**中 国 地 质 大 学**

**嵌入式实验报告**



课程名称 嵌入式原理及应用

任课教师

学生姓名

学生班级

学生学号

日 期: 2021 年 10 月 31 日

**目录**

[windows环境下裸机开发实验报告 3](#_Toc8914)

[一、实验二：ARM汇编程序的设计 3](#_Toc24236)

[1.程序代码及注释 3](#_Toc240)

[2.AXD调试成功截图 4](#_Toc30107)

[二、实验三：ARM汇编与C混合编程 5](#_Toc11873)

[三、实验四：ARM中断实验 6](#_Toc1069)

[四、思考题 7](#_Toc29986)

[五、 体会和建议 8](#_Toc13094)

[Linux环境下裸机开发实验报告 9](#_Toc20126)

[一、 实验五：熟悉Linux操作系统 9](#_Toc8990)

[二、 实验六：Linux常用命令的熟悉 9](#_Toc31384)

[1. 常规命令行 9](#_Toc22567)

[2、命令行执行截图 9](#_Toc10935)

[三、 实验六、实验八的实验过程 10](#_Toc27586)

[1.实验六实验过程 10](#_Toc20251)

[2. 实验八实验过程 11](#_Toc32658)

[3. 实验截图 11](#_Toc13532)

[四、 实验九、实验十的实验过程 12](#_Toc32347)

[1. 实验九过程及截图 12](#_Toc28021)

[2. 实验十过程及截图 12](#_Toc26199)

[五、 思考题 14](#_Toc28242)

[六、 体会和建议 14](#_Toc28397)

