



中國地質大學

嵌入式课程实验报告

指导老师 : 刘玮

姓 名 : 陈涵毅

班 级 : 231203

学 号 : 20201002157

二〇二二年十月

实验报告 1

1.1 实验二 ARM 汇编程序设计

实验代码

;定义端口 B 寄存器预定义

GPBCON EQU 0x56000010

GPBDAT EQU 0x56000014

GPBUP EQU 0x56000018

AREA Init, CODE, READONLY;该伪指令定义了一个代码段, 段名为 Init,
属性只读

ENTRY;程序的入口点标识

ResetEntry

;下面这三条语句, 主要是用来设置 GPB5--GPB8 为输出属性

ldr r0,= GPBCON ;将寄存器 GPBCON 的地址存放到寄存器 r0 中

ldr r1,=0x15400

str r1,[r0] ;将 r1 中的数据存放到地址为 r0 的内存单元中

;下面这三条语句, 设置 GPB5--GPB8 禁止上拉电阻

ldr r0,= GPBUP

ldr r1,=0xffff

str r1,[r0]

ldr r2,=GPBDAT ; 将寄存器 GPBDAT 的地址存放到寄存器 r2 中

aa

ldr r1,=0x01e0

ledloop

str r1,[r2] ;使 GPB--GPB8 输出高电平, LED1--LED4 全灭

bl delay ;调用延迟子程序

sub r1,r1,#0x20 ;依此改变 r1 字节

str r1,[r2] ; 使 GPB5 输出低电平,GPB6--GPB8 输出高电平,LED1 亮 1

bl delay ;调用延迟子程序

cmp r1,#0 ;将 r1 与 0 比较

bne ledloop ;不断的循环, LED1-LED7 将不停的闪烁

b aa ;跳转到 aa 处, 重新置位

;下面是延迟子程序

delay

ldr r3,#0xbffff ;设置延迟的时间

delay1

sub r3,r3,#1 ;r3=r3-1

cmp r3,#0x0 ;将 r3 的值与 0 相比较

bne delay1 ;比较的结果不为 0 (r3 不为 0), 继续调用 delay1,否则执行下一条语句

mov pc,lr ;返回

END ;程序结束符

AXD 中的调试截图:

