	01 = write processor latch for each plane (loaded by previous read) 10 = each plane gets 8 bits of value of corresponding data bit 0-3 (fast color fill) 11 = not valid
[2]	Test condition (0=normal, 1=GRPHC outputs high impedance)
[3]	Read mode (0=selected plane, 1=color compare results)
[4]	Odd/even (1) (normally matches 3C5.04[2])
[5]	Shift even bits from even maps, odd from odd (1) This option implements CGA-compatible, four-color modes with two adjacent bits per pixel.

3CF.06	Miscellaneous
[0]	Graphics (1=disable char gen latches) / alpha (0)
[1]	Chain odd maps after even maps (1)
[2-3]	Memory mapping in processor address space 00 = A000 for 128KB 01 = A000 for 64KB—high-resolution graphics 10 = B000 for 32KB—monochrome alpha 11 = B800 for 32KB—CGA-compatible

3CF.07	Color don't care
[0-3]	Do not consider planes for which bits are set when doing color compare reads (read mode 1).

3CF.08	Bit mask
[0-7]	Bits set to 0 are protected from modification in all planes (that is, they are written from memory latches.) To preserve data, location must be read before writing.

CRT CONTROLLER (write only except where indicated)

Registers marked with an asterisk are new or different from corresponding registers in Motorola 6845 CRT Controller used in the IBM Monochrome Monitor Adapter and the IBM Color/Graphics Monitor Adapter.

3x4	Address
3x5.00	Horizontal total (total characters – 2)
3x5.01	Horizontal display end (total displayed-1)
*3x5.02	Start horizontal blank (character count)
*3x5.03	End horizontal blank
[0-4]	Start blank + blank width in chars –> 5 bits
[5-6]	Display enable skew (0-3 character times)
*3x5.04	Start horizontal retrace (character position)
*3x5.05	End horizontal retrace
*[0-4]	Start retrace + retrace width in chars —> 5 bits
*[5-6]	Horizontal retrace delay (0-3 character times)
*[7]	Start field on odd/even memory address—used for horizontal pixel panning (0=even, 1=odd)
*3x5.06	Vertical total (scan lines)
*3x5.07	Overflow (contains bit 8 for the following values)
*[0]	Vertical total (index 06)
*[1]	Vertical display enable end (index 12)
*[2]	Vertical retrace start (index 10)
*[3]	Start vertical blank (index 15)
*[4]	Line compare (index 18)
*[5]	Cursor location (index 0A)
*3x5.08	Preset row scan
*[0-4]	First scan line after vertical retrace (for vertical pixel panning)
3x5.09	Max scan line (0-31)
3x5.0A	Cursor start (scan line 0-31)
3x5.0B	Cursor end
[0-4]	Last scan line (0-31)
[5-6]	Cursor skew (0-3 characters)
3x5.0C	Start address high
3x5.0D	Start address low Together form a 16-bit word address (see 3x5.05[7])
3x5.0E	Cursor location high (read/write)
3x5.0F	Cursor location low (read/write)
3x5.10	Light pen high (read only)
3x5.11	Light pen low (read only)
*3x5.10	Vertical retrace start
*3x5.11	Vertical retrace end
*[0-3]	Start retrace + width of retrace in scan lines –> 4 bits
*[4]	Clear vertical interrupt (0=clear)
*[5]	Enable vertical interrupt (0=enable on IRQ2)
*3x5.12	Vertical display end (last scan line)
*3x5.13	Offset (additional offset in words to next logical line)
*3x5.14	Underline location (scan line 0-31)
*3x5.15	Start vertical blanking (scan line)
*3x5.16	End vertical blanking
*[0-4]	Start blank + blank width in scan lines –> 5 bits
*3x5.17	Mode control
*[0]	Compatibility mode (0=row scan A0 used for MA13 for 8KB offset between even and odd scan lines in CGA graphics modes)
*[1]	Select row scan counter (0=row scan A1 used for MA14)
*[2]	Horizontal retrace select (0=normal, 1=divide by 2 to double vertical resolution)
*[3]	Count by 2 (0=normal, 1=clock memory address with character clock / 2 for word refresh address)
*[4]	Output control (0=enable, 1=force high impedance)
*[5]	Address wrap for CGA compatibility in word address mode (0=MA13 to MA0 output, 1=MA15 to MA0 out). Use MA13 in odd/even mode with 64KB, MA15 with >64KB.
*[6]	Word address (0) / byte address (1) mode. In WAM, internal MA0-14 are output on MA1-15, and MA13 or MA15 (see 3x5.17[5]) on MA0.
*[7]	Hardware reset (0=reset, 1=normal operation)
*3x5.18	Line compare (scan line) When scan line counter reaches this value, internal MA is cleared to zero. Used for split screen.

