· 源代码、PPT免费下载

案例解说单片机C语言开发

——基于AVR+Proteus仿真

程国钢 编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

案例解说单片机C语言开发

——基于AVR+Proteus仿真

程国钢　编著

電子工業出版社

Publishing　　Industry

北京 • BEIJING

内 容 简 介

ICCAVR 是 ATmega128 单片机软件开发环境，Proteus 是目前应用最广泛的硬件仿真环境。本书基于 ICCAVR 和 Proteus 介绍了 ATmega128 单片机体系结构、C 语言、内部资源，以及常用扩展器件的使用方法。全书分为 3 个部分：第 1 章和第 2 章是基础部分，介绍了 ICCAVR 和 Proteus 的基础用法。第 3～9 章是基础应用部分，各个章节基于 ICCAVR 和 Proteus 介绍了 ATmega128 单片机的内部资源和典型外部扩展器件的使用方法，对于这些资源和器件进行了基础知识和 Proteus 库的介绍，还提供了详尽的实例。第 10 章是综合应用部分，介绍了 ATmega128 单片机应用系统的基础设计方法，提供了包括频率计、简易数字时钟、可控自校准数字电源、仓库自动通风控制系统在内的 4 个大型综合应用实例。

本书提供了大量实例，它们都有详细的设计思路、典型器件列表、Proteus 应用电路、C 语言应用代码和仿真运行结果。

本书包含丰富的单片机内部资源和外围模块的应用实例，并且都基于 Proteus 仿真，简单直观，适合具有初步单片机基础的单片机工程师进阶学习，以及高等院校电子类专业的学生和单片机爱好者阅读，也可以作为工程设计人员的参考手册。

图书在版编目（CIP）数据

案例解说单片机 C 语言开发：基于 AVR+Proteus 仿真/程国钢编著. —北京：电子工业出版社，2012.9
ISBN 978-7-121-18018-7

Ⅰ. ①案… Ⅱ. ①程… Ⅲ. ①单片微型计算机—C 语言—程序设计 Ⅳ. ①TP368.1②TP312

中国版本图书馆 CIP 数据核字（2012）第 198806 号

策划编辑：陈韦凯
责任编辑：陈韦凯　　特约编辑：刘丽丽
印　　刷：
装　　订：北京京师印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
开　　本：787×1 092　1/16　印张：24.25　字数：621 千字
印　　次：2012 年 9 月第 1 次印刷
印　　数：4 000 册　　定价：49.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

一、行业背景

ATmega128 单片机具有体积小、功能强、价格低的特点，在工业控制、数据采集、智能仪表、机电一体化、家用电器等领域有着广泛的应用。其应用可以大大提高生产、生活的自动化水平。近年来，随着嵌入式的应用越来越广泛，ATmega128 单片机的开发也变得更加灵活和高效，ATmega128 单片机的开发和应用已经成为嵌入式应用领域的一个重大课题。

二、关于本书

本书以 Proteus 硬件仿真环境和 ICCAVR 软件开发环境为依托，介绍了 ATmega128 单片机的应用方法，包括 ATmega128 单片机的体系结构、C 语言、内部资源的应用、外部器件的扩展应用方法，以及 ATmega128 单片机综合应用系统的开发方法和实例。

本书对于各个内部资源和外围器件介绍的组织结构如下：器件基础、Proteus 中的器件库说明、Proteus 应用电路、应用代码、仿真结果和总结；读者既可以了解该器件的基础知识和对应的驱动代码，也可以在 Proteus 中进行仿真并且观察仿真结果。为与 Proteus 软件中的电路图保持一致，本书电路图中电阻、电容单位的不规范处不作更改，如 10k 不改为 10kΩ，10uF 不改为 10μF，读者能够理解即可。

