

## HT49 MCU 控制 HT93LC46 的读写

文件编码：HA0044s

### 介绍：

HT93LC46EEPROM 是 Holtek 制造的 1K 位系列的 EEPROM (电子可擦除只读存储器), 一般它用于微控制器的固定数据的存储。在本文中, 我们将以 Holtek 公司 8 位微控制器 HT49 系列为例, 介绍该芯片常用的操作功能代码。用户只需把代码加到程序中, 并且在使用 HT93LC46 之前将引脚 CS/SK/DI/DO 连接即可。

### 功能实现：

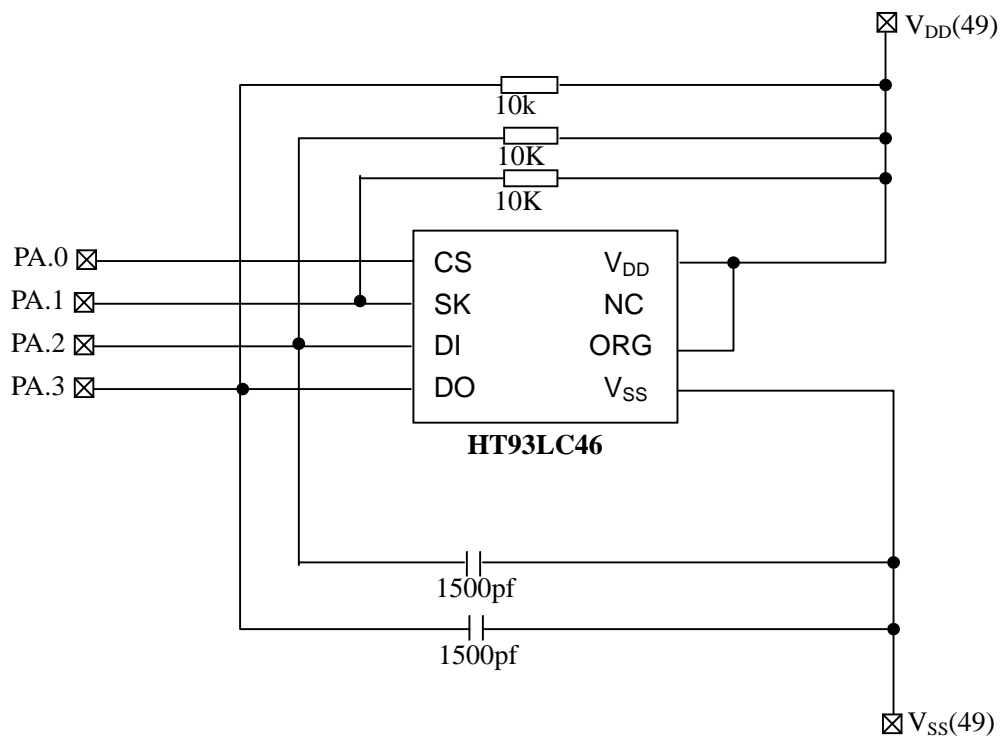
在本文中将会用到一插入文件 HT93LC46S.ASM 和一汇编源文件 OP16\_93LC46S.ASM。在 HT - IDE3000 开发环境下, 在使用所提供的接口函数前按下述的操作步骤做：

- 步骤 1：把 OP16\_93LC46S.ASM 加到项目下 (用[Project/Edit]指令)
- 步骤 2：根据你的电路, 修改 HT93LC46S.INC 文件连接 CS/SK/DI/DO 引脚
- 步骤 3：包括 HT93LC46.ASM 在内的源文件调用接口函数

注意：在使用这些函数前要适当地设置输入/输出口的模式

### 使用说明：

#### 1. 硬件线路图



## 2. 掩模选择

PA0 ~ PA3 : NMOS

PA0 ~ PA3 : PULL HIGH

### 例程：

生成一个目标控制器为 HT49R30A-1 的项目 (Project), 然后加入 OP16\_93LC46S.ASM 到这个项目中。

在这个例子中, 首先使用 HT93LC46\_EWEN 函数使 HT93LC46 写置能; 然后, 使用 HT93LC46\_WRITE 函数向存储器写入相应的地址; 最后, 使用 HT93LC46\_READ 指令将刚才写入的数据读出并检查写入的数据是否正确。

程序如下：

```

;=====
; 文件名: op16_93lc46s.asm
; 日期: 2003/11/26
; MCU: HT49R30A-1
; EEPROM: HT93LC46, 64x16bits
;=====

includeht49r30a-1.inc
includeht93lc46s.asm

data    .section    'data'
temp    db    ?

main    .section    at 0    'code'
start:
    clr    pa
    clr    porta
    set    porta.3
    mov    a,porta
    mov    pa,a

;-----
    call    ht93lc46_ewen    ;ht93lc46写置能
    nop
    mov    a,3fh            ;ORG脚接VCC, 64x16
    mov    temp,a

write:
    mov    a, temp
    mov    dataaddr, a
    mov    data2, a
    cpl    acc
    mov    data3, a
    call    ht93lc46_write    ;ht93lc46写

