第 10 章习题

训练题 10-1 解：参考程序如下：

```
#include<REG52.H>
#include <intrins.h>
#define uchar unsigned char
#define uint unsigned int
sbit   CE =P1^0;                    //定义使能
sbit   SDI=P1^5;                    //定义数据输入端口
sbit   SDO=P1^6;                    //定义数据输出端口
sbit   CLK=P1^7;                    //定义时钟端口
sbit   ACC7=ACC^7;                  //定义累加器 A 的第 8 位
sbit   ACC0=ACC^0;                  //定义累加器 A 的第 1 位

void DS1306_Write(uchar addr,uchar date)        //DS1306 写程序
{
    uchar i;
    CE=0;
    _nop_();
    CE=1;                           //使能端置 1，数据有效
    for(i=0;i<8;i++)                //传送地址
    {
        CLK=0;                      //清时钟总线
        ACC=addr;
        SDI=ACC7;                   //将 addr 高位数据送到 SDI
        CLK=1;                      //时钟上升沿发送数据
        _nop_();
        _nop_();
        addr=(addr<<1);         //准备传送下一位数据

    }
    for(i=0;i<8;i++)                //传送数据
    {
        CLK=0;                      //清时钟总线
        ACC=date;
        SDI=ACC7;                   //将 date 高位数据送到 SDI
        CLK=1;                      //时钟上升沿发送数据
        _nop_();
        _nop_();
        date=(date<<1);         //准备传送下一位数据
    }
    CE=0;                           //使能端置 0，数据没效
}

uchar DS1306_Read(uchar addr)   // DS1306 读程序
{
    uchar i,DA;
```

```
    CE=0;
    _nop_();
    CE=1;                       //使能端置 1，数据有效
    for(i=0;i<8;i++)            //传送地址
    {
        CLK=0;                  //清时钟总线
        ACC=addr;
        SDI=ACC7;               //将 addr 高位数据送到 SDI
        CLK=1;                  //时钟上升沿发送数据
        _nop_();
        _nop_();
        addr=(addr<<1);         //准备传送下一位数据
    }
    for(i=0;i<8;i++)            //接收数据
    {

        CLK=0;                  //时钟下降沿将数据输出
        ACC=(ACC<<1);
        ACC0=SDO;               //将接收到的数据放在 ACC 中
        _nop_();
        _nop_();
        CLK=1;                  //将时钟线置高
        _nop_();
        _nop_();
    }
    DA=ACC;                     //将接收到的一个字节数据赋给 DA
    CE=0;                       //使能端置 0，数据没效
    return(DA);                 //返回 DA 值
}

void DS1306_int()               // 对 DS1306 初始化，即“对表”程序
{
    DS1306_Write(0x8f,0x04);
    DS1306_Write(0x80,0x00);    //秒初始化 "00"秒
    DS1306_Write(0x81,0x09);    //分初始化 "09"分
    DS1306_Write(0x82,0x10);    //小时初始化 "10"时
    DS1306_Write(0x83,0x03);    //星期初始化 星期“3”
    DS1306_Write(0x84,0x19);    //日初始化“19”日
    DS1306_Write(0x85,0x04);    //月初始化 “04”月
    DS1306_Write(0x86,0x10);    //年初始化“10”年
    DS1306_Write(0x8f,0x44);    //写保护，禁止向 DS1306 写数据
}
void main()
{
    uchar S,M,H,D,W,Mo,Y;
    DS1306_int();
```

```
    while(1)
    {
        S=DS1306_Read(0x00);      //读取秒数据
        P0=S;
        M=DS1306_Read(0x01);      //读取分数据
        H=DS1306_Read(0x02);      //读取小时数据
        W=DS1306_Read(0x03);      //读取周数据
        D=DS1306_Read(0x04);      //读取日数据
        Mo=DS1306_Read(0x05);     //读取月数据
        Y=DS1306_Read(0x06);      //读取秒数据
    }
}
```

