

第4章 串行硬件的设置

现在还有那么一些人，他们自己拥有一台计算机，不把钱花在 T1 因特网链路上。日常新闻和电子邮件收发只依赖于采用公用电话网络的 SLIP 链路、UUCP 网络和电子公告板系统 (BBS)。

本章旨在帮助那些依靠 modem 接入因特网的人们维护他们的链路。但是，本章不准备就某些问题进行详细讨论（比如，配置 modem 以供拨号之用）。所有这些主题的每一个细节都可在 HOWTO 中找到，它位于 www.redhat.com/mirrors/LDP/HOWTO/Serial-HOWTO.html（由 David S. Lawyer (bf347@lanf.org) 编写。最初是由 Greg Hankins (greg@cc.gatech.edu) 编写)。

4.1 Modem 通信软件

用于 Linux 的通信软件不胜枚举。大部份都是终端软件，允许用户接驳另一台计算机。过去，用于 Linux 的终端软件叫作 kermi。现在，有种称呼是 Spartan。随着 IT 业的不断进步，各种终端程序层出不穷，而且越来越方便好用，它们可支持脚本语言呼叫和登录到远程计算机系统等等。其中之一是 Minicom，该软件与早期 DOS 用户惯用的有些终端软件很相似。另外，还有基于 X 的通信软件，比如 Seyon。与此同时，还有丰富的基于 Linux 的 BBS 通信软件供那些想运行电子公告板的用户选择。大家可在 sunsite.unc.edu/pub/Linux/system/Network 内找到其中一些通信软件。

除了终端软件之外，还有一些软件利用一条专用的串行线路实现数据的非交互式传输。这种技术的好处是自动将邮件下载回本机，而不是在线浏览，从而节省时间。另一方面，它要求有较多的磁盘空间。

概括地说，这类通信软件就叫作 UUCP。它是一个通信套件，把一台主机上的文件复制到另一台主机，执行远程主机上的程序等等。它常用于私人网络内的邮件及新闻传输。Ian Taylor 编写的 UUCP 套件也可在 Linux 下运行，详情参见下一章。其他非交互式通信软件则用于 Fidonet。诸如 ifmail 之类的 Ficonet 应用程序端口也是可以用的。

SLIP (串行线路 Internet 协议) 介于交互式和非交互式传输之间。许多人用 SLIP 拨号接入校园网或其他类型的公用 SLIP 服务器，运行 FTP 会话等。但是，SLIP 也可用于永久性地或半永久性的连接上，供局域网到局域网的联结所用，但只限于 ISDN。

4.2 串行设备概述

内核为访问串行设备所提供的设备一般称为 tty。其完整称呼是 teletype，它一度是早期 Unix 时代的主要终端产品。今天，这个术语代表的是任何一种基于字符的数据终端。本章中，我们用它来代表内核设备。

Linux 将 tty 分为三类：(虚拟) 控制台、伪终端 (类似于一个双向管道，供 X11 之类的应用程序使用) 和串行设备。后者也被算做 tty，因为它们允许终端或远程计算机通过串行链路实

现交互式会话。

tty的配置参数较多，可利用 `ioctl(2)` 系统方法调用来设定这些参数。多数参数都只适用于串行设备，因为它们需要更大的灵活性来处理不同类型的连接。

最突出的线路参数是 `line speed`（线路速率）和 `parity`（奇偶校验）。另外还有一些标记用于大小写转换、回车到链接速率的转换。tty驱动程序还支持不同的 `line disciplines`（线路规则，即线路协议），线路规则会令设备驱动程序的行为截然不同。例如，SLIP驱动程序就是通过一条特殊的线路规则来实施的。

应该怎样来描述线路速率呢？最恰当的术语是“位速率”（bit rate）。数据通信中，通过通信线路传输二进制位的速度。通常用每秒位数或比特/秒表示（bps）。有时，也会听到有人称之为“波特率”（baud rate），其实这种称呼是很不恰当的。两个术语不能互换。波特率指的是某个串行设备的物理特性，即时钟频率，是每秒钟传送的信息位数量。而位速率，代表的是两个通信点之间的现有串行连接当前所处的状态，即每秒钟内传送的平均信息位数量。记住这两个值不同是很重要的，因为多数设备在每个电子脉冲处理的位数不止一位。

4.3 访问串行设备

像系统内所有的设备一样，对串行端口的访问是通过特定的设备文件（位于 `/dev` 目录下）来完成的。有两类与串行驱动程序相关的设备文件，每个端口都有自己的设备文件。采用的文件不同，设备的行为也会有所不同。

第一类用于端口拨号时；它有一个主要的编号4，其文件名分别为 `ttyS0`、`ttyS1` 等等。第二类用于通过端口拨出时；其文件名为 `cua0`、`cua1` 等等，其主要编号是5（Linux设备有一个主编号和副编号。在做一个很长的目录清单（`ls -l`）时，也需列出设备编号。我们将在清单 4-1 中向大家展示一个示例。主编号是5，副编号在64到67之间）。

两种类型的副编号完全相同。如果你的 modem 在端口 COM1 到 COM4 之间的其中一个端口上，其副编号就会是 COM 端口号再加上 63。如果你的设置与此不符，比如使用的是一张支持多串行线路的网卡，就有必要查看 Serial HOWTO 来了解详情。

