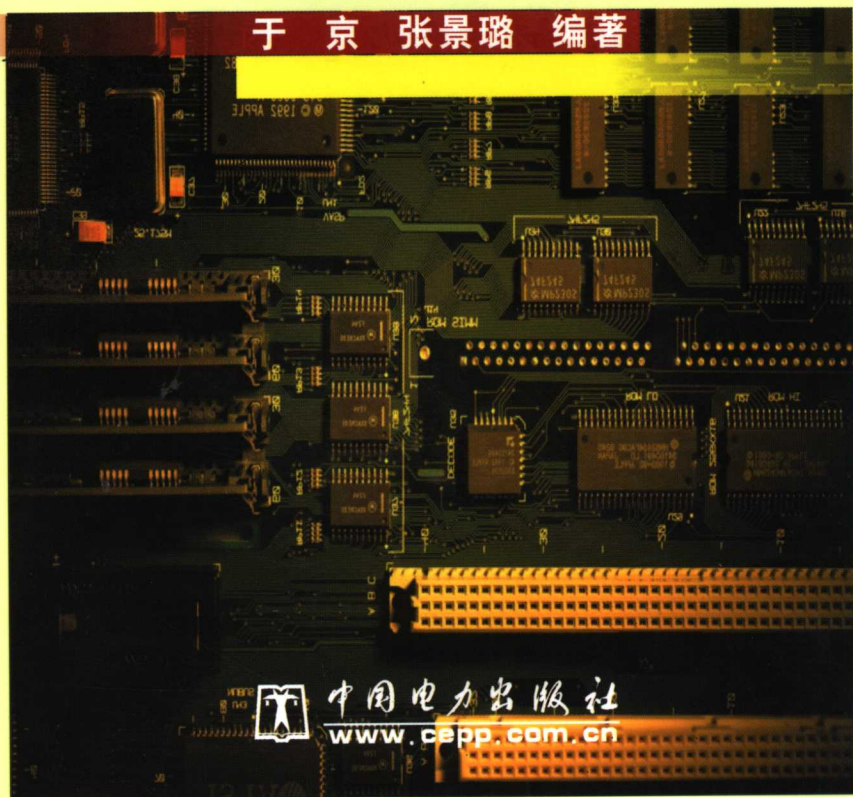


51 XILIE DANPIANJI

51系列单片机 C程序设计与应用案例

于京 张景璐 编著



中国电力出版社

www.cepp.com.cn

51 XILIE DANPIANJI

51系列单片机 C程序设计与应用案例

于 京 张景璐 编著



中国电力出版社

www.cepp.com.cn

内容提要

本书作者在长期从事单片机实际开发和教学工作的基础上,通过精心设计的一系列环节,使内容自成体系,循序渐进。目的是让读者可以从零起步。本书避开冗长的理论描述,通过完整的案例分析,引导读者快速进入设计开发领域。同时作者也愿意将自己实际的开发设计经验与大家分享,以期使读者看完本书就能够设计、开发出符合当今趋势的单片机应用系统。

本书在简要介绍单片机基础的同时,重点增加了数字电路的知识,使读者不用再去查找其他参考书就能够进行系统的开发。全书重点介绍了51系列单片机的结构特点和编程方法,并对每一个问题都附有真实、完整的应用案例。

本书可供单片机设计、开发人员和大中专院校电子专业的师生参考、学习使用

图书在版编目(CIP)数据

51 系列单片机 C 程序设计与应用案例/于京, 张景璐编著.
—北京: 中国电力出版社, 2006
ISBN 7-5083-4085-X

I. 5… II. ①于…②张… III. 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 (2006) 第 007177 号

中国电力出版社出版、发行

(北京三里河路 6 号 100044 <http://www.cepp.com.cn>)

北京市同江印刷厂印刷

各地新华书店经售

*

2006 年 3 月第一版 2006 年 3 月北京第一次印刷

850 毫米×1168 毫米 32 开本 7.5 印张 194 千字

印数 0001—4000 册 定价 15.00 元

版权专有 翻印必究

(本书如有印装质量问题, 我社发行部负责退换)

前 言



首先说明这本书不属于单片机的理论书籍，它可以说是一本实例分析，或是一册单片机开发实践总结。但是您会发现从第一个例子开始入手学习单片机是非常简单的，而学习完最后一个例子，您甚至可以完成一个中等规模的单片机应用了。再次强调：整个学习过程并不难。因为这些例子的安排是精心准备的，是直观的、可理解的，而不是只告诉你“将 0xA0 送入地址 88H，这样就实现了发送数据的功能”。作者认为这种例子不够完整，对读者学习单片机应用没有意义。

