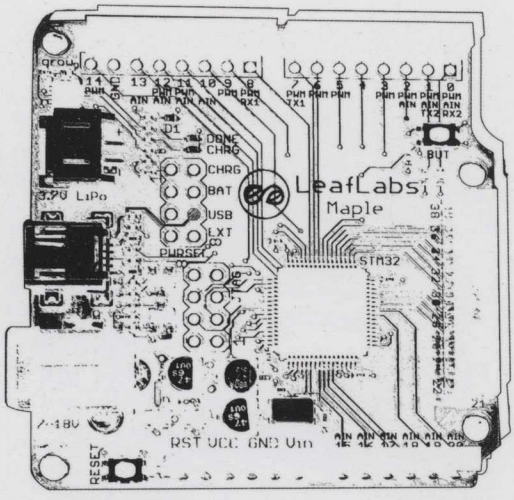
HZ BOOKS
华章科技

完全兼容Arduino的开源硬件项目Maple的权威操作指南。
MIT LeafLabs实验室作品，采用72MHz处理器，比Arduino更快的“Arduino”。
与工业控制完美对接，基于工业界最流行的32位STM32处理器。
全彩色印刷，可选配入门套件，动手体验更佳。

数字匠人



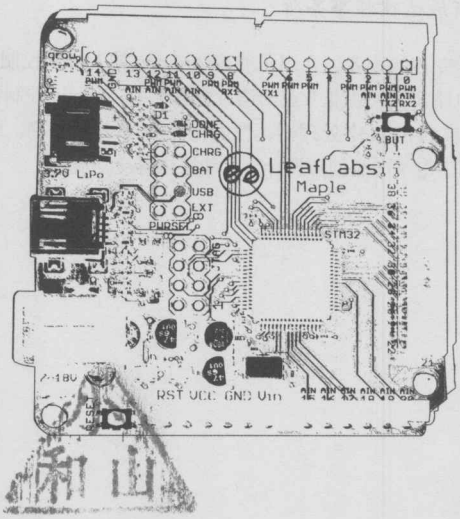
STM32篇

Arduino 开发实战指南

姚汉 编著

机械工业出版社
China Machine Press

数字匠人



STM32篇

Arduino 开发实战指南

姚汉 编著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Arduino 开发实战指南：STM32 篇 / 姚汉编著 . —北京：机械工业出版社，2013.11

ISBN 978-7-111-44582-1

Ⅰ. A… Ⅱ. 姚… Ⅲ. 单片微型计算机 – 指南 Ⅳ. TP368.1-62

中国版本图书馆 CIP 数据核字（2013）第 254049 号

本书主要面向非电子专业的读者，介绍了兼容 Arduino 的 Maple 控制器的编程方法与基础电路设计。书中有大量代码和硬件电路实例，使非专业的读者更易上手。Maple 控制器编程简单，功能强大，采用了基于 ARM Cortex-M3 内核的 STM32 处理器，性能比 AVR 单片机更高。Maple 控制器开发上与广泛使用的 Arduino 相兼容，可用于电子产品创意设计、互动媒体等交互应用。

机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码 100037）

责任编辑：朱秀英

藁城市京瑞印刷有限公司印刷

2014 年 1 月第 1 版第 1 次印刷

186mm × 240mm • 9.75 印张

标准书号：ISBN 978-7-111-44582-1

定 价：59.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：（010）88378991 88361066 投稿热线：（010）88379604

购书热线：（010）68326294 88379649 68995259 读者信箱：hzjsj@hzbook.com

前　言

早年的DIY爱好者集中于电子制作，后来由于计算机的出现，电子类DIY逐渐被边缘化，取而代之的是对程序的研究。而Arduino这类开源硬件为物理电路连接的现实世界与虚拟的程序世界架起了一座桥梁，通过它们可以方便地利用程序控制物理的驱动器，或是读取传感器数据，与现实世界交互，为很多环境交互应用提供了实施的平台。因此，近年来开源硬件与物理计算得以迅速发展，并有愈演愈烈的趋势，吸引着各行各业越来越多的人使用。

