

第 12 章

硬件破解

在前面的几章中，你已经学到了如何把树莓派变成一个可以运行各种软件的平台。而在本章中，你将看到任何桌面或笔记本电脑可以运行的软件，即使是那些远超过树莓派低功耗处理器计算能力的软件，树莓派仍可以管理它。

树莓派有一项特殊功能（这一功能超越了普通的电脑能力）：在树莓派的印刷电路板的左上角有 26 个通用输入输出管脚。

GPIO 使树莓派能够与其他组件和电路通信，并允许它充当控制器再去控制一个更大的电子电路。通过 GPIO 端口，可以让树莓派探测温度，充当伺服系统并与其他移动计算设备使用包括串行外设接口（SPI）和内部集成电路（I²C）在内的各种不同的协议。

在开始搭建使用 GPIO 端口的电路之前，你还需要一些额外的设备。

12.1 电子元件

开始着手搭建可被 GPIO 端口控制的电路前，你需要各种元件和工具。下面的列表就提供了一个购物列表的示例：

- 电路板：电子实验电路板提供了间距为 2.54 毫米的带孔网格，元件可以在这些孔上插入和拔出。每个网格的下方是一系列的连接点，它允许同一行的元件在不需要线的情况下连接在一起。实验电路板是电子工作的一个宝贵工具，因为它允许你快速制作试验电路而无需焊接。
- 电线：尽管电线实验电路板可以让元件无需电线就能连接到一起，但你还是需要电线把一行连接到另一行，这些被称为跳线。如果你在实验电路板上使用的话，最好使用实心线而不是绞芯线。实心线更容易插入实验电路板的孔。实心线也方便买到各种颜色，使你能根据颜色为不同的连接编码。
- 电阻：电路中最常见的是称为电阻器的元件，本章中的示例项目也不例外。电阻的单位是欧姆，符号为 Ω 。在开始前最好准备一些常见阻值的电阻：2.2 k Ω 、10 k Ω 、68 Ω 。一些零售商会卖一些电阻工具包，其中包括

多种有用阻值的电阻。

- 按钮：一个非常常见的输入元件，当按钮被按下时一个完整的电路就接通了。在最基本的层面上，键盘就是多个按钮的集合。如果你正在为树莓派设计能提供简单输入的电路，就用按钮来完成开关吧。
- LED 管：发光二极管（LED）是最常见的输出设备。当有电压时，LED 灯点亮，让你对 GPIO 端口上的针脚是高电平还是低电平有个视觉感受。在为树莓派购买 LED 灯时，应当选择功率低的。GPIO 端口的电压不是那么大，对于那些大电流的 LED 灯，比如亮白或亮蓝的型号将需要外部电源，这将需要额外的称为晶体管的供电元件。

此外，如果你打算在完成了实验电路模型后做一些永久电路（本章将在后面介绍这些），你还将需要以下这些东西：

- 条纹板：可以被认为是一次性的实验电路板。同实验电路板一样，孔分布在一个间隔为 2.54 毫米的网格上。与实验电路板不同的是，元件需要被焊接到连接的地方，然后你就有了一个永久的电子电路。
- 电烙铁：当你需要将元件永久地连接到电路时，你就需要焊接。你不必花巨资购买一个电烙铁，但如果你的预算允许，最好买一个温度可控的型号，这会是一个明智的投资。确保你买的电烙铁是有一个小尖的，锥型头的电烙铁不适合精密的电路焊接操作。
- 焊料：你的电烙铁焊接需要焊料。焊料是一种混合物，它混合了称为清洗物质的导电金属。确保你买的焊料是适合电路焊接的，太粗的或者用于管道焊接的焊料虽然便宜，但可能会损害脆弱的电路，因为它需要更多的热量去融化。
- 支架和海绵：热烙铁在不使用的时候需要有地方放置，同时在使用电烙铁的时候还要清洗它的尖端。一些电烙铁附带了一个支架和清洁海绵，如果没有，你需要单独买一个。
- 侧铣刀：插孔元件会有长脚，这会使你在焊后留出多余的长脚。侧铣刀能让你简单快速地修剪这些多余的脚而又不破坏焊点。

- 镊子：电子元件通常又小又精密，一个好的镊子是必需的。如果你想使用表面贴装元件，而不是去焊接，那么镊子是绝对必要的。没有镊子，如果你尝试用手拿着元件去焊接，你可能会烧伤你的手指！
- 工作台：通常称为帮手，它是用来在上面摆放、夹取并焊接元件的。有些工作台还包括了放大镜，而一些好的工作台会有台灯来照亮工作区。
- 万用表：万用表是一种有多个功能的测试仪表，能测量电压、电阻和电容，以及测试电路连通性。虽然万用表不是绝对必要的，但它在诊断电路问题时是极其有用的。专业万用表单位是相当昂贵的，但是一个简单的型号却是相当便宜的，而且是每一个人都值得买。
- 拆焊芯：焊接出错，是指一些不是永久性连接的地方被连接了。拆焊芯是一种编织金属磁，它可以放在一个焊点上融化并把焊锡带走。实践使用中，可以用拆焊芯回收废弃电子设备的元件，这是一个简单而又便宜的收集常见元件的方法。

12.2 解读电阻颜色编码

大部分的电子元件都会被显著地标注。例如，电容器的电容会将法拉直接印在上面，晶体的频率也会同样地被印上去。

电阻是一个例外，在它上面通常不会有标注。相反的，可以从电阻表面雕刻的颜色带来计算欧姆值。学习这些解码是一个硬件黑客的重要技能，因为一旦从包装里拿出来，算出它的阻值的唯一方法是使用万用表，但这又是一个麻烦而又缓慢的测量工具。

还好，电阻器颜色编码遵循一定的模式。图 12-1 显示了典型的四段电阻。此图的高分辨率彩色版本在树莓派用户指南网页上可以找

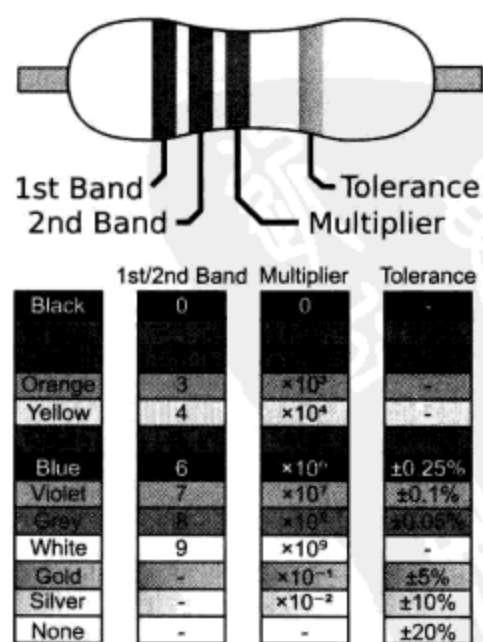


图 12-1 一个四色段的电阻和电阻的颜色编码表

到, <http://www.wiley.com/go/raspberrypiuserguide>。前两个色段等于欧姆的电阻值的颜色。第三个色段是乘数, 其中前两个数字相乘, 得出实际的电阻值。最后一个色段表示电阻的误差, 或者说是精度。低误差电阻比高误差的电阻更接近其标记的阻值, 但是你会为此花更多的钱。

