Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Technology 3 (2012) 324 – 333

Procedia Technology

The 2012 Iberoamerican Conference on Electronics Engineering and Computer Science

FPGA Open Architecture Design for a VGA Driver

Carlos Alberto Ramos-Arreguín[a]*, Juan Carlos Moya Morales[a], Juan Manuel Ramos-Arreguín[a], Jesus Carlos Pedraza-Ortega[a], Saúl Tovar-Arriaga[a], Marco Antonio Aceves Fernandez[a],Jose de Jesus Rangel-Magdaleno[b]

[a]Faculty of Computer Science, Universidad Autónoma de Querétaro
[b]Universidad Politécnica de Puebla
Av. De las Ciencias s/n. C.P., 76230Juriquilla Querétaro, Qro., México
San Mateo Cuanalá, Juan C. Bonilla, Puebla, México

Abstract

This work presents an open architecture proposal for a VGA (Video Generic Array) controller to be used into embedded systems based in FPGA. Several developers of hardware design, which use some hardware description language, have a video library to be used with VHDL or Verilog language, but in most of cases, we need to buy an expensive annual license, and can be used only for the manufacturer hardware. This controller is developed using only VHDL based in the IEEE standards, to ensure the portability with any manufacturer, and this is part of the contribution of this work. The controller designed is generic, due that it can be used for any resolution used by a commercial monitor, including the widescreen monitor. The generic controller will be used for image processing research. The work presents two kinds of test: first, we display the eight basics color generated with the RGB (Red, Green, Blue) values; second, an image is stored into an external memory RAM (Random Access Memory), and the FPGA reads and display the image into a CRT (Cathode Ray Tube) and LCD (Liquid Crystal Display) monitor.

Keywords: FPGA; VGA; resolution.

---

1. Introduction

At present, the use of FPGA in research and development of applied digital systems for specific tasks is increasing. This is due to the advantages FPGAs has over other programmable devices. These advantages are: high clock frequency, high operations per second, code portability, code libraries reusability, low cost, parallel processing, capability of interacting with high or low interfaces, security and Intellectual Property (IP) retention, among others [1].

One of the topics where FPGA has been used is image processing, attributable to the high degree of parallelism in

---

* Corresponding author. Tel.: +52 442 192 1200 ext 5941.
E-mail address: c.ramos@ieee.org.

doi:10.1016/j.protcy.2012.03.035

comparison with a PC or a microcontroller where processing is sequential. Nowadays, there is much work carried out in the field of image processing based on FPGA, where image processing based algorithms have been widely implemented [2 - 11].
In order to be able to see the results obtained from the implemented algorithms on an FPGA, a standard VGA (Video Graphics Array) display may be chosen. However, a video controller is necessary to show the results in the display. This controller must be designed based on the datasheets of the FPGA's developing board, where the synchronization timing is specified. The VGA resolution corresponds to 640x480 pixels, created by IBM PCs in the 80's decade [12], which is widely support by both CRT and LCD technologies.

In this work, the video controller is proposed, showing from the standard VGA theory, to the results obtained in the development board for FGPA's Spartan 3 [13], Spartan 3AN [14], Nexys 2 [15], and Basys 2 [16].

2. Signal Timing for a 60 Hz, 640x480 VGA Display

CRT-based VGA displays use amplitude-modulated, moving electron beams (or cathode rays) to display information on a phosphor-coated screen. LCD displays use an array of switches that can impose a voltage across a small amount of liquid crystal, thereby changing light permittivity through the crystal on a pixel-by-pixel basis. LCD displays have evolved to use the same signal timings as CRT displays. Within a CRT display, current waveforms pass through the coils to produce magnetic fields that deflect electron beams to transverse the display surface in "raster" pattern, horizontally from left to right and vertically from top to bottom. Fig. 1 shows an example on exhibit time and vertical and horizontal sync on a CRT monitor [13 - 17].

