תוכנית דוגמא b800h.asm

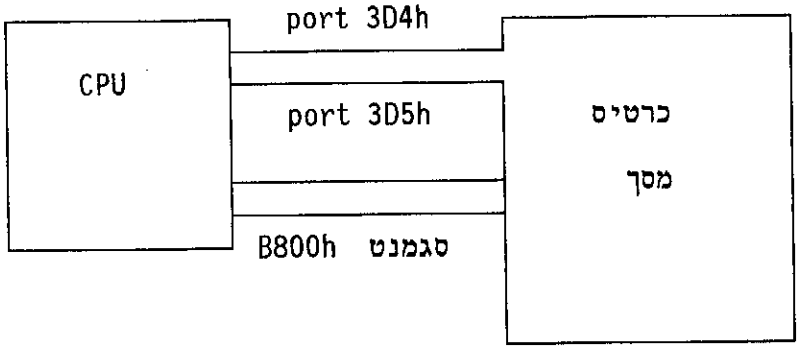
התוכנית הזו נועדה להמחיש את נושא ה-Memory Mapped I/O ופלט למסך כולל ה-Hardware Scrolling.

תוכנית זו משתמשת במסך הגרפי במוד טקסט צבעוני 25x40.

בתנאים אלו כתיבה לתצוגה היא דרך סגמנט B800h (זיכרון פיזי 736k).

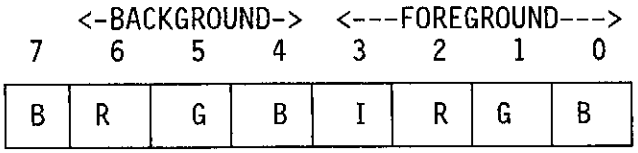
התוכנית כותבת 'A' -ים, 'B' -ים, 'C' -ים, 'D' -ים בצבעים שונים למסך דרך סגמנט B800h, ומדפדפת ביניהם ע"י כתיבה ל-ports 3D4h, 3D5h מה שקרוי דיפדוף חומרה (Hardware Scrolling).

לצורך נקודת ההשקפה שלנו המודל של החיבורים נראה כך:



פורמטים של אינפורמציה

במוד טקסט המערכת מפרשת את תוכן סגמנט B800h כצמדים של בתים המתארים כל אחד משבצת אחת במסך. הבית הנמוך (היסטים זוגיים) מפורש כקוד ה-ASCII של הבית שאותו צריך להציג. הבית בכתובת הגבוהה (היסטים איזוגיים) מפורש כבית של מאפיינים (Attribute byte):



B - Blink (1 - Foreground character blinks)
R - Red
G - Green
B - Blue
I - Intensity (0 - Low, 1 - High)

כמו כן ה-CPU מחובר למסך דרך זוג ה-ports 3D4h - 3D5h, שדרכם קובעים (בין השאר) את דף התצוגה, ומיקום ה-cursor. אנו קובעים את מוד המסך ע"י פסיקת תוכנה BIOS 10h.

בתוכנית הזו כרטיס המסך מקבלת את ההנחיות הבאות דרך פורטים 3D4h - 3D5h:

לפי התוכן הנכתב לפורט 3D4h הכרטיס מפרש / מחבר את פורט 3D5h לאוגר הפנימי שלו המתאים.

הערכים שאנחנו נשתמש (יש יותר):

תוכן 10 בפורט 3D4h: פורט 3D5h מפורש כגבול עליון לתצוגת ה-cursor (קו 0 עד 14).

תוכן 11 בפורט 3D4h: פורט 3D5h מפורש כגבול תחתון לתצוגת ה-cursor (קו 1 עד 15).

תוכן 12 בפורט 3D4h: פורט 3D5h מפורש כבית עליון לנקודת ההתחלה של תצוגת המסך (היסט בבתים מתחילת סגמנט B800h).

תוכן 13 בפורט 3D4h: פורט 3D5h מפורש כבית תחתון לנקודת ההתחלה של תצוגת המסך (היסט בבתים מתחילת סגמנט B800h).

תוכן 14 בפורט 3D4h: פורט 3D5h מפורש כבית עליון למיקום ה-cursor (היסט בבתים מתחילת סגמנט B800h).

תוכן 15 בפורט 3D4h: פורט 3D5h מפורש כבית תחתון למיקום ה-cursor (היסט בבתים מתחילת סגמנט B800h).

