

## 第 15 章 串口通讯

(翻译中出现的任何问题或错误，请广大读者及时反馈给我：freebsdhandbook@163.com)

### 15.1 概要

UNIX 都是支持串口通讯的。事实上，早期的 UNIX 系统就是利用串行线来输入和输出数据的。那时通常的“terminal”包含了一个每秒 10 个字符的串行打印机和键盘时，现在这些事情已经有所变化了。本章将介绍一些利用 FreeBSD 进行串行通讯的方法。

读完本章，你将了解到：

- 如何通过终端连接到 FreeBSD 系统。
- 如何使用 modem 拨号到远程域。
- 如何允许远程用户通过 modem 登陆到你的系统。
- 如何从串行控制台引导你的系统。

阅读本章之前，你应当了解：

- 如何配置和安装一个新的内核。（第 9 章）
- 理解 unix 的权限和进程。（第 3 章）
- 通过网络设备访问技术参考手册。

### 15.2 介绍

#### 15.2.1 术语

bps

每秒比特---数据的传输速率

DTE

数据终端设备---如你的电脑

DCE

数据通讯设备---如你的 modem

RS-232

用于硬件串行通讯的 EIA 标准

## 15.2.2 线缆和端口

要从你的 FreeBSD 系统连接到一个 modem 或终端,你需要有一个串行端口,和连接到你串行设备的适当的线缆。如果你比较熟悉硬件和线缆,你可以跳过这节。

### 15.2.2.1 线缆

有好几种线缆。两个最普通的类型是 null-modem 线缆和标准 RS-232 线缆。你的硬件的规格说明会有详细描述。

#### 15.2.2.1.1 Null-modem 线缆

一个 null-modem 线缆会直接通过象 signal ground 这样的信号。如果你想自己选择线缆,你可以做一个使用 null-modem 的线缆。这个线缆在一个 DB-25 连接器上会显示 RS-232C 信号名和 pin 号码。

Signal	Pin #		Pin #	Signal
TxD	2	连接到	3	RxD
RxD	3	连接到	2	TxD
DTR	20	连接到	6	DSR
DSR	6	连接到	20	DTR
SG	7	连接到	7	SG
DCD	8	连接到	4	RTS
RTS	4		5	CTS
CTS	5	连接到	8	DCD

#### 15.2.2.1.2 标准 RS-232C 线缆

一个标准的串行线缆会直接通过所有的 RS-232C 信号。这是连接一个 modem 到你的 FreeBSD 系统的线缆类型,线缆的类型需要针对一些终端。

### 15.2.2.2 端口

串行端口是 FreeBSD 主机与终端传输数据的设备。这节描述了端口的种类和他们在 FreeBSD 上的地址。

#### 15.2.2.2.1 几种端口

有好几种端口存在。你在购买和做线缆之前，你需要确定它是否适合你机器的接线端。

绝大多数的接线端有 DB25 端口。个人电脑包括运行 FreeBSD 的 PC 机，有 DB25 或 DB9 端口。如果你有一个多端口的串行卡，你可以使用 RJ-12 或 RJ-45 端口。请仔细看看硬件的说明。

#### 15.2.2.2.2 端口名称

在 FreeBSD 中，你可以通过/dev 目录中的一个记录来访问每个串行端口。有两种不同的记录：

- 呼入端口被命名为/dev/ttydN，这儿 N 是端口号，从零开始。通常，你使用呼入端口作为接线端。呼入端口要求数据线使用载波检测信号来工作。
- 呼出端口被命名为/dev/cuaaN。你通常不使用呼出端口作为接线端，只使用 modem。如果串行线或接线端不支持载波检测数据传输，你可以使用呼出端口。

如果你已经连接到了第一个串行端口，那你需要使用/dev/ttyd0 来应用接线端。如果它是在第二个串行口，那就是/dev/ttyd1，等等。

### 15.2.3 内核配置

FreeBSD 默认支持 4 个串行端口。在 MS-DOS 下，这些是 COM1:，COM2:，COM3: 和 COM4:。FreeBSD 当前支持 dumb 多端口串行接口卡，如 BocaBoard 1008 and 2016，就象许多 Digi board 和 Stallion Technologies 制造的智能多接口卡。默认的内核只会寻找标准的 COM 接口。

要看看你的内核是否支持你的串口，需要在内核启动时查看一些信息，或使用 /sbin/dmesg 命令重新播出内核启动信息。特别的，寻找以 sio 为特征的启动信息。

**提示：**要看看带有 sio 的信息，可以使用下面的命令：

```
# /sbin/dmesg | grep 'sio'
```

例如，在一个带有 4 个串行口的系统上，这些是串行口特定的内核启动信息：

```
sio0 at 0x3f8-0x3ff irq 4 on isa
```

```
sio0: type 16550A
```

```
sio1 at 0x2f8-0x2ff irq 3 on isa
```

```
sio1: type 16550A
```

```
sio2 at 0x3e8-0x3ef irq 5 on isa
```

```
sio2: type 16550A
```

```
sio3 at 0x2e8-0x2ef irq 9 on isa
```

```
sio3: type 16550A
```

如果你的内核没有认出你的所有串行口，你需要为你的系统定制一个内核。更多有关配置内核的细节，可以看看第 9 章。

在你的内核配置文件中相关的设备行是这样的：

```
device    sio0    at isa? port "IO_COM1" tty irq 4 vector siointr
```

```
device    sio1    at isa? port "IO_COM2" tty irq 3 vector siointr
```

```
device    sio2    at isa? port "IO_COM3" tty irq 5 vector siointr
```

```
device    sio3    at isa? port "IO_COM4" tty irq 9 vector siointr
```

你可以注释掉或完全删除你没有的设备。请看看 `sio` 的有关如何为你的多端口主板写入配置信息的联机手册。如果你使用了一个不同版本的 FreeBSD 的配置文件，请小心点，因为设备的标记在两个版本之间已经改变了。

### 15.2.4 设备指定文件

内核中的绝大多数设备可以通过 `device special files` 来访问，它就在 `/dev` 目录下。`Sio` 设备通过 `/dev/ttydN` (`dial-in`) 和 `/dev/cuaaN` (`call-out`) 来访问。FreeBSD 也提供了初始化的设备 (`/dev/ttyidN` and `/dev/cuai0N`) 和锁定的设备 (`/dev/ttyldN` and

/dev/cua10M)。初始化的设备在每次端口被打开时被用来初始化设备通讯端口参数。例如使用 CTS/RTS 的流控制信号的 `crtscts`。锁定设备被用来锁定端口的标记以阻止用户或程序改变某个参数；看看 `termios`, `sio`, and `stty` 的联机手册了解更多信息。

