24 系列 EEPROM 专用模拟 I²C 软件包应用说明

一、概述

24 系列 EEPROM 是一种 I²C 接口的存储器，无须外围元件，操作简单方便，应用广泛。在 24 系列 EEPROM 的操作中，常常会有一些问题，如页写功能、写入时间、16 位存储地址（即软件包中所说的子地址），若用户使用通用的 I²C 软件包，则这些问题用户得自己考虑。为此，我公司开发了一个 24 系列 EEPROM 专用的模拟 I²C 软件包，软件包中充分考虑了以上几个问题，使用对 24 系列 EEPROM 操作更为简便。

二、软件包接口及功能

此软件包是从本公司的模拟 I²C（V1.0）改进过来的，其使用方式与模拟 I²C（V1.0）基本一致，这里就只说明一下不相同的地方，及要注意的地方。

接口子程序如下：

IRDBYTE	（无子地址）读单字节数据	（现行地址读）
IWRBYTE	（无子地址）写单字节数据	（现行地址写）
IRDNBYTE	（有子地址）读 N 字节数据	（随机地址读）
IWRNBYTE	（有子地址）写 N 字节数据	（随要地址写）

说明：现行地址读/写即专指无子地址的器件，不给定子地址的读／写操作。

软件包占用内部资源： R0，R1，R2，R3，ACC，Cy。

使用前须定义变量： SLA 器件从地址，SUBA 器件子地址，NUMBYTE 读/写的字节数 ，位变量 ACK 。

使用前须定义常量： SDA 、SCL 总线位，MTD 发送数据缓冲区首址，MRD 接收数据缓冲区首址。

(※ 接口子程序出口参数 ACK 为 0 时表示从器件无应答)

在使用 IRDNBYTE、IWRNBYTE 时可以使用 16 位子地址，使用 16 位子地址时，要在程序开头定义条件汇编符号 SUBA16，然后还要定义一个变量是 SUBA1，这样 SUBA1、SUBA 就构成了 16 位子址址，其中 SUBA1 为高 8 位。在用户程序开头定义$SET (SUBA16)即可使用 16 位子地址，若没有此定义则只使用 8 位子地址。使用 24WC32、64、128、256 时要这样定义$SET (SUBA16)。

IWRNBYTE 接口程序的功能是指定地址写多字节数据，其中包含了页跳转逻辑功能（是以 8 字节为一页的方式），及页写延时逻辑功能（延时 10mS，若用户系统时钟大于 12MHz，则要进行延时的修改）。

