FID Fast Image Display for (.BMP & .PCX) Images

Rawaa Qasha *rawa_qasha@uomosul.edu.iq*

Ahmed S. Nori *ahmed.s.nori@uomosul.edu.iq*

Dept. of Computer Science
College of Computer Science and Mathematics
University of Mosul

Received on: 10/04/2002 **Accepted on: 12/10/2003**

ABSTRACT

Video display speed constitutes a very important factor in modern software performance. The best way to achieve fastest display is by accessing Video RAM and programming video card directly. In addition to the speed, this method provides flexibility and high performance video display operations. Besides that, dealing with 64K and 16.7 M color mode can be achieved only by this method.

Fast Image Display (FID) software is developed to display two popular types of images (BMP, PCX) using direct access to VRAM method with various SVGA modes differing in resolutions and number of colors. Assembly instructions and C++ language have been used to write Software parts.

Keywords: VRAM, BMP, PCX, SVGA.

عرض سريع للصور ل (.BMP و .PCX)

رواء قاشا **احمد سامي نوري**

كلية علوم الحاسوب والرياضيات / جامعة الموصل

تاريخ الاستلام: 2002/04/10 **تاريخ القبول: 2003/10/12**

الملخص

تشكل سرعة العرض الفيديوي عاملاً مهماً في أداء معظم التطبيقات والبرمجيات الحديثة، والطريقة المثلى لتوفير سرعة عالية في أثناء عملية العرض هي بالتعامل مع ذاكرة بطاقة الرسوميات وبرمجة المكونات المادية لهذه البطاقة بصورة مباشرة. فضلا عن السرعة، فإن هذه الطريقة تمتاز بالمرونة في السيطرة على عملية العرض والأداء العالي خلال التعامل مع بطاقة الرسوميات، فضلاً عن كونها الطريقة الوحيدة للتعامل مع الأطوار ذات64 ألف و 16 مليون لون.

نظامFID يوفر إمكانية عرض نوعين من ملفات الصور (BMP, PCX) بطريقة الوصول المباشر لذاكرة بطاقة الرسوميات لعدد من أطوار SVGA المختلفة الألوان والدقة حسب ما توفرها هذه الأطوار. وقد أُستخدمت لغة التجميع و C++ في بناء هذا البرنامج.
الكلمات المفتاحية: ذاكرة بطاقة الرسوميات، BMP، PCX، SVGA.

Introduction

As a general rule, the video display is created using either DOS or BIOS services that appropriate to the task in hand.

DOS and BIOS provide several functions for display manipulation, these functions are slow because the mechanism used to call these functions is slow and call BIOS interrupt takes much longer than a routine within a program. [1][7]

Besides that, BIOS supports capabilities to deal with at most 256 color modes, i.e graphics modes with more than 256 color cannot be managed by BIOS functions. While all modern video cards support graphics modes with high and true color with high resolution and required fast manipulation for graphics display. Direct manipulating video system offers fast and professional mechanism for graphics display and provides tools to deal with high color (64K color) and true color (16.7M color) graphics modes.

Direct Access Display method

Each video card contains an appropriate amount of memory called Video RAM (VRAM) that stores information related to each pixel appears on the screen. Direct access technique

accesses pixel's address in VRAM to change its information, this address can be calculated by means of its coordinates on the screen. For all applications to work on all PC's, the addresses used by each system must be the same. Therefore, to make individual pixel visible on the screen, the program must calculate the exact address of this pixel and pass the related color value to that address. [2][6]

Address calculation for a pixel depends on the number of bits required to represent each pixel and the organization of VRAM, which differs in each graphics mode. This method begins by setting an appropriate mode and calculates pixel address, then change its color.

Video RAM

Video cards offer different graphics modes with different resolutions, and various numbers of colors depending on the size of VRAM supported by the cards.