עם הרצת התוכנית הזו המסך יציג סידרה 50 שורות של 'A' -ים בצבע תכלת שבהדרגה יוחלפו ב-50 שורות של 'B' סגולים שיוחלפו בהדרגה ב-'C'-ים חומים שייוחלפו ב-50 שורות של 'D'-ים לבנים. ה-cursor נמצא בתחילת אחת השורות תחת אחד ה-'B' -ים. עם ההגעה לסוף ה-'D'-ים יתחיל תהליך הפוך של דפדוף חזרה עד לתחילת ה-'A' -ים וחוזר חלילה. לחיצת מקש כלשהוא (למעט Esc) יגרום לעצירת הדפדוף ולחיצה נוספת של מקש כלשהוא יחדש אותו. לחיצת Esc יגרום לסיום התוכנית.

בפועל התוכנית כותבת לזיכרון המסך יותר משהמסך יכול להציג. ע"י שינוי נקודת ההתחלה של התצוגה בכל פעם מוצגת חלק אחר של זיכרון כרטיס המסך. הפעולה הזו, של שינוי התצוגה לא ע"י כתיבת ערכים חדשים אלא ע"י שינוי נקודת ההתחלה נקרא Hardware Scrolling.

מהלך התוכנית היא

- להעביר את המסך למוד טקסט 25x40 - 25 שורות של 40 תוים לשורה (בניגוד ל-80 תוים לשורה בדרך כלל) ע"י רוטינת ה-BIOS INT 10h אופציה AH = 0, AL = 1. לא ניכנס לפרטים בנושא INT 10h הזה כאן. נאמר רק שבמוד הזה מסך מציג את התוכן של סגמנט B800h.

- מילוי התוכן הרצוי לזיכרון המסך. הדבר נעשה בארבעה שלבים ע"י הצבת הערך הרצוי ל-AX וכתיבת הערך 2000 פעם. למשל כתיבת ה-'A'-ים נעשה ע"י

```
MOV AL,'A'
MOH AH,03h
MOV CX,2000
Loop1:
    MOV ES:[DI],AX
    ADD DI,2
    LOOP Loop1
```

לכאורה הפקודה MOV ES:[DI],AX היא פקודת כתיבה לזיכרון אבל כתיבה לסגמנט B800h משמעותו פעולת **פלט** דומה יותר ל-OUT מאשר כתיבה לזיכרון. המשמעות כאן היא כתיבה לזיכרון המסך של 50 שורות של 'A'-ים בצבע

תכלת.

- קביעת מיקום ה-cursor. הדבר נעשה ע"י הפקודות

```
MOV BX,40*63
MOV DX,3D4h
MOV AL,14
MOV AH,BH
OUT DX,AX
;
MOV AL,15
MOV AH,BL
OUT DX,AX
```

ה-OUT הראשון יהיה הבית המשמעותי יותר וה-OUT השני הבית המשמעותי פחות. הפקודות הללו ממקמים את ה-cursor בעמוד הראשון בשורה ה-64 של זיכרון המסך.

מהשלב הזה התוכנית תכנס ללולאה של

- המתנה
- בדיקת לחיצת מקש
- שינוי נקודת ההתחלה של התצוגה של המסך

שינוי נקודת ההתחלה של תצוגת המסך נעשה בשני שלבים:

הבית המשמעותי יותר נכתב ע"י הפקודות

```
MOV DX,3D4h
MOV AL,12
MOV AH,BH
OUT DX,AX
```