使用接口子程序时要先设置好 SLA、[SUBA]、[SUBA1]、NUMBYTE、[MTD 缓冲区]。

三、软件包原码

```
;-------------------------------------------------------------------------------------------------
;                MCS-51 单片机模拟 I2C 软件包(24 系列 EEPROM 专用)
;     文件名：VI2C_24A.INC
;     功能说明：本模拟 I2C 软件包包含了 I2C 操作的底层子程序，使用前要定义
;好 SCL 和 SDA。在标准 8051 模式(12 Clock)下,对主频要求是不高于 12MHz(即 1 个
;机器周期 1us);若 Fosc>12MHz 则要增加相应的 NOP 指令数。(总线时序符合 I2C 标
```

```
;准模式,100Kbit/S)
;      版本说明：本版本是针对 24 系列 EEPROM 专门设置的，用此软件包进行 EEPR
;OM 读写时，用户不必考虑页写问题，及 16 位存储器地址的问题。
;                                  更新时间：2002.06.05
;-------------------------------------------------------------------------------------------------------

;启动 I2C 总线子程序
START:  SETB    SDA
        NOP
        SETB    SCL             ;起始条件建立时间大于 4.7us
        NOP
        NOP
        NOP
        NOP
        NOP
        CLR     SDA
        NOP                     ;起始条件锁定时大于 4us
        NOP
        NOP
        NOP
        NOP
        CLR     SCL             ;钳住总线，准备发数据
        NOP
        RET

;结束总线子程序
STOP:   CLR     SDA
        NOP
        SETB    SCL             ;发送结束条件的时钟信号
        NOP                     ;结束总线时间大于 4us
        NOP
        NOP
        NOP
        NOP
        SETB    SDA             ;结束总线
        NOP                     ;保证一个终止信号和起始信号的空闲时间大于 4.7us
        NOP
        NOP
        NOP
        RET

;发送应答信号子程序
MACK:   CLR     SDA             ;将 SDA 置 0
        NOP
```

```
        NOP
        SETB    SCL
        NOP                     ;保持数据时间，即 SCL 为高时间大于 4.7us
        NOP
        NOP
        NOP
        NOP
        CLR     SCL
        NOP
        NOP
        RET

;发送非应答信号
MNACK:  SETB    SDA             ;将 SDA 置 1
        NOP
        NOP
        SETB    SCL
        NOP
        NOP                     ;保持数据时间，即 SCL 为高时间大于 4.7us
        NOP
        NOP
        NOP
        CLR     SCL
        NOP
        NOP
        RET

;检查应答位子程序
;返回值，ACK=1 时表示有应答
CACK:   SETB    SDA
        NOP
        NOP
        SETB    SCL
        CLR     ACK
        NOP
        NOP
        MOV     C,SDA
        JC      CEND
        SETB    ACK             ;判断应答位
CEND:   NOP
        CLR     SCL
        NOP
        RET
```

```
;发送字节子程序
;字节数据放入 ACC
;每发送一字节要调用一次 CACK 子程序，取应答位
WRBYTE: MOV     R0,#08H
WLP:    RLC     A               ;取数据位
        JC      WR1
        SJMP    WR0             ;判断数据位
WLP1:   DJNZ    R0,WLP
        NOP
        RET
WR1:    SETB    SDA             ;发送 1
        NOP
        SETB    SCL
        NOP
        NOP
        NOP
        NOP
        NOP
        CLR     SCL
        SJMP    WLP1
WR0:    CLR     SDA             ;发送 0
        NOP
        SETB    SCL
        NOP
        NOP
        NOP
        NOP
        NOP
        CLR     SCL
        SJMP    WLP1

;读取字节子程序
;读出的值在 ACC
;每取一字节要发送一个应答/非应答信号
RDBYTE: MOV     R0,#08H
RLP:    SETB    SDA
        NOP
        SETB    SCL             ;时钟线为高，接收数据位
        NOP
        NOP
        MOV     C,SDA           ;读取数据位
        MOV     A,R2
        CLR     SCL             ;将 SCL 拉低，时间大于 4.7us
        RLC     A               ;进行数据位的处理
```

```
        MOV     R2,A
        NOP
        NOP
        NOP
        DJNZ    R0,RLP              ;未够 8 位，再来一次
        RET



;=========================================================================
;=========================================================================
;       以下是用户接口子程序
;

;无子地址器件写字节数据
;入口参数: 数据为 ACC、器件从地址 SLA
;占用:  A、R0、CY
IWRBYTE: PUSH   ACC
IWBLOOP: LCALL  START               ;起动总线
        MOV     A,SLA
        LCALL   WRBYTE              ;发送器件从地址
        LCALL   CACK
        JNB     ACK,RETWRB          ;无应答则跳转
        POP     ACC                 ;写数据
        LCALL   WRBYTE
        LCALL   CACK
        LCALL   STOP
        RET
RETWRB:  POP    ACC
        LCALL   STOP
        RET

;无子地址器件读字节数据
;入口参数: 器件从地址 SLA
;出口参数: 数据为 ACC
;占用： A 、R0、R2 、CY
IRDBYTE: LCALL  START
        MOV     A,SLA               ;发送器件从地址
        INC     A
        LCALL   WRBYTE
        LCALL   CACK
        JNB     ACK,RETRDB
        LCALL   RDBYTE              ;进行读字节操作
        LCALL   MNACK               ;发送非应信号
```

```
RETRDB: LCALL   STOP            ;结束总线
        RET

;向器件指定子地址写 N 字节数据(24 系列 EEPROM 专用)
