Video Graphics Array – VGA -

Abstract
This report is to give the reader an insight in how VGA works and background knowledge for the VGA project and practical training (lab).

Contents

1 Introduction

Video Graphics Array (**VGA**) was first marketed in 1987 by IBM. Since then it has been a well established standard, used in many applications. The term **VGA** is often used to refer to a resolution of 640×480. It can also mean the connector which is still widely used to carry analog video signals of all resolutions. All PC computer systems today must support **VGA** since it is the most basic form for showing graphics on a computer monitor. When you boot a computer for example, the bios tells the graphic card to set the monitor in **VGA** mode. No specific graphic card device driver should be needed. This is done because it is the simplest way to show a picture on screen and the most compatible mode. However when the operating system has been loaded it also loads a device driver for the graphics card. This allows for higher resolutions, faster refresh rates and more colors, if the monitor supports it. When this happens you are usually no longer in **VGA** mode.
Other modes are:
- Super Video Graphics Adapter (SVGA)
- Extended Graphics Array (XGA)
- Ultra Extended Graphics Array (UXGA).

It is not uncommon today (2008) that a monitor can support a resolution of 1600×1200 and 16.8 million colors. This is however only possible if the graphic card supports it.
I will not be discussing these other modes in any detail since they are very similar to the VGA mode.

VGA specifications
These features must be supported by the monitor and graphic card to be truly VGA compatible.

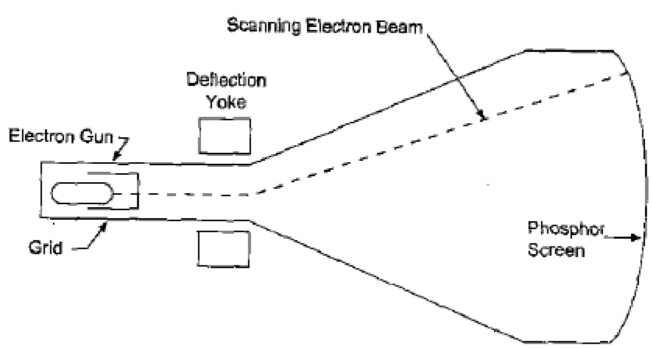
- 256 Kb Video RAM
- 16-color and 256-color modes

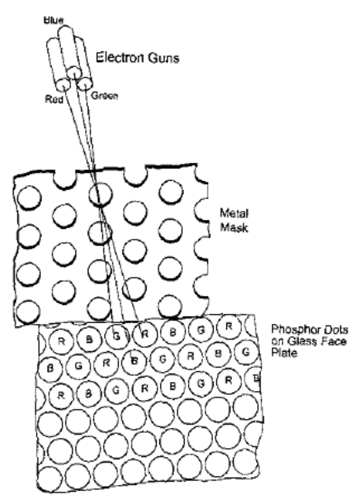
- 262144-value color palette (six bits each for red, green, and blue)
- Selectable 25 MHz or 28 MHz master clock
- Maximum of 720 horizontal pixels
- Maximum of 480 lines
- Refresh rates at up to 70 Hz
- Planar mode: up to 16 colors (4 bit planes)
- Packed-pixel mode: 256 colors (Mode 13h)
- Hardware smooth scrolling support
- Some "Raster Ops" support
- Barrel shifter
- Split screen support
- Soft fonts

Standard VGA graphics modes are

- 640×480 in 16 colors
- 640×350 in 16 colors
- 320×200 in 16 colors
- 320×200 in 256 colors (Mode 13h)

2 VGA Hardware and timing

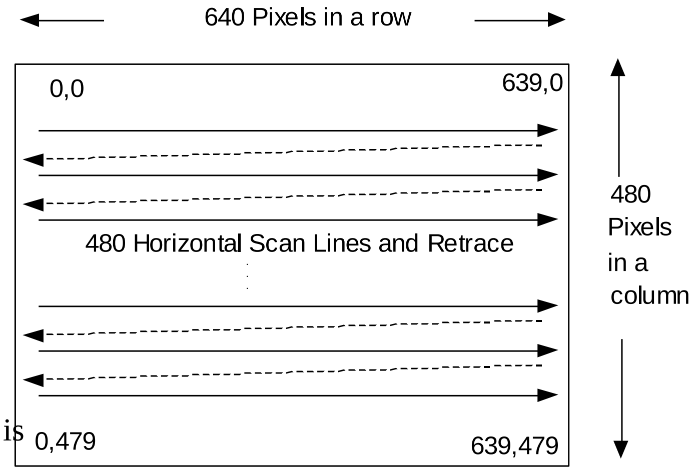




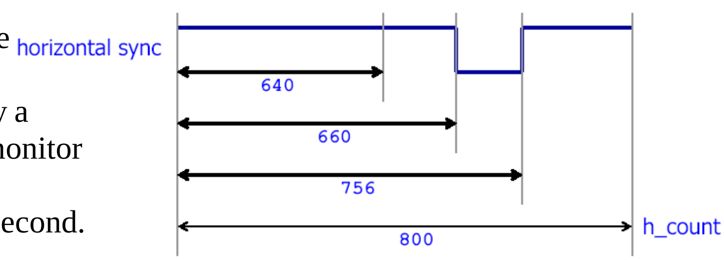
The main component inside the **VGA** monitor is a **Color CRT (Cathode Ray Tube)**