A standard VGA card provides 256K bytes of VRAM, while SVGA card and its extended modes requires more than this amount, because of the new graphics modes with high resolution up to 1600×1200 and Large color spectrum (32K, 64K, 16M color).[8]

VRAM is connected with the video card and the microprocessor. Microprocessor accesses VRAM as it can with

ordinary RAM, while display hardware takes values from this memory and displays them on the screen. [2][9]

The organization of VRAM varies as the graphics modes differ. In general, VRAM is accessed through segment A starting at A000:0000h when the video card operates in graphics modes, which requires only 64K bytes since, only segment A is available to the video card from the total addressable memory of PC., that means, 256K bytes of VRAM in standard VGA must addressed in 64K bytes area, so that VRAM is divided into four equal sections, each section is called a bitplanes. Note that, the four bitplanes are not occupying different address spaces in memory, instead, all four bitplanes occupy the same address space. They all start at address A000H. [1][3][11]

Latch Registers

Communication between the processor and VRAM is managed using four 8-bit registers known as the latch registers. There is one-latch register for each bitplane, see figure (1). [12]

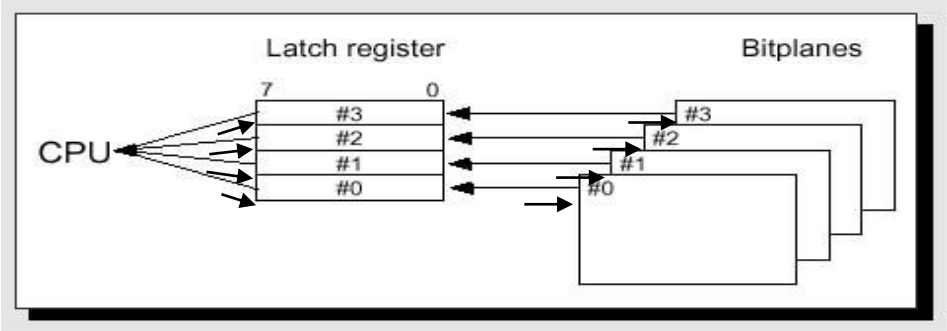


Figure (1): **Bitplanes and Latch register**

Organization of VRAM in 256 color modes

In these modes, each pixel needs a byte in the VRAM to store its information in one of the four bitplanes. Thus all pixels are spreaded on the four bitplanes as follows:

Each four consecutive pixels information are found in the same offset address from the bitplanes. For example, offset address (000h) in each bitplanes hold information related to the first four pixels in the screen starting at the upper left corner. First pixel in bitplane 0, seconds one in bitplane 1 and so on. The next four pixels are stored in offset address 0002h. In the same way, the remaining pixels are stored in the bitplanes as shown in figure (2):

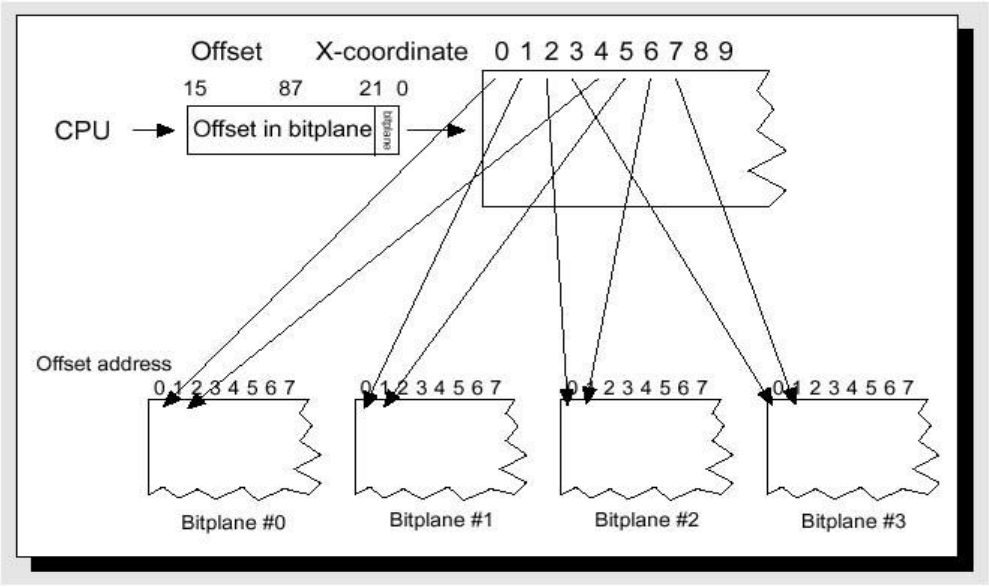


Figure (2): **Alternate 256 color mode as seen by the CRTC**

Accessing VRAM in 256-color mode

According to the organization described in the previous section, each bitplane is addressed individually, so all pixels are not available in the 64K byte at A0000h at the same time. Therefore accessing the address of a pixel with (x, y) as the horizontal and vertical coordinates respectively is calculated by multiplying y coordinates by (mode horizontal resolution /4), then the X coordinate is divided by 4, since there are four pixels located at the same address. The sum of these two calculations is the address in one of the four bitplanes, see table (1), mathematically [3][5]:

