

实验一 嵌入式开发环境搭建实验

【实验目的】

1. 掌握嵌入式 Linux 开发的流程
2. 熟悉嵌入式 Linux 的环境搭建

【实验设备】

1. 装有 Ubuntu 虚拟机的 PC 机一台
2. 实验箱一台
3. 母-母串口线或 USB A-B 延长线一根
4. 网线一根

【实验要求】

1. 能够熟练掌握嵌入式 Linux 的开发流程。
2. 熟练嵌入式 Linux 开发的环境搭建。

【实验原理】

1. 预备知识

绝大多数 Linux 软件开发都是以本地方式进行的，即本机（HOST）开发、调试，本机运行的方式。这种方式通常不适合于嵌入式系统的软件开发，因为对于嵌入式系统的开发，没有足够的资源在本机（即板子上系统）运行开发工具和调试工具。通常的嵌入式系统的软件开发采用一种交叉编译调试的方式。交叉编译调试环境建立在宿主机（即一台 PC 机）上，对应的开发板叫做目标板。

运行 Linux 的 PC（宿主机）开发时使用宿主机上的交叉编译、汇编及连接工具形成可执行的二进制代码（这种可执行代码并不能在宿主机上执行，而只能在目标板上执行），然后把可执行文件下载到目标机上运行。调试时的方法很多，可以使用串口，以太网口等，具体使用哪种调试方法可以根据目标机处理器提供的支持作出选择。宿主机和目标板的处理器一般不相同，宿主机为 Intel 处理器，而目标板如凌阳 Cortex-A8 实验仪，核心芯片为三星 S5PV210。GNU 编译器提供这样的功能，在编译器编译时可以选择开发所需的宿主机和目标机从而建立开发环境。所以在进行嵌入式开发前第一步的工作就是要安装一台装有指定操作系统的 PC 机作宿主开发机，对于嵌入式 LINUX，宿主机上的操作系统一般要求为 Ubuntu。嵌入式开发通常要求宿主机配置有网络，支持 NFS（为交叉开发时 mount 所用）。然后要在宿主机上建立交叉编译调试的开发环境。

2. 嵌入式 Linux 开发流程

嵌入式 linux 开发，根据应用需求的不同有不同的配置开发方法，但是一般都要经过以下过程：

■ 建立开发环境

操作系统一般使用 Ubuntu，选择定制安装或全部安装，通过网络下载相应的 GCC 交叉编译器进行安装（比如 arm-linux-gcc、arm-uclibc-gcc），或者安装产品厂家提供的交叉编译器。

■ 配置开发主机

配置 MINICOM，一般参数为波特率 115200，数据位 8 位，停止位 1，无奇偶校验，软硬件控制流设为无。在 WINDOWS 下的超级终端的配置也是这样。MINICOM 软件的作用是作为调试嵌入式开发板信息输出的监视器和键盘输入的工具。

配置网络，主要是配置 NFS 网络文件系统，需要关闭防火墙，简化嵌入式网络调试环境设置过程。

■ 建立引导装载程序 BootLoader

从网络上下载一些公开源代码的 BootLoader，如 U-Boot、BLOB、VIVI、LILO、ARM-Boot、Redboot

等，根据自己具体芯片进行移植修改。有些芯片没有内置引导装载程序，比如三星的 ARM7、ARM9、Cortex-A8 系列芯片，这样就需要编写烧写开发板上 flash 的烧写程序，网络上有免费下载的 WINDOWS 下通过 JTAG 并口简易仿真器烧写 ARM 外围 flash 芯片的程序。也有 LINUX 下开源代码的 J-Flash 程序。如果不能烧写自己的开发板，就需要根据自己的具体电路进行源代码修改。这是让系统可以正常运行的第一步。如果你购买了厂家的仿真器当然比较容易烧写 flash 了，但是其中的核心技术是无法了解的。这对于需要迅速开发应用的人来说可以极大地提高开发速度。下载别人已经移植好的 Linux 操作系统，如 UCLinux、ARM-Linux、PPC-Linux 等，如果有专门针对你所使用的 CPU 移植好的 Linux 操作系统那是再好不过，下载后再添加自己的特定硬件的驱动程序，进行调试修改，对于带 MMU 的 CPU 可以使用模块方式调试驱动，对于 UCLinux 这样的系统只能编译进内核进行调试。

