International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 14 (2018) pp. 11572-11576

VGA Configuration Algorithm using VHDL

[1]Christian Plaza, [2]Olga Ramos, [3]Dario Amaya

Virtual Applications Group-GAV, Nueva Granada Military University –UMNG
Bogotá, Colombia.

Abstract

Nowadays it is important to visualize information, with the purpose of generate real-time data about a device or a process to a user, this would help to identify problems and generate solutions in a simpler way. A VGA controller can work with different types of resolution; this can be used to create an incredible amount of characters, which can be useful to visualize a lot of information in an easier way, a VGA controller generates synchronization signals produce on a FPGA (programed on VHDL language), this allows the correct visualization on a VGA monitor. Some of the most significant factors that can affect the frequency of synchronisms are resolution and the refresh frequency, A VGA controller can be used in many applications due to the high processing speed of the FPGA, it is perfect for color recognition.

Keywords: image processing, image visualization, synchronism, VGA port.

INTRODUCTION

Nowadays there are different kinds of electronics devices that allows the visualization and interaction of information with the user, in a direct way through an electronic device, this can make real time modifications depending on the current input values that receive the device. One of the most used system is a monitor with a VGA input. [1]

FPGA: (Field Programmable Gate Array) is a reprogrammable electronic device that allow the parallel programing, the response time it is really fast, it is an economic alternative compared to other elements because it is reprogrammable. [2]

VGA Monitor: is a device that enable the image visualization through a VGA port, these has different resolution, and works by shifting an electron beam through the screen for viewing images. [3]

The VGA Port (*Video Graphics Array),* is a graphic card that allows to visualizer image on a monitor, through an electronic device that generate sync signals, this signals can be change to be compatible with different monitors and resolutions. [4]

In a VGA controller it is really important to consider two differences kinds of signals, the horizontal and vertical synchronization signals. This allows to visualize an image through the monitor, the values of frequency depend on the kind of resolution that is required and the update frequency, the last one indicates the number of times that a monitor is going to repaint in one second, the standard value is 60 Hz. [5]

There are different types of resolution, also there are different update frequency (this one had normalize values). For each one of these there are different values for the pixel clock, front porch, back porch and sync pulse. Besides the sync polarity changes depending on the current VGA controller, in some cases they are going to be positive and other negatives. [6]

The first thing that is required for the image visualization is the vertical and horizontal sync, this are composed by four types of elements: the first one is the Back Porch, this one indicates the beginning of the sync, next it precedes by the Sync Pulse this generate an inversion of polarity in the synchronism, after this continues the Front Porch returning to the initial value of the polarity and finally the number of pixel in the display. [7]

When the Front Porch, the Sync Pulse and the back porch are running it is known as the Blanking Time and the most significant characteristic is that in all that time the monitor it is not going to generate any kind of image, when the count arrive the number of pixels of the display, the visualization of images with the respective RGB values takes place on the monitor. [8]

When the image visualization is taking place the FPGA send data in a digital way there correspond to the value of the color, also in the monitor, there are three different types of analog signals that create the color, with a maximum value of 256 colors. The color intensity depend of the voltage value from the analog signal, most of the FPGA had a digital analog converser to realize this kind of conversion. [9]

The main goal of these project is generate a VGA controller using a FPGA device for the color recognition, using a CMOS sensor as an input, the FPGA device can send the desired information through the VGA port, so this can be visualize on a monitor.[10]

METHODS AND MATERIALS

For the project which main object is the color identification on a VGA monitor. Software and Hardware were selected to have a decent efficiency during the project development.

Image visualization on a VGA

In the Figure 1 it can be seen the FPGA and the VGA monitor connections.

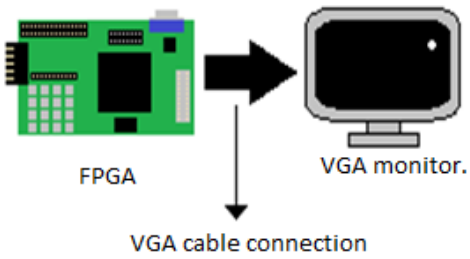

Figure 1. Connection diagram FPGA VGA.

There are two important design stages on these project, which will be described below. The first one consist on the creation of a clock that allow the count of the pixels, a controller will be implemented (this contains the horizontal and vertical sync signals) for the required resolution, and the other one visualize an image on a determined pixel.[11]

FPGA block diagram

The figure 2 show a block representation of the FPGA programming.

