

国外计算机科学经典教材

THOMSON

Embedded C Programming and the Atmel AVR

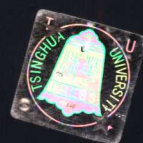
嵌入式 C 编程 与 Atmel AVR

Richard Barnett

(美) Larry O'Cull 著

Sarah Cox

周俊杰 等译



清华大学出版社

嵌入式 C 编程与 Atmel AVR

Richard Barnett
(美) Larry O'Cull 著
Sarah Cox

周俊杰 等译

清华大学出版社

北 京

Richard Barnett, Larry O'Cull, Sarah Cox

Embedded C Programming and the Atmel AVR

EISBN: 1-4018-1206-6

Copyright © 2003 by Delmar Learning, a division of Thomson Learning.

Original language published by Thomson Learning(a division of Thomson Learning Asia Pte Ltd).

All Rights reserved.

本书原版由汤姆森学习出版集团出版。版权所有，盗印必究。

Tsinghua University Press is authorized by Thomson Learning to publish and distribute exclusively this Simplified Chinese edition. This edition is authorized for sale in the People's Republic of China only (excluding Hong Kong, Macao SAR and Taiwan). Unauthorized export of this edition is a violation of the Copyright Act. No part of this publication may be reproduced or distributed by any means, or stored in a database or retrieval system, without the prior written permission of the publisher.

本中文简体字翻译版由汤姆森学习出版集团授权清华大学出版社独家出版发行。此版本仅限在中华人民共和国境内(不包括中国香港、澳门特别行政区及中国台湾地区)销售。未经授权的本书出口将被视为违反版权法的行为。未经出版者预先书面许可，不得以任何方式复制或发行本书的任何部分。

981-243-982-x

北京市版权局著作权合同登记号 图字：01-2003-2176

本书封面贴有清华大学出版社激光防伪标签，无标签者不得销售。

图书在版编目(CIP)数据

嵌入式 C 编程与 Atmel AVR/(美)巴雷特，(美)古尔等著；周俊杰等译.—北京：清华大学出版社，2003

书名原文：Embedded C Programming and the Atmel AVR

ISBN 7-302-06955-7

I.嵌… II.①巴…②古…③周… III.①单片微型计算机，AVR②C 语言—程序设计 IV.①TP368.1②TP312

中国版本图书馆 CIP 数据核字(2003)第 064032 号

出 版 者：清华大学出版社

地 址：北京清华大学学研大厦

<http://www.tup.com.cn>

邮 编：100084

社 总 机：010-62770175

客户服务：010-62776969

组稿编辑：曹康

文稿编辑：陈宗斌

封面设计：康博

版式设计：康博

印 刷 者：北京牛山世兴印刷厂

发 行 者：新华书店总店北京发行所

开 本：185×260 印张：27 字数：684 千字

版 次：2003 年 9 月第 1 版 2003 年 9 月第 1 次印刷

书 号：ISBN 7-302-06955-7/TP·5137

印 数：1~4000

定 价：54.00 元

前 言

本书不仅讲授 C 语言嵌入式微控制器的编程方法,还提供了应用 Atmel 公司生产的 AVR RISC 系列微控制器的相关知识。

适用对象

本书是针对两类不同的读者而设计的。

- 第一类读者是电子计算机工程、电子工程、电气工程技术、电子工程技术及计算机工程技术专业的学生。本书很适合以下两类学生:
 - ◆ 没有学过 C 语言的学生:本书可以作为两学期或 4 个季度课程的课本。这样,学生就可以先学习 C 语言编程,再学习如何在嵌入式微控制器设计中应用 C 语言。按照这个学习顺序,他们就可以学习更复杂的嵌入式应用程序,这些程序应该都能运行在嵌入式微控制器中,而几乎不需要硬件知识。第 1 章(嵌入式 C 语言导论)将作为后续课程的参考资料。
 - ◆ 学过 C 语言的学生:本书可以作为一学期或两个季度课程的课本。在这种情况下,学生只需要学习第 1 章中有关嵌入式环境编程的内容,然后就可以迅速进入硬件知识的学习。第 1 章的内容为以后的学习提供必要的参考信息。
- 第二类读者是那些想在工作领域中增加微控制器应用知识的工程师和技术人员。根据不同的需要,第 1 章可以作为学习的内容,或者只是作为参考资料(这将由读者的编程水平而定)。而有关 Atmel AVR 微控制器硬件知识的章节,将引导读者去学习一个新的微控制器。同时它们也都可作为以后学习研究的参考资料。