■ 建立根文件系统

从 www.busybox.net 下载使用 BusyBox 软件进行功能裁减，产生一个最基本的根文件系统，再根据自己的应用需要添加其他的程序。默认的启动脚本一般都不会符合应用的需要，所以就要修改根文件系统中的启动脚本，它的存放位置位于/etc 目录下，包括：/etc/init.d/rc.S、/etc/profile、/etc/.profile 等，自动挂装文件系统的配置文件/etc/fstab，具体情况会随系统不同而不同。根文件系统在嵌入式系统中一般设为只读，需要使用 mkcramfs、genromfs 等工具产生烧写映像文件。建立应用程序的 flash 磁盘分区，一般使用 JFFS2 或 YAFFS 文件系统，这需要在内核中提供这些文件系统的驱动，有的系统使用一个线性 Flash（NOR 型）512K~32M，有的系统使用非线性 Flash（NAND 型）8~512M，有的两个同时使用，需要根据应用规划 Flash 的分区方案。

■ 开发应用程序

应用程序可以下载到根文件系统中，也可以放入 YAFFS、JFFS2 文件系统中，有的应用程序不使用根文件系统，而是直接将应用程序和内核设计在一起，这有点类似于 UCOS-II 的方式。

烧写内核、根文件系统、应用程序

■ 发布产品

3. 对宿主 PC 机的性能要求

- 由于 Ubuntu 安装后占用空间约为 2.4G~5G 之间，还要安装 ARM-LINUX 开发软件，因此对开发计算机的硬盘空间要求较大。

硬件要求：

- CPU：高于奔腾 500M，推荐高于奔腾 1.0G；
- 内存：大于 128M，推荐 256M；
- 硬盘：大于 10G，推荐高于 40G。

【实验步骤】

实验箱是一个完整的计算机系统，其内部运行了一个与 PC 上类似的 Linux 系统。在一般的开发过程中，我们需要首先在 PC 端做一些准备工作，这些设置包括：实验箱与 PC 的硬件连接、串口通信软件设置、网络环境设置。

■ 实验箱与 PC 的硬件连接

一般情况下，实验箱同时需要两种方式与 PC 建立连接：串口和以太网。首先使用标准 9 针串口线，

将实验仪的 UART0 与 PC 的串口相连；然后，使用实验箱附带的网线，将实验箱的以太网接口与 PC 的网卡直接相连，或者将实验箱与路由器相连。这样就完成了硬件连接，如图 1.1 所示。



图 1.1 实验箱与 PC 的基本硬件连接

■ 串口通信软件设置

在 PC 端需要使用串口通信软件来对实验箱进行控制。通常情况下，使用 Windows 系统自带的“超级终端”工具即可（或者用户也可以使用其他同类型的软件，这里仅针对“超级终端”做详细设置说明）。

首先在“开始”菜单中，找到“程序”→“附件”→“通讯”→“超级终端”，如图 1.2 所示。

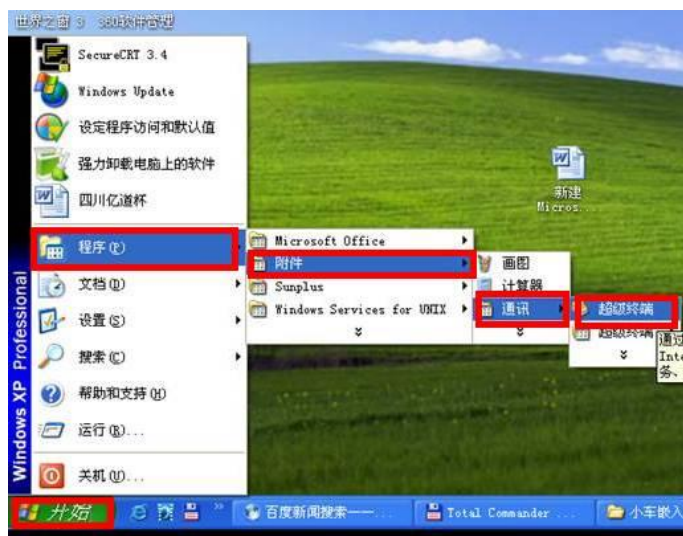


图 1.2 打开超级终端

设置超级终端名称，任意名称即可，如图 1.3 所示：

选择串口，例如：如果自己串口线接在串口 1 上就选择 COM1，如图 1.4 所示：

通达部分后期版本可使用USB给实验箱供电，com口能显示但超级终端连接不上

使用USB串口通信时，选择COM1外的其他串口（一般为COM3），其他不变



图 1.3 输入连接的名称



图 1.4 选择连接的串口

设置串口属性,每秒位数设置为 115200, 数据流控制选择无, 如图 1.5 所示:



图 1.5 选择串口的设置属性

此时, 将物联网多网设计平台的电源打开, A8 实验仪的拨动开关拨至“ON”, 并按下实验仪上的“Power”键(位于显示屏右下位置), 可以在超级终端中看到图 1.6 所示的启动提示信息。

