International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 20 (2018) pp. 14728-14732

Driver for Visualization of Graphics on VGA Screens using FPGA's

Edwar Jacinto Gómez, Fredy Martínez Sarmiento and Holman Montiel Ariza

Universidad Distrital Francisco José de Caldas, Facultad Tecnológica
Calle 68 D Bis A Sur No. 49F – 70, Bogotá D.C, Colombia.

Abstract

The research developed in this paper shows the design and implementation of a Core that is used as a video controller, with which a Stand-Alone system can be controlled at several levels. The system can display monochromatic messages on a VGA port, as well as images in 8-bit format using a minimum of memory. This controller can be implemented in any low cost FPGA, which allows the rest of the hardware resource to be used for other tasks, making the system fully reusable in a lot of applications.

Keywords: Core VGA, FPGA, Binary image , ROM.

INTRODUCTION

The handling of video in a digital application uses a significant part of the computational capacity, for this reason there have been countless jobs to design hardware that perform this work [1], these designs point to different types of uses: video as a display for Stand-Alone systems, real-time on-board camera visualization [2-4], high-speed image processing [5-8] and in general as support hardware in embedded processing in diverse applications where they use SOC's (System On Chip) as a processing core [9], design of video drivers in soft-core applications [10] and in general any possible use that requires information processing at high speed.

Visualization tasks in a digital system have a significant computational load in terms of the use of resources, specifically the frequency of the processor, which in order to fulfill high-resolution visualization tasks must be higher than 100Mhz [1]; On the other hand, it is sometimes necessary to make applications with multiple video inputs (cameras) [2] [3] that require the processing of real-time video in 2D and 3D [11], since it is currently one of the most incurred fields and in which the use of FPGA's is imperative, devices that are able to perform tasks of artificial intelligence, machine learning and deep learning in real time at high speeds. Another possible use of FPGAs at this level and with low cost, would be the applications that carry out tasks of signals acquisition in real time [12], where each of these variables could be visualized locally with a VGA port, since these devices are so fast it could be transmitted to other devices in parallel.

In this paper, a system is designed that drives the VGA port that could be carried to a chip of specific ASIC type [13] or in an application embedded in a modern FPGA, which means that it can be integrated to almost any system in a fast way and using few hardware resources; in general this CORE allows the realization of VGA applications without requiring a personal computer to display the information to the end user.

MATERIALS AND METHODS

The design of a video controller was done in a standard hardware description language, described as a generic CORE to visualize binary messages in a character-type monochromatic format or 8-bit color images. The proposed design is composed of the following functional blocks:

- **Image to ROM**: Script in Matlab that allows getting from a standard image file to a ROM memory described in VHDL.
- **Screen Synchronism Module**: Block in charge of generating the VGA protocol synchronization signals, the horizontal and vertical counters with which the coordinates are calculated.
- **ROM memory reading**: Block that performs the reading of the information in the required times and order, in order to visualize it statically and bounded on the screen.
- **Calculation of coordinates**: Block in charge of calculating the coordinates in which the information will be displayed and which allows these images to move on the screen according to the user's requirements.

DEVELOPMENT AND IMPLEMENTATION

Converting the image to ROM memory

The script has as an input file an image in standard digital format, in this case tests were performed with images in JPEG format, which must have certain characteristics, which make them easily transformable to an 8-bit depth format; an image with colors that differentiate in a simple way is looked for, in other words, it looks for images where the color is achieved using a pure channel or the combination of some of them. On the other hand, we look for images with an aspect ratio of 1x2 or 1x1, in such a way that it reaches a size of 16x32 or 32x32, since the FPGA has blocks of 1KByte maximum, this is done to avoid using more than one block memory by image.

After selecting the appropriate image, its resolution is reduced until it reaches the aforementioned amount of information, after which a separation of the RGB channels of the original image is performed and binary masks are made to discard the least significant bits of each color: 3 for the red channel, 3 for the green channel and only 2 for the blue channel; This is done to be fully compatible with 8-bit technology, which is something that can be easily modified to support another hardware configuration.

VGA system synchronization

From a core, see Fig. 1, the horizontal and vertical synchronization signals (HS and VS) are generated, together with a signal to indicate that work is being done within the screen space (Blank) and a pair of counters that generate a guide for the new coordinate system (hcount, vcount), these two counters are responsible for managing the addresses with which the memories are going to be read to show the information to be displayed on the VGA screen.