### 15.2.4.1 编译设备指定文件

**注意：**FreeBSD 5.0 已把自动创建设备接点的 `devfs` 文件系统作为是必需的。如果你在运行启用 `devfs` 的 FreeBSD 版本，你就可以跳过这一节。

一个在 `/dev` 目录下的叫做 `MAKEDEV` 的 shell 脚本管理着设备的特殊文件。要使用 `MAKEDEV` 来编译拨号设备，以使用 `COM1`：(port 0)，`cd` 进入 `/dev`，然后执行命令 `MAKEDEV ttyd0`。同样地，要编译拨号设备以使用 `COM2`：(port 1)，可以执行 `MAKEDEV ttyd1`。`MAKEDEV` 不仅仅创建 `/dev/ttydN` 设备特殊文件，也创建 `/dev/cuaaN`，`/dev/cuai aN`，`/dev/cual aN`，`/dev/ttyldN`，和 `/dev/ttyidN` 节点。

编译完支持新设备的特殊文件之后，需要检查文件的权限来确定谁可以在这些文件上读写—你可能不想让普通的用户来使用你的 modem 吧。默认的在 `/dev/cua*` 上的权限应当比较充分：

```
crw-rw----  1 uucp  dialer  28, 129 Feb 15 14:38 /dev/cuaa1
crw-rw----  1 uucp  dialer  28, 161 Feb 15 14:38 /dev/cuai a1
crw-rw----  1 uucp  dialer  28, 193 Feb 15 14:38 /dev/cual a1
```

这些许可允许用户 `uucp` 和在组拨号上的用户使用呼出设备。

### 15.2.5 串行端口配置

`ttydN` (或 `cuaaN`) 设备是你要打开你的应用程序的规则设备。当一个进程打开某个设备，它将有一个终端 I/O 的默认设置。你可以在命令行看这些设置：

```
# stty -a -f /dev/ttyd1
```

当你修改了这个设备的设置，这个设置会生效，除非设备被关闭。当它被重新打开时，它将回到默认设置。要修改默认设置，你可以打开和调整 `initial state` 设备的设置。例如，要打开 `CLOCAL` 模式，8 位通讯，默认的 `XON/XOFF` 流控制，键入：

```
# stty -f /dev/ttyid5 clocal cs8 ixon ixoff
```

系统的串行设备的初试化可以在/etc/rc.serial 中被控制。这个文件影响了串行设备的默认设置。

要防止某个设置被一个应用程序所修改,需要调整 lock state 设备。例如,要锁定 ttyd5 的速度为 57600 bps,键入:

```
# stty -f /dev/ttyid5 57600
```

现在,一个打开 ttyd5 和设法改变端口速度的应用程序将被固定在 57600 bps。很自然地,你需要确定初始情形,和锁定设备的写入状态,只有 root 才能写。

### 15.3 终端

当你不在电脑控制台或不在一个连接的网络上时,终端提供了一个方便和低成本地访问你的 FreeBSD 系统的方法。本节描述了如何在 FreeBSD 上使用终端。

#### 15.3.1 终端的用法和类型

早期的 UNIX 系统没有控制台。人们通过将终端连接到电脑的串行口来登陆和使用程序。它很象用一个 modem 和一些终端软件来拨号进入一个远程的系统只执行文本的工作。

今天的 PC 已经可以使用高质量的图形了,但与今天的其他 UNIX 操作系统一样,建立一个登陆的能力仍然存在;通过使用一个终端连接到一个没有使用的串行口,你就能登陆和运行任何文本程序或在 X Window System 中运行一个 xterm 窗口程序。

对于商业用户,你可以配上任何终端到 FreeBSD 系统,然后把它们放在员工的桌面上。对于一个家庭用户,可以使用一台比较老的 IBM PC 或 Macintosh 运行一个终端连接到一台运行 FreeBSD 的高性能机器上。

对于 FreeBSD,有三种终端:

- Dumb 终端
- 充当终端的 PCs
- X 终端

下面的小节将描述每一种。

### 15.3.1.1 Dumb 终端

Dumb 终端可以适应好几种硬件，让你通过串行线连接到电脑。他们被叫做 dumb 是因为他们只能够用来显示、发送和接受文本。你不能在它上面运行任何程序。

有好几百种 dumb 终端被制造，包括 Digital Equipment Corporation's VT-100 和 Wyse's WY-75。只有几种可以在 FreeBSD 上工作。一些高端的终端可以显示图形，但只有某些软件包可以使用这些高级特性。

### 15.3.1.2 充当终端的 PCs

如果一个 dumb 终端有足够的力量来显示、发送和接受文本，那这些个人电脑可以作为一个 dumb 终端。你所需要的只是适当的线缆和一些终端模拟软件。

这样一个配置被广泛运用于家庭。例如，如果你的妻子忙于在你的 FreeBSD 系统控制台上工作，你也可以从一台低档的个人电脑登陆到 FreeBSD 系统执行一些文本的工作。

### 15.3.1.3 X 终端

X 终端是最复杂的终端系统。它们通常需要使用以太网来连接。它们能显示任何 X 应用程序。我们介绍 X 终端只是为了感兴趣。然而，本章不会涉及 X 终端的安装、配置或使用。

## 15.3.2 配置

本节描述了你在一个终端上启用一个登陆活动，需要你在 FreeBSD 系统上配置些什么。假设你已经配置好了你的内核来支持串行端口，你就可以连接了。

回想起第 7 章，init 进程依赖于系统启动时所有的处理控制和初始化。通过 init 来执行的一些任务将先读取/etc/ttys 文件，然后在可用的终端上启用一个 getty 进程。getty 进程可用来阅读一个登陆名和启动登陆程序。

然而，要配置你 FreeBSD 系统的终端，你需要以 root 来执行下面的步骤：

1. 如果它不在那儿，你需要为串行端口在/dev 目录下添加一行记录到/etc/ttys。
2. 指定/usr/libexec/getty 在端口上运行，然后从/etc/gettytab 文件指定适当的 *getty* 类型。

3. 指定默认的终端类型。
4. 设置端口为 on。
5. 确定端口是否为 secure。
6. 迫使 `init` 重新读取 `/etc/ttys` 文件。

作为可选的步骤，你可以通过在 `/etc/gettytab` 中建立一个记录，在第 2 步创建一个定制的 *getty* 类型来使用。本章不会介绍如何做；你可以看看 [gettytab](#) 和 [getty](#) 的联机手册了解更多信息。

