IBM

Personal Computer

IBM Enhanced Graphics Adapter

IBM ENHANCED GRAPHICS ADAPTER

Contents

Description 1

Major Components 3

Modes of Operation 5

Basic Operations 8

Registers 12

Programming Considerations 62

Programming the Registers 62

RAM Loadable Character Generator 69

Creating a 512 Character Set 70

Creating an 80 by 43 Alphanumeric Mode 71

Vertical Interrupt Feature 72

Creating a Split Screen 73

Compatibility Issues 74

Interface 76

Feature Connector 76

Specifications 79

System Board Switches 79

Configuration Switches 80

Direct Drive Connector 83

Light Pen Interface 84

Jumper Descriptions 85

Logic Diagrams 87

BIOS Listing 103

Vectors with Special Meanings 103

Index Index-1

Description

The IBM Enhanced Graphics Adapter (EGA) is a graphics controller that supports both color and monochrome direct drive displays in a variety of modes. In addition to the direct drive port, a light pen interface is provided. Advanced features on the adapter include bit-mapped graphics in four planes and a RAM (Random Access Memory) loadable character generator. Design features in the hardware substantially reduce the software overhead for many graphics functions.

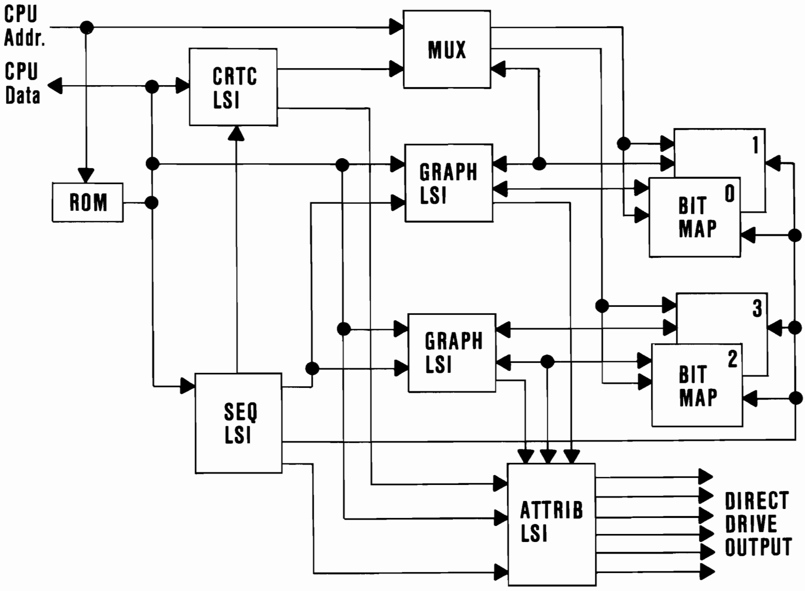
The Enhanced Graphics Adapter provides Basic Input Output System (BIOS) support for both alphanumeric (A/N) modes and all-points-addressable (APA) graphics modes, including all modes supported by the Monochrome Display Adapter and the Color/Graphics Monitor Adapter. Other modes provide APA 640x350 pel graphics support for the IBM Monochrome Display, full 16 color support in both 320x200 pel and 640x200 pel resolutions for the IBM Color Display, and both A/N and APA support with resolution of 640x350 for the IBM Enhanced Color Display. In alphanumeric modes, characters are formed from one of two ROM (Read Only Memory) character generators on the adapter. One character generator defines 7x9 characters in a 9x14 character box. For Enhanced Color Display support, the 9x14 character set is modified to provide an 8x14 character set. The second character generator defines 7x7 characters in an 8x8 character box. These generators contain dot patterns for 256 different characters. The character sets are identical to those provided by the IBM Monochrome Display Adapter and the IBM Color/Graphics Monitor Adapter.

The adapter contains 64K bytes of storage configured as four 16K byte bit planes. Memory expansion options are available to expand the adapter memory to 128K bytes or 256K bytes.

The adapter is packaged on a single 13-1/8 inch (333.50 mm) card. The direct drive port is a right-angle mounted connector at the rear of the adapter and extends through the rear panel of the system unit. Also on the card are five large scale integration (LSI) modules custom designed for this controller.

Located on the adapter is a feature connector that provides access to internal functions through a 32-pin berg connector. A separate 64-pin connector provides an interface for graphics memory expansion.

The following is a block diagram of the Enhanced Graphics Adapter:



Enhanced Graphics Adapter Block Diagram

Major Components

CRT Controller

The CRT (Cathode Ray Tube) Controller (CRTC) generates horizontal and vertical synchronous timings, addressing for the regenerative buffer, cursor and underline timings, and refresh addressing for the dynamic RAMs.

Sequencer

The Sequencer generates basic memory timings for the dynamic RAMs and the character clock for controlling regenerative memory fetches. It allows the processor to access memory during active display intervals by inserting dedicated processor memory cycles periodically between the display memory cycles. Map mask registers are available to protect entire memory maps from being changed.

Graphics Controller

The Graphics Controller directs the data from the memory to the attribute controller and the processor. In graphics modes, memory data is sent in serialized form to the attribute chip. In alpha modes the memory data is sent in parallel form, bypassing the graphics controller. The graphics controller formats the data for compatible modes and provides color comparators for use in color painting modes. Other hardware facilities allow the processor to write 32 bits in a single memory cycle, (8 bits per plane) for quick color presetting of the display areas, and additional logic allows the processor to write data to the display on non-byte boundaries.

Attribute Controller

The Attribute Controller provides a color palette of 16 colors, each of which may be specified separately. Six color outputs are available for driving a display. Blinking and underlining are controlled by this chip. This chip takes data from the display memory and formats it for display on the CRT screen.

Display Buffer

The display buffer on the adapter consists of 64K bytes of dynamic read/write memory configured as four 16K byte video bit planes. Two options are available for expanding the graphics memory. The Graphics Memory Expansion Card plugs into the memory expansion connector on the adapter, and adds one bank of 16K to each of the four bit planes, increasing the graphics memory to 128K bytes. The expansion card also provides DIP sockets for further memory expansion. Populating the DIP sockets with the Graphics Memory Module Kit adds two additional 16K banks to each bit plane, bringing the graphics memory to its maximum of 256K bytes.

The address of the display buffer can be changed to remain compatible with other video cards and application software. Four locations are provided. The buffer can be configured at segment address hex A0000 for a length of 128K bytes, at hex A0000 for a length of 64K bytes, at hex B0000 for a length of 32K bytes, or at hex B8000 for a length of 32K bytes.

BIOS

A read-only memory (ROM) Basic Input Output System (BIOS) module on the adapter is linked to the system BIOS. This ROM BIOS contains character generators and control code and is mapped into the processor address at hex C0000 for a length of 16K bytes.

Support Logic

The logic on the card surrounding the LSI modules supports the modules and creates latch buses for the CRT controller, the processor, and character generator. Two clock sources (14 MHz and 16 MHz) provide the dot rate. The clock is multiplexed under processor I/O control. Four I/O registers also resident on the card are not part of the LSI devices.

Modes of Operation

IBM Color Display

The following table describes the modes supported by BIOS on the IBM Color Display:

MODE #	TYPE	COLORS	ALPHA FORMAT	BUFFER START	BOX SIZE	MAX. PAGES	RESOLUTION
0	A/N	16	40x25	B8000	8x8	8	320x200
1	A/N	16	40x25	B8000	8x8	8	320x200
2	A/N	16	80x25	B8000	8x8	8	640x200
3	A/N	16	80x25	B8000	8x8	8	640x200
4	APA	4	40x25	B8000	8x8	1	320x200
5	APA	4	40x25	B8000	8x8	1	320x200
6	APA	2	80x25	B8000	8x8	1	640x200
D	APA	16	40x25	A0000	8x8	2/4/8	320x200
E	APA	16	80x25	A0000	8x8	1/2/4	640x200

Modes 0 through 6 emulate the support provided by the IBM Color/Graphics monitor Adapter.

Modes 0,2 and 5 are identical to modes 1,3 and 4 respectively at the adapter's direct drive interface.

The Maximum Pages fields for modes D and E indicate the number of pages supported when 64K, 128K or 256K bytes of graphics memory is installed, respectively.

IBM Monochrome Display

The following table describes the modes supported by BIOS on the IBM Monochrome Display.

MODE #	TYPE	COLORS	ALPHA FORMAT	BUFFER START	BOX SIZE	MAX. PAGES	RESOLUTION
7	A/N	4	80x25	B0000	9x14	8	720x350
F	APA	4	80x25	A0000	8x14	1/2	640x350

Mode 7 emulates the support provided by the IBM Monochrome Display Adapter.

IBM Enhanced Color Display

The Enhanced Graphics Adapter supports attachment of the IBM Enhanced Color Display. The IBM Enhanced Color Display is capable of running at the standard television frequency of 15.75 KHz as well as running 21.85 KHz. The table below summarizes the characteristics of the IBM Enhanced Color Display:

Parameter	TV Frequency	High Resolution
Horiz Scan Rate	15.75 KHz.	21.85 KHz.
Vertical Scan Rate	60 Hz.	60 Hz.
Video Bandwidth	14.318 MHz.	16.257 MHz.
Displayable Colors	16 Maximum	16 or 64
Character Size	7 by 7 Pels	7 by 9 Pels
Character Box Size	8 by 8 Pels	8 by 14 Pels
Maximum Resolution	640x200 Pels	640 by 350 Pels
Alphanumeric Modes	0,1,2,3	0,1,2,3
Graphics Modes	4,5,6,D,E	10

In the television frequency mode, the IBM Enhanced Color Display displays information identical in color and resolution to the IBM Color Display.

In the high resolution mode, the adapter provides enhanced alphanumeric character support. This enhanced alphanumeric support consists of transforming the 8 by 8 character box into an 8 by 14 character box, and providing 16 colors out of a palette of

64 possible display colors. Display colors are changed by altering the programming of the color palette registers in the Attribute Controller. In alphanumeric modes, any 16 of 64 colors are displayable. the screen resolution is 320x350 for modes 0 and 1, and 640x350 for modes 2 and 3.

The resolution displayed on the IBM Enhanced Color Display is selected by the switch settings on the Enhanced Graphics Adapter.

The Enhanced Color Display is compatible with all modes listed for the IBM Color Display. the following table describes additional modes supported by BIOS for the IBM Enhanced Color Display:

MODE #	TYPE	COLORS	ALPHA FORMAT	BUFFER START	BOX SIZE	MAX. PAGES	RESOLUTION
0*	A/N	16/64	40x25	B8000	8x14	8	320x350
1*	A/N	16/64	40x25	B8000	8x14	8	320x350
2*	A/N	16/64	80x25	B8000	8x14	8	640x350
3*	A/N	16/64	80x25	B8000	8x14	8	640x350
10*	APA	4/16 16/64	80x25	A0000	8x14	1/2	640x350

* Note that modes 0, 1, 2, and 3, are also listed for IBM Color Display support. BIOS provides enhanced support for these modes when an Enhanced Color Display is attached.

The values in the "COLORS" field indicate 16 colors of a 64 color palette or 4 colors of a sixteen color palette.

In mode 10, The dual values for the "COLORS" field and the "MAX. PAGES" field indicate the support provided when 64K or when greater than 64K of graphics memory is installed, respectively.

Basic Operations

Alphanumeric Modes

The data format for alphanumeric modes on the Enhanced Graphics Adapter is the same as the data format on the IBM Color/Graphics Monitor Adapter and the IBM Monochrome Display Adapter. As an added function, bit three of the attribute byte may be redefined by the Character Map Select register to act as a switch between character sets. This gives the programmer access to 512 characters at one time. This function is valid only when memory has been expanded to 128K bytes or more.

When an alphanumeric mode is selected, the BIOS transfers character patterns from the ROM to bit plane 2. The processor stores the character data in bit plane 0, and the attribute data in bit plane 1. The programmer can view bit planes 0 and 1 as a single buffer in alphanumeric modes. The CRTC generates sequential addresses, and fetches one character code byte and one attribute byte at a time. The character code and row scan count address bit plane 2, which contains the character generators. The appropriate dot patterns are then sent to the palette in the attribute chip, where color is assigned according to the attribute data.

Graphics Modes

320x200 Two and Four Color Graphics (Modes 4 and 5)

Addressing, mapping and data format are the same as the 320x200 pel mode of the Color/Graphics Monitor Adapter. The display buffer is configured at hex B8000. Bit image data is stored in bit planes 0 and 1.

640x200 Two Color Graphics (Mode 6)

Addressing, mapping and data format are the same as the 640x200 pel black and white mode of the Color/Graphics

Monitor Adapter. The display buffer is configured at hex B8000. Bit image data is stored in bit plane 0.

640x350 Monochrome Graphics (Mode F)

This mode supports graphics on the IBM Monochrome Display with the following attributes: black, video, blinking video, and intensified video. Resolution of 640x350 requires 56K bytes to support four attributes. By chaining maps 0 and 1, then maps 2 and 3 together, two 32K bit planes can be formed. This chaining is done only when necessary (less than 128K of graphics memory). The first map is the video bit plane, and the second map is the intensity bit plane. Both planes reside at hex address A0000.

Two bits, one from each bit plane, define one picture element (pel) on the screen. The bit definitions for the pels are given in the following table. The video bit plane is denoted by C0 and the Intensity Bit Plane is denoted by C2.

C2	CO	Pixel Color	Valid Attributes
0	0	Black	0
0	1	Video	3
1	0	Blinking Video	C
1	1	Intensified Video	F

The byte organization in memory is sequential. The first eight pels on the screen are defined by the contents of memory in location A000:0H, the second eight pels by location A000:1H, and so on. The first pel within any one byte is defined by bit 7 in the byte. The last pel within the byte is defined by bit 0 in the byte.

Monochrome graphics works in odd/even mode, which means that even CPU addresses go into even bit planes and odd CPU addresses go into odd bit planes. Since both bit planes reside at address A0000, the user must select which plane or planes he desires to update. This is accomplished by the map mask register of the sequencer. (See the table above for valid attributes).

16/64 Color Graphics Modes (Mode 10)

These modes support graphics in 16 colors on either a medium or high resolution monitor. The memory in these modes consists of using all four bit planes. Each bit plane represents a color as shown below. The bit planes are denoted as C0,C1,C2 and C3 respectively.

C0 = Blue Pels
C1 = Green Pels
C2 = Red Pels
C3 = Intensified Pels

Four bits (one from each plane) define one pel on the screen. The color combinations are illustrated in the following table:

I	R	G	B	Color
0	0	0	0	Black
0	0	0	1	Blue
0	0	1	0	Green
0	0	1	1	Cyan
0	1	0	0	Red
0	1	0	1	Magenta
0	1	1	0	Brown
0	1	1	1	White
1	0	0	0	Dark Gray
1	0	0	1	Light Blue
1	0	1	0	Light Green
1	0	1	1	Light Cyan
1	1	0	0	Light Red
1	1	0	1	Light Magenta
1	1	1	0	Yellow
1	1	1	1	Intensified White

The display buffer resides at address A0000. The map mask register of the sequencer is used to select any or all of the bit planes to be updated when a memory write to the display buffer is executed by the CPU.

Color Mapping

The Enhanced Graphics Adapter supports 640x350 Graphics for both the IBM Monochrome and the IBM Enhanced Color

Displays. Four color capability is supported on the EGA without the Graphics Memory Expansion Card (base 64 KB), and sixteen colors are supported when the Graphics Memory Expansion Card is installed on the adapter (128 KB or above). This section describes the differences in the colors displayed depending upon the graphics memory available. Note that colors 0H, 1H, 4H, and 7H map directly regardless of the graphics memory available.

Character Attribute	Monochrome	Mode 10H 64KB	Mode 10H >64KB
00H*	Black	Black	Black
01H*	Video	Blue	Blue
02H	Black	Black	Green
03H	Video	Blue	Cyan
04H*	Blinking	Red	Red
05H	Intensified	White	Magenta
06H	Blinking	Red	Brown
07H*	Intensified	White	White
08H	Black	Black	Dark Gray
09H	Video	Blue	Light Blue
0AH	Black	Black	Light Green
0BH	Video	Blue	Light Cyan
0CH	Blinking	Red	Light Red
0DH	Intensified	White	Light Magenta
0EH	Blinking	Red	Yellow
0FH	Intensified	White	Intensified White

* Graphics character attributes which map directly regardless of the graphics memory available.

Registers

External Registers

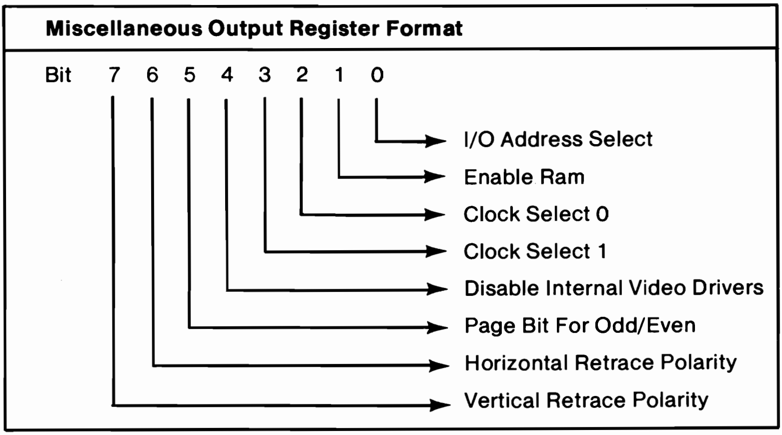
This section contains descriptions of the registers of the Enhanced Graphics Adapter that are not contained in an LSI device.

Name	Port	Index
Miscellaneous Output Register	3C2	-
Feature Control Register	3?A	-
Input Status Register 0	3C2	-
Input Status Register 1	3?2	-
? = B in Monochrome Modes	? = D in Color Modes	

Miscellaneous Output Register

This is a write-only register. The processor output port address is hex 3C2. A hardware reset causes all bits to reset to zero.

Miscellaneous Output Register Format



Bit 0 3BX/3DX CRTC I/O Address—This bit maps the CRTC I/O addresses for IBM Monochrome or Color/Graphics Monitor Adapter emulation. A logical 0 sets CRTC addresses to 3BX and Input Status Register 1 's address to 3BA for Monochrome emulation. A logical 1 sets CRTC

addresses to 3DX and Input Status Register 1's address to 3DA for Color/Graphics Monitor Adapter emulation.

Bit 1 Enable RAM—A logical 0 disables RAM from the processor; a logical 1 enables RAM to respond at addresses designated by the Control Data Select value programmed into the Graphics Controllers.

Bit 2-Bit 3 Clock Select—These two bits select the clock source according to the following table:

Bits		
3	**2**	
0	**0-**	Selects 14 MHz clock from the processor I/O channel
0	**1-**	Selects 16 MHz clock on-board oscillator
1	**0-**	Selects external clock source from the feature connector.
1	**1-**	Not used

Bit 4 Disable Internal Video Drivers—A logical 0 activates internal video drivers; a logical 1 disables internal video drivers. When the internal video drivers are disabled, the source of the direct drive color output becomes the feature connector direct drive outputs.

Bit 5 Page Bit For Odd/Even—Selects between two 64K pages of memory when in the Odd/Even modes (0,1,2,3,7). A logical 0 selects the low page of memory; a logical 1 selects the high page of memory.

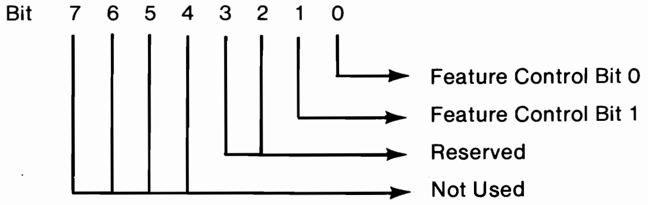
Bit 6 Horizontal Retrace Polarity—A logical 0 selects positive horizontal retrace; a logical 1 selects negative horizontal retrace.

Bit 7 Vertical Retrace Polarity—A logical 0 selects positive vertical retrace; a logical 1 selects negative vertical retrace. The IBM Monochrome display requires a negative vertical retrace polarity.

Feature Control Register

This is a write-only register. The processor output register is hex 3BA or 3DA.

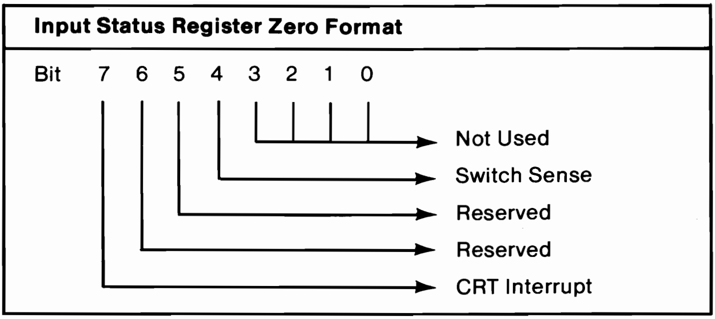
Feature Control Register Format



Bits 0 and 1 Feature Control Bits—These bits are used to convey information to the feature connector. The output of these bits goes to the FEAT 0 (pin 19) and FEAT 1 (pin 17) of the feature connector.

Input Status Register Zero

This is a read-only register. The processor input port address is hex 3C2.



Input Status Register Zero Format

Bit 4 Switch Sense—When set to 1, this bit allows the processor to read the four configuration switches on the board. The setting of the CLKSEL field determines which switch is being read. The switch configuration can be determined by reading byte 40:88H in RAM.

Bit 3: Switch 4 ; Logical 0 = switch closed
Bit 2: Switch 3 ; Logical 0 = switch closed
Bit 1: Switch 2 ; Logical 0 = switch closed
Bit 0: Switch 1 ; Logical 0 = switch closed

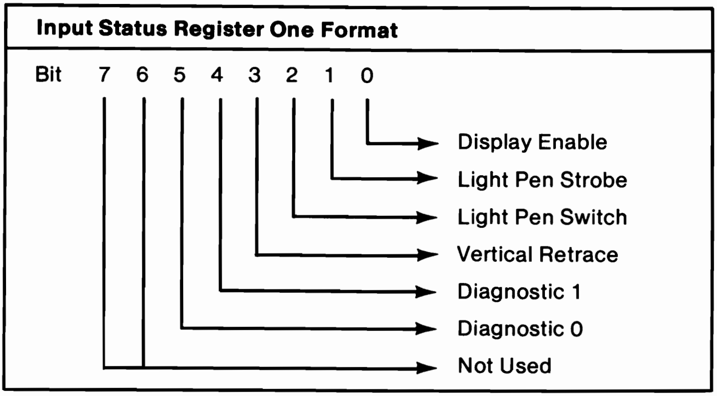
Bits 5 and 6 Feature Code—These bits are input from the Feat (0) and Feat (1) pins on the feature connector.

Bit 7 CRT Interrupt—A logical 1 indicates video is being displayed on the CRT screen; a logical 0 indicates that vertical retrace is occurring.

Input Status Register One

This is a read-only register. The processor port address is hex 3BA or hex 3DA.

Input Status Register One Format



Bit 0 Display Enable—Logical 0 indicates the CRT raster is in a horizontal or vertical retrace interval. This bit is the real time status of the display enable signal. Some programs use this status bit to restrict screen updates to inactive display intervals. The Enhanced Graphics Adapter does not require the CPU to update the screen buffer during inactive display intervals to avoid glitches in the display image.

Bit 1 Light Pen Strobe—A logical 0 indicates that the light pen trigger has not been set; a logical 1 indicates that the light pen trigger has been set.

Bit 2 Light Pen Switch—A logical 0 indicates that the light pen switch is closed; a logical 1 indicates that the light pen switch is open.

Bit 3 Vertical Retrace—A logical 0 indicates that video information is being displayed on the CRT screen; a logical 1 indicates the CRT is in a vertical retrace interval. This bit can be programmed to interrupt the processor on interrupt level 2 at the start of the vertical retrace. This is done through bits 4 and 5 of the Vertical Retrace End Register of the CRTC.

Bits 4 and 5 Diagnostic Usage—These bits are selectively connected to two of the six color outputs of the

Attribute Controller. The Color Plane Enable register controls the multiplexer for the video wiring. The following table illustrates the combinations available and the color output wiring.

Color Plane Register		**Input Status Register One**	
Bit 5	**Bit 4**	**Bit 5**	**Bit 4**
0	0	Red	Blue
0	1	Secondary Blue	Green
1	0	Secondary Red	Secondary Green
1	1	Not Used	Not Used

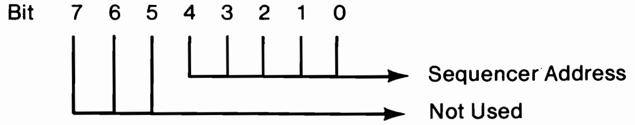
Sequencer Registers

Name	Port	Index
Address	3C4	-
Reset	3C5	00
Clocking Mode	3C5	01
Map Mask	3C5	02
Character Map Select	3C5	03
Memory Mode	3C5	04

Sequencer Address Register

The Address Register is a pointer register located at address hex 3C4. This register is loaded with a binary value that points to the sequencer data register where data is to be written. This value is referred to as "Index" in the table above.

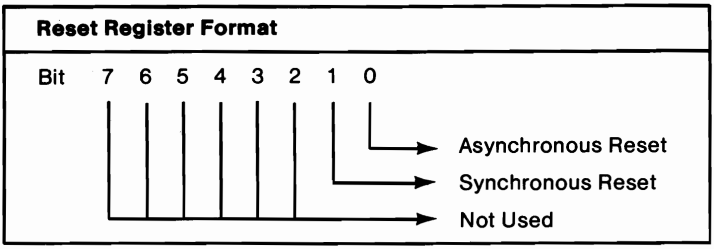
Sequencer Address Register Format



Bit 0–Bit 3 Sequencer Address Bits—A binary value pointing to the register where data is to be written.

Reset Register

This is a write-only register pointed to when the value in the address register is hex 00. The output port address for this register is hex 3C5.



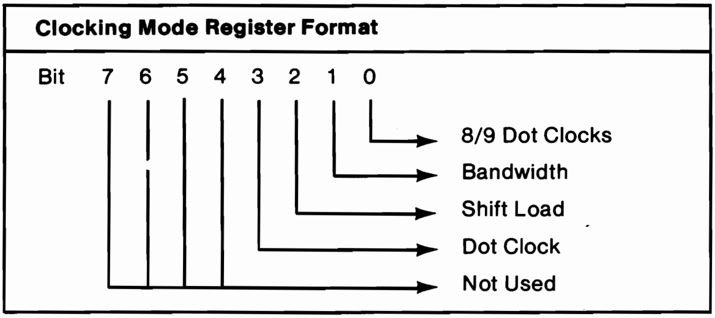
Reset Register Format

Bit 0 Asynchronous Reset—A logical 0 commands the sequencer to asynchronous clear and halt. All outputs are placed in the high impedance state when this bit is a 0. A logical 1 commands the sequencer to run unless bit 1 is set to zero. Resetting the sequencer with this bit can cause data loss in the dynamic RAMs.

Bit 1 Synchronous Reset—A logical 0 commands the sequencer to synchronous clear and halt. Bits 1 and 0 must both be ones to allow the sequencer to operate. Reset the sequencer with this bit before changing the Clocking Mode Register, if memory contents are to be preserved.

Clocking Mode Register

This is a write-only register pointed to when the value in the address register is hex 01. The output port address for this register is hex 3C5.



Clocking Mode Register Format

Bit 0 8/9 Dot Clocks—A logical 0 directs the sequencer to generate character clocks 9 dots wide; a logical 1 directs the sequencer to generate character clocks 8 dots wide. Monochrome alphanumeric mode (07H) is the only mode that uses character clocks 9 dots wide. All other modes must use 8 dots per character clock.

Bit 1 Bandwidth—A logical 0 makes CRT memory cycles occur on 4 out of 5 available memory cycles; a logical 1 makes CRT memory cycles occur on 2 out of 5 available memory cycles. Medium resolution modes require less data to be fetched from the display buffer during the horizontal scan time. This allows the CPU greater access time to the display buffer. All high resolution modes must provide the CRTC with 4 out of 5 memory cycles in order to refresh the display image.

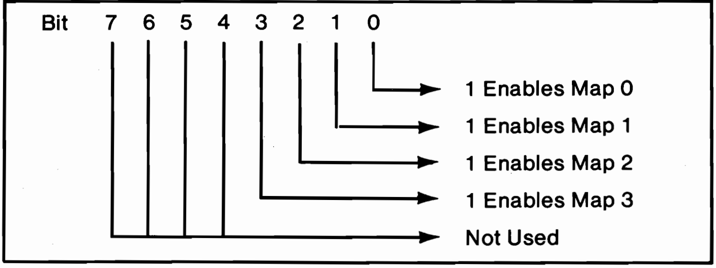
Bit 2 Shift Load—When set to 0, the video serializers are reloaded every character clock; when set to 1, the video serializers are loaded every other character clock. This mode is useful when 16 bits are fetched per cycle and chained together in the shift registers.

Bit 3 Dot Clock—A logical 0 selects normal dot clocks derived from the sequencer master clock input. When this bit is set to 1, the master clock will be divided by 2 to generate the dot clock. All the other timings will be stretched since they are derived from the dot clock. Dot clock divided by two is used for 320x200 modes (0, 1, 4, 5) to provide a pixel rate of 7 MHz, (9 MHz for mode D).

Map Mask Register

This is a write-only register pointed to when the value in the address register is hex 02. The output port address for this register is hex 3C5.

Map Mask Register Format

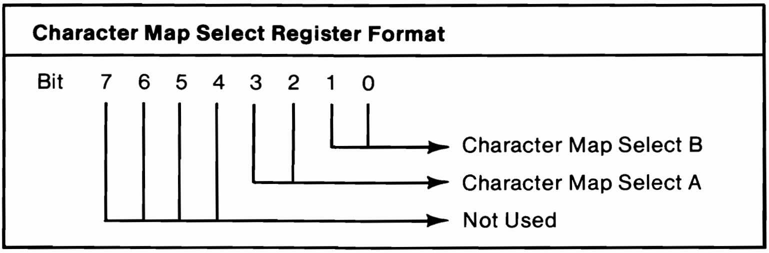


Bit 0–Bit 3 Map Mask—A logical 1 in bits 3 through 0 enables the processor to write to the corresponding maps 3 through 0. If this register is programmed with a value of 0FH, the CPU can perform a 32-bit write operation with only one memory cycle. This substantially reduces the overhead on the CPU during display update cycles in graphics modes. Data scrolling operations are also enhanced by setting this register to a value of 0FH and writing the display buffer address with the data stored in the CPU data latches. This is a read-modify-write operation. When odd/even modes are selected, maps 0 and 1 and maps 2 and 3 should have the same map mask value.

Character Map Select Register

This is a write-only register pointed to when the value in the address register is hex 03. The output port address for this register is 3C5.

Character Map Select Register Format



Bit 0-Bit 1 Character Map Select B—Selects the map used to generate alpha characters when attribute bit 3 is a 0, according to the following table:

Bits		Map	
1	0	Selected	Table Location
Value			
0	0	0	1st 8K of Plane 2 Bank 0
0	1	1	2nd 8K of Plane 2 Bank 1
1	0	2	3rd 8K of Plane 2 Bank 2
1	1	3	4th 8K of Plane 2 Bank 3

Bit 2-Bit 3 Character Map Select A—Selects the map used to generate alpha characters when attribute bit 3 is a 1, according to the following table:

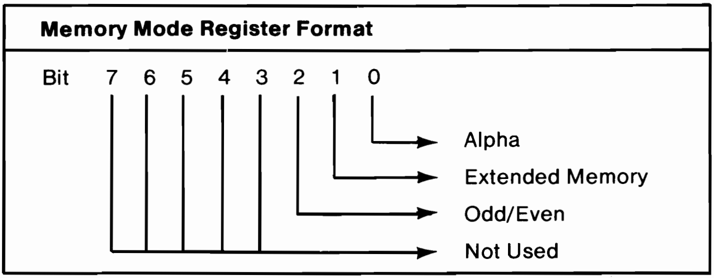
Bits		Map	
3	2	Selected	Table Location
Value			
0	0	0	1st 8K of Plane 2 Bank 0
0	1	1	2nd 8K of Plane 2 Bank 1
1	0	2	3rd 8K of Plane 2 Bank 2
1	1	3	4th 8K of Plane 2 Bank 3

In alphanumeric modes, bit 3 of the attribute byte normally has the function of turning the foreground intensity on or off. This bit however may be redefined as a switch between character sets. This function is enabled when there is a difference between the value in Character Map Select A and the value in Character Map Select B. Whenever these two values are the same, the character select function is disabled. The memory mode register bit 1 must be a 1 (indicates the memory extension card is installed in the unit) to enable this function; otherwise, bank 0 is always selected.

128K of graphics memory is required to support two character sets. 256K supports four character sets. Asynchronous reset clears this register to 0. This should be done only when the sequencer is reset.

Memory Mode Register

This is a write-only register pointed to when the value in the address register is hex 04. The processor output port address for this register is 3C5.



Memory Mode Register Format

Bit 0 Alpha—A logical 0 indicates that a non-alpha mode is active. A logical 1 indicates that alpha mode is active and enables the character generator map select function.

Bit 1 Extended Memory—A logical 0 indicates that the memory expansion card is not installed. A logical 1 indicates that the memory expansion card is installed and enables access to the extended memory through address bits 14 and 15.

Bit 2 Odd/Even—A logical 0 directs even processor addresses to access maps 0 and 2, while odd processor addresses access maps 1 and 3. A logical 1 causes processor addresses to sequentially access data within a bit map. The maps are accessed according to the value in the map mask register.

CRT Controller Registers

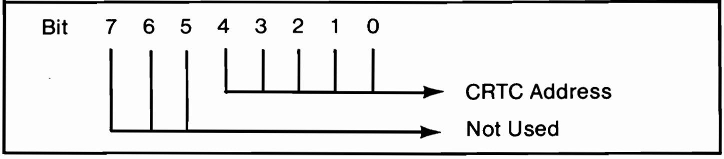
Name	Port	Index
Address Register	3?4	-
Horizontal Total	3?5	00
Horizontal Display End	3?5	01
Start Horizontal Blank	3?5	02
End Horizontal Blank	3?5	03
Start Horizontal Retrace	3?5	04
End Horizontal Retrace	3?5	05
Vertical Total	3?5	06
Overflow	3?5	07
Preset Row Scan	3?5	08
Max Scan Line	3?5	09
Cursor Start	3?5	0A
Cursor End	3?5	0B
Start Address High	3?5	0C
Start Address Low	3?5	0D
Cursor Location High	3?5	0E
Cursor Location Low	3?5	0F
Vertical Retrace Start	3?5	10
Light Pen High	3?5	10
Vertical Retrace End	3?5	11
Light Pen Low	3?5	11
Vertical Display End	3?5	12
Offset	3?5	13
Underline Location	3?5	14
Start Vertical Blank	3?5	15
End Vertical Blank	3?5	16
Mode Control	3?5	17
Line Compare	3?5	18

? = B in Monochrome Modes and D in Color Modes

CRT Controller Address Register

The Address register is a pointer register located at hex 3B4 or hex 3D4. If an IBM Monochrome Display is attached to the adapter, address 3B4 is used. If a color display is attached to the adapter, address 3D4 is used. This register is loaded with a binary value that points to the CRT Controller data register where data is to be written. This value is referred to as "Index" in the table above.

CRT Controller Address Register Format

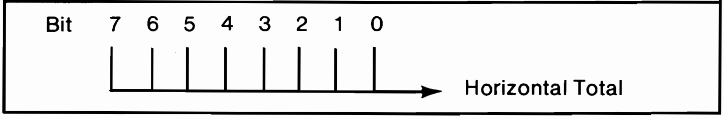


Bit 0-Bit 4 CRT Controller Address Bits—A binary value pointing to the CRT Controller register where data is to be written.

Horizontal Total Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 00. The processor output port address for this register is hex 3B5 or hex 3D5.

Horizontal Total Register Format



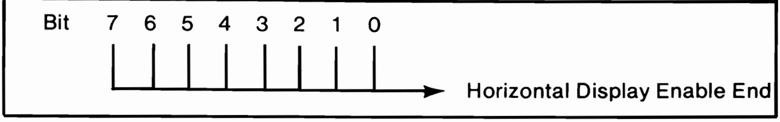
This register defines the total number of characters in the horizontal scan interval including the retrace time. The value directly controls the period of the horizontal retrace output signal. An internal horizontal character counter counts character clock inputs to the CRT Controller, and all horizontal and vertical timings are based upon the horizontal register. Comparators are used to compare register values with horizontal character values to provide horizontal timings.

Bit 0-Bit 7 Horizontal Total—The total number of characters less 2.

Horizontal Display Enable End Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 01. The processor output port address for this register is hex 3B5 or hex 3D5.

Horizontal Display Enable End Register Format



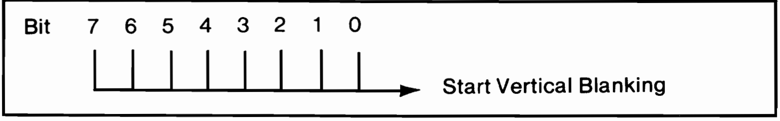
This register defines the length of the horizontal display enable signal. It determines the number of displayed character positions per horizontal line.

Bit 0-Bit 7 Horizontal display enable end —A value one less than the total number of displayed characters.

Start Horizontal Blanking Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 02. The processor output port address for this register is hex 3B5 or hex 3D5.

Start Horizontal Blanking Register Format



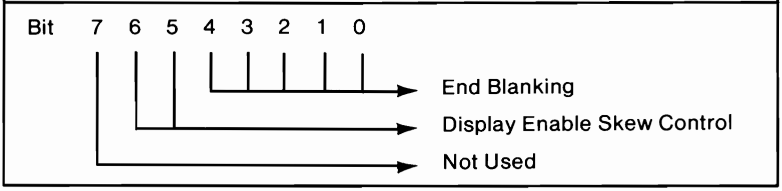
This register determines when the horizontal blanking output signal becomes active. The row scan address and underline scan line decode outputs are multiplexed on the memory address outputs and cursor outputs respectively during the blanking interval. These outputs are latched external to the CRT Controller with the falling edge of the BLANK output signal. The row scan address and underline signals remain on the output signals for one character count beyond the end of the blanking signal.

Bit 0–Bit 7 Start Horizontal Blanking—The horizontal blanking signal becomes active when the horizontal character counter reaches this value.

End Horizontal Blanking Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 03. The processor output port address for this register is hex 3B5 or hex 3D5.

End Horizontal Blanking Register Format



This register determines when the horizontal blanking output signal becomes inactive. The row scan address and underline scan line decode outputs are multiplexed on the memory address outputs and the cursor outputs respectively during the blanking interval. These outputs are latched external to the CRT Controller with the falling edge of the BLANK output signal. The row scan address and underline signals remain on the output signals for one character count beyond the end of the blanking signal.

Bit 0–Bit 4 End Horizontal Blanking—A value equal to the five least significant bits of the horizontal character counter value at which time the horizontal blanking signal becomes inactive (logical 0). To obtain a blanking signal of width W, the following algorithm is used: Value of Start Blanking Register + Width of Blanking signal in character clock units = 5-bit result to be programmed into the End Horizontal Blanking Register.

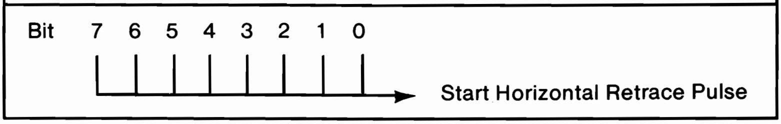
Bit 5-Bit 6 Display Enable Skew Control—These two bits determine the amount of display enable skew. Display enable skew control is required to provide sufficient time for the CRT Controller to access the display buffer to obtain a character and attribute code, access the character generator font, and then go through the Horizontal Pel Panning Register in the Attribute Controller. Each access requires the display enable signal to be skewed one character clock unit so that the video output is in synchronization with the horizontal and vertical retrace signals. The bit values and amount of skew are shown in the following table:

Bits **6 5**	
0 0	Zero character clock skew
0 1	One character clock skew
1 0	Two character clock skew
1 1	Three character clock skew

Start Horizontal Retrace Pulse Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 04. The processor output port address for this register is hex 3B5 or hex 3D5.

Start Horizontal Retrace Pulse Register Format

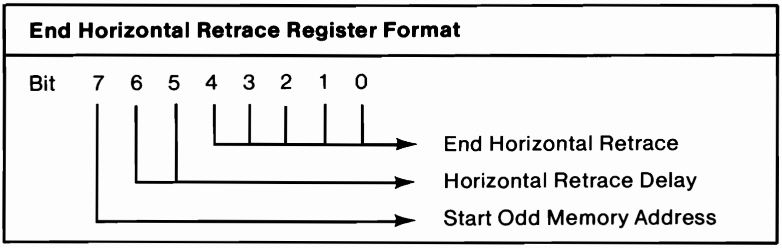


This register is used to center the screen horizontally, and to specify the character position at which the Horizontal Retrace Pulse becomes active.

Bit 0–Bit 7 Start Horizontal Retrace Pulse—The value programmed is a binary count of the character position number at which the signal becomes active.

End Horizontal Retrace Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 05. The processor output port address for this register is hex 3B5 or hex 3D5.


End Horizontal Retrace Register Format

This register specifies the character position at which the Horizontal Retrace Pulse becomes inactive (logical 0).

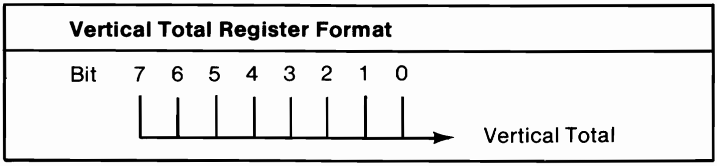
Bit 0–Bit 4 End Horizontal Retrace—A value equal to the five least significant bits of the horizontal character counter value at which time the horizontal retrace signal becomes inactive (logical 0). To obtain a retrace signal of width W, the following algorithm is used: Value of Start Retrace Register + width of horizontal retrace signal in character clock units = 5-bit result to be programmed into the End Horizontal Retrace Register.

Bit 5–Bit 6 Horizontal Retrace Delay—These bits control the skew of the horizontal retrace signal. Binary 00 equals no Horizontal Retrace Delay. For some modes, it is necessary to provide a horizontal retrace signal that takes up the entire blanking interval. Some internal timings are generated by the falling edge of the horizontal retrace signal. To guarantee the signals are latched properly, the retrace signal is started before the end of the display enable signal, and then skewed several character clock times to provide the proper screen centering.

Bit 7 Start Odd/Even Memory Address—This bit controls whether the first CRT memory address output after a horizontal retrace begins with an even or an odd address. A logical 0 selects even addresses; a logical 1 selects odd addresses. This bit is used for horizontal pel panning applications. Generally, this bit should be set to a logical 0.

Vertical Total Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 06. The processor output port address for this register is hex 3B5 or 3D5.

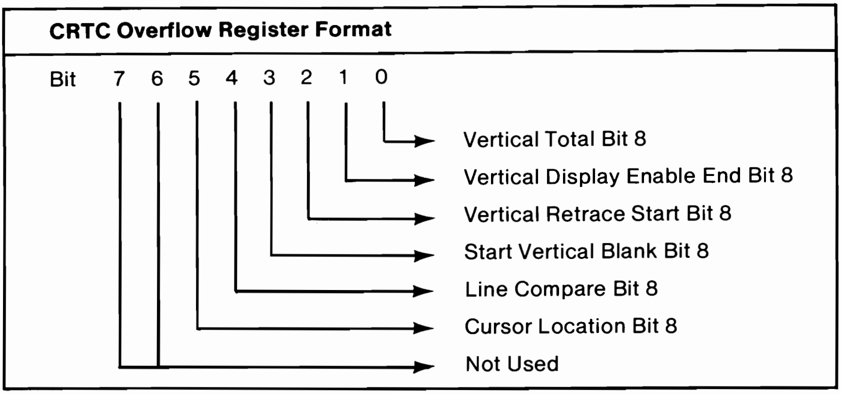

Vertical Total Register Format

Bit 0-Bit 7 Vertical Total—This is the low-order eight bits of a nine-bit register. The binary value represents the number of horizontal raster scans on the CRT screen, including vertical retrace. The value in this register determines the period of the vertical retrace signal. Bit 8 of this register is contained in the CRT Controller Overflow Register hex 07 bit 0.

CRT Controller Overflow Register

This is a write-only register pointed to when the value in the CRT Controller Address Register is hex 07. The processor output port address for this register is hex 3B5 or hex 3D5.

CRTC Overflow Register Format


Bit 7 6 5 4 3 2 1 0
Vertical Total Bit 8
Vertical Display Enable End Bit 8
Vertical Retrace Start Bit 8
Start Vertical Blank Bit 8
Line Compare Bit 8
Cursor Location Bit 8
Not Used

Bit 0 Vertical Total—Bit 8 of the Vertical Total register (index hex 06).

Bit 1 Vertical Display Enable End—Bit 8 of the Vertical Display Enable End register (index hex 12).

Bit 2 Vertical Retrace Start—Bit 8 of the Vertical Retrace Start register (index hex 10).

Bit 3 Start Vertical Blank—Bit 8 of the Start Vertical Blank register (index hex 15).

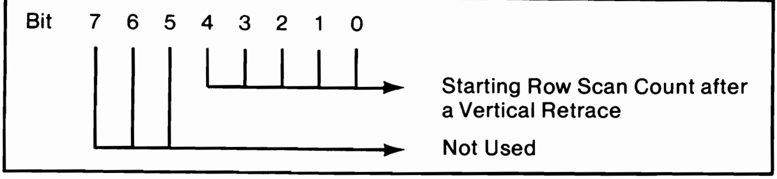
Bit 4 Line Compare—Bit 8 of the Line Compare register (index hex 18).

Bit 5 Cursor Location—Bit 8 of the Cursor Location register (index hex 0A).

Preset Row Scan Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 08. The processor output port address for this register is hex 3B5 or hex 3D5.

Preset Row Scan Register Format



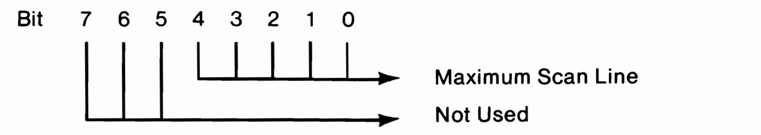
This register is used for pel scrolling.

Bit 0–Bit 4 Preset Row Scan (Pel Scrolling)—This register specifies the starting row scan count after a vertical retrace. The row scan counter increments each horizontal retrace time until a maximum row scan occurs. At maximum row scan compare time the row scan is cleared (not preset).

Maximum Scan Line Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 09. The processor output port address for this register is hex 3B5 or hex 3D5.

Maximum Scan Line Register Format

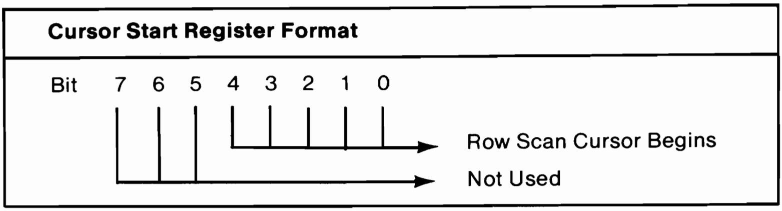


Bit 0–Bit 4 Maximum Scan Line—This register specifies the number of scan lines per character row. The number to be programmed is the maximum row scan number minus one.

Cursor Start Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 0A. The processor output port

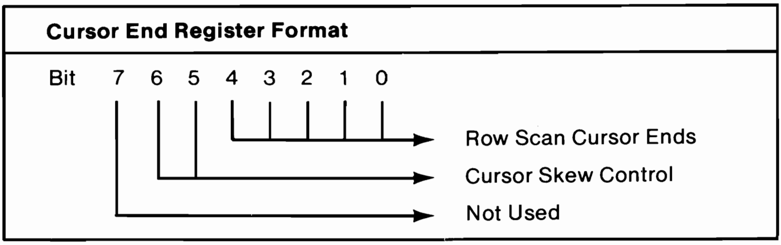
address for this register is hex 3B5 or hex 3D5.



Bit 0-Bit 4 Cursor Start—This register specifies the row scan of a character line where the cursor is to begin. The number programmed should be one less than the starting cursor row scan.

Cursor End Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 0B. The processor output port address for this register is hex 3B5 or hex 3D5.



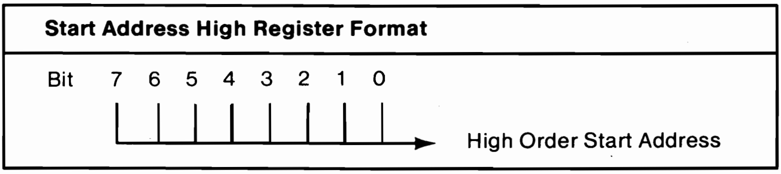
Bit 0-Bit 4 Cursor End—These bits specify the row scan where the cursor is to end.

Bit 5-Bit 6 Cursor Skew—These bits control the skew of the cursor signal.

Bits		
6	5	
0	0	Zero character clock skew
0	1	One character clock skew
1	0	Two character clock skew
1	1	Three character clock skew

Start Address High Register

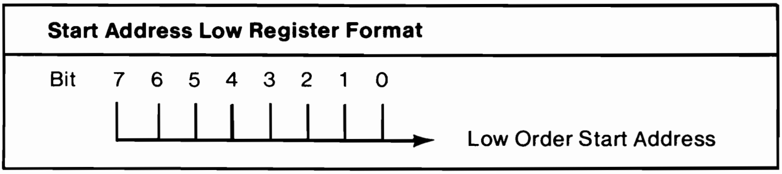
This is a read/write register pointed to when the value in the CRT Controller address register is hex 0C. The processor input/output port address for this register is hex 3B5 or hex 3D5.


Start Address High Register Format

Bit 0–Bit 7 Start Address High—These are the high-order eight bits of the start address. The 16-bit value, from the high-order and low-order start address registers, is the first address after the vertical retrace on each screen refresh.

Start Address Low Register

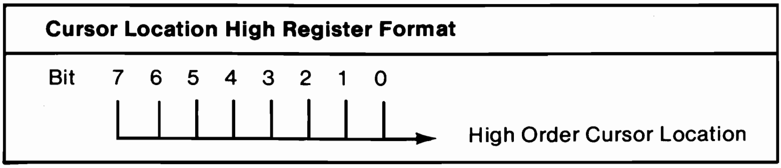
This is a read/write register pointed to when the value in the CRT Controller address register is hex 0D. The processor input/output port address for this register is hex 3B5 or hex 3D5.


Start Address Low Register Format

Bit 0–Bit 7 Start Address Low—These are the low-order 8 bits of the start address.

Cursor Location High Register

This is a read/write register pointed to when the value in the CRT Controller address register is hex 0E. The processor input/output port address for this register is hex 3B5 or hex 3D5.

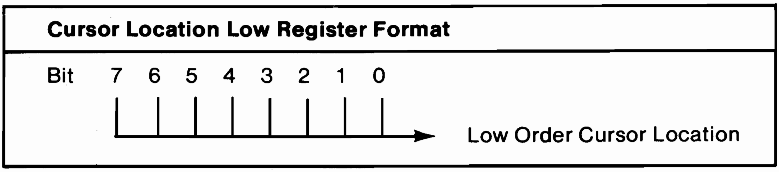


Cursor Location High Register Format

Bit 0–Bit 7 Cursor Location High—These are the high-order 8 bits of the cursor location.

Cursor Location Low Register

This is a read/write register pointed to when the value in the CRT Controller address register is hex 0F. The processor input/output port address for this register is hex 3B5 or Hex 3D5.



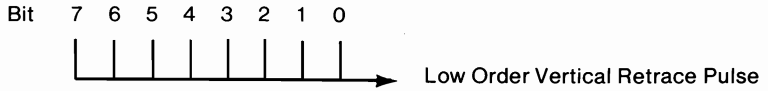
Cursor Location Low Register Format

Bit 0–Bit 7 Cursor Location Low— These are the low-order 8 bits of the cursor location.

Vertical Retrace Start Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 10. The processor output port address for this register is hex 3B5 or hex 3D5.

Vertical Retrace Start Register Format

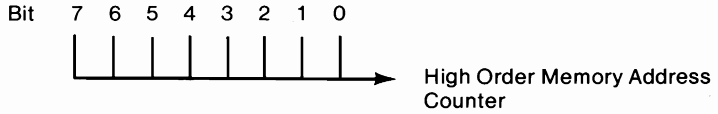


Bit 0–Bit 7 Vertical Retrace Start—This is the low-order 8 bits of the vertical retrace pulse start position programmed in horizontal scan lines. Bit 8 is in the overflow register location hex 07.

Light Pen High Register

This is a read-only register pointed to when the value in the CRT Controller address register is hex 10. The processor input port address for this register is hex 3B5 or hex 3D5.

Light Pen High Register Format



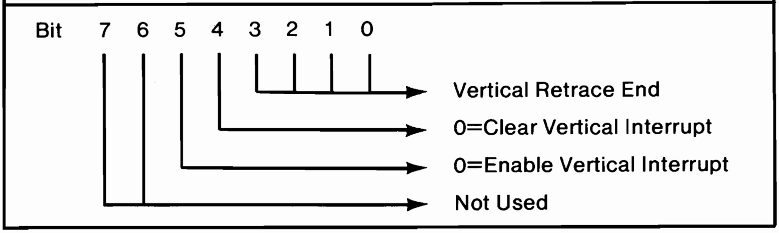
Bit 0–Bit 7 Light Pen High—This is the high order 8 bits of the memory address counter at the time the light pen was triggered.

Vertical Retrace End Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 11. The processor output port

address for this register is hex 3B5 or hex 3D5.

Vertical Retrace End Register Format



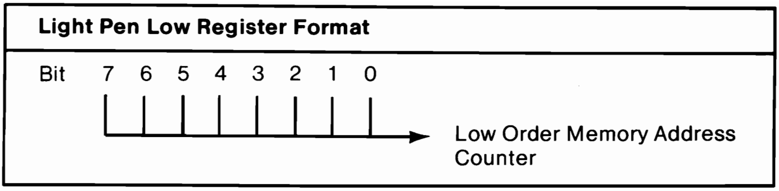
Bit 0-Bit 3 Vertical Retrace End—These bits determine the horizontal scan count value when the vertical retrace output signal becomes inactive. The register is programmed in units of horizontal scan lines. To obtain a vertical retrace signal of width W, the following algorithm is used: Value of Start Vertical Retrace Register + width of vertical retrace signal in horizontal scan units = 4-bit result to be programmed into the End Horizontal Retrace Register.

Bit 4 Clear Vertical Interrupt—A logical 0 will clear a vertical interrupt.

Bit 5 Enable Vertical Interrupt—A logical 0 will enable vertical interrupt.

Light Pen Low Register

This is a read-only register pointed to when the value in the CRT Controller address register is hex 11. The processor input port address for this register is hex 3B5 or 3D5.

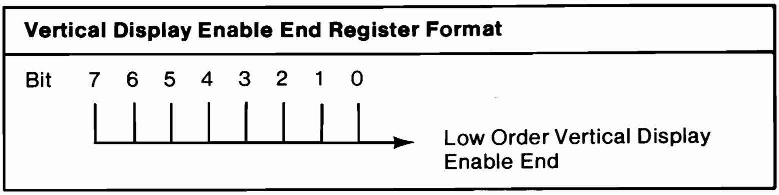


Light Pen Low Register Format

Bit 0-Bit 7 Light Pen Low—This is is the low-order 8 bits of the memory address counter at the time the light pen was triggered.

Vertical Display Enable End Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 12. The processor output port address for this register is hex 3B5 or hex 3D5.



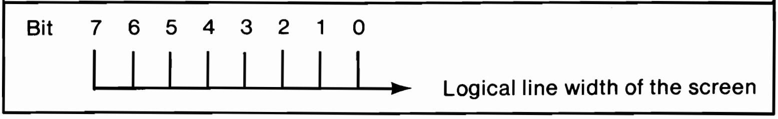
Vertical Display Enable End Register Format

Bit 0-Bit 7 Vertical Display Enable End—These are the low-order 8 bits of the vertical display enable end position. This address specifies which scan line ends the active video area of the screen. Bit 8 is in the overflow register location hex 07.

Offset Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 13. The processor output port address for this register is hex 3B5 or hex 3D5.

Offset Register Format

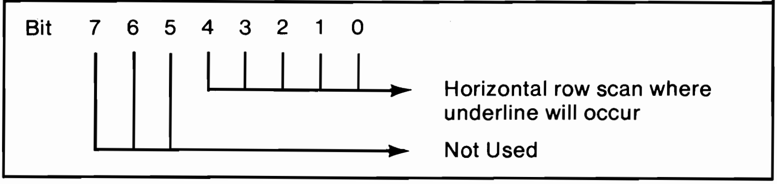


Bit 0–Bit 7 Offset—This register specifies the logical line width of the screen. The starting memory address for the next character row is larger than the current character row by this amount. The Offset Register is programmed with a word address. Depending upon the method of clocking the CRT Controller, this word address is either a word or double word address.

Underline Location Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 14. The processor output port address for this register is hex 3B5 or hex 3D5.

Underline Location Register Format

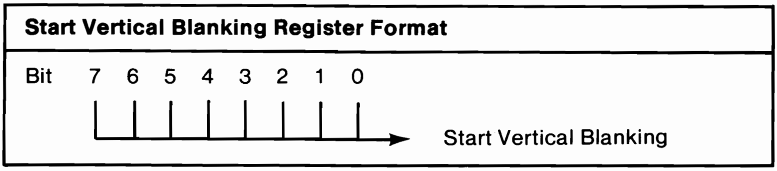


Bit 0–Bit 4 Underline Location—This register specifies the horizontal row scan on which underline will occur. The value programmed is one less than the scan line number desired.

Start Vertical Blanking Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 15. The processor output port

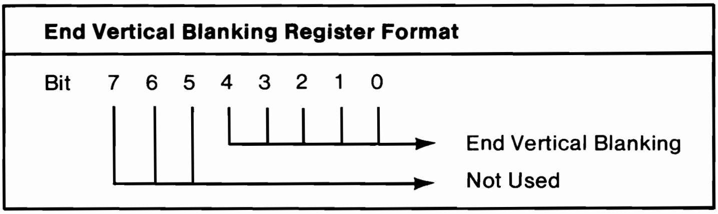
address for this register is hex 3B5 or hex 3D5.


Start Vertical Blanking Register Format

Bit 0–Bit 7 Start Vertical Blank—These are the low 8 bits of the horizontal scan line count, at which the vertical blanking signal becomes active. Bit 8 bit is in the overflow register hex 07.

End Vertical Blanking Register

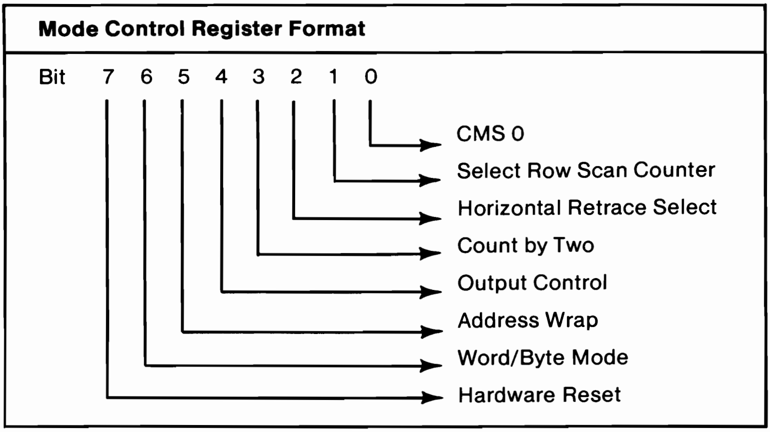
This is a write-only register pointed to when the value in the CRT Controller address register is hex 16. The processor output port address for this register is hex 3B5 or hex 3D5.


End Vertical Blanking Register Format

Bit 0–Bit 4 End Vertical Blank—This register specifies the horizontal scan count value when the vertical blank output signal becomes inactive. The register is programmed in units of horizontal scan lines. To obtain a vertical blank signal of width W, the following algorithm is used: Value of Start Vertical Blank Register + width of vertical blank signal in horizontal scan units = 5-bit result to be programmed into the End Vertical Blank Register.

Mode Control Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 17. The processor output port address for this register is hex 3B5 or hex 3D5.



Mode Control Register Format

Bit 0 Compatibility Mode Support— When this bit is a logical 0, the row scan address bit 0 is substituted for memory address bit 13 during active display time. A logical 1 enables memory address bit 13 to appear on the memory address output bit 13 signal of the CRT Controller. The CRT Controller used on the IBM Color/Graphics Monitor Adapter is the 6845. The 6845 has 128 horizontal scan line address capability. To obtain 640 by 200 graphics resolution, the CRTC was programmed for 100 horizontal scan lines with 2 row scan addresses per character row. Row scan address bit 0 became the most significant address bit to the display buffer. Successive scan lines of the display image were displaced in memory by 8K bytes. This bit allows compatibility with the 6845 and Color Graphics APA modes of operation.

Bit 1 Select Row Scan Counter—A logical 0 selects row scan counter bit 1 on MA 14 output pin. A logical 1 selects MA 14 counter bit on MA 14 output pin.

Bit 2 Horizontal Retrace Select—This bit selects Horizontal Retrace or Horizontal Retrace divided by 2 as the clock that controls the vertical timing counter. This bit can be used to effectively double the vertical resolution capability of the CRT Controller. The vertical counter has a maximum resolution of 512 scan lines due to the 9-bit wide Vertical Total Register. If the vertical counter is clocked with the horizontal retrace divided by 2 clock, then the vertical resolution is doubled to 1024 horizontal scan lines. A logical 0 selects HRTC and a logical 1 selects HRTC divided by 2.

Bit 3 Count By Two— When this bit is set to 0, the memory address counter is clocked with the character clock input. A logical 1 clocks the memory address counter with the character clock input divided by 2. This bit is used to create either a byte or word refresh address for the display buffer.

Bit 4 Output Control—A logical 0 enables the module output drivers. A logical 1 forces all outputs into high impedance state.

Bit 5 Address Wrap—This bit selects Memory Address counter bit MA 13 or bit MA 15, and it appears on the MA 0 output pin in the word address mode. If you are not in the word address mode, MA 0 counter output appears on the MA 0 output pin. A logical 1 selects MA 15. In odd/even mode, bit MA 13 should be selected when the 64K memory is installed on the board. Bit MA 15 should be selected when greater then 64K memory is installed. This function is used to implement Color Graphics Monitor Adapter compatibility.

Bit 6 Word Mode or Byte Mode—When this bit is a logical 0, the Word Mode shifts all memory address counter bits down one bit, and the most significant bit of the counter appears on the least significant bit of the memory address outputs. See table below for address output details. A logical 1 selects the Byte Address mode.

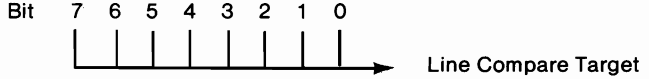
	Internal Memory Address Counter Wiring to the Output Multiplexer	
CRTC Out Pin	**Byte Address Mode**	**Word Address Mode**
MA 0/RFA 0	MA 0	MA 15 or MA 13
MA 1/RFA 1	MA 1	MA 0
MA 2/RFA 2	MA 2	MA 1
MA 3/RFA 3	MA 3	MA 2
*	*	*
*	*	*
*	*	*
MA 14/RS 3	MA 14	MA 13
MA 15/RS 4	MA 15	MA 14

Bit 7 Hardware Reset—A logical 0 forces horizontal and vertical retrace to clear. A logical 1 forces horizontal and vertical retrace to be enabled.

Line Compare Register

This is a write-only register pointed to when the value in the CRT Controller address register is hex 18. The processor output port address for this register is hex 3B5 or hex 3D5.

Line Compare Register Format



Bit 0–Bit 7 Line Compare—This register is the low-order 8 bits of the compare target. When the vertical

counter reaches this value, the internal start of the line counter is cleared. This allows an area of the screen to be immune to scrolling. Bit 8 of this register is in the overflow register hex 07.

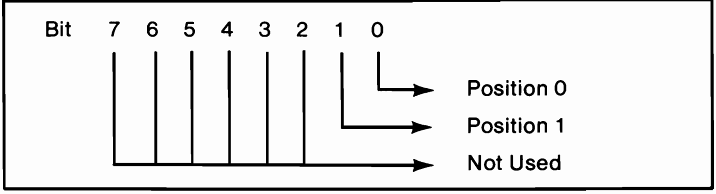
Graphics Controller Registers

Name	Port	Index
Graphics 1 Position	3CC	-
Graphics 2 Position	3CA	-
Graphics 1 & 2 Address	3CE	-
Set/Reset	3CF	00
Enable Set/Reset	3CF	01
Color Compare	3CF	02
Data Rotate	3CF	03
Read Map Select	3CF	04
Mode Register	3CF	05
Miscellaneous	3CF	06
Color Don't Care	3CF	07
Bit Mask	3CF	08

Graphics 1 Position Register

This is a write-only register. The processor output port address for this register is hex 3CC.

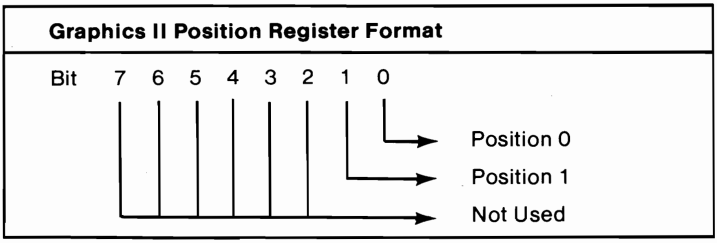
Graphics I Position Register Format



Bit 0-Bit 1 Position—These 2 bits are binary encoded hierarchy bits for the graphics chips. The position register controls which 2 bits of the processor data bus each chip responds to. Graphics 1 must be programmed with a position register value of 0 for this card.

Graphics 2 Position Register

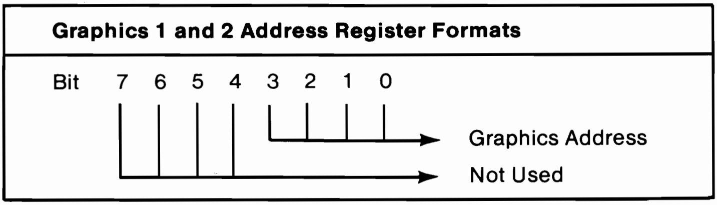
This is a write-only register. The processor output port address for this register is hex 3CA.


Graphics II Position Register Format

Bit 0–Bit 1 Position—These 2 bits are binary encoded hierarchy bits for the graphics chips. The position register controls which 2 bits of the processor data bus to which each chip responds. Graphics 2 must be programmed with a position register value of 1 for this card.

Graphics 1 and 2 Address Register

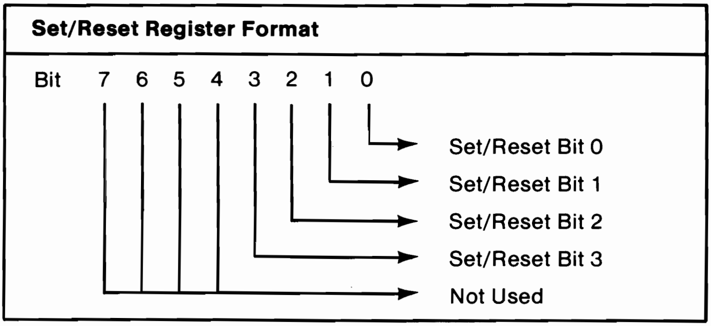
This is a write-only register and the processor output port address for this register is hex 3CE.


Graphics 1 and 2 Address Register Formats

Bit 0–Bit 3 Graphics 1 and 2 Address Bits—This output loads the address register in both graphics chips simultaneously. This register points to the data register of the graphics chips.

Set/Reset Register

This is a write-only register pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 00 before writing can take place. The processor output port address for this register is hex 3CF.

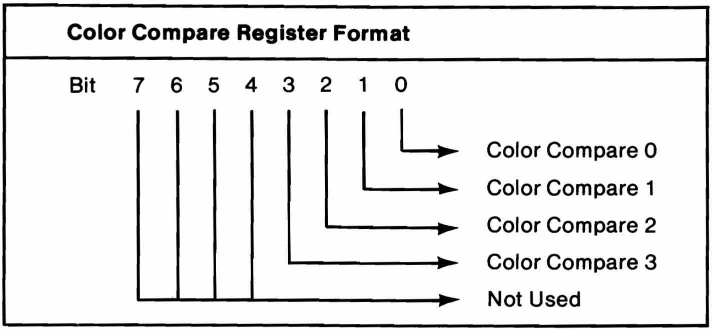


Set/Reset Register Format

Bit 0–Bit 3 Set/Reset—These bits represent the value written to the respective memory planes when the processor does a memory write with write mode 0 selected and set/reset mode is enabled. Set/Reset can be enabled on a plane by plane basis with separate OUT commands to the Set/Reset register.

Enable Set/Reset Register

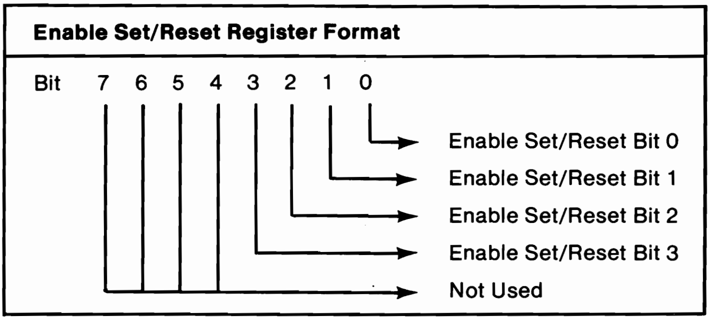
This is a write-only register and is pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 01 before writing can take place. The processor output port for this register is hex 3CF.


Color Compare Register Format

Bit 0–Bit 3 Enable Set/Reset—These bits enable the set/reset function. The respective memory plane is written with the value of the Set/Reset register provided the write mode is 0. When write mode is 0 and Set/Reset is not enabled on a plane, that plane is written with the value of the processor data.

Color Compare Register

This is a write-only register pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 02 before writing can take place. The processor output port address for this register is hex 3CF.

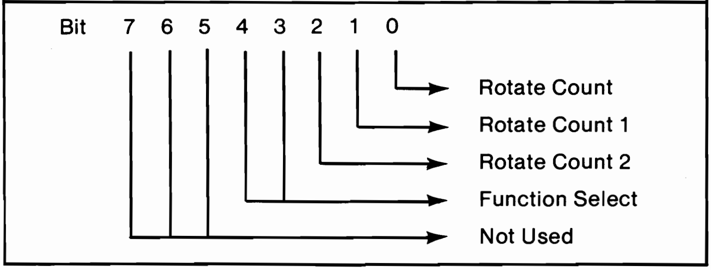

Enable Set/Reset Register Format

Bit 0–Bit 3 Color Compare—These bits represent a 4 bit color value to be compared. If the processor sets read mode 1 on the graphics chips, and does a memory read, the data returned from the memory cycle will be a 1 in each bit position where the 4 bit planes equal the color compare register.

Data Rotate Register

This is a write-only register pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 03 before writing can take place. The processor output port address for this register is hex 3CF.

Data Rotate Register Format



Bit 0–Bit 2 Rotate Count—These bits represent a binary encoded value of the number of positions to rotate the processor data bus during processor memory writes. This operation is done when the write mode is 0. To write unrotated data the processor must select a count of 0.

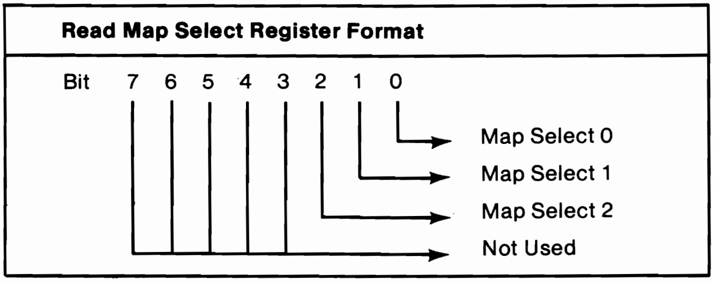
Bit 3–Bit 4 Function Select—Data written to memory can operate logically with data already in the processor latches. The bit functions are defined in the following table.

Bits		
4	3	
0	0	Data unmodified.
0	1	Data AND'ed with latched data.
1	0	Data OR'ed with latched data.
1	1	Data XOR'ed with latched data.

Data may be any of the choices selected by the Write Mode Register except processor latches. If rotated data is selected, the rotate applies before the logical function.

Read Map Select Register

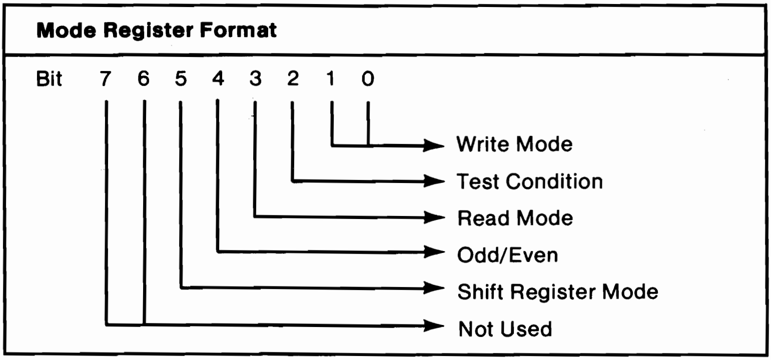
This is a write-only register pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 04 before writing can take place. The processor output port address for this register is hex 3CF.


Read Map Select Register Format

Bit 0-Bit 2 Map Select—These bits represent a binary encoded value of the memory plane number from which the processor reads data. This register has no effect on the color compare read mode described elsewhere in this section.

Mode Register

This is a write-only register pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 05 before writing can take place. The processor output port address for this register is 3CF.



Bit 0–Bit 1 Write Mode

Bits
1 0

0 0 Each memory plane is written with the processor data rotated by the number of counts in the rotate register, unless Set/Reset is enabled for the plane. Planes for which Set/Reset is enabled are written with 8 bits of the value contained in the Set/Reset register for that plane.

0 1 Each memory plane is written with the contents of the processor latches. These latches are loaded by a processor read operation.

1 0 Memory plane *n* (0 through 3) is filled with 8 bits of the value of data bit *n*.

1 1 Not Valid

The logic function specified by the function select register also applies.

Bit 2 Test Condition—A logical 1 directs graphics controller outputs to be placed in high impedance state for testing.

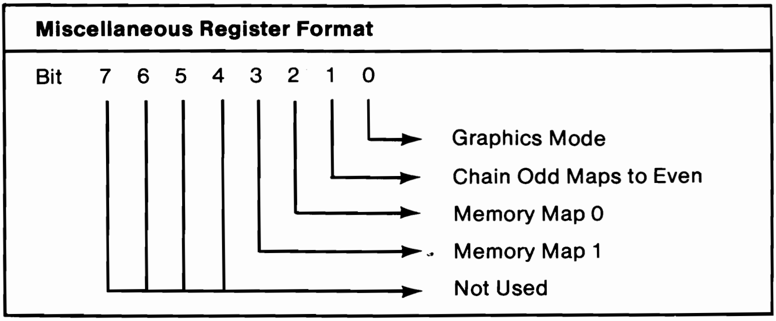
Bit 3 Read Mode—When this bit is a logical 0, the processor reads data from the memory plane selected by the read map select register. When this bit is a logical 1, the processor reads the results of the comparison of the 4 memory planes and the color compare register.

Bit 4 Odd/Even—A logical 1 selects the odd/even addressing mode, which is useful for emulation of the Color Graphics Monitor Adapter compatible modes. Normally the value here follows the value of the Memory Mode Register bit 3 of the Sequencer.

Bit 5 Shift Register—A logical 1 directs the shift registers on each graphics chip to format the serial data stream with even numbered bits on the even numbered maps and odd numbered bits on the odd maps.

Miscellaneous Register

This is a write-only register pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 06 before writing can take place. The processor output port for this register is hex 3CF.


Miscellaneous Register Format

Bit 0 Graphics Mode—This bit controls alpha-mode addressing. A logical 1 selects graphics mode. When set to graphics mode, the character generator address latches are disabled.

Bit 1 Chain Odd Maps To Even Maps—When set to 1, this bit directs the processor address bit 0 to be replaced by a higher order bit and odd/even maps to be selected with odd/even values of the processor A0 bit, respectively.

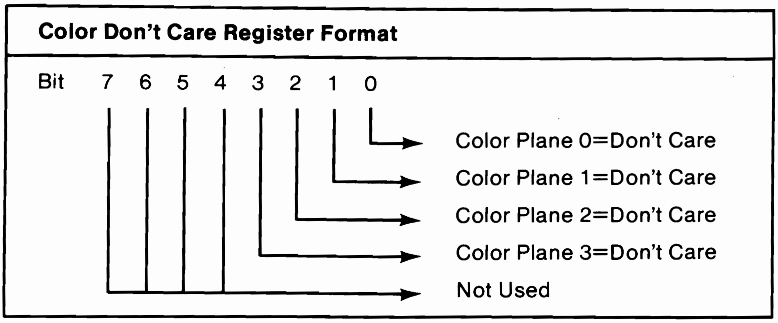
Bit 2-Bit 3 Memory Map—These bits control the mapping of the regenerative buffer into the processor address space.

Bits **3 2**	
0 0	Hex A000 for 128K bytes.
0 1	Hex A000 for 64K bytes.
1 0	Hex B000 for 32K bytes
1 1	Hex B800 for 32K bytes.

If the display adapter is mapped at address hex A000 for 128K bytes, no other adapter can be installed in the system.

Color Don't Care Register

This is a write-only register and is pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 07 before writing can take place. The processor output port for this register is hex 3CF.


Color Don't Care Register Format

Bit 0 Color Don't Care—Color plane 0=don't care when reading color compare when this bit is set to 1.

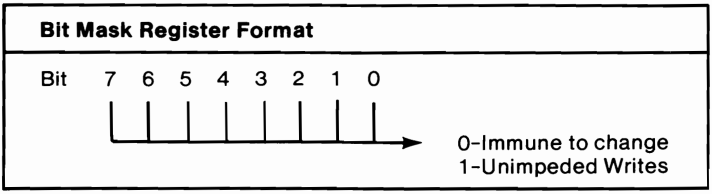
Bit 1 Color Don't Care—Color plane 1=don't care when reading color compare when this bit is set to 1.

Bit 2 Color Don't Care—Color plane 2=don't care when reading color compare when this bit is set to 1.

Bit 3 Color Don't Care—Color plane 3=don't care when reading color compare when this bit is set to 1.

Bit Mask Register

This is a write-only register and is pointed to by the value in the Graphics 1 and 2 address register. This value must be hex 08 before writing can take place. The processor output port for this register is hex 3CF.


Bit Mask Register Format

Bit 0-Bit 7 Bit Mask—Any bit programmed to *n* causes the corresponding bit *n*in each bit plane to be immune to change provided that the location being written was the last location read by the processor. Bits programmed to a 1 allow unimpeded writes to the corresponding bits in the bit planes.

The bit mask applies to any data written by the processor (rotate, AND'ed, OR'ed, XOR'ed, DX, and S/R). To preserve bits using the bit mask, data must be latched internally by reading the location. When data is written to preserve the bits, the most current data in latches is written in those positions. The bit mask applies to all bit planes simultaneously.

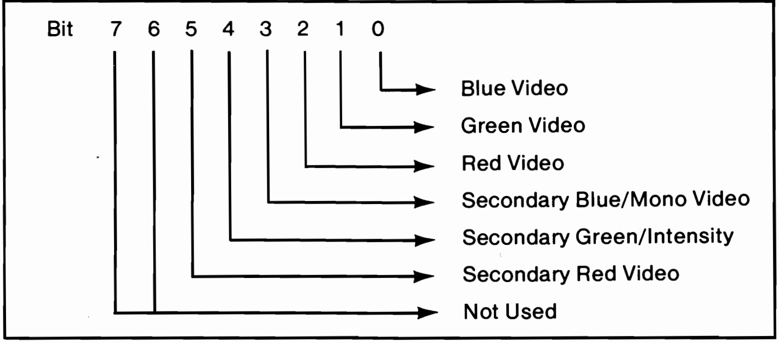
Attribute Controller Registers

Name	Port	Index
Address Register	3C0	-
Palette Registers	3C0	00-0F
Mode Control Register	3C0	10
Overscan Color Register	3C0	11
Color Plane Enable Register	3C0	12
Horizontal Pel Panning Register	3C0	13

Attribute Address Register

This is a write-only register. The processor output port is hex 3C0.

Palette Registers Hex 00 through Hex 0F Format

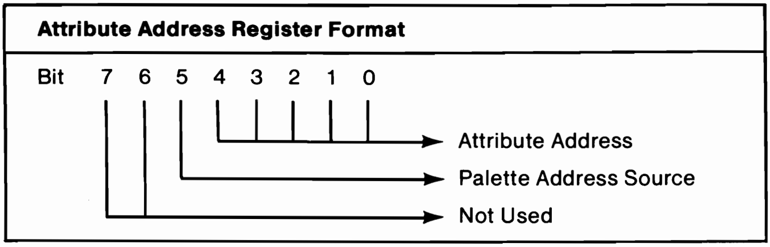


Bit 0-Bit 4 Attribute Address Bits—The Address Register is a pointer register located at hex 3C0. This register is loaded with a binary value that points to the attribute data register where data is to be written. The Attribute Controller does not have an address bit input to control selection of the address and data registers. An internal address flip-flop controls selection of either the address or data registers. To initialize the flip-flop, an IOR instruction is issued to the Attribute Controller at address 3BA or 3DA. This clears the flip-flop, and selects the Address Register. After the Address Register has been loaded, the next OUT instruction loads the data register. The flip-flop toggles each time an OUT is issued to the Attribute Controller.

Bit 5 Palette Address Source—When loading the color palette registers, bit 5 must be cleared to 0. To enable the memory data to access the color palette, bit 5 must be set to 1.

Palette Register Hex 00 through Hex 0F

This is a write-only register. The processor output port is hex 3C0.

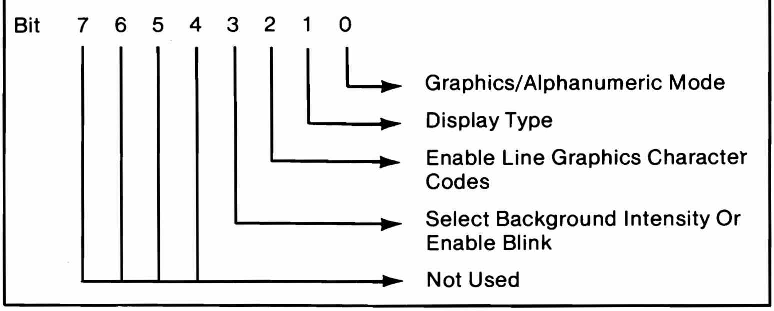

Attribute Address Register Format

Bit 0–Bit 5 Palette—These 6-bit registers allow a dynamic mapping between the text attribute or graphic color input value and the display color in the CRT screen. A logical 1 selects the appropriate color. A logical 0 de-selects. The color palette register should be modified only during the vertical retrace interval to avoid glitches in the displayed image. Note that some color monitors do not have an intensity input and only a maximum of eight colors are be available. Monitors with four color inputs display sixteen colors, and monitors with six color inputs display 64 colors.