开源硬件中最为人熟知的是Arduino，它将嵌入式编程中复杂的寄存器操作改为简化的C++语言，大幅度降低了嵌入式设计的门槛。我也是从Arduino开始使用的，但是在使用的过程中逐渐发现Arduino的不足——计算能力小与芯片资源过少。后来我了解到Maple。作为一款以ARM Cortex-M3为处理器核心的Arduino兼容硬件，Maple在具有Arduino易开发特性的同时，在工作频率、储存器等芯片资源上有了很大提高。利用Maple可以开发出很多优秀的互动作品，实现一些更复杂的创意想法。

现在，国内的Maple中文资源比较缺乏，这极大地限制了开源硬件爱好者对Maple的了解与使用，给很多正在使用的朋友带来了不便，也阻碍了很多有兴趣的朋友对它的交流。所以作为第一批将其引入国内并推广的人，我写了这本书，希望能给Maple入门使用者带来一些便利，以使他们更好地利用Maple实现自己的创意。

全书主要分为四部分内容，第一部分主要讲解Maple控制器上Arduino兼容库及其使用，第二部分讲解一些基础的元器件知识，第三部分主要讲解如何使用Maple控制器由简到难地完成各种实验，第四部分则结合开源嵌入式实时操作系统进行简单的嵌入术应用开发。

希望这本书能够促进Maple和其他开源硬件的推广和应用。也希望使用Maple的朋友能够在网络上分享自己的创意与实施过程，并一起推动Maple这款开源硬件的完善和改进。由于作者水平有限，书中难免存在错误与疏漏，欢迎广大读者指正。

作者

目　录

第 1 章　Maple 简介

1.1　Maple 与 Arduino 的关系

Maple 是由麻省理工学院的学生所组成的 Leaflab 实验室开发的，其目的是让科技不再局限于高投入的实验室，将低成本、易开发的嵌入式设备推广开来。

在设计上，Maple 对 Arduino 有很高的兼容性，例如，部分引脚的排列和功能、提供的操作函数（除了 Arduino 中的 tone() 与 pluseIn() 函数不支持）。Maple 与 Arduino 最大的不同在于它所使用的处理器是 32 位的 ARM 处理器，而 Arduino 采用的是 8 位的 AVR 处理器。与 AVR 处理器相比，ARM 处理器在处理速度、RAM 容量、Flash 容量、引脚数量、成本上都有很大的优势。

尽管性能上不如 Maple，但是与 Maple 相比，Arduino 出现得更早，更加成熟，有更多的扩展和库。

1.2　Maple 的衍生版本

和 Arduino 一样，在发展的过程中，Maple 也出现了为不同目标应用设计的不同版本，这里简单介绍下 Maple Rev5 以外的版本。

1.2.1　Maple RET6

Maple RET6 是 Maple Rev5 的改进版本。硬件上，它与 Maple Rev5 的区别是它使用的是 STM32F103RET6 处理器。与 Maple Rev5 所使用的 STM32F103RBT6（只有一个字符的差别）相比，RET6 多了两路 DAC 输出，片上 Flash 与 SRAM 储存器容量分别由 128 KB 和 20 KB 提升为 512 KB 与 64 KB，其他方面差别不大。

1.2.2　Maple Mini

Maple Mini 具有兼容面包板的双列直插外形，如图 1-1 所示，可以直接用于面包板。外形只有 51.3 cm × 1.82 cm，也非常适合于对于尺寸和重量有较高限制的场合。例如，微型 4 轴飞行器、微型机器人等。Maple Mini 采用的处理器为 STM32F103CBT6，与 Maple 相

比，GPIO（General Purpose Input Output, 通用输入输出）数有所减少。如果需要作为便携设备使用，Maple Mini 是首选。

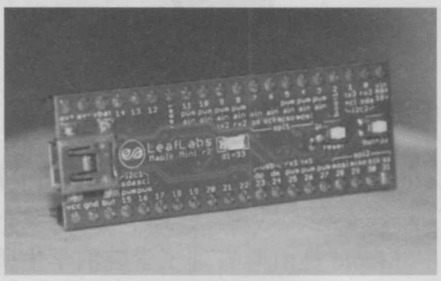


图 1-1 Maple Mini（图片来自 leaflabs.com）

1.2.3 Maple Native

Maple Native 是为了充分发挥处理器性能而设计的高性能开发板，使用了高端 STM32F103 处理器。与 Maple 相比，Maple Native 具有两倍于 Maple 的 GPIO 数并提供外部扩展 SRAM，能够应对高要求的应用。Maple Native 具有 DAC，能够真正实现高精度的模拟输出（Maple 和 Arduino 只能通过 PWM 来输出模拟量）。同时，与其他的高性能开发板相比，在顾及高性能的设计之外，Maple Native 仍然提供与 Arduino 类似的库。