示例电阻是这样读的, 首先读左起的两个色带, 它们被标记为红色和红色。红色, 在图 12-1 的表中, 等于 2, 因此最初读到的是 22。接下来的色是绿色, 即乘数相当于 10^5 或 10 万。22 乘以 10 万等于 220 万, 这个电阻的欧姆值就是 220 万 Ω 。

还有五色段电阻。和四色段电阻的读法类似, 就是前三个色段表示的数字, 第四个是乘数, 第五个是误差。

12.3 采购组件

如果你以前没有涉足电子产品, 你会很难弄到元件和工具。幸好, 有大量的在线和离线供应商, 专门用来买到完成项目所需的难以找到的元件。

12.3.1 在线零售商

两个世界上最大的元件和工具零售商是 RS 和 Farnell。它们在世界各地都有办事处和仓库, 也都有大量硬件可供选择。你很可能至少会熟悉两个零售商之一。在写这篇文章的时候 RS 和 Farnell 是唯一两家被授权生产树莓派硬件的公司, 所以如果你在读这本书时正在考虑购买树莓派, 你得向这两家零售商中的一家订购。

RS 和 Farnell 主要经营企业间交易, 这意味着它们的主要收入来源是从电子公司购买大量零部件。不过, 他们会很乐意允许消费者在注册了各自的网络店铺后订购小到单个元件。

当你在订购小订单时, 请注意可能会有其他费用。RS 会为第二天到货服务收费, 对于大订单来说, 收费是非常合理的, 它会大大超过小额订单购买的组件的费用。另一方面, 第二天到货服务 Farnell 是不收费的, 但订单的总额应大于网站

规定的最小值。

取决于你住的地方不同，你可以在 RS 或 Farnell 上将购买的东西凑成一个订单。这节省了邮费并让你更快收到货物，但也有可能的是你住的地方在选项里没有。零售商的网站可以在以下位置找到：

RS Components: <http://www.rs-online.com>

Farnell: <http://www.farnell.com>

12.3.2 离线零售商

你也许会发现你立即需要一个元件，而隔天到达的快递又不够块。也许你可能只需要一个电阻或一个小小的线材，并且金额又不能达到任何在线零售商之一的最小订单金额。谢天谢地，有专门经营电子组件的实体店铺。虽然这些店铺在过去几十年中都不多，但大多数主要城镇和城市里至少包含一个实体店，而且都备有最常用的工具和元件。

在英国，最受欢迎的电子元件商店是 Maplin 电子。于 1972 年在 Essex 成立以来，公司已从一个小的邮购商店发展到了目前包括在英国各地开设的 187 家门店。大多数城市居民都能找到 Maplin 电子商店，如果你缺一些常用部件，它可以是一个方便的来源。

Maplin 电子还提供了通过其网站邮购以及单击预订的服务，但要注意其业务是针对消费者的。从 Maplin 购买许多元件比从 RS 或 Farnell 购买要贵很多，因为该公司是依赖于预订而非大批量采购来盈利的。

在美国和其他国家，Radio Shack 是最受欢迎的电子产品连锁商店。它成立于 1921 年，号称在世界各地有 7150 多家门店，公司拥有常见的电子元件和工具，个人通过其网站可以购买或订购。

和 Maplin 电子在英国一样，Radio Shack 是以 B2C 方式经营的。其结果是，买家在从 Radio Shack 购买时会比从 RS 或 Farnell 订购花更多的钱。然而，大量的 Radio Shack 商店使买家在购买急需物品或一次性物品的时候更方便。

Maplin 和 Radio Shack 都有具备能与你直接交谈的员工，这是它们的一项优势。两家公司的员工都具备电子产品方面的知识，乐意提供意见并在你不确定需要什么部件时为你提供帮助。零售商的网站可以在以下位置找到：

Maplin Electronics: <http://www.maplin.co.uk>

Radio Shack: <http://www.radioshack.com>

12.3.3 业余爱好专家

除主要的连锁商店外，还有业余爱好者在其中工作的小公司。虽然他们与那些大型连锁店相形见绌，但他们都是公司以外的高手并能够提供个性化的建议。

Arduino 随许多爱好商店兴起，它旨在建立一个开放源码项目教育型单片机样机平台。树莓派非常欢迎像 Arduino 一样的公司（尽管目的不同），大多数调查显示除了支持树莓派外也支持其现有的产品。

从爱好专家那儿购买有几个优点。如果产品是作为树莓派的产品销售的，那么他们已经针对具体用途进行了测试。一些公司还针对不同的平台设计自己的扩展硬件，当然树莓派也不例外。这些设备，是为满足共同需要而设计的，它可能包含了其他端口和额外的硬件，或者拥有可以扩展目标设备的功能。

在英国，最受欢迎的一个爱好专家是 oomlout。它由 Arduino 硬件的开源爱好者成立，它是一个扩展工具包以及包括推按钮、显示和晶体管在内的常见元件的很好来源。与大型零售商不同，oomlout 的元件提供一切必要的额外附件（例如方便按钮组装的上拉电阻）在一些可能的地方，也会尽可能地提供示例源代码，让你更快地使用。

在美国，Adafruit 提供类似的服务。它是为了给 Arduino 电路板提供开源插件而成立的，Adafruit 提供了多种选择，包括第一个专门为树莓派主板设计的扩展组件和工具包（更多详细信息见第 13 章）。

它们的网页可以在下面找到：

oomlout: <http://www.oomlout.co.uk>

Adafruit: <http://www.adafruit.com>

12.4 GPIO 端口

树莓派的 GPIO 端口位于印刷电路板的左上方，标有 P1。这是一个 26 管脚的端口，装有两行 13 个 2.54 毫米公头。这些头间距尤为重要，引脚间距 2.54 毫米（0.1 英寸）在电子产品中是非常常见的，也是包括 stripboards 和实验电路板在内的标准间距。

