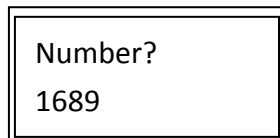
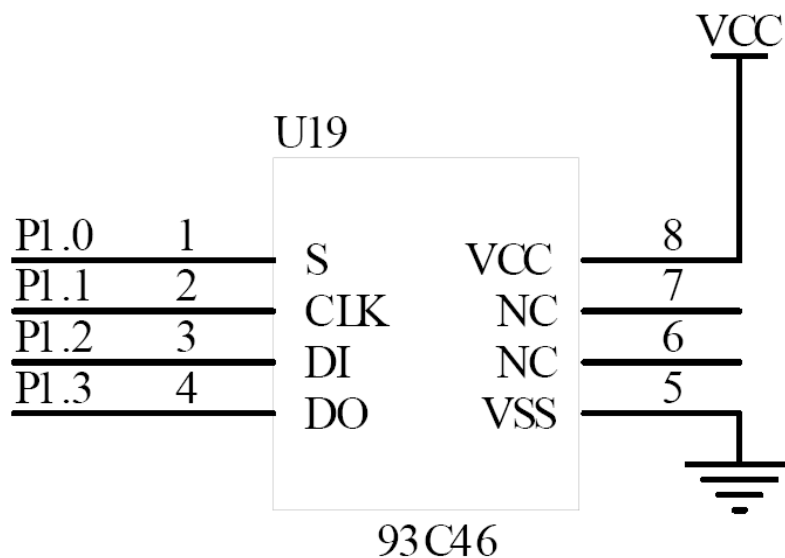


A. 題目

被指定到本題者必須在LCM 進階選單的選項3上呈現出”MEM”等文字，在週邊板上有一個編號U16 的MicroWire 式串列記憶體93C46，應試者在參考電路圖後必須設計一個可以將4 位數字儲存到93C46 的程式，當藉由矩陣鍵盤輸入四個數字的過程中”LCM”也要同時顯示該組數字。當系統在進階選單狀態時，若操作者按下數字鍵”3”進入資料寫入功能，此時 LCM 畫面第一列顯示”Number?”等文字，而第二列呈現空白狀態等待命令或數字資料的輸入，藉由矩陣鍵盤依序輸入四個0-9 的數字時，這四個數字必須同時顯示在LCM 畫面的第二列上，一旦連續輸入四個數字完畢後，該組數字就自動存入編號U16的串列記憶體裡，若按下” Mu” 鍵則返回主功能表畫面，實際操作參考畫面如如下所示。



再者，關閉電路板電源再重新打開電源，已經寫入儲存在串列記憶體中的四個數字必須依然存放在93C46 中，此時依序輸入”Mu”鍵再按下數字鍵”3”進入MEM 功能選項狀態，直接按下” En” 鍵則代表要讀出93C46 的四個數字資料，此時LCM 畫面上應該立即顯示剛剛寫入記憶體中的四個數字資料；至於資料寫入記憶體的位址可由應試者自行決定，但寫入與讀出的位址必須相同。在日後的應用上可作為進入程式操作的密碼保護系統，或者當週邊板要作為智慧型自走車使用時，用來儲存自走車行走的步數、速度及方向等資料。



參考電路圖

B. 參考程式：(原始檔名 9346.a)

```
;serial memory 93c46
    org 00h
;-----register define-----
s9346_cs    equ p1.0
s9346_clk   equ p1.1
s9346_di    equ p1.2
s9346_do    equ p1.3
;-----
temp_9346   equ 0x75
count_9346  equ 0x74
data_hb     equ 0x73    ;data register
data_lb     equ 0x72    ;data register
adrs_9346   equ 0x20    ;single bit define
```

接腳和暫存器宣告

主程式從這裡開始

這裡是 93C46 初始狀態，讀寫都需要

```
mov p2,#0xff    ;if p2=>8 leds(lb)
mov p3,#0xff    ;if p3=>8 leds(hb)
;
clr s9346_cs    ;initial status
clr s9346_clk   ;initial status
clr s9346_di    ;initial status
setb s9346_do   ;;initial status(notice)
```

```
mov adrs_9346,#0x00    ;write 93c46@00h
mov data_hb,#0x0a      ;if data high byte=0ah
mov data_lb,#0x55      ;if data high byte=55h
call write_data
;
mov data_hb,#0x00      ;clear data_hb
mov data_lb,#0x00      ;claer data_lb
```

