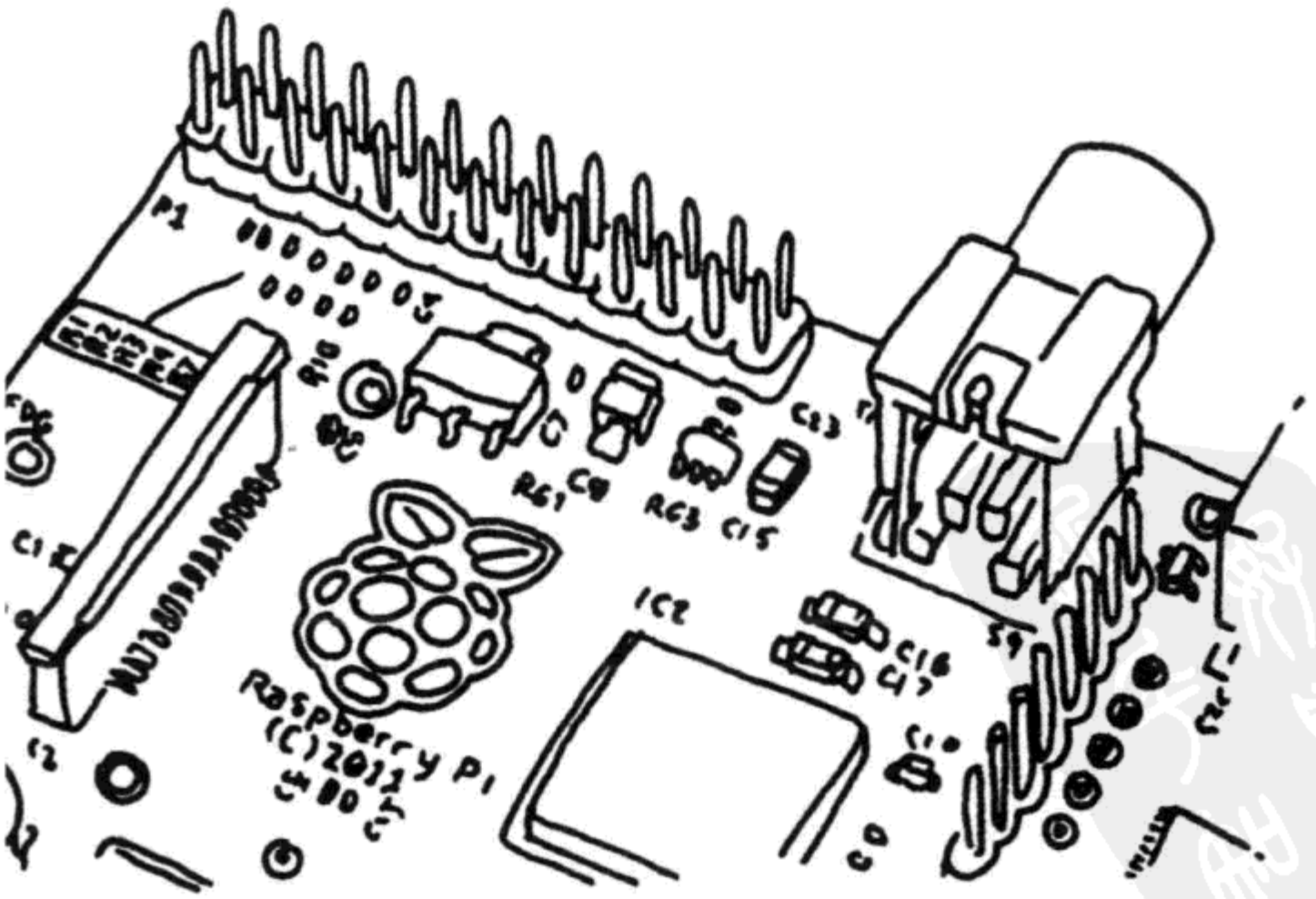
第 7 章

基本输入输出

Basic Input and Output



Raspberry Pi 本质上是一台廉价的 Linux 电脑，但它与我们平时用于上网、写邮件或进行文字处理的台式电脑或笔记本仍有一些不同之处。其中一个最明显的差别就是，Raspberry Pi 的主板上提供了一组 GPIO（General Purpose Input & Output，通用输入输出）接口，这些接口可以直接用于开发一些电子相关的项目，如图 7.1 所示。