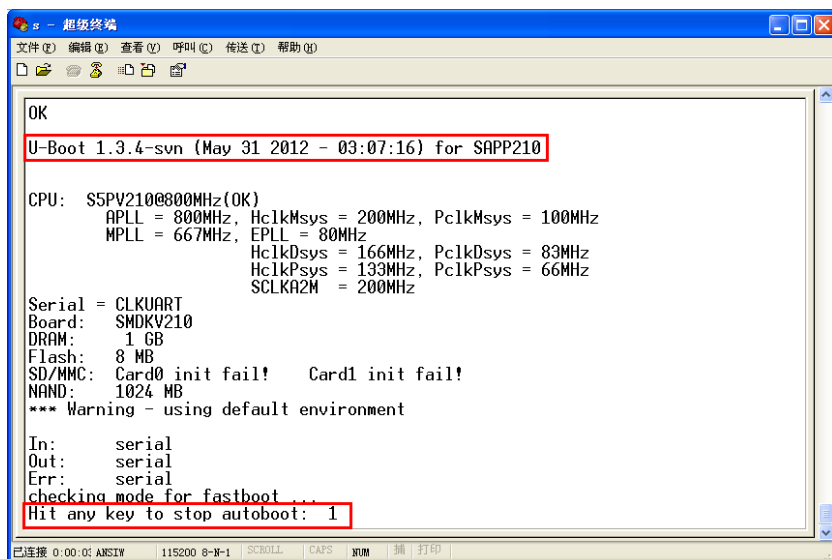


图 1.6 U-boot 启动界面

看到“Hit any key to stop autoboot”的提示，表示实验仪正在准备启动 Linux 系统。此时，如果不做任何操作，则在倒计时结束后将会启动 Linux。如果在倒计时的过程中按下键盘的空格键，即可进入到 U-Boot 的命令行，可以对系统启动参数进行调整，或者可以重新安装操作系统等。

待系统正常启动之后，可以看到“SAPP210.XXXX login:”的提示，如图 1.7 所示。其中，XXXX 根据不同的实验箱可能会有所不同。此时，表示 Linux 系统已经正常启动，等待用户登录。

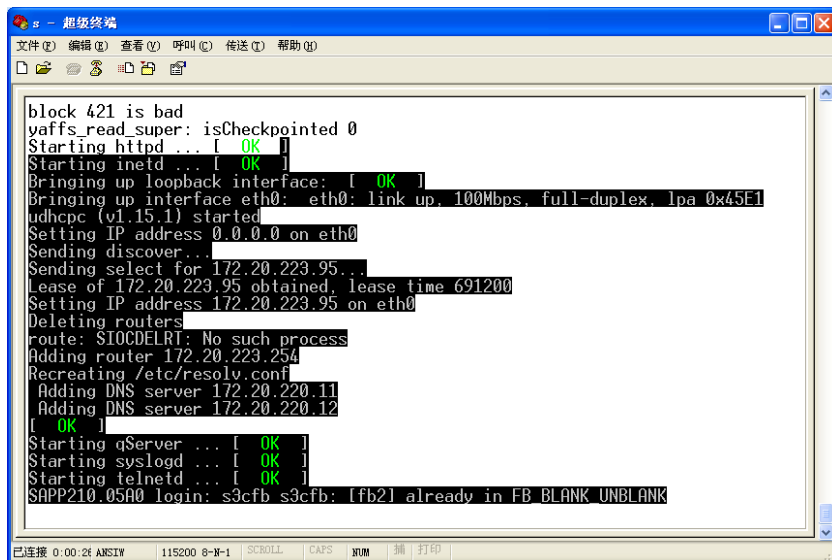


图 1.7 Linux 启动完成

按下 Enter 键，进入登陆，输入用户名“root”，密码“111111”，即可登录到系统，如图 1.8 所示。注意，密码输入时超级终端中不会有任何显示。登录成功之后，可以看到类似于“[root@SAPP210/root]#”的提示。

