

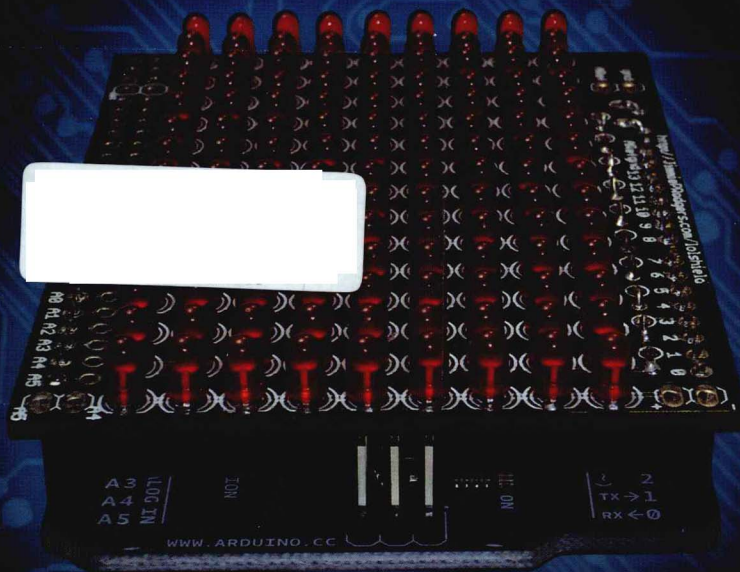
Arduino Internals

Arduino 技术内幕

- 全面深入理解Arduino开发，灵活打造炫酷作品！
- 用Arduino轻松实现自动控制，放飞你的创意！
- 依循设计原则，在Arduino DIY中体验创造的乐趣！

[美] Dale Wheat 著

翁恺 译



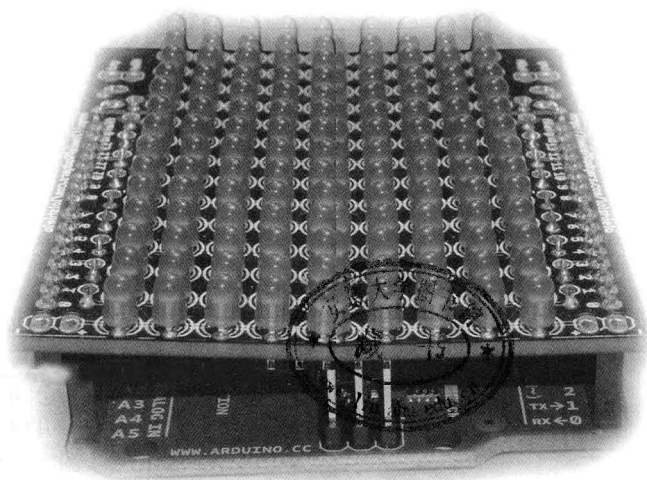
人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

Arduino Internals

Arduino 技术内幕

[美] Dale Wheat 著 翁恺 译



人民邮电出版社
北京

图书在版编目 (C I P) 数据

Arduino技术内幕 / (美) 惠特 (Wheat, D.) 著 ; 翁
恺译. — 北京 : 人民邮电出版社, 2013. 4
(图灵程序设计丛书)
书名原文: Arduino internals
ISBN 978-7-115-31192-4

I. ①A… II. ①惠… ②翁… III. ①单片微型计算机
—基本知识 IV. ①TP368.1

中国版本图书馆CIP数据核字(2013)第040105号

内 容 提 要

本书摒弃了大量基础知识, 全面深入揭秘 Arduino 的内部工作原理。其中, 它通过“LED 闪灯”、“LED 照明”、“数字钟”等示例作品由浅入深、由易而难介绍 Arduino 开发, 并最后以一个“自动机器人”示例总结全书, 不仅介绍了可用的软硬件及软硬件间的协作, 阐述了软硬件的选取与设计, 而且分析了项目管理(包括文档化、团队工作与协作开发以及如何选择许可方式)方面的内容。另外, 读者还可从本书了解设计原则与优秀的编程技巧、编程风格, 学会动手设计实现优秀的 Arduino 作品。

本书适合所有 Arduino 用户学习参考。

图灵程序设计丛书 Arduino技术内幕

-
- ◆ 著 [美] Dale Wheat
 - 译 翁 恺
 - 责任编辑 毛倩倩
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京鑫正大印刷有限公司印刷
 - ◆ 开本: 800×1000 1/16
印张: 20.75
字数: 490千字 2013年4月第1版
印数: 1-3 000册 2013年4月北京第1次印刷
- 著作权合同登记号 图字: 01-2011-8029号
ISBN 978-7-115-31192-4
-

定价: 69.00元

读者服务热线: (010)51095186转604 印装质量热线: (010)67129223

反盗版热线: (010)67171154

版权声明

Original English language edition, entitled *Arduino Internals* by Dale Wheat, published by Apress, 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705 USA.

