模拟 I²C 总线 C51 程序软件包

一、 C51 程序软件包

此软件包用在单主方式下的 I²C 总线，硬件接口是 SDA，SCL，使用 MCU 的 I/O 口作 SDA、SCL。

软件包的接口界面：

（1） bit ISendByte(uchar sla,uchar c) （无子地址）读单字节数据 （现行地址读）

（2） bit IRcvByte(uchar sla,uchar *c) （无子地址）写单字节数据 （现行地址写）

（3） bit ISendStr(uchar sla,uchar suba,uchar *s,uchar no)（有子地址）读 N 字节数据

（4） bit IRcvStr(uchar sla,uchar suba,uchar *s,uchar no) （有子地址）写 N 字节数据

- 每一个函数都有返回值，当返回值为 1 时表示操作成功，否则操作失败。
- 参数说明：sla 为器件从地址， suba 为器件子地址， *s 数据接收/发送区指针，no 接收/发送字节数。
- 现行地址读/写：有子地址器件，不给定子地址的读/写操作。
- 设计有/无子地址子程序是根据 I²C 器件的特点，把地址和数据彻底分开。

使用时只要把 VI2C_C51.LIB 复制到 C51\LIB，把 VI2C_C51.H 复制到 C51\INC，然后在用户程序开头加入

```
#include <VI2C_C51.h>
```

即可以使用上面的函数。但这样做有两点限制 ：

（1） I²C 总线 SDA、SCL 只能使用 MCU 的 P1.3(SDA)、P1.2(SCL)。

（2） MCU 的 Fosc<=12MHz（时钟周期为标准 80C51 模式即 12 Clock)。当你的系统不希望受这两个条件限制时，你可以对 VIIC_C51.C 的如下设置

```
sbit SDA=P1^3;                  /*模拟 I²C 数据传送位*/
sbit SCL=P1^2;                  /*模拟 I²C 时钟控制位*/
```

