的了解和研究。特别是在实习中遇到过很多的问题，有些问题通过查资料或者在同学和老师的帮助下得以解决，也有些问题还是不太明白。也就是说，掌握的只是过于局限和片面，在遇到问题时不能快速准确的找到问题的根源。因此在今后的学习当中我们小组会努力学习，在Linux的世界当中遨游，争取做一名优秀的驱动工程师。

# 十、附录

程序代码