必备知识

本书要求读者有数字系统和逻辑设计的知识。第 1 章中的相关知识已成功地在微控制器基础课程上使用了。本书是两个学期的基础数字逻辑课程的后续课程(相当于大学二年级水平,没有编程基础)。同时本书也是很好的高级微控制器选修课的教材。很多时候,选修了该课的学生都会保留这本书,以作为以后高级课程项目设计时的参考资料,有的甚至把它当作参考书带到工厂去了。

本书的组织

本书是根据内容的逻辑主题来划分各部分的，所以教师可以依照本书的组织顺序来讲授，从 C 语言开始，然后是 AVR 的硬件，再进入更高级的主题。也可以根据自己具体的需要选择相应的内容。每一个主题都是相对独立，自成一体的。每章后面的练习和上机实习题目都是针对相应主题的，这样可以使读者方便、容易地在具体应用中使用它们。

各章内容简介

第 1 章 嵌入式 C 语言导论

本章详细地讲述了 C 语言，并按部就班地将其应用到嵌入式微控制器的编程中。每个编程概念都有一到两个例子演示它们的用法。学习完本章后，学生就可以编写 C 语言程序来解决问题。

第 2 章 Atmel RISC 处理器

本章内容覆盖了微控制器的基本结构及其内部的每一个标准外设。同样利用示例程序来演示每个外设的常规用法。完成了前面两章的学习之后，学生应该可以利用 AVR RISC 微控制器来解决问题了。

第 3 章 标准 I/O 和预处理函数

本章向学生介绍 C 语言的内部函数及用法。同样，也是利用示例程序来演示这些内部函数的用法。完成第 3 章的学习后，就可以利用库函数加快编程速度并提高解决问题的能力。

第 4 章 CodeVisionAVR C 编译器和集成开发环境

本章是 CodeVisionAVR 编译器及其集成开发环境的使用手册。在本章中，可以学习如何利用 CodeVisionAVR 和 IDE 来有效地创建和调试 C 语言程序。

第 5 章 项目开发

本章集中讨论了应用微控制器的项目的开发过程。通过开发一个完整的无线室内/户外气象台系统来演示这一过程。在本章中，可以学习如何高效地开发一个项目，以取得最大的成效。

附录

附录 A 库函数参考 包含到本书出版时为止，所有可以获得的内部库函数的完整参考资料。

附录 B CodeVisionAVR 和 STK500 入门 本附录是利用 CodeVisonAVR 开发 Atmel STK500 芯片的快速使用指南。

附录 C AVR 微控制器编程 这是 AVR 设备的 FLASH 存储器区域编程的操作指南，通过该指南，可以让您明白编程的原理。

附录 D 安装和使用 CableAVR 该用户手册介绍了如何安装和使用 Cable AVR 软件及硬件。

附录 E MegaAVR-DEV 开发板 MegaAVR-DEV 开发板的说明书和简图。

附录 F ASCII 字符表。

附录 G AVR 指令集汇总 这是示例程序中所用到的汇编语言指令的简要总结。

附录 H 部分练习答案。

基本目的

围绕着微控制器的技术在不断地发展与推进，不停地为微控制器提供更多的功能和更快的速度。这使人们更普遍使用高级语言(例如 C 语言)来处理一些对时间要求很严格的任务，以前这种任务只能用汇编语言来完成。同时，微控制器的应用变得越来越容易了，这使它成为一种很好的教学设备。很多学校都采用微控制器作为他们课程的实验器材。另外，微控制器开发板的价钱已经降低到了一个可以接受的水平，很多学校都可以要求学生自己买开发板作为整个课程的配件，这样每个学生都可以拥有自己的开发板。有的课程需要有 C 语言的基础，有的课程则把 C 语言和嵌入式微控制器的应用综合起来。

