

中国地质大学（武汉）自动化学院

嵌入式实验报告

课 程： 嵌入式系统实验报告二

学 号： 20201000128

班 级： 231202

姓 名： 刘瑾瑾

指导老师： 刘玮

二〇二二年十月

一、结合实验五，说说你所理解的 Linux 操作系统；

Linux，全称 GNU/Linux，是一种免费使用和自由传播的类 UNIX 操作系统，是一个基于 POSIX 的多用户、多任务、支持多线程和多 CPU 的操作系统。它能运行主要的 Unix 工具软件、应用程序和网络协议。它支持 32 位和 64 位硬件。

(1) Linux 的基本思想有两点：第一，一切都是文件；第二，每个文件都有确定的用途。其中第一条详细来讲就是系统中的所有都归结为一个文件，包括命令、硬件和软件设备、操作系统、进程等等对于操作系统内核而言，都被视为拥有各自特性或类型的文件。

(2) 完全免费：Linux 是一款免费的操作系统，用户可以通过网络或其他途径免费获得，并可以任意修改其源代码。

(3) 多用户、多任务：Linux 支持多用户，各个用户对于自己的文件设备有自己特殊的权利，保证了各用户之间互不影响。多任务则是现代电脑最主要的一个特点，Linux 可以使多个程序同时并独立地运行。

(4) 良好的界面：Linux 同时具有字符界面和图形界面。在字符界面用户可以通过键盘输入相应的指令来进行操作。它同时也提供了类似 Windows 图形界面的 X-Window 系统，用户可以使用鼠标对其进行操作。

(5) 支持多种平台：Linux 可以运行在多种硬件平台上，如具有 x86、680x0、SPARC、Alpha 等处理器的平台。同时 Linux 也支持多处理器技术。多个处理器同时工作，使系统性能大大提高。

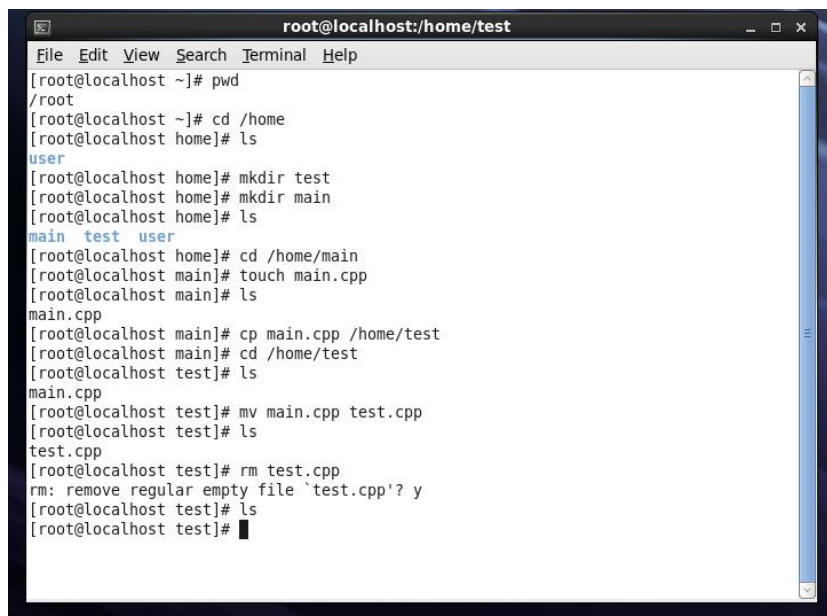
二、列出你在实验六中执行的一些常规命令行，给出该行命令的功能解释，并附上命令行执行截图；

常规命令行及功能：

<code>pwd</code>	显示工作目录
<code>cd</code>	切换工作目录
<code>ls</code>	列出目录内容
<code>touch</code>	创建文件
<code>mkdir</code>	创建目录
<code>cp</code>	复制文件或目录

