

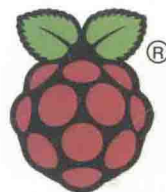
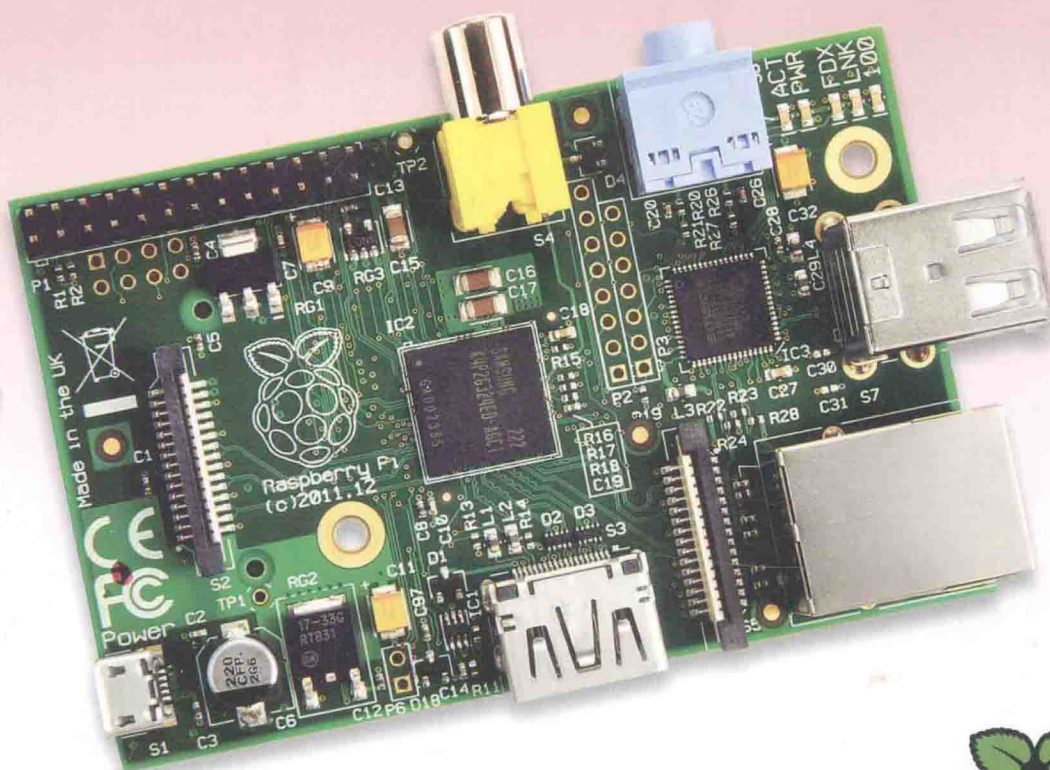
Raspberry Pi 实战攻略

【英】Gray Girling 著 翁恺 译

Owners' Workshop Manual

革命性的微型计算机实用指南

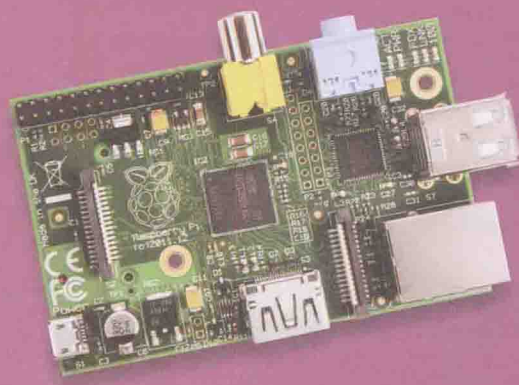
树莓派基金会创始人Eben 和Liz Upton作序推荐



人民邮电出版社
POSTS & TELECOM PRESS



爱上开源



Raspberry Pi 实战攻略

【英】Gray Girling 著
翁恺 译

人民邮电出版社
北京

图书在版编目(CIP)数据

Raspberry Pi实战攻略 / (英) 格林 (Girling, G.)
著; 翁恺译. — 北京: 人民邮电出版社, 2014.6
(爱上开源)
ISBN 978-7-115-35329-0

I. ①R… II. ①格… ②翁… III. ①Linux操作系统
IV. ①TP316.89

中国版本图书馆CIP数据核字(2014)第082011号

版权声明

Originally published in English by Haynes Publishing under the title: The Raspberry Pi Manual written by Gray Girling © Gray Girling 2012

内 容 提 要

在 Raspberry Pi (中文名: 树莓派) 基金会创始人 Eben Upton 的建议下, 博通公司工程师 Gray Girling 博士撰写了本书, 在书中为众多树莓派用户提供了独一无二的相关资料, 这是一本易懂而准确的关于树莓派应用的指导手册。