LISTING 1: SMALL.ASM

```
Page    60,132
Title   SMALL -- Load EGA 8x8 Font for 25 or 43 Line Screens
Subttl  Thomas V. Hoffmann, January 1985

;--This program selects 80x25 alpha color mode (mode 3), loads the
;  EGA character generator with the 8x8 font, and causes BIOS to
;  recalculate the video parameters for maximum screen dimensions.
;
;   With 350-line displays, this gives 43 lines per screen.
;   With 200-line displays, this gives 25 lines per screen.
;

Stack   segment para stack 'stack'
        dw      64 dup (0)
Stack   Ends

Bdata   segment at 40H          ;-- BIOS data segment
        org     63H
CRTC    dw      ?               ; Base I/O address of CRTC
        org     87H
info    db      ?               ; Bit 0=1 inhibits cursor emulation
Bdata   Ends

Code    segment para public 'code'
Small   proc    far
        Push    es              ; Push ES:0 for return to DOS
        Sub     ax,ax
        Push    ax
        Mov     ax,Bdata        ; Set DS to BIOS data segment
        Mov     ds,ax
        assume  ds:Bdata

        Mov     ax,0003H        ; Set 80-column alpha mode
        Int     10H

        Mov     ax,1112H        ; Load 8x8 font
        Mov     bl,0            ;   into block 0
        Int     10H             ;   and recalc screen

; This code sets the EGA CRTC cursor register directly, after
; inhibiting the BIOS cursor emulation function.  On 350-line
; displays, this prevents BIOS from setting the cursor to lines
; 11 and 12, which are not displayed for 8-line characters.

        Or      info,1          ; Inhibit cursor emulation
        Mov     ax,0100H        ; Set cursor
        Mov     bh,0            ;   for page 0
        Mov     cx,0600H        ;   to last two lines
        Int     10H             ;    (start on 6, off on 0)
page

; This code sets the underline location register in the CRTC to
; the last line of the character box (line 7).  BIOS incorrectly
; sets it to line 8, which is not displayed.

        Mov     dx,CRTC         ; Get CRTC base address
        Mov     al,14H          ; Select underline loc register
        Out     dx,al
        Inc     dx              ; Point DX to CRTC data register
        Mov     al,7            ; Set underline loc to line 7
        Out     dx,al

; This code enables the EGA BIOS print screen routine, which
; can handle non-standard display dimensions.  In this case
; it handles 43 lines of characters on 350-line displays.

        Mov     ax,1200H        ; Select EGA screen print
        Mov     bl,20H          ;   routine
        Int     10H

        Ret                     ; Return to DOS
Small   Endp
Code    Ends
        End
```

LISTING 2: GRAPH16.BAS

```
 1 '-- GRAPH16.BAS  16-Color Graphics Example
 3 '   This program is very slow, even when compiled.
 4 '   It is intended as an example only.
 5 '   The EGA must be the currently active display adapter.
25 '   Runs in Mode E (640 by 200, 16 colors)
30 '   Memory Map:  4 Planes at &HA000
40 '   8 Pixels per byte, non-interleaved
60 DEFINT A-Z
70 CLS
75 '-- The following line is for the compiled version only.
80 CALL SETMODE      '-- Set Mode after BASIC initialization
100 DEF SEG=&HA000 '-- Video buffer
110 INPUT "How many boxes? ", NBOXES
200 FOR BOX=1 TO NBOXES
210    X1=RND*639: Y1=RND*199
220    X2=RND*639: Y2=RND*199
230    C=RND*15
240    GOSUB 900
250 NEXT BOX
260 BEEP
270 WHILE INKEY$="": WEND
280 SYSTEM
900 '----------
901 ' Fill Box from (x1,y1)-(x2,y2) in color C
910 FOR X=X1 TO X2
920   FOR Y=Y1 TO Y2
930     GOSUB 1000
940   NEXT Y
950 NEXT X
960 RETURN
1001 '-- Put Pixel (color=C) at Location (X,Y)
1010 ROWBYTE = INT (X/8)
1020 BITMASK = 2 ^ (7 - (X MOD 8) )
1030 BYTEOFFSET = (Y * 80) + ROWBYTE
1040 ' Mask all but desired pixel position
1050 OUT &H3CE,8         '-- Graphics Bit Mask Register
1060 OUT &H3CF,BITMASK   '-- Mask all but desired pixel
1070 ' Read previous contents to latches (all maps)
1080 OUT &H3C4,2         '-- Sequencer Map Mask
1090 OUT &H3C5,&HFF      '-- Enable all 4 maps
1100 JUNK = PEEK (BYTEOFFSET)
1110 ' Blank the pixel
1120 POKE BYTEOFFSET,0
1130 ' Now set desired color in sequencer map mask
1140 OUT &H3C4,2         '-- Sequencer Map Mask
1150 OUT &H3C5,C         '-- Desired Color
1160 ' Write 1's to selected planes
1170 POKE BYTEOFFSET,&HFF
1180 RETURN
```

LISTING 3: SETMODE.ASM

```
     Page 60,132
Title     SETMODE -- Set Mode E for GRAPH16 BASIC example.

Subttl    Thomas V. Hoffmann, January 1985

Code segment   para public 'code'
     public    SetMode
SetMode   proc far

     Mov  ax,000EH  ; Set 320 by 200, 16 color mode
     Int  10H

     Ret            ; Return to BASIC

SetMode   Endp
Code Ends
     End
```

LISTING 4: DUALFONT.BAS

```
100 '-- Dual Font Example
110 COLOR 7,0: CLS
120 PRINT "EGA Dual Character Fonts"
130 PRINT
140 COLOR   4,0: PRINT "Font 0 is 7x9 in 8x14 box."
150 COLOR 8+4,0: PRINT "Font 1 is 5x7 in 8x8 box."
160 PRINT
170 COLOR   6,0: PRINT "Fonts can be ";
180 COLOR 8+6,0: PRINT              "mixed ";
190 COLOR   6,0: PRINT                    "on a line."
200 PRINT
210 COLOR   1,0: PRINT "Blue is underlined ";
220 COLOR 8+1,0: PRINT                    "in either font"
230 PRINT
240 COLOR   1,6: PRINT " but only on a black background. "
300 WHILE INKEY$="": WEND
310 WHILE INKEY$="": WEND
320 SYSTEM
```