הבית המשמעותי פחות נכתב ע"י הפקודות

```
MOV AL,13
MOV AH,BL
OUT DX,AX
```

```
;
; B800h.ASM
; This program demonstrates the use of hardware scrolling.
;
Stack               SEGMENT PARA STACK 'STACK'
                    DB      256 DUP(0)
Stack               ENDS
;
Data                SEGMENT PARA PUBLIC 'Data'
Count               DB      0       ; Number of lines scrollled down
Base                DW      0
;
Direction           DB      0       ; The scroll direction
;                                     0=Down    1=Up
Data                ENDS
Code                SEGMENT PARA PUBLIC 'Code'
Start               PROC    FAR
;
;Standard program prologue
;
  ASSUME            CS:Code
  PUSH              DS              ; Save PSP segment address
  MOV               AX,0
  PUSH              AX              ; Save RET address offset (PSP+0)
  MOV               AX,Data
  MOV               DS,AX           ; Establish Data segment addressability
  ASSUME            DS:Data
;
;Part1 : Initialize the display adapter
;
  MOV               AH,0            ; Select function = 'SET MODE'
  MOV               AL,1            ; 40 BY 25 Color image
  INT               10H             ; Adapter initilizedD. Page 0 displayed
;
  MOV               AX,0B800h       ; Segment address of memory of color adapter
                                    ; in text mode
  MOV               ES,AX           ; Set up extra segment register
  MOV               DI,0            ; Initial offset address into segment
  MOV               AL,'A'          ; Character A to fill adapter memory
  MOV               AH,03h          ; Attribute byte: BLUE + GREEN = LIGHT BLUE
;
  MOV               CX,2000
Loop1:
        MOV ES:[DI],AX
        ADD DI,2
        LOOP Loop1
;
  MOV               AL,'B'          ; Character B to fill adapter memoory
  MOV               AH,05h          ; Attribute byte: BLUE + RED = PURPLE
;
  MOV               CX,2000
Loop2:
        MOV ES:[DI],AX
        ADD DI,2
        LOOP Loop2
```

```
;
  MOV               AL,'C'          ; Character C to fill adapter memory
  MOV               AH,06h          ; Attribute byte: GREEN + RED = BROWN
;
  MOV               CX,2000
Loop3:
        MOV ES:[DI],AX
        ADD DI,2
        LOOP Loop3
;
  MOV               AL,'D'          ; Character D to fill adapter memory
  MOV               AH,07h          ; Attribute byte: GREEN + RED + BLUE = WHITE
;
  MOV               CX,2000
Loop4:
        MOV ES:[DI],AX
        ADD DI,2
        LOOP Loop4
;
;
; Set the cursor address registers
;
  MOV               BX,40*63        ; Cursor position: 40th row, Col 0
  MOV               DX,3D4h         ; 3D4h - 3D5h: Display adapter ports
  MOV               AL,14           ; 14 - Cursor address high byte register
  MOV               AH,BH           ; Load AH with desired high byte value
  OUT               DX,AX           ; Output AL to port 3D4h, AH to port 3D5h
;
  MOV               AL,15           ; 15 - Cursor address low byte register
  MOV               AH,BL           ; Load AH with desired low byte value
  OUT               DX,AX           ; Output AL to port 3D4h, AH to port 3D5h
;
;
;PART 2 : Scroll the display every second until a key is hit
;
Delay:
  ; Delay AL*CX operations
  MOV                AL,200
Tenth:
  MOV                CX,60000
Dloop:
  LOOP              Dloop
  DEC               AL
  CMP               AL,0
  JNE               Tenth
;
  MOV               AH,1            ; User pressed key?
  INT               16H
  JZ                Scroll          ; No, scroll
  MOV               AH,0            ; Yes, read the key
  INT               16H
  CMP               AH,1            ; Is it Esc?
  JE                ToReturn        ; Yes - return to DOS
  MOV               AH,0            ; No: Wait for another key
  INT               16H
  CMP               AH,1            ; Was it Esc?
  JE                ToReturn        ; Yes - return to DOS
```

```
;                                   ; No - Continue
Scroll:
  MOV           BX,Base             ; Retrieve present location
  CMP           Direction,0         ; Backwards or Forwards?
  JNE           Backwards           ; Direction = 1: Backwards
;                                     Direction = 0: Forwards
  INC           Count               ; Advance 1 line
  CMP           Count,160           ; End of 4 pages?
  JB            Down_Ok             ; No
;
  MOV           Direction,1         ; Yes, reverse direction
Down_Ok:
  ADD           BX,40               ; No: Advance 1 line
  MOV           Base,BX             ; Store location
;
  JMP           SHORT  Update
;
Backwards:
  DEC           Count               ; Backtack 1 line
  CMP           Count,0             ; Reached first line?
  JNE           Up_Ok               ; No
;
  MOV           Direction,0         ; Yes, reverse direction
Up_Ok:
  SUB           BX,40               ; Backtack 1 line
  MOV           Base,BX             ; Store location
;
Update:
  MOV           DX,3D4h             ; 3D4h - 3D5h: Display adapter ports
  MOV           AL,12               ; Display position high byte register
  MOV           AH,BH               ; Move desired high byte value to AH
  OUT           DX,AX               ; Output AL to port 3D4h, AH to port 3D5h
  MOV           AL,13               ; Display position low byte register
  MOV           AH,BL               ; Move desired low byte value to AH
  OUT           DX,AX               ; Output AL to port 3D4h, AH to port 3D5h
;
  JMP           Delay                 ; Repeat the process
;
ToReturn:
                MOV  AX,3           ; Restore adapter mode
                INT  10h
                RET                 ; Return to DOS
Start           ENDP
Code            ENDS
  END           Start
```