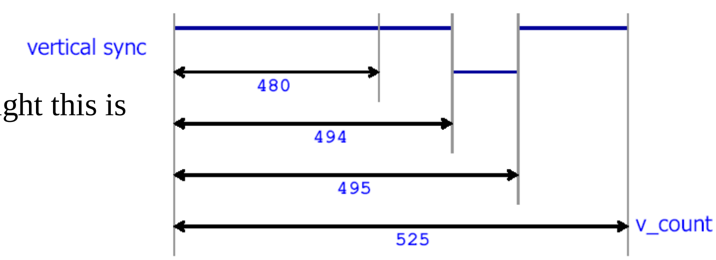
The picture you see on the PC monitor, are made up of electron-beams hitting a phosphor plate on the front of the screen. When the electrons hits the phosphor plate it lights up. The deflection yoke generates a magnetic or electrostatic field which pulls the electrons in different directions allowing it to "paint" the the screen in an orderly fashion. The electron beam is made up of three beams each controlling a color. They are red,

green and blue. On the phosphor plate there are small dots each gathered in a group of **red**, **green** and **blue**. This is actually different kinds of phosphor each lighting up with the corresponding colors. We can make all other colors from these primary colors by controlling the intensity on each of the beams. Since each group on the screen, or pixel, as we call it is very small, we can “paint” a very detailed picture using this method. The metal mask task is to assure that the beams only hit one pixel or **RGB** group at a time
I will now be discussing the way the VGA makes up a 640 x 480 resolution with a 60Hz refresh rate.



The refresh rate tells us how many times the picture is drawn on the screen. We need a refresh rate of at least 30Hz so the human eye will not experience flicker when watching the monitor.
As you can see on the picture to the right the electron beam goes from the left to the right painting each pixel as it passes by. When it reaches the end at the right it goes down one pixel vertical and goes all to the left end and starts painting horizontal again.
When the beam has gone 480 pixels down it goes all the way up to the top left corner and the process repeats. With a refresh rate of 60 Hz the entire screen is repainted 60 times per second.
At each end horizontal and vertical there is sync and retrace time depending on what refresh rate being used. So the time before reaching the left end after a horizontal or vertical sweep includes not only a retrace but a sync time as well. This is used so the monitor knows that the signal it receives is a correct one.
For 60 Hz it steps to 800 x 525 pixels 60 times per second.
This gives a pixel clock of about 25.2 MHz.
In the project you can use a 25MHz clock.
This means that the time to draw one pixel is about 39,7 ns (we use 40 ns). On the picture to the lower right this is illustrated. If we start when pixels are being painted horizontally we see that it paints 640 pixels first.
Then a hold time, to 660 pixels.
Then a sync time, from 660 to 756.
Then at the end, a new hold time from 756 to 800.
During the time from 640 to 800 pixels the electron beam should be off (black).
It is the same with the vertical timing only that the sync period happens when we have painted the entire screen vertical.

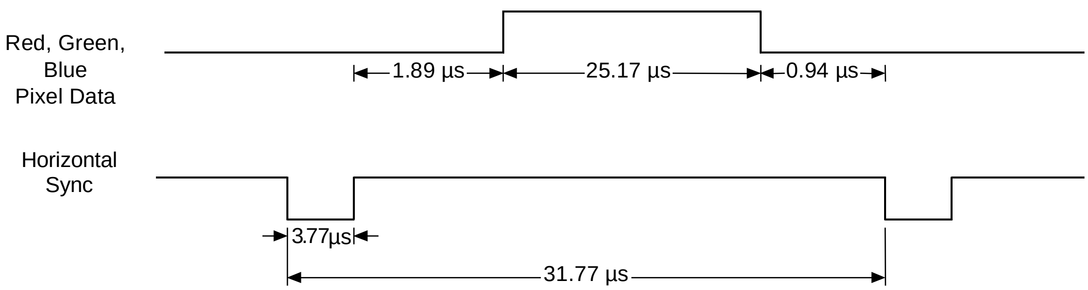




The two pictures below explain the same but here it is described with time instead of pixels.
We start with the horizontal sweep again.
This time we start from when a sync period starts.