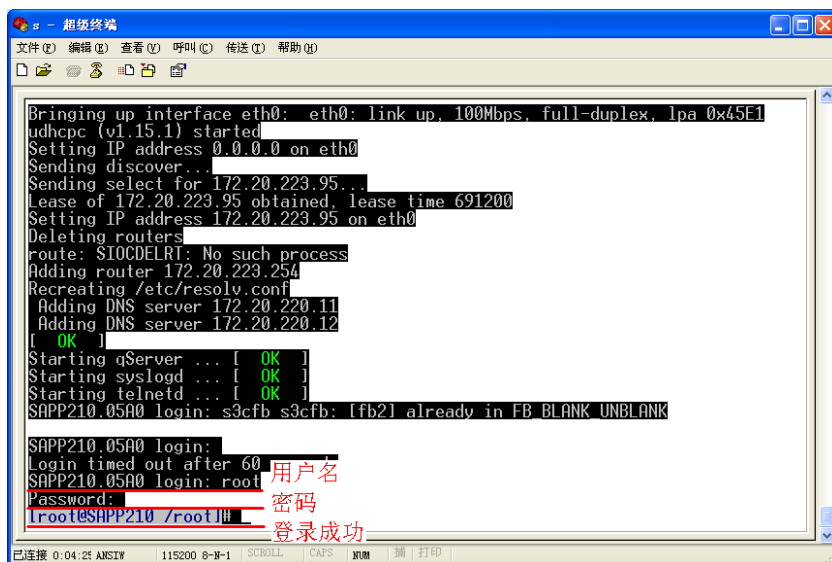


图 1.8 登录到实验箱的 Linux 系统

可以看到，实验箱的 Linux 系统启动过程中，会输出一些带有颜色的符号，导致超级终端软件的屏幕出现黑白相间的花屏。可以执行“clear”命令来清除屏幕，如图 1.9 所示。

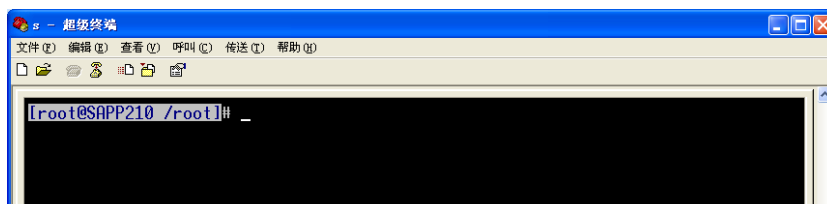


图 1.9 使用 clear 命令清屏

■ 网络环境设置

如果实验箱使用网线连入局域网，而局域网中存在 DHCP 服务器，则实验箱启动过程中，将会自动获取到 IP 地址，正如上面的图 1.7 中看到的这些提示一样：

```
udhcpd (v1.15.1) started
Setting IP address 0.0.0.0 on eth0
Sending discover...
Sending select for 172.20.223.95...
Lease of 172.20.223.95 obtained, lease time 691200
Setting IP address 172.20.223.95 on eth0
Deleting routers
route: SIOCDELRT: No such process
Adding router 172.20.223.254
Recreating /etc/resolv.conf
  Adding DNS server 172.20.220.11
  Adding DNS server 172.20.220.12
[ OK ]
```

其中，“172.20.223.95”即为实验箱的 IP 地址。将实验箱连入局域网，用 DHCP 服务器为其分配 IP 地址，是我们推荐的做法。

然而，如果没有局域网的条件，或者局域网不具备 DHCP 服务器，则也可以通过手动配置的方式，来为实验仪分配 IP 地址。使用手动配置实验仪 IP 地址的方法必须设置电脑为静态 IP，方法如下：

● 设置电脑为静态 IP

在桌面上“网上邻居”图标上点击鼠标右键，选择“属性”，如图 1.10 所示：

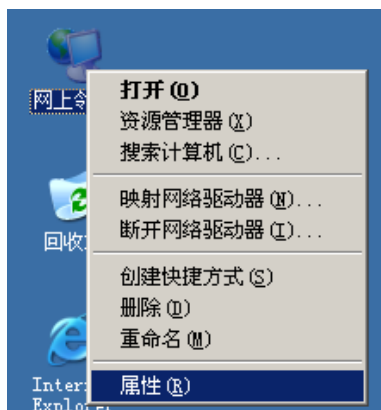


图 1.10 查看网上邻居的属性

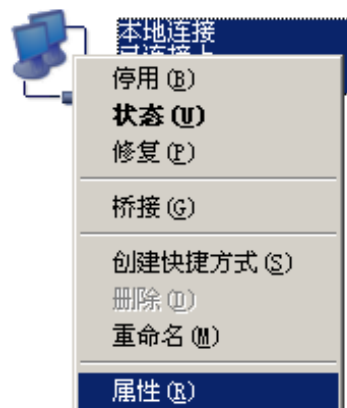


图 1.11 查看本地连接的属性

在打开的窗口中，找到“本地连接”，点击鼠标右键，选择“属性”，如图 1.11 所示；

在打开的窗口中，选择“Internet 协议 (TCP/IP)”，并点击“属性”按钮，如图 8.12 所示；

在弹出的“Internet 协议 (TCP/IP) 属性”对话框中，按照如图 1.13 设置 IP，单击“确定”按钮，就为电脑设置好静态 IP 了。注意：在本例中，将电脑的 IP 地址设置为“192.168.87.1”，如用户对计算机网络熟悉，也可以按照自己的需要进行设置。