```

```

dec    temp
snz    temp.7
jmp    write
call   ht93lc46_ewds    ;ht93lc46写除能
;-----
; 开始读数据并比较
mov    a,3fh            ;ORG脚接VCC , 64x16
mov    temp,a
read:
clr    data2
clr    data3
mov    a, temp
mov    dataaddr, a
call   ht93lc46_read    ;ht93lc46读

mov    a, temp          ;比较数据
xor    a, data2
snz    z
jmp    fail
cpla   temp
xor    a, data3
snz    z
jmp    fail
dec    temp
snz    temp.7
jmp    read
;-----
call   ht93lc46_ewen    ;ht93lc46写置能
nop
call   ht93lc46_eral    ;ht93lc46擦除全部空间
nop
mov    a,3fh            ;ORG脚接VCC , 64x16
mov    temp,a
read_1:
clr    data3
clr    data2
mov    a, temp
mov    dataaddr, a
call   ht93lc46_read    ;ht93lc46读

mov    a, 0ffh          ;比较数据
xor    a, data2
snz    z

```

```

    jmp    fail
    mov    a, 0ffh
    xor    a, data3
    snz    z
    jmp    fail
    dec    temp
    snz    temp.7
    jmp    read_1
;-----
    mov    a, 04bh
    mov    data2, a
    mov    a, 0c3h
    mov    data3, a
    call   ht93lc46_wral    ;ht93lc46写全部空间
    nop
    mov    a, 3fh          ;ORG脚接VCC , 64x16
    mov    temp, a
read_2:
    clr    data2
    clr    data3
    mov    a, temp
    mov    dataaddr, a
    call   ht93lc46_read   ;ht93lc46读

    mov    a, 04bh        ;比较数据
    xor    a, data2
    snz    z
    jmp    fail
    mov    a, 0c3h
    xor    a, data3
    snz    z
    jmp    fail
    dec    temp
    snz    temp.7
    jmp    read_2
;-----
    mov    a, 3fh          ;ORG脚接VCC , 64x16
    mov    temp, a
write_1:
    mov    a, temp
    mov    dataaddr, a
    call   ht93lc46_erase  ;ht93lc46擦除指定地址
    dec    temp

```

```

    snz    temp.7
    jmp    write_1
    nop
    mov    a,3fh                ;ORG脚接VCC , 64x16
    mov    temp,a
read_3:
    clr    data3
    clr    data2
    mov    a, temp
    mov    dataaddr, a
    call   ht93lc46_read        ;ht93lc46读

    mov    a, 0ffh              ;比较数据
    xor    a, data2
    snz    z
    jmp    fail
    mov    a, 0ffh
    xor    a, data3
    snz    z
    jmp    fail
    dec    temp
    snz    temp.7
    jmp    read_3

    jmp    $                      ;操作成功
fail:
    jmp    $                      ;操作失败
;-----
;程序op16_93lc46s.asm结束
;-----

```

以下为附件1 (HT93LC46S.ASM) , 源文件调用接口函数, 包含了HT93LC46的七条指令。

```

;=====
; 文件名:ht93lc46.ASM
; 日期:2003/11/26
; ROM使用情况:C8H
; RAM使用情况:08H
;=====
#defineht93lc46_asm

;操作码
oc_read    equ    10000000b
oc_erase   equ    11000000b

```

```

oc_write    equ    01000000b
oc_ewen     equ    00110000b
oc_ewds     equ    00000000b
oc_eral     equ    00100000b
oc_wral     equ    00010000b

;=====
;如果ORG脚接VCC, 则屏蔽bit8, 64x16
;如果ORG脚接VSS, 则屏蔽bit16, 128x8
;=====
#define bit16
;#define bit8

;=====
;可根据你的电路修改各引脚CS/SK/DI/DO的定义
;=====
porta       equ    [70h]
sk          equ    porta.1
di          equ    porta.2
do          equ    porta.3
cs          equ    porta.0
;pa        equ    [12h]
doin       equ    pa.3

;=====
;以下部分不可修改
;=====
ifndef     ht93lc46_asm
extern     dataaddr    :byte
extern     data3       :byte
extern     data2       :byte
extern     ht93lc46_write:near
extern     ht93lc46_read :near
extern     ht93lc46_ewen :near
extern     ht93lc46_ewds :near
extern     ht93lc46_eral :near
extern     ht93lc46_wral :near
extern     ht93lc46_erase:near
endif

;=====
;宣告提供给外部程序使用的变量
public     dataaddr    ;数据的页内地址