Pixel address (X, Y) = Y × (mode horizontal resolution /4) + X /4

The number of the bitplane that holds the desired pixel must be loaded to one of the sequencer controller registers (Map Mask register). This number is obtained from the two lowest bits of the X-coordinate.

Table (1): **256-color SVGA modes supported by FID software**

Mode No.	Graphic Resolution		No. of bits /pixel	Maximum memory
	Horizontal	Vertical		
101H	640	480	8	307,200
103H	800	600	8	480,000
105H	1024	768	8	786,432
107H	1280	1024	8	1,310,720

The following equations are used to access any pixel in the modes described in table 1:

- Offset = Y × (160) + int (X /4) for mode 101H
- Offset = Y × (200) + int (X /4) for mode 103H
- Offset = Y × (256) + int (X /4) for mode 105H
- Offset = Y × (320) + int (X /4) for mode 107H

High color SVGA modes

Each pixel in high color mode is represented by two bytes in VRAM. These two bytes are divided into three parts that corresponding to red, green, and blue intensities. the least five significant bits represents red intensity followed by six bits for green intensity, and five most significant bits for blue intensity. Organization of VRAM in 64K high color mode is depicted figure (3)

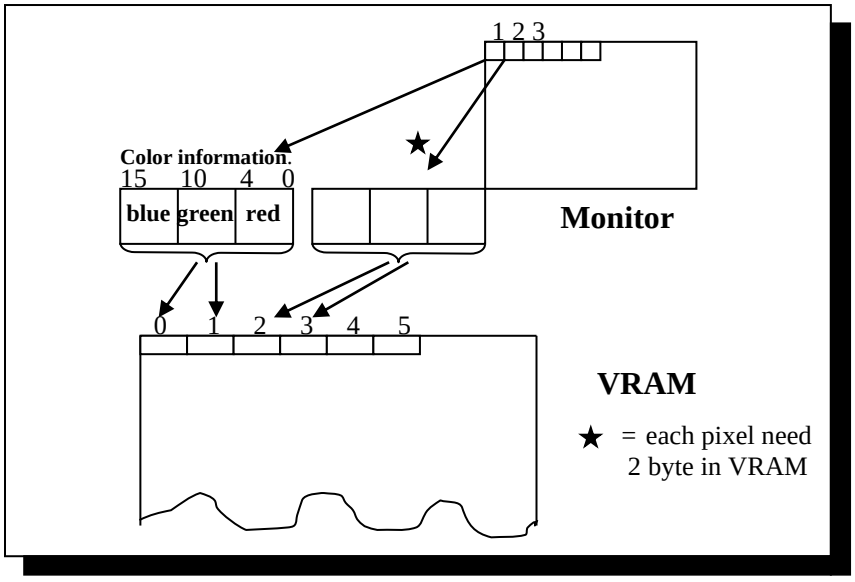


Figure (3): **Organization of VRAM in 64K-high color mode.**

Accessing VRAM in 64K-high color mode

Pixel information in high color mode is managed in the 64KB area in A segment. The two bytes that represent any pixel are existed in two consecutive address locations in the video memory, see table (2).

Address of the first byte for a pixel of X, Y coordinate is calculated using the following equation:

Pixel-address (X, Y) = Y×(mode horizontal resolution×2) + X×2

Offset address for the second byte can be obtained by incrementing first address by one.

Table (2): **High colors SVGA modes used by FID software**

Mode No.	Graphic Resolution		No. of bits /pixel	Maximum memory
	Horizontal	Vertical		
10EH	320	200	16	128 KB
111H	640	480	16	614,400
114H	800	600	16	960,000
117H	1024	768	16	1,572,864
11AH	1280	1024	16	2,621,440

To access any pixel in any of the five modes, the following equations are used for each mode respectively.