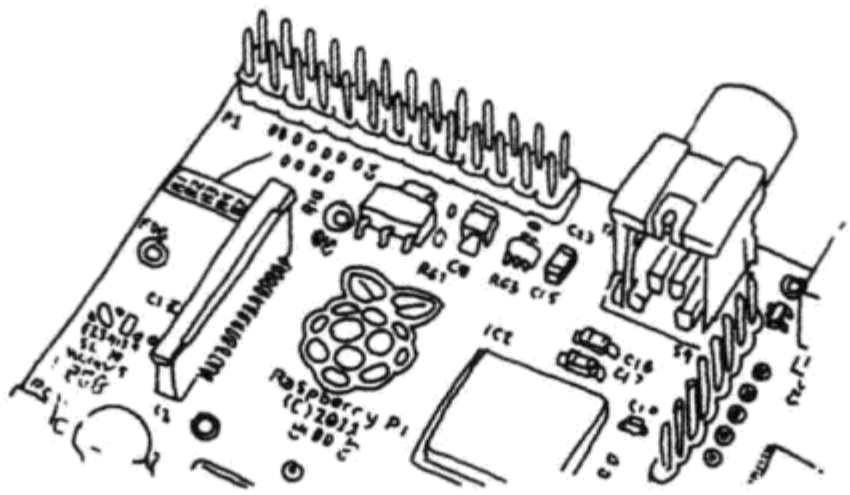


图7.1　Raspberry Pi的GPIO接口

这些GPIO接口可以作为控制信号输出口来控制一些硬件设备，如发光二极管（LED）、电动机或继电器。它们也可以作为信号输入接口，用于读取按钮或开关的状态，获取键盘上按下的键，或者获得温度、光线、运动和距离等各种传感器的状态。

Raspberry Pi 的 GPIO 有一个小小的缺点——它无法直接连接到以模拟信号输出的传感器上，如模拟亮度传感器或温度传感器。必须通过一个模数转换器（ADC）芯片才能让 Raspberry Pi 读取模拟值。相关的内容请参考附录 C。

如果电脑具备 GPIO，它的最大优势在于你可以编写程序从 GPIO 口上读取状态或根据不同的条件向 GPIO 口输出值，就与给普通台式电脑编写程序一样方便。Raspberry Pi 与其他具备 GPIO 口的单片机开发板不同之处在于，它还具备键盘、鼠标、显示器及以太网口等输入输出设备。如果你有过其他单片机的开发经验（如 Arduino 开发板），Raspberry Pi 的 GPIO 接口虽然比它要少，但 Raspberry Pi 自带的其他输入输出设备不需要额外的电路或接线就可以直接使用，用起来更为方便。

具备键盘、鼠标、显示器等接口不是 Raspberry Pi 区别于其他单片机开发板的唯一优点，在开发电子项目时，它还具备以下的优点。

文件系统

可以直接读写 Linux 文件系统，会使很多项目实现起来更为容易。比如，把一个温度传感器连接到 Raspberry Pi，每秒读取一次温度值。你可以把每次读到的值添加到一个日志文件中，这个文件后期可以很方便地下载到电脑中并用于绘制一条温度变化曲线。你甚至可以直接在 Raspberry Pi 上完成这项绘图的工作。

Linux 工具

Raspberry Pi 的 Linux 发行版中包含了一系列的命令行工具，你可以使用这些工具来操作文件、控制进程或把很多任务进行自动化处理。这些灵活的工具都可以在你的项目中直接使用。由于有很多 Linux 用户都依赖这些工具来完成日常工作，所以你遇到问题时，可以很方便地在网上搜索到答案。有关 Linux 相关的问题，你常常可以在 Stack Overflow 网站（*http://stackoverflow.com*）上找到答案。如果遇到了 Raspberry Pi 上特有的问题，可以尝试访问 Raspberry Pi

的论坛（*http://www.raspberrypi.org/phpBB3/*）或 Stack Overflow 上的 Raspberry Pi 版块（*http://raspberrypi.stackexchange.com*）。

编程语言

在 Raspberry Pi 这样的嵌入式 Linux 系统中，系统支持很多不同的开发语言，可以选择你自己最喜欢的一种来进行开发。在本书中，主要使用 Shell 和 Python 作为开发语言，但你也可以很方便地把它们转换为 C、Java、Perl 或其他任何一种开发语言。

使用输入输出接口

除了 Raspberry Pi 本身以外，你还需要一些其他的配件，才可以完成下面的实验。这些配件通常可以在本地的电子市场里购买到，也可以在网上商店选购。下面是一个简要的清单。

- 面包板。
- 各种发光二极管（LED）。
- 杜邦线（公头对公头）。
- 母头对公头连接线，用于连接 Raspberry Pi 的 GPIO 接口到面包板上，这种线比较少见。
- 小按钮。
- 各种电阻。

为了更容易地连接 Raspberry Pi 与面包板，我们推荐使用 Adafruit 的 Pi Cobbler Breakout 套件，使用它可以省去母头对公头的连接线。这个套件是以散件的形式提供的，你需要自己进行焊接，焊接的过程并不复杂，可以参考 Adafruit 的教程（*http://learn.adafruit.com/adafruit-pi-cobbler-kit/overview*）一步步地完成。*MAKE* 的 Raspberry Pi 入门套装（*http://oreil.ly/pikit*）中包含了 Adafruit 的 Pi Cobbler Breakout 套件，并同时提供了上面所列出的其他相

关元件（除了母头对公头连接线，因为用了 Adafruit 的 Pi Cobbler Breakout 套件后，这种连接线就不需要了）。

图 7.2 中，我们为 Raspberry Pi 的每个 GPIO 接口引脚标记了对应的编号，在你的程序中，需要使用这些编号来操作对应的引脚。图中没有标记的引脚默认被设置为其他的用途。