本书还有一个特点，那就是它用一些章节讲了“数字电路”和 C 语言，这是因为作者发现外版教材和国内教材的最大区别就是：外版教材对读者的起始要求低，常用一些篇幅介绍本书需要的基础知识，这样读者比较容易上手，学起来也有兴趣。所以本书中加上数字电路和 C 语言内容的目的在于：如果你需要马上开发一个项目，而你几乎已经忘记了学校所学的基础知识，那么这一本书就足够了，不用再去找其他参考书。

不可否认，如果你对基础知识掌握得越好，做出的单片机系统效率就越高，这一点在单片机系统开发中是很重要的。也就是说，将系统做得出来是一回事，系统运行的效率高就是另一回事了。比如下面这个例子：你看得出来它们的区别吗？

定义数组 `a[10]`，其中存放 10 个数值，找出需要的数值的下标，并存入 `inx`：

```
for(i=0;i<10;i++)  
if(a[i]==sample)inx=i;
```

另一种做法：定义 `a` 有 11 个元素，用 `a[1]` 到 `a[10]` 存放样本：

```
for(a[0]=sample,inx=10;a[inx]!=sample;inx--);
```

第二种做法的效率比第一种提高 40% 以上，对于单片机这种资源相对不足的系统来说是很重要的。本书的例程中有一些注释，告诉读者用代码提高系统运行效率的技巧，这是本书另一个可取之处。

还有，本书使用 C 语言来开发单片机系统，原因在于：对开发来说，C 语言开发效率高、易懂、易组织、复用性强。并且经过不断完善，C 语言写的单片机程序其运行效率已经接近汇编程序了。书中还给出了 I²C 总线和 SPI 总线的全部驱动程序，使读者能够马上应用到自己的设计开发中。

说了这么多就是想告诉读者，您能很容易地使用本书构建你自己的单片机系统。当然，由于时间紧迫，书中难免有疏漏的地方，欢迎读者批评指正。

作者

目 录



前言

第一章 数字逻辑基础	1
第一节 数制与布尔代数	1
一、数制	1
二、十六进制和八进制	4
三、十进制数的二进制编码 (Binary-Coded-Decimal)	5
第二节 布尔代数	7
一、布尔代数	7
二、布尔代数的基本定理及法则	8
三、布尔代数的表示方法	9
第三节 布尔代数的化简	10
一、布尔代数化简的重要定理	10
二、布尔代数的化简	11
第四节 组合逻辑电路	22
一、组合逻辑表示	23
二、组合逻辑	25
三、常用的组合逻辑电路	27
小结	33
第二章 时序逻辑基础	35
第一节 时序逻辑电路的特点	35
一、时序逻辑电路与组合逻辑电路的区别	35
二、组合逻辑电路与时序逻辑电路范畴	36

第二节 存储单元	37
一、锁存器	38
二、透明锁存器	39
第三节 触发器 (Flip - Flop)	40
一、基本 RS 触发器	41
二、D 触发器	44
三、T 触发器	47
第四节 总线与三态器件	48
第五节 存储器	51
第六节 时序图	53
第七节 串行与并行	57
小结	59
第三章 单片机的 C 语言基础	60
第一节 概论	60
一、使用 C 语言对单片机编程的特点	61
二、MCS - 51 的 C 语言编译器	61
三、Cx51 的程序结构	62
第二节 Cx51 的数据与运算	63
一、数据与数据类型	63
二、常量与变量	66
三、单片机 MCS - 51 的存储器结构和存储模式	67
四、位变量 (BIT) 及其 Cx51 定义	70
五、Cx51 运算符、表达式及其规则	73
第三节 单片机的 C 语言程序设计	80
一、顺序结构的程序设计	81
二、选择结构的程序设计	81
三、循环结构的程序设计	84
第四节 Cx51 构造数据类型	87