本书满足了以上两种课程的要求，并可以作为学生学习后续课程的参考书。

使用的硬件

本书中大部分编程应用实例都是由 Progressive Resources, LLC 公司提供的 AVR 测试版开发板(详细说明请参看附录 E)开发的。这一开发板很适合教学使用，也是很好的通用开发板。Atmel AVR 微控制器非常容易使用。把它插在模型板上，外加一个晶体振荡器，几个电容；以及 4 根用于编程的数据线就可以让它运行得很好。两种方法都可以取得很好的效果。

本书的例子已经在微控制器 AT90S8535 或者 Atmega163 上测试过。AVR 系列微控制器的一大优点是，它们的结构以及设备的编程方式都很相似。因此，所提供的例子可以在任何 Atmel AVR 微控制器上运行，只要它上面有相应的外设和资源——没有必要为 AVR 系列其他的微控制器改变代码。因此，本书也适用于 AVR 系列的其他微控制器。

本书涵盖了最常用的外设，当遇到 AVR 系列其他成员的特殊外设时，本书的代码可以作为参考模板。

作者简介

本书是 3 位作者高度协作的成果。每部分都是先由其中一个人编写，写完以后再由其他两个人认真地审阅，如果有需要，他们会重写其中大部分的内容。所以很难指出那一个作者对书中那一个部分负责。

本书的作者是：

Richard H. Barnett

电子工程技术教授

珀杜大学

Barnett 博士在过去的 18 年里一直从事嵌入式微控制器领域的教学工作。从 Intel 公司的 8085 开始, 到 8051 系列嵌入式微控制器, 现在则利用 Atmel AVR 设备从事高级嵌入式微控制器的教育工作。在暑假和另外两个假期里, 他广泛地从事多处理器嵌入式系统的研究工作, 并将它们应用到各种面向控制的应用中。另外, 他还积极地在上述领域从事咨询工作。在珀杜大学任职期间, 他当了 10 年的航空电子工业工程师。

在教学方面, Barnett 博士获得了很多奖项, 包括作为珀杜大学中最好的教师而获得了 Charles B. Murphy 奖。他同时被列入珀杜大学的知名教师手册中, 这个知名教师手册上只列举了珀杜大学中最有影响的 225 位教师。本书是他写的第二本教材。

如果您有任何意见或者建议, 可以通过珀杜大学的电话 765-494-7497 和他取得联系, 也可以发送电子邮件到 rbarnett@purdue.edu

Larry D. O'Cull

资深操作员

Progressive Resources LLC 公司

O'Cull 先生从珀杜大学的电子工程技术学院获得理学学士学位。他开始是为 CNC(Computer Numeric Controlled, 计算机数值控制)设备设计软件和控制系统的, 后来转向电子工程的其他领域, 并为各种各样的系统和设备开发软件, 如视觉系统, 激光机械设备, 医疗诊断设备和各种工业品或消费品等。同时, 他还是多项专利的拥有者或者共同拥有者。

O'Cull 在从事了几年的电子和软件工程师工作以及工程管理工作后, 于 1995 年成立了 Progressive Resources 公司。Progressive Resources LLC 公司(<http://prllc.com>)专门从事商业、工业和消费品的新产品开发工作。该公司还是 Atmel AVR 顾问团的成员。

如果有任何意见或者建议, 可以发送电子邮件到 locull@prllc.com 和他取得联系。

Sarah A. Cox

公司

软件开发主任

Cox 女士在珀杜大学学习软件设计并获得了计算机和电子工程两个理学学士学位。

她曾在很大的顾问公司中工作过一段时间, 为数据库管理系统提供咨询, 之后她被微处理器设计的快速节奏和无限魅力吸引了。在成为 Progressive Resources LLC 公司的合作者之前, 她曾独立完成了很多医疗测试设备的设计。

在 Progressive Resources 公司中, Cox 女士为各种项目开发软件, 范围涉及最小的消费品到工业产品和测试设备。这些项目跨越了很多领域, 包括自动化、医疗、娱乐、儿童发展、公共安全/教育、声音和图像压缩、建筑等。除此之外, 她还是很多专利的共同拥有者。她也为系统编程编写软件, 为 Atmel AVR 处理器编写开发工具。

