

HT48 MCU 读写 HT24 系列 EEPROM 的应用范例

文件编码: HA0016s

简介:

HT24 系列的 EEPROM 是通过 I²C 协议控制其读写的。HT48 系列单片机的接口部分是 CMOS I/O 口, 可以用来很方便地采用 I²C 协议控制周边器件。

HT24 系列的 EEPROM 总共 8 个管脚, 三个为芯片地址脚 A0、A1、A2, 在单片机对它进行操作时, 从 SDA 输入 A0、A1、A2 数据和芯片外部 A0、A1、A2 所接地址需一一对应。一个为芯片写保护脚 WP, WP 脚接低电平时, 芯片可进行读写操作; WP 脚接高时, 芯片只可进行读, 不可进行写。另外两个管脚为电源脚 VCC, VSS。

用单片机对 HT24 系列的 EEPROM 进行控制时, HT24 系列的 EEPROM 的外部管脚 VCC、VSS、WP、A0、A1、A2 根据需要, 对应接上, SDA、SCL 接到单片机控制脚上。

引脚名称	I/O	功能描述
A0~A2	I	地址输入
VSS	I	电源负极输入
SDA	I/O	串行数据输入/输出
SCL	I	串行数据传送时钟信号输入
WP	I	写保护
VCC	I	电源正极输入

HT24 系列的 EEPROM 根据型号不同, EEPROM 的容量大小不同, 当 EEPROM 的空间大于 1 页 (256bytes) 时, 即大于 2048bits, 则 HT48 MCU 需要控制需要控制 A0、A1、A2 来确定写 HT24 系列的 EEPROM 的第几页, HT24 系列的 EEPROM 空间大小如下表所示:

型号	引脚 A0、A1 及 A2 使用方法	容量大小
HT24LC02	A0、A1、A2 引脚作为器件地址输入, 从 SDA 输入 A0、A1、A2 数据和芯片引脚 A0、A1、A2 所接状态需一一对应	2K (256×8)
HT24LC04	A1、A2 引脚作为器件地址输入, 从 SDA 输入 A1、A2 数据和芯片引脚 A1、A2 所接状态需一一对应, A0 引脚浮空	4K (512×8, 2pages)
HT24LC08	A2 引脚器件地址输入, 从 SDA 输入 A2 数据和芯片引脚 A2 所接状态需一一对应, 其余引脚浮空	8K (1024×8, 4pages)
HT24LC16	A0、A1、A2 全部浮空, 不必接	16K (2048×8, 8pages)

使用说明:

本文是以 HT48R30A-1 控制 HT24LC04 为例的。I²C 协议只需要两根线进行控制，一根时钟线 SCL，一根数据线 SDA。用单片机对 HT24LC04 进行控制时，HT24LC04 外部管脚 VCC、VSS、WP、A1、A2 根据需要，对应接上（本例中是与 VSS 相接，即 A1、A2=00），SDA、SCL 接到单片机控制脚上。在这个例程中 SCL 接到 pa.3 脚，SDA 接到 pa.1 脚。程序的过程是这样的：先向 EEPROM 中某个地址写 55H，写完后，再将 EEPROM 中内容读出来，并将读出数据和 55H 进行比较，若数据不相等程序跳到 fail_out 中；若相等，最后程序跳到 ok_end 中。

例程中只是对第 0 页中的特定地址进行写操作后，再将该地址的内容读出。关于 HT24 系列的 EEPROM 其它操作只要严格按其时序要求，并注意从 SDA 输入 A0、A1、A2 数据和芯片引脚 A0、A1、A2 所接状态需一一对应即可。

例程:

```

;-----
;mask option:
;WDT: disabled
;WDTinstr:one clear instruction
;PA wake up:noen
;pullhigh:all
;WDT OSC :on chip RC
;OSC:crystal
;sysvolt:5.000V
;sysfreq:1000kHz,internal
;product:24SKDIP_B
include ht48r30a-1.inc
; file name: 4810wr.asm
; 作者: 盛扬半导体(上海)有限公司软件部
; 目的: 熟悉 HT48 系列控制 HT24 系列的 EEPROM 单片机的流程

eeprom .section 'data'
    scl          equ    pa.3          ;定义 pa.3 为时钟脚, pa3 接 SCL
    scl_c        equ    pac.3
    sda          equ    pa.1          ;定义 pa.1 为数据脚, pa1 接 SDA
    sda_c        equ    pac.1
    read_out     equ    [70h]        ;读出数据暂存器
    write_in     equ    [71h]        ;写入数据暂存器
    word_address equ    [72h]        ;读写地址暂存器
    data_8       equ    [73h]
    delay_5      equ    [77h]
    delay        equ    [78h]

eepromc .section 'code'
    org    00h