以及__Nop()的个数进行修改，然后再把这个文件包含到用户程序中。

```
/*****************************************************************************
   VI2C_C51.C
   此程序是 I²C 操作平台（主方式的软件平台）的底层的 C 子程序,如发送数据
及接收数据,应答位发送,并提供了几个直接面对器件的操作函数，它很方便的与用
户程序连接并扩展。
   注意:函数是采用软件延时的方法产生 SCL 脉冲,对高晶振频率要作一定的修改
(本例是 1us 机器周期,即晶振频率要小于 12MHz)。
*****************************************************************************/
#include <reg764.h>                 /*头文件的包含*/
#include <intrins.h>
#define   uchar unsigned char       /*宏定义*/
#define   uint   unsigned int
#define   _Nop()   _nop_()          /*定义空指令*/
/*端口位定义*/
sbit SDA=P1^3;                  /*模拟 I2C 数据传送位*/
sbit SCL=P1^2;                  /*模拟 I2C 时钟控制位*/
```

```
/*状态标志*/
bit ack;                        /*应答标志位*/

/*****************************************************************
起动总线函数
函数原型: void  Start_I2c();
功能:启动 I2C 总线,即发送 I2C 起始条件
*****************************************************************/
void Start_I2c()
{
  SDA=1;    /*发送起始条件的数据信号*/
  _Nop();
  SCL=1;
  _Nop();     /*起始条件建立时间大于 4.7us,延时*/
  _Nop();
  _Nop();
  _Nop();
  _Nop();
  SDA=0;    `/*发送起始信号*/
  _Nop();     /*  起始条件锁定时间大于 4μs*/
  _Nop();
  _Nop();
  _Nop();
  _Nop();
  SCL=0;    /*钳住 I2C 总线，准备发送或接收数据  */
  _Nop();
  _Nop();
}

/*********************************************
结束总线函数
函数原型: void  Stop_I2c();
功能:结束 I2C 总线,即发送 I2C 结束条件
*********************************************/
void Stop_I2c()
{
  SDA=0;   /*发送结束条件的数据信号*/
  _Nop();    /*发送结束条件的时钟信号*/
  SCL=1;   /*结束条件建立时间大于 4μs*/
  _Nop();
  _Nop();
  _Nop();
  _Nop();
  _Nop();
```

```
   SDA=1;   /*发送 I²C 总线结束信号*/
   _Nop();
   _Nop();
   _Nop();
   _Nop();
}

/*****************************************************************
字节数据传送函数
函数原型: void    SendByte(uchar c);
功能:将数据 c 发送出去,可以是地址,也可以是数据,发完后等待应答,并对此状
      态位进行操作(不应答或非应答都使 ack=0  假) 。发送数据正常，ack=1;
      ack=0 表示被控器无应答或损坏。
*****************************************************************/
void    SendByte(uchar c)
{
 uchar BitCnt;

 for(BitCnt=0;BitCnt<8;BitCnt++)   /*要传送的数据长度为 8 位*/
     {
      if((c<<BitCnt)&0x80)SDA=1;     /*判断发送位*/
        else    SDA=0;
      _Nop();
      SCL=1;                        /*置时钟线为高，通知被控器开始接收数据位*/
       _Nop();
       _Nop();                       /*保证时钟高电平周期大于 4μs*/
       _Nop();
       _Nop();
       _Nop();
      SCL=0;
     }
     _Nop();
     _Nop();
     SDA=1;                        /*8 位发送完后释放数据线，准备接收应答位*/
     _Nop();
     _Nop();
     SCL=1;
     _Nop();
     _Nop();
     _Nop();
     if(SDA==1)ack=0;
        else ack=1;              /*判断是否接收到应答信号*/
     SCL=0;
     _Nop();
```

```
    _Nop();
}

/*****************************************************************
字节数据传送函数
函数原型: uchar   RcvByte();
功能:用来接收从器件传来的数据,并判断总线错误(不发应答信号),
    发完后请用应答函数。
*****************************************************************/
uchar   RcvByte()
{
  uchar retc;
  uchar BitCnt;

  retc=0;
  SDA=1;                    /*置数据线为输入方式*/
  for(BitCnt=0;BitCnt<8;BitCnt++)
      {
        _Nop();
        SCL=0;          /*置时钟线为低，准备接收数据位*/
        _Nop();
        _Nop();             /*时钟低电平周期大于 4.7μs*/
        _Nop();
        _Nop();
        _Nop();
        SCL=1;          /*置时钟线为高使数据线上数据有效*/
        _Nop();
        _Nop();
        retc=retc<<1;
        if(SDA==1)retc=retc+1; /*读数据位,接收的数据位放入 retc 中  */
        _Nop();
        _Nop();
      }
  SCL=0;
  _Nop();
  _Nop();
  return(retc);
}

/*****************************************************************
应答子函数
原型:   void Ack_I2c(bit a);
功能:主控器进行应答信号,(可以是应答或非应答信号)
*****************************************************************/
```

```
void Ack_I2c(bit a)
{
  if(a==0)SDA=0;      /*在此发出应答或非应答信号 */
          else SDA=1;
  _Nop();
  _Nop();
  _Nop();
  SCL=1;
    _Nop();
    _Nop();                    /*时钟低电平周期大于 4μs*/
    _Nop();
    _Nop();
    _Nop();
 SCL=0;                        /*清时钟线，钳住 I2C 总线以便继续接收*/
    _Nop();
    _Nop();
}

/*****************************************************************
向无子地址器件发送字节数据函数
函数原型: bit   ISendByte(uchar sla,ucahr c);
功能:从启动总线到发送地址，数据，结束总线的全过程,从器件地址 sla。如果
     返回 1 表示操作成功，否则操作有误。
*****************************************************************/
bit ISendByte(uchar sla,uchar c)
{
   Start_I2c();                 /*启动总线*/
   SendByte(sla);               /*发送器件地址*/
     if(ack==0)return(0);
   SendByte(c);                 /*发送数据*/
     if(ack==0)return(0);
  Stop_I2c();                   /*结束总线*/
  return(1);
}

/*****************************************************************
向有子地址器件发送多字节数据函数
函数原型: bit   ISendStr(uchar sla,uchar suba,ucahr *s,uchar no);
功能:从启动总线到发送地址，子地址,数据，结束总线的全过程,从器件地址 sla，
     子地址 suba，发送内容是 s 指向的内容，发送 no 个字节。如果返回 1 表示
     操作成功，否则操作有误。
*****************************************************************/
bit ISendStr(uchar sla,uchar suba,uchar *s,uchar no)
{
```

```
    uchar i;

    Start_I2c();                    /*启动总线*/
    SendByte(sla);                  /*发送器件地址*/
      if(ack==0)return(0);
    SendByte(suba);                 /*发送器件子地址*/
      if(ack==0)return(0);
    for(i=0;i<no;i++)
     {
      SendByte(*s);                 /*发送数据*/
        if(ack==0)return(0);
      s++;
     }
 Stop_I2c();                        /*结束总线*/
  return(1);
}

/*********************************************************************
向无子地址器件读字节数据函数
函数原型: bit   IRcvByte(uchar sla,ucahr *c);
功能:从启动总线到发送地址，读数据，结束总线的全过程,从器件地址 sla，返
     回值在 c。如果返回 1 表示操作成功，否则操作有误。
*********************************************************************/
bit IRcvByte(uchar sla,uchar *c)
{
    Start_I2c();                    /*启动总线*/
    SendByte(sla+1);                /*发送器件地址*/
      if(ack==0)return(0);
    *c=RcvByte();                   /*读取数据*/
      Ack_I2c(1);                   /*发送非就答位*/
  Stop_I2c();                       /*结束总线*/
  return(1);
}

/*********************************************************************
向有子地址器件读取多字节数据函数
函数原型: bit   ISendStr(uchar sla,uchar suba,ucahr *s,uchar no);
功能:从启动总线到发送地址，子地址,读数据，结束总线的全过程,从器件地址 sla，
     子地址 suba，读出的内容放入 s 指向的存储区，读 no 个字节。如果返回 1
     表示操作成功，否则操作有误。
*********************************************************************/
bit IRcvStr(uchar sla,uchar suba,uchar *s,uchar no)
{
    uchar i;
```

```
    Start_I2c();                    /*启动总线*/
    SendByte(sla);                  /*发送器件地址*/
      if(ack==0)return(0);
    SendByte(suba);                 /*发送器件子地址*/
      if(ack==0)return(0);
    Start_I2c();
    SendByte(sla+1);
       if(ack==0)return(0);

    for(i=0;i<no-1;i++)
     {
      *s=RcvByte();                 /*发送数据*/
       Ack_I2c(0);                  /*发送就答位*/
      s++;
     }
    *s=RcvByte();
     Ack_I2c(1);                    /*发送非应位*/
  Stop_I2c();                       /*结束总线*/
  return(1);
}
                              /*      END         */
```