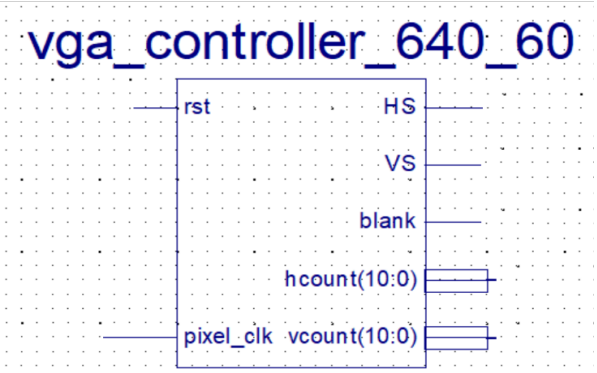

Figure 1: Core that generates system synchronization signals

ROM memory reading

This block is responsible for calculating the addresses for ROM reading (Addr_ROM) from the horizontal and vertical counters, this ROM can store binary information, with which it can display information type monochrome characters or can also contain binaries of any 8-bit depth image. From the information that is read from the ROM, a division of the data is carried out in the three RGB color channels that the CRT monitor receives, in the exact times to perform the task of visualizing the information, for this specific case, the data entry In_ROM receives a vector of 31 positions, which is converted into color channels to display a complete message on the screen.

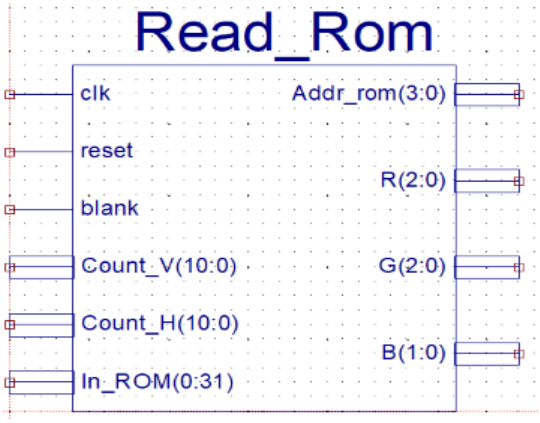

Figure 2: Block responsible for reading ROM data

Frequency generator base system

In this case it is required that the system has a clock of 25 MHz that will be the base of time that governs the system in general, besides this it requires another time base to perform slower tasks, such as performing motion effects in the texts or images to be shown on the screen, in this case a native block of the Xilinx FPGA's called DCM (Digital Clock manager) was used, which allows simply changing some constants to generate several base frequencies without the use of additional hardware resources.