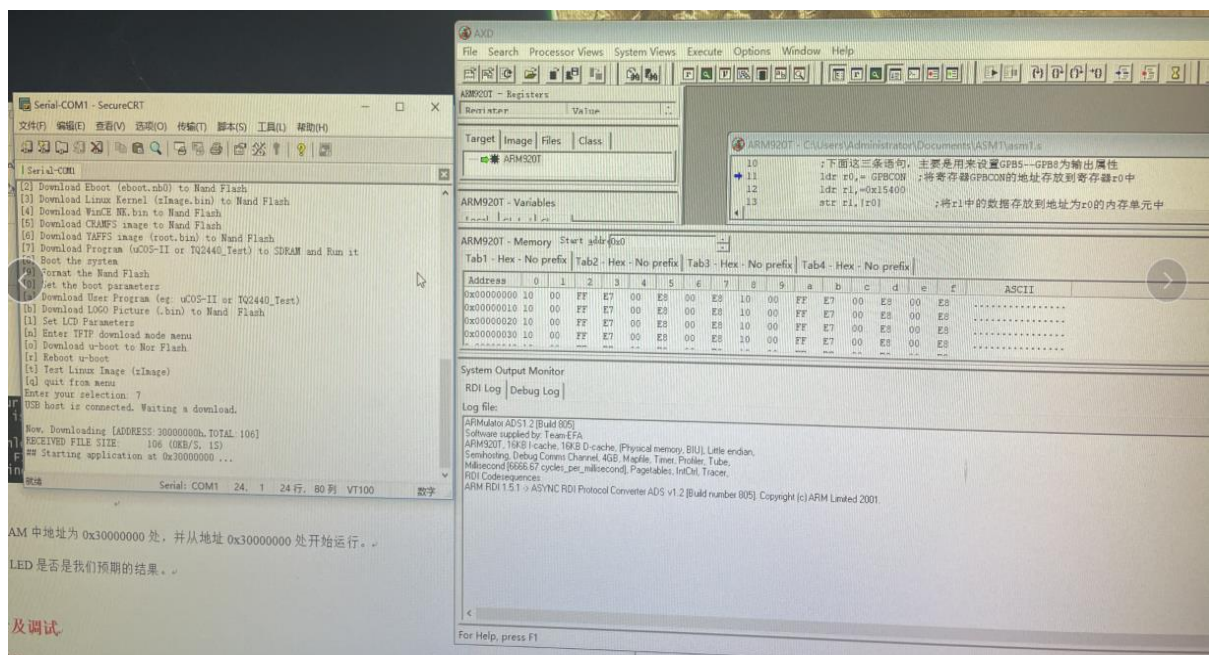


图 1 AXD 调试图

1.2 实验三 ARM 汇编与 C 混合编程软件实现流程图：

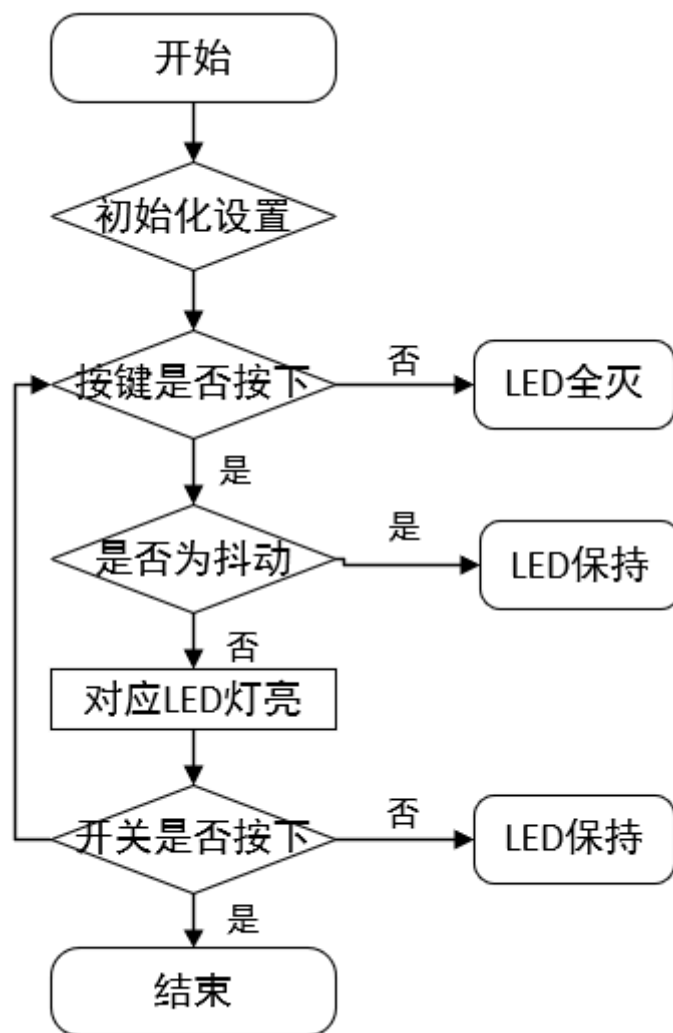


图 2 实验三 ARM 汇编与 C 混合编程软件实现流程图

1.3 实验四 ARM 中断实验软件实现流程图：

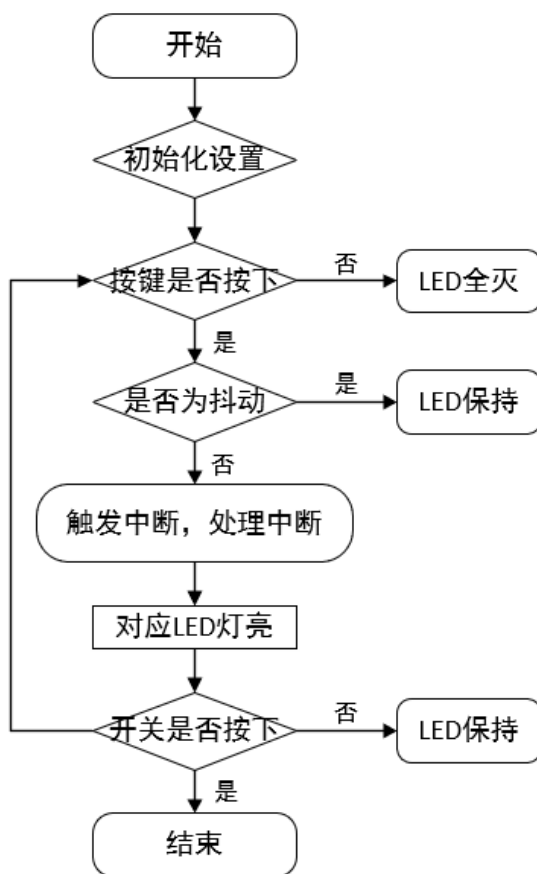


图 3 实验四 ARM 中断实验软件实现流程图

1.4 思考题

1、在嵌入式系统编程当中，汇编语言和 C 语言分别有什么优势？是否可以完全摒弃其中一种语言？为什么？

汇编语言是一种用文字助记符来表示机器指令的符号语言，是最接近机器码的一种语言。其主要优点是占用资源少、程序执行效率高。使用汇编语言能面向机器并较好地发挥机器的特性，得到质量较高的程序。

C 语言是一种编译型程序设计语言，它兼顾了多种高级语言的特点，并具备汇编语言的功能。C 语言有功能丰富的库函数、运算速度快、编译效率高、有良好的可移植性，而且可以直接实现对系统硬件的控制。C 语言是一种结构化程序设计语言，它支持当前程序设计中广泛采用的由顶向下结构化程序设计技术。此外，C 语言程序具有完善的模块程序结构，从而为软件开发中采用模块化程序设计方法提供了有力的保障。因此，使用 C 语言进行程序设计已成为软件开发的一个主流。用 C 语言来编写目标系统软件，会大大缩短开发周期，且明显地增加软件的可读性，便于改进和扩充，从而研制出规模更大、性能更完备的系统。

总体来说用 C 语言是单片机开发与应用的趋势，但我们不能完全摒弃其中一种语言，在不同的编程环境下有不同的要求，不同的语言有不同的优势，比如采用混合编程的方法。

2、ARM 汇编调用 C 语言以及 C 语言调用 ARM 汇编时，如何传递参数？实验 2,3 程序中

参数是如何传递的？

