```
                        PAGE 118,121
                        TITLE VIDEO1 --- 03/24/86  VIDEO DISPLAY BIOS
                        .LIST
0000                    CODE    SEGMENT BYTE PUBLIC

                                PUBLIC  ACT_DISP_PAGE
                                PUBLIC  READ_AC_CURRENT
                                PUBLIC  READ_CURSOR
                                PUBLIC  READ_DOT
                                PUBLIC  READ_LPEN
                                PUBLIC  SCROLL_DOWN
                                PUBLIC  SCROLL_UP
                                PUBLIC  SET_COLOR
                                PUBLIC  SET_CPOS
                                PUBLIC  SET_CTYPE
                                PUBLIC  SET_MODE
                                PUBLIC  WRITE_AC_CURRENT
                                PUBLIC  WRITE_C_CURRENT
                                PUBLIC  WRITE_DOT
                                PUBLIC  WRITE_TTY
                                PUBLIC  VIDEO_IO_1
                                PUBLIC  VIDEO_STATE

                                EXTRN   BEEP:NEAR               ; SPEAKER BEEP ROUTINE
                                EXTRN   CRT_CHAR_GEN:NEAR       ; CHARACTER GENERATOR GRAPHICS TABLE
                                EXTRN   DDS:NEAR                ; LOAD (DS) WITH DATA SEGMENT SELECTOR
                                EXTRN   M51:WORD                ; REGEN BUFFER LENGTH TABLE
                                EXTRN   M6:BYTE                 ; COLUMNS PER MODE TABLE
                                EXTRN   M7:BYTE                 ; MODE SET VALUE PER MODE TABLE

                        ;--- INT 10 H ----------------------------------------------------------------
                        ;                                                                           :
                        ; VIDEO_IO                                                                 :
                        ;       THESE ROUTINES PROVIDE THE CRT DISPLAY INTERFACE                    :
                        ;       THE FOLLOWING FUNCTIONS ARE PROVIDED:                               :
                        ;                                                                           :
                        ;   (AH)= 00H   SET MODE (AL) CONTAINS MODE VALUE                           :
                        ;               (AL) = 00H  40X25 BW MODE (POWER ON DEFAULT)                :
                        ;               (AL) = 01H  40X25 COLOR                                     :
                        ;               (AL) = 02H  80X25 BW                                        :
                        ;               (AL) = 03H  80X25 COLOR                                     :
                        ;                           GRAPHICS MODES                                  :
                        ;               (AL) = 04H  320X200 COLOR                                   :
                        ;               (AL) = 05H  320X200 BW MODE                                 :
                        ;               (AL) = 06H  640X200 BW MODE                                 :
                        ;               (AL) = 07H   80X25 MONOCHROME (USED INTERNAL TO VIDEO ONLY) :
                        ;               *** NOTES -BW MODES OPERATE SAME AS COLOR MODES, BUT COLOR   :
                        ;                          BURST IS NOT ENABLED                             :
                        ;                         -CURSOR IS NOT DISPLAYED IN GRAPHICS MODE         :
                        ;   (AH)= 01H   SET CURSOR TYPE                                            :
                        ;               (CH) =  BITS 4-0 = START LINE FOR CURSOR                    :
                        ;                       ** HARDWARE WILL ALWAYS CAUSE BLINK                 :
                        ;                       ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING   :
                        ;                          OR NO CURSOR AT ALL                              :
                        ;               (CL) =  BITS 4-0 = END LINE FOR CURSOR                      :
                        ;   (AH)= 02H   SET CURSOR POSITION                                        :
                        ;               (DH,DL) = ROW,COLUMN  (00H,00H) IS UPPER LEFT               :
                        ;               (BH) = PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)         :
                        ;   (AH)= 03H   READ CURSOR POSITION                                       :
                        ;               (BH) = PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)         :
                        ;               ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR              :
                        ;                       (CH,CL) = CURSOR MODE CURRENTLY SET                 :
                        ;   (AH)= 04H   READ LIGHT PEN POSITION                                    :
                        ;               ON EXIT:                                                    :
                        ;               (AH) = 00H -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED       :
                        ;               (AH) = 01H -- VALID LIGHT PEN VALUE IN REGISTERS            :
                        ;                       (DH,DL) = ROW,COLUMN OF CHARACTER LP POSITION       :
                        ;                       (CH) = RASTER LINE (0-199)                          :
                        ;                       (BX) = PIXEL COLUMN (0-319,639)                     :
                        ;   (AH)= 05H   SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES)     :
                        ;               (AL) = NEW PAGE VALUE (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3) :
                        ;   (AH)= 06H   SCROLL ACTIVE PAGE UP                                      :
                        ;               (AL) = NUMBER OF LINES, ( LINES BLANKED AT BOTTOM OF WINDOW ) :
                        ;                       (AL) = 00H MEANS BLANK ENTIRE WINDOW                :
                        ;               (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL         :
                        ;               (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL        :
                        ;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                   :
                        ;   (AH)= 07H   SCROLL ACTIVE PAGE DOWN                                    :
                        ;               (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP OF WINDOW :
                        ;                       (AL) = 00H MEANS BLANK ENTIRE WINDOW                :
                        ;               (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL         :
                        ;               (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL        :
                        ;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                   :
                        ;                                                                           :
                        ;   CHARACTER HANDLING ROUTINES                                             :
                        ;                                                                           :
                        ;   (AH)= 08H   READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION         :
                        ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)            :
                        ;               ON EXIT:                                                    :
                        ;               (AL) = CHAR READ                                            :
                        ;               (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)       :
                        ;   (AH)= 09H   WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION        :
                        ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)            :
                        ;               (CX) = COUNT OF CHARACTERS TO WRITE                         :
                        ;               (AL) = CHAR TO WRITE                                        :
                        ;               (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR (GRAPHICS):
                        ;                       SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.          :
                        ;   (AH)= 0AH   WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION             :
                        ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)            :
                        ;               (CX) = COUNT OF CHARACTERS TO WRITE                         :
                        ;               (AL) = CHAR TO WRITE                                        :
                        ;                       NOTE: USE FUNCTION (AH)= 09H IN GRAPHICS MODES      :
                        ;       FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE       :
                        ;               CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE       :
                        ;               MAINTAINED IN THE SYSTEM ROM.  ONLY THE 1ST 128 CHARS       :
                        ;               ARE CONTAINED THERE.  TO READ/WRITE THE SECOND 128 CHARS,   :
                        ;               THE USER MUST INITIALIZE THE POINTER AT INTERRUPT 1FH       :
                        ;               (LOCATION 0007CH) TO POINT TO THE 1K BYTE TABLE CONTAINING   :
                        ;               THE CODE POINTS FOR THE SECOND 128 CHARS (128-255).         :
                        ;       FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION FACTOR :
                        ;               CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID RESULTS ONLY   :
                        ;               FOR CHARACTERS CONTAINED ON THE SAME ROW.  CONTINUATION TO   :
                        ;               SUCCEEDING LINES WILL NOT PRODUCE CORRECTLY.                 :
                        ;                                                                           :
```

```
                        ;    GRAPHICS INTERFACE                                                              :
                        ;                                                                                    :
                        ;    (AH)= 0BH  SET COLOR PALETTE                                                    :
                        ;               (BH) = PALETTE COLOR ID BEING SET (0-127)                            :
                        ;               (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID                     :
                        ;                      NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT HAS        :
                        ;                            MEANING ONLY FOR 320X200 GRAPHICS.                      :
                        ;                      COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15)               :
                        ;                      COLOR ID = 1 SELECTS THE PALETTE TO BE USED:                   :
                        ;                                   0 = GREEN(1)/RED(2)/YELLOW(3)                    :
                        ;                                   1 = CYAN(1)/MAGENTA(2)/WHITE(3)                  :
                        ;                      IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET FOR              :
                        ;                            PALETTE COLOR 0 INDICATES THE BORDER COLOR              :
                        ;                            TO BE USED (VALUES 0-31, WHERE 16-31 SELECT             :
                        ;                            THE HIGH INTENSITY BACKGROUND SET.                      :
                        ;    (AH)= 0CH  WRITE DOT                                                            :
                        ;               (DX) = ROW NUMBER                                                    :
                        ;               (CX) = COLUMN NUMBER                                                 :
                        ;               (AL) = COLOR VALUE                                                   :
                        ;                      IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS EXCLUSIVE         :
                        ;                      ORed WITH THE CURRENT CONTENTS OF THE DOT                      :
                        ;    (AH)= 0DH  READ DOT                                                             :
                        ;               (DX) = ROW NUMBER                                                    :
                        ;               (CX) = COLUMN NUMBER                                                 :
                        ;               (AL) RETURNS THE DOT READ                                            :
                        ;                                                                                    :
                        ;    ASCII TELETYPE ROUTINE FOR OUTPUT                                               :
                        ;                                                                                    :
                        ;    (AH)= 0EH  WRITE TELETYPE TO ACTIVE PAGE                                        :
                        ;               (AL) = CHAR TO WRITE                                                 :
                        ;               (BL) = FOREGROUND COLOR IN GRAPHICS MODE                             :
                        ;               NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET              :
                        ;    (AH)= 0FH  CURRENT VIDEO STATE                                                  :
                        ;               RETURNS THE CURRENT VIDEO STATE                                      :
                        ;               (AL) = MODE CURRENTLY SET ( SEE (AH)= 00H FOR EXPLANATION)           :
                        ;               (AH) = NUMBER OF CHARACTER COLUMNS ON SCREEN                         :
                        ;               (BH) = CURRENT ACTIVE DISPLAY PAGE                                   :
                        ;    (AH)= 10H  RESERVED                                                             :
                        ;    (AH)= 11H  RESERVED                                                             :
                        ;    (AH)= 12H  RESERVED                                                             :
                        ;    (AH)= 13H  WRITE STRING                                                         :
                        ;                          ES:BP  -  POINTER TO STRING TO BE WRITTEN                 :
                        ;                          CX     -  LENGTH OF CHARACTER STRING TO WRITTEN           :
                        ;                          DX     -  CURSOR POSITION FOR STRING TO BE WRITTEN        :
                        ;                          BH     -  PAGE NUMBER                                     :
                        ;                 (AL)= 00H  WRITE CHARACTER STRING                                  :
                        ;                            BL     -  ATTRIBUTE                                     :
                        ;                            STRING IS  <CHAR,CHAR, ... ,CHAR>                       :
                        ;                            CURSOR NOT MOVED                                        :
                        ;                 (AL)= 01H  WRITE CHARACTER STRING AND MOVE CURSOR                  :
                        ;                            BL     -  ATTRIBUTE                                     :
                        ;                            STRING IS  <CHAR,CHAR, ... ,CHAR>                       :
                        ;                            CURSOR IS MOVED                                         :
                        ;                 (AL)= 02H  WRITE CHARACTER AND ATTRIBUTE STRING                    :
                        ;                                  (VALID FOR ALPHA MODES ONLY)                      :
                        ;                            STRING IS  <CHAR,ATTR,CHAR,ATTR ..  ,CHAR,ATTR>         :
                        ;                            CURSOR IS NOT MOVED                                     :
                        ;                 (AL)= 03H  WRITE CHARACTER AND ATTRIBUTE STRING AND MOVE CURSOR    :
                        ;                                  (VALID FOR ALPHA MODES ONLY)                      :
                        ;                            STRING IS  <CHAR,ATTR,CHAR,ATTR ..  ,CHAR,ATTR>         :
                        ;                            CURSOR IS MOVED                                         :
                        ;                   NOTE:  CARRIAGE RETURN, LINE FEED, BACKSPACE, AND BELL ARE       :
                        ;                          TREATED AS COMMANDS RATHER THAN PRINTABLE CHARACTERS.     :
                        ;                                                                                    :
                        ;       BX,CX,DX,SI,DI,BP,SP,DS,ES,SS PRESERVED DURING CALLS EXCEPT FOR              :
                        ;       BX,CX,DX RETURN VALUES ON FUNCTIONS 03H,04H,0DH AND 0DH. ON ALL CALLS        :
                        ;       AX IS MODIFIED.                                                              :
                        ;-------------------------------------------------------------------------------------

                                ASSUME  CS:CODE,DS:DATA,ES:NOTHING

0000 005F R             MI      DW      OFFSET  SET_MODE          ;  TABLE OF ROUTINES WITHIN VIDEO I/O
0002 0146 R                     DW      OFFSET  SET_CTYPE
0004 016B R                     DW      OFFSET  SET_CPOS
0006 0193 R                     DW      OFFSET  READ_CURSOR
0008 0789 R                     DW      OFFSET  READ_LPEN
000A 01AA R                     DW      OFFSET  ACT_DISP_PAGE
000C 0213 R                     DW      OFFSET  SCROLL_UP
000E 02B1 R                     DW      OFFSET  SCROLL_DOWN
0010 0303 R                     DW      OFFSET  READ_AC_CURRENT
0012 0360 R                     DW      OFFSET  WRITE_AC_CURRENT
0014 0392 R                     DW      OFFSET  WRITE_C_CURRENT
0016 01CC R                     DW      OFFSET  SET_COLOR
0018 0454 R                     DW      OFFSET  WRITE_DOT
001A 0443 R                     DW      OFFSET  READ_DOT
001C 0702 R                     DW      OFFSET  WRITE_TTY
001E 01F2 R                     DW      OFFSET  VIDEO_STATE
0020 013D R                     DW      OFFSET  VIDEO_RETURN    ; RESERVED
0022 013D R                     DW      OFFSET  VIDEO_RETURN    ; RESERVED
0024 013D R                     DW      OFFSET  VIDEO_RETURN    ; RESERVED
0026 03BF R                     DW      OFFSET  WRITE_STRING    ; CASE 13H, WRITE STRING
= 0028                  MIL     EQU     $-MI

0028                    VIDEO_IO_1      PROC    NEAR            ;       ENTRY POINT FOR ORG 0F065H
0028 FB                         STI                             ; INTERRUPTS BACK ON
0029 FC                         CLD                             ; SET DIRECTION FORWARD
002A 80 FC 14                   CMP     AH,MIL/2                ; TEST FOR WITHIN TABLE RANGE
002D 73 2F                      JNB     M4                      ; BRANCH TO EXIT IF NOT A VALID COMMAND

002F 06                         PUSH    ES
0030 1E                         PUSH    DS                      ; SAVE WORK AND PARAMETER REGISTERS
0031 52                         PUSH    DX
0032 51                         PUSH    CX
0033 53                         PUSH    BX
0034 56                         PUSH    SI
0035 57                         PUSH    DI
0036 55                         PUSH    BP
0037 BE ---- R                  MOV     SI,DATA                 ; POINT DS: TO DATA SEGMENT
003A 8E DE                      MOV     DS,SI
003C 8B F0                      MOV     SI,AX                   ; SAVE COMMAND/DATA INTO (SI) REGISTER
003E A0 0010 R                  MOV     AL,BYTE PTR @EQUIP_FLAG ; GET THE EQUIPMENT FLAG VIDEO BITS
0041 24 30                      AND     AL,30H                  ; ISOLATE CRT SWITCHES
0043 3C 30                      CMP     AL,30H                  ; IS SETTING FOR MONOCHROME CARD?