Mode Control Register

This is a write-only register pointed to by the value in the Attribute address register. This value must be hex 10 before writing can take place. The processor output port address for this register is hex 3C0.

Mode Control Register Format


Bit 7 6 5 4 3 2 1 0
Graphics/Alphanumeric Mode
Display Type
Enable Line Graphics Character Codes
Select Background Intensity Or Enable Blink
Not Used

Bit 0 Graphics/Alphanumeric Mode—A logical 0 selects selects alphanumeric mode. A logical 1 selects graphics mode.

Bit 1 Monochrome Display/Color Display—A logical 0 selects color display attributes. A logical 1 selects IBM Monochrome Display attributes.

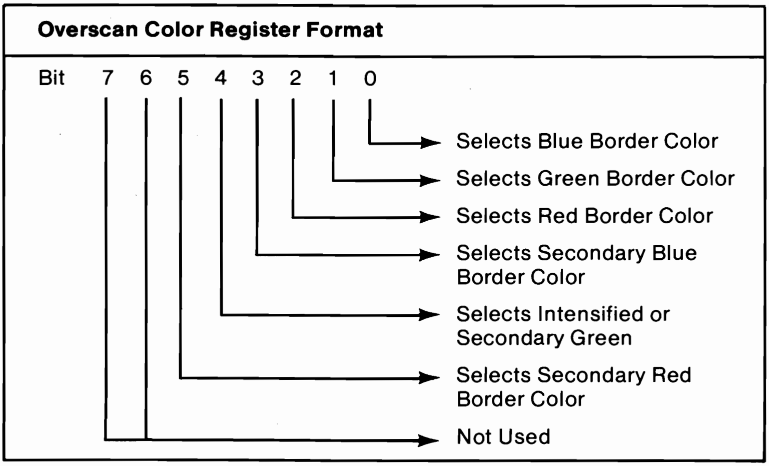
Bit 2 Enable Line Graphics Character Codes—When this bit is set to 0, the ninth dot will be the same as the background. A logical 1 enables the special line graphics character codes for the IBM Monochrome Display adapter. This bit when enabled forces the ninth dot of a line graphic character to be identical to the eighth dot of the character. The line graphics character codes for the Monochrome Display Adapter are Hex C0 through Hex DF.

For character fonts that do not utilize the line graphics character codes in the range of Hex C0 through Hex DF, bit 2 of this register should be a logical 0. Otherwise unwanted video information will be displayed on the CRT screen.

Bit 3 Enable Blink/Select Background Intensity—A logical 0 selects the background intensity of the attribute input. This mode was available on the Monochrome and Color Graphics adapters. A logical 1 enables the blink attribute in alphanumeric modes. This bit must also be set to 1 for blinking graphics modes.

Overscan Color Register

This is a write-only register pointed to by the value in the Attribute address register. This value must be hex 11 before writing can take place. The processor output port address for this register is hex 3C0.

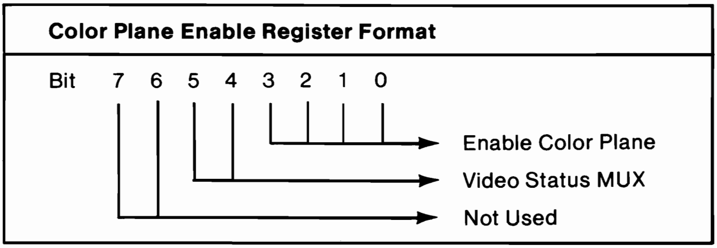

Overscan Color Register Format

Bit 7 6 5 4 3 2 1 0
Selects Blue Border Color
Selects Green Border Color
Selects Red Border Color
Selects Secondary Blue Border Color
Selects Intensified or Secondary Green
Selects Secondary Red Border Color
Not Used

Bit 0-Bit 5 Overscan Color—This 6-bit register determines the overscan (border) color displayed on the CRT screen. For monochrome display this register should be set to a value of 0. A logical 1 selects the appropriate color.

Color Plane Enable Register

This is a write-only register pointed to by the value in the Attribute address register. This value must be hex 12 before writing can take place. The processor output port address for this register is 3C0.


Color Plane Enable Register Format

Bit 0-Bit 3 Enable Color Plane—Writing a logical 1 in any of bits 0 through 3 enables the respective display memory color plane.

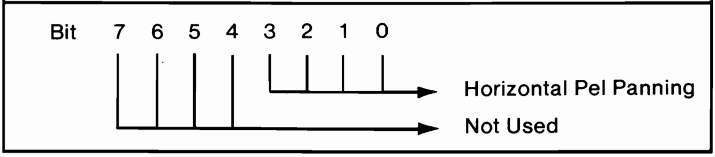
Bit 4-Bit 5 Video Status MUX—Selects two of the six color outputs to be available on the status port. The following table illustrates the combinations available and the color output wiring.

COLOR PLANE ENABLE REGISTER		INPUT STATUS REGISTER ONE	
Bit 5	**Bit 4**	**Bit 5**	**Bit 4**
0	0	Red	Blue
0	1	Secondary Blue	Green
1	0	Secondary Red	Secondary Green
1	1	Not Used	Not Used

Horizontal Pel Panning Register

This is a write-only register pointed to by the value in the Attribute address register. This value must be hex 12 before writing can take place. The processor output port address for this register is hex 3C0.

Horizontal Pel Panning Register Format



Bit 0-Bit 3 Horizontal Pel Panning—This 4 bit register selects the number of picture elements (pels) to shift the video data horizontally to the left. Pel panning is available in both A/N and APA modes. In Monochrome A/N mode, the image can be shifted a maximum of 9 pels. In all other A/N and APA modes, the image can be shifted a maximum of 8 pels. The sequence for shifting the image is given below:

9 pels/character : 8, 0, 1, 2, 3, 4, 5, 6, 7 (Monochrome A/N mode only)

8 pels/character : 0, 1, 2, 3, 4, 5, 6, 7 (All other Modes)

Programming Considerations

Programming the Registers

Each of the LSI devices has an address register and a number of data registers. The address register serves as a pointer to the other registers on the LSI device. It is a write-only register that is loaded by the processor by executing an 'OUT' instruction to its I/O address with the index of the selected data register.

The data registers on each LSI device are accessed through a common I/O address. They are distinguished by the pointer (index) in the address register. To write to a data register, the address register is loaded with the index of the appropriate data register, then the selected data register is loaded by executing an 'OUT' instruction to the common I/O address.

The external registers that are not part of an LSI device and the Graphics I and II registers are not accessed through an address register; they are written to directly.

The following tables define the values that are loaded into the registers by BIOS to support the different modes of operation supported by this adapter.

Register			Mode of Operation																	
Name	Port	Index	0	1	2	3	4	5	6	7	D	E	F	10	F⁑	10⁑	0*	1*	2*	3*
Miscellaneous	3C2	–	23	23	23	23	23	23	23	A6	23	23	A2	A7	A2	A7	A7	A7	A7	A7
Feature Cntrl	3?A	–	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Input Stat 0	3C2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Input Stat 1	3?2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
? = B in monochrome modes ? = D in color modes																				
*Values for these modes when the IBM Enhanced Color Display is attached																				
⁑Values for these modes when greater than 64K Graphics Memory is installed																				

External Registers

Register			Mode of Operation																	
Name	Port	Index	0	1	2	3	4	5	6	7	D	E	F	10	F⁑	10⁑	0*	1*	2*	3*
Seq Address	3C4	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Reset	3C5	00	03	03	03	03	03	03	03	03	03	03	03	03	03	03	03	03	03	03
Clock Mode	3C5	01	0B	0B	01	01	0B	0B	01	00	0B	01	05	05	01	01	0B	0B	01	01
Map Mask	3C5	02	03	03	03	03	03	03	01	03	0F	0F	0F	0F	0F	0F	03	03	03	03
Char Gen Sel	3C5	03	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Memory Mode	3C5	04	03	03	03	03	02	02	06	03	06	06	00	00	06	06	03	03	03	03
*Values for these modes when the IBM Enhanced Color Display is attached																				
⁑Values for these modes when greater than 64K Graphics Memory is installed																				

Sequencer Registers

Register			Mode of Operation																	
Name	Port	Index	0	1	2	3	4	5	6	7	D	E	F	10	F⁑	10⁑	0*	1*	2*	3*
Address Reg	3?4	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Horiz Total	3?5	00	37	37	70	70	37	37	70	60	37	70	60	5B	60	5B	2D	2D	5B	5B
Hrz Disp End	3?5	01	27	27	4F	4F	27	27	4F	4F	27	4F	4F	4F	4F	4F	27	27	4F	4F
Strt Hrz Blk	3?5	02	2D	2D	5C	5C	2D	2D	59	56	2D	56	56	53	56	53	2B	2B	53	53
End Hrz Blk	3?5	03	37	37	2F	2F	37	37	2D	3A	37	2D	1A	17	3A	37	2D	2D	37	37
Strt Hrz Retr	3?5	04	31	31	5F	5F	30	30	5E	51	30	5E	50	50	50	52	28	28	51	51
End Hrz Retr	3?5	05	15	15	07	07	14	14	06	60	14	06	E0	BA	60	00	6D	6D	5B	5B
Vert Total	3?5	06	04	04	04	04	04	04	04	70	04	04	70	6C	70	6C	6C	6C	6C	6C
Overflow	3?5	07	11	11	11	11	11	11	11	1F	11	11	1F	1F	1F	1F	1F	1F	1F	1F
Preset Row SC	3?5	08	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Max Scan Line	3?5	09	07	07	07	07	01	01	01	0D	00	00	00	00	00	00	0D	0D	0D	0D
Cursor Start	3?5	0A	06	06	06	06	00	00	00	0B	00	00	00	00	00	00	0B	0B	0B	0B
Cursor End	3?5	0B	07	07	07	07	00	00	00	0C	00	00	00	00	00	00	0C	0C	0C	0C
Strt Addr Hi	3?5	0C	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Strt Addr Lo	3?5	0D	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

? = B in monochrome modes   ? = D in color modes

*Values for these modes when the IBM Enhanced Color Display is attached

⁑Values for these modes when greater than 64K Graphics Memory is installed

CRT Controller Registers (1 of 2)

Register			Mode of Operation																	
Name	Port	Index	0	1	2	3	4	5	6	7	D	E	F	10	F⁑	10⁑	0*	1*	2*	3*
Cursor LC Hi	3?5	0E	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Cursor LC Low	3?5	0F	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Vrt Retr Strt	3?5	10	E1	E1	E1	E1	E1	E1	E0	5E	E1	E0	5E	5E	5E	5E	5E	5E	5E	5E
Light Pen Hi	3?5	10	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Vert Retr End	3?5	11	24	24	24	24	24	24	23	2E	24	23	2E	2B	2E	2B	2B	2B	2B	2B
Light Pen Low	3?5	11	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Vrt Disp End	3?5	12	C7	C7	C7	C7	C7	C7	C7	5D	C7	C7	5D	5D	5D	5D	5D	5D	5D	5D
Offset	3?5	13	14	14	28	28	14	14	28	28	14	28	14	14	28	28	14	14	28	28
Underline Loc	3?5	14	08	08	08	08	00	00	00	0D	00	00	0D	0F	0D	0F	0F	0F	0F	0F
Strt Vert Blk	3?5	15	E0	E0	E0	E0	E0	E0	DF	5E	E0	DF	5E	5F	5E	5F	5E	5E	5E	5E
End Vert Blk	3?5	16	F0	F0	F0	F0	F0	F0	EF	6E	F0	EF	6E	0A	6E	0A	0A	0A	0A	0A
Mode Control	3?5	17	A3	A3	A3	A3	A2	A2	C2	A3	E3	E3	8B	8B	E3	E3	A3	A3	A3	A3
Line Compare	3?5	18	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

? = B in monochrome modes ? = D in color modes

*Values for these modes when the IBM Enhanced Color Display is attached

⁑Values for these modes when greater than 64K Graphics Memory is installed

CRT Controller Registers (2 of 2)

Register			Mode of Operation																	
Name	Port	Index	0	1	2	3	4	5	6	7	D	E	F	10	F⁑	10⁑	0*	1*	2*	3*
Grphx I Pos	3CC	–	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Grphx II Pos	3CA	–	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01
Grphx I II AD	3CE	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Set Reset	3CF	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Enable S/R	3CF	01	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Color Compare	3CF	02	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Data Rotate	3CF	03	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Read Map Sel	3CF	04	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Mode Register	3CF	05	10	10	10	10	30	30	00	10	00	00	10	10	00	00	10	10	10	10
Miscellaneous	3CF	06	0E	0E	0E	0E	0F	0F	0D	0A	05	05	07	07	05	05	0E	0E	0E	0E
Color No Care	3CF	07	00	00	00	00	00	00	00	00	0F	0F	0F	0F	0F	0F	00	00	00	00
Bit Mask	3CF	08	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

*Values for these modes when the IBM Enhanced Color Display is attached

⁑Values for these modes when greater than 64K Graphics Memory is installed

Graphics SI Registers

Register			Mode of Operation																	
Name	Port	Index	0	1	2	3	4	5	6	7	D	E	F	10	F⁑	10⁑	0*	1*	2*	3*
Address	3?A	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Palette	3C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Palette	3C0	01	01	01	01	01	13	13	17	08	01	01	08	01	08	01	01	01	01	01
Palette	3C0	02	02	02	02	02	15	15	17	08	02	02	00	00	00	02	02	02	02	02
Palette	3C0	03	03	03	03	03	17	17	17	08	03	03	00	00	00	03	03	03	03	03
Palette	3C0	04	04	04	04	04	02	02	17	08	04	04	18	04	18	04	04	04	04	04
Palette	3C0	05	05	05	05	05	04	04	17	08	05	05	18	07	18	05	05	05	05	05
Palette	3C0	06	06	06	06	06	06	06	17	08	06	06	00	00	00	06	14	14	14	14
Palette	3C0	07	07	07	07	07	07	07	17	08	07	07	00	00	00	07	07	07	07	07
Palette	3C0	08	10	10	10	10	10	10	17	10	10	10	00	00	00	38	38	38	38	38
Palette	3C0	09	11	11	11	11	11	11	17	18	11	11	08	01	08	39	39	39	39	39
Palette	3C0	0A	12	12	12	12	12	12	17	18	12	12	00	00	00	3A	3A	3A	3A	3A
Palette	3C0	0B	13	13	13	13	13	13	17	18	13	13	00	00	00	3B	3B	3B	3B	3B

? = B in monochrome modes   ? = D in color modes

*Values for these modes when the IBM Enhanced Color Display is attached

⁑Values for these modes when greater than 64K Graphics Memory is installed

Attribute Registers (1 of 2)

Register			Mode of Operation																	
Name	Port	Index	0	1	2	3	4	5	6	7	D	E	F	10	F⁑	10⁑	0*	1*	2*	3*
Palette	3C0	0C	14	14	14	14	14	14	17	18	14	14	00	04	00	3C	3C	3C	3C	3C
Palette	3C0	0D	15	15	15	15	15	15	17	18	15	15	18	07	18	3D	3D	3D	3D	3D
Palette	3C0	0E	16	16	16	16	16	16	17	18	16	16	00	00	00	3E	3E	3E	3E	3E
Palette	3C0	0F	17	17	17	17	17	17	18	17	17	00	00	00	3F	3F	3F	3F	3F	3F
Mode Control	3C0	10	08	08	08	08	01	01	01	0E	01	01	0B	0B	0B	01	08	08	08	08
Overscan	3C0	11	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Color Plane	3C0	12	0F	0F	0F	0F	03	03	01	0F	0F	0F	05	05	05	0F	0F	0F	0F	0F
Hrz Panning	3C0	13	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

*Values for these modes when the IBM Enhanced Color Display is attached

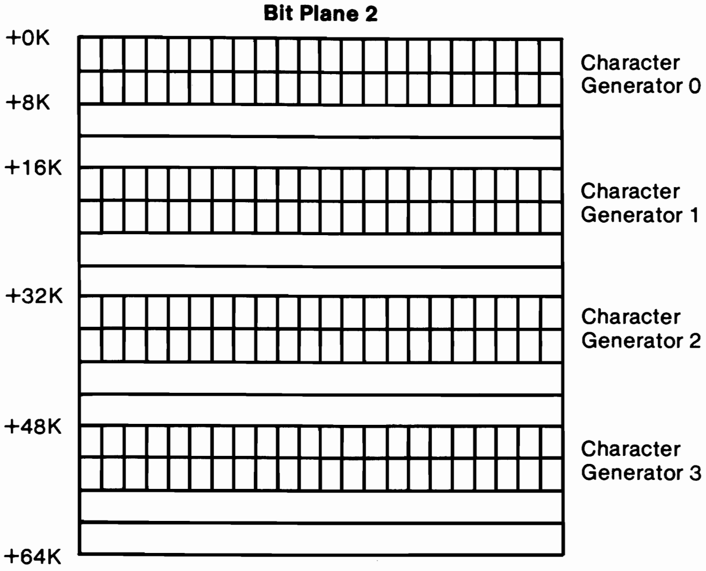
⁑Values for these modes when greater than 64K Graphics Memory is installed

Attribute Registers (2 of 2)

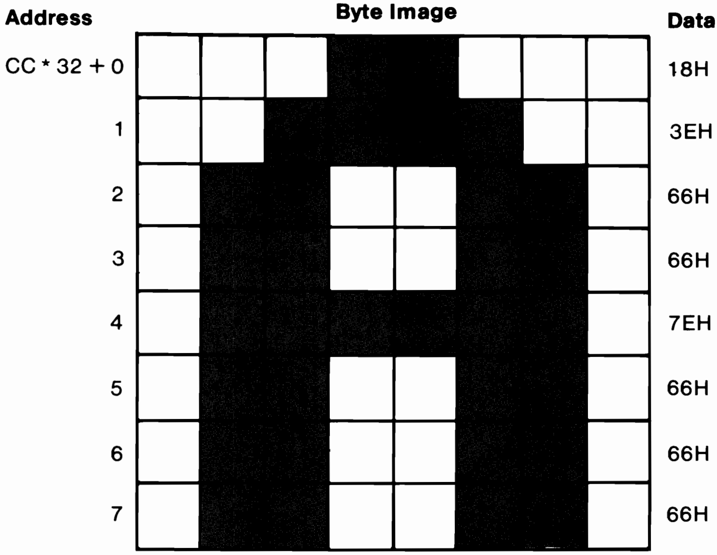
RAM Loadable Character Generator

The character generator on the adapter is RAM loadable and can support characters up to 32 scan lines high. Two character generators are stored within the BIOS and one is automatically loaded into the RAM by the BIOS when an alphanumeric mode is selected. The Character Map Select Register can be programmed to define the function of bit 3 of the attribute byte to be a character generator switch. This allows the user to select between any two character sets residing in bit plane 2. This effectively gives the user access to 512 characters instead of 256. character tables may be loaded off line.The adapter must have 128K bytes of storage to support this function. Up to four tables can be loaded can be loaded with 256K of graphics memory installed.

The structure of the character tables is described in the following figure. The character generator is in bit plane 2 and must be protected using the map mask function.



The following figure illustrates the structure of each character pattern. If the CRT controller is programmed to generate *n* row scans, then *n* bytes must be filled in for each character in the character generator. The example assumes eight row scans per character.



CC = Value of the character code. For example, 41H in the case of an ASCII "A".

Creating a 512 Character Set

This section describes how to create a 512 character set on the IBM Color Display. Note that only 256 characters can be printed on the printer. This is a special application which the Enhanced Graphics Adapter will support. The 9 by 14 characters will be displayed when attribute bit 3 is a logical 0, and the IBM Color/Graphics Monitor Adapter 8 by 8 characters will be displayed when the attribute bit 3 is a logical 1. This example is for demonstrative purposes only. The assembly language routine for creating 512 characters is given below. Debug 2.0 was used for this example. The starting assembly address is 100 and the character string is stored in location 200. This function requires 128K or more of graphics memory.

```
a100
mov ax,1102         ;load 8x8 character font in character
mov bl,02           ;generator number 2
int 10

mov ax,1103         ;select 512 character operation
mov bl,08           ;if attribute bit 3=1 use 8x8 font
int 10              ;if attribute bit 3=0 use 9x14 font

mov ax,1000         ;set color plane enable to 7H to disable
mov bx,0712         ;attribute bit 3 in the color palette
int 10              ;lookup table

mov ax,1301
mov bx,000F         ;write char. string with attribute bit 3=1
mov cx,003A         ;cx = character string length
mov dx,1600         ;write character on line 22 of display
mov bp,0200         ;pointer to character string location
push cs
pop es
int 10

mov ax ,1301
mov bx,0007         ;write char. string with attribute bit 3=0
mov cx,003A         ;cx = character string length
mov dx,1700         ;write character on line 23 of display
mov bp,0200         ;pointer to character string location
push cs
pop es
int 10
int 3

a200 db             "This character string is used to show 512
                    characters"
```

Creating an 80 by 43 Alphanumeric Mode

The following examples show how to create 80 column by 43 row, both alphanumeric and graphics, images on the IBM Monochrome Display. The BIOS Interface supports an 80 column by *n* row display by using the character generator load routine call. The print screen routine must be revectored to handle the additional character rows on the screen. The assembly language required for both an alphanumeric and a graphics screen is shown below.

```
mov al,7          ;Monochrome alphanumeric mode
int 10            ;video interrupt call
mov ax,1112       ;character generator BIOS routine
mov bl,0          ;load 8 by 8 double dot character font
int 10            ;video interrupt call
mov ax,1200       ;alternate screen routine
move bl,20        ;select alternate print screen routine
int 10            ;video interrupt call
int 3

mov ax,f          ;Monochrome graphic mode
int 10            ;video interrupt call
mov ax,1123       ;character generator BIOS routine
mov bl,0          ;load 8 by 8 double dot character font
mov dl,2B         ;43 character rows
int 10            ;video interrupt call
mov ax,1200       ;alternate screen routine
mov bl,20         ;alternate print screen routine
int 10            ;video interrupt call
int 3
```

Vertical Interrupt Feature

The Enhanced Graphics Adapter can be programmed to create an interrupt each time the vertical display refresh time has ended. An interrupt handler routine must be written by the application to take advantage of this feature. The CRT Vertical interrupt is on IRQ2. The CPU can poll the Enhanced Graphics Adapter Input Status Register 0 (bit 7) to determine whether the CRTC caused the interrupt to occur.

The Vertical Retrace End Register (11H) in the CRT controller contains two bits which are used to control the interrupt circuitry. The remaining bits must be output as per the value in the mode table.

Bit 5 Enable Vertical Interrupt—A logical 0 will enable vertical interrupt.

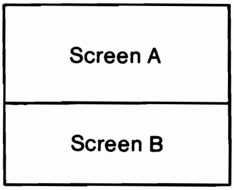
Bit 4 Clear Vertical Interrupt—A logical 0 will clear a vertical interrupt.

The sequence of events which occur in an interrupt handler are outlined below.

1. Clear IRQ latch and enable driver
2. Enable IRQ latch
3. Wait for vertical interrupt
4. Poll Interrupt Status Register 0 to determine if CRTC has caused the interrupt
5. If CRTC interrupt, then clear IRQ latch; if not, then branch to next interrupt handler.
6. Enable IRQ latch
7. Update Enhanced Graphics Adapter during vertical blanking interval
8. Wait for next vertical interrupt

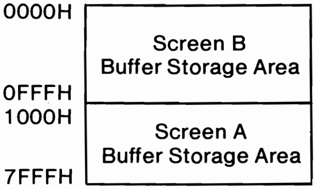
Creating a Split Screen

The Enhanced Graphics Adapter hardware supports an alphanumeric mode dual screen display. The top portion of the screen is designated as screen A, and the bottom portion of the screen is designated as screen B as per the following figure.



Dual Screen Definition

The following figure shows the screen mapping for a system containing a 32K byte alphanumeric storage buffer. Note that the Enhanced Graphics Adapter has a 32K byte storage buffer in alphanumeric mode. Information displayed on screen A is defined by the start address high and low registers (0CH and 0DH) of the CRTC. Information displayed on screen B always begins at address 0000H.



Screen Mapping Within the Display Buffer Address Space

The Line Compare Register (18H) of the CRT Controller is utilized to perform the split screen function. The CRTC has an internal horizontal scan counter, and logic which compares the horizontal scan counter value to the Line Compare Register value and clears the memory address generator when a compare occurs. The linear address generator then sequentially addresses the display buffer starting at location zero, and each subsequent row address is is determined by the 16 bit addition of the start of line latch and the offset register.

Screen B can be smoothly scrolled onto the CRT screen by updating the Line compare in synchronization with the vertical retrace signal. The information on screen B is immune from scrolling operations which utilize the Start Address High and Low registers to scroll through the Screen A address map.

Compatibility Issues

The CRT Controller on the IBM Enhanced Graphics Adapter is a custom design, and is different than the 6845 controller used on the IBM Monochrome Monitor Adapter and the IBM Color/Graphics Monitor Adapter. It should be noted that several CRTC register addresses differ between the adapters. The following figure illustrates the registers which do not map directly across the two controllers.

Register	6485 Function	EGA CRTC Function
02H	Start Horiz. Retrace	Start Horiz. Blanking
03H	End Horiz. Retrace	End Horiz. Blanking
04H	Vertical Total	Start Horiz. Retrace
05H	Vertical Total Adjust	End Horiz. Retrace
06H	Vertical Displayed	Vertical Total
07H	Vertical Sync Position	Overflow
08H	Interlace Mode and Skew	Preset Row Scan

Existing applications which utilize the BIOS interface will generally be compatible with the Enhanced Graphics Adapter.

Horizontal screen centering was required on the IBM Color/Graphics Monitor Adapter in order to center the screen when generating composite video. This was done through the Horizontal Sync Position Register. Since the Enhanced Graphics Adapter does not support a composite video monitor, programs which do screen centering may cause loss of the screen image if centering is attempted.

The Enhanced Graphics Adapter offers a wider variety of displayable monochrome character attributes than the IBM Monochrome Display Adapter. Some attribute values may display differently between the two Adapters. The values listed in the table below, in any combinations with the blink and intensity attributes, will display identically.

Background R G B	Foreground R G B	Function
0 0 0	0 0 0	Non-Display
0 0 0	0 0 1	Underline
0 0 0	1 1 1	White Character/Black Background
1 1 1	0 0 0	Reverse Video

Software which explicitly addresses 3D8 (Mode Select Register) or 3D9 (Color Select Register) on the Color Graphics Monitor Adapter may produce different results on the Enhanced Graphics Adapter. For example, blinking which is disabled by writing to 3D8 on the Color Graphics Adapter will not be disabled on the Enhanced Graphics Adapter.

Interface

Feature Connector

The following is a description of the Enhanced Graphics Adapter feature connector. Note that signals coming from the Enhanced Graphics Adapter are labeled “inputs” and the signals coming to the Enhanced Graphics Adapter through the feature connector are labeled “outputs”.

Signal	Description
J2	This pin is connected to auxiliary jack 2 on the rear panel of the adapter.
R’OUT	Secondary red output
ATRS/L	Attribute shift load. This signal controls the serialization of the video information. The shift register parallel loads at the dot clock leading edge when this signal is low.
G OUT	Primary green output
R’	Secondary red input
R	Primary red input
FC1	This signal is input from bit 1 (Feature Control Bit 1) of the Feature Control Register.
FC0	This signal is input from bit 0 (Feature Control Bit 0) of the Feature control Register.
FEAT 0	This signal is output to bit 5 (Feature Code 0) of Input Status Register 0.
B’/V	Secondary blue input/Monochrome video
VIN	Vertical retrace input

Internal This signal is output to bit 4 (Disable Internal Video Drivers) of the Miscellaneous Output Register.

V OUT Vertical retrace output

J1 This pin is connected to auxiliary jack 1 on the rear panel of the adapter.

G'OUT Secondary green output

B'OUT Secondary blue output

B OUT Blue output

G Green input

B Blue input

R OUT Red output

BLANK This is a composite horizontal and vertical blanking signal from the CRTC.

FEAT 1 This signal is output to bit 6 (Feature Code 1) of Input Status Register 0.

G'/I Secondary green/Intensity input

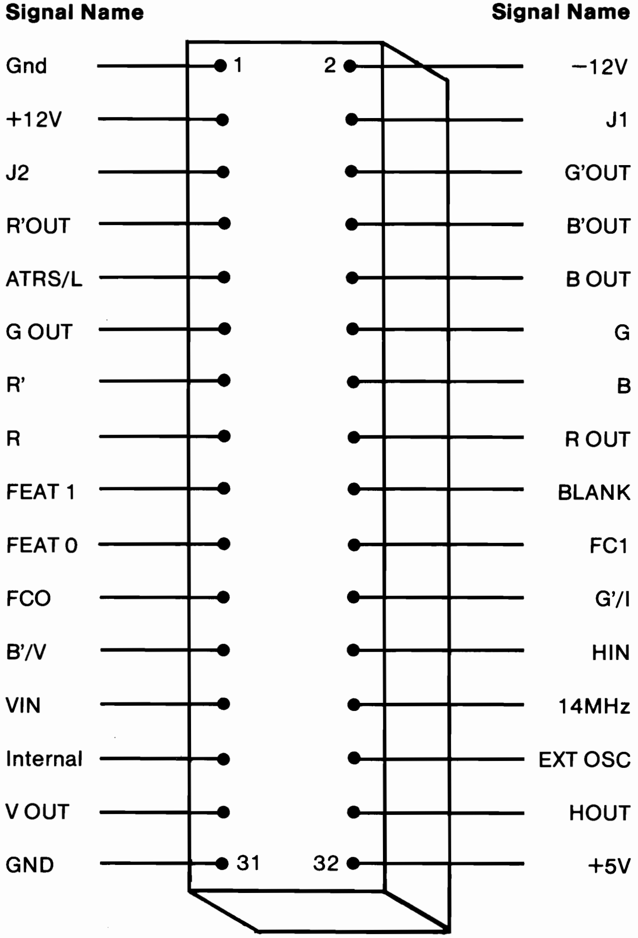
HIN Horizontal retrace input from the CRTC

14MHZ 14 MHz signal from the system board

EXT OSC External dot clock output

HOUT Horizontal retrace output

The following figure shows the layout and pin numbering of the feature connector.

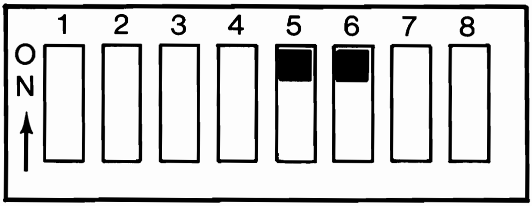


Feature Connector Diagram

Specifications

System Board Switches

The following figure shows the proper system board DIP switch settings for the IBM Enhanced Graphics Adapter when used with the Personal Computer and the Personal Computer XT. The switch block locations are illustrated in the Technical Reference Manual "System Board Component Diagram". The Personal Computer has two DIP switch blocks; the switch settings shown pertain to DIP Switch Block 1. The Personal Computer XT has one DIP switch block.

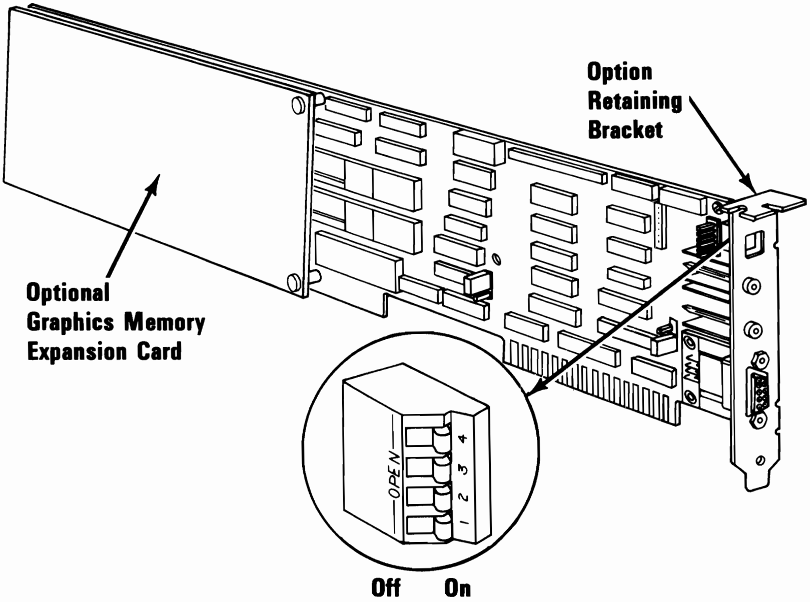


Switch Block (1)

Note: The DIP switches must be set as shown whenever the IBM Enhanced Graphics Adapter is installed, regardless of display type. This is true even when a second display adapter is installed in the system.

Configuration Switches

The following diagram shows the location and orientation of the configuration switches on the Enhanced Graphics Adapter.



Configuration Switch Settings

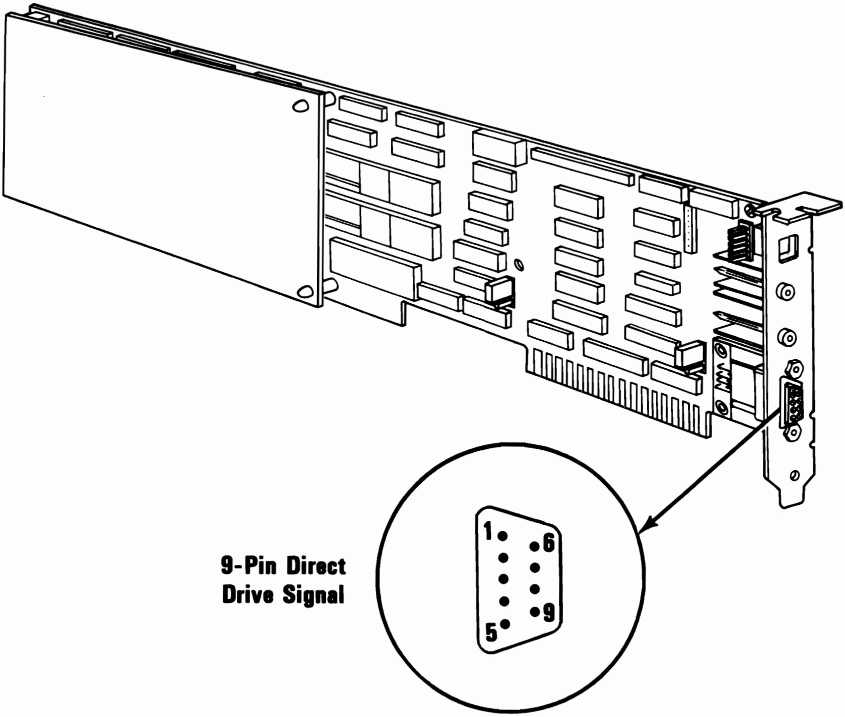
The configuration switches on the Enhanced Graphics Adapter determine the type of display support the adapter provides, as follows:

Switch Settings for Enhanced Graphics Adapter as Primary Display Adapter						
				Configuration		
SW1	SW2	SW3	SW4	Enhanced Adapter	Monochrome Adapter	Color/Graphics Adapter
On	Off	Off	On	Color Display 40x25	Secondary	–
Off	Off	Off	On	Color Display 80x25	Secondary	–
On	On	On	Off	Enhanced Display Emulation Mode	Secondary	–
Off	On	On	Off	Enhanced Display Hi Res Mode	Secondary	–
On	Off	On	Off	Monochrome	–	Secondary 40x25
Off	Off	On	Off	Monochrome	–	Secondary 80x25

Switch Settings for Enhanced Graphics Adapter as Secondary Display Adapter

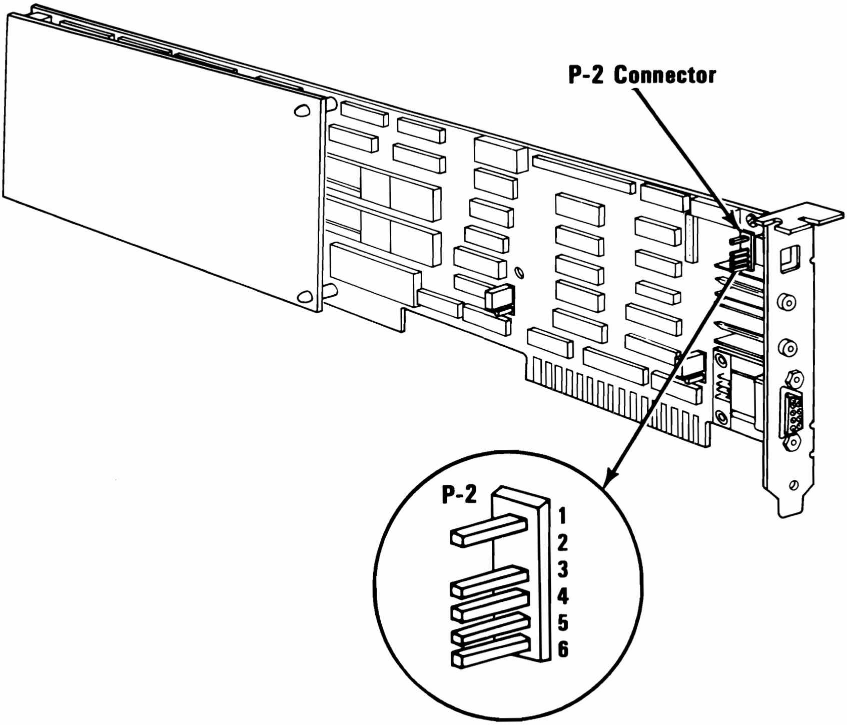
				Configuration		
SW1	SW2	SW3	SW4	Enhanced Adapter	Monochrome Adapter	Color/Graphics Adapter
On	On	On	On	Color Display 40x25	Primary	–
Off	On	On	On	Color Display 80x25	Primary	–
On	Off	On	On	Enhanced Display Emulation Mode	Primary	–
Off	Off	On	On	Enhanced Display Hi Res Mode	Primary	–
On	On	Off	On	Monochrome	–	Primary 40x25
Off	On	Off	On	Monochrome	–	Primary 80x25

Direct Drive Connector



	Signal Name - Description	Pin	
Direct Drive Display	Ground	1	**Enhanced Graphics Adapter**
	Secondary Red	2	
	Primary Red	3	
	Primary Green	4	
	Primary Blue	5	
	Secondary Green/Intensity	6	
	Secondary Blue/Mono Video	7	
	Horizontal Retrace	8	
	Vertical Retrace	9	

Light Pen Interface



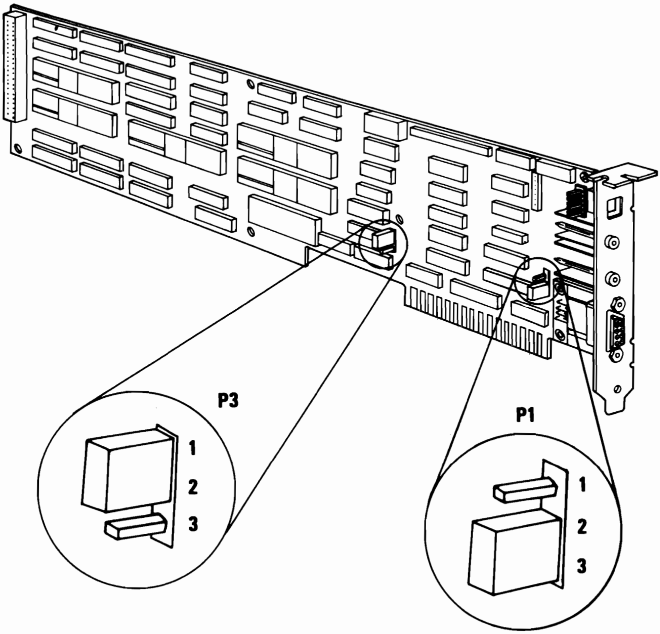
	P-2 Connector	Pin	
Light Pen Attachment	+Light Pen Input	1	Enhanced Graphics Adapter
	Not used	2	
	+Light Pen Switch	3	
	Ground	4	
	+5 Volts	5	
	12 Volts	6	

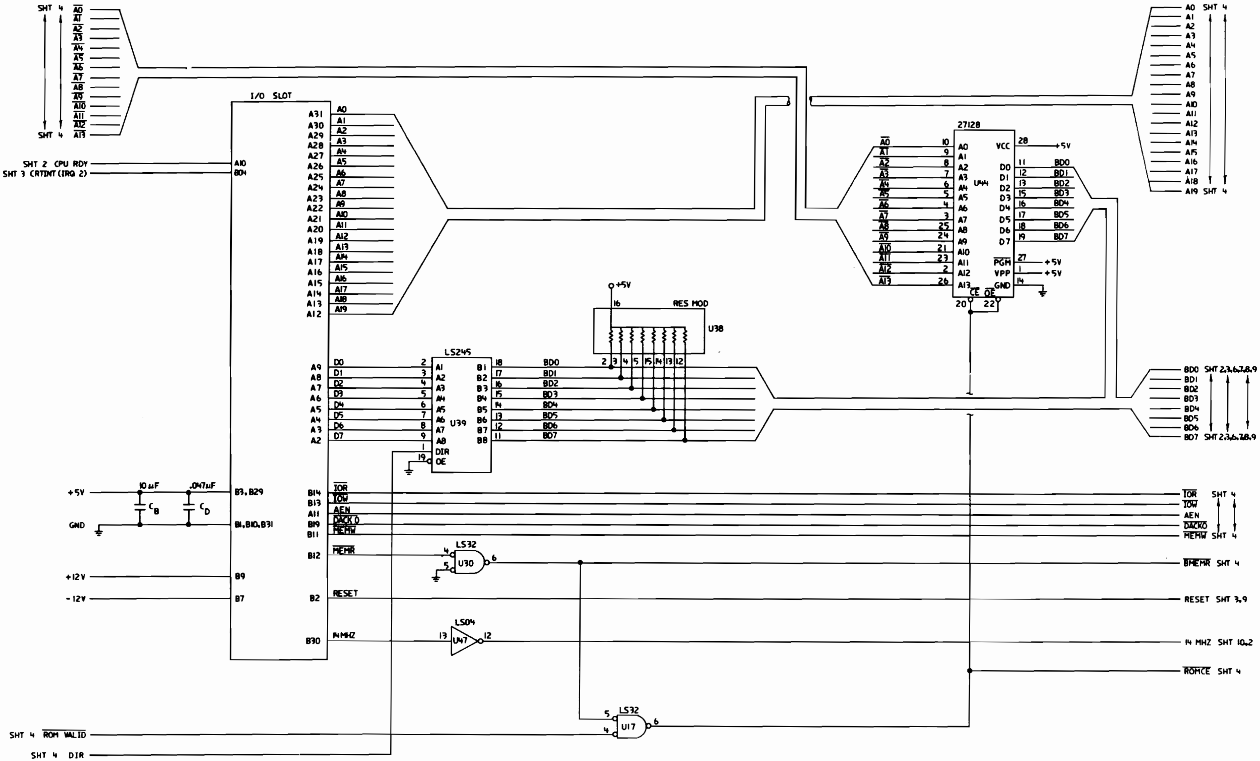
Jumper Descriptions

Located on the adapter are two jumpers designated P1 and P3. Jumper P1 changes the function of pin 2 on the direct drive interface. When placed on pins 2 and 3, jumper P1 selects ground as the function of direct drive interface, pin 2. This selection is for displays that support five color outputs, such as the IBM Color Display. When P1 is placed on pins 1 and 2, red prime output is placed on pin 2 of the direct drive interface connector. This supports the IBM Enhanced Color Display, which utilizes six color outputs on the direct drive interface.

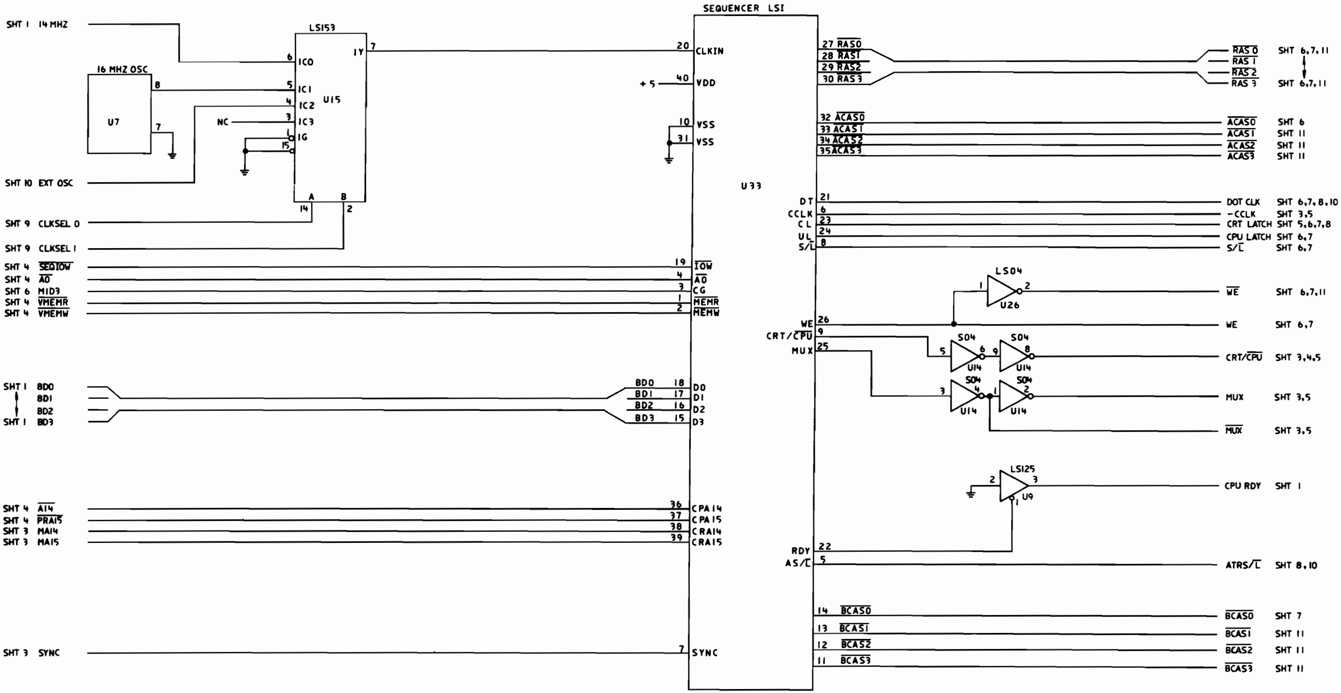
Jumper P3 changes the I/O address port of the Enhanced Graphics Adapter within the system. In its normal position, (pins 1 and 2), all Enhanced Graphics Adapter addresses are in the range 3XX. Moving jumper P3 to pins 2 and 3 changes the addresses to 2XX. Operation of the adapter in the 2XX mode is not supported in BIOS.

The following figure shows the location of the jumpers and numbering of the connectors.

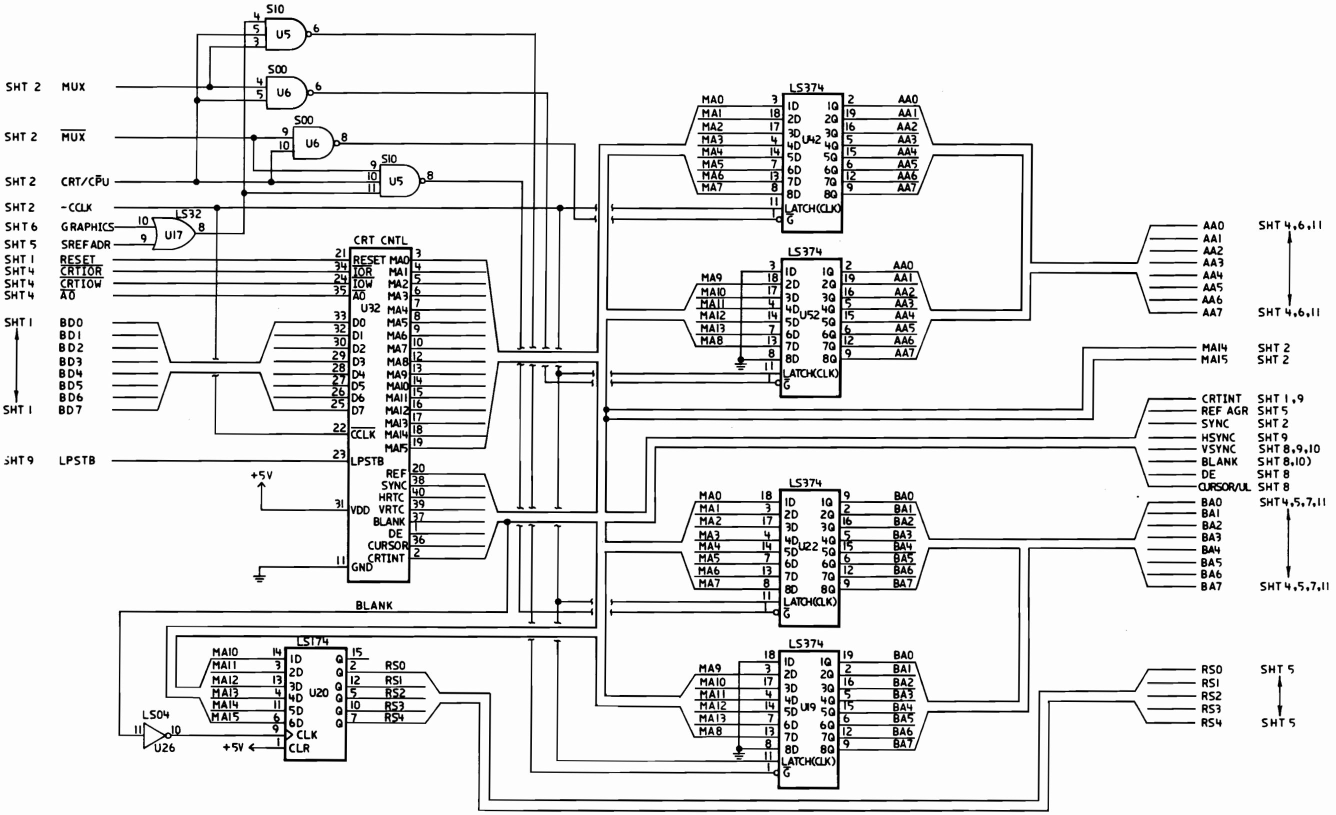




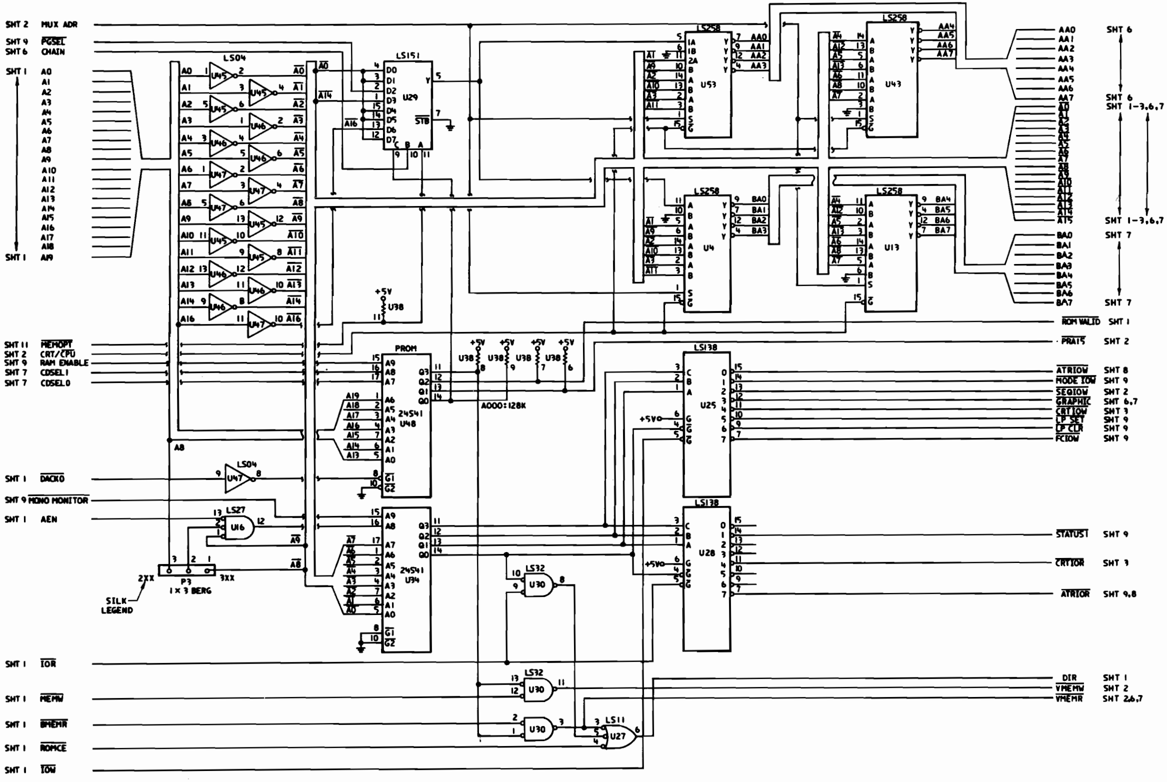
Enhanced Graphics Adapter Sheet 1 of 11



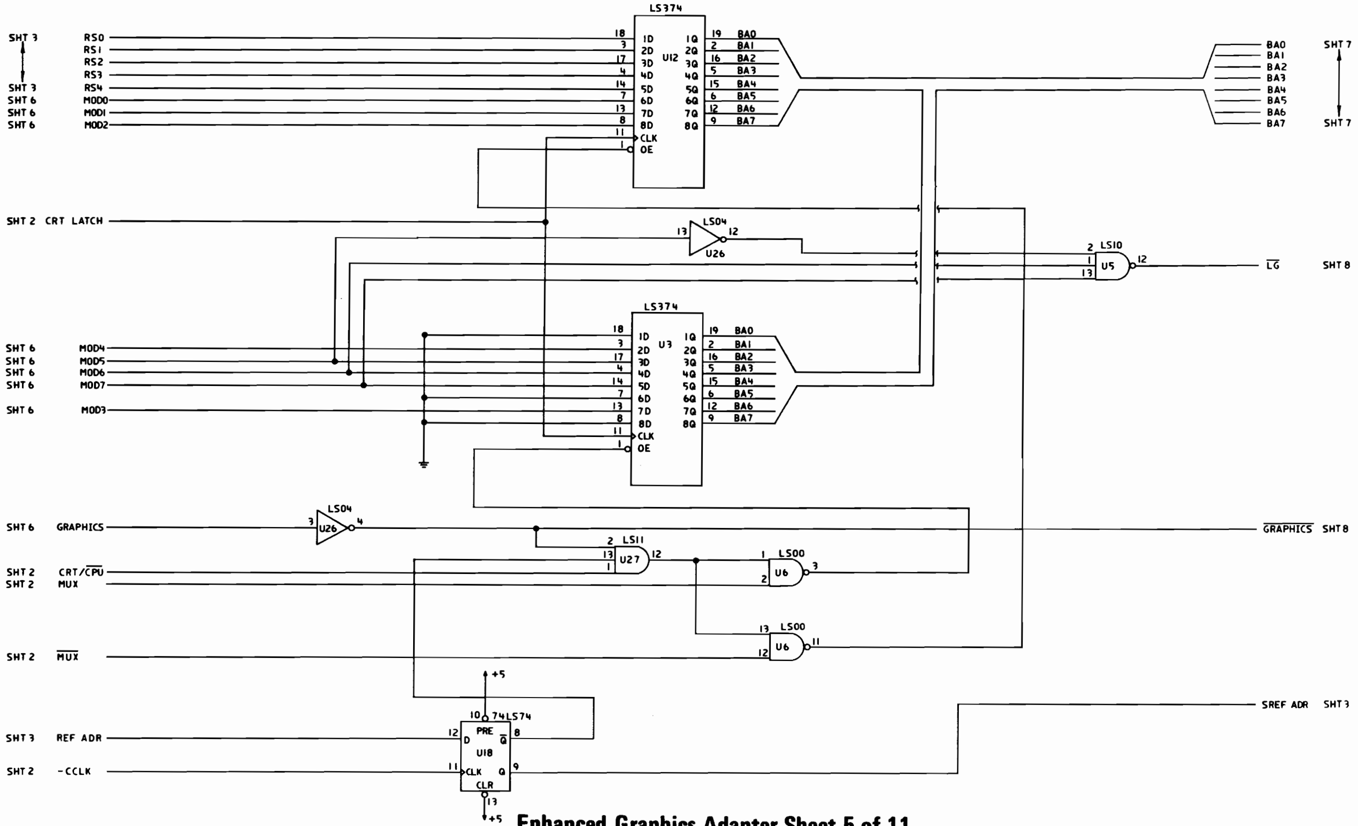
Enhanced Graphics Adapter Sheet 2 of 11



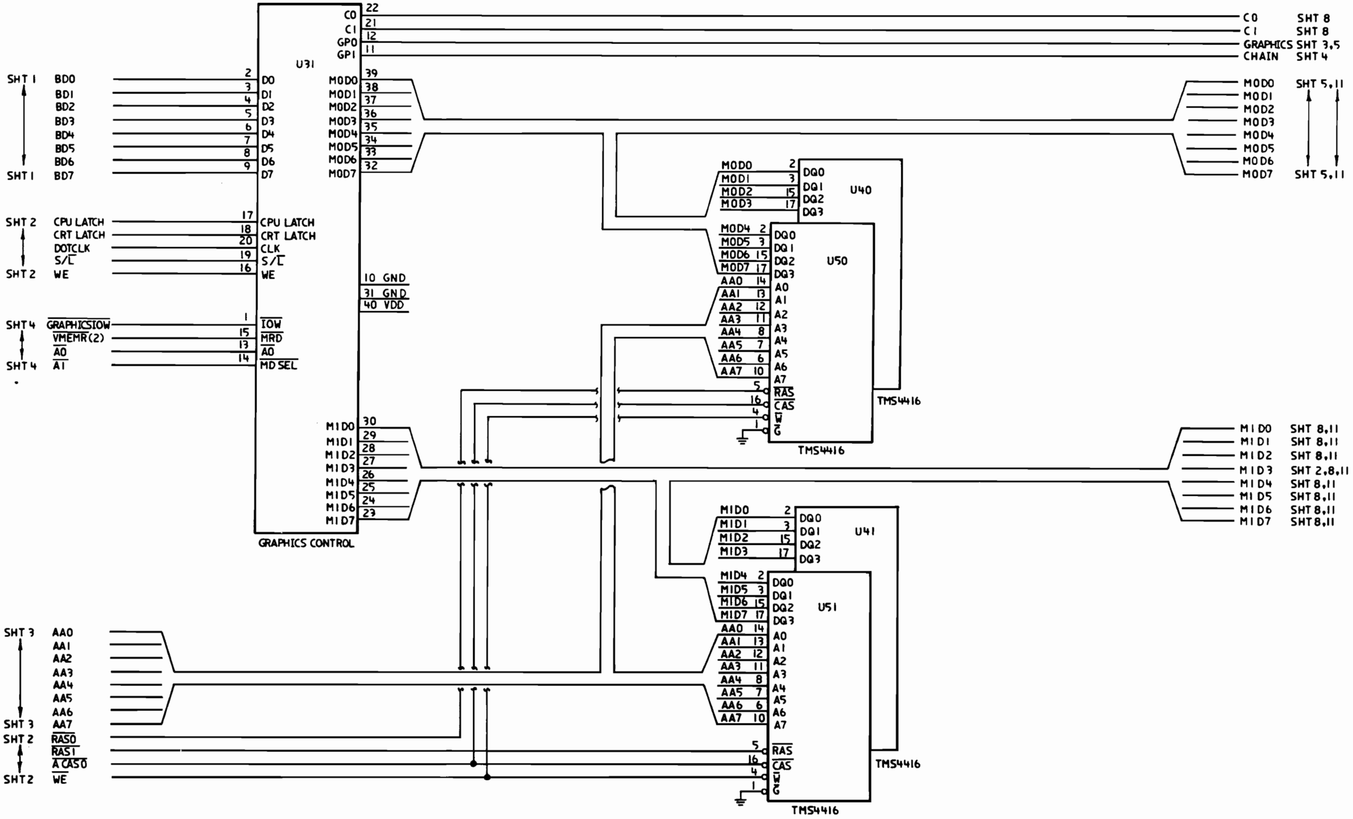
Enhanced Graphics Adapter Sheet 3 of 11



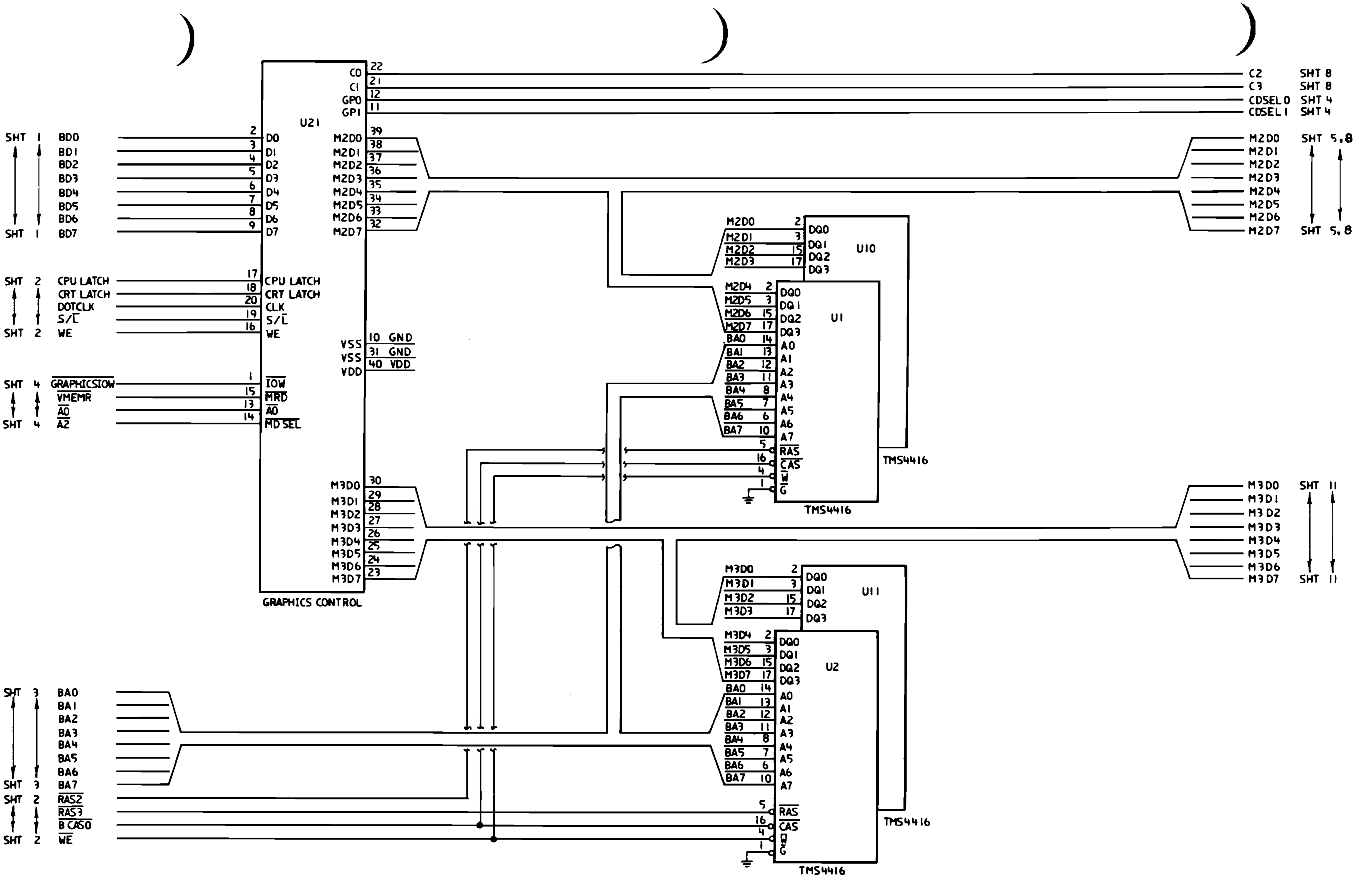
Enhanced Graphics Adapter Sheet 4 of 11



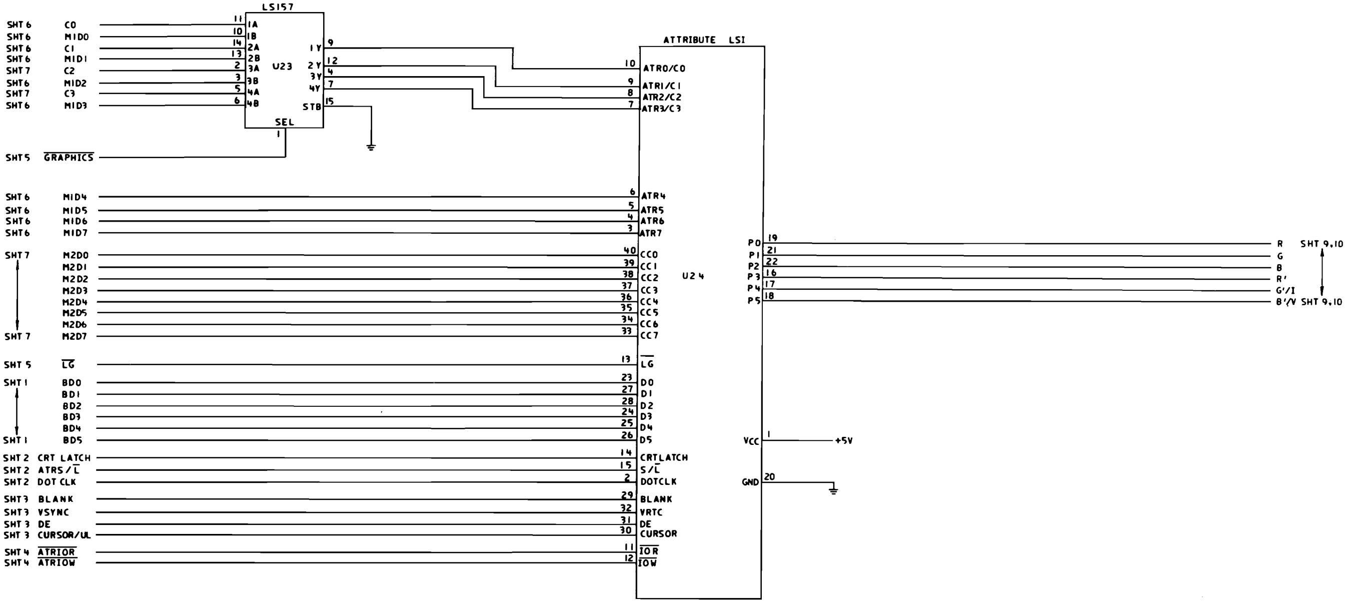
Enhanced Graphics Adapter Sheet 5 of 11



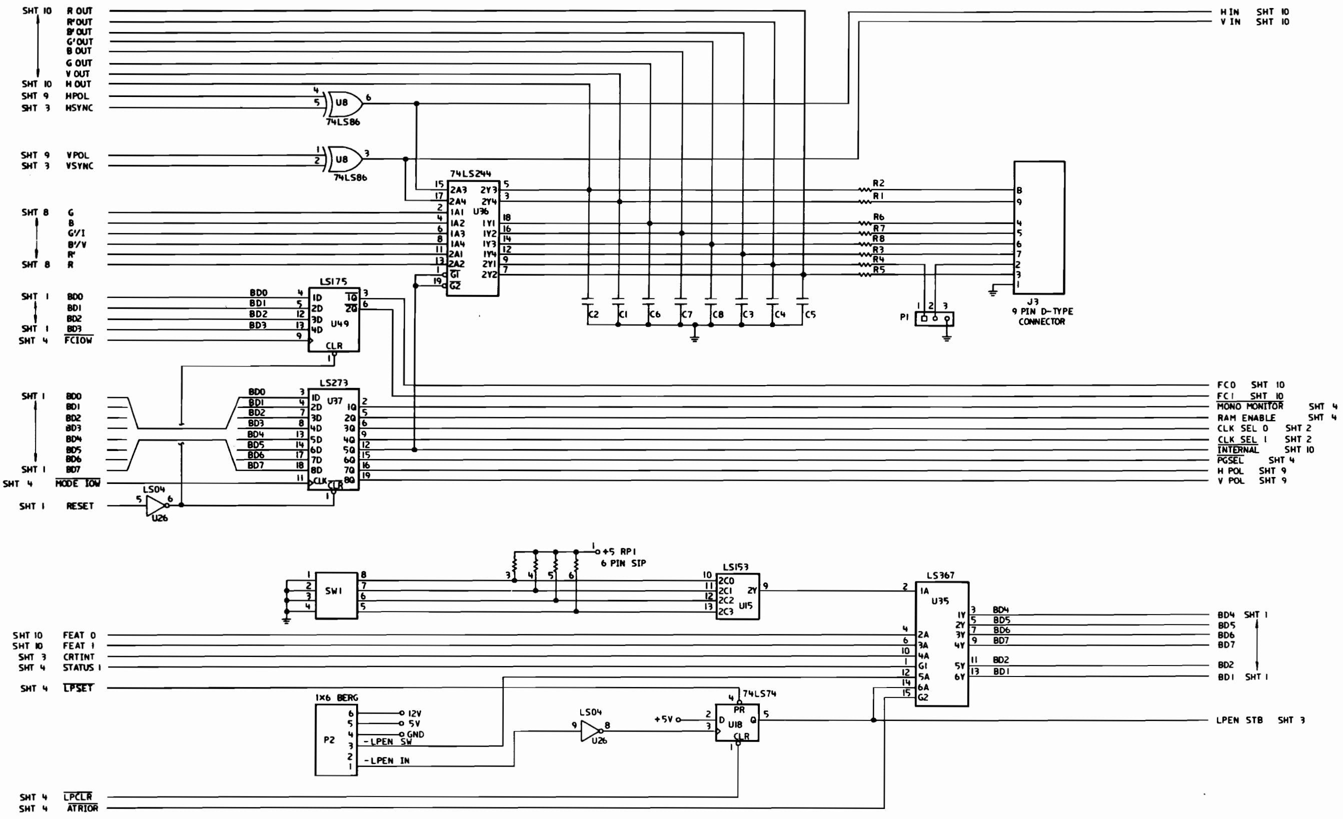
Enhanced Graphics Adapter Sheet 6 of 11



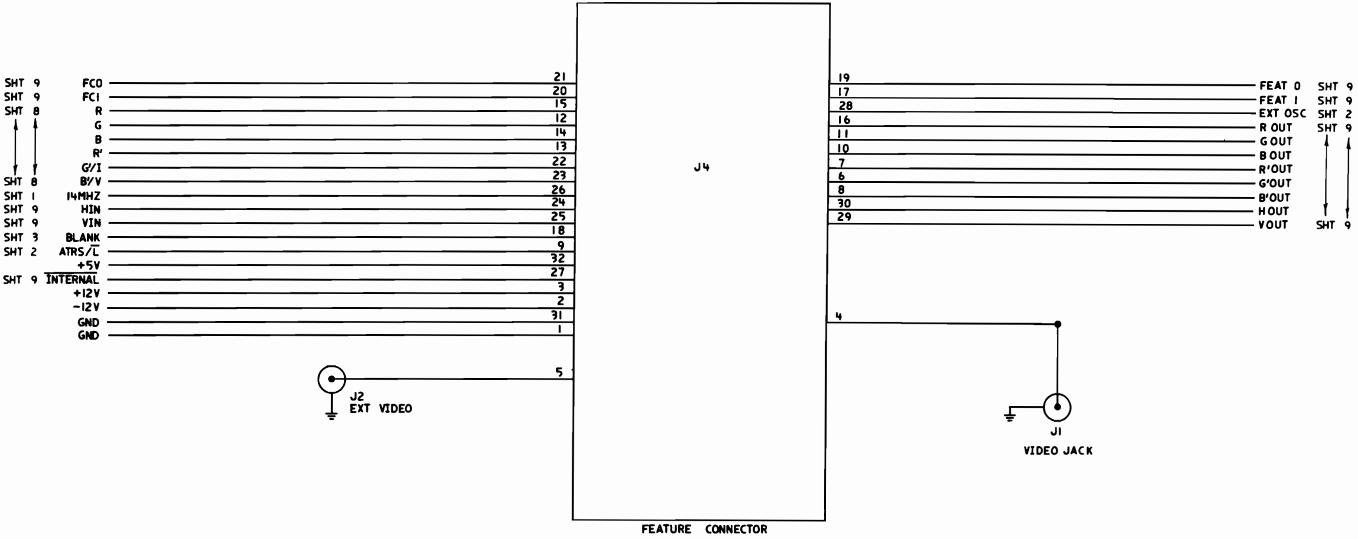
Enhanced Graphics Adapter Sheet 7 of 11



Enhanced Graphics Adapter Sheet 8 of 11



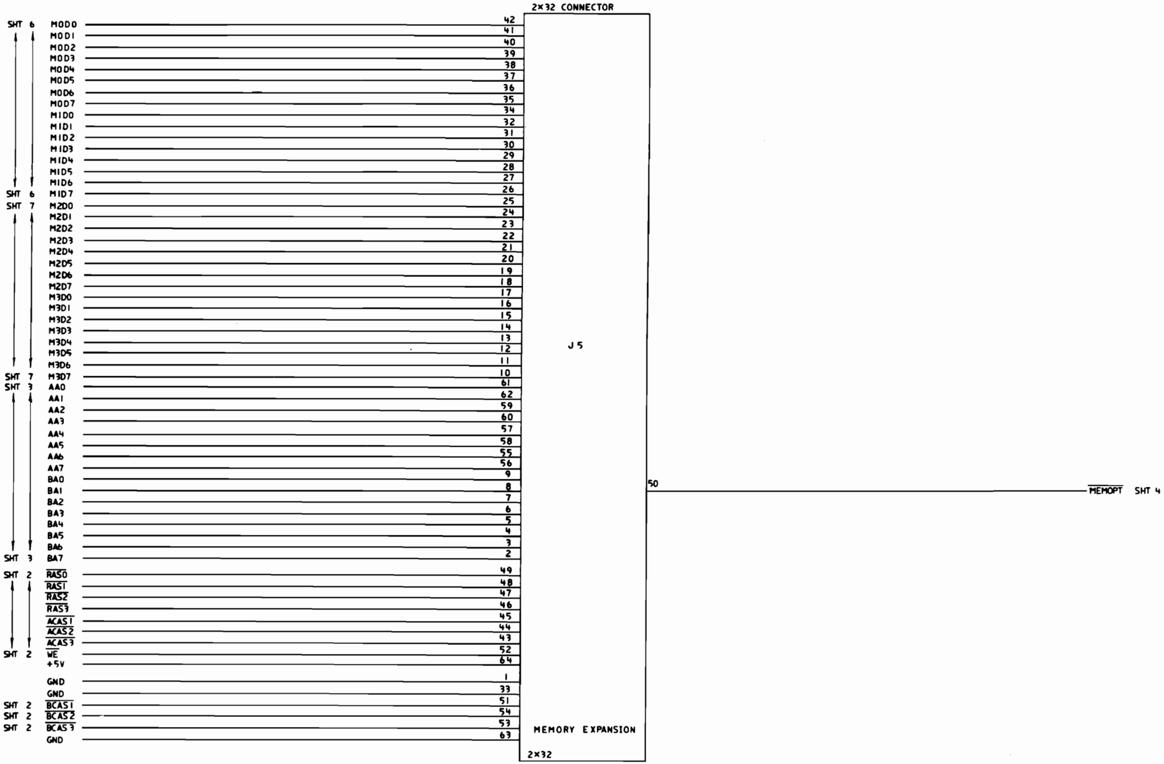
Enhanced Graphics Adapter Sheet 9 of 11



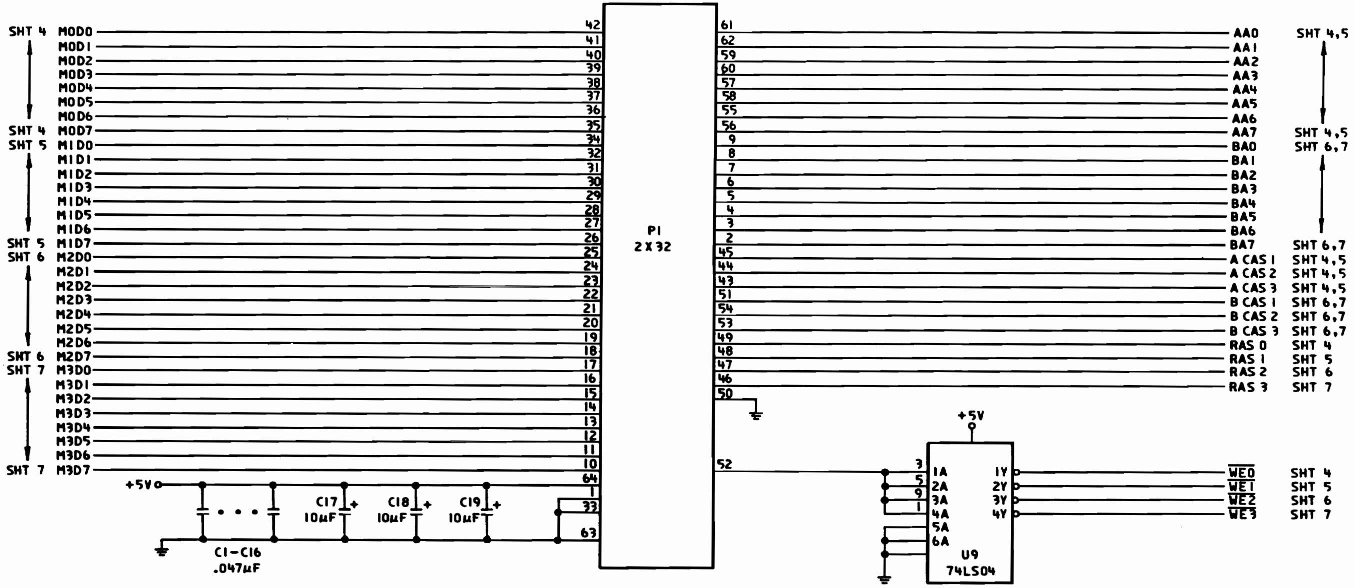
NOTE :

1 GROUNDS - ONE AT EACH END OF CONNECTOR.

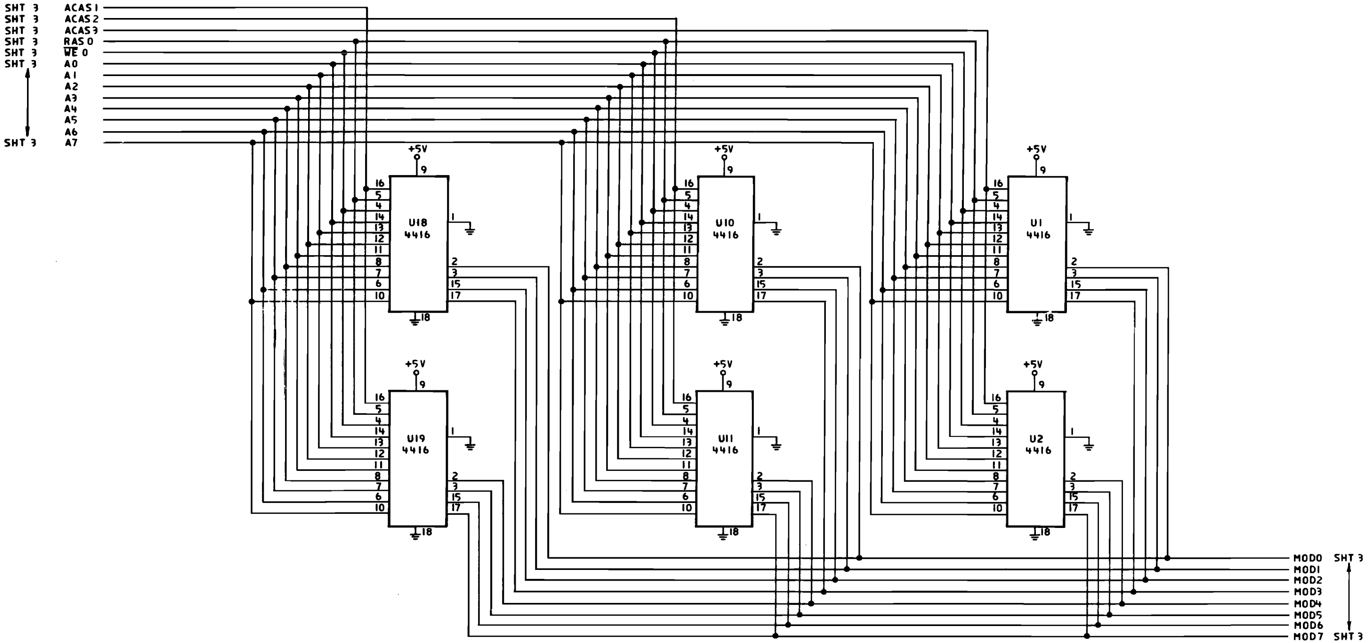
Enhanced Graphics Adapter Sheet 10 of 11



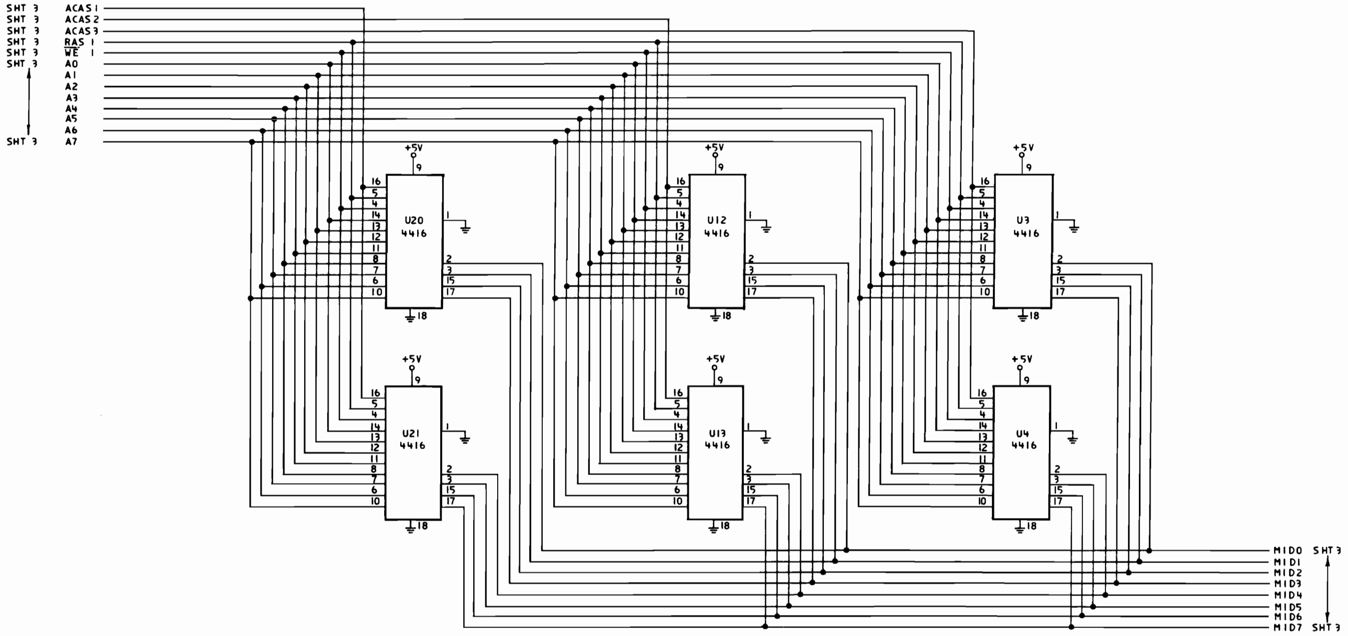
Enhanced Graphics Adapter Sheet 11 of 11



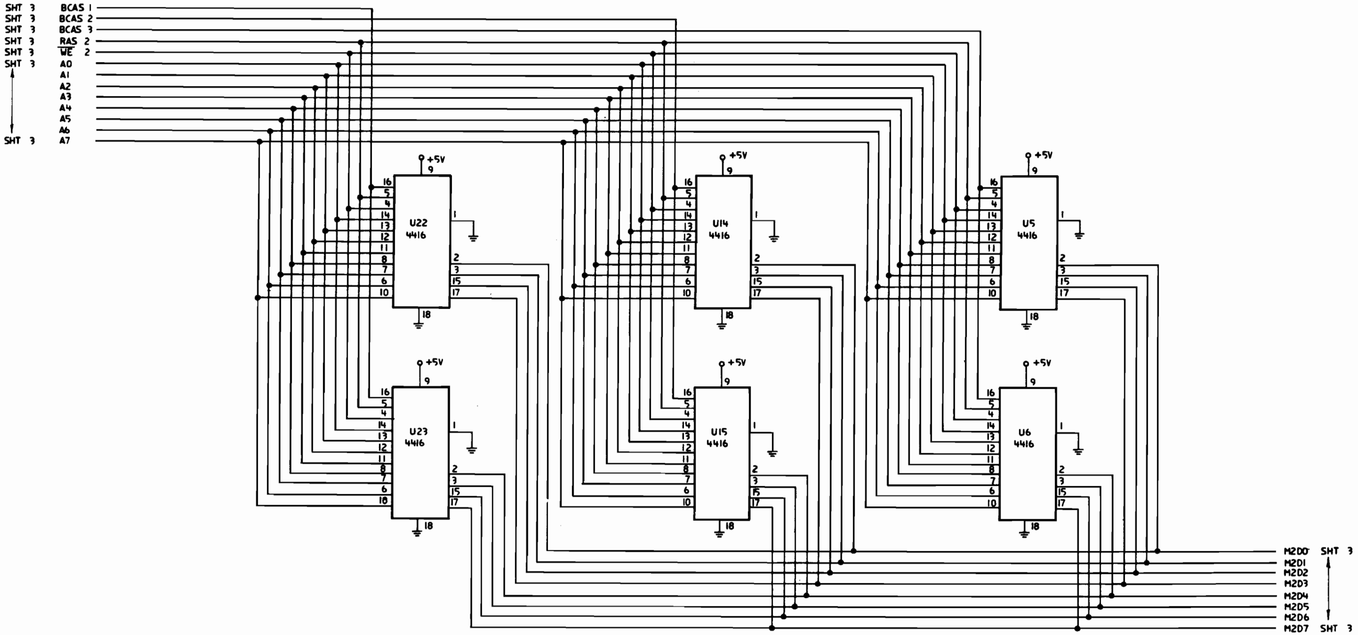
Graphics Memory Expansion Card Sheet 1 of 5



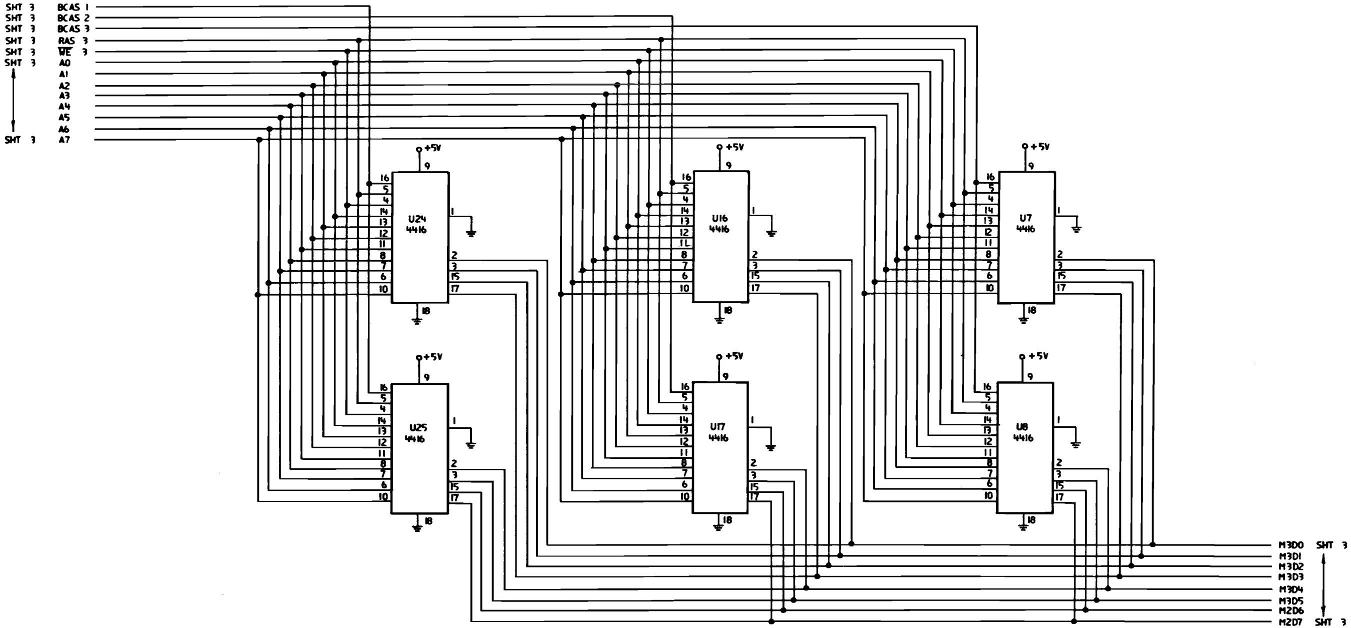
Graphics Memory Expansion Card Sheet 2 of 5



Graphics Memory Expansion Card Sheet 3 of 5



Graphics Memory Expansion Card Sheet 4 of 5



Graphics Memory Expansion Card Sheet 5 of 5

Vectors with Special Meanings

Interrupt Hex 42 - Reserved

When an IBM Enhanced Graphics Adapter is installed, the BIOS routines use interrupt 42 to revector the video pointer.