### 15.3.2.1 添加一个记录到 `/etc/ttys`

`/etc/ttys` 文件列出了你 FreeBSD 系统上允许登陆的所有端口。例如，第一个虚拟控制台 `ttyv0` 在这个文件中有一个记录。你可以使用这个记录登陆进控制台。这个文件也包含其他虚拟控制台的记录、串行口、和伪-`ttys`。对于一个硬连线的终端，只要列出串行端口的 `/dev` 记录。

默认的 FreeBSD 安装包括了支持最初四个串行口 `ttyd0` 到 `ttyd3` 的 `/etc/ttys` 文件。如果你从那些端口来使用终端，你不需要添加另一个记录。

#### 例 15-1. 添加终端记录到 `/etc/ttys`

建议我们连接两个终端给系统：一个 Wyse-50 和一个老的运行 Procomm 终端软件模拟一个 VT-100 终端的 286 IBM PC。在 `/etc/ttys` 文件中的相应的记录是这样的：

```
ttyd1 ① "/usr/libexec/getty std.38400"② wy50③ on④ insecure⑤
```

```
ttyd5  "/usr/libexec/getty std.19200" vt100 on insecure
```

①，第一部分指定了终端特殊文件的名称，它可以在 `/dev` 中找到。

②，第二部分是在这行执行的命令，通常是 `getty`。Getty 初始化然后打开一行，设置速度，用户名的命令和执行登陆程序。

`getty` 程序在它的命令行接收一个参数，*getty* 类型。一个 *getty* 类型会在终端行描述一个特征，象 `bps` `rate` 和 `parity`。`getty` 程序从 `/etc/gettytab` 文件读取这些字符。文件 `/etc/gettytab` 包含了许多老的和新的终端线的记录。在很多例子中，启动文本 `std` 的记录

将用硬连线终端来工作。这些记录忽略了奇偶性。这是一个从 110 到 115200 的每 bps 的 std 记录。当然，你可以添加你自己的记录到这个文件。Gettytab 的联机手册提供了更多的信息。

当在 `/etc/ttys` 中设置 *getty* 类型的时候，确信在终端上的通讯设置匹配。

在我们的例子中，Wyse-50 不使用奇偶性，用 38400 bps 来连接。286 PC 不使用奇偶性，用 19200bps 来连接。

③，第三部分是通常连接到那个 tty 线的终端类型。对于拨号端口，不知名的或拨出的通常被用在这个地方。对于硬连线的终端，终端类型不会改变，所以你可以从 `termcap` 数据库文件中放置一个真正的终端类型。

在我们的例子中，Wyse-50 使用真正的终端类型，而运行 `Procomm` 的 286 PC 将被设置成在 VT-100 上的模拟。

④，如果端口被启用，可以指定第四个部分。在第二部分，把它放在这儿将执行初始化进程来启动程序 `getty`。如果你在这部分推迟，将没有 `getty`，在端口上因此就没有登陆。

⑤ 最后部分被用来指定端口是否安全。标记一个安全的端口意味着你信任它允许用 `root` 帐户从哪个端口登陆。不安全的端口不允许 `root` 登陆。在一个不安全的端口上，用户必须用无特权的帐户登陆，然后使用 `su` 或一个相似的机制来获得超级用户的权限。

### 15.3.2.2 重新读取 `/etc/ttys` 来强制初始化

对 `/etc/ttys` 文件做一个必要的修改后，你必须发送一个 `SIGHUP` 信号给初始化进程来迫使它重新读取它的配置文件，例如：

```
# kill -HUP 1
```

如果能够被正确设置，所有的线缆都是适当的，终端将可以启用了，然后一个 `getty` 进程将在每个终端被运行，你将在你的终端上看到登陆命令行。

### 15.3.3 你的连接可能出现的问题

即使你小心翼翼地注意细节，你仍然可能会在设置终端时出错。这儿有一个有关问题的现象和解决办法的列表：

### 1, 没有登陆命令出现：

确定终端被嵌入和打开了。如果把一台个人电脑充当一个终端，确信终端模拟软件运行在正确的串口上。

确信线缆被稳固地连接在终端和 FreeBSD 电脑上。确信用了正确的电缆。

确定终端和 FreeBSD 的传输速率和奇偶设置已经一致了。如果你有一个图象显示终端，确信对比度已经调节好了。如果它是一个可打印的终端，确信纸张和墨水已经准备好了。

确定一个 getty 进程正在运行和服务终端。例如，可以用 ps 命令来得到运行 getty 程序的列表，键入：

```
# ps -axww|grep getty
```

你将看到一个终端的记录。例如，下面的显示表明一个 getty 正在第二个串行端口 ttyd1 运行，正在/etc/gettytab 中使用 std. 38400 的记录：

```
22189 d1 ls+ 0:00.03 /usr/libexec/getty std.38400 ttyd1
```

如果没有 getty 进程运行，确信你已经在/etc/ttys 中启用了端口。在修改完 ttys 文件后，记得运行 kill -HUP 1。

### 2, 出现一个“垃圾”而不是一个登陆命令行

确信终端和 FreeBSD 使用相同的 bps 传输率和奇偶校验设置。检查一下 getty 进程确信当前使用的正确的 getty 类型。如果没有，编辑/etc/ttys 然后运行 kill -HUP 1。

### 3, 当键入密码时，字符两个两个出现

将终端（或终端模拟软件）从“half duplex”或“local echo”换成“full duplex”。

## 15.4 拨入服务

配置 FreeBSD 系统来用拨入服务与连接到终端是非常相似的，除非你正在使用 modem 来拨号。

### 15.4.1 外置和内置 modem

外置 modem 看起来很容易拨号。因为，外置 modem 可以通过储存在非易失性的 RAM 中的

参数来配置，他们通常提供指示器来显示重要的 RS-232 信号的状态。不停闪光的信号灯能给用户留下比较深刻的印象，而且指示器也可以用来查看 modem 是否正常地工作。

内置 modem 通常缺乏非易失性的 RAM，所以对它们的配置可能被限制在通过 DIP 开关来设置。如果你的内置 modem 有指示灯，你也很难看得到。

### 15.4.1.1 Modems 和线缆

如果你使用一个外置的 modem，那你将需要适当的电缆线。一个标准的串行线应当足够长以至普通的信号能够连接上：

- Transmitted Data (SD)
- Received Data (RD)
- Request to Send (RTS)
- Clear to Send (CTS)
- Data Set Ready (DSR)
- Data Terminal Ready (DTR)
- Carrier Detect (CD)
- Signal Ground (SG)

FreeBSD 需要对速度超过 2400bps 的 RTS 和 CTS 信号进行流控制，当一个呼叫被回复或线路被挂起的时候，CD 信号就会被侦测到，一个任务完成之后，DTR 信号就会刷新 modem。一些线缆不需要任何信号就可以连接，所以如果你有问题，当线路被挂起时，一个登陆任务就会丢失，你可能会在线缆上有问题。