# windows环境下裸机开发实验报告

## 一、实验二：ARM汇编程序的设计

### 1.程序代码及注释

;汇编指令实验

;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

init

ldr r1,=0xffff

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

bl delay ;调用延迟子程序

ldr r4,=0 ;相当于计数器，计数16次

ldr r1,=0x1c0 ;点亮第一个灯

ledloop

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

bl delay ;调用延迟子程序

eor r1,r1,#0x1e0 ;由于是低电平有效，所以需要先取反再加1，实现二进制点灯

add r1,r1,#0x20 ;第五位+1

eor r1,r1,#0x1e0 ;取反

add r4,r4,#0x1 ;计数加1

cmp r4,#0xf ;r4的值与0xf比较，相当于计数16次

beq init ;如果相等，说明已进行一次二进制点灯，则跳转到初始化程序重新开始

b ledloop ;不断的循环，LED1-LED7将不停的闪烁

;下面是延迟子程序

delay

ldr r3,=0xeffff ;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

### 2.AXD调试成功截图

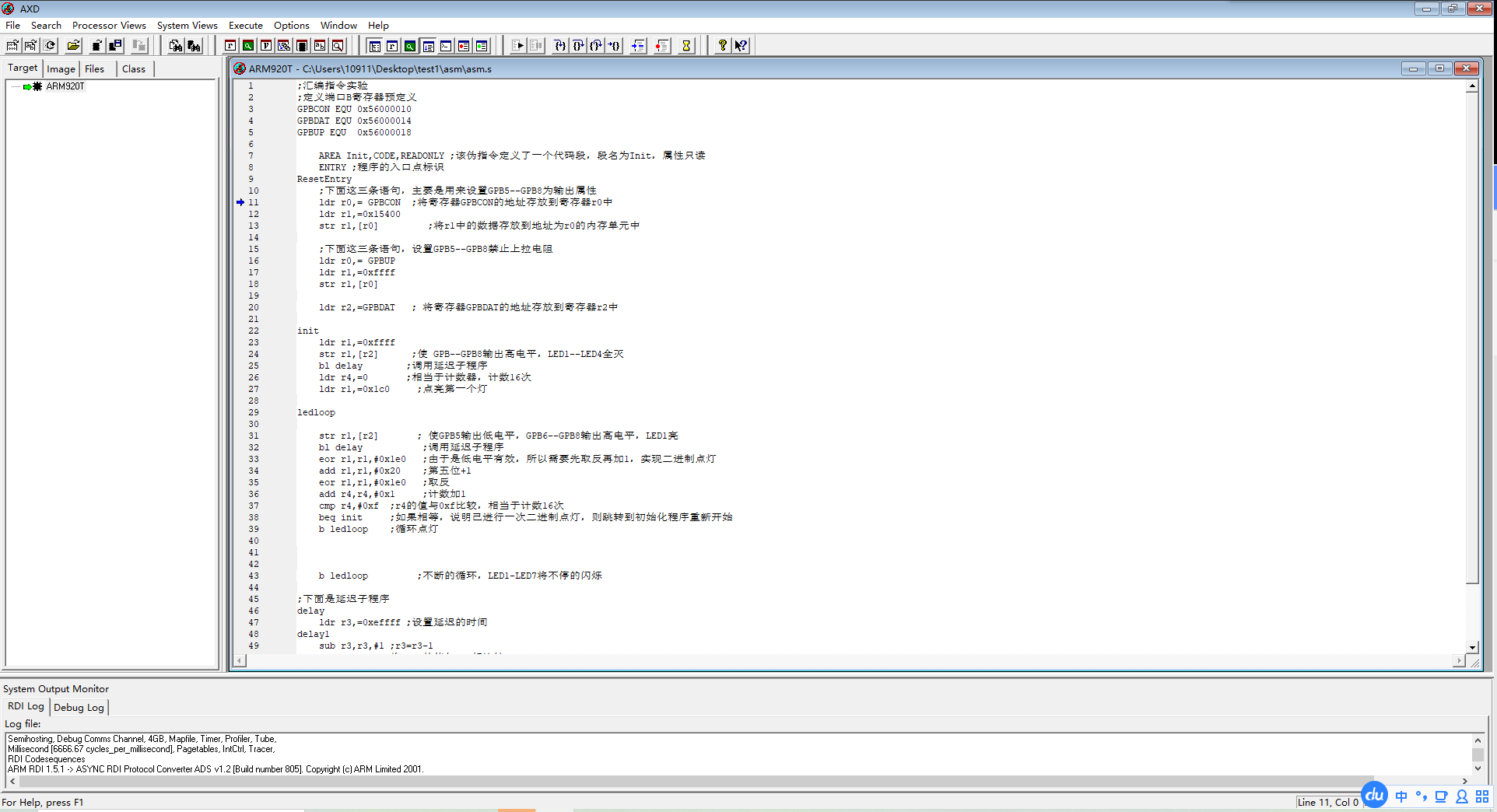


图 1 AXD调试成功截图

## 二、实验三：ARM汇编与C混合编程

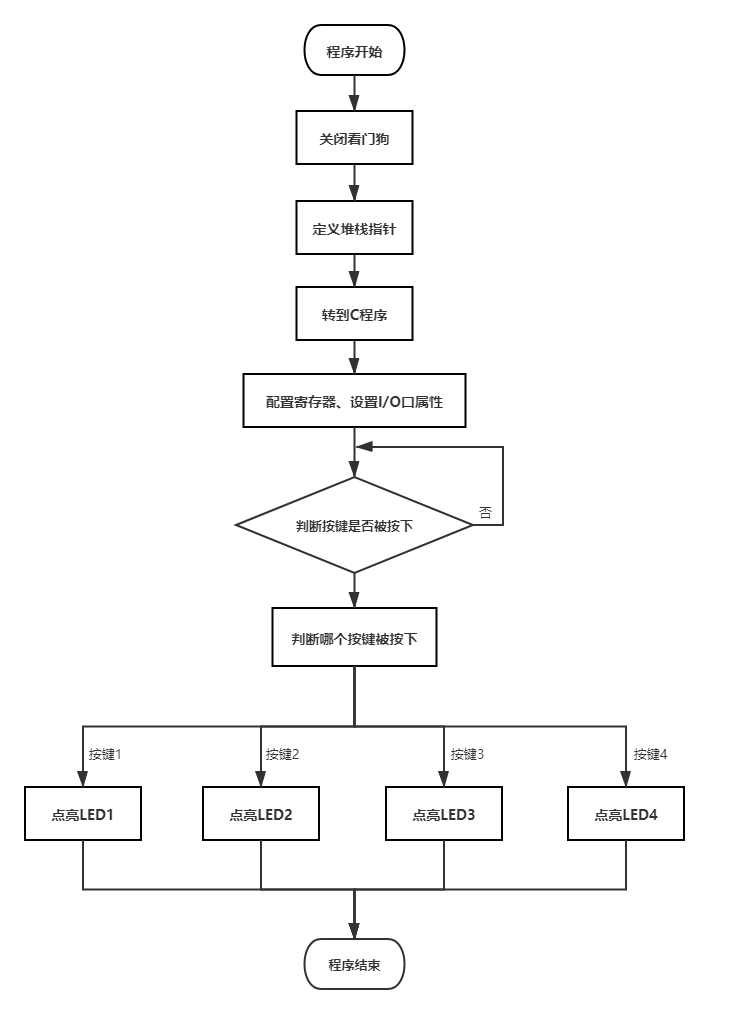