雖然 93C46 一次可寫入 2 個 byte，但在此只用到 data_hb 一個 byte，程式比較好寫，data_lb 就先忽略不管。

```
mov adrs_9346,#0x00    ;read 93c46@00h
call read_data
mov p3,data_hb
mov p2,data_lb
```

主程式到此為止

```
end_9346:    jmp end_9346
```

```
;+++++
```

```
;=====clock signal
```

```
clk_pus:
    setb s9346_clk
    nop
    clr s9346_clk
```

從這裡開始是副程式，以下全部複製到考試程式的副程式區

```

        ret

;=====write start bit
wr_sb:
        setb s9346_cs
        setb s9346_di
        call clk_pus
        ret

;=====write 1-byte data
wr_byte:
        mov count_9346,#8
wrbit:   mov a,temp_9346
        rlc a
        mov temp_9346,a
        jc wrh
        clr s9346_di
        call clk_pus
        jmp wrlp9346
wrh:     setb s9346_di
        call clk_pus
wrlp9346: djnz count_9346,wrbit
        ret

;=====read 1-byte data
rd_byte:
        mov count_9346,#8
        clr a
rdbit:   call clk_pus
        jb s9346_do,rdh
        clr c
        rlc a
        jmp rdlp9346
rdh:     setb c
        rlc a
rdlp9346: djnz count_9346,rdbit
        mov temp_9346,a
        ret

;=====erase/write enable
ew_ena:
        mov temp_9346,#00110000b ;opcode=00,address=11xxxx
        call wr_sb
        call wr_byte
        clr s9346_di
        clr s9346_cs
        ret

```

```

;=====erase/write disable
ew_dis:
    mov temp_9346,#00000000b ;opcode=00,address=00xxxx
    call wr_sb
    call wr_byte
    clr s9346_di
    clr s9346_cs
    ret

;=====erase 1-data,write 0xff to 1-address
erase_data:
    setb adrs_9346.7 ;opcode=11
    setb adrs_9346.6
    mov temp_9346,adrs_9346
    call wr_sb
    call wr_byte
    clr s9346_cs
    clr s9346_di
    nop
    setb s9346_cs
erasewait:  nop
            jnb s9346_do,erasewait
            clr s9346_cs
            ret

;=====
;*****write 16 bits data to 93c46
write_data:
    call ew_ena
    call erase_data
    clr adrs_9346.7 ;opcode=01
    setb adrs_9346.6
    mov temp_9346,adrs_9346
    call wr_sb
    call wr_byte
    nop
    mov temp_9346,data_hb
    call wr_byte
    mov temp_9346,data_lb
    call wr_byte
    clr s9346_di
    clr s9346_cs
    nop
    setb s9346_cs
wrwait:    nop

```

```

        jnb  s9346_do,wrwait
        call ew_dis
        clr  s9346_cs
        ret
;*****read 16 bits data from 93c46
read_data:
        setb adrs_9346.7      ;opcode=10
        clr  adrs_9346.6
        mov  temp_9346,adrs_9346
        call wr_sb
        call wr_byte
        nop
        call rd_byte
        mov  data_hb,temp_9346
        call rd_byte
        mov  data_lb,temp_9346
        clr  s9346_di
        clr  s9346_cs
        ret
;
        end

```

C. 串列記憶體程式設計說明：

以下的步驟得背起來

步驟一：將程式開頭暫存器宣告程式複製到暫存器宣告區

```
s9346_cs      equ    p1.0
s9346_clk     equ    p1.1
s9346_di      equ    p1.2
s9346_do      equ    p1.3
;-----
temp_9346     equ    0x75
count_9346    equ    0x74
data_hb       equ    0x73      ;data register
data_lb       equ    0x72      ;data register
adrs_9346     equ    0x20      ;single bit define
```

步驟二：將所有副程式到複製到副程式區，範圍從第二頁標示處到第五頁，記住 end 這一行不要也順便 copy 過去，不然程式只會組譯到 end 那一行，後面的程式就被忽略掉了

步驟三：設計按數字鍵寫入 93C46 並顯示於 LCM 和讀取 93C46 資料副程式。(1 位數)

利用考場提供之程式

```
clr    s9346_cs      ;93C46 初始設定
clr    s9346_clk      ;
clr    s9346_di      ;
setb   s9346_do      ;
;
mov     adrs_9346, #0x00 ;設定寫入 93C46 的位址
mov     data_hb, #0x0a   ;設定高位元組資料=0x0a
mov     data_lb, #0x55 ;設定低位元組資料=0x55
call    write_data
;
mov     data_hb, #0x00 ;先清除 data_hb，等一下看看能不能正確讀到資料
mov     data_lb, #0x00 ;清除 data_lb
;
mov     adrs_9346, #0x00 ;設定 93C46 資料存放的位址
call    read_data       ;呼叫讀取資料副程式
mov     p3, data_hb      ;將讀到的高位元組資料顯示於 p3
```

```

mov    p2, data_lb      ;將讀到的低位元組資料顯示於 p2
end_9346:  jmp          end_9346