如果有任何意见或者建议, 请将电子邮件发送到 sac@prllc.com 和她取得联系。

引 言

嵌入式微控制器就是在同一块集成电路中集成了 CPU、主要外设和所需内存的微型计算机，实际上就是在一块芯片上的微型计算机。

嵌入式微控制器的使用历史已经超过 30 年了。Intel 公司的 8051 系列是最早把内存、I/O、算术逻辑部件(ALU)、程序 ROM，以及其他外设都封装到一个小巧的芯片中的微控制器之一。这些处理器现在还被用来设计新产品。跟在 Intel 公司之后进入嵌入式微控制器竞争市场的公司有 General Instruments、National Semiconductor、Motorola、Philips/Signetics、Zilog、AMD、Hitachi、Toshiba 和 Microchip 公司，以及其他公司。

最近几年，Amtel 成为 FLASH 内存技术开发的世界领头人。FLASH 是一种具有非易失性(nonvolatile)但可以重复编程的存储器，它通常用在数码像机、便携式音响设备和 PC 主板上。这种存储技术由于能提供嵌入式系统编程的解决方案，所以大大推进了 Atmel 在微控制器领域中的发展。这样，结合 AVR RISC(Reduced Instruction Set Computing，精简计算指令集)核心结构的发展，就能在对功耗要求很低的情况下提供令人惊讶的功能。

这个高科技领域的另一大发展就是专门针对这类微控制器的高级语言编译器的出现。这些编译器生成代码和优化代码的能力惊人。由于 C 编程语言它有自由的风格和结构，以及代码的可移植性，使它在这方面的应用上具有优势。这类语言最关键的益处在于它创建的智能属性池可以重用。这些底层的开发可以降低随后设计的开发周期，从而降低成本。

目前，最好的一个 C 语言开发工具就是 CodeVisionAVR。它是由 HP InfoTech S.R.L 的 Pavel Haiduc 编写的。这个完整的集成开发环境(IDE)可以允许在 PC 机的 Windows 应用程序中进行编辑、编译、分块编程和调试。

AVR 和其他 RISC 微控制器的流行、集成水平的不断提高(一个芯片中集成越来越多器件，而电路板上的器件却越来越少)，以及在利用这些技术开发产品时对“协调思想”的需求，促使我们产生了编写这本书的动机。您可能有过为 PC 编写 C 语言程序，或者为微控制器编写汇编语言程序的经历。但是在为嵌入式微控制器编写 C 语言代码时，要改变所用的方法，以达到期望的结果——短小、高效、可靠及可重用的代码。本书的目的之一就是为初学者提供一个比较好的起点，同时，本书对于那些在嵌入式微控制器设计上有丰富经验的人来说，也是一本有用的参考书。

目 录

第 1 章 嵌入式 C 语言导论	1
1.1 本章目标	1
1.2 引言	1
1.3 基本概念	1
1.4 变量和常量	4
1.4.1 变量类型	4
1.4.2 变量的作用域	4
1.4.3 常量	5
1.4.4 枚举和定义	7
1.4.5 存储类型	8
1.4.6 类型转换	8
1.5 输入/输出操作	9
1.6 运算符和表达式	10
1.6.1 赋值运算符和算术运算符	10
1.6.2 逻辑运算符与关系运算符	13
1.6.3 自增运算符、自减运算符和复合赋值运算符	14
1.6.4 条件表达式	15
1.6.5 运算符优先级	16
1.7 控制语句	17
1.7.1 while 循环	17
1.7.2 do/while 循环	18
1.7.3 for 循环	19
1.7.4 if/else 语句	20
1.7.5 switch/case 语句	23
1.7.6 break、continue 和 goto 语句	25
1.8 函数	29
1.8.1 原型和函数组织	30
1.8.2 函数返回值	32
1.8.3 递归函数	33
1.9 指针和数组	37
1.9.1 指针	37
1.9.2 数组	40
1.9.3 多维数组	42