图 2 实验三软件实现流程图

## 三、实验四：ARM中断实验

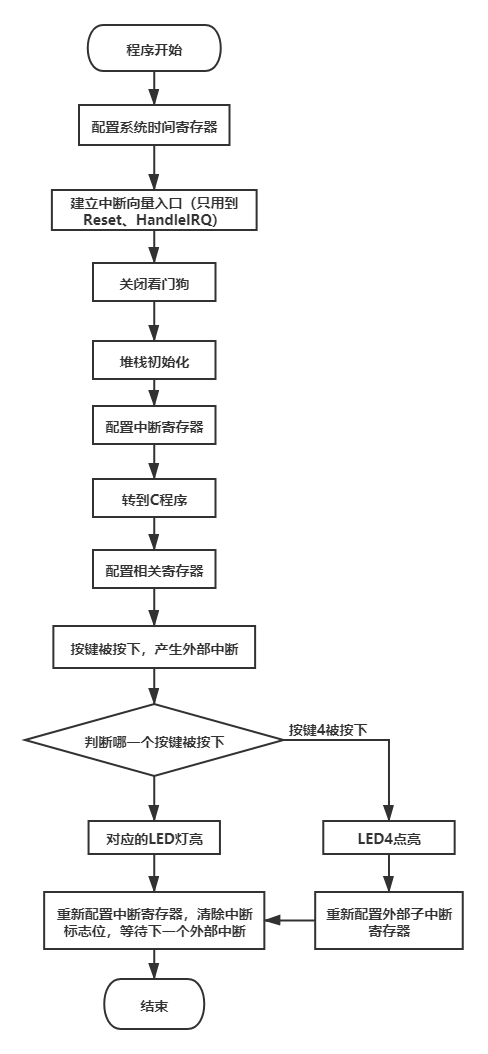


图 3 实验四软件实现流程图

## 四、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？**

答：

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。

C语言是一种结构化的高级语言。其优点是可读性好，移植容易，编程难度较低，是普遍使用的一种计算机语言。

不能完全摈弃其中一种语言，对于汇编语言，不同的CPU，其汇编语言可能会有差异，所以不易移植；而C语言占用的资源较多，执行效率没有汇编高。同时在嵌入式开发的过程中，我们常常在汇编语言中进行C程序运行环境的初始化（包括系统初始化和代码搬移初始化），再跳转到C程序。往往两种代码相互配合，才能写出开销小，执行效率高的程序，所以不能完全摈弃其中的一种语言。

**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？实验二，三程序中参数是如何传递的？**

答：

C语言调用汇编时，参数传递是通过ATPCS的规定来进行的，如果函数参数不多于4个，则使用寄存器R0~R3来进行传递；多于4个参数时将借助栈来传递。

实验二LED函数中，有三个参数，分别通过R0、R1、R2三个寄存器来传递；delay()有两个参数，通过R0、R1来传递。

实验三中，Reset()用到了sp堆栈指针，说明参数超过了四个；delay()有两个参数，通过R0、R1来传递；EnterCritical()和ExitCritical()用R0、R1传递参数。