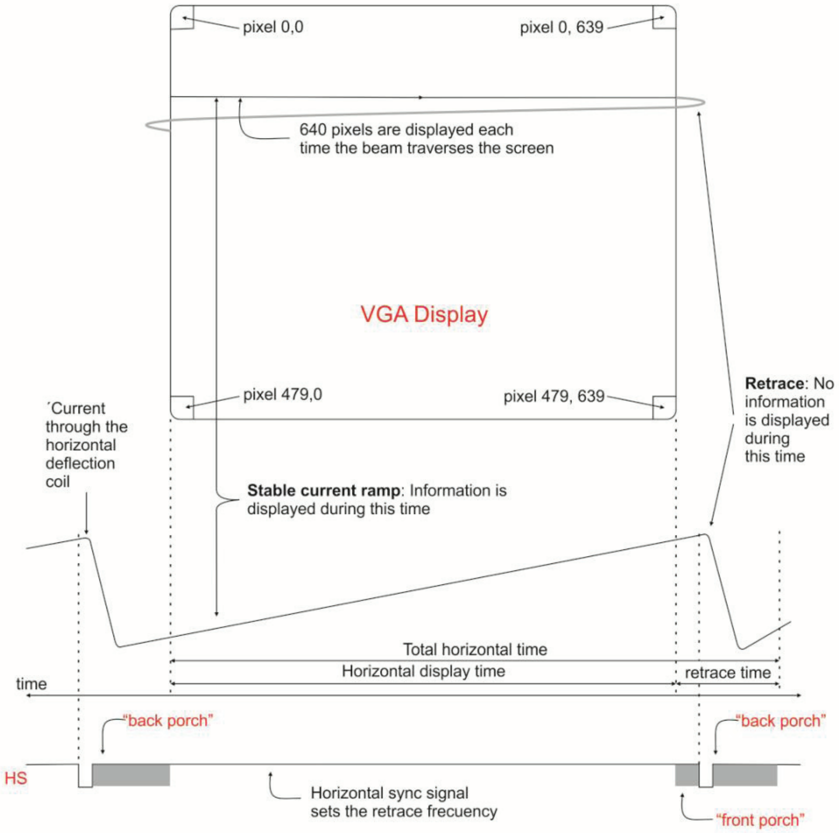

pixel 0,0
pixel 0, 639
640 pixels are displayed each time the beam traverses the screen
VGA Display
pixel 479,0
pixel 479, 639
Retrace: No information is displayed during this time
'Current through the horizontal deflection coil
Stable current ramp: Information is displayed during this time
Total horizontal time
Horizontal display time
retrace time
time
"back porch"
"back porch"
HS
Horizontal sync signal sets the retrace frecuency
"front porch"

Fig 1. CRT Display Timing VGA Example.

The timing needed for horizontal and vertical (HS and VS, respectively) VGA port sync of a monitor with a clock frequency of 25MHz with display resolution of 640-pixel by 480 row, are shown on table 1.

Table 1.640x480 Mode VGA Timing

Symbol	Parameter	Vertical Sync			Horizontal Sync	
		Time	Clocks	Lines	Time	Clocks
T_S	Sync pulse time	16.7 ms	416, 800	521	32 µs	800
T_{DISP}	Display time	15.36 ms	384, 000	480	25.6 µs	640
T_{PW}	Pulse width	64 µs	1, 600	2	3.84 µs	96
T_{FP}	Front porch	320 µs	8, 000	10	640 ns	16
T_{BP}	Back porch	928µs	23, 200	29	1.92 µs	48

The timing of VGA driver according to table 1 is shown on Fig. 2.

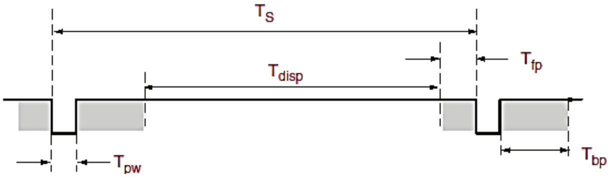


Fig. 2. VGA Control Timing

Digilent® provides in the reference manual of the development boards Basys-2® and Nexys-2®, the block diagram of the VGA controller, which is shown on Fig. 3.

