

刘文涛 编著

单片机语言

C51典型应用设计



本书附光盘



人民邮电出版社
POSTS & TELECOM PRESS

单片机语言 C51 典型应用设计

刘文涛 编 著

人 民 邮 电 出 版 社

图书在版编目 (CIP) 数据

单片机语言 C51 典型应用设计/刘文涛编著. —北京: 人民邮电出版社, 2005.10

ISBN 7-115-13820-6

I. 单... II. 刘... III. 单片微型计算机—程序设计 IV. TP368.1

中国版本图书馆 CIP 数据核字 (2005) 第 102458 号

内 容 提 要

本书是《单片机语言 C51 程序设计》的配套书。它通过典型的应用程序介绍了 8051 单片机语言 C51 的软件编程技术。全书共有 100 多个典型应用实例程序, 每个程序都给出了 C51 高级语言程序清单并做了详细注释, 而且这些程序都有实际应用参考价值。通过本书的学习和实践, 读者能够掌握 C51 语言的编程应用技巧和方法, 同时, 本书的所有程序都是通过编译的, 并且放入随书附赠的光盘中。

本书可供从事单片机开发、计算机应用开发的科技人员学习和参考。

单片机语言 C51 典型应用设计

- ◆ 编 著 刘文涛
责任编辑 张 伟
- ◆ 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 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京鸿佳印刷厂印刷
新华书店总店北京发行所经销
- ◆ 开本: 787×1092 1/16
印张: 21.5
字数: 538 千字
印数: 1—5 000 册
- 2005 年 10 月第 1 版
2005 年 10 月北京第 1 次印刷

ISBN 7-115-13820-6/TN · 2577

定价: 35.00 元 (附光盘)

读者服务热线: (010) 67129264 印装质量热线: (010) 67129223

前 言

Keil C51 标准 C 编译器为 8051 微控制器的软件开发提供了 C 语言环境, 同时保留了汇编代码高效、快速的特点。C51 编译器的功能不断增强, 使开发者可以更加贴近 CPU 本身及其衍生产品。C51 已被完全集成到 μ Vision3 的集成开发环境中, 这个集成开发环境包含编译器、汇编器、实时操作系统、项目管理器和调试器。 μ Vision3 IDE 可为它们提供单一而灵活的开发环境。

Keil C51 是目前最高效、灵活的 8051 开发平台。它可以支持所有 8051 的衍生产品, 也可以支持所有兼容的仿真器, 同时支持其他第三方开发工具。因此, 它无疑是 8051 开发用户的最佳选择。

本书是《单片机语言 C51 程序设计》的配套书, 它通过大量的实例详细介绍了利用 Keil C51 进行 51 系列单片机开发的方法和技巧。主要内容包括: 第 1 章应用基础实例, 第 2 章 C51 应用技巧, 第 3 章中断应用程序, 第 4 章定时器与时钟应用程序, 第 5 章串行口应用程序, 第 6 章接口以及其他应用程序, 第 7 章 A/D 转换应用程序, 第 8 章 LCD 液晶显示应用程序, 第 9 章键盘输入法应用程序, 第 10 章 I²C 总线应用程序, 第 11 章综合应用程序。

本书强调先进性和实用性。全书各章列举了大量应用程序实例, 并全部收入到配套光盘中。光盘中的源程序都已经调试并通过, 可以直接应用到开发项目中去。

本书由刘文涛编著。参加编写工作的还有沙盼盼、赵木青、宋萌、张娜、王波波、兰吉昌、姜艳波、孙宁、顾正大、艾丽香、刘春红、辛征、李志、王晶, 作者在此谨表谢意!