注意：本题程序还可以优化。即将单字节数据读/写方式，改为批量数据读/写方式。

训练题 10-2 解：驱动 CAT1161 的汇编程序如下，仅供参考。

```
        FADDR     EQU    0A0h              ；CAT1161- I2C 总线固定地址
        ；  寄存器定义
        INDEX     EQU    R0                ；  缓冲区指针
        kount     EQU    R1                ；  字节计数器
        zdata     EQU    R1                ；  数据寄存器
        Addr      EQU    R2                ；  字节地址
        buffer    EQU    40H               ；  片内缓冲区首址
        ；  I2C 总线定义.
        SCL       BIT    P1.5              ；  串行时钟
        SDA       BIT    P1.6              ；  串行数据

        ORG       0000H
        LSJMP     ONRESET
        ORG       0080H
ONRESET：MOV      SP，#5FH
        SETB      SDA                      ；初始化总线
        SETB      SCL
        MOV       Addr，#0FFH
        MOV       zdata，#55H
        CLR       A
        LCALL     write_byte               ；向 CAT1161-0FFH 单元，写入 55H
        LCALL     DELY10mS

        MOV       Addr，#0FFH
        CLR       A
        LCALL     read_random              ；从 CAT1161-0FFH 单元，读数据验证=55H?

        MOV       Addr，#0FFH
        MOV       zdata，#66H
        MOV       A，#7
        LCALL     write_byte               ；向 CAT1161-7FFH 单元，写入 66H
```

```
            LCALL   DELY10Ms

            MOV     Addr，#0FFH
            MOV     A，#7
            LCALL   read_random     ；从 CAT1161-0FFH 单元，读数据验证=66H？

            MOV     Addr，#10H
            MOV     kount，#10H
            MOV     A，#1
            LCALL   write_block     ；向 CAT1161-100H 开始单元写入 16 字节数据
            LCALL   DELY10mS

            MOV     Addr，#10H
            MOV     kount，#10H
            MOV     A，#1
            LCALL   read_block:     ；从 CAT1161-100H 单元开始读 16 字节验证？
            SJMP    $
START：     SETB    SDA
            SETB    SCL
            JNB     SDA，X40        ；  总线不正确跳转
            JNB     SCL，X40        ；  不正确跳转
            NOP                     ；  延时
            CLR     SDA
            NOP                     ；  强制延时
            NOP
            NOP
            NOP
            NOP
            CLR     SCL
            CLR     C               ；  清错误标志
            SJMP    X41
X40：       SETB    C               ；  置错误标志
X41：       RET
STOP：      CLR     SDA             ；  总线停止
            NOP
            NOP
            SETB    SCL
            NOP
            NOP
            NOP
            NOP
            NOP
            SETB    SDA
            RET
SHOUT：     PUSH    B               ；字节写，数据在 A 中，高位先写
            MOV     B，#8
```

```
X42：   RLC     A           ；  移数据位进 CY
        MOV     SDA，C      ；  输出数据位
        NOP                 ；  SCL 低，数据写入
        SETB    SCL
        NOP
        NOP
        NOP
        NOP
        CLR     SCL
        DJNZ    B，X42      ；  下一位
        SETB    SDA         ；  ACK
        NOP
        NOP
        SETB    SCL
        NOP
        NOP
        NOP
        NOP
        MOV     C，SDA      ；  提取 ACK 位
        CLR     SCL
        POP     B
        RET
SHIN：  SETB    SDA         ；  读字节，数据在 A 中
        PUSH    B
        MOV     B，#8       ；  bit cOunt
X43：   NOP                 ；  enfOrce SCL lOw And data setup
        NOP
        NOP
        SETB    SCL         ；  rAise clOck
        NOP                 ；  enfOrce SCL high
        NOP
        MOV     C，SDA      ；  移出数据进 CY
        RLC     A           ；  移入 A 中
        CLR     SCL     `
        DJNZ    B，X43      ；  读下一位
        POP     B
        RET
ACK：   CLR     SDA         ；  主控器应答子程序，SDA 低有效
        NOP
        NOP
        SETB    SCL
        NOP
        NOP
        NOP
        NOP
        CLR     SCL
```

```
        RET