1.3 Maple 的硬件资源

Maple 采用的是意法半导体集团生产的 STM32F103RBT 处理器，这款处理器是以 ARM Cortex-M3 为核心，工作在 72 MHz 的频率下。与 Arduino 所使用的 8 位处理器相比，Maple 的处理器性能和硬件资源有了很大的提升，但是却有相近的成本，为电子积木拓宽了应用的空间。Maple 及其衍生版本与 Arduino 的性能对比如表 1-1 所示。Maple 的布局如图 1-2 所示。

表 1-1 Maple 及其衍生版本与 Arduino 性能对比

比 较 项	Maple Rev5	Maple Mini	Maple Native	Arduino UNO
处理器型号	STM32F103RBT6	STM32F103CBT6	STM32F103ZET6	AtMega328
处理器时钟	72 MHz	72 MHz	72 MHz	16 ～ 20 MHz
SRAM	20 KB	20 KB	64 KB+1 MB(EXT)	2 KB
Flash	128 KB	128 KB	512 KB	32 KB
EEPROM	Flash 虚拟	Flash 虚拟	Flash 虚拟	1 KB
可用 GPIO	43	34	106	14
ADC	15(12 位分辨率)	9(12 位分辨率)	21(12 位分辨率)	6(10 位分辨率)
PWM	15(16 位分辨率)	12(16 位分辨率)	17(16 位分辨率)	6(8 位分辨率)
硬件 SPI	2	2	3	1
硬件 I^2C	2	2	2	1
硬件 USART	3	3	3	1
硬件 USB	1	1	1	无
中断	42 个共用 16 通道		106 个共用 16 通道	2 个
DAC	0	0	2(12 位分辨率)	0

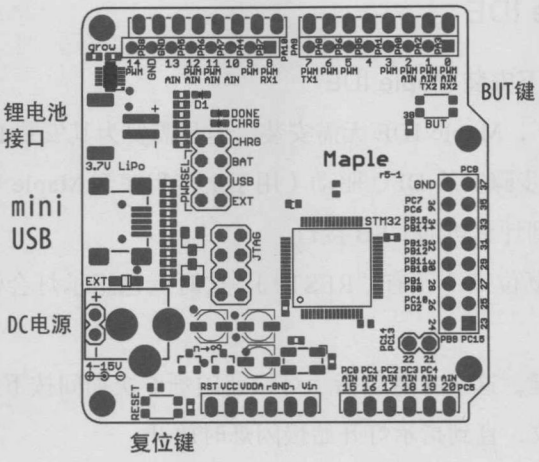


图 1-2　Maple 的布局

1.4　libmaple 简介

libmaple 是由 Leaflab 开发的，用于 STM32 库，对 STM32 底层操作进行了重新封装，大幅提高了操作的简便性。

libmaple 分为两部分：第一部分，以 C 程序编写的底层库，将其称为 libmaple，提供对硬件高度灵活的操作，是严格意义上的 libmaple，也是本章将要介绍的 libmaple，在源码中的 /libmaple 目录下；第二部分，以 C++ 编写的 Wiring 风格的 API，该部分提供对第 2 章中介绍的 Arduino 兼容库部分的支持，位于源码中的 /wirish 目录下。

libmaple 的源码位于 GitHub 上，可以在 https://github.com/leaflabs/libmaple 中找到。

libmaple 附随于 Maple IDE 中，可以直接在 Maple IDE 的代码中加入对 libmaple 头文件的引用，同时 libmaple 也可以用于其他开发平台包，给习惯使用专业嵌入式开发 IDE 的用户使用。

1.5　Maple IDE 的安装和使用

1.5.1　下载 Maple IDE

目前 Maple IDE 支持 Mac、Windows 和 Linux，可以在其官方网站 http://leaflabs.com/docs/maple-ide-install.html 下载到最新版本的 Maple IDE。将下载的 ZIP 文件解压到合适的位置。