```

```

    jmp    start
    org    020h
start:
    mov    a,55h
    mov    write_in,a           ;写入 55H
    mov    a,14h               ;写入 14H 为要操作的 EEPROM 的地址
    mov    word_address,a
write_data:
    clr    sda_c
    clr    scl_c

    set    sda
    set    scl
    clr    sda                 ;起始信号
    clr    scl

    set    sda                 ;1
    set    scl
    clr    scl

    clr    sda                 ;0
    set    scl
    clr    scl

    set    sda                 ;1
    set    scl
    clr    scl

    clr    sda                 ;0
    set    scl
    clr    scl

    clr    sda                 ;A2,A1,A0=000
    set    scl
    clr    scl

    set    scl
    clr    scl

    set    scl
    clr    scl

    set    scl                 ;写 0, 设定为写模式
    clr    scl

```

```

    set    sda_c
    set    scl
wait_ack:
    sz     sda                ;等待应答信号
    jmp    wait_ack
    clr    scl
    clr    sda_c
    mov    a,08h              ;设传输数据长度 8
    mov    data_8,a
random_write:
    clr    sda
    sz     word_address.7
    set    sda
    set    scl
    clr    scl

    rl     word_address
    sdz    data_8
    jmp    random_write
    set    sda_c
    set    scl
fdev:
    sz     sda                ;等待应答信号
    jmp    fdev
    clr    scl
    clr    sda_c
    mov    a,08h
    mov    data_8,a
dtat_in:
    clr    sda
    sz     write_in.7
    set    sda
    set    scl
    clr    scl
    rl     write_in
    sdz    data_8
    jmp    dtat_in

    set    scl
    clr    scl

    clr    sda
    set    scl

```

```

set    sda                                ;停止信号

mov    a, 30h
mov    delay_5, a
mov    a, 05h
mov    delay, a
delay1:
    sdz    delay_5
    jmp    delay1
    sdz    delay
    jmp    delay1
;
;读
do_read:
    clr    read_out
    clr    sda_c
    clr    scl_c

    set    sda
    set    scl
    clr    sda                                ;起始信号

    clr    scl
    set    sda                                ;1
    set    scl
    clr    scl

    clr    sda                                ;0
    set    scl
    clr    scl

    set    sda                                ;1
    set    scl
    clr    scl

    clr    sda                                ;0
    set    scl
    clr    scl

    clr    sda
    set    scl                                ;A0,A1,A2=0
    clr    scl

    set    scl                                ;0

```

```

clr    scl

set    scl                ;0
clr scl

clr    sda                ;写模式，写地址
set    scl
clr    scl

set    sda_c
set    scl

wait:
sz     sda
jmp    wait
clr    scl
mov    a,08h
mov    data_8,a
clr    sda_c
read_address_in:
clr    sda
sz     word_address.7
set    sda
set    scl

clr    scl
rl     word_address
sdz   data_8
jmp    read_address_in

set    sda_c
set    scl

ack:
sz     sda
jmp    ack
clr    scl
clr    sda_c

read_data:
set    sda
set    scl
clr    sda                ;start bit
clr    scl

```

```

set    sda                ;1
set    scl
clr    scl

clr    sda                ;0
set    scl
clr    scl

set    sda                ;1
set    scl
clr    scl

clr    sda                ;0
set    scl
clr    scl

clr    sda                ;A2,A1,A0
set    scl
clr    scl

set    scl
clr    scl

set    scl
clr    scl

set    sda                ;读模式
set    scl
clr    scl

set    sda_c
w_ack:
sz     sda
jmp    w_ack

set    scl                ;down_edge data out
mov    a,08h
mov    data_8,a
set    sda_c
random_out:
set    scl
call   del
clr    scl

```

```

call    del
rl      read_out
clr     read_out.0
sz      sda
set     read_out.0
sdz     data_8
jmp     random_out
mov     a,read_out
mov     [41h],a
clr     sda_c                ;for stop
set     scl
set     sda
clr     scl
clr     sda
set     sda                ;stop end

mov     a,055h
xor     a,read_out
sz      acc
jmp     fail_out
jmp     ok_end

fail_out:
jmp     $

ok_end:
jmp     $

del:                ;for delay
nop
nop
nop
ret

```

写此程序的注意点:

- 1、需要注意 HT24 系列 EEPROM 中 A0 的用法。在 HT24LC04 中 A0 是作为地址位用的。
- 2、读出数据时是在下降沿，写入数据是在上升沿。
- 3、注意：读取操作的时钟频率不应该太高。在演示中系统时钟频率为 1MHz，如果系统时钟频率比较高的话，则应该加上延时操作，否则读取数据会出现错误。

本例适用于读写 HT24 系列的 EEPROM 单片机第 0 页的特定地址操作，若要对其它页进行操作，改变相应的 A0、A1、A2 数据，从 SDA 输入即可。