GPIO 端口的每个引脚都有其自己的用处，几个针脚一起工作，就能形成特定的电路。

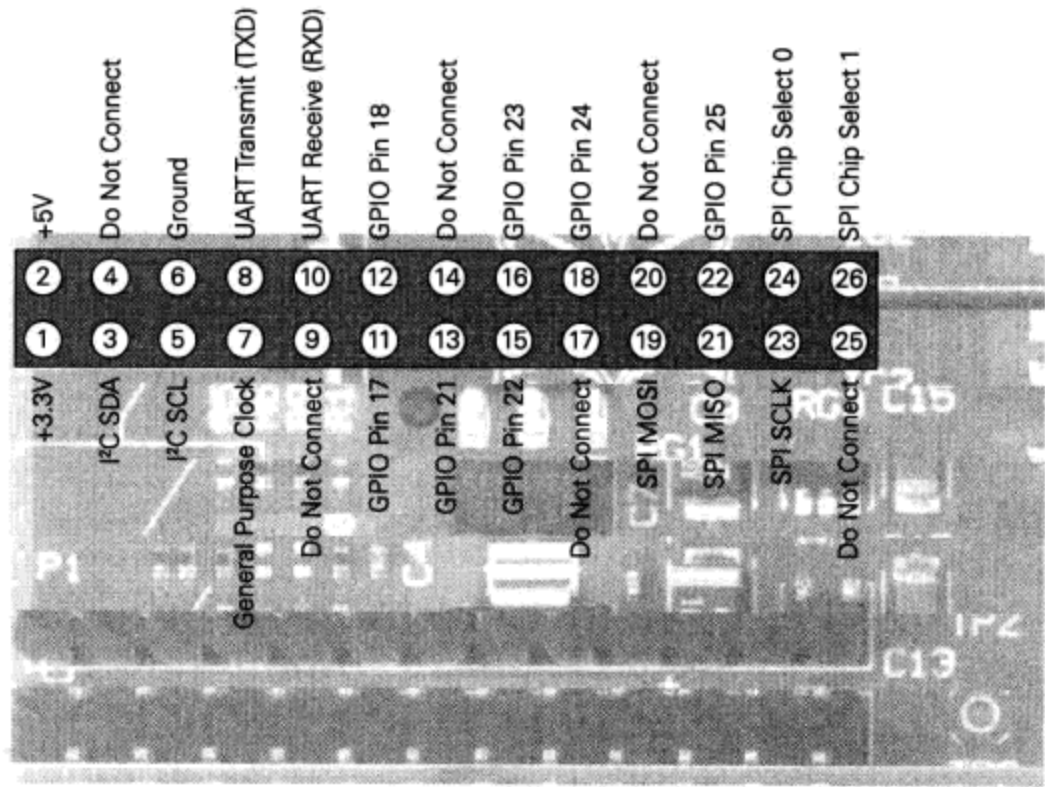


图 12-2 树莓派的 GPIO 端口及其定义

GPIO 端口的针脚数分为两行，奇数行在上，偶数在下。在使用树莓派的 GPIO 端口时，记住这一点很重要。大多数其他设备使用不同的针脚编号，同时因为树莓派本身无标记，因此很容易混淆针脚。

警告 不要连接标记为 Do Not Connect 的针脚，这些是留给树莓派的 BCM2835 系统芯片（SoC）的内部功能用的。连接任何的这些针脚都会造成损坏。

虽然树莓派 GPIO 的 pin2 端口提供了 5V 电源，从微型 USB 集线器获得的电源，但树莓派的内部工作电压是 3.3V。这意味着树莓派上的组件 3.3 V 电源供电的。如果你打算创建一个接口的电路与树莓派的 GPIO 端口连接，请确保你使用的是使用 3.3V 逻辑电路兼容的组件或在电路连接到树莓派前，通过电压调节。

警告 5V 电源连接到树莓派 GPIO 端口的任何管脚，或连接电源引脚做空（pin1 和 pin2）到任何其他 pin 将导致损坏。由于端口直接连接到高通 BCM2835 芯片的针脚处理器，你将无法修复任何你所造成损害。在 GPIO 端口附近操作时，必须格外小心。

GPIO 端口提供了 7 个通用管脚针：pin11、pin12、pin13、pin15、pin16、pin18 和 pin22。此外，pin 7，默认提供时钟信号也可以做一般用途，也就是共 8 个管脚。这些针脚可以在两个状态之间切换：正电压 3.3V 的高电平，等于地面的位置或 0 伏特的低电平。这相当于 1 和 0 的二进制逻辑，并可用于打开或关闭其他组件。以后你会在本章的后面有更多的了解。

警告 树莓派的内部逻辑工作在 3.3V。这与许多常见的微控制器设备不同，如流行的 Arduino 及其变种，它们通常运行在 5V。Arduino 电路设计的设备可能无法与树莓派同时使用，除非使用了电平转换器或光隔离器。同样，单片机的 5V 管脚直接连接到树莓派 GPIO 端口将不工作，甚至可能会造成永久的损坏。

除了这些通用的针脚，GPIO 端口还有特别的总线专用管脚。以下各小节介绍了这些总线。

12.4.1 UART 串行总线

异步接收器/发送器（UART）串行总线提供了一个简单的两线串行接口。当串口在 `cmdline.txt` 文件中被配置后（如第 6 章所述），这个串行总线就被用作传递消息。把树莓派的 UART 串行总线连接到一个能显示的设备上就能显示来自 Linux 内核的消息。如果你在启动树莓派时遇到了麻烦，这可以作为一个方便

的诊断工具，尤其当显示器不显示任何东西时。

UART 串行总线 pin8 和 pin10 都能被访问，pin8 发送信号，pin10 接收信号。发送速度可以在 `cmdline.txt` 文件中设置，通常是 115200 位/秒 (bps)。

12.4.2 I²C 总线

顾名思义，内部集成电路 (I²C) 总线是为多个集成电路 (IC) 之间提供通信用的。在树莓派中，这种集成电路的核心是 Broadcom BCM2835 芯片处理器系统。这些管脚包括位于树莓派上的上拉电阻器，这意味着不需要外部电阻就能访问 I²C。

I²C 总线能通过 pin3 和 pin5 访问，pin3 提供串行数据线 (SDA) 信号，pin 5 提供串行时钟 (SCL) 信号。I²C 总线仅有由 BCM2835 芯片提供的两个中的一个可供使用，它被称为 I²C0。第二个，I²C1，是被树莓派电路板的电阻中断的，不能用于一般用途的使用。

12.4.3 SPI 总线

串行外设接口 (SPI) 总线是一种同步串行总线，主要用于微控制器和其他设备的系统编程 (ISP)。与 UART 和 I²C 总线不同，这是一个四线总线与多个芯片选择线路，使它能够与多个目标设备进行通信。

树莓派的 SPI 总线是 pin19、pin21 和 pin23，及一对 pin24 和 pin26 上的芯片选择线。pin19 提供 SPI 主输出、从输入 (MOSI) 信号，pin21 提供 SPI 主输入、从输出 (MISO) 信号，pin23 提供了串行时钟 (SLCK) 用于同步的通信，pin24 和 pin26 提供两个独立的从设备的芯片选择信号。

尽管树莓派的 BCM2835 片上系统处理器存在着其他的总线，但它们没有被引到 GPIO 端口，因此无法使用。

12.5 通过 Python 使用 GPIO 端口

学习了理论之后，是时候去实践了。本节中，你将学习如何安装库以能够方

便地通过 Python 来使用树莓派 GPIO 端口上的针脚。你还会看到两个简单的电子电路演示如何使用 GPIO 端口输入和输出。

如你在本书的第 11 章中所见, Python 是一个友好而功能强大的编程语言。但它不是在每个情景都能作为一个完美的选择的。虽然在本章搭建的简单电路里它工作正常, 但它并不建议在所谓的确定性实时操作中使用。对于大多数用户, 这并不重要。如果你打算把树莓派用在核反应堆核心或复杂的机器人平台, 你也许需要研究使用一个较低级别的语言, 如 C++, 甚至汇编程序, 运行在专用实时微控制器。