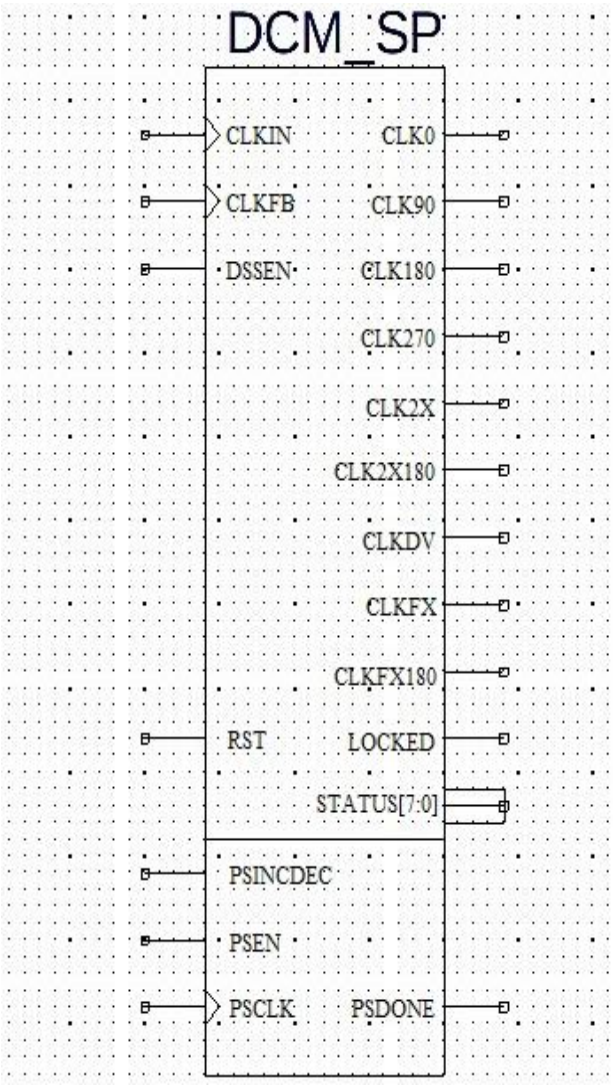

Figure 3: DCM Digital Clock Manager System clock controller

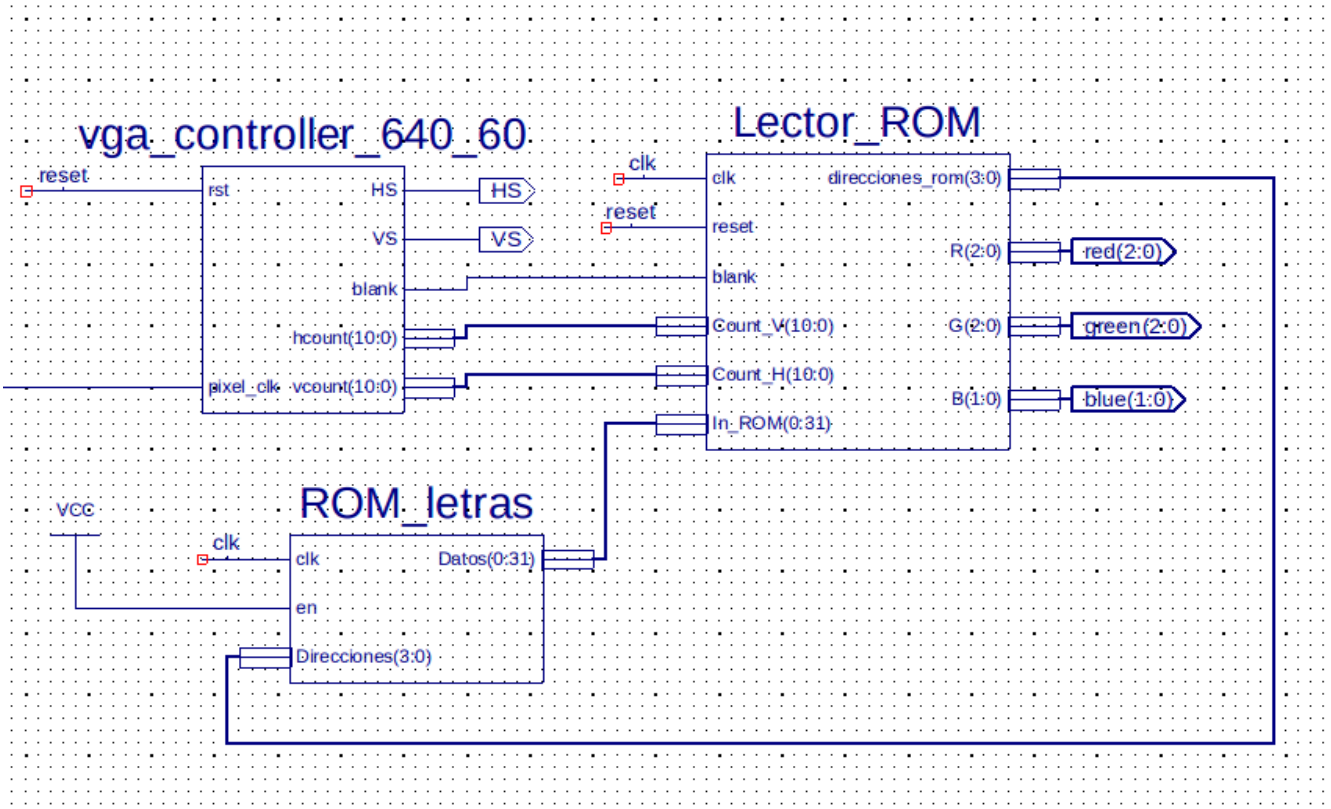

Figure 4: General block diagram of the system

It should be noted that even the smallest FPGA has several blocks that perfectly control the system clock, achieving precise frequencies essential for this type of application and most importantly without spending any type of hardware resource.

VGA driver

Having these basic blocks, we have a system that can show on a VGA monitor any type of information, originally it can be monochromatic character information, but in the end, several ROM memories can be generated with images in 8-bit format, these images can be displayed with different aspect ratios, achieving background effects, movement and resizing of the images to be displayed.

Fig. 4 shows a general block diagram of the system, which can be used to display different types of information, depending on the ROM memory, where the information to be displayed is stored on the screen.