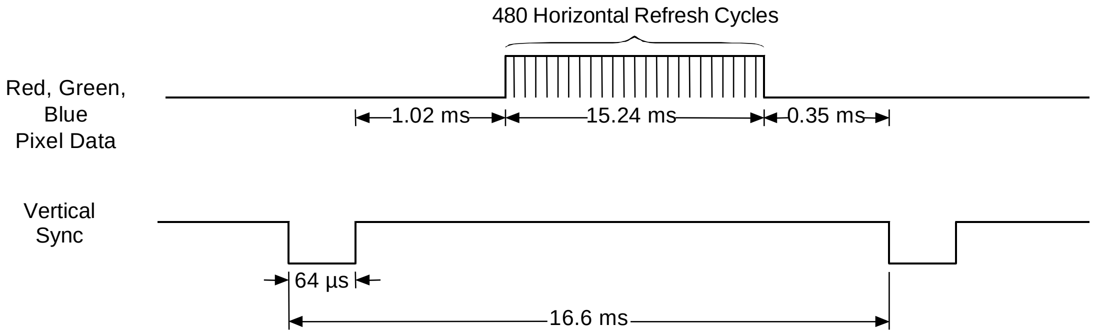
We see that horizontal sync lasts 3.77us. From the picture above it is supposed to last 756 – 660 = 96 pixels. If we multiply this with 39,7 ns. We get approximately 3.77us.

Horizontal



As you can se on the above picture the **RGB** pixel data is not painted before 1.89us after the sync period. Then it has 25,17us to write the 640 pixels of **RGB** data to screen. Each pixel lasting for about 39,7 ns. Then a new hold and retrace time of 0.94us before the entire process repeats it selves. The entire horizontal sweep lasts 31.77us.

Vertical



Now looking at the vertical sweep, we se that it is very similar to the horizontal sweep. There is a sync time and retrace time here as well. We know there is 480 horizontal sweeps per frame, each lasting 31.77us. If we multiply this with 480, we get 15.24ms as shown on the picture above. You can also see that **RGB** data is not displayed all the time during these 15.24ms. These as you know are made up of the retrace and sync time in the horizontal sweep.
In the project you can use the 25MHz clock, so the timing will be tiny different.

The **RGB** data is actually three analog voltages ranging from 0 to 0.7 Volt. Older displays like the **CGA [Color Graphics Adapter]** used digital signals to control the color. This might seem a bit odd that we now use analog signals, but it has been found to be the best way. Analog signals have by nature an endless resolution. However in any practical application there are limitations to what color resolution that can be achieved.

Modern graphic cards use a **RAMDAC**. This is a digital to analog converter used to generate the needed **RGB** signals.

Below are the common VGA connectors used and pin out table.

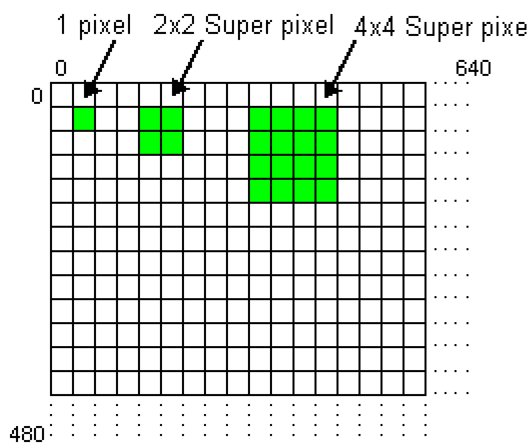
VGA Connector Pin Out

Pin #	15-Pin D	15-Pin D	----	9-Pin D	9-Pin D
---	**Name**	**Description**	----	**Name**	**Description**
1	RED Video	Red Video	----	RED Video	Red Video
2	GREEN Video	Green Video	----	GREEN Video	Green Video
3	BLUE Video	Blue Video	----	BLUE Video	Blue Video
4	ID2	Monitor ID, Bit #2	----	HSYNC	Horizontal Sync
5	GND	Ground	----	VSYNC	Vertical Sync
6	RGND	Red Ground	----	RGND	Red Ground
7	GGND	Green Ground	----	GGND	Green Ground
8	BGND	Blue Ground	----	BGND	Blue Ground
9	Key	No pin installed	----	SGND	Sync Ground
10	SGND	Sync Ground	----	----	----
11	ID0	Monitor ID Bit #0	----	----	----
12	ID1	Monitor ID Bit #1	----	----	----
13	HSYNC	Horizontal Sync	----	----	----
14	VSYNC	Vertical Sync	----	----	----
15	ID3	Monitor ID Bit #3	----	----	----