Copyright © 2011 by Dale Wheat. Simplified Chinese-language edition copyright © 2013 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由Apress L.P.授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

献 辞

献给我的妻子 Anne。她的爱、耐心、信心、信任和支持使得本书终于得以写成。不过她有自己的技术专长领域（她是一位注册护士），所以要是她来阅读这本书，估计只会读读这一段吧。我觉得这样也还行啦，真的很感谢她。

前 言

让我们来彻底了解 Arduino 吧。Arduino 看上去就是一个简单的机器，其实不然。为了让它易学易用，很多人付出了艰巨的努力。然而，人们在好心好意地简化 Arduino 操作的同时，却会将它一些内在的潜能掩盖起来。事实上，在它简单的外表下，还有很多很多可以挖掘的东西。

确实如此。

这正是一本关于 Arduino 工作原理的书。Arduino 有效地体现了一些经过多年演化的设计理念，假使你深入理解了其所涉及的复杂技术，就可以改动或重新编排 Arduino 的“内部元素”以满足自己特定的设计需求。

目标读者

本书适合各个层次的 Arduino 用户阅读参考，无论跃跃欲试的新手，还是经验丰富的专家都能从中获益。也许你已经知道 Arduino 是什么，明白它能做什么，也许你已经有了这一块 Arduino 电路板，或者能从某处弄一块来试试。对于阅读本书来说，你无需有这些基础，只要你对 Arduino 和它内部秘密有浓厚的兴趣就足够了。

Arduino 的入门介绍已经随处可见了，本书就不在那些非常基础的事情上过多着墨了。本书深入探讨了很多内容，因此也希望能伴你在深入研究 Arduino 的路上走得更远。我可不想让你一学会如何让 LED 闪烁就再也用不上这本书了。

关于本书

本书不打算正儿八经地教你学基础电子学，不过，假如你求知若渴又聪颖好学，也许可以从本书的示例作品和练习中获得大量有用的电子学基础知识。我们认真准备了这些示例，并进行了严格的技术审校，确保它们都体现了正确的设计原则。

本书也不是程序设计的入门书。读者应该会用编辑器编辑源代码，并按照简单的指示完成指定的操作。当然，天性好奇的你从书中那些力求清晰的代码中不知不觉学到编程技巧和风格。

我们知道，任何一个章节都没办法事无巨细地彻底说清楚某个问题，所以我们既给出了实用的信息，也列出了参考文献，供你查找详细资料。Arduino 和其他任何复杂的系统一样，是不断发展着的。正如赫拉克利特在 2500 多年前说过的，“人不能两次踏进同一条河流”，因为河水一

直在流动。这话在当年是正确的，现在依然正确无误，尤其对于正在快速演变的 Arduino 来说更是如此。因此，罗列过多细节并不合适。

注意，本书绝不是没有头绪地胡乱堆叠技术资料。Arduino 光鲜闪亮的外表下面，层层叠叠隐藏着对复杂设计难题的复杂解决方案。本书的根本目的，就是以富有意义、易于理解的顺序一层层拨开这些迷雾。

最后，本书也并非只是随意选择一些毫无关联的 Arduino 项目。书中的项目和练习都是围绕说明和强调那些重要的设计理念精心编写的，而且由易而难，逐步深化，帮助你尽情探索 Arduino 及各种可能性。

本书内容

现在来简单看一下本书的内容，看看我们究竟能深入探索到什么程度。我们在各种技术资料中做了充分的权衡取舍，以使每一章都能独立成篇。

按照本书的章节顺序阅读当然好，不过按照兴趣跳跃阅读也没问题。书中的内容还是很丰富的，各章基本上遵循由一般到具体的原则编排，要是跳来跳去发现一下子读不明白了，可以往前翻几章，确保获得足够的基础知识后再继续往后阅读。