**3. C 语言中和汇编语言中是如何操作寄存器的？**

答：

C语言使用#define定义一个名称，将该名称指向寄存器的地址，通过对该名称指针的操作，从而对寄存器进行操作。

汇编中可以直接通过各种汇编指令对寄存器进行操作。

**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。**

答：

实验二、实验三中“ARM Linker”下“Output”选项卡中“RO Base”的地址设为0x30000000，在实验四中设为0x00000000。这是因为实验二、实验三是从Nor Flash启动，0x30000000是SDRAM的物理起始地址；而实验四使用了中断，必须从0x00000000开始，否则无法正常进入中断子程序，导致程序无法正常运行。

**5. 在中断实验中为什么要把可执行程序下载到 NAND Flash 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？**

答：

由于内存表中的中断源排列顺序是固定的，不能随便更改，即中断向量表是从0x00000000开始，每一种异常对应一个地址（如IRQ对应0x00000018，FIQ对应0x0000001c）。如果程序直接下载到SDRAM（对应物理起始地址为0x30000000）中，会使中断向量表的位置发生偏移，无法正常进入中断，导致程序无法正常运行。而如果把程序下载到NAND Flash（对应物理起始地址0x00000000）中，中断向量表位置正确，程序可以正常运行。

**6. 结合实验，叙述 NAND Flash 启动的流程。**

答：

首先将按键拨到Nor Flash模式，在Nor Flash模式下将代码下载到NAND Flash，关闭电源。将开关拨到NAND Flash模式，打开电源使能NAND Flash控制器的自动引导模式，NAND Flash的首4K个字节被复制到内部缓存Steppingstone；CPU开始执行Steppingstone中的引导程序，将更多的代码拷贝到SDRAM中，然后跳转到SDRAM中运行。

## 体会和建议

体会：通过前两次的嵌入式实验，我对windows环境下裸机开发有了初步的了解。由于实验刚开始时，对知识点掌握的不是很透彻，只能机械的按照实验指导书一步步的操作。在进行实验一至三时还比较顺利，但是在完成实验四的过程中遇到了许多的困难，实验四的EINT4是一个次级中断，这个问题困扰了我很久，最终通过查阅芯片手册和网上的资料得以解决。我认为，对于嵌入式的学习，仅仅局限于上课听讲是不够的，只有在自己动手之后才能对知识点有更深入的理解。

建议：在本次实验中只对流水灯进行了简单的操作，开发板上还有许多资源没有被使用，或许可以在之后的实验中加入这一部分资源的教学，帮助同学们更好的了解一这款开发板。

# Linux环境下裸机开发实验报告

## 实验五：熟悉Linux操作系统

我对Linux操作系统的理解：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。相比windows，Linux有更多的开源代码，支持灵活剪裁配置，灵活性高，功能强大，运行稳定。在实验五中，我们在虚拟机上运行Linux系统，初次尝试在Linux系统上进行操作，在Linux系统中“一切皆文件”，对Linux系统的操作本质上是对文件进行操作，这与我们熟悉的windows完全不同。实验五只是运行Linux系统的一种方式，还有很多种运行方式值得我们慢慢去探索。