一、数组	87
二、结构	89
三、指针	91
小结	93
第四章 单片机开发初步	94
第一节 MCS-51 单片机的基本结构	94
一、MCS-51 单片机的结构	94
二、MCS-51 单片机的时钟和时序	97
三、MCS-51 的复位电路和复位状态	98
四、最简单的单片机电路	101
第二节 案例开发：按钮计次	105
一、HD7279 的使用方法	105
二、HD7279 的案例剖析	108
第五章 MCS-51 的定时、计数与中断	115
第一节 计数和定时的原理	115
第二节 MCS-51 单片机的定时/计数器	116
一、基本结构	116
二、控制字	117
三、工作方式详解	118
第三节 MCS-51 单片机的中断机制	120
一、单片机的中断原理	120
二、中断应用案例一：利用外部中断完成对 PI 口 的检测	124
三、中断应用案例二：定时器程序的设置	126
第六章 MCS-51 的串口应用	129
第一节 MCS-51 的串口工作原理	129

一、通信的基本概念	129
二、MCS-51 串口的状态寄存器和缓冲器	130
三、MCS-51 串口的工作方式	132
四、波特率的设计	134
第二节 一个串口应用的实例	135
一、RS232 接口	135
二、实例剖析	136
第七章 接口技术	141
第一节 概述	141
第二节 I ² C 总线	141
一、简介	141
二、I ² C 总线的驱动模拟	146
第三节 SPI 总线	150
第四节 实用案例: 电平控制	152
一、X9221 的功能介绍	153
二、X9221 的时序分析和指令系统	154
三、本例的功能和系统设计	156
四、代码分析	157
第五节 实用案例: A/D 转换	158
一、A/D 转换的基本知识	158
二、TLC542 的功能介绍	159
三、TLC542 的时序分析和指令系统 (如图 7-10 所示)	160
四、本例的功能和系统设计	161
第六节 实用案例: 存储芯片 AT24 系列的使用	164
一、AT24 系列 E ² PROM 的功能介绍	164
二、AT24 系列 E ² PROM 的时序分析和指令系统	166

三、AT24 系列的读写	167
第七节 实用案例: X5045 的使用	169
一、X5045 的功能介绍	169
二、X5045 的使用方法	169
三、本例的功能和系统设计	173
第八章 系统扩展	177
第一节 基本概念	177
第二节 ROM 扩展	178
第三节 RAM 扩展	179
第四节 地址译码法	180
第五节 常用的 MCS-51 内核单片机	180
一、PIC 系列	182
二、ATMEL 的 89S5X 系列	183
三、华邦系列	185
第九章 实际单片机开发完整过程	187
第一节 万年历的开发	187
一、元器件的介绍	187
二、AT89S52 的 ISP 功能使用	188
三、PCF8563 的功能介绍	190
四、应用实例	191
第二节 基于 A/D 转换的数据采集系统	202
第三节 华邦 W77E58 的应用	206
第十章 KEIL 的使用	213
第一节 如何在 KEIL 中调试程序	213
第二节 KEIL 中的指针与数据类型	218
一、存储类型与存储区关系	218

二、指针类型和存储区的关系	220
附图 A “按钮计次”系统的原理图	223
附图 B “利用外部中断完成对 P1 口的检测”系统的原理图	224
附图 C “定时器程序设置”系统的原理图	225
附图 D “串口应用”系统的原理图	226
附图 E “电平控制”系统的原理图	227
附图 F “万年历系统”的原理图	228

第一章 数字逻辑基础

本章主要介绍数字系统中各种数制的表示及编码，还有数字逻辑基础知识，包括布尔代数的基本概念、基本公式以及布尔代数的表示方法和化简。本章还介绍数字电路中各种基本门电路和常用组合逻辑电路的工作原理以及使用，如多路复用器、编码器与译码器等。

第一节 数制与布尔代数

一、数制

众所周知，在日常生活中我们经常碰到计数的问题，比如在交易的时候。由于人们经常要借助数字系统帮助人们处理日常信息，因此数字系统中同样会经常涉及到计数的问题。

那么，我们大家最熟悉的数制是哪种数制呢？没错，是十进制。十进制是人类日常生活中不可分割的一部分。但是数字系统与日常生活中的进制不同，有的只是 0 和 1 两个符号，因而在数字系统中最常用的是二进制，以及与之关系密切的八进制和十六进制。

（一）十进制

所谓十进制就是以 10 为基数的计数体制，其计数规律是逢十进一。十进制可以用字母 D 表示。

例 1-1 试用位权来表示十进制数 1234。

解：

$$\begin{array}{rcll} 1 & 2 & 3 & 4 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ & & 4 \times 10^0 = & 4 \\ & & 3 \times 10^1 = 3 \times 10 = & 30 \\ & & 2 \times 10^2 = 2 \times 100 = & 200 \\ & & 1 \times 10^3 = 1 \times 1000 = & +1000 \\ & & & \hline & & & 1234 \end{array}$$

