3. Praktikum

1. Aufgabe Direkte Bildschirm-Text-Ausgabe ohne Systemfunktionen

Das Unterprogramm-Paket **eadirekt.asm aus Praktikum 1 Aufgabe 2.2 ist in einem zweiten Schritt in ein UP-Paket** umzuwandeln, das keinerlei Funktionen zur Bildschirm-Ausgabe des DOS-INT 21h bzw Bildschirm-Ausgabe des BIOS-INT 10h verwendet, sondern **"direkt" im Textmodus auf den Bildschirm ausgeben** soll
Die INT 10h-Funktion 12h in grafinit (Grafikbildschirm initialisieren) braucht nicht ersetzt zu werden, denn sie betrifft die **Grafik**-Ausgabe.

Die INT10h-BIOS-Funktionen 0Eh (in putch) soll hierfür durch ein Unterprogramm **TEXTAUS** nachgebildet (emuliert) werden. Dem UP TEXTAUS werden dazu die gleichen Parameter wie dem INT 10h übergeben .
Da in diesem Unterprogramm aber weder DOS noch BIOS-Funktionen verwendet werden dürfen, bleibt hier nur der Zugriff durch das **direkte** Schreiben in den **Bildschirmspeicher**.

Die neu entstandene Version der UP-Bibliothek **eadirekt** ist ebenfalls mit Aufgabe 1.3 des Praktikums 1 (Euklid –Algorithmus) zu testen.

Vorschlag zur Vorgehensweise:
Die komfortable INT10h-Funktion 0Eh (- führt Cursor weiter
- führt an unterer Bildschirmgrenze einen einzeiligen ScrollUp durch
- interpretiert Steuerzeichen wie LF, CR und Backspace)

zuerst in kleinere, speziellere BIOS-INT10h- Funktionen in entsprechender Reihenfolge **auflösen**

0Fh = Abfrage des aktuellen Bildschirmstatus
03h = Position des Cursors bestimmen
09h/0Ah = Schreiben in Cursorposition
02h = Cursor auf Bildschirm weiterpositionieren
eventuell 06h = Fenster auf aktueller Seite nach oben Scrollen

und dann die einzelnen Funktionen **<u>mit direktem Zugriff</u>** ausprogrammieren und nach jedem Abänderungs - schritt austesten.
Skript: Video Seite 16 – 17 , System-Funktionen

Hinweise:
Da die Weiterführung des **blinkenden Cursors nur mit direkter Programmierung** des CRT-Controllers möglich ist und dies in der Vorlesung nicht behandelt wird, kann entweder auf die Mitführung des blinkenden Cursors verzichtet oder es kann dafür das unten angegebene Unterprogramm CURSORDI verwendet werden.
Skript: Video Seite 9 - 10
Weiterhin soll eine Ausgabe nur im Textmodus erfolgen. Wird TEXTAUS im Grafikmodus aufgerufen, so soll keine Ausgabe des Zeichens erfolgen, sondern nur mit einer Fehlermeldung (in diesem Fall kann hier ausnahmsweise die Meldungsausgabe mit einer Systemfunktion erfolgen INT 21h oder INT 10h . **Warum ??**) das Unterprogramm beendet werden.
Das Programm braucht nur die klassischen Textmodi 0, 1, 2, 3 und 7 (CGA und MDA) korrekt abhandeln.

Zur Demonstration werden die folgenden drei Unterprogramme im Quellcode zur Verfügung gestellt:

ZEICHADR - bestimmt aus Bildschirmposition die Bildspeicheradresse, wobei die Segmentadresse für Monochrom- oder Color-Karten aus den dabei unterschiedlichen Portadressen des CRT-Controllers ermittelt wird.