0045 BF B800                    MOV     DI,0B800H               ; GET SEGMENT FOR COLOR CARD
```

SECTION 5

```
0048 75 03                      JNE     M2                      ; SKIP IF NOT MONOCHROME CARD
004A BF B000                    MOV     DI,0B000H               ; ELSE GET SEGMENT FOR MONOCHROME CARD
004D                    M2:
004D 8E C7                      MOV     ES,DI                   ; SET UP TO POINT AT VIDEO MEMORY AREAS
004F 8A C4                      MOV     AL,AH                   ; PLACE COMMAND IN LOW BYTE OF (AX)
0051 98                         CBW                             ; AND FORM BYTE OFFSET WITH COMMAND
0052 D1 E0                      SAL     AX,1                    ; TIMES 2 FOR WORD TABLE LOOKUP
0054 96                         XCHG    SI,AX                   ; MOVE OFFSET INTO LOOK UP REGISTER (SI)
                                                                ;  AND RESTORE COMMAND/DATA INTO (AX)
0055 8A 26 0049 R               MOV     AH,@CRT_MODE            ; MOVE CURRENT MODE INTO (AH) REGISTER
0059 2E: FF A4 0000 R           JMP     WORD PTR CS:[SI+OFFSET M1]      ; GO TO SELECTED FUNCTION
005E                    M4:                                     ;       COMMAND NOT VALID
005E CF                         IRET                            ; DO NOTHING IF NOT IN VALID RANGE
005F                    VIDEO_IO_1      ENDP
                        ;--------------------------------------------------------
                        ; SET_MODE                                             :
                        ;       THIS ROUTINE INITIALIZES THE ATTACHMENT TO     :
                        ;       THE SELECTED MODE.  THE SCREEN IS BLANKED.     :
                        ; INPUT                                                 :
                        ;       @EQUIP_FLAG BITS 5-4 = MODE/WIDTH              :
                        ;           11 = MONOCHROME (FORCES MODE 7)            :
                        ;           01 = COLOR ADAPTER 40x25 (MODE 0 DEFAULT)  :
                        ;           10 = COLOR ADAPTER 80x25 (MODE 2 DEFAULT)  :
                        ;         (AL) = COLOR MODE REQUESTED ( RANGE  0 - 6 )  :
                        ; OUTPUT                                                :
                        ;       NONE                                           :
                        ;--------------------------------------------------------
005F                    SET_MODE        PROC    NEAR
005F BA 03D4                    MOV     DX,03D4H                ; ADDRESS OF COLOR CARD
0062 8B 3E 0010 R               MOV     DI,@EQUIP_FLAG          ; GET EQUIPMENT FLAGS SETTING
0066 81 E7 0030                 AND     DI,30H                  ; ISOLATE CRT SWITCHES
006A 83 FF 30                   CMP     DI,30H                  ; IS BW CARD INSTALLED AS PRIMARY
006D 75 06                      JNE     M8C                     ; SKIP AND CHECK IF COLOR
006F B0 07                      MOV     AL,7                    ; ELSE INDICATE INTERNAL BW CARD MODE
0071 B2 B4                      MOV     DL,0B4H                 ; SET ADDRESS OF BW (MONOCHROME) CARD
0073 EB 0D                      JMP     SHORT M8                ; CONTINUE WITH FORCED MODE 7
0075                    M8C:
0075 3C 07                      CMP     AL,7                    ; CHECK FOR VALID COLOR MODES 0-6
0077 72 09                      JB      M8                      ; CONTINUE IF BELOW MODE 7
0079 B0 00                      MOV     AL,0                    ; FORCE DEFAULT 40x25 BW MODE
007B 83 FF 10                   CMP     DI,10H                  ; CHECK FOR @EQUIP_FLAG AT 40x25 COLOR
007E 74 02                      JE      M8                      ; CONTINUE WITH MODE 0 IF NOT
0080 B0 02                      MOV     AL,2                    ; ELSE FORCE MODE 2
0082                    M8:
0082 A2 0049 R                  MOV     @CRT_MODE,AL            ; SAVE MODE IN GLOBAL VARIABLE
0085 89 16 0063 R               MOV     @ADDR_6845,DX           ; SAVE ADDRESS OF BASE
0089 C6 06 0084 R 18            MOV     @ROWS,25-1              ; INITIALIZE DEFAULT ROW COUNT OF 25
008E 1E                         PUSH    DS                      ; SAVE POINTER TO DATA SEGMENT
008F 50                         PUSH    AX                      ; SAVE MODE NUMBER (AL)
0090 98                         CBW                             ; CLEAR HIGH BYTE OF MODE
0091 8B F0                      MOV     SI,AX                   ; SET TABLE POINTER, INDEXED BY MODE
0093 2E: 8A 84 0000 E           MOV     AL,CS:[SI + OFFSET M7]  ; GET THE MODE SET VALUE FROM TABLE
0098 A2 0065 R                  MOV     @CRT_MODE_SET.AL        ; SAVE THE MODE SET VALUE
009B 24 37                      AND     AL,037H                 ; VIDEO OFF, SAVE HIGH RESOLUTION BIT
009D 52                         PUSH    DX                      ; SAVE OUTPUT PORT VALUE
009E 83 C2 04                   ADD     DX,4                    ; POINT TO CONTROL REGISTER
00A1 EE                         OUT     DX,AL                   ; RESET VIDEO TO OFF TO SUPPRESS ROLLING
00A2 5A                         POP     DX                      ; BACK TO BASE REGISTER
                                ASSUME  DS:ABS0
00A3 2B DB                      SUB     BX,BX                   ; SET UP FOR ABS0 SEGMENT
00A5 8E DB                      MOV     DS,BX                   ; ESTABLISH VECTOR TABLE ADDRESSING
00A7 C5 1E 0074 R               LDS     BX,@PARM_PTR            ; GET POINTER TO VIDEO PARMS
                                ASSUME  DS:CODE
00AB 58                         POP     AX                      ; RECOVER MODE NUMBER IN (AL)
00AC B9 0010                    MOV     CX,16                   ; LENGTH OF EACH ROW OF TABLE
00AF 3C 02                      CMP     AL,2                    ; DETERMINE WHICH ONE TO USE
00B1 72 0E                      JC      M9                      ; MODE IS 0 OR 1
00B3 03 D9                      ADD     BX,CX                   ; NEXT ROW OF INITIALIZATION TABLE
00B5 3C 04                      CMP     AL,4
00B7 72 08                      JC      M9                      ; MODE IS 2 OR 3
00B9 03 D9                      ADD     BX,CX                   ; MOVE TO GRAPHICS ROW OF INIT_TABLE
00BB 3C 07                      CMP     AL,7
00BD 72 02                      JC      M9                      ; MODE IS 4,5, OR 6
00BF 03 D9                      ADD     BX,CX                   ; MOVE TO BW CARD ROW OF INIT_TABLE

                        ;-----  BX POINTS TO CORRECT ROW OF INITIALIZATION TABLE

00C1                    M9:                                     ; OUT_INIT
00C1 50                         PUSH    AX                      ; SAVE MODE IN (AL)
00C2 8B 47 0A                   MOV     AX,[BX+10]              ; GET THE CURSOR MODE FROM THE TABLE
00C5 86 E0                      XCHG    AH,AL                   ; PUT CURSOR MODE IN CORRECT POSITION
00C7 1E                         PUSH    DS                      ; SAVE TABLE SEGMENT POINTER
                                ASSUME  DS:DATA
00C8 E8 0000 E                  CALL    DDS                     ; POINT DS TO DATA SEGMENT
00CB A3 0060 R                  MOV     @CURSOR_MODE,AX         ; PLACE INTO BIOS DATA SAVE AREA
                                ASSUME  DS:CODE
00CE 1F                         POP     DS                      ; RESTORE THE TABLE SEGMENT POINTER
00CF 32 E4                      XOR     AH,AH                   ; AH IS REGISTER NUMBER DURING LOOP

                        ;-----  LOOP THROUGH TABLE, OUTPUTTING REGISTER ADDRESS, THEN VALUE FROM TABLE

00D1                    M10:                                    ;  INITIALIZATION LOOP
00D1 8A C4                      MOV     AL,AH                   ; GET 6845 REGISTER NUMBER
00D3 EE                         OUT     DX,AL
00D4 42                         INC     DX                      ; POINT TO DATA PORT
00D5 FE C4                      INC     AH                      ; NEXT REGISTER VALUE
00D7 8A 07                      MOV     AL,[BX]                 ; GET TABLE VALUE
00D9 EE                         OUT     DX,AL                   ; OUT TO CHIP
00DA 43                         INC     BX                      ; NEXT IN TABLE
00DB 4A                         DEC     DX                      ; BACK TO POINTER REGISTER
00DC E2 F3                      LOOP    M10                     ; DO THE WHOLE TABLE
00DE 58                         POP     AX                      ; GET MODE BACK INTO (AL)
00DF 1F                         POP     DS                      ; RECOVER SEGMENT VALUE
                                ASSUME  DS:DATA

                        ;-----  FILL REGEN AREA WITH BLANK

00E0 33 FF                      XOR     DI,DI                   ; SET UP POINTER FOR REGEN
00E2 89 3E 004E R               MOV     @CRT_START,DI           ; START ADDRESS SAVED IN GLOBAL
00E6 C6 06 0062 R 00            MOV     @ACTIVE_PAGE,0          ; SET PAGE VALUE
00EB B9 2000                    MOV     CX,8192                 ; NUMBER OF WORDS IN COLOR CARD
00EE 3C 04                      CMP     AL,4                    ; TEST FOR GRAPHICS
```

```
                        JC      M12                     ; NO_GRAPHICS_INIT
                        CMP     AL,7                    ; TEST FOR BW CARD
                        JE      M11                     ; BW_CARD_INIT
                        XOR     AX,AX                   ; FILL FOR GRAPHICS MODE
                        JMP     SHORT M13               ; CLEAR_BUFFER
                M11:                                    ; BW_CARD_INIT
                        MOV     CH,08H                  ; BUFFER SIZE ON BW CARD (2048)
                M12:                                    ; NO_GRAPHICS_INIT
                        MOV     AX,' '+7*H              ; FILL CHAR FOR ALPHA + ATTRIBUTE
                M13:                                    ; CLEAR_BUFFER
                        REP     STOSW                   ; FILL THE REGEN BUFFER WITH BLANKS

                ;-----  ENABLE VIDEO AND CORRECT PORT SETTING

                        MOV     DX,@ADDR_6845           ; PREPARE TO OUTPUT TO VIDEO ENABLE PORT
                        ADD     DX,4                    ; POINT TO THE MODE CONTROL REGISTER
                        MOV     AL,@CRT_MODE_SET        ; GET THE MODE SET VALUE
                        OUT     DX,AL                   ; SET VIDEO ENABLE PORT

                ;-----  DETERMINE NUMBER OF COLUMNS, BOTH FOR ENTIRE DISPLAY
                ;-----  AND THE NUMBER TO BE USED FOR TTY INTERFACE

                        MOV     AL,CS:[SI + OFFSET M6]  ; GET NUMBER OF COLUMNS ON THIS SCREEN
                        CBW                             ; CLEAR HIGH BYTE
                        MOV     @CRT_COLS,AX            ; INITIALIZE NUMBER OF COLUMNS COUNT

                ;-----  SET CURSOR POSITIONS

                        AND     SI,000EH                ; WORD OFFSET INTO CLEAR LENGTH TABLE
                        MOV     AX,CS:[SI + OFFSET M5]  ; LENGTH TO CLEAR
                        MOV     @CRT_LEN,AX             ; SAVE LENGTH OF CRT -- NOT USED FOR BW
                        MOV     CX,8                    ; CLEAR ALL CURSOR POSITIONS
                        MOV     DI,OFFSET @CURSOR_POSN
                        PUSH    DS                      ; ESTABLISH SEGMENT
                        POP     ES                      ;   ADDRESSING
                        XOR     AX,AX
                        REP     STOSW                   ; FILL WITH ZEROES

                ;-----  SET UP OVERSCAN REGISTER

                        INC     DX                      ; SET OVERSCAN PORT TO A DEFAULT
                        MOV     AL,30H                  ; 30H VALUE FOR ALL MODES EXCEPT 640X200
                        CMP     @CRT_MODE,6             ; SEE IF THE MODE IS 640X200 BW
                        JNZ     M14                     ; IF NOT 640X200, THEN GO TO REGULAR
                        MOV     AL,3FH                  ; IF IT IS 640X200, THEN PUT IN 3FH
                M14:
                        OUT     DX,AL                   ; OUTPUT THE CORRECT VALUE TO 3D9 PORT
                        MOV     @CRT_PALETTE,AL         ; SAVE THE VALUE FOR FUTURE USE

                ;-----  NORMAL RETURN FROM ALL VIDEO RETURNS

                VIDEO_RETURN:
                        POP     BP
                        POP     DI
                        POP     SI
                        POP     BX
                M15:                                    ; VIDEO_RETURN_C
                        POP     CX
                        POP     DX
                        POP     DS
                        POP     ES                      ; RECOVER SEGMENTS
                        IRET                            ; ALL DONE
                SET_MODE        ENDP
                ;--------------------------------------------------
                ; SET_CTYPE
                ;       THIS ROUTINE SETS THE CURSOR VALUE
                ; INPUT
                ;       (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
                ; OUTPUT
                ;       NONE
                ;--------------------------------------------------------
                SET_CTYPE       PROC    NEAR
                        MOV     AH,10                   ; 6845 REGISTER FOR CURSOR SET
                        MOV     @CURSOR_MODE,CX         ; SAVE IN DATA AREA
                        CALL    M16                     ; OUTPUT CX REGISTER
                        JMP     VIDEO_RETURN

                ;-----  THIS ROUTINE OUTPUTS THE CX REGISTER TO THE 6845 REGISTERS NAMED IN (AH)


                M16:
                        MOV     DX,@ADDR_6845           ; ADDRESS REGISTER
                        MOV     AL,AH                   ; GET VALUE
                        OUT     DX,AL                   ; REGISTER SET
                        INC     DX                      ; DATA REGISTER
                        JMP     $+2                     ; I/O DELAY
                        MOV     AL,CH                   ; DATA
                        OUT     DX,AL
                        DEC     DX
                        MOV     AL,AH
                        INC     AL                      ; POINT TO OTHER DATA REGISTER
                        OUT     DX,AL                   ; SET FOR SECOND REGISTER
                        INC     DX
                        JMP     $+2                     ; I/O DELAY
                        MOV     AL,CL                   ; SECOND DATA VALUE
                        OUT     DX,AL
                        RET                             ; ALL DONE
                SET_CTYPE       ENDP

                ;--------------------------------------------
                ; SET_CPOS
                ;       THIS ROUTINE SETS THE CURRENT CURSOR POSITION TO THE
                ;       NEW X-Y VALUES PASSED
                ; INPUT
                ;       DX - ROW,COLUMN OF NEW CURSOR
                ;       BH - DISPLAY PAGE OF CURSOR
                ; OUTPUT
                ;       CURSOR IS SET AT 6845 IF DISPLAY PAGE IS CURRENT DISPLAY
                ;--------------------------------------------
                SET_CPOS        PROC    NEAR
                        MOV     AL,BH                   ; MOVE PAGE NUMBER TO WORK REGISTER
                        CBW                             ; CONVERT PAGE TO WORD VALUE
                        SAL     AX,1                    ; WORD OFFSET
                        XCHG    AX,SI                   ; USE INDEX REGISTER
```

```
0171 89 94 0050 R               MOV     [SI+OFFSET @CURSOR_POSN],DX     ; SAVE THE POINTER
0175 38 3E 0062 R               CMP     @ACTIVE_PAGE,BH
0179 75 05                      JNZ     M17                     ; SET_CPOS_RETURN
017B 8B C2                      MOV     AX,DX                   ; GET ROW/COLUMN TO AX
017D E8 0182 R                  CALL    M18                     ; CURSOR_SET
0180                    M17:                                    ; SET_CPOS_RETURN
0180 EB BB                      JMP     VIDEO_RETURN
0182                    SET_CPOS        ENDP

                        ;-----  SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR

0182                    M18     PROC    NEAR
0182 E8 0204 R                  CALL    POSITION                ; DETERMINE LOCATION IN REGEN BUFFER
0185 8B C8                      MOV     CX,AX
0187 03 0E 004E R               ADD     CX,@CRT_START           ; ADD IN THE START ADDRESS FOR THIS PAGE
018B D1 F9                      SAR     CX,1                    ; DIVIDE BY 2 FOR CHAR ONLY COUNT
018D B4 0E                      MOV     AH,14                   ; REGISTER NUMBER FOR CURSOR
018F E8 0151 R                  CALL    M16                     ; OUTPUT THE VALUE TO THE 6845
0192 C3                         RET
0193                    M18     ENDP
                        ;--------------------------------------------------
                        ; READ_CURSOR
                        ;       THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE
                        ;       6845, FORMATS IT, AND SENDS IT BACK TO THE CALLER
                        ; INPUT
                        ;       BH - PAGE OF CURSOR
                        ; OUTPUT
                        ;       DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION
                        ;       CX - CURRENT CURSOR MODE
                        ;--------------------------------------------
0193                    READ_CURSOR     PROC    NEAR
0193 8A DF                      MOV     BL,BH
0195 32 FF                      XOR     BH,BH
0197 D1 E3                      SAL     BX,1                    ; WORD OFFSET
0199 8B 97 0050 R               MOV     DX,[BX+OFFSET @CURSOR_POSN]
019D 8B 0E 0060 R               MOV     CX,@CURSOR_MODE
01A1 5D                         POP     BP
01A2 5F                         POP     DI
01A3 5E                         POP     SI
01A4 5B                         POP     BX
01A5 58                         POP     AX                      ; DISCARD SAVED CX AND DX
01A6 58                         POP     AX
01A7 1F                         POP     DS
01A8 07                         POP     ES
01A9 CF                         IRET
01AA                    READ_CURSOR     ENDP
                        ;--------------------------------------------------
                        ; ACT_DISP_PAGE
                        ;       THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
                        ;       THE FULL USE OF THE MEMORY SET ASIDE FOR THE VIDEO ATTACHMENT
                        ; INPUT
                        ;       AL HAS THE NEW ACTIVE DISPLAY PAGE
                        ; OUTPUT
                        ;       THE 6845 IS RESET TO DISPLAY THAT PAGE
                        ;--------------------------------------------
01AA                    ACT_DISP_PAGE   PROC    NEAR
01AA A2 0062 R                  MOV     @ACTIVE_PAGE,AL         ; SAVE ACTIVE PAGE VALUE
01AD 98                         CBW                             ; CONVERT (AL) TO WORD
01AE 50                         PUSH    AX                      ; SAVE PAGE VALUE
01AF F7 26 004C R               MUL     WORD PTR @CRT_LEN       ; DISPLAY PAGE TIMES REGEN LENGTH
01B3 A3 004E R                  MOV     @CRT_START,AX           ; SAVE START ADDRESS FOR LATER
01B6 8B C8                      MOV     CX,AX                   ; START ADDRESS TO CX
01B8 D1 F9                      SAR     CX,1                    ; DIVIDE BY 2 FOR 6845 HANDLING
01BA B4 0C                      MOV     AH,12                   ; 6845 REGISTER FOR START ADDRESS
01BC E8 0151 R                  CALL    M16
01BF 5B                         POP     BX                      ; RECOVER PAGE VALUE
01C0 D1 E3                      SAL     BX,1                    ; *2 FOR WORD OFFSET
01C2 8B 87 0050 R               MOV     AX,[BX + OFFSET @CURSOR_POSN]   ; GET CURSOR FOR THIS PAGE
01C6 E8 0182 R                  CALL    M18                     ; SET THE CURSOR POSITION
01C9 E9 013D R                  JMP     VIDEO_RETURN
01CC                    ACT_DISP_PAGE   ENDP
                        ;--------------------------------------------
                        ; SET_COLOR
                        ;       THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
                        ;       AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
                        ; INPUT
                        ;       (BH) HAS COLOR ID
                        ;               IF BH=0, THE BACKGROUND COLOR VALUE IS SET
                        ;                       FROM THE LOW BITS OF BL (0-31)
                        ;               IF BH=1, THE PALETTE SELECTION IS MADE
                        ;                       BASED ON THE LOW BIT OF BL:
                        ;                               0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
                        ;                               1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
                        ;       (BL) HAS THE COLOR VALUE TO BE USED
                        ; OUTPUT
                        ;       THE COLOR SELECTION IS UPDATED
                        ;--------------------------------------------
01CC                    SET_COLOR       PROC    NEAR
01CC 8B 16 0063 R               MOV     DX,@ADDR_6845           ; I/O PORT FOR PALETTE
01D0 83 C2 05                   ADD     DX,5                    ; OVERSCAN PORT
01D3 A0 0066 R                  MOV     AL,@CRT_PALETTE         ; GET THE CURRENT PALETTE VALUE
01D6 0A FF                      OR      BH,BH                   ; IS THIS COLOR 0?