ARM 汇编程序调用 C 程序：在调用之前首先要通过 IMPORT 伪指令进行声明，必须根据 C 语言模块中需要的参数个数，以及 ATPCS 参数规则，完成参数传递，即前四个参数通过 R0-R3 传递，后面的参数通过堆栈传递，然后再利用 B、BL 指令调用。

C 语言调用 ARM 汇编：C 程序调用汇编程序首先通过 extern 声明要调用的汇编程序模块，声明中形参个数要与汇编程序模块中需要的变量个数一致，且参数传递要满足 ATPCS 规则，然后在 C 程序中调用。

实验二为汇编语言中调用 C 语言，本次实验例程中数据传递是通过寄存器 R3 传递的数据。实验三为汇编调用 C 语言，通过 bl 指令调用 Main 函数。

3、C 语言中和汇编语言中是如何操作寄存器的？

C 语言通过位操作的方式给寄存器赋值，汇编是通过寄存器寻址等寻址方式来操作寄存器。

4、比较实验二、三和四中 ADS 下的工程设置的有何异同，并分析其理由。

实验二和三 ADS 工程中 Linker 里面的 output RO base 地址设为 0x30000000，这是 S3C2440 的 SDRAM 的首地址；在实验四 ADS 工程中 output RO base 地址改为 0x00000000，这是 S3C2440 的 Nand Flash 的首地址。实验二和三程序是在 Nor Flash 模式下运行，实验四是在 Nand Flash 模式下。

5、在中断实验中为什么要把可执行程序下载到 NANDFlash 中运行，而不是直接下载到 SDRAM 中运行？如果直接下载到 SDRAM 中运行会发生什么情况？

中断向量表位于地址 0x30000000 以下，如 IRQ 中断向量地址为 0x00000018、FIQ 中断向量地址为 0x0000001C；而 SDRAM 一般是映射到地址 0x30000000 以后，若下载到 SDRAM 中，中断向量地址将找不到中断函数，则程序无法正常运行。