- mv 移动或更名现有的文件或目录
- rm 删除文件或目录

命令执行截图：



```
root@localhost:/home/test
File Edit View Search Terminal Help
[root@localhost ~]# pwd
/root
[root@localhost ~]# cd /home
[root@localhost home]# ls
user
[root@localhost home]# mkdir test
[root@localhost home]# mkdir main
[root@localhost home]# ls
main test user
[root@localhost home]# cd /home/main
[root@localhost main]# touch main.cpp
[root@localhost main]# ls
main.cpp
[root@localhost main]# cp main.cpp /home/test
[root@localhost main]# cd /home/test
[root@localhost test]# ls
main.cpp
[root@localhost test]# mv main.cpp test.cpp
[root@localhost test]# ls
test.cpp
[root@localhost test]# rm test.cpp
rm: remove regular empty file `test.cpp'? y
[root@localhost test]# ls
[root@localhost test]#
```

图 1 命令执行截图

三、列出实验六、实验八的实验过程，编写并编译自己的第一个 c 程序（硬件无关），并附实验截图;

实验六：

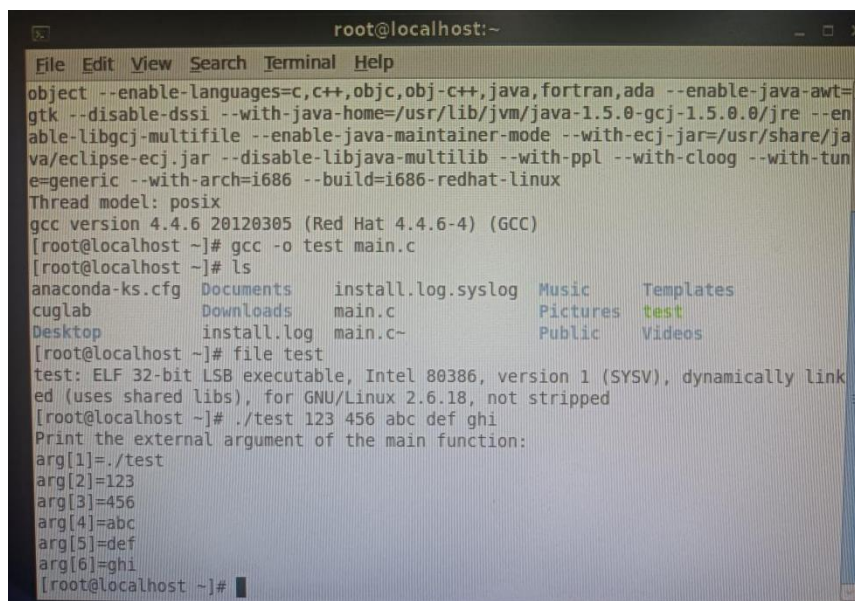
- (1) pwd 显示工作目录
- (2) cd 切换工作目录为 home
- (3) ls 列出工作目录下的文件
- (4) mkdir 创建 main 和 test 两个目录
- (5) touch 在 main 目录下创建 main.cpp 文件
- (6) cp 将 main 目录下的 main.cpp 文件复制到 test 目录下
- (7) mv 将 test 目录下的 main.cpp 更名为 test.cpp
- (8) rm 在 test 目录下删除 test.cpp 文件

实验八：

- (1) 确认 main.c 在当前工作目录下，执行以下命令：
#gcc -o test main.c
- (2) 执行 ls，查看当前目录，发现新增一个文件 test。

(3) 执行命令 `#file test` 可以查看 `test` 文件详细信息。

(4) 执行可执行文件 `test: #./test 123 456 abc def ghi`



```
root@localhost:~  
File Edit View Search Terminal Help  
object --enable-languages=c,c++,objc,obj-c++,java,fortran,ada --enable-java-awt=  
gtk --disable-dssi --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-1.5.0.0/jre --en  
able-libgcj-multifile --enable-java-maintainer-mode --with-ecj-jar=/usr/share/ja  
va/eclipse-ecj.jar --disable-libjava-multilib --with-ppl --with-cloog --with-tun  
e=generic --with-arch=i686 --build=i686-redhat-linux  
Thread model: posix  
gcc version 4.4.6 20120305 (Red Hat 4.4.6-4) (GCC)  
[root@localhost ~]# gcc -o test main.c  
[root@localhost ~]# ls  
anaconda-ks.cfg  Documents      install.log.syslog  Music      Templates  
cuglab           Downloads     main.c             Pictures   test  
Desktop          install.log  main.c~           Public     Videos  
[root@localhost ~]# file test  
test: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically link  
ed (uses shared libs), for GNU/Linux 2.6.18, not stripped  
[root@localhost ~]# ./test 123 456 abc def ghi  
Print the external argument of the main function:  
arg[1]=./test  
arg[2]=123  
arg[3]=456  
arg[4]=abc  
arg[5]=def  
arg[6]=ghi  
[root@localhost ~]#
```

