# 实验五　终端及串口编程

## 基本信息

**实验课程：**Linux程序设计 **设课形式：**非独立

**课程学分：**2 **实验项目：**进程控制

**项目类型：**验证 **项目学时：**2

## 实验预习

**实验目的和要求：**

掌握串行通信编程的流程

学习串行通信端口的设置

串行通信中相关函数的应用

**实验条件：**

1、装有Linux操作系统的微型计算机；

## 实验过程

**5.1串行通信概述**

* 现在的串行通信端口（RS-232）是计算机上的标准配置。最为常见的应用是连接调制解调器进行数据传输。
* 计算机通常包含COM1和COM2两个串行通信端口。一般计算机的COM端口从外观上看有9个针脚。
* 在Linux中，所有的设备文件都位于“/dev”下，其中COM1、COM2对应的设备名依次为“/dev/ttyS0”、“/dev/ttyS1”。
* Linux对设备的操作方法和对文件的操作方法相同，因此，对串口的读写就可以使用简单的“read”、“write”函数来完成，所不同的是要对串口的一些参数进行配置。



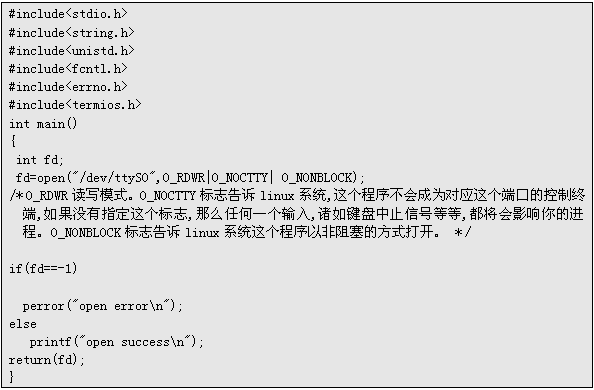
**5.2 串行通信程序的设计**

**5.2.1 串行通信程序设计流程**

**http:/www.gjwtech.com/scomm/pic/T64-1.gif**

**5.2.2 打开通讯端口**

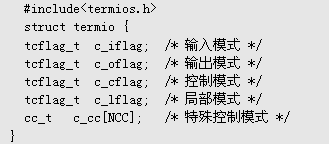
* **在Linux中把串口设备视为普通文件，可使用open()函数来打开串口设备。**
* **例5.1：打开PC的COM1串行通信端口。**
* **源程序代码com1open.c**

