

C 8051单片机 语言 彻底应用

赖麒文 编著

特色

- ◆突破艰涩难懂的汇编语言编写，改用易学易懂的C语言
- ◆由浅入深介绍程序结构、设计技巧与理念
- ◆循序渐进扩展程序设计的逻辑推理与思考方法
- ◆参考范例设计或直接选取实例函数套用到实际开发中
- ◆从基础、硬件电路设计和软件编程方法彻底剖析实用的商品化程序

 科学出版社

 文魁资讯股份有限公司

8051 单片机

C 语言彻底应用

赖麒文 编著

科学出版社

2002

内 容 简 介

本书介绍 8051 单片机 C 语言结合硬件编程应用的工程方法。本书通过一个个实用的例子分析,讲解了 C 语言实现自动控制和界面的设计方法、技巧以及常见问题剖析。

本书适合 8051 单片机应用设计人员参考。

图书在版编目(CIP)数据

8051 单片机 C 语言彻底应用/赖麒文编著. —北京:科学出版社, 2002
ISBN 7-03-009054-3

I . 8… II . 赖… III . 单片微型计算机, 8051-C 语言-程序设计
IV . TP312

中国版本图书馆 CIP 数据核字 (2001) 第 074091 号

本书繁体字版原书名为《8051 单片机之 C 语言彻底应用》,由文魁信息股份有限公司出版,版权属赖麒文所有。本书简体字中文版由文魁信息股份有限公司授权科学出版社独家出版。未经本书原版出版者和本书出版者书面许可,任何单位和个人不得以任何形式或任何手段复制或传播本书的部分或全部。

版权所有,翻印必究。

图字: 01-2001-2190 号

MS556/02

科学出版社 出版

北京东黄城根北街16号

邮政编码:100717

<http://www.sciencep.com>

新蕾印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2002 年 1 月第 一 版 开本: 710×1000 1/16

2002 年 1 月第一次印刷 印张: 32

印数: 1-5 000 字数: 628 000

定价: 42.00 元

(如果有印装质量问题,我社负责调换(环伟))

序

现在，市场上的 8051 单片机开发类书籍讲述的多是汇编语言，有关 C 语言的书也是琳琅满目。但是，像本书这样结合硬件讲述用 C 语言开发单片机的应用图书却不多见。本书以 8051 单片机为例，对 C 语言在单片机开发应用相关的程序和电路设计进行了深入浅出的讲述。

8051 单片机广泛应用于键盘、电话机、显示器、电冰箱、洗衣机、空调等等电器中。本书提供了来自产品的现成实例，读者可以把本书作为 8051 软硬件设计的工具书。本书中的这些电路和函数可以非常简便地应用在商品化软硬件开发过程中。也许书中的某些范例程序对你而言一时还派不上用场，但也请你仔细阅读，了解到底有哪些函数、用途是什么以及程序的设计理念。

在本书提供的大量函数中，部分函数的功能和目的是相同的，罗列程序的不同写法并对重要部分一再重复说明，是希望使读者更明了地理解函数，本书的宗旨是提供一些灵感启发读者活用 C 语言。我们希望读者在设计某种功能的程序时，可以方便地参考本书，快速了解程序设计技巧和观念。无论是吸收书中的方法直接套用某个函数，还是将书中的范例融会贯通、举一反三。这样对提高自己的编程水平会有很大帮助。本书附录中给出了书中所有 C 语言范例的源程序编译成的 8051 汇编代码，以方便使用不同类型计算机的读者参考。

目 录

第 1 章 C 语言基本概念	1
1-1 程序的初步	1
1-2 C 程序的运算符	2
1-3 C 程序的流程控制	3
第 2 章 程序的开始	19
主程序 main()	19
#include "define.h"	22
#include "cpu8052.h"	27
#include "global.h"	32
#include <intrins.h>	36
#include <math.h>	36
第 3 章 开机后的启动流程	38
PowerOnInitial()	38
InitialCpu()	39
InitialCpuIO()	41
InitialEeprom()	42
InitialVariable()	43
第 4 章 延时例程	45
DelayX1ms(count)	45
DelayX1ms1(count)	46
DelayX1ms2(count)	46
DelayX10ms(count)	47
DelayX10ms1(count)	48
Delay50uS (count)	49
ShortDelay(count)	49
Timer40msDelay(count)	50