6、结合实验，叙述 NAND Flash 启动的流程。

- (1)配置 DMA 控制器的 4 个寄存器，通道使能后，等待 FLASH 发出的搬运请求；
- (2)配置 NAND FLASH 控制器的 3 个寄存器，选择适合的地址、时序参数与所用的 FLASH 芯片吻合；
- (3)分别在 r8 ~ r11 中放入程序需要的备用值；
- (4)将需要在 SDRAM 中运行的 4 条指令搬入 SDRAM 0x30000000 处；
- (5)执行 Nop 指令，Nop 指令用于填充一页 NANDFLASH 中的剩余空间；
- (6)执行在页末的指令，将 PC 指针指向 SDRAM 的 0x30000000 处；
- (7)执行 SDRAM 中的指令，首先启动 NANDFLASH 的数据传输，将程序搬往 SDRAM 的 0x30001000 处。其次执行一个循环语句，等待第一页的程序搬完，之后将 PC 指针指向 0x30001000 处，启动程序从 0x30001000 处正式开始执行。

1.5 体会与建议

在第一次和第二次实验中，我学会了在 window 环境下使用 ADS 软件对于开发板进行了裸机开发，对于 ADS 的基本使用以及编译、烧录有了一定的认识，能够在实验例程代码的基础上进行自主编程达到后续实验要求。认识到了按键控制和中断控制 LED 的亮灭的区别。

建议：实验过程中可加入自主编写 c 文件进行交叉编译的过程，对于开发板的中断原理和 Nand Flash 启动过程可以进行更多考察。

实验报告 2

2.1 结合实验五，说说你所理解的 Linux 操作系统

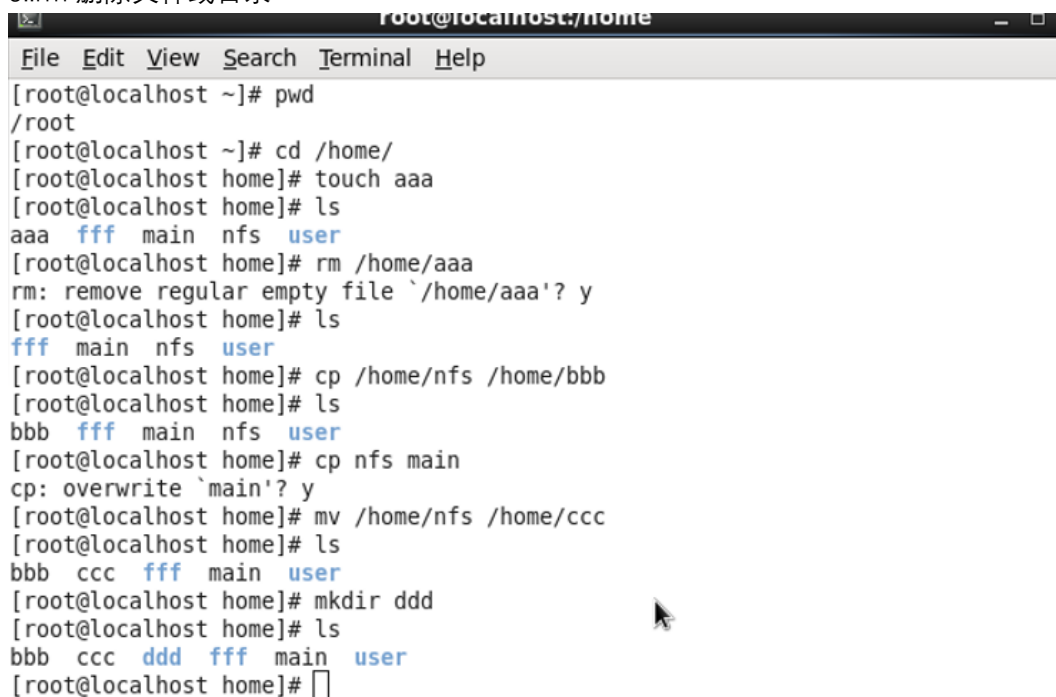
在查询相关资料后我对于 Linux 有以下认识：Linux 是最知名和最常用的开源操作系统。作为一个操作系统，Linux 是一个软件，位于计算机上的所有其他软件的下面，从这些程序接收请求并将这些请求转发到计算机硬件。

我们使用术语“Linux”来指代 Linux 内核，也是通常与 Linux 内核捆绑在一起的程序，工具和服务，以提供所有必需的组件全功能操作系统。将此集合称为 GNU / Linux，因为包括的许多工具都是 GNU 组件。但是并不是所有的 Linux 安装都使用 GNU 组件作为其操作系统的一部分。