本书的各个章节说明如下：

- 第 1 章主要介绍了 ATmega128 单片机基础知识，包括 ATmega128 单片机的体系结构、C 语言和 ICCAVR 软件开发环境的基础使用方法。
- 第 2 章主要介绍了 Proteus 硬件仿真环境的基础使用方法，包括其与 ICCAVR 软件开发环境联合调试的方法。
- 第 3 章主要介绍了 ATmega128 单片机的内部资源应用，包括 I/O 引脚、外部中断、定时/计数器、串行口、TWI 接口模块、SPI 接口模块和内置看门狗模块，并且介绍了在 ICCAVR 中建立用户自己的库函数的方法。
- 第 4 章主要介绍了 ATmega128 单片机的人机交互通道使用方法，包括发光二极管、数码管、液晶模块、数字键盘等。
- 第 5 章主要介绍了 ATmega128 单片机的信号采集通道使用方法，包括 A/D 通道、时钟日历芯片、温度传感器等。
- 第 6 章主要介绍了 ATmega128 单片机的信号输出通道使用方法，包括 D/A 通道、I/O 扩展芯片等。
- 第 7 章主要介绍了 ATmega128 单片机的存储器使用方法，包括内部 E^2PROM 和外部 RAM 芯片。
- 第 8 章主要介绍了 ATmega128 单片机的执行机构使用方法，包括三极管、电动机、蜂鸣器等。

- 第 9 章主要介绍了 ATmega128 单片机的通信扩展模块使用方法，包括 RS-232、RS-422 和 RS-485 通信扩展。
- 第 10 章主要介绍了 ATmega128 单片机的综合应用实例，这是对前面所有章节内容的综合实践，提供了频率计、简易数字时钟、可控自校准数字电源等应用实例。

三、本书特色

- 适合具有初步单片机基础的单片机工程师进阶学习，以及高等院校电子类专业的学生和单片机爱好者阅读。
- 涵盖了 ATmega128 单片机从内部资源到用户输入通道、A/D 信号采集、温度/湿度传感芯片、有线通信模块等常用资源或者扩展器件。
- 基于 Proteus 硬件开发环境提供了相应的仿真运行实例及其输出结果。
- 对于相应的资源或者器件的介绍，都按照基础知识、Proteus 库介绍、实例的设计思路和应用代码、实例的仿真运行输出和总结循序渐进的方式进行。
- 提供了大量的 Proteus 应用电路和 ICCAVR 开发环境的工程文件（读者可以登录 www.hxedu.com.cn 查找本书免费下载），可以直接运行仿真。

本书主要由程国钢编著。同时，参与本书编写工作的还有孙明、唐伟、王杨、顾辉、李成、刘启才、陈杰、郑宏、张霁芬、张计、陈军、张强、杨明、张玉兰。在此，对以上人员致以诚挚的谢意。

由于时间仓促，以及学识水平所限，错误之处在所难免，请广大读者批评指正。

编著者

目　录

第 1 章　ATmega128 应用基础

单片机是单片微型计算机的简称，是一种将运算控制器、存储器、寄存器 I/O 接口，以及一些常用的功能模块都集成到一块芯片上的计算机，常用于工业控制、小型家电等需要嵌入式控制的场合。ATmega128 是 ATMEL 公司研发的增强型内置 Flash 的 RISC 精简指令集高速 8 位单片机（AVR）中性能最高的型号，本章将简要介绍 ATmega128 的体系结构、C 语言和 ICCAVR 软件开发环境。