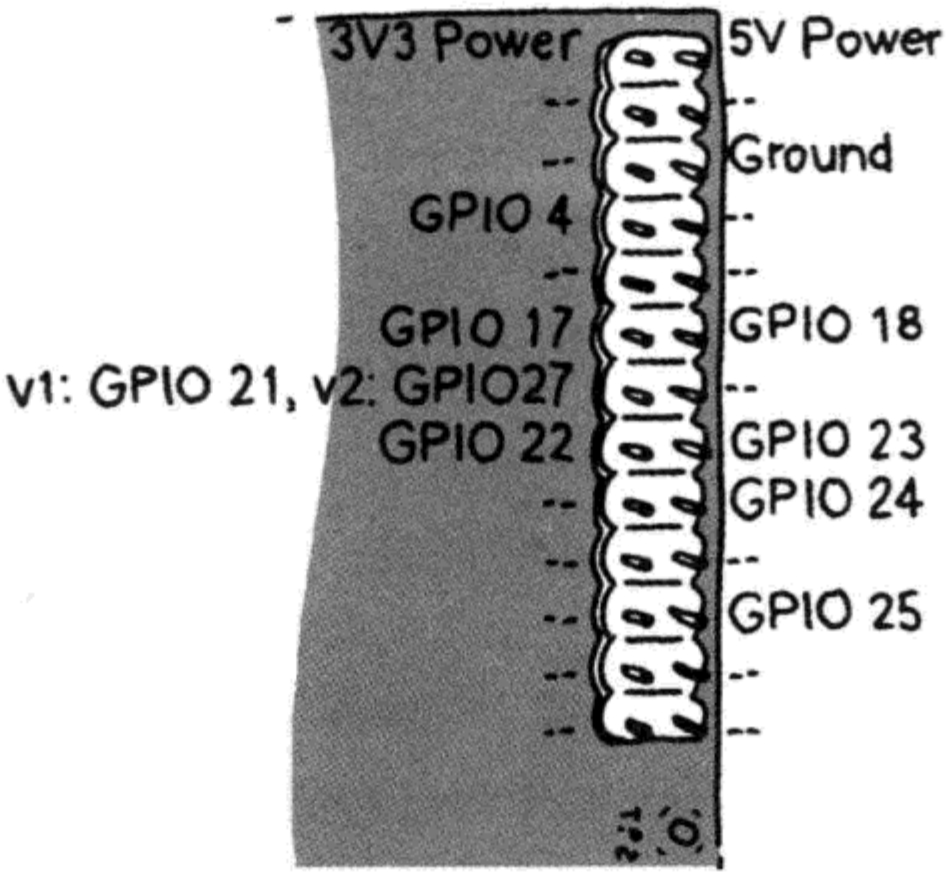


图7.2　Raspberry Pi上默认的GPIO接口。在较新版本的Raspberry Pi上，GPIO 21被改成GPIO 27

也许你注意到图 7.2 中有一个针脚上被标注了两个不同的 GPIO 口编号。在较新版本的 Raspberry Pi 主板上，GPIO 21 变成了 GPIO 27。可以在命令行中执行 *cat /proc/cpuinfo* 来获知你的 Raspberry Pi 属于哪个版本，如果显示出来的版本号是 0002 或 0003，你使用的板子就是第一版的。如果显示出来的版本号比 0003 更大或者包含字母，则你使用的板子是第二版的。

数字信号输出：点亮 LED

使用 GPIO 接口的最简单的方法就是连接一个发光二极管（LED），你可以使用 Linux 命令行来点亮或熄灭 LED。一旦你理解了这些命令背后的工作原理，就可以很容易地使用 LED 来提醒你收到了一封新的邮件、下雨天出门前要带上雨伞或到了上床睡觉的时间。除了控制 LED 外，使用一个继电器来定时控制一个灯的开关，也是很容易的。

面包板使用方法入门

如果你以前从来没有使用过面包板，那首先要了解的是面包板上的插孔是如何连接在一起的。图 7.3 中，我们把一块常见的面包板上彼此相连的插孔用阴影包络在一起。要注意，左右两侧的供电总线之间并不相连。如果需要让两侧的供电总线同时提供电源和接地，你需要使用两根公头对公头的连接线把它们连到一起。

1. 使用公头对母头的连接线，把 Raspberry Pi 的 GPIO 25 与面包板相连。请参考图 7.2 了解 Raspberry Pi 的引脚与 GPIO 接口编号之间的关系。

2. 使用另一根连接线，把 Raspberry Pi 的接地（Ground）脚与面包板上供电总线的负极相接。

3. 现在，你可以连接 LED 了（图 7.4）。在开始连接之前，你需要知道 LED 是有极性的：LED 的两个引脚的接线不能颠倒。LED 的引脚中，较长的一根是正极，应当与 GPIO 口相连；较短的一根则是负极，需要接地。另一种区别正负的方式是从 LED 的顶部往下看，