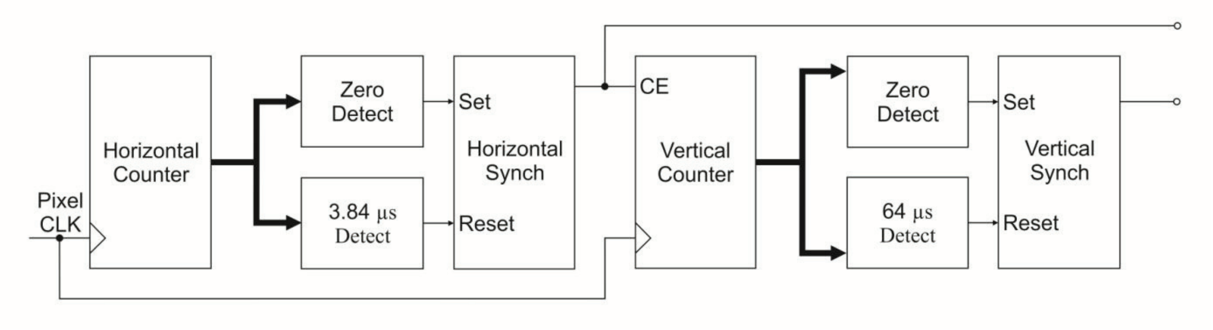


Fig. 3. Digilent VGA Sync General Block Diagram.

The VGA driver ought to generate HS and VS signals and coordinates the delivery of the video stream, based on the Pixel CLK (25MHz), this clock defined the needed time to display the pixel information. The signal VS defines the frequency of the display refresh rate, or the frequency in which all the information of the display is re-drawn. The controller decodes the output of the horizontal counter module to generate the HS signal time. This counter may be used to locate the pixel of a given row. Also, the output for the module Vertical Counter that increases the HS pulse may be used to generate the VS output time and this counter may be used to locate any row [15].

3. Development of the Proposed Controller

The controller proposed in this work is also based on table 1. The main difference is that instead of using the times for each part, the counting is carried out by the number of lines. The resolution values used for 640x480 are shown on table 2.

Table 2.Used resolution values for the proposed controller for a 640x480.

Symbol	Parameter	Vertical Sync Lines	Horizontal Sync Lines
T_S	Sync Pulse	521	800
T_{DISP}	Display Time	480	640
T_{PW}	Pulse Width	2	96
T_{FP}	Front porch	10	16
T_{BP}	Back porch	29	48

The VGA Driver architecture is obtained following the methodology TOP-DOWN [17]. Fig. 4 presents a TOP-DOWN methodology diagram, where a 1^{st} level design (main entity) is presented, the second level consists in specify the main entity components; in this case, every module don't have internal modules, in last level we find the module description of operation.