第 5 章 基本输入输出	52
Led_1()	52
LedOn()	53
Input1()	56
Input2()	57
Input3()	63
Input4()	65
Input5()	66
第 6 章 中断的应用	69
CountMain1()	69
Timer0ISR_2()	73
CountMain2()	74
CountMain3()	76
One_INT0ISR()	77
More_INT0ISR()	79
Timer1ISR_1()	83
第 7 章 公用函数	85
UnSignVar()	85
SignVar()	85
ByteVariableAdd1()	86
ByteVariableAdd2()	87
ByteVariableSub()	88
ByteProcess()	89
WordVariableAdd1()	90
WordVariableAdd2()	91
WordVariableSub1()	92
WordVariableSub2()	93
WordProcess()	94
Hex2Bcd1(value)	95
Hex2Bcd2(value)	96
Hex2Bcd3(value)	97
Value255_100(value)	98
Value100_128a(value)	99

Value100_128b(value)	100
RamClear()	101
ZeroContinue(counter)	102
第 8 章 显示器的应用	103
LedFlash0()	103
LedFlash1()	104
LedFlash2()	105
LedFlash3()	106
LedFlash4(ontime,offtime)	107
LedFlash5(count,ontime,offtime)	108
LedFlash6(count,ontime,offtime)	109
LedFlashGetkey(count,ontime,offtime)	110
LedMain1()	111
LedMain2()	112
LedTiming()	115
LedMain3()	120
LedMain4()	122
LedMain5()	124
第 9 章 蜂鸣器的应用	132
Beep1()	132
Beep2(tone)	134
Beep3(soundlong,tone)	135
Beep4(count,soundlong,tone)	136
BeepGetkey(count,soundlong,tone)	137
Alarm1(soundlong,tone)	139
Alarm2(count,soundlong,tone)	140
AlarmGetkey(count,soundlong,tone)	141
BeepLed(count,soundlong,tone)	143
HardWareBeep1()	145
HardWareBeep2()	147
HardWareBeep3()	149

第 10 章 演奏歌曲的应用	151
Sound()	152
Music1()	153
Music2()	155
Music3()	157
Music4(number)	159
第 11 章 七段显示器的应用	162
BcdDisplay1()	162
BcdDisplay2()	164
BcdDisplay3()	166
BcdDisplay4()	169
第 12 章 点阵显示器的应用	171
Dot5x7_Display1()	171
Dot5x7_Display2()	173
Dot5x7_Display3()	174
Dot5x7_Display4()	179
Dot5x7_Display5()	180
Dot5x7_Display6()	182
第 13 章 解码器的应用	184
Output74138_1()	184
Output74138_2()	185
Output74138_3()	187
Output74138_4()	190
第 14 章 扩充输出端口的应用	193
Output4094_1(value)	193
Output4094_5(outputstate,value)	197
第 15 章 脉冲的应用	200
OutPulse1()	200
OutPulse2(count)	201
OutPulse3()	202
OutPulse4()	203

PulseDetect1()	205
PulseDetect2()	207
PulseDetect3()	208
PulseGenerator()	210
PulseDuty1_Timer1ISR()	212
PulseDuty2_Timer1ISR()	214
CheckPulseCome()	215
CheckPulseWidth()	216
CheckPulseData()	218
CheckPulseHiLow()	220
PulseDecoder()	223
EncoderProcess()	225
第 16 章 多任务器的应用	229
Input4051_1()	229
Input4051_2()	230
Input4051_3()	231
Input4051_4()	232
Input4051_5()	235
Input4051_6()	237
Input4067_1()	239
Input4067_2()	243
Input4067_3()	245
第 17 章 键盘操作的应用	247
InputKey1()	247
InputKey2()	248
InputKey3()	250
ScanKey1()	251
ScanKey2()	254
GetKey1()	257
GetKey2()	259
KeyCheck()	262
KeyCountCheck()	264
KeyProcess()	265

第 18 章 可控制电源电压的应用	271
LM7805()	271
LM317()	272
Dac08()	273
SawTooth()	275
TriAngle()	276
Square()	278
第 19 章 存储芯片 93C66 的应用	280
PushEeprom93c66()	280
EepWriteData(adr,value)	282
PopEeprom93c66()	286
ReadROM(adr)	287
第 20 章 IIC BUS 的应用	289
IIC BUS 概念	289
IIC 总线协议	290
开始(Start)	292
地址: (Address)	292
读/写(Read/Write)	292
确认(Acknowledge)	292
数据(Data)	293
停止(Stop)	293
IIC BUS 时序(Timming)	293
I2cStart()	294
I2cStop()	295
I2cWait()	296
I2cSentByte(bytedata)	296
I2cSentByte1(bytedata)	298
I2cReceiveByte()	300
SendAcknowledge(ack)	301
I2cByteWrite(device,address,bytedata)	301
I2cByteWrite1(device,address,bytedata)	302
I2cByteWrite2(device,address,bytedata)	304
I2cByteRead(device,address)	306