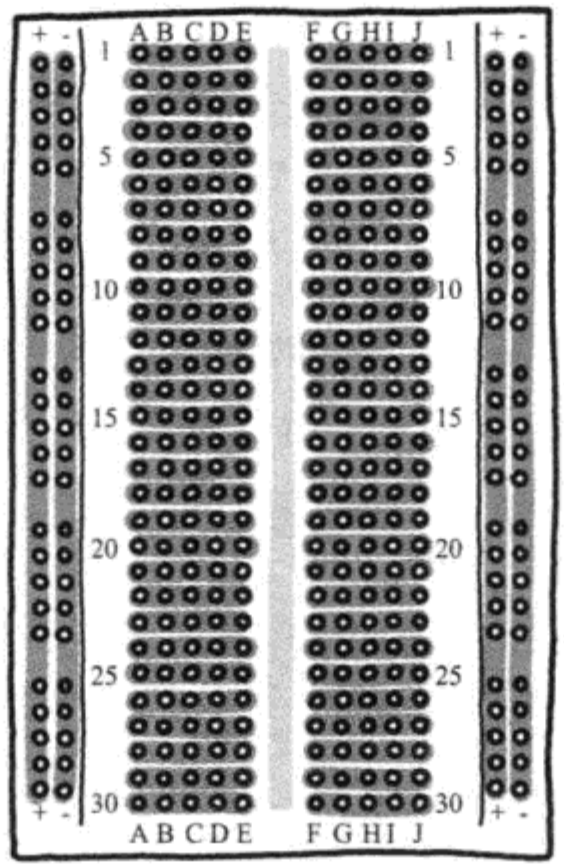


图7.3　面包板

LED 边缘被切平的一侧的引脚是 LED 的负极，需要接地。把 LED 的正极插入面包板上接 Raspberry Pi GPIO 25 的同一行，这就把它与 GPIO 25 相连了。同时，还需要把 LED 的负极与面包板上的电源总线的负极相接。

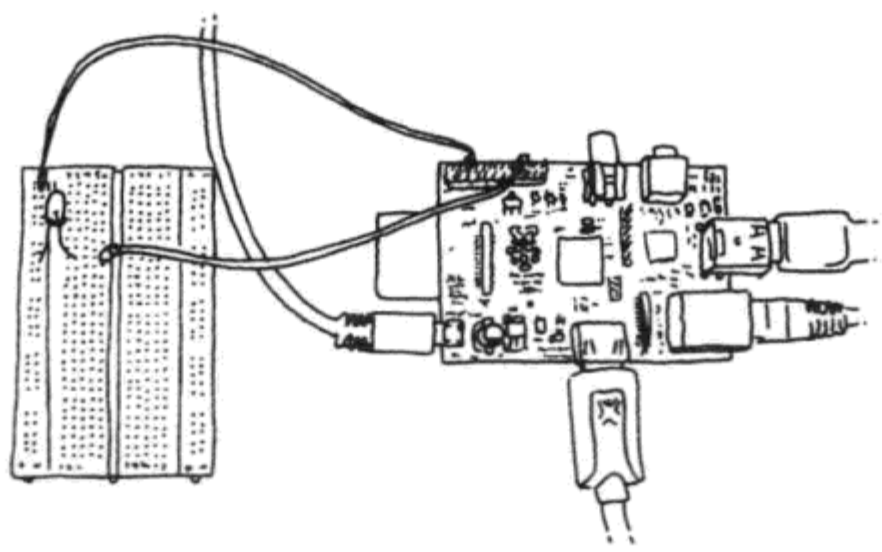


图7.4　把LED与Raspberry Pi相连[①]

① 如果书中此类手绘图中不容易看清实际的电路连接情况，请访问本书的支持网站 *http://rpi.freemindworld.com* 查看清晰的面包板连线图。——译者注

4. 接好你的键盘、鼠标和显示器，启动你的 Raspberry Pi 并登录系统。进入命令行模式，准备开始后面的操作。如果你在 X Window 环境中，双击桌面上的 LX 终端（LXTerminal）图标，打开一个命令行窗口。

5. 为了从命令行操作 Raspberry Pi 的输入输出接口，你需要以 root 权限（管理员权限）来运行相关的命令。通过输入 sudo su 并回车，可以把命令行切换到管理员权限模式下：

```
pi@raspberrypi ~ $ sudo su
root@raspberrypi:/home/pi#
```

可以注意到，命令行的提示符发生了变化，提示你后续的命令都将以 root 权限运行。

root 账号具有管理员权限，可以对系统的所有文件和功能进行操作。如果你输入了一个可能破坏整个系统的命令，系统也不会对自己实施太多的保护措施，所以，当你以 root 权限来操作系统时，要格外小心。不过，在 Raspberry Pi 上，即使你做了一些错误的操作破坏了系统，也不需要过于担心，因为你总是可以通过重新烧录 SD 卡的镜像来安装一个全新的 Linux。当你完成了需要 root 权限的相关操作后，可以输入 exit 回到 pi 用户权限下。

6. 在开始使用命令来操作 GPIO 25 上所接的 LED 前，你需要先把这个引脚的操作接口从内核空间暴露到用户空间中，使用这个命令：

```
root@raspberrypi:/home/pi# echo 25 > /sys/class/gpio/export
```

使用 echo 命令把你想操作的 GPIO 编号（25）写入 */sys/class/gpio* 下的 *export* 文件，当你把 GPIO 编号写入这个文件后，*/sys/class/gpio* 下会自动新建一个 */sys/class/gpio/gpio25* 目录，里面包含了操作这个 GPIO 口所需要的控制文件。

7. 用 cd 命令切换到这个新建的目录下，并用 ls 命令列出文件清单：

```
root@raspberrypi:/home/pi# cd /sys/class/gpio/gpio25
root@raspberrypi:/sys/class/gpio/gpio25# ls
active_low  direction  edge  power  subsystem  uevent  value
```

cd 命令是 Change Directory（改变目录）的缩写，它能把当前的工作目录切换到指定的目录下，这样就不需要每次都输入目录的完整路径。ls 命令可以列出目录下的文件和子目录。在这个目录下，你需要操作两个文件：direction 和 value。