如果你有一块树莓派, 想让它顺利工作起来; 编写出你的第一个“贪吃蛇”的游戏, 甚至让它成为你要打造的全功能媒体中心的核心, 那就一定要阅读本书。本书中那些丰富而详尽的树莓派编程实例及外围设备拓展方法, 将让你成为树莓派应用的高手。

本书不仅适合树莓派用户阅读, 还非常适合高校相关专业师生阅读, 帮助他们从一个崭新的角度来学习计算机硬件及软件编程的相关知识。

-
- ◆ 著 [英] Gray Girling
译 翁 恺
责任编辑 房 桦
责任印制 周昇亮
- ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京精彩雅恒印刷有限公司印刷
- ◆ 开本: 787×1092 1/16
印张: 14
字数: 325千字
印数: 1—3 000册
- 2014年6月第1版
2014年6月北京第1次印刷
- 著作权合同登记号 图字: 01-2013-8479号
-

定价: 69.00元

读者服务热线: (010) 81055339 印装质量热线: (010) 81055316

反盗版热线: (010) 81055315

广告经营许可证: 京崇工商广字第0021号

前言

7年前，当时能买到的最长的一块洞洞板摆在我们的厨房餐桌上，旁边还有几瓶家庭自制的果酱。我把那几瓶黏黏的果酱拿开，可以看到那块洞洞板上立着一些晶体管，还有一块Atmel的芯片，那是Eben做得非常干净的手工焊接。这块板可以在老式的模拟电视机上产生非常简单的3D图形，仅此而已。那时候叫作ABC Micro的这个原型，是长长的硬件迭代演化中的第一个。随着硬件的不断演化，目前这块叫作树莓派的板子已经让我们感觉很陌生了。

今天，我们意外地发现自己在做着突然闪现的数以百万的计算机生意：在开卖的日子对付所有的供应商的网站，为了应付这一年的需求拼命奔波，累得喘不过气来。我们要帮助构建全新的学校课程体系，就用今天你手里握着的这块硬件：一个完全成熟的Linux PC，小到可以放进口袋，便宜到用零花钱就可以购买。不像2006年的餐桌型号，今天的树莓派可以播放和记录蓝光质量的高清视频，在任何电视机上都可以，HDMI或模拟的都可以；它可以运行你“丢”进去的任何Linux编程语言程序。它可以和真实世界交互：人们用他们的Pi来控制酿酒和烘焙时的温度和压力，用手机来开启车库的门，从高空的氢气球上发送图片回来，构造车载计算机，监视婴儿，为祖父母做一键播放的有声读物。要是你没什么想法，也可以只是用它来看电视，或者学习如何编程。

在写本书的时候，距离开始发售树莓派才不到一年的时间。我们所宣称的目标始终是用它

来鼓励孩子学习编程，让任何可能有兴趣的孩子都能用这个工具，让它廉价而且容易获得。在我们三十多岁的时候，我们遇到的孩子和我们不同，他们不像我们在20世纪80年代那么容易可以接触到编程，那时候我们有BBC Micro和ZX Spectrum（就像在中国20世纪80年代中期的基于Z80 CPU的TP-801单板机和R1学习机、Laser 310电脑，以及后来的仿Apple II的“中华学习机”）。用于编程的机器的市场被从两端吃掉了。在上端，它们被家庭PC弱肉强食了。家庭PC在家里有一个中心地位：爸爸妈妈要工作、处理银行事务和访问互联网。很少有家长会让他们把孩子把家里PC的机箱打开，捣鼓里面的东西。而从下端，许多这样的20世纪80年代的机器被单一用途的游戏机给挤占了。游戏机生产出来明显是不可编程的，他们要你购买游戏放进去，而不是自己免费做一个游戏进去。在游戏方面他们很成功（我家的电视机下面就有两个），但是对于很多家庭而言，其实那是家里唯一能让孩子认识到这是一台计算机的东西。我们想要做一个东西，不想做手机或平板的东西，重点是在创造而不是消费。我们想要做一个东西，用非常少的钱，就能有游戏和很棒的多媒体能力，以此来吸引孩子；同时我们希望能展现出这样的一种潜力，孩子们只要用这个东西，就能做他们自己各种想做的东西。

当然，不是只有孩子在购买派，虽然我们估计大约20%已经销售的树莓派最终是在孩子手里的。我们从来，从来没有能想象到反响会

是如此巨大、如此热烈，而且如此出乎我们意料得精彩。这其中，有人花了时间做了一件家具，包括一个华丽的白杨木架子、6个甜菜根和一个树莓派。每次有人抚摸某根甜菜，就会发出美妙的音乐，你很快就会发现自已与生俱来就有根据形状记住曲谱的能力。我们仍然坚守着最初的目标：这一年，树莓派基金会和大量其他机构一起，要把糟糕透顶的 ICT（中小学的信息与通信技术）课程体系替换掉。那套现在已经不存在了的课程体系，把整个一代孩子对机器计算的全部兴趣整成了几个电子表格和 PowerPoint 幻灯片。我们正在踏实前进，要给孩子看到计算不是数学，而是创造、游戏，是监视父母的传感器，是算命盒子，或是演奏音乐的蔬菜根。我们在向众多而且越来越多的教师和教育工作者宣传，我们在和课外机构及父母团体合作。

