DOS Super VGA / VESA programming notes - by Myles

Introduction:

-------------

SuperVGA programming uses screen data that can easily exceed

1 MB in size. However, the normal VGA card only offers a 64k

memory access at A000h (some cards provide 128k).

How do you get to all that video memory ?

You know it's there - after all, your card documentation states

that you have e.g. 512k of memory on your video card, enough to

handle e.g. 800 x 600 x 256.

You may even have experimented and found that you can change to

this mode, but can only write pixels to a narrow band at the

top of the screen.(Surprise! That narrow band is 64k in size.)

(Except, of course, if it is 128k - read on.)

The answer is in a technique called bank switching, where you

use the 64k at A000h as a window onto your video card's memory.

The techniques and functions to do this vary from card to

card, because of historical reasons. Read on.

History:

--------

IBM defined MDA (monochrome), CGA, EGA, the MCGA (found in

some early PS/2 models, it never became a wide-spread

standard), and in 1987, the VGA (and the 8514/A).

VGA was backwards compatible with all the previous standards,

including the 320x200,256-colour MCGA mode. It also

introduced a new mode, 640x480 with 16 colours, which was

basically an extension of EGA's 640x350,16-colour mode, with

one major advantage - square pixels (so that if you plotted

the points for a circle, it looked like a circle, not an

ellipse). There was also a 640\*480 2-colour mode on the MCGA.

If you wanted higher resolutions you had to pay quite a bit

for IBM's professional level card, the 8514/A, a

high-resolution card capable of 256-colour modes up to

1024x768 (interlaced).

However, this card was proprietary - IBM didn't

release the register-level details.

Each of the other video card manufacturers then came up with

their own high resolution cards. However, they were all

implemented differently. 640x480 in 256 colours, for

example, may have been mode 34h on one manufacturer's card,

mode 56h on another's, and mode 62h on yet another. (N.B.

These are just false numbers to give you the idea.) Similarly

for 800x600, 1024x768, and other modes.

Also, at least one manufacturer implemented 128k banks.

Manufacturers also differed on whether banks (as a subsection

of the video card's memory) could start only 64k apart,

or 4k apart (i.e. possibly overlapping), or ....

You get the picture. Non-standard.

Programmers have to write routines to detect (or ask the

user:-p) which card is present, and then write card-specific

routines to handle the graphics routines for that card. If a

new card comes out, from a different manufacturer or even a

different model of card from the same manufacturer,

programmers have to write new routines to support that card,

which means that first they have to get details from the

manufacturer, and if they do it properly, they needed to find

a card to test their routines, (or a beta tester to do it for

them). You could provide these routines either in the body

of your code, or write external drivers or configuration

files. It is possible to write moderately generic code that

loads the specific card details from external files.

The manufacturers usually provide drivers for a few programs

(Autocad, Windows, etc.). Also, the information supplied

in the user manual that comes with the card is usually only the

mode numbers for that card, not the bank-switching code. Sigh.

You have to chase them up to get programming details.

Not surprisingly, most programs didn't support ALL the cards

available, and many programmers chose not to support any

SuperVGA cards. It wasn't worth the effort, and your

customers always ask when you are going to support THEIR

particular card.

The manufacturers were miffed enough by this that they

actually got together and formed the Video Electronics

Standards Association (VESA). They defined a new standard

programmer's interface so that programmers would only have to

write one set of graphics routines for SuperVGA. This was

the VESA standard. It has nothing to do with VESA Local Bus

(VLB), which is another standard from the same group about a

completely different hardware problem. (You think that's

confusing - VESA are coming/have come out with another VESA

standards for sound device interfacing, and more. And then

there's the Advanced Graphics Interface standard, which is

also different, I hear) The VESA SuperVGA standard is also

known as the VESA VBE standard (Video BIOS Extensions, or VGA

BIOS Extensions - depending on who you talk to - I think

the latter is correct, but I'm not sure).

The VESA VBE standard

---------------------

The VESA VBE standard implements video card routines through

an extension of the Interrupt 10h BIOS routines

(subfunction 4Fh).

Many video cards in existence implement VESA through the use

of a VESA VBE TSR (TSR = Terminate and Stay Resident

program, also known as a memory-resident program.), often

known as a VESA driver (not entirely accurate), although

newer video cards may implement the VESA VBE standard in

hardware. Let me re-iterate that this is an entirely

different matter from whether they are a VESA Local Bus (VLB)

card.

A VESA VBE driver should have been included on the disk

of drivers and utilities you got with your video card.

However, this, along with a number of VESA VBE drivers

available on Internet, may be out of date. Most VESA TSRs

are specific to a particular card.