如果你的项目确实需要真正的实时操作, 树莓派可能是一个不错的选择。相反, 可以考虑使用单片机平台, 例如流行的开源的 Arduino, 或一个来自德州仪器的 MSP430 系列微控制器。这些设备都可以通过 GPIO 或 USB 与树莓派交互, 并提供专业的实时控制和传感的环境。

12.5.1 安装 GPIO 的 Python 库

自树莓派发布以来, 许多开发人员创建了称为库的软件模块来充分利用其各种功能。特别是, 实际上, 程序员已经解决了树莓派访问 GPIO 端口的需求, 而不必知道底层编程。

就像在第 11 章中描述的 pygame 那样, 这些库旨在扩展 Python 语言的功能。安装这些库使 Python 能够轻松访问树莓派的 GPIO 端口, 虽然这意味着有人打算使用你创建的软件还必须在开始前下载并安装库。

尽管可以通过 web 浏览器下载 Python 库, 但很明显通过终端安装更快。请按照下列步骤操作:

1. 从树莓派的附件菜单打开一个终端窗口, 如果没有加载桌面环境就使用控制台。
2. 输入 `wget http://raspberrypi-gpio-python.googlecode.com/files/RPi.GPIO-0.2.0.tar.gz` 下载库到 home 目录。如果有新版本发布,

只需把版本号 0.2.0 换成当前版本。

3. 输入 `tar xvzf RPi.GPIO-0.2.0.tar.gz` 展开文件。命令是大小写敏感的，所以请确保输入大写字母。

4. 输入 `cd RPi.GPIO-0.2.0` 来切换到新建目录。如果你下一次下载了新版本的库，只需要更换所下载的版本号。

5. 输入 `sudo python setup.py install` 以安装 Python 库。

虽然 GPIO 库现已安装到 Python 中，但默认情况下不会加载它。就像 pygame，库需要显式地导入到你的程序。如要使用库，在程序开始输入 `import RPi.GPIO as GPIO`。在下列示例中你将了解更多。

树莓派 GPIO 端口不提供任何针对防止电压尖脉或电路短路的保护。

警告 始终确保你的电路在其连接到树莓派之前是安全的。如果可能，使用例如隔离板 Gertboard（在第 13 章中讨论）来提供保护。

计算限流电阻值

LED 需要限流电阻以防被烧坏。如果没有电阻，LED 灯将可能只能使用很短的时间就坏了，就需要更换 LED 灯。知道是一回事，但同样重要的是要选择合适阻值的电阻。太高的阻值会让 LED 灯非常暗淡或不发光，而太低的阻值会将它烧毁。

计算所需的电阻值，你需要知道 LED 灯的正向电流。这是 LED 灯在被烧坏之前最大的电流值，以毫安（mA）记。你还需要知道 LED 的正向电压。后者按伏特计，应为 3.3V 或更低。LED 灯需要外部电源和称为晶体管的开关设备。

计算所需阻值的最简单方法是用公式 $R = (V - F) / I$ ，其中 R 是电阻的欧姆值， V 是 LED 灯的电压， F 是 LED 灯的正向电压， I 是在 LED 灯正向电流的最大安培值（毫安的 1000 倍是安培）。

以一个典型的正向电流为 25 mA 和正向电压为 1.7V 的红色 LED 灯为例，

使用树莓派 GPIO 端口提供的 3.3 V 电源，可以计算所需的电阻为 $(3.3 - 1.7)/0.025 = 64$ 。因此， 64Ω 或更高的电阻会保护 LED 灯。这些数值很少与出售的电阻的阻值匹配，所以当你选择一个电阻时，总是选用更高阻值的电阻以确保 LED 灯得到保护。最接近可用的值是 68Ω ，这将充分保护 LED 灯。

如果你不知道你的 LED 灯的正向电压和电流（例如，如果 LED 灯没有附带文档或是被回收过来的），谨慎起见，选用一个相当大的电阻。如果指示灯太暗，你可以改小，但是坏掉的 LED 灯是无法修复的。

12.5.2 GPIO 的输出

在第一个示例中，你需要构建一个由一个 LED 灯和电阻组成的简单电路。LED 灯将提供可视化效果以确认 Python 程序做了你要树莓派 GPIO 端口所要做的事，电阻会限制 LED 灯的电流以防被烧坏。

组装电路时，你需要一个实验电路板、两根跳线、一个 LED 灯和恰当的限流电阻（如侧边栏“计算限流电阻值”中所述）。尽管可以在没有实验电路板的情况下通过缠线来装配电路，但实验电路板是值得去买的，它会使拆装原型电路更简单。

假设使用了一个实验电路板，电路按下列方式装配，如图 12-3 所示。

1. 将 LED 灯插入到实验电路板中，以便长脚（负极）在一行中而短脚（阴极）在另一行。如果你把 LED 灯的两根脚接到了同一行，它将不会工作。
2. 在 LED 灯短脚的同一行中插入电阻的一根脚，并将电阻的另一根脚接到一个空行。电阻的接脚的方向并不重要，因为电阻是一种非极性（不区分方向）的设备。
3. 使用跳线连接树莓派 GPIO 端口的 pin11（或连接 GPIO 端口相应 pin 到实验电路板上），使其和 LED 长腿在同一行。
4. 使用另一根跳线，连接树莓派 GPIO 端口的 pin6（或连接 GPIO 端口相应 pin 到实验电路板上）到包含电阻器而没有 LED 灯引脚的一行上。