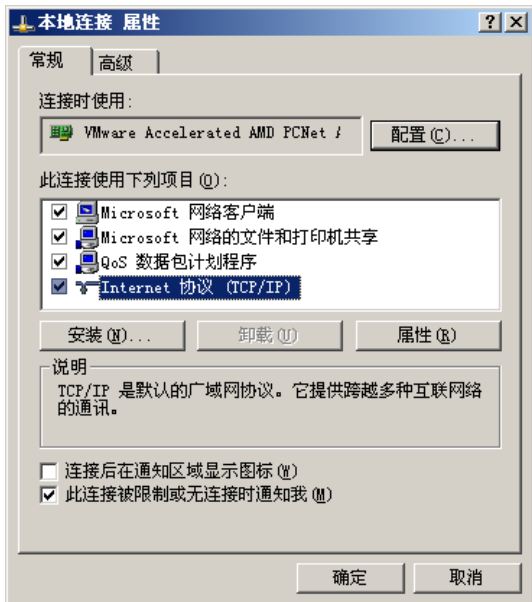


图 1.12 查看 Internet 协议属性

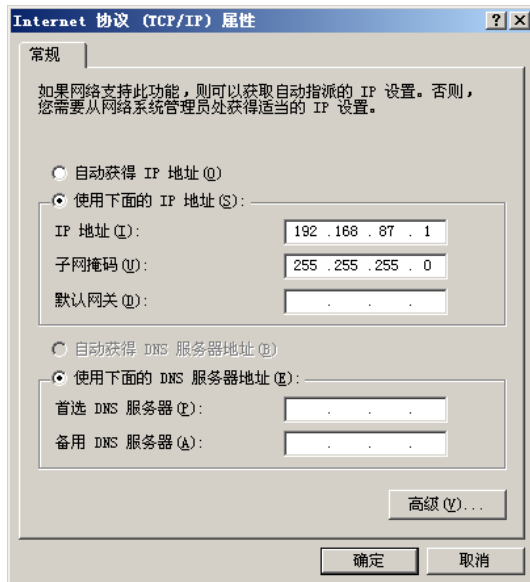


图 1.13 设置电脑静态 IP

● 配置实验仪 IP 地址

在超级终端中，执行命令“`ipconfig eth0 -i 192.168.87.130 -m 255.255.255.0 -g 192.168.87.1`”，即可为实验箱手动配置 IP 地址，如图 1.14 所示。

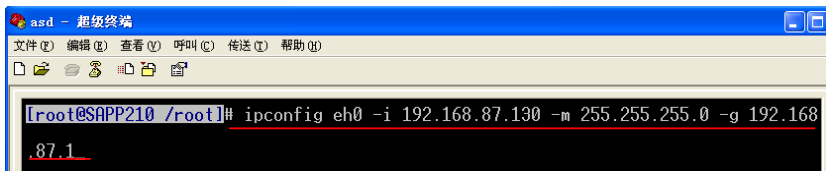


图 1.14 手动配置实验箱的 IP 地址

其中，-i 后面的参数是实验箱的 IP 地址；-m 后面的参数是子网掩码；-g 后面的参数是网关地址。如果不需要网关，可以将-g 和其后面的参数省略。

设置完成之后，需要执行“`service network restart`”命令重启网络服务，使设置生效，如图 1.15 所示。需要注意的是，实验箱的 IP 地址需要设置为与电脑同一个网段，例如，在本例中，电脑的 IP 地址为“192.168.87.1/255.255.255.0”，而实验箱的 IP 地址为“192.168.87.130/255.255.255.0”。