象其它 unix 类的操作系统一样，FreeBSD 使用硬件信号来寻找出一个呼叫什么时候会回复或一个线路会被挂起。FreeBSD 避免发送命令给 modem 或监视 modem 的状况。如果你熟悉连接 modem 到 BBS，这可能是很难的。

### 15.4.2 串行接口的考虑

FreeBSD 支持以 NS8250-，NS16450-，NS16550-和 NS16550A 为基础的 EIA RS-232C 通讯接口。8250 和 16450 设备有单字符缓冲。16550 设备提供了一个 16 个字符的缓冲，可以提

高更多的系统性能。因为单字符缓冲设备比 16 个字符的缓冲需要更多的系统资源来工作，所以基于 16550A 的接口卡可能更好。如果系统没有活动的串行口，或有一个巨大的负载，16 字符缓冲的卡对于低错误率的通讯来说更好。

### 15.4.3 快速预览

对于终端，`init` 会在每个配置串行口上为每个拨入连接产生一个 `getty` 进程。例如，如果一个 modem 被附带在 `/dev/ttyd0` 中，用命令 `ps ax` 可以显示下面这些：

```
4850 ?? |      0:00.09 /usr/libexec/getty V19200 ttyd0
```

当一个用户拨上 modem，并使用它进行连接时，CD 线就会被 modem 认出。内核注意到载波信号已经被检测到，需要完成 `getty` 的端口的打开。`Getty` 发送一个登陆：在指定的初始线速度上的命令行。`Getty` 会检查合法的字符是否被接收，在一个典型的配置中，如果发现垃圾，`getty` 就会设法调节线速度，直到它接收到合理的字符。

用户在键入他/她的登陆名称后，`getty` 执行 `/usr/bin/login`，这会要求用户输入密码来完成登陆，然后启动用户的 shell。

### 15.4.4 配置文件

在 `/etc` 目录中，有三个你将需要编辑的系统配置文件，来允许拨号访问到你的 FreeBSD 系统。第一，`/etc/gettytab` 包含了针对 `/usr/libexec/getty` 守护程序的配置信息。第二，保存信息来告诉 `/sbin/init` 什么 tty 设备将有运行在他们系统上的 `getty` 进程。最后，你可以把端口初始化命令放在 `/etc/rc.serial` 脚本中。

在 `unix` 上，关于拨号 modem 的想法主要有两种。一种是把本地接口配置成一个固定速率，以至一个远程用户拨号进入时都保持一个固定速率。这样配置的好处是远程用户总是可以立即看到一个系统的登陆界面。这种下降趋势是系统不知道一个用户真实的数据速率是多少，所以象 `eamcs` 全屏程序将不会调接屏幕刷新来确保对比较慢的连接有比较好的回应。

其他的配置 RS-232 modem 的方法是随着远程用户连接的速度的变化而变化。例如，连接到 modem 的 V.32bis (14.4 Kbps) 连接可以使 modem 在 19.2 Kbps 上运行 RS-232 接口，而连接使得 RS-232 接口运行在 2400 bps 上。

因为 `getty` 不了解任何特殊的 modem 的连接速度报告，`getty` 会给出一个登陆：在一个初始速度和检测字符的信息会作出回应。如果用户看到垃圾，假定他们知道他们键入了 `enter` 键，知道他看到了一个熟悉的命令行界面。

如果数据速率不相匹配，`getty` 会把用户键入的任何东西都看作“junk”，设法回到下面的速度，然后给出登陆：命令行界面。很明显，这个登陆顺序看起来不如 `locked-speed` 的方法，但一个连接在低速率上的用户将更好地交互接收来自全屏幕程序的回应。

### 15.4.4.1 /etc/gettytab

`/etc/gettytab` 是一个用来配置 `getty` 信息的 `termcap` 风格的文件。请看看 `gettytab` 的联机手册了解完整的文件格式和功能列表。

#### 15.4.4.1.1 锁定速率的配置

如果你把你的 modem 的数据通讯率锁定在一个特殊的速率上，你不需要对 `/etc/gettytab` 文件做任何变化。

#### 15.4.4.1.2 匹配速率的配置

你将需要在 `/etc/gettytab` 中设置一个记录来给出 `getty` 的你希望用到你的 modem 上的有关速度的信息。如果你有一个 2400 bps 的 modem，你可以使用已存在的 `D2400` 的记录。

```
#  
  
# Fast dialup terminals, 2400/1200/300 rotary (can start either way)  
  
#  
  
D2400|d2400|Fast-Dial-2400:\  
  
          :nx=D1200:tc=2400-baud:  
  
3|D1200|Fast-Dial-1200:\  
  
          :nx=D300:tc=1200-baud:  
  
5|D300|Fast-Dial-300:\  
  
          :nx=D2400:tc=300-baud:
```

如果你有一个更高速度的 modem，你必须在/etc/gettytab 中添加一个记录；这儿是一个你可以使用的一个最高 19.2 Kbps 的接口用在 14.4 Kbps 的 modem 上的记录：

```
#

# Additions for a V.32bis Modem

#

um|V300|High Speed Modem at 300,8-bit:\

      :nx=V19200:tc=std.300:

un|V1200|High Speed Modem at 1200,8-bit:\

      :nx=V300:tc=std.1200:

uo|V2400|High Speed Modem at 2400,8-bit:\

      :nx=V1200:tc=std.2400:

up|V9600|High Speed Modem at 9600,8-bit:\

      :nx=V2400:tc=std.9600:

uq|V19200|High Speed Modem at 19200,8-bit:\

      :nx=V9600:tc=std.19200:
```

上面使用 19.2 Kbps 的连接速度的例子，也可以使用 9600 bps (for V.32)，2400 bps，1200 bps，300 bps，直到 19.2 Kbps。通讯率的调节使用 nx= (“next table”)来实现。每条线使用一个 tc= (“table continuation”)的记录来加速对于一个特殊数据率的标准设置。

如果你有一个 28.8 Kbps 的 modem，或你想使用它的 14.4Kbps，你需要使用一个更高的超过 19.2 Kbps 的通讯速率的 modem。这是一个启动 57.6 Kbps 的 gettytab 记录的例子：

```
#

# Additions for a V.32bis or V.34 Modem
```

```
# Starting at 57.6 Kbps

#

vm|VH300|Very High Speed Modem at 300,8-bit:\

      :nx=VH57600:tc=std.300:

vn|VH1200|Very High Speed Modem at 1200,8-bit:\

      :nx=VH300:tc=std.1200:

vo|VH2400|Very High Speed Modem at 2400,8-bit:\

      :nx=VH1200:tc=std.2400:

vp|VH9600|Very High Speed Modem at 9600,8-bit:\

      :nx=VH2400:tc=std.9600:

vq|VH57600|Very High Speed Modem at 57600,8-bit:\

      :nx=VH9600:tc=std.57600:
```