(二) 二进制

二进制与十进制的区别在于数码的个数和进位规律有很大的区别。顾名思义，二进制的计数规律为逢二进一，是以 2 为基数的计数体制，即：1 + 1 = 10。

二进制的位权表示，如图 1-1 所示。

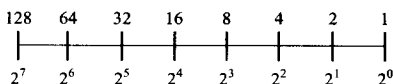


图 1-1 二进制数的位权图

例如：我们可以将二进制数 1010 表示为： $1010 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

一般任意一个数 N 用二进制数可表示为：

$$(N)_B = \sum_{j=-\infty}^{j=\infty} K_j \times 2^j$$

其中字母 B 表示二进制数。

例 1-2 试将二进制数 $(11010110)_B$ 转换为十进制数。

解：方法是按各位权展开，求出各项式之和。

$$\begin{array}{cccccccc} 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ \uparrow & & & & & & \uparrow & \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{array} = 128 + 64 + 16 + 4 + 2 = 214$$

$$(11010110)_B = 214_D$$

(三) 十-二进制之间的转换

既然一个数可以用二进制和十进制两种不同形式来表示，十-二进制之间经常需要转换，十进制转换为二进制的方法可以用除 2 求余的方法。

例 1-3 将 $(36)_D$ 转换为二进制数。

解：方法是，不断地用 2 分解十进制整数，并将余数按得到的顺序由低位到高位排列，即可得到对应的二进制数。

2	36	余 0	b_0
2	18	余 0	b_1
2	9	余 1	b_2
2	4	余 0	b_3
2	2	余 0	b_4
2	1	余 1	b_5
	0		

所以 $(36)_D = (b_5 b_4 b_3 b_2 b_1 b_0)_B = (100100)_B$

当一个要转换的数不太大时，可以采用另一种比较简单的方法，这种方法的思想是从需要转换的十进制数找出与之最接近的2的幂次方，并从这个十进制数中减去该2的幂次方，在剩下的余数中重复这种做法，直到余数为0或1。然后将所得到的这些2的幂次方与二进制数中的位权相比，相同的位标记为1，其余的为0，这样就得到与十进制数对应的二进制数。

例 1-4 现在来看看 175 这个十进制数

解：

175		1	0	1	0	1	1	1	1
- 128	$\rightarrow 2^7$								
47									
- 32	$\rightarrow 2^5$								
15									
- 8	$\rightarrow 2^3$								
7									
- 4	$\rightarrow 2^2$								
3									
- 2	$\rightarrow 2^1$								
1	$\rightarrow 2^0$								

$(175)_D = (10101111)_B$

数据通过减去相应的权，如果够减，则相应的位权为1，如175先减去128，如果够减，对应第7位为1，175先减去128的余数47，继续减去64，不够减，对应第6位为0，以此类推，直到结果为0或1。

(四) 在数字设备中使用二进制的优缺点

1. 使用二进制的优点：数字装置简单可靠，所用元器件少；

只有两个数码 0 和 1，因此它的每一位数都可用任何具有两个不同稳定状态的元器件来表示；基本运算规则简单，运算操作方便。

2. 使用二进制的缺点：用二进制表示一个比较大的数时，位数多。

二、十六进制和八进制

由于当二进制数的位数很多时，阅读和书写都很麻烦，因此为了减少书写一个数字的位数，在计算机中经常采用十六进制或八进制数来表示二进制数。十六进制中包含的 16 个数字是 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F，其中 A ~ F 分别对应于十进制中的 10 ~ 15。参照十进制数和二进制数的一般表达式，可以得到十六进制数的一般表达式为：

$$(N)_H = \sum_{j=-\infty}^{j=\infty} K_j \times 16^j$$

其中，字母 H 表示十六进制数。

例 1-5 将十六进制数 4E6 转换为十进制数。

解：方法是按十六进制数位权展开，十六进制数位权分别是 $16^0 = 1$ ， $16^1 = 16$ ， $16^2 = 256$ 等，求出各项式之和。

$$\begin{aligned}(4E6)_H &= 4 \times 16^2 + 14 \times 16^1 + 6 \times 16^0 \\ &= 4 \times 256 + 14 \times 16 + 6 \times 1 = (1254)_D\end{aligned}$$