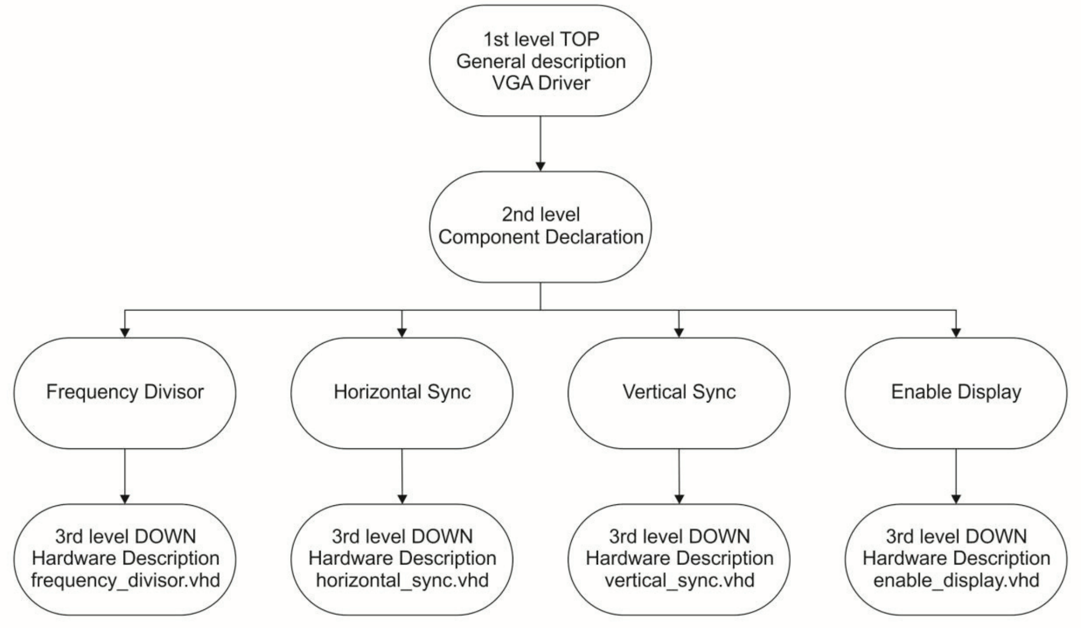


Figure 4. Design with TOP-DOWN Methodology.

Therefore, the general block diagram (called "entity") is shown on Fig 5.

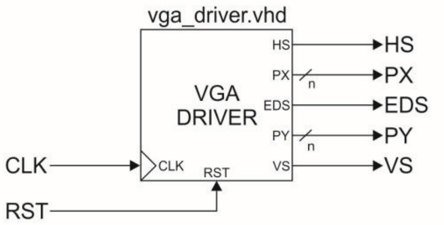


Fig. 5. General VGA sync controller diagram with clock frequency of 25MHz.

For the Fig. 5, the principal signals on the entity are shown in table 3, including a brief description.

Table 3. Principal System Signals

Signal	Description
CLK	Main Clock signal coming from a 50MHz clock from the developing board Nexys 2.
RST	System´smainreset
HS	Horizontal synchronization signal
PX	Indicates the actual position from the horizontal line
EDS	Enables the actual pixel display on the monitor, the pixel is displayed when EDS is on high state.
PY	Indicates the actual pixel position on the vertical line
VS	Vertical synchronization signal
N	Indicates the bit size of the signal. This is set according to the resolution specifications. For a 640x480 pixel resolution, it requires a zise 10 data bus.

The internal architecture of the VGA controller, consists of four modules to generate the output signals shown on Fig. 5, to get a successful synchronization with the monitor. The interconnections with the other modules (Frecuency Divisor, Horizontal Sync, Vertical Sync and Enable Display) are shown on Fig. 6.

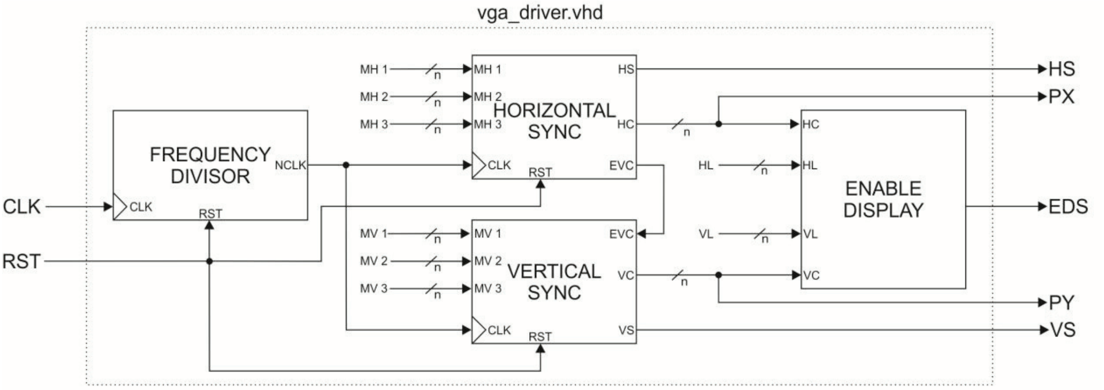


Fig. 6. Component Interconnection of the VGA Driver.