现在，我们假设你的 modem 位于 COM2 端口上。因此，其副编号是 65，主编号是 5，用于拨出。另外还应该有一个拥有这些编号的设备 `cua1`。然后，列出 `/dev` 目录中所有的 tty。第5和第6列将分别展示主编号和副编号，如清单 4-1 所示。

清单 4-1 设备 `cua1` 中的主编号和副编号

```
$ ls -l /dev/cua*
crw-rw-rw- 1 root root 5, 64 Nov 30 19:31 /dev/cua0
crw-rw-rw- 1 root root 5, 65 Nov 30 22:08 /dev/cua1
crw-rw-rw- 1 root root 5, 66 Oct 28 11:56 /dev/cua2
crw-rw-rw- 1 root root 5, 67 Mar 19 1992 /dev/cua3
```

如果没有此类的设备，必须建立一个：become super-user and type

```
# mknod -m 666 /dev/cua1 c 5 65
# chown root.root /dev/cua1
```

有人建议令 `/dev/modem` 作为一个象征性的链接，链接到自己的 modem 设备，以便临时性的用户无须去记住颇费脑筋的 `cua1`。但是，没有人愿意在这个程序使用 modem，又在另一个程序内去取真正的设备文件名。正因为此，这类程序通常采用一个所谓的“锁文件”（lock

files) 来表示设备正在使用中。按照惯例, cua1的锁文件名一般是LCK..cua1。针对同一个端口, 使用不同的设备文件将意味着程序不能识别彼此的锁文件, 因而两者都会同时采用这个设备。其结果是两个应用程序根本不能运行。

4.4 串行硬件

目前, Linux对采用RS-232标准的串行卡提供了广泛的支持。RS-232是当前PC领域内最常用的串行通信标准。它利用大量回路来同步传送单一的信息位。新增的线路将用于标示载波(供modem所用)和握手的存在。

尽管硬件握手是可选的, 但它非常有用。它允许通信的任何一方标示是否已准备接收数据, 或另一方是否应该在接收方处理完接收到的数据之后, 再继续发送数据。用于这些用途的线路分别称作“清除发送”(Clear to Send, CTS)和“准备发送”(Ready to Send, RTS), 它们代表硬件握手名, 即RTS/CTS。

在PC领域内, RS-232接口通常是一颗UART芯片(源于国家半导体16450芯片)或新版本的NSC 16550A(以前, 有过NSC 16550芯片, 但其FIFO一直未能起过任何作用)。其他牌子(全球最引人注意的是装备Rockwell芯片组的modem)也采用了完全不同的另类芯片, 可以模拟16550芯片。

16450和16550芯片之间的主要区别是后者有16个字节的FIFO缓冲, 而前者只有1字节的缓冲。这样一来, 16450适用于9600波特以下的数据传送速率, 而高速率则要求16550芯片。除了前面提到的芯片外, Linux还支持8250芯片, 该芯片是最初的UART(通用异步收发机), 用于PC-AT。

默认配置方案中, 内核会对COM1到COM4之间的四个标准串行端口进行查看。就像前面描述的那样, 这些端口将分配到64到67之间的设备副编号。

如果打算正确配置自己的串行端口, 你应该随rc.serial脚本一起, 安装Ted Tso的setserial命令。应该在系统启动时间, 从/etc/rc调用这个脚本。典型的rc.serial脚本像下面这样:

```
# /etc/rc.serial - serial line configuration script.
#
# Do wild interrupt detection
/sbin/setserial -W /dev/cua*
# Configure serial devices
/sbin/setserial /dev/cua0 auto irq skip test autoconfig
/sbin/setserial /dev/cua1 auto irq skip test autoconfig
/sbin/setserial /dev/cua2 auto irq skip test autoconfig
/sbin/setserial /dev/cua3 auto irq skip test autoconfig

# Display serial device configuration
/sbin/setserial -bg /dev/cua*
```

关于setserial命令的参数说明, 请参考相应文档。

如果内核没有侦测到你的串行卡, 或setserial-bg命令显示出一个错误的设置, 你必须清楚提供正确的参数值, 实施整个配置。对使用装备Rockwell芯片组的内置modem的用户, 经常会碰到这个问题。比如, 将UART芯片认作是NSC 16450芯片, 而事实上, 它是兼容于NSC 16550的芯片, 面对这种情况, 你只有将配置命令改为

```
/sbin/setserial /dev/cua1 auto irq skip test autoconfig
uart 16550
```

类似的情况也发生在COM端口、基础地址和IRQ设置上。详情参阅 `setserial(8)` 手册。

如果你的modem支持硬件握手，就应该确定启用这一特性。奇怪的是，默认情况下，多数通信软件都不会启用它；所以你必须自行手动设定。这一过程最好利用 `stty` 命令，在 `rc.serial` 脚本内进行：

```
$ stty crtscts < /dev/cua1
```

要想查看硬件握手是否开始起作用，则采用

```
$ stty -a < dev/cua1
```

这样，就能看到该设备的所谓状态标记了；前面带有负号 - 的标记（比如 `- crtscts`），表示该标记已关闭。