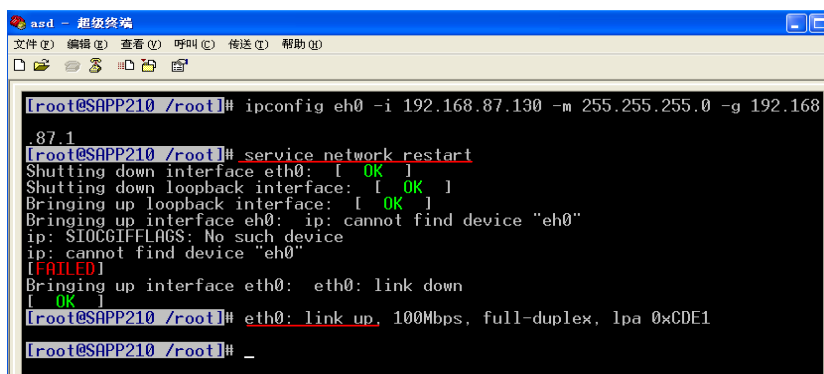


图 1.15 重启网络服务

当看到“eth0: link up”的提示，表示配置已经生效。

如需查看实验箱当前的 IP 地址，可以执行命令“ifconfig eth0”，如图 1.16 所示。

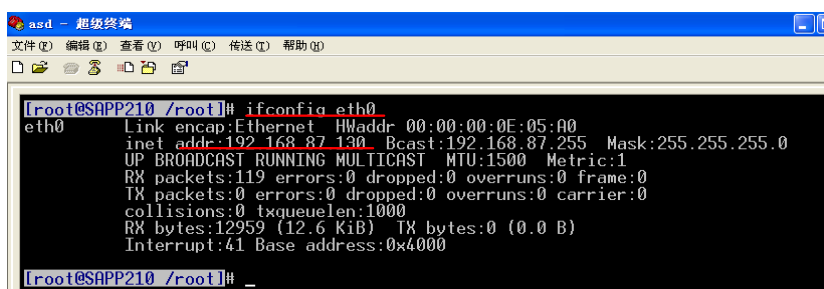


图 1.16 查看当前的 IP 地址

注：如需将实验箱重新配置成自动获取 IP 地址，只需执行命令“ipconfig eth0 -a”，并重启网络服务即可。

5. 在 Ubuntu 下编译嵌入式 C 程序

Ubuntu 10.10 with contiki

此过程是嵌入式 Linux 开发非常重要的一个过程，后续绝大部分的实验都会重复使用本节的方法，以便在 Ubuntu 下编译可以在实验箱上运行的程序。

用户无论在 Windows 下，或者在 Ubuntu 下编写完 C 程序之后，都必须首先保证该源程序文件存储到 Ubuntu 系统内。在本例中，我们首先在 Ubuntu 系统中的“主文件夹”中保存一个名为“hello.c”的文件。打开“主文件夹”的方法如图 1.17 所示。



图 1.17 打开 Ubuntu 系统的主文件夹

在主文件夹下，单击右键选“创建文档”->“空文件”，保存文档名为“hello.c”，如图 1.18 所示。

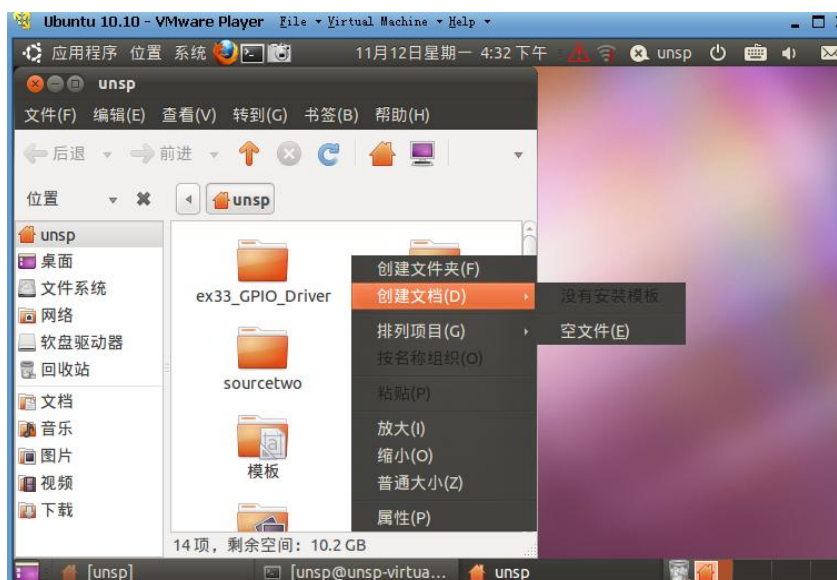


图 1.18 在主文件夹下创建 hello.c 文档

在 hello.c 文档中输入以下内容：

```
#include <stdio.h>
int main(int argc, char *argv[])
{
    printf("Hello, world!\n");
    return 0;
}
```

为了编译这个 C 程序，我们需要打开 Ubuntu 系统中的“终端”程序，该程序是一个命令行工具，可以运行标准的 Linux 命令，并可以用来编译程序。打开“终端”程序有两种方法：在 Ubuntu 系统左上角的“应用程序”菜单下，依次找到“附件”→“终端”，点击即可打开，如图 1.19 所示。