Interrupt Hex 43 - IBM Enhanced Graphics Video Parameters

When an IBM Enhanced Graphics Adapter is installed, the BIOS routines use this vector to point to a data region containing the parameters required for the initializing of the IBM Enhanced Graphics Adapter. Note that the format of the table must adhere to the BIOS conventions established in the listing. The power-on routines initialize this vector to point to the parameters contained in the IBM Enhanced Graphics Adapter ROM.

Interrupt Hex 44 - Graphics Character Table

When an IBM Enhanced Graphics Adapter is installed the BIOS routines use this vector to point to a table of dot patterns that will be used when graphics characters are to be displayed. This table will be used for the first 128 code points in video modes 4, 5, and 6. This table will be used for 256 characters in all additional graphics modes. See the appropriate BIOS interface for additional information on setting and using the graphics character table pointer.

```
           PAGE,120
           TITLE   ENHANCED GRAPHICS ADAPTER BIOS
           EXTRN   CGMN:NEAR, CGDDOT:NEAR, INT_1F_1:NEAR, CGMN_FDG:NEAR
           EXTRN   END_ADDRESS:NEAR

   ;----------------------------------------------------------------
   ;       THE  BIOS ROUTINES ARE MEANT TO BE ACCESSED THROUGH     :
   ;       SOFTWARE INTERRUPTS ONLY.  ANY ADDRESSES PRESENT IN     :
   ;       THE LISTINGS  ARE INCLUDED  ONLY FOR  COMPLETENESS,     :
   ;       NOT FOR  REFERENCE.   APPLICATIONS WHICH  REFERENCE     :
   ;       ABSOLUTE   ADDRESSES   WITHIN   THE   CODE  SEGMENT     :
   ;       VIOLATE THE STRUCTURE AND DESIGN OF BIOS.               :
   ;----------------------------------------------------------------

   .LIST
C          INCLUDE         VFRONT.INC
C          SUBTTL  VFRONT.INC
C          PAGE
C
C  ;--- INT 10 ----------------------------------------------------------------
C  ; VIDEO_IO                                                                :
C  ;       THESE ROUTINES PROVIDE THE CRT INTERFACE                          :
C  ;       THE FOLLOWING FUNCTIONS ARE PROVIDED:                             :
C  ;       (AH)=0  SET MODE (AL) CONTAINS MODE VALUE                         :
C  ;                                                                         :
C  ;       AL AD   TYPE    RES      NOTES  DF-DIM  DISPLAY      MAX PGS    :
C  ;       ------------------------------------------------------------    :
C  ;     * 0 B8   ALPHA   640X200           40X25   COLOR - BW      8       :
C  ;        1 B8   ALPHA   640X200           40X25   COLOR           8       :
C  ;     * 2 B8   ALPHA   640X200           80X25   COLOR - BW      8       :
C  ;        3 B8   ALPHA   640X200           80X25   COLOR           8       :
C  ;        4 B8   GRPHX   320X200           40X25   COLOR           1       :
C  ;        5 B8   GRPHX   320X200           40X25   COLOR - BW      1       :
C  ;        6 B8   GRPHX   640X200           80X25   COLOR - BW      1       :
C  ;     * 7 BO   ALPHA   720X350           80X25   MONOCHROME      8       :
C  ;                                                                         :
C  ;        8      RESERVED                                                  :
C  ;        9      RESERVED                                                  :
C  ;        A      RESERVED                                                  :
C  ;        B      RESERVED - INTERNAL USE                                   :
C  ;        C      RESERVED - INTERNAL USE                                   :
C  ;                                                                         :
C  ;        D A0   GRPHX   320X200           40X25   COLOR           8       :
C  ;        E A0   GRPHX   640X200           80X25   COLOR           4       :
C  ;        F A0   GRPHX   640X350           80X25   MONOCHROME      2       :
C  ;       10 A0   GRPHX   640X350           80X25   HI RES          2       :
C  ;                                                                         :
C  ;                                                                         :
C  ;               NOTE : HIGH BIT AL SET PREVENTS REGEN BUFFER CLEAR ON  :
C  ;                       MODES RUNNING ON THE COMBO VIDEO ADAPTER        :
C  ;                                                                         :
C  ;               *** NOTE BW MODES OPERATE SAME AS COLOR MODES, BUT      :
C  ;                        COLOR BURST IS NOT ENABLED                     :
C  ;       (AH)=1  SET CURSOR TYPE                                           :
C  ;               (CH) =  BITS 4-0 = START LINE FOR CURSOR                :
C  ;                       ** HARDWARE WILL ALWAYS CAUSE BLINK             :
C  ;                       ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC        :
C  ;                          BLINKING OR NO CURSOR AT ALL                 :
C  ;               (CL) =  BITS 4-0 = END LINE FOR CURSOR                  :
C  ;       (AH)=2  SET CURSOR POSITION                                       :
C  ;               (DH,DL) = ROW,COLUMN  (0,0) IS UPPER LEFT               :
C  ;               (BH) = PAGE NUMBER                                        :
C  ;       (AH)=3  READ CURSOR POSITION                                      :
C  ;               (BH) = PAGE NUMBER                                        :
C  ;               ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR            :
C  ;                          (CH,CL) = CURSOR MODE CURRENTLY SET            :
C  ;       (AH)=4  READ LIGHT PEN POSITION                                   :
C  ;               ON EXIT:                                                  :
C  ;               (AH) = 0 -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED     :
C  ;               (AH) = 1 -- VALID LIGHT PEN VALUE IN REGISTERS            :
C  ;                       (DH,DL) = ROW,COLUMN OF CHARACTER LP POSN       :
C  ;                       (CH) = RASTER LINE (0-199)                      :
C  ;                       (CX) = RASTER LINE (0-NNN) NEW GRAPHICS MODES   :
C  ;                       (BX) = PIXEL COLUMN (0-319,639)                 :
C  ;       (AH)=5  SELECT ACTIVE DISPLAY PAGE                                :
C  ;               (AL) = NEW PAGE VALUE, SEE AH=0 FOR PAGE INFO             :
C  ;       (AH)=6  SCROLL ACTIVE PAGE UP                                     :
C  ;               (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT BOTTOM   :
C  ;                      OF WINDOW                                          :
C  ;                       AL = 0 MEANS BLANK ENTIRE WINDOW                :
C  ;               (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL     :
C  ;               (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL    :
C  ;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE               :
C  ;       (AH)=7  SCROLL ACTIVE PAGE DOWN                                   :
C  ;               (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP      :
C  ;                      OF WINDOW                                          :
C  ;                       AL = 0 MEANS BLANK ENTIRE WINDOW                :
C  ;               (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL     :
C  ;               (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL    :
C  ;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE               :
C  ;                                                                         :
C  ;       CHARACTER HANDLING ROUTINES                                       :
C  ;                                                                         :
C  ;       (AH) = 8 READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION    :
C  ;               (BH) = DISPLAY PAGE                                       :
C  ;               ON EXIT:                                                  :
C  ;               (AL) = CHAR READ                                          :
C  ;               (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)   :
C  ;       (AH) = 9 WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION   :
C  ;               (BH) = DISPLAY PAGE                                       :
C  ;               (CX) = COUNT OF CHARACTERS TO WRITE                       :
C  ;               (AL) = CHAR TO WRITE                                      :
C  ;               (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR     :
C  ;                      (GRAPHICS)                                         :
C  ;                       SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.      :
C  ;       (AH) = A  WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION       :
C  ;               (BH) = DISPLAY PAGE                                       :
C  ;               (CX) = COUNT OF CHARACTERS TO WRITE                       :
C  ;               (AL) = CHAR TO WRITE                                      :
C  ;       FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE  :
C  ;               CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE  :
C  ;               MAINTAINED IN THE SYSTEM ROM.  ONLY THE 1ST 128 CHARS   :
C  ;               ARE CONTAINED THERE.  TO READ/WRITE THE SECOND 128      :
C  ;               CHARS, THE USER MUST INITIALIZE THE POINTER AT          :
C  ;               INTERRUPT 1FH (LOCATION 0007CH) TO POINT TO THE 1K BYTE :
C  ;               TABLE CONTAINING THE CODE POINTS FOR THE SECOND         :
C  ;               128 CHARS (128-255).                                      :
C  ;                                                                         :
C  ;               FOR THE NEW GRAPHICS MODES 256 GRAPHICS CHARS ARE       :
C  ;               SUPPLIED IN THE SYSTEM ROM.                               :
C  ;                                                                         :
C  ;       FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION :
C  ;               FACTOR CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID    :
C  ;               RESULTS ONLY FOR CHARACTERS CONTAINED ON THE SAME ROW.  :
C  ;               CONTINUATION TO SUCCEEDING LINES WILL NOT PRODUCE       :
C  ;               CORRECTLY.                                                :
```

```
C  ;                                                                       :
C  ;       GRAPHICS INTERFACE                                              :
C  ;       (AH) = B  SET COLOR PALETTE                                     :
C  ;               FOR USE IN COMPATIBILITY MODES                          :
C  ;               (BH) = PALETTE COLOR ID BEING SET (0-127)               :
C  ;               (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID        :
C  ;                  NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT   :
C  ;                        HAS MEANING ONLY FOR 320X200 GRAPHICS.         :
C  ;                       COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15):
C  ;                       COLOR ID = 1 SELECTS THE PALETTE TO BE USED:    :
C  ;                                 0 = GREEN(1)/RED(2)/BROWN(3)          :
C  ;                                 1 = CYAN(1)/MAGENTA(2)/WHITE(3)       :
C  ;                       IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET    :
C  ;                               FOR PALETTE COLOR 0 INDICATES THE       :
C  ;                               BORDER COLOR TO BE USED (VALUES 0-31,   :
C  ;                               WHERE 16-31 SELECT THE HIGH INTENSITY   :
C  ;                               BACKGROUND SET).                        :
C  ;       (AH) = C  WRITE DOT                                             :
C  ;               (BH) = PAGE                                             :
C  ;               (DX) = ROW NUMBER                                       :
C  ;               (CX) = COLUMN NUMBER                                    :
C  ;               (AL) = COLOR VALUE                                      :
C  ;                       IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS     :
C  ;                       EXCLUSIVE OR'D WITH THE CURRENT CONTENTS OF     :
C  ;                       THE DOT                                         :
C  ;       (AH) = D  READ DOT                                              :
C  ;               (BH) = PAGE                                             :
C  ;               (DX) = ROW NUMBER                                       :
C  ;               (CX) = COLUMN NUMBER                                    :
C  ;               (AL) RETURNS THE DOT READ                               :
C  ;                                                                       :
C  ; ASCII TELETYPE ROUTINE FOR OUTPUT                                     :
C  ;                                                                       :
C  ;       (AH) = E  WRITE TELETYPE TO ACTIVE PAGE                         :
C  ;               (AL) = CHAR TO WRITE                                    :
C  ;               (BL) = FOREGROUND COLOR IN GRAPHICS MODE                :
C  ;               NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET :
C  ;                                                                       :
C  ;       (AH) = F  CURRENT VIDEO STATE                                   :
C  ;               RETURNS THE CURRENT VIDEO STATE                         :
C  ;               (AL) = MODE CURRENTLY SET  (SEE AH=0 FOR EXPLANATION)   :
C  ;               (AH) = NUMBER OF CHARACTER COLUMNS ON SCREEN            :
C  ;               (BH) = CURRENT ACTIVE DISPLAY PAGE                      :
C  ;                                                                       :
C  ;       (AH) = 10  SET PALETTE REGISTERS                                :
C  ;                                                                       :
C  ;               (AL) = 0        SET INDIVIDUAL PALETTE REGISTER         :
C  ;                BL = PALETTE REGISTER TO BE SET                        :
C  ;                BH = VALUE TO SET                                      :
C  ;                                                                       :
C  ;               AL = 1        SET OVERSCAN REGISTER                     :
C  ;               BH = VALUE TO SET                                       :
C  ;                                                                       :
C  ;               AL = 2        SET ALL PALETTE REGISTERS AND OVERSCAN    :
C  ;               ES:DX POINTS TO A 17 BYTE TABLE                         :
C  ;               BYTES 0 - 15 ARE THE PALETTE VALUES, RESPECTIVELY       :
C  ;               BYTE 16 IS THE OVERSCAN VALUE                           :
C  ;                                                                       :
C  ;               AL = 3          TOGGLE INTENSIFY/BLINKING BIT           :
C  ;                         BL - 0  ENABLE INTENSIFY                      :
C  ;                         BL - 1  ENABLE BLINKING                       :
C  ;                                                                       :
C  ;       (AH) = 11  CHARACTER GENERATOR ROUTINE                          :
C  ;               NOTE : THIS CALL WILL INITIATE A MODE SET, COMPLETELY   :
C  ;                       RESETTING THE VIDEO ENVIRONMENT BUT MAINTAINING :
C  ;                       THE REGEN BUFFER.                               :
C  ;                                                                       :
C  ;               AL = 00   USER ALPHA LOAD                               :
C  ;                       ES:BP - POINTER TO USER TABLE                   :
C  ;                       CX    - COUNT TO STORE                          :
C  ;                       DX    - CHARACTER OFFSET INTO TABLE             :
C  ;                       BL    - BLOCK TO LOAD                           :
C  ;                       BH    - NUMBER OF BYTES PER CHARACTER           :
C  ;               AL = 01   ROM MONOCHROME SET                            :
C  ;                       BL    - BLOCK TO LOAD                           :
C  ;               AL = 02   ROM 8X8 DOUBLE DOT                            :
C  ;                       BL    - BLOCK TO LOAD                           :
C  ;               AL = 03   SET BLOCK SPECIFIER                           :
C  ;                       BL    - CHAR GEN BLOCK SPECIFIER                :
C  ;                               D3-D2  ATTR BIT 3 ONE,  CHAR GEN 0-3    :
C  ;                               D1-D0  ATTR BIT 3 ZERO, CHAR GEN 0-3    :
C  ;                       NOTE : WHEN USING AL = 03 A FUNCTION CALL       :
C  ;                               AX = 1000H                              :
C  ;                               BX = 0712H                              :
C  ;                               IS RECOMMENDED TO SET THE COLOR PLANES  :
C  ;                               RESULTING IN 512 CHARACTERS AND EIGHT   :
C  ;                               CONSISTENT COLORS.                      :
C  ;                                                                       :
C  ;       NOTE : THE FOLLOWING INTERFACE (AL=1X) IS SIMILAR IN FUNCTION   :
C  ;               TO (AL=0X) EXCEPT THAT :                                :
C  ;                       - PAGE ZERO MUST BE ACTIVE                      :
C  ;                       - POINTS (BYTES/CHAR) WILL BE RECALCULATED      :
C  ;                       - ROWS WILL BE CALCULATED FROM THE FOLLOWING:   :
C  ;                               INT[(200 OR 350) / POINTS] - 1          :
C  ;                       - CRT_LEN WILL BE CALCULATED FROM :             :
C  ;                               (ROWS + 1) * CRT_COLS * 2               :
C  ;                       - THE CRTC WILL BE REPROGRAMMED AS FOLLOWS :    :
C  ;                               R09H = POINTS - 1       MAX SCAN LINE   :
C  ;                                       R09H DONE ONLY IN MODE 7        :
C  ;                               ROAH = POINTS - 2       CURSOR START    :
C  ;                               ROBH = 0                CURSOR END      :
C  ;                               R12H =                  VERT DISP END   :
C  ;                                   [(ROWS + 1) * POINTS] - 1           :
C  ;                               R14H = POINTS           UNDERLINE LOC   :
C  ;                                                                       :
C  ;               THE ABOVE REGISTER CALCULATIONS MUST BE CLOSE TO THE    :
C  ;               ORIGINAL TABLE VALUES OR UNDETERMINED RESULTS WILL      :
C  ;               OCCUR.                                                  :
C  ;                                                                       :
C  ;               NOTE : THE FOLLOWING INTERFACE IS DESIGNED TO BE        :
C  ;                       CALLED ONLY IMMEDIATELY AFTER A MODE SET HAS    :
C  ;                       BEEN ISSUED. FAILURE TO ADHERE TO THIS PRACTICE :
C  ;                       MAY CAUSE UNDETERMINED RESULTS.                 :
C  ;                                                                       :
C  ;               AL = 10   USER ALPHA LOAD                               :
C  ;                       ES:BP - POINTER TO USER TABLE                   :
C  ;                       CX    - COUNT TO STORE                          :
C  ;                       DX    - CHARACTER OFFSET INTO TABLE             :
C  ;                       BL    - BLOCK TO LOAD                           :
C  ;                       BH    - NUMBER OF BYTES PER CHARACTER           :
C  ;               AL = 11   ROM MONOCHROME SET                            :
C  ;                       BL    - BLOCK TO LOAD                           :
C  ;               AL = 12   ROM 8X8 DOUBLE DOT                            :
C  ;                       BL    - BLOCK TO LOAD                           :
C  ;                                                                       :
C  ;                                                                       :
```

```
                    C ;                NOTE : THE FOLLOWING INTERFACE IS DESIGNED TO BE        :
                    C ;                       CALLED ONLY IMMEDIATELY AFTER A MODE SET HAS     :
                    C ;                       BEEN ISSUED. FAILURE TO ADHERE TO THIS PRACTICE :
                    C ;                       MAY CAUSE UNDETERMINED RESULTS.                  :
                    C ;                                                                        :
                    C ;                AL = 20   USER GRAPHICS CHARS  INT 01FH (8X8)           :
                    C ;                       ES:BP - POINTER TO USER TABLE                    :
                    C ;                AL = 21   USER GRAPHICS CHARS                           :
                    C ;                       ES:BP - POINTER TO USER TABLE                    :
                    C ;                       CX    - POINTS (BYTES PER CHARACTER)             :
                    C ;                       BL    - ROW SPECIFIER                            :
                    C ;                                                                        :
                    C ;                               BL = 0  USER                             :
                    C ;                                       DL - ROWS                        :
                    C ;                               BL = 1  14 (0EH)                         :
                    C ;                               BL = 2  25 (19H)                         :
                    C ;                               BL = 3  43 (2BH)                         :
                    C ;                                                                        :
                    C ;                AL = 22   ROM 8 X 14 SET                                :
                    C ;                       BL    - ROW SPECIFIER                            :
                    C ;                AL = 23   ROM 8 X 8 DOUBLE DOT                          :
                    C ;                       BL    - ROW SPECIFIER                            :
                    C ;                                                                        :
                    C ;                                                                        :
                    C ;                AL = 30   INFORMATION                                   :
                    C ;                               CX    - POINTS                           :
                    C ;                               DL    - ROWS                             :
                    C ;                       BH    - 0       RETURN CURRENT INT 1FH PTR       :
                    C ;                               ES:BP - PTR TO TABLE                     :
                    C ;                       BH    - 1       RETURN CURRENT INT 44H PTR       :
                    C ;                               ES:BP - PTR TO TABLE                     :
                    C ;                       BH    - 2       RETURN ROM 8 X 14 PTR            :
                    C ;                               ES:BP - PTR TO TABLE                     :
                    C ;                       BH    - 3       RETURN ROM DOUBLE DOT PTR        :
                    C ;                               ES:BP - PTR TO TABLE                     :
                    C ;                       BH    - 4       RETURN ROM DOUBLE DOT PTR (TOP) :
                    C ;                               ES:BP - PTR TO TABLE                     :
                    C ;                       BH    - 5       RETURN ROM ALPHA ALTERNATE 9X14 :
                    C ;                               ES:BP - PTR TO TABLE                     :
                    C ;                                                                        :
                    C ;         (AH) = 12   ALTERNATE SELECT                                   :
                    C ;                                                                        :
                    C ;                BL = 10  RETURN EGA INFORMATION                         :
                    C ;                       BH = 0 - COLOR MODE IN EFFECT <3><D><X>          :
                    C ;                            1 - MONOC MODE IN EFFECT <3><B><X>          :
                    C ;                       BL = MEMORY VALUE                                :
                    C ;                               0 0 - 064K       0 1 - 128K              :
                    C ;                               1 0 - 192K       1 1 - 256K              :
                    C ;                       CH = FEATURE BITS                                :
                    C ;                       CL = SWITCH SETTING                              :
                    C ;                                                                        :
                    C ;                BL = 20  SELECT ALTERNATE PRINT SCREEN ROUTINE          :
                    C ;                                                                        :
                    C ;         (AH) = 13   WRITE STRING                                       :
                    C ;                       ES:BP - POINTER TO STRING TO BE WRITTEN          :
                    C ;                       CX    - CHARACTER ONLY COUNT                     :
                    C ;                       DX    - POSITION TO BEGIN STRING, IN CURSOR      :
                    C ;                               TERMS                                    :
                    C ;                       BH    - PAGE NUMBER                              :
                    C ;                                                                        :
                    C ;                AL = 0                                                  :
                    C ;                       BL    - ATTRIBUTE                                :
                    C ;                       STRING - (CHAR, CHAR, CHAR, ...)                 :
                    C ;                       CURSOR NOT MOVED                                 :
                    C ;                AL = 1                                                  :
                    C ;                       BL    - ATTRIBUTE                                :
                    C ;                       STRING - (CHAR, CHAR, CHAR, ...)                 :
                    C ;                       CURSOR  IS MOVED                                 :
                    C ;                AL = 2                                                  :
                    C ;                       STRING - (CHAR, ATTR, CHAR, ATTR, ...)           :
                    C ;                       CURSOR NOT MOVED                                 :
                    C ;                AL = 3                                                  :
                    C ;                       STRING - (CHAR, ATTR, CHAR, ATTR, ...)           :
                    C ;                       CURSOR  IS MOVED                                 :
                    C ;                                                                        :
                    C ;                NOTE : CHAR RET, LINE FEED, BACKSPACE, AND BELL ARE     :
                    C ;                       TREATED AS COMMANDS RATHER THAN PRINTABLE        :
                    C ;                       CHARACTERS.                                      :
                    C ;                                                                        :
                    C ;-------------------------------------------------------------------------
                    C
                    C SRLOAD  MACRO   SEGREG,VALUE
                    C         IFNB <VALUE>
                    C         IFIDN <VALUE>,<0>
                    C         SUB     DX,DX
                    C         ELSE
                    C         MOV     DX,VALUE
                    C         ENDIF
                    C         ENDIF
                    C         MOV     SEGREG,DX
                    C         ENDM
                    C
                    C
                    C ;----- LOW MEMORY SEGMENT
                    C
0000                C ABS0    SEGMENT AT      0
0014                C         ORG     005H*4                      ; PRINT SCREEN VECTOR
0014                C INT5_PTR        LABEL   DWORD
0040                C         ORG     010H*4                      ; VIDEO I/O VECTOR
0040                C VIDEO           LABEL   DWORD
007C                C         ORG     01FH*4                      ; GRAPHIC CHARS 128-255
007C                C EXT_PTR         LABEL   DWORD
                    C
0108                C         ORG     042H*4                      ; REVECTORED 10H*4
0108                C PLANAR_VIDEO    LABEL   DWORD
                    C
010C                C         ORG     043H*4                      ; GRAPHIC CHARS 0-255
010C                C GRX_SET         LABEL   DWORD
                    C
0410                C         ORG     0410H
0410                C EQUIP_LOW       LABEL   BYTE
0410  ????          C EQUIP_FLAG      DW      ?
                    C
                    C ;----- REUSE RAM FROM PLANAR BIOS
                    C
0449                C         ORG     449H
0449  ??            C CRT_MODE        DB      ?
044A  ????          C CRT_COLS        DW      ?
044C  ????          C CRT_LEN         DW      ?
044E  ????          C CRT_START       DW      ?
0450      08 [      C CURSOR_POSN     DW      8 DUP(?)
               ???? C
            ]       C
                    C
0460  ????          C CURSOR_MODE     DW      ?
0462  ??            C ACTIVE_PAGE     DB      ?
```

```
0463  ????        C  ADDR_6845       DW      ?
0465  ??          C  CRT_MODE_SET    DB      ?
0466  ??          C  CRT_PALETTE     DB      ?
                  C
0472              C          ORG     0472H
0472  ????        C  RESET_FLAG      DW      ?
0484              C          ORG     0484H
0484  ??          C  ROWS    DB      ?                       ; ROWS ON THE SCREEN
0485  ????        C  POINTS  DW      ?                       ; BYTES PER CHARACTER
                  C
0487  ??          C  INFO    DB      ?
                  C
                  C  ; INFO
                  C  ;       D7 - HIGH BIT OF MODE SET, CLEAR/NOT CLEAR REGEN
                  C  ;       D6 - MEMORY  D6 D5 = 0 0 - 064K    0 1 - 128K
                  C  ;       D5 - MEMORY          1 0 - 192K    1 1 - 256K
                  C  ;       D4 - RESERVED
                  C  ;       D3 - EGA ACTIVE MONITOR (0), EGA NOT ACTIVE (1)
                  C  ;       D2 - WAIT FOR DISPLAY ENABLE (1)
                  C  ;       D1 - EGA HAS A MONOCHROME ATTACHED (1)
                  C  ;       D0 - SET C_TYPE EMULATE ACTIVE (0)
                  C
0488  ??          C  INFO_3  DB      ?
                  C
                  C  ; INFO_3
                  C  ;       D7-D4   FEATURE BITS
                  C  ;       D3-D0   SWITCHES
                  C
04A8              C          ORG     04A8H
04A8              C  SAVE_PTR        LABEL   DWORD
                  C
                  C  ;----- SAVE_PTR
                  C  ;
                  C  ;       SAVE_PTR IS A POINTER TO A TABLE AS DESCRIBED AS FOLLOWS :
                  C  ;
                  C  ;       DWORD_1         VIDEO PARAMETER TABLE POINTER
                  C  ;       DWORD_2         DYNAMIC SAVE AREA POINTER
                  C  ;       DWORD_3         ALPHA MODE AUXILIARY CHAR GEN POINTER
                  C  ;       DWORD_4         GRAPHICS MODE AUXILIARY CHAR GEN POINTER
                  C  ;       DWORD_5         RESERVED
                  C  ;       DWORD_6         RESERVED
                  C  ;       DWORD_7         RESERVED
                  C  ;
                  C  ;       DWORD_1         PARAMETER TABLE POINTER
                  C  ;                       INITIALIZED TO BIOS EGA PARAMETER TABLE.
                  C  ;                       THIS VALUE MUST EXIST.
                  C  ;
                  C  ;       DWORD_2         PARAMETER SAVE AREA POINTER
                  C  ;                       INITIALIZED TO 0000:0000, THIS VALUE IS OPTIONAL.
                  C  ;                       WHEN NON-ZERO, THIS POINTER WILL BE USED AS POINTER
                  C  ;                       TO A RAM AREA WHERE CERTAIN DYNAMIC VALUES ARE TO
                  C  ;                       BE SAVED. WHEN IN EGA OPERATION THIS RAM AREA WILL
                  C  ;                       HOLD THE 16 EGA PALETTE REGISTER VALUES PLUS
                  C  ;                       THE OVERSCAN VALUE IN BYTES 0-16D RESPECTIVELY.
                  C  ;                       AT LEAST 256 BYTES MUST BE ALLOCATED FOR THIS AREA.
                  C  ;
                  C  ;       DWORD_3         ALPHA MODE AUXILIARY POINTER
                  C  ;                       INITIALIZED TO 0000:0000, THIS VALUE IS OPTIONAL.
                  C  ;                       WHEN NON-ZERO, THIS POINTER IS USED AS A POINTER
                  C  ;                       TO A TABLES DESCRIBED AS FOLLOWS :
                  C  ;
                  C  ;                       BYTE    BYTES/CHARACTER
                  C  ;                       BYTE    BLOCK TO LOAD, SHOULD BE ZERO FOR NORMAL
                  C  ;                               OPERATION
                  C  ;                       WORD    COUNT TO STORE, SHOULD BE 256D FOR NORMAL
                  C  ;                               OPERATION
                  C  ;                       WORD    CHARACTER OFFSET, SHOULD BE ZERO FOR NORMAL
                  C  ;                               OPERATION
                  C  ;                       DWORD   POINTER TO A FONT TABLE
                  C  ;                       BYTE    DISPLAYABLE ROWS
                  C  ;                               IF 'FF' THE MAXIMUM CALCULATED VALUE WILL BE
                  C  ;                               USED, ELSE THIS VALUE WILL BE USED
                  C  ;                       BYTE    CONSECUTIVE BYTES OF MODE VALUES FOR WHICH
                  C  ;                               THIS FONT DESCRIPTION IS TO BE USED.
                  C  ;                               THE END OF THIS STREAM IS INDICATED BY A
                  C  ;                               BYTE CODE OF 'FF'
                  C  ;
                  C  ;                       NOTE : USE OF THIS POINTER MAY CAUSE UNEXPECTED
                  C  ;                              CURSOR TYPE OPERATION. FOR AN EXPLANATION
                  C  ;                              OF CURSOR TYPE SEE AH = 01 IN THE INTERFACE
                  C  ;                              SECTION.
                  C  ;
                  C  ;       DWORD_4         GRAPHICS MODE AUXILIARY POINTER
                  C  ;                       INITIALIZED TO 0000:0000, THIS VALUE IS OPTIONAL.
                  C  ;                       WHEN NON-ZERO, THIS POINTER IS USED AS A POINTER
                  C  ;                       TO A TABLES DESCRIBED AS FOLLOWS :
                  C  ;
                  C  ;                       BYTE    DISPLAYABLE ROWS
                  C  ;                       WORD    BYTES PER CHARACTER
                  C  ;                       DWORD   POINTER TO A FONT TABLE
                  C  ;                       BYTE    CONSECUTIVE BYTES OF MODE VALUES FOR WHICH
                  C  ;                               THIS FONT DESCRIPTION IS TO BE USED.
                  C  ;                               THE END OF THIS STREAM IS INDICATED BY A
                  C  ;                               BYTE CODE OF 'FF'
                  C  ;
                  C  ;       DWORD_5 THRU DWORD_7
                  C  ;                       RESERVED AND SET TO 0000:0000.
                  C  ;
                  C
0500              C          ORG     0500H
0500  ??          C  STATUS_BYTE     DB      ?
0501              C  ABS0    ENDS
                  C
= 0061            C  PORT_B          EQU     61H                     ; 8255 PORT B ADDR
= 0040            C  TIMER           EQU     40H
                  C
                  C  ;----- EQUATES FOR CARD PORT ADDRESSES
                  C
= 00C4            C          SEQ_ADDR        EQU     0C4H
= 00C5            C          SEQ_DATA        EQU     0C5H
= 00D4            C          CRTC_ADDR       EQU     0D4H
= 00B4            C          CRTC_ADDR_B     EQU     0B4H
= 00D5            C          CRTC_DATA       EQU     0D5H            ; OR 0B5H
= 00CC            C          GRAPH_1_POS     EQU     0CCH
= 00CA            C          GRAPH_2_POS     EQU     0CAH
= 00CE            C          GRAPH_ADDR      EQU     0CEH
= 00CF            C          GRAPH_DATA      EQU     0CFH
= 00C2            C          MISC_OUTPUT     EQU     0C2H
= 00C2            C          IN_STAT_0       EQU     0C2H
= 00BA            C          INPUT_STATUS_B  EQU     0BAH
= 00DA            C          INPUT_STATUS    EQU     0DAH
= 00DA            C          ATTR_READ       EQU     0DAH
= 00C0            C          ATTR_WRITE      EQU     0C0H
                  C
                  C  ;----- EQUATES FOR ADDRESS REGISTER VALUES
                  C
```

```
= 0000                C          S_RESET          EQU      00H
= 0001                C          S_CLOCK          EQU      01H
= 0002                C          S_MAP            EQU      02H
= 0003                C          S_CGEN           EQU      03H
= 0004                C          S_MEM            EQU      04H
                      C
= 0000                C          C_HRZ_TOT        EQU      00H
= 0001                C          C_HRZ_DSP        EQU      01H
= 0002                C          C_STRT_HRZ_BLK   EQU      02H
= 0003                C          C_END_HRZ_BLK    EQU      03H
= 0004                C          C_STRT_HRZ_SYN   EQU      04H
= 0005                C          C_END_HRZ_SYN    EQU      05H
= 0006                C          C_VRT_TOT        EQU      06H
= 0007                C          C_OVERFLOW       EQU      07H
= 0008                C          C_PRE_ROW        EQU      08H
= 0009                C          C_MAX_SCAN_LN    EQU      09H
= 000A                C          C_CRSR_START     EQU      0AH
= 000B                C          C_CRSR_END       EQU      0BH
= 000C                C          C_STRT_HGH       EQU      0CH
= 000D                C          C_STRT_LOW       EQU      0DH
= 000E                C          C_CRSR_LOC_HGH   EQU      0EH
= 000F                C          C_CRSR_LOC_LOW   EQU      0FH
= 0010                C          C_VRT_SYN_STRT   EQU      10H             ; WRITE ONLY
= 0010                C          C_LGHT_PEN_HGH   EQU      10H             ; READ ONLY
= 0011                C          C_VRT_SYN_END    EQU      11H             ; WRITE ONLY
= 0011                C          C_LGHT_PEN_LOW   EQU      11H             ; READ ONLY
= 0012                C          C_VRT_DSP_END    EQU      12H
= 0013                C          C_OFFSET         EQU      13H
= 0014                C          C_UNDERLN_LOC    EQU      14H
= 0015                C          C_STRT_VRT_BLK   EQU      15H
= 0016                C          C_END_VRT_BLK    EQU      16H
= 0017                C          C_MODE_CNTL      EQU      17H
= 0018                C          C_LN_COMP        EQU      18H
                      C
= 0000                C          G_SET_RESET      EQU      00H
= 0001                C          G_ENBL_SET       EQU      01H
= 0002                C          G_CLR_COMP       EQU      02H
= 0003                C          G_DATA_ROT       EQU      03H
= 0004                C          G_READ_MAP       EQU      04H
= 0005                C          G_MODE           EQU      05H
= 0006                C          G_MISC           EQU      06H
= 0007                C          G_COLOR          EQU      07H
= 0008                C          G_BIT_MASK       EQU      08H
                      C
= 0010                C          P_MODE           EQU      10H
= 0011                C          P_OVERSC         EQU      11H
= 0012                C          P_CPLANE         EQU      12H
= 0013                C          P_HPEL           EQU      13H
                      C
                      C          SUBTTL

                          ;----- CODE SEGMENT

0000                      CODE   SEGMENT PUBLIC

                      C          INCLUDE          VPOST.INC
                      C          SUBTTL  VPOST.INC
                      C          PAGE
                      C
                      C   ;----- POST
                      C
                      C          ASSUME  CS:CODE,DS:ABSO
0000                  C          ORG     0H
0000  55              C          DB      055H                           ; SIGNATURE
0001  AA              C          DB      0AAH                           ;  BYTES
0002  20              C          DB      020H                           ; LENGTH INDICATOR
                      C
                      C   ;----- NOTE : DO NOT USE THE SIGNATURE BYTES AS A PRESENCE TEST
                      C
                      C   ;      PLANAR VIDEO SWITCH SETTINGS
                      C
                      C   ;      0 0 - UNUSED
                      C   ;      0 1 - 40 X 25 COLOR
                      C   ;      1 0 - 80 X 25 COLOR
                      C   ;      1 1 - 80 X 25 MONOCHROME
                      C   ; NOTE : 0 0 MUST BE SET WHEN THIS ADAPTER IS INSTALLED.
                      C
                      C   ;      VIDEO ADAPTER SWITCH SETTINGS
                      C
                      C   ;      0 0 0 0 - MONOC PRIMARY, EGA COLOR, 40X25
                      C   ;      0 0 0 1 - MONOC PRIMARY, EGA COLOR, 80X25
                      C   ;      0 0 1 0 - MONOC PRIMARY, EGA HI RES EMULATE (SAME AS 0001)
                      C   ;      0 0 1 1 - MONOC PRIMARY, EGA HI RES ENHANCED
                      C   ;      0 1 0 0 - COLOR 40 PRIMARY, EGA MONOCHROME
                      C   ;      0 1 0 1 - COLOR 80 PRIMARY, EGA MONOCHROME
                      C
                      C   ;      0 1 1 0 - MONOC SECONDARY, EGA COLOR, 40X25
                      C   ;      0 1 1 1 - MONOC SECONDARY, EGA COLOR, 80X25
                      C   ;      1 0 0 0 - MONOC SECONDARY, EGA HI RES EMULATE (SAME AS 0111)
                      C   ;      1 0 0 1 - MONOC SECONDARY, EGA HI RES ENHANCED
                      C   ;      1 0 1 0 - COLOR 40 SECONDARY, EGA MONOCHROME
                      C   ;      1 0 1 1 - COLOR 80 SECONDARY, EGA MONOCHROME
                      C
                      C   ;      1 1 0 0 - RESERVED
                      C   ;      1 1 0 1 - RESERVED
                      C   ;      1 1 1 0 - RESERVED
                      C   ;      1 1 1 1 - RESERVED
                      C
                      C   ;----- SETUP ROUTINE FOR THIS MODULE
                      C
0003                  C   VIDEO_SETUP    PROC     FAR
0003  EB 28           C          JMP     SHORT    L1
0005  32 34 30 30     C          DB      '2400'
0009  36 32 37 37 33 35  C       DB      '6277356 (C)COPYRIGHT IBM 1984'
      36 20 28 43 29 43  C
      4F 50 59 52 49 47  C
      48 54 20 49 42 4D  C
      20 31 39 38 34     C
0026  39 2F 31 33 2F 38  C       DB      '9/13/84'
      34              C
                      C
                      C   ;----- SET UP VIDEO VECTORS
                      C
002D                  C   L1:
002D  B6 03           C          MOV     DH,3
002F  B2 DA           C          MOV     DL,INPUT_STATUS
0031  EC              C          IN      AL,DX
0032  B2 BA           C          MOV     DL,INPUT_STATUS_B
0034  EC              C          IN      AL,DX
0035  B2 C0           C          MOV     DL,ATTR_WRITE
0037  B0 00           C          MOV     AL,0
0039  EE              C          OUT     DX,AL
                      C
                      C          SRLOAD  DS,0
003A  2B D2           C+         SUB     DX,DX
003C  8E DA           C+         MOV     DS,DX
```

```
003E  FA                       C                CLI
003F  C7 06 0040 R 0CD7 R      C                MOV     WORD PTR VIDEO,OFFSET COMBO_VIDEO
0045  8C 0E 0042 R             C                MOV     WORD PTR VIDEO+2, CS
0049  C7 06 0108 R F065        C                MOV     WORD PTR PLANAR_VIDEO,0F065H
004F  C7 06 010A R F000        C                MOV     WORD PTR PLANAR_VIDEO+2,0F000H
0055  C7 06 04A8 R 010C R      C                MOV     WORD PTR SAVE_PTR,OFFSET SAVE_TBL
005B  8C 0E 04AA R             C                MOV     WORD PTR SAVE_PTR+2, CS
005F  C7 06 007C R 0000 E      C                MOV     WORD PTR EXT_PTR, OFFSET INT_1F_1
0065  8C 0E 007E R             C                MOV     WORD PTR EXT_PTR+2, CS
0069  C7 06 010C R 0000 E      C                MOV     WORD PTR GRX_SET, OFFSET CGDDOT
006F  8C 0E 010E R             C                MOV     WORD PTR GRX_SET+2, CS
0073  FB                       C                STI
                               C
                               C        ;----- POST FOR COMBO VIDEO CARD
                               C
0074  C6 06 0487 R 04          C                MOV     INFO,00000100B
0079  E8 009B R                C                CALL    RD_SWS
007C  88 1E 0488 R             C                MOV     INFO_3,BL
0080  E8 00CE R                C                CALL    F_BTS
0083  08 06 0488 R             C                OR      INFO_3,AL
0087  8A 1E 0488 R             C                MOV     BL,INFO_3
008B  E8 00F3 R                C                CALL    MK_ENV
008E  E9 0244 R                C                JMP     POST
0091                           C        SKIP:
0091  CB                       C                RET
0092                           C        VIDEO_SETUP     ENDP
                               C
                               C
0092                           C        POR_1   PROC    NEAR
0092  EE                       C                OUT     DX,AL
0093  50                       C                PUSH    AX
0094  58                       C                POP     AX
0095  EC                       C                IN      AL,DX
0096  24 10                    C                AND     AL,010H
0098  D0 E8                    C                SHR     AL,1
009A  C3                       C                RET
009B                           C        POR_1   ENDP
                               C
                               C        ;----- READ THE SWITCH SETTINGS ON THE CARD
                               C
009B                           C        RD_SWS  PROC    NEAR
                               C                ASSUME  DS:ABS0
009B  B6 03                    C                MOV     DH,3
009D  B2 C2                    C                MOV     DL,MISC_OUTPUT
009F  B0 01                    C                MOV     AL,1
00A1  EE                       C                OUT     DX,AL
                               C
                               C        ;----- COULD BE 0,4,8,C
                               C
00A2  B0 0D                    C                MOV     AL,0DH
00A4  E8 0092 R                C                CALL    POR_1
00A7  D0 E8                    C                SHR     AL,1
00A9  D0 E8                    C                SHR     AL,1
00AB  D0 E8                    C                SHR     AL,1
00AD  8A D8                    C                MOV     BL,AL
                               C
00AF  B0 09                    C                MOV     AL,9
00B1  E8 0092 R                C                CALL    POR_1
00B4  D0 E8                    C                SHR     AL,1
00B6  D0 E8                    C                SHR     AL,1
00B8  0A D8                    C                OR      BL,AL
                               C
00BA  B0 05                    C                MOV     AL,5
00BC  E8 0092 R                C                CALL    POR_1
00BF  D0 E8                    C                SHR     AL,1
00C1  0A D8                    C                OR      BL,AL
                               C
00C3  B0 01                    C                MOV     AL,1
00C5  E8 0092 R                C                CALL    POR_1
00C8  0A D8                    C                OR      BL,AL
                               C
00CA  80 E3 0F                 C                AND     BL,0FH
00CD  C3                       C                RET
00CE                           C        RD_SWS  ENDP
                               C
                               C        ;----- OBTAIN THE FEATURE BITS FROM DAUGHTER CARD
                               C
00CE                           C        F_BTS   PROC    NEAR
00CE  B6 03                    C                MOV     DH,3
00D0  B2 BA                    C                MOV     DL,0BAH
00D2  B0 01                    C                MOV     AL,1
00D4  EE                       C                OUT     DX,AL
00D5  B2 DA                    C                MOV     DL,0DAH
00D7  EE                       C                OUT     DX,AL
00D8  B2 C2                    C                MOV     DL,IN_STAT_0
00DA  EC                       C                IN      AL,DX                           ; READ FEATURE BITS
00DB  24 60                    C                AND     AL,060H
00DD  D0 E8                    C                SHR     AL,1
00DF  8A D8                    C                MOV     BL,AL
00E1  B2 BA                    C                MOV     DL,0BAH
00E3  B0 02                    C                MOV     AL,2
00E5  EE                       C                OUT     DX,AL
00E6  B2 DA                    C                MOV     DL,0DAH
00E8  EE                       C                OUT     DX,AL
00E9  B2 C2                    C                MOV     DL,IN_STAT_0
00EB  EC                       C                IN      AL,DX                           ; READ FEATURE BITS
00EC  24 60                    C                AND     AL,060H
00EE  D0 E0                    C                SHL     AL,1
00F0  0A C3                    C                OR      AL,BL
00F2  C3                       C                RET
00F3                           C        F_BTS   ENDP
                               C
                               C        ;----- ESTABLISH THE VIDEO ENVIRONMENT, KEYED OFF OF THE SWITCHES
                               C
00F3                           C        MK_ENV  PROC    NEAR
                               C                ASSUME  DS:ABS0
00F3  2A FF                    C                SUB     BH,BH
00F5  80 E3 0F                 C                AND     BL,0FH
00F8  D1 E3                    C                SAL     BX,1
00FA  52                       C                PUSH    DX
00FB  B6 03                    C                MOV     DH,3
00FD  8A E6                    C                MOV     AH,DH
00FF  5A                       C                POP     DX
0100  80 E4 01                 C                AND     AH,1
0103  FE C4                    C                INC     AH
0105  F6 D4                    C                NOT     AH
0107  2E: FF A7 0128 R         C                JMP     WORD PTR CS:[BX + OFFSET T5]
                               C
010C                           C        SAVE_TBL        LABEL   DWORD
010C  0717 R                   C                DW      OFFSET  VIDEO_PARMS             ; PARMS
010E  C000                     C                DW      0C000H                          ; PARMS
0110  0000                     C                DW      0                               ; PAL SAVE AREA
0112  0000                     C                DW      0                               ; PAL SAVE AREA
0114  0000                     C                DW      0                               ; ALPHA TABLES
0116  0000                     C                DW      0                               ; ALPHA TABLES
0118  0000                     C                DW      0                               ; GRAPHICS TABLES
```

```
011A  0000                C          DW      0                    ; GRAPHICS TABLES
                          C
011C  0000                C          DW      0
011E  0000                C          DW      0
0120  0000                C          DW      0
0122  0000                C          DW      0
0124  0000                C          DW      0
0126  0000                C          DW      0
                          C
0128                      C  T5      LABEL   WORD
0128  0173 R              C          DW      OFFSET PST_0
012A  017E R              C          DW      OFFSET PST_1
012C  017E R              C          DW      OFFSET PST_2
012E  0189 R              C          DW      OFFSET PST_3
0130  0194 R              C          DW      OFFSET PST_4
0132  01A8 R              C          DW      OFFSET PST_5
0134  01BC R              C          DW      OFFSET PST_6
0136  01C7 R              C          DW      OFFSET PST_7
                          C
0138  01C7 R              C          DW      OFFSET PST_8
013A  01D2 R              C          DW      OFFSET PST_9
013C  01DD R              C          DW      OFFSET PST_A
013E  01F1 R              C          DW      OFFSET PST_B
0140  0204 R              C          DW      OFFSET PST_OUT
0142  0204 R              C          DW      OFFSET PST_OUT
0144  0204 R              C          DW      OFFSET PST_OUT
0146  0204 R              C          DW      OFFSET PST_OUT
                          C
0148                      C  ENV_X   PROC    NEAR                 ; SET 40X25 COLOR ALPHA
0148  80 26 0410 R CF     C          AND     EQUIP_LOW,0CFH
014D  80 0E 0410 R 10     C          OR      EQUIP_LOW,010H
0152  B8 0001             C          MOV     AX,1H
0155  CD 10               C          INT     10H
0157  C3                  C          RET
0158                      C  ENV_X   ENDP
                          C
0158                      C  ENV_0   PROC    NEAR                 ; SET 80X25 COLOR ALPHA
0158  80 26 0410 R CF     C          AND     EQUIP_LOW,0CFH
015D  80 0E 0410 R 20     C          OR      EQUIP_LOW,020H
0162  B8 0003             C          MOV     AX,03H
0165  CD 10               C          INT     10H
0167  C3                  C          RET
0168                      C  ENV_0   ENDP
                          C
0168                      C  ENV_3   PROC    NEAR                 ; SET MONOCHROME ALPHA
0168  80 0E 0410 R 30     C          OR      EQUIP_LOW,030H
016D  B8 0007             C          MOV     AX,07H
0170  CD 10               C          INT     10H
0172  C3                  C          RET
0173                      C  ENV_3   ENDP
                          C
                          C
0173                      C  PST_0:
0173  20 26 0487 R        C          AND     INFO,AH
0177  E8 0148 R           C          CALL    ENV_X
017A  E8 0168 R           C          CALL    ENV_3
017D  C3                  C          RET
017E                      C  PST_1:
017E                      C  PST_2:
017E  20 26 0487 R        C          AND     INFO,AH
0182  E8 0158 R           C          CALL    ENV_0
0185  E8 0168 R           C          CALL    ENV_3
0188  C3                  C          RET
0189                      C  PST_3:
0189  20 26 0487 R        C          AND     INFO,AH
018D  E8 0158 R           C          CALL    ENV_0
0190  E8 0168 R           C          CALL    ENV_3
0193  C3                  C          RET
0194                      C  PST_4:
0194  B6 03               C          MOV     DH,3
0196  B2 C2               C          MOV     DL,MISC_OUTPUT
0198  B0 00               C          MOV     AL,0
019A  EE                  C          OUT     DX,AL
019B  F6 D4               C          NOT     AH
019D  08 26 0487 R        C          OR      INFO,AH
01A1  E8 0168 R           C          CALL    ENV_3
01A4  E8 0148 R           C          CALL    ENV_X
01A7  C3                  C          RET
01A8                      C  PST_5:
01A8  B6 03               C          MOV     DH,3
01AA  B2 C2               C          MOV     DL,MISC_OUTPUT
01AC  B0 00               C          MOV     AL,0
01AE  EE                  C          OUT     DX,AL
01AF  F6 D4               C          NOT     AH
01B1  08 26 0487 R        C          OR      INFO,AH
01B5  E8 0168 R           C          CALL    ENV_3
01B8  E8 0158 R           C          CALL    ENV_0
01BB  C3                  C          RET
01BC                      C  PST_6:
01BC  20 26 0487 R        C          AND     INFO,AH
01C0  E8 0168 R           C          CALL    ENV_3
01C3  E8 0148 R           C          CALL    ENV_X
01C6  C3                  C          RET
01C7                      C  PST_7:
01C7                      C  PST_8:
01C7  20 26 0487 R        C          AND     INFO,AH
01CB  E8 0168 R           C          CALL    ENV_3
01CE  E8 0158 R           C          CALL    ENV_0
01D1  C3                  C          RET
01D2                      C  PST_9:
01D2  20 26 0487 R        C          AND     INFO,AH
01D6  E8 0168 R           C          CALL    ENV_3
01D9  E8 0158 R           C          CALL    ENV_0
01DC  C3                  C          RET
01DD                      C  PST_A:
01DD  B6 03               C          MOV     DH,3
01DF  B2 C2               C          MOV     DL,MISC_OUTPUT
01E1  B0 00               C          MOV     AL,0
01E3  EE                  C          OUT     DX,AL
01E4  F6 D4               C          NOT     AH
01E6  08 26 0487 R        C          OR      INFO,AH
01EA  E8 0148 R           C          CALL    ENV_X
01ED  E8 0168 R           C          CALL    ENV_3
01F0  C3                  C          RET
01F1                      C  PST_B:
01F1  B6 03               C          MOV     DH,3
01F3  B2 C2               C          MOV     DL,MISC_OUTPUT
01F5  B0 00               C          MOV     AL,0
01F7  EE                  C          OUT     DX,AL
01F8  F6 D4               C          NOT     AH
01FA  08 26 0487 R        C          OR      INFO,AH
01FE  E8 0158 R           C          CALL    ENV_0
0201  E8 0168 R           C          CALL    ENV_3
0204                      C  PST_OUT:
0204  C3                  C          RET
0205                      C  MK_ENV  ENDP
```

```
                      C
                      C ;-------------------------------------------------------------------------
                      C ; THIS ROUTINE TESTS THE CRT CARD INTERNAL DATA BUS AND IN A LIMITED     :
                      C ; WAY TESTS THE CRTC VIDEO CHIP BY WRITING/READING FROM CURSOR REGISTER :
                      C ; CARRY IS SET IF AN ERROR IS FOUND                                      :
                      C ;                                                                        :
                      C ; REGISTERS BX,SI,ES,DS ARE PRESERVED.                                   :
                      C ; REGISTERS AX,CX,DX ARE MODIFYED.                                       :
                      C ;-------------------------------------------------------------------------
0205                  C CD_PRESENCE_TST PROC    NEAR
0205  53              C         PUSH    BX                      ; SAVE BX
0206  BB 007F         C         MOV     BX,07FH                 ; INITIAL WORD PATTERN BYTE
0209  8B FB           C         MOV     DI,BX
020B  50              C         PUSH    AX                      ; SAVE PORT ADDRESS
020C  E8 022C R       C         CALL    RD_CURSOR               ;
020F  8B F0           C         MOV     SI,AX                   ; SAVE ORIGINAL VALUE
0211  58              C         POP     AX                      ; RECOVER PORT ADDRESS
0212  50              C         PUSH    AX                      ; SAVE PORT ADDRESS
0213  E8 0236 R       C         CALL    WR_CURSOR               ; WRITE CURSOR
0216  58              C         POP     AX                      ; RECOVER PORT ADDRESS
0217  50              C         PUSH    AX                      ; SAVE PORT ADDRESS
0218  E8 022C R       C         CALL    RD_CURSOR               ; READ IT BACK
021B  3B C7           C         CMP     AX,DI                   ; SAME?
021D  58              C         POP     AX
021E  75 03           C         JNZ     NOT_PRESENT             ; EXIT IF NOT EQUAL
0220  EB 05 90        C         JMP     TST_EX
0223                  C NOT_PRESENT:
0223  33 C0           C         XOR     AX,AX                   ; SET NOT PRESENT
0225  5B              C         POP     BX
0226  C3              C         RET
0227                  C TST_EX:
0227  B8 0001         C         MOV     AX,1                    ; SET PRESENT ON EXIT
022A  5B              C         POP     BX                      ; RESTORE BX
022B  C3              C         RET
022C                  C CD_PRESENCE_TST ENDP
                      C
                      C ;-------------------------------------------------------------------------
                      C ; MODULE NAME  RD_CURSOR                                                 :
                      C ;  READ CURSOR POSITION [ADDRESS] (FROM CRTC) TO  AX                     :
                      C ;                                                                        :
                      C ; REGISTER AX IS MODIFIED.                                               :
                      C ;-------------------------------------------------------------------------
022C                  C RD_CURSOR       PROC NEAR
022C  52              C         PUSH    DX                      ; SAVE REGS USED
022D  8B D0           C         MOV     DX,AX
022F  B0 0E           C         MOV     AL,C_CRSR_LOC_HGH
0231  EE              C         OUT     DX,AL
0232  42              C         INC     DX
0233  EC              C         IN      AL,DX
                      C                                         ; RETURN WITH CURSOR POS IN AX
                      C                                         ; RESTORE REGS USED
0234  5A              C         POP     DX
0235  C3              C         RET
0236                  C RD_CURSOR       ENDP
                      C
                      C ;-------------------------------------------------------------------------
                      C ; MODULE NAME  WR_CURSOR                                                 :
                      C ;  WRITE CURSOR POSITION [ADDRESS] (TO CRTC) WITH CONTENTS OF AX         :
                      C ;                                                                        :
                      C ; ALL REGISTERS PRESERVED                                                :
                      C ;------------------------------------------------------------------------:
0236                  C WR_CURSOR       PROC NEAR
                      C                                         ; SAVE REGS USED
0236  50              C         PUSH    AX
0237  52              C         PUSH    DX
0238  8B D0           C         MOV     DX,AX
023A  B4 0E           C         MOV     AH,C_CRSR_LOC_HGH       ; CURSOR LOCATION HIGH INDEX
023C  B0 7F           C         MOV     AL,07FH                 ; TEST VALUE
023E  E8 0D15 R       C         CALL    OUT_DX
                      C                                         ; RETURN WITH CURSOR POS IN AX
                      C                                         ; RESTORE REGS USED
0241  5A              C         POP     DX
0242  58              C         POP     AX
0243  C3              C         RET
0244                  C WR_CURSOR       ENDP
                      C
0244                  C POST:
                      C ;-----------------------------------------------------
                      C ;       INITIALIZE AND START CRT CONTROLLER (6845)    :
                      C ;       ON COLOR GRAPHICS AND MONOCHROME CARDS        :
                      C ;       TEST VIDEO READ/WRITE STORAGE.                :
                      C ; DESCRIPTION                                         :
                      C ;       RESET THE VIDEO ENABLE SIGNAL.                :
                      C ;       SELECT ALPHANUMERIC MODE, 40 * 25, B & W.     :
                      C ;       READ/WRITE DATA PATTERNS TO STG. CHECK STG    :
                      C ;       ADDRESSABILITY.                               :
                      C ;-----------------------------------------------------
                      C         ASSUME  DS:ABS0,ES:ABS0
0244  E8 0CFE R       C         CALL    DDS
0247  F6 06 0487 R 02 C         TEST    INFO,2
024C  75 12           C         JNZ     COLOR_PRESENCE_TST
024E  B8 03B4         C         MOV     AX,03B4H
0251  E8 0205 R       C         CALL    CD_PRESENCE_TST
0254  3D 0001         C         CMP     AX,1
0257  74 03           C         JE      CONT1
0259  E9 0317 R       C         JMP     POD14
025C                  C CONT1:
025C  B4 30           C         MOV     AH,30H                  ; MONOCHROME CARD INSTALLED
025E  EB 10           C         JMP     SHORT   OVER
0260                  C COLOR_PRESENCE_TST:
0260  B8 03D4         C         MOV     AX,03D4H
0263  E8 0205 R       C         CALL    CD_PRESENCE_TST
0266  3D 0001         C         CMP     AX,1
0269  74 03           C         JE      CONT2
026B  E9 0317 R       C         JMP     POD14
026E                  C CONT2:
026E  B4 20           C         MOV     AH,20H                  ; COLOR GRAPHICS CARD INSTALLED
0270                  C OVER:
0270  50              C         PUSH    AX                      ; RESAVE VALUE
0271  BB B000         C         MOV     BX,0B000H               ; BEG VIDEO RAM ADDR B/W CD
0274  BA 03B8         C         MOV     DX,3B8H                 ; MODE CONTROL B/W
0277  B9 1000         C         MOV     CX,4096                 ; RAM BYTE CNT FOR B/W CD
027A  B0 01           C         MOV     AL,1                    ; SET MODE FOR BW CARD
027C  80 FC 30        C         CMP     AH,30H                  ; B/W VIDEO CARD ATTACHED?
027F  74 08           C         JE      E9                      ; YES - GO TEST VIDEO STG
0281  B7 B8           C         MOV     BH,0B8H                 ; BEG VIDEO RAM ADDR COLOR CD
0283  B2 D8           C         MOV     DL,0D8H                 ; MODE CONTROL COLOR
0285  B5 40           C         MOV     CH,40H                  ; RAM BYTE CNT FOR COLOR CD
0287  FE C8           C         DEC     AL                      ; SET MODE TO 0 FOR COLOR CD
0289                  C E9:                                     ; TEST_VIDEO_STG:
0289  EE              C         OUT     DX,AL                   ; DISABLE VIDEO FOR COLOR CD
                      C
028A  8B 2E 0472 R    C         MOV     BP,DS:RESET_FLAG        ; POD INITIALIZED BY KBD RESET
                      C
028E  81 FD 1234      C         CMP     BP,1234H                ; POD INITIATED BY KBD RESET?
0292  8E C3           C         MOV     ES,BX                   ; POINT ES TO VIDEO RAM STG
```

```
0294  74 07          C          JE      E10                     ; YES - SKIP VIDEO RAM TEST
0296  8E DB          C          MOV     DS,BX                   ; POINT DS TO VIDEO RAM STG
                     C          ASSUME  DS:NOTHING,ES:NOTHING
0298  E8 02DF R      C          CALL    STGTST_CNT              ; GO TEST VIDEO R/W STG
029B  75 2E          C          JNE     E17                     ; R/W STG FAILURE - BEEP SPK
                     C  ;----------------------------------------------------------------
                     C  ;       SETUP VIDEO DATA ON SCREEN FOR VIDEO LINE TEST.          :
                     C  ; DESCRIPTION                                                    :
                     C  ;       ENABLE VIDEO SIGNAL AND SET MODE.                        :
                     C  ;       DISPLAY A HORIZONTAL BAR ON SCREEN.                      :
                     C  ;----------------------------------------------------------------
029D                 C  E10:
029D  58             C          POP     AX                      ; GET VIDEO SENSE SWS (AH)
029E  50             C          PUSH    AX                      ; SAVE IT
029F  B8 7020        C          MOV     AX,7020H                ; WRT BLANKS IN REVERSE VIDEO
02A2  2B FF          C          SUB     DI,DI                   ; SETUP STARTING LOC
02A4  B9 0028        C          MOV     CX,40                   ; NO. OF BLANKS TO DISPLAY
02A7  F3/ AB         C          REP     STOSW                   ; WRITE VIDEO STORAGE
                     C  ;-----------------------------------------------------
                     C  ;       CRT INTERFACE LINES TEST                      :
                     C  ; DESCRIPTION                                         :
                     C  ;       SENSE ON/OFF TRANSITION OF THE VIDEO ENABLE   :
                     C  ;       AND HORIZONTAL SYNC LINES.                    :
                     C  ;-----------------------------------------------------
02A9  58             C          POP     AX                      ; GET VIDEO SENSE SW INFO
02AA  50             C          PUSH    AX                      ; SAVE IT
02AB  80 FC 30       C          CMP     AH,30H                  ; B/W CARD ATTACHED?
02AE  BA 03BA        C          MOV     DX,03BAH                ; SETUP ADDR OF BW STATUS PORT
02B1  74 02          C          JE      E11                     ; YES - GO TEST LINES
02B3  B2 DA          C          MOV     DL,0DAH                 ; COLOR CARD IS ATTACHED
02B5                 C  E11:                                    ; LINE_TST:
02B5  B4 08          C          MOV     AH,8
02B7                 C  E12:                                    ; OFLOOP_CNT:
02B7  2B C9          C          SUB     CX,CX
02B9                 C  E13:
02B9  EC             C          IN      AL,DX                   ; READ CRT STATUS PORT
02BA  22 C4          C          AND     AL,AH                   ; CHECK VIDEO/HORZ LINE
02BC  75 04          C          JNZ     E14                     ; ITS ON - CHECK IF IT GOES OFF
02BE  E2 F9          C          LOOP    E13                     ; LOOP TILL ON OR TIMEOUT
02C0  EB 09          C          JMP     SHORT E17               ; GO PRINT ERROR MSG
02C2                 C  E14:
02C2  2B C9          C          SUB     CX,CX
02C4                 C  E15:
02C4  EC             C          IN      AL,DX                   ; READ CRT STATUS PORT
02C5  22 C4          C          AND     AL,AH                   ; CHECK VIDEO/HORZ LINE
02C7  74 0A          C          JZ      E16                     ; ITS ON - CHECK NEXT LINE
02C9  E2 F9          C          LOOP    E15                     ; LOOP IF OFF TILL IT GOES ON
02CB                 C  E17:                                    ;  CRT_ERR
02CB  BA 0102        C          MOV     DX,102H
02CE  E8 06C8 R      C          CALL    ERR_BEEP                ; GO BEEP SPEAKER
02D1  EB 06          C          JMP     SHORT E18
02D3                 C  E16:                                    ; NXT_LINE
02D3  B1 03          C          MOV     CL,3                    ; GET NEXT BIT TO CHECK
02D5  D2 EC          C          SHR     AH,CL
02D7  75 DE          C          JNZ     E12                     ; GO CHECK HORIZONTAL LINE
02D9                 C  E18:                                    ;  DISPLAY_CURSOR:
02D9  58             C          POP     AX                      ; GET VIDEO SENSE SWS (AH)
02DA  EB 3B          C          JMP     SHORT   POD14
                     C
                     C  ;-----------------------------------------------------------------------
                     C  ;       THIS SUBROUTINE PERFORMS A READ/WRITE STORAGE TEST ON           :
                     C  ;       A 16K BLOCK OF STORAGE.                                         :
                     C  ; ENTRY REQUIREMENTS:                                                   :
                     C  ;       ES = ADDRESS OF STORAGE SEGMENT BEING TESTED                    :
                     C  ;       DS = ADDRESS OF STORAGE SEGMENT BEING TESTED                    :
                     C  ;       WHEN ENTERING AT STGTST_CNT, CX MUST BE LOADED WITH             :
                     C  ;       THE BYTE COUNT.                                                 :
                     C  ; EXIT PARAMETERS:                                                     :
                     C  ;       ZERO FLAG = 0 IF STORAGE ERROR (DATA COMPARE OR PARITY CHECK.   :
                     C  ;               AL = 0 DENOTES A PARITY CHECK. ELSE AL=XOR'ED BIT       :
                     C  ;                  PATTERN OF THE EXPECTED DATA PATTERN VS THE          :
                     C  ;                  ACTUAL DATA READ.                                    :
                     C  ;       AX,BX,CX,DX,DI, AND SI ARE ALL DESTROYED.                       :
                     C  ;-----------------------------------------------------------------------
02DC                 C  STGTST  PROC    NEAR
02DC  B9 4000        C          MOV     CX,4000H                ; SETUP CNT TO TEST A 16K BLK
02DF                 C  STGTST_CNT:
02DF  FC             C          CLD                             ; SET DIR FLAG TO INCREMENT
02E0  8B D9          C          MOV     BX,CX                   ; SAVE CNT (4K FOR VIDEO OR 16K)
02E2  B8 AAAA        C          MOV     AX,0AAAAH               ; GET DATA PATTERN TO WRITE
02E5  BA FF55        C          MOV     DX,0FF55H               ; SETUP OTHER DATA PATTERNS TO USE
02E8  2B FF          C          SUB     DI,DI                   ; DI = OFFSET 0 RELATIVE TO ES REG
02EA  F3/ AA         C          REP     STOSB                   ; WRITE STORAGE LOCATIONS
02EC                 C  C3:                                     ;  STG01
02EC  4F             C          DEC     DI                      ; POINT TO LAST BYTE JUST WRITTEN
02ED  FD             C          STD                             ; SET DIR FLAG TO GO BACKWARDS
02EE                 C  C4:
02EE  8B F7          C          MOV     SI,DI
02F0  8B CB          C          MOV     CX,BX                   ; SETUP BYTE CNT
02F2                 C  C5:                                     ; INNER TEST LOOP
02F2  AC             C          LODSB                           ; READ OLD TEST BYTE [SI]+
02F3  32 C4          C          XOR     AL,AH                   ; DATA READ AS EXPECTED ?
02F5  75 1E          C          JNE     C7                      ;  NO - GO TO ERROR ROUTINE
02F7  8A C2          C          MOV     AL,DL                   ; GET NEXT DATA PATTERN TO WRITE
02F9  AA             C          STOSB                           ; WRITE INTO LOCATION JUST READ
02FA  E2 F6          C          LOOP    C5                      ;  DECREMENT COUNT AND LOOP CX
                     C
02FC  22 E4          C          AND     AH,AH                   ; ENDING 0 PATTERN WRITTEN TO STG?
02FE  74 13          C          JZ      C6X                     ; YES - RETURN TO CALLER WITH AL=0
0300  8A E0          C          MOV     AH,AL                   ; SETUP NEW VALUE FOR COMPARE
0302  86 F2          C          XCHG    DH,DL                   ; MOVE NEXT DATA PATTERN TO DL
0304  22 E4          C          AND     AH,AH                   ; READING ZERO PATTERN THIS PASS ?
0306  75 04          C          JNZ     C6                      ; CONTINUE TEST SEQUENCE TILL 0
0308  8A D4          C          MOV     DL,AH                   ;  ELSE SET 0 FOR END READ PATTERN
030A  EB E0          C          JMP     C3                      ;  AND MAKE FINAL BACKWARDS PASS
030C                 C  C6:
030C  FC             C          CLD                             ; SET DIR FLAG TO GO FORWARD
030D  47             C          INC     DI                      ; SET POINTER TO BEG LOCATION
030E  74 DE          C          JZ      C4                      ; READ/WRITE FORWARD IN STG
0310  4F             C          DEC     DI                      ; ADJUST POINTER
0311  EB D9          C          JMP     C3                      ; READ/WRITE BACKWARD IN STG
0313                 C  C6X:
0313  B0 00          C          MOV     AL,000H                 ; AL=0 DATA COMPARE OK
0315                 C  C7:
0315  FC             C          CLD                             ; SET DIRECTION FLAG BACK TO INC
0316  C3             C          RET
0317                 C  STGTST  ENDP
                     C
                     C  ;-----------------------------------------------------
                     C  ;                 EGA CRT ATTACHMENT TEST             :
                     C  ;                                                     :
                     C  ; 1. INIT CRT TO 40X25 - BW  ****SET TO MODE****      :
                     C  ; 2. CHECK FOR VERTICAL AND VIDEO ENABLES, AND CHECK  :
                     C  ;    TIMING OF SAME                                   :
                     C  ; 3. CHECK VERTICAL INTERRUPT                         :
                     C  ; 4. CHECK RED, BLUE, GREEN, AND INTENSIFY DOTS       :
```

```
                     C  ; 5. INIT TO 40X25 - COLOR/MONO ****SET TO MODE****       :
                     C  ;-----------------------------------------------------------
                     C
                     C  ;----- NOMINAL TIME IS B286H FOR 60 HZ.
                     C  ;----- NOMINAL TIME IS A2FEH FOR 50 HZ.
                     C
 = A0AC              C  MAX_VERT_COLOR  EQU     0A0ACH                       ; MAX TIME FOR VERT/VERT
                     C                                                       ;  (NOMINAL + 10%)
 = C460              C  MIN_VERT_COLOR  EQU     0C460H                       ; MIN TIME FOR VERT/VERT
                     C                                                       ;  (NOMINAL - 10%)
 = 00C8              C  CENAB_PER_FRAME EQU     200                          ; NUM OF ENABLES PER FRAME
 = 8D99              C  MAX_VERT_MONO   EQU     08D99H                       ; MAX TIME FOR VERT/VERT
                     C                                                       ;  (NOMINAL + 10%)
 = B862              C  MIN_VERT_MONO   EQU     0B862H                       ; MIN TIME FOR VERT/VERT
                     C                                                       ;  (NOMINAL - 10%)
 = 015E              C  EENAB_PER_FRAME EQU     350                          ; ENHANCED ENABLES PER FRAME
 = 015E              C  MENAB_PER_FRAME EQU     350                          ; NUM OF ENABLES PER FRAME
                     C
 = 0043              C  TIM_CTL         EQU     043H                         ; 8253 TIMER CONTROL PORT
 = 0040              C  TIMERO          EQU     040H                         ; 8253 TIMER/CNTER 0 PORT
                     C
 0317                C  POD14   PROC    NEAR
 0317  83 EC 0A      C          SUB     SP,0AH                               ; RESERVE 5 WORDS ON STACK
 031A  8B EC         C          MOV     BP,SP                                ; INIT SCRATCH PAD POINTER
                     C
                     C          ASSUME  DS:ABS0,ES:ABS0
 031C  E8 0CFE R     C          CALL    DDS
 031F  B0 30         C          MOV     AL,00110000B                         ; SET TIMER 0 TO MODE 0
                     C
 0321  E6 43         C          OUT     TIM_CTL,AL
 0323  B0 00         C          MOV     AL,00H
 0325  E6 40         C          OUT     TIMERO,AL                            ; SEND FIRST BYTE TO TIMER
 0327  F6 06 0487 R 02 C        TEST    INFO,2
 032C  74 1F         C          JZ      COLOR_EGA_V
 032E  E8 0168 R     C          CALL    ENV_3                                ; SET UP IN MONOCHROME
 0331  C7 46 02 015E C          MOV     WORD PTR[BP][2],MENAB_PER_FRAME ; NUM.OF FRAMES FOR MONO
 0336  C7 46 04 8D99 C          MOV     WORD PTR[BP][4],MAX_VERT_MONO   ; MAX TIME FOR VERT/VERT
 033B  C7 46 06 B862 C          MOV     WORD PTR[BP][6],MIN_VERT_MONO   ; MIN TIME FOR VERT/VERT
 0340  B2 B4         C          MOV     DL,CRTC_ADDR_B                       ; MONO CRTC REG
 0342  B4 01         C          MOV     AH,C_HRZ_DSP                         ; HORIZ. TOTAL DIPLAY
 0344  B0 27         C          MOV     AL,27H                               ;  TO 40 COL
 0346  E8 0D15 R     C          CALL    OUT_DX                               ;
 0349  B2 BA         C          MOV     DL,INPUT_STATUS_B                    ; 3BA
 034B  EB 2A         C          JMP     SHORT   COMMON
 034D                C  COLOR_EGA_V:
 034D  E8 0148 R     C          CALL    ENV_X                                ; SET UP IN 40X25 COLOR
 0350  E8 0E9A R     C          CALL    BRST_DET                             ; ENHANCED MODE
 0353  73 11         C          JNC     COLOR_V                              ; NO,40X25
 0355  B2 D4         C          MOV     DL,CRTC_ADDR                         ; BRST MODE ONLY!
 0357  B4 01         C          MOV     AH,1                                 ; HRZ DSP END
 0359  B0 14         C          MOV     AL,20                                ; MODIFY FOR TEST ONLY
 035B  E8 0D15 R     C          CALL    OUT_DX
 035E  C7 46 02 015E C          MOV     WORD PTR[BP][2],EENAB_PER_FRAME ; NUM.OF FRAMES FOR COLOR
 0363  EB 06 90      C          JMP     BRST_COLOR_V
 0366                C  COLOR_V:
                     C
 0366  C7 46 02 00C8 C          MOV     WORD PTR[BP][2],CENAB_PER_FRAME ; NUM.OF FRAMES FOR COLOR
 036B                C  BRST_COLOR_V:
 036B  C7 46 04 A0AC C          MOV     WORD PTR[BP][4],MAX_VERT_COLOR  ; MAX TIME FOR VERT/VERT
 0370  C7 46 06 C460 C          MOV     WORD PTR[BP][6],MIN_VERT_COLOR  ; MIN TIME FOR VERT/VERT
 0375  B2 DA         C          MOV     DL,INPUT_STATUS                      ; SET ADDRESSING TO VIDEO
                     C                                                       ;  ATTR STATUS
 0377                C  COMMON:
 0377  B8 0500       C          MOV     AX,0500H                             ; SET TO VIDEO PAGE 0
 037A  CD 10         C          INT     10H
 037C  2B C9         C          SUB     CX,CX
                     C
                     C  ;----- LOOK FOR VERTICAL
                     C
 037E                C  POD14_1:
 037E  EC            C          IN      AL,DX                                ; GET STATUS
 037F  A8 08         C          TEST    AL,00001000B                         ; VERTICAL THERE YET?
 0381  75 07         C          JNE     POD14_2                              ; CONTINUE IF IT IS
 0383  E2 F9         C          LOOP    POD14_1                              ; KEEP LOOKING TILL COUNT
 0385  B3 00         C          MOV     BL,00                                ;  EXHAUSTED
 0387  E9 0448 R     C          JMP     POD14_ERR                            ; NO VERTICAL
                     C
                     C  ;----- GOT VERTICAL - START TIMER
                     C
 038A                C  POD14_2:
 038A  B0 00         C          MOV     AL,0
 038C  E6 40         C          OUT     TIMERO,AL                            ; SEND 2ND BYTE TO TIMER TO
                     C                                                       ; START IT
 038E  2B DB         C          SUB     BX,BX                                ; INIT. ENABLE COUNTER
                     C  ;----- WAIT FOR VERTICAL TO GO AWAY
 0390  33 C9         C          XOR     CX,CX
 0392                C  POD14_25:
 0392  EC            C          IN      AL,DX                                ; GET STATUS
 0393  A8 08         C          TEST    AL,00001000B                         ; VERTICAL STILL THERE
 0395  74 07         C          JZ      POD14_3                              ; CONTINUE IF IT'S GONE
 0397  E2 F9         C          LOOP    POD14_25                             ; KEEP LOOKING TILL COUNT
 0399  B3 01         C          MOV     BL,01H                               ;  EXHAUSTED
 039B  E9 0448 R     C          JMP     POD14_ERR                            ; VERTICAL STUCK ON
                     C
                     C  ;----- NOW START LOOKING FOR ENABLE TRANSITIONS
                     C
 039E                C  POD14_3:
 039E  2B C9         C          SUB     CX,CX
 03A0                C  POD14_4:
 03A0  EC            C          IN      AL,DX                                ; GET STATUS
 03A1  A8 01         C          TEST    AL,00000001B                         ; ENABLE ON YET?
 03A3  74 15         C          JE      POD14_5                              ; GO ON IF IT IS
 03A5  A8 08         C          TEST    AL,00001000B                         ; VERTICAL ON AGAIN?
 03A7  75 23         C          JNE     POD14_75                             ; CONTINUE IF IT IS
 03A9  E2 F5         C          LOOP    POD14_4                              ; KEEP LOOKING IF NOT
 03AB  B3 02         C          MOV     BL,02H
 03AD  E9 0448 R     C          JMP     POD14_ERR                            ; ENABLE STUCK OFF
 03B0                C  POD14_4A:
 03B0  B3 03         C          MOV     BL,03H
 03B2  E9 0448 R     C          JMP     POD14_ERR                            ; VERTICAL STUCK ON
 03B5                C  POD14_4B:
 03B5  B3 04         C          MOV     BL,04H
 03B7  E9 0448 R     C          JMP     POD14_ERR                            ; ENABLE STUCK ON
                     C
                     C  ;----- MAKE SURE VERTICAL WENT OFF WITH ENABLE GOING ON
                     C
 03BA                C  POD14_5:
 03BA  A8 08         C          TEST    AL,00001000B                         ; VERTICAL OFF?
 03BC  75 F2         C          JNZ      POD14_4A                            ; GO ON IF IT IS
                     C  ;----- NOW WAIT FOR ENABLE TO GO OFF                 ;  (ERROR IF NOT)
 03BE                C  POD14_6:
 03BE  EC            C          IN      AL,DX                                ; GET STATUS
 03BF  A8 01         C          TEST    AL,00000001B                         ; ENABLE OFF YET?
 03C1  E1 FB         C          LOOPE   POD14_6                              ; KEEP LOOKING IF NOT
 03C3  E3 F0         C          JCXZ    POD14_4B                             ;  YET LOW
                     C  ;----- ENABLE HAS TOGGLED, BUMP COUNTER AND TEST FOR NEXT VERTICAL
```

```
03C5                 C  POD14_7:
03C5  43             C          INC     BX                         ; BUMP ENABLE COUNTER
03C6  74 04          C          JZ      POD14_75                   ; IF COUNTER WRAPS,
                     C                                             ;  SOMETHING IS WRONG
03C8  A8 08          C          TEST    AL,00001000B               ; DID ENABLE GO LOW
                     C                                             ;  BECAUSE OF VERTICAL
03CA  74 D2          C          JZ      POD14_3                    ; IF NOT, LOOK FOR ANOTHER
                     C                                             ;  ENABLE TOGGLE
                     C  ;----- HAVE HAD COMPLETE VERTICAL-VERTICAL CYCLE, NOW TEST RESULTS
03CC                 C  POD14_75:
03CC  B0 00          C          MOV     AL,00                      ; LATCH TIMERO
03CE  E6 43          C          OUT     TIM_CTL,AL
03D0  3B 5E 02       C          CMP     BX,WORD PTR[BP][2]         ; NUMBER OF ENABLES BETWEEN
                     C                                             ;  VERTICALS O.K.?
03D3  74 04          C          JE      POD14_8
03D5  B3 05          C          MOV     BL,05H
03D7  EB 6F          C          JMP     SHORT POD14_ERR
03D9                 C  POD14_8:
03D9  E4 40          C          IN      AL,TIMERO                  ; GET TIMER VALUE LOW
03DB  8A E0          C          MOV     AH,AL                      ; SAVE IT
03DD  90             C          NOP
03DE  E4 40          C          IN      AL,TIMERO                  ; GET TIMER HIGH
03E0  86 E0          C          XCHG    AH,AL
03E2  90             C          NOP
03E3  90             C          NOP
03E4  3B 46 04       C          CMP     AX,WORD PTR[BP][4]         ; MAXIMUM VERTICAL TIMING
03E7  7D 04          C          JGE     POD14_9
03E9  B3 06          C          MOV     BL,06H
03EB  EB 5B          C          JMP     SHORT POD14_ERR
03ED                 C  POD14_9:
03ED  3B 46 06       C          CMP     AX,WORD PTR[BP][6]         ; MINIMUM VERTICAL TIMING
03F0  7E 04          C          JLE     POD14_10
03F2  B3 07          C          MOV     BL,07H
03F4  EB 52          C          JMP     SHORT   POD14_ERR
                     C
                     C  ;----- SEE IF RED, GREEN, BLUE AND INTENSIFY DOTS WORK
                     C
                     C  ;----- FIRST, SET A LINE OF REVERSE VIDEO, INTENSIFIED BLANKS INTO BUFFER
03F6                 C  POD14_10:
03F6  B8 09DB        C          MOV     AX,09DBH                   ; WRITE CHARS, BLANKS
03F9  BB 000F        C          MOV     BX,000FH                   ; PAGE 0, REVERSE VIDEO,
                     C                                             ;  HIGH INTENSITY
03FC  B9 0050        C          MOV     CX,80                      ; 80 CHARACTERS
03FF  CD 10          C          INT     10H
0401  EC             C          IN      AL,DX
0402  52             C          PUSH    DX                         ; SAVE INPUT STATUS
0403  B2 C0          C          MOV     DL,ATTR_WRITE              ; ATTRIBUTE ADDRESSS
0405  B4 0F          C          MOV     AH,0FH                     ; PALETTE REG 'F'
0407  B0 3F          C          MOV     AL,03FH                    ; TEST VALUE
0409  E8 0D15 R      C          CALL    OUT_DX                     ; VIDEO STATUS MUX
040C  B8 000F        C          MOV     AX,0FH                     ; START WITH BLUE DOTS
040F  5A             C          POP     DX
0410                 C  POD14_13:
0410  50             C          PUSH    AX                         ; SAVE
0411  52             C          PUSH    DX                         ; SAVE INPUT STATUS
0412  B2 C0          C          MOV     DL,ATTR_WRITE              ; ATTRIBUTE ADDRESSS
0414  B4 32          C          MOV     AH,32H                     ; COLOR PLANE ENABLE
0416  E8 0D15 R      C          CALL    OUT_DX                     ; VIDEO STATUS MUX
0419  5A             C          POP     DX                         ; RECOVER INPUT STATUS
041A  58             C          POP     AX
041B  2B C9          C          SUB     CX,CX
                     C  ;----- SEE IF DOT COMES ON
041D                 C  POD14_14:
041D  EC             C          IN      AL,DX                      ; GET STATUS
041E  A8 30          C          TEST    AL,00110000B               ; DOT THERE?
0420  75 09          C          JNZ     POD14_15                   ; LOOK FOR DOT TO TURN OFF
0422  E2 F9          C          LOOP    POD14_14                   ; CONTINUE TEST FOR DOT ON
0424  B3 10          C          MOV     BL,10H
0426  0A DC          C          OR      BL,AH                      ; OR IN DOT BEING TESTED
0428  EB 1E 90       C          JMP     POD14_ERR                  ; DOT NOT COMING ON
                     C  ;----- SEE IF DOT GOES OFF
042B                 C  POD14_15:
042B  2B C9          C          SUB     CX,CX
042D                 C  POD14_16:
042D  EC             C          IN      AL,DX                      ; GET STATUS
042E  A8 30          C          TEST    AL,00110000B               ; IS DOT STILL ON?
0430  74 08          C          JE      POD14_17                   ; GO ON IF DOT OFF
0432  E2 F9          C          LOOP    POD14_16                   ; ELSE, KEEP WAITING FOR
                     C                                             ;  DOT TO GO OFF
0434  B3 20          C          MOV     BL,20H
0436  0A DC          C          OR      BL,AH                      ; OR IN DOT BEING TESTED
0438  EB 0E          C          JMP     SHORT POD14_ERR
                     C
                     C  ;----- ADJUST TO POINT TO NEXT DOT
                     C
043A                 C  POD14_17:
043A  FE C4          C          INC     AH
043C  80 FC 30       C          CMP     AH,030H                    ; ALL 3 DOTS DONE?
043F  74 25          C          JE      POD14_18                   ; GO END
0441  80 CC 0F       C          OR      AH,0FH                     ; MAKE 0F,1F,2F
0444  8A C4          C          MOV     AL,AH
0446  EB C8          C          JMP     POD14_13                   ; GO LOOK FOR ANOTHER DOT
0448                 C  POD14_ERR:
0448  B9 0006        C          MOV     CX,6
044B  BA 0103        C          MOV     DX,0103H                   ; ONE LONG AND THREE SHORT
044E  E8 06C8 R      C          CALL    ERR_BEEP
0451  83 C4 0A       C          ADD     SP,0AH                     ; BALANCE STACK
0454  B0 36          C          MOV     AL,00110110B               ; RE-INIT TIMER 0
0456  E6 43          C          OUT     TIM_CTL,AL
0458  2A C0          C          SUB     AL,AL
045A  E6 40          C          OUT     TIMERO,AL
045C  90             C          NOP
045D  90             C          NOP
045E  E6 40          C          OUT     TIMERO,AL
0460  BD 0001        C          MOV     BP,1
0463  E9 0091 R      C          JMP     SKIP
                     C          ASSUME  DS:ABSO
0466                 C  POD14_18:
0466  E8 0CFE R      C          CALL    DDS
0469  B8 0500        C          MOV     AX,0500H                   ; SET TO VIDEO PAGE 0
046C  CD 10          C          INT     10H
046E  B0 36          C          MOV     AL,00110110B               ; RE-INIT TIMER 0
0470  E6 43          C          OUT     TIM_CTL,AL
0472  2A C0          C          SUB     AL,AL
0474  E6 40          C          OUT     TIMERO,AL
0476  90             C          NOP
0477  90             C          NOP
0478  E6 40          C          OUT     TIMERO,AL
047A  83 C4 0A       C          ADD     SP,0AH                     ; REMOVE SCRATCH PAD
047D  BD 0000        C          MOV     BP,0                       ; MAKE BP NON ZERO
0480                 C  POD14           ENDP
                     C
                     C  ;----- TEST STORAGE
                     C
0480                 C  MEM_TEST:
0480  1E             C          PUSH    DS
```

```
0481  E8 0CFE R            C            CALL    DDS
                           C            ASSUME DS:ABS0
0484  F6 06 0487 R 02      C            TEST    INFO,2
0489  74 12                C            JZ      D_COLOR_M
048B  80 0E 0410 R 30      C            OR      EQUIP_LOW,030H
0490  B8 000F              C            MOV     AX,0FH
0493  80 0E 0487 R 60      C            OR      INFO,060H
0498  B8 000F              C            MOV     AX,0FH
049B  EB 0D                C            JMP     SHORT D_OUT_M
049D                       C   D_COLOR_M:
049D  80 26 0410 R CF      C            AND     EQUIP_LOW,0CFH
04A2  80 0E 0410 R 20      C            OR      EQUIP_LOW,020H
04A7  B8 000E              C            MOV     AX,0EH                       ; INTERNAL COLOR MODE
04AA                       C   D_OUT_M:                                      ; TEST IN COLOR
04AA  CD 10                C            INT     10H
04AC  83 EC 06             C            SUB     SP,6                         ; RESERVE 3 WORDS ON STACK
04AF  8B EC                C            MOV     BP,SP                        ; SET BP
04B1  B8 A000              C            MOV     AX,0A000H                    ; PUT BUFFER ADDRESS IN AX
                           C            ASSUME  DS:NOTHING,ES:NOTHING
04B4  8E D8                C            MOV     DS,AX                        ; SET UP SEG REGS TO POINT
04B6  8E C0                C            MOV     ES,AX                        ;  TO BUFFER AREA
04B8  C7 46 02 0000        C            MOV     WORD PTR[BP][2],0            ; INITIALIZE
04BD  C7 46 04 0000        C            MOV     WORD PTR[BP][4],0            ; INITIALIZE
04C2  B6 03                C            MOV     DH,3
04C4  B2 C4                C            MOV     DL,SEQ_ADDR
04C6  B8 0201              C            MOV     AX,0201H
04C9  E8 0D15 R            C            CALL    OUT_DX
04CC  B2 CE                C            MOV     DL,GRAPH_ADDR                ; ADDRESS READ MAP SELECT
04CE  B8 0400              C            MOV     AX,0400H
04D1  E8 0D15 R            C            CALL    OUT_DX
04D4  52                   C            PUSH    DX
04D5  B2 DA                C            MOV     DL,ATTR_READ                 ; SET UP ATTRIBUTE
04D7  EC                   C            IN      AL,DX
04D8  B2 C0                C            MOV     DL,ATTR_WRITE                ; ATTRIBUTE WRITE ADDRESS
04DA  B8 3200              C            MOV     AX,3200H
04DD  E8 0D15 R            C            CALL    OUT_DX
04E0  E8 068F R            C            CALL    HOW_BIG                      ; GO FIND AMOUNT OF MEMORY
04E3  80 FC 00             C            CMP     AH,0
04E6  74 03                C            JZ      AA1
04E8  E9 05CD R            C            JMP     EGA_MEM_ERROR
04EB                       C   AA1:
04EB  E8 05D9 R            C            CALL    MEMORY_OK                    ; GO TEST IT
04EE  80 FC 00             C            CMP     AH,0
04F1  74 03                C            JZ      AA2
04F3  E9 05CD R            C            JMP     EGA_MEM_ERROR
04F6                       C   AA2:
04F6  5A                   C            POP     DX
04F7  B2 C4                C            MOV     DL,SEQ_ADDR
04F9  B8 0202              C            MOV     AX,0202H
04FC  E8 0D15 R            C            CALL    OUT_DX
04FF  B2 CE                C            MOV     DL,GRAPH_ADDR                ; ADDRESS OF READ MAP
0501  B8 0401              C            MOV     AX,0401H
0504  E8 0D15 R            C            CALL    OUT_DX
0507  52                   C            PUSH    DX
0508  B2 DA                C            MOV     DL,ATTR_READ                 ; SET UP ATTRIBUTE
050A  EC                   C            IN      AL,DX
050B  B2 C0                C            MOV     DL,ATTR_WRITE                ; ATTRIBUTE WRITE ADDRESS
050D  B8 3200              C            MOV     AX,3200H
0510  E8 0D15 R            C            CALL    OUT_DX
0513  C7 46 04 0000        C            MOV     WORD PTR [BP][4],0           ; INITIALIZE
0518  E8 068F R            C            CALL    HOW_BIG                      ; GO FIND AMOUNT OF MEMORY
051B  80 FC 00             C            CMP     AH,0
051E  74 03                C            JZ      AA3
0520  E9 05CD R            C            JMP     EGA_MEM_ERROR
0523                       C   AA3:
0523  E8 05D9 R            C            CALL    MEMORY_OK                    ; GO TEST IT
0526  80 FC 00             C            CMP     AH,0
0529  74 03                C            JZ      AA4
052B  E9 05CD R            C            JMP     EGA_MEM_ERROR
052E                       C   AA4:
052E  5A                   C            POP     DX
052F  B2 C4                C            MOV     DL,SEQ_ADDR
0531  B8 0204              C            MOV     AX,0204H
0534  E8 0D15 R            C            CALL    OUT_DX
0537  52                   C            PUSH    DX
0538  B2 CE                C            MOV     DL,GRAPH_ADDR                ; ADDRESS OF READ MAP
053A  B8 0402              C            MOV     AX,0402H
053D  E8 0D15 R            C            CALL    OUT_DX
0540  B2 DA                C            MOV     DL,ATTR_READ                 ; SET UP ATTRIBUTE
0542  EC                   C            IN      AL,DX
0543  B2 C0                C            MOV     DL,ATTR_WRITE                ; ATTRIBUTE WRITE ADDRESS
0545  B8 3200              C            MOV     AX,3200H
0548  E8 0D15 R            C            CALL    OUT_DX
054B  C7 46 04 0000        C            MOV     WORD PTR[BP][4],0            ; INITIALIZE
0550  E8 068F R            C            CALL    HOW_BIG                      ; GO FIND AMOUNT OF MEMORY