8. direction 文件用于指定这个 GPIO 接口会被用于输入（接一个按钮）或是输出（接一个 LED）。由于目前你所需要控制的 LED 连接在 GPIO 25 接口上，所以你应该把这个接口设置为输出模式：

```
root@raspberrypi:/sys/class/gpio/gpio25# echo out > direction
```

9. 使用 echo 命令向 value 文件中输出 1，你就可以点亮这个 GPIO 口上所接的 LED：

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 1 > value
```

10. 输入回车以后，LED 就被点亮了！要把它熄灭，也很简单，只要用 echo 命令向 value 文件中输出 0 就可以了：

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 0 > value
```

虚拟文件

在这个实验中所操作的很多“文件”其实并不真实存在于 Raspberry Pi 的 SD 卡上，它们是 Linux 虚拟文件系统的一部分。Linux 虚拟文件系统提供了用户模式下操作系统底层硬件的一种简易的接口。其实，也可以通过向 Raspberry Pi 内存的一个指定位置写入一些特定的值来点亮或熄灭 LED，但是这比使用虚拟文件系统操作要复杂得多，也更容易出错。

既然通过写入文件可以控制向 GPIO 接口输出信息，那应该如何从GPIO读入数据呢？也许你会猜，是不是该通过“读取文件”呢？完全正确，下面我们就会来完成这项工作。

数字信号输入：读取按钮状态

如图 7.5 所示，使用简单的按钮开关来产生数字输入信号非常合适，它们可以稳妥地插入到面包板上。

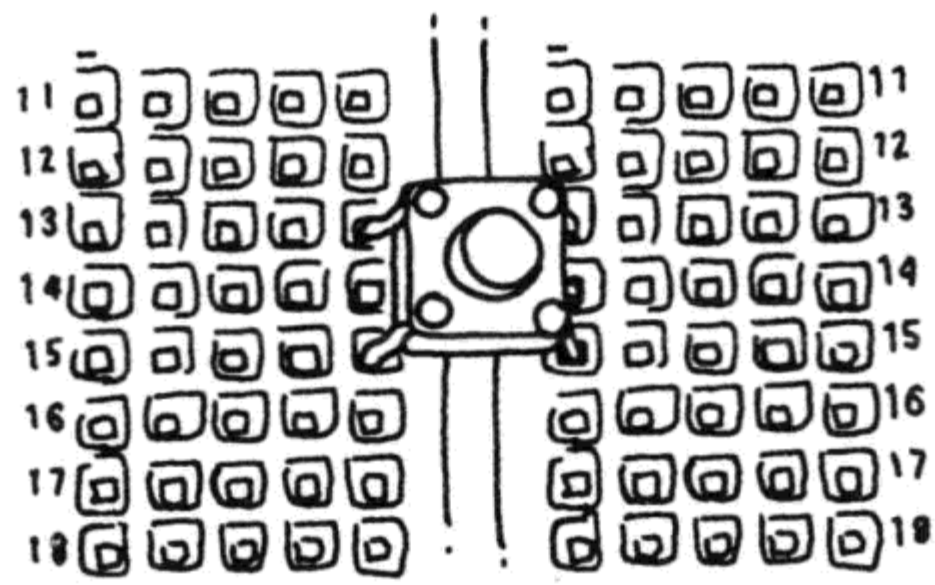


图7.5 按 钮

这类按钮开关在电子实验中很常用，了解它们的内在构造可以更好地帮助你设计电路。观察面包板上这些按钮（图 7.5）：上部的两个引脚之间永远相通，下部的两个引脚亦然。当你把按钮按下的时候，这两组引脚之间就被连通了。

当从 Raspberry Pi 的 GPIO 接口上读取数据时，实际上你是在检查这个接口是被接在 3.3V 的电源上还是被接地了。必须牢记，输入接口必须连接一个确定的信号（如 3.3V 电源或者接地），如果尝试读取一个既未接电源又未接地的引脚，你会得到一个不稳定的结果。当学会了按钮的工作原理，你可以试着将磁控开关、游戏手柄，甚至自动售货机上的硬币识别器作为数字输入信号连接到 GPIO 接口上。

1. 把按钮插入面包板，并使它横跨在面包板中央绝缘条的两边。

2. 使用连接线，把 Raspberry Pi 的 GPIO 24 与按钮上部的一个引脚相连。

3. 把 Raspberry Pi 的 3.3V 供电口与面包板上供电总线的正极相连。

注意：需要输入高电平时，只能把 3.3V 的电源与按钮相接，不能接 5V 的电源。向 Raspberry Pi 的 GPIO 接口输入大于 3.3V 的电压将对会彻底烧坏 Raspberry Pi。