;入口参数:  器件从地址 SLA、器件子地址 SUBA 、发送数据缓冲区 MTD、发送字节数 NUMBYTE
;占用：A 、R0 、R1 、R2、R3 、CY
;说明：以 8 字节为 1 页的方式计算，防止页写的翻转问题。即你写入数据是不用考虑页写入问
;       题，即本子程序会帮你转页。本子程序是以 8 字节为一页进行换页，调用此子程序会
;       更改 SUBA 和[SUBA1]的值。
IWRNBYTE: MOV   A,NUMBYTE
        MOV     R3,A
        LCALL   START           ;起动总线
        MOV     A,SLA
        LCALL   WRBYTE          ;发送器件从地址
        LCALL   CACK
        JNB     ACK,RETWRN      ;无应答则退出
$IF(SUBA16)                     ;若是使用 16 位存储地址，则发送高 8 位地址
        MOV     A,SUBA1         ;如 24WC32、64、128、256 就是 16 位存储地址
        LCALL   WRBYTE
        LCALL   CACK
$ENDIF
        MOV     A,SUBA          ;指定子地址
        LCALL   WRBYTE
        LCALL   CACK

;开始发送数据
        MOV     R1,#MTD
WRDA:   MOV     A,@R1
        LCALL   WRBYTE          ;开始写入数据
        LCALL   CACK
        JNB     ACK,IWRNBYTE

;发送完 1 字节后，马上判断是否页满，若是则结束总线，并等待写周期结束。
;然后重新启动总线，再次发送从地址，子地址，写数据。

        INC     SUBA            ;子地址更新
$IF(SUBA16)
        MOV     A,SUBA
        JNZ     WRDA_L1
        INC     SUBA1           ;若是 16 位子地址，且 SUBA 有进位，则要对 SUBA1 进行更改
$ENDIF

WRDA_L1:MOV     A,SUBA
        ANL     A,#0FH
```

```
            JZ      WRDA_L3         ;若已经到页首，重新进行页定位
            CJNE    A,#08H,WRDA_L4  ;若 SUBA=08H(页首),重新进行页定位
WRDA_L3:LCALL   STOP
            MOV     R2,#0           ;等待页写完成，延时 10mS
DELAY10: NOP
            ACALL   DELAY40NOP
            NOP
            DJNZ    R2,DELAY10
            ;页写完成，重新启动总线
            LCALL   START           ;起动总线
            MOV     A,SLA
            LCALL   WRBYTE          ;发送器件从地址
            LCALL   CACK
            JNB     ACK,RETWRN      ;无应答则退出
$IF(SUBA16)                         若是使用 16 位存储地址，则发送高 8 位地址
             MOV    A,SUBA1         ;如 24WC32、64、128、256 就是 16 位存储地址
             LCALL  WRBYTE
             LCALL  CACK
$ENDIF
            MOV     A,SUBA          ;指定子地址
            LCALL   WRBYTE
            LCALL   CACK

WRDA_L4:INC     R1
            DJNZ    R3,WRDA         ;判断写完没有
RETWRN: LCALL   STOP
            RET

;延时 40uS
;NOP 的机器码为 00H
DELAY40NOP:
             DB  00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
             DB  00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
             DB  00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
             DB  00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
              RET

;向器件指定子地址读取 N 字节数据(24 系列 EEPROM 专用)
;入口参数:   器件从地址 SLA、器件子地址 SUBA、接收字节数 NUMBYTE
;出口参数:   接收数据缓冲区 MTD
;占用：A、 R0、 R1、 R2、 R3、 CY
IRDNBYTE:MOV    R3,NUMBYTE
            LCALL   START
            MOV     A,SLA
```

```
            LCALL   WRBYTE          ;发送器件从地址
            LCALL   CACK
            JNB     ACK,RETRDN
$IF(SUBA16)                         若是使用 16 位存储地址，则发送高 8 位地址
            MOV     A,SUBA1         ;如 24WC32、64、128、256 就是 16 位存储地址
            LCALL   WRBYTE
            LCALL   CACK
$ENDIF
            MOV     A,SUBA          ;指定子地址
            LCALL   WRBYTE
            LCALL   CACK
            LCALL   START           ;重新起动总线
            MOV     A,SLA
            INC     A               ;准备进行读操作
            LCALL   WRBYTE
            LCALL   CACK
            JNB     ACK,IRDNBYTE
            MOV     R1,#MRD
RDN1:       LCALL   RDBYTE          ;读操作开始
            MOV     @R1,A
            DJNZ    R3,SACK
            LCALL   MNACK           ;最后一字节发非应答位
RETRDN:     LCALL   STOP            ;并结束总线
            RET
SACK:       LCALL   MACK
            INC     R1
            SJMP    RDN1

;*********************************************************************************************
;                                   请注意
;
;     占用内部资源：          R0，R1，R2，R3，ACC，Cy。
;     在你的程序里要做以下定义：
;     1、定义变量： SLA 器件从地址  SUBA 器件子地址  [SUBA1 器件子地址低 8 位]  NUMBYTE
;读／写的字节数 ，位变量 ACK
;     2、定义常量： SDA SCL 总线位  MTD 发送数据缓冲区首址    MRD 接收数据缓冲区首址
;  （ACK 为调试／测试位,ACK 为 0 时表示无器件应答或总线出错）
;
;     在使用本软件包时，请在你的程序的未尾加入$INCLUDE (VI2C_24A.INC)即可。VI2C_24A.INC 文件
;复制到 IDE 包含文件所要求的目录(如：工作目录或 INC 目录)
;     若没有定义 SUBA16，则软件包是使用 8 位节子地址 SUBA，若你定义了 SUBA16 则是使用 16 位的
;子地址，还要定义变量 SUBA1，即使用 24WC32、64、128、256 时要这样定义$SET (SUBA16)。
;     用户可以对"用户接口子程序"进行裁减，即把自已不使用的接口子程序删掉,以节省程序空间。
;*********************************************************************************************
```