I2cSentData(bytecnt)	307
I2cReceiveData(bytecnt)	308
DataSetBit(device,addr,bitno)	309
DataClearBit(device,addr,bitno)	310
第 21 章 PWM IC 的应用	312
PWM_Output()	312
TEST_DacOut()	313
第 22 章 IC 24C08 的应用	318
Eeprom 24c08 命令格式	318
EepromByteWrite0(bank,addr,value)	319
EepromByteRead0(bank,addr)	321
EepromByteWrite(addr,bytedata)	322
EepromByteRead(addr)	322
EepromPageWrite()	323
EepromPageRead()	324
EepromWrite(subaddress,count)	324
EepromRead(subaddress,count)	326
SendEEPROMData()	328
SendData()	329
RcvData()	330
GoMaster(slaveaddr)	332
SendByte(value)	333
SendStop()	335
DdcChecksum(adr)	336
第 23 章 存储器 IC 24C32 的应用	338
EEPROM24c32WriteByte_1(addr,value)	338
EEPROM24c32WriteByte_2(addr,value)	339
EEPROM24c32WriteMulti_1(addr,count)	341
EEPROM24c32WriteMulti_2(addr,count)	342
EEPROM24c32ReadByte_1(addr)	343
EEPROM24c32ReadByte_2(addr)	344
EEPROM24c32ReadWord_1(addr)	346
EEPROM24c32ReadWord_2(addr)	347

EEPROM24c32ReadMulti_1(addr,count)	349
EEPROM24c32ReadMulti_2(addr,count)	350
第 24 章 OSD IC 的应用	352
OsdStart()	352
OsdStop()	353
OsdSentByte(bytedata)	354
OsdReceiveByte()	356
OsdFormatA_0(row,col,value)	357
OsdFormatA(row,col,value)	358
OsdFrameControl(vertd,hord,height,width,rowspace)	358
OsdLocationSet(vertical,horizontal)	359
OsdRamClear()	360
OsdEnable(yes)	361
OsdOpenUp()	361
OsdNormal()	362
OsdResetFont()	362
OsdClearRow(start,end,color)	363
OsdClearRow1(start,end,color)	364
OsdPrintIcon(row,col,icon,color)	366
OsdStringAdr0(*string,sel)	367
OsdStringAdr(*string,total,sel,fglanguage)	368
OsdPrintString(row,col,color,*string)	369
OsdPrintString1(row,col,color,*string)	371
OsdDisableWindow1(sub_window)	373
OsdSetWindow(sub_window,row_start,row_end, column_start,column_end,attribute)	374
OsdBarHandle(row,col,color)	375
OsdBarHandle1(row,col,color)	378
OsdDisplayValue(row,col,color)	380
OsdDisplayCount(count)	383
附录 A 头文件	385
附录 B 汇编程序	404

第 1 章 C 语言基本概念

1-1 程序的初步

/* -----*/: C 程序的注释

➤ void main(void)

C 程序从 main 开始执行，前一个 void，表示无返回值；后一个 void，表示不传参数。

int i ;

int: 声明整数类型

i : 变量名

; : 语句结束符号

表 1-1 数据类型的长度

数据类型	位数	字节数	值 域
bit	1		0 ~ 1
signed char	8	1	-128 ~ +127
unsigned char	8	1	0 ~ 255
enum	16	2	-32768 ~ +32767
signed short	16	2	-32768 ~ +32767
unsigned short	16	2	0 ~ 65535
signed int	16	2	-32768 ~ +32767
unsigned int	16	2	0 ~ 65535
signed long	32	4	-2147483648 ~ 2147483647
unsigned long	32	4	0 ~ 4294967295
float	32	4	0.175494E-38 ~ 0.402823E+38
sbit	1		0 ~ 1
sfr	8	1	0 ~ 255
sfr16	16	2	0 ~ 65535