如果你有一个低速的 CPU 或一个庞大负载的系统，你没有 16550A 的串行端口，你可能会在 57.6 Kbps 上得到 sio 错误。

#### 15.4.4.2 /etc/ttys

/etc/ttys 文件的配置在例 15-1 中介绍过。配置 modem 是相似的，但我们必须指定一个不同的终端类型。锁定速度和匹配速度配置的通用格式是：

```
ttyd0  "/usr/libexec/getty xxx"  dialup on
```

上面的第一条是这个记录的设备特定文件—ttyd0 意味着/dev/ttyd0 是这个 getty 将被监视的文件。第二条"/usr/libexec/getty xxx" 是将运行在设备上的处理 init。第三条，dialup，是默认的终端类型。第四个参数，on，指出了线路是可操作的 init。也可能会有第五个参数，secure，但它将只被用作拥有物理安全的终端(如系统终端)。

默认的终端类型可以依赖于本地参考。拨号是传统的默认的终端类型，以至用户可以定制他们的登陆脚本来注意终端什么时候拨号，和自动调节他们的终端类型。然而，作者发现它很容易在他的站点上指定 vt102 作为默认的终端类型，既然用户刚才在他们的远程系统上使用的是 VT102 模拟器。

你对/etc/ttys 作修改之后，你可以发送 `init` 进程给一个 HUP 信号来重读文件。你可以使用下面的命令来发送信号。：

```
# kill -HUP 1
```

如果这是你的第一次设置系统，你可能要在发信号 `init` 之前等一下，等到你的 modem 被正确地配置并连接好。

### 15.4.4.2.1 锁定速率的配置

对于一个锁定速率的配置，你的 `ttys` 记录必须有一个为 `getty` 提供固定速率的记录。对于一个速度被锁定在 19.2kbps 的 modem，`ttys` 记录是这样的：

```
ttyd0 "/usr/libexec/getty std.19200" dialup on
```

如果你的 modem 被锁定在一个不同的数据速率，为 `std.speed` 使用适当的速率来代替 `std.19200`。确信你使用了一个在/etc/gettytab 中列出的正确的类型。

### 15.4.4.2.2 匹配速度的配置

在一个匹配速率的配置中，你的 `ttys` 记录需要参考在/etc/gettytab 中适当的起始 `auto-baud` 记录。例如，如果你为一个以 19.2 Kbps 开始的匹配速度的 modem 添加上面建议的记录的话，你的 `ttys` 记录可能是这样的：

```
ttyd0 "/usr/libexec/getty V19200" dialup on
```

### 15.4.4.3 /etc/rc.serial

高速 modem，象 V.32, V.32bis, 和 V.34 modems，需要使用硬件(RTS/CTS)流控制。你可以在/etc/rc.serial 中添加 `stty` 命令来设置硬件流控制标记。

例如，在拨入和拨出初始设备的#1's (COM2:)串行端口上设置 `termios` 标记 `crtcscts`，下面这些行会被添加到/etc/rc.serial 中：

```
# Serial port initial configuration
```

```
stty -f /dev/ttyid1 crtscts
```

```
stty -f /dev/cuai01 crtscts
```

### 15.4.5 Modem 设置

如果你有一个 modem，它的参数能被存储在非易失性的 RAM 中，你将必须使用一个终端程序来设置参数。使用同样的通讯速率来连接 modem 作为初始速度 getty 将使用和配置 modem 的非易失性 RAM 来匹配这些要求：

- CD asserted when connected
- DTR asserted for operation; dropping DTR hangs up line and resets modem
- CTS transmitted data flow control
- Disable XON/XOFF flow control
- RTS received data flow control
- Quiet mode (no result codes)
- No command echo

请读读你的 modem 的文档找到你需要用什么命令和 DIP 接口设置。例如，要在一个 USRobotics Sportster 14,400 的外置 modem 上设置上面的参数，你可以用下面这些命令：

```
ATZ
```

```
AT&C1&D2&H1&I0&R2&W
```

你也可能想要在 modem 上寻找机会调节这个设置，例如它是否使用 V.42bis 和 MNP5 压缩。USR Sportster 14,400 外置 modem 也有一些用来设置的 DIP 开关，也许你可以使用这些设置作为一个例子：

- Switch 1: UP -- DTR Normal
- Switch 2: Do not care (Verbal Result Codes/Numeric Result Codes)
- Switch 3: UP -- Suppress Result Codes

- Switch 4: DOWN -- No echo, offline commands
- Switch 5: UP -- Auto Answer
- Switch 6: UP -- Carrier Detect Normal
- Switch 7: UP -- Load NVRAM Defaults
- Switch 8: Do not care (Smart Mode/Dumb Mode)

### 15.4.5.1 锁定速率的配置

对于一个锁定速率的配置，你需要配置 modem 来获得一个不依赖于通讯率的稳定的 modem-to-computer 的数据率。在一个 USR Sportster 14,400 外置 modem 上，这些命令将锁定 modem-to-computer 的数据率：

```
ATZ
```

```
AT&B1&W
```

### 15.4.5.2 匹配速率的配置

对于一个变速的配置，你需要配置你的 modem 来调节它的串行端口数据率来匹配接收的数据率。在一个 USR Sportster 14,400 的外置 modem 上，这些命令将锁定 modem 的错误修正数据率适合命令要求的速度，但允许串行端口速率适应 non-error-corrected 的连接：

```
ATZ
```

```
AT&B2&W
```

### 15.4.5.3 检查 modem 的配置

大多数高速的 modem 提供了用来查看当前操作参数的命令。在 USR Sportster 14,400 外置 modem 上，命令 AT15 显示了存储在非易失性 RAM 中的设置。要看看正确的 modem 操作参数，可以使用命令 ATZ 然后是 AT14。

如果你有一个不同牌子的 modem，检查 modem 的使用手册看看如何双重检查你的 modem 的配置参数。

### 15.4.6 问题解答

这儿是几个检查拨号 modem 的步骤。

### 15.4.6.1 检查 FreeBSD 系统

把你的 modem 连接到 FreeBSD 系统，启动系统，然后，如果你的 modem 有一个指示灯，当登陆时看看 modem 的 DTR 指示灯是否亮：会在系统控制台出现命令行---如果它亮，意味着 FreeBSD 已经在适当的通讯端口启动了一个 getty 进程，等待 modem 接受一个呼叫。