1.1　ATmega128 的体系结构

ATmega128 是基于 RISC 结构的 8 位高性能、低功耗的处理器，是 AVR 单片机系列中整体性能最强的一款，其主要特点如下所述。

- 支持 131 条 AVR 指令，其中大多数指令执行时间为单个时钟周期，执行速度快。
- 内部有 32 个 8 位通用工作寄存器，硬件乘法器只需两个时钟周期，当工作频率为 16MHz 时性能高达 16MIPS。
- 内置 4KB 的片内 SRAM，128KB 的系统内可编程 Flash，4KB 的 E^2PROM。
- 内置具有独立锁定位的可选 Boot 代码区，并且可以通过片上 Boot 程序实现系统内编程。
- 内置 4 个灵活的具有比较模式和 PWM 功能的定时/计数器 T/C 和一个实时时钟 RTC。
- 内置 8 通道 10 位 ADC，可以组合为 8 个单端通道或者 7 组差分通道，其中有两个具有可编程增益（1X, 10X 或 200X）的差分通道。
- 内置片内模拟比较器。
- 内置具有独立片内振荡器的可编程看门狗定时器。
- 内置多种串行通信接口，包括 TWI（I^2C）两线接口、两路可编程 USART、可工作于主机/从机模式的 SPI 串行接口。
- 提供 53 个可编程的 I/O 端口、64 引脚 TQFP 封装、64 引脚 MLF 封装。
- 支持符合 JTAG 标准的边界扫描，提供和 IEEE 1149.1 标准兼容的 JTAG 硬件接口。
- 支持 2.7～5.5V（ATmega128L）和 4.5～5.5V（ATmega128）工作电压，前者工作频率为 0～8MHz，后者为 0～16MHz。

图 1.1 所示的是 ATmega128 的内部结构示意图，其由 ALU、通用寄存器、程序存储器、数据 RAM、中断模块和内部扩展资源等组成，各个部分详细说明如下所述。