1.9.4 指向函数的指针	44
1.10 结构与共用体	48
1.10.1 结构	48
1.10.2 结构数组	50
1.10.3 指向结构的指针	51
1.10.4 共用体	52
1.10.5 typedef 运算符	54
1.10.6 位和位域	55
1.10.7 sizeof 运算符	56
1.11 存储器类型	57
1.11.1 常量和变量	57
1.11.2 指针	59
1.11.3 寄存器变量	59
1.12 实时方法	62
1.12.1 使用中断	62
1.12.2 状态机	65
1.13 本章小结	71
1.14 练习	71
1.15 上机实习	73
第 2 章 Atmel Risc 处理器	75
2.1 本章目标	75
2.2 引言	75
2.3 体系结构概述	75
2.4 存储器	76
2.4.1 FLASH 代码存储器	77
2.4.2 数据存储器	77
2.4.3 EEPROM 存储器	81
2.5 复位和中断功能	82
2.5.1 中断	83
2.5.2 复位	86
2.6 并行 I/O 端口	88
2.7 计时器/计数器	93
2.7.1 计时器/计数器预定标器和输入选择器	93
2.7.2 Timer 0	94
2.7.3 Timer 1	97
2.7.4 Timer 2	108
2.8 使用 UART 进行串行通信	112

2.9 模拟接口	118
2.9.1 模数转换背景知识	118
2.9.2 模数转换器外设	119
2.9.3 模拟比较器	122
2.10 利用 SPI 进行串行通信	127
2.11 AVR RISC 汇编语言指令集	130
2.12 本章小结	132
2.13 练习	136
2.14 上机实习	137
第 3 章 标准 I/O 和预处理函数	139
3.1 本章目标	139
3.2 引言	139
3.3 字符输入/输出函数 getchar()和 putchar()	139
3.4 标准输出函数	145
3.4.1 输出字符串函数 puts()	145
3.4.2 从 FLASH 输出字符串函数 PUTSF()	146
3.4.3 格式打印函数 printf()	147
3.4.4 字符串格式打印函数 sprintf()	149
3.5 标准输入函数	150
3.5.1 获得字符串函数 gets()	150
3.5.2 格式扫描函数 scanf()	151
3.5.3 字符串格式扫描函数 sscanf()	152
3.6 预处理指令	153
3.6.1 #include 指令	153
3.6.2 #define 指令	154
3.6.3 #ifdef, #ifndef, #else 和 #endif 指令	156
3.6.4 #pragma 指令	162
3.6.5 其他宏和指令	165
3.7 本章小结	166
3.8 练习	166
3.9 上机实习	167
第 4 章 CodeVisionAVR C 编译器和集成开发环境	168
4.1 本章目标	168
4.2 引言	168
4.3 集成开发(IDE)环境操作	169
4.3.1 项目	169
4.3.2 源文件	171

4.3.3	编辑文件	175
4.3.4	打印文件	176
4.3.5	文件导航器	176
4.4	C 编译器选项	177
4.4.1	UART	178
4.4.2	存储器模式	178
4.4.3	优化	179
4.4.4	程序类型	179
4.4.5	SRAM	179
4.4.6	编译	179
4.5	编译和生成项目	180
4.5.1	编译项目	180
4.5.2	生成项目	182
4.6	对目标设备编程	183
4.6.1	芯片	183
4.6.2	FLASH 和 EEPROM	183
4.6.3	FLASH 锁定位	185
4.6.4	保险位	185
4.6.5	Boot Lock Bit 0 和 Boot Lock Bit 1	185
4.6.6	签名	185
4.6.7	芯片擦除	186
4.6.8	编程速度	186
4.6.9	Program All	186
4.6.10	其他编程器	186
4.7	CodeWizardAVR 代码生成器	188
4.7.1	Chip 选项卡	189
4.7.2	Ports 选项卡	190
4.7.3	External IRQ 选项卡	191
4.7.4	Timers 选项卡	192
4.7.5	UART 选项卡	193
4.7.6	ADC 选项卡	194
4.7.7	Project Information 选项卡	195
4.7.8	生成源代码	196
4.8	终端工具	207
4.9	Atmel AVR Studio 调试器	208
4.9.1	为 AVR Studio 新建一个 COFF 文件	208
4.9.2	从 CodeVisionAVR 启动 AVR Studio	209
4.9.3	打开文件进行调试	209