树莓派目前所取得的成功，主要归功于围绕着这个板子建立起来的圈子。我们在 www.raspberrypi.org 的论坛，每分钟就能进来一个帖子，在那里，新手、过去的电子爱好者和这两种人之间的每个人，都有话要说：我们保证如果有什么你做不来的，这里有人会在一天之内告诉你该怎么做。这里有经营着像是树莓酱全球网络这样的企业的人，树莓酱是新人和有经验的人的聚会；也有像是做《MagPi》杂志的人，这是由一个小团体免费撰写和制作的杂志。他们不仅有爱好和热情，还乐于助人。树莓派基金会人员很少，你想不到我们是有多么地感谢所有那些在这第一年里帮助过我们的人们：你们用 Pi 做了很酷

的东西，告诉人们什么是树莓派，在网络上宣传派，在学校和小餐馆组织树莓派的聚会，撰写文档，提出有启发意义的问题，把一个这样的小计算机送给一个刚刚有点儿兴趣的小朋友。

最后，这就是工具论：有了一点点正确的知识，就像这本充满了知识的书，你就可以做出任何东西来。

Eben Upton 和 Liz Upton

树莓派基金会

2013 年 1 月

作者致谢

要不是 Eben Upton 的建议，这本书根本就不会写出来。Eben 是树莓派的架构师，也是我在剑桥的博通公司里的同事。博通公司本身也应该害羞地走进这聚光灯里来，它慷慨地让我有时间来写这本书。

我还要感谢在博通的许多其他同事，他们提供了独一无二的相关资料，他们中有 Dom Cobley、Gert van Loo 和 Eben。还要感谢 Dan Trowell、Gordon Hollingworth、James Chapman 和 Haynes 出版社友好的编辑们，他们都对本书提出了很有价值的评价。

第一章 介绍

历史	2	树莓派“项目”	4	本书布局	6
ARM时代	2	令人赞叹的技术	4		
BBC Micro	3	自由的意义	5		
黑暗的年代	3				

第二章 先咬为快

咬一小口	8	初次运行	15	高级配置	20
装配软件	8	使用桌面	16	其他插座和端口	20
装配硬件	10	配置树莓派	17	接下去做什么	21

第三章 烹饪之法

关于编程语言	24	操作系统介绍	29	程序结构	39
机器代码	24	线程	30	函数	40
汇编代码	24	内核	30	数据结构	41
编程语言	24	文件系统	33	类	43
程序的运行	24	Python入门	34	文档	43
树莓派的语言	25	Python的伟大之处	34	例子——Docker	44
名字	25	概览	35	其他例子	48
变量	26	准备工作	35	帮助	48
类型	27	Geany	36	Python模块	49
程序结构	27	使用Geany	36	Linux介绍	50
重复	28	注释	38	Linux发布版本	51
数据结构	29	名字	38	命令行shell	51
注释	29	变量	38	启动与引导	65
				网络	66

第四章 软件菜谱

定期运行程序	68	在Linux程序中直接访问	88	访问Windows共享	110
准备	68	创建UART设备	88	把树莓派用作文件服务器	110
cron表	71	通过SSH访问树莓派	88	工作原理	110
帮助	71	工作原理	88	树莓派上的准备	111
抓取网页	72	用途	89	SMB客户端设置	113
工作原理	72	树莓派上的准备	89	通过Linux图形用户界面浏览Samba	115
准备工作	72	从Unix计算机上使用	90	通过VNC访问树莓派	119
处理网页数据	72	从Windows计算机上使用	91	工作原理	120
使用互联网	78	通过X访问	93	树莓派上的准备	120
OSI和互联网	78	工作原理	94	运行VNC服务器	120
协议	78	运行一个X服务器	95	VNC客户端设置	123
在Python中使用套接字	84	实现一个网页服务器	98	以Windows远程桌面方式访问	124
在Bash中使用套接字	84	为什么需要网页服务器	98	工作原理	125
通过串口线访问	84	工作原理	99	树莓派的准备	125
释放UART	85	准备	99	RDP服务名字	126
在C中做终端控制	86	CGI脚本	104	Windows RDP客户端设置	126
在Bash中使用UART	87	响应查询	106	Linux RDP客户端设置	126
在Python中做终端控制	87	一个俏皮话的网站应用程序	107		