If you don't have a VESA VBE driver that provides support

for version 1.2 of the VESA VBE standard, look for a

shareware VESA utility, last seen as UNIVBE51.ZIP, by Kendall

Bennett of SciTech Software. This is a shareware TSR that

provides VESA extensions for practically every SuperVGA card

in existence. Inexpensive personal registration. Licences

available for including it with your own programs.

Version 5.1 supports version 2.0 of the VESA standard.

The current version of the VESA VBE standard is 2.0 - you

should really try and get a driver which supports at least

version 1.2 , as there are a number of useful extra extras

beyond earlier versions (such as 32k/64k/16.7M colour modes)

VESA subfunctions (as of VESA VBE version 1.2) are:

subfunction 00 - get SuperVGA information

subfunction 01 - get SVGA mode information

subfunction 02 - set SuperVGA video mode

subfunction 03 - get current video mode

subfunction 04 - save/restore SuperVGA video state

subfunction 05 - bank switch

subfunction 06 - get/set logical scan line length

subfunction 07 - get/set display start

subfunction 08 - get/set DAC palette control

VESA VBE video modes (as of version 1.2)

100h : 640x400 256-colour

101h : 640x480 256-colour

102h : 800x600 16-colour

103h : 800x600 256-colour

104h : 1024x768 16-colour

105h : 1024x7686 256-colour

106h : 1280x1024 16-colour

107h : 1280x1024 256-colour

108h : 80x60 text

109h : 132x25 text

10Ah : 132x43 text

10Bh : 132x50 text

10Ch : 132x60 text

10Dh : 320x200 32k-colour (1:5:5:5)

10Eh : 320x200 64k-colour (5:6:5)

10Fh : 320x200 16.8M-colour (8:8:8)

110h : 640x480 32k-colour (1:5:5:5)

111h : 640x480 64k-colour (5:6:5)

112h : 640x480 16.8M-colour (8:8:8)

113h : 800x600 32k-colour (1:5:5:5)

114h : 800x600 64k-colour (5:6:5)

115h : 800x600 16.8M-colour (8:8:8)

116h : 1024x768 32k-colour (1:5:5:5)

117h : 1024x768 64k-colour (5:6:5)

118h : 1024x768 16.8M-colour (8:8:8)

119h : 1280x1024 32k-colour (1:5:5:5)

11Ah : 1280x1024 64k-colour (5:6:5)

11Bh : 1280x1024 16.8M-colour (8:8:8)

Version 2.0 of the VESA video standard has recently

become available, but I'm still gathering information

about it.

It covers extra features like protected mode, flat

memory video space (no banks), etc.

Stay tuned. In the meantime ...

-----------------------------------------------------------

That 64k of memory - what to put into it:

-----------------------------------------

For 256-colour modes, each byte in the 64k is simply a

palette value, or the colour number of a pixel (Yes,

256-colour graphics is just pixel painting by the numbers).

This means the color number is just an array index into a 256

member array of 6-bit red, green and blue values, giving you

256 colours out of 256k (2^(6\*6\*6)) possible values.

-------------------------------------------------------------

HiColor and TrueColor cards use a different RAMDAC chip (a

digital to analogue chip converting the digital values in

video memory to analogue output for the monitor).

HiColor and TrueColor modes represent their colours directly

- you specify the red-green-blue values for each pixel,

rather than choosing from a limited array of colours.

15-bit modes provide 32k colours - each colour is represented

as a two-byte value xrrrrrgggggbbbbb, where the x bit is

unused. 16-bit modes provide 64k colours, and provide an

extra bit for green (I think) - rrrrrggggggbbbbb. 24-bit

modes use three-byte values, one byte (8-bits) each for red,

green, and blue.

SuperVGA does not support the 32-bit and 64-bit modes found

on some specialist hardware, although you may sometimes find

32-bit (and rarely even 64-bit ?) files.

Some SuperVGA cards e.g. the S3-864, have pseudo-32-bit

modes, enabling the use of 32-bit instructions to move

data into video memory. The 4th byte is not used, giving you

a 24-bit mode with unused "gaps" in video memory, but increasing

the speed of transfer.

32-bit files provide another byte for an alpha value - which is

a transparency value used for overlaying one image on another -

also good for such things as anti-aliasing edges. 64-bit

files are like 32-bit, but provide 2 bytes each for red,

green, blue, and alpha.

-----------------------------------------------------------

A note on VESA and speed :

Some people have stated that programming using the VESA VBE

is s-l-o-w, apparently because they hear the word "BIOS"

and tune out, because in the past it has been emphasised that

using the BIOS to e.g. draw a pixel, is incredibly slow, compared

to writing your own routines to just move graphics data into

video memory.