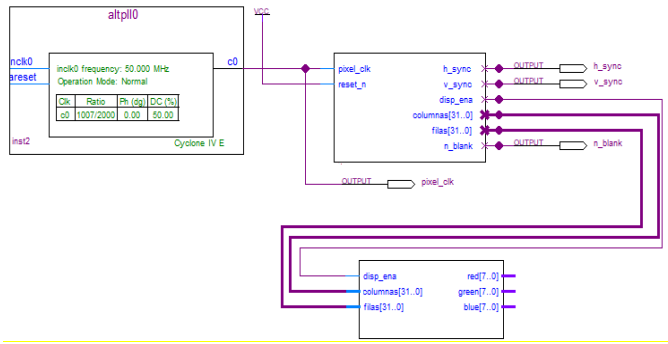

Figure 2. FPGA block program.

Flowchart

In the figure 3 the flowchart of the developed program will be presented.

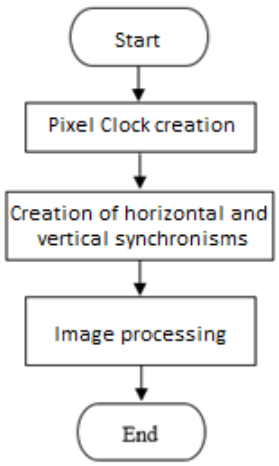

Figure 3. Flowchart

Implemented code

For the creation of the VGA controller the code on the Figure 4 was implemented.

```
ARCHITECTURE behavior OF controlador_vga IS
   CONSTANT h_total  :  INTEGER := 800;
   CONSTANT v_total  :  INTEGER := 525;
BEGIN
```

Figure 4. VGA controller, initialization of variables

The first thing to determinate are the maximum values for the vertical and horizontal sync signals, those values can change depending on the resolution and the update frequency, for these project it will be use a 640x480 resolution working with 60 Hz obtaining the following values described in the [12]

Table 1. Different values for a 640x480 resolution using 60 Hz.

Characteristics	Horizontal	Vertical
Display	640	480
Front Porch	16	10
Sync Pulse	96	2
Back Porch	48	33
Polarity synchronism	Negative	Negative
Total	800	525

The sum of the Display, Front Porch, Sync Pulse, Back Porch, it will give as a result the total period of the duration for the sync pulse, using the polarity of the synchronism will perform a state change in the signal so that the controller works correctly figure 5.

```
IF(cont_hor < h_total - 1) THEN
   cont_hor := cont_hor + 1;
ELSE
   cont_hor := 0;
   IF(cont_ver < v_total - 1) THEN
      cont_ver := cont_ver + 1;
   ELSE
      cont_ver := 0;
   END IF;
END IF;
```

Figure 5. VGA controller, vertical and horizontal counter.

Obtained the total period value of the vertical and horizontal sync signals, a sum conditional was implemented, while the counter is less from the total period value of the signal a unit value will be added to it, but when the value is more than the total value the counter will be reset Figure 6.

```
IF(cont_hor < 656) THEN
   h_sync <= '1';
ELSE
   h_sync <= '0';
END IF;

IF(cont_hor >752) THEN
   h_sync <= '1';
ELSE
   h_sync <= '0';
END IF;

IF(cont_ver < 490 ) THEN
   v_sync <= '1';
ELSE
   v_sync <= '0';
END IF;

IF(cont_ver>492) THEN
   v_sync <= '1';
ELSE
   v_sync <= '0';
END IF;
```

Figure 6. VGA controller, synchronism pulse activation

With the pixel counter it can be determinate in which time the horizontal and vertical sync pulse should be activated (using the parameters from the table 1) the horizontal sync signal will be activated on this pixel range [656 – 752], for the vertical sync signal the following range [490 – 492], for both cases the signal was on a logic state '1', will be change for a '0' while the pixel counter remain on the previous ranges Figure 7.[13]

```
IF(cont_hor < 640) THEN
   columnas <= cont_hor;
END IF;
IF(cont_ver < 480) THEN
   filas <= cont_ver;
END IF;
```

Figure 7. VGA controller, rows and columns assignment

A new variable is created for maintain the current position of the pixel with respect to the resolution implemented on the controller Figure 8.

```
IF(cont_hor < 640 AND cont_ver < 480) THEN
   disp_ena <= '1';
ELSE
   disp_ena <= '0';
END IF;
```

Figure 8. VGA controller, enable activation.