NAK：   SETB    SDA             ；  主控器应答子程序，SDA 高
        NOP
        NOP
        SETB    SCL
        NOP
        NOP
        NOP
        NOP
        CLR     SCL
        RET

write_block：；  连续写，字节地址由 ADDR 中指出
        ；  KOUNT 为计数器；CY=1，操作或应答失败
        LCALL   START
        JC      X38             ；  总线失败
        RL      A
        ORL     A，  #FADDR     ；  CAT1161 从地址（包括 A10~A8）
        LCALL   SHOUT           ；  写从地址
        JC      X37             ；  应答失败
        MOV     A，Addr         ；  送字节地址
        LCALL   SHOUT
        JC      X37             ；  无应答
        MOV     INDEX，#buffer  ；  存于片内 buffer 区
X36：   MOV     A，  @INDEX
        LCALL   SHOUT
        JC      X37             ；  无应答
        INC     INDEX
        DJNZ    kount，  X36    ；  下一字节
        CLR     C               ；  清错误标志
X37：   LCALL   STOP
X38：   RET

read_block:：；  连续读（选择读方式），器件字节地址由 ADDR 指出
        ；  KOUNT 为计数器；CY=1，操作或应答失败
        LCALL   START
        JC      X35             ；  总线失败
        RL      A
        ORL     A，  #FADDR     ；  CAT1161 从地址（包括 A10~A8）
        SETB    Acc.0           ；  读命令
        LCALL   SHOUT
        JC      X34             ；  无应答
        MOV     A，Addr         ；  送字节地址
        LCALL   SHOUT
        JC      X34             ；  无应答
        LCALL   START           ；  START
```

```
            JC      X34             ；  无应答
            MOV     A， #FADDR          ； CAT1161 从地址
            SETB    Acc.0           ；  写命令
            LCALL   SHOUT           ；  送器件地址
            JC      X34
            MOV     INDEX， #buffer ；  读出的数据存于 buffer
X31：       LCALL   SHIN            ；  接收数据字节
            MOV     @INDEX，A       ；  存数据
            CLNE    kount，#1，X32  ；  不是最后字节转
            LCALL   NAK             ；  最后一个字节不应答
            SJMP    X33             ；  完成
X32：       LCALL   ACK             ；  每个字节都应答 AcknOwledge
            INC     INDEX
            DJNZ    kount，X31      ；  下一字节
X33：       CLR     C
X34：       LCALL   STOP
X35：       RET

write_byte：LCALL   START           ；  字节写
            JC      X49
            RL      A
            ORL     A， #FADDR      ；  CAT1161 从地址（包括 A10~A8）
            LCALL   SHOUT
            JC      X48
            MOV     A，Addr
            LCALL   SHOUT
            JC      X48
            MOV     A，zdata        ；  A 得到数据
            LCALL   SHOUT
            JC      X48
            CLR     C
X48：       LCALL   STOP
X49：       RET

read_current：LCALL START           ；  当前/立即读方式
            JC      X45             ；  AbOrt if bus nOt AvAilAble
            RL      A
            ORL     A， #FADDR      ；  CAT1161 从地址（包括 A10~A8）
            SETB    Acc.0           ；  读命令
            LCALL   SHOUT
            JC      X44
            LCALL   SHIN            ；  读字节
            LCALL   NAK             ；  不应答
            CLR     C               ；  清错误标志
X44：       LCALL   STOP
X45：       RET
```

```
read_random：LCALL    START              ；  选择/自由读方式
             JC       X47
             RL       A
             ORL      A，  #FADDR        ；  CAT1161 从地址（包括 A10~A8）
             SETB     Acc.0              ；  读命令
             LCALL    SHOUT
             JC       X46
             MOV      A，addr
             LCALL    SHOUT
             JC       X46
             LCALL    read_current       ；   接当前/立即读方式完成
             SJMP     X47                ；  退出
X46：        LCALL    STOP
X47：        POP      B
             RET
             END
DELY10mS：   MOV      R7，#40
AGAIN：      MOV      R6，#0
WAIT：       DJNZ     R6，WAIT
             DJNZ     R7，AGAIN
             RET                         ；  12MHz 主频时，延时约为 10mS
```