```

以上程式的功能為，先在 93C46 位址 00 的地方存入 0a55，再將暫存器 data_hb、data_lb 清為 0，再透過讀取程式，將資料存到暫存器 data_hb、data_lb，再顯示於 P3、P2，看看與當初寫入的值是否相符。

93C46 一個為只可以存放 2 個位元組的資料，為了方便程式撰寫，我們浪費一點，一個位址只存放一個位元組的資料，所以原程式中有關 data_lb 這個位址存放資料或讀取的部分就先刪掉，不要管它存甚麼值。

在此我們將上述修改成 2 個副程式，mem_wr 能寫入一個位元組的資料到 93C46 並且顯示於 LCM，mem_rd 能將從 93C46 讀取到的資料顯示於 LCM，整個程式修改如下(紅字部分是要額外加上的)

```

mem_wr:      ;寫入 93C46 並顯示於 LCM 副程式
            call    line2      ;游標移到第二行開頭

```

```

            clr     s9346_cs    ;93C46 初始設定
            clr     s9346_clk
            clr     s9346_di
            setb    s9346_do

```

將數值改成 ASCII 碼
0~9 ASCII 碼 =>30H~39H
相差 30H

```

;
mov     adrs_9346, #1      ;寫入第一個數字

```

```

mov     a, keynum
add     a, #30h
mov     data_hb, a
call    write_data
mov     abuf, data_hb
call    wr_lcdat

```

將數值顯示於 LCD,注意這裡要顯示的
數字已經不在累加器 a，要從 data_hb
複製過來

```

ret

```

```

mem_rd:      ;讀取 93C46 資料並顯示於 LCM 副程式
            call    line2      ;游標移到第二行開頭

```

```

            clr     s9346_cs    ;93C46 初始設定
            clr     s9346_clk
            clr     s9346_di
            setb    s9346_do

```

框內這幾行與上面相同，直接
複製過來後再改副程式名稱
就可以了

```

mov     adrs_9346, #1      ;從 93C46 讀第一個數字
call    read_data
mov     abuf, data_hb

```

將數值顯示於 LCD

```
call wr_lcddat
```

```
ret
```

步驟四：找出原程式控制串列記憶體功能表顯示位置，這個副程式目前只能顯示畫面和按"MU"鍵返回主功能

MEM_menu:

```
mov dptr,#MEM_scr1
```

```
call sho_lcm1
```

```
mov dptr,#MEM_scr2
```

```
call sho_lcm2
```

顯示 Number? 字串，第二行不顯示，清除為空白

MEM_scan:

```
call scan_key
```

```
mov a, keynum
```

```
cjne a, #0ah, MEM_scan ;判斷是否為 Menu 鍵
```

```
ret
```

按"MU"鍵離開副程式

步驟五：加上按鍵判斷程式（紅色字體），

（判斷是按"0"~"9"鍵數字鍵、"En"鍵、或是"MU"鍵返回主功能）

MEM_menu:

```
mov dptr,#MEM_scr1
```

```
call sho_lcm1
```

```
mov dptr,#MEM_scr2
```

```
call sho_lcm2
```

框內這幾行的寫法噢 Number Set 那題相似，直接複製過來後再改內容就可以了

MEM_scan:

```
call scan_key
```

```
mov a, keynum
```

```
clr c
```

```
subb a, #10 ;減 10
```

```
jnc m1 ;判斷是否有借位(小於 10)，如果大於 10 就判斷是否是"En"鍵
```

```
call mem_wr ;寫入 93C46 並顯示於 LCM 副程式
```

```
jmp MEM_scan
```

m1: mov a, keynum ;原來 a 的值已經改變，要重新載入一次

```
cjne a, #11, m2 ;判斷是否是"En"鍵(按"En"鍵，讀回值是 11)
```

```
call mem_rd
```

按"En"鍵，讀取記憶體內容

```
jmp MEM_scan
```

m2:

```
cjne a, #0ah, MEM_scan ;判斷是否為 Menu 鍵
```

```
ret
```

程式修改到這裡，應該可以按一個數字鍵將資料寫入 93C46 並顯示於 LCM，按” En” 鍵讀取 93C46 一個數字並顯示於 LCM。修改到這裡，程式初步結果可參考程式 5-6-1.a