第五章 硬件菜谱

使用U盘和移动硬盘	128	电气极限	146	用sysfs创建PC设备	159
可以用什么	128	GPIO软件的工作原理	147	用C语言编程	160
使用USB Wi-Fi网卡	131	绕过内核的软件	149	用Python语言编程	161
树莓派上的USB	131	第三方板子用的软件	150	进一步的资料	161
Wi-Fi网卡和Linux	131	Python支持	150	通过UART连接PC	161
Wi-Fi网卡驱动程序	132	更多的GPIO线	151	扩展插座上的引脚	161
工作原理	132	特殊功能引脚	151	到PC的串行连线	162
准备	133	连接SPI设备	152	做一条USB串口线	162
使用蓝牙设备	137	引脚	152	测试连接	163
准备	137	工作原理	152	用电池工作	165
发送文件	139	使用Linux驱动程序	153	错误的信息	165
发光二极管LED	142	C编程	154	电压问题	165
LED功能	142	进一步的资料	155	电流问题	166
LED软件控制	142	连接一个PC设备	156	能源问题	166
通过GPIO连接硬件	145	引脚	156	直接输出的电池包	166
扩展插座	145	工作原理	157	带稳压器的电池包	167
第三方扩展	145	PC的软件	157	带DC-DC转换器的电池包	167
供电	146	使用Linux驱动程序	157	成品转换器	168
		命令行工具	159	停机	168

第六章 套餐

MP3网站	170	游戏设计	173	更多	190
准备	170	接口和抽象	185	媒体中心	190
关于Edna	170	其他例子	185	获得XBMC	191
解压Edna	170	推特提醒器	185	使用电视机遥控器	192
配置Edna	171	硬件	186	初次运行	192
使用Edna	171	Pi-Face	186	把树莓派当作DVD播放器	198
贪吃蛇游戏	172	硬件准备	186	通过SMB访问XBMC	198
用Python写游戏	172	软件准备	187	XBMC还能做更多	199
准备	173	测试这个玩具	189		
游戏描述	173	扫描全世界的推文	189		

第七章 附录

配置	201	引导文件系统	207	Hello World	211
超频属性	201	多媒体库	207	库	212
HDMI和显示配置属性	202	库	208	音频演示	213
HDMI模式	203	编解码器和开源部件	208	视频演示	214
电视机设置	205	文件系统位置和安装	208	表面纹理演示	214
UART配置属性	205	OpenMAX IL 1.1	209	渲染演示	215
SD控制器配置属性	205	OpenGL ES 1.1和2.0	210	整体演示	215
ARM装载程序配置属性	206	OpenVG 1.1	210	编码的例子	215
编解码器许可	206	EGL 1.4	210	矢量图形的例子	216
安全模式	206	C语言例子程序	211	字体的例子	216
其他选项	207	准备	211		

第一章 介绍

历史

在很久很久以前，历史上曾有过那么一段时间，一台计算机可以填满大学计算机系的一整间屋子的，唯一能和计算机交流的方式是在纸卡片上打孔，那时候大众认为计算机是要使用巨大的磁带盘的。正是在那样的“石器时代”，我迈出了我的书呆子之路的第一步。那时候互联网曾经是一个美国军方研究项目，电子邮件曾经很时尚，电话曾经是一种摆在厅里用布护套线连着的胶木东西。

对一个尚在成长中的极客而言，恐怕不会再有那样美好的时光了。从那以后，总有一些曾让人激动的东西，在仅仅 10 年里，它们的速度、尺寸或密度就成倍增长了。这不是随口说说，这经验定律已经有效了五十多年了。这个定律叫作摩尔定律，最初是在 1965 年由 Gordon Moore 提出来的，他是 Intel 的创始人之一，他发现集成电路（芯片）上的晶体管的数量每两年就翻一番。1978 年的时候，这个数据是准确的，这定律足以让人们臆测，认为到现在这个时候，我们就应该可以把所有的电影都存放在家里的电脑上了。即使今天，我们也还是可以用这个定律来预测将来。



有一件事情还没能预测，就是玩的东西会用到怎样的技术、谁会从中赚钱以及它对社会能造

成怎样的影响。谁能猜到对 1890 年美国人口普查数据的处理最终催生了 IBM？安排给 Lyons 公司的茶餐厅的配送间接形成了英国的大型机业务？而 CERN（欧洲原子能研究中心）里想要分享信息的一位英国物理学家给我们发明了万维网？

不能简单地说伟大的贡献都是不期而至，应该说伟大的贡献绝大多数都是从微小的创意开始的。在计算机领域有很多这样的故事，小的开始最终变成了巨大的成功，而在那些发轫之初的、核心的一两个人，所需的只是（a）正确的技术背景和（b）一些善于发掘新奇事物的天赋和运气——在正确的时间做正确的事情。当然一块树莓派和我们这本书并不能带给你运气，它只能对你的技术背景有所助益。

ARM 时代

如果你的记性很好，也许会记得曾经最通用的计算机是 IBM 制造的。经过一段很长的大型机（“主机”）的朝代。

随着个人计算机的兴起，一个新的朝代在 Intel 所发明的新的计算机指令集上建立起来。如果你曾经用过 Windows 操作系统，就应该已经运行过使用这种指令集的计算机了。和 IBM 的一样，这种指令集也是逐渐发展演变的，随着每个新的处理器的发布，新的指令会不断加入。现在，最新的 Intel 处理器还可以执行最老版本的指令。这十分宏伟的、全部的指令被叫作 Intel x86 体系。