However, apart from setting values and graphics mode, usually

at the beginning of your program, and getting information

pertaining to the mode you are using (a few BIOS calls at

most, done once only), you don't have to use the BIOS much at

all. Once you've set a bank, the VESA standard allows you to

write directly to video memory (well, a 64k subset of it),

indeed almost encourages it.

Also, you can use function 05 to change banks, but you'll find

the VESA VBE standard also includes a function returning the

direct address of the video card's bank changing routine, so

you can use it directly in a far call.

Alas, a number of video card manufacturers have not pointed this

address to a fast, optimized bank-changing routine, and a

considerable gain in speed can still be achieved by writing

card-specific drivers. \*Sigh\*

So VESA \*can\* be slower than a card-specific driver.

But not because it's using BIOS functions.

-----------------------------------------------------------

Warning: VESA is not a completely simple solution. You have

to find out whether the card it is being used on has 2

"windows", whether one or both windows are readable and/or

writeable, how big are the jumps by which you can move the

window around (granularity), how big the windows are, etc.

While these routines will only have to be written once, there

is a bit of work to be done at the start to make your VESA

routines generic to all VESA cards.

-----------------------------------------------------------

Information and code.

---------------------

\*\*\*\*\*\* N.B. I am not willing to

supply FTP sites for these files -

I suggest you either use ARCHIE, or find out HOW

to use archie - [news.answers](http://groups.google.com/group/news.answers) and [comp.answers](http://groups.google.com/group/comp.answers) may

be good places to start.

You may also find information there on FTP mail

servers, allowing you FTP services via email.

Requests for FTP sites will be deleted from my mail

without consideration - my time is, unfortunately, limited.

That said, I recommend the following FTP sites to get you

started (you may also have local mirrors of these sites) -

garbo.uwasa.fi: e.g. /pc/turbopas,/pc/graphics

oak.oakland.edu: e.g. /SimTel/msdos/graphics,/SimTel/msdos/turbopas

x2ftp.oulu.fi: /pub/msdos/programming

SVGABG55.ZIP - Jordan Hargrave's set of Super VGA BGI drivers.

If your card is not one of those catered for, it will also use

a VESA driver. Shareware, register for source. Uses the Graph

unit. This is just like using the Borland BGI. Good.

VESATP11.ZIP - shareware TPU , register for source.

On initialising a SuperVGA mode, you get a SuperVGA advert.

Otherwise much like the BGI interface, except you don't use the

Graph unit. Good.

I am reliably informed a new, improved version of this unit

will be out some time soon.

There is a graphics file viewer (VIEWER.ZIP ?) which uses

this unit (or it's predecessor, VESATP10.ZIP ?) for it's

graphics handling.

EGOF11-6.ZIP

EGOF11-7.ZIP - shareware, Turbo Pascal units. Mode X and VESA.

Haven't seen them myself yet.

VGADOC3.ZIP - includes card-specific information and code

(Turbo Pascal), also includes VESA information and code. Will

identify your card (including the DAC - 256 colour, 32k

colour, 16.7M colour), and will do a quick screen demo in any

available mode. Very extensive, freeware. Top quality.

Compiler/author, Finn Thoegersen.

VESASP12.ZIP - an unofficial version of the official VESA

standard, typed in by a friendly demo coder going under the

pseudonym of Patch. Includes ASM information on VESA

routines. I believe a version of this may also be found in

the PC Game Programmer's Encyclopedia, PCGPE\*.ZIP, at

x2ftp.oulu.fi, somewhere under pub/msdos/programming.

VESADRV2.ZIP - a collection of VESA drivers, not necessarily

implementing version 1.2 of the VESA standard. Some, at

least, are earlier.

VDRIV.ZIP - an even older collection of VESA drivers.

VESA24\_2.ZIP - C/ASM source for VESA usage.

Originally by Randy Buckland, with modifications by Don

Lewis.

VGAKIT52.ZIP - C ? More info ?

ALLSWAGS.ZIP

SWAG\*.ZIP - this collection of Turbo Pascal code snippets

and it's updates have some VESA routines included.

(I believe there is a C equivalent called something like

SNIPPETS, which may have VESA routines - I haven't seen it.)

UNIVBE51.ZIP - the universal VESA VBE driver. Better than

some hardware-specific drivers. Might well provide functions

not provided by VESA functionality in hardware. Shareware.

Regularly updated and improved. By Kendall Bennett,

from SciTech Software. Good.

MGL - MegaGraphics Library - for C/C++, also from SciTech

Software. Shareware.