□ 第 1 章 硬件

看看有哪些 Arduino 模型，大致了解它们的内部工作原理。

□ 第 2 章 软件

Arduino 开发的每一个层次上都会用到各种软件，本章全面概览一下这些软件。

□ 第 3 章 Atmel AVR

Arduino 的“大脑”也大有名堂。了解它能做什么、不能做什么，是一个项目成功的关键。

□ 第 4 章 支持硬件

电路板上其他元件也起着重要的作用，了解它们能做什么、不能做什么，有助于做出更好的作品。

□ 第 5 章 Arduino 软件

利用 Arduino 提供的自由软件可以方便快捷地开始工作。它看上去很简单，但其实不然。

□ 第 6 章 优化

简单写就的代码总是可以做很多优化。本章将告诉你如何做优化，以及如何用精确的测量技术来验证优化。

□ 第 7 章 硬件和软件的结合

硬件和软件的协作，尤其是充分协作时，可以实现惊艳的结果。仔细研究 AVR 片上的外围设备，可以用较少的代码实现更多的功能。通过例子，本章详细解释了通用 I/O 端口、USART（串口）、计数器、定时器、PWM 输出和模拟输入，还介绍了如何使用中断来实现硬件软件之间的有效协作。

□ 第 8 章 示例作品

LED 闪烁？真的？本章先展示如何通过权限控制 LED 闪烁，然后做一个更复杂的作品：数字钟。

□ 第 9 章 项目管理

处理一些开发过程中的事情，比如文档、协作和许可等。

□ 第 10 章 硬件设计

非常全面地介绍了设计 Arduino 兼容或部分兼容的硬件所需的各种知识，包括电源供应、处理器选择和 shield 接口设计，也介绍了一些硬件设计自动化工具。

□ 第 11 章 软件设计

不光使用代码草图，还要根据具体情况决定如何设计和使用软件。本章讨论了另外一些开发环境、库的开发和文档，以及 PC 端的应用程序。基于本章的知识，你甚至可以开发自己的软件工具。

□ 第 12 章 组网

理论上，对 Arduino 进行组网非常简单，就好像只需要决定和谁连接、连上了要传输什么内容就可以了。不过，实际上没有这么简单。本章介绍如何利用 Arduino 内置的通信能力实现与各种设备的通信。当然，加上一些专门的硬件可以扩展网络的通信能力，甚至加上一些不太贵的硬件就有可能做出一个简单的 Web 服务器。

□ 第 13 章 其他示例

由一系列独立项目有机组合做一个小型自动机器人的例子。需要用到本章之前所有的知识才能完成这个大作品。

小结

本书致力于探索和描述 Arduino 体系结构的内部工作机制，为 Arduino 用户提供参考。掌握这些知识有助于更快更好地开发 Arduino 作品。本书也希望能点燃读者对嵌入式系统设计的兴趣，激发读者今后开发出更好的作品。

祝大家好运，学习开心！

目 录

第 1 章 硬件	1	2.2.3 第三步: 给芯片编程	28
1.1 什么是 Arduino	1	2.2.4 第四步: 测试和调试	29
1.2 Arduino Uno	2	2.2.5 第五步: 重复	29
1.2.1 处理器	3	2.3 半自动化	29
1.2.2 串口	4	2.4 更进一步	33
1.2.3 电源	5	2.5 小结	33
1.2.4 扩展插座	6	第 3 章 Atmel AVR	34
1.2.5 盾板	8	3.1 起源	34
1.3 Arduino Mega 2560	11	3.2 AVR 芯片家族	34
1.4 之前的硬件	12	3.3 若有疑问: 芯片器件手册	35
1.4.1 Arduino Serial	12	3.4 芯片封装	35
1.4.2 Arduino USB	13	3.4.1 双列直插 (DIP)	35
1.4.3 Arduino Extreme	14	3.4.2 表面安装器件 (SMD)	36
1.4.4 Arduino Nuova Generazione (新一代)	14	3.4.3 多出来的引脚	37
1.4.5 Arduino Diecimila	14	3.5 管脚定义	37
1.4.6 Arduino Duemilanove	15	3.6 AVR 内核	43
1.4.7 Arduino Mega	15	3.6.1 时钟源	45
1.5 谁在做 Arduino	15	3.6.2 地址空间	46
1.5.1 正式许可的产品	15	3.6.3 指令集	49
1.5.2 其他人	17	3.7 片内外围设备	54
1.6 自己做 Arduino	17	3.7.1 通用输入/输出 (I/O)	55
1.6.1 Arduino 印刷电路板	17	3.7.2 外部中断	55
1.6.2 面包板 Arduino	18	3.7.3 定时器/计数器	56
1.7 小结	19	3.7.4 USART	57
第 2 章 软件	21	3.7.5 两线串行接口 (TWI), 即 I ² C	57
2.1 主机和目标机	21	3.7.6 模拟输入	58
2.2 一步步来	21	3.8 小结	58
2.2.1 第一步: 写代码	22	第 4 章 支持硬件	59
2.2.2 第二步: 编译代码	27	4.1 电路图	59
		4.1.1 元件类型	60