今天，也许你会惊讶于听到，x86 体系已经不再是世界上使用得最多的指令集了。这个头衔属于 ARM 体系，ARM 处理器遍地都是，你自己手上很可能就已经有几个了。到 2011 年，据说已经生产了超过 150 亿个 ARM 处理器了。

既然世界人口估计为 70 亿左右，这就意味着每个活人的每条“胳膊”（英文单词就是 arm）上至少可以有一片 ARM 处理器了。未来的计算机考古学家通过发掘我们时代的遗迹，也许会把我们的时代称作“ARM 时代”。如果你希望能成为这个 ARM 时代的积极分子，那么你就需要有一片 ARM 来做点实验，正好，树莓派用的正是 ARM。

BBC Micro

“ARM”曾经表示“Acorn RISC Machine（Acorn 家的 RISC 机器）”，原本是生产 BBC Micro 的 Acorn 公司的一个部门做的机器。后来（随着 Acorn 公司遗憾地结束），这个缩写变成了“Advanced RISC Machine（先进的 RISC 机器）”。



BBC 微型计算机

在英国，有许多老者，BBC Micro 伴随着他们的成长，他们常常深情地回忆起那部小机器是如何点燃他们对无处不在的计算的兴趣的。这部小机器是 BBC 作为 PC 来支持他们的计算机解放计划的，在他们的电视连续剧“20 世纪 80 年代的计算机程序”中，他们需要有一台使用的计算机。

当我在剑桥大学的计算机实验室的一个同事，Ram Bannerjee，帮助搭建了原型电路板时我才第一次见到 BBC Micro。Acorn 在一个竞标

中满足了一些很紧张的截止时间要求后，才能把他们的机器叫作“BBC Micro”。Ram 用了一个研究项目中的一些成果，使得他可以相当快地把元件接好线。我非常幸运得以参与一些前期测试工作，后来就加入了 Acorn 继续从事这个工作。

如果你看过我们在 Haynes 出版的姐妹篇《视频游戏手册》，你会读到过气的游戏平台的历史。虽然那本书里没有 BBC Micro，不过有缩减版的 Electron。在 BBC Micro 上可以很好地玩游戏（人们真的玩过），不过，源于它的教育初衷，它期望用户自己写游戏。打开这种计算机（以今天的标准来看是非常快的），首先看到的是一个提示符，要你输入程序的第一行。如果你用过的计算机都是要求你首先选择一个要运行的程序的话，你就不会发现你其实是在用这种已经习惯的方式来使用计算机的。我还记得第一次使用 Windows 操作系统（Windows 95）的时候有多么的失落，我找不到熟悉的方式来控制计算机：

没有提示符、没有编程环境——那是一台属于别的程序员的计算机，不是我的。

```
BBC Computer 32K
Acorn DFS
BASIC
>
>10 PRINT "HELLO WORLD"
>
>RUN
HELLO WORLD
>
>
```

开机编程

黑暗的年代

自从 20 世纪 90 年代以来，人们家里的计算机主要是让人们使用提供给他们的软件的。当人们需要做什么事情的时候，只是用已有的，可以组合或是稍做调整就可以方便地完成任任务的软件。现在常见的做法，要不是上网络去搜索正好具有这个功能的新软件，要不就干脆放弃了。

就一件事情上自己做这样的“调整”也许不那么重要，不过计算机不是就用一次，一辈

子不做简单的事情（简单的事情必定会发展成更复杂的），累积起来就是教育的大问题。所以说，不知道从哪里开始做这样的事情构成了微小、但是实在的障碍，把新软件的潜在的开发者（以及相应的技术）白白耽误了。

为了在儿童中灌输“你是消费者，不是生产者”的观念，英国教育系统在信息通信和技术（ICT）课程中树立了这样的标准，大量地强调使用事实上的标准程序，而很少或根本不关心这些程序最初是怎么制作出来的，这些更有趣的细节知识。

树莓派“项目”

到孩子们毕业的时候，他们被灌输了大量只和文字处理、电子表格和数据库有关的工具的知识，而很少知道它们背后的基础技术。

经历了 20 世纪 80 年代和 20 世纪 90 年代的有理想、有知识的新生人类数量的膨胀之后，大学的计算机系现在发现他们正在面对的是日益增多的对编程爱好毫无经验的中学毕业生。