4. 把按钮下部的一个引脚接到供电总线的正极上，这时，当你按下按钮时，3.3V 的电压会接入 GPIO 24 接口。

5. 还记得我们说过要保证数字输入接口应该是接 3.3V 电源或

接地吗？在目前的情况下，你没有按下按钮，那么 GPIO 24 是没有接到电源或接地的，因此，它是悬空的。这种情况会带来不稳定的数据读取结果，所以必须解决这个问题。使用一个 10kΩ 的电阻（电阻色环为：棕、黑、橙[①]，最后一环为银色或金色），把按钮靠近输入的一端用电阻与 Raspberry Pi 的地线相接，这样当你没有按下按钮时，Raspberry Pi 的 GPIO 是通过电阻接地的。

由于电流总是会往接地电阻最小的通路流动，当你按下按钮时，3.3V 的电压会接入 Raspberry Pi 的输入接口，因为这条通路的电阻小于 10kΩ。

当所有的准备工作都完成时，整个电路看上去应该如图 7.6 所示。

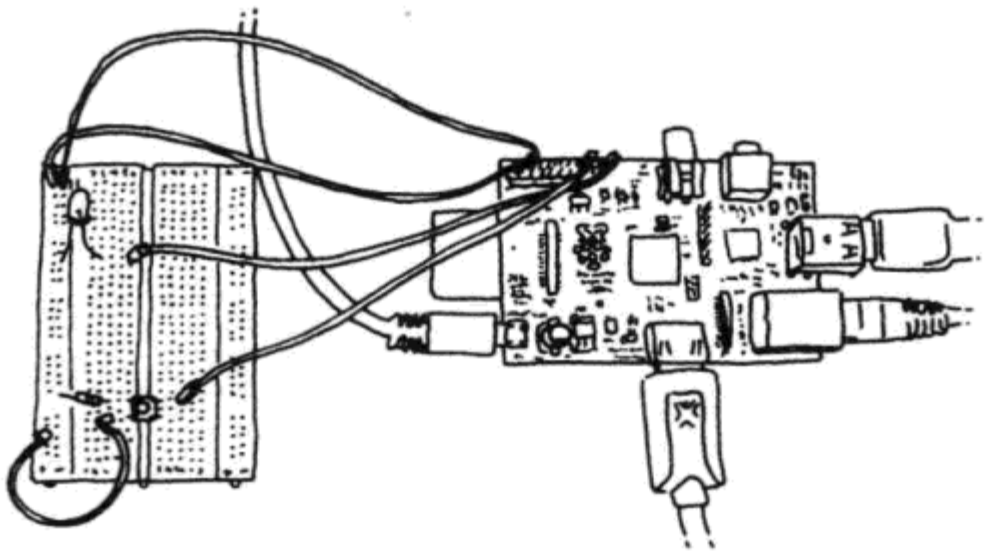


图7.6　把按钮与Raspberry Pi相连

6. 电路已经连接好了，这时我们就可以从命令行上读取按钮的状态了。如果你还没有取得 root 权限，就先执行 sudo su。

7. 与前一个例子一样，第一步先要把输入接口暴露到用户态：

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 24 > /sys/
class/gpio/export
```

① 现在市面上有些电阻采用 5 色环标注阻值，10kΩ 电阻对应的前 4 个色环是棕、黑、黑、棕，第 5 个色环表示阻值精度。——译者注

8. 把工作目录切换到上一步操作所生成的目录中：

```
root@raspberrypi:/sys/class/gpio/gpio25# cd /sys/class/gpio/
    gpio24
```

9. 设置接口的工作模式为输入：

```
root@raspberrypi:/sys/class/gpio/gpio24# echo in > direction
```

10. 现在，你可以用 cat 命令来读取按钮的状态了，cat 命令用于在命令行下显示文件的内容。cat 这个命令的命名来源于它可以用来把多个文件拼接（Concatenate）在一起，它也可以用于显示文件内容。

```
root@raspberrypi:/sys/class/gpio/gpio24# cat value
0
```

11. 显示的0表示这个接口目前是接地的。你可以按住按钮不放，同时再次执行这个命令：

```
root@raspberrypi:/sys/class/gpio/gpio24# cat value
1
```

12. 如果你看到它显示了1，就表示你的电路已经正常工作了！

如果需要执行一个以前执行过的命令，可以通过不断地按光标向上键查询执行过的命令，找到你想要的命令时，按回车键执行。

现在你已经学会了如何通过 Linux 的命令行来控制 LED 或读取按钮的状态，下面让我们一起来用一些常用的 Linux 内置工具操作数字输入输出接口来完成一个简单的项目。

项目：定时台灯

假设明天一早你需要离开家去外地度假，希望你不在家的这段时间里，不要吸引小偷来光顾。用一个定时开关定时控制家里的台灯可以实现这个目的，可是附近的电器商店已经关门了，明天早上飞机起飞前你也没有足够时间去商店购买现成的定时开关。还好，你是 Raspberry Pi 的爱好者，你手头有一些实用的器件：

- Raspberry Pi 主板；
- 面包板；
- 连接线；
- PowerSwitch Tail II；
- 电线。

有了这些器材，你就可以自己动手编写一个定时开关，需要使用两个强大的 Linux 工具：Shell 脚本和 `cron`。

脚本命令

所谓 Shell 脚本，是指包含了很多 Shell 命令（如我们在上一节中用来控制 GPIO 接口的命令）的文件。看看下面的 Shell 脚本和我们对关键行的解释。