4.1.2 元件编号	60	6.2 压缩闪存程序	83
4.1.3 元件值	61	6.2.1 闪烁是如何实现的	84
4.1.4 元件值误差	61	6.2.2 衡量节省空间的优化措施	84
4.1.5 元件的其他参数	62	6.2.3 代码分析	84
4.1.6 连接	62	6.2.4 没有 pinMode() 的生活	85
4.2 给电路板供电	63	6.2.5 缩写和简语	86
4.2.1 柱式电源插座	63	6.2.6 二进制写法	87
4.2.2 输入电源调节	63	6.2.7 深入分析	87
4.2.3 稳压器	65	6.2.8 轻松翻转	88
4.2.4 电源电路的演变	65	6.2.9 进一步精简	88
4.3 串口	67	6.2.10 更有效地“浪费时间”	89
4.4 处理器	69	6.2.11 更低级别的代码	89
4.4.1 功耗	69	6.3 用简单的串口通信来节省空间	90
4.4.2 I/O 驱动能力	70	6.3.1 “Hello, world!” 做了什么	91
4.4.3 -RESET 信号	70	6.3.2 写配置寄存器	91
4.4.4 时间基准	70	6.3.3 发送数据	93
4.4.5 去耦电容	71	6.3.4 一串字符	94
4.4.6 闪光灯	71	6.3.5 输出数字	95
4.5 扩展空间	71	6.4 节省 SRAM	96
4.6 结构外形	72	6.4.1 测量要用的 SRAM	96
4.7 通用串行总线 (USB) : 信号加电源	73	6.4.2 最小裸机	98
4.8 小结	74	6.4.3 内存分区	99
第 5 章 Arduino 软件	75	6.4.4 变量的位置	100
5.1 开源软件	76	6.4.5 使用恰当的数据类型	100
5.2 多平台支持	76	6.4.6 再探字符串	101
5.3 Arduino 的传承与发展	76	6.5 低功耗, 还是高速度?	102
5.4 软件安装	77	6.6 电子测量	103
5.5 实践	77	6.6.1 用 Arduino 做测试仪器	103
5.6 用户界面	78	6.6.2 尽可能快	105
5.6.1 File (文件) 菜单	79	6.6.3 让它慢下来	108
5.6.2 Edit (编辑) 菜单和 Edit 关联菜单	80	6.6.4 进一步降低功耗	110
5.6.3 Sketch (程序) 菜单	80	6.7 小结	112
5.6.4 Tools (工具) 菜单	81	第 7 章 硬件加软件	113
5.6.5 Help (帮助) 菜单	82	7.1 可用的外围设备	113
5.7 小结	82	7.1.1 串口	113
第 6 章 优化	83	7.1.2 通用数字输入输出	119
6.1 可行性	83	7.1.3 定时器和计数器	121
		7.1.4 脉宽调制 (PWM) 输出	124
		7.1.5 模拟输入	129