```

```

public    data3                ;存取的数据的高八位
public    data2                ;存取的数据的低八位
;宣告提供给外部程序使用的子程序
public    ht93lc46_eral       ;
public    ht93lc46_wral       ;
public    ht93lc46_ewen       ;
public    ht93lc46_ewds       ;
public    ht93lc46_write      ;
public    ht93lc46_read       ;
public    ht93lc46_erase      ;
public    delay

;=====
;数据段
ht93lc46data .section 'data'
dataaddr    db ? ;操作地址
data3       db ? ;操作数据寄存器,高八位
data2       db ? ;操作数据寄存器
data1       db ? ;进行移位的数据寄存器
movb        db ? ;循环移位次数寄存器
reg         db ? ;时延数据寄存器
reg1        db ? ;时延数据寄存器
;=====

;=====
;代码段
ht93lc46code .section 'code'

;=====
; READ--读取数据
; 描述:从EEPROM的指定地址处读取数据
; 入口参数 dataaddr:byte 指定地址
; 出口参数: data2 :byte 读取的数据的低八位
;          data3 :byte 读取的数据的高八位
; 堆栈使用: 1
;=====
ht93lc46_read    proc
    call    ht93_start    ;开始信号
    mov     a, oc_read
    mov     data1, a
    mov     a,2           ;写入2位op-code代码
    call    wbit
    mov     a, dataaddr

```

```

mov    data1, a
rl     data1
ifdef  bit8
    mov    a,7           ;写入7位dataaddr
endif
ifdef  bit16
    rl     data1
    mov    a,6           ;写入6位dataaddr
endif
call   wbit
nop
call   rbit
ifdef  bit16
    mov    a, data2
    mov    data3, a
    call   rbit
endif
clr    cs
call   porta2pa
ret

ht93lc46_read    endp
;=====
; WRITE--写入数据
; 描述：往EEPROM的指定地址处写入数据
; 入口参数：dataaddr:byte 指定地址
;          data2 :byte 写入的数据的低八位
;          data3 :byte 写入的数据的高八位
; 出口参数：    无
; 堆栈使用：1
;=====
ht93lc46_write   proc
    call   ht93_start
    mov    a, oc_write
    mov    data1, a
    mov    a,2           ;写入2位op-code代码
    call   wbit
    mov    a, dataaddr
    mov    data1, a
    rl     data1
    ifdef  bit8
        mov    a,7           ;写入7位dataaddr
    endif
    ifdef  bit16

```

```

        rl      data1
        mov     a,6           ;写入6位dataaddr
    endif
    call     wbit
    nop
    ifdef    bit16
        mov     a, data3
        mov     data1, a
        mov     a,8
        call    wbit
    endif
    mov     a, data2
    mov     data1, a
    mov     a,8
    call    wbit
    clr     cs
    call    porta2pa
    call    delay
    call    mverify
    clr     cs
    call    porta2pa
    ret
ht93lc46_write     endp
;=====
; ERASE--擦除数据
; 描述：往EEPROM指定地址处写入数据1
; 入口参数：dataaddr:byte 指定地址
; 出口参数：无
; 堆栈使用：1
;=====
ht93lc46_erase     proc
    call     ht93_start
    mov     a, oc_erase
    mov     data1, a
    mov     a,2           ;写入2位op-code代码
    call    wbit
    mov     a, dataaddr
    mov     data1, a
    rl      data1
    ifdef    bit8
        mov     a,7           ;写入7位dataaddr
    endif
    ifdef    bit16

```

```

        rl      data1
        mov     a,6           ;写入6位dataaddr
    endif
    call    wbit
    clr     cs
    call    porta2pa
    call    delay
    call    mverify
    clr     cs
    call    porta2pa
    ret
ht93lc46_erase    endp
;=====
; EWDS--写除能
; 描述：写除能，使EEPROM不能进行写入的操作
; 入口参数：无
; 出口参数：无
; 堆栈使用：1
;=====
ht93lc46_ewds    proc
    call    ht93_start
    mov     a, oc_ewds
    mov     data1, a
    mov     a,8           ;写入8位op-code代码
    call    wbit
    ifdef  bit8
        clr     sk
        call    porta2pa
        set     sk
        call    porta2pa
        nop
    endif
    clr     sk
    call    porta2pa
    clr     cs
    call    porta2pa
    ret
ht93lc46_ewds    endp
;=====
; EWEN--写置能
; 描述：写置能，使EEPROM能进行写入的操作
; 入口参数：无
; 出口参数：无