4.9.4 开始、中断和跟踪	209
4.9.5 设置和清除断点	210
4.9.6 查看和修改寄存器和变量的值	210
4.9.7 查看和修改机器状态	211
4.10 本章小结	211
4.11 练习	212
4.12 上机实习	213
第 5 章 项目开发	214
5.1 本章目标	214
5.2 引言	214
5.3 概念开发阶段	214
5.4 项目开发过程的步骤	214
5.4.1 定义阶段	214
5.4.2 设计阶段	216
5.4.3 测试定义阶段	217
5.4.4 建立和测试硬件原型阶段	217
5.4.5 系统集成和开发阶段	218
5.4.6 系统测试阶段	218
5.4.7 庆祝阶段	218
5.5 项目开发过程总结	218
5.6 示例项目：一个气象监测器	219
5.6.1 构思阶段	219
5.6.2 定义阶段	219
5.6.3 测量方法在设计方面的考虑	224
5.6.4 室外装置的硬件设计	235
5.6.5 室外装置的软件设计	237
5.6.6 室内装置的硬件设计	239
5.6.7 室内装置的软件设计	241
5.6.8 测试定义阶段	243
5.6.9 建立和测试原型硬件阶段	244
5.6.10 系统集成和软件开发阶段——室外装置	250
5.6.11 系统集成和软件开发阶段——室内装置	258
5.6.12 系统测试阶段	284
5.7 挑战	288
5.8 本章小结	288
5.9 练习	288
5.10 上机实习	289

附录 A 库函数参考	290
附录 B CodeVisionAVR 和 STK500 入门	369
附录 C AVR 微控制器编程	381
附录 D 安装并使用 TheCableAVR	384
附录 E MegAVR-DEV 开发板	394
附录 F ASCII 字符表	397
附录 G AVR 指令集汇总	401
附录 H 部分练习答案	409

第1章 嵌入式C语言导论

1.1 本章目标

通过本章的学习，您应该掌握以下内容：

- 定义、描述和识别变量类型与常量类型，以及这些类型的作用域和用法
- 为各种大小的数值数据和字符串构造变量常量声明
- 将枚举应用于变量声明
- 通过赋值运算符给变量和常量赋值
- 评估 C 语言中使用的所有运算符的效果
- 解释每种控制语句对程序流程产生的效果
- 创建包含有变量、运算符和控制语句的函数，以完成特定任务
- 使用指针、数组、结构和共用类型作为函数变量
- 用本章的概念创建 C 语言程序来完成任务

1.2 引言

本章介绍了将 C 语言应用于嵌入式微控制器程序时必须了解的基本知识，包括作为 CodeVisionAVR C 语言的一部分的 C 语言扩展。您将从基本概念的学习开始，然后编写完整的程序，同时通过可在微控制器上实现的例子来加强学习。

本章的介绍是根据程序员需要学习的内容安排的：

- 声明变量和常量
- 简单的输入输出(I/O)，让程序可以通过 I/O 来使用微控制器的并行端口
- 变量和常量赋值，对变量执行算术运算
- C 构造语句与控制语句，通过这些语句来形成完整的 C 程序

最后一节将介绍更高级的主题，例如指针、数组、结构和共用类型，以及它们在 C 程序中的用法。最后以实时程序设计和中断等高级概念来结束本章。

1.3 基本概念

编写 C 程序在某种意义上来说就好像是用砖建房子：首先打好地基，使用沙子和水泥造砖，把这些砖堆砌起来，然后建成房子。在嵌入式 C 语言程序中，指令集放在一块形成函数，函数

被当作更高一级的运算，由函数结合组成程序。

每个 C 语言程序至少有一个函数，即 `main()` 函数。`main()` 函数是 C 语言程序的基础，是程序代码执行的起点。所有的函数都是通过 `main()` 函数直接或间接调用的。尽管函数可以是完整的和独立的，但变量和参数可以同函数结合在一起。