- Offset = Y × (320 × 2) + X × 2 for mode 10EH
- Offset = Y × (640 × 2) + X × 2 for mode 111H
- Offset = Y × (800 × 2) + X × 2 for mode 114H

- Offset = Y × (1024 × 2) + X × 2 for mode 117H
- Offset = Y × (1280 × 2) + X × 2 for mode 11AH

Organization of VRAM in True color modes

Depending on SVGA card system, pixels are represented by different no. of bytes,

(a) 24-bit true color mode assign 3 bytes for each pixel, one for red intensity, one for green intensity and other for blue intensity, as depicted

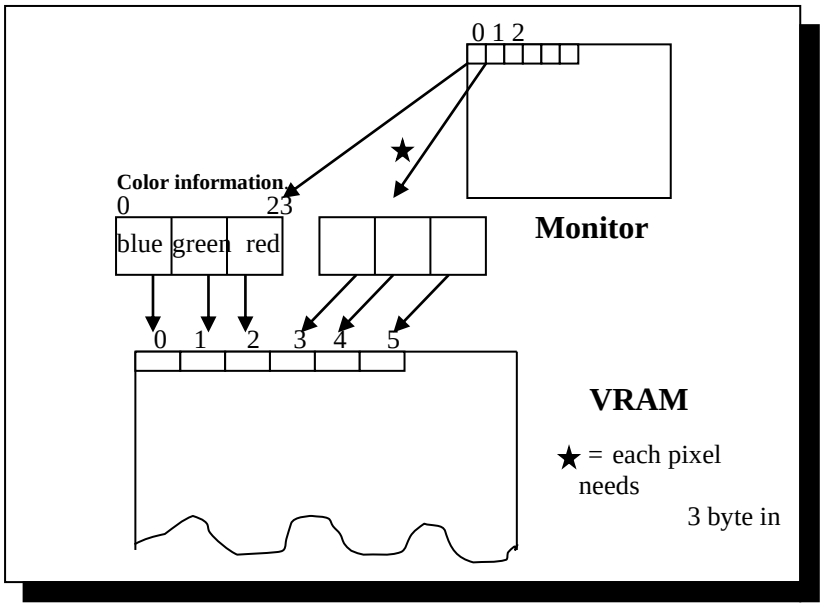


Figure (4): **Organization of VRAM in 24-True color mode.**

(b) in figure

(c) (4).

(d) 32-bit true color assigns 4 bytes for each pixel, three for red, green, blue intensities, and the last byte is added for future using, figure (5) depicts the organization of VRAM in 32-true color mode.

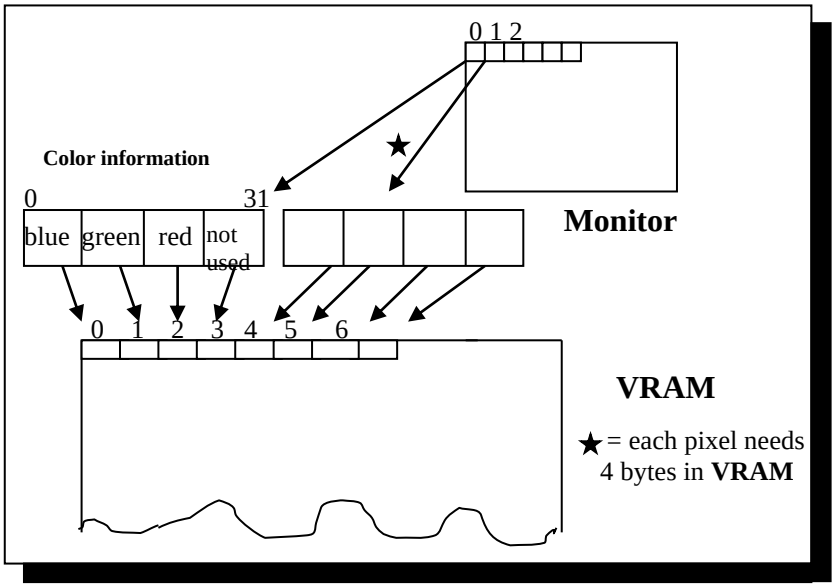


Figure (5): **Organization of VRAM in 32-True color mode.**

Accessing VRAM in True color modes

Here, there is no need for bitplanes, instead the information is managed in the 64K byte area in segment A.