由于时间仓促以及水平所限, 书中难免有不妥之处, 恳请广大读者给予指正。

编 者

目 录

第 1 章 应用基础实例	1
例 1 8051 bit I/O 程序	1
例 2 8051 计数器 0 程序	1
例 3 8051 中断 0 程序	2
例 4 8051 I/O 口程序	2
例 5 8051 定时器 0 模式 1 程序	2
例 6 8051 定时器 0 脉冲程序	3
例 7 8051 定时器模式 2 程序	4
例 8 8051 内存读写程序	4
例 9 8051 定时器 2 程序	5
例 10 8051 看门狗程序	6
例 11 ADUC812 的 A/D 转换	7
例 12 ADUC814 的 A/D 转换	7
例 13 ADUC816 的 A/D 转换	8
例 14 ADUC831 的 A/D 转换	9
例 15 Apnt 的 A/D 转换	9
例 16 AT898252	13
例 17 c505 的 A/D 转换	15
例 18 c509 的 A/D 转换	16
例 19 c515 的 A/D 转换	17
例 20 80c517 定时器 2	17
例 21 80c537 A/D 转换	18
例 22 5111 A/D 转换	18
例 23 ADUC812 的 EEPROM 程序	20
例 24 ADUC816 A/D 转换	20
例 25 C8051 F310 实例程序	21
例 26 T89c51cc01 实例程序	22
例 27 SYS420 实例程序	23
例 28 rx2 实例程序	23
例 29 rtx51ints 实例程序	24
例 30 P87LPC769 实例程序	25
例 31 87LPC767 实例程序	27
例 32 MSC1212YS 实例程序	27
例 33 MSC1212yx 实例程序	29

例 34	MSC1211yx 实例程序	30
例 35	MSC1210yx 实例程序	32
例 36	MSCB817 实例程序	33
例 37	MAX765x 实例程序	37
例 38	LPC-Time 实例程序	38
例 39	CC01_ADC 实例程序	40
例 40	DS320_WDT 实例程序	41
例 41	DS80C40 实例程序	42
例 42	DS400 实例程序	43
例 43	F020 实例程序	43
例 44	LPC900 实例程序	44
例 45	LPC935 实例程序	45
例 46	LPC_ADC 实例程序	46
第 2 章 C51 应用技巧		48
例 1	浮点运算程序	48
例 2	复杂计算程序	48
例 3	温度对比程序	49
例 4	for 运算程序	49
例 5	比较	49
例 6	比较赋值	50
例 7	嵌套 for 语句	50
例 8	开关语句	51
例 9	数组 1	51
例 10	结构	52
例 11	联合	52
例 12	共用体	53
例 13	指针	53
例 14	子函数调用	54
例 15	多层循环	54
例 16	二维数组	55
例 17	指针变量	55
例 18	指针类型	55
例 19	指针数组	56
例 20	指针数组 2	56
例 21	指针数组 3	57
例 22	字符指针	57
例 23	指向字符串的指针数组	57
例 24	指向指针的指针	58
例 25	指针变量函数	59

例 26	交换指针	60
例 27	二维指针数组函数调用	60
例 28	函数调用指针数组	61
例 29	函数指针变量	61
例 30	结构指针	62
例 31	动态分配	63
例 32	嵌套的结构体	63
例 33	为指针分配存储空间	64
例 34	表	65
例 35	比较程序	67
例 36	查找程序	68
例 37	冒泡程序	69
第 3 章	中断应用程序	71
例 1	利用外部中断实现 LED 灯的简单控制	71
例 2	定时器中断	73
例 3	串行口中断	74
第 4 章	定时器与时钟应用程序	76
例 1	4 个数字 LED 显示时间程序	76
例 2	秒程序	81
例 3	24 小时时钟程序	90
例 4	DS12887 应用程序	99
第 5 章	串行口应用程序	101
例 1	串行口双工应用程序	101
例 2	采用中断方式进行串口通信	107
例 3	与 PC 机串口通信程序	114
例 4	采用查询方式进行串口通信	118
第 6 章	接口以及其他应用程序	121
例 1	LED 指示灯	121
例 2	DS1820 单芯片温度测量	123
例 3	实时时钟芯片 DS1302	125
例 4	常用的几种码制转换	128
例 5	菜单	131
第 7 章	A/D 转换应用程序	134
例 1	ADC0809 转换实例	134
例 2	AT2051 的 A/D 转换程序	140