如果 DTR 指示灯不亮，通过控制台登陆到 FreeBSD 系统，然后执行一个 `ps ax` 来看 FreeBSD 是否正在正确的端口运行一个 getty 进程。你将在进程显示中看到象这样的一行：

```
114 ?? |      0:00.10 /usr/libexec/getty V19200 ttyd0
115 ?? |      0:00.10 /usr/libexec/getty V19200 ttyd1
```

如果你看到这样的：

```
114 d0 |      0:00.10 /usr/libexec/getty V19200 ttyd0
```

说明，modem 不接受呼叫，这意味着 getty 已经在通讯端口打开了。这可以指出线缆有问题或 modem 错误配置，因为 getty 不能打开通讯端口。

如果你没有看到任何 getty 进程等待打开渴望的 `ttydN` 端口，在 `/etc/ttys` 中双击你的记录看看那儿是否有错误。另外，检查日志文件 `/var/log/messages` 看看是否有一些来自 `init` 或 `getty` 的关于一些问题的日志信息。如果有任何信息，仔细检查配置文件 `/etc/ttys` 和 `/etc/gettytab`，还有适当的设备文件 `/dev/ttydN`，是否有错误，丢失记录，或丢失了设备指定文件。

### 15.4.6.2 尝试接入

设法拨入系统；确信使用 8 位，没有奇偶检验，在远程系统上的 1 阻止位。如果你不能立刻得到一个命令行，试试每隔一秒键入 `enter`。如果你仍没有看到一个登陆：设法发送一个 `BREAK`。如果你正使用一个高速的 modem 来拨号，请在锁定拨号 modem 的接口速度后再试试。

如果你不能得到一个登陆：`prompt`，再检查一下 `/etc/gettytab`，重复检查：

- 在 `/etc/ttys` 中指定的初始可用的名称与 `/etc/gettytab` 中的一个可用的相匹配。

- 每个 `nx=` 记录与另一个 `gettytab` 可用名称匹配。
- 每个 `tc=` 记录与另一个 `gettytab` 可用名称相匹配。

如果你拨号但 FreeBSD 系统上的 modem 没有回应，确信 modem 能回应电话。如果 modem 看起来配置正确了，通过检查 modem 的指示灯来确认 DTR 线连接正确。

如果你做了好几次，它仍然无法工作，打断一会，待会再试试。如果还不能工作，也许你应该发一封电子邮件给 <freebsd-questions@FreeBSD.org> 寻求帮助。

### 15.5 拨出设备

下面将让你的主机通过 modem 连接到另一台电脑上。这只要建立一个终端作为远程主机。这可以用来登陆进一个 BBS。

如果你用 PPP 有问题，那这种连接在 internet 上下载一个文件是非常有帮助的。如果你必须 FTP 一些东西，而 PPP 断了，使用终端连接到 FTP。然后使用 `zmodem` 来把它传输到你的机器上。

#### 15.5.1 我的 Stock Hayes Modem 不被支持, 我该怎么办?

事实上，联机手册对于这个的描述已经过期了。有一个通用的 Hayes 拨号已经建构在里面了。只要在你的 `/etc/remote` 文件中使用 `at=hayes`。

Hayes 驱动器不够“聪明”只能认出一些比较新的 modem 的高级特性---如 `BUSY`, `NO DIALTONE`, or `CONNECT 115200` 的信息将被搞乱。当你使用的时候，你必须把这些信息关掉。

另外，拨号的延迟是 60 秒。你的 modem 可能使用另外的时间或提示认为有其他的通讯问题。试试 `ATS7=45&W`。

实际上，有些提示不完全被支持。解决方法是编辑 `/usr/src/usr.bin/tip/tip` 目录中的 `tipconf.h` 文件。很明显，你需要它的源代码。

把行 `#define HAYES 0` 修改成 `#define HAYES 1`。然后 `make, make install`。这样就好了。

#### 15.5.2 我如何键入这些命令?

在 `/etc/remote` 文件中编译一个叫做 `direct` 的记录。例如，如果你的 modem 连接在第一个串行端口，`/dev/cuaa0`，就放进下面这行：

```
cuaa0: dv=/dev/cuaa0: br#19200: pa=none
```

在 br 项中使用最高的传输速率。然后键入 cuaa0, 你就可以连接到你的 modem 了。如果在你的系统上没有/dev/cuaa0, 可以这样:

```
# cd /dev
```

```
# MAKEDEV cuaa0
```

或以 root 使用 cu:

```
# cu -l line -sspeed
```

*line* 是串行端口(如 ./dev/cuaa0), *speed* 是速度(如 57600)。当你键入~. 就退出。

### 15.5.3 在 pn 现@标记不能工作?

在电话号码项@标记告诉电脑在/etc/phones 文件中查找一个电话号码。但@标记也是一个在象/etc/remote 这样的可用文件中的特殊的字符。用一个反斜线符号退出:

```
pn=\@
```

### 15.5.4 我如何在命令行拨电话号码?

在你的/etc/remote 文件中通常放着一个叫做 generic 的记录。例如:

```
tip115200|Dial any phone number at 115200 bps:\
```

```
      : dv=/dev/cuaa0: br#115200: at=hayes: pa=none: du:
```

```
tip57600|Dial any phone number at 57600 bps:\
```

```
      : dv=/dev/cuaa0: br#57600: at=hayes: pa=none: du:
```

然后, 你可以这样:

```
# tip -115200 5551234
```

如果你更喜欢 cu, 使用一个通用的 cu 记录:

```
cu115200|Use cu to dial any number at 115200bps:\
```

```
: dv=/dev/cuaa1: br#57600: at=hayes: pa=none: du:
```

然后键入：

```
# cu 5551234 -s 115200
```

### 15.5.5 通过一个终端服务器我能访问许多主机。

不用等待，除非你每次连接需要键入 CONNECT <host>，使用 tip 的 cm 功能。例如，这些记录在 /etc/remote 中：

```
pain|pain.deep13.com|Forrester's machine:\
```

```
:cm=CONNECT pain\n:tc=deep13:
```

```
muffin|muffin.deep13.com|Frank's machine:\
```

```
:cm=CONNECT muffin\n:tc=deep13:
```

```
deep13:Gizmonics Institute terminal server:\
```

```
:dv=/dev/cuaa2: br#38400: at=hayes: du: pa=none: pn=5551234:
```

## 15.6 设置串行控制台

### 15.6.1 介绍

FreeBSD 可以通过一个串行口只使用一个哑终端就可以启动一个系统。这样一种配置只有两种人能使用：希望在机器上安装 FreeBSD 的系统管理员，他没有键盘或显示器，还有就是调试内核或设备驱动程序的开发人员。