If the video card support 24-bit true color, then the three bytes for each pixel are founded in three consecutive addresses in VRAM. So to change color information for any pixel with X & Y coordinates, calculate its first byte offset address by multiplying Y coordinate by scanline width (mode horizontal resolution × 3), and multiply the X-coordinate by 3. Then the sum of these two

calculations can be used to access the first byte, see table (3), mathematically [3][4][10]:

Pixel Offset (X, Y) = Y × (mode horizontal resolution × 3)+X× 3

The offset address for the other two bytes is obtained by incrementing the first address.

Also in 32-bit true color mode, the four bytes that representing any pixel are found in four consecutive addresses and the offset address for the first byte is calculated using the following equation:

Pixel Offset (X, Y) = Y×(mode horizontal resolution × 4) + X × 4

This offset address is incremented to obtain the other three bytes.

Table (3): **True colors SVGA modes accessed by FID software**

Mode No.	Graphic Resolution		No. of bits /pixel	Maximum memory
	Horizontal	Vertical		
10FH	320	200	32	256 KB
10FH	320	400	32	512 KB
112H	640	480	32	1,228,800
115H	800	600	32	1,920,000
118H	1024	768	32	3,145,728

To access any pixel in any of the four listed modes, the following equations are used for each mode respectively.

24-bit True color equations:

- Offset = Y × (320 × 3) + X × 3 for mode 10FH
- Offset = Y × (640 × 3) + X × 3 for mode 112H
- Offset = Y × (800 × 3) + X × 3 for mode 115H
- Offset = Y × (1024 × 3) + X × 3 for mode 118H

32-bit True color equations:

- Offset = Y × (320 × 4) + X × 4 for mode 10FH
- Offset = Y × (640 × 4) + X × 4 for mode 112H
- Offset = Y × (800 × 4) + X × 4 for mode 115H
- Offset = Y × (1024 × 3) + X × 4 for mode 118H

Accessing the extended VRAM in SVGA card

As mentioned earlier, SVAG cards contain large amount of VRAM to store the whole screen pixels in high resolution and true color modes. These modes need more than 256 KB of VRAM, so accessing the extended VRAM is accomplished by mapping portions of the video memory into the 64 K window at A000h. So that VRAM is divided into pages, which corresponding to the granularity of 64 K window.

Window granularity is defined as the number of bytes between the closest two bytes in VRAM that can be placed on any address in the 64 K window, it can take 1K, 2K, 4K … etc. To determine the granularity allowed by SVGA systems, VESA

BIOS function (01H) must be used, which returns information about a specific SVGA mode in a block structure.

Comparison (DOS, BIOS and Direct Access)

Its too difficult to decide with a specific manner the speed of video display by using each of (DOS & BIOS) services and the Direct Access way, because the results depends on the programming details and the system hardware components. But, as an average, the speed ratio of Direct Access to DOS is about 20:1 and to BIOS is about 10:1.