二、使用方法举例

```
#include <VI2C_C51.h>               /*头文件的包含*/
#define   uchar unsigned char       /*宏定义*/
#define   uint   unsigned int
#define    CSI24WCXX   0XA0
#define    PCF8574       0X40
#define    SAA1064       0X70
#include <SAA1064_C51.h>                  /*器件驱动函数*/
uchar disp_ram[4];                        /*定义显示缓冲区*/
uchar code chg_tab[20]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,
                        0x7f,0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71,
                        0x73,0x40,0x80,0x50};
      /*此表为 0—F ，P，—，.，r 二十个字模 */

/*********************************************************************
读 CSI24WC02 函数
原型: void Read_S24();
功能:读取 24WC02 上指定子地址的内容,并在 LED 上显示出来使用前起动总线
      发出读 PCF8574 地址
*********************************************************************/
```

```
void  Read_S24()
{
   uchar    dat,addr;

   while(1)
      {
         IRcvByte(PCF8574,&addr);              /*读 I/O 口的数据作地址*/
         IRcvStr(CSI24WCXX,addr,&dat,1); /*读 CSI24WCXX 指定地址上的数据*/

         Disp_SAA1064(chg_tab[addr>>4],1); /*将地址,数据进行显示*/
         Disp_SAA1064(chg_tab[addr&0x0f],2);
         Disp_SAA1064(chg_tab[dat>>4],3);
         Disp_SAA1064(chg_tab[dat&0x0f],4);
      }
}
```