就象第 7 章描述的，FreeBSD 可以使用一个三步的启动过程。最先两步被储存在 FreeBSD 启动磁盘的启动 slice 的启动代码块中。启动块然后就被加载，接着运行第三步启动引导器 (/boot/loader)。

为了设置串行控制台，你必须配置启动块代码，启动引导器代码和内核。

### 15.6.2 串行控制台的配置

1. 准备一个串行线缆。

你需要使用一个 null-modem 的线缆或一个标准的串行线和一个 null-modem 适配器。看看第 15.2.2 节有关串行线的讨论。

### 2. 拔去你的键盘。

绝大多数的 PC 在开机检测的时候会检测到键盘，如果键盘没有被检测到，将会出现一个错误。一些机器会提示丢失键盘，就不会继续引导系统。

如果你的电脑出现错误，但仍能继续启动，你可以不必理它。

如果你的电脑没有键盘拒绝启动，那你需要配置 BIOS 来避免这个错误。看看你的主板的使用说明了解更多细节。

**提示：**在 BIOS 中设置键盘 Not installed 并不意味着你不能使用键盘。这样做只是告诉 BIOS 不要在机器开机检测时检测键盘，以至不会提示说系统找不到键盘。即使你设置了 Not installed，只要把你的键盘插上去仍然可以使用。

**注意：**如果你的系统有一个 PS/2 鼠标，如果机会好的话，你也可以象键盘一样把它拔下来，这是因为 PS/2 鼠标与键盘的一些硬件是共享的，你的鼠标插上去，系统会认为键盘仍在那儿。

### 3. 插一个哑终端到 COM1: (si o0)。

如果你没有一个哑终端，你可以使用一个比较老的带有一个 modem 程序的 PC/XT 机器，或在其他 unix 机器的串行口。如果你没有 COM1: (si o0)，去找一个。这时，你没有办法只能选择 COM1: 来启动系统。如果你已经在另一台设备上使用 COM1:，你必须临时删除那个设备，然后安装一个新的系统启动块和内核。

### 4. 确信你的内核配置文件已经为 COM1: (si o0) 设置了适当的标记：

有关的标记是：

0x10

启用控制台支持。其他的控制台标记会被忽略，除非它被设置了。现在，绝大多数的设置都有控制台的支持；这个标记的第一个就是首选的。这个单独选项是不能确保串行口适用于控制台的，设置下面的标记或加上下面描述的 -h 选项，和这个放在一起。

0x20

不管下面有没有讨论，都迫使这个选项支持控制台。这个标记在 FreeBSD 2. X 中替换了 COMCONSOLE 选项。标记 0x20 必须和 0x10 一起使用。

0x40

保存这个设置，确保这个设置不能用于普通访问。你不要把这个标记放在你要使用的串口设置中。这个标记的唯一的用处是在进行远程内核调试时用于指派单位。看看开发人员手册了解更多信息。

**注意：**在 FreeBSD 4.0-CURRENT 和以后的版本中，标记 0x40 通常是不同的，有另一个标记可以来指定一个串行口用于远程调试。

例：

```
device sio0 at isa? port "IO_COM1" tty flags 0x10 irq 4
```

看看 sio 的联机手册了解更多信息。如果标记没有被设置，你必须运行 UserConfig 或重新编译内核。

5. 在启动磁盘的 a 分区的根目录创建一个 boot.config 文件。

这个文件将指导启动块代码如何启动系统。为了激活串行控制台，你必须有一个或多个下面的选项---如果你要多个选项，在同一行必须都包含它们：

-h

切换内部和串行控制台。你使用这个来交换控制台设备。例如，如果你从内部控制台启动，你可以使用-h 来直接使用启动引导器和内核来使用串行口作为它的控制台设备。另外，如果你从串行口启动，你可以使用-h 来告诉启动引导器和内核使用显示设备作为控制台。

-D

切换单一和双重控制台配置。在单一配置中，控制台将是本机的控制台（显示设备）或串行口。在双重控制台配置中，显示设备和串行口将同时成为控制台，无论-h 的选项的情形。然而，双控制台配置只在启动块运行的过程中起作用。一旦启动引导器获得控制，由-h 选项指定的控制台将成为唯一的控制台。

-P

在启动时，探测键盘。如果键盘找不到，-D 和-h 选项会自动设置。

**注意：**由于启动块的当前版本的限制，-P 选项只能探测扩展的键盘。少于 101 键的键盘将无法被探测到。如果你碰到这个情况，你必须避免使用-P 选项。不幸的是这个问题还没有解决。

使用-P 选项来自动选择控制台，或使用-h 选项来激活控制台。

你也可以使用 boot 联机文档中所描述的其他选项。

除了-P 选项，所有选项将被传给启动引导器(/boot/loader)。启动引导器将通过检查-h 选项的状态来决定是显示设备成为控制台，还是串行口成为控制台。这意味着如果你指定-D 选项，但在/boot.config 中没有-h 选项，你在启动块时使用串行口作为控制台；启动引导器将使用内部显示设备作为控制台。

### 6. 启动机器

当你启动你的 FreeBSD 时，启动块将把/boot.config 的内容发给控制台。例如：

```
/boot.config: -P
```

```
Keyboard: no
```

如果你把-P 放在/boot.config 中并指出键盘存在或不存在，那将只出现第二行。这些信息会被定位到串行口或内部控制台，或两个都是，完全取决于/boot.config 中的选项。

选项	定位信息
none	internal console
-h	serial console
-D	serial and internal consoles
-Dh	serial and internal consoles
-P, keyboard present	internal console
-P, keyboard absent	serial console

出现上面信息后，在启动块加载启动引导器和更多信息被映到屏幕之前将有一个小小的停顿。在通常情况下，你不需要打断启动进程，但为了确信设置是否正确，你也可以这样做。

键入任何键，而不是 Enter，控制台会打断启动进程。启动块将进入命令行模式。你看到：

```
>> FreeBSD/i386 BOOT

Default: 0:wd(0,a)/boot/loader

boot:
```

检验上面出现的信息，可能是串行口，或内部控制台，或两个都是，完全取决于你在 `/boot.config` 中的选项。如果信息出现在正确的控制台，键入 Enter 继续启动进程。

如果你要使用串行控制台，但你没有看到命令行，那可能设置有问题。这时，你键入 `-h` 然后单击 Enter/Return 来告诉启动块选择串行口作为控制台。一旦系统起来了，回去检查一下是什么出问题了。