1.5.2 安装 Maple IDE

1. Windows 环境下安装 Maple IDE

在 Windows 环境下，Maple IDE 无需安装，但是需要为其安装 DFU 和虚拟串口两个驱动程序。首先按照如下步骤安装 DFU 驱动（用于上传程序到 Maple）程序：

1）将 Maple 连接到计算机的 USB 接口。

2）按下左下角的复位键（印有“REST”），这时蓝色指示灯会快速闪烁 6 次，然后慢闪烁几次。

3）再次按下复位键，这次在蓝色指示灯快速闪烁 6 次期间按下另一个键（右上角，印有“BUT”）并保持不放，直到指示灯开始慢闪烁时放开。

4）这时 Maple 会处于永久 Bootloader 状态（Perpetual Bootloader Mode），蓝色指示灯会一直闪烁，使你有机会安装 DFU 驱动程序。

5）Windows 会提示你需要驱动程序，人工指定驱动程序所在目录位置，选择 Maple IDE 文件夹中的 drivers\mapleDrv\dfu。

下一步安装虚拟串口驱动程序（用于通过 USB 与 Maple 进行串口通信）：

1）复位 Maple，等待蓝色指示灯停止闪烁（退出 bootloader 转而运行用户程序，新的 Maple 会运行制造时预留的测试程序）。

2）一旦 Maple 运行了用户程序，Windows 会提示安装驱动程序，同样，人工指定驱动程序所在目录为 Maple IDE 文件夹中的 drivers\mapleDrv\serial。

现在可以双击 Maple IDE 程序运行 Maple IDE 了。

2. Linux 环境下安装 Maple IDE

在 Linux 环境下安装 Maple IDE 的步骤如下：

1）首先确认是否已经安装了 JRE，如果没有，可以通过如下命令安装：

```
sudo aptitude install openjdk-6-jre
```

2）解压压缩文件到合适的位置（例如桌面）。

3）打开解压压缩文件得到的文件夹，运行其中的 install-udev-rules.sh，出现要求输入用于获得管理员权限的密码的提示。

4）用“sudo restart udev”命令重启 udev。

5）双击程序运行 Maple IDE。

3. Mac 环境下安装 Maple IDE

在 Mac 环境下安装 Maple IDE 的步骤如下：

1）双击你所下载的 DMG 文件使该镜像被加载。

2）拖动 Maple IDE 按钮到应用程序文件夹。

3）运行 Maple IDE。

1.5.3　第一个程序

我们从 File → Examples → Digital 中选择一个简单的例程 Blink，如图 1-3 所示，这个程序会让蓝色指示灯闪烁。

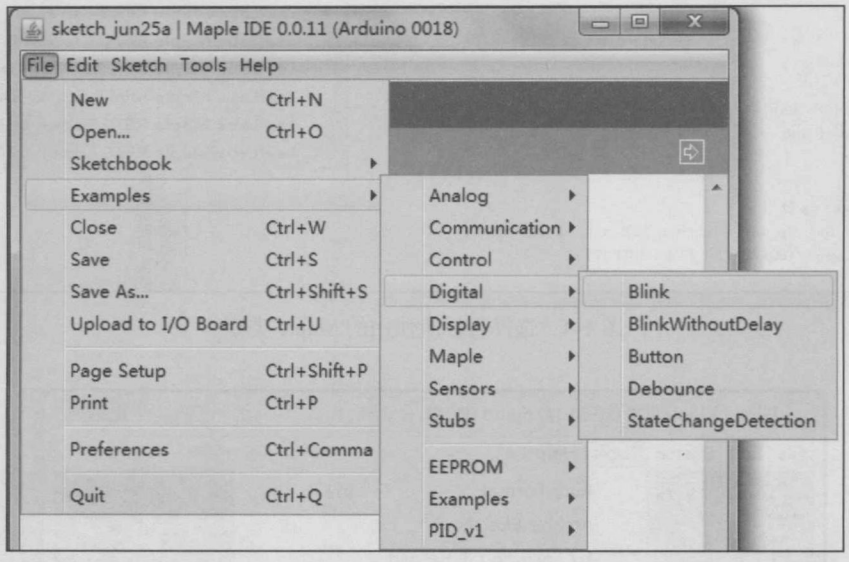


图 1-3　打开例程“Blink”