01D8 75 0E                      JNZ     M20                     ; OUTPUT COLOR 1

                        ;-----  HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR

01DA 24 E0                      AND     AL,0E0H                 ; TURN OFF LOW 5 BITS OF CURRENT
01DC 80 E3 1F                   AND     BL,01FH                 ; TURN OFF HIGH 3 BITS OF INPUT VALUE
01DF 0A C3                      OR      AL,BL                   ; PUT VALUE INTO REGISTER
01E1                    M19:                                    ; OUTPUT THE PALETTE
01E1 EE                         OUT     DX,AL                   ; OUTPUT COLOR SELECTION TO 3D9 PORT
01E2 A2 0066 R                  MOV     @CRT_PALETTE,AL         ; SAVE THE COLOR VALUE
01E5 E9 013D R                  JMP     VIDEO_RETURN

                        ;-----  HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED

01E8                    M20:
01E8 24 DF                      AND     AL,0DFH                 ; TURN OFF PALETTE SELECT BIT
01EA D0 EB                      SHR     BL,1                    ; TEST THE LOW ORDER BIT OF BL
01EC 73 F3                      JNC     M19                     ; ALREADY DONE
01EE 0C 20                      OR      AL,20H                  ; TURN ON PALETTE SELECT BIT
01F0 EB EF                      JMP     M19                     ; GO DO IT
01F2                    SET_COLOR       ENDP
```

```
                        ;--------------------------------------------------
                        ; VIDEO STATE
                        ;  RETURNS THE CURRENT VIDEO STATE IN AX
                        ;  AH = NUMBER OF COLUMNS ON THE SCREEN
                        ;  AL = CURRENT VIDEO MODE
                        ;  BH = CURRENT ACTIVE PAGE
                        ;--------------------------------------------------
01F2                    VIDEO_STATE     PROC    NEAR
01F2 8A 26 004A R               MOV     AH,BYTE PTR @CRT_COLS   ; GET NUMBER OF COLUMNS
01F6 A0 0049 R                  MOV     AL,@CRT_MODE            ; CURRENT MODE
01F9 8A 3E 0062 R               MOV     BH,@ACTIVE_PAGE         ; GET CURRENT ACTIVE PAGE
01FD 5D                         POP     BP                      ; RECOVER REGISTERS
01FE 5F                         POP     DI
01FF 5E                         POP     SI
0200 59                         POP     CX                      ; DISCARD SAVED BX
0201 E9 0141 R                  JMP     M15                     ; RETURN TO CALLER
0204                    VIDEO_STATE     ENDP
                        ;--------------------------------------
                        ; POSITION
                        ;       THIS SERVICE ROUTINE CALCULATES THE REGEN BUFFER ADDRESS
                        ;       OF A CHARACTER IN THE ALPHA MODE
                        ; INPUT
                        ;       AX = ROW, COLUMN POSITION
                        ; OUTPUT
                        ;       AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
                        ;--------------------------------------
0204                    POSITION        PROC    NEAR
0204 53                         PUSH    BX                      ; SAVE REGISTER
0205 93                         XCHG    BX,AX                   ; SAVE ROW/COLUNM POSITION IN (BX)
0206 A0 004A R                  MOV     AL,BYTE PTR @CRT_COLS   ; GET COLUMNS PER ROW COUNT
0209 F6 E7                      MUL     BH                      ; DETERMINE BYTES TO ROW
020B 32 FF                      XOR     BH,BH
020D 03 C3                      ADD     AX,BX                   ; ADD IN COLUMN VALUE
020F D1 E0                      SAL     AX,1                    ; * 2 FOR ATTRIBUTE BYTES
0211 5B                         POP     BX
0212 C3                         RET
0213                    POSITION        ENDP
                        ;--------------------------------------------
                        ; SCROLL UP
                        ;       THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP
                        ;       ON THE SCREEN
                        ; INPUT
                        ;       (AH) = CURRENT CRT MODE
                        ;       (AL) = NUMBER OF ROWS TO SCROLL
                        ;       (CX) = ROW/COLUMN OF UPPER LEFT CORNER
                        ;       (DX) = ROW/COLUMN OF LOWER RIGHT CORNER
                        ;       (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE
                        ;       (DS) = DATA SEGMENT
                        ;       (ES) = REGEN BUFFER SEGMENT
                        ; OUTPUT
                        ;       NONE -- THE REGEN BUFFER IS MODIFIED
                        ;--------------------------------------------
                                ASSUME  DS:DATA,ES:DATA
0213                    SCROLL_UP       PROC    NEAR

0213 E8 02EE R                  CALL    TEST_LINE_COUNT
0216 80 FC 04                   CMP     AH,4
0219 72 08                      JC      N1                      ; HANDLE SEPARATELY
021B 80 FC 07                   CMP     AH,7                    ; TEST FOR BW CARD
021E 74 03                      JE      N1
0220 E9 04B0 R                  JMP     GRAPHICS_UP
0223                    N1:                                     ; UP_CONTINUE
0223 53                         PUSH    BX                      ; SAVE FILL ATTRIBUTE IN BH
0224 8B C1                      MOV     AX,CX                   ; UPPER LEFT POSITION
0226 E8 0260 R                  CALL    SCROLL_POSITION         ; DO SETUP FOR SCROLL
0229 74 31                      JZ      N7                      ; BLANK_FIELD
022B 03 F0                      ADD     SI,AX                   ; FROM ADDRESS
022D 8A E6                      MOV     AH,DH                   ; # ROWS IN BLOCK
022F 2A E3                      SUB     AH,BL                   ; # ROWS TO BE MOVED
0231                    N2:                                     ; ROW_LOOP
0231 E8 02A1 R                  CALL    N10                     ; MOVE ONE ROW
0234 03 F5                      ADD     SI,BP
0236 03 FD                      ADD     DI,BP                   ; POINT TO NEXT LINE IN BLOCK
0238 FE CC                      DEC     AH                      ; COUNT OF LINES TO MOVE
023A 75 F5                      JNZ     N2                      ; ROW_LOOP
023C                    N3:                                     ; CLEAR_ENTRY
023C 58                         POP     AX                      ; RECOVER ATTRIBUTE IN AH
023D B0 20                      MOV     AL,' '                  ; FILL WITH BLANKS
023F                    N4:                                     ; CLEAR_LOOP
023F E8 02AA R                  CALL    N11                     ; CLEAR THE ROW
0242 03 FD                      ADD     DI,BP                   ; POINT TO NEXT LINE
0244 FE CB                      DEC     BL                      ; COUNTER OF LINES TO SCROLL
0246 75 F7                      JNZ     N4                      ; CLEAR_LOOP
0248                    N5:                                     ; SCROLL_END
0248 E8 0000 E                  CALL    DDS
024B 80 3E 0049 R 07            CMP     @CRT_MODE,7             ; IS THIS THE BLACK AND WHITE CARD
0250 74 07                      JE      N6                      ; IF SO, SKIP THE MODE RESET
0252 A0 0065 R                  MOV     AL,@CRT_MODE_SET        ; GET THE VALUE OF THE MODE SET
0255 BA 03D8                    MOV     DX,03D8H                ; ALWAYS SET COLOR CARD PORT
0258 EE                         OUT     DX,AL
0259                    N6:                                     ; VIDEO_RET_HERE
0259 E9 013D R                  JMP     VIDEO_RETURN
025C                    N7:                                     ; BLANK_FIELD
025C 8A DE                      MOV     BL,DH                   ; GET ROW COUNT
025E EB DC                      JMP     N3                      ; GO CLEAR THAT AREA
0260                    SCROLL_UP       ENDP

                        ;-----  HANDLE COMMON SCROLL SET UP HERE

0260                    SCROLL_POSITION PROC    NEAR
0260 E8 0204 R                  CALL    POSITION                ; CONVERT TO REGEN POINTER
0263 03 06 004E R               ADD     AX,@CRT_START           ; OFFSET OF ACTIVE PAGE
0267 8B F8                      MOV     DI,AX                   ; TO ADDRESS FOR SCROLL
0269 8B F0                      MOV     SI,AX                   ; FROM ADDRESS FOR SCROLL
026B 2B D1                      SUB     DX,CX                   ; DX = #ROWS, #COLS IN BLOCK
026D FE C6                      INC     DH
026F FE C2                      INC     DL                      ; INCREMENT FOR 0 ORIGIN
0271 32 ED                      XOR     CH,CH                   ; SET HIGH BYTE OF COUNT TO ZERO
0273 8B 2E 004A R               MOV     BP,@CRT_COLS            ; GET NUMBER OF COLUMNS IN DISPLAY
0277 03 ED                      ADD     BP,BP                   ; TIMES 2 FOR ATTRIBUTE BYTE
0279 A0 004A R                  MOV     AL,BYTE PTR @CRT_COLS   ; GET CHARACTERS PER LINE COUNT
027C F6 E3                      MUL     BL                      ; DETERMINE OFFSET TO FROM ADDRESS
027E 03 C0                      ADD     AX,AX                   ; *2 FOR ATTRIBUTE BYTE
0280 50                         PUSH    AX                      ; SAVE LINE COUNT
```

SECTION 5

```
0281 A0 0049 R                  MOV     AL,@CRT_MODE            ; GET CURRENT MODE
0284 06                         PUSH    ES                      ; ESTABLISH ADDRESSING TO REGEN BUFFER
0285 1F                         POP     DS                      ;  FOR BOTH POINTERS
0286 3C 02                      CMP     AL,2                    ; TEST FOR COLOR CARD SPECIAL CASES HERE
0288 72 13                      JB      N9                      ; HAVE TO HANDLE 80X25 SEPARATELY
028A 3C 03                      CMP     AL,3
028C 77 0F                      JA      N9
                        ;-----                                  ; 80X25 COLOR CARD SCROLL
028E 52                         PUSH    DX
028F BA 03DA                    MOV     DX,3DAH                 ; GUARANTEED TO BE COLOR CARD HERE
0292                    N8:                                     ; WAIT_DISP_ENABLE
0292 EC                         IN      AL,DX                   ; GET PORT
0293 A8 08                      TEST    AL,RVRT                 ; WAIT FOR VERTICAL RETRACE
0295 74 FB                      JZ      N8                      ; WAIT_DISP_ENABLE
0297 B0 25                      MOV     AL,25H
0299 B2 D8                      MOV     DL,0D8H                 ; ADDRESS CONTROL PORT
029B EE                         OUT     DX,AL                   ; TURN OFF VIDEO DURING VERTICAL RETRACE
029C 5A                         POP     DX
029D                    N9:
029D 58                         POP     AX                      ; RESTORE LINE COUNT
029E 0A DB                      OR      BL,BL                   ; 0 SCROLL MEANS BLANK FIELD
02A0 C3                         RET                             ; RETURN WITH FLAGS SET
02A1                    SCROLL_POSITION ENDP

                        ;-----  MOVE_ROW
02A1                    N10     PROC    NEAR
02A1 8A CA                      MOV     CL,DL                   ; GET # OF COLS TO MOVE
02A3 56                         PUSH    SI
02A4 57                         PUSH    DI                      ; SAVE START ADDRESS
02A5 F3/ A5                     REP     MOVSW                   ; MOVE THAT LINE ON SCREEN
02A7 5F                         POP     DI
02A8 5E                         POP     SI                      ; RECOVER ADDRESSES
02A9 C3                         RET
02AA                    N10     ENDP

                        ;-----  CLEAR_ROW
02AA                    N11     PROC    NEAR
02AA 8A CA                      MOV     CL,DL                   ; GET # COLUMNS TO CLEAR
02AC 57                         PUSH    DI
02AD F3/ AB                     REP     STOSW                   ; STORE THE FILL CHARACTER
02AF 5F                         POP     DI
02B0 C3                         RET
02B1                    N11     ENDP
                        ;--------------------------------------
                        ; SCROLL_DOWN
                        ;       THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
                        ;       BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
                        ;       WITH A DEFINED CHARACTER
                        ; INPUT
                        ;       (AH) = CURRENT CRT MODE
                        ;       (AL) = NUMBER OF LINES TO SCROLL
                        ;       (CX) = UPPER LEFT CORNER OF REGION
                        ;       (DX) = LOWER RIGHT CORNER OF REGION
                        ;       (BH) = FILL CHARACTER
                        ;       (DS) = DATA SEGMENT
                        ;       (ES) = REGEN SEGMENT
                        ; OUTPUT
                        ;       NONE -- SCREEN IS SCROLLED
                        ;--------------------------------------
02B1                    SCROLL_DOWN     PROC    NEAR
02B1 FD                         STD                             ; DIRECTION FOR SCROLL DOWN
02B2 E8 02EE R                  CALL    TEST_LINE_COUNT         ;
02B5 80 FC 04                   CMP     AH,4                    ; TEST FOR GRAPHICS
02B8 72 08                      JC      N12
02BA 80 FC 07                   CMP     AH,7                    ; TEST FOR BW CARD
02BD 74 03                      JE      N12
02BF E9 0507 R                  JMP     GRAPHICS_DOWN
02C2                    N12:                                    ; CONTINUE_DOWN
02C2 53                         PUSH    BX                      ; SAVE ATTRIBUTE IN BH
02C3 8B C2                      MOV     AX,DX                   ; LOWER RIGHT CORNER
02C5 E8 0260 R                  CALL    SCROLL_POSITION         ; GET REGEN LOCATION
02C8 74 20                      JZ      N16
02CA 2B F0                      SUB     SI,AX                   ; SI IS FROM ADDRESS
02CC 8A E6                      MOV     AH,DH                   ; GET TOTAL # ROWS
02CE 2A E3                      SUB     AH,BL                   ; COUNT TO MOVE IN SCROLL
02D0                    N13:
02D0 E8 02A1 R                  CALL    N10                     ; MOVE ONE ROW
02D3 2B F5                      SUB     SI,BP
02D5 2B FD                      SUB     DI,BP
02D7 FE CC                      DEC     AH
02D9 75 F5                      JNZ     N13
02DB                    N14:
02DB 58                         POP     AX                      ; RECOVER ATTRIBUTE IN AH
02DC B0 20                      MOV     AL,' '
02DE                    N15:
02DE E8 02AA R                  CALL    N11                     ; CLEAR ONE ROW
02E1 2B FD                      SUB     DI,BP                   ; GO TO NEXT ROW
02E3 FE CB                      DEC     BL
02E5 75 F7                      JNZ     N15
02E7 E9 0248 R                  JMP     N5                      ; SCROLL_END
02EA                    N16:
02EA 8A DE                      MOV     BL,DH
02EC EB ED                      JMP     N14
02EE                    SCROLL_DOWN     ENDP

                        ;-----  IF  AMOUNT OF LINES TO BE SCROLLED = AMOUNT OF LINES IN WINDOW
                        ;-----    THEN  ADJUST AL; ELSE  RETURN;

02EE                    TEST_LINE_COUNT PROC    NEAR

02EE 8A D8                      MOV     BL,AL                   ; SAVE LINE COUNT IN BL
02F0 0A C0                      OR      AL,AL                   ; TEST IF AL IS ALREADY ZERO
02F2 74 0E                      JZ      BL_SET                  ; IF IT IS THEN RETURN...