תוכנית דוגמא cursor2.asm

התוכנית הזו נועדה להדגים אספקטים נוספים של ניהול המסך. מה שהתוכנית הזו עושה היא להציג את התו 'A' באמצע המסך כאשר מלבדו המסך ריק וה-cursor מהבהב מתחתיו. כל לחיצה עת מקש כלשהוא (למעט Esc) יגרום ל-cursor לשנות צורה, מקו תחתון ל-'A' עד ל-cursor המירבי של כיסוי כל המשבצת וחזרה. לחיצת Esc יביא לסיום התוכנית.

הגודל של ה-cursor נקבע לפי גבול עליון וגבול תחתון. הקו העליון ביותר ממוספר 0 והתחתון ביותר הוא 15. התוכנית יכולה לבחור גבול עליון מ-0 ל-14 וגבול תחתון כל מספר עד 15 מתחתיו. לפיכך ה-cursor המירבי הוא מ-0 ל-15 ומינימלי "הרגיל": הוא מ-14 ל-15.

הגבול העליון של ה-cursor נקבע לפי תוכן port 3D5h בתנאי שב-port 3D4h ישנו הערך 10 (Ah).

הגבול התחתון של ה-cursor נקבע לפי תוכן port 3D5h בתנאי שב-port 3D4h ישנו הערך 11 (Bh).

לפיכך קביעת ה-cursor המירבי נעשה ע"י הפקודות

```
MOV DX,3D4h
MOV AX,000Ah
OUT DX,AX
MOV AX,0F0Bh
OUT DX,AX
```

וקביעת ה-cursor המינימלי נעשה ע"י הפקודות

```
MOV DX,3D4h
MOV AX,0E0Ah
OUT DX,AX
MOV AX,0F0Bh
OUT DX,AX
```

```
MOV AX, 0B800h
MOV ES,AX
MOV DI,0
MOV AL,' '
MOV AH,0Eh
MOV CX,1000
CLD
REP STOSW
```

שמשמעותו הגדרת המסך כרווחים בצבע צהוב מודגש (למעשה הכל שחור) על פני 1000 התוים הראשונים (25 שורות ראשונות או מסך שלם) של המסך.
STOSW היא פקודת מחרוזת המציבה את התוכן של AX בכתובת [ES:[DI וקידום DI אוטומטית ב-2. משמעות שלושת הפקודות

```
MOV CX,1000
CLD
REP STOSW
```

משמעותו "כתוב את תוכן AX לתוך 1000 בתים החל מכתובת [ES:[DI".
זוהי אפשרות יותר יעילה מאשר מימוש רגיל שקול מבחינת התוצאה של

```
MOV CX,1000
Loop1:
   MOV ES:[DI],AX
   ADD DI,2
   LOOP Loop1
```

משמעות ה-REP STOSW היא בדרך כלל פעולת כתיבה לזיכרון לכל דבר אבל בסגמנט המסוים מאד B800h פירושו Memory Mapped I/O לכרטיס המסך.

שינוי המשבצת האמצעית במסך ל-'A' צהוב על רקע שחור נעשה ע"י הפקודה

```
MOV BYTE PTR ES:[2*(12*40+20)],'A'
```

משמעותו תוכן התו מספר 20 (ה-21 מהקצה השמאלי) בשורה מספר 12 (ה-13 מהקצה העליון) של המסך יהיה 'A'. הצבע הצהוב נקבע קודם לכן ע"י ה-REP MOVSW.