树莓派基金会（一家英国慈善机构）的人看到了一台真正的个人计算机在改变这样的状况时所能起的作用，就像 30 年前的 BBC Micro 曾经起到过的那样。

令人赞叹的技术

树莓派里的芯片是令人赞叹的，事实上，应该感谢摩尔定律，每一年芯片都在进步。每过一两年，进步带来的差距累积起来，可能就足以让你想要扔掉旧相机、笔记本电脑和电视机来买一批新的。十几二十年之后，累积的差距就在革命的边缘了。原本需要一个大柜子的，或者满屋子柜子的东西，现在就只是芯片里的小角落。

我的第一所大学在 20 世纪 70 年代后期有过一台巨大的 CDC 7600 计算机，它连接了很

多终端。那是当时最前沿的设计，里面有一些独立的计算机处理器，据说每秒能做 3600 万浮点运算。那时候，我们在憧憬即将到来的下一代计算机能具有一个“向量”单元，可以用一条指令在一连串数据上重复很多很多次相同的运算。

因为树莓派没有 120 英里手工接线的连线，也不消耗几百千瓦的电力，它比几百万英镑的 CDC 7600 的成本可是要少得不是一点点。不仅如此，它也是一个多处理器的机器，里面有两个额外的“VideoCore（视频核心）”处理器，每个里面都有一个相当强大的向量处理器，每个处理器每秒可以做 240 亿次浮点运算；不需要使用那种足以供给整个村庄的电力，树莓派里的芯片可以用手机电池来运行。虽然不像小尺寸手机强大的计算能力那么出名，但是树莓派确实只需要非常少的电力就可以工作。

无疑如果在 20 世纪 80 年代的早期就有了树莓派，这块芯片是足以服务一所大学的。不过今天，它的目的仅仅是用作一部手机而已。

树莓派里的芯片

树莓派里的芯片非常少，重要的东西几乎就在一片芯片里，它就在板子的中央。

如果仔细观察，你也许能够看到这片芯片是有两层的，有点像一个甘草什锦糖。芯片的上面是计算机的全部内存，下面可能看上去比较薄，不过里面有很多元件。这片芯片里的功能部件是如此紧密地布置着，它们之间的距离只有 40nm（纳米）。如果在它的表面掉落了一根头发，就可以盖

住 1 000 行的电子元件。



在中央的 BEM 2835 芯片

除了 ARM 和两个 VideoCore，它还包括了能支持 VideoCore 处理从各种来源收到的视频和音频、并把数据发送给各个地方去的所有元件。

ARM 本身每秒要运行惊人的 7 亿条指令。告诉你这有多快，它执行一条指令的时间，光只能前进大约 43cm（这正好解释了为什么芯片不能很大）。

看到“VideoCore”这样的名字，你也许会猜测这部分擅长做视频输出，你是对的。它能很好地处理任何常见来源（比如 DVD 或蓝光碟）的最高分辨率的视频。它的 3D 图形性能至少是上一代游戏引擎的水平，而且令人惊叹的是不需要任何特殊的手段就可以保持芯片不发烫。

自由的意义

在还能买到 BBC Micro，而且摩尔定律还在左右着计算机硬件的面貌的岁月里，另外一场独立的运动也在计算机软件领域发生着：一场追求“自由”软件的运动。

这场运动可能并非如你想象的那样，只是让大家可以开心地不花钱得到东西，因为“自由软件”里的“自由（free）”并不总是“免费啤酒”里的“免费（free）”的意思。有时候，它更像是“自由发言”或“自由人”里的“自由”，就是说，软件是“自由”的，而不是不恰当地受到诸如许可、专利或法规的束缚。

随着计算机变得越来越多，“雷锋程序员”的用户也变得越来越。许多“雷锋程序员”发现能用他们的编程努力帮助这么多的人，这对他们具有无法抗拒的吸引力，这正是成千上万的自由软件项目的成果所在。国际知识产权法律（比如版权）给予了作者对他们所写的程序的权利，可是除非采取措施，否则一个大作品经过成百

上千的分发之后，它的法律权利很快就无法维护了。像自由软件基金会这样的组织，非常努力地提供特殊的许可形式，让自由软件得以发展，并与威胁其生存的法律体系进行斗争。这是重要的，一旦没有了“自由”，就有太多的可能把一个“免费”的程序变成一个付费的程序。

现在已经有很多杰出的自由软件，具有和商业软件等价（如果不能超过的话）的能力。不过，更为重要的是，越来越多的人认识到，在大型“自由”软件项目中所使用的完全开放的方法，和商业项目中必须采用的封闭开发相比，是天然具有优势的。开发者的数量众多，加上他们开展的批评与自我批评，以及大量会指出程序错误的用户，就可以达到商业化运作无法企及的软件正确程度。

构成树莓派核心的 Linux 操作系统和 Python 编程语言，就是这样的自由或“开放”软件（其实基本上所有的树莓派软件都是）。从某种意义上说，树莓派计划的目标之一就是推广开放软件的使用。