0553  80 FC 00             C            CMP     AH,0
0556  74 03                C            JZ      AA5
0558  EB 73 90             C            JMP     EGA_MEM_ERROR
055B                       C   AA5:
055B  E8 05D9 R            C            CALL    MEMORY_OK                    ; GO TEST IT
055E  80 FC 00             C            CMP     AH,0
0561  74 03                C            JZ      AA6
0563  EB 68 90             C            JMP     EGA_MEM_ERROR
0566                       C   AA6:
0566  5A                   C            POP     DX
0567  B2 C4                C            MOV     DL,SEQ_ADDR
0569  B8 0208              C            MOV     AX,0208H
056C  E8 0D15 R            C            CALL    OUT_DX
056F  B2 CE                C            MOV     DL,GRAPH_ADDR                ; ADDRESS OF READ MAP
0571  B8 0403              C            MOV     AX,0403H
0574  E8 0D15 R            C            CALL    OUT_DX
0577  52                   C            PUSH    DX
0578  B2 DA                C            MOV     DL,ATTR_READ                 ; SET UP ATTRIBUTE
057A  EC                   C            IN      AL,DX
057B  B2 C0                C            MOV     DL,ATTR_WRITE                ; ATTRIBUTE WRITE ADDRESS
057D  B8 3200              C            MOV     AX,3200H
0580  E8 0D15 R            C            CALL    OUT_DX
0583  C7 46 04 0000        C            MOV     WORD PTR[BP][4],0            ; INITIALIZE
0588  E8 068F R            C            CALL    HOW_BIG                      ; GO FIND AMOUNT OF MEMORY
058B  80 FC 00             C            CMP     AH,0
058E  75 3D                C            JNZ     EGA_MEM_ERROR
0590  E8 05D9 R            C            CALL    MEMORY_OK                    ; GO TEST IT
0593  80 FC 00             C            CMP     AH,0
0596  75 35                C            JNZ     EGA_MEM_ERROR
0598  55                   C            PUSH    BP                           ; SAVE SCRATCH PAD POINTER
0599  BD 0000              C            MOV     BP,0                         ; RESET BP FOR XT
059C                       C   EGA_MEM_EXIT:
059C  5E                   C            POP     SI                           ; RESTORE
059D  5A                   C            POP     DX
059E  E8 0CFE R            C            CALL    DDS                          ; SET DATA SEGMENT
                           C            ASSUME  DS:ABS0
05A1  36: 8B 5C 02         C            MOV     BX,WORD PTR SS:[SI][2]       ; GET EGA MEMORY SIZE
05A5  B1 06                C            MOV     CL,06H                       ; DIVIDE BY 64 TO GET
05A7  D3 EB                C            SHR     BX,CL                        ;  NUMBER OF 64KB BLOCKS
05A9  4B                   C            DEC     BX
05AA  B1 05                C            MOV     CL,05H
```

```
05AC  D3 E3                C          SHL     BX,CL
05AE  80 E3 60             C          AND     BL,01100000B                 ; ISOLATE BITS 5 AND 6
                           C
05B1  80 26 0487 R 9F      C          AND     INFO,10011111B
                           C
05B6  08 1E 0487 R         C          OR      INFO,BL
                           C
05BA  80 0E 0487 R 04      C          OR      INFO,00000100B               ; 04H  SET 3XX ACTIVE
05BF  8A 1E 0488 R         C          MOV     BL,INFO_3
05C3  E8 00F3 R            C          CALL    MK_ENV
05C6  83 C4 06             C          ADD     SP,6                         ; RESTORE STACK
05C9  1F                   C          POP     DS
05CA  E9 0091 R            C          JMP     SKIP                         ; GO TO END
05CD                       C  EGA_MEM_ERROR:
05CD  BA 0103              C          MOV     DX,0103H                     ; ONE LONG AND THREE SHORT
05D0  E8 06C8 R            C          CALL    ERR_BEEP
05D3  55                   C          PUSH    BP                           ; SAVE SCRATCH PAD POINTER
05D4  BD 0001              C          MOV     BP,1                         ; INDICATE ERROR FOR XT
05D7  EB C3                C          JMP     EGA_MEM_EXIT
                           C
                           C  ;----- THIS ROUTINE FINDS AMOUNT OF MEMORY GOOD
                           C
05D9                       C  MEMORY_OK       PROC    NEAR
05D9  BB A000              C          MOV     BX,0A000H                    ; SET PTR. TO BUFFER SEG
05DC  8E DB                C          MOV     DS,BX                        ; SET SEG.REG.
05DE  8E C3                C          MOV     ES,BX
05E0  8B 46 04             C          MOV     AX,WORD PTR[BP][4]           ; SET COUNT FOR 32K WORDS
05E3  8A E8                C          MOV     CH,AL                        ; SET AMOUNT OF BUFFER
05E5  2A C9                C          SUB     CL,CL                        ;  TO BE TESTED
05E7  D1 E1                C          SHL     CX,1                         ; MULTIPLY BY TWO
05E9  E8 05FB R            C          CALL    PODSTG
05EC  80 FC 00             C          CMP     AH,0                         ; TEST FOR ERROR
05EF  75 09                C          JNZ     MEMORY_OK_ERR                ; IF ERROR GO PRINT IT
05F1                       C  MEMORY_OK_EX:
05F1  8B 46 04             C          MOV     AX,WORD PTR[BP][4]           ; AMOUNT OF MEMORY FOUND
05F4  01 46 02             C          ADD     WORD PTR[BP][2],AX           ; AMOUNT OF MEMORY GOOD
05F7  B8 0000              C          MOV     AX,0
05FA                       C  MEMORY_OK_ERR:
05FA  C3                   C          RET
05FB                       C  MEMORY_OK       ENDP
                           C
                           C  ;-------------------------------------------------------------
                           C  ; THIS ROUTINE PERFORMS A READ/WRITE TEST ON A BLOCK OF STORAGE :
                           C  ; (MAX. SIZE = 32KW). IF "WARM START", FILL BLOCK WITH 0000 AND :
                           C  ; RETURN.                                                       :
                           C  ; ON ENTRY:                                                     :
                           C  ;   ES = ADDRESS OF STORAGE TO BE TESTED                        :
                           C  ;   DS = ADDRESS OF STORAGE TO BE TESTED                        :
                           C  ;   CX = WORD COUNT OF STORAGE BLOCK TO BE TESTED               :
                           C  ;        (MAX. = 8000H (32K WORDS))                             :
                           C  ; ON EXIT:                                                      :
                           C  ;   ZERO FLAG = OFF IF STORAGE ERROR                            :
                           C  ; AX,BX,CX,DX,DI,SI ARE ALL DESTROYED.                          :
                           C  ;-------------------------------------------------------------
05FB                       C  PODSTG  PROC    NEAR
05FB  55                   C          PUSH    BP
05FC  FC                   C          CLD                                  ; SET DIR TO INCREMENT
05FD  2B FF                C          SUB     DI,DI                        ; SET DI=0000 REL TO START
                           C                                               ;  OF SEGMENT
05FF  2B C0                C          SUB     AX,AX                        ; INITIAL DATA PATTERN FOR
                           C                                               ;  00-FF TEST
0601  E8 0CFE R            C          CALL    DDS
                           C          ASSUME  DS:ABS0
0604  8B 1E 0472 R         C          MOV     BX,DS:RESET_FLAG             ; WARM START?
0608  81 FB 1234           C          CMP     BX,1234H
060C  8C C2                C          MOV     DX,ES
060E  8E DA                C          MOV     DS,DX                        ; RESTORE DS
0610  74 62                C          JE      PODSTG_5                     ; GO DO FILL WITH 0000
                           C                                               ;  IF WARM START
0612  81 FB 4321           C          CMP     BX,4321H                     ; DCP WARM START?
0616  74 5C                C          JE      PODSTG_5                     ; DO FILL IF SO
0618                       C  PODSTG_1:
0618  88 05                C          MOV     [DI],AL                      ; WRITE TEST DATA
061A  8A 05                C          MOV     AL,[DI]                      ; GET IT BACK
061C  32 C4                C          XOR     AL,AH                        ; COMPARE TO EXPECTED
061E  75 40                C          JNZ     PODSTG_ERR0                  ; ERROR EXIT IF MISCOMPARE
0620  FE C4                C          INC     AH                           ; FORM NEW DATA PATTERN
0622  8A C4                C          MOV     AL,AH
0624  75 F2                C          JNZ     PODSTG_1                     ; LOOP TILL ALL 256 DATA
                           C                                               ;  PATTERNS DONE
0626  8B E9                C          MOV     BP,CX                        ; SAVE WORD COUNT
0628  B8 AA55              C          MOV     AX,0AA55H                    ; LOAD DATA PATTERN
062B  8B D8                C          MOV     BX,AX
062D  BA 55AA              C          MOV     DX,055AAH                    ; LOAD OTHER DATA PATTERN
0630  F3/ AB               C          REP     STOSW                        ; FILL WORDS FROM LOW TO
                           C                                               ;  HIGH WITH AAAA
0632  4F                   C          DEC     DI                           ; POINT TO LAST WORD
0633  4F                   C          DEC     DI                           ;  WRITTEN
0634  FD                   C          STD                                  ; SET DIR FLAG TO GO DOWN
0635  8B F7                C          MOV     SI,DI                        ; SET INDEX REGS. EQUAL
0637  8B CD                C          MOV     CX,BP                        ; RECOVER WORD COUNT
0639                       C  PODSTG_2:                                    ; GO FROM HIGH TO LOW
0639  AD                   C          LODSW                                ; GET WORD FROM MEMORY
063A  33 C3                C          XOR     AX,BX                        ; EQUAL WHAT S/B THERE?
063C  75 22                C          JNZ     PODSTG_ERR0                  ; GO ERROR EXIT IF NOT
063E  8B C2                C          MOV     AX,DX                        ; GET 55 DATA PATTERN AND
0640  AB                   C          STOSW                                ;  STORE IN LOC JUST READ
0641  E2 F6                C          LOOP    PODSTG_2                     ; LOOP TILL ALL BYTES DONE
0643  8B CD                C          MOV     CX,BP                        ; RECOVER WORD COUNT
0645  FC                   C          CLD                                  ; BACK TO INCREMENT
0646  46                   C          INC     SI                           ; ADJUST PTRS
0647  46                   C          INC     SI
0648  8B FE                C          MOV     DI,SI
064A                       C  PODSTG_3:                                    ; LOW TO HIGH DOING WORDS
064A  AD                   C          LODSW                                ; GET A WORD
064B  33 C2                C          XOR     AX,DX                        ; SHOULD COMPARE TO DX
064D  75 11                C          JNZ     PODSTG_ERR0                  ; GO ERROR IF NOT
064F  AB                   C          STOSW                                ; WRITE 0000 BACK TO LOC
                           C                                               ;  JUST READ
0650  E2 F8                C          LOOP    PODSTG_3                     ; LOOP TILL DONE
                           C
0652  FD                   C          STD                                  ; BACK TO DECREMENT
0653  4E                   C          DEC     SI                           ; ADJUST POINTER DOWN TO
                           C                                               ;  LAST WORD WRITTEN
0654  4E                   C          DEC     SI
0655  8B CD                C          MOV     CX,BP                        ; GET WORD COUNT
0657                       C  PODSTG_4:
0657  AD                   C          LODSW                                ; GET WORD
0658  0B C0                C          OR      AX,AX                        ; = TO 0000
065A  75 04                C          JNZ     PODSTG_ERR0                  ; ERROR IF NOT
065C  E2 F9                C          LOOP    PODSTG_4                     ; LOOP TILL DONE
065E  EB 11                C          JMP     SHORT PODSTG_ERR2
0660                       C  PODSTG_ERR0:
0660  8B C8                C          MOV     CX,AX                        ; SAVE BITS IN ERROR
0662  32 E4                C          XOR     AH,AH
0664  0A ED                C          OR      CH,CH                        ; HIGH BYTE ERROR?
```

```
0666  74 02                 C               JZ      PODSTG_ERR1
0668  B4 01                 C               MOV     AH,1                    ; SET HIGH BYTE ERROR
066A                        C       PODSTG_ERR1:
066A  0A C9                 C               OR      CL,CL                   ; LOW BYTE ERROR?
066C  74 03                 C               JZ      PODSTG_ERR2
066E  80 C4 02              C               ADD     AH,2
0671                        C       PODSTG_ERR2:
0671  5D                    C               POP     BP
0672  FC                    C               CLD                             ; SET DIR FLAG BACK TO INC
0673  C3                    C               RET                             ; RETURN TO CALLER
0674                        C       PODSTG_5:                               ; SIMPLE FILL WITH 0000 ON
                            C                                               ;  WARM-START
0674  50                    C               PUSH    AX                      ; SAVE
0675  52                    C               PUSH    DX                      ; SAVE VALUE
0676  B6 03                 C               MOV     DH,3
0678  B2 C4                 C               MOV     DL,SEQ_ADDR             ; SEQ_ADDR  REGISTER
067A  B8 020F               C               MOV     AX,020FH
067D  E8 0D15 R             C               CALL    OUT_DX                  ; DO IT
0680  5A                    C               POP     DX                      ; RESTORE
0681  58                    C               POP     AX                      ; RESTORE
0682  F3/ AB                C               REP     STOSW
0684  E8 0CFE R             C               CALL    DDS
                            C               ASSUME  DS:ABS0
0687  89 1E 0472 R          C               MOV     DS:RESET_FLAG,BX
068B  8E DA                 C               MOV     DS,DX                   ; RESTORE DS
068D  EB E2                 C               JMP     PODSTG_ERR2             ; AND EXIT
068F                        C       PODSTG  ENDP
                            C
                            C       ;----- DETERMINE SIZE OF BUFFER
                            C
068F                        C       HOW_BIG         PROC NEAR
068F  8C DA                 C               MOV     DX,DS                   ; SET PNTR TO BUFFER LOC
0691  2B DB                 C               SUB     BX,BX                   ; BASIC COUNT OF 00K
0693                        C       FILL_LOOP:
0693  8E C2                 C               MOV     ES,DX                   ; SET SEG. REG
0695  2B FF                 C               SUB     DI,DI
0697  B8 AA55               C               MOV     AX,0AA55H               ; TEST PATTERN
069A  8B C8                 C               MOV     CX,AX
069C  26: 89 05             C               MOV     ES:[DI],AX              ; SEND TO MEMORY
069F  B0 0F                 C               MOV     AL,0FH                  ; PUT SOMETHING IN AL
06A1  26: 8B 05             C               MOV     AX,ES:[DI]              ; GET PATTERN FROM MEMORY
06A4  33 C1                 C               XOR     AX,CX                   ; COMPARE PATTERNS
06A6  75 14                 C               JNZ     HOW_BIG_END             ; GO END IF NO COMPARE
06A8  B9 2000               C               MOV     CX,2000H                ; SET COUNT FOR 8K WORDS
06AB  F3/ AB                C       REP     STOSW                           ; FILL 8K WORDS
06AD  81 C2 0400            C               ADD     DX,0400H                ; POINT TO NEXT 16K BLOCK
06B1  83 C3 10              C               ADD     BX,16                   ; BUMP COUNT BY 16KB
06B4  80 FE B0              C               CMP     DH,0B0H
06B7  75 DA                 C               JNZ     FILL_LOOP               ; AREA YET ?(B000OH)
06B9  EB 01 90              C               JMP     HOW_BIG_END
06BC                        C       HOW_BIG_END:
06BC  80 FE A0              C               CMP     DH,0A0H                 ; 1ST 16KB OK
06BF  74 06                 C               JZ      HB_ERROR_EXIT
06C1                        C       RESUME:
06C1  01 5E 04              C               ADD     WORD PTR[BP][4],BX      ; SAVE BUFFER FOUND
06C4  B8 0000               C               MOV     AX,0
06C7                        C       HB_ERROR_EXIT:
06C7  C3                    C               RET
06C8                        C       HOW_BIG         ENDP
                            C
                            C       ;--------------------------------------
                            C       ;SUBROUTINES FOR POWER ON DIAGNOSTICS  :
                            C       ;----------------------------------------------------------------------
                            C       ;THIS PROCEDURE WILL ISSUE ONE LONG TONE (3 SEC) AND ONE OR          :
                            C       ;MORE SHORT TONES (1 SEC) TO INDICATE A FAILURE ON THE PLANAR        :
                            C       ;BOARD ,A BAD RAM MODULE,OR A PROBLEM WITH THE CRT.                  :
                            C       ;ENTRY REQUIREMENTS:                                                 :
                            C       ;       DH=NUMBER OF LONG TONES TO BEEP                              :
                            C       ;       DL=NUMBER OF SHORT TONES TO BEEP.                            :
                            C       ;----------------------------------------------------------------------
06C8                        C       ERR_BEEP    PROC    NEAR
06C8  9C                    C               PUSHF                           ; SAVE FLAGS
06C9  FA                    C               CLI                             ; DISABLE SYSTEM INTS
06CA  1E                    C               PUSH    DS
06CB  E8 0CFE R             C               CALL    DDS
                            C               ASSUME  DS:ABS0
06CE  0A F6                 C               OR      DH,DH                   ; ANY LONG TONES TO BEEP
06D0  74 0B                 C               JZ      G3                      ; NO, DO THE SHORT ONES
06D2                        C       G1:                                     ; LONG BEEP
06D2  B3 06                 C               MOV     BL,6                    ; COUNTER FOR BEEPS
06D4  E8 0D20 R             C               CALL    BEEP                    ; DO THE BEEP
06D7                        C       G2:
06D7  E2 FE                 C               LOOP    G2                      ; DELAY BETWEEN BEEPS
06D9  FE CE                 C               DEC     DH                      ; ANY MORE TO DO
06DB  75 F5                 C               JNZ     G1                      ; DO IT
06DD                        C       G3:
06DD  B3 01                 C               MOV     BL,1                    ; COUNTER FOR A SHORT BEEP
06DF  E8 0D20 R             C               CALL    BEEP                    ; DO IT
06E2                        C       G4:
06E2  E2 FE                 C               LOOP    G4                      ; DELAY BETWEEN BEEPS
06E4  FE CA                 C               DEC     DL                      ; DONE WITH SHORT BEEPS
06E6  75 F5                 C               JNZ     G3                      ; DO MORE
06E8                        C       G5:
06E8  E2 FE                 C               LOOP G5                         ; DELAY BEFORE RETURN
06EA                        C       G6:
06EA  E2 FE                 C               LOOP G6
06EC  1F                    C               POP     DS                      ; RESTORE CONTENTS OF DS
06ED  9D                    C               POPF                            ; RESTORE FLAGS
06EE  C3                    C               RET
06EF                        C       ERR_BEEP        ENDP
                            C
                            C               SUBTTL


06EF                                T2      LABEL   WORD
06EF  0EB3 R                                DW      OFFSET  AH0             ; MODE SET
06F1  10EF R                                DW      OFFSET  AH1             ; SET CURSOR TYPE
06F3  1157 R                                DW      OFFSET  AH2             ; SET CURSOR POSITION
06F5  1186 R                                DW      OFFSET  AH3             ; READ CURSOR POSITION
06F7  119D R                                DW      OFFSET  AH4             ; READ LIGHT PEN POSITION
06F9  12A4 R                                DW      OFFSET  AH5             ; ACTIVE DISPLAY PAGE
06FB  150E R                                DW      OFFSET  AH6             ; SCROLL DOWN
06FD  15B0 R                                DW      OFFSET  AH7             ; SCROLL UP
06FF  17D2 R                                DW      OFFSET  AH8             ; READ CHAR/ATTRIBUTE
0701  1899 R                                DW      OFFSET  AH9             ; WRITE CHAR/ATTRIBUTE
0703  18DD R                                DW      OFFSET  AHA             ; WRITE CHARACTER ONLY
0705  1A75 R                                DW      OFFSET  AHB             ; SET COLOR PALETTE
0707  1BCB R                                DW      OFFSET  AHC             ; WRITE DOT
0709  1C9F R                                DW      OFFSET  AHD             ; READ DOT
070B  1D01 R                                DW      OFFSET  AHE             ; WRITE TTY
070D  1D85 R                                DW      OFFSET  AHF             ; CURRENT VIDEO STATE
070F  1DC5 R                                DW      OFFSET  AH10            ; SET PALETTE REGISTERS
0711  1F98 R                                DW      OFFSET  AH11            ; CHAR GENERATOR ROUTINE
0713  20BF R                                DW      OFFSET  AH12            ; ALTERNATE SELECT
0715  2118 R                                DW      OFFSET  AH13            ; WRITE STRING
= 0028                              T2L     EQU     $-T2
```

```

                         C          INCLUDE         VPARMS.INC
                         C          SUBTTL  VPARMS.INC
                         C          PAGE
0717                     C  VIDEO_PARMS     LABEL   BYTE
                         C
                         C  ; STRUCTURE OF THIS TABLE
                         C  ;
                         C  ;       COLUMNS, ROWS, PELS PER CHARACTER
                         C  ;       PAGE LENGTH
                         C  ;       SEQUENCER PARAMETERS
                         C  ;       MISCELLANEOUS REGISTER
                         C  ;       CRTC PARAMETERS
                         C  ;       ATTRIBUTE PARAMETERS
                         C  ;       GRAPHICS PARAMETERS
                         C
= 0000                   C  BASE_1  EQU     $ - VIDEO_PARMS
0717                     C  BASE_1_L        LABEL   BYTE
                         C
                         C  ;----- DEFAULT MODES
                         C
                         C  ;--0--
0717  28 18 08           C          DB      40D,24D,08D
071A  0800               C          DW      00800H
                         C
= 0005                   C  TFS_LEN EQU     $ - BASE_1_L
                         C
071C                     C  SEQ_PARMS       LABEL   BYTE
071C  0B 03 00 03        C          DB      00BH,003H,000H,003H
= 0004                   C  M1      EQU     $ - SEQ_PARMS
                         C
0720  23                 C          DB      023H
                         C
0721                     C  CRT_PARMS       LABEL   BYTE
0721  37 27 2D 37 31 15  C          DB      037H,027H,02DH,037H,031H,015H
0727  04 11 00 07 06 07  C          DB      004H,011H,000H,007H,006H,007H
072D  00 00 00 00 E1 24  C          DB      000H,000H,000H,000H,0E1H,024H
0733  C7 14 08 E0 F0 A3  C          DB      0C7H,014H,008H,0E0H,0F0H,0A3H
0739  FF                 C          DB      0FFH
= 0019                   C  M4      EQU     $-CRT_PARMS
                         C
= 0023                   C  LN_4    EQU     $ - BASE_1_L
                         C
073A                     C  ATTR_PARMS      LABEL   BYTE
073A  00 01 02 03 04 05  C          DB      000H,001H,002H,003H,004H,005H
0740  06 07 10 11 12 13  C          DB      006H,007H,010H,011H,012H,013H
0746  14 15 16 17 08 00  C          DB      014H,015H,016H,017H,008H,000H
074C  0F 00              C          DB      00FH,000H
= 0014                   C  M5      EQU     $-ATTR_PARMS
                         C
= 0037                   C  LN_2    EQU     $ - BASE_1_L
074E                     C  GRAPH_PARMS     LABEL   BYTE
074E  00 00 00 00 00 10  C          DB      000H,000H,000H,000H,000H,010H
0754  0E 00 FF           C          DB      00EH,000H,0FFH
= 0009                   C  M6      EQU     $-GRAPH_PARMS
                         C
= 0040                   C  M_TBL_LEN       EQU     $ - BASE_1_L
                         C
                         C  ;--1--
0757  28 18 08           C          DB      40D,24D,08D
075A  0800               C          DW      00800H
                         C
075C  0B 03 00 03        C          DB      00BH,003H,000H,003H
                         C
0760  23                 C          DB      023H
                         C
0761  37 27 2D 37 31 15  C          DB      037H,027H,02DH,037H,031H,015H
0767  04 11 00 07 06 07  C          DB      004H,011H,000H,007H,006H,007H
076D  00 00 00 00 E1 24  C          DB      000H,000H,000H,000H,0E1H,024H
0773  C7 14 08 E0 F0 A3  C          DB      0C7H,014H,008H,0E0H,0F0H,0A3H
0779  FF                 C          DB      0FFH
                         C
077A  00 01 02 03 04 05  C          DB      000H,001H,002H,003H,004H,005H
0780  06 07 10 11 12 13  C          DB      006H,007H,010H,011H,012H,013H
0786  14 15 16 17 08 00  C          DB      014H,015H,016H,017H,008H,000H
078C  0F 00              C          DB      00FH,000H
                         C
078E  00 00 00 00 00 10  C          DB      000H,000H,000H,000H,000H,010H
0794  0E 00 FF           C          DB      00EH,000H,0FFH
                         C
                         C  ;--2--
0797  50 18 08           C          DB      80D,24D,08D
079A  1000               C          DW      01000H
                         C
079C  01 03 00 03        C          DB      001H,003H,000H,003H
                         C
07A0  23                 C          DB      023H
                         C
07A1  70 4F 5C 2F 5F 07  C          DB      070H,04FH,05CH,02FH,05FH,007H
07A7  04 11 00 07 06 07  C          DB      004H,011H,000H,007H,006H,007H
07AD  00 00 00 00 E1 24  C          DB      000H,000H,000H,000H,0E1H,024H
07B3  C7 28 08 E0 F0 A3  C          DB      0C7H,028H,008H,0E0H,0F0H,0A3H
07B9  FF                 C          DB      0FFH
                         C
07BA  00 01 02 03 04 05  C          DB      000H,001H,002H,003H,004H,005H
07C0  06 07 10 11 12 13  C          DB      006H,007H,010H,011H,012H,013H
07C6  14 15 16 17 08 00  C          DB      014H,015H,016H,017H,008H,000H
07CC  0F 00              C          DB      00FH,000H
                         C
07CE  00 00 00 00 00 10  C          DB      000H,000H,000H,000H,000H,010H
07D4  0E 00 FF           C          DB      00EH,000H,0FFH
                         C
                         C  ;--3--
07D7  50 18 08           C          DB      80D,24D,08D
07DA  1000               C          DW      01000H
                         C
07DC  01 03 00 03        C          DB      001H,003H,000H,003H
                         C
07E0  23                 C          DB      023H
                         C
07E1  70 4F 5C 2F 5F 07  C          DB      070H,04FH,05CH,02FH,05FH,007H
07E7  04 11 00 07 06 07  C          DB      004H,011H,000H,007H,006H,007H
07ED  00 00 00 00 E1 24  C          DB      000H,000H,000H,000H,0E1H,024H
07F3  C7 28 08 E0 F0 A3  C          DB      0C7H,028H,008H,0E0H,0F0H,0A3H
07F9  FF                 C          DB      0FFH
                         C
07FA  00 01 02 03 04 05  C          DB      000H,001H,002H,003H,004H,005H
0800  06 07 10 11 12 13  C          DB      006H,007H,010H,011H,012H,013H
0806  14 15 16 17 08 00  C          DB      014H,015H,016H,017H,008H,000H
080C  0F 00              C          DB      00FH,000H
                         C
080E  00 00 00 00 00 10  C          DB      000H,000H,000H,000H,000H,010H
0814  0E 00 FF           C          DB      00EH,000H,0FFH
                         C
                         C  ;--4--
0817  28 18 08           C          DB      40D,24D,08D
```

```
081A  4000                  1891   C                DW      04000H
                            1892   C
081C  0B 03 00 02           1893   C                DB      00BH,003H,000H,002H
                            1894   C
0820  23                    1895   C                DB      023H
                            1896   C
0821  37 27 2D 37 30 14     1897   C                DB      037H,027H,02DH,037H,030H,014H
0827  04 11 00 01 00 00     1898   C                DB      004H,011H,000H,001H,000H,000H
082D  00 00 00 00 E1 24     1899   C                DB      000H,000H,000H,000H,0E1H,024H
0833  C7 14 00 E0 F0 A2     1900   C                DB      0C7H,014H,000H,0E0H,0F0H,0A2H
0839  FF                    1901   C                DB      0FFH
                            1902   C
083A  00 13 15 17 02 04     1903   C                DB      000H,013H,015H,017H,002H,004H
0840  06 07 10 11 12 13     1904   C                DB      006H,007H,010H,011H,012H,013H
0846  14 15 16 17 01 00     1905   C                DB      014H,015H,016H,017H,001H,000H
084C  03 00                 1906   C                DB      003H,000H
                            1907   C
084E  00 00 00 00 00 30     1908   C                DB      000H,000H,000H,000H,000H,030H
0854  0F 00 FF              1909   C                DB      00FH,000H,0FFH
                            1910   C
                            1911   C  ;--5--
0857  28 18 08              1912   C                DB      40D,24D,08D
085A  4000                  1913   C                DW      04000H
                            1914   C
085C  0B 03 00 02           1915   C                DB      00BH,003H,000H,002H
                            1916   C
0860  23                    1917   C                DB      023H
                            1918   C
0861  37 27 2D 37 30 14     1919   C                DB      037H,027H,02DH,037H,030H,014H
0867  04 11 00 01 00 00     1920   C                DB      004H,011H,000H,001H,000H,000H
086D  00 00 00 00 E1 24     1921   C                DB      000H,000H,000H,000H,0E1H,024H
0873  C7 14 00 E0 F0 A2     1922   C                DB      0C7H,014H,000H,0E0H,0F0H,0A2H
0879  FF                    1923   C                DB      0FFH
                            1924   C
087A  00 13 15 17 02 04     1925   C                DB      000H,013H,015H,017H,002H,004H
0880  06 07 10 11 12 13     1926   C                DB      006H,007H,010H,011H,012H,013H
0886  14 15 16 17 01 00     1927   C                DB      014H,015H,016H,017H,001H,000H
088C  03 00                 1928   C                DB      003H,000H
                            1929   C
088E  00 00 00 00 00 30     1930   C                DB      000H,000H,000H,000H,000H,030H
0894  0F 00 FF              1931   C                DB      00FH,000H,0FFH
                            1932   C
                            1933   C  ;--6--
0897  50 18 08              1934   C                DB      80D,24D,08D
089A  4000                  1935   C                DW      04000H
                            1936   C
089C  01 01 00 06           1937   C                DB      001H,001H,000H,006H
                            1938   C
08A0  23                    1939   C                DB      023H
                            1940   C
08A1  70 4F 59 2D 5E 06     1941   C                DB      070H,04FH,059H,02DH,05EH,006H
08A7  04 11 00 01 00 00     1942   C                DB      004H,011H,000H,001H,000H,000H
08AD  00 00 00 00 E0 23     1943   C                DB      000H,000H,000H,000H,0E0H,023H
08B3  C7 28 00 DF EF C2     1944   C                DB      0C7H,028H,000H,0DFH,0EFH,0C2H
08B9  FF                    1945   C                DB      0FFH
                            1946   C
08BA  00 17 17 17 17 17     1947   C                DB      000H,017H,017H,017H,017H,017H
08C0  17 17 17 17 17 17     1948   C                DB      017H,017H,017H,017H,017H,017H
08C6  17 17 17 17 01 00     1949   C                DB      017H,017H,017H,017H,001H,000H
08CC  01 00                 1950   C                DB      001H,000H
                            1951   C
08CE  00 00 00 00 00 00     1952   C                DB      000H,000H,000H,000H,000H,000H
08D4  0D 00 FF              1953   C                DB      00DH,000H,0FFH
                            1954   C
                            1955   C  ;--7--
08D7  50 18 0E              1956   C                DB      80D,24D,14D
08DA  1000                  1957   C                DW      01000H
                            1958   C
08DC  00 03 00 03           1959   C                DB      000H,003H,000H,003H
                            1960   C
08E0  A6                    1961   C                DB      0A6H
                            1962   C
08E1  60 4F 56 3A 51 60     1963   C                DB      060H,04FH,056H,03AH,051H,060H
08E7  70 1F 00 0D 0B 0C     1964   C                DB      070H,01FH,000H,00DH,00BH,00CH
08ED  00 00 00 00 5E 2E     1965   C                DB      000H,000H,000H,000H,05EH,02EH
08F3  5D 28 0D 5E 6E A3     1966   C                DB      05DH,028H,00DH,05EH,06EH,0A3H
08F9  FF                    1967   C                DB      0FFH
                            1968   C
08FA  00 08 08 08 08 08     1969   C                DB      000H,008H,008H,008H,008H,008H
0900  08 08 10 18 18 18     1970   C                DB      008H,008H,010H,018H,018H,018H
0906  18 18 18 18 0E 00     1971   C                DB      018H,018H,018H,018H,00EH,000H
090C  0F 08                 1972   C                DB      00FH,008H
                            1973   C
090E  00 00 00 00 00 10     1974   C                DB      000H,000H,000H,000H,000H,010H
0914  0A 00 FF              1975   C                DB      00AH,000H,0FFH
                            1976   C
                            1977   C  ;--8--
0917  28 18 08              1978   C                DB      40D,24D,08D
091A  4000                  1979   C                DW      04000H
                            1980   C
091C  00 00 00 03           1981   C                DB      000H,000H,000H,003H
                            1982   C
0920  23                    1983   C                DB      023H
                            1984   C
0921  37 27 2D 37 31 15     1985   C                DB      037H,027H,02DH,037H,031H,015H
0927  04 11 00 07 06 07     1986   C                DB      004H,011H,000H,007H,006H,007H
092D  00 00 00 00 E1 24     1987   C                DB      000H,000H,000H,000H,0E1H,024H
0933  C7 14 08 E0 F0 A3     1988   C                DB      0C7H,014H,008H,0E0H,0F0H,0A3H
0939  FF                    1989   C                DB      0FFH
                            1990   C
093A  00 01 02 03 04 05     1991   C                DB      000H,001H,002H,003H,004H,005H
0940  06 07 10 11 12 13     1992   C                DB      006H,007H,010H,011H,012H,013H
0946  14 15 16 17 08 00     1993   C                DB      014H,015H,016H,017H,008H,000H
094C  0F 00                 1994   C                DB      00FH,000H
                            1995   C
094E  00 00 00 00 00 10     1996   C                DB      000H,000H,000H,000H,000H,010H
0954  0E 00 FF              1997   C                DB      00EH,000H,0FFH
                            1998   C
                            1999   C  ;--9--
0957  28 18 08              2000   C                DB      40D,24D,08D
095A  4000                  2001   C                DW      04000H
                            2002   C
095C  00 00 00 03           2003   C                DB      000H,000H,000H,003H
                            2004   C
0960  23                    2005   C                DB      023H
                            2006   C
0961  37 27 2D 37 31 15     2007   C                DB      037H,027H,02DH,037H,031H,015H
0967  04 11 00 07 06 07     2008   C                DB      004H,011H,000H,007H,006H,007H
096D  00 00 00 00 E1 24     2009   C                DB      000H,000H,000H,000H,0E1H,024H
0973  C7 14 08 E0 F0 A3     2010   C                DB      0C7H,014H,008H,0E0H,0F0H,0A3H
0979  FF                    2011   C                DB      0FFH
                            2012   C
097A  00 01 02 03 04 05     2013   C                DB      000H,001H,002H,003H,004H,005H
0980  06 07 10 11 12 13     2014   C                DB      006H,007H,010H,011H,012H,013H
0986  14 15 16 17 08 00     2015   C                DB      014H,015H,016H,017H,008H,000H
098C  0F 00                 2016   C                DB      00FH,000H
```

```
                           C
098E  00 00 00 00 00 10    C           DB     000H,000H,000H,000H,000H,010H
0994  0E 00 FF             C           DB     00EH,000H,0FFH
                           C
                           C  ;--A--
0997  28 18 08             C           DB     40D,24D,08D
099A  4000                 C           DW     04000H
                           C
099C  00 00 00 03          C           DB     000H,000H,000H,003H
                           C
09A0  23                   C           DB     023H
                           C
09A1  37 27 2D 37 31 15    C           DB     037H,027H,02DH,037H,031H,015H
09A7  04 11 00 07 06 07    C           DB     004H,011H,000H,007H,006H,007H
09AD  00 00 00 00 E1 24    C           DB     000H,000H,000H,000H,0E1H,024H
09B3  C7 14 08 E0 F0 A3    C           DB     0C7H,014H,008H,0E0H,0F0H,0A3H
09B9  FF                   C           DB     0FFH
                           C
09BA  00 01 02 03 04 05    C           DB     000H,001H,002H,003H,004H,005H
09C0  06 07 10 11 12 13    C           DB     006H,007H,010H,011H,012H,013H
09C6  14 15 16 17 08 00    C           DB     014H,015H,016H,017H,008H,000H
09CC  0F 00                C           DB     00FH,000H
                           C
09CE  00 00 00 00 00 10    C           DB     000H,000H,000H,000H,000H,010H
09D4  0E 00 FF             C           DB     00EH,000H,0FFH
                           C
                           C  ;--B--
09D7  50 18 08             C           DB     80D,24D,08D
09DA  1000                 C           DW     01000H
                           C
09DC  01 04 00 07          C           DB     001H,004H,000H,007H
                           C
09E0  23                   C           DB     023H
                           C
09E1  70 4F 5C 2F 5F 07    C           DB     070H,04FH,05CH,02FH,05FH,007H
09E7  04 11 00 07 06 07    C           DB     004H,011H,000H,007H,006H,007H
09ED  00 00 00 00 E1 24    C           DB     000H,000H,000H,000H,0E1H,024H
09F3  C7 28 08 E0 F0 A3    C           DB     0C7H,028H,008H,0E0H,0F0H,0A3H
09F9  FF                   C           DB     0FFH
                           C
09FA  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0A00  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0A06  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0A0C  0F 00                C           DB     00FH,000H
                           C
0A0E  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0A14  04 00 FF             C           DB     004H,000H,0FFH
                           C  ;--C--
0A17  50 18 0E             C           DB     80D,24D,14D
0A1A  1000                 C           DW     01000H
                           C
0A1C  00 04 00 07          C           DB     000H,004H,000H,007H
                           C
0A20  A6                   C           DB     0A6H
                           C
0A21  60 4F 56 3A 51 60    C           DB     060H,04FH,056H,03AH,051H,060H
0A27  70 1F 00 0D 0B 0C    C           DB     070H,01FH,000H,00DH,00BH,00CH
0A2D  00 00 00 00 5E 2E    C           DB     000H,000H,000H,000H,05EH,02EH
0A33  5D 28 0D 5E 6E A3    C           DB     05DH,028H,00DH,05EH,06EH,0A3H
0A39  FF                   C           DB     0FFH
                           C
0A3A  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0A40  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0A46  00 00 00 00 0E 00    C           DB     000H,000H,000H,000H,00EH,000H
0A4C  0F 08                C           DB     00FH,008H
                           C
0A4E  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0A54  04 00 FF             C           DB     004H,000H,0FFH
                           C  ;--D--
0A57  28 18 08             C           DB     40D,24D,08D
0A5A  2000                 C           DW     02000H
                           C
0A5C  0B 0F 00 06          C           DB     00BH,00FH,000H,006H
                           C
0A60  23                   C           DB     023H
                           C
0A61  37 27 2D 37 30 14    C           DB     037H,027H,02DH,037H,030H,014H
0A67  04 11 00 00 00 00    C           DB     004H,011H,000H,000H,000H,000H
0A6D  00 00 00 00 E1 24    C           DB     000H,000H,000H,000H,0E1H,024H
0A73  C7 14 00 E0 F0 E3    C           DB     0C7H,014H,000H,0E0H,0F0H,0E3H
0A79  FF                   C           DB     0FFH
                           C
0A7A  00 01 02 03 04 05    C           DB     000H,001H,002H,003H,004H,005H
0A80  06 07 10 11 12 13    C           DB     006H,007H,010H,011H,012H,013H
0A86  14 15 16 17 01 00    C           DB     014H,015H,016H,017H,001H,000H
0A8C  0F 00                C           DB     00FH,000H
                           C
0A8E  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0A94  05 0F FF             C           DB     005H,00FH,0FFH
                           C  ;--E--
0A97  50 18 08             C           DB     80D,24D,08D
0A9A  4000                 C           DW     04000H
                           C
0A9C  01 0F 00 06          C           DB     001H,00FH,000H,006H
                           C
0AA0  23                   C           DB     023H
                           C
0AA1  70 4F 59 2D 5E 06    C           DB     070H,04FH,059H,02DH,05EH,006H
0AA7  04 11 00 00 00 00    C           DB     004H,011H,000H,000H,000H,000H
0AAD  00 00 00 00 E0 23    C           DB     000H,000H,000H,000H,0E0H,023H
0AB3  C7 28 00 DF EF E3    C           DB     0C7H,028H,000H,0DFH,0EFH,0E3H
0AB9  FF                   C           DB     0FFH
                           C
0ABA  00 01 02 03 04 05    C           DB     000H,001H,002H,003H,004H,005H
0AC0  06 07 10 11 12 13    C           DB     006H,007H,010H,011H,012H,013H
0AC6  14 15 16 17 01 00    C           DB     014H,015H,016H,017H,001H,000H
0ACC  0F 00                C           DB     00FH,000H
                           C
0ACE  00 00 00 00 00 00    C           DB     000H,000H,000H,000H,000H,000H
0AD4  05 0F FF             C           DB     005H,00FH,0FFH
                           C  ;--F--
0AD7  50 18 0E             C           DB     80D,24D,14D
0ADA  8000                 C           DW     08000H
                           C
0ADC  05 0F 00 00          C           DB     005H,00FH,000H,000H
                           C
0AE0  A2                   C           DB     0A2H
                           C
0AE1  60 4F 56 1A 50 E0    C           DB     060H,04FH,056H,01AH,050H,0E0H
0AE7  70 1F 00 00 00 00    C           DB     070H,01FH,000H,000H,000H,000H
0AED  00 00 00 00 5E 2E    C           DB     000H,000H,000H,000H,05EH,02EH
0AF3  5D 14 0D 5E 6E 8B    C           DB     05DH,014H,00DH,05EH,06EH,08BH
0AF9  FF                   C           DB     0FFH
                           C
0AFA  00 08 00 00 18 18    C           DB     000H,008H,000H,000H,018H,018H
0B00  00 00 00 08 00 00    C           DB     000H,000H,000H,008H,000H,000H
```

```
0B06  00 18 00 00 0B 00      C            DB      000H,018H,000H,000H,00BH,000H
0B0C  05 00                  C            DB      005H,000H
                             C
0B0E  00 00 00 00 00 10      C            DB      000H,000H,000H,000H,000H,010H
0B14  07 0F FF               C            DB      007H,00FH,0FFH
                             C  ;-10--
0B17  50 18 0E               C            DB      80D,24D,14D
0B1A  8000                   C            DW      08000H
                             C
0B1C  05 0F 00 00            C            DB      005H,00FH,000H,000H
                             C
0B20  A7                     C            DB      0A7H
                             C
0B21  5B 4F 53 17 50 BA      C            DB      05BH,04FH,053H,017H,050H,0BAH
0B27  6C 1F 00 00 00 00      C            DB      06CH,01FH,000H,000H,000H,000H
0B2D  00 00 00 00 5E 2B      C            DB      000H,000H,000H,000H,05EH,02BH
0B33  5D 14 0F 5F 0A 8B      C            DB      05DH,014H,00FH,05FH,00AH,08BH
0B39  FF                     C            DB      0FFH
                             C
0B3A  00 01 00 00 04 07      C            DB      000H,001H,000H,000H,004H,007H
0B40  00 00 00 01 00 00      C            DB      000H,000H,000H,001H,000H,000H
0B46  04 07 00 00 01 00      C            DB      004H,007H,000H,000H,001H,000H
0B4C  05 00                  C            DB      005H,000H
                             C
0B4E  00 00 00 00 00 10      C            DB      000H,000H,000H,000H,000H,010H
0B54  07 0F FF               C            DB      007H,00FH,0FFH
                             C
= 0440                       C  BASE_2    EQU     $ - VIDEO_PARMS
                             C
                             C  ;-----    > 16K MODE VALUES
                             C
                             C  ;--F--
0B57  50 18 0E               C            DB      80D,24D,14D
0B5A  8000                   C            DW      08000H
                             C
0B5C  01 0F 00 06            C            DB      001H,00FH,000H,006H
                             C
0B60  A2                     C            DB      0A2H
                             C
0B61  60 4F 56 3A 50 60      C            DB      060H,04FH,056H,03AH,050H,060H
0B67  70 1F 00 00 00 00      C            DB      070H,01FH,000H,000H,000H,000H
0B6D  00 00 00 00 5E 2E      C            DB      000H,000H,000H,000H,05EH,02EH
0B73  5D 28 0D 5E 6E E3      C            DB      05DH,028H,00DH,05EH,06EH,0E3H
0B79  FF                     C            DB      0FFH
                             C
0B7A  00 08 00 00 18 18      C            DB      000H,008H,000H,000H,018H,018H
0B80  00 00 00 08 00 00      C            DB      000H,000H,000H,008H,000H,000H
0B86  00 18 00 00 0B 00      C            DB      000H,018H,000H,000H,00BH,000H
0B8C  05 00                  C            DB      005H,000H
                             C
0B8E  00 00 00 00 00 00      C            DB      000H,000H,000H,000H,000H,000H
0B94  05 0F FF               C            DB      005H,00FH,0FFH
                             C
                             C  ;-10--
0B97  50 18 0E               C            DB      80D,24D,14D
0B9A  8000                   C            DW      08000H
                             C
0B9C  01 0F 00 06            C            DB      001H,00FH,000H,006H
                             C
0BA0  A7                     C            DB      0A7H
                             C
0BA1  5B 4F 53 37 52 00      C            DB      05BH,04FH,053H,037H,052H,000H
0BA7  6C 1F 00 00 00 00      C            DB      06CH,01FH,000H,000H,000H,000H
0BAD  00 00 00 00 5E 2B      C            DB      000H,000H,000H,000H,05EH,02BH
0BB3  5D 28 0F 5F 0A E3      C            DB      05DH,028H,00FH,05FH,00AH,0E3H
0BB9  FF                     C            DB      0FFH
                             C
0BBA  00 01 02 03 04 05      C            DB      000H,001H,002H,003H,004H,005H
0BC0  14 07 38 39 3A 3B      C            DB      014H,007H,038H,039H,03AH,03BH
0BC6  3C 3D 3E 3F 01 00      C            DB      03CH,03DH,03EH,03FH,001H,000H
0BCC  0F 00                  C            DB      00FH,000H
                             C
0BCE  00 00 00 00 00 00      C            DB      000H,000H,000H,000H,000H,000H
0BD4  05 0F FF               C            DB      005H,00FH,0FFH
                             C
                             C
= 04C0                       C  BASE_3    EQU     $ - VIDEO_PARMS
                             C
                             C  ;----- HI RES ALTERNATE VALUES
                             C
                             C  ;--0--
0BD7  28 18 0E               C            DB      40D,24D,14D
0BDA  0800                   C            DW      00800H
                             C
0BDC  0B 03 00 03            C            DB      00BH,003H,000H,003H
                             C
0BE0  A7                     C            DB      0A7H
                             C
0BE1  2D 27 2B 2D 28 6D      C            DB      02DH,027H,02BH,02DH,028H,06DH
0BE7  6C 1F 00 0D 06 07      C            DB      06CH,01FH,000H,00DH,006H,007H
0BED  00 00 00 00 5E 2B      C            DB      000H,000H,000H,000H,05EH,02BH
0BF3  5D 14 0F 5E 0A A3      C            DB      05DH,014H,00FH,05EH,00AH,0A3H
0BF9  FF                     C            DB      0FFH
                             C
0BFA  00 01 02 03 04 05      C            DB      000H,001H,002H,003H,004H,005H
0C00  14 07 38 39 3A 3B      C            DB      014H,007H,038H,039H,03AH,03BH
0C06  3C 3D 3E 3F 08 00      C            DB      03CH,03DH,03EH,03FH,008H,000H
0C0C  0F 00                  C            DB      00FH,000H
                             C
0C0E  00 00 00 00 00 10      C            DB      000H,000H,000H,000H,000H,010H
0C14  0E 00 FF               C            DB      00EH,000H,0FFH
                             C
                             C  ;--1--
0C17  28 18 0E               C            DB      40D,24D,14D
0C1A  0800                   C            DW      00800H
                             C
0C1C  0B 03 00 03            C            DB      00BH,003H,000H,003H
                             C
0C20  A7                     C            DB      0A7H
                             C
0C21  2D 27 2B 2D 28 6D      C            DB      02DH,027H,02BH,02DH,028H,06DH
0C27  6C 1F 00 0D 06 07      C            DB      06CH,01FH,000H,00DH,006H,007H
0C2D  00 00 00 00 5E 2B      C            DB      000H,000H,000H,000H,05EH,02BH
0C33  5D 14 0F 5E 0A A3      C            DB      05DH,014H,00FH,05EH,00AH,0A3H
0C39  FF                     C            DB      0FFH
                             C
0C3A  00 01 02 03 04 05      C            DB      000H,001H,002H,003H,004H,005H
0C40  14 07 38 39 3A 3B      C            DB      014H,007H,038H,039H,03AH,03BH
0C46  3C 3D 3E 3F 08 00      C            DB      03CH,03DH,03EH,03FH,008H,000H
0C4C  0F 00                  C            DB      00FH,000H
                             C
0C4E  00 00 00 00 00 10      C            DB      000H,000H,000H,000H,000H,010H
0C54  0E 00 FF               C            DB      00EH,000H,0FFH
                             C
                             C  ;--2--
0C57  50 18 0E               C            DB      80D,24D,14D
```

```
0C5A  1000                C         DW       01000H
                          C
0C5C  01 03 00 03         C         DB       001H,003H,000H,003H
                          C
0C60  A7                  C         DB       0A7H
                          C
0C61  5B 4F 53 37 51 5B   C         DB       05BH,04FH,053H,037H,051H,05BH
0C67  6C 1F 00 0D 06 07   C         DB       06CH,01FH,000H,00DH,006H,007H
0C6D  00 00 00 00 5E 2B   C         DB       000H,000H,000H,000H,05EH,02BH
0C73  5D 28 0F 5E 0A A3   C         DB       05DH,028H,00FH,05EH,00AH,0A3H
0C79  FF                  C         DB       0FFH
                          C
0C7A  00 01 02 03 04 05   C         DB       000H,001H,002H,003H,004H,005H
0C80  14 07 38 39 3A 3B   C         DB       014H,007H,038H,039H,03AH,03BH
0C86  3C 3D 3E 3F 08 00   C         DB       03CH,03DH,03EH,03FH,008H,000H
0C8C  0F 00               C         DB       00FH,000H
                          C
0C8E  00 00 00 00 00 10   C         DB       000H,000H,000H,000H,000H,010H
0C94  0E 00 FF            C         DB       00EH,000H,0FFH
                          C
                          C ;--3--
0C97  50 18 0E            C         DB       80D,24D,14D
0C9A  1000                C         DW       01000H
                          C
0C9C  01 03 00 03         C         DB       001H,003H,000H,003H
                          C
0CA0  A7                  C         DB       0A7H
                          C
0CA1  5B 4F 53 37 51 5B   C         DB       05BH,04FH,053H,037H,051H,05BH
0CA7  6C 1F 00 0D 06 07   C         DB       06CH,01FH,000H,00DH,006H,007H
0CAD  00 00 00 00 5E 2B   C         DB       000H,000H,000H,000H,05EH,02BH
0CB3  5D 28 0F 5E 0A A3   C         DB       05DH,028H,00FH,05EH,00AH,0A3H
0CB9  FF                  C         DB       0FFH
                          C
0CBA  00 01 02 03 04 05   C         DB       000H,001H,002H,003H,004H,005H
0CC0  14 07 38 39 3A 3B   C         DB       014H,007H,038H,039H,03AH,03BH
0CC6  3C 3D 3E 3F 08 00   C         DB       03CH,03DH,03EH,03FH,008H,000H
0CCC  0F 00               C         DB       00FH,000H
                          C
0CCE  00 00 00 00 00 10   C         DB       000H,000H,000H,000H,000H,010H
0CD4  0E 00 FF            C         DB       00EH,000H,0FFH
                          C
                          C
                          C         SUBTTL

                            ;----- VECTOR INTO <AH> SPECIFIED FUNCTION

0CD7                        COMBO_VIDEO      PROC     NEAR
0CD7  FB                            STI                                     ; INTERRUPTS ON
0CD8  FC                            CLD                                     ; SET DIRECTION FORWARD
0CD9  55                            PUSH     BP                             ; SAVE THE REGISTER SET
0CDA  06                            PUSH     ES
0CDB  1E                            PUSH     DS
0CDC  52                            PUSH     DX
0CDD  51                            PUSH     CX
0CDE  53                            PUSH     BX
0CDF  56                            PUSH     SI
0CE0  57                            PUSH     DI

0CE1  50                            PUSH     AX                             ; SAVE AX VALUE
0CE2  8A C4                         MOV      AL,AH                          ; GET INTO LOW BYTE
0CE4  32 E4                         XOR      AH,AH                          ; ZERO TO HIGH BYTE
0CE6  D1 E0                         SAL      AX,1                           ; * 2 FOR TABLE LOOKUP
0CE8  8B F0                         MOV      SI,AX                          ; PUT INTO SI FOR BRANCH
0CEA  3D 0028                       CMP      AX,T2L                         ; TEST FOR WITHIN RANGE
0CED  72 06                         JB       M2                             ; BRANCH AROUND BRANCH
0CEF  58                            POP      AX                             ; RECOVER REGISTER
0CF0  CD 42                         INT      42H                            ; PASS UNRECOGNIZED CALL
0CF2  E9 219E R                     JMP      V_RET                          ; RETURN TO CALLER
0CF5                        M2:
                                    ASSUME   DS:ABS0
0CF5  E8 0CFE R                     CALL     DDS
0CF8  58                            POP      AX                             ; RECOVER
0CF9  2E: FF A4 06EF R              JMP      WORD PTR CS:[SI + OFFSET T2]   ; JMP TO AH=0 THRU AH=XX

                            ;----- UTILITY ROUTINES

                            ;----- SET DS TO THE DATA SEGMENT

0CFE                        DDS     PROC     NEAR
0CFE  50                            PUSH     AX                             ; SAVE REGISTER
0CFF  2B C0                         SUB      AX,AX
0D01  8E D8                         MOV      DS,AX
0D03  58                            POP      AX                             ; RESTORE REGISTER
0D04  C3                            RET
0D05                        DDS     ENDP

0D05                        WHAT_BASE        PROC     NEAR
                                    ASSUME   DS:ABS0
0D05  1E                            PUSH     DS                             ; SAVE DATA SEGMENT
0D06  E8 0CFE R                     CALL     DDS                            ; GET LOW MEMORY SEGMENT
0D09  8B 16 0463 R                  MOV      DX,ADDR_6845                   ; GET CRTC ADDRESS
0D0D  80 E2 F0                      AND      DL,0F0H                        ; STRIP OFF LOW NIBBLE
0D10  80 CA 0A                      OR       DL,0AH                         ; SET TO STATUS REGISTER
0D13  1F                            POP      DS
0D14  C3                            RET
0D15                        WHAT_BASE        ENDP

0D15                        OUT_DX  PROC     NEAR                           ; AH=INDEX,AL=DATA,DX=PORT
0D15  86 C4                         XCHG     AL,AH                          ; GET INDEX VALUE
0D17  EE                            OUT      DX,AL                          ; SET INDEX REG
0D18  42                            INC      DX                             ; SET DX TO DATA REG
0D19  86 C4                         XCHG     AL,AH                          ; GET DATA VALUE
0D1B  EE                            OUT      DX,AL                          ; SET DATA REG
0D1C  4A                            DEC      DX                             ; SET DX BACK TO INDEX
0D1D  C3                            RET
0D1E                        OUT_DX  ENDP

                            ;----- ROUTINE TO SOUND BEEPER

0D1E                        BP_1    PROC     NEAR
0D1E  EE                            OUT      DX,AL
0D1F  C3                            RET
0D20                        BP_1    ENDP

0D20                        BEEP    PROC     NEAR
0D20  52                            PUSH     DX
0D21  BA 0043                       MOV      DX,TIMER+3
0D24  B0 B6                         MOV      AL,10110110B                   ; SEL TIM 2,LSB,MSB,BINARY
0D26  E8 0D1E R                     CALL     BP_1                           ; WRITE THE TIMER MODE REG
0D29  B8 0533                       MOV      AX,533H                        ; DIVISOR FOR 1000 HZ
0D2C  4A                            DEC      DX
0D2D  E8 0D1E R                     CALL     BP_1                           ; WRITE TIMER 2 CNT - LSB
0D30  8A C4                         MOV      AL,AH
0D32  E8 0D1E R                     CALL     BP_1                           ; WRITE TIMER 2 CNT - MSB
0D35  BA 0061                       MOV      DX,PORT_B
```

```
0D38  EC                             IN      AL,DX                    ; GET SETTING OF PORT
0D39  8A E0                          MOV     AH,AL                    ; SAVE THAT SETTING
0D3B  0C 03                          OR      AL,03                    ; TURN SPEAKER ON
0D3D  E8 0D1E R                      CALL    BP_1
0D40  2B C9                          SUB     CX,CX                    ; SET CNT TO WAIT 500 MS
0D42                         G7:
0D42  E2 FE                          LOOP    G7                       ; DELAY BEFORE TURNING OFF
0D44  FE CB                          DEC     BL                       ; DELAY CNT EXPIRED?
0D46  75 FA                          JNZ     G7                       ; NO-CONTINUE BEEPING SPK
0D48  8A C4                          MOV     AL,AH                    ; RECOVER VALUE OF PORT
0D4A  E8 0D1E R                      CALL    BP_1
0D4D  5A                             POP     DX
0D4E  C3                             RET                              ; RETURN TO CALLER
0D4F                         BEEP    ENDP

                             ;----- FIND THE PARAMETER TABLE VECTOR IN THE SAVE TABLE

0D4F                         SET_BASE        PROC    NEAR
                                     ASSUME  DS:ABS0
0D4F  E8 0CFE R                      CALL    DDS
0D52  C4 1E 04A8 R                   LES     BX,SAVE_PTR              ; GET PTR TP PTR TABLE
0D56  26: C4 1F                      LES     BX,DWORD PTR ES:[BX]     ; GET PARAMETER PTR
0D59  C3                             RET
0D5A                         SET_BASE        ENDP

                             ;----- ESTABLISH ADDRESSING TO THE CORRECT MODE TABLE ENTRY

0D5A                         MAKE_BASE       PROC    NEAR
                                     ASSUME  DS:ABS0
0D5A  51                             PUSH    CX
0D5B  52                             PUSH    DX
0D5C  E8 0D4F R                      CALL    SET_BASE                 ; GET PARM TBL PTR
0D5F  8A 26 0449 R                   MOV     AH,CRT_MODE
0D63  F6 06 0487 R 60                TEST    INFO,060H                ; TEST FOR BASE CARD
0D68  74 18                          JZ      B_M_1                    ; MIN MEMORY

                             ;----- WE HAVE A MEMORY EXPANSION OPTION HERE

0D6A  80 FC 0F                       CMP     AH,0FH
0D6D  75 07                          JNE     B_M_2
0D6F  81 C3 0440                     ADD     BX,BASE_2 - BASE_1
0D73  EB 33 90                       JMP     B_M_OUT
0D76                         B_M_2:
0D76  80 FC 10                       CMP     AH,010H
0D79  75 07                          JNE     B_M_1
0D7B  81 C3 0480                     ADD     BX,BASE_2 + M_TBL_LEN - BASE_1
0D7F  EB 27 90                       JMP     B_M_OUT
0D82                         B_M_1:
0D82  80 FC 03                       CMP     AH,03H
0D85  77 14                          JA      B_M_3                    ; SKIP ENHANCED PORTION

                             ;----- CHECK THE SWITCH SETTING FOR ENHANCEMENT

0D87  A0 0488 R                      MOV     AL,INFO_3
0D8A  24 0F                          AND     AL,0FH
0D8C  3C 03                          CMP     AL,03H                   ; SECONDARY EMULATE SETTING
0D8E  74 07                          JE      BRS
0D90  3C 09                          CMP     AL,09H                   ; PRIMARY EMULATE SETTING
0D92  74 03                          JE      BRS
0D94  EB 05 90                       JMP     B_M_3

                             ;----- WE WILL PERFORM ENHANCEMENT

0D97                         BRS:
0D97  81 C3 04C0                     ADD     BX,BASE_3 - BASE_1       ; VECTOR TO ENHANCEMENT TBL
0D9B                         B_M_3:
0D9B  8A 0E 0449 R                   MOV     CL,CRT_MODE
0D9F  2A ED                          SUB     CH,CH
0DA1  E3 05                          JCXZ    B_M_4

                             ;----- THIS LOOP WILL MOVE THE PTR TO THE INDIVIDUAL MODE ENTRY

0DA3                         B_M_5:
0DA3  83 C3 40                       ADD     BX,M_TBL_LEN             ; LENGTH OF ONE MODE ENTRY
0DA6  E2 FB                          LOOP    B_M_5
0DA8                         B_M_4:
0DA8                         B_M_OUT:
0DA8  5A                             POP     DX
0DA9  59                             POP     CX
0DAA  C3                             RET
0DAB                         MAKE_BASE       ENDP

                             ;----- PROGRAM THE EGA REGISTERS FROM THE PARAMETER TABLE

0DAB                         SET_REGS        PROC    NEAR
                                     ASSUME  DS:ABS0,ES:NOTHING

                             ;----- PROGRAM THE SEQUENCER

0DAB  E8 0D5A R                      CALL    MAKE_BASE                ; GET TABLE PTR
0DAE  83 C3 05                       ADD     BX,TFS_LEN               ; MODE TO SEQUENCER PARMS
0DB1  B6 03                          MOV     DH,3
0DB3  B2 C4                          MOV     DL,SEQ_ADDR
0DB5  B8 0001                        MOV     AX,0001H                 ; RESET SEQUENCER
0DB8  FA                             CLI                              ; DISABLE INTERRUPTS
0DB9  E8 0D15 R                      CALL    OUT_DX
0DBC  26: 8A 07                      MOV     AL,ES:[BX]               ; GET SEQUENCER VALUE
0DBF  FE C4                          INC     AH                       ; NEXT INDEX
0DC1  E8 0D15 R                      CALL    OUT_DX                   ; SET IT
0DC4                         D1:
0DC4  FE C4                          INC     AH                       ; NEXT INDEX REGISTER
0DC6  43                             INC     BX                       ; NEXT TABLE ENTRY
0DC7  26: 8A 07                      MOV     AL,ES:[BX]
0DCA  E8 0D15 R                      CALL    OUT_DX
0DCD  80 FC 05                       CMP     AH,M1+1
0DD0  72 F2                          JB      D1

0DD2  26: 8A 07                      MOV     AL,ES:[BX]
0DD5  43                             INC     BX
0DD6  B2 C2                          MOV     DL,MISC_OUTPUT
0DD8  EE                             OUT     DX,AL
0DD9  B2 C4                          MOV     DL,SEQ_ADDR
0DDB  B8 0003                        MOV     AX,0003H
0DDE  E8 0D15 R                      CALL    OUT_DX                   ; START SEQUENCER
0DE1  FB                             STI                              ; ENABLE INTERRUPTS

                             ;----- PROGRAM THE CRT CONTROLLER

0DE2  8B 16 0463 R                   MOV     DX,ADDR_6845             ; CRTC INDEX REGISTER
0DE6  2A E4                          SUB     AH,AH                    ; COUNTER
0DE8                         X1:
0DE8  26: 8A 07                      MOV     AL,ES:[BX]               ; GET VALUE FROM TABLE
0DEB  E8 0D15 R                      CALL    OUT_DX                   ; SET CRTC REGISTER
0DEE  43                             INC     BX                       ; NEXT TABLE ENTRY
0DEF  FE C4                          INC     AH                       ; NEXT INDEX VALUE
0DF1  80 FC 19                       CMP     AH,M4                    ; TEST REGISTER COUNT
```

```
0DF4  72 F2                               JB      X1                          ; DO THE REST
0DF6  26: 8B 47 F1                        MOV     AX,ES:[BX][-0FH]            ; GET CURSOR MODE
0DFA  86 E0                               XCHG    AH,AL
0DFC  A3 0460 R                           MOV     CURSOR_MODE,AX              ; SET LOW RAM VALUE

                          ;----- PROGRAM THE ATTRIBUTE CHIP

0DFF  8B F3                               MOV     SI,BX
0E01  E8 0D05 R                           CALL    WHAT_BASE
0E04  EC                                  IN      AL,DX
0E05  B2 C0                               MOV     DL,ATTR_WRITE
0E07  2A E4                               SUB     AH,AH                       ; INDEX COUNTER
0E09                      D3:
0E09  26: 8A 07                           MOV     AL,ES:[BX]                  ; GET DATA VALUE
0E0C  86 E0                               XCHG    AH,AL
0E0E  EE                                  OUT     DX,AL
0E0F  86 E0                               XCHG    AH,AL
0E11  EE                                  OUT     DX,AL
0E12  43                                  INC     BX                          ; NEXT DATA VALUE
0E13  FE C4                               INC     AH                          ; NEXT INDEX VALUE
0E15  80 FC 14                            CMP     AH,M5                       ; TEST REGISTER COUNT
0E18  72 EF                               JB      D3                          ; DO THE REST

0E1A  B0 00                               MOV     AL,0
0E1C  EE                                  OUT     DX,AL

                          ;----- CHECK IF PALETTE REGISTER VALUES ARE TO BE SAVED

0E1D  1E                                  PUSH    DS
0E1E  06                                  PUSH    ES
0E1F  C4 3E 04A8 R                        LES     DI,SAVE_PTR                 ; GET TABLE PTR
0E23  26: C4 7D 04                        LES     DI,DWORD PTR ES:[DI][4]     ; GET PALETTE PTR
0E27  8C C0                               MOV     AX,ES
0E29  0B C7                               OR      AX,DI
0E2B  74 09                               JZ      SAVE_OUT                    ; IF ZERO, NO SAVE OCCURS

                          ;----- STORE AWAY THE PALETTE VALUES IN RAM SAVE AREA

0E2D  1F                                  POP     DS
0E2E  1E                                  PUSH    DS
0E2F  B9 0010                             MOV     CX,16D
0E32  F3/ A4              REP             MOVSB                               ; SAVE THE PALETTE REGS
0E34  46                                  INC     SI
0E35  A4                                  MOVSB                               ; SAVE THE OVERSCAN REG
0E36                      SAVE_OUT:
0E36  07                                  POP     ES
0E37  1F                                  POP     DS

                          ;----- PROGRAM THE GRAPHICS CHIPS

0E38  B2 CC                               MOV     DL,GRAPH_1_POS
0E3A  B0 00                               MOV     AL,0
0E3C  EE                                  OUT     DX,AL
0E3D  B2 CA                               MOV     DL,GRAPH_2_POS
0E3F  B0 01                               MOV     AL,1
0E41  EE                                  OUT     DX,AL
0E42  B2 CE                               MOV     DL,GRAPH_ADDR
0E44  2A E4                               SUB     AH,AH
0E46                      D4:
0E46  26: 8A 07                           MOV     AL,ES:[BX]                  ; PARAMETER BYTE
0E49  E8 0D15 R                           CALL    OUT_DX                      ; SET IT
0E4C  43                                  INC     BX                          ; NEXT BYTE
0E4D  FE C4                               INC     AH                          ; NEXT REGISTER
0E4F  80 FC 09                            CMP     AH,M6
0E52  72 F2                               JB      D4                          ; CONTINUE
0E54  C3                                  RET
0E55                      SET_REGS                ENDP

                          ;----- MODE SET REGEN CLEAR ROUTINE

0E55                      BLANK   PROC    NEAR                                ; FILL REGEN WITH BLANKS
                                  ASSUME  DS:ABS0,ES:NOTHING
0E55  A0 0487 R                   MOV     AL,INFO                             ; SEE IF BLANK IS TO OCCUR
0E58  A8 80                       TEST    AL,080H                             ; MODE SET HIGH BIT
0E5A  75 39                       JNZ     OUT_1                               ; SKIP BLANK FOR REGEN
0E5C  BA B800                     MOV     DX,0B800H                           ; COLOR MODE REGEN ADDRESS
0E5F  A0 0449 R                   MOV     AL,CRT_MODE                         ; CURRENT MODE SET
0E62  3C 06                       CMP     AL,6                                ; 0-6 ARE COLOR MODES
0E64  76 0A                       JBE     CG0
0E66  BA B000                     MOV     DX,0B000H                           ; MONOCHROME REGEN ADDRESS
0E69  3C 07                       CMP     AL,7                                ; MONOCHROME MODE
0E6B  74 03                       JE      CG0
0E6D  BA A000                     MOV     DX,0A000H                           ; REMAINING MODES
0E70                      CG0:
0E70  BB 0720                     MOV     BX,0720H                            ; ALPHA BLANK VALUE
0E73  3C 04                       CMP     AL,4                                ; ALPHAMODES 0-3
0E75  72 06                       JB      WW1
0E77  3C 07                       CMP     AL,7                                ; ALPHA MODE
0E79  74 02                       JE      WW1
0E7B  2B DB                       SUB     BX,BX                               ; GRAPHICS BLANK VALUE
0E7D                      WW1:
                                  SRLOAD  ES                                  ; SET THE REGEN SEGMENT
0E7D  8E C2           +           MOV     ES,DX
0E7F  8B 0E 044C R                MOV     CX,CRT_LEN
0E83  E3 10                       JCXZ    OUT_1
0E85  B9 8000                     MOV     CX,08000H
0E88  80 FE A0                    CMP     DH,0A0H
0E8B  74 02                       JE      N_BA
0E8D  B5 40                       MOV     CH,040H
0E8F                      N_BA:
0E8F  8B C3                       MOV     AX,BX                               ; BLANK VALUE
0E91  2B FF                       SUB     DI,DI                               ; CLEAR POINTER
0E93  F3/ AB                      REP     STOSW                               ; CLEAR THE PAGE
0E95                      OUT_1:
0E95  C3                          RET                                         ; RETURN TO CALLER
0E96                      BLANK   ENDP

0E96                      PH_5    PROC    NEAR
0E96  E8 1DB7 R                   CALL    PAL_ON
0E99  C3                          RET
0E9A                      PH_5    ENDP

                          ;----- SEE IF WE ARE TO SUPPORT 640 X 350 ON A 640 X 200 MODE

0E9A                      BRST_DET        PROC    NEAR
                                  ASSUME  DS:ABS0
0E9A  50                          PUSH    AX
0E9B  1E                          PUSH    DS
0E9C  E8 0CFE R                   CALL    DDS
0E9F  A0 0488 R                   MOV     AL,INFO_3
0EA2  1F                          POP     DS
0EA3  24 0F                       AND     AL,0FH
0EA5  3C 03                       CMP     AL,03H                              ; EMULATE MODE
0EA7  74 07                       JE      B_YES
0EA9  3C 09                       CMP     AL,09H                              ; EMULATE MODE
0EAB  74 03                       JE      B_YES
```

```
0EAD  58                              POP     AX
0EAE  F8                              CLC
0EAF  C3                              RET
0EB0                          B_YES:
0EB0  58                              POP     AX
0EB1  F9                              STC
0EB2  C3                              RET
0EB3                          BRST_DET        ENDP

                              ;----- MODE SET

0EB3                          AHO:
                                      ASSUME  DS:ABSO
0EB3  FA                              CLI
0EB4  C7 06 010C R 0000 E             MOV     WORD PTR GRX_SET, OFFSET CGDDOT
0EBA  8C 0E 010E R                    MOV     WORD PTR GRX_SET+2, CS
0EBE  FB                              STI
0EBF  80 26 0487 R F3                 AND     INFO,11110011B              ; TURN OFF RETRACE BIT
                                                                          ;  EGA ACTIVE BIT
0EC4  50                              PUSH    AX                          ; SAVE
0EC5  F6 06 0487 R 02                 TEST    INFO,2
0ECA  74 2C                           JZ      ST_1                        ; THERE IS NO MONOCHROME
0ECC  A1 0410 R                       MOV     AX,EQUIP_FLAG               ; THERE IS A MONOCHROME
0ECF  24 30                           AND     AL,030H                     ; CHECK THE EQUIPMENT FLAG
0ED1  3C 30                           CMP     AL,030H                     ;  FOR MONOCHROME CALL
0ED3  74 48                           JE      ST_2                        ; IT IS A MONOCHROME CALL

                              ;----- FALL THROUGH => REGULAR COLOR CARD SETUP

0ED5  C6 06 0484 R 18                 MOV     ROWS,024D
0EDA  C7 06 0485 R 0008               MOV     POINTS,8
0EE0  58                              POP     AX                          ; RECOVER
0EE1  80 0E 0487 R 08                 OR      INFO,00001000B              ; EGA NOT ACTIVE
0EE6  3C 01                           CMP     AL,1
0EE8  76 09                           JBE     ST_7                        ; WAIT FOR RETRACE ON
0EEA  3C 04                           CMP     AL,4                        ; MODES 2,3 ONLY
0EEC  73 05                           JAE     ST_7
0EEE  80 0E 0487 R 04                 OR      INFO,00000100B              ; DO RETRACE
0EF3                          ST_7:
0EF3  CD 42                           INT     42H                         ; OTHER ADAPTER MODE CALL
0EF5  E9 219E R                       JMP     V_RET                       ; BACK TO CALLER

                              ;----- AT THIS POINT THERE IS NO MONOCHROME ATTACHED TO THE ADAPTER

0EF8                          ST_1:
0EF8  A1 0410 R                       MOV     AX,EQUIP_FLAG               ; TEST THE EQUIPMENT FLAG
0EFB  24 30                           AND     AL,030H                     ;  TO SEE IF THIS IS A
0EFD  3C 30                           CMP     AL,030H                     ;  MONOCHROME SETUP CALL
0EFF  75 40                           JNE     ST_3                        ; MUST BE COLOR TO CARD

                              ;----- FALL THROUGH => REGULAR MONOCHROME CARD SETUP

0F01  C6 06 0484 R 18                 MOV     ROWS,024D
0F06  C7 06 0485 R 000E               MOV     POINTS,014D
0F0C  58                              POP     AX                          ; RECOVER
0F0D  CD 42                           INT     42H                         ; OTHER ADAPTER MODE CALL
0F0F  C7 06 0460 R 0B0C               MOV     CURSOR_MODE,0B0CH           ; FIX PLANAR VALUE
0F15  80 0E 0487 R 08                 OR      INFO,8                      ; THE EGA IS NOT ACTIVE
0F1A  E9 219E R                       JMP     V_RET                       ; BACK TO CALLER

                              ;----- MONOCHROME SETUP TO THE ADAPTER

0F1D                          ST_2:
0F1D  58                              POP     AX                          ; RECOVER
0F1E  50                              PUSH    AX                          ; SAVE
0F1F  B6 03                           MOV     DH,3
0F21  24 80                           AND     AL,080H                     ; PICK OFF THE CLEAR BIT
0F23  80 26 0487 R 7F                 AND     INFO,07FH                   ; MASK OFF THE OTHER BITS
0F28  08 06 0487 R                    OR      INFO,AL                     ; SAVE REGEN CLEAR BIT
0F2C  58                              POP     AX                          ; RECOVER TRUE CALL VALUE
0F2D  24 7F                           AND     AL,07FH                     ; ALREADY DEALT WITH D7
0F2F  3C 0F                           CMP     AL,0FH                      ; A MONOCHROME MODE
0F31  74 02                           JE      ST_2A                       ; DO THIS MODE
0F33  B0 07                           MOV     AL,7                        ; REGULAR MONOCHROME
0F35                          ST_2A:
0F35  A2 0449 R                       MOV     CRT_MODE,AL                 ; SAVE MODE VALUE
0F38  B2 B4                           MOV     DL,CRTC_ADDR_B              ; IT IS 3-B-X
0F3A  89 16 0463 R                    MOV     ADDR_6845,DX                ; SAVE CRTC ADDRESS
0F3E  EB 1C 90                        JMP     QQ1                         ; CONTINUE THE MODE SET

                              ;----- COLOR SETUP TO THE ADAPTER

0F41                          ST_3:
0F41  58                              POP     AX                          ; RECOVER PARAMETER VALUE
0F42  50                              PUSH    AX                          ; SAVE IT
0F43  B6 03                           MOV     DH,3
0F45  24 80                           AND     AL,080H                     ; ISOLATE REGEN CLEAR BIT
0F47  80 26 0487 R 7F                 AND     INFO,07FH                   ; PREPARE INFO BYTE
0F4C  08 06 0487 R                    OR      INFO,AL                     ; SET IT, OR NOT
0F50  58                              POP     AX                          ; RECOVER TRUE MODE CALL
0F51  24 7F                           AND     AL,07FH                     ; DONE WITH D7
0F53  A2 0449 R                       MOV     CRT_MODE,AL                 ; SAVE THIS MODE
0F56  B2 D4                           MOV     DL,CRTC_ADDR                ; 3-D-X
0F58  89 16 0463 R                    MOV     ADDR_6845,DX                ; SAVE CRTC ADDRESS
0F5C                          QQ1:
0F5C  C7 06 044E R 0000               MOV     CRT_START,0                 ; SAVE START ADDRESS
0F62  C6 06 0462 R 00                 MOV     ACTIVE_PAGE,0               ; RESET PAGE VALUE TO ZERO
                                      ASSUME  ES:NOTHING
0F67  B9 0008                         MOV     CX,8                        ; 8 PAGES OF CURSOR VALUES
0F6A  BF 0450 R                       MOV     DI,OFFSET CURSOR_POSN       ; OFFSET
0F6D  1E                              PUSH    DS                          ; ESTABLISH
0F6E  07                              POP     ES                          ;  ADDRESSING
0F6F  2B C0                           SUB     AX,AX                       ; 0 THOSE CURSOR LOCATIONS
0F71  F3/ AB                          REP     STOSW                       ; CLEAR OUT SAVED VALUES

0F73  E8 0D5A R                       CALL    MAKE_BASE

0F76  26: 8A 07                       MOV     AL,ES:[BX]                  ; GET COLUMN COUNT
0F79  2A E4                           SUB     AH,AH                       ; ZERO HIGH BYTE
0F7B  A3 044A R                       MOV     CRT_COLS,AX                 ; STORE COLUMN VALUE

0F7E  26: 8A 47 01                    MOV     AL,ES:[BX][1]               ; GET ROW VALUE
0F82  A2 0484 R                       MOV     ROWS,AL                     ; STORE ROW VALUE

0F85  26: 8A 47 02                    MOV     AL,ES:[BX][2]               ; GET THE BYTES/CHAR
0F89  2A E4                           SUB     AH,AH                       ; ZERO HIGH BYTE
0F8B  A3 0485 R                       MOV     POINTS,AX                   ; STORE BYTES/CHAR

0F8E  26: 8B 47 03                    MOV     AX,ES:[BX][3]               ; GET PAGE SIZE
0F92  A3 044C R                       MOV     CRT_LEN,AX                  ; STORE PAGE LENGTH

0F95  2B DB                           SUB     BX,BX                       ; ZERO
0F97  B0 01                           MOV     AL,1                        ; MONOCHROME ALPHA CHAR GEN
0F99  8A 26 0449 R                    MOV     AH,CRT_MODE                 ; GET CURRENT MODE
0F9D  80 FC 07                        CMP     AH,7                        ; IS IT MONOCHROME
0FA0  74 0C                           JE      ENTRY_2                     ; 9X14 FONT
```

```
0FA2  80 FC 03                      CMP      AH,03H
0FA5  77 35                         JA       ENTRY_1

0FA7  E8 0E9A R                     CALL     BRST_DET
0FAA  72 02                         JC       ENTRY_2

0FAC  B0 02                         MOV      AL,2                      ; COLOR ALPHA CHAR GEN
0FAE                        ENTRY_2:
0FAE  E8 1EAE R                     CALL     CH_GEN                    ; LOAD ALPHA CHAR GEN
0FB1  E8 0CFE R                     CALL     DDS
0FB4  8A 26 0449 R                  MOV      AH,CRT_MODE               ; GET CURRENT MODE
0FB8  80 FC 07                      CMP      AH,7                      ; IS IT MONOCHROME
0FBB  74 03                         JE       FDG_IT                    ; 9X14 FONT
0FBD  EB 1D 90                      JMP      ENTRY_1
0FC0                        FDG_IT:
0FC0  BD 0000 E                     MOV      BP,OFFSET CGMN_FDG        ; TABLE POINTER
0FC3  BB 0E00                       MOV      BX,0E00H                  ; 14 BYTES PER CHAR
0FC6                        FDG:
0FC6  0E                            PUSH     CS                        ; GET THE ROM SEGMENT
0FC7  07                            POP      ES                        ;  INTO ES
0FC8  26: 8B 56 00                  MOV      DX,ES:[BP]                ; GET THE CHAR HEX CODE
0FCC  0B D2                         OR       DX,DX                     ; ZERO = NO MORE CHARS
0FCE  74 0C                         JZ       ENTRY_1                   ; NO MORE
0FD0  B9 0001                       MOV      CX,1                      ; DO ONE CHAR AT A TIME
0FD3  45                            INC      BP                        ; MOVE TO FIRST CODE POINT
0FD4  E8 1EF6 R                     CALL     DO_MAP2                   ; STORE THE CODE POINT
0FD7  83 C5 0E                      ADD      BP,014D                   ; ADJUST BP TO NEXT CODE
0FDA  EB EA                         JMP      FDG                       ; DO ANOTHER
0FDC                        ENTRY_1:
0FDC  E8 0DAB R                     CALL     SET_REGS
0FDF  E8 0E55 R                     CALL     BLANK                     ; CLEAR OUT THE BUFFER
0FE2  E8 0E96 R                     CALL     PH_5

                                    ASSUME   DS:ABS0
0FE5  E8 0CFE R                     CALL     DDS
0FE8  80 3E 0449 R 0F               CMP      CRT_MODE,0FH
0FED  72 06                         JB       MS_1
0FEF  C7 06 010C R 0000 E           MOV      WORD PTR GRX_SET , OFFSET CGMN
0FF5                        MS_1:
0FF5  80 3E 0449 R 07               CMP      CRT_MODE,7
0FFA  77 09                         JA       SAVE_GRPH
0FFC  74 4B                         JE       SAVE_ALPH
0FFE  80 3E 0449 R 03               CMP      CRT_MODE,3
1003  76 44                         JBE      SAVE_ALPH
1005                        SAVE_GRPH:
1005  C4 1E 04A8 R                  LES      BX,SAVE_PTR
1009  83 C3 0C                      ADD      BX,0CH
100C  26: C4 1F                     LES      BX,DWORD PTR ES:[BX]
100F  8C C0                         MOV      AX,ES
1011  0B C3                         OR       AX,BX
1013  74 32                         JZ       J4J                       ; JMP AH0_DONE
1015  BE 0007                       MOV      SI,07H
1018                        SG_1:
1018  26: 8A 00                     MOV      AL,ES:[BX][SI]
101B  3C FF                         CMP      AL,0FFH
101D  74 7A                         JE       AH0_DONE
101F  3A 06 0449 R                  CMP      AL,CRT_MODE
1023  74 03                         JE       SG_2
1025  46                            INC      SI
1026  EB F0                         JMP      SG_1
1028                        SG_2:
1028  FA                            CLI
1029  26: 8A 07                     MOV      AL,BYTE PTR ES:[BX]
102C  FE C8                         DEC      AL
102E  A2 0484 R                     MOV      ROWS,AL
1031  26: 8B 47 01                  MOV      AX,WORD PTR ES:[BX][1]
1035  A3 0485 R                     MOV      POINTS,AX
1038  26: 8B 47 03                  MOV      AX,WORD PTR ES:[BX][3]
103C  A3 010C R                     MOV      WORD PTR GRX_SET,AX
103F  26: 8B 47 05                  MOV      AX,WORD PTR ES:[BX][5]
1043  A3 010E R                     MOV      WORD PTR GRX_SET + 2,AX
1046  FB                            STI
1047                        J4J:
1047  EB 50                         JMP      SHORT AH0_DONE
1049                        SAVE_ALPH:
1049  C4 1E 04A8 R                  LES      BX,SAVE_PTR
104D  83 C3 08                      ADD      BX,08H
1050  26: C4 1F                     LES      BX,DWORD PTR ES:[BX]
1053  8C C0                         MOV      AX,ES
1055  0B C3                         OR       AX,BX
1057  74 40                         JZ       AH0_DONE
1059  BE 000B                       MOV      SI,0BH
105C                        SA_1:
105C  26: 8A 00                     MOV      AL,ES:[BX][SI]
105F  3C FF                         CMP      AL,0FFH
1061  74 36                         JE       AH0_DONE
1063  3A 06 0449 R                  CMP      AL,CRT_MODE
1067  74 03                         JE       SA_2
1069  46                            INC      SI
106A  EB F0                         JMP      SA_1
106C                        SA_2:
106C  26: 8A 27                     MOV      AH,ES:[BX]
106F  26: 8A 47 01                  MOV      AL,ES:[BX][1]
1073  26: 8B 4F 02                  MOV      CX,ES:[BX][2]
1077  26: 8B 57 04                  MOV      DX,ES:[BX][4]
107B  26: 8B 6F 06                  MOV      BP,ES:[BX][6]
107F  26: 8E 47 08                  MOV      ES,ES:[BX][8]
1083  53                            PUSH     BX
1084  8B D8                         MOV      BX,AX
1086  B8 1110                       MOV      AX,1110H
1089  CD 10                         INT      10H
108B  5B                            POP      BX
108C  26: 8A 47 0A                  MOV      AL,ES:[BX][0AH]
1090  3C FF                         CMP      AL,0FFH
1092  74 05                         JE       AH0_DONE
1094  FE C8                         DEC      AL
1096  A2 0484 R                     MOV      ROWS,AL

                            ;----- SET THE LOW RAM VALUES FOR COMPATIBILITY (3D8 AND 3D9 SAVE BYTES)

1099                        AH0_DONE:
1099  E8 0CFE R                     CALL     DDS
109C  80 3E 0449 R 07               CMP      CRT_MODE,7
10A1  77 1E                         JA       DNDCS
10A3  BB 10C8 R                     MOV      BX,OFFSET COMPAT_MODE
10A6  A0 0449 R                     MOV      AL,CRT_MODE
10A9  2A E4                         SUB      AH,AH
10AB  03 D8                         ADD      BX,AX
10AD  2E: 8A 07                     MOV      AL,CS:[BX]
10B0  A2 0465 R                     MOV      CRT_MODE_SET,AL
10B3  B0 30                         MOV      AL,030H
10B5  80 3E 0449 R 06               CMP      CRT_MODE,6
10BA  75 02                         JNE      DO_PAL
10BC  B0 3F                         MOV      AL,03FH
10BE                        DO_PAL:
10BE  A2 0466 R                     MOV      CRT_PALETTE,AL
```

```
10C1                       DNDCS:
10C1  8B 0E 0460 R                   MOV     CX,CURSOR_MODE
10C5  EB 28 90                       JMP     AH1

10C8                       COMPAT_MODE       LABEL    BYTE
10C8  2C 28 2D 29 2A 2E              DB      02CH,028H,02DH,029H,02AH,02EH
10CE  1E 29                          DB      01EH,029H

                        C            INCLUDE V1-5.INC
                        C            SUBTTL  V1-5.INC
                        C            PAGE
                        C
10D0                    C  CALC_CURSOR       PROC     NEAR
                        C            ASSUME  DS:ABS0
10D0  80 FD 00          C            CMP     CH,0                     ; CHECK FOR FULL HEIGHT
10D3  75 04             C            JNE     CC_1                     ; NORMAL CHECK
10D5  FE C1             C            INC     CL                       ; ADJUST END VALUE
10D7  EB 0A             C            JMP     SHORT   CALC_OUT
10D9                    C  CC_1:
10D9  FE C1             C            INC     CL                       ; ADJUST FOR EGA REGISTERS
10DB  3A 0E 0485 R      C            CMP     CL,BYTE PTR POINTS       ; WILL IT WRAP
10DF  72 02             C            JB      CALC_OUT                 ; NO, ITS OK
10E1  2A C9             C            SUB     CL,CL                    ; EGA METHOD FOR CURSOR END
10E3                    C  CALC_OUT:
10E3  51                C            PUSH    CX                       ; SAVE CURSOR TYPE VALUE
10E4  2A CD             C            SUB     CL,CH                    ; END - START
10E6  80 F9 10          C            CMP     CL,010H                  ; LOW NIBBLE EQUAL
10E9  59                C            POP     CX                       ; RESTORE
10EA  75 02             C            JNE     COMP_4
10EC  FE C1             C            INC     CL                       ; ADD 1 FOR CORRECT CURSOR
10EE                    C  COMP_4:
10EE  C3                C            RET                              ; BACK TO CALLER
10EF                    C  CALC_CURSOR       ENDP
                        C
                        C  ;--------------------------------------------------------------
                        C  ; SET_CTYPE        SET CURSOR TYPE                             :
                        C  ;         THIS ROUTINE SETS THE CURSOR VALUE                   :
                        C  ; INPUT                                                        :
                        C  ;         (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE    :
                        C  ; OUTPUT                                                       :
                        C  ;         NONE                                                 :
                        C  ;--------------------------------------------------------------
= 0004                  C            CUT_OFF         EQU      4
10EF                    C  AH1:
                        C            ASSUME  DS:ABS0
10EF  B4 0A             C            MOV     AH,C_CRSR_START          ; CRTC REG FOR CURSOR SET
10F1  89 0E 0460 R      C            MOV     CURSOR_MODE,CX           ; SAVE IN DATA AREA
10F5  F6 06 0487 R 08   C            TEST    INFO,8                   ; EGA ACTIVE BIT
10FA  75 33             C            JNZ     DO_SET                   ; 0=EGA, 1=OLD CARDS
                        C
                        C  ;----- THIS SECTION WILL EMULATE CURSOR OFF ON THE EGA
                        C
10FC  8A C5             C            MOV     AL,CH                    ; GET START VALUE
10FE  24 60             C            AND     AL,060H                  ; TURN OFF CURSOR ?
1100  3C 20             C            CMP     AL,020H                  ; TEST THE BITS
1102  75 05             C            JNE     AH1_A                    ; SKIP CURSOR OFF
1104  B9 1E00           C            MOV     CX,01E00H                ; EMULATE CURSOR OFF
1107  EB 26             C            JMP     SHORT DO_SET
                        C
                        C  ;----- THIS SECTION : ADJUST THE CURSOR AND TEST FOR ENHANCED OPERATION
                        C
1109                    C  AH1_A:
1109  F6 06 0487 R 01   C            TEST    INFO,1                   ; CURSOR EMULATE BIT
110E  75 1F             C            JNZ     DO_SET                   ; 0=EMULATE, 1=VALUE AS-IS
1110  80 3E 0449 R 03   C            CMP     CRT_MODE,3               ; POSSIBLE EMULATION
1115  77 15             C            JA      AH1_S                    ; NO, SET THE CURSOR TYPE
1117  E8 0E9A R         C            CALL    BRST_DET                 ; SEE IF EMULATE MODE
111A  73 10             C            JNC     AH1_S                    ; NOT EMULATING
111C  80 FD 04          C            CMP     CH,CUT_OFF               ; TEST START
111F  76 03             C            JBE     AH1_B                    ; SKIP ADJUST
1121  80 C5 05          C            ADD     CH,5                     ; ADJUST
1124                    C  AH1_B:
1124  80 F9 04          C            CMP     CL,CUT_OFF               ; TEST END
1127  76 03             C            JBE     AH1_S                    ; SKIP ADJUST
1129  80 C1 05          C            ADD     CL,5
112C                    C  AH1_S:
112C  E8 10D0 R         C            CALL    CALC_CURSOR              ; ADJUST END REGISTER
112F                    C  DO_SET:
112F  E8 1135 R         C            CALL    M16                      ; OUTPUT CX REG
1132  E9 219E R         C            JMP     V_RET                    ; RETURN TO CALLER
                        C
                        C  ;----- THIS ROUTINE OUTPUTS THE CX REGISTER TO THE CRTC REGS NAMED IN AH
                        C
1135                    C  M16:
1135  8B 16 0463 R      C            MOV     DX,ADDR_6845             ; ADDRESS REGISTER
1139  8A C5             C            MOV     AL,CH                    ; DATA
113B  E8 0D15 R         C            CALL    OUT_DX                   ; OUTPUT THE VALUE
113E  FE C4             C            INC     AH                       ; NEXT REGISTER
1140  8A C1             C            MOV     AL,CL                    ; SECOND DATA VALUE
1142  E8 0D15 R         C            CALL    OUT_DX                   ; OUTPUT THE VALUE
1145  C3                C            RET                              ; ALL DONE
                        C
                        C  ;--------------------------------------------------------------
                        C  ; POSITION                                                     :
                        C  ;         THIS SERVICE ROUTINE CALCULATES THE REGEN BUFFER     :
                        C  ;         ADDRESS OF A CHARACTER IN THE ALPHA MODE             :
                        C  ; INPUT                                                        :
                        C  ;         AX = ROW, COLUMN POSITION                            :
                        C  ; OUTPUT                                                       :
                        C  ;         AX = OFFSET OF CHAR POSITION IN REGEN BUFFER         :
                        C  ;--------------------------------------------------------------
1146                    C  POSITION          PROC     NEAR
1146  53                C            PUSH    BX                       ; SAVE REGISTER
1147  8B D8             C            MOV     BX,AX
1149  8A C4             C            MOV     AL,AH                    ; ROWS TO AL
114B  F6 26 044A R      C            MUL     BYTE PTR CRT_COLS        ; DETERMINE BYTES TO ROW
114F  32 FF             C            XOR     BH,BH                    ; ZERO OUT
1151  03 C3             C            ADD     AX,BX                    ; ADD IN COLUMN VALUE
1153  D1 E0             C            SAL     AX,1                     ; * 2 FOR ATTRIBUTE BYTES
1155  5B                C            POP     BX                       ; RESTORE REGISTER
1156  C3                C            RET
1157                    C  POSITION          ENDP
                        C
                        C  ;--------------------------------------------------------------
                        C  ; SET_CPOS         SET CURSOR POSITION                         :
                        C  ;         THIS ROUTINE SETS THE CURRENT CURSOR POSITION TO THE :
                        C  ;         NEW X-Y VALUES PASSED                                :
                        C  ; INPUT                                                        :
                        C  ;         DX - ROW,COLUMN OF NEW CURSOR                        :
                        C  ;         BH - DISPLAY PAGE OF CURSOR                          :
                        C  ; OUTPUT                                                       :
                        C  ;         CURSOR IS SET AT CRTC IF DISPLAY PAGE IS CURRENT     :
                        C  ;         DISPLAY                                              :
                        C  ;--------------------------------------------------------------
1157                    C  AH2:
1157  E8 115D R         C            CALL    SET_CPOS
```

```
115A  E9 219E R          C              JMP     V_RET
                         C
115D                     C      SET_CPOS:
115D  8A CF              C              MOV     CL,BH
115F  32 ED              C              XOR     CH,CH                           ; ESTABLISH LOOP COUNT
1161  D1 E1              C              SAL     CX,1                            ; WORD OFFSET
1163  8B F1              C              MOV     SI,CX                           ; USE INDEX REGISTER
1165  89 94 0450 R       C              MOV     [SI+OFFSET CURSOR_POSN],DX      ; SAVE THE POINTER
1169  38 3E 0462 R       C              CMP     ACTIVE_PAGE,BH
116D  75 05              C              JNZ     M17                             ; SET_CPOS_RETURN
116F  8B C2              C              MOV     AX,DX                           ; GET ROW/COLUMN TO AX
1171  E8 1175 R          C              CALL    M18                             ; CURSOR_SET