SVGAKT51.ZIP - for C/C++, also from SciTech Software. Freeware ?

SciTech also released a freeware product called WinDirect

(maybe included in SVGAKIT51.ZIP ?) which enables you to

turn off the GDI in MS-Windows and run full-screen VESA

graphics, rather than using WinG in a window, while still

leaving other Windows functionality available.

Have a look in ftp.scitech.com, under devel/svgakt51.zip

INTER45?.ZIP - Ralf Brown's extensive interrupt list in

electronic form, includes VESA int 10h extensions

- latest version includes version 2.0 functions.

(Also available as RBNG45.ZIP - Norton Guides format

- a good DOS TSR reader for this format, apart from the

official commercial one, is NGHELP(10?).ZIP)

SVGACC23.ZIP - library for MS-compat. C/C++

SVGAPV23.ZIP - library for MS PDS and MS VBDOS

SVGAQB23.ZIP - library for MS QuickBASIC 4.x

More details on these and others as and when I have the time,

or somebody passes me the information.

-------------------------------------------------------------

Other sources I have seen/used:

------------------------------

N.B. Only the most recent of these cover up to version 1.2 of

the standard. I've not yet seen a printed version that

covers version 2.0.

"PC Interrupts", 2nd ed. by Ralf Brown and Jim Kyle.

Addison-Wesley, 1994. ISBN 0201624850

"Super VGA graphics programming secrets" by Steve Rimmer.

Windcrest/McGraw-Hill, 1993. ISBN 0-8306-4427-X (hbk) (C/ASM)

ISBN 0-8306-4428-8 (pbk)

"PC INTERN System Programming : the encyclopedia of DOS

programming know how" by Michael Tischer. Abacus, 1992. ISBN

1-55755-145-6 (C/TP/ASM)

"PC Magazine Turbo Pascal 6.0 : techniques and utilities", by

Neil J. Rubenking. Ziff-Davis Press, 1991. ISBN 1-56276-010-6

"Programmer's guide to the EGA and VGA cards", 2nd ed. by

Richard F. Ferraro. Addison-Wesley, 1990. ISBN

Program Now (UK programmer's magazine), September 1993,

p.60-64, Dave Bolton's Turbo Pascal programming column,

"Raising the VESA standard."

Dr Dobbs Journal, April 1990, p. 65H-70. "VESA VGA BIOS

extensions : a software standard for Super VGA" by Bo

Ericsson.

This one doesn't have VESA, but it's goes into plenty of

detail, with lots of code, on programming VGA and earlier

(ASM/C): "Programmer's guide to the PC & PS/2 video

subsytems" by Richard Wilton. Microsoft Press, 1987. ISBN

1-55615-103-9

The second edition, now out, has a brief section on VESA

"Programmer's guide to PC video subsystems", 2nd ed.,

by Richard Wilton. Microsoft Press, 1994. ISBN 1-55615-641-3.

------------------------------------------------------

The VESA standard itself is available from:

Video Electronics Standards Association

2150 North First Street, Suite 440

San Jose, California. 95131-2029

Phone (408) 435-0333

FAX (408) 435-8225

You may see an address of South Bascombe Avenue or some such

quoted in some sources. This is an old address, and mail is

no longer forwarded from this address, so don't use it.

When I last enquired, version 1.2 was still the standard.

Prices were as follows :

Cost: $20 to non-members, for VBE 1.2

$50 for the complete VESA Programmers Toolkit, which includes

VBE 1.2, programmers guidelines for direct color modes, SVPMI

1.0 SuperVGA protected mode interface, the VESA XGA extensions

standard, the standard for 800x600 mode (an older one ?), video

cursor interface and the VGA pass-through connector standard.

If you are not USA, add $20 international shipping charge.

Make sure you specify the VESA VBE 1.2 standard

(for VGA BIOS extensions), or the VESA Programmers Toolkit, as

VESA also has standards for the VESA Local Bus, an audio

interface, power management signalling, and others, all of

which are "VESA standards". They will FAX you an order form if

you give them your FAX number.

I guess by now you should probably specify version 2.0 8-).

-----------------------------------------------------------

Credits :

My thanks to those who have posted informative messages

on [comp.lang.pascal](http://groups.google.com/group/comp.lang.pascal) and [comp.graphics](http://groups.google.com/group/comp.graphics), and to the authors

and editors of the above packages and references.

Special thanks goes to Barry Naujok for his help,

corrections, information; to Kendall Bennett for his

occasional informative postings; and to Patch for typing

in the file in VESASP12.ZIP.

Any mistakes or errors are my own.