像 Linux 或 Python 这样的大型程序是复杂的，使用这样的软件需要相关技能的学习，这既是代价（因为要花费大量时间来适应软件）也是收益（因为将来就能顺利地使用了）。这些技能在将来用得越多，收益就越大，而如果这软件不能用了，或者开始变得代价高昂了，收益就变小了。开放软件的一大好处就是这些风险都大大降低了。学习用 Python 来做日常小事有一点点可能不好的地方——比如，它可能会比那些小事本身成本还高，或者，突然市面上就有能解决你问题的软件出现了。总的来说，选择学习“开放”软件比选择相应的商业软件要安全得多。

本书布局

对新玩具稍微有点兴趣的人，首先要做的是让它动起来，所以，在介绍之后，紧接着的第二章就是关于如何组装各个部分及开始启动。

不过，本书并不只是给那些有一定基础的人的关于如何使用树莓派单板计算机的手册，它收集了一些内容，能有助于搬走横亘在那些有想法、想要尝试各种可能的人面前的障碍。

当然，如此宽泛的目标是不可能在一本书里完全实现的，很多章节都“只是恰好足够”让人开始，然后读者可以用其他地方能够得到的更详细的资料来进一步展开。

这种障碍很多并不是直接是针对树莓派的，而是学习不同类别的技术时的简单问题。所以，就好像要用正确的方式烹饪可以吃的树莓派一样，接下去的一章是烹饪课：介绍程序设计语言和操作系统。如果你已经有这些知识的背景，可以跳过去。

一块树莓派并不是整套大餐，只是大餐的

一部分。树莓派可以用来构成很多作品的核心，可能要用很多不同的东西来做成自己的作品。为此，本书的第三章着重提供各种小菜谱，给你尽可能多的信息来收集各种有用的点滴部件。

接下来的第四章着重介绍已有的软件菜谱，能帮助你作品以某种方式与远程用户交互、定时做事情或是提供各种网络服务。

第五章给出能用板上的插座和端口的硬件菜谱，包括很多隐藏在板子角落里的一排排引脚的功能。这章也稍微解释了一下大多数软件的情况，这样就可以在自己的程序里使用这些硬件端口。耗电的问题也讨论了一下。

最后本书给出了一些套餐的简单例子，你可能会有兴趣。有些，例如“贪吃蛇”游戏和Twitter通知玩具就适合作为自己作品的起点，其他的像全功能媒体中心就是完整的作品，拿来用就好了。

最后，本书的众多例子的程序代码可以在www.haynes.co.uk/rpiresource 找到。

第二章

先咬为快

咬一小口

我们来把新买的树莓派组装起来，让它工作起来，为此主要要做3件事情：

- 把树莓派需要的软件放到一张SD卡上；
- 插入SD卡；
- 开机运行。

下面我们一个一个地细谈。假如你遇到了本书没有预见到的问题，请使用你常用的搜索引擎，也可以先试试树莓派论坛网站：

<http://www.raspberrypi.org/>

装配软件

如果你的树莓派来的时候就带着一张SD卡，就可以跳过这一节看下面的。

树莓派上可以运行几个版本的Linux，在下面的网址可以下载到官方推荐的版本：

<http://www.raspberrypi.org/downloads/>

首先要做的是选择一个版本下载到你的计算机，然后才可以写入到SD卡。

当然此时你需要一张空白的SD卡。SD卡有各种容量的，树莓派的软件只需要占用略小于2GB的容量，不过SD卡也可以用来存放自己的文件，所以SD卡的容量越大越好。4GB的卡应该是最小的了。

你的下载电脑

用Windows或Linux都可以来做SD卡，这部电脑需要的主要功能有：

- 互联网连接；
- 一个内置的SD卡读卡器（许多笔记本电脑都有）或一个独立的SD卡读卡器（有很多这种东西，插在电脑的USB口上即可使用）；

- 足够存放SD卡映像文件的空间（至少2G字节空间，这很少会成为问题，除非你用的是非常古老的电脑）。

下载SD映像

在下面的网页上选择你要的系统：

<http://www.raspberrypi.org/downloads>

本书假设你要下载Debian Linux的“wheezy”版本，比如，像下面这个：



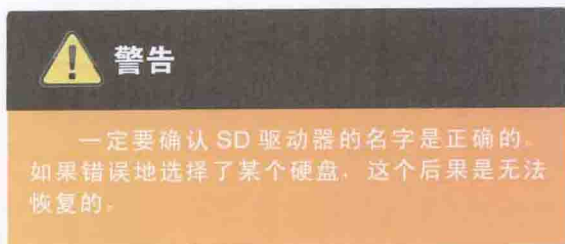
单击这个链接就到了下载页面，要下载的文件是一个大的“.zip”文件，所以需要一点时间来完成下载。

记下浏览器把下载的文件放到哪里了（一般在Windows和Linux都是在你的“Downloads（下载）”目录里。接着需要解压这个文件。



写 SD 卡是危险的

无论用的是哪个操作系统，可以写 SD 卡映像的程序也可以把硬盘的全部内容都改了。



用 Windows 写 SD 卡

打开下载文件所在的目录，解压那个“.zip”文件。大多数版本的 Windows 在文件的右键菜单上都有“Extract（解压）”或“Extract All...（全部解压）”的选项。