02F4 50                         PUSH    AX                      ; SAVE AX
02F5 8A C6                      MOV     AL,DH                   ; SUBTRACT LOWER ROW FROM UPPER ROW
02F7 2A C5                      SUB     AL,CH
02F9 FE C0                      INC     AL                      ; ADJUST DIFFERENCE BY 1
02FB 3A C3                      CMP     AL,BL                   ; LINE COUNT = AMOUNT OF ROWS IN WINDOW?
02FD 58                         POP     AX                      ; RESTORE AX
02FE 75 02                      JNE     BL_SET                  ; IF NOT THEN WE'RE ALL SET
0300 2A DB                      SUB     BL,BL                   ; OTHERWISE SET BL TO ZERO
0302                    BL_SET:
0302 C3                         RET                             ; RETURN
0303                    TEST_LINE_COUNT ENDP
```

```
                        PAGE
                        ;--------------------------------------------------------------------------------
                        ; READ_AC_CURRENT                                                              :
                        ;       THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER AT THE CURRENT          :
                        ;       CURSOR POSITION AND RETURNS THEM TO THE CALLER                        :
                        ; INPUT                                                                       :
                        ;       (AH) = CURRENT CRT MODE                                               :
                        ;       (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )                              :
                        ;       (DS) = DATA SEGMENT                                                   :
                        ;       (ES) = REGEN SEGMENT                                                  :
                        ; OUTPUT                                                                      :
                        ;       (AL) = CHARACTER READ                                                 :
                        ;       (AH) = ATTRIBUTE READ                                                 :
                        ;--------------------------------------------------------------------------------
                                ASSUME  DS:DATA,ES:DATA

0303                    READ_AC_CURRENT PROC    NEAR
0303 80 FC 04                   CMP     AH,4                    ; IS THIS GRAPHICS
0306 72 08                      JC      P10

0308 80 FC 07                   CMP     AH,7                    ; IS THIS BW CARD
030B 74 03                      JE      P10

030D E9 0642 R                  JMP     GRAPHICS_READ
0310                    P10:                                    ;  READ_AC_CONTINUE
0310 E8 032C R                  CALL    FIND_POSITION           ; GET REGEN LOCATION AND PORT ADDRESS
0313 8B F7                      MOV     SI,DI                   ; ESTABLISH ADDRESSING IN SI
0315 06                         PUSH    ES                      ; GET REGEN SEGMENT FOR QUICK ACCESS
0316 1F                         POP     DS

                        ;-----  WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80

0317 0A DB                      OR      BL,BL                   ; CHECK MODE FLAG FOR COLOR CARD IN 80
0319 75 0D                      JNZ     P13                     ; ELSE SKIP RETRACE WAIT - DO FAST READ
031B                    P11:                                    ;  WAIT FOR HORZ RETRACE LOW OR VERTICAL
031B FB                         STI                             ; ENABLE INTERRUPTS FIRST
031C 90                         NOP                             ; ALLOW FOR SMALL INTERRUPT WINDOW
031D FA                         CLI                             ; BLOCK INTERRUPTS FOR SINGLE LOOP
031E EC                         IN      AL,DX                   ; GET STATUS FROM THE ADAPTER
031F A8 01                      TEST    AL,RHRZ                 ; IS HORIZONTAL RETRACE LOW
0321 75 F8                      JNZ     P11                     ; WAIT UNTIL IT IS
0323                    P12:                                    ;  NOW WAIT FOR EITHER RETRACE HIGH
0323 EC                         IN      AL,DX                   ; GET STATUS
0324 A8 09                      TEST    AL,RVRT+RHRZ            ; IS HORIZONTAL OR VERTICAL RETRACE HIGH
0326 74 FB                      JZ      P12                     ; WAIT UNTIL EITHER IS ACTIVE
0328                    P13:
0328 AD                         LODSW                           ; GET THE CHARACTER AND ATTRIBUTE
0329 E9 013D R                  JMP     VIDEO_RETURN            ;  EXIT WITH (AX)

032C                    READ_AC_CURRENT ENDP


032C                    FIND_POSITION   PROC    NEAR            ;       SETUP FOR BUFFER READ OR WRITE
032C 86 E3                      XCHG    AH,BL                   ; SWAP MODE TYPE WITH ATTRIBUTE
032E 8B E8                      MOV     BP,AX                   ; SAVE CHARACTER/ATTR IN (BP) REGISTER
0330 80 EB 02                   SUB     BL,2                    ; CONVERT DISPLAY MODE TYPE TO A
0333 D0 EB                      SHR     BL,1                    ;  ZERO VALUE FOR COLOR IN 80 COLUMN
0335 8A C7                      MOV     AL,BH                   ; MOVE DISPLAY PAGE TO LOW BYTE
0337 98                         CBW                             ; CLEAR HIGH BYTE FOR BYTE OFFSET
0338 8B F8                      MOV     DI,AX                   ; MOVE DISPLAY PAGE (COUNT) TO WORK REG
033A D1 E7                      SAL     DI,1                    ; TIMES 2 FOR WORD OFFSET
033C 8B 95 0050 R               MOV     DX,[DI+OFFSET @CURSOR_POSN]     ; GET ROW/COLUMN OF THAT PAGE
0340 74 09                      JZ      P21                     ; SKIP BUFFER ADJUSTMENT IF PAGE ZERO

0342 33 FF                      XOR     DI,DI                   ; ELSE SET BUFFER START ADDRESS TO ZERO
0344                    P20:
0344 03 3E 004C R               ADD     DI,@CRT_LEN             ; ADD LENGTH OF BUFFER FOR ONE PAGE
0348 48                         DEC     AX                      ; DECREMENT PAGE COUNT
0349 75 F9                      JNZ     P20                     ; LOOP TILL PAGE COUNT EXHAUSTED

034B                    P21:                                    ; DETERMINE LOCATION IN REGEN IN PAGE
034B A0 004A R                  MOV     AL,BYTE PTR @CRT_COLS   ; GET COLUMNS PER ROW COUNT
034E F6 E6                      MUL     DH                      ; DETERMINE BYTES TO ROW
0350 32 F6                      XOR     DH,DH
0352 03 C2                      ADD     AX,DX                   ; ADD IN COLUMN VALUE
0354 D1 E0                      SAL     AX,1                    ; * 2 FOR ATTRIBUTE BYTES
0356 03 F8                      ADD     DI,AX                   ; ADD LOCATION TO START OF REGEN PAGE
0358 8B 16 0063 R               MOV     DX,@ADDR_6845           ; GET BASE ADDRESS OF ACTIVE DISPLAY
035C 83 C2 06                   ADD     DX,6                    ;  DX= STATUS PORT ADDRESS OF ADAPTER
035F C3                         RET                             ;  BP= ATTRIBUTE/CHARACTER (FROM BL/AL)
                                                                ;  DI= POSITION (OFFSET IN REGEN BUFFER)
0360                    FIND_POSITION   ENDP                    ;  BL= MODE FLAG (ZERO FOR 80X25 COLOR)
```

```
                        PAGE
                        ;--------------------------------------------------------------------------------
                        ; WRITE_AC_CURRENT                                                              :
                        ;       THIS ROUTINE WRITES THE ATTRIBUTE AND CHARACTER                         :
                        ;       AT THE CURRENT CURSOR POSITION                                         :
                        ; INPUT                                                                        :
                        ;       (AH) = CURRENT CRT MODE                                                :
                        ;       (BH) = DISPLAY PAGE                                                    :
                        ;       (CX) = COUNT OF CHARACTERS TO WRITE                                    :
                        ;       (AL) = CHAR TO WRITE                                                   :
                        ;       (BL) = ATTRIBUTE OF CHAR TO WRITE                                      :
                        ;       (DS) = DATA SEGMENT                                                    :
                        ;       (ES) = REGEN SEGMENT                                                   :
                        ; OUTPUT                                                                       :
                        ;       DISPLAY REGEN BUFFER UPDATED                                           :
                        ;--------------------------------------------------------------------------------

 0360                   WRITE_AC_CURRENT        PROC    NEAR
 0360 80 FC 04                  CMP     AH,4                    ; IS THIS GRAPHICS
 0363 72 08                     JC      P30
 0365 80 FC 07                  CMP     AH,7                    ; IS THIS BW CARD
 0368 74 03                     JE      P30
 036A E9 058E R                 JMP     GRAPHICS_WRITE
 036D                   P30:                                    ;  WRITE_AC_CONTINUE
 036D E8 032C R                 CALL    FIND_POSITION           ; GET REGEN LOCATION AND PORT ADDRESS
                                                                ; ADDRESS IN (DI) REGISTER
 0370 0A DB                     OR      BL,BL                   ; CHECK MODE FLAG FOR COLOR CARD AT 80
 0372 74 06                     JZ      P32                     ; SKIP TO RETRACE WAIT IF COLOR AT 80

 0374 95                        XCHG    AX,BP                   ; GET THE ATTR/CHAR SAVED FOR FAST WRITE
 0375 F3/ AB                    REP     STOSW                   ; STRING WRITE THE ATTRIBUTE & CHARACTER
 0377 EB 16                     JMP     SHORT  P35              ; EXIT FAST WRITE ROUTINE

                        ;-----  WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80

 0379                   P31:                                    ;  LOOP FOR EACH ATTR/CHAR WRITE
 0379 95                        XCHG    BP,AX                   ; PLACE ATTR/CHAR BACK IN SAVE REGISTER
 037A                   P32:                                    ;  WAIT FOR HORZ RETRACE LOW OR VERTICAL
 037A FB                        STI                             ; ENABLE INTERRUPTS FIRST
 037B 90                        NOP                             ; ALLOW FOR INTERRUPT WINDOW
 037C FA                        CLI                             ; BLOCK INTERRUPTS FOR SINGLE LOOP
 037D EC                        IN      AL,DX                   ; GET STATUS FROM THE ADAPTER
 037E A8 08                     TEST    AL,RVRT                 ; CHECK FOR VERTICAL RETRACE FIRST
 0380 75 09                     JNZ     P34                     ; DO FAST WRITE NOW IF VERTICAL RETRACE
 0382 A8 01                     TEST    AL,RHRZ                 ; IS HORIZONTAL RETRACE LOW THEN
 0384 75 F4                     JNZ     P32                     ; WAIT UNTIL IT IS
 0386                   P33:                                    ;  WAIT FOR EITHER RETRACE HIGH
 0386 EC                        IN      AL,DX                   ; GET STATUS AGAIN
 0387 A8 09                     TEST    AL,RVRT+RHRZ            ; IS HORIZONTAL OR VERTICAL RETRACE HIGH
 0389 74 FB                     JZ      P33                     ; WAIT UNTIL EITHER IS ACTIVE
 038B                   P34:
 038B 95                        XCHG    AX,BP                   ; GET THE ATTR/CHAR SAVED IN (BP)
 038C AB                        STOSW                           ; WRITE THE ATTRIBUTE AND CHARACTER
 038D E2 EA                     LOOP    P31                     ; AS MANY TIMES AS REQUESTED - TILL CX=0
 038F                   P35:
 038F E9 013D R                 JMP     VIDEO_RETURN            ; EXIT

 0392                   WRITE_AC_CURRENT        ENDP

                        ;--------------------------------------------------------------------------------
                        ; WRITE_C_CURRENT                                                               :
                        ;       THIS ROUTINE WRITES THE CHARACTER AT                                   :
                        ;       THE CURRENT CURSOR POSITION, ATTRIBUTE UNCHANGED                       :
                        ; INPUT                                                                        :
                        ;       (AH) = CURRENT CRT MODE                                                :
                        ;       (BH) = DISPLAY PAGE                                                    :
                        ;       (CX) = COUNT OF CHARACTERS TO WRITE                                    :
                        ;       (AL) = CHAR TO WRITE                                                   :
                        ;       (DS) = DATA SEGMENT                                                    :
                        ;       (ES) = REGEN SEGMENT                                                   :
                        ; OUTPUT                                                                       :
                        ;       DISPLAY REGEN BUFFER UPDATED                                           :
                        ;--------------------------------------------------------------------------------

 0392                   WRITE_C_CURRENT PROC    NEAR
 0392 80 FC 04                  CMP     AH,4                    ; IS THIS GRAPHICS
 0395 72 08                     JC      P40
 0397 80 FC 07                  CMP     AH,7                    ; IS THIS BW CARD
 039A 74 03                     JE      P40
 039C E9 058E R                 JMP     GRAPHICS_WRITE
 039F                   P40:
 039F E8 032C R                 CALL    FIND_POSITION           ; GET REGEN LOCATION AND PORT ADDRESS
                                                                ; ADDRESS OF LOCATION IN (DI)

                        ;-----  WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80

 03A2                   P41:                                    ;  WAIT FOR HORZ RETRACE LOW OR VERTICAL
 03A2 FB                        STI                             ; ENABLE INTERRUPTS FIRST
 03A3 0A DB                     OR      BL,BL                   ; CHECK MODE FLAG FOR COLOR CARD IN 80
 03A5 75 0F                     JNZ     P43                     ; ELSE SKIP RETRACE WAIT - DO FAST WRITE
 03A7 FA                        CLI                             ; BLOCK INTERRUPTS FOR SINGLE LOOP
 03A8 EC                        IN      AL,DX                   ; GET STATUS FROM THE ADAPTER
 03A9 A8 08                     TEST    AL,RVRT                 ; CHECK FOR VERTICAL RETRACE FIRST
 03AB 75 09                     JNZ     P43                     ; DO FAST WRITE NOW IF VERTICAL RETRACE
 03AD A8 01                     TEST    AL,RHRZ                 ; IS HORIZONTAL RETRACE LOW THEN
 03AF 75 F1                     JNZ     P41                     ; WAIT UNTIL IT IS
 03B1                   P42:                                    ;  WAIT FOR EITHER RETRACE HIGH