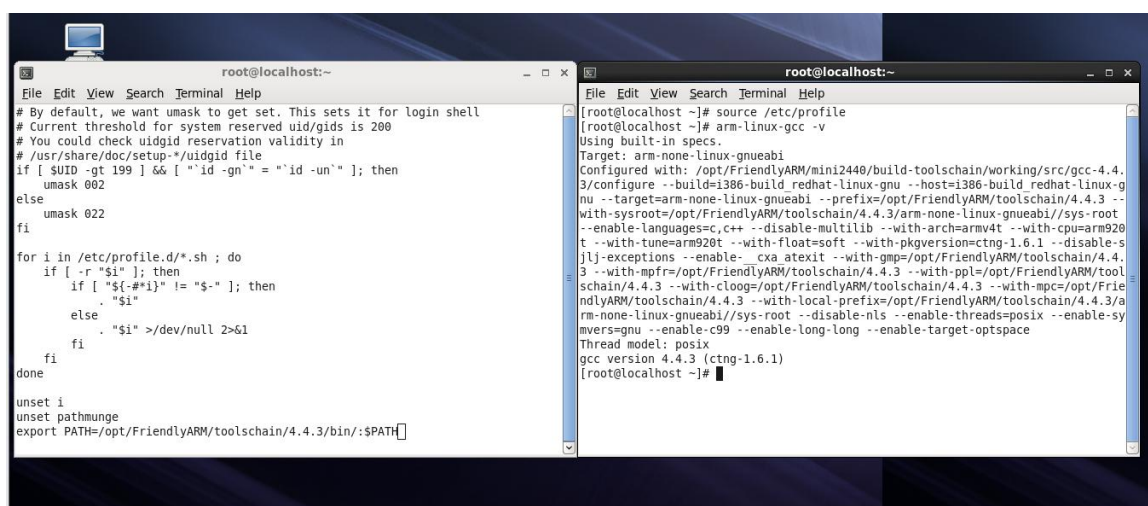
图 2 实验八实验截图

四、列出实验九、十的实验过程，并附实验截图;

实验九:

(1) 安装交叉编译工具

(2) 修改系统环境变量



```
root@localhost:~  
File Edit View Search Terminal Help  
# By default, we want umask to get set. This sets it for login shell  
# Current threshold for system reserved uid/gids is 200  
# You could check uidgid reservation validity in  
# /usr/share/doc/setup-*/uidgid file  
if [ $UID -gt 199 ] && [ "$(id -gn)" = "id -un" ]; then  
    umask 002  
else  
    umask 022  
fi  
for i in /etc/profile.d/*.sh; do  
    if [ -r "$i" ]; then  
        if [ "${#i}" != "$-" ]; then  
            . "$i"  
        else  
            . "$i" >/dev/null 2>&1  
        fi  
    fi  
done  
unset i  
unset pathmunge  
export PATH=/opt/FriendlyARM/toolschain/4.4.3/bin/:$PATH
```

```
root@localhost:~  
File Edit View Search Terminal Help  
[root@localhost ~]# source /etc/profile  
[root@localhost ~]# arm-linux-gcc -v  
Using built-in specs.  
Target: arm-none-linux-gnueabi  
Configured with: /opt/FriendlyARM/mini2440/build-toolschain/working/src/gcc-4.4.  
3/configure --build=i386-build redhat-linux-gnu --host=i386-build redhat-linux-g  
nu --target=arm-none-linux-gnueabi --prefix=/opt/FriendlyARM/toolschain/4.4.3 --  
with-sysroot=/opt/FriendlyARM/toolschain/4.4.3/arm-none-linux-gnueabi/sys-root  
--enable-languages=c,c++ --disable-multilib --with-arch=armv4t --with-cpu=arm920  
t --with-tune=arm920t --with-float=soft --with-pkgversion=ctng-1.6.1 --disable-s  
jlj-exceptions --enable- cxa atexit --with-gmp=/opt/FriendlyARM/toolschain/4.4.  
3 --with-mpfr=/opt/FriendlyARM/toolschain/4.4.3 --with-ppl=/opt/FriendlyARM/tool  
schain/4.4.3 --with-cloog=/opt/FriendlyARM/toolschain/4.4.3 --with-mpc=/opt/Frie  
ndlyARM/toolschain/4.4.3 --with-local-prefix=/opt/FriendlyARM/toolschain/4.4.3/a  
rm-none-linux-gnueabi/sys-root --disable-nls --enable-threads=posix --enable-sy  
mvers-gnu --enable-c99 --enable-long-long --enable-target-optspace  
Thread model: posix  
gcc version 4.4.3 (ctng-1.6.1)  
[root@localhost ~]#
```