通常我们说的“linux”其实是指 linux 内核，而 linux 操作系统其实是 GNU/linux (GNU/linux 是指 linux 内核+GNU 组织的软件)。

2.2 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

- 1.pwd 显示工作目录
- 2.cd 切换工作目录
- 3.ls 列出目录内容
- 4.touch 创建文件
- 5..rm 删除文件或目录



```
root@localhost:~/home
File Edit View Search Terminal Help
[root@localhost ~]# pwd
/root
[root@localhost ~]# cd /home/
[root@localhost home]# touch aaa
[root@localhost home]# ls
aaa fff main nfs user
[root@localhost home]# rm /home/aaa
rm: remove regular empty file `/home/aaa'? y
[root@localhost home]# ls
fff main nfs user
[root@localhost home]# cp /home/nfs /home/bbb
[root@localhost home]# ls
bbb fff main nfs user
[root@localhost home]# cp nfs main
cp: overwrite `main'? y
[root@localhost home]# mv /home/nfs /home/ccc
[root@localhost home]# ls
bbb ccc fff main user
[root@localhost home]# mkdir ddd
[root@localhost home]# ls
bbb ccc ddd fff main user
[root@localhost home]#
```

图 4 命令行执行截图

2.3 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

实验六过程

1. 登录 Linux 终端
2. 打开 terminal 命令终端
3. 在终端输入 Linux 常用命令，如 `pwd`, `cd`, `ls`, `rm` 等

实验八过程

1. 打开 terminal 命令终端
2. 在 shell 的提示符（即命令行终端）下键入 `gcc-v`，屏幕上就会显示出目前正在使用的 `gcc` 的版本信息
3. 对于实验七中编辑的 `main.c` 源文件编译为可执行文件并执行，依此输入
`#gcc -o test main.c`
`#file test`
`#./test 123 456 a`

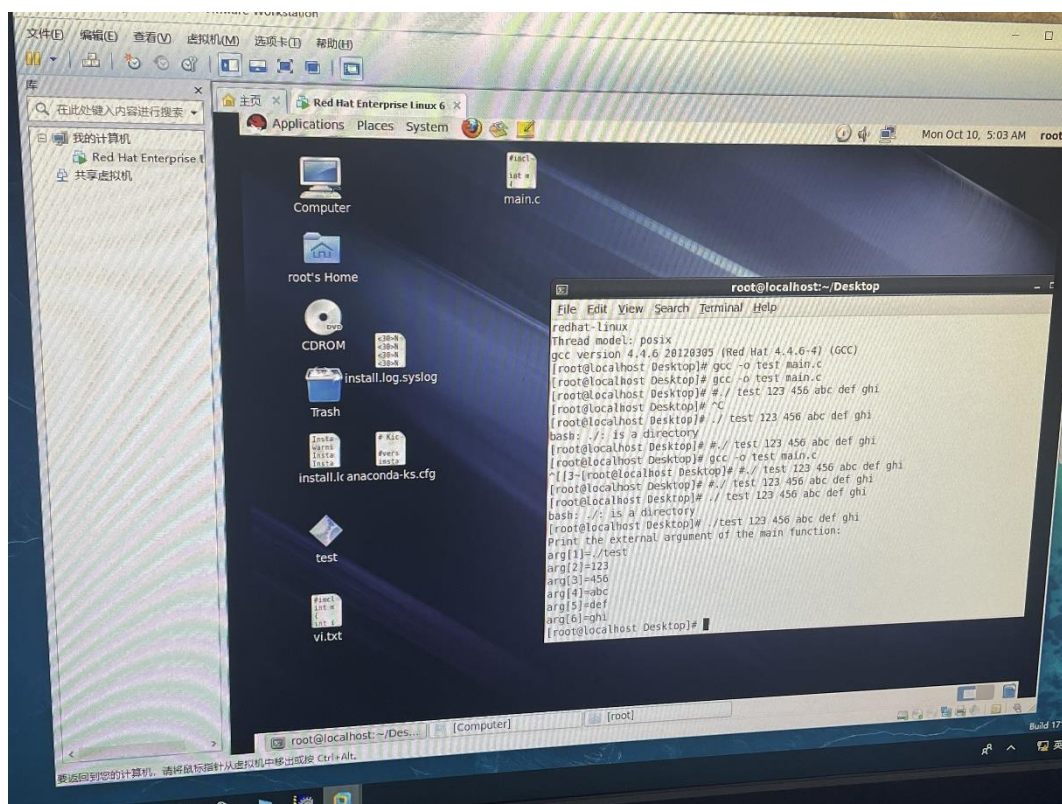


图 5 实验八运行截图

2.4 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# tar xvfz arm-linux-gcc-4.4.3-20100728.tar.gz -C