然后进入 Tools → Board → LeafLabs Maple ... to Flash（根据你所选使用的型号选择），如图 1-4 所示。每种型号的 Maple 都有两个选项 to Flash 和 to RAM。to RAM 会将程序上传到 Maple 的 RAM 存储器，与上传到 Flash 相比，上传速度更快，并且通过简单的复位或者重新上电就可以清除程序。但是 Flash 空间更大，并且是可以在掉电后存储程序的唯一选择。如果需要调试程序，那么 to RAM 模式能够提高你的效率。

现在，可以单击左上角的（Verify）按钮来编译程序，编译的过程与结果会显示在底部的窗口中。该过程主要用于验证程序，检验程序是否有语法或函数引用错误。

现在，将 Maple 用 USB Mini-B 接口的 USB 线连接到计算机上。每次复位、重编程或接上 USB，蓝色指示灯都会闪烁一段时间，表示它进入了 bootloader 状态。等到 Maple 开始执行用户程序，进入 Tools → Serial Port 选择 Maple 在你系统中所使用的串口设备，如图 1-5 所示。在不同的计算机不同的 USB 接口上，Maple 会占用不同的串口，可以通过查看设备管理器来了解 Maple 所占用的串口。

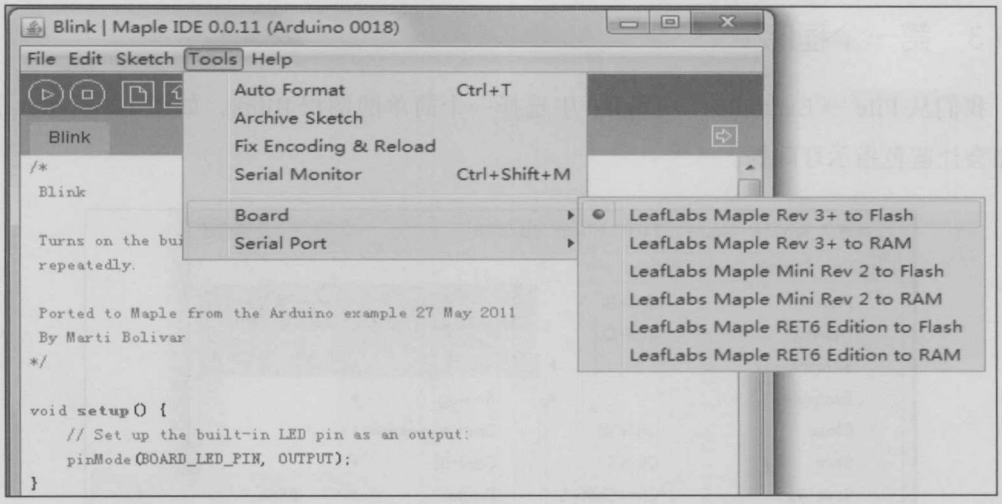


图 1-4　选择目标所使用的 Maple 类型

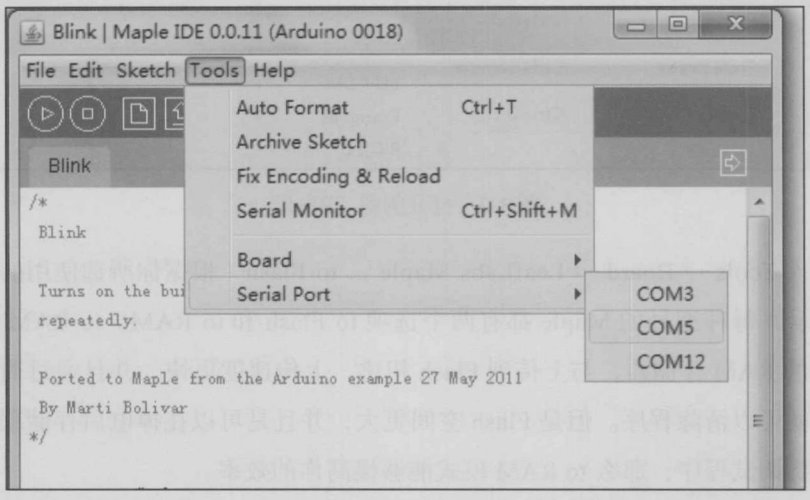


图 1-5　选择 Maple 所使用的串口

单击（Upload）按钮开始上传程序，上传过程会在下方窗口显示。

Maple IDE 是利用 USB 虚拟串口对 Maple 发送复位信号，复位后 Maple 会进入 DFU 模式片刻等待上传，如果没有上传程序，则会在数秒后开始执行原有的用户程序。