图 1.19 在“应用程序”中打开“终端”

或者直接点击 Ubuntu 系统中“系统”菜单右侧的“终端”快捷图标，如图 1.20 所示。



图 1.20 使用快捷图标打开“终端”

“终端”程序打开之后，可以看到图 1.21 所示的界面。

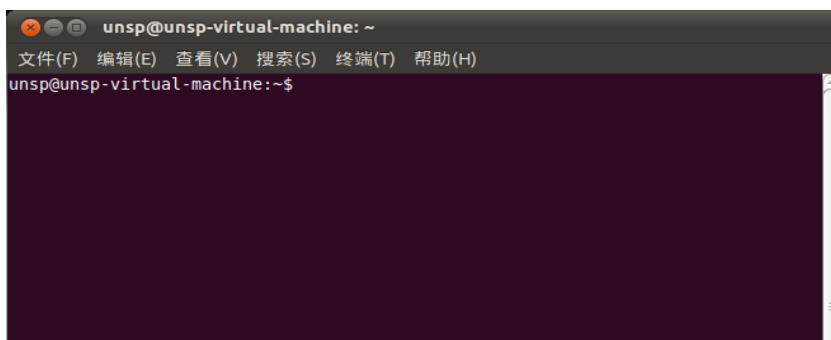


图 1.21 “终端”程序界面

终端程序打开之后，默认的工作目录即为“主文件夹”，在终端中输入“ls”命令，可以看到 hello.c 文件，如图 1.22 所示。

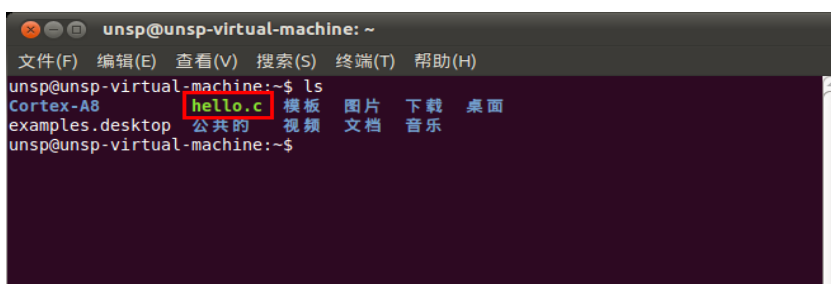


图 1.22 使用 ls 命令确认文件是否存在当前目录

输入命令“arm-linux-gcc -o hello hello.c”，即可将 hello.c 程序编译成为实验箱可以运行的可执行文件，如图 1.23 所示。其中，“arm-linux-gcc”表示用于 ARM 系列芯片的编译器，“-o hello”表示编译之后生成的可执行文件的名字为“hello”，最后的“hello.c”即为待编译的 C 源程序文件。

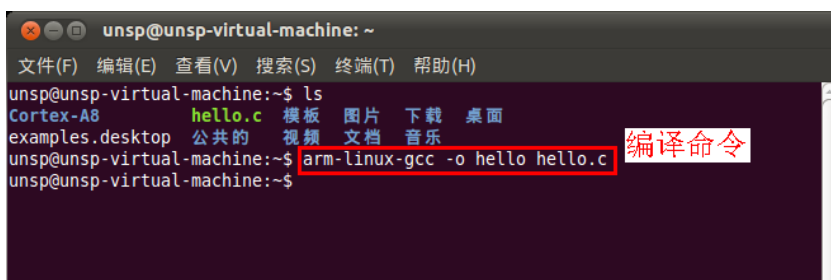


图 1.23 编译 hello.c 程序

使用 ls 命令查看，可以看到目录下多了一个“hello”文件，该文件即为用于实验箱的可执行程序文件，如图 1.24 所示。

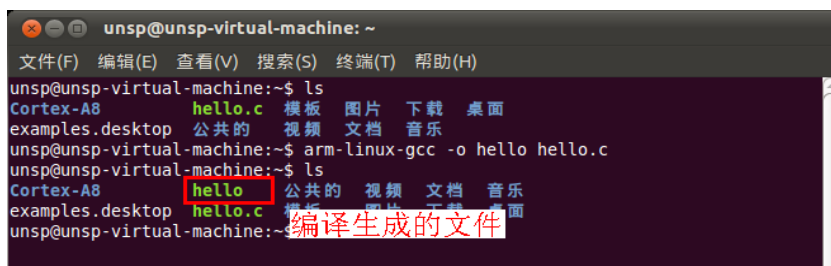


图 1.24 确认编译的文件是否生成

6. 将编译生成的文件复制到实验箱上并运行

首先将编译好的可执行程序文件从 Ubuntu 中复制到 Windows 系统，然后打开“我的电脑”，在地址栏中输入“ftp://开发板的 IP 地址”，如图 1.25 所示，其中，开发板的 IP 地址可以参考图 1.16 中的“ifconfig eth0”命令来查看。