警告

树莓派 GPIO 端口在连接电线时要非常小心。如本章前面所述，如果你连错接脚，你可能会对树莓派造成严重损害。

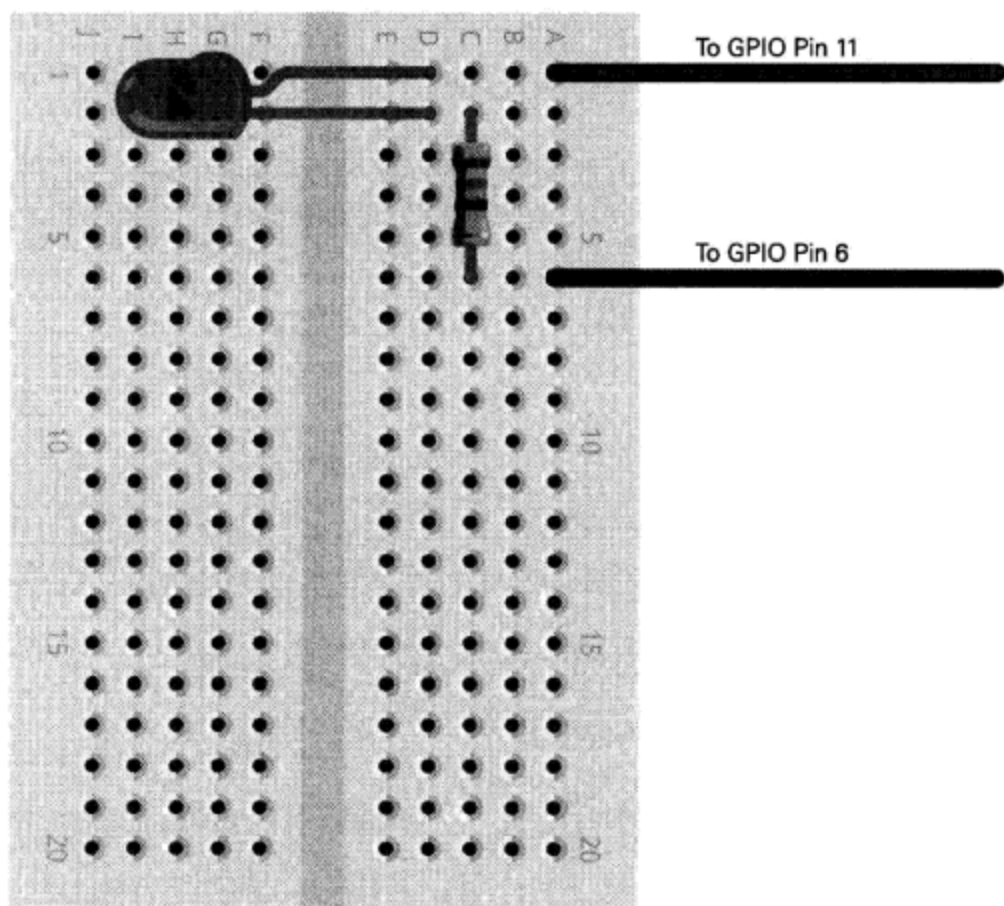


图 12-3 实验电路板的简单 LED 输出灯

此时，什么都不会发生。这很正常，默认情况下树莓派的 GPIO 管脚是关闭的。如果要立即检查电路，将线材从 pin11 连到 pin1 让 LED 灯亮起来。小心不要将其连接到 pin2，尽管限流电阻适用于 3.3V，但在连接到 5V 电源时，它不足以保护 LED 灯。记得在继续前移回 pin11。

要想让 LED 灯做点有用的事，不如开始一个新项目。如第 11 章中介绍的，你也可以使用纯文本编辑器或内置的 IDLE 软件在推荐的 Debian 发行版中开始项目。

在使用本章前面安装的 GPIO 库前，你需要将其导入到你的项目。因此，将以下文字作为文件的开始：

```
import RPi.GPIO as GPIO
```

记住，Python 是区分大小写的，所以请确保输入的是 `RPi.GPIO`。让 Python 可以使用时间的功能（换言之，使指示灯闪烁，而不仅仅是把它开启和关闭），你

还需要导入时间的模块。向项目中添加以下行：

```
import time
```

库导入后，是时候解决 GPIO 端口了。GPIO 库可通过指令 `GPIO.output` 和 `GPIO.input` 很轻松地使用通用端口。但是在使用之前，你需要初始化作为输入或输出的管脚。在此示例中，`pin11` 为输出，所以要将下面一行添加到项目中：

```
GPIO.setup(11, GPIO.OUT)
```

这告诉 GPIO 库，树莓派 GPIO 端口上的 `pin11` 设置为输出。如果你是控制其他设备，你可以添加更多 `GPIO.setup` 到项目中。但是这里，一个就够了。

在配置为输出管脚后，你可以打开和关闭其 3.3V 电源来演示一个简单的演示二值逻辑。指令 `GPIO.output(11, True)` 将打开管脚，而 `GPIO.output(11, False)` 则关闭它。管脚将保持最近的状态，所以如果你只给出打开管脚的命令然后退出你的 Python 程序，管脚将一直打开，直到收到其他命令。

虽然你可以添加 `GPIO.output(11, True)` 为 Python 项目将管脚打开，但使它不断闪烁会更有趣。首先，添加下面一行到程序中创建一个无限循环：

```
while True:
```

接下来，输入下面几行，先打开管脚，等待 2 秒钟，然后再将其关闭，再等待 2 秒。确保每行开头都有 4 个空格，表示它是无限 `while` 循环的一部分：

```
GPIO.output(11, True)
time.sleep(2)
GPIO.output(11, False)
time.sleep(2)
```

那么，完成的程序应当会像这样：

```
import RPi.GPIO as GPIO
import time
GPIO.setup(11, GPIO.OUT)
while True:
    GPIO.output(11, True)
    time.sleep(2)
```



```
GPIO.output(11, False)
time.sleep(2)
```



图 12-4 程序 gpiooutput.py, 在 nano 中编辑, 光标停在最后一行

将文件保存为 gpiooutput.py。如果你使用的是 Python 开发环境, 如 SPE, 不要尝试在编辑器内运行该程序。大多数树莓派的 Linux 发行版的 GPIO 端口使用必需使用 root 用户, 那么运行该程序将需要使用命令 `sudo python gpiooutput.py`。在终端就可以开始了。如果一切顺利, 你应该看到 LED 灯开始以一定时间间隔闪烁, 至此, 你已经创建了第一个在树莓派上的输出设备。

如果没有正常工作, 不要着急。首先, 检查所有连接。实验电路板孔非常小, 这使你很容易觉得已经将元件插入到某一行中, 但实际上却插在了另一行。接下来, 检查对 GPIO 端口上的管脚连接。树莓派上没有标签, 错误是很容易发生的。最后, 再次检查你的组件, 如果 LED 灯的正向电压高于 3.3 V 或限流电阻太大, 指示灯就不会亮。

尽管这只是一个基本示例, 但它很好地示范了一些基本概念。如果要扩展其功能, 可以用用来警报的蜂鸣器或机器人平台中的伺服电机来取代 LED 灯。用于激活和停用 GPIO 管脚的代码可以集成到其他程序, 用于在新邮件到达时点亮 LED 灯, 或当一个朋友加入了 IRC 频道时发出一个信号。

12.5.3 GPIO 的输入

使用 GPIO 作为输出无疑是有用的，但是如果你可以将其与一个或多个输入结合，它将变得更加实用。在以下示例中，你将了解如何连接一个按钮，在 Python 中用它来切换到另一个 GPIO 端口上的管脚并读取其状态。

同之前 LED 灯输出的示例一样，此输入的示例将使用 Python GPIO 库。假设你已安装了此库，你可以开始构建电路了。（如果你还没有安装 Python GPIO 库，跳回前面几页，然后按照说明安装。）

如果你已经建立了 GPIO 输出的示例，你可以断开其与树莓派的连接或让其继续连着，此示例使用不同的针脚，所以这两个示例可以很好地并存。如果你选择保留前面的示例连接，你要确保所使用的新元件连接在实验电路板的不同行上，不然你会发现不能正常工作。

按如下步骤搭建电路：

1. 将按钮开关插入实验电路板。大部分开关有两个或四个接脚。你只需要关心四根接脚中的两根。如果该按钮具有四条腿，它们会是成对的。检查推按钮的数据表以找出配对的接脚。

2. 将 $10\text{k}\Omega$ 电阻连接到按钮接脚的一行及一个未使用的行。这是一个上拉电阻，并为树莓派提供参考电压，以便当按钮被按下时可以被感知。