Finally a digital flag was implement to restrict the image visualization on the VGA port while the pixel counter was on the range [0 – 640] (horizontal counter), and [0 – 480] (vertical counter).

ANALYSIS OF RESULTS

In the current part, the different results obtained on the image processing will be explain.

The result show on the Figure 9, is the implementation and visualization of a square on the VGA monitor, the beginning of coordinates is located on the top left corner of the monitor, a square will be draw while the pixel counter is in a certain coordinates (the pixel will have a violet color), if the counter is outside the range the pixel will be set up as green. [14]

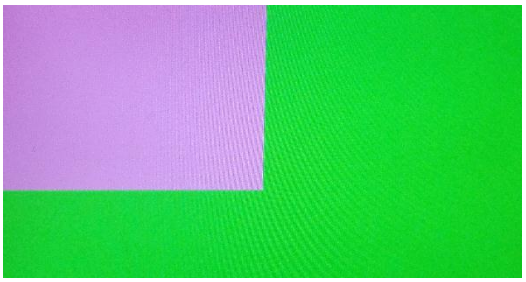

Figure 9. Final result

Knowing the current position of the pixel can provide different conditions, this can be used to generate different amounts of geometrics figures, in this particular case a square, the OTHERS indicates that the others bits will have the same value as the one indicated on the function.

```
PROCESS(disp_ena, filas, columna)
BEGIN

   IF(disp_ena = '1') THEN
      IF(filas < 250 AND columna < 250) THEN
         rojo <= (OTHERS => '1');
         verde <= (OTHERS => '0');
         azul <= (OTHERS => '1');
      ELSE
         rojo <= (OTHERS => '0');
         verde <= (OTHERS => '1');
         azul <= (OTHERS => '0');
      END IF;
   ELSE
      rojo <= (OTHERS => '0');
      verde <= (OTHERS => '0');
      azul <= (OTHERS => '0');
   END IF;

END PROCESS;
```

Figure 10. Code Test 1

The second test is similar to the before, the changes made were the coordinate of the square and the color that are going to be used, in this particular case the color yellow (mix between red and blue) and cyan were used Figure 11. [15]

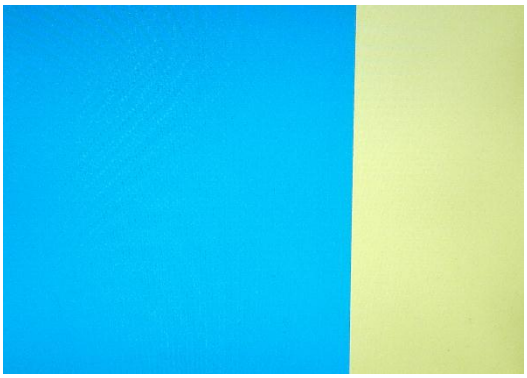

Figure 11. Test 2, result.

If the pixel clock is outside the range the RGB output will be set up as 0, as it can be seen on the figure 12.

```
PROCESS(disp_ena, filas, columna)
BEGIN

   IF(disp_ena = '1') THEN
      IF(filas < 500 AND columna < 480) THEN
         rojo <= (OTHERS => '0');
         verde <= (OTHERS => '1');
         azul <= (OTHERS => '1');
      ELSE
         rojo <= (OTHERS => '1');
         verde <= (OTHERS => '1');
         azul <= (OTHERS => '0');
      END IF;
   ELSE
      rojo <= (OTHERS => '0');
      verde <= (OTHERS => '0');
      azul <= (OTHERS => '0');
   END IF;

END PROCESS;
```

Fig. 12 Code test 2

On the third test a sine signal simulation was proposed, for this a design of circle compose of 37 pixels was created, the current pattern is going to localize on different coordinates, also an axis allow the guidance of these points on the VGA monitor. Figure 13. [16]

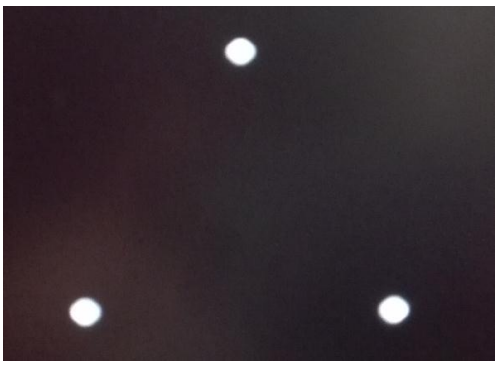

Figure 13. Display of points.