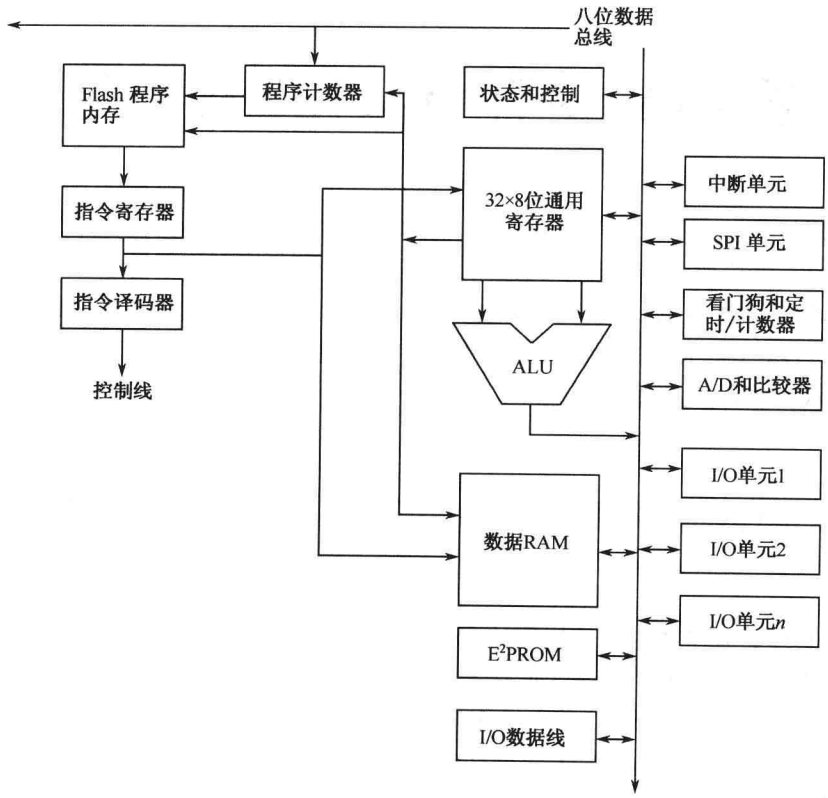

图 1.1　ATmega128 的结构

- ALU（运算器）：支持通用寄存器之间及通用寄存器和常数之间的算术和逻辑运算，ALU 也可以执行单寄存器操作，当运算完成之后更新相应的状态寄存器的内容以反映操作结果。ATmega128 通过有条件或者无条件的跳转指令和调用指令来控制程序的工作流程，从而可以直接寻址整个地址空间。
- 通用寄存器：其中 6 个寄存器可以联合起来构成 3 个 16 位的 X、Y、Z 间接寻址寄存器，可以用来寻址数据空间以实现高效的地址运算，其中一个指针还可以作为程序存储器查询表的地址指针。
- 程序存储器：这是可以在线编程的 Flash，其快速访问寄存器由 32 个 8 位通用寄存器组成，访问时间为一个时钟周期，从而 ATmega128 可以实现单时钟周期的运算器操作，在典型的运算器操作中，两个分别位于不同通用寄存器中的操作数被同时访问，然后执行运算，结果再被送回到通用寄存器，整个指令执行过程仅需一个时钟周期。
- 中断模块：ATmega128 内置一个灵活的中断模块，其控制寄存器位于 I/O 空间内，并且有一个位于状态寄存器 SREG 中的全局中断使能位，每个中断在中断矢量表里都有独立的中断矢量，其优先级与该中断矢量在中断矢量表的位置有关，中断矢量地址越低，其优先级越高。
- 内部扩展资源：ATmega128 内部集成了多种外部资源，包括 SPI 接口、ADC 接口、E^2PROM、串行模块等。

1.1.1　ATmega128 的内核

ATmega128 的内核由算术逻辑单元（ALU）、状态寄存器（SREG）、通用寄存器、堆栈、RAM 页面选择寄存器（RAMPZ）、中断和复位处理模块等构成。

1．算术逻辑单元

算术逻辑单元是 ATmega128 内核中执行各种算术和逻辑运算操作的部件，其基本操作包括加、减、乘、除四则运算（算术运算），与、或、非、异或等逻辑操作（逻辑预算），以及移位、比较和传送等操作。ALU 与 32 个通用工作寄存器直接相连，ATmega128 的寄存器与寄存器之间、寄存器与常数之间的 ALU 运算只需要一个时钟周期。ALU 的操作分为三类：算术、逻辑和位操作。此外 ALU 还能支持无/有符号数和分数乘法。

2．状态寄存器

ATmega128 的状态寄存器包含了最近执行的算术指令的相关结果信息，用户可以根据这些信息来实现条件操作，状态寄存器 SREG 的内部结构如表 1.1 所示，其详细说明如下所述。

表 1.1　ATmega128 的状态寄存器 SREG

BIT	I	T	H	S	V	N	Z	C
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

- I：全局中断使能位，当 I 被置位时使能全局中断，单独的中断使能由其他独立的控制寄存器控制；如果 I 被清零，则不论单独中断标志置位与否，都不会产生中断。任意一个中断发生后 I 将被清零，而执行 RETI（中断服务程序退出）指令后 I 恢复置位以使能中断，I 也可以通过 SEI（置位全局中断使能位）和 CLI（清除全局中断使能位）指令来置位和清零。
- T：位复制存储位，位复制指令 BLD 和 BST 利用 T 作为目的或源地址，BST 指令把寄存器的某一位复制到 T，而 BLD 把 T 复制到寄存器的某一位。
- H：半进位标志位，H 被置位时表示算术操作发生了半进位，此标志对于 BCD 运算非常有用。
- S：负数标志位，用于存放 N 与 2 的补码和溢出标志 V 的异或结果。
- V：补码溢出标志，支持二进制补码运算。
- N：负数标志位，表明算术或逻辑操作结果为负。
- Z：零标志位，表明算术或逻辑操作结果为零。
- C：进位标志位，表明算术或逻辑操作发生了进位。

3．通用寄存器

ATmega128 有 32 个通用寄存器，其针对 ATmega128 的指令集进行了优化，支持以下的输入/输出方案。

- 输入为一个 8 位操作数，输出一个 8 位结果。
- 输入为两个 8 位操作数，输出一个 8 位结果。
- 输入为两个 8 位操作数，输出一个 16 位结果。
- 输入为一个 16 位操作数，输出一个 16 位结果。

图 1.2 所示的是 ATmega128 的通用寄存器结构示意图，每个通用寄存器都有一个对应的地址，将它们直接映射到用户数据空间的头 32 个地址，X、Y、Z 寄存器可以设置为指向任意寄存器的指针。