## 实验六：Linux常用命令的熟悉

### 常规命令行

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、mkdir 创建目录

⑥、cp 复制文件或目录

⑦、mv 移动或更名现有的文件或目录

⑧、rm 删除文件或目录

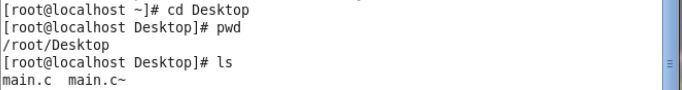
⑨、ifconfig 网络配置

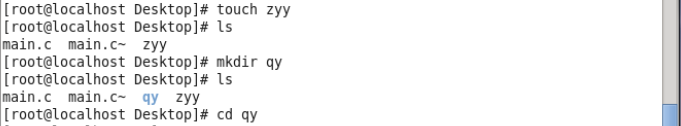
⑩、echo 显示一串字符

⑪、help 帮助

⑫、date 显示当前的日期和时间

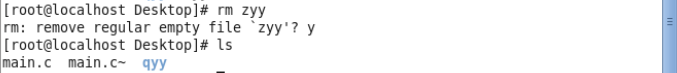
### 2、命令行执行截图







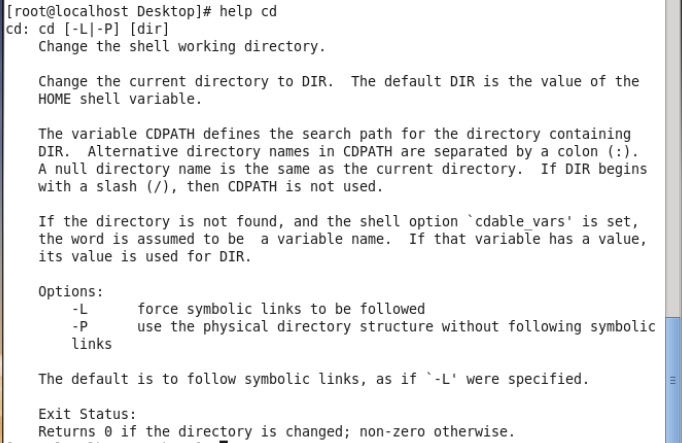












## 实验六、实验八的实验过程

### 1.实验六实验过程

①、打开虚拟机，登录Linux操作系统

②、打开命令终端Terminal

③、输入命令行

### 实验八实验过程

1. 执行命令#vi main.c，输入代码
2. 执行命令gcc -v确认gcc以正确安装
3. 执行命令gcc -o test main.c生成可执行文件test
4. 执行ls确认test文件已正确生成
5. 执行file test查看文件信息
6. 执行命令./test 123 456 abc def ghi 执行可执行文件test
7. 查看运行结果

### 实验截图

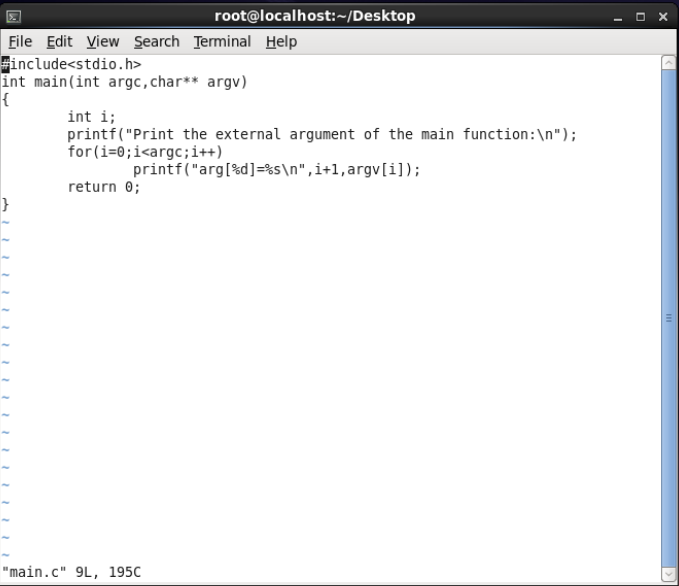


图 4在vi中编写main.c

图 5 查看gcc信息



图 6生成可执行文件

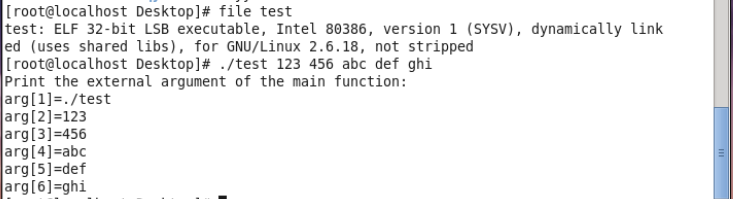


图 7查看文件信息并执行文件

## 实验九、实验十的实验过程

### 实验九过程及截图

1. 执行命令tar xvzf arm-linux-gcc-4.4.3-20100728.tar.gz -C /解压实验提供的交叉编译工具压缩包
2. 修改环境变量
3. 查看修改的环境变量是否生效
4. 执行命令arm-linux-gcc -v查看交叉编译工具信息