מיקום ה-cursor נעשה בשני שלבים כמו בתוכנית b800h.asm ע"י הפקודות

```
MOV BX,12*40+20
MOV DX,3D4H
MOV AL,14
MOV AH,BH
OUT DX,AX
MOV AL,15
MOV AH,BL
OUT DX,AX
```

כאשר DX = 3D4h ו- AL = 14 ו-AH פירושו קביעת בית עליון של מיקום ה-cursor, כאשר DX = 3D4h ו- AL = 15 ו-AH פירושו קביעת בית תחתון של מיקום ה-cursor.

```
;
; Cursor2.asm
;
;This program demonstrates cursor manipulation.
;
;
Stak                SEGMENT PARA STACK 'STACK'
DB                   256 DUP(0)
Stak                ENDS
;
Data                 SEGMENT PARA PUBLIC 'Data'
CursorPos            DW      0      ; Number of lines scrolled down
Base                 DW      0
;
Data                 ENDS
Code                 SEGMENT PARA PUBLIC 'Code'
                     .386  ; Enable 386 commands
SetCursorPos            PROC    NEAR
; Set Cursor to position in BX
; Input: BX
; Output: None
;
  MOV                DX,3D4H  ; Point to 3D4H - 3D5H port pair
  MOV                AL,14    ; Address of cursor register pos high byte
  MOV                AH,BH    ; Get desired value of cursor pos high byte
  OUT                DX,AX    ; Port(3D4h) = 14, Port(3D5h) = Value of BH
;
  MOV                AL,15    ; Address of cursor register pos low byte
  MOV                AH,BL    ; Get desired value pf cursor pos low byte
  OUT                DX,AX    ; Port(3D4h) = 15, Port(3D5h) = Value of BL
;
  RET                         ; Return to caller
SetCursorPos            ENDP
;
Start                PROC    FAR
;
;STANDARD PROGRAM PROLOGUE
;
  ASSUME             CS:Code
  PUSH               DS           ; Save PSP segment address
  MOV                AX,0
  PUSH               AX           ; Save INT 20h address offset (PSP+0)
  MOV                AX,Data
  MOV                DS,AX        ; Establish Data segment addressability
  ASSUME             DS:Data
;
;Part1 : Initialize the display adapter
;
  MOV                AH,0           ; Select function = 'Set mode'
  MOV                AL,1           ; 40 by 25 color image
  INT                10h            ; Adapter initialized. Page 0 displayed
;
  MOV                AX,0B800h      ; Segment address of memory on color adapter
;
  MOV                ES,AX          ; Set up extra segment register
  MOV                DI,0           ; Initial offset address into segment
  MOV                AL,' '         ; Character space to fill adapter memory
  MOV                AH,0Eh         ; Attribute byte : Intense yellow
  MOV                CX,1000        ; Initialize count, 1 Screen
  CLD                               ; Write forward
  REP                STOSW          ; Write
```

```
;
; Write 'A' in mid screen
;
  MOV               BYTE PTR ES:[2*(12*40+20)],'A'
;
; Set the cursor address registers
;
  MOV               BX,12*40+20
  CALL              SetCursorPos
;
;PART 2 : Wait for key strike
;
; Wait for key
;
NextLoop:
  MOV               AH,0        ; Wait and read key
  INT               16h         ;
  CMP               AH,1        ; Is it Esc?
  JE                ToReturn    ; Yes - Return to DOS
;
;  Not esc key - change cursor
;
  ; 3D4H  Graphics adapter address register port
  ; 3D5H  Graphics adapter data register port
;
    MOV              DX,3D4h   ; Point TO 3D4h - 3D5h port pair
    MOV              AX,000Ah  ; Cursor start address (0Ah) - Value 0 (00h)
    OUT              DX,AX     ; Port(3D4h) = 0Ah, Port(3D5h) = 01h
    MOV              AX,0F0Bh  ; Cursor end address - Value 15 (0Eh)
    OUT              DX,AX     ; Port(3D4h) = 0Bh, Port(3D5h) = 0Eh
;
; Wait for key
;
  MOV               AH,0
  INT               16h
  CMP               AH,1        ; Is it Esc?
  JE                ToReturn    ; Yes - Return to DOS
;
    MOV              DX,3D4h   ; Point to 3D4H - 3D5H port pair
    MOV              AX,0E0Ah  ; Cursor start address (0Ah) - Value 14 (0Dh)
    OUT              DX,AX     ; Port(3D4h) = 0Ah, Port(3D5h) = 01h
    MOV              AX,0F0Bh  ; Cursor end address - Value 15 (0Eh)
    OUT              DX,AX     ; Port(3D4h) = 0Bh, Port(3D5h) = 0Eh
;
  JMP               NextLoop    ; Repeat main loop
;
ToReturn:
                    MOV  AX,2
                    INT  10h
                    RET
Start               ENDP
Code                ENDS
  END               Start
```