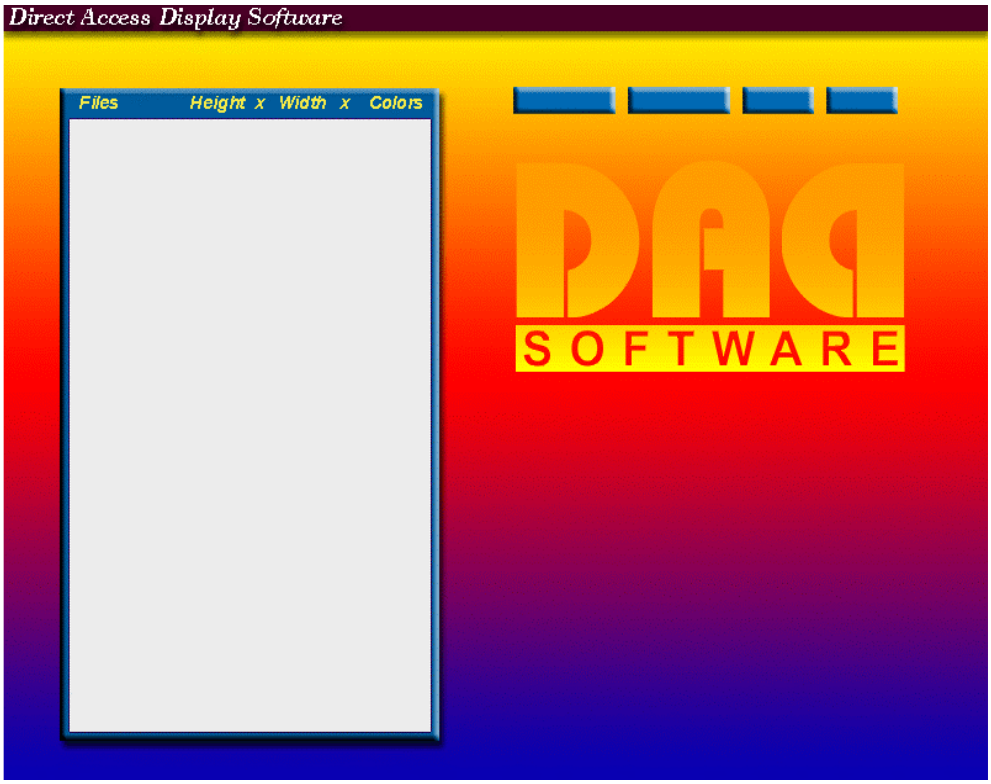
Besides, for the graphics display, BIOS provides dealing with graphics modes up to 256 colors, but for the (true & high) color graphics modes they can't be accessed only through the Direct Access way.

Conclusion

Direct access display is the fastest way for video display that supports powerful abilities to manipulate the extended memory associated with the video modes of SVGA cards by controlling the operations of video mode initialization and VRAM access.

These operations performed by writing directly to VRAM, manipulating video card controllers registers and directly programming the Palette and DAC registers. Direct access

abilities make it possible to deal with high and True color modes (24-bit, 32-bit) in high resolutions.



REFERENCES

1. Deltenei, Justin, 2000, Inverse Reality, http:// www.inversereality.org/tutorials/graphics%20programming/graphics programming.html
2. Deltenei, Justin, 2000, Inverse Reality, http:// www.inversereality.org/tutorials/graphics%20programming/plottingpixelsVRAM.html
3. Introduction to VESA Programming,

www.monstersoft.com/tutorial/1/vesa-intro.html

4. Jones, Chris, 1998, DOS Games Programming, http:// saturn.spaceports.com/~dosuser/dosgames.htm#programming%20SVGA.
5. Lafore, R., 1994, "Turbo C Programming for the PC", The Waite Group Inc.
6. Norton, P., 1986, "Inside the IBM PC", Brady Books. Inc.
7. Phoenix Technologies Ltd., 1989, "System BIOS for IBM PC/ XT/ AT Computers and Compatibles", Addison-Wesley
8. Rosch, L. W., 1997, "Hardware Bible", Premier Edition, Techmedia.
9. Sandler, C., 1998, "Fix Your Own PC", 4th Edition, M.I.S. Press.
10. Tischer, M. & Jennrich, B., 1996, "PC Intern", 6th Edition, Abacus.
11. Wilton, R., 1987, "Programmer's Guide to PC and PS/2", Microsoft Press.
12. Wood, W., June 1989, "The PC Graphics Maze", Electronics and wireless World, V. 6, N. 4.

Appendix -A-

FID software Review

The objective behind designing this software is to reach the high-speed display achieved by accessing VRAM directly through the displaying of several types of images, and provides an essential information about video display environment. FID offers the following capabilities:

1- Fast displaying BMP and PCX images with their information including image high and width, and its number of colors.
2- Four high-resolution modes to display 256 color image, five high color modes to display high color images and five true color modes to display true color images with various resolutions.
3- Displaying information when any mode has been selected includes:
   - Mode resolution.
   - Granularity of the VRAM window.
   - Number of bytes required by each pixel line in VRAM.
   - Number of bits per screen pixel.
4- Displaying information about the current VESA card and its capabilities such as:
   - Card signature

- VESA version
- Card manufacturer name.
- Listing the supported card modes.

The following keys can be used to interact with the interface:

Alt	+ Drive letter	to change to any drive.
	M	to display Mode info.
	C	to display Card info.
	H	to display Help list.
	A	to display About list.
up, down arrow		to select image file & mode.
ESC		to Exit the software.