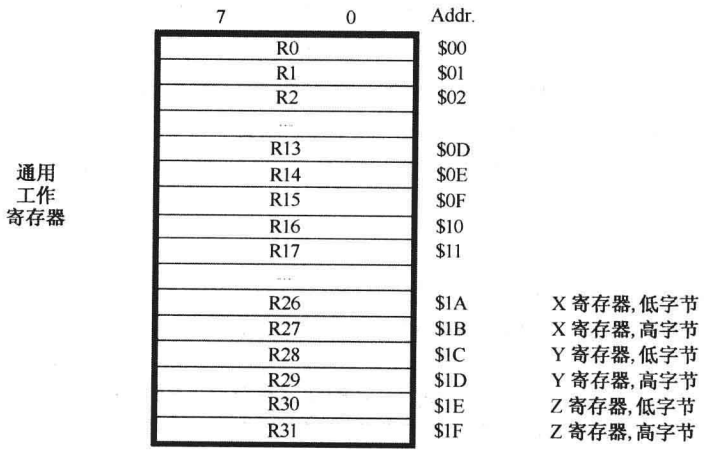

图 1.2 ATmega128 的通用寄存器

通用寄存器 R26～R31 除了用做通用寄存器外，还可以作为数据间接寻址用的地址指针 X、Y、Z，如图 1.3 所示。在不同的寻址模式中，这些地址寄存器可以实现固定偏移量，自动加 1 和自动减 1 功能。

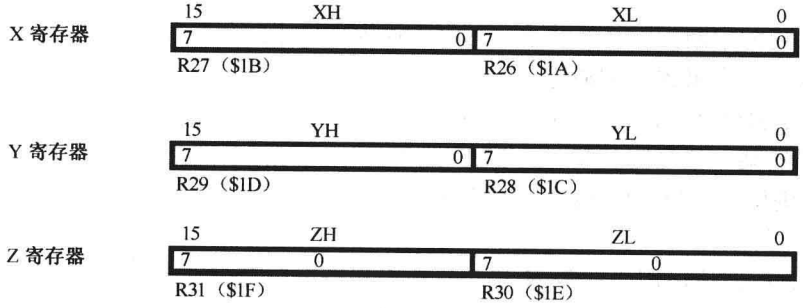

图 1.3 X、Y、Z 寄存器

4．堆栈

ATmega128 的堆栈主要用来保存临时数据、局部变量、子程序和中断子程序的返回地址，堆栈指针总是指向堆栈的顶部。

注意：ATmega128 的堆栈是向下生长的，即当有新数据被推入堆栈时，堆栈指针的数值将减小。

ATmega128 的堆栈指针指向位于 SRAM 的函数或者中断堆栈，在调用子程序和使能中断之前必须先初始化堆栈，并且堆栈指针必须指向高于 0x60 的地址空间，堆栈的详细增减说明如下所述。

- 当使用 PUSH 指令将数据推入堆栈时，堆栈指针减 1。
- 当子程序或中断返回地址被推入堆栈时，指针将减 2。
- 使用 POP 指令将数据弹出堆栈时，堆栈指针加 1。
- 使用 RET 或 RETI 指令从子程序或中断返回时堆栈指针加 2。

ATmega128 的堆栈指针其实质上是 I/O 空间中的两个 8 位寄存器，如表 1.2 所示。

表 1.2　ATmega128 的堆栈指针寄存器

BIT	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0

5．页面选择寄存器

ATmega128 支持不超过 64KB 的数据存储器空间，页面选择寄存器用于选择 ATmega128 访问的是哪一个数据存储器空间，其内部结构如表 1.3 所示，详细说明如下所述。

表 1.3　ATmega128 的页面选择寄存器

BIT	—	—	—	—	—	—	—	RAMPZ0
读/写	R	R	R	R	R	R	R	R/W
初始值	0	0	0	0	0	0	0	0

- BIT7～BIT1：保留位。
- RAMPZ0：RAM 页面扩展选择指针，当 RAMPZ0 = 0 时，ELPM/SPM 指令用于访问 ATmega128 寄存器的低 64KB 地址空间 0x0000～0x7FFF；当 RAMPZ0 = 1 时，ELPM/SPM 指令用于访问 ATmega128 寄存器的高 64KB 地址空间 0x7FFF～0xFFFF。