1174                     C      M17:                                            ; SET_CPOS_RETURN
1174  C3                 C              RET
                         C
                         C      ;----- SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR
                         C
1175                     C      M18     PROC    NEAR
1175  E8 1146 R          C              CALL    POSITION                        ; DETERMINE LOC IN REGEN
1178  8B C8              C              MOV     CX,AX
117A  03 0E 044E R       C              ADD     CX,CRT_START                    ; ADD IN THE START ADDR
                         C                                                      ;  FOR THIS PAGE
117E  D1 F9              C              SAR     CX,1                            ; / 2 FOR CHAR ONLY COUNT
1180  B4 0E              C              MOV     AH,C_CRSR_LOC_HGH               ; REGISTER NUMBER FOR CURSOF
1182  E8 1135 R          C              CALL    M16                             ; SET VALUE TO CRTC
1185  C3                 C              RET
1186                     C      M18     ENDP
                         C
                         C      ;--------------------------------------------------------------
                         C      ; READ_CURSOR                                                  :
                         C      ;       THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM       :
                         C      ;       MEMORY AND SENDS IT BACK TO THE CALLER                 :
                         C      ; INPUT                                                        :
                         C      ;       BH - PAGE OF CURSOR                                    :
                         C      ; OUTPUT                                                       :
                         C      ;       DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION        :
                         C      ;       CX - CURRENT CURSOR MODE                               :
                         C      ;--------------------------------------------------------------
1186                     C      AH3:
1186  8A DF              C              MOV     BL,BH                           ; PAGE VALUE
1188  32 FF              C              XOR     BH,BH                           ; ZERO UPPER BYTE
118A  D1 E3              C              SAL     BX,1                            ; WORD OFFSET
118C  8B 97 0450 R       C              MOV     DX,[BX + OFFSET CURSOR_POSN]    ; GET CURSOR FOR THIS PAGE
1190  8B 0E 0460 R       C              MOV     CX,CURSOR_MODE                  ; GET THE CURSOR MODE
1194  5F                 C              POP     DI
1195  5E                 C              POP     SI
1196  5B                 C              POP     BX
1197  58                 C              POP     AX                              ; DISCARD CX
1198  58                 C              POP     AX                              ; DISCARD DX
1199  1F                 C              POP     DS
119A  07                 C              POP     ES
119B  5D                 C              POP     BP
119C  CF                 C              IRET
                         C
                         C      ;----- READ LIGHT PEN POSITION
                         C
119D                     C      AH4:
119D  A0 0449 R          C              MOV     AL,CRT_MODE
11A0  3C 07              C              CMP     AL,07H
11A2  77 37              C              JA      READ_LPEN
                         C
11A4  F6 06 0487 R 02    C              TEST    INFO,2
11A9  74 07              C              JZ      EGA_IS_COLOR
                         C
                         C      ;----- MONOCHROME HERE (MONOC BIT 1)
                         C
11AB  3C 07              C              CMP     AL,07H
11AD  74 2C              C              JE      READ_LPEN
11AF  EB 05 90           C              JMP     OLD_LP
                         C
                         C      ;----- EGA IS COLOR HERE (MONOC BIT 0)
                         C
11B2                     C      EGA_IS_COLOR:
11B2  3C 06              C              CMP     AL,06H
11B4  76 25              C              JBE     READ_LPEN
11B6                     C      OLD_LP:
11B6  CD 42              C              INT     42H                             ; CALL EXISTING CODE
11B8  5F                 C              POP     DI
11B9  5E                 C              POP     SI
11BA  83 C4 06           C              ADD     SP,6                            ; DISCARD SAVED BX,CX,DX
11BD  1F                 C              POP     DS
11BE  07                 C              POP     ES
11BF  5D                 C              POP     BP
11C0  CF                 C              IRET
                         C
                         C      ;--------------------------------------------------------------
                         C      ; LIGHT PEN                                                    :
                         C      ;       THIS ROUTINE TESTS THE LIGHT PEN SWITCH AND THE LIGHT  :
                         C      ;       PEN TRIGGER. IF BOTH ARE SET, THE LOCATION OF THE LIGHT :
                         C      ;       PEN IS DETERMINED. OTHERWISE, A RETURN WITH NO         :
                         C      ;       INFORMATION IS MADE.                                   :
                         C      ; ON EXIT                                                      :
                         C      ;       (AH) = 0 IF NO LIGHT PEN INFORMATION IS AVAILABLE      :
                         C      ;                BX,CX,DX ARE DESTROYED                        :
                         C      ;       (AH) = 1 IF LIGHT PEN IS AVAILABLE                     :
                         C      ;                (DH,DL) = ROW,COLUMN OF CURRENT LIGHT PEN     :
                         C      ;                          POSITION                            :
                         C      ;                (CH) = RASTER POSITION (OLD MODES)            :
                         C      ;                (CX) = RASTER POSITION (NEW MODES)            :
                         C      ;                (BX) = BEST GUESS AT PIXEL HORIZONTAL POSITION :
                         C      ;--------------------------------------------------------------
                         C              ASSUME  CS:CODE,DS:ABS0
                         C      ;----- SUBTRACT_TABLE
11C1                     C      V1      LABEL   BYTE
11C1  06 06 07 07 05 05  C              DB      006H,006H,007H,007H,005H,005H   ; 0-5
11C7  04 05 00 00 00 00  C              DB      004H,005H,000H,000H,000H,000H   ; 6-B
11CD  00 05 06 04 04 04  C              DB      000H,005H,006H,004H,004H,004H   ; C-11
11D3  04 06 06 04 07 04  C              DB      004H,006H,006H,004H,007H,004H   ; 12-17
11D9  07 04              C              DB      007H,004H                       ; 18-19
                         C
11DB                     C      READ_LPEN       PROC    NEAR
                         C
                         C      ;----- WAIT FOR LIGHT PEN TO BE DEPRESSED
                         C
11DB  8B 16 0463 R       C              MOV     DX,ADDR_6845                    ; GET BASE ADDRESS OF 6845
11DF  83 C2 06           C              ADD     DX,6                            ; POINT TO STATUS REGISTER
11E2  EC                 C              IN      AL,DX                           ; GET STATUS REGISTER
11E3  A8 04              C              TEST    AL,4                            ; TEST LIGHT PEN SWITCH
11E5  B4 00              C              MOV     AH,0                            ; SET NO LIGHT PEN RETURN
11E7  74 03              C              JZ      V9                              ;  CODE
11E9  E9 1291 R          C              JMP     V6                              ; NOT SET, RETURN
                         C
                         C      ;----- NOW TEST FOR LIGHT PEN TRIGGER
                         C
11EC                     C      V9:
11EC  A8 02              C              TEST    AL,2                            ; TEST LIGHT PEN TRIGGER
```

```
11EE  75 03                   C          JNZ     V7A                 ; RETURN WITHOUT RESETTING
                              C                                      ;  TRIGGER
11F0  E9 129B R               C          JMP     V7                  ; EXIT LIGHT PEN ROUTINE
                              C
                              C  ;----- TRIGGER HAS BEEN SET, READ THE VALUE IN
                              C
11F3                          C  V7A:
11F3  B4 10                   C          MOV     AH,16               ; LIGHT PEN REGISTERS
                              C
                              C  ;----- INPUT REGS POINTED TO BY AH, AND CONVERT TO ROW COLUMN IN DX
                              C
11F5  8B 16 0463 R            C          MOV     DX,ADDR_6845        ; ADDRESS REGISTER
11F9  8A C4                   C          MOV     AL,AH               ; REGISTER TO READ
11FB  EE                      C          OUT     DX,AL               ; SET IT UP
11FC  42                      C          INC     DX                  ; DATA REGISTER
11FD  50                      C          PUSH    AX
11FE  EC                      C          IN      AL,DX               ; GET THE VALUE
11FF  8A E8                   C          MOV     CH,AL               ; SAVE IN CX
1201  58                      C          POP     AX
1202  4A                      C          DEC     DX                  ; ADDRESS REGISTER
1203  FE C4                   C          INC     AH
1205  8A C4                   C          MOV     AL,AH               ; SECOND DATA REGISTER
1207  EE                      C          OUT     DX,AL
1208  42                      C          INC     DX                  ; POINT TO DATA REGISTER
1209  EC                      C          IN      AL,DX               ; GET THE 2ND DATA VALUE
120A  8A E5                   C          MOV     AH,CH               ; AX HAS INPUT VALUE
                              C
                              C  ;----- AX HAS THE VALUE READ IN FROM THE 6845
                              C
120C  8A 1E 0449 R            C          MOV     BL,CRT_MODE
1210  2A FF                   C          SUB     BH,BH               ; MODE VALUE TO BX
1212  2E: 8A 9F 11C1 R        C          MOV     BL,CS:V1[BX]        ; AMOUNT TO SUBTRACT
1217  2B C3                   C          SUB     AX,BX               ; TAKE IT AWAY
1219  8B 1E 044E R            C          MOV     BX,CRT_START        ; SCREEN ADDRESS
121D  D1 EB                   C          SHR     BX,1                ; DIVIDE BY 2
121F  2B C3                   C          SUB     AX,BX               ; ADJUST TO ZERO START
1221  79 02                   C          JNS     V2                  ; IF POSITIVE, GET MODE
1223  2B C0                   C          SUB     AX,AX               ; <0 PLAYS AS 0
                              C
                              C  ;----- DETERMINE MODE OF OPERATION
                              C
1225                          C  V2:                                 ; DETERMINE_MODE
1225  B1 03                   C          MOV     CL,3                ; SET *8 SHIFT COUNT
1227  80 3E 0449 R 04         C          CMP     CRT_MODE,4          ; GRAPHICS OR ALPHA
122C  72 4D                   C          JB      V4                  ; ALPHA_PEN
122E  80 3E 0449 R 07         C          CMP     CRT_MODE,7
1233  74 46                   C          JE      V4                  ; ALPHA_PEN
                              C
1235  80 3E 0449 R 06         C          CMP     CRT_MODE,06H
123A  77 28                   C          JA      V8
123C  75 02                   C          JNE     V8X
123E  D1 E8                   C          SHR     AX,1
                              C
                              C  ;----- OLD GRAPHICS MODES
                              C
1240                          C  V8X:
1240  B2 28                   C          MOV     DL,40               ; DIVISOR FOR GRAPHICS
1242  F6 F2                   C          DIV     DL                  ; ROW(AL) AND COLUMN(AH)
                              C                                      ;  AL RANGE 0-99,
                              C                                      ;  AH RANGE 0-39
                              C  ;----- DETERMINE GRAPHIC ROW POSITION
                              C
1244  8A E8                   C          MOV     CH,AL               ; SAVE ROW VALUE IN CH
1246  02 ED                   C          ADD     CH,CH               ; *2 FOR EVEN/ODD FIELD
1248  8A DC                   C          MOV     BL,AH               ; COLUMN VALUE TO BX
124A  2A FF                   C          SUB     BH,BH               ; *8 FOR MEDIUM RES
124C  80 3E 0449 R 06         C          CMP     CRT_MODE,6          ; MEDIUM OR HIGH RES
1251  75 04                   C          JNE     V3                  ; NOT_HIGH_RES
1253  B1 04                   C          MOV     CL,4                ; SHIFT VALUE FOR HIGH RES
1255  D0 E4                   C          SAL     AH,1                ; COLUMN VALUE *2 FOR HIGH RES
1257                          C  V3:                                 ; NOT_HIGH_RES
1257  D3 E3                   C          SHL     BX,CL               ; *16 FOR HIGH RES
                              C
                              C  ;----- DETERMINE ALPHA CHAR POSITION
                              C
1259  8A D4                   C          MOV     DL,AH               ; COLUMN VALUE FOR RETURN
125B  8A F0                   C          MOV     DH,AL               ; ROW VALUE
125D  D0 EE                   C          SHR     DH,1                ; DIVIDE BY 4
125F  D0 EE                   C          SHR     DH,1                ;  FOR VALUE IN 0-24 RANGE
1261  EB 2C 90                C          JMP     V5                  ; LIGHT_PEN_RETURN_SET
1264                          C  V8:
                              C
                              C  ;----- NEW GRAPHICS MODES
                              C
1264  99                      C          CWD                         ; PREPARE TO DIVIDE
1265  F7 36 044A R            C          DIV     CRT_COLS            ; AX = ROW, DX = COLUMN
1269  8B DA                   C          MOV     BX,DX               ; SAVE REMAINDER
126B  D3 E3                   C          SAL     BX,CL               ; PEL COLUMN
126D  8B C8                   C          MOV     CX,AX               ; PEL ROW
126F  52                      C          PUSH    DX                  ; SAVE FROM DIVIDE
1270  99                      C          CWD                         ; PREPARE TO DIVIDE
1271  F7 36 0485 R            C          DIV     POINTS              ; DIVIDE BY BYTES/CHAR
1275  5A                      C          POP     DX                  ; RECOVER
1276  8A F0                   C          MOV     DH,AL               ; CHARACTER ROW
1278  EB 15 90                C          JMP     V5
                              C
                              C  ;----- ALPHA MODE ON LIGHT PEN
                              C
127B                          C  V4:                                 ; ALPHA_PEN
127B  F6 36 044A R            C          DIV     BYTE PTR CRT_COLS   ; ROW,COLUMN VALUE
127F  8A F0                   C          MOV     DH,AL               ; ROWS TO DH
1281  8A D4                   C          MOV     DL,AH               ; COLS TO DL
1283  8A DC                   C          MOV     BL,AH               ; COLUMN VALUE
1285  32 FF                   C          XOR     BH,BH               ;  TO BX
1287  D3 E3                   C          SAL     BX,CL
1289  F6 26 0485 R            C          MUL     BYTE PTR POINTS
128D  8B C8                   C          MOV     CX,AX
128F                          C  V5:                                 ; LIGHT_PEN_RETURN_SET
128F  B4 01                   C          MOV     AH,1                ; INDICATE EVERTHING SET
1291                          C  V6:                                 ; LIGHT_PEN_RETURN
1291  52                      C          PUSH    DX                  ; SAVE RETURN VALUE
                              C                                      ;  (IN CASE)
1292  8B 16 0463 R            C          MOV     DX,ADDR_6845        ; GET BASE ADDRESS
1296  83 C2 07                C          ADD     DX,7                ; POINT TO RESET PARM
1299  EE                      C          OUT     DX,AL               ; ADDRESS, NOT DATA,
                              C                                      ;  IS IMPORTANT
129A  5A                      C          POP     DX                  ; RECOVER VALUE
129B                          C  V7:                                 ; RETURN_NO_RESET
129B  5F                      C          POP     DI
129C  5E                      C          POP     SI
129D  83 C4 06                C          ADD     SP,6                ; DISCARD SAVED BX,CX,DX
12A0  1F                      C          POP     DS
12A1  07                      C          POP     ES
12A2  5D                      C          POP     BP
12A3  CF                      C          IRET
12A4                          C  READ_LPEN       ENDP
```

```
                          C
                          C  ;--------------------------------------------------------------
                          C  ; ACT_DISP_PAGE           SELECT ACTIVE DISPLAY PAGE          :
                          C  ;        THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING  :
                          C  ;        FOR MULTIPLE PAGES OF DISPLAYED VIDEO.               :
                          C  ; INPUT                                                       :
                          C  ;        AL HAS THE NEW ACTIVE DISPLAY PAGE                   :
                          C  ; OUTPUT                                                      :
                          C  ;        THE CRTC IS RESET TO DISPLAY THAT PAGE               :
                          C  ;--------------------------------------------------------------
12A4                      C  AH5:
12A4   A2 0462 R          C          MOV     ACTIVE_PAGE,AL                  ; SAVE ACTIVE PAGE VALUE
12A7   8B 0E 044C R       C          MOV     CX,CRT_LEN                      ; GET SAVED LENGTH OF
                          C                                                  ;  REGEN BUFFER
12AB   98                 C          CBW                                     ; CONVERT AL TO WORD
12AC   50                 C          PUSH    AX                              ; SAVE PAGE VALUE
12AD   F7 E1              C          MUL     CX                              ; DISPLAY PAGE TIMES
                          C                                                  ;  REGEN LENGTH
12AF   A3 044E R          C          MOV     CRT_START,AX                    ; SAVE START ADDRESS FOR
                          C                                                  ;  LATER REQUIREMENTS
12B2   8B C8              C          MOV     CX,AX                           ; START ADDRESS TO CX
12B4   8A 1E 0449 R       C          MOV     BL,CRT_MODE
12B8   80 FB 07           C          CMP     BL,7                            ; DO NOT DIVIDE BY TWO
12BB   77 02              C          JA      ADP_1
12BD                      C  ADP_2:
12BD   D1 F9              C          SAR     CX,1                            ; / 2 FOR CRTC HANDLING
12BF                      C  ADP_1:
12BF   B4 0C              C          MOV     AH,C_STRT_HGH                   ; REG FOR START ADDRESS
12C1   E8 1135 R          C          CALL    M16
12C4   5B                 C          POP     BX                              ; RECOVER PAGE VALUE
12C5   D1 E3              C          SAL     BX,1                            ; *2 FOR WORD OFFSET
12C7   8B 87 0450 R       C          MOV     AX,[BX + OFFSET CURSOR_POSN]    ; GET CURSOR FOR THIS PAGE
12CB   E8 1175 R          C          CALL    M18                             ; SET THE CURSOR POSITION
12CE   E9 219E R          C          JMP     V_RET
                          C
                          C          SUBTTL

                          C          INCLUDE VSCROLL.INC
                          C          SUBTTL  VSCROLL.INC
                          C          PAGE
                          C
12D1                      C  FLTA    PROC    NEAR                            ; CHECK FOR SCROLL COUNT
12D1   50                 C          PUSH    AX
12D2   8A E6              C          MOV     AH,DH                           ; LOWER ROW
12D4   2A E5              C          SUB     AH,CH                           ; UPPER ROW
12D6   FE C4              C          INC     AH                              ; NUMBER TO SCROLL
12D8   3A E0              C          CMP     AH,AL                           ; SAME AS REQUESTED
12DA   58                 C          POP     AX
12DB   75 02              C          JNE     LTA
12DD   2A C0              C          SUB     AL,AL                           ; YES, SET TO 0 FOR BLANK
12DF                      C  LTA:
12DF   C3                 C          RET
12E0                      C  FLTA    ENDP
                          C
12E0                      C  CRANK   PROC    NEAR                            ; MOVE ROWS OF PELS UP
12E0   53                 C          PUSH    BX
                          C          ASSUME  DS:ABSO
12E1   1E                 C          PUSH    DS                              ; SAVE DATA SEGMENT
12E2   E8 0CFE R          C          CALL    DDS                             ; SET DATA SEGMENT
12E5   8B 1E 044A R       C          MOV     BX,CRT_COLS
12E9   1F                 C          POP     DS
12EA                      C  CRANK_A:
12EA   51                 C          PUSH    CX                              ; SAVE MOVE COUNT
12EB   8A CA              C          MOV     CL,DL                           ; COLUMN COUNT
12ED   2A ED              C          SUB     CH,CH                           ; CLEAR HIGH BYTE
12EF   56                 C          PUSH    SI                              ; SAVE POINTERS
12F0   57                 C          PUSH    DI
12F1   F3/ A4             C          REP     MOVSB                           ; MOVE THAT ROW
12F3   5F                 C          POP     DI                              ; RECOVER POINTERS
12F4   5E                 C          POP     SI
12F5   03 F3              C          ADD     SI,BX                           ; NEXT ROW
12F7   03 FB              C          ADD     DI,BX                           ; NEXT ROW
12F9   59                 C          POP     CX                              ; RECOVER ROW COUNT
12FA   E2 EE              C          LOOP    CRANK_A                         ; DO MORE
12FC   5B                 C          POP     BX
12FD   C3                 C          RET                                     ; RETURN TO CALLER
12FE                      C  CRANK   ENDP
                          C
12FE                      C  CRANK_4 PROC    NEAR                            ; MOVE ROWS OF PELS DOWN
12FE   53                 C          PUSH    BX
                          C          ASSUME  DS:ABSO
12FF   1E                 C          PUSH    DS                              ; SAVE DATA SEGMENT
1300   E8 0CFE R          C          CALL    DDS                             ; SET DATA SEGMENT
1303   8B 1E 044A R       C          MOV     BX,CRT_COLS
1307   1F                 C          POP     DS
1308                      C  CRANK_B:
1308   51                 C          PUSH    CX                              ; SAVE MOVE COUNT
1309   8A CA              C          MOV     CL,DL                           ; COLUMN COUNT
130B   2A ED              C          SUB     CH,CH                           ; CLEAR HIGH BYTE
130D   56                 C          PUSH    SI                              ; SAVE POINTERS
130E   57                 C          PUSH    DI
130F   F3/ A4             C          REP     MOVSB                           ; MOVE THAT ROW
1311   5F                 C          POP     DI                              ; RECOVER POINTERS
1312   5E                 C          POP     SI
1313   2B F3              C          SUB     SI,BX                           ; NEXT ROW
1315   2B FB              C          SUB     DI,BX                           ; NEXT ROW
1317   59                 C          POP     CX                              ; RECOVER ROW COUNT
1318   E2 EE              C          LOOP    CRANK_B                         ; DO MORE
131A   5B                 C          POP     BX
131B   C3                 C          RET                                     ; RETURN TO CALLER
131C                      C  CRANK_4 ENDP
                          C
131C                      C  PART_1  PROC    NEAR                            ; FILL ROW AFTER SCROLL
131C   52                 C          PUSH    DX
131D   B6 03              C          MOV     DH,3
131F   B2 C4              C          MOV     DL,SEQ_ADDR                     ; SEQUENCER
1321   B8 020F            C          MOV     AX,020FH                        ; MAP MASK
1324   E8 0D15 R          C          CALL    OUT_DX                          ; ALL MAPS ON
1327   5A                 C          POP     DX
1328   2B C0              C          SUB     AX,AX                           ; ZERO
132A   8A CA              C          MOV     CL,DL                           ; COLUMN COUNT
132C   2A ED              C          SUB     CH,CH
132E   57                 C          PUSH    DI                              ; SAVE POINTER
132F   F3/ AA             C          REP     STOSB                           ; CLEAR ONE ROW OF PELS
1331   5F                 C          POP     DI                              ; RECOVER POINTER
1332   8A C6              C          MOV     AL,DH                           ; GET COLOR VALUE
1334   52                 C          PUSH    DX
1335   B6 03              C          MOV     DH,3
1337   B2 C4              C          MOV     DL,SEQ_ADDR                     ; SEQUENCER
1339   B4 02              C          MOV     AH,02H                          ; MAP MASK
133B   E8 0D15 R          C          CALL    OUT_DX                          ; SET THE COLOR
133E   5A                 C          POP     DX
133F   B0 FF              C          MOV     AL,0FFH                         ; ALL BITS ON
1341   8A CA              C          MOV     CL,DL                           ; COLUMN COUNT
1343   57                 C          PUSH    DI                              ; SAVE POINTER
1344   F3/ AA             C          REP     STOSB                           ; TURN ON THOSE BITS IN
```

```
                          C                                         ;  ENABLED PLANES
1346  5F                  C                POP     DI               ; RECOVER POINTER
1347  C3                  C                RET                      ; RETURN TO CALLER
1348                      C     PART_1     ENDP
                          C
1348                      C     PART_2     PROC    NEAR
1348  B6 03               C                MOV     DH,3
134A  B2 C4               C                MOV     DL,SEQ_ADDR      ; SEQUENCER
134C  B8 020F             C                MOV     AX,020FH         ; MAP MASK, ALL MAPS
134F  E8 0D15 R           C                CALL    OUT_DX           ; ENABLE THE MAPS
1352  C3                  C                RET                      ; RETURN TO CALLER
1353                      C     PART_2     ENDP
                          C
1353                      C     BLNK_3     PROC    NEAR             ; BLANK FOR SCROLL UP
1353  1E                  C                PUSH    DS               ; SAVE DATA SEGMENT
                          C                ASSUME  DS:ABSO
1354  E8 0CFE R           C                CALL    DDS              ; GET LOW MEMORY SEGMENT
1357  8A F7               C                MOV     DH,BH            ; ATTRIBUTE FOR BLANK LINE
1359  2A FF               C                SUB     BH,BH            ; CLEAR HIGH BYTE
135B  50                  C                PUSH    AX               ; SAVE
135C  52                  C                PUSH    DX               ; SAVE BECAUSE OF MULTIPLY
135D  8B C3               C                MOV     AX,BX            ; ROW COUNT
135F  F7 26 0485 R        C                MUL     POINTS           ; CHARACTER HEIGHT
1363  8B D8               C                MOV     BX,AX            ; NET VALUE TO BX
1365  5A                  C                POP     DX               ; RECOVER
1366  58                  C                POP     AX
                          C
1367  1F                  C                POP     DS
                          C                ASSUME  DS:NOTHING
1368                      C     S13:
1368  E8 131C R           C                CALL    PART_1           ; BLANK OUT ROW WITH COLOR
                          C                ASSUME  DS:ABSO
136B  1E                  C                PUSH    DS               ; SAVE SEGMENT
136C  E8 0CFE R           C                CALL    DDS              ; LOW MEMORY SEGMENT
136F  03 3E 044A R        C                ADD     DI,CRT_COLS      ; NEXT ROW
1373  1F                  C                POP     DS               ; RECOVER
1374  4B                  C                DEC     BX               ; NEXT
1375  75 F1               C                JNZ     S13              ; DO MORE
1377  E8 1348 R           C                CALL    PART_2
137A  C3                  C                RET                      ; RETURN TO CALLER
137B                      C     BLNK_3     ENDP
                          C
137B                      C     BLNK_4     PROC    NEAR             ; BLANK FOR SCROLL DOWN
137B  1E                  C                PUSH    DS               ; SAVE DATA SEGMENT
                          C                ASSUME  DS:ABSO
137C  E8 0CFE R           C                CALL    DDS              ; GET LOW MEMORY SEGMENT
137F  8A F7               C                MOV     DH,BH            ; ATTRIBUTE FOR BLANK LINE
1381  2A FF               C                SUB     BH,BH            ; CLEAR HIGH BYTE
1383  50                  C                PUSH    AX               ; SAVE
1384  52                  C                PUSH    DX               ; SAVE BECAUSE OF MULTIPLY
1385  8B C3               C                MOV     AX,BX            ; ROW COUNT
1387  F7 26 0485 R        C                MUL     POINTS           ; CHARACTER HEIGHT
138B  8B D8               C                MOV     BX,AX            ; NET VALUE TO BX
138D  5A                  C                POP     DX               ; RECOVER
138E  58                  C                POP     AX
                          C
138F  1F                  C                POP     DS
                          C                ASSUME  DS:NOTHING
1390                      C     S13_4:
1390  E8 131C R           C                CALL    PART_1           ; BLANK OUT ROW WITH COLOR
                          C                ASSUME  DS:ABSO
1393  1E                  C                PUSH    DS               ; SAVE SEGMENT
1394  E8 0CFE R           C                CALL    DDS              ; LOW MEMORY SEGMENT
1397  2B 3E 044A R        C                SUB     DI,CRT_COLS      ; NEXT ROW
139B  1F                  C                POP     DS               ; RECOVER
139C  4B                  C                DEC     BX               ; NEXT
139D  75 F1               C                JNZ     S13_4            ; DO MORE
139F  E8 1348 R           C                CALL    PART_2
13A2  C3                  C                RET                      ; RETURN TO CALLER
13A3                      C     BLNK_4     ENDP
                          C
                          C     ;--------------------------------------------------------
                          C     ; SCROLL UP                                              :
                          C     ;          THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP   :
                          C     ;          ON THE SCREEN                                 :
                          C     ; INPUT                                                  :
                          C     ;          (AH) = CURRENT CRT MODE                       :
                          C     ;          (AL) = NUMBER OF ROWS TO SCROLL               :
                          C     ;          (CX) = ROW/COLUMN OF UPPER LEFT CORNER        :
                          C     ;          (DX) = ROW/COLUMN OF LOWER RIGHT CORNER       :
                          C     ;          (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE   :
                          C     ;          (DS) = DATA SEGMENT                           :
                          C     ;          (ES) = REGEN BUFFER SEGMENT                   :
                          C     ; OUTPUT                                                 :
                          C     ;          NONE -- THE REGEN BUFFER IS MODIFIED          :
                          C     ;--------------------------------------------------------
                          C                ASSUME  CS:CODE,DS:ABSO,ES:NOTHING
13A3                      C     SCROLL_UP          PROC    NEAR
13A3  8A D8               C                MOV     BL,AL            ; SAVE LINE COUNT IN BL
13A5  E8 16EB R           C                CALL    MK_ES
13A8  80 FC 04            C                CMP     AH,4             ; TEST FOR GRAPHICS MODE
13AB  72 08               C                JB      N1               ; HANDLE SEPERATELY
13AD  80 FC 07            C                CMP     AH,7             ; TEST FOR BW CARD
13B0  74 03               C                JE      N1
13B2  E9 1474 R           C                JMP     GRAPHICS_UP
13B5                      C     N1:                                 ; UP_CONTINUE
13B5  53                  C                PUSH    BX               ; SAVE FILL ATTR IN BH
13B6  8B C1               C                MOV     AX,CX            ; UPPER LEFT POSITION
13B8  E8 13F2 R           C                CALL    SCROLL_POSITION  ; DO SETUP FOR SCROLL
13BB  74 31               C                JZ      N7               ; BLANK_FIELD
13BD  03 F0               C                ADD     SI,AX            ; FROM ADDRESS
13BF  8A E6               C                MOV     AH,DH            ; # ROWS IN BLOCK
13C1  2A E3               C                SUB     AH,BL            ; # ROWS TO BE MOVED
13C3                      C     N2:                                 ; ROW_LOOP
13C3  E8 1432 R           C                CALL    N10              ; MOVE ONE ROW
13C6  03 F5               C                ADD     SI,BP
13C8  03 FD               C                ADD     DI,BP            ; NEXT LINE IN BLOCK
13CA  FE CC               C                DEC     AH               ; COUNT OF LINES TO MOVE
13CC  75 F5               C                JNZ     N2               ; ROW_LOOP
13CE                      C     N3:                                 ; CLEAR_ENTRY
13CE  58                  C                POP     AX               ; RECOVER ATTRIBUTE IN AH
13CF  B0 20               C                MOV     AL,' '           ; FILL WITH BLANKS
13D1                      C     N4:                                 ; CLEAR_LOOP
13D1  E8 143B R           C                CALL    N11              ; CLEAR THE ROW
13D4  03 FD               C                ADD     DI,BP            ; POINT TO NEXT LINE
13D6  FE CB               C                DEC     BL               ; LINES TO SCROLL
13D8  75 F7               C                JNZ     N4               ; CLEAR_LOOP
13DA                      C     N5:                                 ; SCROLL_END
13DA  E8 0CFE R           C                CALL    DDS
13DD  80 3E 0449 R 07     C                CMP     CRT_MODE,7       ; IS THIS THE B/W CARD
13E2  74 07               C                JE      N6               ; SKIP THE MODE RESET
13E4  A0 0465 R           C                MOV     AL,CRT_MODE_SET  ; GET THE MODE SET
13E7  BA 03D8             C                MOV     DX,03D8H         ; ALWAYS SET COLOR CARD
13EA  EE                  C                OUT     DX,AL
13EB                      C     N6:                                 ; VIDEO_RET_HERE
13EB  E9 219E R           C                JMP     V_RET
```

```
13EE                  C  N7:                                       ; BLANK_FIELD
13EE  8A DE           C          MOV     BL,DH                     ; GET ROW COUNT
13F0  EB DC           C          JMP     N3                        ; GO CLEAR THAT AREA
13F2                  C  SCROLL_UP       ENDP
                      C
                      C  ;----- HANDLE COMMON SCROLL SET UP HERE
                      C
13F2                  C  SCROLL_POSITION PROC    NEAR
13F2  F6 06 0487 R 04 C          TEST    INFO,4
13F7  74 12           C          JZ      N9
                      C
                      C  ;----- 80X25 COLOR CARD SCROLL
                      C
13F9  52              C          PUSH    DX
13FA  B6 03           C          MOV     DH,3
13FC  B2 DA           C          MOV     DL,0DAH                   ; COLOR CARD HERE
13FE  50              C          PUSH    AX
13FF                  C  N8:                                       ; WAIT_DISP_ENABLE
13FF  EC              C          IN      AL,DX
1400  A8 08           C          TEST    AL,8                      ; WAIT FOR VERT RETRACE
1402  74 FB           C          JZ      N8                        ; WAIT_DISP_ENABLE
1404  B0 25           C          MOV     AL,25H
1406  B2 D8           C          MOV     DL,0D8H                   ; DX=3D8
1408  EE              C          OUT     DX,AL                     ; TURN OFF VIDEO
1409  58              C          POP     AX                        ;  DURING VERTICAL RETRACE
140A  5A              C          POP     DX
140B                  C  N9:
140B  E8 1146 R       C          CALL    POSITION                  ; CONVERT TO REGEN POINTER
140E  03 06 044E R    C          ADD     AX,CRT_START              ; OFFSET OF ACTIVE PAGE
1412  8B F8           C          MOV     DI,AX                     ; TO ADDRESS FOR SCROLL
1414  8B F0           C          MOV     SI,AX                     ; FROM ADDRESS FOR SCROLL
1416  2B D1           C          SUB     DX,CX                     ; DX = #ROWS, #COLS
1418  FE C6           C          INC     DH
141A  FE C2           C          INC     DL                        ; INCREMENT FOR 0 ORIGIN
141C  32 ED           C          XOR     CH,CH                     ; ZERO HIGH BYTE OF COUNT
141E  8B 2E 044A R    C          MOV     BP,CRT_COLS               ; NUM OF COLS IN DISPLAY
1422  03 ED           C          ADD     BP,BP                     ; TIMES 2 FOR ATTR BYTE
1424  8A C3           C          MOV     AL,BL                     ; GET LINE COUNT
1426  F6 26 044A R    C          MUL     BYTE PTR CRT_COLS         ; OFFSET TO FROM ADDRESS
142A  03 C0           C          ADD     AX,AX                     ; *2 FOR ATTRIBUTE BYTE
142C  06              C          PUSH    ES                        ; ESTABLISH ADDRESSING
142D  1F              C          POP     DS                        ;  FOR BOTH POINTERS
142E  80 FB 00        C          CMP     BL,0                      ; 0 MEANS BLANK FIELD
1431  C3              C          RET                               ; RETURN WITH FLAGS SET
1432                  C  SCROLL_POSITION ENDP
                      C
                      C  ;----- MOVE_ROW
                      C
1432                  C  N10     PROC    NEAR
1432  8A CA           C          MOV     CL,DL                     ; GET # OF COLS TO MOVE
1434  56              C          PUSH    SI
1435  57              C          PUSH    DI                        ; SAVE START ADDRESS
1436  F3/ A5          C          REP     MOVSW                     ; MOVE THAT LINE ON SCREEN
1438  5F              C          POP     DI
1439  5E              C          POP     SI                        ; RECOVER ADDRESSES
143A  C3              C          RET
143B                  C  N10     ENDP
                      C
                      C  ;----- CLEAR_ROW
                      C
143B                  C  N11     PROC    NEAR
143B  8A CA           C          MOV     CL,DL                     ; GET # COLUMNS TO CLEAR
143D  57              C          PUSH    DI
143E  F3/ AB          C          REP     STOSW                     ; STORE THE FILL CHARACTER
1440  5F              C          POP     DI
1441  C3              C          RET
1442                  C  N11     ENDP
                      C
                      C  ;--------------------------------------------------------
                      C  ; SCROLL_DOWN                                            :
                      C  ;       THIS ROUTINE MOVES THE CHARACTERS WITHIN A       :
                      C  ;       DEFINED BLOCK DOWN ON THE SCREEN, FILLING THE    :
                      C  ;       TOP LINES WITH A DEFINED CHARACTER               :
                      C  ; INPUT                                                  :
                      C  ;       (AH) = CURRENT CRT MODE                          :
                      C  ;       (AL) = NUMBER OF LINES TO SCROLL                 :
                      C  ;       (CX) = UPPER LEFT CORNER OF REGION               :
                      C  ;       (DX) = LOWER RIGHT CORNER OF REGION              :
                      C  ;       (BH) = FILL CHARACTER                            :
                      C  ;       (DS) = DATA SEGMENT                              :
                      C  ;       (ES) = REGEN SEGMENT                             :
                      C  ; OUTPUT                                                 :
                      C  ;       NONE -- SCREEN IS SCROLLED                       :
                      C  ;--------------------------------------------------------
1442                  C  SCROLL_DOWN     PROC    NEAR
1442  FD              C          STD                               ; SCROLL DOWN
1443  8A D8           C          MOV     BL,AL                     ; LINE COUNT TO BL
1445  E8 16EB R       C          CALL    MK_ES
1448  53              C          PUSH    BX                        ; SAVE ATTRIBUTE IN BH
1449  8B C2           C          MOV     AX,DX                     ; LOWER RIGHT CORNER
144B  E8 13F2 R       C          CALL    SCROLL_POSITION           ; GET REGEN LOCATION
144E  74 20           C          JZ      N16
1450  2B F0           C          SUB     SI,AX                     ; SI IS FROM ADDRESS
1452  8A E6           C          MOV     AH,DH                     ; GET TOTAL # ROWS
1454  2A E3           C          SUB     AH,BL                     ; COUNT TO MOVE IN SCROLL
1456                  C  N13:
1456  E8 1432 R       C          CALL    N10                       ; MOVE ONE ROW
1459  2B F5           C          SUB     SI,BP
145B  2B FD           C          SUB     DI,BP
145D  FE CC           C          DEC     AH
145F  75 F5           C          JNZ     N13
1461                  C  N14:
1461  58              C          POP     AX                        ; RECOVER ATTRIBUTE IN AH
1462  B0 20           C          MOV     AL,' '
1464                  C  N15:
1464  E8 143B R       C          CALL    N11                       ; CLEAR ONE ROW
1467  2B FD           C          SUB     DI,BP                     ; GO TO NEXT ROW
1469  FE CB           C          DEC     BL
146B  75 F7           C          JNZ     N15
146D  E9 13DA R       C          JMP     N5                        ; SCROLL_END
1470                  C  N16:
1470  8A DE           C          MOV     BL,DH
1472  EB ED           C          JMP     N14
1474                  C  SCROLL_DOWN     ENDP
                      C
                      C  ;------------------------------------------------------------------
                      C  ; SCROLL UP                                                        :
                      C  ;       THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT         :
                      C  ; ENTRY                                                            :
                      C  ;       CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL              :
                      C  ;       DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL             :
                      C  ;        BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS              :
                      C  ;       BH = FILL VALUE FOR BLANKED LINES                          :
                      C  ;       AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE         :
                      C  ;            FIELD)                                                :
                      C  ;       DS = DATA SEGMENT                                          :
                         C ;         ES = REGEN SEGMENT                                        :
                         C ; EXIT                                                              :
                         C ;         NOTHING, THE SCREEN IS SCROLLED                           :
                         C ;--------------------------------------------------------------------
1474                     C GRAPHICS_UP     PROC    NEAR
1474  8A D8              C         MOV     BL,AL                           ; SAVE LINE COUNT IN BL
1476  8B C1              C         MOV     AX,CX                           ; GET UPPER LEFT POSITION
                         C                                                 ;  INTO AX REG
                         C
                         C ;----- USE CHARACTER SUBROUTINE FOR POSITIONING
                         C ;----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
                         C
1478  E8 16A7 R          C         CALL    GRAPH_POSN
147B  8B F8              C         MOV     DI,AX                           ; SAVE RESULT AS
                         C                                                 ;  DESTINATION ADDRESS
                         C
                         C ;----- DETERMINE SIZE OF WINDOW
                         C
147D  2B D1              C         SUB     DX,CX
147F  81 C2 0101         C         ADD     DX,101H                         ; ADJUST VALUES
1483  D0 E6              C         SAL     DH,1                            ; MULTIPLY # ROWS BY 4
                         C                                                 ;  SINCE 8 VERT DOTS/CHAR
1485  D0 E6              C         SAL     DH,1                            ;  AND EVEN/ODD ROWS
                         C
                         C ;----- DETERMINE CRT MODE
                         C
1487  80 3E 0449 R 06    C         CMP     CRT_MODE,6                      ; TEST FOR MEDIUM RES
148C  73 04              C         JNC     R7                              ; FIND_SOURCE
                         C
                         C ;----- MEDIUM RES UP
                         C
148E  D0 E2              C         SAL     DL,1                            ; * 2,
1490  D1 E7              C         SAL     DI,1                            ;  SINCE 2 BYTES/CHAR
                         C
                         C ;----- DETERMINE THE SOURCE ADDRESS IN THE BUFFER
                         C
1492                     C R7:                                             ; FIND_SOURCE
1492  06                 C         PUSH    ES                              ; GET SEGMENTS BOTH
1493  1F                 C         POP     DS                              ;  POINTING TO REGEN
1494  2A ED              C         SUB     CH,CH                           ; 0 TO HIGH OF COUNT REG
1496  D0 E3              C         SAL     BL,1                            ; NUMBER OF LINES *4
1498  D0 E3              C         SAL     BL,1
149A  74 2D              C         JZ      R11                             ; IF 0, BLANK ENTIRE FIELD
149C  8A C3              C         MOV     AL,BL                           ; NUMBER OF LINES IN AL
149E  B4 50              C         MOV     AH,80                           ; 80 BYTES/ROW
14A0  F6 E4              C         MUL     AH                              ; OFFSET TO SOURCE
14A2  8B F7              C         MOV     SI,DI                           ; SET UP SOURCE
14A4  03 F0              C         ADD     SI,AX                           ;  ADD IN OFFSET TO IT
14A6  8A E6              C         MOV     AH,DH                           ; NUMBER OF ROWS IN FIELD
14A8  2A E3              C         SUB     AH,BL                           ; DETERMINE NUMBER TO MOVE
                         C
                         C ;----- LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
                         C
14AA                     C R8:                                             ; ROW_LOOP
14AA  E8 14CD R          C         CALL    R17                             ; MOVE ONE ROW
14AD  81 EE 1FB0         C         SUB     SI,2000H-80                     ; MOVE TO NEXT ROW
14B1  81 EF 1FB0         C         SUB     DI,2000H-80
14B5  FE CC              C         DEC     AH                              ; NUMBER OF ROWS TO MOVE
14B7  75 F1              C         JNZ     R8                              ; CONTINUE TILL ALL MOVED
                         C
                         C ;----- FILL IN THE VACATED LINE(S)
                         C
14B9                     C R9:                                             ; CLEAR_ENTRY
14B9  8A C7              C         MOV     AL,BH                           ; ATTRIBUTE TO FILL WITH
14BB                     C R10:
14BB  E8 14E6 R          C         CALL    R18                             ; CLEAR THAT ROW
14BE  81 EF 1FB0         C         SUB     DI,2000H-80                     ; POINT TO NEXT LINE
14C2  FE CB              C         DEC     BL                              ; NUMBER OF LINES TO FILL
14C4  75 F5              C         JNZ     R10                             ; CLEAR_LOOP
14C6  E9 219E R          C         JMP     V_RET
14C9                     C R11:                                            ; BLANK_FIELD
14C9  8A DE              C         MOV     BL,DH                           ; SET BLANK COUNT TO
                         C                                                 ;  EVERYTHING IN FIELD
14CB  EB EC              C         JMP     R9                              ; CLEAR THE FIELD
14CD                     C GRAPHICS_UP     ENDP
                         C
                         C ;----- ROUTINE TO MOVE ONE ROW OF INFORMATION
                         C
14CD                     C R17     PROC    NEAR
14CD  8A CA              C         MOV     CL,DL                           ; NUM OF BYTES IN THE ROW
14CF  56                 C         PUSH    SI
14D0  57                 C         PUSH    DI                              ; SAVE POINTERS
14D1  F3/ A4             C         REP     MOVSB                           ; MOVE THE EVEN FIELD
14D3  5F                 C         POP     DI
14D4  5E                 C         POP     SI
14D5  81 C6 2000         C         ADD     SI,2000H
14D9  81 C7 2000         C         ADD     DI,2000H                        ; POINT TO THE ODD FIELD
14DD  56                 C         PUSH    SI
14DE  57                 C         PUSH    DI                              ; SAVE THE POINTERS
14DF  8A CA              C         MOV     CL,DL                           ; COUNT BACK
14E1  F3/ A4             C         REP     MOVSB                           ; MOVE THE ODD FIELD
14E3  5F                 C         POP     DI
14E4  5E                 C         POP     SI                              ; POINTERS BACK
14E5  C3                 C         RET                                     ; RETURN TO CALLER
14E6                     C R17     ENDP
                         C
                         C ;----- CLEAR A SINGLE ROW
                         C
14E6                     C R18     PROC    NEAR
14E6  8A CA              C         MOV     CL,DL                           ; NUMBER OF BYTES IN FIELD
14E8  57                 C         PUSH    DI                              ; SAVE POINTER
14E9  F3/ AA             C         REP     STOSB                           ; STORE THE NEW VALUE
14EB  5F                 C         POP     DI                              ; POINTER BACK
14EC  81 C7 2000         C         ADD     DI,2000H                        ; POINT TO ODD FIELD
14F0  57                 C         PUSH    DI
14F1  8A CA              C         MOV     CL,DL
14F3  F3/ AA             C         REP     STOSB                           ; FILL THE ODD FIELD
14F5  5F                 C         POP     DI
14F6  C3                 C         RET                                     ; RETURN TO CALLER
14F7                     C R18     ENDP
                         C
14F7                     C MEM_DET PROC    NEAR
                         C         ASSUME  DS:ABSO
14F7  50                 C         PUSH    AX
14F8  1E                 C         PUSH    DS
14F9  E8 0CFE R          C         CALL    DDS
14FC  8A 26 0487 R       C         MOV     AH,INFO
1500  80 E4 60           C         AND     AH,060H
1503  1F                 C         POP     DS
1504  58                 C         POP     AX
1505  74 02              C         JZ      MIN
1507  F9                 C         STC
1508  C3                 C         RET
1509                     C MIN:
1509  F8                 C         CLC
150A  C3                 C         RET
```

```
150B                  C   MEM_DET ENDP
                      C
                      C   ;----- SCROLL ACTIVE PAGE UP
                      C
150B                  C   SC_2:
150B  E9 13A3 R       C           JMP     SCROLL_UP
                      C
150E                  C   AH6:
                      C           ASSUME  DS:ABS0
150E  E8 12D1 R       C           CALL    FLTA
1511  8A 26 0449 R    C           MOV     AH,CRT_MODE               ; GET CURRENT MODE
1515  80 FC 07        C           CMP     AH,07H
1518  76 F1           C           JBE     SC_2                      ; ANY OF THE OLD MODES
151A  80 FC 0D        C           CMP     AH,0DH
151D  73 17           C           JAE     GRAPHICS_UP_2             ; NEW GRAPHICS MODES
151F  E9 219E R       C           JMP     V_RET                     ; NOT A RECOGNIZED MODE
                      C
1522                  C   GR_ST_1 PROC    NEAR
1522  BA A000         C           MOV     DX,0A000H                 ; REGEN BUFFER
1525  BD 0511         C           MOV     BP,0511H                  ; GRAPHICS WRITE MODE
1528  80 FC 0F        C           CMP     AH,0FH
152B  72 08           C           JB      VV1
152D  E8 14F7 R       C           CALL    MEM_DET
1530  73 03           C           JNC     VV1
1532  BD 0501         C           MOV     BP,0501H                  ; GRAPHICS WRITE MODE
1535                  C   VV1:
1535  C3              C           RET
1536                  C   GR_ST_1 ENDP
                      C
1536                  C   GRAPHICS_UP_2   PROC    NEAR
                      C           ASSUME  DS:ABS0
1536  52              C           PUSH    DX
1537  E8 1522 R       C           CALL    GR_ST_1                   ; SET SEGMENT, WRITE MODE
                      C           SRLOAD  ES                        ; SET REGEN
153A  8E C2           C+          MOV     ES,DX
153C  5A              C           POP     DX
153D  8A D8           C           MOV     BL,AL                     ; NUMBER OF LINES
153F  8B C1           C           MOV     AX,CX                     ; UPPER LEFT CORNER
1541  53              C           PUSH    BX
1542  8A 3E 0462 R    C           MOV     BH,ACTIVE_PAGE            ; ACTIVE PAGE FOR SCROLL
1546  E8 16C6 R       C           CALL    GRX_PSN                   ; ADDRESS IN REGEN
1549  5B              C           POP     BX
154A  8B F8           C           MOV     DI,AX                     ; SET POINTER
154C  2B D1           C           SUB     DX,CX                     ; DETERMINE WINDOW
154E  81 C2 0101      C           ADD     DX,0101H                  ; ADJUST
1552  2A E4           C           SUB     AH,AH                     ; ZERO HIGH BYTE
1554  8A C3           C           MOV     AL,BL                     ; LINE COUNT
1556  52              C           PUSH    DX
1557  F7 26 0485 R    C           MUL     POINTS                    ; BYTES PER CHARACTER
155B  F7 26 044A R    C           MUL     CRT_COLS                  ; COLUMNS
155F  8B F7           C           MOV     SI,DI                     ; SET UP SOURCE INDEX
1561  03 F0           C           ADD     SI,AX                     ; ADJUST
                      C           ASSUME  DS:NOTHING
1563  06              C           PUSH    ES
1564  1F              C           POP     DS
1565  5A              C           POP     DX
1566  0A DB           C           OR      BL,BL                     ; LINE COUNT
1568  74 3F           C           JZ      AR9
156A  8A CE           C           MOV     CL,DH
156C  2A CB           C           SUB     CL,BL
156E  2A ED           C           SUB     CH,CH
                      C
                      C           ASSUME  DS:ABS0
1570  1E              C           PUSH    DS
1571  E8 0CFE R       C           CALL    DDS                       ; LOW MEMORY SEGMENT
1574  50              C           PUSH    AX
1575  52              C           PUSH    DX
1576  8B C1           C           MOV     AX,CX
1578  F7 26 0485 R    C           MUL     POINTS                    ; BYTES PER CHAR
157C  8B C8           C           MOV     CX,AX                     ; SET THE COUNT
157E  5A              C           POP     DX
157F  58              C           POP     AX
                      C           ASSUME  DS:NOTHING
1580  1F              C           POP     DS
                      C
1581  52              C           PUSH    DX
1582  8B C5           C           MOV     AX,BP
1584  B6 03           C           MOV     DH,3
1586  B2 CE           C           MOV     DL,GRAPH_ADDR             ; GRAPHICS
1588  E8 0D15 R       C           CALL    OUT_DX
158B  B2 C4           C           MOV     DL,SEQ_ADDR               ; SEQUENCER
158D  B8 020F         C           MOV     AX,020FH                  ; ENABLE ALL MAPS
1590  E8 0D15 R       C           CALL    OUT_DX
1593  5A              C           POP     DX
1594  E8 12E0 R       C           CALL    CRANK                     ; SCROLL THE SCREEN
                      C
1597  52              C           PUSH    DX
1598  4D              C           DEC     BP
1599  8B C5           C           MOV     AX,BP
159B  B6 03           C           MOV     DH,3
159D  B2 CE           C           MOV     DL,GRAPH_ADDR
159F  E8 0D15 R       C           CALL    OUT_DX
15A2  5A              C           POP     DX
15A3                  C   AR10:
15A3  E8 1353 R       C           CALL    BLNK_3
15A6  E9 219E R       C           JMP     V_RET
15A9                  C   AR9:
15A9  8A DE           C           MOV     BL,DH                     ; BLANK ENTIRE WINDOW
15AB  EB F6           C           JMP     AR10
15AD                  C   GRAPHICS_UP_2   ENDP
                      C
                      C   ;----- SCROLL ACTIVE DISPLAY PAGE DOWN
                      C
15AD                  C   SC_3:
15AD  E9 1442 R       C           JMP     SCROLL_DOWN
                      C
15B0                  C   AH7:
                      C           ASSUME  DS:ABS0
15B0  E8 12D1 R       C           CALL    FLTA
15B3  8A 26 0449 R    C           MOV     AH,CRT_MODE
15B7  80 FC 03        C           CMP     AH,03H                    ; OLD COLOR ALPHA
15BA  76 F1           C           JBE     SC_3
15BC  80 FC 07        C           CMP     AH,07H                    ; MONOCHROME ALPHA
15BF  74 EC           C           JE      SC_3
                      C
15C1  80 FC 0D        C           CMP     AH,0DH                    ; NEW GRAPHICS MODES
15C4  73 0C           C           JAE     GRAPHICS_DN_2
15C6  80 FC 06        C           CMP     AH,06H                    ; OLD GRAPHICS MODES
15C9  77 04           C           JA      M_0
15CB  B4 07           C           MOV     AH,07H
15CD  CD 42           C           INT     42H
15CF                  C   M_0:
15CF  E9 219E R       C           JMP     V_RET
                      C
15D2                  C   GRAPHICS_DN_2   PROC    NEAR
15D2  FD              C           STD                               ; DIRECTION TO DECREMENT
```

```
15D3  8A D8              C             MOV     BL,AL                   ; LINE COUNT
15D5  52                 C             PUSH    DX                      ; SAVE LOWER RIGHT
15D6  E8 1522 R          C             CALL    GR_ST_1
                         C             SRLOAD  ES                      ; SET REGEN SEGMENT
15D9  8E C2              C+            MOV     ES,DX
15DB  5A                 C             POP     DX
15DC  8B C2              C             MOV     AX,DX
15DE  FE C4              C             INC     AH                      ; MOV CHAR ROW UP BY ONE
15E0  53                 C             PUSH    BX
15E1  8A 3E 0462 R       C             MOV     BH,ACTIVE_PAGE
15E5  E8 16C6 R          C             CALL    GRX_PSN                 ; ADDRESS IN REGEN
15E8  5B                 C             POP     BX
15E9  2B 06 044A R       C             SUB     AX,CRT_COLS             ; ONE SCAN OVERSHOOT
15ED  8B F8              C             MOV     DI,AX
15EF  2B D1              C             SUB     DX,CX                   ; CALCULATE WINDOW
15F1  81 C2 0101         C             ADD     DX,0101H                ; ADJUST COUNT
15F5  2A E4              C             SUB     AH,AH
15F7  8A C3              C             MOV     AL,BL
15F9  52                 C             PUSH    DX
15FA  F7 26 0485 R       C             MUL     POINTS                  ; BYTES PER CHAR
15FE  F7 26 044A R       C             MUL     CRT_COLS                ; BYTES PER ROW
1602  8B F7              C             MOV     SI,DI
1604  2B F0              C             SUB     SI,AX
                         C             ASSUME  DS:NOTHING
1606  06                 C             PUSH    ES                      ; SET DS TO
1607  1F                 C             POP     DS                      ;  THE REGEN SEGMENT
1608  5A                 C             POP     DX
1609  0A DB              C             OR      BL,BL                   ; SCROLL COUNT
160B  74 40              C             JZ      DXR9                    ; BLANK ENTIRE WINDOW
160D  8A CE              C             MOV     CL,DH
160F  2A CB              C             SUB     CL,BL
1611  2A ED              C             SUB     CH,CH
                         C
                         C             ASSUME  DS:ABS0
1613  1E                 C             PUSH    DS
1614  E8 0CFE R          C             CALL    DDS
1617  50                 C             PUSH    AX
1618  52                 C             PUSH    DX
1619  8B C1              C             MOV     AX,CX
161B  F7 26 0485 R       C             MUL     POINTS                  ; BYTES PER CHAR
161F  8B C8              C             MOV     CX,AX
1621  5A                 C             POP     DX
1622  58                 C             POP     AX
                         C             ASSUME  DS:NOTHING
1623  1F                 C             POP     DS
                         C
1624  52                 C             PUSH    DX
1625  8B C5              C             MOV     AX,BP
1627  B6 03              C             MOV     DH,3
1629  B2 CE              C             MOV     DL,GRAPH_ADDR           ; GRAPHICS
162B  E8 0D15 R          C             CALL    OUT_DX
162E  B2 C4              C             MOV     DL,SEQ_ADDR             ; SEQUENCER
1630  B8 020F            C             MOV     AX,020FH                ; ENABLE ALL MAPS
1633  E8 0D15 R          C             CALL    OUT_DX
1636  5A                 C             POP     DX
1637  E8 12FE R          C             CALL    CRANK_4                 ; SCROLL THE SCREEN
                         C
163A  52                 C             PUSH    DX
163B  4D                 C             DEC     BP
163C  8B C5              C             MOV     AX,BP
163E  B6 03              C             MOV     DH,3
1640  B2 CE              C             MOV     DL,GRAPH_ADDR
1642  E8 0D15 R          C             CALL    OUT_DX
1645  5A                 C             POP     DX
1646                     C     DXR10:
1646  E8 137B R          C             CALL    BLNK_4
1649  FC                 C             CLD
164A  E9 219E R          C             JMP     V_RET
164D                     C     DXR9:
164D  8A DE              C             MOV     BL,DH                   ; BLANK ENTIRE WINDOW
164F  EB F5              C             JMP     DXR10
1651                     C     GRAPHICS_DN_2   ENDP
                         C
                         C             SUBTTL

                         C             INCLUDE         VGRW.INC
                         C             SUBTTL  VGRW.INC
                         C             PAGE
                         C
                         C             ASSUME  DS:ABS0
1651                     C     FIND_POSITION   PROC    NEAR
1651  8A CF              C             MOV     CL,BH                   ; DISPLAY PAGE TO CX
1653  32 ED              C             XOR     CH,CH
1655  8B F1              C             MOV     SI,CX                   ; MOVE TO SI FOR INDEX
1657  D1 E6              C             SAL     SI,1                    ; * 2 FOR WORD OFFSET
1659  8B 84 0450 R       C             MOV     AX,[SI+ OFFSET CURSOR_POSN]  ; ROW/COLUMN OF THAT PAGE
165D  33 DB              C             XOR     BX,BX                   ; SET START ADDRESS TO 0
165F  E3 06              C             JCXZ    P5                      ; NO_PAGE
1661                     C     P4:                                     ; PAGE_LOOP
1661  03 1E 044C R       C             ADD     BX,CRT_LEN              ; LENGTH OF BUFFER
1665  E2 FA              C             LOOP    P4
1667                     C     P5:                                     ; NO_PAGE
1667  E8 1146 R          C             CALL    POSITION                ; DETERMINE LOC IN REGEN
166A  03 D8              C             ADD     BX,AX                   ; ADD TO START OF REGEN
166C  C3                 C             RET
166D                     C     FIND_POSITION   ENDP
                         C
                         C     ;--------------------------------------------------------
                         C     ; EXPAND_MED_COLOR
                         C     ;       THIS ROUTINE EXPANDS THE LOW 2 BITS IN BL TO    :
                         C     ;       FILL THE ENTIRE BX REGISTER                     :
                         C     ; ENTRY                                                 :
                         C     ;       BL = COLOR TO BE USED ( LOW 2 BITS )            :
                         C     ; EXIT                                                  :
                         C     ;       BX = COLOR TO BE USED ( 8 REPLICATIONS OF THE   :
                         C     ;       2 COLOR BITS )                                  :
                         C     ;--------------------------------------------------------
166D                     C     S19     PROC    NEAR
166D  80 E3 03           C             AND     BL,3                    ; ISOLATE THE COLOR BITS
1670  8A C3              C             MOV     AL,BL                   ; COPY TO AL
1672  51                 C             PUSH    CX                      ; SAVE REGISTER
1673  B9 0003            C             MOV     CX,3                    ; NUMBER OF TIMES
1676                     C     S20:
1676  D0 E0              C             SAL     AL,1
1678  D0 E0              C             SAL     AL,1                    ; LEFT SHIFT BY 2
167A  0A D8              C             OR      BL,AL                   ; ANOTHER COLOR VERSION
                         C                                             ;  INTO BL
167C  E2 F8              C             LOOP    S20                     ; FILL ALL OF BL
167E  8A FB              C             MOV     BH,BL                   ; FILL UPPER PORTION
1680  59                 C             POP     CX                      ; REGISTER BACK
1681  C3                 C             RET                             ; ALL DONE
1682                     C     S19     ENDP
                         C     ;--------------------------------------------------------
                         C     ; EXPAND_BYTE
                         C     ;       THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES   :
                         C     ;       ALL OF THE BITS, TURNING THE 8 BITS INTO        :
```

```
                        C  ;         16 BITS. THE RESULT IS LEFT IN AX                    :
                        C  ;-------------------------------------------------------------
1682                    C  S21       PROC    NEAR
1682  52                C            PUSH    DX                              ; SAVE REGISTERS
1683  51                C            PUSH    CX
1684  53                C            PUSH    BX
1685  2B D2             C            SUB     DX,DX                           ; RESULT REGISTER
1687  B9 0001           C            MOV     CX,1                            ; MASK REGISTER
168A                    C  S22:
168A  8B D8             C            MOV     BX,AX                           ; BASE INTO TEMP
168C  23 D9             C            AND     BX,CX                           ; USE MASK TO EXTRACT BIT
168E  0B D3             C            OR      DX,BX                           ; PUT INTO RESULT REGISTER
1690  D1 E0             C            SHL     AX,1
1692  D1 E1             C            SHL     CX,1                            ; SHIFT BASE AND MASK BY 1
1694  8B D8             C            MOV     BX,AX                           ; BASE TO TEMP
1696  23 D9             C            AND     BX,CX                           ; EXTRACT THE SAME BIT
1698  0B D3             C            OR      DX,BX                           ; PUT INTO RESULT
169A  D1 E1             C            SHL     CX,1                            ; SHIFT ONLY MASK NOW,
                        C                                                    ;  MOVING TO NEXT BASE
169C  73 EC             C            JNC     S22                             ; USE MASK BIT COMING OUT
                        C                                                    ;  TO TERMINATE
169E  8B C2             C            MOV     AX,DX                           ; RESULT TO PARM REGISTER
16A0  5B                C            POP     BX
16A1  59                C            POP     CX                              ; RECOVER REGISTERS
16A2  5A                C            POP     DX
16A3  C3                C            RET                                     ; ALL DONE
16A4                    C  S21       ENDP
                        C
16A4                    C  S26       PROC    NEAR
16A4  A1 0450 R         C            MOV     AX,CURSOR_POSN                  ; GET CURRENT CURSOR
16A7                    C  GRAPH_POSN        LABEL   NEAR
16A7  53                C            PUSH    BX                              ; SAVE REGISTER
16A8  8B D8             C            MOV     BX,AX                           ; SAVE A COPY OF CURSOR
16AA  8A C4             C            MOV     AL,AH                           ; GET ROWS TO AL
16AC  F6 26 044A R      C            MUL     BYTE PTR CRT_COLS               ; MULTIPLY BY BYTES/COLUMN
16B0  D1 E0             C            SHL     AX,1                            ; *4 SINCE 4 ROWS/BYTE
16B2  D1 E0             C            SHL     AX,1
16B4  2A FF             C            SUB     BH,BH                           ; ISOLATE COLUMN VALUE
16B6  03 C3             C            ADD     AX,BX                           ; DETERMINE OFFSET
16B8  5B                C            POP     BX                              ; RECOVER POINTER
16B9  C3                C            RET                                     ; ALL DONE
16BA                    C  S26       ENDP
                        C
                        C  ;--------------------------------------------------------------------
                        C  ; GR_CUR                                                            :
                        C  ; ENTRY                                                             :
                        C  ;         BH = DISPLAY PAGE                                         :
                        C  ; EXIT                                                              :
                        C  ;         AX = CURSOR POSITION FOR REQUESTED PAGE                   :
                        C  ;--------------------------------------------------------------------
16BA                    C  GR_CUR:
                        C            ASSUME  DS:ABS0
16BA  53                C            PUSH    BX                              ; SAVE REGISTER
16BB  8A DF             C            MOV     BL,BH                           ; GET TO LOW BYTE
16BD  2A FF             C            SUB     BH,BH                           ; ZERO HIGH BYTE
16BF  D1 E3             C            SAL     BX,1                            ; *2 FOR WORD COUNT
16C1  8B 87 0450 R      C            MOV     AX,[BX + OFFSET CURSOR_POSN]    ; CURSOR, REQUESTED PAGE
16C5  5B                C            POP     BX                              ; RECOVER REGISTER
                        C  ;--------------------------------------------------------------------
                        C  ; GRX_PSN                                                           :
                        C  ; ENTRY                                                             :
                        C  ;         AX = CURSOR POSITION IN DESIRED PAGE                      :
                        C  ;         BH = DESIRED PAGE                                         :
                        C  ; EXIT                                                              :
                        C  ;         AX = BYTE OFFSET INTO REGEN                               :
                        C  ;--------------------------------------------------------------------
16C6                    C  GRX_PSN   PROC    NEAR
16C6  53                C            PUSH    BX                              ; SAVE
16C7  51                C            PUSH    CX                              ; SAVE
16C8  52                C            PUSH    DX                              ; SAVE
16C9  2A ED             C            SUB     CH,CH                           ; ZERO
16CB  8A CF             C            MOV     CL,BH                           ; PAGE NUMBER
16CD  8B D8             C            MOV     BX,AX                           ; ROW, COLUMN
16CF  8A C4             C            MOV     AL,AH                           ; ROW
16D1  F6 26 044A R      C            MUL     BYTE PTR CRT_COLS               ; ROW * COLUMNS/ROW
16D5  F7 26 0485 R      C            MUL     POINTS                          ; BYTES PER ROW
16D9  2A FF             C            SUB     BH,BH                           ; ZERO TO LEAVE COL VALUE
16DB  03 C3             C            ADD     AX,BX                           ; ADD IN COLUMN
16DD  8B 1E 044C R      C            MOV     BX,CRT_LEN                      ; PAGE LENGTH
16E1  E3 04             C            JCXZ    GP_2                            ; NO PAGE OFFSET
16E3                    C  GP_3:
16E3  03 C3             C            ADD     AX,BX                           ; ADD IN THE PAGE LENGTH
16E5  E2 FC             C            LOOP    GP_3                            ; DO FOR NUMBER OF PAGES
16E7                    C  GP_2:
16E7  5A                C            POP     DX                              ; RECOVER
16E8  59                C            POP     CX                              ; RECOVER
16E9  5B                C            POP     BX                              ; RECOVER
16EA  C3                C            RET
16EB                    C  GRX_PSN ENDP
                        C
16EB                    C  MK_ES:
16EB  BE B800           C            MOV     SI,0B800H
16EE  8B 3E 0410 R      C            MOV     DI,EQUIP_FLAG
16F2  81 E7 0030        C            AND     DI,030H
16F6  83 FF 30          C            CMP     DI,030H
16F9  75 03             C            JNE     P6_A
16FB  BE B000           C            MOV     SI,0B000H
16FE                    C  P6_A:
16FE  8E C6             C            MOV     ES,SI
1700  C3                C            RET
                        C
                        C  ;----------------------------------------------------------
                        C  ; READ_AC_CURRENT                                          :
                        C  ;         THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER   :
                        C  ;         AT THE CURRENT CURSOR POSITION AND RETURNS THEM  :
                        C  ;         TO THE CALLER                                    :
                        C  ;INPUT                                                    :
                        C  ;         (AH) = CURRENT CRT MODE                          :
                        C  ;         (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )         :
                        C  ;         (DS) = DATA SEGMENT                              :
                        C  ;         (ES) = REGEN SEGMENT                             :
                        C  ;OUTPUT                                                   :
                        C  ;         (AL) = CHAR READ                                 :
                        C  ;         (AH) = ATTRIBUTE READ                            :
                        C  ;----------------------------------------------------------
                        C            ASSUME  CS:CODE,DS:ABS0,ES:NOTHING
1701                    C  READ_AC_CURRENT PROC     NEAR
1701  E8 16EB R         C            CALL    MK_ES
1704  E8 1651 R         C            CALL    FIND_POSITION
1707  8B F3             C            MOV     SI,BX                           ; ADDRESSING IN SI
                        C
1709  8B 16 0463 R      C            MOV     DX,ADDR_6845                    ; GET BASE ADDRESS
170D  83 C2 06          C            ADD     DX,6                            ; POINT AT STATUS PORT
                        C
1710  F6 06 0487 R 04   C            TEST    INFO,4
                        C
```

```
1715  06                C           PUSH    ES
1716  1F                C           POP     DS                              ; SEGMENT FOR QUICK ACCESS
                        C
1717  74 0B             C           JZ      P3A
                        C
                        C  ;-----'WAIT FOR HORIZONTAL RETRACE
                        C
1719                    C  P2:                                              ; WAIT FOR RETRACE LOW
1719  EC                C           IN      AL,DX                           ; GET STATUS
171A  A8 01             C           TEST    AL,1                            ; IS HORZ RETRACE LOW
171C  75 FB             C           JNZ     P2                              ; WAIT UNTIL IT IS
171E  FA                C           CLI                                     ; NO MORE INTERRUPTS
171F                    C  P3:                                              ; WAIT FOR RETRACE HIGH
171F  EC                C           IN      AL,DX                           ; GET STATUS
1720  A8 01             C           TEST    AL,1                            ; IS IT HIGH
1722  74 FB             C           JZ      P3                              ; WAIT UNTIL IT IS
1724                    C  P3A:
1724  AD                C           LODSW                                   ; GET THE CHAR/ATTR
1725  E9 219E R         C           JMP     V_RET
1728                    C  READ_AC_CURRENT ENDP
                        C
                        C  ;-----------------------------------------------------
                        C  ; MED_READ_BYTE                                      :
                        C  ;        THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN  :
                        C  ;        BUFFER, COMPARE AGAINST THE CURRENT FOREGROUND :
                        C  ;        COLOR, AND PLACE THE CORRESPONDING ON/OFF BIT  :
                        C  ;        PATTERN INTO THE CURRENT POSITION IN THE SAVE  :
                        C  ;        AREA                                         :
                        C  ; ENTRY                                              :
                        C  ;        SI,DS = POINTER TO REGEN AREA OF INTEREST     :
                        C  ;        BX = EXPANDED FOREGROUND COLOR                :
                        C  ;        BP = POINTER TO SAVE AREA                     :
                        C  ; EXIT                                               :
                        C  ;        BP IS INCREMENT AFTER SAVE                    :
                        C  ;-----------------------------------------------------
1728                    C  S23      PROC    NEAR
1728  8A 24             C           MOV     AH,[SI]                         ; GET FIRST BYTE
172A  8A 44 01          C           MOV     AL,[SI+1]                       ; GET SECOND BYTE
172D  B9 C000           C           MOV     CX,0C000H                       ; 2 BIT MASK TO TEST
                        C                                                   ;  THE ENTRIES
1730  B2 00             C           MOV     DL,0                            ; RESULT REGISTER
1732                    C  S24:
1732  85 C1             C           TEST    AX,CX                           ; IS THIS BACKGROUND?
1734  F8                C           CLC                                     ; CLEAR CARRY IN HOPES
                        C                                                   ;  THAT IT IS
1735  74 01             C           JZ      S25                             ; IF 0, IT IS BACKGROUND
1737  F9                C           STC                                     ; WASN'T, SO SET CARRY
1738                    C  S25:
1738  D0 D2             C           RCL     DL,1                            ; MOVE THAT BIT INTO THE
173A  D1 E9             C           SHR     CX,1                            ;  RESULT
173C  D1 E9             C           SHR     CX,1                            ; MOVE THE MASK TO THE
                        C                                                   ;  RIGHT BY 2 BITS
173E  73 F2             C           JNC     S24                             ; DO IT AGAIN IF MASK
                        C                                                   ;  DIDN'T FALL OUT
1740  88 56 00          C           MOV     [BP],DL                         ; STORE RESULT IN SAVE
1743  45                C           INC     BP                              ; ADJUST POINTER
1744  C3                C           RET                                     ; ALL DONE
1745                    C  S23      ENDP
                        C
                        C
1745                    C  GRAPHICS_READ    PROC    NEAR
1745  E8 16EB R         C           CALL    MK_ES
1748  E8 16A4 R         C           CALL    S26                             ; CONVERTED TO OFFSET
174B  8B F0             C           MOV     SI,AX                           ; SAVE IN SI
174D  83 EC 08          C           SUB     SP,8                            ; ALLOCATE SPACE TO SAVE
                        C                                                   ;  THE READ CODE POINT
1750  8B EC             C           MOV     BP,SP                           ; POINTER TO SAVE AREA
                        C
                        C  ;----- DETERMINE GRAPHICS MODES
                        C
1752  80 3E 0449 R 06   C           CMP     CRT_MODE,6
1757  06                C           PUSH    ES
1758  1F                C           POP     DS                              ; POINT TO REGEN SEGMENT
1759  72 1A             C           JC      S13P                            ; MEDIUM RESOLUTION
                        C
                        C  ;----- HIGH RESOLUTION READ
                        C
                        C  ;----- GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
                        C
175B  B6 04             C           MOV     DH,4                            ; NUMBER OF PASSES
175D                    C  S12P:
175D  8A 04             C           MOV     AL,[SI]                         ; GET FIRST BYTE
175F  88 46 00          C           MOV     [BP],AL                         ; SAVE IN STORAGE AREA
1762  45                C           INC     BP                              ; NEXT LOCATION
1763  8A 84 2000        C           MOV     AL,[SI+2000H]                   ; GET LOWER REGION BYTE
1767  88 46 00          C           MOV     [BP],AL                         ; ADJUST AND STORE
176A  45                C           INC     BP
176B  83 C6 50          C           ADD     SI,80                           ; POINTER INTO REGEN
176E  FE CE             C           DEC     DH                              ; LOOP CONTROL
1770  75 EB             C           JNZ     S12P                            ; DO IT SOME MORE
1772  EB 17 90          C           JMP     S15P                            ; GO MATCH THE SAVED CODE
                        C                                                   ;  POINTS
                        C
                        C  ;----- MEDIUM RESOLUTION READ
                        C
1775                    C  S13P:                                            ; MED_RES_READ
1775  D1 E6             C           SAL     SI,1                            ; OFFSET*2, 2 BYTES/CHAR
1777  B6 04             C           MOV     DH,4                            ; NUMBER OF PASSES
1779                    C  S14P:
1779  E8 1728 R         C           CALL    S23                             ; GET PAIR BYTES
                        C                                                   ;  INTO SINGLE SAVE
177C  81 C6 2000        C           ADD     SI,2000H                        ; GO TO LOWER REGION
1780  E8 1728 R         C           CALL    S23                             ; GET THIS PAIR INTO SAVE
1783  81 EE 1FB0        C           SUB     SI,2000H-80                     ; ADJUST POINTER BACK INTO
1787  FE CE             C           DEC     DH                              ;  UPPER
1789  75 EE             C           JNZ     S14P                            ; KEEP GOING UNTIL 8 DONE
                        C
                        C  ;----- SAVE AREA HAS CHARACTER IN IT, MATCH IT
                        C
178B                    C  S15P:                                            ; FIND_CHAR
178B  1E                C           PUSH    DS
178C  E8 0CFE R         C           CALL    DDS
178F  C4 3E 010C R      C           LES     DI,GRX_SET                      ; ESTABLISH ADDRESSING
1793  1F                C           POP     DS
1794  83 ED 08          C           SUB     BP,8                            ; ADJUST POINTER TO
                        C                                                   ;  BEGINNING OF SAVE AREA
1797  8B F5             C           MOV     SI,BP
1799  FC                C           CLD                                     ; ENSURE DIRECTION
179A  B0 00             C           MOV     AL,0                            ; CURRENT CODE POINT BEING
179C                    C  S16P:                                            ;  MATCHED
179C  16                C           PUSH    SS                              ; ADDRESSING TO STACK
179D  1F                C           POP     DS                              ; FOR THE STRING COMPARE
179E  BA 0080           C           MOV     DX,128                          ; NUMBER TO TEST AGAINST
17A1                    C  S17P:
17A1  56                C           PUSH    SI                              ; SAVE SAVE AREA POINTER
17A2  57                C           PUSH    DI                              ; SAVE CODE POINTER
```

```
17A3  B9 0008           C            MOV     CX,8                    ; NUMBER OF BYTES TO MATCH
17A6  F3/ A6            C            REPE    CMPSB                   ; COMPARE THE 8 BYTES
17A8  5F                C            POP     DI                      ; RECOVER THE POINTERS
17A9  5E                C            POP     SI
17AA  74 1D             C            JZ      S18P                    ; IF ZERO FLAG SET,
                        C                                            ;  THEN MATCH OCCURRED
17AC  FE C0             C            INC     AL                      ; NO MATCH, MOVE TO NEXT
17AE  83 C7 08          C            ADD     DI,8                    ; NEXT CODE POINT
17B1  4A                C            DEC     DX                      ; LOOP CONTROL
17B2  75 ED             C            JNZ     S17P                    ; DO ALL OF THEM
                        C
                        C   ;----- CHAR NOT MATCHED, MIGHT BE IN USER SUPPLIED SECOND HALF
                        C
17B4  3C 00             C            CMP     AL,0                    ; AL <> 0 IF ONLY 1ST
                        C                                            ;  HALF SCANNED
17B6  74 11             C            JE      S18P                    ; IF = 0, THEN ALL HAS
                        C                                            ;  BEEN SCANNED
                        C            ASSUME  DS:ABS0
17B8  E8 0CFE R         C            CALL    DDS
17BB  C4 3E 007C R      C            LES     DI,EXT_PTR              ; GET POINTER
17BF  8C C0             C            MOV     AX,ES                   ; SEE IF THE PNTR EXISTS
17C1  0B C7             C            OR      AX,DI                   ; IF ALL 0, DOESN'T EXIST
17C3  74 04             C            JZ      S18P                    ; NO SENSE LOOKING
17C5  B0 80             C            MOV     AL,128                  ; ORIGIN FOR SECOND HALF
17C7  EB D3             C            JMP     S16P                    ; GO BACK AND TRY FOR IT
                        C
                        C   ;----- CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
                        C
17C9                    C   S18P:
17C9  83 C4 08          C            ADD     SP,8                    ; READJUST THE STACK,
                        C                                            ;  THROW AWAY SAVE
17CC  E9 219E R         C            JMP     V_RET                   ; ALL DONE
17CF                    C   GRAPHICS_READ    ENDP
                        C
                        C
                        C   ;----- READ CHARACTER/ATTRIBUTE AT CURRENT CURSOR POSITION
                        C
17CF                    C   AH8S:
17CF  E9 1701 R         C            JMP     READ_AC_CURRENT
                        C
17D2                    C   AH8:
                        C            ASSUME  DS:ABS0
17D2  8A 26 0449 R      C            MOV     AH,CRT_MODE             ; GET THE CURRENT MODE
17D6  80 FC 07          C            CMP     AH,07H
17D9  74 F4             C            JE      AH8S
17DB  80 FC 03          C            CMP     AH,03H
17DE  76 EF             C            JBE     AH8S
17E0  80 FC 06          C            CMP     AH,06H
17E3  77 03             C            JA      Z_1
17E5  E9 1745 R         C            JMP     GRAPHICS_READ
17E8                    C   Z_1:
17E8  80 FC 0F          C            CMP     AH,0FH
17EB  72 52             C            JB      GRX_RD2
17ED  E8 14F7 R         C            CALL    MEM_DET
17F0  72 4D             C            JC      GRX_RD2
17F2  EB 0A             C            JMP     SHORT GRX_RD1
17F4  80 FC 0D          C            CMP     AH,0DH                  ; RANGE TEST
17F7  73 46             C            JAE     GRX_RD2                 ; FOUR MAP READ
17F9  B0 00             C            MOV     AL,0
17FB  E9 219E R         C            JMP     V_RET
                        C
17FE                    C   GRX_RD1 PROC     NEAR
                        C            ASSUME  DS:ABS0
                        C            SRLOAD  ES,0A000H               ; REGEN SEGEMNT
17FE  BA A000           C+           MOV     DX,0A000H
1801  8E C2             C+           MOV     ES,DX
1803  E8 16BA R         C            CALL    GR_CUR                  ; BYTE OFFSET INTO REGEN
1806  8B F0             C            MOV     SI,AX                   ; SAVE IN SI
1808  8B 1E 0485 R      C            MOV     BX,POINTS               ; BYTES PER CHARACTER
180C  2B E3             C            SUB     SP,BX                   ; ALLOCATE SPACE TO SAVE
                        C                                            ;  THE READ CODE POINT
180E  8B EC             C            MOV     BP,SP                   ; POINTER TO SAVE AREA
                        C
                        C   ;----- GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
                        C
1810  53                C            PUSH    BX                      ; SAVE BYTES PER CHARACTER
1811  24 01             C            AND     AL,1                    ; ODD OR EVEN BYTE
1813  8A C8             C            MOV     CL,AL                   ; USE FOR SHIFT
1815  B0 05             C            MOV     AL,5                    ; COLOR COMP VALUE (C0-C2)
1817  D2 E0             C            SHL     AL,CL                   ; (C1-C3) IF ODD BYTE
1819  B4 07             C            MOV     AH,G_COLOR              ; COLOR COMPARE REGISTER
181B  B6 03             C            MOV     DH,3
181D  B2 CE             C            MOV     DL,GRAPH_ADDR
181F  E8 0D15 R         C            CALL    OUT_DX                  ; SET GRAPHICS CHIP
1822  B8 0518           C            MOV     AX,518H                 ; READ MODE
1825  E8 0D15 R         C            CALL    OUT_DX                  ; SET GRAPHICS CHIP
1828                    C   S12_1:
1828  26: 8A 04         C            MOV     AL,ES:[SI]              ; GET FIRST BYTE
182B  F6 D0             C            NOT     AL
182D  88 46 00          C            MOV     SS:[BP],AL              ; SAVE IN STORAGE AREA
1830  45                C            INC     BP                      ; NEXT LOCATION
1831  03 36 044A R      C            ADD     SI,CRT_COLS             ; POINTER INTO REGEN
1835  4B                C            DEC     BX                      ; LOOP CONTROL
1836  75 F0             C            JNZ     S12_1                   ; DO IT SOME MORE
1838  5B                C            POP     BX                      ; RECOVER BYTES PER CHAR
1839  B8 0510           C            MOV     AX,510H                 ; UNDO READ MODE
183C  EB 32 90          C            JMP     GRX_RECG                ; CHAR REGONTION ROUTINE
183F                    C   GRX_RD1 ENDP
                        C
183F                    C   GRX_RD2 PROC     NEAR
                        C            ASSUME  DS:ABS0
                        C            SRLOAD  ES,0A000H               ; REGEN SEGMENT
183F  BA A000           C+           MOV     DX,0A000H
1842  8E C2             C+           MOV     ES,DX
1844  E8 16BA R         C            CALL    GR_CUR                  ; BYTE OFFSET INTO REGEN
1847  8B F0             C            MOV     SI,AX                   ; SAVE IN SI
1849  8B 1E 0485 R      C            MOV     BX,POINTS               ; BYTES PER CHARACTER
184D  2B E3             C            SUB     SP,BX                   ; ALLOCATE SPACE TO SAVE
                        C                                            ;  THE READ CODE POINT
184F  8B EC             C            MOV     BP,SP                   ; POINTER TO SAVE AREA
                        C
                        C   ;----- GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
                        C
1851  B6 03             C            MOV     DH,3
1853  B2 CE             C            MOV     DL,GRAPH_ADDR           ; GRAPHICS CHIP
1855  B8 0508           C            MOV     AX,508H                 ; COLOR COMPARE
1858  E8 0D15 R         C            CALL    OUT_DX                  ; SET THE REGISTER
185B  53                C            PUSH    BX                      ; SAVE BYTES PER CHARACTER
185C                    C   S12:
185C  26: 8A 04         C            MOV     AL,ES:[SI]              ; GET COLOR COMPARED BYTE
185F  F6 D0             C            NOT     AL                      ; ADJUST
1861  88 46 00          C            MOV     SS:[BP],AL              ; SAVE IN STORAGE AREA
1864  45                C            INC     BP                      ; NEXT LOCATION
1865  03 36 044A R      C            ADD     SI,CRT_COLS             ; POINTER INTO REGEN
1869  4B                C            DEC     BX                      ; LOOP CONTROL
186A  75 F0             C            JNZ     S12                     ; DO IT SOME MORE
```

```
186C  5B                  C            POP     BX                          ; RECOVER BYTES PER CHAR
186D  B8 0500             C            MOV     AX,500H                     ; UNDO READ MODE
1870                      C  GRX_RD2 ENDP
                          C
1870                      C  GRX_RECG:
                          C
                          C  ;----- SAVE AREA HAS CHARACTER IN IT, MATCH IT
                          C
1870  E8 0D15 R           C            CALL    OUT_DX                      ; SET READ MODE BACK
1873  C4 3E 010C R        C            LES     DI,GRX_SET                  ; GET FONT DEFINITIONS
1877  2B EB               C            SUB     BP,BX                       ; ADJUST POINTER TO
                          C                                                ;  BEGINNING OF SAVE AREA
1879  8B F5               C            MOV     SI,BP
187B  FC                  C            CLD                                 ; ENSURE DIRECTION
187C  B0 00               C            MOV     AL,0                        ; CODE POINT BEING MATCHED
187E  16                  C            PUSH    SS                          ; ADDRESSING TO STACK
187F  1F                  C            POP     DS                          ;  FOR THE STRING COMPARE
1880  BA 0100             C            MOV     DX,256D                     ; NUMBER TO TEST AGAINST
1883                      C  S17_5:
1883  56                  C            PUSH    SI                          ; SAVE SAVE AREA POINTER
1884  57                  C            PUSH    DI                          ; SAVE CODE POINTER
1885  8B CB               C            MOV     CX,BX                       ; NUMBER OF BYTES TO MATCH
1887  F3/ A6              C            REPE    CMPSB                       ; COMPARE THE 8 BYTES
1889  5F                  C            POP     DI                          ; RECOVER THE POINTERS
188A  5E                  C            POP     SI
188B  74 07               C            JZ      S18_5                       ; IF ZFL SET, THEN MATCH
                          C                                                ;  OCCURRED
188D  FE C0               C            INC     AL                          ; NO MATCH, ON TO NEXT
188F  03 FB               C            ADD     DI,BX                       ; NEXT CODE POINT
1891  4A                  C            DEC     DX                          ; LOOP CONTROL
1892  75 EF               C            JNZ     S17_5                       ; DO ALL OF THEM
1894                      C  S18_5:                                        ; AL=CHAR, 0 IF NOT FOUND
1894  03 E3               C            ADD     SP,BX                       ; READJUST THE STACK
1896  E9 219E R           C            JMP     V_RET
                          C