7.1.6 外部中断	133	9.3 选择许可方式	195
7.1.7 中断手册	135	9.3.1 专利和商标	195
7.2 小结	137	9.3.2 版权	196
第 8 章 示例作品	139	9.3.3 开源	197
8.1 不只是闪烁的 LED: 从简单开始	139	9.3.4 公共域	198
8.1.1 足够慢	142	9.4 小结	199
8.1.2 基本完成优化的六通道调光 器	144	第 10 章 硬件设计	200
8.1.3 变暗了	147	10.1 了解硬件	200
8.2 闪烁 LED 的其他用处	154	10.1.1 必须要有的东西	201
8.2.1 红外遥控	154	10.1.2 想要有的东西	202
8.2.2 TV-B-Gone	156	10.2 红外接近传感器	203
8.3 许多闪烁的 LED	157	10.2.1 一个简易原型	205
8.3.1 一个直接驱动的例子	157	10.2.2 一些简单的改进	207
8.3.2 用 LED 驱动器直接驱动	159	10.2.3 印制电路板	210
8.3.3 多路复用技术	160	10.2.4 PCB 布局技术	211
8.4 数字钟	174	10.2.5 第一次尝试	212
8.4.1 精度	179	10.2.6 更小的版本	214
8.4.2 用户界面	180	10.2.7 连接	215
8.4.3 其他功能	180	10.3 自己定制的 Arduino	216
8.5 小结	181	10.3.1 兼容现有 Arduino 和 1 号板	216
第 9 章 项目管理	182	10.3.2 电源的可能选择	217
9.1 文档	182	10.3.3 处理器选择	218
9.1.1 源代码注释	183	10.3.4 还有什么?	219
9.1.2 空格	184	10.4 设计用的软件	219
9.1.3 按你的意思编程, 按你的代码 解释	185	10.4.1 CadSoft EAGLE	220
9.1.4 自动文档编制	186	10.4.2 EAGLE 贴士	220
9.1.5 写给你的“读者”看	186	10.5 小结	221
9.1.6 硬件文档	187	第 11 章 软件设计	222
9.1.7 再进一步	188	11.1 Arduino 高级话题	222
9.2 团队工作和协同开发	189	11.1.1 写 Arduino 库	222
9.2.1 博客	190	11.1.2 备选内核	229
9.2.2 论坛	190	11.2 不用 Arduino	239
9.2.3 维基	191	11.2.1 再探 Bare Metal	239
9.2.4 修订控制系统	191	11.2.2 其他开发环境	242
9.2.5 关于修订版号和版本号的 说明	192	11.3 小结	245
9.2.6 放作品的网站	192	第 12 章 组网	246
		12.1 点对点组网	246
		12.1.1 通过串口交谈	246

12.1.2	Arduino 间的对话	248	13.4.3	非接触传感器	278
12.2	MIDI: 乐器数字接口	260	13.4.4	声音传感器	279
12.3	互联网	264	13.4.5	指示器、控制器和其他形式 的通信	279
12.4	小结	268	13.5	控制系统	281
第 13 章	更多示例作品	269	13.5.1	开环系统	281
13.1	一个自动机器人	269	13.5.2	闭环系统	282
13.2	电源	270	13.6	示例机器人作品	283
13.3	动作控制	270	13.6.1	一个练习机器人	284
13.4	传感器	276	13.6.2	下一个机器人	304
13.4.1	光敏传感器	276	13.6.3	你的终极机器人	318
13.4.2	接触传感器	277	13.7	小结	319

自从2005年面世以来，Arduino的硬件一直在缓慢升级换代。Arduino是硬件和软件的有机结合，因此充分理解这两个领域及其交叉领域的知识很重要。本章考察Arduino硬件的总体概况，对某些部分做些细致的研究，另外也会涉及它的历史，并就读者如何影响它将来的发展给出建议。

1.1 什么是 Arduino

由于Arduino已经有了一些年头，而且经历了一些升级换代，如今能够称作Arduino的电路板已经很多了，而且每天都还在增多。Arduino团队正式发布的是Arduino Uno和大一些的Arduino Mega 2560。

提到Arduino，大多数人想到的都是那个小型、长方形的（很可能是蓝色的）PCB（Printed Circuit Board，印刷电路板），它就是所谓的I/O电路板，请看图1-1。