Handling image memory (ROM)

In Fig. 4 a general block diagram is shown, where a ROM memory reader with a single memory is shown, in that memory the character type information can be stored, which is constructed in a simple way by means of some numerical manipulation program, such as excel, Matlab or some equivalent, this information shown in Fig. 5 can have a size of a few simple pixels, but can be increased to reach the full screen.

The ROM can also store images of a small depth, which are compatible with the digital to analog converter that is designed for the application, in this case 8 bits were chosen, 3 for red and green and only 2 for blue. The most interesting thing is that one can store a series of ROM memories with different images in such a way that can make small animations using certain background images and some other moving images that fit the needs of the application.

It isimportant to take into account the storage capacity in internal RAM memory of the FPGA, in this case it worked with a low cost Xilinx FPGA, which only has a few tens of kilobytes of storage capacity, but currently and with the explosion of the scale of integration that today is around ten nanometers per transistor; Xilinx and Altera have in the market economic FPGAs with a minimum of 1 Megabyte of internal storage, so that applications could be made with higher resolution images and in general with more pixels to be displayed on the screen.

To be able to show the different images on the VGA screen, multiplexing must be done in time, at a refresh rate higher than the persistence of the human eye, it is advisable to make this change between the images shown at a minimum of 60 Hz, verifying practice if this change is sufficient for the displayed images to be seen fluently.

```
signal ROM : rom_type:= ("00000000000000000000000000000000",
                         "00011111100111111100000111111000",
                         "00111111110111111110001111111000",
                         "00110000110110000110001000000000",
                         "00110000110110000110011000000000",
                         "00110000110110000100011000000000",
                         "00111111110111111000011000000000",
                         "00111111110111111100011000000000",
                         "00110000110110000110011000000000",
                         "00110000110110000110011000000000",
                         "00110000110110000110011000000000",
                         "00110000110110000100001000000000",
                         "00110000110111111000001111111000",
                         "00110000110111111000000111111000",
                         "00000000000000000000000000000000",
                         "00000000000000000000000000000000"
                         );
```

Figure 5: Binary image, simple monochromatic visualization

Object movement control

One of the aspects to take into account in this paper, is that a design scheme is proposed where the coordinates of the images or characters to be displayed are generated, this calculation of coordinates depends on the application, in other words, if only it requires displaying a monochromatic text type message, it could be shown horizontally in a certain part of the screen, in that case only an incremental calculation of the coordinates in the horizontal counter of the display block that reads the ROM information would be required, this increment would be seen simply as the increment of a counter in the reading pointer of the horizontal ROM memory, this increment must be done at a much lower time base than that of the refresh of the screen to realize an animation effect in the final user. Next, the code in VHDL language that performs the movement of a 256 color image is shown:

```
if blank = '0'  then
if ((count_H > move_x) and (count_H < (24*4+move_x))) and  count_V < 32*4 then

R  <= pixel_actual(7 downto 5);
G  <= pixel_actual(4 downto 2);
B  <= pixel_actual(1 downto 0);

if count_H_temp(7 downto 3) < 24 then
addr_rom  <= ("11000" * count_V(6 downto 2)) + ("00000" & count_H_temp(6 downto 2));
        else
                addr_rom  <=  (others => '0');
        end if;

else…
```

This section of code will be responsible for performing the coordinate calculations at a refresh rate according to the needs and constraints of the user, and in general this calculation of coordinates can be described in hardware as a frequency divider that increases a counter that adds to the read pointer of the ROM memory and position the information read in another part of the screen, all this at a speed of multiplexing and refresh higher than the persistence of the human eye, to avoid stroboscopic effects that are noticed at the moment of showing the information on the VGA screen.

A visualization test with movement is shown in Fig. 6, where an image is shown to which the whole process described in this work was performed and the visualization of a character of the 8-bit games with a small animation is achieved of movement.

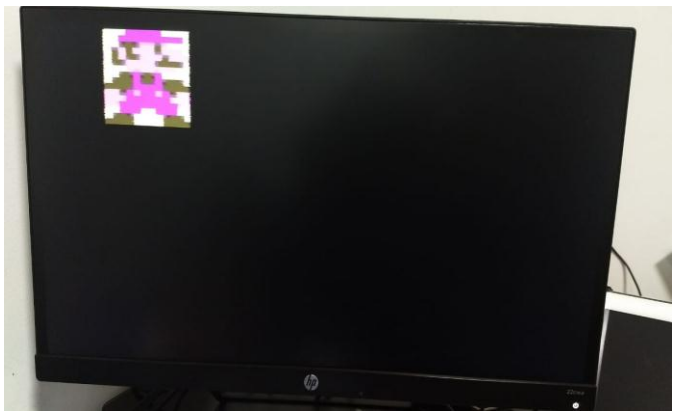

Figure 6: Real view of an 8-bit image

CONCLUSIONS

A video CORE was made in a low cost FPGA, which is capable of displaying binary monochromatic images and / or 8-bit images only using the 1% of the logical gates available in a Spartan 3E 100K gates and a single block of 1K RAM memory by 8 bits.