                          C  ;----- WRITE CHARACTER/ATTRIBUTE AT CURRENT CURSOR POSITION
                          C
                          C  ;-----------------------------------------------
                          C  ; WRITE_AC_CURRENT                              :
                          C  ;       THIS ROUTINE WRITES THE ATTRIBUTE       :
                          C  ;       AND CHARACTER AT THE CURRENT CURSOR     :
                          C  ;       POSITION                                :
                          C  ; INPUT·                                        :
                          C  ;       (AH) = CURRENT CRT MODE                 :
                          C  ;       (BH) = DISPLAY PAGE                     :
                          C  ;       (CX) = COUNT OF CHARACTERS TO WRITE     :
                          C  ;       (AL) = CHAR TO WRITE                    :
                          C  ;       (BL) = ATTRIBUTE OF CHAR TO WRITE       :
                          C  ;       (DS) = DATA SEGMENT                     :
                          C  ;       (ES) = REGEN SEGMENT                    :
                          C  ; OUTPUT                                        :
                          C  ;       NONE                                    :
                          C  ;-----------------------------------------------
1899                      C  AH9:
                          C            ASSUME  DS:ABS0
1899  E8 0CFE R           C            CALL    DDS
189C  8A 26 0449 R        C            MOV     AH,CRT_MODE
                          C
18A0  80 FC 04            C            CMP     AH,4                        ; IS THIS GRAPHICS
18A3  72 08               C            JC      P6
18A5  80 FC 07            C            CMP     AH,7                        ; IS THIS BW CARD
18A8  74 03               C            JE      P6
18AA  EB 74 90            C            JMP     GRAPHICS_WRITE
18AD                      C  P6:                                           ; WRITE_AC_CONTINUE
18AD  E8 16EB R           C            CALL    MK_ES
18B0  8A E3               C            MOV     AH,BL                       ; GET ATTRIBUTE TO AH
18B2  50                  C            PUSH    AX                          ; SAVE ON STACK
18B3  51                  C            PUSH    CX                          ; SAVE WRITE COUNT
18B4  E8 1651 R           C            CALL    FIND_POSITION
18B7  8B FB               C            MOV     DI,BX                       ; ADDRESS TO DI REGISTER
18B9  59                  C            POP     CX                          ; WRITE COUNT
18BA  5B                  C            POP     BX                          ; CHARACTER IN BX REG
18BB  8B 16 0463 R        C            MOV     DX,ADDR_6845                ; GET BASE ADDRESS
18BF  83 C2 06            C            ADD     DX,6                        ; POINT AT STATUS PORT
                          C
                          C  ;----- WAIT FOR HORIZONTAL RETRACE
                          C
18C2                      C  P7:
18C2  F6 06 0487 R 04     C            TEST    INFO,4
18C7  74 0B               C            JZ      P9A
18C9                      C  P8:
18C9  EC                  C            IN      AL,DX                       ; GET STATUS
18CA  A8 01               C            TEST    AL,1                        ; IS IT LOW
18CC  75 FB               C            JNZ     P8                          ; WAIT UNTIL IT IS
18CE  FA                  C            CLI                                 ; NO MORE INTERRUPTS
18CF                      C  P9:
18CF  EC                  C            IN      AL,DX                       ; GET STATUS
18D0  A8 01               C            TEST    AL,1                        ; IS IT HIGH
18D2  74 FB               C            JZ      P9                          ; WAIT UNTIL IT IS
18D4                      C  P9A:
18D4  8B C3               C            MOV     AX,BX                       ; RECOVER THE CHAR/ATTR
18D6  AB                  C            STOSW                               ; PUT THE CHAR/ATTR
18D7  FB                  C            STI                                 ; INTERRUPTS BACK ON
18D8  E2 E8               C            LOOP    P7                          ; AS MANY TIMES
18DA  E9 219E R           C            JMP     V_RET
                          C
                          C  ;----- WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION
                          C
                          C  ;-----------------------------------------------
                          C  ; WRITE_C_CURRENT                               :
                          C  ;       THIS ROUTINE WRITES THE CHARACTER AT    :
                          C  ;       THE CURRENT CURSOR POSITION, ATTRIBUTE  :
                          C  ;       UNCHANGED                               :
                          C  ; INPUT                                         :
                          C  ;       (AH) = CURRENT CRT MODE                 :
                          C  ;       (BH) = DISPLAY PAGE                     :
                          C  ;       (CX) = COUNT OF CHARACTERS TO WRITE     :
                          C  ;       (AL) = CHAR TO WRITE                    :
                          C  ;       (DS) = DATA SEGMENT                     :
                          C  ;       (ES) = REGEN SEGMENT                    :
                          C  ; OUTPUT                                        :
                          C  ;       NONE                                    :
                          C  ;-----------------------------------------------
18DD                      C  AHA:
                          C            ASSUME  DS:ABS0
18DD  E8 0CFE R           C            CALL    DDS
18E0  8A 26 0449 R        C            MOV     AH,CRT_MODE
                          C
18E4  80 FC 04            C            CMP     AH,4                        ; IS THIS GRAPHICS
18E7  72 08               C            JC      P10
18E9  80 FC 07            C            CMP     AH,7                        ; IS THIS BW CARD
18EC  74 03               C            JE      P10
                          C
18EE  EB 30 90            C            JMP     GRAPHICS_WRITE
18F1                      C  P10:
18F1  E8 16EB R           C            CALL    MK_ES
```

```
18F4  50                    C           PUSH    AX                          ; SAVE ON STACK
18F5  51                    C           PUSH    CX                          ; SAVE WRITE COUNT
18F6  E8 1651 R             C           CALL    FIND_POSITION
18F9  8B FB                 C           MOV     DI,BX                       ; ADDRESS TO DI
18FB  59                    C           POP     CX                          ; WRITE COUNT
18FC  5B                    C           POP     BX                          ; BL HAS CHAR TO WRITE
                            C
                            C   ;----- WAIT FOR HORIZONTAL RETRACE
                            C
18FD  8B 16 0463 R          C           MOV     DX,ADDR_6845                ; GET BASE ADDRESS
1901  83 C2 06              C           ADD     DX,6                        ; POINT AT STATUS PORT
1904                        C   P11:
1904  F6 06 0487 R 04       C           TEST    INFO,4
1909  74 0B                 C           JZ      P13A
190B                        C   P12:
190B  EC                    C           IN      AL,DX                       ; GET STATUS
190C  A8 01                 C           TEST    AL,1                        ; IS IT LOW
190E  75 FB                 C           JNZ     P12                         ; WAIT UNTIL IT IS
1910  FA                    C           CLI                                 ; NO MORE INTERRUPTS
1911                        C   P13:
1911  EC                    C           IN      AL,DX                       ; GET STATUS
1912  A8 01                 C           TEST    AL,1                        ; IS IT HIGH
1914  74 FB                 C           JZ      P13                         ; WAIT UNTIL IT IS
1916                        C   P13A:
1916  8A C3                 C           MOV     AL,BL                       ; RECOVER CHAR
1918  AA                    C           STOSB                               ; PUT THE CHAR/ATTR
1919  FB                    C           STI                                 ; INTERRUPTS BACK ON
191A  47                    C           INC     DI                          ; BUMP POINTER PAST ATTR
191B  E2 E7                 C           LOOP    P11                         ;  AS REQUESTED
191D  E9 219E R             C           JMP     V_RET
                            C
                            C   ;--------------------------------------------------------------
                            C   ; GRAPHICS WRITE                                               :
                            C   ;       THIS ROUTINE WRITES THE ASCII CHARACTER TO THE         :
                            C   ;       CURRENT POSITION ON THE SCREEN.                        :
                            C   ; ENTRY                                                        :
                            C   ;       AL = CHARACTER TO WRITE                                :
                            C   ;       BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR   :
                            C   ;        IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN     :
                            C   ;        BUFFER (0 IS USED FOR THE BACKGROUND COLOR)           :
                            C   ;       CX = NUMBER OF CHARS TO WRITE                          :
                            C   ;       DS = DATA SEGMENT                                      :
                            C   ;       ES = REGEN SEGMENT                                     :
                            C   ; EXIT                                                         :
                            C   ;       NOTHING IS RETURNED                                    :
                            C   ;                                                              :
                            C   ; GRAPHICS READ                                                :
                            C   ;       THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT  :
                            C   ;       CURSOR POSITION ON THE SCREEN BY MATCHING THE DOTS ON  :
                            C   ;       THE SCREEN TO THE CHARACTER GENERATOR CODE POINTS      :
                            C   ; ENTRY                                                        :
                            C   ;       NONE  (0 IS ASSUMED AS THE BACKGROUND COLOR)           :
                            C   ; EXIT                                                         :
                            C   ;       AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF    :
                            C   ;            NONE FOUND)                                       :
                            C   ;                                                              :
                            C   ; FOR COMPATIBILITY ROUTINES, THE IMAGES USED TO FORM CHARS ARE :
                            C   ;  CONTAINED IN ROM FOR THE 1ST 128 CHARS.  TO ACCESS CHARS    :
                            C   ;  IN THE SECOND HALF, THE USER MUST INITIALIZE THE VECTOR AT  :
                            C   ;  INTERRUPT 1FH (LOCATION 0007CH) TO POINT TO THE USER        :
                            C   ;  SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).               :
                            C   ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS              :
                            C   ;--------------------------------------------------------------
                            C           ASSUME  CS:CODE,DS:ABSO,ES:NOTHING
1920                        C   GRAPHICS_WRITE  PROC    NEAR
1920  80 FC 07              C           CMP     AH,7
1923  72 03                 C           JB      S1_A
1925  E9 19D7 R             C           JMP     GRX_WRT
1928                        C   S1_A:
1928  E8 16EB R             C           CALL    MK_ES
192B  B4 00                 C           MOV     AH,0                        ; 0 TO HIGH OF CODE POINT
192D  50                    C           PUSH    AX                          ; SAVE CODE POINT VALUE
                            C
                            C   ;----- DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
                            C
192E  E8 16A4 R             C           CALL    S26                         ; LOC IN REGEN BUFFER
1931  8B F8                 C           MOV     DI,AX                       ; REGEN POINTER IN DI
                            C
                            C   ;----- DETERMINE REGION TO GET CODE POINTS FROM
                            C
1933  58                    C           POP     AX                          ; RECOVER CODE POINT
1934  3C 80                 C           CMP     AL,80H                      ;  IS IT IN SECOND HALF
1936  73 06                 C           JAE     S1                          ; YES
                            C
                            C   ;----- IMAGE IS IN FIRST HALF, CONTAINED IN ROM
                            C
1938  C5 36 010C R          C           LDS     SI,GRX_SET
193C  EB 06                 C           JMP     SHORT S2                    ; DETERMINE_MODE
                            C
                            C   ;----- IMAGE IS IN SECOND HALF, IN USER RAM
                            C
193E                        C   S1:                                         ; EXTEND_CHAR
193E  2C 80                 C           SUB     AL,80H                      ; 0 ORIGIN FOR SECOND HALF
1940  C5 36 007C R          C           LDS     SI,EXT_PTR
                            C
                            C   ;----- DETERMINE GRAPHICS MODE IN OPERATION
                            C
1944                        C   S2:                                         ; DETERMINE_MODE
1944  D1 E0                 C           SAL     AX,1                        ; MULTIPLY CODE POINT
1946  D1 E0                 C           SAL     AX,1                        ;  VALUE BY 8
1948  D1 E0                 C           SAL     AX,1
194A  03 F0                 C           ADD     SI,AX                       ; SI HAS OFFSET OF
194C  1E                    C           PUSH    DS                          ;  DESIRES CODES
194D  E8 0CFE R             C           CALL    DDS
1950  80 3E 0449 R 06       C           CMP     CRT_MODE,6
1955  1F                    C           POP     DS
1956  72 2C                 C           JC      S7                          ; TEST FOR MEDIUM RES MODE
                            C
                            C   ;----- HIGH RESOLUTION MODE
                            C
1958                        C   S3:                                         ; HIGH_CHAR
1958  57                    C           PUSH    DI                          ; SAVE REGEN POINTER
1959  56                    C           PUSH    SI                          ; SAVE CODE POINTER
195A  B6 04                 C           MOV     DH,4                        ; NUMBER OF TIMES THROUGH
195C                        C   S4:                                         ;  LOOP
195C  AC                    C           LODSB                               ; GET BYTE FROM CODE POINT
195D  F6 C3 80              C           TEST    BL,80H                      ; SHOULD WE USE THE
1960  75 16                 C           JNZ     S6                          ;  FUNCTION TO PUT CHAR IN
1962  AA                    C           STOSB                               ; STORE IN REGEN BUFFER
1963  AC                    C           LODSB
1964                        C   S5:
1964  26: 88 85 1FFF        C           MOV     ES:[DI+2000H-1],AL          ; STORE IN SECOND HALF
1969  83 C7 4F              C           ADD     DI,79                       ; MOVE TO NEXT ROW IN REGEN
196C  FE CE                 C           DEC     DH                          ; DONE WITH LOOP
196E  75 EC                 C           JNZ     S4
1970  5E                    C           POP     SI
```

```
1971  5F               C              POP    DI                         ; RECOVER REGEN POINTER
1972  47               C              INC    DI                         ; POINT TO NEXT CHAR POS
1973  E2 E3            C              LOOP   S3                         ; MORE CHARS TO WRITE
1975  E9 219E R        C              JMP    V_RET
1978                   C      S6:
1978  26: 32 05        C              XOR    AL,ES:[DI]                 ; XOR WITH CURRENT
197B  AA               C              STOSB                             ; STORE THE CODE POINT
197C  AC               C              LODSB                             ; AGAIN FOR ODD FIELD
197D  26: 32 85 1FFF   C              XOR    AL,ES:[DI+2000H-1]
1982  EB E0            C              JMP    S5                         ; BACK TO MAINSTREAM
                       C
                       C      ;----- MEDIUM RESOLUTION WRITE
                       C
1984                   C      S7:                                       ; MED_RES_WRITE
1984  8A D3            C              MOV    DL,BL                      ; SAVE HIGH COLOR BIT
1986  D1 E7            C              SAL    DI,1                       ; OFFSET*2, 2 BYTES/CHAR
1988  E8 166D R        C              CALL   S19                        ; EXPAND BL TO FULL WORD
198B                   C      S8:                                       ;  OF COLOR
198B  57               C              PUSH   DI                         ; SAVE REGEN POINTER
198C  56               C              PUSH   SI                         ; SAVE THE CODE POINTER
198D  B6 04            C              MOV    DH,4                       ; NUMBER OF LOOPS
198F                   C      S9:
198F  AC               C              LODSB                             ; GET CODE POINT
1990  E8 1682 R        C              CALL   S21                        ; DOUBLE UP ALL THE BITS
1993  23 C3            C              AND    AX,BX                      ; CONVERT THEM TO FORE-
                       C                                                ;  GROUND COLOR (0 BACK)
1995  F6 C2 80         C              TEST   DL,80H                     ; IS THIS XOR FUNCTION
1998  74 07            C              JZ     S10                        ; NO, STORE IT IN AS IT IS
199A  26: 32 25        C              XOR    AH,ES:[DI]                 ; DO FUNCTION WITH HALF
199D  26: 32 45 01     C              XOR    AL,ES:[DI+1]               ;  AND WITH OTHER HALF
19A1                   C      S10:
19A1  26: 88 25        C              MOV    ES:[DI],AH                 ; STORE FIRST BYTE
19A4  26: 88 45 01     C              MOV    ES:[DI+1],AL               ; STORE SECOND BYTE
19A8  AC               C              LODSB                             ; GET CODE POINT
19A9  E8 1682 R        C              CALL   S21
19AC  23 C3            C              AND    AX,BX                      ; CONVERT TO COLOR
19AE  F6 C2 80         C              TEST   DL,80H                     ; IS THIS XOR FUNCTION
19B1  74 0A            C              JZ     S11                        ; NO, JUST STORE THE VALUE
19B3  26: 32 A5 2000   C              XOR    AH,ES:[DI+2000H]           ; FUNCTION WITH FIRST HALF
19B8  26: 32 85 2001   C              XOR    AL,ES:[DI+2001H]           ; AND WITH SECOND HALF
19BD                   C      S11:
19BD  26: 88 A5 2000   C              MOV    ES:[DI+2000H],AH
19C2  26: 88 85 2001   C              MOV    ES:[DI+2000H+1],AL         ; STORE IN SECOND PORTION
19C7  83 C7 50         C              ADD    DI,80                      ; POINT TO NEXT LOCATION
19CA  FE CE            C              DEC    DH
19CC  75 C1            C              JNZ    S9                         ; KEEP GOING
19CE  5E               C              POP    SI                         ; RECOVER CODE PONTER
19CF  5F               C              POP    DI                         ; RECOVER REGEN POINTER
19D0  47               C              INC    DI                         ; POINT TO NEXT CHAR
19D1  47               C              INC    DI
19D2  E2 B7            C              LOOP   S8                         ; MORE TO WRITE
19D4  E9 219E R        C              JMP    V_RET
19D7                   C      GRAPHICS_WRITE  ENDP
                       C
                       C      ;-------------------------------------------------------------
                       C      ; ENTRY                                                       :
                       C      ;       AL = CHAR TO WRITE                                    :
                       C      ;       BH = DISPLAY PAGE                                     :
                       C      ;       BL = ATTRIBUTE/COLOR                                  :
                       C      ;       CX = COUNT OF CHARS TO WRITE                          :
                       C      ;-------------------------------------------------------------
19D7                   C      GRX_WRT PROC    NEAR
                       C              ASSUME  DS:ABS0, ES:NOTHING
19D7  80 FC 0F         C              CMP    AH,0FH                     ; 640X350 GRAPHICS
19DA  72 0E            C              JB     NO_ADJ1
19DC  E8 14F7 R        C              CALL   MEM_DET                    ; BASE CARD
19DF  72 09            C              JC     NO_ADJ1
19E1  80 E3 85         C              AND    BL,10000101B               ; 85H, XOR C2 C0 MASK
19E4  8A E3            C              MOV    AH,BL
19E6  D0 E4            C              SHL    AH,1                       ; EXPAND C0 TO C1, C2 TO C3
19E8  0A DC            C              OR     BL,AH                      ; BUILD ?(80H) + (0,3,C,F)
19EA                   C      NO_ADJ1:
19EA  2A E4            C              SUB    AH,AH                      ; ZERO
19EC  F7 26 0485 R     C              MUL    POINTS                     ; OFFSET FONT TABLE BASE
19F0  50               C              PUSH   AX                         ; FONT TABLE DISPLACEMENT
19F1  E8 16BA R        C              CALL   GR_CUR                     ; GET OFFSET INTO REGEN
19F4  8B F8            C              MOV    DI,AX                      ;  INTO DESTINATION
19F6  8B 2E 0485 R     C              MOV    BP,POINTS                  ; BYTES PER CHAR
                       C              SRLOAD ES,0A000H                  ; REGEN SEGEMNT
19FA  BA A000          C+             MOV    DX,0A000H
19FD  8E C2            C+             MOV    ES,DX
19FF  C5 36 010C R     C              LDS    SI,GRX_SET                 ; ADDRESSING TO FONTS
1A03  58               C              POP    AX                         ; RECOVER OFFSET
1A04  03 F0            C              ADD    SI,AX                      ; CHARACTER IN TABLE
1A06  B6 03            C              MOV    DH,3
1A08                   C      S20A:
1A08  F6 C3 80         C              TEST   BL,080H                    ; TEST FOR XOR
1A0B  74 0B            C              JZ     NO_XOR                     ; NO XOR
1A0D  B2 CE            C              MOV    DL,GRAPH_ADDR
1A0F  B8 0318          C              MOV    AX,0318H                   ; GRAPHICS CHIP XOR
1A12  E8 0D15 R        C              CALL   OUT_DX                     ; SET REGISTER
1A15  EB 1E 90         C              JMP    F_2                        ; SKIP BLANK
1A18                   C      NO_XOR:                                   ; BLANK BOX FOR CHAR
1A18  57               C              PUSH   DI                         ; SAVE REGEN POINTER
1A19  B2 C4            C              MOV    DL,SEQ_ADDR
1A1B  B8 020F          C              MOV    AX,020FH                   ; ENABLE ALL MAPS
1A1E  E8 0D15 R        C              CALL   OUT_DX
1A21  2B C0            C              SUB    AX,AX                      ; STORE ZERO
1A23  51               C              PUSH   CX                         ; SAVE CHARACTER COUNT
1A24  8B CD            C              MOV    CX,BP                      ; GET BYTE COUNT
1A26  1E               C              PUSH   DS
1A27  E8 0CFE R        C              CALL   DDS
1A2A                   C      S13A:
1A2A  AA               C              STOSB                             ; ZERO REGEN BYTE
1A2B  03 3E 044A R     C              ADD    DI,CRT_COLS                ; NEXT BYTE OF BOX
1A2F  4F               C              DEC    DI                         ; ADJUST
1A30  E2 F8            C              LOOP   S13A                       ; NEXT BYTE
1A32  1F               C              POP    DS
1A33  59               C              POP    CX                         ; RECOVER CHARACTER COUNT
1A34  5F               C              POP    DI                         ; RECOVER REGEN POINTER
1A35                   C      F_2:
1A35  B2 C4            C              MOV    DL,SEQ_ADDR
1A37  B4 02            C              MOV    AH,02H                     ; SET MAP MASK
1A39  8A C3            C              MOV    AL,BL                      ;  FOR COLOR
1A3B  E8 0D15 R        C              CALL   OUT_DX                     ; SET THE CHIP
1A3E  57               C              PUSH   DI                         ; SAVE OFFSET IN REGEN
1A3F  53               C              PUSH   BX                         ; SAVE COLOR VALUE
1A40  51               C              PUSH   CX                         ; SAVE CHACTER COUNT
1A41  8B DD            C              MOV    BX,BP                      ; LOOP CONTROL, BYTES/CHAR
1A43  1E               C              PUSH   DS                         ; SAVE FONT SEGMENT
1A44  E8 0CFE R        C              CALL   DDS                        ; SET LOW RAM SEGMENT
                       C              ASSUME DS:ABS0
1A47  8B 0E 044A R     C              MOV    CX,CRT_COLS                ; GET COLUMN COUNT
1A4B  1F               C              POP    DS                         ; RESTORE FONT SEGMENT
                       C              ASSUME DS:NOTHING
1A4C                   C      S1K:                                      ; WRITE OUT THE CHARACTER
```

```
1A4C  8A 04                 C          MOV     AL,DS:[SI]                  ; CODE POINT
1A4E  26: 8A 25             C          MOV     AH,ES:[DI]                  ; LATCH DATA
1A51  26: 88 05             C          MOV     ES:[DI],AL                  ; WRITE ONE BYTE OF FONT
1A54  46                    C          INC     SI                          ; NEXT FONT POINT
1A55  03 F9                 C          ADD     DI,CX                       ; ONE ROW BELOW LAST POINT
1A57  4B                    C          DEC     BX                          ; BYTES PER CHAR COUNTER
1A58  75 F2                 C          JNZ     S1K                         ; DO NEXT ROW OF CHARACTER
                            C
1A5A  59                    C          POP     CX                          ; CHARACTER COUNT
1A5B  5B                    C          POP     BX                          ; COLOR VALUE
1A5C  2B F5                 C          SUB     SI,BP                       ; ADJUST PTR TO FONT TABLE
1A5E  5F                    C          POP     DI                          ; REGEN POINTER
1A5F  47                    C          INC     DI                          ; NEXT CHAR POSN IN REGEN
1A60  E2 A6                 C          LOOP    S20A                        ; WRITE ANOTHER CHARACTER
                            C
1A62  B2 CE                 C          MOV     DL,GRAPH_ADDR
1A64  B8 0300               C          MOV     AX,0300H                    ; NORMAL WRITE, NO ROTATE
1A67  E8 0D15 R             C          CALL    OUT_DX                      ; SET THE CHIP
1A6A  B2 C4                 C          MOV     DL,SEQ_ADDR
1A6C  B8 020F               C          MOV     AX,020FH                    ; ENABLE ALL MAPS
1A6F  E8 0D15 R             C          CALL    OUT_DX                      ; SET THE CHIP
1A72  E9 219E R             C          JMP     V_RET
1A75                        C   GRX_WRT ENDP
                            C
                            C          SUBTTL

                                ;----- SET COLOR PALETTE

1A75                            AHB:
                                       ASSUME  DS:ABS0
1A75  80 3E 0463 R B4                  CMP     BYTE PTR ADDR_6845,0B4H
1A7A  74 09                            JE      M21_B                       ; CALL VALID ONLY FOR COLOR
1A7C  F6 06 0487 R 02                  TEST    INFO,2                      ; SEE IF ITS THE OLD COLOR CARD
1A81  74 05                            JZ      M21_A                       ; IF NOT, HANDLE IT HERE
1A83  CD 42                            INT     42H                         ; OLD CODE CALL
1A85                            M21_B:
1A85  E9 219E R                        JMP     V_RET                       ; BACK TO CALLER
1A88                            M21_A:
1A88  2B C0                            SUB     AX,AX
1A8A  8B E8                            MOV     BP,AX
1A8C  C4 3E 04A8 R                     LES     DI,SAVE_PTR
1A90  83 C7 04                         ADD     DI,4
1A93  26: C4 3D                        LES     DI,DWORD PTR ES:[DI]
1A96  8C C0                            MOV     AX,ES
1A98  0B C7                            OR      AX,DI
1A9A  74 01                            JZ      NOT4AHB
1A9C  45                               INC     BP
1A9D                            NOT4AHB:
1A9D  E8 1DC0 R                        CALL    PAL_INIT
1AA0  0A FF                            OR      BH,BH
1AA2  75 65                            JNZ     M20

                                ;----- HANDLE BH = 0 HERE
                                ;      ALPHA MODES  => BL = OVERSCAN COLOR
                                ;      GRAPHICS     => BL = OVERSCAN AND BACKGROUND COLOR

                                ;----- MOVE INTENSITY BIT FROM D3 TO D4 FOR COMPATIBILITY

1AA4  8A FB                            MOV     BH,BL
1AA6  A0 0466 R                        MOV     AL,CRT_PALETTE
1AA9  24 E0                            AND     AL,0E0H
1AAB  80 E3 1F                         AND     BL,01FH
1AAE  0A C3                            OR      AL,BL
1AB0  A2 0466 R                        MOV     CRT_PALETTE,AL
1AB3  8A DF                            MOV     BL,BH
1AB5  80 E7 08                         AND     BH,08H
1AB8  D0 E7                            SHL     BH,1
1ABA  8A E8                            MOV     CH,AL
1ABC  80 E5 EF                         AND     CH,0EFH
1ABF  0A ED                            OR      CH,CH
1AC1  80 E3 0F                         AND     BL,0FH
1AC4  8A FB                            MOV     BH,BL
1AC6  D0 E3                            SHL     BL,1
1AC8  80 E3 10                         AND     BL,010H
1ACB  80 E7 07                         AND     BH,07H
1ACE  0A DF                            OR      BL,BH

1AD0  A0 0449 R                        MOV     AL,CRT_MODE
1AD3  3C 03                            CMP     AL,3
1AD5  76 0E                            JBE     M21

                                ;----- GRAPHICS MODE DONE HERE (SET PALETTE 0 AND OVERSCAN)

1AD7  B4 00                            MOV     AH,0
1AD9  8A C3                            MOV     AL,BL
1ADB  E8 1D9F R                        CALL    PAL_SET

1ADE  0B ED                            OR      BP,BP
1AE0  74 03                            JZ      M21
1AE2  26: 88 1D                        MOV     ES:[DI],BL

                                ;----- ALPHA MODE DONE HERE (SET OVERSCAN REGISTER)

1AE5                            M21:
1AE5  80 3E 0449 R 03                  CMP     CRT_MODE,3                  ; CHECK FOR AN ENHANCED MODE
1AEA  77 05                            JA      SET_OVRSC                   ; NO CHANCE
1AEC  E8 0E9A R                        CALL    BRST_DET                    ; SEE IF WE ARE ENHANCED
1AEF  72 07                            JC      SKIP_OVRSC                  ; THERE IS NO BORDER
1AF1                            SET_OVRSC:
1AF1  B4 11                            MOV     AH,011H                     ; OVERSCAN REGISTER
1AF3  8A C3                            MOV     AL,BL
1AF5  E8 1D9F R                        CALL    PAL_SET                     ; SET THE BORDER
1AF8                            SKIP_OVRSC:
1AF8  0B ED                            OR      BP,BP
1AFA  74 04                            JZ      M21Y
1AFC  26: 88 5D 10                     MOV     ES:[DI][16D],BL
1B00                            M21Y:
1B00  8A DD                            MOV     BL,CH
1B02  80 E3 20                         AND     BL,020H
1B05  B1 05                            MOV     CL,5
1B07  D2 EB                            SHR     BL,CL

                                ;----- HANDLE BH = 1 HERE
                                ;      ALPHA MODES  => NO EFFECT
                                ;      GRAPHICS     => LOW BIT OF BL = 0
                                ;                         PALETTE 0 = BACKGROUND
                                ;                         PALETTE 1 = GREEN
                                ;                         PALETTE 2 = RED
                                ;                         PALETTE 3 = BROWN
                                ;                   => LOW BIT OF BL = 1
                                ;                         PALETTE 0 = BACKGROUND
                                ;                         PALETTE 1 = CYAN
                                ;                         PALETTE 2 = MAGENTA
                                ;                         PALETTE 3 = WHITE
                                ;
```

```
1B09                                    M20:
1B09  80 3E 0449 R 03                           CMP     CRT_MODE,3
1B0E  76 4A                                     JBE     M80

1B10  A0 0466 R                                 MOV     AL,CRT_PALETTE
1B13  24 DF                                     AND     AL,0DFH
1B15  80 E3 01                                  AND     BL,1
1B18  74 02                                     JZ      M22
1B1A  0C 20                                     OR      AL,020H
1B1C                                    M22:
1B1C  A2 0466 R                                 MOV     CRT_PALETTE,AL
1B1F  24 10                                     AND     AL,010H
1B21  0C 02                                     OR      AL,2
1B23  0A D8                                     OR      BL,AL
1B25  B4 01                                     MOV     AH,1
1B27  8A C3                                     MOV     AL,BL
1B29  E8 1D9F R                                 CALL    PAL_SET

1B2C  0B ED                                     OR      BP,BP
1B2E  74 04                                     JZ      M22Y
1B30  26: 88 5D 01                              MOV     ES:[DI][1],BL
1B34                                    M22Y:

1B34  FE C3                                     INC     BL
1B36  FE C3                                     INC     BL
1B38  B4 02                                     MOV     AH,2
1B3A  8A C3                                     MOV     AL,BL
1B3C  E8 1D9F R                                 CALL    PAL_SET

1B3F  0B ED                                     OR      BP,BP
1B41  74 04                                     JZ      M27Y
1B43  26: 88 5D 02                              MOV     ES:[DI][2],BL
1B47                                    M27Y:

1B47  FE C3                                     INC     BL
1B49  FE C3                                     INC     BL
1B4B  B4 03                                     MOV     AH,3
1B4D  8A C3                                     MOV     AL,BL
1B4F  E8 1D9F R                                 CALL    PAL_SET

1B52  0B ED                                     OR      BP,BP
1B54  74 04                                     JZ      M80
1B56  26: 88 5D 03                              MOV     ES:[DI][3],BL

1B5A                                    M80:
1B5A  E8 1DB7 R                                 CALL    PAL_ON
1B5D  E9 219E R                                 JMP     V_RET

                                     C          INCLUDE         VDOT.INC
                                     C          SUBTTL  VDOT.INC
                                     C          PAGE
                                     C  ;------------------------------------------------
                                     C  ; ENTRY                                          :
                                     C  ;       DX = ROW                                 :
                                     C  ;       CX = COLUMN                              :
                                     C  ;       BH = PAGE                                :
                                     C  ; EXIT                                           :
                                     C  ;       BX = OFFSET INTO REGEN                   :
                                     C  ;       AL = BIT MASK FOR COLUMN BYTE            :
                                     C  ;------------------------------------------------
1B60                                 C  DOT_SUP_1       PROC    NEAR
                                     C
                                     C  ;----- OFFSET = PAGE OFFSET + ROW * BYTES/ROW + COLUMN/8
                                     C
1B60  F7 26 044A R                   C          MUL     WORD PTR CRT_COLS               ; ROW * BYTES/ROW
1B64  51                             C          PUSH    CX                              ; SAVE COLUMN VALUE
1B65  D1 E9                          C          SHR     CX,1                            ; DIVIDE BY EIGHT TO
1B67  D1 E9                          C          SHR     CX,1                            ;  DETERMINE THE BYTE THAT
1B69  D1 E9                          C          SHR     CX,1                            ;  THIS DOT IS IN
                                     C                                                  ;  (8 BITS/BYTE)
1B6B  03 C1                          C          ADD     AX,CX                           ; BYTE OFFSET INTO PAGE
1B6D  8A DF                          C          MOV     BL,BH                           ; GET PAGE INTO BL
1B6F  2A FF                          C          SUB     BH,BH                           ; ZERO
1B71  8B CB                          C          MOV     CX,BX                           ; COUNT VALUE
1B73  8B 1E 044C R                   C          MOV     BX,CRT_LEN                      ; LENGTH OF ONE PAGE
1B77  E3 04                          C          JCXZ    DS_2                            ; PAGE ZERO
1B79                                 C  DS_3:
1B79  03 C3                          C          ADD     AX,BX                           ; BUMP TO NEXT PAGE
1B7B  E2 FC                          C          LOOP    DS_3                            ; DO FOR THE REST
1B7D                                 C  DS_2:
1B7D  59                             C          POP     CX                              ; RECOVER COLUMN VALUE
1B7E  8B D8                          C          MOV     BX,AX                           ; REGEN OFFSET
1B80  80 E1 07                       C          AND     CL,07H                          ; SHIFT COUNT FOR BIT MASK
1B83  B0 80                          C          MOV     AL,080H                         ; MASK BIT
1B85  D2 E8                          C          SHR     AL,CL                           ; POSITION MASK BIT
1B87  C3                             C          RET
1B88                                 C  DOT_SUP_1       ENDP
                                     C
                                     C  ;--------------------------------------------------------
                                     C  ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION    :
                                     C  ; OF THE INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.   :
                                     C  ; ENTRY --                                              :
                                     C  ;  DX = ROW VALUE (0-199)                               :
                                     C  ;  CX = COLUMN VALUE (0-639)                            :
                                     C  ; EXIT --                                               :
                                     C  ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST   :
                                     C  ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST          :
                                     C  ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH   :
                                     C  ;  DH = # BITS IN RESULT                                :
                                     C  ;--------------------------------------------------------
1B88                                 C  R3      PROC    NEAR
1B88  53                             C          PUSH    BX                      ; SAVE BX DURING OPERATION
1B89  50                             C          PUSH    AX                      ; WILL SAVE AL DURING OPERATION
                                     C
                                     C  ;----- DETERMINE 1ST BYTE IN IDICATED ROW BY MULTIPLYING ROW VALUE BY 40
                                     C  ;----- ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW
                                     C
1B8A  B0 28                          C          MOV     AL,40
1B8C  52                             C          PUSH    DX                      ; SAVE ROW VALUE
1B8D  80 E2 FE                       C          AND     DL,0FEH                 ; STRIP OFF ODD/EVEN BIT
1B90  F6 E2                          C          MUL     DL                      ; AX HAS ADDRESS OF 1ST BYTE
                                     C                                          ;  OF INDICATED ROW
1B92  5A                             C          POP     DX                      ; RECOVER IT
1B93  F6 C2 01                       C          TEST    DL,1                    ; TEST FOR EVEN/ODD
1B96  74 03                          C          JZ      R4                      ; JUMP IF EVEN ROW
1B98  05 2000                        C          ADD     AX,2000H                ; OFFSET TO LOCATION OF ODD ROWS
1B9B                                 C  R4:                                     ; EVEN_ROW
1B9B  8B F0                          C          MOV     SI,AX                   ; MOVE POINTER TO SI
1B9D  58                             C          POP     AX                      ; RECOVER AL VALUE
1B9E  8B D1                          C          MOV     DX,CX                   ; COLUMN VALUE TO DX
                                     C
                                     C  ;----- DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
                                     C
                                     C  ;----------------------------------------------------------------
                                     C  ; SET UP THE REGISTERS ACCORDING TO THE MODE                    :
                                     C  ;  CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES)  :
```

```
                          C   ;   CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M)         :
                          C   ;   BL = MASK TO SELECT BITS FROM POINTED BYTE (80H/C0H FOR H/M) :
                          C   ;   BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M)      :
                          C   ·--------------------------------------------------------------------
                          C
1BA0  BB 02C0             C           MOV     BX,2C0H
1BA3  B9 0302             C           MOV     CX,302H                  ; SET PARMS FOR MED RES
1BA6  80 3E 0449 R 06     C           CMP     CRT_MODE,6
1BAB  72 06               C           JC      R5                       ; HANDLE IF MED ARES
1BAD  BB 0180             C           MOV     BX,180H
1BB0  B9 0703             C           MOV     CX,703H                  ; SET PARMS FOR HIGH RES
                          C
                          C   ;----- DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
                          C
1BB3                      C   R5:
1BB3  22 EA               C           AND     CH,DL                    ; ADDRESS OF PEL WITHIN BYTE TO CH
                          C
                          C   ;----- DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
                          C
1BB5  D3 EA               C           SHR     DX,CL                    ; SHIFT BY CORRECT AMOUNT
1BB7  03 F2               C           ADD     SI,DX                    ; INCREMENT THE POINTER
1BB9  8A F7               C           MOV     DH,BH                    ; GET THE # OF BITS IN RESULT TO DH
                          C
                          C   ;----- MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
                          C
1BBB  2A C9               C           SUB     CL,CL                    ; ZERO INTO STORAGE LOCATION
1BBD                      C   R6:
1BBD  D0 C8               C           ROR     AL,1                     ; LEFT JUSTIFY THE VALUE
                          C                                            ;  IN AL (FOR WRITE)
1BBF  02 CD               C           ADD     CL,CH                    ; ADD IN THE BIT OFFSET VALUE
1BC1  FE CF               C           DEC     BH                       ; LOOP CONTROL
1BC3  75 F8               C           JNZ     R6                       ; ON EXIT, CL HAS SHIFT COUNT
                          C                                            ;  TO RESTORE BITS
1BC5  8A E3               C           MOV     AH,BL                    ; GET MASK TO AH
1BC7  D2 EC               C           SHR     AH,CL                    ; MOVE THE MASK TO CORRECT LOCATION
1BC9  5B                  C           POP     BX                       ; RECOVER REG
1BCA  C3                  C           RET                              ; RETURN WITH EVERYTHING SET UP
1BCB                      C   R3      ENDP
                          C
                          C   ;--------------------------------------------------------------------
                          C   ; READ DOT  -- WRITE DOT                                            :
                          C   ;       THESE ROUTINES WILL WRITE A DOT, OR READ THE DOT AT         :
                          C   ;       THE INDICATED LOCATION                                      :
                          C   ; ENTRY --                                                          :
                          C   ;   DX = ROW (0-199)    (THE ACTUAL VALUE DEPENDS ON THE MODE)      :
                          C   ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )       :
                          C   ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,       :
                          C   ;       REQ'D FOR WRITE DOT ONLY, RIGHT JUSTIFIED)                  :
                          C   ;       BIT 7 OF AL=1 INDICATES XOR THE VALUE INTO THE LOCATION     :
                          C   ;   DS = DATA SEGMENT                                               :
                          C   ;   ES = REGEN SEGMENT                                              :
                          C   ;                                                                   :
                          C   ; EXIT                                                              :
                          C   ;       AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY             :
                          C   ;--------------------------------------------------------------------
                          C
                          C   ;----- WRITE DOT
                          C
1BCB                      C   AHC:
                          C           ASSUME  DS:ABS0
1BCB  80 3E 0449 R 07     C           CMP     CRT_MODE,7
1BD0  77 2A              .C           JA      WRITE_DOT_2
                          C
1BD2                      C   WRITE_DOT       PROC    NEAR
                          C           ASSUME  DS:ABS0,ES:NOTHING
1BD2  52                  C           PUSH    DX
                          C           SRLOAD  ES,0B800H
1BD3  BA B800             C+          MOV     DX,0B800H
1BD6  8E C2               C+          MOV     ES,DX
1BD8  5A                  C           POP     DX
1BD9  50                  C           PUSH    AX                       ; SAVE DOT VALUE
1BDA  50                  C           PUSH    AX                       ;  TWICE
1BDB  E8 1B88 R           C           CALL    R3                       ; DETERMINE BYTE POSITION OF THE DOT
1BDE  D2 E8               C           SHR     AL,CL                    ; SHIFT TO SET UP THE BITS FOR OUTPUT
1BE0  22 C4               C           AND     AL,AH                    ; STRIP OFF THE OTHER BITS
1BE2  26: 8A 0C           C           MOV     CL,ES:[SI]               ; GET THE CURRENT BYTE
1BE5  5B                  C           POP     BX                       ; RECOVER XOR FLAG
1BE6  F6 C3 80            C           TEST    BL,80H                   ; IS IT ON
1BE9  75 0D               C           JNZ     R2                       ; YES, XOR THE DOT
1BEB  F6 D4               C           NOT     AH                       ; SET THE MASK TO REMOVE THE
1BED  22 CC               C           AND     CL,AH                    ;  INDICATED BITS
1BEF  0A C1               C           OR      AL,CL                    ; OR IN THE NEW VALUE OF THOSE BITS
1BF1                      C   R1:                                      ; FINISH_DOT
1BF1  26: 88 04           C           MOV     ES:[SI],AL               ; RESTORE THE BYTE IN MEMORY
1BF4  58                  C           POP     AX
1BF5  E9 219E R           C           JMP     V_RET
1BF8                      C   R2:                                      ; XOR_DOT
1BF8  32 C1               C           XOR     AL,CL                    ; EXCLUSIVE OR THE DOTS
1BFA  EB F5               C           JMP     R1                       ; FINISH UP THE WRITING
1BFC                      C   WRITE_DOT       ENDP
                          C
1BFC                      C   WRITE_DOT_2     PROC    NEAR
1BFC  80 3E 0449 R 0F     C           CMP     CRT_MODE,0FH
1C01  72 0D               C           JB      NO_ADJ2
1C03  E8 14F7 R           C           CALL    MEM_DET                           ; BASE CARD
1C06  72 08               C           JC      NO_ADJ2
1C08  24 85               C           AND     AL,10000101B                      ; 85H, XOR C2 C0 MASK
1C0A  8A E0               C           MOV     AH,AL
1C0C  D0 E4               C           SHL     AH,1                              ; EXPAND C0 TO C1, C2 TO C3
1C0E  0A C4               C           OR      AL,AH                             ; BUILD ?(80H) + (0,3,C,F)
1C10                      C   NO_ADJ2:
1C10  50                  C           PUSH    AX
1C11  8B C2               C           MOV     AX,DX                             ; ROW VALUE
1C13  E8 1B60 R           C           CALL    DOT_SUP_1                         ; BX=OFFSET, AL=BIT MASK
1C16  B6 03               C           MOV     DH,3
1C18  B2 CE               C           MOV     DL,GRAPH_ADDR                     ; GRAPHICS CHIP
1C1A  B4 08               C           MOV     AH,G_BIT_MASK                     ; BIT MASK REGISTER
1C1C  E8 0D15 R           C           CALL    OUT_DX                            ; SET BIT MASK
1C1F  52                  C           PUSH    DX
                          C           SRLOAD  ES,0A000H                         ; REGEN SEGMENT
1C20  BA A000             C+          MOV     DX,0A000H
1C23  8E C2               C+          MOV     ES,DX
1C25  5A                  C           POP     DX
1C26  58                  C           POP     AX                                ; RECOVER COLOR
1C27  8A E8               C           MOV     CH,AL                             ; SAVE COLOR
1C29  F6 C5 80            C           TEST    CH,080H                           ; SEE IF XOR
1C2C  74 0A               C           JZ      WD_A                              ; NO XOR
1C2E  B4 03               C           MOV     AH,G_DATA_ROT                     ; DO XOR
1C30  B0 18               C           MOV     AL,018H                           ; XOR FUNCTION
1C32  E8 0D15 R           C           CALL    OUT_DX                            ; SET THE REGISTER
1C35  EB 12 90            C           JMP     WD_B                              ; SKIP THE BLANK
1C38                      C   WD_A:                                             ; BLANK THE DOT
1C38  B2 C4               C           MOV     DL,SEQ_ADDR                       ; SEQUENCER
1C3A  B4 02               C           MOV     AH,S_MAP                          ; MAP MASK
1C3C  B0 FF               C           MOV     AL,0FFH                           ; ENABLE ALL MAPS
1C3E  E8 0D15 R           C           CALL    OUT_DX                            ; SET THE REGISTER
```

```
1C41  26: 8A 07         C           MOV     AL,ES:[BX]              ; LATCH DATA
1C44  2A C0             C           SUB     AL,AL                   ; ZERO
1C46  26: 88 07         C           MOV     ES:[BX],AL              ; BLANK THE DOT
1C49                    C   WD_B:                                   ; SET THE COLOR MAP MASK
1C49  B2 C4             C           MOV     DL,SEQ_ADDR             ; SEQUENCER
1C4B  B4 02             C           MOV     AH,S_MAP                ; MAP MASK REGISTER
1C4D  8A C5             C           MOV     AL,CH                   ; COLOR VALUE
1C4F  24 0F             C           AND     AL,0FH                  ; VALUES 0-15
1C51  E8 0D15 R         C           CALL    OUT_DX                  ; SET IT
1C54  26: 8A 07         C           MOV     AL,ES:[BX]              ; LATCH DATA
1C57  B0 FF             C           MOV     AL,0FFH                 ; WRITE VALUE
1C59  26: 88 07         C           MOV     ES:[BX],AL              ; SET THE DOT
                        C
                        C   ;----- NORMALIZE THE ENVIRONMENT
                        C
1C5C  E8 0D15 R         C           CALL    OUT_DX                  ; ALL MAPS ON
1C5F  B2 CE             C           MOV     DL,GRAPH_ADDR           ; GRAPHICS CHIPS
1C61  B4 03             C           MOV     AH,G_DATA_ROT           ; XOR REGISTER
1C63  2A C0             C           SUB     AL,AL                   ; NORMAL WRITES
1C65  E8 0D15 R         C           CALL    OUT_DX                  ; SET IT
1C68  B4 08             C           MOV     AH,G_BIT_MASK           ; BIT MASK
1C6A  B0 FF             C           MOV     AL,0FFH                 ; ALL BITS ON
1C6C  E8 0D15 R         C           CALL    OUT_DX                  ; SET IT
1C6F  E9 219E R         C           JMP     V_RET                   ; WRITE DOT DONE
1C72                    C   WRITE_DOT_2     ENDP
                        C
1C72                    C   RD_S    PROC    NEAR
                        C           ASSUME  DS:ABS0
1C72  50                C           PUSH    AX
1C73  52                C           PUSH    DX
                        C           SRLOAD  ES,0A000H
1C74  BA A000           C+          MOV     DX,0A000H
1C77  8E C2             C+          MOV     ES,DX
1C79  5A                C           POP     DX
1C7A  58                C           POP     AX
1C7B  8B C2             C           MOV     AX,DX
1C7D  E8 1B60 R         C           CALL    DOT_SUP_1
1C80  B5 07             C           MOV     CH,7
1C82  2A E9             C           SUB     CH,CL
1C84  2B D2             C           SUB     DX,DX
1C86  B0 00             C           MOV     AL,0
1C88  C3                C           RET
1C89                    C   RD_S    ENDP
                        C
1C89                    C   RD_1S   PROC    NEAR
1C89  8A CD             C           MOV     CL,CH
1C8B  B4 04             C           MOV     AH,4
1C8D  52                C           PUSH    DX
1C8E  B6 03             C           MOV     DH,3
1C90  B2 CE             C           MOV     DL,GRAPH_ADDR
1C92  E8 0D15 R         C           CALL    OUT_DX
1C95  5A                C           POP     DX
1C96  26: 8A 27         C           MOV     AH,ES:[BX]
1C99  D2 EC             C           SHR     AH,CL
1C9B  80 E4 01          C           AND     AH,1
1C9E  C3                C           RET
1C9F                    C   RD_1S   ENDP
                        C
                        C   ;----- READ DOT
                        C
1C9F                    C   AHD:
                        C           ASSUME  DS:ABS0
1C9F  80 3E 0449 R 07   C           CMP     CRT_MODE,7
1CA4  77 18             C           JA      R_1
                        C
1CA6                    C   READ_DOT        PROC    NEAR
                        C           ASSUME  DS:ABS0,ES:NOTHING
1CA6  52                C           PUSH    DX
                        C           SRLOAD  ES,0B800H
1CA7  BA B800           C+          MOV     DX,0B800H
1CAA  8E C2             C+          MOV     ES,DX
1CAC  5A                C           POP     DX
1CAD  E8 1B88 R         C           CALL    R3                  ; DETERMINE BYTE POSITION OF DOT
1CB0  26: 8A 04         C           MOV     AL,ES:[SI]          ; GET THE BYTE
1CB3  22 C4             C           AND     AL,AH               ; MASK OFF THE OTHER BITS IN THE BYTE
1CB5  D2 E0             C           SHL     AL,CL               ; LEFT JUSTIFY THE VALUE
1CB7  8A CE             C           MOV     CL,DH               ; GET NUMBER OF BITS IN RESULT
1CB9  D2 C0             C           ROL     AL,CL               ; RIGHT JUSTIFY THE RESULT
1CBB  E9 219E R         C           JMP     V_RET
1CBE                    C   READ_DOT        ENDP
                        C
1CBE                    C   R_1:
1CBE  80 3E 0449 R 0F   C           CMP     CRT_MODE,0FH
1CC3  72 25             C           JB      READ_DOT_2
1CC5  E8 14F7 R         C           CALL    MEM_DET
1CC8  72 20             C           JC      READ_DOT_2
                        C
1CCA                    C   READ_DOT_1      PROC    NEAR            ; 2 MAPS
                        C           ASSUME  DS:ABS0, ES:NOTHING
1CCA  E8 1C72 R         C           CALL    RD_S
1CCD  E8 1C89 R         C           CALL    RD_1S
1CD0  0A D4             C           OR      DL,AH
1CD2  D0 E4             C           SHL     AH,1
1CD4  0A D4             C           OR      DL,AH
1CD6  B0 02             C           MOV     AL,2
1CD8  E8 1C89 R         C           CALL    RD_1S
1CDB  D0 E4             C           SHL     AH,1
1CDD  D0 E4             C           SHL     AH,1
1CDF  0A D4             C           OR      DL,AH
1CE1  D0 E4             C           SHL     AH,1
1CE3  0A D4             C           OR      DL,AH
1CE5  8A C2             C           MOV     AL,DL
1CE7  E9 219E R         C           JMP     V_RET
1CEA                    C   READ_DOT_1      ENDP
                        C
1CEA                    C   READ_DOT_2      PROC    NEAR            ; 4 MAPS
                        C           ASSUME  DS:ABS0, ES:NOTHING
1CEA  E8 1C72 R         C           CALL    RD_S
1CED                    C   RD_2A:
1CED  E8 1C89 R         C           CALL    RD_1S
1CF0  8A C8             C           MOV     CL,AL
1CF2  D2 E4             C           SHL     AH,CL
1CF4  0A D4             C           OR      DL,AH
1CF6  FE C0             C           INC     AL
1CF8  3C 03             C           CMP     AL,3
1CFA  76 F1             C           JBE     RD_2A
1CFC  8A C2             C           MOV     AL,DL
1CFE  E9 219E R         C           JMP     V_RET
1D01                    C   READ_DOT_2      ENDP
                        C
                        C   ;-------------------------------------------------------------------------
                        C   ; WRITE_TTY       WRITE TELETYPE TO ACTIVE PAGE                            :
                        C   ;         THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE VIDEO   :
                        C   ;         CARD.  THE INPUT CHARACTER IS WRITTEN TO THE CURRENT CURSOR      :
                        C   ;         POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION. IF THE   :
                        C   ;         CURSOR LEAVES THE LAST COLUMN OF THE FIELD,  THE COLUMN IS SET   :
```

```
                     C  ;        TO ZERO,  AND THE ROW VALUE IS INCREMENTED.   IF THE ROW VALUE  :
                     C  ;        LEAVES THE FIELD,  THE CURSOR IS PLACED ON THE LAST ROW, FIRST  :
                     C  ;        COLUMN,  AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.   WHEN  :
                     C  ;        THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE NEWLY  :
                     C  ;        BLANKED LINE IS READ FROM THE  CURSOR POSITION ON THE PREVIOUS  :
                     C  ;        LINE BEFORE THE SCROLL,  IN CHARACTER MODE.  IN GRAPHICS MODE,  :
                     C  ;        THE 0 COLOR IS USED.                                            :
                     C  ; ENTRY                                                                  :
                     C  ;        (AH) = CURRENT CRT MODE                                         :
                     C  ;        (AL) = CHARACTER TO BE WRITTEN                                  :
                     C  ;         NOTE THAT BACK SPACE, CAR RET, BELL AND LINE FEED ARE HANDLED  :
                     C  ;         AS COMMANDS RATHER THAN AS DISPLAYABLE GRAPHICS                 :
                     C  ;        (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A        :
                     C  ;         GRAPHICS MODE                                                  :
                     C  ; EXIT                                                                   :
                     C  ;        ALL REGISTERS SAVED                                             :
                     C  ;-------------------------------------------------------------------------
1D01                 C  AHE:
                     C          ASSUME  CS:CODE,DS:ABS0
1D01  50             C          PUSH    AX                              ; SAVE REGISTERS
1D02  8A 3E 0462 R   C          MOV     BH,ACTIVE_PAGE                  ; GET THE ACTIVE PAGE
1D06  53             C          PUSH    BX                              ; SAVE
1D07  8A DF          C          MOV     BL,BH                           ; GET PAGE TO BL
1D09  32 FF          C          XOR     BH,BH                           ; CLEAR HIGH BYTE
1D0B  D1 E3          C          SAL     BX,1                            ; *2 FOR WORD OFFSET
1D0D  8B 97 0450 R   C          MOV     DX,[BX + OFFSET CURSOR_POSN]    ; CURSOR, ACTIVE PAGE
1D11  5B             C          POP     BX                              ; RECOVER
                     C
                     C  ;----- DX NOW HAS THE CURRENT CURSOR POSITION
                     C
1D12  3C 0D          C          CMP     AL,0DH                          ; IS IT CARRIAGE RETURN
1D14  74 5C          C          JE      U9                              ; CAR_RET
1D16  3C 0A          C          CMP     AL,0AH                          ; IS IT A LINE FEED
1D18  74 5C          C          JE      U10                             ; LINE_FEED
1D1A  3C 08          C          CMP     AL,08H                          ; IS IT A BACKSPACE
1D1C  74 4C          C          JE      U8                              ; BACK_SPACE
1D1E  3C 07          C          CMP     AL,07H                          ; IS IT A BELL
1D20  74 5C          C          JE      U11                             ; BELL
                     C
                     C  ;----- WRITE THE CHAR TO THE SCREEN
                     C
1D22  B4 0A          C          MOV     AH,10                           ; WRITE CHAR ONLY
1D24  B9 0001        C          MOV     CX,1                            ; ONLY ONE CHAR
1D27  CD 10          C          INT     10H                             ; WRITE THE CHAR
                     C
                     C  ;----- POSITION THE CURSOR FOR NEXT CHAR
                     C
1D29  FE C2          C          INC     DL
1D2B  3A 16 044A R   C          CMP     DL,BYTE PTR CRT_COLS            ; TEST FOR COLUMN OVERFLOW
1D2F  75 35          C          JNZ     U7                              ; SET_CURSOR
1D31  2A D2          C          SUB     DL,DL                           ; COLUMN FOR CURSOR
1D33  3A 36 0484 R   C          CMP     DH,ROWS
1D37  75 2B          C          JNZ     U6                              ; SET_CURSOR_INC
                     C
                     C  ;----- SCROLL REQUIRED
                     C
1D39                 C  U1:
1D39  E8 115D R      C          CALL    SET_CPOS                        ; SET THE CURSOR
                     C
                     C  ;----- DETERMINE VALUE TO FILL WITH DURING SCROLL
                     C
1D3C  A0 0449 R      C          MOV     AL,CRT_MODE                     ; GET THE CURRENT MODE
1D3F  3C 04          C          CMP     AL,4
1D41  72 06          C          JB      U2                              ; READ-CURSOR
1D43  2A FF          C          SUB     BH,BH                           ; FILL WITH BACKGROUND
1D45  3C 07          C          CMP     AL,7
1D47  75 06          C          JNE     U3                              ; SCROLL-UP
1D49                 C  U2:                                             ; READ-CURSOR
1D49  B4 08          C          MOV     AH,8
1D4B  CD 10          C          INT     10H                             ; READ CHAR/ATTR
1D4D  8A FC          C          MOV     BH,AH                           ; STORE IN BH
1D4F                 C  U3:                                             ; SCROLL-UP
1D4F  B8 0601        C          MOV     AX,601H                         ; SCROLL ONE LINE
1D52  2B C9          C          SUB     CX,CX                           ; UPPER LEFT CORNER
1D54  8A 36 0484 R   C          MOV     DH,ROWS                         ; LOWER RIGHT ROW
1D58  8A 16 044A R   C          MOV     DL,BYTE PTR CRT_COLS            ; LOWER RIGHT COLUMN
1D5C  FE CA          C          DEC     DL
1D5E                 C  U4:                                             ; VIDEO-CALL-RETURN
1D5E  CD 10          C          INT     10H                             ; SCROLL UP THE SCREEN
1D60                 C  U5:                                             ; TTY-RETURN
1D60  58             C          POP     AX                              ; RESTORE THE CHARACTER
1D61  E9 219E R      C          JMP     V_RET                           ; RETURN TO CALLER
1D64                 C  U6:                                             ; SET-CURSOR-INC
1D64  FE C6          C          INC     DH                              ; NEXT ROW
1D66                 C  U7:                                             ; SET-CURSOR
1D66  B4 02          C          MOV     AH,2
1D68  EB F4          C          JMP     U4                              ; ESTABLISH THE NEW CURSOR
                     C
                     C  ;----- BACK SPACE FOUND
                     C
1D6A                 C  U8:
1D6A  0A D2          C          OR      DL,DL                           ; ALREADY AT END OF LINE
1D6C  74 F8          C          JZ      U7                              ; SET CURSOR
1D6E  FE CA          C          DEC     DL                              ; NO -- JUST MOVE IT BACK
1D70  EB F4          C          JMP     U7                              ; SET_CURSOR
                     C
                     C  ;----- CARRIAGE RETURN FOUND
                     C
1D72                 C  U9:
1D72  2A D2          C          SUB     DL,DL                           ; MOVE TO FIRST COLUMN
1D74  EB F0          C          JMP     U7                              ; SET_CURSOR
                     C
                     C  ;----- LINE FEED FOUND
                     C
1D76                 C  U10:
1D76  3A 36 0484 R   C          CMP     DH,ROWS                         ; BOTTOM OF SCREEN
1D7A  75 E8          C          JNE     U6                              ; YES, SCROLL THE SCREEN
1D7C  EB BB          C          JMP     U1                              ; NO, JUST SET THE CURSOR
                     C
                     C  ;----- BELL FOUND
                     C
1D7E                 C  U11:
1D7E  B3 02          C          MOV     BL,2                            ; SET UP COUNT FOR BEEP
1D80  E8 0D20 R      C          CALL    BEEP                            ; SOUND THE POD BELL
1D83  EB DB          C          JMP     U5                              ; TTY_RETURN
                     C
                     C
                     C  ;----- CURRENT VIDEO STATE
                     C
1D85                 C  AHF:
                     C          ASSUME  DS:ABS0
1D85  8A 26 044A R   C          MOV     AH,BYTE PTR CRT_COLS            ; GET NUMBER OF COLUMNS
1D89  8A 3E 0462 R   C          MOV     BH,ACTIVE_PAGE
1D8D  A0 0487 R      C          MOV     AL,INFO
1D90  24 80          C          AND     AL,080H
1D92  0A 06 0449 R   C          OR      AL,CRT_MODE
```

```
1D96  5F                 C          POP     DI
1D97  5E                 C          POP     SI
1D98  59                 C          POP     CX                              ; DISCARD BX
1D99  59                 C          POP     CX
1D9A  5A                 C          POP     DX
1D9B  1F                 C          POP     DS
1D9C  07                 C          POP     ES
1D9D  5D                 C          POP     BP
1D9E  CF                 C          IRET
                         C
                         C          SUBTTL


1D9F                          PAL_SET PROC    NEAR
1D9F  50                              PUSH    AX
1DA0  E8 0D05 R                       CALL    WHAT_BASE
1DA3  FA                              CLI
1DA4                          VR:
1DA4  EC                              IN      AL,DX
1DA5  A8 08                           TEST    AL,08H                          ; VERTICAL RETRACE
1DA7  74 FB                           JZ      VR
1DA9  58                              POP     AX
1DAA  B2 C0                           MOV     DL,ATTR_WRITE
1DAC  86 C4                           XCHG    AL,AH
1DAE  EE                              OUT     DX,AL
1DAF  86 C4                           XCHG    AL,AH
1DB1  EE                              OUT     DX,AL
1DB2  B0 20                           MOV     AL,020H
1DB4  EE                              OUT     DX,AL
1DB5  FB                              STI
1DB6  C3                              RET
1DB7                          PAL_SET ENDP

1DB7                          PAL_ON  PROC    NEAR
1DB7  E8 1DC0 R                       CALL    PAL_INIT
1DBA  B2 C0                           MOV     DL,ATTR_WRITE
1DBC  B0 20                           MOV     AL,020H
1DBE  EE                              OUT     DX,AL
1DBF  C3                              RET
1DC0                          PAL_ON  ENDP

1DC0                          PAL_INIT        PROC    NEAR
1DC0  E8 0D05 R                       CALL    WHAT_BASE
1DC3  EC                              IN      AL,DX
1DC4  C3                              RET
1DC5                          PAL_INIT        ENDP

                              ;----- SET PALETTE REGISTERS

1DC5                          AH10:
                                      ASSUME  DS:ABS0
1DC5  F6 06 0487 R 02                 TEST    INFO,2
1DCA  75 07                           JNZ     BM_OK                           ; IN MONOCHROME MODE

                              ;----- HERE THE EGA IS IN A COLOR MODE

1DCC  80 3E 0463 R B4                 CMP     BYTE PTR ADDR_6845,0B4H
1DD1  74 33                           JE      BM_OUT
1DD3                          BM_OK:
1DD3  8A E0                           MOV     AH,AL
1DD5  0A E4                           OR      AH,AH
1DD7  75 30                           JNZ     BM_1

                              ;----- SET INDIVIDUAL REGISTER

1DD9  2B ED                           SUB     BP,BP
1DDB  C4 3E 04A8 R                    LES     DI,SAVE_PTR
1DDF  83 C7 04                        ADD     DI,4
1DE2  26: C4 3D                       LES     DI,DWORD PTR ES:[DI]
1DE5  8C C0                           MOV     AX,ES
1DE7  0B C7                           OR      AX,DI
1DE9  74 01                           JZ      TLO_1
1DEB  45                              INC     BP
1DEC                          TLO_1:

1DEC  E8 1DC0 R                       CALL    PAL_INIT
1DEF  8A E3                           MOV     AH,BL
1DF1  8A C7                           MOV     AL,BH
1DF3  E8 1D9F R                       CALL    PAL_SET
1DF6  E8 1DB7 R                       CALL    PAL_ON
1DF9  0B ED                           OR      BP,BP
1DFB  74 09                           JZ      BM_OUT
1DFD  8A C7                           MOV     AL,BH
1DFF  2A FF                           SUB     BH,BH
1E01  03 FB                           ADD     DI,BX
1E03  26: 88 05                       MOV     ES:[DI],AL
1E06                          BM_OUT:
1E06  E9 219E R                       JMP     V_RET

1E09                          BM_1:
1E09  FE CC                           DEC     AH
1E0B  75 2D                           JNZ     BM_2

1E0D  2B ED                           SUB     BP,BP
1E0F  C4 3E 04A8 R                    LES     DI,SAVE_PTR
1E13  83 C7 04                        ADD     DI,4
1E16  26: C4 3D                       LES     DI,DWORD PTR ES:[DI]
1E19  8C C0                           MOV     AX,ES
1E1B  0B C7                           OR      AX,DI
1E1D  74 01                           JZ      TLO_2
1E1F  45                              INC     BP
1E20                          TLO_2:

                              ;----- SET OVERSCAN REGISTER

1E20  E8 1DC0 R                       CALL    PAL_INIT
1E23  B4 11                           MOV     AH,011H
1E25  8A C7                           MOV     AL,BH
1E27  E8 1D9F R                       CALL    PAL_SET
1E2A  E8 1DB7 R                       CALL    PAL_ON

1E2D  0B ED                           OR      BP,BP
1E2F  74 D5                           JZ      BM_OUT
1E31  83 C7 11                        ADD     DI,011H
1E34  26: 88 3D                       MOV     ES:[DI],BH

1E37  E9 219E R                       JMP     V_RET

1E3A                          BM_2:
1E3A  FE CC                           DEC     AH
1E3C  75 40                           JNZ     BM_3

                              ;----- SET 16 PALETTE REGISTERS AND OVERSCAN REGISTER

1E3E  1E                              PUSH    DS
1E3F  06                              PUSH    ES
```

```
1E40  C4 3E 04A8 R                          LES     DI,SAVE_PTR
1E44  83 C7 04                              ADD     DI,4
1E47  26: C4 3D                             LES     DI,DWORD PTR ES:[DI]        ; ES:DI PTR TO PAL SAVE AREA
1E4A  8C C0                                 MOV     AX,ES
1E4C  0B C7                                 OR      AX,DI
1E4E  74 09                                 JZ      TLO_3

1E50  1F                                    POP     DS                          ; PARAMETER ES
1E51  1E                                    PUSH    DS
1E52  8B F2                                 MOV     SI,DX                       ; PARAMETER OFFSET
1E54  B9 0011                               MOV     CX,17D
1E57  F3/ A4                        REP     MOVSB

1E59                                TLO_3:
1E59  07                                    POP     ES
1E5A  1F                                    POP     DS

1E5B  8B DA                                 MOV     BX,DX
1E5D  E8 1DC0 R                             CALL    PAL_INIT
1E60  2A E4                                 SUB     AH,AH
1E62                                BM_2A:
1E62  26: 8A 07                             MOV     AL,ES:[BX]
1E65  E8 1D9F R                             CALL    PAL_SET
1E68  FE C4                                 INC     AH
1E6A  43                                    INC     BX
1E6B  80 FC 10                              CMP     AH,010H
1E6E  72 F2                                 JB      BM_2A
1E70  FE C4                                 INC     AH
1E72  26: 8A 07                             MOV     AL,ES:[BX]
1E75  E8 1D9F R                             CALL    PAL_SET
1E78  E8 1DB7 R                             CALL    PAL_ON
1E7B  E9 219E R                             JMP     V_RET

1E7E                                BM_3:
1E7E  FE CC                                 DEC     AH
1E80  75 29                                 JNZ     BM_4

                                    ;----- TOGGLE INTENSIFY/BLINKING BIT

1E82  53                                    PUSH    BX
1E83  E8 0D5A R                             CALL    MAKE_BASE
1E86  83 C3 33                              ADD     BX,010H + LN_4
1E89  26: 8A 07                             MOV     AL,ES:[BX]
1E8C  5B                                    POP     BX

1E8D  0A DB                                 OR      BL,BL
1E8F  75 0A                                 JNZ     BM_6

                                    ;----- ENABLE INTENSIFY

1E91  80 26 0465 R DF                       AND     CRT_MODE_SET,11011111B
1E96  24 F7                                 AND     AL,0F7H
1E98  EB 0C 90                              JMP     BM_7
1E9B                                BM_6:
1E9B  FE CB                                 DEC     BL
1E9D  75 07                                 JNZ     BM_7

                                    ;----- ENABLE BLINK

1E9F  80 0E 0465 R 20                       OR      CRT_MODE_SET,020H
1EA4  0C 08                                 OR      AL,08H
1EA6                                BM_7:
1EA6  B4 10                                 MOV     AH,P_MODE
1EA8  E8 1D9F R                             CALL    PAL_SET
1EAB                                BM_4:
1EAB  E9 219E R                             JMP     V_RET

                                 C          INCLUDE         VCHGEN.INC
                                 C          SUBTTL          VCHGEN.INC
                                 C          PAGE
                                 C  ;--------------------------------------------------
                                 C  ; ENTRY                                           :
                                 C  ;       AL = 0  USER SPECIFIED FONT               :
                                 C  ;            1  8 X 14 FONT                       :
                                 C  ;            2  8 X 8 DOUBLE DOT                  :
                                 C  ;       BL = BLOCK TO LOAD                        :
                                 C  ;--------------------------------------------------
1EAE                             C  CH_GEN:
1EAE  50                         C          PUSH    AX                          ; SAVE THE INVOLVED REGS
1EAF  55                         C          PUSH    BP
1EB0  53                         C          PUSH    BX
1EB1  51                         C          PUSH    CX
1EB2  52                         C          PUSH    DX
1EB3  06                         C          PUSH    ES
                                 C
                                 C          ASSUME  DS:ABS0
1EB4  E8 0CFE R                  C          CALL    DDS                         ; SET DATA SEGMENT
1EB7  A0 0449 R                  C          MOV     AL,CRT_MODE                 ; GET THE CURRENT MODE
1EBA  50                         C          PUSH    AX                          ; SAVE IT
1EBB  3C 07                      C          CMP     AL,7                        ; IS THIS MONOCHROME
1EBD  74 07                      C          JE      H14                         ; MONOCHROME VALUES
1EBF  C6 06 0449 R 0B            C          MOV     CRT_MODE,0BH                ; COLOR VALUES
1EC4  EB 05                      C          JMP     SHORT H15                   ; SKIP
1EC6                             C  H14:
1EC6  C6 06 0449 R 0C            C          MOV     CRT_MODE,0CH                ; MONOCHROME VALUES
1ECB                             C  H15:
1ECB  E8 0DAB R                  C          CALL    SET_REGS
1ECE  E8 0CFE R                  C          CALL    DDS                         ; RESET THE DATA SEGMENT
1ED1  58                         C          POP     AX                          ; RECOVER OLD MODE VALUE
1ED2  A2 0449 R                  C          MOV     CRT_MODE,AL                 ; RETURN TO LOW MEMORY
                                 C
1ED5  07                         C          POP     ES                          ; RESTORE REGS THAT WERE
1ED6  5A                         C          POP     DX                          ;  USED BY THE MODE SET
1ED7  59                         C          POP     CX                          ;  ROUTINES
1ED8  5B                         C          POP     BX
1ED9  5D                         C          POP     BP
1EDA  58                         C          POP     AX
                                 C
1EDB  0A C0                      C          OR      AL,AL                       ; SET FLAGS
1EDD  74 17                      C          JZ      DO_MAP2                     ; USER SPECIFIED FONT
1EDF  0E                         C          PUSH    CS                          ; SET SEGMENT TO
1EE0  07                         C          POP     ES                          ;  THIS MODULE
1EE1  2B D2                      C          SUB     DX,DX                       ; ZERO OUT START OFFSET
1EE3  B9 0100                    C          MOV     CX,0256D                    ; CHAR COUNT (FULL SET)
1EE6  FE C8                      C          DEC     AL                          ; WHICH PARAMETER
1EE8  75 07                      C          JNZ     H7                          ; MUST BE ONE
1EEA  B7 0E                      C          MOV     BH,014D                     ; BYTES PER CHARACTER
1EEC  BD 0000 E                  C          MOV     BP,OFFSET CGMN              ; 8 X 14 TABLE OFFSET
1EEF  EB 05                      C          JMP     SHORT DO_MAP2               ; STORE IT
1EF1                             C  H7:
1EF1  B7 08                      C          MOV     BH,8                        ; 8 X 8 FONT
1EF3  BD 0000 E                  C          MOV     BP,OFFSET CGDDOT            ; ROM 8 X 8 DOUBLE DOT
                                 C
                                 C  ;--------------------------------------------------
                                 C  ; ALPHA CHARACTER GENERATOR LOAD                  :
```

```
                         C  ;                                               :
                         C  ; ENTRY                                         :
                         C  ;         ES:BP - POINTER TO TABLE              :
                         C  ;         CX    - COUNT OF CHARS                :
                         C  ;         DX    - CHAR COUNT OFFSET INTO MAP 2  :
                         C  ;         BH    - BYTES PER CHARACTER           :
                         C  ;         BL    - MAP 2 BLOCK TO LOAD           :
                         C  ;-----------------------------------------------
1EF6                     C  DO_MAP2:
1EF6  06                 C          PUSH    ES                  ; FONT TABLE SEGMENT
1EF7  1F                 C          POP     DS                  ; ADDRESSING TO TABLE
1EF8  52                 C          PUSH    DX                  ; SAVE REGISTER
                         C          SRLOAD  ES,0A000H           ; ADDRESSING TO MAP 2
1EF9  BA A000            C+         MOV     DX,0A000H
1EFC  8E C2              C+         MOV     ES,DX
1EFE  5A                 C          POP     DX                  ; RECOVER REGISTER
1EFF  51                 C          PUSH    CX                  ; MULTIPLY BY 020H SINCE
1F00  B1 05              C          MOV     CL,5                ;  MAXIMUM BYTES PER
1F02  D3 E2              C          SHL     DX,CL               ;  CHARACTER IS 32D=020H
1F04  59                 C          POP     CX                  ; RECOVER
1F05  0A DB              C          OR      BL,BL               ; WHICH 16K BLOCK TO LOAD
1F07  74 08              C          JZ      H3                  ; BLOCK ZERO
1F09                     C  H4:
1F09  81 C2 4000         C          ADD     DX,04000H           ; INCREMENT TO NEXT BLOCK
1F0D  FE CB              C          DEC     BL                  ; ANY MORE
1F0F  75 F8              C          JNZ     H4                  ; DO ANOTHER
1F11                     C  H3:
1F11  8A C7              C          MOV     AL,BH               ; BYTES PER CHARACTER
1F13  2A E4              C          SUB     AH,AH               ; ZERO
1F15  8B FA              C          MOV     DI,DX               ; OFFSET INTO MAP
1F17  8B F5              C          MOV     SI,BP               ; OFFSET INTO TABLE
1F19  E3 0D              C          JCXZ    LD_OVER             ; CHARACTER COUNT
1F1B                     C  LD:
1F1B  51                 C          PUSH    CX                  ; SAVE CHARACTER COUNT
1F1C  8B C8              C          MOV     CX,AX               ; ONE ENTIRE CHARACTER
1F1E  F3/ A4             C          REP     MOVSB               ;  AT A TIME
1F20  2B F8              C          SUB     DI,AX               ; ADJUST OFFSET
1F22  83 C7 20           C          ADD     DI,020H             ; NEXT CHARACTER POSITION
1F25  59                 C          POP     CX                  ; RECOVER CHARACTER COUNT
1F26  E2 F3              C          LOOP    LD                  ; DO THE REST
1F28                     C  LD_OVER:
1F28  C3                 C          RET
                         C
1F29                     C  BRK_1:
                         C          ASSUME  DS:ABS0
1F29  E8 0CFE R          C          CALL    DDS                 ; SET LOW MEMORY SEGMENT
1F2C  A3 0485 R          C          MOV     POINTS,AX           ; GET BYTES/CHARACTER
1F2F  8B 16 0463 R       C          MOV     DX,ADDR_6845        ; CRTC REGISTER
1F33  80 3E 0449 R 07    C          CMP     CRT_MODE,7
1F38  75 05              C          JNE     H11A
1F3A  B4 14              C          MOV     AH,C_UNDERLN_LOC    ; R14H
1F3C  E8 0D15 R          C          CALL    OUT_DX              ; SET THE UNDERLINE LOC
1F3F                     C  H11A:
1F3F  FE C8              C          DEC     AL                  ; POINTS - 1
1F41  B4 09              C          MOV     AH,C_MAX_SCAN_LN    ; R09H
1F43  E8 0D15 R          C          CALL    OUT_DX              ; SET THE CHARACTER HEIGHT
1F46  FE C8              C          DEC     AL                  ; POINTS - 2
                         C
1F48  8A E8              C          MOV     CH,AL               ; CURSOR START
1F4A  8A C8              C          MOV     CL,AL               ; CURSOR END
1F4C  FE C1              C          INC     CL                  ; ADJUST END
1F4E  B4 01              C          MOV     AH,1                ; SET C_TYPE BIOS CALL
1F50  CD 10              C          INT     10H                 ; SET THE CURSOR
                         C
1F52  8A 1E 0449 R       C          MOV     BL,CRT_MODE         ; GET THE CURRENT MODE
1F56  B8 015E            C          MOV     AX,350D             ; MAX SCANS ON SCREEN
1F59  80 FB 03           C          CMP     BL,3                ; 640X200 ALPHA MODES
1F5C  77 08              C          JA      H11                 ; MUST BE 350
1F5E  E8 0E9A R          C          CALL    BRST_DET
1F61  72 03              C          JC      H11
1F63  B8 00C8            C          MOV     AX,200D             ; SET FOR 200
1F66                     C  H11:
1F66  99                 C          CWD                         ; PREPARE TO DIVIDE
1F67  F7 36 0485 R       C          DIV     POINTS              ; MAX ROWS ON SCREEN
1F6B  48                 C          DEC     AX                  ; ADJUST
1F6C  A2 0484 R          C          MOV     ROWS,AL             ; SAVE ROWS
1F6F  FE C0              C          INC     AL                  ; READJUST
1F71  2A E4              C          SUB     AH,AH               ; CLEAR
1F73  F7 26 0485 R       C          MUL     POINTS              ; ROWS*BYTES/CHAR
1F77  48                 C          DEC     AX                  ; ADJUST
1F78  8B 16 0463 R       C          MOV     DX,ADDR_6845        ; CRTC ADDRESS
1F7C  B4 12              C          MOV     AH,C_VRT_DSP_END    ; SCANS DISPLAYED
1F7E  E8 0D15 R          C          CALL    OUT_DX              ; SET IT
1F81  A0 0484 R          C          MOV     AL,ROWS             ; GET CHARACTER ROWS
1F84  FE C0              C          INC     AL                  ; ADJUST
1F86  F6 26 044A R       C          MUL     BYTE PTR CRT_COLS   ; ROWS*COLUMNS
1F8A  D1 E0              C          SHL     AX,1                ; *2 FOR ALPHA MODE
1F8C  05 0100            C          ADD     AX,256D             ; SPACE BETWEEN PAGES
1F8F  A3 044C R          C          MOV     CRT_LEN,AX          ; BYTES PER PAGE
1F92  E8 0E96 R          C          CALL    PH_5                ; VIDEO ON
1F95  E9 219E R          C          JMP     V_RET               ; RETURN TO CALLER
                         C
                         C  ;----- LOADABLE CHARACTER GENERATOR ROUTINES
                         C
1F98                     C  AH11:
1F98  3C 10              C          CMP     AL,010H             ; CHECK PARAMETER
1F9A  73 37              C          JAE     AH11_ALPHA1         ; NEXT STAGE
                         C
                         C  ;----- ALPHA MODE ACTIVITY HERE
                         C
1F9C  3C 03              C          CMP     AL,03H              ; RANGE CHECK
1F9E  73 17              C          JAE     H1                  ; NEXT STAGE
1FA0  E8 1EAE R          C          CALL    CH_GEN              ; SET THE CHAR GEN
1FA3  E8 0DAB R          C          CALL    SET_REGS
1FA6  E8 0E96 R          C          CALL    PH_5                ; VIDEO ON
                         C          ASSUME  DS:ABS0
1FA9  E8 0CFE R          C          CALL    DDS                 ; SET THE DATA SEGMENT
1FAC  8B 0E 0460 R       C          MOV     CX,CURSOR_MODE      ; GET THE MODE
1FB0  B4 01              C          MOV     AH,1                ; SET C_TYPE
1FB2  CD 10              C          INT     10H                 ; EMULATE CORRECT CURSOR
1FB4  E9 219E R          C          JMP     V_RET               ; RETURN TO CALLER
                         C
                         C  ;----- SET THE CHARACTER GENERATOR BLOCK SELECT REGISTER
                         C
1FB7                     C  H1:
1FB7  75 17              C          JNE     H2                  ; NOT IN RANGE
1FB9  B6 03              C          MOV     DH,3
1FBB  B2 C4              C          MOV     DL,SEQ_ADDR         ; SEQUENCER
                         C
1FBD  B8 0001            C          MOV     AX,1                ; AH=S_RESET, AL=1
1FC0  E8 0D15 R          C          CALL    OUT_DX
                         C
1FC3  B4 03              C          MOV     AH,S_CGEN           ; CHAR BLOCK REGISTER
1FC5  8A C3              C          MOV     AL,BL               ; GET THE VALUE
1FC7  E8 0D15 R          C          CALL    OUT_DX              ; SET IT
                         C
```

```
1FCA  B8 0003         C            MOV     AX,3                          ; AH=S_RESET, AL=3
1FCD  E8 0D15 R       C            CALL    OUT_DX
1FD0                  C  H2:
1FD0  E9 219E R       C            JMP     V_RET                         ; RETURN TO CALLER
                      C
1FD3                  C  AH11_ALPHA1:
                      C            ASSUME  DS:ABS0
1FD3  3C 20           C            CMP     AL,020H
1FD5  73 26           C            JAE     AH11_GRAPHICS
                      C
                      C  ;----- ALPHA MODE ACTIVITY HERE
                      C
1FD7  2C 10           C            SUB     AL,010H                       ; ADJUST TO 0 - N
1FD9  3C 02           C            CMP     AL,02H                        ; RANGE CHECK
1FDB  77 F3           C            JA      H2                            ; INVALID CALL
1FDD  50              C            PUSH    AX                            ; SAVE
1FDE  53              C            PUSH    BX
1FDF  E8 1EAE R       C            CALL    CH_GEN                        ; LOAD THE CHAR GEN
1FE2  E8 0DAB R       C            CALL    SET_REGS
1FE5  5B              C            POP     BX
1FE6  58              C            POP     AX                            ; RESTORE
1FE7  8A E0           C            MOV     AH,AL                         ; CALLING PARAMETER
1FE9  0A E4           C            OR      AH,AH                         ; USER MODE
1FEB  8A C7           C            MOV     AL,BH
1FED  74 09           C            JZ      H13                           ; DO NOT SET BYTES/CHAR
1FEF  B0 08           C            MOV     AL,8                          ; 8 X 8 FONT
1FF1  80 FC 01        C            CMP     AH,1                          ; IS THE CALL FOR MONOC
1FF4  75 02           C            JNE     H13                           ; NO, LEAVE IT AT 8
1FF6  B0 0E           C            MOV     AL,14D                        ; MONOC SET
1FF8                  C  H13:
1FF8  2A E4           C            SUB     AH,AH                         ; CLEAR UPPER BYTE
1FFA  E9 1F29 R       C            JMP     BRK_1                         ; CONTINUE
                      C
                      C  ;----- GRAPHICS MODE ACTIVITY HERE
                      C
1FFD                  C  AH11_GRAPHICS:
                      C            ASSUME  DS:ABS0
1FFD  3C 30           C            CMP     AL,030H
1FFF  73 6A           C            JAE     AH11_INFORM
2001  2C 20           C            SUB     AL,020H
2003  75 11           C            JNZ     F10
                      C
                      C  ;----- COMPATIBILITY, UPPER HALF GRAPHICS CHARACTER SET
                      C
                      C            ASSUME  DS:ABS0
                      C            SRLOAD  DS,0
2005  2B D2           C+           SUB     DX,DX
2007  8E DA           C+           MOV     DS,DX
2009  FA              C            CLI
200A  89 2E 007C R    C            MOV     WORD PTR EXT_PTR , BP
200E  8C 06 007E R    C            MOV     WORD PTR EXT_PTR + 2 , ES
2012  FB              C            STI
2013                  C  F11:
2013  E9 219E R       C            JMP     V_RET
2016                  C  F10:
                      C            ASSUME  DS:ABS0
2016  52              C            PUSH    DX
                      C            SRLOAD  DS,0
2017  2B D2           C+           SUB     DX,DX
2019  8E DA           C+           MOV     DS,DX
201B  5A              C            POP     DX
201C  3C 03           C            CMP     AL,03H                        ; RANGE CHECK
201E  77 F3           C            JA      F11
2020  FE C8           C            DEC     AL
2022  74 14           C            JZ      F19
2024  0E              C            PUSH    CS
2025  07              C            POP     ES
2026  FE C8           C            DEC     AL
2028  75 08           C            JNZ     F13
202A  B9 000E         C            MOV     CX,14D
202D  BD 0000 E       C            MOV     BP,OFFSET CGMN                ; ROM 8 X 14 CHARACTER SET
2030  EB 06           C            JMP     SHORT F19
2032                  C  F13:
2032  B9 0008         C            MOV     CX,8
2035  BD 0000 E       C            MOV     BP,OFFSET CGDDOT              ; ROM 8 X 8 DOUBLE DOT
2038                  C  F19:
2038  FA              C            CLI
2039  89 2E 010C R    C            MOV     WORD PTR GRX_SET , BP
203D  8C 06 010E R    C            MOV     WORD PTR GRX_SET + 2 , ES
2041  FB              C            STI
                      C            ASSUME  DS:ABS0
2042  E8 0CFE R       C            CALL    DDS
2045  89 0E 0485 R    C            MOV     POINTS,CX
2049  8A C3           C            MOV     AL,BL
204B  BB 2067 R       C            MOV     BX,OFFSET RT
204E  0A C0           C            OR      AL,AL
2050  75 05           C            JNZ     DR_3
2052  8A C2           C            MOV     AL,DL
2054  EB 09 90        C            JMP     DR_1
2057                  C  DR_3:
2057  3C 03           C            CMP     AL,3
2059  76 02           C            JBE     DR_2
205B  B0 02           C            MOV     AL,2
205D                  C  DR_2:
205D  2E: D7          C            XLAT    CS:RT
205F                  C  DR_1:
205F  FE C8           C            DEC     AL
2061  A2 0484 R       C            MOV     ROWS,AL
2064  E9 219E R       C            JMP     V_RET
                      C
2067                  C  RT        LABEL   BYTE
2067  00 0E 19 2B     C            DB      00D,14D,25D,43D
                      C
                      C
                      C  ;----- INFORMATION RETURN DONE HERE
                      C
206B                  C  AH11_INFORM:
                      C            ASSUME  DS:ABS0
206B  3C 30           C            CMP     AL,030H
206D  74 03           C            JE      F6
206F                  C  F5:
206F  E9 219E R       C            JMP     V_RET
2072                  C  F6:
2072  8B 0E 0485 R    C            MOV     CX,POINTS
2076  8A 16 0484 R    C            MOV     DL,ROWS
207A  80 FF 07        C            CMP     BH,7
207D  77 F0           C            JA      F5
207F  80 FF 01        C            CMP     BH,1
2082  77 18           C            JA      F7
                      C
                      C            ASSUME  DS:ABS0
2084  52              C            PUSH    DX
                      C            SRLOAD  DS,0
2085  2B D2           C+           SUB     DX,DX
2087  8E DA           C+           MOV     DS,DX
2089  5A              C            POP     DX
```

```
208A  0A FF                C        OR      BH,BH
208C  75 07                C        JNZ     F9
208E  C4 2E 007C R         C        LES     BP,EXT_PTR
2092  EB 1A 90             C        JMP     INFORM_OUT
2095                       C  F9:
2095  C4 2E 010C R         C        LES     BP,GRX_SET
2099  EB 13 90             C        JMP     INFORM_OUT
                           C
                           C  ;----- HANDLE BH = 2  THRU  BH = 5 HERE  RETURN ROM TABLE POINTERS
                           C
209C                       C  F7:
                           C        ASSUME  DS:ABSO
209C  80 EF 02             C        SUB     BH,2
209F  8A DF                C        MOV     BL,BH
20A1  2A FF                C        SUB     BH,BH
20A3  D1 E3                C        SAL     BX,1
20A5  81 C3 20B7 R         C        ADD     BX,OFFSET TBL_5
20A9  2E: 8B 2F            C        MOV     BP,CS:[BX]
20AC  0E                   C        PUSH    CS
20AD  07                   C        POP     ES
                           C
20AE                       C  INFORM_OUT:
20AE  5F                   C        POP     DI
20AF  5E                   C        POP     SI
20B0  5B                   C        POP     BX
20B1  58                   C        POP     AX                               ; DISCARD SAVED CX
20B2  58                   C        POP     AX                               ; DISCARD SAVED DX
20B3  1F                   C        POP     DS
20B4  58                   C        POP     AX                               ; DISCARD SAVED ES
20B5  58                   C        POP     AX                               ; DISCARD SAVED BP
20B6  CF                   C        IRET
                           C
                           C  ;----- TABLE OF CHARACTER GENERATOR OFFSETS
                           C
20B7                       C  TBL_5 LABEL   WORD
20B7  0000 E               C        DW      OFFSET  CGMN
20B9  0000 E               C        DW      OFFSET  CGDDOT