训练题：10-3

得到 ROM 编码为 0101 器件的过程如表 训练题 10-1 所示。

→ROM 编码的“排除”搜索算法进行的方向→											
器件码	两读	一写	器件码	两读	一写	器件码	两读	一写	器件码	两读	一写
0000	00	0	000	00	1	——	01	0		10	0/1
1111											转入
0101			101			01			1		下一
1010											轮
识别位		0			1			0		1	循环

得到 ROM 编码为 1010 器件的过程如表 训练题 10-2 所示。

→ROM 编码的“排除”搜索算法进行的方向→											
器件码	两读	一写	器件码	两读	一写	器件码	两读	一写	器件码	两读	一写
0000	00	1		00	0		10	1	10	01	0/1
1111			111			——					转入
0101											下一
1010			010			10			0		轮
识别位		1			0			1		0	循环

得到 ROM 编码为 1111 器件的过程如表 训练题 10-3 所示。

→ROM 编码的“排除”搜索算法进行的方向→

器件码	两读	一写	器件码	两读	一写	器件码	两读	一写	器件码	两读	一写
0000	00	1		00	1		10	1		10	0/1
1111			111			11			1		转入
0101											下一
1010			010			——					轮
识别位		1			1			1		1	循环

训练题 10-4

解：参考的 C51 程序如下：

```
#include <intrins.h>
#define uchar unsigned char
#define uint unsigned int
#define DQ P1_0
static bdata uchar sclkdata;
sbit sclkdata0 = sclkdata^0;sbit sclkdata1 = sclkdata^1;/*DS1820 用变量*/
sbit sclkdata2 = sclkdata^2;sbit sclkdata3= sclkdata^3;
sbit sclkdata4 = sclkdata^4;sbit sclkdata7 = sclkdata^7;
sbit sclkdata5 = sclkdata^5;sbit sclkdata6 = sclkdata^6;
idata uchar datal, datah, id_buff[9];
idata uint temepra_ture;
code uchar Table_CRC[256]={      /*cnCRC_CCITT*/
    0,94,188,226,97,63,221,131,194,156,126,32,163,253,31,65,
    157,195,33,127,252,162,64,30,95,1,227,189,62,96,130,220,
    35,125,159,193,66,28,254,160,255,191,93,3,128,222,60,98,
    190,224,2,92,223,129,99,61,124,34,192,158,29,67,161,255,
    70,24,250,164,39,121,155,197,132,218,56,102,229,187,89,7,
    219,133,103,57,186,228,6,88,25,71,165,251,120,38,196,154,
    101,59,217,135,4,90,184,230,167,249,27,69,198,152,122,36,
    248,166,68,26,153,199,37,123,58,100,134,216,91,5,231,185,
    140,210,48,110,237,179,81,15,78,16,242,172,47,113,147,205,
    17,79,173,243,112,46,204,146,211,141,111,49,178,236,14,80,
    175,241,19,77,206,144,114,44,109,51,209,143,12,82,176,238,
    50,108,142,208,83,13,239,177,240,174,76,18,145,207,45,115,
    202,148,118,40,171,245,23,73,8,86,180,234,105,55,213,139,
    87,9,235,181,54,104,138,212,149,203,41,119,244,170,72,22,
    233,183,85,11,136,214,52,106,43,117,151,201,74,20,246,168,
    116,42,200,150,21,75,169,247,182,232,10,84,215,137,107,53};

/*********************以下是 2401 函数组*************************/
delay(void)                                  /*延时 0.05ms 函数*/
{
    uchar i;
    for (i=0;i<4;i++)
    {;}
}
```

```
write()                                 /*写一个字节函数*/
{
    uchar i,j;
    for (j=0;j<8;j++)
    {   DQ = 0;
        _nop_();
        _nop_();                        /*延时  2us*/
        if( sclkdata0==1)
        DQ = 1;
        else DQ = 0;
        for (i=0;i<5;i++);              /*等待 64us*/
        _nop_();
        _nop_();                        /*延时  2us*/
        DQ = 1;                         /*释放总线*/
        sclkdata >>=1;
    }
}

read()                                  /*读一个字节函数*/
{
    uchar i,j;
    for (j=0;j<8;j++)
    {
        sclkdata >>=1;
        DQ = 0;
        _nop_();
        _nop_();                        /*延时  2us*/
        DQ = 1;                         /*延时  2uS */
        sclkdata = sclkdata;sclkdata = sclkdata;sclkdata = sclkdata;sclkdata = sclkdata;
        sclkdata = sclkdata;sclkdata = sclkdata; /*延时  12uS */
        if (DQ == 1)
        sclkdata7 =1;
        else sclkdata7 =0;
        for (i=0;i<4;i++);              /*等待 51us*/
     }
}

void reset(void)                        /*复位函数*/
{
    uchar tt;                           /*这种临时变量最省空间*/
    tt = 0;
    DQ = 0;                             /*复位脉冲开始*/
    delay();delay();delay();delay();delay();
    delay();delay();delay();delay();    /*保持低电平 0.5ms*/
    DQ = 1;                             /*释放总线*/
```

```
    delay();                        /*保持低电平 0.05ms*/
    while (DQ ==1)                  /*等待总线变低*/
        {
            tt++;
            if (tt>200)
            break;
        }
    delay();                        /*保持低电平 0.05ms*/
    tt = 0;
    while (DQ ==0)                  /*等待总线变高*/
        {
            tt++;
            if (tt>200)
            break;
        }
    DQ = 1;                         /*释放总线*/
}

void read2401(void)
{
    uchar j;
    reset();                        /*DS1822 复位*/

    sclkdata = 0x33;                /*启动读取序列号*/
    write();

    for (j=0;j<8;j++)
    {
        read();
        id_buff[i] = sclkdata;
    }
}

/*计算 1 位 CRC 值，采用 CRC-CCITT*/
uchar CRC_8(aData, aSize)
uchar *aData;
uchar aSize;
{
    /*  aData：表示要传输的数据首地址，aSize 表示数据的字节个数*/
    static idata uchar i;
    static idata uchar nAccum,x;

    nAccum = 0;
    for (i = 0; i < aSize; i++)
    {
        x = (nAccum  ^ *aData++);
```

```
        nAccum = Table_CRC[x];
        /*nAccum = (nAccum ) ^ Table_CRC[(nAccum ) ^ *aData++]; */
    }
    return nAccum;
}

/*************************************************************************/
main()
{
    uchar i,j,CRC;

    Read2401();                         /*读 2401 的 ID*/
    CRC = CRC_8(id_buff,7);             /*CRC 校验*/
    if (CRC == id_buff [7])
    {
        Sclkdata2 = 1;                  /*CRC 校验成功，sclkdata2 = 1*/
    }
    else
    {
        Sclkdata2 = 0;                  /*CRC 校验不成功，sclkdata2 = 1*/
    }
}
```

刘焕成　2011 年 03 月 08 日修改 2