图 8查看交叉编译工具信息

### 实验十过程及截图

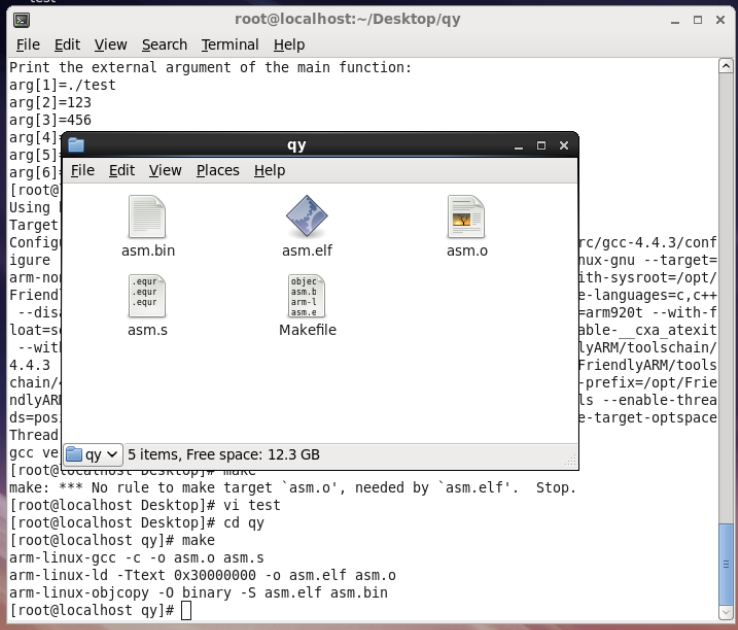
1. 编写makefile文件
2. 执行cd命令转到makefile文件目录下
3. 执行make命令，生成可执行文件asm.bin
4. 将asm.bin拖到windows环境下
5. 按照windows环境下裸机开发时使用的方法，通过串口超级中断将可执行文件下载到SDRAM中执行  
   