20BB  0000 E               C        DW      OFFSET  INT_1F_1
20BD  0000 E               C        DW      OFFSET  CGMN_FDG
                           C
                           C        SUBTTL

                              ;----- ALTERNATE SELECT

20BF                          AH12:
                                    ASSUME  DS:ABSO
20BF  80 FB 10                      CMP     BL,010H                          ; RETURN ACTIVE CALL
20C2  72 51                         JB      ACT_1
20C4  74 1B                         JE      ACT_3
20C6  80 FB 20                      CMP     BL,020H                          ; ALTERNATE PRINT SCREEN
20C9  74 03                         JE      ACT_2
20CB  E9 219E R                     JMP     V_RET                            ; INVALID CALL
20CE                          ACT_2:                                         ; NEW PRINT SCREEN
                                    SRLOAD  DS,0
20CE  2B D2                +        SUB     DX,DX
20D0  8E DA                +        MOV     DS,DX
20D2  FA                            CLI
20D3  C7 06 0014 R 21A7 R           MOV     WORD PTR INT5_PTR, OFFSET PRINT_SCREEN
20D9  8C 0E 0016 R                  MOV     WORD PTR INT5_PTR+2, CS
20DD  FB                            STI
20DE  E9 219E R                     JMP     V_RET
20E1                          ACT_3:
20E1  8A 3E 0487 R                  MOV     BH,INFO                          ; LOOKING FOR MONOC BIT
20E5  80 E7 02                      AND     BH,2                             ; ISOLATE
20E8  D0 EF                         SHR     BH,1                             ; ADJUST

20EA  A0 0487 R                     MOV     AL,INFO                          ; LOOKING FOR MEMORY
20ED  24 60                         AND     AL,01100000B                     ; MEMORY BITS
20EF  B1 05                         MOV     CL,5                             ; SHIFT COUNT
20F1  D2 E8                         SHR     AL,CL                            ; ADJUST MEM VALUE
20F3  8A D8                         MOV     BL,AL                            ; RETURN REGISTER

20F5  8A 0E 0488 R                  MOV     CL,INFO_3                        ; FEATURE/SWITCH
20F9  8A E9                         MOV     CH,CL                            ; DUPLICATE IN CH
20FB  80 E1 0F                      AND     CL,0FH                           ; MASK OFF SWITCH VALUE
20FE  D0 ED                         SHR     CH,1                             ; MOVE FEATURE VALUE
2100  D0 ED                         SHR     CH,1
2102  D0 ED                         SHR     CH,1
2104  D0 ED                         SHR     CH,1
2106  80 E5 0F                      AND     CH,0FH                           ; MASK IT

2109  5F                            POP     DI
210A  5E                            POP     SI
210B  5A                            POP     DX                               ; DISCARD BX
210C  5A                            POP     DX                               ; DISCARD CX
210D  5A                            POP     DX
210E  1F                            POP     DS
210F  07                            POP     ES
2110  5D                            POP     BP
2111  CF                            IRET
2112                          AH12_X:
2112  E9 219E R                     JMP     V_RET                            ; RETURN TO CALLER
2115                          ACT_1:
2115                          STR_OUTZ:
2115  E9 219E R                     JMP     V_RET                            ; RETURN TO CALLER

                              ;----- WRITE STRING

2118                          AH13:
2118  3C 04                         CMP     AL,04                            ; RANGE CHECK
211A  73 F9                         JAE     STR_OUTZ                         ; INVALID PARAMETER
211C  E3 F7                         JCXZ    STR_OUTZ
211E  53                            PUSH    BX                               ; SAVE REGISTER
211F  8A DF                         MOV     BL,BH                            ; GET PAGE TO LOW BYTE
2121  2A FF                         SUB     BH,BH
2123  D1 E3                         SAL     BX,1                             ; *2 FOR WORD OFFSET
2125  8B B7 0450 R                  MOV     SI,[BX + OFFSET CURSOR_POSN]     ; GET CURSOR POSITION
2129  5B                            POP     BX                               ; RESTORE
212A  56                            PUSH    SI                               ; CURRENT VALUE ON STACK

212B  50                            PUSH    AX
212C  B8 0200                       MOV     AX,0200H                         ; SET THE CURSOR POSITION
212F  CD 10                         INT     10H
2131  58                            POP     AX
2132                          STR_1:
2132  51                            PUSH    CX
2133  53                            PUSH    BX
2134  50                            PUSH    AX
2135  86 E0                         XCHG    AH,AL
2137  26: 8A 46 00                  MOV     AL,ES:[BP]                       ; GET THE CHAR TO WRITE
213B  45                            INC     BP
213C  3C 0D                         CMP     AL,0DH                           ; CARRIAGE RETURN
213E  74 3D                         JE      STR_CR_LF
2140  3C 0A                         CMP     AL,0AH                           ; LINE FEED
2142  74 39                         JE      STR_CR_LF
2144  3C 08                         CMP     AL,08H                           ; BACKSPACE
```

```
2146  74 35                    JE      STR_CR_LF
2148  3C 07                    CMP     AL,07H                         ; BELL
214A  74 31                    JE      STR_CR_LF
214C  B9 0001                  MOV     CX,1                           ; COUNT OF CHARACTERS
214F  80 FC 02                 CMP     AH,2                           ; CHECK WHERE ATTR IS
2152  72 05                    JB      DO_STR                         ; NOT IN THE STRING
2154  26: 8A 5E 00             MOV     BL,ES:[BP]                     ; GET THE ATTRIBUTE
2158  45                       INC     BP                             ; NEXT ITEM IN STRING
2159                     DO_STR:
2159  B4 09                    MOV     AH,09H                         ; WRITE THE CHAR/ATTR
215B  CD 10                    INT     10H
215D  FE C2                    INC     DL                             ; NEXT CURSOR POSITION
215F  3A 16 044A R             CMP     DL,BYTE PTR CRT_COLS           ; COLUMN OVERFLOW
2163  72 11                    JB      STR_2                          ; NOT YET
2165  3A 36 0484 R             CMP     DH,ROWS
2169  75 07                    JNE     STR_3
216B  B8 0E0A                  MOV     AX,0E0AH
216E  CD 10                    INT     10H
2170  FE CE                    DEC     DH
2172                     STR_3:
2172  FE C6                    INC     DH                             ; NEXT ROW
2174  2A D2                    SUB     DL,DL                          ; COLUMN ZERO
2176                     STR_2:
2176  B8 0200                  MOV     AX,0200H                       ; SET THE CURSOR
2179  CD 10                    INT     10H
217B  EB 0E                    JMP     SHORT   STR_4
217D                     STR_CR_LF:
217D  B4 0E                    MOV     AH,0EH
217F  CD 10                    INT     10H
2181  8A DF                    MOV     BL,BH                          ; GET PAGE TO LOW BYTE
2183  2A FF                    SUB     BH,BH
2185  D1 E3                    SAL     BX,1                           ; *2 FOR WORD OFFSET
2187  8B 97 0450 R             MOV     DX,[BX + OFFSET CURSOR_POSN]   ; GET CURSOR POSITION
218B                     STR_4:
218B  58                       POP     AX
218C  5B                       POP     BX
218D  59                       POP     CX
218E  E2 A2                    LOOP    STR_1

2190  5A                       POP     DX                             ; RECOVER CURSOR POSITION
                                                                      ;  FROM PUSH SI ABOVE
2191  3C 01                    CMP     AL,1
2193  74 09                    JE      STR_OUT
2195  3C 03                    CMP     AL,3
2197  74 05                    JE      STR_OUT
2199  B8 0200                  MOV     AX,0200H                       ; SET CURSOR POSITION
219C  CD 10                    INT     10H
219E                     STR_OUT:
                                                                      ; ALLOW FALL THROUGH


219E                     V_RET  PROC   NEAR                           ; VIDEO BIOS RETURN
219E  5F                       POP     DI
219F  5E                       POP     SI
21A0  5B                       POP     BX
21A1  59                       POP     CX
21A2  5A                       POP     DX
21A3  1F                       POP     DS
21A4  07                       POP     ES
21A5  5D                       POP     BP
21A6  CF                       IRET
21A7                     V_RET  ENDP

21A7                     COMBO_VIDEO     ENDP

                      C          INCLUDE         VPRSC.INC
                      C          SUBTTL  VPRSC.INC
                      C          PAGE
                      C  ;--------------------------------------------------------------------
                      C  ;   INTERRUPT 5                                                      :
                      C  ;       THIS LOGIC WILL BE INVOKED BY INTERRUPT 05H TO PRINT THE     :
                      C  ;       SCREEN. THE CURSOR POSITION AT THE TIME THIS ROUTINE IS INVOKED :
                      C  ;       WILL BE SAVED AND RESTORED UPON COMPLETION. THE ROUTINE IS   :
                      C  ;       INTENDED TO RUN WITH INTERRUPTS ENABLED. IF A SUBSEQUENT     :
                      C  ;       'PRINT SCREEN' KEY IS DEPRESSED DURING THE TIME THIS ROUTINE :
                      C  ;       IS PRINTING IT WILL BE IGNORED.                              :
                      C  ;       ADDRESS 50:0 CONTAINS THE STATUS OF THE PRINT SCREEN:        :
                      C  ;                                                                    :
                      C  ;       50:0    =0      EITHER PRINT SCREEN HAS NOT BEEN CALLED      :
                      C  ;                       OR UPON RETURN FROM A CALL THIS INDICATES    :
                      C  ;                       A SUCCESSFUL OPERATION.                      :
                      C  ;               =1      PRINT SCREEN IS IN PROGRESS                  :
                      C  ;               =255    ERROR ENCOUNTERED DURING PRINTING            :
                      C  ;--------------------------------------------------------------------
                      C          ASSUME  CS:CODE,DS:ABS0
21A7                  C  PRINT_SCREEN    PROC    FAR
21A7  FB              C          STI                         ; MUST RUN WITH INTS ENABLED
21A8  1E              C          PUSH    DS                  ; MUST USE 50:0 FOR DATA
21A9  50              C          PUSH    AX                  ;  AREA STORAGE
21AA  53              C          PUSH    BX
21AB  51              C          PUSH    CX                  ; USE THIS LATER FOR CURSOR LIMITS
21AC  52              C          PUSH    DX                  ; WILL HOLD CURRENT CURSOR POS
21AD  E8 0CFE R       C          CALL    DDS
21B0  80 3E 0500 R 01 C          CMP     STATUS_BYTE,1       ; SEE IF PRINT ALREADY IN PROGRESS
21B5  74 63           C          JZ      EXIT                ; JUMP IF PRINT IN PROGRESS
21B7  C6 06 0500 R 01 C          MOV     STATUS_BYTE,1       ; INDICATE PRINT NOW IN PROGRESS
21BC  B4 0F           C          MOV     AH,15               ; WILL REQUEST THE CURRENT MODE
21BE  CD 10           C          INT     10H                 ;       [AL]=MODE (NOT USED)
                      C                                      ;       [AH]=NUMBER COLUMNS/LINE
                      C                                      ;       [BH]=VISUAL PAGE
                      C  ;-----------------------------------------------------------
                      C  ;       AT THIS POINT WE KNOW THE COLUMNS/LINE ARE IN       :
                      C  ;       [AX] AND THE PAGE IF APPLICABLE IS IN [BH]. THE STACK :
                      C  ;       HAS DS,AX,BX,CX,DX PUSHED. [AL] HAS VIDEO MODE      :
                      C  ;-----------------------------------------------------------
21C0  8A CC           C          MOV     CL,AH               ; WILL MAKE USE OF [CX] REG TO
21C2  8A 2E 0484 R    C          MOV     CH,ROWS             ; CONTROL ROW & COLUMNS
21C6  FE C5           C          INC     CH                  ; ADJUST
21C8  E8 2220 R       C          CALL    CRLF                ; CAR RETURN LINE FEED ROUTINE
21CB  51              C          PUSH    CX                  ; SAVE SCREEN BOUNDS
21CC  B4 03           C          MOV     AH,3                ; WILL NOW READ THE CURSOR.
21CE  CD 10           C          INT     10H                 ; AND PRESERVE THE POSITION
21D0  59              C          POP     CX                  ; RECALL SCREEN BOUNDS
21D1  52              C          PUSH    DX                  ; RECALL [BH]=VISUAL PAGE
21D2  33 D2           C          XOR     DX,DX               ; SET CURSOR POSITION TO [0,0]
                      C  ;-----------------------------------------------------------
                      C  ;       THE LOOP FROM PRI10 TO THE INSTRUCTION PRIOR TO PRI20  :
                      C  ;       IS THE LOOP TO READ EACH CURSOR POSITION FROM THE      :
                      C  ;       SCREEN AND PRINT.                                      :
                      C  ;-----------------------------------------------------------
21D4                  C  PRI10:
21D4  B4 02           C          MOV     AH,2                ; TO INDICATE CURSOR SET REQUEST
21D6  CD 10           C          INT     10H                 ; NEW CURSOR POS ESTABLISHED
21D8  B4 08           C          MOV     AH,8                ; TO INDICATE READ CHARACTER
21DA  CD 10           C          INT     10H                 ; CHARACTER NOW IN [AL]
21DC  0A C0           C          OR      AL,AL               ; SEE IF VALID CHAR
```

```
21DE  75 02              C            JNZ     PRI15                 ; JUMP IF VALID CHAR
21E0  B0 20              C            MOV     AL,' '                ; MAKE A BLANK
21E2                     C  PRI15:
21E2  52                 C            PUSH    DX                    ; SAVE CURSOR POSITION
21E3  33 D2              C            XOR     DX,DX                 ; INDICATE PRINTER 1
21E5  32 E4              C            XOR     AH,AH                 ; TO INDICATE PRINT CHAR IN [AL]
21E7  CD 17              C            INT     17H                   ; PRINT THE CHARACTER
21E9  5A                 C            POP     DX                    ; RECALL CURSOR POSITION
21EA  F6 C4 29           C            TEST    AH,029H               ; TEST FOR PRINTER ERROR
21ED  75 21              C            JNZ     ERR10                 ; JUMP IF ERROR DETECTED
21EF  FE C2              C            INC     DL                    ; ADVANCE TO NEXT COLUMN
21F1  3A CA              C            CMP     CL,DL                 ; SEE IF AT END OF LINE
21F3  75 DF              C            JNZ     PRI10                 ; IF NOT PROCEED
21F5  32 D2              C            XOR     DL,DL                 ; BACK TO COLUMN 0
21F7  8A E2              C            MOV     AH,DL                 ; [AH]=0
21F9  52                 C            PUSH    DX                    ; SAVE NEW CURSOR POSITION
21FA  E8 2220 R          C            CALL    CRLF                  ; LINE FEED CARRIAGE RETURN
21FD  5A                 C            POP     DX                    ; RECALL CURSOR POSITION
21FE  FE C6              C            INC     DH                    ; ADVANCE TO NEXT LINE
2200  3A EE              C            CMP     CH,DH                 ; FINISHED?
2202  75 D0              C            JNZ     PRI10                 ; IF NOT CONTINUE
                         C
2204  5A                 C            POP     DX                    ; RECALL CURSOR POSITION
2205  B4 02              C            MOV     AH,2                  ; TO INDICATE CURSOR SET REQUEST
2207  CD 10              C            INT     10H                   ; CURSOR POSITION RESTORED
2209  C6 06 0500 R 00    C            MOV     STATUS_BYTE,0         ; INDICATE FINISHED
220E  EB 0A              C            JMP     SHORT EXIT            ; EXIT THE ROUTINE
2210                     C  ERR10:
2210  5A                 C            POP     DX                    ; GET CURSOR POSITION
2211  B4 02              C            MOV     AH,2                  ; TO REQUEST CURSOR SET
2213  CD 10              C            INT     10H                   ; CURSOR POSITION RESTORED
2215  C6 06 0500 R FF    C            MOV     STATUS_BYTE,0FFH      ; INDICATE ERROR
221A                     C  EXIT:
221A  5A                 C            POP     DX                    ; RESTORE ALL THE REGISTERS USED
221B  59                 C            POP     CX
221C  5B                 C            POP     BX
221D  58                 C            POP     AX
221E  1F                 C            POP     DS
221F  CF                 C            IRET
2220                     C  PRINT_SCREEN      ENDP
                         C
                         C  ;------ CARRIAGE RETURN, LINE FEED SUBROUTINE
                         C
2220                     C  CRLF      PROC    NEAR
2220  33 D2              C            XOR     DX,DX                 ; PRINTER 0
2222  32 E4              C            XOR     AH,AH                 ; WILL NOW SEND INITIAL CR, LF
                         C                                          ;  TO PRINTER
2224  B0 0D              C            MOV     AL,0DH                ; CR
2226  CD 17              C            INT     17H                   ; SEND THE LINE FEED
2228  32 E4              C            XOR     AH,AH                 ; NOW FOR THE CR
222A  B0 0A              C            MOV     AL,0AH                ; LF
222C  CD 17              C            INT     17H                   ; SEND THE CARRIAGE RETURN
222E  C3                 C            RET
222F                     C  CRLF      ENDP
                         C
                         C            SUBTTL

222F                        CODE      ENDS
                                      END


                                      PAGE,120
                                      SUBTTL  MONOCHROME CHARACTER GENERATOR
0000                        CODE      SEGMENT PUBLIC
                                      PUBLIC  CGMN
0000                        CGMN      LABEL   BYTE
                                                                                          ; BW 8*14 PATTTERN
0000  00 00 00 00 00 00               DB      000H,000H,000H,000H,000H,000H,000H,000H ; TOP_HALF_00
      00 00
0008  00 00 00 00 00 00               DB      000H,000H,000H,000H,000H,000H           ; BOTTOM_HALF 00
000E  00 00 7E 81 A5 81               DB      000H,000H,07EH,081H,0A5H,081H,081H,0BDH ; TH_01
      81 BD
0016  99 81 7E 00 00 00               DB      099H,081H,07EH,000H,000H,000H           ; BT_01
001C  00 00 7E FF DB FF               DB      000H,000H,07EH,0FFH,0DBH,0FFH,0FFH,0C3H ; TH_02
      FF C3
0024  E7 FF 7E 00 00 00               DB      0E7H,0FFH,07EH,000H,000H,000H           ; BT_02
002A  00 00 00 6C FE FE               DB      000H,000H,000H,06CH,0FEH,0FEH,0FEH,0FEH ; TH_03
      FE FE
0032  7C 38 10 00 00 00               DB      07CH,038H,010H,000H,000H,000H           ; BT_03
0038  00 00 00 10 38 7C               DB      000H,000H,000H,010H,038H,07CH,0FEH,07CH ; TH_04
      FE 7C
0040  38 10 00 00 00 00               DB      038H,010H,000H,000H,000H,000H           ; BT_04
0046  00 00 18 3C 3C E7               DB      000H,000H,018H,03CH,03CH,0E7H,0E7H,0E7H ; TH_05
      E7 E7
004E  18 18 3C 00 00 00               DB      018H,018H,03CH,000H,000H,000H           ; BT_05
0054  00 00 18 3C 7E FF               DB      000H,000H,018H,03CH,07EH,0FFH,0FFH,07EH ; TH_06
      FF 7E
005C  18 18 3C 00 00 00               DB      018H,018H,03CH,000H,000H,000H           ; BT_06
0062  00 00 00 00 00 18               DB      000H,000H,000H,000H,000H,018H,03CH,03CH ; TH_07
      3C 3C
006A  18 00 00 00 00 00               DB      018H,000H,000H,000H,000H,000H           ; BT_07
0070  FF FF FF FF FF E7               DB      0FFH,0FFH,0FFH,0FFH,0FFH,0E7H,0C3H,0C3H ; TH_08
      C3 C3
0078  E7 FF FF FF FF FF               DB      0E7H,0FFH,0FFH,0FFH,0FFH,0FFH           ; BT_08
007E  00 00 00 00 3C 66               DB      000H,000H,000H,000H,03CH,066H,042H,042H ; TH_09
      42 42
0086  66 3C 00 00 00 00               DB      066H,03CH,000H,000H,000H,000H           ; BT_09
008C  FF FF FF FF C3 99               DB      0FFH,0FFH,0FFH,0FFH,0C3H,099H,0BDH,0BDH ; TH_0A
      BD BD
0094  99 C3 FF FF FF FF               DB      099H,0C3H,0FFH,0FFH,0FFH,0FFH           ; BT_0A
009A  00 00 1E 0E 1A 32               DB      000H,000H,01EH,00EH,01AH,032H,078H,0CCH ; TH_0B
      78 CC
00A2  CC CC 78 00 00 00               DB      0CCH,0CCH,078H,000H,000H,000H           ; BT_0B
00A8  00 00 3C 66 66 66               DB      000H,000H,03CH,066H,066H,066H,03CH,018H ; TH_0C
      3C 18
00B0  7E 18 18 00 00 00               DB      07EH,018H,018H,000H,000H,000H           ; BT_0C
00B6  00 00 3F 33 3F 30               DB      000H,000H,03FH,033H,03FH,030H,030H,030H ; TH_0D
      30 30
00BE  70 F0 E0 00 00 00               DB      070H,0F0H,0E0H,000H,000H,000H           ; BT_0D
00C4  00 00 7F 63 7F 63               DB      000H,000H,07FH,063H,07FH,063H,063H,063H ; TH_0E
      63 63
00CC  67 E7 E6 C0 00 00               DB      067H,0E7H,0E6H,0C0H,000H,000H           ; BT_0E
00D2  00 00 18 18 DB 3C               DB      000H,000H,018H,018H,0DBH,03CH,0E7H,03CH ; TH_0F
      E7 3C
00DA  DB 18 18 00 00 00               DB      0DBH,018H,018H,000H,000H,000H           ; BT_0F

00E0  00 00 80 C0 E0 F8               DB      000H,000H,080H,0C0H,0E0H,0F8H,0FEH,0F8H ; TH_10
      FE F8
00E8  E0 C0 80 00 00 00               DB      0E0H,0C0H,080H,000H,000H,000H           ; BT_10
00EE  00 00 02 06 0E 3E               DB      000H,000H,002H,006H,00EH,03EH,0FEH,03EH ; TH_11
      FE 3E
00F6  0E 06 02 00 00 00               DB      00EH,006H,002H,000H,000H,000H           ; BT_11
00FC  00 00 18 3C 7E 18               DB      000H,000H,018H,03CH,07EH,018H,018H,018H ; TH_12
      18 18
0104  7E 3C 18 00 00 00               DB      07EH,03CH,018H,000H,000H,000H           ; BT_12
010A  00 00 66 66 66 66               DB      000H,000H,066H,066H,066H,066H,066H,066H ; TH-13
```

```
      66 66
0112  00 66 66 00 00 00    DB    000H,066H,066H,000H,000H,000H           ; BT_13
0118  00 00 7F DB DB DB    DB    000H,000H,07FH,0DBH,0DBH,0DBH,07BH,01BH ; TH_14
      7B 1B
0120  1B 1B 1B 00 00 00    DB    01BH,01BH,01BH,000H,000H,000H           ; BT_14
0126  00 7C C6 60 38 6C    DB    000H,07CH,0C6H,060H,038H,06CH,0C6H,0C6H ; TH_15
      C6 C6
012E  6C 38 0C C6 7C 00    DB    06CH,038H,00CH,0C6H,07CH,000H           ; BT_15
0134  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H,000H,000H ; TH_16
      00 00
013C  FE FE FE 00 00 00    DB    0FEH,0FEH,0FEH,000H,000H,000H           ; BT_16
0142  00 00 18 3C 7E 18    DB    000H,000H,018H,03CH,07EH,018H,018H,018H ; TH_17
      18 18
014A  7E 3C 18 7E 00 00    DB    07EH,03CH,018H,07EH,000H,000H           ; BT_17
0150  00 00 18 3C 7E 18    DB    000H,000H,018H,03CH,07EH,018H,018H,018H ; TH_18
      18 18
0158  18 18 18 00 00 00    DB    018H,018H,018H,000H,000H,000H           ; BT_18
015E  00 00 18 18 18 18    DB    000H,000H,018H,018H,018H,018H,018H,018H ; TH_19
      18 18
0166  7E 3C 18 00 00 00    DB    07EH,03CH,018H,000H,000H,000H           ; BT_19
016C  00 00 00 00 18 0C    DB    000H,000H,000H,000H,018H,00CH,0FEH,00CH ; TH_1A
      FE 0C
0174  18 00 00 00 00 00    DB    018H,000H,000H,000H,000H,000H           ; BT_1A
017A  00 00 00 00 30 60    DB    000H,000H,000H,000H,030H,060H,0FEH,060H ; TH_1B
      FE 60
0182  30 00 00 00 00 00    DB    030H,000H,000H,000H,000H,000H           ; BT_1B
0188  00 00 00 00 00 C0    DB    000H,000H,000H,000H,000H,0C0H,0C0H,0C0H ; TH_1C
      C0 C0
0190  FE 00 00 00 00 00    DB    0FEH,000H,000H,000H,000H,000H           ; BT_1C
0196  00 00 00 00 28 6C    DB    000H,000H,000H,000H,028H,06CH,0FEH,06CH ; TH_1D
      FE 6C
019E  28 00 00 00 00 00    DB    028H,000H,000H,000H,000H,000H           ; BT_1D
01A4  00 00 00 10 38 38    DB    000H,000H,000H,010H,038H,038H,07CH,07CH ; TH_1E
      7C 7C
01AC  FE FE 00 00 00 00    DB    0FEH,0FEH,000H,000H,000H,000H           ; BT_1E
01B2  00 00 00 FE FE 7C    DB    000H,000H,000H,0FEH,0FEH,07CH,07CH,038H ; TH_1F
      7C 38
01BA  38 10 00 00 00 00    DB    038H,010H,000H,000H,000H,000H           ; BT_1F

01C0  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H,000H,000H ; TH_20 SP
      00 00
01C8  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H           ; BT_20 SP
01CE  00 00 18 3C 3C 3C    DB    000H,000H,018H,03CH,03CH,03CH,018H,018H ; TH_21 !
      18 18
01D6  00 18 18 00 00 00    DB    000H,018H,018H,000H,000H,000H           ; BT_21 !
01DC  00 66 66 66 24 00    DB    000H,066H,066H,066H,024H,000H,000H,000H ; TH_22 "
      00 00
01E4  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H           ; BT_22 "
01EA  00 00 6C 6C FE 6C    DB    000H,000H,06CH,06CH,0FEH,06CH,06CH,06CH ; TH_23 #
      6C 6C
01F2  FE 6C 6C 00 00 00    DB    0FEH,06CH,06CH,000H,000H,000H           ; BT_23 #
01F8  18 18 7C C6 C2 C0    DB    018H,018H,07CH,0C6H,0C2H,0C0H,07CH,006H ; TH_24 $
      7C 06
0200  86 C6 7C 18 18 00    DB    086H,0C6H,07CH,018H,018H,000H           ; BT_24 $
0206  00 00 00 00 C2 C6    DB    000H,000H,000H,000H,0C2H,0C6H,00CH,018H ; TH_25 '%'
      0C 18
020E  30 66 C6 00 00 00    DB    030H,066H,0C6H,000H,000H,000H           ; BT_25 '%'
0214  00 00 38 6C 6C 38    DB    000H,000H,038H,06CH,06CH,038H,076H,0DCH ; TH_26 &
      76 DC
021C  CC CC 76 00 00 00    DB    0CCH,0CCH,076H,000H,000H,000H           ; BT_26 &
0222  00 30 30 30 60 00    DB    000H,030H,030H,030H,060H,000H,000H,000H ; TH_27 '
      00 00
022A  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H           ; BT 27 '
0230  00 00 0C 18 30 30    DB    000H,000H,00CH,018H,030H,030H,030H,030H ; TH_28 (
      30 30
0238  30 18 0C 00 00 00    DB    030H,018H,00CH,000H,000H,000H           ; BT_28 (
023E  00 00 30 18 0C 0C    DB    000H,000H,030H,018H,00CH,00CH,00CH,00CH ; TH_29 )
      0C 0C
0246  0C 18 30 00 00 00    DB    00CH,018H,030H,000H,000H,000H           ; BT_29 )
024C  00 00 00 00 66 3C    DB    000H,000H,000H,000H,066H,03CH,0FFH,03CH ; TH_2A *
      FF 3C
0254  66 00 00 00 00 00    DB    066H,000H,000H,000H,000H,000H           ; BT_2A *
025A  00 00 00 00 18 18    DB    000H,000H,000H,000H,018H,018H,07EH,018H ; TH_2B +
      7E 18
0262  18 00 00 00 00 00    DB    018H,000H,000H,000H,000H,000H           ; BT_2B +
0268  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H,000H,000H ; TH_2C ,
      00 00
0270  18 18 18 30 00 00    DB    018H,018H,018H,030H,000H,000H           ; BT_2C ,
0276  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H,0FEH,000H ; TH_2D -
      FE 00
027E  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H           ; BT_2D -
0284  00 00 00 00 00 00    DB    000H,000H,000H,000H,000H,000H,000H,000H ; TH_2E .
      00 00
028C  00 18 18 00 00 00    DB    000H,018H,018H,000H,000H,000H           ; BT_2E .
0292  00 00 02 06 0C 18    DB    000H,000H,002H,006H,00CH,018H,030H,060H ; TH_2F /
      30 60
029A  C0 80 00 00 00 00    DB    0C0H,080H,000H,000H,000H,000H           ; BT_2F /

02A0  00 00 7C C6 CE DE    DB    000H,000H,07CH,0C6H,0CEH,0DEH,0F6H,0E6H ; TH_30 0
      F6 E6
02A8  C6 C6 7C 00 00 00    DB    0C6H,0C6H,07CH,000H,000H,000H           ; BT_30 0
02AE  00 00 18 38 78 18    DB    000H,000H,018H,038H,078H,018H,018H,018H ; TH_31 1
      18 18
02B6  18 18 7E 00 00 00    DB    018H,018H,07EH,000H,000H,000H           ; BT_31 1
02BC  00 00 7C C6 06 0C    DB    000H,000H,07CH,0C6H,006H,00CH,018H,030H ; TH_32 2
      18 30
02C4  60 C6 FE 00 00 00    DB    060H,0C6H,0FEH,000H,000H,000H           ; BT_32 2
02CA  00 00 7C C6 06 06    DB    000H,000H,07CH,0C6H,006H,006H,03CH,006H ; TH_33 3
      3C 06
02D2  06 C6 7C 00 00 00    DB    006H,0C6H,07CH,000H,000H,000H           ; BT_33 3
02D8  00 00 0C 1C 3C 6C    DB    000H,000H,00CH,01CH,03CH,06CH,0CCH,0FEH ; TH_34 4
      CC FE
02E0  0C 0C 1E 00 00 00    DB    00CH,00CH,01EH,000H,000H,000H           ; BT_34 4
02E6  00 00 FE C0 C0 C0    DB    000H,000H,0FEH,0C0H,0C0H,0C0H,0FCH,006H ; TH_35 5
      FC 06
02EE  06 C6 7C 00 00 00    DB    006H,0C6H,07CH,000H,000H,000H           ; BT_35 5
02F4  00 00 38 60 C0 C0    DB    000H,000H,038H,060H,0C0H,0C0H,0FCH,0C6H ; TH_36 6
      FC C6
02FC  C6 C6 7C 00 00 00    DB    0C6H,0C6H,07CH,000H,000H,000H           ; BT_36 6
0302  00 00 FE C6 06 0C    DB    000H,000H,0FEH,0C6H,006H,00CH,018H,030H ; TH_37 7
      18 30
030A  30 30 30 00 00 00    DB    030H,030H,030H,000H,000H,000H           ; BT_37 7
0310  00 00 7C C6 C6 C6    DB    000H,000H,07CH,0C6H,0C6H,0C6H,07CH,0C6H ; TH_38 8
      7C C6
0318  C6 C6 7C 00 00 00    DB    0C6H,0C6H,07CH,000H,000H,000H           ; BT_38 8
031E  00 00 7C C6 C6 C6    DB    000H,000H,07CH,0C6H,0C6H,0C6H,07EH,006H ; TH_39 9
      7E 06
0326  06 0C 78 00 00 00    DB    006H,00CH,078H,000H,000H,000H           ; BT_39 9
032C  00 00 00 18 18 00    DB    000H,000H,000H,018H,018H,000H,000H,000H ; TH_3A :
      00 00
0334  18 18 00 00 00 00    DB    018H,018H,000H,000H,000H,000H           ; BT_3A :
033A  00 00 00 18 18 00    DB    000H,000H,000H,018H,018H,000H,000H,000H ; TH_3B ;
      00 00
0342  18 18 30 00 00 00    DB    018H,018H,030H,000H,000H,000H           ; BT_3B ;
0348  00 00 06 0C 18 30    DB    000H,000H,006H,00CH,018H,030H,060H,030H ; TH_3C <
      60 30
```

```
0350  18 0C 06 00 00 00     DB    018H,00CH,006H,000H,000H,000H           ; BT_3C <
0356  00 00 00 00 00 7E     DB    000H,000H,000H,000H,000H,07EH,000H,000H ; TH_3D =
      00 00
035E  7E 00 00 00 00 00     DB    07EH,000H,000H,000H,000H,000H           ; BT_3D =
0364  00 00 60 30 18 0C     DB    000H,000H,060H,030H,018H,00CH,006H,00CH ; TH_3E >
      06 0C
036C  18 30 60 00 00 00     DB    018H,030H,060H,000H,000H,000H           ; BT_3E >
0372  00 00 7C C6 C6 0C     DB    000H,000H,07CH,0C6H,0C6H,00CH,018H,018H ; TH_3F ?
      18 18
037A  00 18 18 00 00 00     DB    000H,018H,018H,000H,000H,000H           ; BT_3F ?

0380  00 00 7C C6 C6 DE     DB    000H,000H,07CH,0C6H,0C6H,0DEH,0DEH,0DEH ; TH_40 @
      DE DE
0388  DC C0 7C 00 00 00     DB    0DCH,0C0H,07CH,000H,000H,000H           ; BT_40 @
038E  00 00 10 38 6C C6     DB    000H,000H,010H,038H,06CH,0C6H,0C6H,0FEH ; TH_41 A
      C6 FE
0396  C6 C6 C6 00 00 00     DB    0C6H,0C6H,0C6H,000H,000H,000H           ; BT_41 A
039C  00 00 FC 66 66 66     DB    000H,000H,0FCH,066H,066H,066H,07CH,066H ; TH_42 B
      7C 66
03A4  66 66 FC 00 00 00     DB    066H,066H,0FCH,000H,000H,000H           ; BT_42 B
03AA  00 00 3C 66 C2 C0     DB    000H,000H,03CH,066H,0C2H,0C0H,0C0H,0C0H ; TH_43 C
      C0 C0
03B2  C2 66 3C 00 00 00     DB    0C2H,066H,03CH,000H,000H,000H           ; BT_43 C
03B8  00 00 F8 6C 66 66     DB    000H,000H,0F8H,06CH,066H,066H,066H,066H ; TH_44 D
      66 66
03C0  66 6C F8 00 00 00     DB    066H,06CH,0F8H,000H,000H,000H           ; BT_44 D
03C6  00 00 FE 66 62 68     DB    000H,000H,0FEH,066H,062H,068H,078H,068H ; TH_45 E
      78 68
03CE  62 66 FE 00 00 00     DB    062H,066H,0FEH,000H,000H,000H           ; BT_45 E
03D4  00 00 FE 66 62 68     DB    000H,000H,0FEH,066H,062H,068H,078H,068H ; TH_46 F
      78 68
03DC  60 60 F0 00 00 00     DB    060H,060H,0F0H,000H,000H,000H           ; BT_46 F
03E2  00 00 3C 66 C2 C0     DB    000H,000H,03CH,066H,0C2H,0C0H,0C0H,0DEH ; TH_47 G
      C0 DE
03EA  C6 66 3A 00 00 00     DB    0C6H,066H,03AH,000H,000H,000H           ; BT_47 G
03F0  00 00 C6 C6 C6 C6     DB    000H,000H,0C6H,0C6H,0C6H,0C6H,0FEH,0C6H ; TH_48 H
      FE C6
03F8  C6 C6 C6 00 00 00     DB    0C6H,0C6H,0C6H,000H,000H,000H           ; BT_48 H
03FE  00 00 3C 18 18 18     DB    000H,000H,03CH,018H,018H,018H,018H,018H ; TH_49 I
      18 18
0406  18 18 3C 00 00 00     DB    018H,018H,03CH,000H,000H,000H           ; BT_49 I
040C  00 00 1E 0C 0C 0C     DB    000H,000H,01EH,00CH,00CH,00CH,00CH,00CH ; TH_4A J
      0C 0C
0414  CC CC 78 00 00 00     DB    0CCH,0CCH,078H,000H,000H,000H           ; BT_4A J
041A  00 00 E6 66 6C 6C     DB    000H,000H,0E6H,066H,06CH,06CH,078H,06CH ; TH_4B K
      78 6C
0422  6C 66 E6 00 00 00     DB    06CH,066H,0E6H,000H,000H,000H           ; BT_4B K
0428  00 00 F0 60 60 60     DB    000H,000H,0F0H,060H,060H,060H,060H,060H ; TH_4C L
      60 60
0430  62 66 FE 00 00 00     DB    062H,066H,0FEH,000H,000H,000H           ; BT_4C L
0436  00 00 C6 EE FE FE     DB    000H,000H,0C6H,0EEH,0FEH,0FEH,0D6H,0C6H ; TH_4D M
      D6 C6
043E  C6 C6 C6 00 00 00     DB    0C6H,0C6H,0C6H,000H,000H,000H           ; BT_4D M
0444  00 00 C6 E6 F6 FE     DB    000H,000H,0C6H,0E6H,0F6H,0FEH,0DEH,0CEH ; TH_4E N
      DE CE
044C  C6 C6 C6 00 00 00     DB    0C6H,0C6H,0C6H,000H,000H,000H           ; BT_4E N
0452  00 00 38 6C C6 C6     DB    000H,000H,038H,06CH,0C6H,0C6H,0C6H,0C6H ; TH_4F O
      C6 C6
045A  C6 6C 38 00 00 00     DB    0C6H,06CH,038H,000H,000H,000H           ; BT_4F O

0460  00 00 FC 66 66 66     DB    000H,000H,0FCH,066H,066H,066H,07CH,060H ; TH_50 P
      7C 60
0468  60 60 F0 00 00 00     DB    060H,060H,0F0H,000H,000H,000H           ; BT_50 P
046E  00 00 7C C6 C6 C6     DB    000H,000H,07CH,0C6H,0C6H,0C6H,0C6H,0D6H ; TH_51 Q
      C6 D6
0476  DE 7C 0C 0E 00 00     DB    0DEH,07CH,00CH,00EH,000H,000H           ; BT_51 Q
047C  00 00 FC 66 66 66     DB    000H,000H,0FCH,066H,066H,066H,07CH,06CH ; TH_52 R
      7C 6C
0484  66 66 E6 00 00 00     DB    066H,066H,0E6H,000H,000H,000H           ; BT_52 R
048A  00 00 7C C6 C6 60     DB    000H,000H,07CH,0C6H,0C6H,060H,038H,00CH ; TH_53 S
      38 0C
0492  C6 C6 7C 00 00 00     DB    0C6H,0C6H,07CH,000H,000H,000H           ; BT_53 S
0498  00 00 7E 7E 5A 18     DB    000H,000H,07EH,07EH,05AH,018H,018H,018H ; TH_54 T
      18 18
04A0  18 18 3C 00 00 00     DB    018H,018H,03CH,000H,000H,000H           ; BT_54 T
04A6  00 00 C6 C6 C6 C6     DB    000H,000H,0C6H,0C6H,0C6H,0C6H,0C6H,0C6H ; TH_55 U
      C6 C6
04AE  C6 C6 7C 00 00 00     DB    0C6H,0C6H,07CH,000H,000H,000H           ; BT_55 U
04B4  00 00 C6 C6 C6 C6     DB    000H,000H,0C6H,0C6H,0C6H,0C6H,0C6H,0C6H ; TH_56 V
      C6 C6
04BC  6C 38 10 00 00 00     DB    06CH,038H,010H,000H,000H,000H           ; BT_56 V
04C2  00 00 C6 C6 C6 C6     DB    000H,000H,0C6H,0C6H,0C6H,0C6H,0D6H,0D6H ; TH_57 W
      D6 D6
04CA  FE 7C 6C 00 00 00     DB    0FEH,07CH,06CH,000H,000H,000H           ; BT_57 W
04D0  00 00 C6 C6 6C 38     DB    000H,000H,0C6H,0C6H,06CH,038H,038H,038H ; TH_58 X
      38 38
04D8  6C C6 C6 00 00 00     DB    06CH,0C6H,0C6H,000H,000H,000H           ; BT_58 X
04DE  00 00 66 66 66 66     DB    000H,000H,066H,066H,066H,066H,03CH,018H ; TH_59 Y
      3C 18
04E6  18 18 3C 00 00 00     DB    018H,018H,03CH,000H,000H,000H           ; BT_59 Y
04EC  00 00 FE C6 8C 18     DB    000H,000H,0FEH,0C6H,08CH,018H,030H,060H ; TH_5A Z
      30 60
04F4  C2 C6 FE 00 00 00     DB    0C2H,0C6H,0FEH,000H,000H,000H           ; BT_5A Z
04FA  00 00 3C 30 30 30     DB    000H,000H,03CH,030H,030H,030H,030H,030H ; TH_5B [
      30 30
0502  30 30 3C 00 00 00     DB    030H,030H,03CH,000H,000H,000H           ; BT_5B [
0508  00 00 80 C0 E0 70     DB    000H,000H,080H,0C0H,0E0H,070H,038H,01CH ; TH_5C
      38 1C
0510  0E 06 02 00 00 00     DB    00EH,006H,002H,000H,000H,000H           ; BT_5C
0516  00 00 3C 0C 0C 0C     DB    000H,000H,03CH,00CH,00CH,00CH,00CH,00CH ; TH_5D ]
      0C 0C
051E  0C 0C 3C 00 00 00     DB    00CH,00CH,03CH,000H,000H,000H           ; BT_5D ]
0524  10 38 6C C6 00 00     DB    010H,038H,06CH,0C6H,000H,000H,000H,000H ; TH_5E
      00 00
052C  00 00 00 00 00 00     DB    000H,000H,000H,000H,000H,000H           ; BT_5E
0532  00 00 00 00 00 00     DB    000H,000H,000H,000H,000H,000H,000H,000H ; TH_5F _
      00 00
053A  00 00 00 00 FF 00     DB    000H,000H,000H,000H,0FFH,000H           ; BT_5F _

0540  30 30 18 00 00 00     DB    030H,030H,018H,000H,000H,000H,000H,000H ; TH_60 '
      00 00
0548  00 00 00 00 00 00     DB    000H,000H,000H,000H,000H,000H           ; BT_60 '
054E  00 00 00 00 00 78     DB    000H,000H,000H,000H,000H,078H,00CH,07CH ; TH_61 LOWER_CASE A
      0C 7C
0556  CC CC 76 00 00 00     DB    0CCH,0CCH,076H,000H,000H,000H           ; BT_61 LOWER_CASE A
055C  00 00 E0 60 60 78     DB    000H,000H,0E0H,060H,060H,078H,06CH,066H ; TH_62 L.C. B
      6C 66
0564  66 66 7C 00 00 00     DB    066H,066H,07CH,000H,000H,000H           ; BT_62 L.C. B
056A  00 00 00 00 00 7C     DB    000H,000H,000H,000H,000H,07CH,0C6H,0C0H ; TH_63 L.C. C
      C6 C0
0572  C0 C6 7C 00 00 00     DB    0C0H,0C6H,07CH,000H,000H,000H           ; BT_63 L.C. C
0578  00 00 1C 0C 0C 3C     DB    000H,000H,01CH,00CH,00CH,03CH,06CH,0CCH ; TH_64 L.C. D
      6C CC
0580  CC CC 76 00 00 00     DB    0CCH,0CCH,076H,000H,000H,000H           ; BT_64 L.C. D
0586  00 00 00 00 00 7C     DB    000H,000H,000H,000H,000H,07CH,0C6H,0FEH ; TH_65 L.C. E
      C6 FE
```

```
058E  C0 C6 7C 00 00 00        DB   0C0H,0C6H,07CH,000H,000H,000H          ; BT_65 L.C. E
0594  00 00 38 6C 64 60        DB   000H,000H,038H,06CH,064H,060H,0F0H,060H ; TH_66 L.C. F
      F0 60
059C  60 60 F0 00 00 00        DB   060H,060H,0F0H,000H,000H,000H          ; BT_66 L.C. F
05A2  00 00 00 00 00 76        DB   000H,000H,000H,000H,000H,076H,0CCH,0CCH ; TH_67 L.C. G
      CC CC
05AA  CC 7C 0C CC 78 00        DB   0CCH,07CH,00CH,0CCH,078H,000H          ; BT_67 L.C. G
05B0  00 00 E0 60 60 6C        DB   000H,000H,0E0H,060H,060H,06CH,076H,066H ; TH_68 L.C. H
      76 66
05B8  66 66 E6 00 00 00        DB   066H,066H,0E6H,000H,000H,000H          ; BT_68 L.C. H
05BE  00 00 18 18 00 38        DB   000H,000H,018H,018H,000H,038H,018H,018H ; TH_69 L.C. I
      18 18
05C6  18 18 3C 00 00 00        DB   018H,018H,03CH,000H,000H,000H          ; BT_69 L.C. I
05CC  00 00 06 06 00 0E        DB   000H,000H,006H,006H,000H,00EH,006H,006H ; TH_6A L.C. J
      06 06
05D4  06 06 66 66 3C 00        DB   006H,006H,066H,066H,03CH,000H          ; BT_6A L.C. J
05DA  00 00 E0 60 60 66        DB   000H,000H,0E0H,060H,060H,066H,06CH,078H ; TH_6B L.C. K
      6C 78
05E2  6C 66 E6 00 00 00        DB   06CH,066H,0E6H,000H,000H,000H          ; BT_6B L.C. K
05E8  00 00 38 18 18 18        DB   000H,000H,038H,018H,018H,018H,018H,018H ; TH_6C L.C. L
      18 18
05F0  18 18 3C 00 00 00        DB   018H,018H,03CH,000H,000H,000H          ; BT_6C L.C. L
05F6  00 00 00 00 00 EC        DB   000H,000H,000H,000H,000H,0ECH,0FEH,0D6H ; TH_6D L.C. M
      FE D6
05FE  D6 D6 C6 00 00 00        DB   0D6H,0D6H,0C6H,000H,000H,000H          ; BT_6D L.C. M
0604  00 00 00 00 00 DC        DB   000H,000H,000H,000H,000H,0DCH,066H,066H ; TH_6E L.C. N
      66 66
060C  66 66 66 00 00 00        DB   066H,066H,066H,000H,000H,000H          ; BT_6E L.C. N
0612  00 00 00 00 00 7C        DB   000H,000H,000H,000H,000H,07CH,0C6H,0C6H ; TH_6F L.C. O
      C6 C6
061A  C6 C6 7C 00 00 00        DB   0C6H,0C6H,07CH,000H,000H,000H          ; BT_6F L.C. O

0620  00 00 00 00 00 DC        DB   000H,000H,000H,000H,000H,0DCH,066H,066H ; TH_70 L.C. P
      66 66
0628  66 7C 60 60 F0 00        DB   066H,07CH,060H,060H,0F0H,000H          ; BT_70 L.C. P
062E  00 00 00 00 00 76        DB   000H,000H,000H,000H,000H,076H,0CCH,0CCH ; TH_71 L.C. Q
      CC CC
0636  CC 7C 0C 0C 1E 00        DB   0CCH,07CH,00CH,00CH,01EH,000H          ; BT_71 L.C. Q
063C  00 00 00 00 00 DC        DB   000H,000H,000H,000H,000H,0DCH,076H,066H ; TH_72 L.C. R
      76 66
0644  60 60 F0 00 00 00        DB   060H,060H,0F0H,000H,000H,000H          ; BT_72 L.C. R
064A  00 00 00 00 00 7C        DB   000H,000H,000H,000H,000H,07CH,0C6H,070H ; TH_73 L.C. S
      C6 70
0652  1C C6 7C 00 00 00        DB   01CH,0C6H,07CH,000H,000H,000H          ; BT_73 L.C. S
0658  00 00 10 30 30 FC        DB   000H,000H,010H,030H,030H,0FCH,030H,030H ; TH_74 L.C. T
      30 30
0660  30 36 1C 00 00 00        DB   030H,036H,01CH,000H,000H,000H          ; BT_74 L.C. T
0666  00 00 00 00 00 CC        DB   000H,000H,000H,000H,000H,0CCH,0CCH,0CCH ; TH_75 L.C. U
      CC CC
066E  CC CC 76 00 00 00        DB   0CCH,0CCH,076H,000H,000H,000H          ; BT_75 L.C. U
0674  00 00 00 00 00 66        DB   000H,000H,000H,000H,000H,066H,066H,066H ; TH_76 L.C. V
      66 66
067C  66 3C 18 00 00 00        DB   066H,03CH,018H,000H,000H,000H          ; BT_76 L.C. V
0682  00 00 00 00 00 C6        DB   000H,000H,000H,000H,000H,0C6H,0C6H,0D6H ; TH_77 L.C. W
      C6 D6
068A  D6 FE 6C 00 00 00        DB   0D6H,0FEH,06CH,000H,000H,000H          ; BT_77 L.C. W
0690  00 00 00 00 00 C6        DB   000H,000H,000H,000H,000H,0C6H,06CH,038H ; TH_78 L.C. X
      6C 38
0698  38 6C C6 00 00 00        DB   038H,06CH,0C6H,000H,000H,000H          ; BT_78 L.C. X
069E  00 00 00 00 00 C6        DB   000H,000H,000H,000H,000H,0C6H,0C6H,0C6H ; TH_79 L.C. Y
      C6 C6
06A6  C6 7E 06 0C F8 00        DB   0C6H,07EH,006H,00CH,0F8H,000H          ; BT_79 L.C. Y
06AC  00 00 00 00 00 FE        DB   000H,000H,000H,000H,000H,0FEH,0CCH,018H ; TH_7A L.C. Z
      CC 18
06B4  30 66 FE 00 00 00        DB   030H,066H,0FEH,000H,000H,000H          ; BT_7A L.C. Z
06BA  00 00 0E 18 18 18        DB   000H,000H,00EH,018H,018H,018H,070H,018H ; TH_7B L BRAK
      70 18
06C2  18 18 0E 00 00 00        DB   018H,018H,00EH,000H,000H,000H          ; BT_7B L BRAK
06C8  00 00 18 18 18 18        DB   000H,000H,018H,018H,018H,018H,000H,018H ; TH_7C |
      00 18
06D0  18 18 18 00 00 00        DB   018H,018H,018H,000H,000H,000H          ; BT_7C |
06D6  00 00 70 18 18 18        DB   000H,000H,070H,018H,018H,018H,00EH,018H ; TH_7D R BRAK
      0E 18
06DE  18 18 70 00 00 00        DB   018H,018H,070H,000H,000H,000H          ; BT_7D R BRAK
06E4  00 00 76 DC 00 00        DB   000H,000H,076H,0DCH,000H,000H,000H,000H ; TH_7E TILDE
      00 00
06EC  00 00 00 00 00 00        DB   000H,000H,000H,000H,000H,000H          ; BT_7E TILDE
06F2  00 00 00 00 10 38        DB   000H,000H,000H,000H,010H,038H,06CH,0C6H ; TH_7F DELTA
      6C C6
06FA  C6 FE 00 00 00 00        DB   0C6H,0FEH,000H,000H,000H,000H          ; BT_7F DELTA

0700  00 00 3C 66 C2 C0        DB   000H,000H,03CH,066H,0C2H,0C0H,0C0H,0C2H ; TH_80
      C0 C2
0708  66 3C 0C 06 7C 00        DB   066H,03CH,00CH,006H,07CH,000H          ; BT_80
070E  00 00 CC CC 00 CC        DB   000H,000H,0CCH,0CCH,000H,0CCH,0CCH,0CCH ; TH_81
      CC CC
0716  CC CC 76 00 00 00        DB   0CCH,0CCH,076H,000H,000H,000H          ; BT_81
071C  00 0C 18 30 00 7C        DB   000H,00CH,018H,030H,000H,07CH,0C6H,0FEH ; TH_82
      C6 FE
0724  C0 C6 7C 00 00 00        DB   0C0H,0C6H,07CH,000H,000H,000H          ; BT_82
072A  00 10 38 6C 00 78        DB   000H,010H,038H,06CH,000H,078H,00CH,07CH ; TH_83
      0C 7C
0732  CC CC 76 00 00 00        DB   0CCH,0CCH,076H,000H,000H,000H          ; BT_83
0738  00 00 CC CC 00 78        DB   000H,000H,0CCH,0CCH,000H,078H,00CH,07CH ; TH_84
      0C 7C
0740  CC CC 76 00 00 00        DB   0CCH,0CCH,076H,000H,000H,000H          ; BT_84
0746  00 60 30 18 00 78        DB   000H,060H,030H,018H,000H,078H,00CH,07CH ; TH_85
      0C 7C
074E  CC CC 76 00 00 00        DB   0CCH,0CCH,076H,000H,000H,000H          ; BT_85
0754  00 38 6C 38 00 78        DB   000H,038H,06CH,038H,000H,078H,00CH,07CH ; TH_86
      0C 7C
075C  CC CC 76 00 00 00        DB   0CCH,0CCH,076H,000H,000H,000H          ; BT_86
0762  00 00 00 00 3C 66        DB   000H,000H,000H,000H,03CH,066H,060H,066H ; TH_87
      60 66
076A  3C 0C 06 3C 00 00        DB   03CH,00CH,006H,03CH,000H,000H          ; BT_87
0770  00 10 38 6C 00 7C        DB   000H,010H,038H,06CH,000H,07CH,0C6H,0FEH ; TH_88
      C6 FE
0778  C0 C6 7C 00 00 00        DB   0C0H,0C6H,07CH,000H.000H,000H          ; BT_88
077E  00 00 CC CC 00 7C        DB   000H,000H,0CCH,0CCH,000H,07CH,0C6H,0FEH ; TH_89
      C6 FE
0786  C0 C6 7C 00 00 00        DB   0C0H,0C6H,07CH,000H,000H,000H          ; BT_89
078C  00 60 30 18 00 7C        DB   000H,060H,030H,018H,000H,07CH,0C6H,0FEH ; TH_8A
      C6 FE
0794  C0 C6 7C 00 00 00        DB   0C0H,0C6H,07CH,000H,000H,000H          ; BT_8A
079A  00 00 66 66 00 38        DB   000H,000H,066H,066H,000H,038H,018H,018H ; TH_8B
      18 18
07A2  18 18 3C 00 00 00        DB   018H,018H,03CH,000H,000H,000H          ; BT_8B
07A8  00 18 3C 66 00 38        DB   000H,018H,03CH,066H,000H,038H,018H,018H ; TH_8C
      18 18
07B0  18 18 3C 00 00 00        DB   018H,018H,03CH,000H,000H,000H          ; BT_8C
07B6  00 60 30 18 00 38        DB   000H,060H,030H,018H,000H,038H,018H,018H ; TH_8D
      18 18
07BE  18 18 3C 00 00 00        DB   018H,018H,03CH,000H,000H,000H          ; BT_8D
07C4  00 C6 C6 10 38 6C        DB   000H,0C6H,0C6H,010H,038H,06CH,0C6H,0C6H ; TH_8E
      C6 C6
07CC  FE C6 C6 00 00 00        DB   0FEH,0C6H,0C6H,000H,000H,000H          ; BT_8E
```

```
07D2 38 6C 38 00 38 6C      DB   038H,06CH,038H,000H,038H,06CH,0C6H,0C6H ; TH_8F
     C6 C6
07DA FE C6 C6 00 00 00      DB   0FEH,0C6H,0C6H,000H,000H,000H           ; BT_8F

07E0 18 30 60 00 FE 66      DB   018H,030H,060H,000H,0FEH,066H,060H,07CH ; TH_90
     60 7C
07E8 60 66 FE 00 00 00      DB   060H,066H,0FEH,000H,000H,000H           ; BT_90
07EE 00 00 00 00 CC 76      DB   000H,000H,000H,000H,0CCH,076H,036H,07EH ; TH_91
     36 7E
07F6 D8 D8 6E 00 00 00      DB   0D8H,0D8H,06EH,000H,000H,000H           ; BT_91
07FC 00 00 3E 6C CC CC      DB   000H,000H,03EH,06CH,0CCH,0CCH,0FEH,0CCH ; TH_92
     FE CC
0804 CC CC CE 00 00 00      DB   0CCH,0CCH,0CEH,000H,000H,000H           ; BT_92
080A 00 10 38 6C 00 7C      DB   000H,010H,038H,06CH,000H,07CH,0C6H,0C6H ; TH_93
     C6 C6
0812 C6 C6 7C 00 00 00      DB   0C6H,0C6H,07CH,000H,000H,000H           ; BT_93
0818 00 00 C6 C6 00 7C      DB   000H,000H,0C6H,0C6H,000H,07CH,0C6H,0C6H ; TH_94
     C6 C6
0820 C6 C6 7C 00 00 00      DB   0C6H,0C6H,07CH,000H,000H,000H           ; BT_94
0826 00 60 30 18 00 7C      DB   000H,060H,030H,018H,000H,07CH,0C6H,0C6H ; TH_95
     C6 C6
082E C6 C6 7C 00 00 00      DB   0C6H,0C6H,07CH,000H,000H,000H           ; BT_95
0834 00 30 78 CC 00 CC      DB   000H,030H,078H,0CCH,000H,0CCH,0CCH,0CCH ; TH_96
     CC CC
083C CC CC 76 00 00 00      DB   0CCH,0CCH,076H,000H,000H,000H           ; BT_96
0842 00 60 30 18 00 CC      DB   000H,060H,030H,018H,000H,0CCH,0CCH,0CCH ; TH_97
     CC CC
084A CC CC 76 00 00 00      DB   0CCH,0CCH,076H,000H,000H,000H           ; BT_97
0850 00 00 C6 C6 00 C6      DB   000H,000H,0C6H,0C6H,000H,0C6H,0C6H,0C6H ; TH_98
     C6 C6
0858 C6 7E 06 0C 78 00      DB   0C6H,07EH,006H,00CH,078H,000H           ; BT_98
085E 00 C6 C6 38 6C C6      DB   000H,0C6H,0C6H,038H,06CH,0C6H,0C6H,0C6H ; TH_99
     C6 C6
0866 C6 6C 38 00 00 00      DB   0C6H,06CH,038H,000H,000H,000H           ; BT_99
086C 00 C6 C6 00 C6 C6      DB   000H,0C6H,0C6H,000H,0C6H,0C6H,0C6H,0C6H ; TH_9A
     C6 C6
0874 C6 C6 7C 00 00 00      DB   0C6H,0C6H,07CH,000H,000H,000H           ; BT_9A
087A 00 18 18 3C 66 60      DB   000H,018H,018H,03CH,066H,060H,060H,066H ; TH_9B
     60 66
0882 3C 18 18 00 00 00      DB   03CH,018H,018H,000H,000H,000H           ; BT_9B
0888 00 38 6C 64 60 F0      DB   000H,038H,06CH,064H,060H,0F0H,060H,060H ; TH_9C
     60 60
0890 60 E6 FC 00 00 00      DB   060H,0E6H,0FCH,000H,000H,000H           ; BT_9C
0896 00 00 66 66 3C 18      DB   000H,000H,066H,066H,03CH,018H,07EH,018H ; TH_9D
     7E 18
089E 7E 18 18 00 00 00      DB   07EH,018H,018H,000H,000H,000H           ; BT_9D
08A4 00 F8 CC CC F8 C4      DB   000H,0F8H,0CCH,0CCH,0F8H,0C4H,0CCH,0DEH ; TH_9E
     CC DE
08AC CC CC C6 00 00 00      DB   0CCH,0CCH,0C6H,000H,000H,000H           ; BT_9E
08B2 00 0E 1B 18 18 18      DB   000H,00EH,01BH,018H,018H,018H,07EH,018H ; TH_9F
     7E 18
08BA 18 18 18 D8 70 00      DB   018H,018H,018H,0D8H,070H,000H           ; BT_9F

08C0 00 18 30 60 00 78      DB   000H,018H,030H,060H,000H,078H,00CH,07CH ; TH_A0
     0C 7C
08C8 CC CC 76 00 00 00      DB   0CCH,0CCH,076H,000H,000H,000H           ; BT_A0
08CE 00 0C 18 30 00 38      DB   000H,00CH,018H,030H,000H,038H,018H,018H ; TH_A1
     18 18
08D6 18 18 3C 00 00 00      DB   018H,018H,03CH,000H,000H,000H           ; BT_A1
08DC 00 18 30 60 00 7C      DB   000H,018H,030H,060H,000H,07CH,0C6H,0C6H ; TH_A2
     C6 C6
08E4 C6 C6 7C 00 00 00      DB   0C6H,0C6H,07CH,000H,000H,000H           ; BT_A2
08EA 00 18 30 60 00 CC      DB   000H,018H,030H,060H,000H,0CCH,0CCH,0CCH ; TH_A3
     CC CC
08F2 CC CC 76 00 00 00      DB   0CCH,0CCH,076H,000H,000H,000H           ; BT_A3
08F8 00 00 76 DC 00 DC      DB   000H,000H,076H,0DCH,000H,0DCH,066H,066H ; TH_A4
     66 66
0900 66 66 66 00 00 00      DB   066H,066H,066H,000H,000H,000H           ; BT_A4
0906 76 DC 00 C6 E6 F6      DB   076H,0DCH,000H,0C6H,0E6H,0F6H,0FEH,0DEH ; TH_A5
     FE DE
090E CE C6 C6 00 00 00      DB   0CEH,0C6H,0C6H,000H,000H,000H           ; BT_A5
0914 00 3C 6C 6C 3E 00      DB   000H,03CH,06CH,06CH,03EH,000H,07EH,000H ; TH_A6
     7E 00
091C 00 00 00 00 00 00      DB   000H,000H,000H,000H,000H,000H           ; BT_A6
0922 00 38 6C 6C 38 00      DB   000H,038H,06CH,06CH,038H,000H,07CH,000H ; TH_A7
     7C 00
092A 00 00 00 00 00 00      DB   000H,000H,000H,000H,000H,000H           ; BT_A7
0930 00 00 30 30 00 30      DB   000H,000H,030H,030H,000H,030H,030H,060H ; TH_A8
     30 60
0938 C6 C6 7C 00 00 00      DB   0C6H,0C6H,07CH,000H,000H,000H           ; BT_A8
093E 00 00 00 00 00 00      DB   000H,000H,000H,000H,000H,000H,0FEH,0C0H ; TH_A9
     FE C0
0946 C0 C0 00 00 00 00      DB   0C0H,0C0H,000H,000H,000H,000H           ; BT_A9
094C 00 00 00 00 00 00      DB   000H,000H,000H,000H,000H,000H,0FEH,006H ; TH_AA
     FE 06
0954 06 06 00 00 00 00      DB   006H,006H,000H,000H,000H,000H           ; BT_AA
095A 00 C0 C0 C6 CC D8      DB   000H,0C0H,0C0H,0C6H,0CCH,0D8H,030H,060H ; TH_AB
     30 60
0962 DC 86 0C 18 3E 00      DB   0DCH,086H,00CH,018H,03EH,000H           ; BT_AB
0968 00 C0 C0 C6 CC D8      DB   000H,0C0H,0C0H,0C6H,0CCH,0D8H,030H,066H ; TH_AC
     30 66
0970 CE 9E 3E 06 06 00      DB   0CEH,09EH,03EH,006H,006H,000H           ; BT_AC
0976 00 00 18 18 00 18      DB   000H,000H,018H,018H,000H,018H,018H,03CH ; TH_AD
     18 3C
097E 3C 3C 18 00 00 00      DB   03CH,03CH,018H,000H,000H,000H           ; BT_AD
0984 00 00 00 00 36 6C      DB   000H,000H,000H,000H,036H,06CH,0D8H,06CH ; TH_AE
     D8 6C
098C 36 00 00 00 00 00      DB   036H,000H,000H,000H,000H,000H           ; BT_AE
0992 00 00 00 00 D8 6C      DB   000H,000H,000H,000H,0D8H,06CH,036H,06CH ; TH_AF
     36 6C
099A D8 00 00 00 00 00      DB   0D8H,000H,000H,000H,000H,000H           ; BT_AF

09A0 11 44 11 44 11 44      DB   011H,044H,011H,044H,011H,044H,011H,044H ; TH_B0
     11 44
09A8 11 44 11 44 11 44      DB   011H,044H,011H,044H,011H,044H           ; BT_B0
09AE 55 AA 55 AA 55 AA      DB   055H,0AAH,055H,0AAH,055H,0AAH,055H,0AAH ; TH_B1
     55 AA
09B6 55 AA 55 AA 55 AA      DB   055H,0AAH,055H,0AAH,055H,0AAH           ; BT_B1
09BC DD 77 DD 77 DD 77      DB   0DDH,077H,0DDH,077H,0DDH,077H,0DDH,077H ; TH_B2
     DD 77
09C4 DD 77 DD 77 DD 77      DB   0DDH,077H,0DDH,077H,0DDH,077H           ; BT_B2
09CA 18 18 18 18 18 18      DB   018H,018H,018H,018H,018H,018H,018H,018H ; TH_B3
     18 18
09D2 18 18 18 18 18 18      DB   018H,018H,018H,018H,018H,018H           ; BT_B3
09D8 18 18 18 18 18 18      DB   018H,018H,018H,018H,018H,018H,018H,0F8H ; TH_B4
     18 F8
09E0 18 18 18 18 18 18      DB   018H,018H,018H,018H,018H,018H           ; BT_B4
09E6 18 18 18 18 18 F8      DB   018H,018H,018H,018H,018H,0F8H,018H,0F8H ; TH_B5
     18 F8
09EE 18 18 18 18 18 18      DB   018H,018H,018H,018H,018H,018H           ; BT_B5
09F4 36 36 36 36 36 36      DB   036H,036H,036H,036H,036H,036H,036H,0F6H ; TH_B6
     36 F6
09FC 36 36 36 36 36 36      DB   036H,036H,036H,036H,036H,036H           ; BT_B6
0A02 00 00 00 00 00 00      DB   000H,000H,000H,000H,000H,000H,000H,0FEH ; TH_B7
     00 FE
0A0A 36 36 36 36 36 36      DB   036H,036H,036H,036H,036H,036H           ; BT_B7
```

```
0A10  00 00 00 00 00 F8        DB    000H,000H,000H,000H,000H,0F8H,018H,0F8H ; TH_B8
      18 F8
0A18  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_B8
0A1E  36 36 36 36 36 F6        DB    036H,036H,036H,036H,036H,0F6H,006H,0F6H ; TH_B9
      06 F6
0A26  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_B9
0A2C  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H,036H,036H ; TH_BA
      36 36
0A34  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_BA
0A3A  00 00 00 00 00 FE        DB    000H,000H,000H,000H,000H,0FEH,006H,0F6H ; TH_BB
      06 F6
0A42  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_BB
0A48  36 36 36 36 36 F6        DB    036H,036H,036H,036H,036H,0F6H,006H,0FEH ; TH_BC
      06 FE
0A50  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_BC
0A56  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H,036H,0FEH ; TH_BD
      36 FE
0A5E  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_BD
0A64  18 18 18 18 18 F8        DB    018H,018H,018H,018H,018H,0F8H,018H,0F8H ; TH_BE
      18 F8
0A6C  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_BE
0A72  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H,000H,0F8H ; TH_BF
      00 F8
0A7A  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_BF

0A80  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H,018H,01FH ; TH_C0
      18 1F
0A88  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_C0
0A8E  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H,018H,0FFH ; TH_C1
      18 FF
0A96  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_C1
0A9C  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H,000H,0FFH ; TH_C2
      00 FF
0AA4  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_C2
0AAA  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H,018H,01FH ; TH_C3
      18 1F
0AB2  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_C3
0AB8  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H,000H,0FFH ; TH_C4
      00 FF
0AC0  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_C4
0AC6  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H,018H,0FFH ; TH_C5
      18 FF
0ACE  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_C5
0AD4  18 18 18 18 18 1F        DB    018H,018H,018H,018H,018H,01FH,018H,01FH ; TH_C6
      18 1F
0ADC  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_C6
0AE2  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H,036H,037H ; TH_C7
      36 37
0AEA  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_C7
0AF0  36 36 36 36 36 37        DB    036H,036H,036H,036H,036H,037H,030H,03FH ; TH_C8
      30 3F
0AF8  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_C8
0AFE  00 00 00 00 00 3F        DB    000H,000H,000H,000H,000H,03FH,030H,037H ; TH_C9
      30 37
0B06  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_C9
0B0C  36 36 36 36 36 F7        DB    036H,036H,036H,036H,036H,0F7H,000H,0FFH ; TH_CA
      00 FF
0B14  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_CA
0B1A  00 00 00 00 00 FF        DB    000H,000H,000H,000H,000H,0FFH,000H,0F7H ; TH_CB
      00 F7
0B22  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_CB
0B28  36 36 36 36 36 37        DB    036H,036H,036H,036H,036H,037H,030H,037H ; TH_CC
      30 37
0B30  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_CC
0B36  00 00 00 00 00 FF        DB    000H,000H,000H,000H,000H,0FFH,000H,0FFH ; TH_CD
      00 FF
0B3E  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_CD
0B44  36 36 36 36 36 F7        DB    036H,036H,036H,036H,036H,0F7H,000H,0F7H ; TH_CE
      00 F7
0B4C  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_CE
0B52  18 18 18 18 18 FF        DB    018H,018H,018H,018H,018H,0FFH,000H,0FFH ; TH_CF
      00 FF
0B5A  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_CF

0B60  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H,036H,0FFH ; TH_D0
      36 FF
0B68  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_D0
0B6E  00 00 00 00 00 FF        DB    000H,000H,000H,000H,000H,0FFH,000H,0FFH ; TH_D1
      00 FF
0B76  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_D1
0B7C  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H,000H,0FFH ; TH_D2
      00 FF
0B84  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_D2
0B8A  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H,036H,03FH ; TH_D3
      36 3F
0B92  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_D3
0B98  18 18 18 18 18 1F        DB    018H,018H,018H,018H,018H,01FH,018H,01FH ; TH_D4
      18 1F
0BA0  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_D4
0BA6  00 00 00 00 00 1F        DB    000H,000H,000H,000H,000H,01FH,018H,01FH ; TH_D5
      18 1F
0BAE  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_D5
0BB4  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H,000H,03FH ; TH_D6
      00 3F
0BBC  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_D6
0BC2  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H,036H,0FFH ; TH_D7
      36 FF
0BCA  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H           ; BT_D7
0BD0  18 18 18 18 18 FF        DB    018H,018H,018H,018H,018H,0FFH,018H,0FFH ; TH_D8
      18 FF
0BD8  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_D8
0BDE  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H,018H,0F8H ; TH_D9
      18 F8
0BE6  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_D9
0BEC  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H,000H,01FH ; TH_DA
      00 1F
0BF4  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H           ; BT_DA
0BFA  FF FF FF FF FF FF        DB    0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH ; TH_DB
      FF FF
0C02  FF FF FF FF FF FF        DB    0FFH,0FFH,0FFH,0FFH,0FFH,0FFH           ; BT_DB
0C08  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H,000H,0FFH ; TH_DC
      00 FF
0C10  FF FF FF FF FF FF        DB    0FFH,0FFH,0FFH,0FFH,0FFH,0FFH           ; BT_DC
0C16  F0 F0 F0 F0 F0 F0        DB    0F0H,0F0H,0F0H,0F0H,0F0H,0F0H,0F0H,0F0H ; TH_DD
      F0 F0
0C1E  F0 F0 F0 F0 F0 F0        DB    0F0H,0F0H,0F0H,0F0H,0F0H,0F0H           ; BT_DD
0C24  0F 0F 0F 0F 0F 0F        DB    00FH,00FH,00FH,00FH,00FH,00FH,00FH,00FH ; TH_DE
      0F 0F
0C2C  0F 0F 0F 0F 0F 0F        DB    00FH,00FH,00FH,00FH,00FH,00FH           ; BT_DE
0C32  FF FF FF FF FF FF        DB    0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,000H ; TH_DF
      FF 00
0C3A  00 00 00 00 00 00        DB    000H,000H,000H,000H,000H,000H           ; BT_DF

0C40  00 00 00 00 00 76        DB    000H,000H,000H,000H,000H,076H,0DCH,0D8H ; TH_E0
      DC D8
0C48  D8 DC 76 00 00 00        DB    0D8H,0DCH,076H,000H,000H,000H           ; BT_E0
```

```
0C4E  00 00 00 00 7C C6                      DB      000H,000H,000H,000H,07CH,0C6H,0FCH,0C6H ; TH_E1
      FC C6
0C56  C6 FC C0 C0 40 00                      DB      0C6H,0FCH,0C0H,0C0H,040H,000H           ; BT_E1
0C5C  00 00 FE C6 C6 C0                      DB      000H,000H,0FEH,0C6H,0C6H,0C0H,0C0H,0C0H ; TH_E2
      C0 C0
0C64  C0 C0 C0 00 00 00                      DB      0C0H,0C0H,0C0H,000H,000H,000H           ; BT_E2
0C6A  00 00 00 00 FE 6C                      DB      000H,000H,000H,000H,0FEH,06CH,06CH,06CH ; TH_E3
      6C 6C
0C72  6C 6C 6C 00 00 00                      DB      06CH,06CH,06CH,000H,000H,000H           ; BT_E3
0C78  00 00 FE C6 60 30                      DB      000H,000H,0FEH,0C6H,060H,030H,018H,030H ; TH_E4
      18 30
0C80  60 C6 FE 00 00 00                      DB      060H,0C6H,0FEH,000H,000H,000H           ; BT_E4
0C86  00 00 00 00 00 7E                      DB      000H,000H,000H,000H,000H,07EH,0D8H,0D8H ; TH_E5
      D8 D8
0C8E  D8 D8 70 00 00 00                      DB      0D8H,0D8H,070H,000H,000H,000H           ; BT_E5
0C94  00 00 00 00 66 66                      DB      000H,000H,000H,000H,066H,066H,066H,066H ; TH_E6
      66 66
0C9C  7C 60 60 C0 00 00                      DB      07CH,060H,060H,0C0H,000H,000H           ; BT_E6
0CA2  00 00 00 00 76 DC                      DB      000H,000H,000H,000H,076H,0DCH,018H,018H ; TH_E7
      18 18
0CAA  18 18 18 00 00 00                      DB      018H,018H,018H,000H,000H,000H           ; BT_E7
0CB0  00 00 7E 18 3C 66                      DB      000H,000H,07EH,018H,03CH,066H,066H,066H ; TH_E8
      66 66
0CB8  3C 18 7E 00 00 00                      DB      03CH,018H,07EH,000H,000H,000H           ; BT_E8
0CBE  00 00 38 6C C6 C6                      DB      000H,000H,038H,06CH,0C6H,0C6H,0FEH,0C6H ; TH_E9
      FE C6
0CC6  C6 6C 38 00 00 00                      DB      0C6H,06CH,038H,000H,000H,000H           ; BT_E9
0CCC  00 00 38 6C C6 C6                      DB      000H,000H,038H,06CH,0C6H,0C6H,0C6H,06CH ; TH_EA
      C6 6C
0CD4  6C 6C EE 00 00 00                      DB      06CH,06CH,0EEH,000H,000H,000H           ; BT_EA
0CDA  00 00 1E 30 18 0C                      DB      000H,000H,01EH,030H,018H,00CH,03EH,066H ; TH_EB
      3E 66
0CE2  66 66 3C 00 00 00                      DB      066H,066H,03CH,000H,000H,000H           ; BT_EB
0CE8  00 00 00 00 00 7E                      DB      000H,000H,000H,000H,000H,07EH,0DBH,0DBH ; TH_EC
      DB DB
0CF0  7E 00 00 00 00 00                      DB      07EH,000H,000H,000H,000H,000H           ; BT_EC
0CF6  00 00 03 06 7E DB                      DB      000H,000H,003H,006H,07EH,0DBH,0DBH,0F3H ; TH_ED
      DB F3
0CFE  7E 60 C0 00 00 00                      DB      07EH,060H,0C0H,000H,000H,000H           ; BT_ED
0D04  00 00 1C 30 60 60                      DB      000H,000H,01CH,030H,060H,060H,07CH,060H ; TH_EE
      7C 60
0D0C  60 30 1C 00 00 00                      DB      060H,030H,01CH,000H,000H,000H           ; BT_EE
0D12  00 00 00 7C C6 C6                      DB      000H,000H,000H,07CH,0C6H,0C6H,0C6H,0C6H ; TH_EF
      C6 C6
0D1A  C6 C6 C6 00 00 00                      DB      0C6H,0C6H,0C6H,000H,000H,000H           ; BT_EF

0D20  00 00 00 FE 00 00                      DB      000H,000H,000H,0FEH,000H,000H,0FEH,000H ; TH_F0
      FE 00
0D28  00 FE 00 00 00 00                      DB      000H,0FEH,000H,000H,000H,000H           ; BT_F0
0D2E  00 00 00 18 18 7E                      DB      000H,000H,000H,018H,018H,07EH,018H,018H ; TH_F1
      18 18
0D36  00 00 FF 00 00 00                      DB      000H,000H,0FFH,000H,000H,000H           ; BT_F1
0D3C  00 00 30 18 0C 06                      DB      000H,000H,030H,018H,00CH,006H,00CH,018H ; TH_F2
      0C 18
0D44  30 00 7E 00 00 00                      DB      030H,000H,07EH,000H,000H,000H           ; BT_F2
0D4A  00 00 0C 18 30 60                      DB      000H,000H,00CH,018H,030H,060H,030H,018H ; TH_F3
      30 18
0D52  0C 00 7E 00 00 00                      DB      00CH,000H,07EH,000H,000H,000H           ; BT_F3
0D58  00 00 0E 1B 1B 18                      DB      000H,000H,00EH,01BH,01BH,018H,018H,018H ; TH_F4
      18 18
0D60  18 18 18 18 18 18                      DB      018H,018H,018H,018H,018H,018H           ; BT_F4
0D66  18 18 18 18 18 18                      DB      018H,018H,018H,018H,018H,018H,018H,018H ; TH_F5
      18 18
0D6E  D8 D8 70 00 00 00                      DB      0D8H,0D8H,070H,000H,000H,000H           ; BT_F5
0D74  00 00 00 18 18 00                      DB      000H,000H,000H,018H,018H,000H,07EH,000H ; TH_F6
      7E 00
0D7C  18 18 00 00 00 00                      DB      018H,018H,000H,000H,000H,000H           ; BT_F6
0D82  00 00 00 00 76 DC                      DB      000H,000H,000H,000H,076H,0DCH,000H,076H ; TH_F7
      00 76
0D8A  DC 00 00 00 00 00                      DB      0DCH,000H,000H,000H,000H,000H           ; BT_F7
0D90  00 38 6C 6C 38 00                      DB      000H,038H,06CH,06CH,038H,000H,000H,000H ; TH_F8
      00 00
0D98  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H           ; BT_F8
0D9E  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H,018H,018H ; TH_F9
      18 18
0DA6  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H           ; BT_F9
0DAC  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H,000H,018H ; TH_FA
      00 18
0DB4  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H           ; BT_FA
0DBA  00 0F 0C 0C 0C 0C                      DB      000H,00FH,00CH,00CH,00CH,00CH,00CH,0ECH ; TH_FB
      0C EC
0DC2  6C 3C 1C 00 00 00                      DB      06CH,03CH,01CH,000H,000H,000H           ; BT_FB
0DC8  00 D8 6C 6C 6C 6C                      DB      000H,0D8H,06CH,06CH,06CH,06CH,06CH,000H ; TH_FC
      6C 00
0DD0  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H           ; BT_FC
0DD6  00 70 D8 30 60 C8                      DB      000H,070H,0D8H,030H,060H,0C8H,0F8H,000H ; TH_FD
      F8 00
0DDE  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H           ; BT_FD
0DE4  00 00 00 00 7C 7C                      DB      000H,000H,000H,000H,07CH,07CH,07CH,07CH ; TH_FE
      7C 7C
0DEC  7C 7C 00 00 00 00                      DB      07CH,07CH,000H,000H,000H,000H           ; BT_FE
0DF2  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H,000H,000H ; TH_FF
      00 00
0DFA  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H           ; BT_FF
0E00                              CODE       ENDS
                                             END


                                             PAGE,120
                                             SUBTTL  MONOCHROME CHARACTER GENERATOR  -  ALPHA SUPPLEMENT
0000                              CODE       SEGMENT PUBLIC
                                             PUBLIC  CGMN_FDG
0000                              CGMN_FDG           LABEL   BYTE

                                  ; STRUCTURE OF THIS FILE
                                  ;          DB      XXH     WHERE XX IS THE HEX CODE FOR THE FOLLOWING CHAR
                                  ;          DB      [BYTES 0 - 13 OF THAT CHARACTER]
                                  ;          ...
                                  ;          DB      00H     INDICATES NO MORE REPLACEMENTS TO BE DONE


0000  1D                                     DB      01DH                                    ;
0001  00 00 00 00 24 66                      DB      000H,000H,000H,000H,024H,066H,0FFH,066H ; TH_1D
      FF 66
0009  24 00 00 00 00 00                      DB      024H,000H,000H,000H,000H,000H           ; BT_1D
000F  22                                     DB      022H                                    ;
0010  00 63 63 63 22 00                      DB      000H,063H,063H,063H,022H,000H,000H,000H ; TH_22 "
      00 00
0018  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H           ; BT_22 "
001E  2B                                     DB      02BH                                    ;
001F  00 00 00 18 18 18                      DB      000H,000H,000H,018H,018H,018H,0FFH,018H ; TH_2B +
      FF 18
0027  18 18 00 00 00 00                      DB      018H,018H,000H,000H,000H,000H           ; BT_2B +
002D  2D                                     DB      02DH                                    ;
002E  00 00 00 00 00 00                      DB      000H,000H,000H,000H,000H,000H,0FFH,000H ; TH_2D -
      FF 00
```

```
0036  00 00 00 00 00 00                DB      000H,000H,000H,000H,000H,000H              ; BT_2D -
003C  4D                               DB      04DH                                       ;
003D  00 00 C3 E7 FF DB                DB      000H,000H,0C3H,0E7H,0FFH,0DBH,0C3H,0C3H ; TH_4D M
      C3 C3
0045  C3 C3 C3 00 00 00                DB      0C3H,0C3H,0C3H,000H,000H,000H              ; BT_4D M
004B  54                               DB      054H                                       ;
004C  00 00 FF DB 99 18                DB      000H,000H,0FFH,0DBH,099H,018H,018H,018H ; TH_54 T
      18 18
0054  18 18 3C 00 00 00                DB      018H,018H,03CH,000H,000H,000H              ; BT_54 T
005A  56                               DB      056H                                       ;
005B  00 00 C3 C3 C3 C3                DB      000H,000H,0C3H,0C3H,0C3H,0C3H,0C3H,0C3H ; TH_56 V
      C3 C3
0063  66 3C 18 00 00 00                DB      066H,03CH,018H,000H,000H,000H              ; BT_56 V
0069  57                               DB      057H                                       ;
006A  00 00 C3 C3 C3 C3                DB      000H,000H,0C3H,0C3H,0C3H,0C3H,0DBH,0DBH ; TH_57 W
      DB DB
0072  FF 66 66 00 00 00                DB      0FFH,066H,066H,000H,000H,000H              ; BT_57 W
0078  58                               DB      058H                                       ;
0079  00 00 C3 C3 66 3C                DB      000H,000H,0C3H,0C3H,066H,03CH,018H,03CH ; TH_58 X
      18 3C
0081  66 C3 C3 00 00 00                DB      066H,0C3H,0C3H,000H,000H,000H              ; BT_58 X
0087  59                               DB      059H                                       ;
0088  00 00 C3 C3 C3 66                DB      000H,000H,0C3H,0C3H,0C3H,066H,03CH,018H ; TH_59 Y
      3C 18
0090  18 18 3C 00 00 00                DB      018H,018H,03CH,000H,000H,000H              ; BT_59 Y
0096  5A                               DB      05AH                                       ;
0097  00 00 FF C3 86 0C                DB      000H,000H,0FFH,0C3H,086H,00CH,018H,030H ; TH_5A Z
      18 30
009F  61 C3 FF 00 00 00                DB      061H,0C3H,0FFH,000H,000H,000H              ; BT_5A Z
00A5  6D                               DB      06DH                                       ;
00A6  00 00 00 00 00 E6                DB      000H,000H,000H,000H,000H,0E6H,0FFH,0DBH ; TH_6D L.C. M
      FF DB
00AE  DB DB DB 00 00 00                DB      0DBH,0DBH,0DBH,000H,000H,000H              ; BT_6D L.C. M
00B4  76                               DB      076H                                       ;
00B5  00 00 00 00 00 C3                DB      000H,000H,000H,000H,000H,0C3H,0C3H,0C3H ; TH_76 L.C. V
      C3 C3
00BD  66 3C 18 00 00 00                DB      066H,03CH,018H,000H,000H,000H              ; BT_76 L.C. V
00C3  77                               DB      077H                                       ;
00C4  00 00 00 00 00 C3                DB      000H,000H,000H,000H,000H,0C3H,0C3H,0DBH ; TH_77 L.C. W
      C3 DB
00CC  DB FF 66 00 00 00                DB      0DBH,0FFH,066H,000H,000H,000H              ; BT_77 L.C. W
00D2  91                               DB      091H                                       ;
00D3  00 00 00 00 6E 3B                DB      000H,000H,000H,000H,06EH,03BH,01BH,07EH ; TH_91
      1B 7E
00DB  D8 DC 77 00 00 00                DB      0D8H,0DCH,077H,000H,000H,000H              ; BT_91
00E1  9B                               DB      09BH                                       ;
00E2  00 18 18 7E C3 C0                DB      000H,018H,018H,07EH,0C3H,0C0H,0C0H,0C3H ; TH_9B
      C0 C3
00EA  7E 18 18 00 00 00                DB      07EH,018H,018H,000H,000H,000H              ; BT_9B
00F0  9D                               DB      09DH                                       ;
00F1  00 00 C3 66 3C 18                DB      000H,000H,0C3H,066H,03CH,018H,0FFH,018H ; TH_9D
      FF 18
00F9  FF 18 18 00 00 00                DB      0FFH,018H,018H,000H,000H,000H              ; BT_9D
00FF  9E                               DB      09EH                                       ;
0100  00 FC 66 66 7C 62                DB      000H,0FCH,066H,066H,07CH,062H,066H,06FH ; TH_9E
      66 6F
0108  66 66 F3 00 00 00                DB      066H,066H,0F3H,000H,000H,000H              ; BT_9E
010E  F1                               DB      0F1H                                       ;
010F  00 00 18 18 18 FF                DB      000H,000H,018H,018H,018H,0FFH,018H,018H ; TH_F1
      18 18
0117  18 00 FF 00 00 00                DB      018H,000H,0FFH,000H,000H,000H              ; BT_F1
011D  F6                               DB      0F6H                                       ;
011E  00 00 18 18 00 00                DB      000H,000H,018H,018H,000H,000H,0FFH,000H ; TH_F6
      FF 00
0126  00 18 18 00 00 00                DB      000H,018H,018H,000H,000H,000H              ; BT_F6
012C  00                               DB      000H                                       ; NO MORE
012D                           CODE    ENDS
                                       END


                                       PAGE,120
                                       SUBTTL  DOUBLE DOT CHARACTER GENERATOR
0000                           CODE    SEGMENT PUBLIC
                                       PUBLIC  CGDDOT,INT_1F_1
0000                           CGDDOT  LABEL   BYTE
                                                                                          ; DOUBLE DOT
0000  00 00 00 00 00 00                DB      000H,000H,000H,000H,000H,000H,000H,000H ; D_00
      00 00
0008  7E 81 A5 81 BD 99                DB      07EH,081H,0A5H,081H,0BDH,099H,081H,07EH ; D_01
      81 7E
0010  7E FF DB FF C3 E7                DB      07EH,0FFH,0DBH,0FFH,0C3H,0E7H,0FFH,07EH ; D_02
      FF 7E
0018  6C FE FE FE 7C 38                DB      06CH,0FEH,0FEH,0FEH,07CH,038H,010H,000H ; D_03
      10 00
0020  10 38 7C FE 7C 38                DB      010H,038H,07CH,0FEH,07CH,038H,010H,000H ; D_04
      10 00
0028  38 7C 38 FE FE 7C                DB      038H,07CH,038H,0FEH,0FEH,07CH,038H,07CH ; D_05
      38 7C
0030  10 10 38 7C FE 7C                DB      010H,010H,038H,07CH,0FEH,07CH,038H,07CH ; D_06
      38 7C
0038  00 00 18 3C 3C 18                DB      000H,000H,018H,03CH,03CH,018H,000H,000H ; D_07
      00 00
0040  FF FF E7 C3 C3 E7                DB      0FFH,0FFH,0E7H,0C3H,0C3H,0E7H,0FFH,0FFH ; D_08
      FF FF
0048  00 3C 66 42 42 66                DB      000H,03CH,066H,042H,042H,066H,03CH,000H ; D_09
      3C 00
0050  FF C3 99 BD BD 99                DB      0FFH,0C3H,099H,0BDH,0BDH,099H,0C3H,0FFH ; D_0A
      C3 FF
0058  0F 07 0F 7D CC CC                DB      00FH,007H,00FH,07DH,0CCH,0CCH,0CCH,078H ; D_0B
      CC 78
0060  3C 66 66 66 3C 18                DB      03CH,066H,066H,066H,03CH,018H,07EH,018H ; D_0C
      7E 18
0068  3F 33 3F 30 30 70                DB      03FH,033H,03FH,030H,030H,070H,0F0H,0E0H ; D_0D
      F0 E0
0070  7F 63 7F 63 63 67                DB      07FH,063H,07FH,063H,063H,067H,0E6H,0C0H ; D_0E
      E6 C0
0078  99 5A 3C E7 E7 3C                DB      099H,05AH,03CH,0E7H,0E7H,03CH,05AH,099H ; D_0F
      5A 99

0080  80 E0 F8 FE F8 E0                DB      080H,0E0H,0F8H,0FEH,0F8H,0E0H,080H,000H ; D_10
      80 00
0088  02 0E 3E FE 3E 0E                DB      002H,00EH,03EH,0FEH,03EH,00EH,002H,000H ; D_11
      02 00
0090  18 3C 7E 18 18 7E                DB      018H,03CH,07EH,018H,018H,07EH,03CH,018H ; D_12
      3C 18
0098  66 66 66 66 66 00                DB      066H,066H,066H,066H,066H,000H,066H,000H ; D_13
      66 00
00A0  7F DB DB 7B 1B 1B                DB      07FH,0DBH,0DBH,07BH,01BH,01BH,01BH,000H ; D_14
      1B 00
00A8  3E 63 38 6C 6C 38                DB      03EH,063H,038H,06CH,06CH,038H,0CCH,078H ; D_15
      CC 78
00B0  00 00 00 00 7E 7E                DB      000H,000H,000H,000H,07EH,07EH,07EH,000H ; D_16
      7E 00
00B8  18 3C 7E 18 7E 3C                DB      018H,03CH,07EH,018H,07EH,03CH,018H,0FFH ; D_17
      18 FF
00C0  18 3C 7E 18 18 18                DB      018H,03CH,07EH,018H,018H,018H,018H,000H ; D_18
```

```
      18 00
00C8  18 18 18 18 7E 3C     DB    018H,018H,018H,018H,07EH,03CH,018H,000H ; D_19
      18 00
00D0  00 18 0C FE 0C 18     DB    000H,018H,00CH,0FEH,00CH,018H,000H,000H ; D_1A
      00 00
00D8  00 30 60 FE 60 30     DB    000H,030H,060H,0FEH,060H,030H,000H,000H ; D_1B
      00 00
00E0  00 00 C0 C0 C0 FE     DB    000H,000H,0C0H,0C0H,0C0H,0FEH,000H,000H ; D_1C
      00 00
00E8  00 24 66 FF 66 24     DB    000H,024H,066H,0FFH,066H,024H,000H,000H ; D_1D
      00 00
00F0  00 18 3C 7E FF FF     DB    000H,018H,03CH,07EH,0FFH,0FFH,000H,000H ; D_1E
      00 00
00F8  00 FF FF 7E 3C 18     DB    000H,0FFH,0FFH,07EH,03CH,018H,000H,000H ; D_1F
      00 00

0100  00 00 00 00 00 00     DB    000H,000H,000H,000H,000H,000H,000H,000H ; SP D_20
      00 00
0108  30 78 78 30 30 00     DB    030H,078H,078H,030H,030H,000H,030H,000H ; ! D_21
      30 00
0110  6C 6C 6C 00 00 00     DB    06CH,06CH,06CH,000H,000H,000H,000H,000H ; " D_22
      00 00
0118  6C 6C FE 6C FE 6C     DB    06CH,06CH,0FEH,06CH,0FEH,06CH,06CH,000H ; # D_23
      6C 00
0120  30 7C C0 78 0C F8     DB    030H,07CH,0C0H,078H,00CH,0F8H,030H,000H ; $ D_24
      30 00
0128  00 C6 CC 18 30 66     DB    000H,0C6H,0CCH,018H,030H,066H,0C6H,000H ; PER CENT D_25
      C6 00
0130  38 6C 38 76 DC CC     DB    038H,06CH,038H,076H,0DCH,0CCH,076H,000H ; & D_26
      76 00
0138  60 60 C0 00 00 00     DB    060H,060H,0C0H,000H,000H,000H,000H,000H ; ' D_27
      00 00
0140  18 30 60 60 60 30     DB    018H,030H,060H,060H,060H,030H,018H,000H ; ( D_28
      18 00
0148  60 30 18 18 18 30     DB    060H,030H,018H,018H,018H,030H,060H,000H ; ) D_29
      60 00
0150  00 66 3C FF 3C 66     DB    000H,066H,03CH,0FFH,03CH,066H,000H,000H ; * D_2A
      00 00
0158  00 30 30 FC 30 30     DB    000H,030H,030H,0FCH,030H,030H,000H,000H ; + D_2B
      00 00
0160  00 00 00 00 00 30     DB    000H,000H,000H,000H,000H,030H,030H,060H ; , D_2C
      30 60
0168  00 00 00 FC 00 00     DB    000H,000H,000H,0FCH,000H,000H,000H,000H ; - D_2D
      00 00
0170  00 00 00 00 00 30     DB    000H,000H,000H,000H,000H,030H,030H,000H ; . D_2E
      30 00
0178  06 0C 18 30 60 C0     DB    006H,00CH,018H,030H,060H,0C0H,080H,000H ; / D_2F
      80 00

0180  7C C6 CE DE F6 E6     DB    07CH,0C6H,0CEH,0DEH,0F6H,0E6H,07CH,000H ; 0 D_30
      7C 00
0188  30 70 30 30 30 30     DB    030H,070H,030H,030H,030H,030H,0FCH,000H ; 1 D_31
      FC 00
0190  78 CC 0C 38 60 CC     DB    078H,0CCH,00CH,038H,060H,0CCH,0FCH,000H ; 2 D_32
      FC 00
0198  78 CC 0C 38 0C CC     DB    078H,0CCH,00CH,038H,00CH,0CCH,078H,000H ; 3 D_33
      78 00
01A0  1C 3C 6C CC FE 0C     DB    01CH,03CH,06CH,0CCH,0FEH,00CH,01EH,000H ; 4 D_34
      1E 00
01A8  FC C0 F8 0C 0C CC     DB    0FCH,0C0H,0F8H,00CH,00CH,0CCH,078H,000H ; 5 D_35
      78 00
01B0  38 60 C0 F8 CC CC     DB    038H,060H,0C0H,0F8H,0CCH,0CCH,078H,000H ; 6 D_36
      78 00
01B8  FC CC 0C 18 30 30     DB    0FCH,0CCH,00CH,018H,030H,030H,030H,000H ; 7 D_37
      30 00
01C0  78 CC CC 78 CC CC     DB    078H,0CCH,0CCH,078H,0CCH,0CCH,078H,000H ; 8 D_38
      78 00
01C8  78 CC CC 7C 0C 18     DB    078H,0CCH,0CCH,07CH,00CH,018H,070H,000H ; 9 D_39
      70 00
01D0  00 30 30 00 00 30     DB    000H,030H,030H,000H,000H,030H,030H,000H ; : D_3A
      30 00
01D8  00 30 30 00 00 30     DB    000H,030H,030H,000H,000H,030H,030H,060H ; ; D_3B
      30 60
01E0  18 30 60 C0 60 30     DB    018H,030H,060H,0C0H,060H,030H,018H,000H ; < D_3C
      18 00
01E8  00 00 FC 00 00 FC     DB    000H,000H,0FCH,000H,000H,0FCH,000H,000H ; = D_3D
      00 00
01F0  60 30 18 0C 18 30     DB    060H,030H,018H,00CH,018H,030H,060H,000H ; > D_3E
      60 00
01F8  78 CC 0C 18 30 00     DB    078H,0CCH,00CH,018H,030H,000H,030H,000H ; ? D_3F
      30 00

0200  7C C6 DE DE DE C0     DB    07CH,0C6H,0DEH,0DEH,0DEH,0C0H,078H,000H ; @ D_40
      78 00
0208  30 78 CC CC FC CC     DB    030H,078H,0CCH,0CCH,0FCH,0CCH,0CCH,000H ; A D_41
      CC 00
0210  FC 66 66 7C 66 66     DB    0FCH,066H,066H,07CH,066H,066H,0FCH,000H ; B D_42
      FC 00
0218  3C 66 C0 C0 C0 66     DB    03CH,066H,0C0H,0C0H,0C0H,066H,03CH,000H ; C D_43
      3C 00
0220  F8 6C 66 66 66 6C     DB    0F8H,06CH,066H,066H,066H,06CH,0F8H,000H ; D D_44
      F8 00
0228  FE 62 68 78 68 62     DB    0FEH,062H,068H,078H,068H,062H,0FEH,000H ; E D_45
      FE 00
0230  FE 62 68 78 68 60     DB    0FEH,062H,068H,078H,068H,060H,0F0H,000H ; F D_46
      F0 00
0238  3C 66 C0 C0 CE 66     DB    03CH,066H,0C0H,0C0H,0CEH,066H,03EH,000H ; G D_47
      3E 00
0240  CC CC CC FC CC CC     DB    0CCH,0CCH,0CCH,0FCH,0CCH,0CCH,0CCH,000H ; H D_48
      CC 00
0248  78 30 30 30 30 30     DB    078H,030H,030H,030H,030H,030H,078H,000H ; I D_49
      78 00
0250  1E 0C 0C 0C CC CC     DB    01EH,00CH,00CH,00CH,0CCH,0CCH,078H,000H ; J D_4A
      78 00
0258  E6 66 6C 78 6C 66     DB    0E6H,066H,06CH,078H,06CH,066H,0E6H,000H ; K D_4B
      E6 00
0260  F0 60 60 60 62 66     DB    0F0H,060H,060H,060H,062H,066H,0FEH,000H ; L D_4C
      FE 00
0268  C6 EE FE FE D6 C6     DB    0C6H,0EEH,0FEH,0FEH,0D6H,0C6H,0C6H,000H ; M D_4D
      C6 00
0270  C6 E6 F6 DE CE C6     DB    0C6H,0E6H,0F6H,0DEH,0CEH,0C6H,0C6H,000H ; N D_4E
      C6 00
0278  38 6C C6 C6 C6 6C     DB    038H,06CH,0C6H,0C6H,0C6H,06CH,038H,000H ; O D_4F
      38 00

0280  FC 66 66 7C 60 60     DB    0FCH,066H,066H,07CH,060H,060H,0F0H,000H ; P D_50
      F0 00
0288  78 CC CC CC DC 78     DB    078H,0CCH,0CCH,0CCH,0DCH,078H,01CH,000H ; Q D_51
      1C 00
0290  FC 66 66 7C 6C 66     DB    0FCH,066H,066H,07CH,06CH,066H,0E6H,000H ; R D_52
      E6 00
0298  78 CC E0 70 1C CC     DB    078H,0CCH,0E0H,070H,01CH,0CCH,078H,000H ; S D_53
      78 00
02A0  FC B4 30 30 30 30     DB    0FCH,0B4H,030H,030H,030H,030H,078H,000H ; T D_54
      78 00
02A8  CC CC CC CC CC CC     DB    0CCH,0CCH,0CCH,0CCH,0CCH,0CCH,0FCH,000H ; U D_55
```

```
      FC 00
02B0  CC CC CC CC CC 78                 DB      0CCH,0CCH,0CCH,0CCH,0CCH,078H,030H,000H ; V D_56
      30 00
02B8  C6 C6 C6 D6 FE EE                 DB      0C6H,0C6H,0C6H,0D6H,0FEH,0EEH,0C6H,000H ; W D_57
      C6 00
02C0  C6 C6 6C 38 38 6C                 DB      0C6H,0C6H,06CH,038H,038H,06CH,0C6H,000H ; X D_58
      C6 00
02C8  CC CC CC 78 30 30                 DB      0CCH,0CCH,0CCH,078H,030H,030H,078H,000H ; Y D_59
      78 00
02D0  FE C6 8C 18 32 66                 DB      0FEH,0C6H,08CH,018H,032H,066H,0FEH,000H ; Z D_5A
      FE 00
02D8  78 60 60 60 60 60                 DB      078H,060H,060H,060H,060H,060H,078H,000H ; [ D_5B
      78 00
02E0  C0 60 30 18 0C 06                 DB      0C0H,060H,030H,018H,00CH,006H,002H,000H ; BACKSLASH D_5C
      02 00
02E8  78 18 18 18 18 18                 DB      078H,018H,018H,018H,018H,018H,078H,000H ; ] D_5D
      78 00
02F0  10 38 6C C6 00 00                 DB      010H,038H,06CH,0C6H,000H,000H,000H,000H ; CIRCUMFLEX D_5E
      00 00
02F8  00 00 00 00 00 00                 DB      000H,000H,000H,000H,000H,000H,000H,0FFH ; _ D_5F
      00 FF

0300  30 30 18 00 00 00                 DB      030H,030H,018H,000H,000H,000H,000H,000H ; ' D_60
      00 00
0308  00 00 78 0C 7C CC                 DB      000H,000H,078H,00CH,07CH,0CCH,076H,000H ; LOWER CASE A D_61
      76 00
0310  E0 60 60 7C 66 66                 DB      0E0H,060H,060H,07CH,066H,066H,0DCH,000H ; L.C. B D_62
      DC 00
0318  00 00 78 CC C0 CC                 DB      000H,000H,078H,0CCH,0C0H,0CCH,078H,000H ; L.C. C D_63
      78 00
0320  1C 0C 0C 7C CC CC                 DB      01CH,00CH,00CH,07CH,0CCH,0CCH,076H,000H ; L.C. D D_64
      76 00
0328  00 00 78 CC FC C0                 DB      000H,000H,078H,0CCH,0FCH,0C0H,078H,000H ; L.C. E D_65
      78 00
0330  38 6C 60 F0 60 60                 DB      038H,06CH,060H,0F0H,060H,060H,0F0H,000H ; L.C. F D_66
      F0 00
0338  00 00 76 CC CC 7C                 DB      000H,000H,076H,0CCH,0CCH,07CH,00CH,0F8H ; L.C. G D_67
      0C F8
0340  E0 60 6C 76 66 66                 DB      0E0H,060H,06CH,076H,066H,066H,0E6H,000H ; L.C. H D_68
      E6 00
0348  30 00 70 30 30 30                 DB      030H,000H,070H,030H,030H,030H,078H,000H ; L.C. I D_69
      78 00
0350  0C 00 0C 0C 0C CC                 DB      00CH,000H,00CH,00CH,00CH,0CCH,0CCH,078H ; L.C. J D_6A
      CC 78
0358  E0 60 66 6C 78 6C                 DB      0E0H,060H,066H,06CH,078H,06CH,0E6H,000H ; L.C. K D_6B
      E6 00
0360  70 30 30 30 30 30                 DB      070H,030H,030H,030H,030H,030H,078H,000H ; L.C. L D_6C
      78 00
0368  00 00 CC FE FE D6                 DB      000H,000H,0CCH,0FEH,0FEH,0D6H,0C6H,000H ; L.C. M D_6D
      C6 00
0370  00 00 F8 CC CC CC                 DB      000H,000H,0F8H,0CCH,0CCH,0CCH,0CCH,000H ; L.C. N D_6E
      CC 00
0378  00 00 78 CC CC CC                 DB      000H,C00H,078H,0CCH,0CCH,0CCH,078H,000H ; L.C. O D_6F
      78 00

0380  00 00 DC 66 66 7C                 DB      000H,000H,0DCH,066H,066H,07CH,060H,0F0H ; L.C. P D_70
      60 F0
0388  00 00 76 CC CC 7C                 DB      000H,000H,076H,0CCH,0CCH,07CH,00CH,01EH ; L.C. Q D_71
      0C 1E
0390  00 00 DC 76 66 60                 DB      000H,000H,0DCH,076H,066H,060H,0F0H,000H ; L.C. R D_72
      F0 00
0398  00 00 7C C0 78 0C                 DB      000H,000H,07CH,0C0H,078H,00CH,0F8H,000H ; L.C. S D_73
      F8 00
03A0  10 30 7C 30 30 34                 DB      010H,030H,07CH,030H,030H,034H,018H,000H ; L.C. T D_74
      18 00
03A8  00 00 CC CC CC CC                 DB      000H,000H,0CCH,0CCH,0CCH,0CCH,076H,000H ; L.C. U D_75
      76 00
03B0  00 00 CC CC CC 78                 DB      000H,000H,0CCH,0CCH,0CCH,078H,030H,000H ; L.C. V D_76
      30 00
03B8  00 00 C6 D6 FE FE                 DB      000H,000H,0C6H,0D6H,0FEH,0FEH,06CH,000H ; L.C. W D_77
      6C 00
03C0  00 00 C6 6C 38 6C                 DB      000H,000H,0C6H,06CH,038H,06CH,0C6H,000H ; L.C. X D_78
      C6 00
03C8  00 00 CC CC CC 7C                 DB      000H,000H,0CCH,0CCH,0CCH,07CH,00CH,0F8H ; L.C. Y D_79
      0C F8
03D0  00 00 FC 98 30 64                 DB      000H,000H,0FCH,098H,030H,064H,0FCH,000H ; L.C. Z D_7A
      FC 00
03D8  1C 30 30 E0 30 30                 DB      01CH,030H,030H,0E0H,030H,030H,01CH,000H ; L BRAK D_7B
      1C 00
03E0  18 18 18 00 18 18                 DB      018H,018H,018H,000H,018H,018H,018H,000H ; | D_7C
      18 00
03E8  E0 30 30 1C 30 30                 DB      0E0H,030H,030H,01CH,030H,030H,0E0H,000H ; R BRAK D_7D
      E0 00
03F0  76 DC 00 00 00 00                 DB      076H,0DCH,000H,000H,000H,000H,000H,000H ; TILDE D_7E
      00 00
03F8  00 10 38 6C C6 C6                 DB      000H,010H,038H,06CH,0C6H,0C6H,0FEH,000H ; DELTA D_7F
      FE 00

0400                           INT_1F_1         LABEL   BYTE

0400  78 CC C0 CC 78 18                 DB      078H,0CCH,0C0H,0CCH,078H,018H,00CH,078H ;  D_80
      0C 78
0408  00 CC 00 CC CC CC                 DB      000H,0CCH,000H,0CCH,0CCH,0CCH,07EH,000H ;  D_81
      7E 00
0410  1C 00 78 CC FC C0                 DB      01CH,000H,078H,0CCH,0FCH,0C0H,078H,000H ;  D_82
      78 00
0418  7E C3 3C 06 3E 66                 DB      07EH,0C3H,03CH,006H,03EH,066H,03FH,000H ;  D_83
      3F 00
0420  CC 00 78 0C 7C CC                 DB      0CCH,000H,078H,00CH,07CH,0CCH,07EH,000H ;  D_84
      7E 00
0428  E0 00 78 0C 7C CC                 DB      0E0H,000H,078H,00CH,07CH,0CCH,07EH,000H ;  D_85
      7E 00
0430  30 30 78 0C 7C CC                 DB      030H,030H,078H,00CH,07CH,0CCH,07EH,000H ;  D_86
      7E 00
0438  00 00 78 C0 C0 78                 DB      000H,000H,078H,0C0H,0C0H,078H,00CH,038H ;  D_87
      0C 38
0440  7E C3 3C 66 7E 60                 DB      07EH,0C3H,03CH,066H,07EH,060H,03CH,000H ;  D_88
      3C 00
0448  CC 00 78 CC FC C0                 DB      0CCH,000H,078H,0CCH,0FCH,0C0H,078H,000H ;  D_89
      78 00
0450  E0 00 78 CC FC C0                 DB      0E0H,000H,078H,0CCH,0FCH,0C0H,078H,000H ;  D_8A
      78 00
0458  CC 00 70 30 30 30                 DB      0CCH,000H,070H,030H,030H,030H,078H,000H ;  D_8B
      78 00
0460  7C C6 38 18 18 18                 DB      07CH,0C6H,038H,018H,018H,018H,03CH,000H ;  D_8C
      3C 00
0468  E0 00 70 30 30 30                 DB      0E0H,000H,070H,030H,030H,030H,078H,000H ;  D_8D
      78 00
0470  C6 38 6C C6 FE C6                 DB      0C6H,038H,06CH,0C6H,0FEH,0C6H,0C6H,000H ;  D_8E
      C6 00
0478  30 30 00 78 CC FC                 DB      030H,030H,000H,078H,0CCH,0FCH,0CCH,000H ;  D_8F
      CC 00

0480  1C 00 FC 60 78 60                 DB      01CH,000H,0FCH,060H,078H,060H,0FCH,000H ;  D_90
      FC 00
0488  00 00 7F 0C 7F CC                 DB      000H,000H,07FH,00CH,07FH,0CCH,07FH,000H ;  D_91
```

```
      7F 00
0490  3E 6C CC FE CC CC        DB    03EH,06CH,0CCH,0FEH,0CCH,0CCH,0CEH,000H ;  D_92
      CE 00
0498  78 CC 00 78 CC CC        DB    078H,0CCH,000H,078H,0CCH,0CCH,078H,000H ;  D_93
      78 00
04A0  00 CC 00 78 CC CC        DB    000H,0CCH,000H,078H,0CCH,0CCH,078H,000H ;  D_94
      78 00
04A8  00 E0 00 78 CC CC        DB    000H,0E0H,000H,078H,0CCH,0CCH,078H,000H ;  D_95
      78 00
04B0  78 CC 00 CC CC CC        DB    078H,0CCH,000H,0CCH,0CCH,0CCH,07EH,000H ;  D_96
      7E 00
04B8  00 E0 00 CC CC CC        DB    000H,0E0H,000H,0CCH,0CCH,0CCH,07EH,000H ;  D_97
      7E 00
04C0  00 CC 00 CC CC 7C        DB    000H,0CCH,000H,0CCH,0CCH,07CH,00CH,0F8H ;  D_98
      0C F8
04C8  C3 18 3C 66 66 3C        DB    0C3H,018H,03CH,066H,066H,03CH,018H,000H ;  D_99
      18 00
04D0  CC 00 CC CC CC CC        DB    0CCH,000H,0CCH,0CCH,0CCH,0CCH,078H,000H ;  D_9A
      78 00
04D8  18 18 7E C0 C0 7E        DB    018H,018H,07EH,0C0H,0C0H,07EH,018H,018H ;  D_9B
      18 18
04E0  38 6C 64 F0 60 E6        DB    038H,06CH,064H,0F0H,060H,0E6H,0FCH,000H ;  D_9C
      FC 00
04E8  CC CC 78 FC 30 FC        DB    0CCH,0CCH,078H,0FCH,030H,0FCH,030H,030H ;  D_9D
      30 30
04F0  F8 CC CC FA C6 CF        DB    0F8H,0CCH,0CCH,0FAH,0C6H,0CFH,0C6H,0C7H ;  D_9E
      C6 C7
04F8  0E 1B 18 3C 18 18        DB    00EH,01BH,018H,03CH,018H,018H,0D8H,070H ;  D_9F
      D8 70

0500  1C 00 78 0C 7C CC        DB    01CH,000H,078H,00CH,07CH,0CCH,07EH,000H ;  D_A0
      7E 00
0508  38 00 70 30 30 30        DB    038H,000H,070H,030H,030H,030H,078H,000H ;  D_A1
      78 00
0510  00 1C 00 78 CC CC        DB    000H,01CH,000H,078H,0CCH,0CCH,078H,000H ;  D_A2
      78 00
0518  00 1C 00 CC CC CC        DB    000H,01CH,000H,0CCH,0CCH,0CCH,07EH,000H ;  D_A3
      7E 00
0520  00 F8 00 F8 CC CC        DB    000H,0F8H,000H,0F8H,0CCH,0CCH,0CCH,000H ;  D_A4
      CC 00
0528  FC 00 CC EC FC DC        DB    0FCH,000H,0CCH,0ECH,0FCH,0DCH,0CCH,000H ;  D_A5
      CC 00
0530  3C 6C 6C 3E 00 7E        DB    03CH,06CH,06CH,03EH,000H,07EH,000H,000H ;  D_A6
      00 00
0538  38 6C 6C 38 00 7C        DB    038H,06CH,06CH,038H,000H,07CH,000H,000H ;  D_A7
      00 00
0540  30 00 30 60 C0 CC        DB    030H,000H,030H,060H,0C0H,0CCH,078H,000H ;  D_A8
      78 00
0548  00 00 00 FC C0 C0        DB    000H,000H,000H,0FCH,0C0H,0C0H,000H,000H ;  D_A9
      00 00
0550  00 00 00 FC 0C 0C        DB    000H,000H,000H,0FCH,00CH,00CH,000H,000H ;  D_AA
      00 00
0558  C3 C6 CC DE 33 66        DB    0C3H,0C6H,0CCH,0DEH,033H,066H,0CCH,00FH ;  D_AB
      CC 0F
0560  C3 C6 CC DB 37 6F        DB    0C3H,0C6H,0CCH,0DBH,037H,06FH,0CFH,003H ;  D_AC
      CF 03
0568  18 18 00 18 18 18        DB    018H,018H,000H,018H,018H,018H,018H,000H ;  D_AD
      18 00
0570  00 33 66 CC 66 33        DB    000H,033H,066H,0CCH,066H,033H,000H,000H ;  D_AE
      00 00
0578  00 CC 66 33 66 CC        DB    000H,0CCH,066H,033H,066H,0CCH,000H,000H ;  D_AF
      00 00

0580  22 88 22 88 22 88        DB    022H,088H,022H,088H,022H,088H,022H,088H ;  D_B0
      22 88
0588  55 AA 55 AA 55 AA        DB    055H,0AAH,055H,0AAH,055H,0AAH,055H,0AAH ;  D_B1
      55 AA
0590  DB 77 DB EE DB 77        DB    0DBH,077H,0DBH,0EEH,0DBH,077H,0DBH,0EEH ;  D_B2
      DB EE
0598  18 18 18 18 18 18        DB    018H,018H,018H,018H,018H,018H,018H,018H ;  D_B3
      18 18
05A0  18 18 18 18 F8 18        DB    018H,018H,018H,018H,0F8H,018H,018H,018H ;  D_B4
      18 18
05A8  18 18 F8 18 F8 18        DB    018H,018H,0F8H,018H,0F8H,018H,018H,018H ;  D_B5
      18 18
05B0  36 36 36 36 F6 36        DB    036H,036H,036H,036H,0F6H,036H,036H,036H ;  D_B6
      36 36
05B8  00 00 00 00 FE 36        DB    000H,000H,000H,000H,0FEH,036H,036H,036H ;  D_B7
      36 36
05C0  00 00 F8 18 F8 18        DB    000H,000H,0F8H,018H,0F8H,018H,018H,018H ;  D_B8
      18 18
05C8  36 36 F6 06 F6 36        DB    036H,036H,0F6H,006H,0F6H,036H,036H,036H ;  D_B9
      36 36
05D0  36 36 36 36 36 36        DB    036H,036H,036H,036H,036H,036H,036H,036H ;  D_BA
      36 36
05D8  00 00 FE 06 F6 36        DB    000H,000H,0FEH,006H,0F6H,036H,036H,036H ;  D_BB
      36 36
05E0  36 36 F6 06 FE 00        DB    036H,036H,0F6H,006H,0FEH,000H,000H,000H ;  D_BC
      00 00
05E8  36 36 36 36 FE 00        DB    036H,036H,036H,036H,0FEH,000H,000H,000H ;  D_BD
      00 00
05F0  18 18 F8 18 F8 00        DB    018H,018H,0F8H,018H,0F8H,000H,000H,000H ;  D_BE
      00 00
05F8  00 00 00 00 F8 18        DB    000H,000H,000H,000H,0F8H,018H,018H,018H ;  D_BF
      18 18

0600  18 18 18 18 1F 00        DB    018H,018H,018H,018H,01FH,000H,000H,000H ;  D_C0
      00 00
0608  18 18 18 18 FF 00        DB    018H,018H,018H,018H,0FFH,000H,000H,000H ;  D_C1
      00 00
0610  00 00 00 00 FF 18        DB    000H,000H,000H,000H,0FFH,018H,018H,018H ;  D_C2
      18 18
0618  18 18 18 18 1F 18        DB    018H,018H,018H,018H,01FH,018H,018H,018H ;  D_C3
      18 18
0620  00 00 00 00 FF 00        DB    000H,000H,000H,000H,0FFH,000H,000H,000H ;  D_C4
      00 00
0628  18 18 18 18 FF 18        DB    018H,018H,018H,018H,0FFH,018H,018H,018H ;  D_C5
      18 18
0630  18 18 1F 18 1F 18        DB    018H,018H,01FH,018H,01FH,018H,018H,018H ;  D_C6
      18 18
0638  36 36 36 36 37 36        DB    036H,036H,036H,036H,037H,036H,036H,036H ;  D_C7
      36 36
0640  36 36 37 30 3F 00        DB    036H,036H,037H,030H,03FH,000H,000H,000H ;  D_C8
      00 00
0648  00 00 3F 30 37 36        DB    000H,000H,03FH,030H,037H,036H,036H,036H ;  D_C9
      36 36
0650  36 36 F7 00 FF 00        DB    036H,036H,0F7H,000H,0FFH,000H,000H,000H ;  D_CA
      00 00
0658  00 00 FF 00 F7 36        DB    000H,000H,0FFH,000H,0F7H,036H,036H,036H ;  D_CB
      36 36
0660  36 36 37 30 37 36        DB    036H,036H,037H,030H,037H,036H,036H,036H ;  D_CC
      36 36
0668  00 00 FF 00 FF 00        DB    000H,000H,0FFH,000H,0FFH,000H,000H,000H ;  D_CD
      00 00
0670  36 36 F7 00 F7 36        DB    036H,036H,0F7H,000H,0F7H,036H,036H,036H ;  D_CE
      36 36
```

```
0678  18 18 FF 00 FF 00                  DB       018H,018H,0FFH,000H,0FFH,000H,000H,000H ;  D_CF
      00 00

0680  36 36 36 36 FF 00                  DB       036H,036H,036H,036H,0FFH,000H,000H,000H ;  D_D0
      00 00
0688  00 00 FF 00 FF 18                  DB       000H,000H,0FFH,000H,0FFH,018H,018H,018H ;  D_D1
      18 18
0690  00 00 00 00 FF 36                  DB       000H,000H,000H,000H,0FFH,036H,036H,036H ;  D_D2
      36 36
0698  36 36 36 36 3F 00                  DB       036H,036H,036H,036H,03FH,000H,000H,000H ;  D_D3
      00 00
06A0  18 18 1F 18 1F 00                  DB       018H,018H,01FH,018H,01FH,000H,000H,000H ;  D_D4
      00 00
06A8  00 00 1F 18 1F 18                  DB       000H,000H,01FH,018H,01FH,018H,018H,018H ;  D_D5
      18 18
06B0  00 00 00 00 3F 36                  DB       000H,000H,000H,000H,03FH,036H,036H,036H ;  D_D6
      36 36
06B8  36 36 36 36 FF 36                  DB       036H,036H,036H,036H,0FFH,036H,036H,036H ;  D_D7
      36 36
06C0  18 18 FF 18 FF 18                  DB       018H,018H,0FFH,018H,0FFH,018H,018H,018H ;  D_D8
      18 18
06C8  18 18 18 18 F8 00                  DB       018H,018H,018H,018H,0F8H,000H,000H,000H ;  D_D9
      00 00
06D0  00 00 00 00 1F 18                  DB       000H,000H,000H,000H,01FH,018H,018H,018H ;  D_DA
      18 18
06D8  FF FF FF FF FF FF                  DB       0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH ;  D_DB
      FF FF
06E0  00 00 00 00 FF FF                  DB       000H,000H,000H,000H,0FFH,0FFH,0FFH,0FFH ;  D_DC
      FF FF
06E8  F0 F0 F0 F0 F0 F0                  DB       0F0H,0F0H,0F0H,0F0H,0F0H,0F0H,0F0H,0F0H ;  D_DD
      F0 F0
06F0  0F 0F 0F 0F 0F 0F                  DB       00FH,00FH,00FH,00FH,00FH,00FH,00FH,00FH ;  D_DE
      0F 0F
06F8  FF FF FF FF 00 00                  DB       0FFH,0FFH,0FFH,0FFH,000H,000H,000H,000H ;  D_DF
      00 00

0700  00 00 76 DC C8 DC                  DB       000H,000H,076H,0DCH,0C8H,0DCH,076H,000H ;  D_E0
      76 00
0708  00 78 CC F8 CC F8                  DB       000H,078H,0CCH,0F8H,0CCH,0F8H,0C0H,0C0H ;  D_E1
      C0 C0
0710  00 FC CC C0 C0 C0                  DB       000H,0FCH,0CCH,0C0H,0C0H,0C0H,0C0H,000H ;  D_E2
      C0 00
0718  00 FE 6C 6C 6C 6C                  DB       000H,0FEH,06CH,06CH,06CH,06CH,06CH,000H ;  D_E3
      6C 00
0720  FC CC 60 30 60 CC                  DB       0FCH,0CCH,060H,030H,060H,0CCH,0FCH,000H ;  D_E4
      FC 00
0728  00 00 7E D8 D8 D8                  DB       000H,000H,07EH,0D8H,0D8H,0D8H,070H,000H ;  D_E5
      70 00
0730  00 66 66 66 66 7C                  DB       000H,066H,066H,066H,066H,07CH,060H,0C0H ;  D_E6
      60 C0
0738  00 76 DC 18 18 18                  DB       000H,076H,0DCH,018H,018H,018H,018H,000H ;  D_E7
      18 00
0740  FC 30 78 CC CC 78                  DB       0FCH,030H,078H,0CCH,0CCH,078H,030H,0FCH ;  D_E8
      30 FC
0748  38 6C C6 FE C6 6C                  DB       038H,06CH,0C6H,0FEH,0C6H,06CH,038H,000H ;  D_E9
      38 00
0750  38 6C C6 C6 6C 6C                  DB       038H,06CH,0C6H,0C6H,06CH,06CH,0EEH,000H ;  D_EA
      EE 00
0758  1C 30 18 7C CC CC                  DB       01CH,030H,018H,07CH,0CCH,0CCH,078H,000H ;  D_EB
      78 00
0760  00 00 7E DB DB 7E                  DB       000H,000H,07EH,0DBH,0DBH,07EH,000H,000H ;  D_EC
      00 00
0768  06 0C 7E DB DB 7E                  DB       006H,00CH,07EH,0DBH,0DBH,07EH,060H,0C0H ;  D_ED
      60 C0
0770  38 60 C0 F8 C0 60                  DB       038H,060H,0C0H,0F8H,0C0H,060H,038H,000H ;  D_EE
      38 00
0778  78 CC CC CC CC CC                  DB       078H,0CCH,0CCH,0CCH,0CCH,0CCH,0CCH,000H ;  D_EF
      CC 00

0780  00 FC 00 FC 00 FC                  DB       000H,0FCH,000H,0FCH,000H,0FCH,000H,000H ;  D_F0
      00 00
0788  30 30 FC 30 30 00                  DB       030H,030H,0FCH,030H,030H,000H,0FCH,000H ;  D_F1
      FC 00
0790  60 30 18 30 60 00                  DB       060H,030H,018H,030H,060H,000H,0FCH,000H ;  D_F2
      FC 00
0798  18 30 60 30 18 00                  DB       018H,030H,060H,030H,018H,000H,0FCH,000H ;  D_F3
      FC 00
07A0  0E 1B 1B 18 18 18                  DB       00EH,01BH,01BH,018H,018H,018H,018H,018H ;  D_F4
      18 18
07A8  18 18 18 18 18 D8                  DB       018H,018H,018H,018H,018H,0D8H,0D8H,070H ;  D_F5
      D8 70
07B0  30 30 00 FC 00 30                  DB       030H,030H,000H,0FCH,000H,030H,030H,000H ;  D_F6
      30 00
07B8  00 76 DC 00 76 DC                  DB       000H,076H,0DCH,000H,076H,0DCH,000H,000H ;  D_F7
      00 00
07C0  38 6C 6C 38 00 00                  DB       038H,06CH,06CH,038H,000H,000H,000H,000H ;  D_F8
      00 00
07C8  00 00 00 18 18 00                  DB       000H,000H,000H,018H,018H,000H,000H,000H ;  D_F9
      00 00
07D0  00 00 00 00 18 00                  DB       000H,000H,000H,000H,018H,000H,000H,000H ;  D_FA
      00 00
07D8  0F 0C 0C 0C EC 6C                  DB       00FH,00CH,00CH,00CH,0ECH,06CH,03CH,01CH ;  D_FB
      3C 1C
07E0  78 6C 6C 6C 6C 00                  DB       078H,06CH,06CH,06CH,06CH,000H,000H,000H ;  D_FC
      00 00
07E8  70 18 30 60 78 00                  DB       070H,018H,030H,060H,078H,000H,000H,000H ;  D_FD
      00 00
07F0  00 00 3C 3C 3C 3C                  DB       000H,000H,03CH,03CH,03CH,03CH,000H,000H ;  D_FE
      00 00
07F8  00 00 00 00 00 00                  DB       000H,000H,000H,000H,000H,000H,000H,000H ;  D_FF
      00 00
0800                            CODE     ENDS
                                         END


                                         PAGE,120
                                         SUBTTL  END ADDRESS
0000                            CODE     SEGMENT PUBLIC
                                PUBLIC   END_ADDRESS
0000                            END_ADDRESS      LABEL    BYTE
0000                            CODE     ENDS
                                         END
```