Tests were performed with different types of information or low resolution images, making small animations, generation of movement and in general processes that require calculation of coordinates of the information to be displayed.

A series of tests were carried out in the classroom, verifying whether students through practical experiences carried out the process of learning to digitize images, playing video through the VGA port, managing memories and working with pointers, accountants and pre-scalers, in order to generate knowledge appropriation regarding the visualization and manipulation of images in VGA format.

ACKNOWLEDGMENT

This work was supported by the Universidad Distrital Francisco José de Caldas Facultad Tecnológica. The views expressed in this paper are not necessarily endorsed by the university. The authors thank the research group ARMOS for the evaluation carried out on prototypes of ideas and strategies.

REFERENCES

[1] V. H. Tran and X. T. Tran, "An efficient architecture design for VGA monitor controller," *2011 Int. Conf. Consum. Electron. Commun. Networks, CECNet 2011 - Proc.*, pp. 3917–3921, 2011.

[2] N. N. E. Murphy, F. Morgan, and J. Manning, "An FPGA based Stereoptic Image Capture System.," *Issc.Exordo.Com*, pp. 95–95, 2012.

[3] K. Takaya and Z. Qian, "FPGA based stereo vision system to display disparity map in realtime," *2012 Int. Conf. Inf. Sci. Appl. ICISA 2012*, no. 1, pp. 3–6, 2012.

[4] C. Plaza, O. Ramos, and D. Amaya, "VGA Configuration Algorithm using VHDL," vol. 13, no. 14, pp. 11572–11576, 2018.

[5] R. Jeyakumar, M. Prakash, S. Sivanantham, and K. Sivasankaran, "FPGA implementation of edge detection using Canny algorithm," *IC-GET 2015 - Proc. 2015 Online Int. Conf. Green Eng. Technol.*, pp. 1–4, 2016.

[6] L. Sheng, J. Shao, M. Xu, and Y. Cui, "Display verification IP core design based on star-extraction and star-recognition image processing on FPGA," *Proc. - 2011 5th Int. Conf. Innov. Mob. Internet Serv. Ubiquitous Comput. IMIS 2011*, pp. 151–155, 2011.

[7] V. Vanishree and K. V. R. Reddy, "Implementation of pipelined sobel edge detection algorithm on FPGA for High speed applications," *Proc. - 2013 Int. Conf. Emerg. Trends Commun. Control. Signal Process. Comput. Appl. IEEE-C2SPCA 2013*, pp. 1–5, 2013.

[8] C. Feng, L. Niu, and P. Shi, "The design of color space conversion based on FPGA," *2015 IEEE Int. Conf. Comput. Commun.*, no. 2, pp. 194–198, 2015.

[9] O. Grairi, "Design and Implementation of a Versatile Display System based on FPGA for Embedded Systems," pp. 2–6, 2015.

[10] E. Zheng, Z. Song, and L. Ma, "Data acquisition and VGA display controller based on NiosII," *ICSPS 2010 - Proc. 2010 2nd Int. Conf. Signal Process. Syst.*, vol. 3, pp. 417–421, 2010.

[11] W. Yun *et al.*, "OFFSET APERTURE BASED HARDWARE ARCHITECTURE FOR REAL-TIME DEPTH EXTRACTION School of Electrical Engineering , KAIST , Daejeon , Republic of Korea Division of Electrical Engineering , Kongju National University , Cheonan , Republic of Korea Center for Inte," pp. 4392–4396, 2017.

[12] W. Aihua, L. Dong, and W. Zhen, "The design Of VGA data communication based on FPGA," *2011 IEEE Int. Symp. IT Med. Educ.*, vol. 1, pp. 649–651, 2011.

[13] R. Szabo and A. Gontean, "Mouse and display driver on a single microchip tested on FPGA and built for an ASIC," *2016 IEEE 22nd Int. Symp. Des. Technol. Electron. Packag. SIITME 2016*, pp. 118–121, 2016.