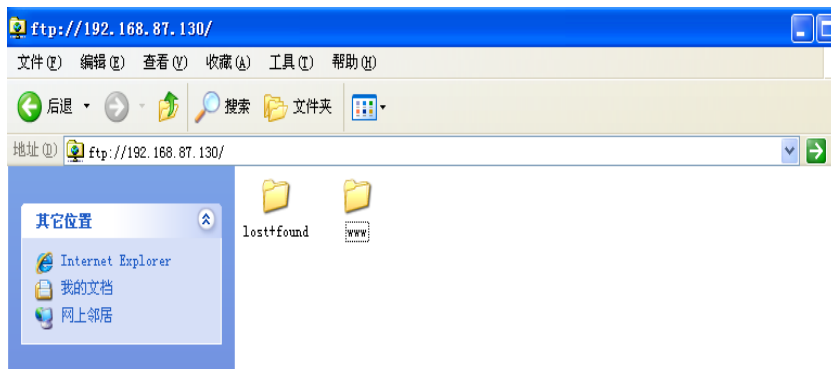


图 1.25 通过 FTP 访问实验箱的文件

接下来，跟操作本地文件一样，我们可以使用复制/粘贴的方式将编译好的 hello 文件放入实验箱内，如图 1.26 所示。

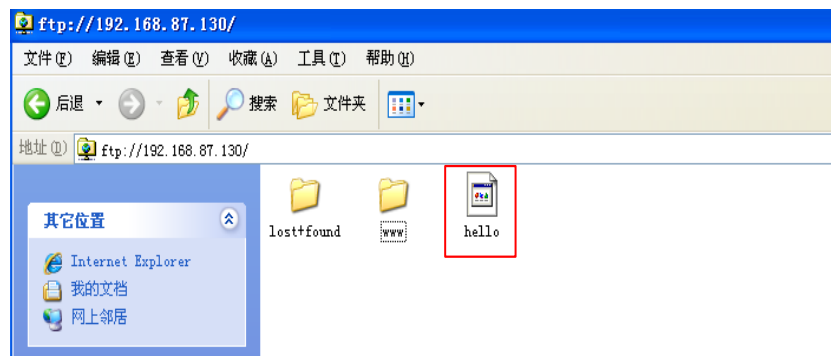


图 1.26 通过 FTP 将文件直接粘贴到实验箱内

在超级终端中，输入“ls”命令，可以看到“hello”文件已经被复制到了实验箱的系统内，如图 1.27所示。

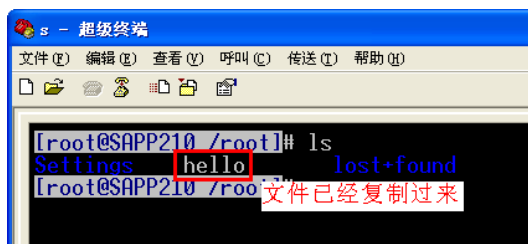


图 1.27 确认文件已经复制到实验箱

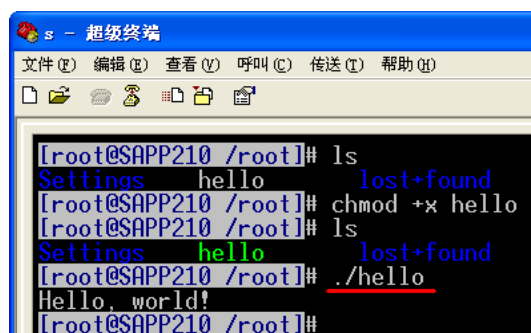
在超级终端中执行命令“chmod +x hello”，为 hello 文件增加可执行权限，如图 1.28 所示。



```
[root@SAPP210 /root]# ls
Settings  hello      lost+found
[root@SAPP210 /root]# chmod +x hello
[root@SAPP210 /root]# ls
Settings  hello      lost+found
[root@SAPP210 /root]#
```

图 1.28 为文件增加可执行权限

最后执行“./hello”命令，即可运行 hello 程序，如图 1.29 所示。



```
[root@SAPP210 /root]# ls
Settings  hello      lost+found
[root@SAPP210 /root]# chmod +x hello
[root@SAPP210 /root]# ls
Settings  hello      lost+found
[root@SAPP210 /root]# ./hello
Hello, world!
[root@SAPP210 /root]#
```

图 1.29 运行程序