Index

A

Attribute Address Register 56
Attribute Controller
  description 3
  registers 56

B

BIOS
  description 4
  vectors with special meanings 103
BIOS listing 103
Bit Mask Register 54

C

character generator
  ROM 1
Character Map Select Register 21
Clocking Mode Register 19
Color Compare Register 48
Color Don't Care Register 53
color mapping 10
Color Plane Enable Register 60
compatibility issues 74
configuration switches 80
CRT Controller
  description 3
  registers 24
CRT Controller Address Register 24
CRT Controller Overflow Register 30
Cursor End Register 33
Cursor Location High Register 35
Cursor Location Low Register 35
Cursor Start Register 32

D

Data Rotate Register 49
direct drive connector 83
display buffer 4

E

Enable Set/Reset Register 47
End Horizontal Blanking Register 27
End Horizontal Retrace Register 29

End Vertical Blanking
  Register 40

F

feature connector 76
Feature Control Register 14

G

Graphics Controller
    description 3
    registers 45
Graphics 1 and 2 Address
  Register 46
Graphics 1 Position
  Register 45
Graphics 2 Position
  Register 46

H

Horizontal Display Enable End
  Register 26
Horizontal Pel Panning
  Register 60
Horizontal Total Register 25

I

Input Status Register One 15
Input Status Register Zero 14
Interface 76
    feature connector 76

L

Light Pen High Register 36
light pen interface 84
Light Pen Low Register 37
Line Compare Register 43

M

Map Mask Register 20
Maximum Scan Line
  Register 32
Memory Mode Register 23
Miscellaneous Output
  Register 12
Miscellaneous Register 52
Mode Control Register 41, 58
Mode Register 50
modes
    alphanumeric 8
    graphics 8
    IBM Color Display 5
    IBM Enhanced Color
      Display 6
    IBM Monochrome
      Display 6

O

Offset Register 38
Overscan Color Register 59

P

Palette Registers 57
Preset Row Scan Register 31
programming
 considerations 62
  compatibility issues 74
  creating a split screen 73
  creating a 512 character set 70
  creating an 80 by 43 alphanumeric mode 71
  programming registers 62
  RAM loadable character generator 69
  vertical interrupt feature 72

R

RAM loadable character generator 69
Read Map Select Register 50
registers
  Attribute Controller 56
  CRT Controller 24
  external 12
  Graphics Controller 45
  Sequencer 18
Reset Register 18

S

Sequencer
  description 3
  registers 18
Sequencer Address Register 18
Set/Reset Register 47
specifications 79
  configuration switch settings 81
  configuration switches 80
  direct drive connector 83
  light pen interface 84
  system board switches 79
Start Address High Register 34
Start Address Low Register 34
Start Horizontal Blanking Register 26
Start Horizontal Retrace Pulse Register 28
Start Vertical Blanking Register 39
support logic 4

U

Underline Location Register 39

V

Vertical Display Enable End Register 38
vertical interrupt feature 72
Vertical Retrace End Register 36
Vertical Retrace Start Register 36
Vertical Total Register 30