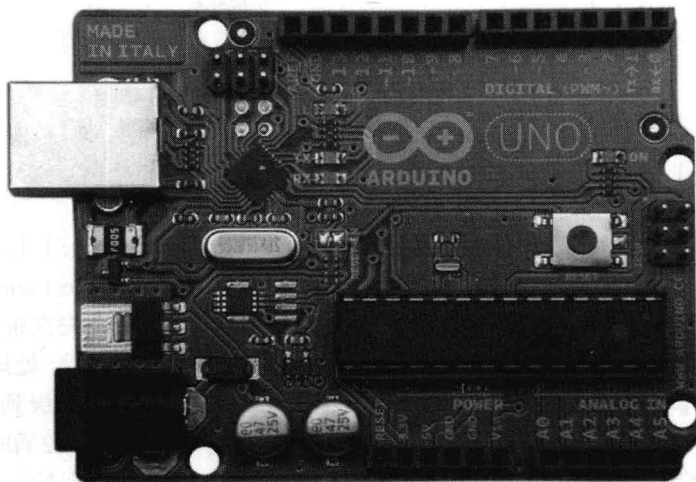


图1-1 Arduino的I/O电路板。对于大多数人来说，这就是听到“Arduino”时想到的东西，其实它只是整个大系统中的一部分

I/O电路板是Arduino系统中有形可见的部分。严格来说，Arduino这个术语涉及硬件、软件、开发团队、设计原理和用户群体的互助精神。当然，你经常会听到人们说“请把那个Arduino递给我”，或是“小心那个Arduino”。

Arduino发轫于意大利的伊夫雷亚（Ivrea）。伊夫雷亚的阿尔杜伊（Arduin）是约1000年前的意大利国王，在当地声名卓著。焦贝蒂广场（Piazza Gioberti）上有一间酒馆以此国王的名字命名，也有人说这家酒馆不过是以其所在的道路阿尔杜伊诺路（Via Arduino）命名的。

Arduino是意大利语中的男性用名，意思是“强壮的朋友”。作为一个专有名词，Arduino总是以首字母大写形式出现，而型号的名字Uno只是在PCB上的图标中才全部字母大写。关于Arduino的历史和传承，还有很多堆积成山的有趣信息资料，请参见Arduino网站（<http://arduino.cc>）。

Arduino I/O电路板传统上是基于Atmel的AVR ATmega8及其后续型号的。I/O电路板上串口、电源电路、扩展插座和其他一些必要的元件。官方的Arduino FAQ说：“这只是一块AVR开发板。”（www.arduino.cc/en/Main/FAQ）这样说的前提是，你知道什么是AVR；当然，读过第3章后你会知道的。（提示：AVR是一块可编程的单片机芯片。）图1-2是I/O电路板简化的方框图。

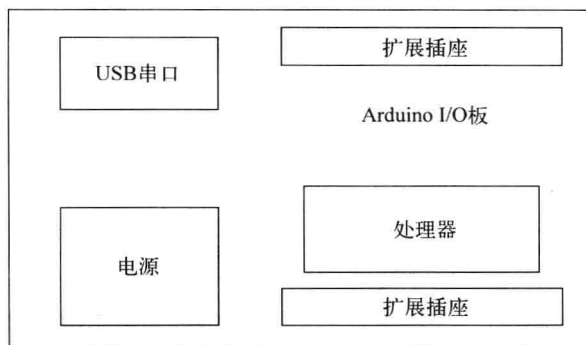


图1-2 Arduino的I/O电路板方框图

1.2 Arduino Uno

Arduino Uno是2011年9月25日在纽约创客大会（New York Maker Faire）上发布的。型号名字Uno是意大利语中“一”的意思，用来表达Arduino软件的1.0版，即Uno Punto Zero（意大利语的“1.0”）版。之前的版本，编号为0001到0022，被认为是alpha版或预先发布版。

Arduino Uno和它之前的版本极其相似，外形是一样的。这么多年以后，处理器从最初的8 KB程序存储器的ATmega8先升级到了16 KB程序存储器的ATmega168，然后升级到了32 KB程序存储器的ATmega328。不过，每次升级都保持了芯片引脚的兼容。9针的RS-232插座和接口电路换成了用USB接口芯片的虚拟串口。电源电路做了优化，提供了额外的过流保护和智能电源供选择。

由于全球范围内都暂时短缺受大众喜爱的ATmega328处理器的28脚双列直插封装芯片（DIP，有别于其他封装，Atmel的器件型号是ATMEGA328P-PU，其第一个P表示低功耗的picoPower技术，

而第二个P表示塑料DIP), 因此有了采用贴片工艺的Arduino Uno版本, 它被命名为Arduino Uno SMD。它的功能和Uno是完全相同的。唯一的缺点是贴片的处理器芯片不能像插座插装的DIP版本那样轻易地从PCB上取下来, 参见图1-3。