第 8 章 LCD 液晶显示应用程序	144
例 1 16×2 字符液晶屏驱动演示程序	144
例 2 128×64 点阵液晶显示驱动程序	147
例 3 192×64 点阵液晶显示驱动程序	151
例 4 240×128 点阵液晶显示驱动程序	171
例 5 320×240 点阵液晶显示驱动程序	184
第 9 章 键盘输入法应用程序	194
例 1 键盘扫描程序	194
例 2 键盘编码程序	197
例 3 拼音输入法	200
例 4 键盘拼音输入法	219
第 10 章 I²C 总线应用程序	239
例 1 I ² C 总线驱动程序	239
例 2 在 PC 上用并行口模拟 I ² C 总线的 C 程序	244
例 3 与 PCF8563 接口	246
第 11 章 综合应用程序	249
例 1 飞机游戏	249
例 2 蛇游戏代码	264
例 3 多级菜单程序代码	280

第 1 章 应用基础实例

本章将介绍与 51 系列硬件相关的 C51 编程实例，这些实例是进行 51 系列单片机开发的基础。通过了解和掌握这些小程序的设计，可以熟悉 51 系列单片机的 I/O、定时器/计数器、中断、串口、A/D 转换以及 EEPROM 读写等基本操作，为进一步进行大型的 51 单片机应用项目的设计打下坚实的基础。

例 1 8051 bit I/O 程序

```
sfr P1 = 0x90;           //确定 P1 (8 bits), P1.0, P1.1
sbit P1_0 = P1^0;        // P1 的 SFR
sbit P1_1 = P1^1;        // P1.0 的 SFR
                          // P1.1 的 SFR
                          //主函数功能

void main (void)
{
    P1_0 = 1;             //配置 P1.0
    while (1)
    {
        P1_1 = P1_0;      // P1.0 赋值给 P1.1
    }
}
```

例 2 8051 计数器 0 程序

```
#include <reg52.h>
#include <stdio.h>

                          //主函数功能

void main (void)
{
    printf ("\r\nCounter 0 Example Program\r\n");
    printf ("\r\nStart toggling P3.4 (T0 input)\r\n\r\n");
    //设置定时器 0, 16-bit 计数器模式
    //定时器 0 运行控制位

    TMOD = (TMOD & 0xF0) | 0x05;
    TR0 = 1;
    //开 P3.4

    while (1)
    {
```

```

    printf ("\rCounter 0 = %2.2bX%2.2bXh", TH0, TL0);
}
}

```

例 3 8051 中断 0 程序

```

#include <REG52.H>
unsigned char ex0_isr_counter = 0;
void ex0_isr (void) interrupt 0
{
    ex0_isr_counter++;           //增加计数
}

void main (void)
{
    IT0 = 1;                     //配置 INTO
                                //配置中断 0

    EX0 = 1;                     //允许 EX0 中断
    EA = 1;                      //允许中断标志
                                //等待

    while (1)
    {
    }
}

```

例 4 8051 I/O 口程序

```

sfr P1 = 0x90;                  //定义端口 1
sfr P3 = 0xB0;                  //定义端口 3
                                //主函数功能

void main (void)
{
    unsigned char pval;          //临时变量端口值

    P1 = 0xFF;                   //端口 P1 输出

    while (1)
    {
        pval = P1;               //P1 口赋值给 pval
        P3 = pval;               //写 pval 赋值给 P3
    }
}

```

例 5 8051 定时器 0 模式 1 程序

```

#include <REG51.H>

```

```

static unsigned long overflow_count = 0;
void timer0_ISR (void) interrupt 1
{
    overflow_count++;           //增加溢出数
}

//主函数功能

void main (void)
{
    //置定时器 0, 16 位定时模式, 定时计
    //数到 65535, 溢出生成中断

    TMOD = (TMOD & 0xF0) | 0x01;

    //置定时器/计数器 0 模式
    ET0 = 1;                   //允许定时器 0 中断
    TR0 = 1;                   //定时器 0 开始运行
    EA = 1;                    //允许中断

    while (1)
    {
    }
}