3 Memory requirements

This all depends on what color and screen resolution is required. Let us say for example, we only want black and white color with a 640 x 480 resolution. We would need on bit

for each pixel on screen. Since there is 8 bits in one byte we get 640 / 8 = 80 bytes horizontally and 480 x 80 = 34,8Kb bytes for the entire screen.
If 16 colors was the requirement we would need (log 16 / log 2) = 4 bits for each pixel. This gives us (640 * 4) / 8 = 320 bytes horizontally and 480 * 320 = 153,6Kb for the entire screen.
We see that having more color resolution creates a much bigger memory requirement. There are methods to avoid this growing memory requirement. One way is to compress the color information in ram and decompress on the fly when it is needed. Another way is to define a so called **super pixel**. A **super pixel** means that we group a square set of pixels and assign one memory location to this group. This can be done by modifying the way pixel row clock and pixel column clock accesses the memory. Using this method we gain greater color resolution at the expense of screen resolution.



These other methods need a more sophisticated hardware or software but it can be very useful if a low memory requirement is needed.
It might seem that we only can update the **VGA** memory when we are not painting pixels, since this is the only time the memory is idle. However there is often used a memory type called **dual port RAM**. This memory has two data address inputs, one data input and one data output. One off the data address inputs are used for writing and the other for reading. This simplifies the timing required to store new video data and is more effective since it can store more data during a frame. This is the most common approach used in modern graphics cards. New graphic cards aimed at computer gaming can easily have 512Mb or more onboard memory. However this memory is not only used to display pixels on screen. Among other things is used to store and modify textures used trough out the computer game. Modern cards also have a CPU (Central processing unit) often called GPU (Graphical processing unit). The reason for this is to offload the computers CPU so it can do other important tasks.

4 Colors

By putting together different levels of the colors Red, Blue and Green we can produce different colors. Below you can see the VGA 256-color map. These 256 colors can be presented in VGA by changing the value of the three RGB-colors.

References

[1] J. D. Neal (1997). VGA Chipset Reference. *Hardware Level VGA and SVGA Video Programming Information Page (http://osdever.net./FreeVGA/vga/vga.htm)*

[2] James O. Hamblen and Michael D. Furman (2001). Rapid prototyping of digital systems. second edition

[3] Eduardo Sanchez. Ecole Polytechnique Fédérale de Lausanne. Paper on **VGA**. Picture reference.

[4] Leroy Davis http://www.interfacebus.com/
Pin out for the **VGA** connector

[5] Lennart Lindh and Zoltan Nagy. On-chip VGA suggestion for low density FPGA`s

Bilaga: En kort sammanfattning

VGA är en förkortning av Video Graphics Array och är en standard för hur bilden överförs från en grafikkrets till en bildskärm. De delarna av protokollet som är viktiga för den här konstruktionen är
följande:

- 640 (max 800) pixlar bred.
- 480 (max 600) linjer hög.
- Linjefrekvens 31,4686kHz.
- Tre signaler som tillsammans ger färgen för varje pixel, R, G och B.
- Två signaler för vertikal och horisontell synkroniseringspuls. HS och VS.

Med 800 pixlar i varje linje och frekvensen 31,4686kHz får vi en pixelfrekvens på 25,17488MHz. Vi kommer att använda 25Mhz eftersom det är hälften av DE2 kortets (om du inte använder ett annat FPGA kort) inbyggda klocka på 50MHz och ligger tillräckligt nära för att fungera. I varje linje så måste timing mellan signalerna komma i en bestämd ordning se nästa figur.

640pixel bred (25,17µS) 480 linjer hög. (15,25ms) Synlig del av bilden. Under denna del är R,G och B aktiva och skapar tillsammans färgen på pixlarna. HS och VS är ”1” dvs. höga. Enligt VGA protokollets specifikation skall 640 pixlar ta 25,17µS. I vårat fall med en pixellocka på 25MHz tar det 25,6µS och det är tillräckligt nära för att fungera.	20 pixlar (0,94µS)	96 pixlar (3,77µs)	44 pixlar (1,89µS)
14 linjer. (0,35ms) Den här delen av bilden kallas ”front porch”. Här är R,G och B inaktiva alltså ”0”. HS och VS är fortfarande ”1”.			
1 linje. (0,06ms) I den här delen är HS och VS låga eller ”0”. R, G och B är ”0”.			
30 linjer. (1,02ms) Den här delen av bilden kallas ”back porch”. Här är R,G och B inaktiva alltså ”0”. HS och VS är ”1”.			

Figur 1: En översikt av hela protokollet