****

**思考题：要求以只读的模式打开计算机的COM2。**

**5.2.3 设置串口属性**

**在Ｌinux中若要对串口进行操作，如改变其波特率、字符大小等，就是对结构体stuct termios中成员的值进行设置。**

****

**注意：在这个结构中最为重要的是c\_iflag，通过对它的赋值，用户可以设置波特率、字符大小、数据位、停止位、奇偶校验位和硬件控制等。**

* **c\_oflag（输出模式）**

****

* **c\_cflag（控制模式）**

****

* **c\_lflag（局部模式）**

****

* **c\_cc[NCCS]（特殊控制字符）**

****

**思考题：在设置串口属性时,要求奇偶校验。请写出设置串口的语句。**

**5.2.4 串口通信程序设计主要语句说明**

* **串口通信一般分为接收端和发送端。**

**接收端：**

1. **打开PC的COM1端口。**

**如果以读写的方式打开COM1端口，语句可写为：**

**fd=open(“/dev/ttyS0”,O\_RDWR | O\_NOCTTY);**

1. **取得当前串口值，并保存至结构体变量oldtio。**

**tcgetattr(fd,&oldtio);**

1. **清除结构体变量newtio。**

**bzero(&newtio,sizeof(newtio));**

1. **设置串口参数。**
2. **假定设置波特率为38400，8个数据位，忽略任何调制解调器状态，同时启动接受器。**

**newtio.c\_cflag=BAUDRATE |CS8 |CLOCAL|CREAD;**

1. **忽略奇偶校验错误**

**newtio.c\_iflag=IGNPAR;**

1. **设输出模式非标准型，同时不回应。**

**ewtio.c\_oflag=0;**

1. **启用正规模式**

**newtio.c\_lflag=ICANON;**

1. **清除所有列队在串口的输入输出**

**tcflush(fd,TCIFLUSH);**

1. **设置当前的串口参数为newtio**

**tcsetattr(fd,TCSANOW,&newtio);**

1. **读取缓存中的数据。**

**read(fd,buf,255);**

1. **关闭串口。**

**close(fd);**

1. **恢复旧的端口参数**

**tcsetattr(fd,TCSANOW,&oldtio);**

**发送端：**

1. **打开PC的COM2端口。**

**fd=open(“/dev/ttyS1”,O\_RDWR | O\_NOCTTY);**

1. **取得当前串口值，并保存至oldtio。**

**tcgetattr(fd,&oldtio);**

1. **清除结构体newtio。**

**bzero(&newtio,sizeof(newtio));**

1. **设置串口参数。**
2. **设置波特率为38400，8个数据位，忽略任何调制解调器状态同时启动接受器。**

**newtio.c\_cflag=BAUDRATE |CS8 |CLOCAL|CREAD;**

1. **忽略奇偶校验错误**

**newtio.c\_iflag=IGNPAR;**

1. **设输出模式非标准型，同时不回应。**

**newtio.c\_oflag=0;**

1. **启用正规模式**

**newtio.c\_lflag=ICANON;**

1. **清除所有列队在串口的输入输出**

**tcflush(fd,TCIFLUSH);**

1. **设置当前的串口为newtio**

**tcsetattr(fd,TCSANOW,&newtio);**

1. **向串口写入数据，储存在缓存中。**

**write(fd,s1,1);**

1. **关闭串口。**

**close(fd);**

1. **恢复旧的端口参数**

**tcsetattr(fd,TCSANOW,&oldtio);/\*恢复旧的端口参数\*/**

**例5.2：通过计算机的COM1和COM2进行通信，利用RS-232来传送信息，其中COM1为发射端，COM2为接收端，当接收端接收到字符‘@’时，结束传输。RS-232的通信格式为38400,n,8,1（38400表示波特率大小，n表示不进行奇偶校验，8表示数据位，1表示停止位）。**