1-2 C 语言的运算符

表 1-2 C 语言运算

运算符	范 例	说 明
+	a+b	a 变量值和 b 变量值相加
-	a-b	a 变量值和 b 变量值相减
*	a*b	a 变量值乘以 b 变量值
/	a/b	a 变量值除以 b 变量值
%	a%b	取 a 变量值除以 b 变量值的余数
=	a=6	将 6 设定给 a 变量, 即 a 变量值等于 6
+=	a+=b	等同于 a=a+b, 将 a 和 b 相加的结果又存回 a
-=	a-=b	等同于 a=a-b, 将 a 和 b 相减的结果又存回 a
=	a=b	等同于 a=a*b, 将 a 和 b 相乘的结果又存回 a
/=	a/=b	等同于 a=a/b, 将 a 和 b 相除的结果又存回 a
%=	a%=b	等同于 a=a%b, a 变量值除以 b 变量值的余数又存回 a
++	a++	a 的值加 1, 即 a=a+1
--	a--	a 的值减 1, 即 a=a-1
>	a>b	测试 a 是否大于 b
<	a<b	测试 a 是否小于 b
==	a==b	测试 a 是否等于 b
>=	a>=b	测试 a 是否大于或等于 b
<=	a<=b	测试 a 是否小于或等于 b
!=	a!=b	测试 a 是否不等于 b
&&	a && b	a 和 b 作逻辑 AND, 两个变量是都是“真”, 结果才为“真” 否则结果为“0”
	a b	a 和 b 作逻辑 OR, 只要有任何一个变量为“真”, 结果就为“真”
!	! a	将 a 变量的值取反, 即原来为“真”则变“假”, 为“假”则变为“真”
>>	a>>b	将 a 按位右移 b 个位
<<	a<<b	将 a 按位左移 b 个位, 右侧补“0”
	a b	a 和 b 的按位做 OR 运算
&	a & b	a 和 b 的按位做 AND 运算
^	a ^ b	a 和 b 的按位做 XOR 运算
~	~a	将 a 的每一位取反
&	a=&b	将 b 变量的地址存入 a 寄存器
*	*a	用来取寄存器所指地址内的值

注意: 在逻辑运算中, 凡是结果为非“0”的数值即为真, 等于“0”为假。

◆ 范例一

```
a=1;
b=++a;
```

其运算过程是 a 值加 1 变为 2，然后再将 2 赋值给 b，所以 b=2，a=2。

◆ 范例二

```
a=1;
b=a++;
```

其运算过程是 a 原先的值 1，先赋值给 b，然后 a 再加 1 变为 2，所以 b=1，a=2。

1-3 C 程序的流程控制

➤ if 语句

```
1. if (条件表达式)
    { 动作 }
```

如果条件表达式的值为真(非零的数)，则执行{ }内的动作，如果条件表达式为假，则略过该动作而继续往下执行。

◆ 范例

```
01 void IfDemol(void)
02 {
03     Byte i,j;
04
05     if ( DisplayState<10 )
06
07         for(i=0; i<5; i++)
08         {
09             j=5 * DisplayState;
10             P2=DISPLAY_TABLE10[j+i];
11             P1=0x01 << i;
12             DelayX1ms(3);
13         }
```

```
14     }  
15 }  
2.  if (条件表达式)  
    {    动作 1    }  
    else  
    {    动作 2    }
```

如果条件表达式为真，则执行动作 1，略过 else 的部分，接着往下执行，如果条件表达式为假，则略过 if 的部分而执行 else 的动作 2，然后再往下执行。

◆ 范例

```
01 void IfDemo2(void)  
02 {  
03     if (LedCount<26)  
04         LedOn( );  
05     else  
06         LedOff( );  
07 }
```

```
3.  if (条件表达式 1)  
    if (条件表达式 2)  
        if (条件表达式 3)  
            { 动作 A  }  
        else  
            { 动作 B  }  
    else  
        { 动作 C  }  
else  
    { 动作 D  }
```

动作 A：是条件表达式 1、2、3 都成立时才会执行。

动作 B：是条件表达式 1、2 成立，但条件表达式 3 不成立时才会执行。

动作 C：是条件表达式 1 成立，条件表达式 2 不成立时才会执行。

动作 D：是条件表达式 1 不成立时才会执行。

◆ 范例

```
01 void IfDemo3(void)  
02 {
```



```
03     if ( FgPulse==0 )
04     {
05         if ( P1_7==0 )
06         {
07             FgPulse=1;
08             LedOn( );
09         }
10     else
11     {
12         FgPulse=0;
13         LedOff( );
14     }
15 }
16 else
17 {
18     if ( P1_7!=0 )
19         FgPulse=0;
20 }
21 }
```

```
4.  if      (条件表达式 1)
      {      动作 A      }
    else if (条件表达式 2)
      {      动作 B      }
    else if (条件表达式 3)
      {      动作 C      }
    else
      {      动作 D      }
```

动作 A: 是条件表达式 1 成立时立即执行。

动作 B: 是条件表达式 1 不成立, 但条件表达式 2 成立时才会执行。

动作 C: 是条件表达式 1、2 不成立, 条件表达式 3 成立时才会执行。

动作 D: 是条件表达式 1、2、3 都不成立时才会执行。

◆ 范例

```
01 void IfDemo4(void)
02 {
```