3. 将上拉电阻的未使用接脚连接到树莓派 GPIO 端口的 pin1。该管脚提供了 3.3V 的参考电压。

4. 树莓派 GPIO 端口的 pin6 连接按钮开关未使用的接脚。该管脚接地。

5. 最后，将树莓派 GPIO 端口的 pin12 连接到按钮开关 $10\text{k}\Omega$ 电阻另一接脚的那一行上。你的实验电路板现在看起来应该如图 12-5 所示。

在本实例中，你刚刚搭建的电路利用树莓派 GPIO 端口的 pin12 和连接到 3.3V 电源的上拉电阻，来让管脚保持高电平。当按钮被按下时，电路接地并呈低电平，使 Python 实例程序可以感知该按钮已被按下。

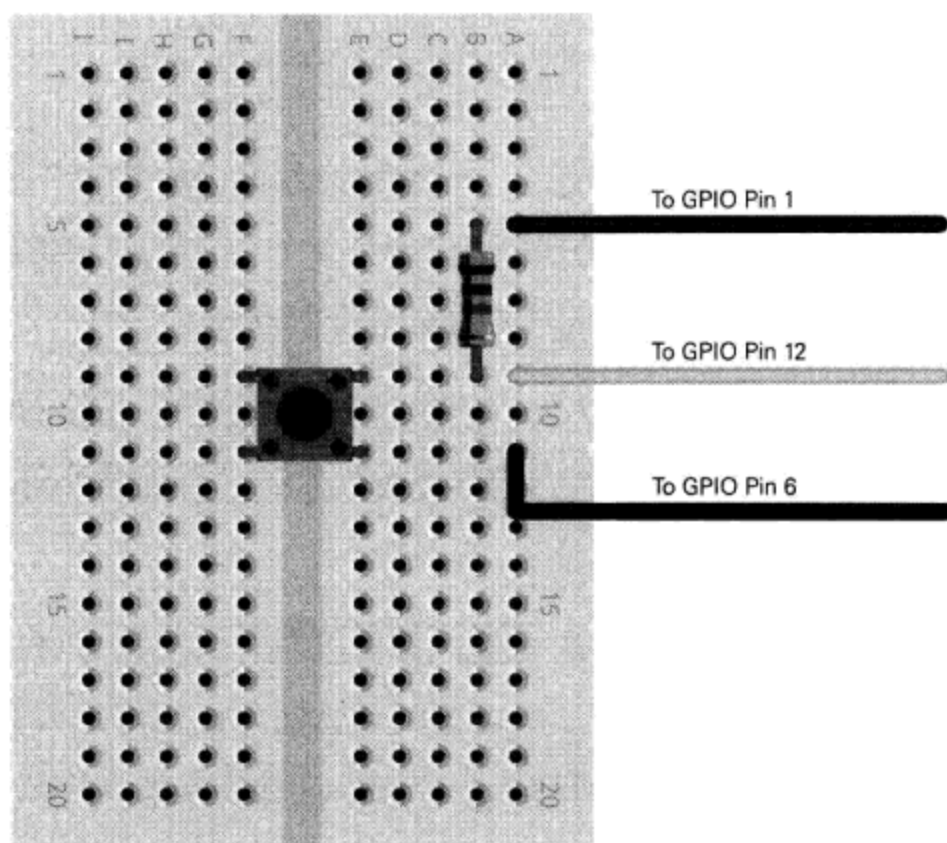


图 12-5 一个简单的按钮输入的示例布局

你可能会奇怪为什么电阻都是必需的，为什么开关不能简单地直接连接 `pin6` 和 `pin12` 或 `pin1`。尽管这是可行的，但这样会构建出一个无法知道是高电平还是低电平的悬空管脚。这样的结果是，即使按钮不被按下电路仍然接通，或者即使按钮被按下却无法被检测到。

在文本编辑器或在树莓派上已有的一个 Python 集成开发环境 (IDEs) 中打开一个新的 Python 文件。首先，跟之前的 GPIO 输出的示例一样，你需要导入相同的 GPIO 库：

```
import RPi.GPIO as GPIO
```

你不需要导入时间库，因为此示例不需要任何时间指令。相反，你可以设置 `pin12` 作为输入。这与在输出实例中设置管脚的方式类似，只是最后一部分要进行相应地指令更改：

```
GPIO.setup(12, GPIO.IN)
```

如果你在这里没有使用 `pin12`，则需要更改在前面指令中的管脚号。

与上一个示例一样，下一步是创建一个无限循环不断检查输入的管脚以查看

它是否一直是低电平（换句话说，是否被按下）。开始循环使用下列代码：

```
while True:
```

读取输入引脚的状态非常类似于设置输出引脚的状态，有一个例外，在你对结果做任何有用的改动以前，你需要将其存储在变量中。下面的指令告诉 Python 创建一个新的变量 `input_value`（如第 11 章所述）并将其设置为当前值 12：

```
input_value = GPIO.input(12)
```

虽然程序现在已经能工作了，但它做不了任何有用的事。为了让你知道发生了什么事件，添加下面的打印指令以获取反馈：


```
if input_value == False:
    print("The button has been pressed.")
    while input_value == False:
        input_value = GPIO.input(12)
```

最后两行第二个 `while` 和第二个 `input_value` 是一个循环，这很重要。即使是在相对高性能的台式机和笔记本处理器来说慢的低性能树莓派的处理器上，Python 运行速度也是非常快的。这个嵌入环告诉 Python 不停地检查 `pin12` 的状态，直到它不再是低电平，此时它知道该按钮已被松开。如果没有这个循环，程序会在按钮按下时不停循环而无论你的反应有多快，你会多次看到打印到屏幕上的消息，这是不正确的。

最后的程序看起来会是这样：

```
import RPi.GPIO as GPIO
GPIO.setup(12, GPIO.IN)
while True:
    input_value = GPIO.input(12)
    if input_value == False:
        print("The button has been pressed.")
        while input_value == False:
            input_value = GPIO.input(12)
```

将文件保存为 `gpinput.py`，然后从终端使用 `sudo python gpinput.py` 来执行它。开始时，什么都不会发生，但是如果你按下按钮，程序将把第六行的消息在打印到终端上（见图 12-6）。松开按钮，然后再按一次，消息将被重复打印。



```
pi@raspberrypi: ~$ python gpioinput.py
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
The button has been pressed.
```

图 12-6 程序 gpioinput.py 的输出

与先前输入的示例一样，这是一个看似简单的程序，但它可以用于多种用途。除了能够读取按钮是否被按下，相同的代码可用于读取独立设备的管脚（如传感器或外部微控制器）是否被拉高或者拉低。

通过扩展代码可以查看多个按钮，每个与独立的 GPIO 管脚连接，你甚至可以创建一个简单的 4 键游戏控制器。例如，你可以将上面的代码与第 11 章中的 Raspberry Snake 游戏结合把树莓派变成一个简单的游戏控制台。你还可以把输入和输出的例子合并到一个程序，等待按钮被按下，然后通过把输出管脚电平拉高来打开 LED 灯。你应当理解本节中的概念，现在尝试创建合并程序吧。如果你卡住了，或者你想检查你的方法，请参考到附录 A，这是一个示例解决方案。