图 3 实验九截图

实验十:

(1) 在 GUN 下编写流水灯程序

(2) 编写流水灯程序的 Makefile 文件

- (3) 使用工程管理工作 **make** 生成可执行程序文件
- (4) 下载可执行程序到开发板，观察流水灯的变化。

```
[root@localhost home]# cd /home/linux_asm  
[root@localhost linux_asm]# make  
arm-linux-gcc -c -o asm.o asm.s  
arm-linux-ld -Ttext 0x30000000 -o asm.elf asm.o  
arm-linux-objcopy -O binary -S asm.elf asm.bin  
[root@localhost linux_asm]#
```



图 4 实验十截图

五、思考题

(1) 思考 **windows** 环境下与 **Linux** 环境下开发裸机程序的区别有什么？

答：Windows 环境下一般使用集成开发环境，集成了编辑器、编译器和链接器，如 keil,ADS 等，一般分为程序编辑，编译和烧录三个步骤，编译的步骤由开发软件决定，开发较简单；Linux 没有集成开发环境，但可使用 GNU 的汇编器 **as**、交叉编译工具 **gcc** 和链接器 **ld**，可以根据使用者编写的 **makefile** 文件来确定编译的步骤，使用 **make** 命令按照 **makefile** 文件编译文件。

(2) **make** 及 **Makefile** 的作用是什么？

答：①**Makefile**：一个决定怎样编译工程的文本文件，按照一定的规则书写；

②**make**：**make** 是一个命令工具，解释 **Makefile** 中的指令，根据当前的 **makefile** 进行工程编译。

(3) 第二篇实验与第三篇实验的 **Linux** 系统一样吗？如不同，不同之处有什么？

答：不一样；

①第二篇实验的 **Linux** 系统是 **Linux** 操作系统，是应用于计算机的操作系统；

②第三篇实验的 **Linux** 系统是嵌入式操作系统，是经过小型化裁剪后，能够固化

在容量只有几百 K 字节或几兆字节的存储器芯片中,应用于特定嵌入式场合的专用的操作系统。

(4) 关于 Linux 操作系统,你还想要了解什么?

答: 关于 Linux 操作系统,我还想了解如何在 Linux 系统中下载安装和卸载应用软件,如何使用 C/C++等编程语言进行开发等, Linux 和 Windows 操作系统相比它的优势在哪里,怎样熟练掌握并使用 Linux 系统。

(5) 关于嵌入式 Linux 操作系统的开发,你还希望学习什么?

答: 希望学习怎样编写 bootload 程序,进行 Linux 内核裁剪时主要修改的代码部分以及为什么修改,怎样编写硬件驱动程序,想体验一下 Linux 操作系统相比于 Windows 操作系统的在软件开发方面的优势。

六、体会和建议

(1) 体会: 通过这几个实验,我对 Linux 系统有了进一步的了解,简单的了解到了 Linux 系统和 Windows 系统的区别;通过自己的实际操作了解了 Linux 操作系统下命令行的使用,学习并掌握了基本的指令,也了解了两个文本编辑器 gedit 和 vi 的使用与区别;通过自己在 Linux 下的编程以及交叉编译工具的使用,对 Linux 系统进行嵌入式系统开发有了基本的了解,可以编写简单的 makefile 文件,使用 make 命令进行编译;初步体验了 Linux 操作系统的移植与驱动,了解了 Linux 操作系统移植的步骤。

(2) 建议: 通过实验所了解 Linux 的比较浅,只是按照实验步骤简单的操作了一下,希望可以多添加一些内容,让我们进一步了解 Linux 操作系统以及 Linux 系统的嵌入式开发。