步驟六：修改 mem_wr 副程式，使其能連續從鍵盤讀取 2 位數字

以下為步驟三完成的 mem_wr 副程式

```
mem_wr:      ;寫入 93C46 並顯示於 LCM 副程式
             call    line2          ;游標移到第二行開頭

             clr     s9346_cs       ;93C46 初始設定
             clr     s9346_clk
             clr     s9346_di
             setb    s9346_do

;
             mov     adrs_9346, #1   ;寫入第一個數字
             mov     a, keynum
             add     a, #30h
             mov     data_hb, a
             call    write_data
             mov     abuf, data_hb
             call    wr_lcddat

             ret
```

這段程式寫入一個字

新的程式從這裡開始加

上述程式只能將一個數字寫入 93C46 並顯示於 LCM，如果還要能再讀取並寫入第 2 個數字，必須再加上鍵盤讀取程式，因為密碼必須要是數字，所以程式中必須判斷讀到的是否是數字 0~9，否則就再重新讀一次，直到讀到數字為止，

```
m3:      call    scan_key      ;從鍵盤讀取第 2 個數字
             mov     a, keynum
             clr     c
             subb    a, #10     ;判斷是不是數字(有沒有大於 10)
             jnc     m3
```

讀到後再將第二個數字寫入 93C46 並顯示於 LCM，這段程式前面已經寫過，複製前面的程式就可以，記得要修改寫入 93C46 的位址。

```
mov     adrs_9346, #2         ;寫入第二個數字的位址
```

```

mov    a, keynum
add    a, #30h
mov    data_hb, a
call   write_data
mov    abuf, data_hb
call   wr_lcddat

```

如此，可得以下完整答案，記得標記位置和存入 93C46 的位址不要重複，免得出錯。

```

mem_wr:
    call    line2

    clr     s9346_cs        ;initial status
    clr     s9346_clk       ;initial status
    clr     s9346_di        ;initial status
    setb    s9346_do        ;;initial status(notice)
;

    mov     adrs_9346, #1    ;write 93c46@00h
    mov     a, keynum
    add     a, #30h
    mov     data_hb, a       ;if data high byte=0ah
    call    write_data
    mov     abuf, data_hb
    call    wr_lcddat

m3:    call    scan_key      ;從鍵盤讀取第 2 個數字
    mov     a, keynum
    clr     c
    subb    a, #10          ;判斷是不是數字(有沒有大於 10)
    jnc     m3

    mov     adrs_9346, #2    ;寫入第 2 個數字的位址
    mov     a, keynum
    add     a, #30h
    mov     data_hb, a      ;
    call    write_data
    mov     abuf, data_hb
    call    wr_lcddat

RET

```

步驟七：修改 mem_rd 副程式，使其能連續從 93C46 讀取並顯示 2 位數字

```
mem_rd:    ;讀取 93C46 資料並顯示於 LCM 副程式
           call    line2                ;游標移到第二行開頭

           clr     s9346_cs             ;93C46 初始設定
           clr     s9346_clk
           clr     s9346_di
           setb    s9346_do

           mov     adrs_9346,#1         ;從 93C46 讀第一個數字
           call    read_data
           mov     abuf,data_hb
           call    wr_lcddat

           ret
```

紅字的部份是讀取一個字，只要再複製一次，修改位址即可，記得讀取和寫入位址一定要互相對應，不然顯示結果會有錯。

```
mem_rd:    ;讀取 93C46 資料並顯示於 LCM 副程式
           call    line2                ;游標移到第二行開頭

           clr     s9346_cs             ;93C46 初始設定
           clr     s9346_clk
           clr     s9346_di
           setb    s9346_do

           mov     adrs_9346,#1         ;從 93C46 讀第一個數字
           call    read_data
           mov     abuf,data_hb
           call    wr_lcddat

           mov     adrs_9346,#2         ;從 93C46 讀第 2 個數字
           call    read_data
           mov     abuf,data_hb
           call    wr_lcddat

           ret
```

修改到此，可以先測試 2 個字的密碼是否可以正常寫入和讀取，確定 OK 後，再進行 4 位數密碼的修改。

步驟八：承步驟六，修改 mem 副程式，使其能連續從鍵盤讀取 4 位數字

mem_wr:

```
call    line2                ;homing to left

clr     s9346_cs              ;initial status
clr     s9346_clk             ;initial status
clr     s9346_di              ;initial status
setb    s9346_do              ;;initial status(notice)
```