```

```

; 堆栈使用：1
;=====
ht93lc46_ewen    proc
    call    ht93_start
    mov     a, oc_ewen
    mov     data1, a
    mov     a,8           ;写入8位op-code代码
    call    wbit
    ifdef  bit8
        clr     sk
        call    porta2pa
        set     sk
        call    porta2pa
        nop
    endif
    clr     sk
    call    porta2pa
    clr     cs
    call    porta2pa
    ret
ht93lc46_ewen    endp
;=====
; ERAL--擦除所有地址
; 描述：将EEPROM中所有地址处写入数据1
; 入口参数：无
; 出口参数：无
; 堆栈使用：1
;=====
ht93lc46_eral    proc
    call    ht93_start
    mov     a, oc_eral
    mov     data1, a
    mov     a,8           ;写入8位op-code代码
    call    wbit
    ifdef  bit8
        clr     sk
        call    porta2pa
        set     sk
        call    porta2pa
        nop
    endif
    clr     sk
    call    porta2pa

```

```

    clr    cs
    call   porta2pa
    call   delay
    call   mverify
    clr    cs
    ret
ht93lc46_eral    endp
;=====
; WRAL--写所有地址
; 描述：将数据写入所有EEPROM空间
; 入口参数：data2  :byte 写入的数据的低八位
;           data3  :byte 写入的数据的高八位
; 出口参数：无
; 堆栈使用：1
;=====
ht93lc46_wral    proc
    call   ht93_start
    mov    a, oc_wral
    mov    data1, a
    mov    a,8                ;写入8位op-code代码
    call   wbit
    ifdef bit8
        clr    sk
        call   porta2pa
        set    sk
        call   porta2pa
        nop
    endif
    clr    sk
    call   porta2pa
    ifdef bit16
        mov    a, data3
        mov    data1, a
        mov    a,8
        call   wbit
    endif
    mov    a, data2
    mov    data1, a
    mov    a,8
    call   wbit
    clr    cs
    call   porta2pa
    call   delay

```

```

    call    mverify
    clr     cs
    call    porta2pa
    ret

ht93lc46_wral    endp
;=====
; 开始信号
;=====
ht93_start      proc
    set     cs
    call    porta2pa
    clr     sk
    call    porta2pa
    set     di
    call    porta2pa
    nop
    nop
    set     sk
    call    porta2pa
    nop
    clr     sk
    call    porta2pa
    ret

ht93_start      endp
;=====
; 写n bit数据子程序, n由acc决定
;=====
wbit            proc
    mov     movb, a
loop1:
    clr     sk
    call    porta2pa
    rl     data1
    snz    data1.0
    jmp     loop1_1
    set     di
    call    porta2pa
    jmp     loop1_2
loop1_1:
    clr     di
    call    porta2pa
loop1_2:
    nop

```

```

set      sk
call     porta2pa
nop
sdz      movb
jmp      loop1
clr      sk
call     porta2pa
ret
wbit          endp

;=====
;读8 bit数据子程序
;=====

rbit          proc
    mov     a, 08h
    mov     movb, a
loop_r:
    rl      data2
    set     do                ;设为输入,接收数据
    call    porta2pa
    set     sk
    call    porta2pa
    nop
    snz    doin
    jmp     loops_0
    set    data2.0
    jmp     loops_1
loops_0:
    clr    data2.0
loops_1:
    clr    sk
    call   porta2pa
    sdz    movb
    jmp    loop_r
    ret
rbit          endp

;=====
;检测DO是否有HIGH信号,即操作是否完成
;=====

mverify      proc
    set     cs
    call    porta2pa
    nop

```

```

    nop
check:
    set    do                ;设为输入,接收应答信号(busy or ready)
    call  porta2pa
    snz   doin
    jmp   check
    ret
mverify      endp
;=====
delay        proc
    set    reg1
    mov    a, 06h
    mov    reg, a
lpy:
    sdz   reg1
    jmp   lpy
    sdz   reg
    jmp   lpy
    ret
delay        endp
;=====
porta2pa     proc
    mov    a, porta
    mov    pa, a
    jmp   $+1        ;49I/O口需要一定的时间延迟
    jmp   $+1
    jmp   $+1
    jmp   $+1
    ret
porta2pa     endp
;=====
;程序 ht93lc46s.ASM结束
;=====

```

写此程序时，需注意以下事项：

1. 因为49系列为简单I/O口，没有I/O的控制寄存器，所以较复杂I/O口更易出现读-改-写现象，所以以上例程使用了一映射寄存器，并对I/O口的操作使用MOV指令；
2. 在每次对I/O口进行SET指令之后，一定要加一段时间的延迟；本例程是在每次I/O口状态修改之后加了一段时间的延迟；
3. 前面所提供的参考电路图中的电阻以及电容值依不同PCB板会有所不同。