2. 修改系统环境变量

vi /etc/profile

export PATH=/opt/FriendlyARM/toolschain/4.4.3/bin/:\$PATH

source /etc/profile

查看修改的环境变量是否生效，可以执行命令： # echo \$PATH

修改的环境变量生效后，安装交叉编译工具的工作就完成了。

查看交叉编译工具的信息，可以执行命令： # arm-linux-gcc -v

实验十：

1.为 main.c 编写一个 Makefile

2.编写流水灯 Makefile 管理工程代码

3.执行 Makefile 文件编译代码

4.下载并运行执行程序

实验截图：

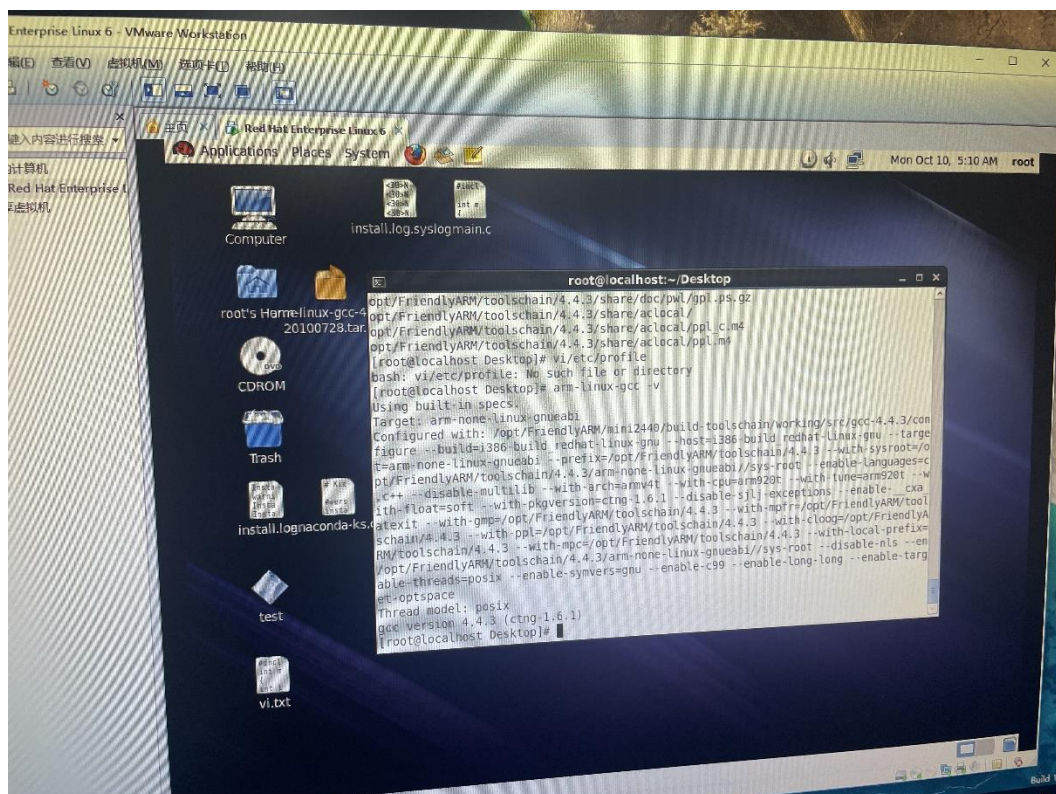


图 6 实验九运行结果

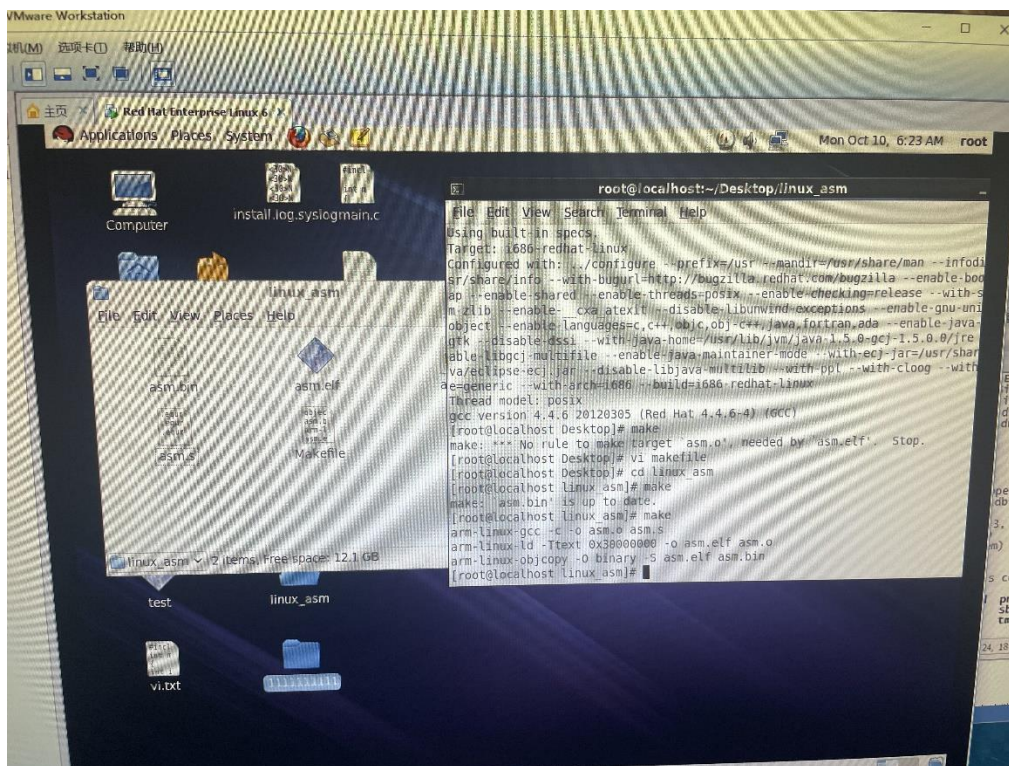


图 7 实验十运行结果

2.5 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？

在 windows 环境下，使用的是 ADS 集成开发环境，可以在 ADS 中实现新工程的创建，添加文件，编译等步骤,并用 AXD 进行仿真调试。程序在开发板上运行的首地址可以在 ADS 的设置中指定。在 Linux 环境下是需要 makefile 来完成，编辑要用 gedit 或 vi 编辑器，将 asm.s 连接到 asm. bin 需要用 arm linux gcc 交叉编译器，程序在开发板上运行的首地址在 Makefile 文件中通过指令指定等等。

2.make 及 makefile 的作用是什么？