USB 虚拟串口相关程序嵌入用户程序中，所以如果前一次上传程序出错，或者错误地选择了串口设备，都可能导致正常的下载操作无法进行。你可以在提示等待 DFU 设备的时候对 Maple 进行复位，或让 Maple 进入“永久 bootloader 状态”之后再进行下载，来规避由于 USB 虚拟串口通信问题造成的无法正常下载的问题。

最后，为了保证一切正常，我们可以上传一个通过 USB 虚拟串口发送文本“Hello,world!”的例程。单击 File → Examples → Stubs → Helloworld 打开该例程，单击（Upload）按钮将其上传到刚才已连接的 Maple 设备上。待 Maple 复位并进入用户程序（蓝色指示灯停止闪烁并听到第二声 USB 设备插入声），单击（Serial Monitor window）按钮打开串口监视器，如果你的串口设置正确，就可以看到 Maple 通过虚拟 USB 串口传回的文字。

1.5.4　Maple IDE 的使用

与 Arduino IDE 一样，Maple IDE 也是在 Processing IDE 的基础上开发的，且简单易用，非常容易上手。下面对 Maple IDE 做一些简单的介绍。Maple IDE 的界面主要有代码编辑区、工具栏和状态栏组成，如图 1-6 所示。

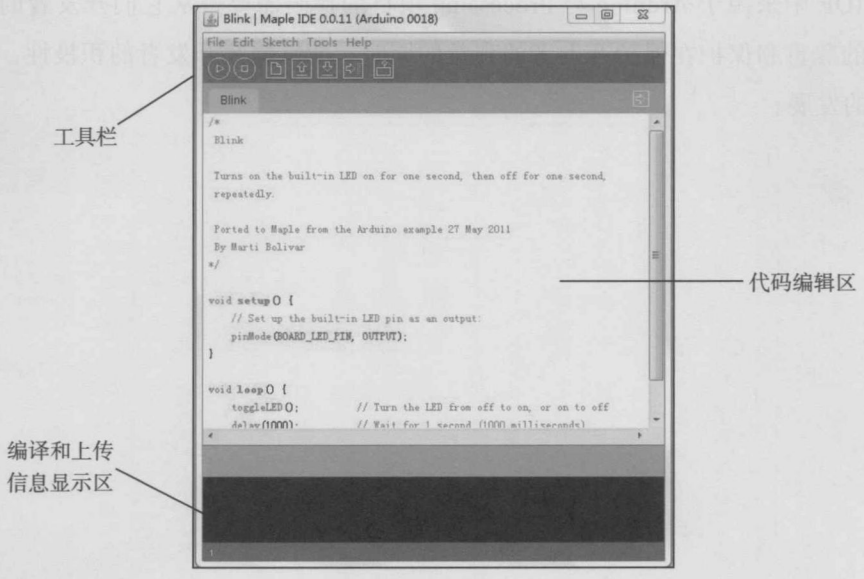


图 1-6　Maple IDE 界面

Maple IDE 的基本功能主要依靠上方的工具栏按钮来实现，如表 1-2 所示为工具栏按钮及其功能。

表 1-2 工具栏按钮及其功能

按 钮	功 能
	单击该按钮会对程序进行编译，在编译过程中检查程序错误，如果没有错误就会产生能在 Maple 上运行的程序
	终止编译过程
	新建一个项目
	打开一个已经存在的项目
	保存当前项目
	将程序上传到 Maple，该过程会重新编译项目
	打开串口监视器（注意：串口监视器打开时不能进行上传，如果上传，窗口监视器会自动关闭。）

1.6 Maple 的开源协议

作为开源硬件的一员，Maple 开发者给予 Maple 最宽松的协议，设计图以 Creative Commons share-a-like 方式发布，任何人都可以任意地使用、修改、重新发布 Maple 的设计和代码。但 Maple IDE 中来源于 Arduino 与 Processing 项目的代码需要遵从它们开发者的协议。对开源协议的尊重和保护在保护开发者的利益的同时，能够提高开发者的积极性，让开源事业有更好的发展。