 03B1 EC                        IN      AL,DX                   ; GET STATUS AGAIN
 03B2 A8 09                     TEST    AL,RVRT+RHRZ            ; IS HORIZONTAL OR VERTICAL RETRACE HIGH
 03B4 74 FB                     JZ      P42                     ; WAIT UNTIL EITHER RETRACE ACTIVE
 03B6                   P43:
 03B6 8B C5                     MOV     AX,BP                   ; GET THE CHARACTER SAVE IN (BP)
 03B8 AA                        STOSB                           ; PUT THE CHARACTER INTO REGEN BUFFER
 03B9 47                        INC     DI                      ; BUMP POINTER PAST ATTRIBUTE
 03BA E2 E6                     LOOP    P41                     ;  AS MANY TIMES AS REQUESTED

 03BC E9 013D R                 JMP     VIDEO_RETURN

 03BF                   WRITE_C_CURRENT ENDP
```

```
                        PAGE
                        ;--------------------------------------------------------------------------------
                        ; WRITE_STRING                                                                  :
                        ;       THIS ROUTINE WRITES A STRING OF CHARACTERS TO THE CRT.                  :
                        ; INPUT                                                                        :
                        ;       (AL) = WRITE STRING COMMAND  0 - 3                                     :
                        ;       (BH) = DISPLAY PAGE (ACTIVE PAGE)                                      :
                        ;       (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN           :
                        ;       (DX) = CURSOR POSITION FOR START OF STRING WRITE                       :
                        ;       (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR  (AL) = 1       :
                        ;       (BP) = SOURCE STRING OFFSET                                            :
                        ;       [0E] = SOURCE STRING SEGMENT (FOR USE IN (ES) IN STACK +14)            :
                        ; OUTPUT                                                                       :
                        ;       NONE                                                                   :
                        ;--------------------------------------------------------------------------------
 03BF                   WRITE_STRING    PROC    NEAR
 03BF 55                        PUSH    BP                      ; SAVE BUFFER OFFSET (BP) IN STACK
 03C0 8B EC                     MOV     BP,SP                   ; GET POINTER TO STACKED REGISTERS
 03C2 8E 46 10                  MOV     ES,[BP]+14+2            ; RECOVER ENTRY (ES) SEGMENT REGISTER
 03C5 5D                        POP     BP                      ; RESTORE BUFFER OFFSET
 03C6 98                        CBW                             ; CLEAR (AH) REGISTER
 03C7 8B F8                     MOV     DI,AX                   ; SAVE (AL) COMMAND IN (DI) REGISTER
 03C9 3C 04                     CMP     AL,04                   ; TEST FOR INVALID WRITE STRING OPTION
 03CB 73 73                     JNB     P59                     ; IF OPTION INVALID THEN RETURN
                        
 03CD E3 71                     JCXZ    P59                     ; IF ZERO LENGTH STRING THEN RETURN
                        
 03CF 8B F3                     MOV     SI,BX                   ; SAVE CURRENT CURSOR PAGE
 03D1 8A DF                     MOV     BL,BH                   ; MOVE PAGE TO LOW BYTE
 03D3 32 FF                     XOR     BH,BH                   ; CLEAR HIGH BYTE
 03D5 87 F3                     XCHG    SI,BX                   ; MOVE OFFSET AND RESTORE PAGE REGISTER
 03D7 D1 E6                     SAL     SI,1                    ; CONVERT TO PAGE OFFSET  (SI= PAGE)
 03D9 FF B4 0050 R              PUSH    [SI+OFFSET @CURSOR_POSN]; SAVE CURRENT CURSOR POSITION IN STACK
 03DD B8 0200                   MOV     AX,0200H                ; SET NEW CURSOR POSITION
 03E0 CD 10                     INT     10H
 03E2                   P50:
 03E2 26: 8A 46 00              MOV     AL,ES:[BP]              ; GET CHARACTER FROM INPUT STRING
 03E6 45                        INC     BP                      ; BUMP POINTER TO CHARACTER
                        
                        ;-----  TEST FOR SPECIAL CHARACTER'S
                        
 03E7 3C 08                     CMP     AL,08H                  ; IS IT A BACKSPACE
 03E9 74 0C                     JE      P51                     ; BACK_SPACE
 03EB 3C 0D                     CMP     AL,CR                   ; IS IT CARRIAGE RETURN
 03ED 74 08                     JE      P51                     ; CAR_RET
 03EF 3C 0A                     CMP     AL,LF                   ; IS IT A LINE FEED
 03F1 74 04                     JE      P51                     ; LINE_FEED
 03F3 3C 07                     CMP     AL,07H                  ; IS IT A BELL
 03F5 75 0A                     JNE     P52                     ; IF NOT THEN DO WRITE CHARACTER
 03F7                   P51:
 03F7 B4 0E                     MOV     AH,0EH                  ; TTY_CHARACTER WRITE
 03F9 CD 10                     INT     10H                       ;WRITE TTY CHARACTER TO THE CRT
 03FB 8B 94 0050 R              MOV     DX,[SI+OFFSET @CURSOR_POSN] ; GET CURRENT CURSOR POSITION
 03FF EB 2D                     JMP     SHORT P54                 ; SET CURSOR POSITION AND CONTINUE
                        
 0401                   P52:
 0401 51                        PUSH    CX
 0402 53                        PUSH    BX
 0403 B9 0001                   MOV     CX,1                    ; SET CHARACTER WRITE AMOUNT TO ONE
 0406 83 FF 02                  CMP     DI,2                    ; IS THE ATTRIBUTE IN THE STRING
 0409 72 05                     JB      P53                     ; IF NOT THEN SKIP
 040B 26: 8A 5E 00              MOV     BL,ES:[BP]              ; ELSE GET NEW ATTRIBUTE
 040F 45                        INC     BP                      ; BUMP STRING POINTER
 0410                   P53:
 0410 B4 09                     MOV     AH,09H                  ;  GOT_CHARACTER
 0412 CD 10                     INT     10H                     ; WRITE CHARACTER TO THE CRT
 0414 5B                        POP     BX                      ; RESTORE REGISTERS
 0415 59                        POP     CX
 0416 FE C2                     INC     DL                      ; INCREMENT COLUMN COUNTER
 0418 3A 16 004A R              CMP     DL,BYTE PTR @CRT_COLS   ; IF COLS ARE WITHIN RANGE FOR THIS MODE
 041C 72 10                     JB      P54                     ;    THEN GO TO COLUMNS SET
 041E FE C6                     INC     DH                      ; BUMP ROW COUNTER BY ONE
 0420 2A D2                     SUB     DL,DL                   ; SET COLUMN COUNTER TO ZERO
 0422 80 FE 19                  CMP     DH,25                   ; IF ROWS ARE LESS THAN 25 THEN
 0425 72 07                     JB      P54                     ;    GO TO ROWS_COLUMNS_SET
                        
 0427 B8 0E0A                   MOV     AX,0E0AH                ; ELSE SCROLL SCREEN ONE LINE
 042A CD 10                     INT     10H                     ; RESET ROW COUNTER TO 24
 042C FE CE                     DEC     DH
 042E                   P54:                                    ;  ROW_COLUMNS_SET
 042E B8 0200                   MOV     AX,0200H                ; SET NEW CURSOR POSITION COMMAND
 0431 CD 10                     INT     10H                     ; ESTABLISH NEW CURSOR POSITION
 0433 E2 AD                     LOOP    P50                     ; DO IT ONCE MORE UNTIL (CX) = ZERO
                        
 0435 5A                        POP     DX                      ; RESTORE OLD CURSOR COORDINATES
 0436 97                        XCHG    AX,DI                   ; RECOVER WRITE STRING COMMAND
 0437 A8 01                     TEST    AL,01H                  ; IF CURSOR WAS NOT TO BE MOVED THEN
 0439 75 05                     JNZ     P59                     ;  THEN EXIT WITHOUT RESETTING OLD VALUE
 043B B8 0200                   MOV     AX,0200H                ; ELSE RESTORE OLD CURSOR POSITION
 043E CD 10                     INT     10H
 0440                   P59:                                    ;  DONE - EXIT WRITE STRING
 0440 E9 013D R                 JMP     VIDEO_RETURN            ; RETURN TO CALLER
                        
 0443                   WRITE_STRING    ENDP
```

```
                        PAGE
                        ;--------------------------------------------
                        ; READ_DOT  -- WRITE DOT
                        ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
                        ;  DOT AT THE INDICATED LOCATION
                        ; ENTRY --
                        ;   DX = ROW (0-199)     (THE ACTUAL VALUE DEPENDS ON THE MODE)
                        ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )
                        ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,
                        ;       REQUIRED FOR WRITE DOT ONLY, RIGHT JUSTIFIED)
                        ;       BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
                        ;   DS = DATA SEGMENT
                        ;   ES = REGEN SEGMENT
                        ;
                        ; EXIT
                        ;       AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
                        ;--------------------------------------------
                                ASSUME  DS:DATA,ES:DATA
0443                    READ_DOT        PROC    NEAR
0443 E8 0477 R                  CALL    R3                      ; DETERMINE BYTE POSITION OF DOT
0446 26: 8A 04                  MOV     AL,ES:[SI]              ; GET THE BYTE
0449 22 C4                      AND     AL,AH                   ; MASK OFF THE OTHER BITS IN THE BYTE
044B D2 E0                      SHL     AL,CL                   ; LEFT JUSTIFY THE VALUE
044D 8A CE                      MOV     CL,DH                   ; GET NUMBER OF BITS IN RESULT
044F D2 C0                      ROL     AL,CL                   ; RIGHT JUSTIFY THE RESULT
0451 E9 013D R                  JMP     VIDEO_RETURN            ; RETURN FROM VIDEO I/O
0454                    READ_DOT        ENDP
                        
0454                    WRITE_DOT       PROC    NEAR
0454 50                         PUSH    AX                      ; SAVE DOT VALUE
0455 50                         PUSH    AX                      ;  TWICE
0456 E8 0477 R                  CALL    R3                      ; DETERMINE BYTE POSITION OF THE DOT
0459 D2 E8                      SHR     AL,CL                   ; SHIFT TO SET UP THE BITS FOR OUTPUT
045B 22 C4                      AND     AL,AH                   ; STRIP OFF THE OTHER BITS
045D 26: 8A 0C                  MOV     CL,ES:[SI]              ; GET THE CURRENT BYTE
0460 5B                         POP     BX                      ; RECOVER XOR FLAG
0461 F6 C3 80                   TEST    BL,80H                  ; IS IT ON
0464 75 0D                      JNZ     R2                      ; YES, XOR THE DOT
0466 F6 D4                      NOT     AH                      ; SET MASK TO REMOVE THE INDICATED BITS
0468 22 CC                      AND     CL,AH
046A 0A C1                      OR      AL,CL                   ; OR IN THE NEW VALUE OF THOSE BITS
046C                    R1:                                     ; FINISH_DOT
046C 26: 88 04                  MOV     ES:[SI],AL              ; RESTORE THE BYTE IN MEMORY
046F 58                         POP     AX
0470 E9 013D R                  JMP     VIDEO_RETURN            ; RETURN FROM VIDEO I/O
0473                    R2:                                     ; XOR_DOT
0473 32 C1                      XOR     AL,CL                   ; EXCLUSIVE OR THE DOTS
0475 EB F5                      JMP     R1                      ; FINISH UP THE WRITING
0477                    WRITE_DOT       ENDP
                        ;--------------------------------------------
                        ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
                        ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
                        ; ENTRY --
                        ;  DX = ROW VALUE (0-199)
                        ;  CX = COLUMN VALUE (0-639)
                        ; EXIT --
                        ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
                        ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
                        ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
                        ;  DH = # BITS IN RESULT
                        ;  BX = MODIFIED
                        ;--------------------------------------------
0477                    R3      PROC    NEAR
                        
                        ;-----  DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
                        ;-----   ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW )
                        
0477 96                         XCHG    SI,AX                   ; WILL SAVE AL AND AH DURING OPERATION
0478 B0 28                      MOV     AL,40
047A F6 E2                      MUL     DL                      ; AX= ADDRESS OF START OF INDICATED ROW
047C A8 08                      TEST    AL,008H                 ; TEST FOR EVEN/ODD ROW CALCULATED
047E 74 03                      JZ      R4                      ; JUMP IF EVEN ROW
0480 05 1FD8                    ADD     AX,2000H-40             ; OFFSET TO LOCATION OF ODD ROWS ADJUST
0483                    R4:                                     ; EVEN ROW
0483 96                         XCHG    SI,AX                   ; MOVE POINTER TO (SI) AND RECOVER (AX)
0484 8B D1                      MOV     DX,CX                   ; COLUMN VALUE TO DX
                        
                        ;-----  DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
                        
                        ; SET UP THE REGISTERS ACCORDING TO THE MODE
                        ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES )
                        ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M )
                        ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/C0H FOR H/M )
                        ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M )
                        
0486 BB 02C0                    MOV     BX,2C0H
0489 B9 0302                    MOV     CX,302H                 ; SET PARMS FOR MED RES
048C 80 3E 0049 R 06            CMP     @CRT_MODE,6
0491 72 06                      JC      R5                      ; HANDLE IF MED RES
0493 BB 0180                    MOV     BX,180H
0496 B9 0703                    MOV     CX,703H                 ; SET PARMS FOR HIGH RES
                        
                        ;-----  DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
0499                    R5:
0499 22 EA                      AND     CH,DL                   ; ADDRESS OF PEL WITHIN BYTE TO CH
                        
                        ;-----  DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
                        
049B D3 EA                      SHR     DX,CL                   ; SHIFT BY CORRECT AMOUNT