```
#!/bin/bash #❶
echo Exporting pin $1. #❷
echo $1> /sys/class/gpio/export #❸
echo Setting direction to out.
echo out > /sys/class/gpio/gpio$1/direction #❹
echo Setting pin high.
echo 1 > /sys/class/gpio/gpio$1/value
```

❶ 所有的 Shell 脚本都以这一行开头。

❷ $1 指代第一个命令行参数。

❸ 这个脚本不是固定地暴露某个特定的 GPIO 接口，而且通过命令行参数来指定接口号。

❹ 注意，这里的 GPIO 接口号也会被命令行参数替换。

把这个文本文件存为 on.sh，并通过 chmod 命令把它设置为可执行：

```
root@raspberrypi:/home/pi# chmod +x on.sh
```

你仍然需要通过 root 权限来执行这个脚本。如果你在执行脚本时遇到“Permission denied.”这样的出错提示，就先执行一下 sudo su。

命令行参数是一种向程序或脚本传递信息的方式，通过在命令后面加上参数，程序或脚本就可以在运行时获取到相应的值。当你在编写一个 Shell 脚本时，$1 代表了第一个命令行参数，$2 代表了第二个命令行参数，以此类推。在上面 on.sh 的例子中，你可以输入想要暴露和打开的 GPIO 接口编号（25）作为命令行参数。通过命令行参数来指定接口编号，而不是在 Shell 脚本中硬编码（Hard-code）具体的接口编号，可以让我们的 Shell 脚本变得更加通用。当你需要对 GPIO 25 进行操作时，只需执行：

```
root@raspberrypi:/home/pi/# ./on.sh 25❶
Exporting pin 25.
Setting direction to out.
Setting pin high.
```

❶ 文件名前的 ./ 表示你正在执行当前工作目录下的脚本。

如果你的 GPIO 25 上仍然连接着 LED，脚本执行后，它应该会被点亮。我们还可以编写另一个名为 *off.sh* 的脚本，用来把 LED 熄灭。这个脚本看起来应该是如下的样子：

```
#!/bin/bash
echo Setting pin low.
echo 0 > /sys/class/gpio/gpio$1/value
echo Unexporting pin $1
echo $1> /sys/class/gpio/unexport
```

为它加上可执行权限并执行它：

```
root@raspberrypi:/home/pi/temp# chmod +x off.sh
root@raspberrypi:/home/pi/temp# ./off.sh 25
Setting pin low.
Unexporting pin 25
```

如果一切正常，先前被点亮的 LED 现在会熄灭。

连接台灯

显然一个小小的 LED 所发出的光线不足以欺骗一个小偷让他误以为你在家中，所以下一步我们需要把台灯与 Raspberry Pi 连接起来。

1. 断开 LED 与 GPIO 25 的连接。

2. 从面包板上引出两条接线，分别引自 Raspberry Pi 的 GPIO 25 与地线。

3. 把从 GPIO 25 上引出的接线，接到 PowerSwitch Tail 上标有“+in”标记的引脚上。

4. 把从地线引出的接线，接到 PowerSwitch Tail 上标有“-in”标记的引脚上。

5. 把 PowerSwitch Tail 插到插座上，并把台灯插到 PowerSwitch Tail 上。请注意，需要把台灯的电源开关打开。

6. 这时，如果执行 `./on.sh 25`，台灯就会被打开；如果执行 `./off.sh 25`，台灯就会被关闭。

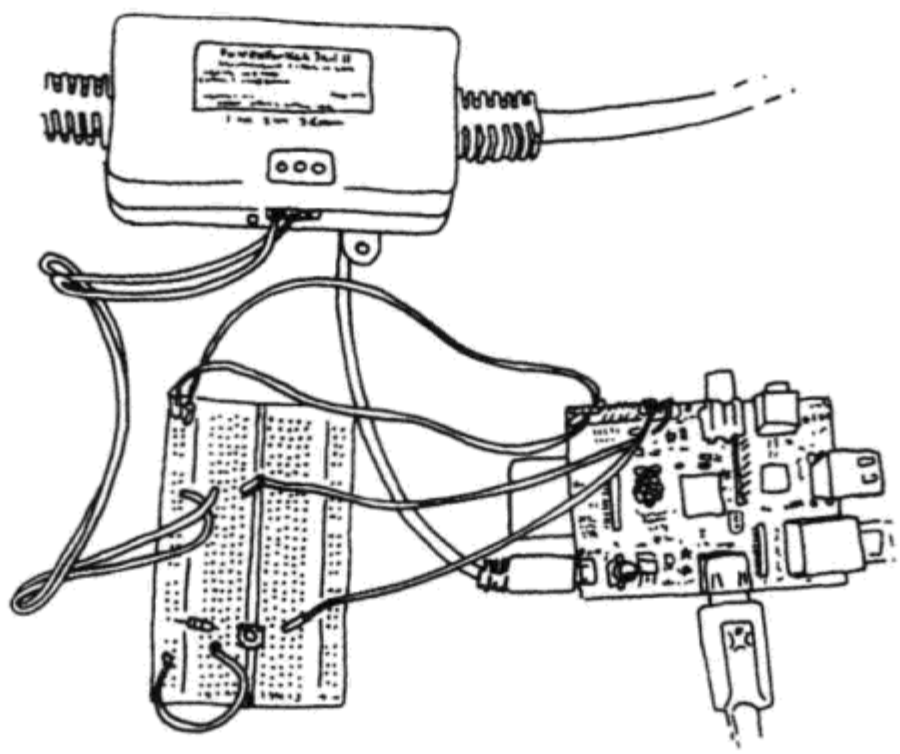


图7.7　连接PowerSwitch Tail与Raspberry Pi

在 PowerSwitch Tail 内部，有一系列电子元件可以帮助你实现用低电压信号（如来自 Raspberry Pi 的 GPIO 输出信息）去控制类似于台灯或电动搅拌机这样的高电压电器。当 PowerSwitch Tail 接通电器电源时，你可以听到“嗒”一声，这个声音来自它内部的核心器件——继电器。继电器相当于是高电压电器的开关，它的开关状态可以由 Raspberry Pi 的低电压信号来控制。

用 cron 设置定时任务

现在你已经把控制 GPIO 接口开关的命令整合成了两个简单的脚本，台灯也已经通过 PowerSwitch Tail 连接到了 Raspberry Pi 并且可以用单条的命令来控制它的开或关。下面就可以通过 cron 来设置每天定时开关灯的时间了。cron 是 Linux 中用于定时执行任

务的程序。通过使用 cron，你可以设置在指定的日期或时间执行指定的命令，或者以指定的时间间隔（如一小时）来执行命令。在我们的项目中，我们可以设定两个定时任务：每天晚上8点打开台灯，每天凌晨 2 点关闭台灯。

与其他时间相关的程序一样，使用 cron 前，需要确保你的 Raspberry Pi 已经按照第 2 章中所描述的方法，设定了正确的日期、时间。

要添加这些定时任务，我们需要修改 Linux 的 cron 表，cron 表中包含了一系列要求 Linux 在指定时间执行的命令[①]：