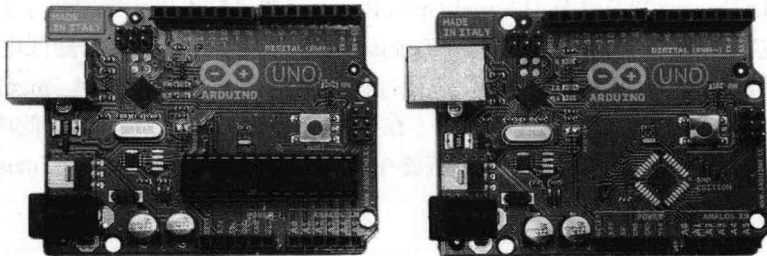


图1-3 Arduino Uno (左) 和 Arduino Uno SMD (右)

如果你开始对Atmel AVR的细节感兴趣, 恭喜你! 所有相关的内容都在第3章, 包括封装选项、引脚如何工作以及能做什么, 其中还很好地介绍了这个强大芯片的内部工作原理。而现在, 我们只关注Atmel AVR如何在这个大的背景——Arduino——中起作用。

1.2.1 处理器

Arduino Uno的大脑是Atmel AVR ATmega328, 那个黑色、长方形、两侧各有一排引脚的塑料块。在SMD版本中, 处理器是直接焊在PCB上的两个小小的黑色方块之一。

这个东西实质上就是单芯片的计算机, 封装了中央处理单元(CPU)、内存阵列、时钟和外围设备, 见图1-4。

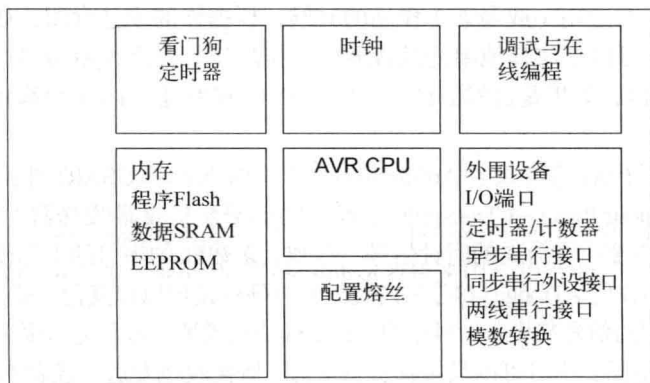


图1-4 ATmega328的简化框图

ATmega328芯片是从最初的Arduino用的处理器ATmega8发展过来的, 它比之前的型号内存更大, 片内外围设备功能更多, 同时耗电更少。ATmega328处理器可以在很宽的供电电压下工作, 从1.8 V到5.5 V都可以, 因此很适合用于电池供电的应用程序。在最低的供电电压下, 处理器最高

只能工作在4 MHz（每秒400万个周期）的时钟频率下。供电电压提高到至少2.7 V，时钟频率就可以提高到10 MHz。如果要以最高的20 MHz的时钟频率工作，芯片至少需要4.5 V的供电电压。Arduino I/O电路板供给ATmega328芯片的是5.0 V，因此它可以工作在最高20 MHz以内任何时钟频率上。

现在发售的ATmega328芯片具有Atmel的picoPower低功耗技术，大幅降低了器件的功耗。具有这个技术的芯片标有一个P后缀，比如ATmega328P。之前的型号要么是低电压运行（比如ATmega328V）但是不能全速的，要么是全速运行但是不能低于2.7 V供电的。picoPower技术消除了这个限制，使得一片芯片既可以全速运行（在适合的供电电压下），也可以低功率减速运行。picoPower使得芯片不再像前一代产品那样需要在型号中标注其速度（比如ATmega328P-PU，之前可能标成ATmega328-20PU）。

注意 某些专门的I/O电路板型号是设计在3.3 V下工作的，这样它们的最高时钟频率就限制在了10 MHz。

尽管新的ATmega328芯片最高可以工作在20 MHz，但是最初的ATmega8最高只能工作在16 MHz，所以在后续所有的Arduino型号中都继续使用16 MHz这个时钟频率以保持兼容性。

关于Atmel的AVR处理器系列的详细资料，请参见第3章。

1.2.2 串口

串口的功能从最初的Arduino开始就没改变过，接插件（connector）变了，但是大家都假装一切都还是一样的。当然，从功能上说，确实是这样。

串口是用来通信的。在开发Arduino作品的时候，通信发生在编写、编译和上传程序的PC和Arduino电路板之间；在应用（或部署）作品的时候，根据你的设计意图，如果需要，串口可能仍然是与PC通信，也可以与其他串行设备通信。作品运行时是否使用串口是可选的，所以也可能不与任何设备通信，如果是这样的话，接收（RX）和发送（TX）引脚也可以用来做通用输入/输出引脚。