```

例 6 8051 定时器 0 脉冲程序

```

#include <reg52.h>
#include <stdio.h>

//定时器 0 溢出中断

unsigned int T0_ISR_count = 0;

void T0_ISR (void) interrupt 1
{
    T0_ISR_count++;
    TF0 = 0;                   //复位中断请求
}

//主函数功能

void main (void)
{
    ET0 = 1;
    EA = 1;

    //置定时器 0, 16 位间隔定时模式

    TMOD = (TMOD & 0xF0) | 0x09;
    while (1)
    {
        //清除定时器溢出记录、定时器高低位数,
        //然后开始定时

        T0_ISR_count = 0;
        TH0 = 0;
    }
}

```

```

TL0 = 0;
TR0 = 1;

printf ("\nStart a pulse.\n");

//写出脉冲开始到结束
while (!INT0);
while (INT0);

//计算脉冲宽度, 时钟循环是 1μs
printf ("The width pulse is: %ld uSec\n",
        (unsigned long)((TH0 << 8) | TL0 | ((unsigned long)T0_ISR_count <<
        16)));
}
}

```

例 7 8051 定时器模式 2 程序

```

#include <reg52.h>
#include <stdio.h>

//定时器 1 中断服务程序
//置断点在溢出计数和运行程序在调试器,
//执行的 100 个时钟循环, 溢出计数, 实
//际是每秒 1/10, 000

static unsigned long overflow_count = 0;

void timer1_ISR (void) interrupt 3
{
    overflow_count++; //增加溢出数
}

//主函数功能

void main (void)
{
    //置定时器 1, 8 位定时器初装值, 定时器数是
    //255, 溢出, 初装值 156, 生成中断

    TMOD = (TMOD & 0x0F) | 0x20;
    TH1 = 256 - 100; //初装值 TL1 到 100 时钟
    TL1 = TH1;
    ET1 = 1; //允许定时器 1 中断
    TR1 = 1; //定时器 1 开始运行
    EA = 1; //允许中断
    while (1)
    {
    }
}

```

例 8 8051 内存读写程序

内存映射为

```

CODE: 0x0000-0xDFFF (ROM)
CODE: 0xE000-0xFFFF ('von Neumann' RAM)
XDATA: 0x0000-0xDFFF (RAM)
XDATA: 0xE000-0xFFFF ('von Neumann' RAM)

#include <intrins.h>
#include <absacc.h>

//载入代码到 0xE000

void load_vonNeumann_memory (void)
{
    XBYTE[0xE000] = 0x02;           //LJMP 0xE000
    XBYTE[0xE001] = 0xE0;
    XBYTE[0xE002] = 0x00;
}

//主函数功能

void main (void)
{
    load_vonNeumann_memory ();

    ((void (code *) (void)) 0xE000) ();
    while (1)                       // 循环
    {
    }
}

```

例 9 8051 定时器 2 程序

```

#include <reg52.h>
static unsigned long overflow_count = 0;

void timer1_ISR (void) interrupt 5
{
    TF2 = 0;           //清除中断请求
    overflow_count++;   //增加溢出数
}

//主函数功能

void main (void)
{
    //置定时器 2~16 位自动初装值
    //定时器数 0xFFFF, 溢出是初装值, 生成中断
    //10000000

    T2CON = 0x80;

    //置初装值到 1000 时钟

    RCAP2L = (65536UL-1000UL);
    RCAP2H = (65536UL-1000UL) >> 8;

    TL2 = RCAP2L;
}

```

```

TH2 = RCAP2H;

ET2 = 1;           //允许定时器 2 中断
TR2 = 1;           //定时器 2 开始运行
EA = 1;            //允许全程中断

while (1)
{
}
}

```

例 10 8051 看门狗程序

8051Fx 系列 PCA 模式 4

```

#include <reg51f.h>

//调整看门狗定时器计算值
//+0xFF00, 先必须写下 CCAP4L, 然后再写下
//CCAP4H

void watchdog_reset (void)
{
    unsigned int newval;

    newval = (CH << 8) | CL;
    newval += 0xFF00;

    CCAP4L = newval;
    CCAP4H = newval >> 8;
}

//主函数功能

void main (void)
{
    unsigned int i;

    //配置 PCA 模块作为看门狗, 确定它不是立即暂停

    watchdog_reset ();
    CCAPM4 = 0x48;

    //配置 PCA 为看门狗定时器

    CMOD = (CMOD & 0x01) | 0x40;

    //开始 PCA 定时器

    CR = 1;

    //不用看门狗复位

    for (i = 0; i < 1000; i++)
    {
        watchdog_reset ();
    }
}