049D 03 F2                      ADD     SI,DX                   ; INCREMENT THE POINTER
049F 8A F7                      MOV     DH,BH                   ; GET THE # OF BITS IN RESULT TO DH
                        
                        ;-----  MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
                        
04A1 2A C9                      SUB     CL,CL                   ; ZERO INTO STORAGE LOCATION
04A3                    R6:
04A3 D0 C8                      ROR     AL,1                    ; LEFT JUSTIFY VALUE IN AL (FOR WRITE)
04A5 02 CD                      ADD     CL,CH                   ; ADD IN THE BIT OFFSET VALUE
04A7 FE CF                      DEC     BH                      ; LOOP CONTROL
04A9 75 F8                      JNZ     R6                      ; ON EXIT, CL HAS COUNT TO RESTORE BITS
04AB 8A E3                      MOV     AH,BL                   ;  GET MASK TO AH
04AD D2 EC                      SHR     AH,CL                   ;  MOVE THE MASK TO CORRECT LOCATION
04AF C3                         RET                             ;  RETURN WITH EVERYTHING SET UP
04B0                    R3      ENDP
```

```
                        ;--------------------------------------------------------
                        ;  SCROLL UP
                        ;    THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
                        ; ENTRY --
                        ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
                        ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
                        ;    BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
                        ;  BH = FILL VALUE FOR BLANKED LINES
                        ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
                        ;  DS = DATA SEGMENT
                        ;  ES = REGEN SEGMENT
                        ; EXIT --
                        ;  NOTHING, THE SCREEN IS SCROLLED
                        ;--------------------------------------------------------
04B0                    GRAPHICS_UP     PROC    NEAR
04B0 8A D8                      MOV     BL,AL                   ; SAVE LINE COUNT IN BL
04B2 8B C1                      MOV     AX,CX                   ; GET UPPER LEFT POSITION INTO AX REG
                        ;-----  USE CHARACTER SUBROUTINE FOR POSITIONING
                        ;-----  ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
04B4 E8 06F0 R                  CALL    GRAPH_POSN
04B7 8B F8                      MOV     DI,AX                   ; SAVE RESULT AS DESTINATION ADDRESS
                        ;-----  DETERMINE SIZE OF WINDOW
04B9 2B D1                      SUB     DX,CX
04BB 81 C2 0101                 ADD     DX,101H                 ; ADJUST VALUES
04BF D0 E6                      SAL     DH,1                    ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
04C1 D0 E6                      SAL     DH,1                    ;  AND EVEN/ODD ROWS
                        ;-----  DETERMINE CRT MODE
04C3 80 3E 0049 R 06            CMP     @CRT_MODE,6             ; TEST FOR MEDIUM RES
04C8 73 04                      JNC     R7                      ; FIND_SOURCE
                        ;-----  MEDIUM RES UP
04CA D0 E2                      SAL     DL,1                    ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
04CC D1 E7                      SAL     DI,1                    ; OFFSET *2 SINCE 2 BYTES/CHAR
                        ;-----  DETERMINE THE SOURCE ADDRESS IN THE BUFFER
04CE                    R7:                                     ; FIND_SOURCE
04CE 06                         PUSH    ES                      ; GET SEGMENTS BOTH POINTING TO REGEN
04CF 1F                         POP     DS
04D0 2A ED                      SUB     CH,CH                   ; ZERO TO HIGH OF COUNT REGISTER
04D2 D0 E3                      SAL     BL,1                    ; MULTIPLY NUMBER OF LINES BY 4
04D4 D0 E3                      SAL     BL,1
04D6 74 2B                      JZ      R11                     ; IF ZERO, THEN BLANK ENTIRE FIELD
04D8 B0 50                      MOV     AL,80                   ; 80 BYTES/ROW
04DA F6 E3                      MUL     BL                      ; DETERMINE OFFSET TO SOURCE
04DC 8B F7                      MOV     SI,DI                   ; SET UP SOURCE
04DE 03 F0                      ADD     SI,AX                   ;  ADD IN OFFSET TO IT
04E0 8A E6                      MOV     AH,DH                   ; NUMBER OF ROWS IN FIELD
04E2 2A E3                      SUB     AH,BL                   ; DETERMINE NUMBER TO MOVE
                        ;-----  LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
04E4                    R8:                                     ; ROW_LOOP
04E4 E8 0564 R                  CALL    R17                     ; MOVE ONE ROW
04E7 81 EE 1FB0                 SUB     SI,2000H-80             ; MOVE TO NEXT ROW
04EB 81 EF 1FB0                 SUB     DI,2000H-80
04EF FE CC                      DEC     AH                      ; NUMBER OF ROWS TO MOVE
04F1 75 F1                      JNZ     R8                      ; CONTINUE TILL ALL MOVED
                        ;-----  FILL IN THE VACATED LINE(S)
04F3                    R9:                                     ; CLEAR_ENTRY
04F3 8A C7                      MOV     AL,BH                   ; ATTRIBUTE TO FILL WITH
04F5                    R10:
04F5 E8 057D R                  CALL    R18                     ; CLEAR THAT ROW
04F8 81 EF 1FB0                 SUB     DI,2000H-80             ; POINT TO NEXT LINE
04FC FE CB                      DEC     BL                      ; NUMBER OF LINES TO FILL
04FE 75 F5                      JNZ     R10                     ; CLEAR_LOOP
0500 E9 013D R                  JMP     VIDEO_RETURN            ; EVERYTHING DONE

0503                    R11:                                    ; BLANK_FIELD
0503 8A DE                      MOV     BL,DH                   ; SET BLANK COUNT TO EVERYTHING IN FIELD
0505 EB EC                      JMP     R9                      ; CLEAR THE FIELD
0507                    GRAPHICS_UP     ENDP
                        ;--------------------------------------------------------
                        ; SCROLL DOWN
                        ;  THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
                        ; ENTRY --
                        ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
                        ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
                        ;    BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
                        ;  BH = FILL VALUE FOR BLANKED LINES
                        ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
                        ;  DS = DATA SEGMENT
                        ;  ES = REGEN SEGMENT
                        ; EXIT --
                        ;  NOTHING, THE SCREEN IS SCROLLED
                        ;--------------------------------------------------------

0507                    GRAPHICS_DOWN   PROC    NEAR
0507 FD                         STD                             ; SET DIRECTION
0508 8A D8                      MOV     BL,AL                   ; SAVE LINE COUNT IN BL
050A 8B C2                      MOV     AX,DX                   ; GET LOWER RIGHT POSITION INTO AX REG
                        ;-----  USE CHARACTER SUBROUTINE FOR POSITIONING
                        ;-----  ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
050C E8 06F0 R                  CALL    GRAPH_POSN
050F 8B F8                      MOV     DI,AX                   ; SAVE RESULT AS DESTINATION ADDRESS
                        ;-----  DETERMINE SIZE OF WINDOW
0511 2B D1                      SUB     DX,CX
0513 81 C2 0101                 ADD     DX,101H                 ; ADJUST VALUES
0517 D0 E6                      SAL     DH,1                    ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
0519 D0 E6                      SAL     DH,1                    ;  AND EVEN/ODD ROWS
                        ;-----  DETERMINE CRT MODE
051B 80 3E 0049 R 06            CMP     @CRT_MODE,6             ; TEST FOR MEDIUM RES
0520 73 05                      JNC     R12                     ; FIND_SOURCE_DOWN
```

SECTION 5

```
                        ;-----  MEDIUM RES DOWN
0522 D0 E2                      SAL     DL,1                    ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
0524 D1 E7                      SAL     DI,1                    ; OFFSET *2 SINCE 2 BYTES/CHAR
0526 47                         INC     DI                      ; POINT TO LAST BYTE

                        ;-----  DETERMINE THE SOURCE ADDRESS IN THE BUFFER

0527                    R12:                                    ; FIND_SOURCE_DOWN
0527 06                         PUSH    ES                      ; BOTH SEGMENTS TO REGEN
0528 1F                         POP     DS
0529 2A ED                      SUB     CH,CH                   ; ZERO TO HIGH OF COUNT REGISTER
052B 81 C7 00F0                 ADD     DI,240                  ; POINT TO LAST ROW OF PIXELS
052F D0 E3                      SAL     BL,1                    ; MULTIPLY NUMBER OF LINES BY 4
0531 D0 E3                      SAL     BL,1
0533 74 2B                      JZ      R16                     ; IF ZERO, THEN BLANK ENTIRE FIELD
0535 B0 50                      MOV     AL,80                   ; 80 BYTES/ROW
0537 F6 E3                      MUL     BL                      ; DETERMINE OFFSET TO SOURCE
0539 8B F7                      MOV     SI,DI                   ; SET UP SOURCE
053B 2B F0                      SUB     SI,AX                   ;  SUBTRACT THE OFFSET
053D 8A E6                      MOV     AH,DH                   ; NUMBER OF ROWS IN FIELD
053F 2A E3                      SUB     AH,BL                   ; DETERMINE NUMBER TO MOVE

                        ;-----  LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS

0541                    R13:                                    ; ROW_LOOP_DOWN
0541 E8 0564 R                  CALL    R17                     ; MOVE ONE ROW
0544 81 EE 2050                 SUB     SI,2000H+80             ; MOVE TO NEXT ROW
0548 81 EF 2050                 SUB     DI,2000H+80
054C FE CC                      DEC     AH                      ; NUMBER OF ROWS TO MOVE
054E 75 F1                      JNZ     R13                     ; CONTINUE TILL ALL MOVED

                        ;-----  FILL IN THE VACATED LINE(S)
0550                    R14:                                    ; CLEAR_ENTRY_DOWN
0550 8A C7                      MOV     AL,BH                   ; ATTRIBUTE TO FILL WITH
0552                    R15:                                    ; CLEAR_LOOP_DOWN
0552 E8 057D R                  CALL    R18                     ; CLEAR A ROW
0555 81 EF 2050                 SUB     DI,2000H+80             ; POINT TO NEXT LINE
0559 FE CB                      DEC     BL                      ; NUMBER OF LINES TO FILL
055B 75 F5                      JNZ     R15                     ; CLEAR_LOOP_DOWN

055D E9 013D R                  JMP     VIDEO_RETURN            ; EVERYTHING DONE

0560                    R16:                                    ; BLANK_FIELD_DOWN
0560 8A DE                      MOV     BL,DH                   ; SET BLANK COUNT TO EVERYTHING IN FIELD
0562 EB EC                      JMP     R14                     ; CLEAR THE FIELD
0564                    GRAPHICS_DOWN   ENDP

                        ;-----  ROUTINE TO MOVE ONE ROW OF INFORMATION

0564                    R17     PROC    NEAR
0564 8A CA                      MOV     CL,DL                   ; NUMBER OF BYTES IN THE ROW
0566 56                         PUSH    SI
0567 57                         PUSH    DI                      ; SAVE POINTERS
0568 F3/ A4                     REP     MOVSB                   ; MOVE THE EVEN FIELD
056A 5F                         POP     DI
056B 5E                         POP     SI
056C 81 C6 2000                 ADD     SI,2000H
0570 81 C7 2000                 ADD     DI,2000H                ; POINT TO THE ODD FIELD
0574 56                         PUSH    SI
0575 57                         PUSH    DI                      ; SAVE THE POINTERS
0576 8A CA                      MOV     CL,DL                   ; COUNT BACK
0578 F3/ A4                     REP     MOVSB                   ; MOVE THE ODD FIELD
057A 5F                         POP     DI
057B 5E                         POP     SI                      ; POINTERS BACK
057C C3                         RET                             ; RETURN TO CALLER
057D                    R17     ENDP

                        ;-----  CLEAR A SINGLE ROW

057D                    R18     PROC    NEAR
057D 8A CA                      MOV     CL,DL                   ; NUMBER OF BYTES IN FIELD
057F 57                         PUSH    DI                      ; SAVE POINTER
0580 F3/ AA                     REP     STOSB                   ; STORE THE NEW VALUE
0582 5F                         POP     DI                      ; POINTER BACK
0583 81 C7 2000                 ADD     DI,2000H                ; POINT TO ODD FIELD
0587 57                         PUSH    DI
0588 8A CA                      MOV     CL,DL
058A F3/ AA                     REP     STOSB                   ; FILL THE ODD FIELD
058C 5F                         POP     DI
058D C3                         RET                             ; RETURN TO CALLER
058E                    R18     ENDP

                        ;--------------------------------------------------
                        ; GRAPHICS WRITE
                        ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
                        ;  POSITION ON THE SCREEN.
                        ; ENTRY --
                        ;  AL = CHARACTER TO WRITE
                        ;  BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
                        ;       IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
                        ;       (0 IS USED FOR THE BACKGROUND COLOR)
                        ;  CX = NUMBER OF CHARS TO WRITE
                        ;  DS = DATA SEGMENT
                        ;  ES = REGEN SEGMENT
                        ; EXIT --
                        ;  NOTHING IS RETURNED
                        ;
                        ; GRAPHICS READ
                        ;   THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT CURSOR
                        ;   POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
                        ;   CHARACTER GENERATOR CODE POINTS
                        ; ENTRY --
                        ;  NONE  (0 IS ASSUMED AS THE BACKGROUND COLOR)
                        ; EXIT --
                        ;  AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF NONE FOUND)
                        ;
                        ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE CONTAINED IN ROM
                        ;  FOR THE 1ST 128 CHARS.  TO ACCESS CHARS IN THE SECOND HALF, THE USER
                        ;  MUST INITIALIZE THE VECTOR AT INTERRUPT 1FH (LOCATION 0007CH) TO
                        ;  POINT TO THE USER SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).