;

```
mov     adrs_9346,#1          ;write 93c46@00h
mov     a,keynum
add     a,#30h
mov     data_hb,a             ;if data high byte=0ah
call    write_data
mov     abuf,data_hb
call    wr_lcddat
```

這段程式從鍵盤讀取寫入一個字，多複製 2 就可以了。

```
m3:    call    scan_key        ;從鍵盤讀取第 2 個數字
mov     a,keynum
clr     c
subb    a,#10                 ;判斷是不是數字(有沒有大於 10)
jnc     m3

mov     adrs_9346,#2          ;寫入第 2 個數字的位址
mov     a,keynum
add     a,#30h
mov     data_hb,a             ;
call    write_data
mov     abuf,data_hb
call    wr_lcddat
```

RET

再複製紅字部分 2 次，如此，可得以下完整答案，記得標記位置和存入 93C46 的位址不要重複，免得出錯。

mem_wr:

```
call    line2                ;homing to left
```

```

    clr    s9346_cs        ;initial status
    clr    s9346_clk       ;initial status
    clr    s9346_di        ;initial status
    setb   s9346_do        ;;initial status(notice)
;

    mov     adrs_9346,#1    ;write 93c46@00h
    mov     a,keynum
    add     a,#30h
    mov     data_hb,a       ;if data high byte=0ah
    call    write_data
    mov     abuf,data_hb
    call    wr_lcddat

m3:    call    scan_key      ;從鍵盤讀取第 2 個數字
    mov     a,keynum
    clr     c
    subb    a,#10           ;判斷是不是數字(有沒有大於 10)
    jnc     m3

    mov     adrs_9346,#2    ;寫入第 2 個數字的位址
    mov     a,keynum
    add     a,#30h
    mov     data_hb,a       ;
    call    write_data
    mov     abuf,data_hb
    call    wr_lcddat

m4:    call    scan_key      ;從鍵盤讀取第 3 個數字
    mov     a,keynum
    clr     c
    subb    a,#10
    jnc     m4

    mov     adrs_9346,#3    ;寫入第 3 個數字的位址
    mov     a,keynum
    add     a,#30h
    mov     data_hb,a
    call    write_data
    mov     abuf,data_hb
    call    wr_lcddat

m5:    call    scan_key      ;從鍵盤讀取第 4 個數字
    mov     a,keynum

```

```

clr    c
subb   a, #10
jnc    m5

mov     adrs_9346, #4      ;寫入第 4 個數字的位址
mov     a, keynum
add     a, #30h
mov     data_hb, a
call    write_data
mov     abuf, data_hb
call    wr_lcddat

ret

```

步驟九：承步驟七，修改 mem_rd 副程式，使其能連續讀取 4 位數字

```

mem_rd:    ;讀取 93C46 資料並顯示於 LCM 副程式
           call    line2      ;游標移到第二行開頭

           clr     s9346_cs    ;93C46 初始設定
           clr     s9346_clk
           clr     s9346_di
           setb    s9346_do

           mov     adrs_9346, #1      ;從 93C46 讀第一個數字
           call    read_data
           mov     abuf, data_hb
           call    wr_lcddat

           mov     adrs_9346, #2      ;從 93C46 讀第二個數字
           call    read_data
           mov     abuf, data_hb
           call    wr_lcddat

           ret

```

紅字的部份是從 93C46 讀取 2 個字，只要再複製一次，修改位址即可，記得讀取和寫入位址一定要互相對應，不然顯示結果會有錯。

```

mem_rd:    ;讀取 93C46 資料並顯示於 LCM 副程式
           call    line2      ;游標移到第二行開頭

```

```

clr    s9346_cs      ;93C46 初始設定
clr    s9346_clk
clr    s9346_di
setb   s9346_do

mov     adrs_9346,#1    ;從 93C46 讀第一個數字
call    read_data
mov     abuf,data_hb
call    wr_lcddat

mov     adrs_9346,#2    ;從 93C46 讀第 2 個數字
call    read_data
mov     abuf,data_hb
call    wr_lcddat

mov     adrs_9346,#3    ;從 93C46 讀第 3 個數字
call    read_data
mov     abuf,data_hb
call    wr_lcddat

mov     adrs_9346,#4    ;從 93C46 讀第 4 個數字
call    read_data
mov     abuf,data_hb
call    wr_lcddat

ret

```

到此串列記憶體存取控制已經完成，詳細內容可參考程式 5-6.a