第 2 章　Maple 的 Arduino 兼容函数库

2.1　基本程序结构

Maple 的基本程序结构有两个入口函数，setup() 与 loop()。setup() 中的代码只会在启动时执行一次，loop() 中的代码会无限循环地执行下去。代码清单 2-1 为 Maple 的基本程序结构。

代码清单 2-1　Maple 的基本程序结构

```
void setup(){
// 这里是初始化使用的程序段，只在启动时执行一次
}
void loop(){
// 主程序位于这里，在这个函数中的程序会无限循环地执行
}
```

Maple 中，与底层相关的大量操作都被隐藏起来了，尽管 Maple 使用的是 C++ 作为编程语言，但是几乎不会出现复杂的 C++ 语法，使得它很适合于对编程与寄存器操作了解不多的初学者使用。

2.2　Maple 静态变量关键字

Maple 中提供了一些静态变量关键字来替代一些常数和引脚，它们可以规避不同型号开发板所产生的兼容性问题。使用这些静态变量有助于提高程序的可移植性与可读性。如表 2-1 所示为静态变量关键字，如表 2-2 所示为串口相关的常量。

表 2-1　静态变量关键字

关 键 字	说　明
CYCLES_PER_MICROSECOND	处理器每毫秒的循环次数。Maple 中为 72
CLOCK_SPEED_MHZ	以兆赫兹为单位的处理器时钟频率
CLOCK_SPEED_HZ	以赫兹为单位的处理器时钟频率
SYSTICK_RELOAD_VAL	Systick 计数器重载次数
BOARD_BUTTON_PIN	代表自带按键（标有“BUT”）所在的引脚
BOARD_LED_PIN	代表 LED 所在引脚的编号

（续）

关 键 字	说　明
BOARD_NR_GPIO_PINS	所有引出的 GPIO 引脚数量。某些引脚可能已经被占用（例如内置的 LED 和按键）。为了了解有哪些引脚被使用，可以使用 boardUsesPin() 函数（或者 boardUsedPins）
BOARD_NR_PWM_PINS	所有可以用于 PWM 输出的引脚数量
BOARD_NR_ADC_PINS	可用 ADC 引脚数量
BOARD_NR_USED_PINS	已使用的 GPIO 引脚数量

表 2-2　串口相关的常量

关 键 字	说　明
BOARD_USART1_TX_PIN,BOARD_USART2_TX_PIN, BOARD_USART3_TX_PIN	3 个 USART 串口的发送引脚
BOARD_USART1_RX_PIN,BOARD_USART2_RX_PIN, BOARD_USART3_RX_PIN	3 个 USART 串口的接收引脚
BOARD_UART4_TX_PIN, BOARD_UART5_TX_PIN	USART4、5 的发送引脚（例如 Maple Native 的）
BOARD_UART4_RX_PIN, BOARD_UART5_RX_PIN	USART4、5 的接收引脚（例如 Maple Native 的）
BOARD_NR_USARTS	可用的 USART 串口数量

2.3　通用输入输出

通用输入输出 (GPIO) 是 Maple 最基本、最常用的功能，用来实现基本的数字量输入和输出。GPIO 的控制主要依赖 pinMode()、digitalWrite()、digitalRead() 三个函数。

由于 Maple 使用的 STM32 采用了更高的工艺生产，所以数字输出的电平为高电平——3.3 V 的 LTTL 电平，而 Arduino 使用的是 ATMGA 系列芯片，制程较陈旧，输出为高电平——5 V 的 TTL 电平。如果需要将 5 V 电平的芯片与 Maple 相接，则需要进行电平转换，否则 5 V 电平的信号直接输入 Maple 可能会造成芯片损坏（部分引脚能够容忍 5 V 电平，对 5 V 电平的数字信号进行读取或是利用开漏输出方式输出 5 V 电平的数字信号）。

2.3.1　pinMode() 函数

形式：void pinMode(uint8 pin, WiringPinMode mode)

参数：pin 为引脚编号。

mode 为引脚的输入输出模式。

pinMode() 函数常放在 setup() 函数中来确定引脚的功能。切记，如果在使用某引脚前没有设定 pinMode() 或者 pinMode 设置模式不正确，引脚输入输出过程可能会出现一些不可预料的错误。pinMode() 输出模式种类如表 2-3 所示。