```

```

//等待复位

while (1)
{
}

```

例 11 ADUC812 的 A/D 转换

```

#include <ADUC812.H>
#include <stdio.h>

void main (void)
{
    unsigned char chan_2_convert;

    //Configure the serial port to run at 9600 Baud.
    SCON = 0x50;
    TMOD |= 0x20;
    TH1 = 0xFA; //波特率为 9600
    TR1 = 1;
    TI = 1;

    //每通道 A/D 转换率
    // 0111 1100
    ADCCON1 = 0x7C;

    while (1)
    {
        unsigned int conv_val;
        unsigned char channel;

        //开始转换, 等待完成
        chan_2_convert = (chan_2_convert + 1) % 8;
        ADCCON2 = (ADCCON2 & 0xF0) | chan_2_convert;
        SCONV = 1;
        while (ADCCON3 & 0x80);

        //读 A/D
        channel = ADCDATAH >> 4;
        conv_val = ADCDATAH | ((ADCDATAH & 0x0F) << 8);

        printf ("ADC Channel %bu = 0x%4.4X\r\n", channel, conv_val);
    }
}

```

例 12 ADUC814 的 A/D 转换

```

#include <ADUC814.H>

```

```

#include <stdio.h>

void main (void)
{
    unsigned char chan_2_convert;

    //每通道 A/D 转换率
    ADCCON1 = 0x8C;

    while (1)
    {
        unsigned int conv_val;

        //开始转换, 等待完成
        chan_2_convert = (chan_2_convert + 1) % 6;
        ADCCON2 = (ADCCON2 & 0xF0) | chan_2_convert;
        SCONV = 1;
        while (ADCCON3 & 0x80);

        //读 A/D
        conv_val = ADCDATAL | ((ADCDATAH & 0x0F) << 8);

        printf ("ADC Channel %bu = 0x%4.4X\r\n", chan_2_convert, conv_val);
    }
}

```

例 13 ADUC816 的 A/D 转换

```

#include <ADUC816.H>
#include <stdio.h>

void main (void)
{
    //每通道 A/D 转换率
    ADC1CCN = 0x08;
    SF = 0x0D;

    while (1)
    {
        unsigned int conv_val;

        //开始转换, 等待完成
        ADCMODE |= 0x12;
        while (RDY1 != 1);
        RDY1 = 0;

        //读 A/D
        conv_val = ADC1L | ((ADC1H) << 8);
    }
}

```



```

printf ("Aux ADC Channel 3 = 0x%4.4X\r\n", conv_val);
}

}

```

例 14 ADUC831 的 A/D 转换

```

#include <ADUC831.H>
#include <stdio.h>

void main (void)
{
    unsigned char chan_2_convert;

    //每通道 A/D 转换率

    ADCCON1 = 0xFC; // 1111 1100
    while (1)
    {
        unsigned int conv_val;
        unsigned char channel;

        //开始转换, 等待完成

        chan_2_convert = (chan_2_convert + 1) % 8;
        ADCCON2 = (ADCCON2 & 0xF0) | chan_2_convert;
        SCONV = 1;
        while (ADCCON3 & 0x80);

        //读 A/D

        channel = ADCDATAH >> 4;
        conv_val = ADCDATAH | ((ADCDATAH & 0x0F) << 8);

        printf ("ADC Channel %bu = 0x%4.4X\r\n", channel, conv_val);
    }
}

```

例 15 Apnt 的 A/D 转换

```

#include <absacc.h>
#include <string.h>
#include <stdio.h>
#include <REG51CC01.H>
#include "can_drv.h" //CAN 头文件

char buffer[8]; // 接收缓冲
unsigned short analog [4]; // AN0~AN3 电压

```