6．中断和复位处理模块

ATmega128 的中断事件都在程序空间对应独立的中断矢量，所有的中断模块都有自己的使能位，当使能位被置位，且状态寄存器的全局中断使能位 I 也被置位时，可以产生中断事件。根据程序计数器 PC 的不同，在引导锁定位 BLB02 或 BLB12 被编程的情况下，中断可能被自动禁止，这个特性提高了软件的安全性。

ATmega128 的程序存储区的最低地址默认为复位矢量和中断矢量，完整的矢量列表参考下文 1.1.7 节的表 1.19，该表也列出了不同中断的优先级；矢量所在的地址越低，优先级越高。RESET 具有最高的优先级，第二个为 INT0（外部中断请求 0）。通过置位通用中断控制寄存器（GICR）的 IVSEL，中断矢量可以移至引导 Flash 的起始处，编程熔丝位 BOOTRST 也可以将复位矢量移至引导 Flash 的起始处。

在任一中断发生时全局中断使能位 I 被清零，从而禁止了所有其他的中断，用户软件可以在中断程序里置位 I 来实现中断嵌套，此时所有的中断都可以中断当前的中断服务程序在执行 RETI 指令后 I 自动置位。

ATmega128 有两种类型的中断，第一种由中断事件触发并置位中断标志位；对于这些中断，程序计数器跳转到实际的中断矢量以执行中断处理程序，同时硬件将清除相应的中断标志，中断标志位也可以通过对其写“1”的方式来清除，当中断发生后，如果相应的中断使能位为“0”，则中断标志位置位，并一直保持到中断执行，或者被软件清除。类似地，如果全局中断标志被清零，则所有已发生的中断都不会被执行，直到 I 被置位。然后挂起的各个中断按中断优先级依次执行。

第二种类型的中断则是只要中断条件满足，就会一直触发，这些中断不需要中断标志位，若中断条件在中断使能之前就消失了，中断不会被触发。

ATmega128 在退出中断后总是回到主程序后并至少执行一条指令才可以去执行其他被挂起的中断。需要注意的是，进入中断服务程序时状态寄存器不会自动保存，中断返回时也不会自动恢复。这些工作必须由用户通过软件来完成。当使用 CLI 指令来禁止中断时，中断禁止立即生效，没有中断可以在执行 CLI 指令后发生，即使它是在执行 CLI 指令的同时发生的。

ATmega128 的中断响应操作最少为 4 个时钟周期，在 4 个时钟周期后，程序跳转到实际的中断处理例程。在这 4 个时钟周期期间 PC 将自动入栈，在通常情况下，中断矢量为一个跳转指令，此跳转需要 3 个时钟周期，如果中断在一个多时钟周期指令执行期间发生，则在此多周期指令执行完毕后 ATmega128 才会执行中断程序。若中断发生时 ATmega128 处于休眠模式，中断响应时间还需增加 4 个时钟周期。此外还要考虑到不同的休眠模式所需要的启动时间。

ATmega128 的中断返回操作也需要 4 个时钟周期，在此期间 PC（两个字节）将被弹出栈，堆栈指针加 2，状态寄存器 SREG 的 I 位被置位。

> **注意：**当使用 SEI 指令使能中断时，在执行任何中断之前一定会首先执行完 SEI 指令之后的那一条指令。

1.1.2 ATmega128 的存储器体系

ATmega128 的存储器体系由程序存储器（Flash）、数据存储器（SRAM），以及 E^2PROM 存储器组成，这 3 个存储器空间都是线性的。

1．程序存储器

ATmega128 具有 128KB 内部 Flash 用于存放程序指令代码，支持在线编程（ISP）和在应用编程（IAP）。因为 ATmega128 的所有的指令为 16 位或 32 位，所以程序存储器被组织成 64KB×16 位的形式，并且被分为引导程序区（BOOT）和应用程序区两个不同的区，如图 1.4 所示。