make 是一个命令工具，是一个解释 makefile 中指令的命令工具，用来执行 Makefile 文件。

Makefile 是一个文件，文件中包含着一些目标，通常目标就是文件名，对每一个目标，提供了实现这个目标的一组命令以及和这个目标有依赖关系的其他目标或文件名。

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？

不一样，二者都是基于 Linux 系统来操作的，但是第二实验中的 Linux 系统是在 Windows 下运行虚拟机，虚拟机里运的是基于 Linux 内核的 Redhat 操作系统。

第三篇实验的 Linux 系统直接运行在开发板上，通过 USB 接口和开发板的通讯线将操作命令从 Windows 的超级终端中传入开发板进行操作。

4.关于 Linux 操作系统，你还想要了解什么？

Linux 多线程多用户的使用、Linux 应用程序、Linux 网阔管理

5.关于嵌入式 Linux 操作系统的开发，你还希望学习什么？

利用 Linux 系统进行一个实际项目的部署和测试，对于开发板进行更多功能的测试，对于 Linux 操作系统指令进行更多的使用。

2.6 体会和建议

在本次实验中，我学习了在 Linux 环境下进行开发板程序的设计及交叉编译流程和对于 Linux 操作系统的移植和驱动。课堂上对于 Linux 的讲解内容不多，对于 Linux 的学习我还处于入门阶段，希望在后续的实习过程中能够深入学习 Linux 知识，后续有能力开展嵌入式 Linux 系统开发。

最后感觉刘玮老师在课堂上和实验课耐心讲解和答疑，在最后一次实验中向我们耐心讲解了前几次实验代码的含义和我们在实验中经常犯错的部分，让我在嵌入式的学习中收获丰富。