                        ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS
                        ;--------------------------------------------------
```

```
                        ASSUME  DS:DATA,ES:DATA
058E                    GRAPHICS_WRITE  PROC    NEAR
058E B4 00                      MOV     AH,0                    ; ZERO TO HIGH OF CODE POINT
0590 50                         PUSH    AX                      ; SAVE CODE POINT VALUE
                        ;-----  DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
0591 E8 06ED R                  CALL    S26                     ; FIND LOCATION IN REGEN BUFFER
0594 8B F8                      MOV     DI,AX                   ; REGEN POINTER IN DI
                        ;-----  DETERMINE REGION TO GET CODE POINTS FROM
0596 58                         POP     AX                      ; RECOVER CODE POINT
0597 3C 80                      CMP     AL,80H                  ; IS IT IN SECOND HALF
0599 73 06                      JAE     S1                      ; YES
                        ;-----  IMAGE IS IN FIRST HALF, CONTAINED IN ROM
059B BE 0000 E                  MOV     SI,OFFSET CRT_CHAR_GEN  ; OFFSET OF IMAGES
059E 0E                         PUSH    CS                      ; SAVE SEGMENT ON STACK
059F EB 18                      JMP     SHORT S2                ; DETERMINE_MODE
                        ;-----  IMAGE IS IN SECOND HALF, IN USER MEMORY
05A1                    S1:                                     ; EXTEND_CHAR
05A1 2C 80                      SUB     AL,80H                  ; ZERO ORIGIN FOR SECOND HALF
05A3 1E                         PUSH    DS                      ; SAVE DATA POINTER
05A4 2B F6                      SUB     SI,SI
05A6 8E DE                      MOV     DS,SI                   ; ESTABLISH VECTOR ADDRESSING
                                ASSUME  DS:ABS0
05A8 C5 36 007C R               LDS     SI,@EXT_PTR             ; GET THE OFFSET OF THE TABLE
05AC 8C DA                      MOV     DX,DS                   ; GET THE SEGMENT OF THE TABLE
                                ASSUME  DS:DATA
05AE 1F                         POP     DS                      ; RECOVER DATA SEGMENT
05AF 52                         PUSH    DX                      ; SAVE TABLE SEGMENT ON STACK
05B0 0B D6                      OR      DX,SI                   ; CHECK FOR VALID TABLE DEFINED
05B2 75 05                      JNZ     S2                      ; CONTINUE IF DS:SI NOT 0000:0000
05B4 58                         POP     AX                      ; ELSE SET (AX)= 0000 FOR "NULL"
05B5 BE 0000 E                  MOV     SI,OFFSET CRT_CHAR_GEN  ; POINT TO DEFAULT TABLE OFFSET
05B8 0E                         PUSH    CS                      ;  IN THE CODE SEGMENT
                        ;-----  DETERMINE GRAPHICS MODE IN OPERATION
05B9                    S2:                                     ;       DETERMINE_MODE
05B9 D1 E0                      SAL     AX,1                    ; MULTIPLY CODE POINT VALUE BY 8
05BB D1 E0                      SAL     AX,1
05BD D1 E0                      SAL     AX,1
05BF 03 F0                      ADD     SI,AX                   ; SI HAS OFFSET OF DESIRED CODES
05C1 80 3E 0049 R 06            CMP     @CRT_MODE,6
05C6 1F                         POP     DS                      ; RECOVER TABLE POINTER SEGMENT
05C7 72 2C                      JC      S7                      ; TEST FOR MEDIUM RESOLUTION MODE
                        ;-----  HIGH RESOLUTION MODE
05C9                    S3:                                     ; HIGH_CHAR
05C9 57                         PUSH    DI                      ; SAVE REGEN POINTER
05CA 56                         PUSH    SI                      ; SAVE CODE POINTER
05CB B6 04                      MOV     DH,4                    ; NUMBER OF TIMES THROUGH LOOP
05CD                    S4:
05CD AC                         LODSB                           ; GET BYTE FROM CODE POINTS
05CE F6 C3 80                   TEST    BL,80H                  ; SHOULD WE USE THE FUNCTION
05D1 75 16                      JNZ     S6                      ;  TO PUT CHAR IN
05D3 AA                         STOSB                           ; STORE IN REGEN BUFFER
05D4 AC                         LODSB
05D5                    S5:
05D5 26: 88 85 1FFF             MOV     ES:[DI+2000H-1],AL      ; STORE IN SECOND HALF
05DA 83 C7 4F                   ADD     DI,79                   ; MOVE TO NEXT ROW IN REGEN
05DD FE CE                      DEC     DH                      ; DONE WITH LOOP
05DF 75 EC                      JNZ     S4
05E1 5E                         POP     SI
05E2 5F                         POP     DI                      ; RECOVER REGEN POINTER
05E3 47                         INC     DI                      ; POINT TO NEXT CHAR POSITION
05E4 E2 E3                      LOOP    S3                      ; MORE CHARS TO WRITE
05E6 E9 013D R                  JMP     VIDEO_RETURN
05E9                    S6:
05E9 26: 32 05                  XOR     AL,ES:[DI]              ; EXCLUSIVE OR WITH CURRENT
05EC AA                         STOSB                           ; STORE THE CODE POINT
05ED AC                         LODSB                           ; AGAIN FOR ODD FIELD
05EE 26: 32 85 1FFF             XOR     AL,ES:[DI+2000H-1]
05F3 EB E0                      JMP     S5                      ; BACK TO MAINSTREAM
                        ;-----  MEDIUM RESOLUTION WRITE
05F5                    S7:                                     ;  MED_RES_WRITE
05F5 8A D3                      MOV     DL,BL                   ; SAVE HIGH COLOR BIT
05F7 D1 E7                      SAL     DI,1                    ; OFFSET*2 SINCE 2 BYTES/CHAR
                                                                ; EXPAND BL TO FULL WORD OF COLOR
05F9 80 E3 03                   AND     BL,3                    ; ISOLATE THE COLOR BITS ( LOW 2 BITS )
05FC B0 55                      MOV     AL,055H                 ; GET BIT CONVERSION MULTIPLIER
05FE F6 E3                      MUL     BL                      ; EXPAND 2 COLOR BITS TO 4 REPLICATIONS
0600 8A D8                      MOV     BL,AL                   ; PLACE BACK IN WORK REGISTER
0602 8A F8                      MOV     BH,AL                   ; EXPAND TO 8 REPLICATIONS OF COLOR BITS
0604                    S8:                                     ;  MED_CHAR
0604 57                         PUSH    DI                      ; SAVE REGEN POINTER
0605 56                         PUSH    SI                      ; SAVE THE CODE POINTER
0606 B6 04                      MOV     DH,4                    ; NUMBER OF LOOPS
0608                    S9:
0608 AC                         LODSB                           ; GET CODE POINT
0609 E8 06C4 R                  CALL    S21                     ; DOUBLE UP ALL THE BITS
060C 23 C3                      AND     AX,BX                   ; CONVERT TO FOREGROUND COLOR ( 0 BACK )
060E 86 E0                      XCHG    AH,AL                   ; SWAP HIGH/LOW BYTES FOR WORD MOVE
0610 F6 C2 80                   TEST    DL,80H                  ; IS THIS XOR FUNCTION
0613 74 03                      JZ      S10                     ; NO, STORE IT IN AS IT IS
0615 26: 33 05                  XOR     AX,ES:[DI]              ; DO FUNCTION WITH LOW/HIGH
0618                    S10:
0618 26: 89 05                  MOV     ES:[DI],AX              ; STORE FIRST BYTE HIGH, SECOND LOW
061B AC                         LODSB                           ; GET CODE POINT
061C E8 06C4 R                  CALL    S21
061F 23 C3                      AND     AX,BX                   ; CONVERT TO COLOR
0621 86 E0                      XCHG    AH,AL                   ; SWAP HIGH/LOW BYTES FOR WORD MOVE
0623 F6 C2 80                   TEST    DL,80H                  ; AGAIN, IS THIS XOR FUNCTION
0626 74 05                      JZ      S11                     ; NO, JUST STORE THE VALUES
0628 26: 33 85 2000             XOR     AX,ES:[DI+2000H]        ; FUNCTION WITH FIRST HALF LOW
```

```
062D                    S11:
062D 26: 89 85 2000             MOV     ES:[DI+2000H],AX        ; STORE SECOND PORTION HIGH
0632 83 C7 50                   ADD     DI,80                   ; POINT TO NEXT LOCATION
0635 FE CE                      DEC     DH
0637 75 CF                      JNZ     S9                      ; KEEP GOING
0639 5E                         POP     SI                      ; RECOVER CODE POINTER
063A 5F                         POP     DI                      ; RECOVER REGEN POINTER
063B 47                         INC     DI                      ; POINT TO NEXT CHAR POSITION
063C 47                         INC     DI
063D E2 C5                      LOOP    S8                      ; MORE TO WRITE
063F E9 013D R                  JMP     VIDEO_RETURN
0642                    GRAPHICS_WRITE  ENDP
                        ;--------------------------------------
                        ; GRAPHICS READ
                        ;--------------------------------------
0642                    GRAPHICS_READ   PROC    NEAR
0642 E8 06ED R                  CALL    S26                     ; CONVERTED TO OFFSET IN REGEN
0645 8B F0                      MOV     SI,AX                   ; SAVE IN SI
0647 83 EC 08                   SUB     SP,8                    ; ALLOCATE SPACE FOR THE READ CODE POINT
064A 8B EC                      MOV     BP,SP                   ; POINTER TO SAVE AREA
                        ;-----  DETERMINE GRAPHICS MODES
064C 80 3E 0049 R 06            CMP     @CRT_MODE,6
0651 06                         PUSH    ES
0652 1F                         POP     DS                      ; POINT TO REGEN SEGMENT
0653 72 19                      JC      S13                     ; MEDIUM RESOLUTION
                        ;-----  HIGH RESOLUTION READ
                        ;-----  GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
0655 B6 04                      MOV     DH,4                    ; NUMBER OF PASSES
0657                    S12:
0657 8A 04                      MOV     AL,[SI]                 ; GET FIRST BYTE
0659 88 46 00                   MOV     [BP],AL                 ; SAVE IN STORAGE AREA
065C 45                         INC     BP                      ; NEXT LOCATION
065D 8A 84 2000                 MOV     AL,[SI+2000H]           ; GET LOWER REGION BYTE
0661 88 46 00                   MOV     [BP],AL                 ; ADJUST AND STORE
0664 45                         INC     BP
0665 83 C6 50                   ADD     SI,80                   ; POINTER INTO REGEN
0668 FE CE                      DEC     DH                      ; LOOP CONTROL
066A 75 EB                      JNZ     S12                     ; DO IT SOME MORE
066C EB 16                      JMP     SHORT S15               ; GO MATCH THE SAVED CODE POINTS
                        ;-----  MEDIUM RESOLUTION READ
066E D1 E6              S13:    SAL     SI,1                    ; OFFSET*2 SINCE 2 BYTES/CHAR
0670 B6 04                      MOV     DH,4                    ; NUMBER OF PASSES
0672                    S14:
0672 E8 06D3 R                  CALL    S23                     ; GET BYTES FROM REGEN INTO SINGLE SAVE
0675 81 C6 1FFE                 ADD     SI,2000H-2              ; GO TO LOWER REGION
0679 E8 06D3 R                  CALL    S23                     ; GET THIS PAIR INTO SAVE
067C 81 EE 1FB2                 SUB     SI,2000H-80+2           ; ADJUST POINTER BACK INTO UPPER
0680 FE CE                      DEC     DH
0682 75 EE                      JNZ     S14                     ; KEEP GOING UNTIL ALL 8 DONE
                        ;-----  SAVE AREA HAS CHARACTER IN IT; MATCH IT
0684                    S15:                                    ; FIND CHAR
0684 BF 0000 E                  MOV     DI,OFFSET CRT_CHAR_GEN  ; ESTABLISH ADDRESSING
0687 0E                         PUSH    CS
0688 07                         POP     ES                      ; CODE POINTS IN CS
0689 83 ED 08                   SUB     BP,8                    ; ADJUST POINTER TO START OF SAVE AREA
068C 8B F5                      MOV     SI,BP
068E B0 00                      MOV     AL,0                    ; CURRENT CODE POINT BEING MATCHED
0690                    S16:
0690 16                         PUSH    SS                      ; ESTABLISH ADDRESSING TO STACK
0691 1F                         POP     DS                      ; FOR THE STRING COMPARE
0692 BA 0080                    MOV     DX,128                  ; NUMBER TO TEST AGAINST
0695                    S17:
0695 56                         PUSH    SI                      ; SAVE SAVE AREA POINTER
0696 57                         PUSH    DI                      ; SAVE CODE POINTER
0697 B9 0004                    MOV     CX,4                    ; NUMBER OF WORDS TO MATCH
069A F3/ A7                     REPE    CMPSW                   ; COMPARE THE 8 BYTES AS WORDS
069C 5F                         POP     DI                      ; RECOVER THE POINTERS
069D 5E                         POP     SI
069E 74 1E                      JZ      S18                     ; IF ZERO FLAG SET, THEN MATCH OCCURRED
06A0 FE C0                      INC     AL                      ; NO MATCH, MOVE ON TO NEXT
06A2 83 C7 08                   ADD     DI,8                    ; NEXT CODE POINT
06A5 4A                         DEC     DX                      ; LOOP CONTROL
06A6 75 ED                      JNZ     S17                     ; DO ALL OF THEM
                        ;-----  CHAR NOT MATCHED, MIGHT BE IN USER SUPPLIED SECOND HALF
06A8 3C 00                      CMP     AL,0                    ;  AL<> 0 IF ONLY 1ST HALF SCANNED
06AA 74 12                      JE      S18                     ;  IF = 0, THEN ALL HAS BEEN SCANNED
06AC 2B C0                      SUB     AX,AX
06AE 8E D8                      MOV     DS,AX                   ;  ESTABLISH ADDRESSING TO VECTOR
                                ASSUME  DS:ABS0
06B0 C4 3E 007C R               LES     DI,@EXT_PTR             ;  GET POINTER
06B4 8C C0                      MOV     AX,ES                   ;  SEE IF THE POINTER REALLY EXISTS
06B6 0B C7                      OR      AX,DI                   ; IF ALL 0, THEN DOESN'T EXIST
06B8 74 04                      JZ      S18                     ; NO SENSE LOOKING
06BA B0 80                      MOV     AL,128                  ; ORIGIN FOR SECOND HALF
06BC EB D2                      JMP     S16                     ; GO BACK AND TRY FOR IT
                                ASSUME  DS:DATA
                        ;-----  CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
06BE 83 C4 08           S18:    ADD     SP,8                    ; READJUST THE STACK, THROW AWAY SAVE
06C1 E9 013D R                  JMP     VIDEO_RETURN            ; ALL DONE
06C4                    GRAPHICS_READ   ENDP
                        ;--------------------------------------
                        ; EXPAND_BYTE
                        ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
                        ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
                        ;  THE RESULT IS LEFT IN AX
                        ;--------------------------------------
06C4                    S21     PROC    NEAR
06C4 51                         PUSH    CX                      ; SAVE REGISTER
06C5 B9 0008                    MOV     CX,8                    ; SHIFT COUNT REGISTER FOR ONE BYTE
06C8                    S22:
06C8 D0 C8                      ROR     AL,1                    ; SHIFT BITS, LOW BIT INTO CARRY FLAG
06CA D1 DD                      RCR     BP,1                    ; MOVE CARRY FLAG (LOW BIT) INTO RESULTS
06CC D1 FD                      SAR     BP,1                    ; SIGN EXTEND HIGH BIT (DOUBLE IT)
06CE E2 F8                      LOOP    S22                     ; REPEAT FOR ALL 8 BITS
06D0 95                         XCHG    AX,BP                   ; MOVE RESULTS TO PARAMETER REGISTER
06D1 59                         POP     CX                      ; RECOVER REGISTER
06D2 C3                         RET                             ; ALL DONE
06D3                    S21     ENDP
```

```
                        ;--------------------------------------------------
                        ; MED_READ_BYTE
                        ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
                        ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
                        ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
                        ;  POSITION IN THE SAVE AREA
                        ; ENTRY --
                        ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
                        ;  BX = EXPANDED FOREGROUND COLOR
                        ;  BP = POINTER TO SAVE AREA
                        ; EXIT --
                        ;  SI AND BP ARE INCREMENTED
                        ;--------------------------------------------------
06D3                    S23     PROC    NEAR
06D3 AD                         LODSW                   ; GET FIRST BYTE AND SECOND BYTES
06D4 86 C4                      XCHG    AL,AH           ; SWAP FOR COMPARE
06D6 B9 C000                    MOV     CX,0C000H       ; 2 BIT MASK TO TEST THE ENTRIES
06D9 B2 00                      MOV     DL,0            ; RESULT REGISTER
06DB                    S24:
06DB 85 C1                      TEST    AX,CX           ; IS THIS SECTION BACKGROUND?
06DD 74 01                      JZ      S25             ; IF ZERO, IT IS BACKGROUND (CARRY=0)
06DF F9                         STC                     ; WASN'T, SO SET CARRY
06E0                    S25:
06E0 D0 D2                      RCL     DL,1            ; MOVE THAT BIT INTO THE RESULT
06E2 D1 E9                      SHR     CX,1
06E4 D1 E9                      SHR     CX,1            ; MOVE THE MASK TO THE RIGHT BY 2 BITS
06E6 73 F3                      JNC     S24             ; DO IT AGAIN IF MASK DIDN'T FALL OUT
06E8 88 56 00                   MOV     [BP],DL         ; STORE RESULT IN SAVE AREA
06EB 45                         INC     BP              ; ADJUST POINTER
06EC C3                         RET                     ; ALL DONE
06ED                    S23     ENDP
                        ;--------------------------------------------
                        ; V4_POSITION
                        ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
                        ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
                        ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
                        ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
                        ;  BE DOUBLED.