有几种不同的串行通信协议。Arduino的串口（内部称作USART外围设备，即Universal Synchronous/Asynchronous Receiver/Transmitter，通用同步/异步接收器/发送器）工作在异步模式，意思是说它不提供也不需要一个分离的时钟信号。这和大多数PC的串口的工作模式是一样的，PC上的串口也叫做RS-232端口。ATmega328芯片内置的串口硬件是可以以其他模式工作的，包括同步模式，同步模式需要专用的信号来表示时钟信息。异步模式只需要一根信号线做发送，另一根做接收。根据应用程序需求的不同，串口可以只接收、只发送、既接收也发送，或者不接收也不发送。

警告 不要把RS-232信号线直接接到Arduino电路板上，一般的RS-232信号线上较高的电压可能会损坏电路板上的电路，包括处理器。当把Arduino电路板连到一个RS-232端口的时候，必须使用一个RS-232到TTL的适配器。

1.2.3 电源

电源电路并不会真的给Arduino提供任何电源，它只是把外部电源传输、稳压和过滤给Arduino。现在的电路已经经过了多年的演变，易于使用——基本上是傻瓜式的过程。电路会自己选择最高可用电压的来源，供给其他各个部分。电路板上甚至还装了一个可恢复保险丝，以防短路造成损坏，减少过热造成损坏的几率。这正是Arduino团队在过去这岁年中倾听用户群体的声音，不断对产品加以改进的成果。

有多种办法可以给Arduino供电。最简单也是最初始的方法，就是通过连接到PC的USB线给它供电。USB标准允许向一个未枚举的USB设备（就是插入USB总线但是没有向主机报告自己身份的设备，比如USB电源转接头）提供5.0 V最大100 mA（即0.1 A）的电流，而枚举了的USB设备可以获得多达500 mA（0.5 A）的电流。这足以提供电力来点亮几个LED灯或是驱动几个低功耗的传感器，但是对于大电流负载，比如继电器、加热器、风扇、马达或电磁阀还是不够的。

若Arduino没有通过USB线连接到PC，可以通过扩展插座上标着5 V和GND（地）的引脚向它输入稳压过的5 V电源。

警告 通过5 V和GND引脚从外部接电时必须接稳压过的5 V电源。未经稳压的电源的电压会随着电网电压以及电路板上负载的变化而变化，就很可能突破狭小的正常电压范围，造成处理器在内的若干元件的永久损坏。标准的Arduino I/O电路板上有一个稳压器，接未稳压的电源时应该用上它。

对于未稳压的电源，电路板上有一个柱式插座，从这里可以输入7 V到12 V的电源，它直接接到一个5 V的稳压电路。理论上输入的电压可以高达20 V，但是这样的话，稳压芯片就有可能过热，而过热则可能永久损害PCB。

最新设计的Arduino的PCB极大地改善了稳压芯片焊在电路板上的接地面，提高了铜箔散热的能力。不过尽管如此，保守估计这个芯片的热阻也会超过100°C/W，也就是说如果有1 W的功率要消耗在这个器件上，器件表面的温度就要比环境温度升高100°C。这真是太热了！千万别欺它太甚！

柱式插座内部有一个2.1 mm直径的针，柱式插座中央的这个针是接正极的，外圈的套管是接地的。插座内芯的针也直接接到扩展插座的Vin引脚。Vin引脚可以用来给插在扩展插座上的盾板（shield）供电，也可以从盾板向Arduino电路板供电。

现在的Arduino I/O电路板有一个非常好的设计，允许同时连接多个电源。智能电源切换电路会选择最高可用电压的电源，然后将其接入稳压器。

如果跳过这个电路直接向Arduino提供稳压的5 V电源——很可能有人会这样做——一定要小心确认接的是5 V的电源，因为所有提供保护的安全措施都被跳过了。当然，如果你头脑很清楚，这样做也无妨。

Arduino Uno电路板上还设了一个专用的3.3 V稳压器。之前的I/O电路板便依赖于FTDI USB接口芯片内部的小稳压器提供的3.3 V电。虽然这样做没给系统增加任何成本，但是这个小稳压器只能在3.3 V上提供最大50 mA（0.05 A）的电流。Arduino Uno自己配了一个3.3 V稳压器（美国