12.6 在实验电路板上更进一步

如本章前面所示，用实验电路板搭建实验电路是很好的。它方便使用，可重复利用，还不损坏元件。

不过，实验电路板也有一些缺点。它们笨重且昂贵，而且连接很松散，这可

能会导致关键部件掉出来，特别是当把实验电路板从一个地方运到另一个地方。图 12-7 很好地展示了这一点，尽管尽了最大努力，但实验电路板的按钮也只是松散地连接在上面，如果用起来不注意就有可能掉下来。

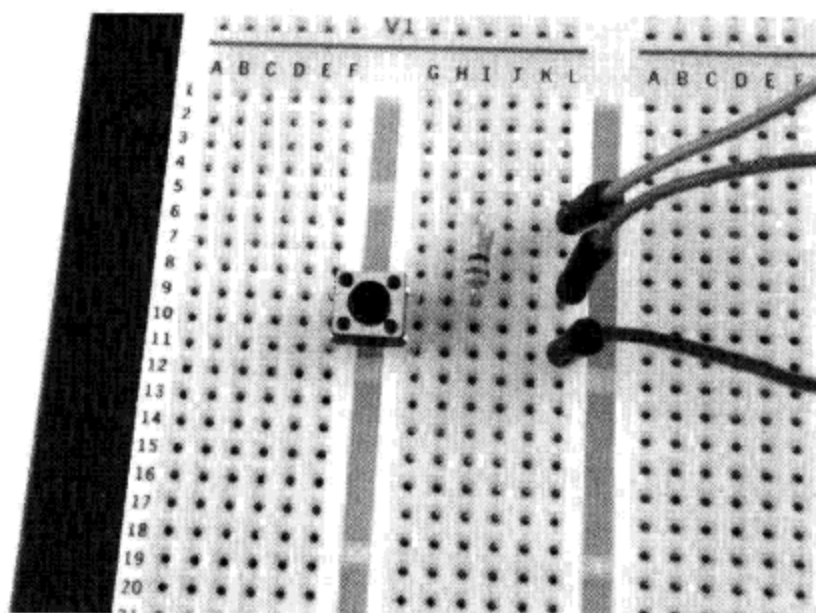


图 12-7 一个元件连接松散的实验电路板样例

这就是为什么树莓派本身是建立在印制电路板（PCB）上而不是实验电路板上的原因，尽管实验电路板在早期的原型设备中使用。尽管可以在家打印和蚀刻自己的电路板，但是有一个简单的中间步骤你可以采用，即使用插座电路板来创建永久的独立电路。

乍看之下，插座电路板类似实验电路板，因为它的表面覆盖着 2.54 毫米间距的小孔。与实验电路板不同的是，插座电路板有一种让电子元件放入并能留在小孔中的巧妙结构，你需要把元件焊接到插座电路板上。插座电路板在市场上通常称为 Veroboard，这是一个在英国和加拿大 Pixel Print 的 Vero 技术公司的商标。

搭建插座电路板电路相对使用实验电路板有许多优点。一片插座电路板相对一个同样大小的实验电路板要便宜许多，并且它还可以改小尺寸以使用更小的电路。它还允许在单个大型插座电路板中构建多个较小的、独立的电路。

由于元件是被焊接到插座电路板上的，这也就比实验电路板原型更耐用。插座电路板电路可以被完整地从一个地方带到另一个地方，而不必担心某一元件错位或丢失。图 12-8 显示了一块反转的插座电路板其底面上的铜轨。

插座电路板非常易于使用，它是设计和制造自定义电路板设计的基石。不过，在你购买插座电路板之前，你应该注意以下几点：

- 插座电路板的类型有很多。一些插座电路板的底面上的铜轨将整行或整列连接起来，而另一种插座电路板会在中间被拆分为两个单独的行，这有点像实验电路板。还有一种通常被称为项目板的插座电路板，根本没有铜轨，这就要求使用电线来连接元件。

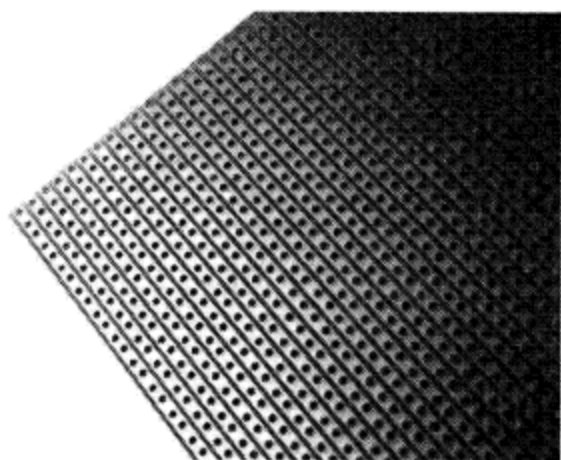


图 12-8 一块插座电路板底面上的铜轨

- 插座电路板可以被做成不同的厚度及使用不同的材料，而且一种类型的插座电路板可能比另一种类型的插座电路板更适用于某一特定项目。比如，在高温环境中，具有耐热功能的插座电路板是更适用的，而较厚的插座电路板则适用于不良环境。
- 为了让元件在插座电路板上布局整齐，可以切断底面上的铜轨来断开元件间的连接。这样既可以避免浪费板子的空间又能设计更复杂的电路。为了达到布局整洁的效果，你需要一个小的称为轨道切割机的手持工具。补充一句，虽然用小钻头也可以，但是如果你打算使用插座电路板，最好还是记得把它加进去。

这里还有使用插座电路板的一些技巧，你如果忽略了，可能会把事情弄得更麻烦：

- 插座电路板底面的铜轨面通常都没有涂层保护。如果你拿手碰到了铜轨，就有可能弄脏它，这会让焊接很困难。除非你打算立即用它，否则你应避免触摸插座电路板背面的铜轨。如果已经不小心碰了，那就需要在焊接之前弄些钢丝很小心地把铜轨上被腐蚀的那一层去掉。
- 与印制电路板不同，插座电路板没有阻焊膜（用来防止焊锡落到别的地

方。这样一来，焊接起来相对 PCB 要更棘手)。如果用一大块焊锡是很容易不小心把线路焊到一起的。如果发生这种情况，使用拆焊芯除去多余的焊锡，然后重新焊接。

- 插座电路板的孔可以容易地对齐到需要的大小，而不是让边缘参差不齐。在对齐插座电路板之后，在花时间装配之前，把你的电路的边缘修整一下。确保你是戴口罩做这件事的，因为从插座电路板上掉下来的灰尘不利于健康。

12.7 焊接简介

有一个电烙铁之后，你还要知道如何使用它。与其他任何技能一样，焊接也需要通过练习才能熟练。如果你按照本节的小提示不断练习，你就能把焊点做得干净而又整齐。

警告

显而易见但又必须指出的是：在使用期间，焊接铁会变得非常热。最好不要碰任何接触电烙铁的金属表面，并远离其尖端。如果可能的话，买一个支架，或去弄一个耐热的支架。不要在使用中离开，如果你把电烙铁不小心跌落，千万不要试图去抓住它！