`main()` 函数通常被认为是最低级的任务，因为它是启动该程序的系统所调用的第 1 个函数。在很多情况下，`main()` 都只包含很少的语句，这些语句的作用仅仅是初始化和指导从一个函数到另一个函数的程序操作。

一个最简单的嵌入式 C 语言程序如下：

```
void main()
{
    while (1) // do forever..
        ;
}
```

以上程序可以很好地进行编译和运行，不过您可能不会知道这一点，因为该程序在运行时没有给出任何界面内容。可以对该程序进行改进，以便您能看出它的存在和功能，并且可以开始学习这种语言的语法元素。

```
#include<stdio.h>

void main()
{
    printf("HELLO WORLD");/*the classic C test program..*/
    while(1)           //do forever..
        ;
}
```

以上程序将结果“HELLO WORLD”输出到标准输出设备上，该输出设备最有可能是一个串行端口。如果在微控制器中运行该程序，那么微控制器会一直等待下去，或者直到微控制器重新启动。这种情况显示了 PC 机程序与嵌入式微控制器程序之间的基本区别：嵌入式应用程序包含无限循环。PC 机有操作系统，一旦程序执行，它把控制权交给操作系统。而嵌入式微控制器没有操作系统，不允许随时任意结束程序，所以每个嵌入式微控制器程序都具有无限循环（这个无限循环位于该程序中的某个位置），例如上面例子中的 `while(1)`。这样可以防止程序无事可做或者做不可预测的随机事件。While 结构将在后面的一节中介绍。

上例程序也提供了一个常见预处理编译器(preprocessor compiler)指令的实例。`#include` 告诉编译器包含 `stdio.h` 文件作为本程序的一部分。`printf()` 函数由外部库提供，可以在该程序中使用，是因为该函数的定义位于 `stdio.h` 中文件中。在以后的章节中，这些概念很快就会放在一块讨论。

上述的例子中，有一些值得注意的元素，其含义见表 1-1：

表 1-1 程序元素

元 素	含 义
;	分号用于指示表达式的末尾。一个最简单的表达式就是一个分号本身
{ }	花括号{}用于描述函数内容的开始与结束，还用于指示将一系列语句作为一个整块来对待
"text"	双引号用来标记文本字符串的开始与结束
//或/*...*/	双斜杠或斜杠加星号用作注释定界符

注释只是程序员加上的注解。注释对于程序的可读性十分重要，不管程序是让其他人阅读还是最初的原作者在以后阅读。本书中给出的注释用来解释示例代码中每一行的功能。注释总是解释程序中的代码行的作用，而不是重复在这一行代码中使用的特定指令。

传统的注释定界符使用斜杠-星号(/*)和星号-斜杠(*/)。斜杠-星号用于注释的开始。编译器一旦遇到斜杠-星号(/*)，就忽略后面的文本(即使是多行文本)，直到遇到星号-斜杠(*/)。对于前面的程序而言，main()函数的第1行就是此分界符的例子。

对于双斜杠分界符，编译器只忽略它所在那一行的文本。例如前面程序中 main()函数的第2行。

随着对细节介绍的深入，我们应记住一些语法规则和基本术语：

- 标识符是变量名或函数名，由字母或下划线后跟一连串的字母和/或数字，和/或下划线组成
- 标识符区分大小写
- 标识符可以为任意长度，但有些编译器只识别有限数量的字符，如前32个字符。因此应当小心
- 特定单词对于编译器具有特殊的含义，被认为是保留字。这些保留字应当以小写形式输入，并且从不应用作标识符。下面列出了这些保留字。

auto	defined	float	Long	static	while
break	do	for	Register	struct	
bit	double	funcused	return	switch	
case	eeeprom	goto	short	typedef	
char	else	if	signed	union	
const	enum	inline	sizeof	unsigned	
continue	extern	int	sfrb	void	
default	flash	interrupt	sfrw	volatile	

- 因为C语言是一种自由形式的语言，除非有双引号引用，否则“空白”均被忽略。空白包括空格(间隔)、制表符和新行(回车符和/或换行符)。