The design pattern that was used for the group of 37 pixels is shown in the Figure 14.

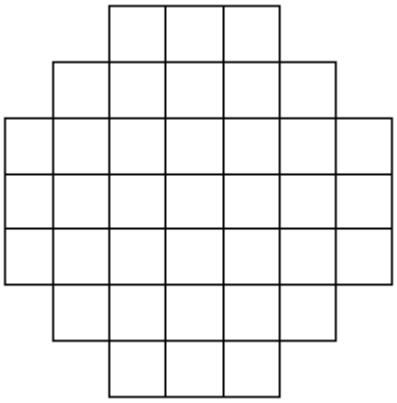

Figure 14. 37 pixel pattern

This pattern allows in some way visualize the pixel on a circular form, this is produce because of the size of the resolution is wide (640x480 pixel), if the resolution were lower the square patterns characteristic of the pixels would be displayed, by reproducing this same figure in different coordinates it can get the shape of a sinusoidal signal, as show on the 15 and 16 figures.

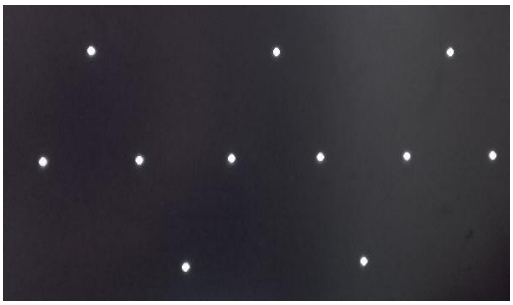

Figure 15. Image processing result

An axis can be a guide for the sinusoidal signal, these can simulate the different intervals of the signal values, and the beginning.

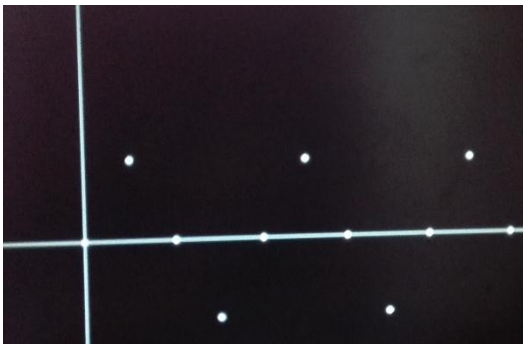

Figure 16. Sinusoidal signal

For the code on the 17 figure, two different variables will be created (X, Y), those variables allow to change the center of the circles, the last one were create using square and rectangles (Figure 19), changing the X and Y values can change the position of the circles without having to modify or create a new code, the verticals and horizontal lines on the Figure 21 will be created using an if conditional on the desired position.

```
PROCESS(disp_ena, filas, columna)
VARIABLE X  :  INTEGER RANGE 0 TO 640 := 200;
VARIABLE Y  :  INTEGER RANGE 0 TO 480 := 400;
BEGIN
   IF(disp_ena = '1') THEN
      IF(filas>=X  AND columna>=Y AND filas<=X+4 AND columna<=Y+4) THEN
         rojo <= (OTHERS => '1');
         verde <= (OTHERS => '1');
         azul <= (OTHERS => '1');
      ELSIF(filas>=X+1 AND columna=Y-1 AND filas<=X+3) THEN
         rojo <= (OTHERS => '1');
         verde <= (OTHERS => '1');
         azul <= (OTHERS => '1');
      ELSIF(filas=X-1 AND columna>=Y+1 AND columna<Y+4) THEN
         rojo <= (OTHERS => '1');
         verde <= (OTHERS => '1');
         azul <= (OTHERS => '1');
      ELSIF(filas>=X+1 AND columna=Y+5 AND filas<=X+3) THEN
         rojo <= (OTHERS => '1');
         verde <= (OTHERS => '1');
         azul <= (OTHERS => '1');
      ELSIF(filas=X+5 AND columna>=Y+1 AND columna<Y+4) THEN
         rojo <= (OTHERS => '1');
         verde <= (OTHERS => '1');
         azul <= (OTHERS => '1');
      ELSE
         rojo <= (OTHERS => '0');
         verde <= (OTHERS => '0');
         azul <= (OTHERS => '0');
      END IF;
   END IF;
END PROCESS;
```

Figure 17. Circle code generator

CONCLUSIONS

The image visualization on a VGA monitor allow to have an bigger control with respect to other visualization methods, with these type of controllers a greater precision and a pixel to pixel color control can be achieve , currently is use because of the increase in the use of technology and its ability to visualize transcendent information.