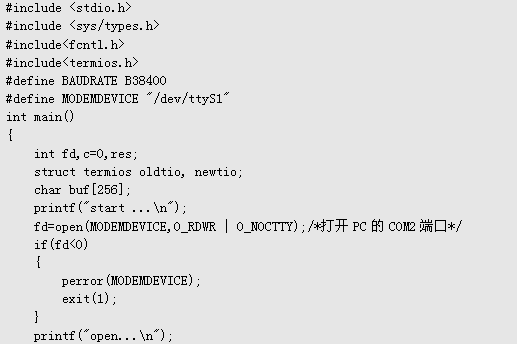
**步骤 1:连线**

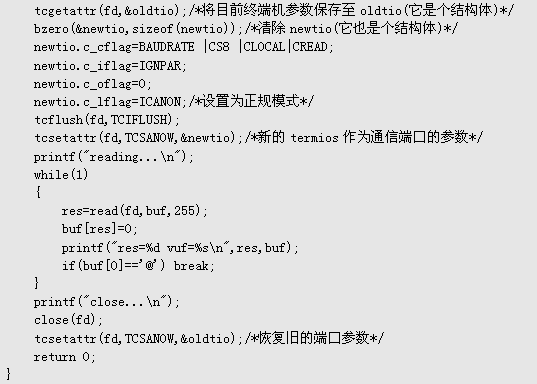
**计算机的COM1和COM2，通过RS-232线连接。**

****

**步骤 2: 编辑源程序代码**

**设接收端的源文件名为：5-2-r.c，发送端的源文件名为：5-2-s.c ，在接收端打开端口com2后，com2口会读取计算机com1口传来的数据，并输出。若com2口接收到的字符为‘@’，则结束传输。**

****

****

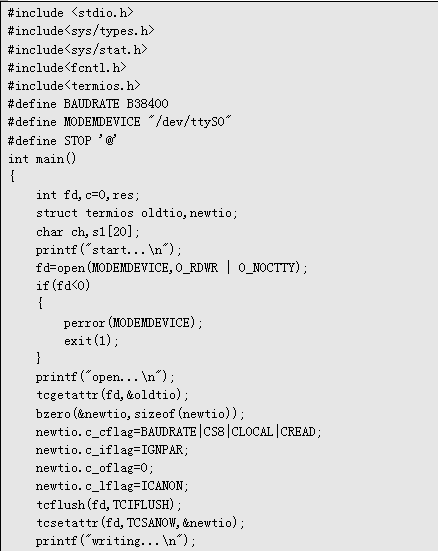
**步骤 3:用gcc编译程序**

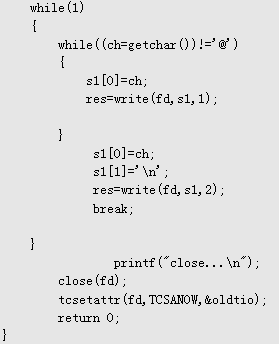
**[root@localhost root]#gcc 5-2-r.c –o 5-2-r**

**步骤 4:编辑发送端源程序代码**

**发送端com1，它会把com1的数据发送给com2。若com2接收的字符为‘@’，则结束传输**

**[root@localhost root]#vim 5-2-s.c**

****

****

**步骤 5:用gcc编译程序**

**[root@localhost root]#gcc 5-2-s.c –o 5-2-s**

**步骤 6: 测试运行结果**

**1. 打开一个终端，运行发送端程序：**

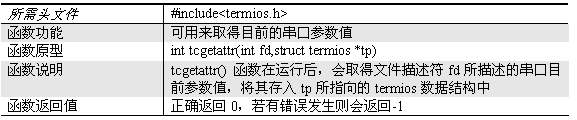
**[root@localhost root]#./ 5-2-r2.**

**2. 打开另一个终端，运行发送端程序，并输入“hello,lupa！”**

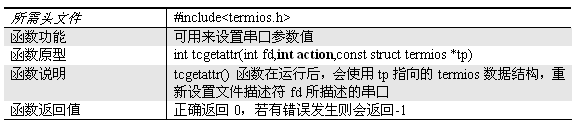
**3. 接着会在接受端看到传来的数据。**

**结果分析：接受端收到发送端传来的字符（hello,lupa!），并统计出了字符数。**

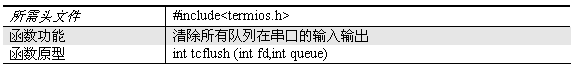
* **tcgetattr函数说明：**

****

* **tcsetattr函数说明：**

****

* **tcflush函数说明：**

****

**思考题**

1. **把例5.2中的RS-232通信格式改为51200,n,8,1，其他设置不变，然后完成次实验。**
2. **参考例5.2，RS-232的通信格式不变，依然是38400,n,8,1，要求发送端先读取文件的内容，然后将其内容发送到接收端，并在屏幕上打印出接受到的内容。**

**思考与实验**

1. **open("/dev/ttyS0",O\_RDWR|O\_NOCTTY|O\_NDELAY);写出这行代码的含义。**
2. **设置串行口属性参数为：波特率38400并且启用偶校验位。**
3. **编写一个串口通信程序，要求使用硬件流控制，8位字符大小，以9600的波特率从一台计算机的com1口发送键盘输入的字符，从另一计算机的com1接受，并在屏幕上打印出接受到的字符。**