                        ; ENTRY -- NO REGISTERS,MEMORY LOCATION @CURSOR_POSN IS USED
                        ; EXIT--
                        ;  AX CONTAINS OFFSET INTO REGEN BUFFER
                        ;--------------------------------------------
06ED                    S26     PROC    NEAR
06ED A1 0050 R                  MOV     AX,@CURSOR_POSN ; GET CURRENT CURSOR
06F0                    GRAPH_POSN      LABEL   NEAR
06F0 53                         PUSH    BX              ; SAVE REGISTER
06F1 8B D8                      MOV     BX,AX           ; SAVE A COPY OF CURRENT CURSOR
06F3 A0 004A R                  MOV     AL,BYTE PTR @CRT_COLS   ; GET BYTES PER COLUMN
06F6 F6 E4                      MUL     AH              ; MULTIPLY BY ROWS
06F8 D1 E0                      SHL     AX,1
06FA D1 E0                      SHL     AX,1            ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
06FC 2A FF                      SUB     BH,BH           ; ISOLATE COLUMN VALUE
06FE 03 C3                      ADD     AX,BX           ; DETERMINE OFFSET
0700 5B                         POP     BX              ; RECOVER POINTER
0701 C3                         RET                     ; ALL DONE
0702                    S26     ENDP
                        ;--- WRITE_TTY ------------------------------------------------------------------------
                        ;                                                                                     :
                        ;   THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE                          :
                        ;     VIDEO CARDS.  THE INPUT CHARACTER IS WRITTEN TO THE CURRENT                     :
                        ;     CURSOR POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION.                  :
                        ;     IF THE CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN                   :
                        ;     IS SET TO ZERO, AND THE ROW VALUE IS INCREMENTED.  IF THE ROW                   :
                        ;     ROW VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW,               :
                        ;     FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.                    :
                        ;     WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE                   :
                        ;     NEWLY BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS             :
                        ;     LINE BEFORE THE SCROLL, IN CHARACTER MODE.  IN GRAPHICS MODE,                   :
                        ;     THE 0 COLOR IS USED.                                                            :
                        ;   ENTRY --                                                                          :
                        ;     (AH) = CURRENT CRT MODE                                                         :
                        ;     (AL) = CHARACTER TO BE WRITTEN                                                  :
                        ;            NOTE THAT BACK SPACE, CARRIAGE RETURN, BELL AND LINE FEED ARE            :
                        ;            HANDLED AS COMMANDS RATHER THAN AS DISPLAY GRAPHICS CHARACTERS           :
                        ;     (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A GRAPHICS MODE           :
                        ;   EXIT --                                                                           :
                        ;     ALL REGISTERS SAVED THROUGH VIDEO_EXIT (INCLUDING (AX))                         :
                        ;-------------------------------------------------------------------------------------
                                ASSUME  DS:DATA
0702                    WRITE_TTY       PROC    NEAR
0702 97                         XCHG    DI,AX           ; SAVE (AX) REGISTER IN (DI) FOR EXIT
0703 B4 03                      MOV     AH,03H          ; READ CURSOR POSITION
0705 8A 3E 0062 R               MOV     BH,@ACTIVE_PAGE ; GET CURRENT PAGE SETTING
0709 CD 10                      INT     10H             ; READ THE CURRENT CURSOR POSITION
070B 8B C7                      MOV     AX,DI           ; RECOVER CHARACTER FROM (DI) REGISTER

                        ;-----  DX NOW HAS THE CURRENT CURSOR POSITION

070D 3C 0D                      CMP     AL,CR           ; IS IT CARRIAGE RETURN OR CONTROL
070F 76 46                      JBE     U8              ; GO TO CONTROL CHECKS IF IT IS

                        ;-----  WRITE THE CHAR TO THE SCREEN
0711 B4 0A              U0:     MOV     AH,0AH          ; WRITE CHARACTER ONLY COMMAND
0713 B9 0001                    MOV     CX,1            ; ONLY ONE CHARACTER
0716 CD 10                      INT     10H             ; WRITE THE CHARACTER

                        ;-----  POSITION THE CURSOR FOR NEXT CHAR

0718 FE C2                      INC     DL
071A 3A 16 004A R               CMP     DL,BYTE PTR @CRT_COLS   ; TEST FOR COLUMN OVERFLOW
071E 75 33                      JNZ     U7              ;  SET_CURSOR
0720 B2 00                      MOV     DL,0            ; COLUMN FOR CURSOR
0722 80 FE 18                   CMP     DH,25-1         ; CHECK FOR LAST ROW
0725 75 2A                      JNZ     U6              ;  SET_CURSOR_INC

                        ;-----  SCROLL REQUIRED
0727 B4 02              U1:     MOV     AH,02H
0729 CD 10                      INT     10H             ; SET THE CURSOR

                        ;-----  DETERMINE VALUE TO FILL WITH DURING SCROLL

072B A0 0049 R                  MOV     AL,@CRT_MODE    ; GET THE CURRENT MODE
072E 3C 04                      CMP     AL,4
0730 72 06                      JC      U2              ; READ-CURSOR
```

SECTION 5

```
                        CMP     AL,7
                        MOV     BH,0                    ; FILL WITH BACKGROUND
                        JNE     U3                      ; SCROLL-UP
                U2:                                     ; READ-CURSOR
                        MOV     AH,08H                  ; GET READ CURSOR COMMAND
                        INT     10H                     ; READ CHAR/ATTR AT CURRENT CURSOR
                        MOV     BH,AH                   ; STORE IN BH
                U3:                                     ; SCROLL-UP
                        MOV     AX,0601H                ; SCROLL ONE LINE
                        SUB     CX,CX                   ; UPPER LEFT CORNER
                        MOV     DH,25-1                 ; LOWER RIGHT ROW
                        MOV     DL,BYTE PTR @CRT_COLS   ; LOWER RIGHT COLUMN
                        DEC     DL
                U4:                                     ; VIDEO-CALL-RETURN
                        INT     10H                     ; SCROLL UP THE SCREEN
                U5:                                     ; TTY-RETURN
                        XCHG    AX,DI                   ; RESTORE THE ENTRY CHARACTER FROM (DI)
                        JMP     VIDEO_RETURN            ; RETURN TO CALLER

                U6:                                     ; SET-CURSOR-INC
                        INC     DH                      ; NEXT ROW
                U7:                                     ; SET-CURSOR
                        MOV     AH,02H
                        JMP     U4                      ; ESTABLISH THE NEW CURSOR

                ;-----  CHECK FOR CONTROL CHARACTERS
                U8:
                        JE      U9                      ; WAS IT A CARRIAGE RETURN
                        CMP     AL,LF                   ; IS IT A LINE FEED
                        JE      U10                     ; GO TO LINE FEED
                        CMP     AL,07H                  ; IS IT A BELL
                        JE      U11                     ; GO TO BELL
                        CMP     AL,08H                  ; IS IT A BACKSPACE
                        JNE     U0                      ; IF NOT A CONTROL, DISPLAY IT

                ;-----  BACK SPACE FOUND

                        OR      DL,DL                   ; IS IT ALREADY AT START OF LINE
                        JE      U7                      ; SET_CURSOR
                        DEC     DX                      ; NO -- JUST MOVE IT BACK
                        JMP     U7                      ; SET_CURSOR

                ;-----  CARRIAGE RETURN FOUND
                U9:     MOV     DL,0                    ; MOVE TO FIRST COLUMN
                        JMP     U7                      ; SET_CURSOR

                ;-----  LINE FEED FOUND
                U10:    CMP     DH,25-1                 ; BOTTOM OF SCREEN
                        JNE     U6                      ; YES, SCROLL THE SCREEN
                        JMP     U1                      ; NO, JUST SET THE CURSOR

                ;-----  BELL FOUND
                U11:    MOV     CX,1331                 ; DIVISOR FOR 896 HZ TONE
                        MOV     BL,31                   ; SET COUNT FOR 31/64 SECOND FOR BEEP
                        CALL    BEEP                    ; SOUND THE POD BELL
                        JMP     U5                      ; TTY_RETURN
                WRITE_TTY       ENDP
                ;--------------------------------------------------------------------------
                ; LIGHT PEN                                                               :
                ;       THIS ROUTINE TESTS THE LIGHT PEN SWITCH AND THE LIGHT           :
                ;       PEN TRIGGER. IF BOTH ARE SET, THE LOCATION OF THE LIGHT         :
                ;       PEN IS DETERMINED. OTHERWISE, A RETURN WITH NO INFORMATION      :
                ;       IS MADE.                                                        :
                ; ON EXIT:                                                             :
                ;       (AH) = 0 IF NO LIGHT PEN INFORMATION IS AVAILABLE               :
                ;               BX,CX,DX ARE DESTROYED                                  :
                ;       (AH) = 1 IF LIGHT PEN IS AVAILABLE                              :
                ;               (DH,DL) = ROW,COLUMN OF CURRENT LIGHT PEN POSITION      :
                ;               (CH) = RASTER POSITION                                  :
                ;               (BX) = BEST GUESS AT PIXEL HORIZONTAL POSITION          :
                ;--------------------------------------------------------------------------
                        ASSUME  DS:DATA
0781 03 03 05 05 03 03  V1      DB      3,3,5,5,3,3,3,4         ; SUBTRACT_TABLE
     03 04

                ;-----  WAIT FOR LIGHT PEN TO BE DEPRESSED

0789                    READ_LPEN       PROC    NEAR
0789 B4 00                      MOV     AH,0                    ; SET NO LIGHT PEN RETURN CODE
078B 8B 16 0063 R               MOV     DX,@ADDR_6845           ; GET BASE ADDRESS OF 6845
078F 83 C2 06                   ADD     DX,6                    ; POINT TO STATUS REGISTER
0792 EC                         IN      AL,DX                   ; GET STATUS REGISTER
0793 A8 04                      TEST    AL,004H                 ; TEST LIGHT PEN SWITCH
0795 74 03                      JZ      V6_A                    ; GO IF YES
0797 E9 081C R                  JMP     V6                      ; NOT SET, RETURN

                ;-----  NOW TEST FOR LIGHT PEN TRIGGER

079A A8 02              V6_A:   TEST    AL,2                    ; TEST LIGHT PEN TRIGGER
079C 75 03                      JNZ     V7A                     ; RETURN WITHOUT RESETTING TRIGGER
079E E9 0826 R                  JMP     V7

                ;-----  TRIGGER HAS BEEN SET, READ THE VALUE IN

07A1 B4 10              V7A:    MOV     AH,16                   ; LIGHT PEN REGISTERS ON 6845

                ;-----  INPUT REGISTERS POINTED TO BY AH, AND CONVERT TO ROW COLUMN IN (DX)

07A3 8B 16 0063 R               MOV     DX,@ADDR_6845           ; ADDRESS REGISTER FOR 6845
07A7 8A C4                      MOV     AL,AH                   ; REGISTER TO READ
07A9 EE                         OUT     DX,AL                   ; SET IT UP
07AA EB 00                      JMP     $+2                     ; I/O DELAY
07AC 42                         INC     DX                      ; DATA REGISTER
07AD EC                         IN      AL,DX                   ; GET THE VALUE
07AE 8A E8                      MOV     CH,AL                   ; SAVE IN CX
07B0 4A                         DEC     DX                      ; ADDRESS REGISTER
07B1 FE C4                      INC     AH
07B3 8A C4                      MOV     AL,AH                   ; SECOND DATA REGISTER
07B5 EE                         OUT     DX,AL
07B6 42                         INC     DX                      ; POINT TO DATA REGISTER
07B7 EB 00                      JMP     $+2                     ; I/O DELAY
07B9 EC                         IN      AL,DX                   ; GET SECOND DATA VALUE
07BA 8A E5                      MOV     AH,CH                   ; AX HAS INPUT VALUE
```

```
                        PAGE
                        ;-----  AX HAS THE VALUE READ IN FROM THE 6845
07BC 8A 1E 0049 R               MOV     BL,@CRT_MODE
07C0 2A FF                      SUB     BH,BH                   ; MODE VALUE TO BX
07C2 2E: 8A 9F 0781 R           MOV     BL,CS:V1[BX]            ; DETERMINE AMOUNT TO SUBTRACT
07C7 2B C3                      SUB     AX,BX                   ; TAKE IT AWAY
07C9 8B 1E 004E R               MOV     BX,@CRT_START
07CD D1 EB                      SHR     BX,1
07CF 2B C3                      SUB     AX,BX                   ; CONVERT TO CORRECT PAGE ORIGIN
07D1 79 02                      JNS     V2                      ; IF POSITIVE, DETERMINE MODE
07D3 2B C0                      SUB     AX,AX                   ; <0 PLAYS AS 0

                        ;-----  DETERMINE MODE OF OPERATION

07D5                    V2:                                     ; DETERMINE_MODE
07D5 B1 03                      MOV     CL,3                    ; SET *8 SHIFT COUNT
07D7 80 3E 0049 R 04            CMP     @CRT_MODE,4             ; DETERMINE IF GRAPHICS OR ALPHA
07DC 72 2A                      JB      V4                      ; ALPHA_PEN
07DE 80 3E 0049 R 07            CMP     @CRT_MODE,7
07E3 74 23                      JE      V4                      ; ALPHA_PEN

                        ;-----  GRAPHICS MODE

07E5 B2 28                      MOV     DL,40                   ; DIVISOR FOR GRAPHICS
07E7 F6 F2                      DIV     DL                      ; DETERMINE ROW(AL) AND COLUMN(AH)
                                                                ;  AL RANGE 0-99, AH RANGE 0-39

                        ;-----  DETERMINE GRAPHIC ROW POSITION

07E9 8A E8                      MOV     CH,AL                   ; SAVE ROW VALUE IN CH
07EB 02 ED                      ADD     CH,CH                   ; *2 FOR EVEN/ODD FIELD
07ED 8A DC                      MOV     BL,AH                   ; COLUMN VALUE TO BX
07EF 2A FF                      SUB     BH,BH                   ; MULTIPLY BY 8 FOR MEDIUM RES
07F1 80 3E 0049 R 06            CMP     @CRT_MODE,6             ; DETERMINE MEDIUM OR HIGH RES
07F6 75 04                      JNE     V3                      ; NOT_HIGH_RES
07F8 B1 04                      MOV     CL,4                    ; SHIFT VALUE FOR HIGH RES
07FA D0 E4                      SAL     AH,1                    ; COLUMN VALUE TIMES 2 FOR HIGH RES
07FC                    V3:                                     ; NOT_HIGH_RES
07FC D3 E3                      SHL     BX,CL                   ; MULTIPLY *16 FOR HIGH RES

                        ;-----  DETERMINE ALPHA CHAR POSITION

07FE 8A D4                      MOV     DL,AH                   ; COLUMN VALUE FOR RETURN
0800 8A F0                      MOV     DH,AL                   ; ROW VALUE
0802 D0 EE                      SHR     DH,1                    ; DIVIDE BY 4
0804 D0 EE                      SHR     DH,1                    ;    FOR VALUE IN 0-24 RANGE
0806 EB 12                      JMP     SHORT V5                ; LIGHT_PEN_RETURN_SET

                        ;-----  ALPHA MODE ON LIGHT PEN

0808                    V4:                                     ; ALPHA_PEN
0808 F6 36 004A R               DIV     BYTE PTR @CRT_COLS      ; DETERMINE ROW,COLUMN VALUE
080C 8A F0                      MOV     DH,AL                   ; ROWS TO DH
080E 8A D4                      MOV     DL,AH                   ; COLS TO DL
0810 D2 E0                      SAL     AL,CL                   ; MULTIPLY ROWS * 8
0812 8A E8                      MOV     CH,AL                   ; GET RASTER VALUE TO RETURN REGISTER
0814 8A DC                      MOV     BL,AH                   ; COLUMN VALUE
0816 32 FF                      XOR     BH,BH                   ;  TO BX
0818 D3 E3                      SAL     BX,CL
081A                    V5:                                     ; LIGHT_PEN_RETURN_SET
081A B4 01                      MOV     AH,1                    ; INDICATE EVERY THING SET
081C                    V6:                                     ; LIGHT_PEN_RETURN
081C 52                         PUSH    DX                      ; SAVE RETURN VALUE (IN CASE)
081D 8B 16 0063 R               MOV     DX,@ADDR_6845           ; GET BASE ADDRESS
0821 83 C2 07                   ADD     DX,7                    ; POINT TO RESET PARM
0824 EE                         OUT     DX,AL                   ; ADDRESS, NOT DATA, IS IMPORTANT
0825 5A                         POP     DX                      ; RECOVER VALUE
0826                    V7:                                     ; RETURN_NO_RESET
0826 5D                         POP     BP
0827 5F                         POP     DI
0828 5E                         POP     SI
0829 1F                         POP     DS                      ; DISCARD SAVED BX,CX,DX
082A 1F                         POP     DS
082B 1F                         POP     DS
082C 1F                         POP     DS
082D 07                         POP     ES
082E CF                         IRET
082F                    READ_LPEN       ENDP
082F                    CODE    ENDS
                                END
```