例 1-6 将二进制数 1001 1100 1011 0100 1000.011 转换为十六进制数，将二进制数 0101 1011.001 转换为十进制数。

解：方法是以小数点为界，整数、小数部分两边分别以四位为一组，不足四位的整数部分高位补 0、小数部分低位补 0，每四位二进制数可以转换为一位十六进制数。

$$(1001\ 1100\ 1011\ 0100\ 1000.011)_B = (9CB48.6)_H$$

$$(0101\ 1011.001)_B = (5B.2)_H$$

$$\begin{aligned}&= (1 \times 2^2 + 1 \times 2^0) \times 16^1 + (1 \times 2^3 + 1 \times 2^1 + 1 \\ &\quad \times 2^0) \times 16^0 + (0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1) \times 16^1\end{aligned}$$

$$= [5 \times 16^1 + 11 \times 16^0 + 2 \times 16^1]_D = 91.125_D$$

同样地，可以得到任意八进制数的表达式为：

$$(N)_O = \sum_{j=-\infty}^{j=\infty} K_j \times 8^j$$

其中，字母 O 表示八进制数。

对于二进制数转换为八进制数时，可将三位二进制数分成一组，对应于一位八进制数。二进制数分别是 000，001，010，011，100，101，110，111，对应八进制数字 0 ~ 7。

例 1-7 将 $(1001\ 1100\ 1011\ 0100\ 1000)_B$ 转换为八进制数。

$$\begin{aligned} \text{解：} & (1001\ 1100\ 1011\ 0100\ 1000)_B \\ &= (10\ 011\ 100\ 101\ 101\ 001\ 000)_B \\ &= (2\ 3\ 4\ 5\ 5\ 1\ 0)_O \end{aligned}$$

三、十进制数的二进制编码 (Binary-Coded-Decimal)

8421BCD 编码 (二 - 十进制码或称 BCD 码)

将十进制数中的 0 ~ 9 十个数，用二进制来进行编码，这种编码称为二 - 十进制码或称 BCD 码。由于使用 4 位二进制数 $b_3b_2b_1b_0$ 表示一位十进制数，可以有 16 种组合，从 0000 到 1111，从中取出十种组合。这其中组合的方法较多，比较常用的是 8421BCD 码。

8421BCD 码用四位自然二进制码中的前十个码字来表示十进制数码，因此，各位的权值依次为： b_3 位的权为 $2^3 = 8$ ， b_2 位的权为 $2^2 = 4$ ， b_1 位的权为 $2^1 = 2$ ， b_0 位的权为 $2^0 = 1$ ，故称 8421 BCD 码。

表 1-1 是常用的二 - 十进制编码，其中包括十进制码与 8421 BCD 码的对应关系。

$$(8)_{10} = (1000)_2 = (1000)_{8421BCD}$$

$$(5)_{10} = (101)_2 = (0101)_{8421BCD}$$

其中，1010 ~ 1111 是无效的 8421BCD 码 8421BCD 码由于

表示简单、直观，所以在数字系统中应用较多。

表 1-1 常用的二-十进制编码

十进制数	8421 BCD 码	2421 BCD 码	余 3 码
0	0000	0000	0011
1	0001	0001	0100
2	0010	0010	0101
3	0011	0011	0110
4	0100	0100	0111
5	0101	1011	1000
6	0110	1100	1001
7	0111	1101	1010
8	1000	1110	1011
9	1001	1111	1100

例 1-8 8421BCD 码与十进制数的转换。

解：8421BCD 码由于表示简单，转换时直接按四位二进制对应一位十进制进行。

$$(1000\ 0001)_{\text{BCD}} = (81)_{10}$$

$$(1001\ 0110)_{\text{BCD}} = (96)_{10}$$

表 1-1 中除了 8421BCD 码外，还有其他的二-十进制码，如 2421 码和余 3 码。

余 3 码也是一种常见的二-十进制码，它是一种无权编码；对比表 1-1，同样的十进制数，余 3 码比 8421BCD 码多 3，即 0011。

余 3 码是一种对 9 的自补码，例如 3 的余 3 码是 0110 求补的结果是 1001，查表是数字 6 的余 3 码，而数字 6 和 3 刚好是对数字 9 的补， $6+3=9$ 。所以，要求 5 的余 3 码，可以先求出 5 对数字 9 的补， $9-5=4$ ，再计算数字 4 的余 3 码是 0111，则 5 的余 3 码求补是 1000。