SCROLLUP - bewirkt das Hinaufschieben des Bildschirminhaltes um eine Zeile

CURSORDI - anhand der BIOS-Position (BIOS-Variable 40:50 h) wird der blinkende Cursor durch direkte Programmierung des CRT-Controllers auf dem Bildschirm dargestellt

Funktionen des (BIOS-) INT 10 h

00h	Setzen des Video-Modus	
01h	Definition des Erscheinungsbildes des Cursors	
02h	Positionierung des Cursors	
03h	Auslesen der Cursor-Position	
04h	Auslesen der Lichtstiftposition	
05h	Auswahl der aktuellen Bildschirmseite	
06h	Textzeilen nach oben schieben (scrollen)	
07h	Textzeilen nach unten schieben (scrollen)	
08h	Auslesen eines Zeichens/Farbe	
09h	Schreiben eines Zeichens/Farbe	
0Ah	Schreiben eines Zeichens	
0Bh U.-fkt. 00h	Auswahl der Rahmen-/Hintergrundfarbe	
0Bh U.-fkt. 01h	Auswahl der Farbpalette	
0Ch	Schreibe Grafikpunkt	
0Dh	Lese Grafikpunkt	
0Eh	Schreiben eines Zeichens	
0Fh	Auslesen des Video-Modus	
13h	Ausgabe einer Zeichenkette	

```
CURSORDI    PROC        ;Blink-Cursor positionieren durch direkte
                        ;Programmierung des CRT-Controllers
            pusha       ;Position wird aus BIOS-Datenbereich ermittelt
            push    es

            mov     ax,40h
            mov     es,ax

            mov     bl,es:62h       ; Cursor-Position bestimmen
            mov     bh,0            ; aus aktiver Page + 40:50h
            shl     bx,1            ; Wortadresse ( *2 )
            add     bx,50h
            mov     DX,es:BX        ; nach DX Cursor-Position (Zeile-Spalte)

            mov     cx,dx           ; in CX Spalte  fuer ZEICHADR
            xor     ch,ch           ;
            mov     dl,dh           ; in DX Zeile
            xor     dh,dh
            call    zeichadr        ; Adresse in ES:DI zurueck

            mov     bx,di
            shr     bx,1     ;aus Byte- wird Wortadresse fuer CRT-Controller

            mov     dx,3D4h  ;Index_Register des CRT CRT-Adress-Register
            mov     al,0Eh   ;High-Adressbyte Cursor-Location-High-Register
            mov     ah,bh    ;High-Byte der Wortadresse
            out     dx,ax

            mov     al,0Fh   ;Low-Adressbyte Cursor-Location-Low-Register
            mov     ah,bl
            out     dx,ax

            pop     es
            popa
            ret

CURSORDI    ENDP
```

```
SCROLLUP   PROC         ;Scroll Up um 1 Zeile nach oben
           pushf
           pusha
           push    DS
           push    ES
           MOV     AX,40h       ;Zeilen u. Spalten aus BIOS ermitteln
           MOV     ES,AX
           MOV     AX,ES:[4Ah]   ;Spalten in AL
           MOV     DL,ES:[84h]   ;Zeilen-1 in DL
           inc     dl            ;da ZEILEN-1
           mul     dl            ;Anzahl Worte die zu verschieben sind

           mov     cx,0
           mov     dx,1
           CALL    ZEICHADR
           mov     si,di         ;Quelladresse nach DS:SI
           push    es
           pop     ds

           mov     cx,0
           mov     dx,0
           CALL    ZEICHADR      ;Zieladresse in ES:DI
           cld
           mov     cx,ax         ;Anzahl Worte zu verschieben
           rep     movsw
           pop     ES
           pop     DS
           popa
           popf
           ret
SCROLLUP   ENDP


;ZEICHADR liefert Adresse einer Bildschirmposition auf aktiver Page
;Eingabe: CX = Spalte , DX = Zeile
;Ausgabe: ES:DI Segment und Offset der Position im Videospeicher
;
ZEICHADR PROC    NEAR
         PUSHF
         PUSH    AX
         PUSH    DS

         MOV     AX,40H
         MOV     DS,AX
         MOV     AX,DS:[63H] ;I/O - Port fuer 6845
         CMP     AX,03B4H
         JNE     COLOR
         MOV     AX,0B000H; Video-Segment MDA (Port 03B4 h)
         JMP     VORWAR
COLOR:   MOV     AX,0B800H; Video-Segment CGA (Port 03D4 h)
VORWAR:  MOV     ES,AX

         MOV     AX,DS:[4Ah]  ;Zeichen pro Zeile
         MUL     DL           ; Zeile * 40/80
         ADD     AX,CX        ; + Spalte
         SAL     AX,1         ; * 2
         MOV     DI,AX
         ADD     DI,DS:[4Eh]  ;+ Offset der aktiven Page
         POP     DS
         POP     AX
         POPF
         RET
ZEICHADR ENDP
```