没有“全部解压...”？

如果你的 Windows 版本较老，或找不到任何解压文件的办法，可以在网络上寻找一个叫做“7-zip”的免费软件，下载安装。这样就会在右键菜单里出现一个“7-zip”的子菜单，其中有一些菜单项可以用来解压这个文件。

推荐用来把磁盘映像文件写到 SD 卡的软件叫作“Win32DiskImager”，如果你还没有安装，在

<http://sourceforge.net/projects/win32diskimager/files/latest/download>

可以下载。

你需要叫作“win32diskimager-binary.zip”

的文件，用和 SD 卡映像一样的方法下载解压。解压这个文件的时候，会产生一些文件，其中就有一个叫作“Win32DiskImager”的程序。

双击运行这个磁盘映像程序，不仅要知道磁盘映像文件所在的地方，还需要知道空白 SD 卡所在的设备的名字。



选择映像文件的时候应该显示如下的对话框：



单击文件夹的图标，找到下载的树莓派映像文件，然后选择放空白 SD 卡的“设备”（仔细阅读前面的警告）。Win32DiskImager 会把所有的内容都写到那个设备上去。

Win32DiskImager 有一个辅助功能，只列出“可移除”的设备，也就是可能是 SD 卡读卡器的设备。如果你的计算机不是太复杂的话，这很可能就是唯一的可移除设备。也许可以观察到 SD 卡插入读卡器后自动打开的目录，或是可以插入或拔出 SD 卡或读卡器来观察“电脑”或“我的电脑”显示的变化来找到这个设备的名字。

一旦找到了正确的设备和 SD 卡映像，单击“write（写入）”按钮，然后等待它完成（这可能要花几分钟时间）。



用 Linux 写 SD 卡

查看 SD 卡插入电脑前后哪个设备被识别出包含分区就可以找到 SD 卡。下面的命令会打印出 Linux 已经找出的所有分区：

```
cat /proc/partitions
```

在插入 SD 卡之前做一次，之后再进行一次。新出现的行就是新插入的 SD 卡。比如，如果插入 SD 卡之前是这样的：

```
% cat /proc/partitions
major    minor    #blocks  name
 8         0    312571224  sda
 8         1    15360000   sda1
 8         2    26291544   sda2
 8         3         1     sda3
 8         5    9764864   sda5
 8         6    261151744  sda6
```

而插入之后是这样的：

```
% cat /proc/partitions
major    minor    #blocks  name
 8         0    312571224  sda
 8         1    15360000   sda1
 8         2    26291544   sda2
 8         3         1     sda3
 8         5    9764864   sda5
 8         6    261151744  sda6
 8        16    1931264    sdb
 8        17    1930680    sdb1
```

就可以看出 sdb（整个 SD 卡）设备和 sdb1（SD 卡的第一个分区）是新的。这些名字对应的就是 /dev/sdb 和 /dev/sdb1 设备。我们要写 SD 映像的设备是 /dev/sdb（注意前面的警告）。

有些 Linux 版本会自动加载插入系统的 SD

卡，但是下面的步骤需要这个设备是没有加载的。已经加载了的设备通常在 /etc/mtab 文件里可以看到。如果 SD 卡被加载了，用下面的命令卸载每个已经加载了的分区（比如 /dev/sdb1）：

```
sudo umount /dev/sdb1
```

然后，用“dd”命令来把映像写入到设备里。如果映像文件是叫作“2012-08-16-wheezy-raspbian.img”，而 SD 读卡器设备是“/dev/sdb”（不是“/dev/sdb1”），写入的命令就是：

```
sudo dd bs=1M if= 2012-08-16-wheezy-raspbian.img of=/dev/sdb
```

这要花费几分钟。在此期间该命令不会产生任何输出。

装配硬件

如果手上没有树莓派所需的全部装备，那些配件一般都是相当便宜的（除了显示器），而恰巧你已经有一台电脑了，你就可以利用这台电脑的功能来方便地使用树莓派。

你可能已经习惯了给 Windows 电脑买装备，而且你发现几乎市面上所有的东西一装进去就能正确运行（没习惯这样也没关系）。不幸的是，给树莓派买配件没那么简单。为了提升购买到能用的装备的机会，试试查看下面的网页，它列出了过去曾经成功或不成功的配件：

http://elinux.org/RPi_VerifiedPeripherals

以后再来谈树莓派可以接的配件的诸多选项，现在先着重看需要什么才能继续下去。

除了树莓派单板计算机，你还需要下面的东西：

- 一个 USB 电源；
- 有树莓派软件的 SD 卡；
- 一个显示器（电视机也可以）；
- 一个 USB 键盘。