启动引导器被加载完后，你将进入启动进程的第三步，你仍然可以在启动引导器通过设定你喜欢的环境来切换内部控制台和串行控制台。看看第 15.6.5 节。

### 15.6.3 摘要

这是几个在这章要讨论的几个设置和选择的控制台的摘要。

#### 15.6.3.1 Case 1: You Set the flags to 0x10 for sio0

```
device sio0 at isa? port "IO_COM1" tty flags 0x10 irq 4
```

Options in <code>/boot.config</code>	Console during boot blocks	Console during boot loader	Console in kernel
nothing	internal	internal	internal
<code>-h</code>	serial	serial	serial
<code>-D</code>	serial and internal	internal	internal
<code>-Dh</code>	serial and internal	serial	serial
<code>-P</code> , keyboard present	internal	internal	internal
<code>-P</code> , keyboard absent	serial and internal	serial	serial

## 15.6.3.2 Case 2: You Set the flags to 0x30 for si o0

```
device si o0 at isa? port "lO_COM1" tty flags 0x30 irq 4
```

Options in /boot.config	Console during boot blocks	Console during boot loader	Console in kernel
nothing	internal	internal	serial
-h	serial	serial	serial
-D	serial and internal	internal	serial
-Dh	serial and internal	serial	serial
-P, keyboard present	internal	internal	serial
-P, keyboard absent	serial and internal	serial	serial

## 15.6.4 串行控制台的提示

## 15.6.4.1 设置一个快速的串行口速度

默认的串行口被设置成 9600 波特, 8 位, 没有奇偶性, 1 个停止位。如果你希望改变速度, 你必须重新编译启动块。在 /etc/make.conf 中添加下面一行, 然后编译新的启动块:

```
BOOT_COMCONSOLE_SPEED=19200
```

如果串行控制台用其他方法来配置而不是在启动时用 -h ,或内核使用的串行控制台与启动块使用的不同, 那你必须在内核配置文件中添加下面这行, 然后编译内核:

```
options CONSPEED=19200
```

## 15.6.4.2 使用串行口而不是 si o0 作为控制台

使用串行口而不是 si o0 作为控制台需要做一些重编译。如果你无论如何都要使用另一个串行口, 重新编译启动块, 启动引导器和内核。

1. 得到内核源代码。
2. 编辑 /etc/make.conf 文件, 然后设置 BOOT\_COMCONSOLE\_PORT 作为你要使用 (0x3F8, 0x2F8, 0x3E8 or 0x2E8) 端口的地址。只有 si o0 到 si o3 (COM1: through COM4:) 能被使用; 多接口串行卡将不会工作。不需要任何中断设置。

3. 创建一个定制的内核配置文件，在你要使用的串行口添加合适的标记。例如，如果要将 `si01(COM2:)` 作为控制台：

4. `device si01 at isa? port "IO_COM2" tty flags 0x10 irq 3`

或

```
device si01 at isa? port "IO_COM2" tty flags 0x30 irq 3
```

其他端口的控制台标记也不要设。

5. 重新编译和安装启动块：

6. `# cd /sys/boot/i386/boot2`

7. `# make`

8. `# make install`

9. 重编译和安装启动引导器：

10. `# cd /sys/boot/i386/loader`

11. `# make`

12. `# make install`

13. 重建和安装内核。

14. 用 `disklabel` 将启动块写到启动磁盘上，然后从新内核启动。

### 15.6.4.3 通过串行线键入 DDB 调试器

如果你想通过串行控制台进行内核调试，你需要在编译内核时加上下面选项：

```
options BREAK_TO_DEBUGGER
```

```
options DDB
```

### 15.6.4.4 在串行控制台上得到一个登陆命令行

你可能希望通过串行线得到一个登陆命令行，现在你可以看到启动信息，通过串行控制台键入内核调试信息。可以这样做。

用一个编辑器打开/etc/ttys 文件，然后定位到下面的行：

```
ttyd0 "/usr/libexec/getty std.9600" unknown off secure
```

```
ttyd1 "/usr/libexec/getty std.9600" unknown off secure
```

```
ttyd2 "/usr/libexec/getty std.9600" unknown off secure
```

```
ttyd3 "/usr/libexec/getty std.9600" unknown off secure
```

ttyd0 到 ttyd3 相当于 COM1 到 COM4。可以打开或关闭某个端口。如果你已经改变了串行口的速度，你必须修改标准的 9600 与当前的例如 19200 相匹配。

你也可以改变终端的类型从不知名的到你串行终端的真实类型。编辑完这个文件，你必须杀掉-HUP 1 来使这个修改启用。

### 15.6.5 从启动引导器修改控制台

前面一节描述了如何通过调整启动块来设定串行控制台。本节将讲到在启动引导器中通过键入一些命令和环境变量来指定控制台。由于启动引导器会被启动进程的第三步所调用，启动块以后，在启动引导器中的设置将忽略在启动块中的设置。

#### 15.6.5.1 设定串行控制台

你可以很容易地指定启动引导器和内核来使用串行控制台，只需要在/boot/loader.rc 中写入下面这行：

```
set console=comconsole
```

你最好把上面一行放在文件的第一行，以至于尽早地在启动时看到串行控制台的启动信息。同样地，你可以指定内部控制台为：

```
set console=vidconsole
```

如果你不设置启动引导环境变量控制台，启动引导器和内核将使用在启动块时用-h 选项指定的控制台。

在版本 3.2 或以后的版本中,你可以在 `/boot/loader.conf.local` 或 `/boot/loader.conf` 中指定控制台,而不是在 `/boot/loader.rc` 中。在这个方法中,你的 `/boot/loader.rc` 文件将是这样的:

```
include /boot/loader.4th
```

```
start
```

然后,创建 `/boot/loader.conf.local` 放上下面的行。

```
console=comconsole
```

或

```
console=vidconsole
```

看看 `loader.conf` 的联机手册了解更多信息。

### 15.6.5.2 使用串行口而不是 `si o0` 作为控制台

你需要使用一个串行口而不是 `si o0` 重新编译启动引导器作为串行控制台。下面的步骤跟第 15.6.4.2 节描述的相似。

### 15.6.6 警告

这篇文章本意是想告诉人们如何设定没有显示设备或键盘的专用服务器。不幸的是,绝大多数系统没有键盘可以让你启动,而没有显示设备就不让你启动。使用 AMI BIOS 的机器可以通过在 CMOS 中将“graphics adapter”项设为“Not installed”来在启动时不需要显示适配器。

然而,许多机器不支持这个选项,如果你的系统没有显示硬件就拒绝启动。对于这些机器,即使你没有显示器,你也必需在你机器上插上显示适配器。建议你试试用 AMI BIOS 的机器。