图 9Linux系统下生成可执行文件

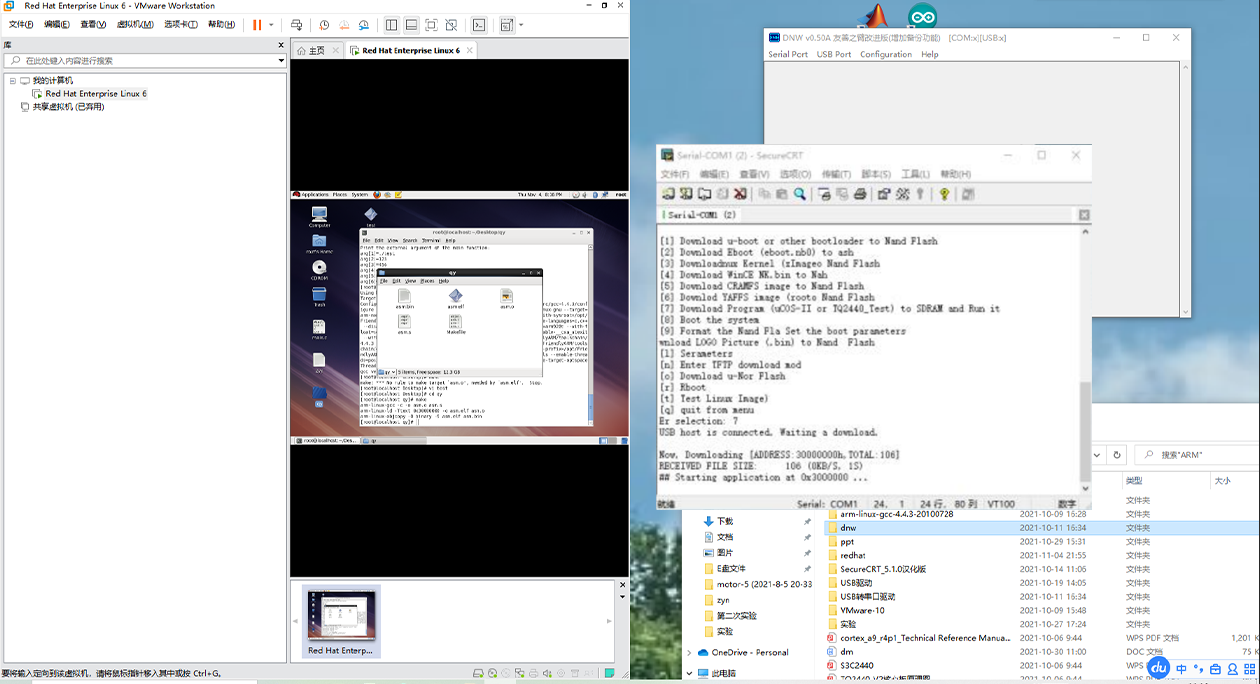


图 10将可执行文件拖到windows下操作

## 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：

在windows环境下使用ADS等软件进行裸机开发，前期需要在软件上进行相应的配置，编写完程序后直接make即可。而在Linux环境下，需要用到arm-linux-gcc交叉编译工具，还需要完成makefile的编写。

1. **make 及 makefile 的作用什么？**

答：

make是一个命令，根据makefile文件编译源代码、连接、生成目标文件、可执行文件。

makefile定义了一系列的规则来指定，哪些文件需要先编译，哪些文件需要后编译，哪些文件需要重新编译，甚至于进行更复杂的功能操作，因为 makefile就像一个Shell脚本一样，其中也可以执行操作系统的命令。makefile带来的好处就是--“自动化编译”,一旦写好，只需要一个make命令，整个工程完全自动编译，极大的提高了软件开发的效率。

使用make工具，可以将大型的开发项目分解成多个易于管理的模块。使用make和makefile工具可以高效处理各个源文件之间的复杂的相互关系，进而取代了复杂的命令行操作，提高了应用程序开发的效率。

1. **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，第二篇实验的Linux系统是在虚拟机上运行的Linux系统；而第三篇实验的Linux系统是在开发板上运行的系统，是经过裁剪定制的Linux系统。

1. **关于 Linux 操作系统，你还想要了解什么？**

答：

我还想了解在哪里可以找到Linux开源的资料，如何在Linux环境下运行C++、python的程序等等。

**5. 关于嵌入式 Linux 操作系统的开发，你还希望学习什么？**

答：

在实验十一和实验十二中，我只是机械的跟着实验指导书一步步的完成了实验，对Linux系统移植是什么？为什么？还有很多疑问。今后希望能多学习到这方面的知识。此外，还希望老师能够对我们后续自主学习Linux提供一些指导意见。

## 体会和建议

体会：在第二篇实验中，本质上只是换了一种方式写代码、换了一种方式进行编译，感觉没有体会到Linux相对于windows环境的优势所在；在第三篇实验中，老师已经给我们准备好了现成的资料，我们只需要按照实验指导书完成即可，没有真正理解为什么要这样操作。我感觉本次实验总体上来说还是照葫芦画瓢，如果换了一款开发板，或者提出了其他开发需求，我们可能就会感觉到无从下手。当然，Linux系统本身就比较复杂，在短短的几节课时间内我们只能对Linux系统有一个初步的了解，今后还要通过不断的学习来了解Linux系统。

建议：本课程课时太少，讲解速度很快，感觉非常难跟上，今后是否能增加授课时间，将课程进度放慢一些，再多讲解一些Linux部分的内容呢？