四、应用举例

例 1：向 24WC02 写入 20 字节数据

```
;为软件包定义变量
ACK         BIT     10H         ;应答标志位变量
SLA         DATA    50H         ;器件从地址变量
SUBA        DATA    51H         ;器件子地址变量
NUMBYTE     DATA    52H         ;读/写的字节数变量
;使用前定义常量
SDA         EQU     P1.3        ; I2C 总线接口定义
SCL         EQU     P1.2
MTD         EQU     30H         ;发送数据缓冲区首址 (缓冲区 30H－ 49H)
MRD         EQU     4AH         ;接收数据缓冲区首址 (缓冲区 4AH－ 4FH)
;定义器件地址
CSI24WCXX   EQU     0A0H

            ORG     0000H
            AJMP    MAIN

            ORG     0080H
MAIN:       MOV     R4,#0F0H    ;延时，等待其它芯片复位好
            DJNZ    R4,$

            MOV     R7,#20
            MOV     R0,#MTD
            MOV     A,#0FH
LOADDAT:    MOV     @R0,A       ;装载数据
            INC     R0
            DJNZ    R7,LOADDAT

            MOV     SLA,# CSI24WCXX
            MOV     SUBA,#00
            MOV     NUMBYTE,#20
            LCALL   IWRNBYTE
            SJMP    $
    $INCLUDE (VI2C_24A.INC)     ;包含 I2C 软件包
    END
```

例 2：向 24WC256 读入 10 字节数据

```
$SET (SUBA16)                   ;使用 16 位子地址
;为软件包定义变量
```

```
ACK         BIT     10H         ;应答标志位变量
SLA         DATA    50H         ;器件从地址变量
SUBA        DATA    51H         ;器件子地址变量（高 8 位）
NUMBYTE     DATA    52H         ;读/写的字节数变量
SUBA1       DATA    53H         ;定义高 8 位子地址变量
;使用前定义常量
SDA         EQU     P1.3        ; I2C 总线接口定义
SCL         EQU     P1.2
MTD         EQU     30H         ;发送数据缓冲区首址 (缓冲区 30H－3FH)
MRD         EQU     40H         ;接收数据缓冲区首址 (缓冲区 40H－4FH)
;定义器件地址
CSI24WCXX   EQU     0A0H

            ORG     0000H
            AJMP    MAIN

            ORG     0080H
MAIN:       MOV     R4,#0F0H    ;延时，等待其它芯片复位好
            DJNZ    R4,$

            MOV     SLA,# CSI24WCXX
            MOV     SUBA,#00
            MOV     SUBA1,#00
            MOV     NUMBYTE,#20
            MOV     SUBA,#00
            MOV     SUBA1,#00
            LCALL   IRDNBYTE
            SJMP    $
    $INCLUDE (VI2C_24A.INC)     ;包含 I2C 软件包
    END
```