焊接就是通过熔化少量金属来连接两个元件。如果你把树莓派反过来，你就会看到，所有较大的元件间的连接使用的是所谓通孔焊接，也就是元件的引脚插进印制电路板的孔然后进行焊接。而较小的元件则是通过表面安装焊接来连接的。

焊锡不是纯金属，它还包含一种称为助熔剂的物质，它旨在去除表面上的任何脏物，以确保尽可能干净地进行焊接。大多数电子焊料包含了 3 至 5 芯焊剂。你也可以单独购买糊状或液态的焊剂，虽然这对于大多数业余焊接作业不是必要的。

当你开始焊接时，确保你有一个干净、明亮的工作区，还要确保良好的通风。焊接烟雾不利于健康，虽然它们在平常少量的焊接作业中达不到危险的水平，但最好还是保持最低程度的接触。

此外，你应该设法保护工作台。滴落的熔化的焊料会烧坏桌面。你可以购买防静电工作垫（如图 12-9 所示），但表面有光泽的杂志也能用。别想用几张便宜的报纸将就，因为焊锡在冷却之前仍能烧穿报纸。



图 12-9 焊接工作区和一块防静电工作垫

如果你做的是精致且需要仔细观察的活，你就应该戴防护眼镜。有时沸腾焊锡会向上溅起，如果它落到你的眼睛里，你将活在一个痛苦的世界里。

但也别让这些警告令你不敢去焊接。虽然焊料是很热的，但它冷却很迅速，同时，燃烧鲜有发生并且是无足轻重的。需要留意设备，但也不必害怕。

在选择了你的工作台并加以保护后，需要对设备进行合理摆放。烙铁应放在你的顺手的一边，并让它的电缆不会落在你的工作区。在插入烙铁以前你要确保烙铁可以自由移动。如果电缆被什么东西缠住了，就可能会燃烧你自己。

你的焊接海绵应该打湿，但也不要弄得湿漉漉的。这很重要，潮湿的海绵将用来清洁烙铁，但如果是干的，它会被点燃并可能弄坏烙铁的尖头。

让烙铁达到工作温度将花费几分钟的时间。如果你买了带温度控制的烙铁，它上面通常有一个在 on 和 off 之间切换的指示灯来指示温度是否已达到或是一个温度读数。（请参阅你的烙铁附带的手册，了解如何读取烙铁的温度。）

一旦达到工作温度，就可以开始用称为焊锡的东西焊接了。请按照下列步骤操作：

1. 把焊锡推向烙铁尖，让少量焊锡在烙铁上熔化。小心不要熔化得太多，这

不仅浪费了焊锡，而且会让过多的焊锡落到工作区上。

2. 用海绵擦拭烙铁。如果发出嘶嘶声并挤出水，那说明海绵太湿了。待海绵冷却然后将其从支架里拿出来，将水挤掉。

3. 不断擦烙铁尖，直到上面被覆盖了一层银焊锡（见图 12-10）。如果有必要，在烙铁尖上用更多的焊料。

在烙铁尖上熔锡可以防止其被损坏，并能让它有效地将热量传递到表面。未能正确地把焊锡放在烙铁尖是造成一个坏焊点的最常见原因。如果你需要焊接很多的话，你可能需要重复这个过程许多次，你会在每次焊接完成后都重复以上工作。一般来说，如果电烙铁尖失去光泽涂层，就应该重复熔锡。

烙铁准备好后，就开始焊接了。把要焊接的东西（例如印刷电路板和元件的管脚）放在工作台上并确保你有一个好的视野。从容器中拉出一段焊料，并开始按以下步骤焊接元件：

1. 如果你在印刷电路板、插座电路板或类似的过孔板上焊接穿孔元件，将元件的管脚穿过孔后向外弯曲，这样当板子反过来的时候元件就不会掉出来了。

2. 把主板固定在站上，把烙铁尖按在元件和主板的铜触点上。烙铁必须同时接触这两样东西，如果烙铁仅接触了其中一个，那最后完成的连接点将很糟。

3. 只需要几秒钟就能把接触的地方加热。数三下，然后把烙铁尖按在元件和主板的铜触点上（如图 12-11 所示）。如果焊料不融化，收回来，再算上几秒钟，然后再试一次。如果它仍然没有融化，试着给烙铁换个位置。



图 12-10 电烙铁尖上作为焊料的镀锡

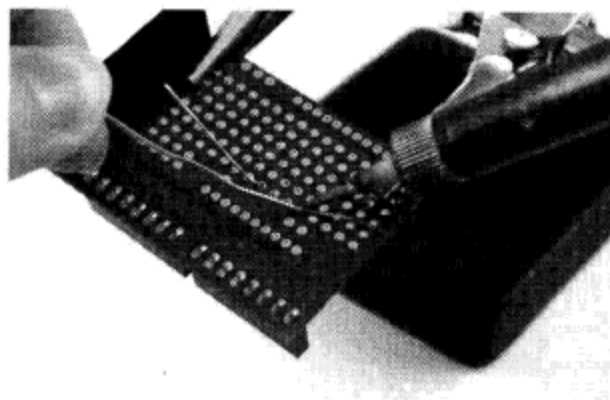


图 12-11 在印制电路板焊接穿孔元件

4. 熔化的焊锡是会流动的，你因此会看到它落进板子上的孔中。这表明该区域受热足够并将是一个好的焊点。如果焊料只是浮动，这表明该区域受热还不足够。

5. 先移开焊锡，然后移开烙铁。（如果你先移开烙铁，焊锡会变硬，把你的金属焊料固定在焊点上！）

如果一切顺利，你会得到一个能用很多年的坚实焊点。如果没有，别气馁，把烙铁重新按在焊点上让焊料熔化，如果需要清理泄漏或多余的焊料就使用拆焊芯。一个完美的焊点应该形状有点像火山，从板子的表面向元件的接脚处不断上升。

不要让烙铁与零件接触太久，这点特别重要。在焊接如集成电路这些不耐热的元件时，这可能会因为烙铁的长时接触而损坏。如果你使用的是带温度控制的焊锡台，在使用中请确保温度被设置到一个适当的水平（查看焊料的包装或数据表）。

当你完成焊接时，记得把烙铁尖的焊锡去掉。不然烙铁可能会在存储过程中严重腐蚀而导致需要被过早更换。

记得加热烙铁的两面。只加热一面的结果会导致所谓的干接头或冷接头，即焊料没有与表面很好地结合。随着时间推移，这些焊点将失效而需要重新焊接。

正如任何技能一样，焊接需要多加练习。许多电子商店销售包括印刷电路板和精选元件在内的套件，你可以使用它来练习穿孔焊接。树莓派的一些扩展主板也提供了需要焊接的套件形式，如 Ciseco Slice of Pi。你将在第13章中了解有关这些的更多内容。