It is really important consider the sync times and the signal polarity, an incorrect polarity on the system will not allow the driver work properly, other aspect to have in count is the phase shift of the synchronism signal these will result in a problem in the visualization of pixels and it is common problem on a driver develop on a micro controller.

REFERENCES

[1] S. M. A. Mazin Rejab Khalil, «IEEE Xplorer,» IEEE Xplorer, 17-18 12 2013. [En línea]. Available: http://ieeexplore.ieee.org.ezproxy.umng.edu.co:2048/stamp/stamp.jsp?arnumber=6998747. [Último acceso: 07 02 2017].

[2] Y. Xu, «EBSCOhost,» 01 05 2017. [En línea]. Available: http://web.b.ebscohost.com.ezproxy.umng.edu.co:2048/ehost/pdfviewer/pdfviewer?vid=1&sid=5edbd901-a62e-4cbb-bf24-96d14181fd37%40 sessionmgr101. [Último acceso: 07 02 2017].

[3] D. G. Bailey, Design for Embedded Image Processing on FPGAs, New Zealand: Wiley, 2011-05-25.

[4] P. Wilson, «Design Recipes for FPGAs: Using Verilog and VHDL,» de Design Recipes for FPGAs: Using Verilog and VHDL, Oxford, Elsevier Science, 2011-02-24, pp. 161-168.

[5] Y. Li, «Computer Principles and Design in Verilog HDL,» de Computer Principles and Design in Verilog HDL, Japan, John Wiley & Sons, Incorporated, 2015-06-30, pp. 466-467.

[6] R. Y. S. M. Q. T. G. Elmar Yusif, «IEEE Xplore,» IEEE Xplore, 2014. [En línea]. Available: http://ieeexplore.ieee.org.ezproxy.umng.edu.co:2048/stamp/stamp.jsp?arnumber=7056798. [Último acceso: 07 02 2017].

[7] A. R. M. K. J. V. Staudt, «IEEE XPLORE,» 15-18 06 2015. [En línea]. Available: http://ieeexplore.ieee.org.ezproxy.umng.edu.co:2048/stamp/stamp.jsp?arnumber=7174718. [Último acceso: 07 02 2017].

[8] L.-p. N. P. S. Chen-lu Feng, «IEEE Xplore,» 21 01 2016. [En línea]. Available:http://ieeexplore.ieee.org.ezproxy.umng.edu.co:2048/stamp/stamp.jsp?arnumber=7387566. [Último acceso: 07 02 2017].

[9] P. Krzysztof (Kris) Iniewski, CMOS Nanoelectronics: Analog and RF VLSI Circuits, New York: McGraw-Hil, 2011.

[10] Eiguo Zhou, Y. L. (2016). Real-time Implementation of Panoramic Mosaic. *IEEE Xplore*, 6. Obtenido de IEEE Xplore: http://ieeexplore.ieee.org.ezproxy.umng.edu.co:2048/stamp/stamp.jsp?arnumber=7784026

[11] P. Krzysztof (Kris) Iniewski, CMOS Nanoelectronics: Analog and RF VLSI Circuits, New York: McGraw-Hil, 2011.

[12] S. Larson, «eewiki,» eewiki, 26 05 2017. [En línea]. Available: https://eewiki.net/pages/viewpage.action?pageId=15925278. [Último acceso: 07 02 2017].

[13] T. K. Eismin, Aircraft Electricity and Electronics, New York: McGraw-Hill Education, 2014.

[14] D. G. F. H. Wayne Beaty, Standard Handbook for Electrical Engineers, New York: McGraw-Hill Companies, 2013.

[15] M. Bass, Handbook of Optics: Volume I - Geometrical and Physical Optics, Polarized Light, Components and Instruments, New York: McGraw-Hill Professional, 2010.

[16] M. G. J. H. Xujun Ye, «IEEE Xplore,» IEEE Xplore, 08 02 2016. [En línea]. Available: http://ieeexplore.ieee.org.ezproxy.umng.edu.co:2048/stamp/stamp.jsp?arnumber=7399957. [Último acceso: 07 02 2017].