Software main functions

FID software uses the following main functions to perform the essential operations to access VRAM and program different video card registers, which belongs to different controllers.

-Get VESA mode information

This function calls VESA function 01H to provide information about the selected VESA mode that will be stored in Mode structure.

Begin

Assigning VESA function 4F01H to AX register;

Load ES:DI registers with the address of Mode structure;

(*see appendix B(2)*)

Call INT 10H;

End.

Appendix -A-

-Get VESA card information

One of the VESA functions is used in this function to provide information related to the current card.

Begin

Assigning VESA function 4F00H to AX register;

Load ES:DI registers with the address of VESA block structure that can be used to hold VESA card information; (*see appendix B(1)*)

Call INT 10H;

End.

-Set pixel with 256 color mode

This function uses one of the equations described previously to access pixel address in 256 color modes depending on the selected mode.

Begin

Calculating offset address for pixel with X, Y coordinate

If offset address exceed 64 K byte then

Call VESA function 4FH, subfunction 05H through INT 10H to access next VRAM window;

Set Map Mask register in the sequencer controller to the content of least significant two bits of X coordinate to deal with proper bitplane;

Assign pixel color to its offset address;

End.

-Set pixel in high color mode

It's used to access pixel position in VRAM using any of the high color mode equation:

Begin

Calculating offset address using high color equation for the first pixel byte;

If offset address exceed 64 K Byte then

Call VESA subfunction 05H by using INT 10H to access next VRAM window

Assign a byte that contains five blue bits and green three most significant bits to the address

Increment address

Assign the remaining green bits with red bits to the second address

End.

-Set pixel in True color mode

Depending on true color mode type, this function uses either 32-bit true color equations or 24-bit equations to access pixel address.

Begin
If selected mode is 32-bit mode then
Calculating offset address using 32-bit equation for the first pixel byte;
else
using 24-bit equation to calculate first pixel byte address;
If offset address exceed 64 K Byte then
Call VESA subfunction 05H by using INT 10H to access next VRAM window
Assigning the three colors values to three consecutive bytes within VRAM;
End.

-Display BMP file using 256 color mode

FID uses this function to read and display BMP file using the above functions.

Begin
Open BMP file
Reading file header information
Reading image header information
Reading RGB color array
Setting selected mode
Setting DAC registers to RGB array
Repeat
Reading byte color information from the file;
Call set pixel with 256-color function;
Until end of image file
End.

-Display BMP file with True color mode

This function can be used to display BMP image in true color modes.

Begin
  Open BMP file
  Reading file header information
  Reading image header information
  Setting true color mode
  Repeat
    Read three bytes color information from the image file;
  Calling set pixel with true color mode function;
  Until end of file
End.

-Display PCX file in 256 color mode

FID uses this function to read, decode and display PCX image file.

Begin
  Open PCX file
  Reading file header information
  Reading RGB array from the file
  Setting 256 color mode
  Setting DAC registers
  Repeat
    Reading one byte color information from the left;
    Decoding color info
    Calling set pixel function with 256 color;
  Until end of image file

End.

Appendix -B-

(1)

SVGA mode structure

Ofs.	Contents	Type
00H	Mode flag,	1 word
02H	Flags for the first access window	1 byte
03H	Flags for the second access window	1 byte
04H	Granularity of the two access windows, in K,	1 word
06H	Size of the two access windows, in K	1 word
08H	Segment address of the first access window	1 word
0AH	Segment address of the second access window	1 word
0CH	FAR pointer to routine for setting the visible region in the two access windows	1 dword
10H	Number of bytes required by each pixel line in video RAM word	1 byte

(2)

VESA card information structure

Ofs.	Contents	Type	Ofs.	Contents	Type
00H	VESA Signature ("VESA")	4 BYTE	06H	FAR pointer to an ASCII string containing the name of the card manufacturer	1 DWORD
04H	VESA Version, main version number	1 BYTE	0AH	Flag that indicates the capabilities of the card currently not used, therefore 0000h	1 DWORD
05H	VESA Version, secondary version number	1 BYTE	0EH	FAR pointer to the list of code numbers for the supported video modes	1 DWORD