```
root@raspberrypi:/home/pi/# crontab -e
```

这条命令会打开一个文本编辑器，用于修改 root 用户的 cron 表。在这个文件的顶端，你能看到一些关于如何修改 cron 表的描述。用光标移动键把光标移动到文件的末尾，然后添加下面两行代码：

```
0 20 * * * /home/pi/on.sh 25
0 2 * * * /home/pi/off.sh 25
```

cron 会忽略任何以井号（#）开头的配置行。如果你需要临时禁用某一个定期任务但又不想删除它，可以

① Linux 下每个用户都有属于自己的 cron，互不干涉，每个用户 cron 表中的任务会以该用户身份所对应的权限运行。由于运行 on.sh 和 off.sh 脚本都需要 root 权限来操作 GPIO 接口，所以 crontab -e 这条命令需要在 root 权限下执行才能修改 root 用户的 cron 表，使用 root 用户的 cron 表才能保证在定时时间到达时，是以 root 的权限来运行相应的脚本。——译者注

在这个任务前添加一个井号。井号开头的行也可以作为cron表中的注释。

按 Control-X 键退出，在提示你是否要保存文件时输入 y，并按回车键接受它提示的默认文件名。当系统把这个文件保存好并退回命令行状态时，屏幕上会显示“installing new crontab”，表示你对 cron 表所做的修改会被 cron 程序运行。

更多有关 cron 的知识

cron 允许你设置在指定日期时间或按指定间隔执行的任务。通过用空格隔开的 5 个时间字段值（如果要指定年份，就会有 6 个字段）以及要执行的命令就可以设定任务。某个字段值上的星号（*）表示这个任务在这个时间间隔上每次都要执行，如表 7.1 所示。

表7.1　每天晚上08:00开灯的cron配置

0	20	*	*	*	/home/pi/on.sh 25
分钟（:00）	小时（晚上 08 点）	每天	每月	一周中的每一天	要执行的命令

如果只想在每周的周一至周五的晚上开灯，可以按表 7.2 设置 cron 表。

表7.2　每个工作日晚上08:00开灯的cron配置

0	20	*	*	1-5	/home/pi/on.sh 25
分钟(:00)	小时（晚上08点）	每天	每月	周一～周五	要执行的命令

如果想写个 Shell 脚本定期检查你是否收到新的信件，有就通

过电子邮件通知你，你可以用表 7.3 中的办法来设置一个每 5min 运行一次的任务。

表7.3　每5min检查一次邮件的cron配置

*/5	*	*	*	*	/home/pi/checkMail.sh
每 5min	每小时	每天	每月	一周中的每一天	要执行的命令

其中，*/5 就表示“每 5min”。

由此可见，cron 是一个非常强大的工具，你可以用它来指定在特定的日期或时间，或以特定的时间间隔来定期执行任务。

进一步学习

eLinux 的 Raspberry Pi GPIO 参考页面

（*http://elinux.org/RPi_Low-level_peripherals*）

这个页面上提供了有关 Raspberry Pi 的 GPIO 接口最为完整全面的信息。

Adafruit: 通过 MCP230xx 扩展 Raspberry Pi 的 GPIO 接口

（*http://learn.adafruit.com/mcp230xx-gpio-expander-on-the-raspberry-pi*）

如果你觉得 Raspberry Pi 上的 GPIO 接口不够用，Adafruit 提供了如何通过 MCP23008 芯片增加 8 个额外的 GPIO 接口或用 MCP23017 增加 16 个额外的 GPIO 接口的方法。