The module called Frequency Divisor generates a frequency of 25MHz to work with a 640x640 resolution. The output signal NCLK corresponds to the 25MHz signal, which is received by the modules Horizontal_Sync and Vertical_Sync to generate both sync signals needed for monitor display process. This module description is shown on Fig. 7.

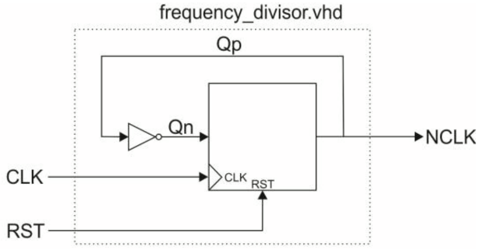


Fig. 7. Frequency Divider Module.

The 25MHz signal, NCLK, is connected to the ***vertical_sync.vhd*** y ***horizontal_sync.vhd*** modules such as the clock signal. The modules used to generate the horizontal sync long with its signals are shown on Fig. 8.

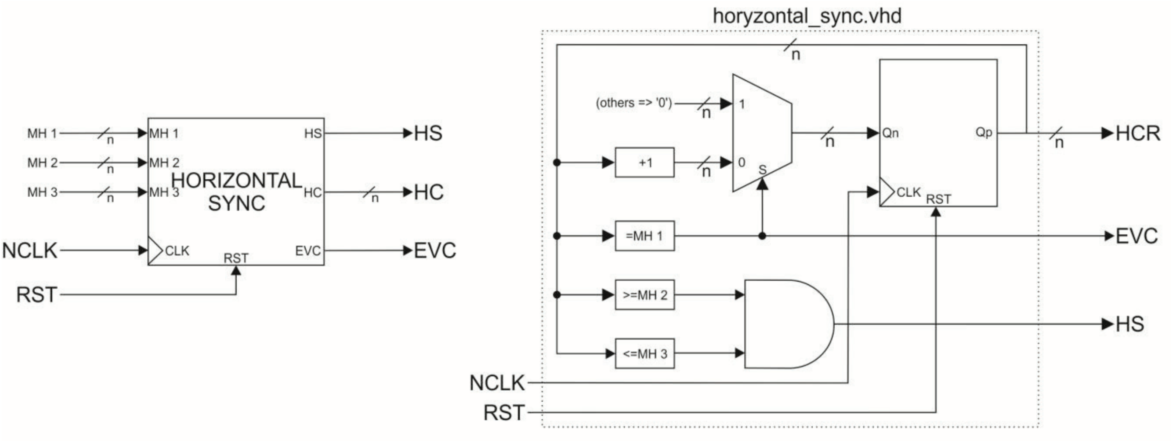


Fig. 8. (a) Entity defined for the Horizontal_sync module; (b) Architecture of the Horizontal_sync module.

The signals **MH1**, **MH2**, **MH3** y **HC** are data buses of size n and their values are obtained from equations 1-4.

$$MH1 = T_S - 1 \tag{1}$$
$$MH2 = T_{DISP} + T_{FP} - 1 \tag{2}$$
$$MH3 = T_{DISP} + T_{FP} - T_{PW} - 1 \tag{3}$$
$$HC = T_{DISP} \tag{4}$$

Fig. 8 (b) shows the internal connection from the module **Horizontal Sync,** where a 2-1 multiplexor is used. The signal **S** indicates whether the value is incremented or reset to 0. Afterwards, the output is sent to a D Flip-Flop (FF) synchronized to 25HMz. This FF is used to store the actual data until the following data is received. The signal **HC** indicates the horizontal position of the actual refresh position. The signal **EVC**, indicates the moment in which the count reached its peak value (**MH1**) and enables the count for module V**ertical Sync.** The signal **HS** allows the horizontal sync with the monitor interface when its value is high. This value is generated while the current count is greater or equal than **MH2** and **MH3**.

Fig. 9 shows the module description that generates the vertical sync and its corresponding signals.

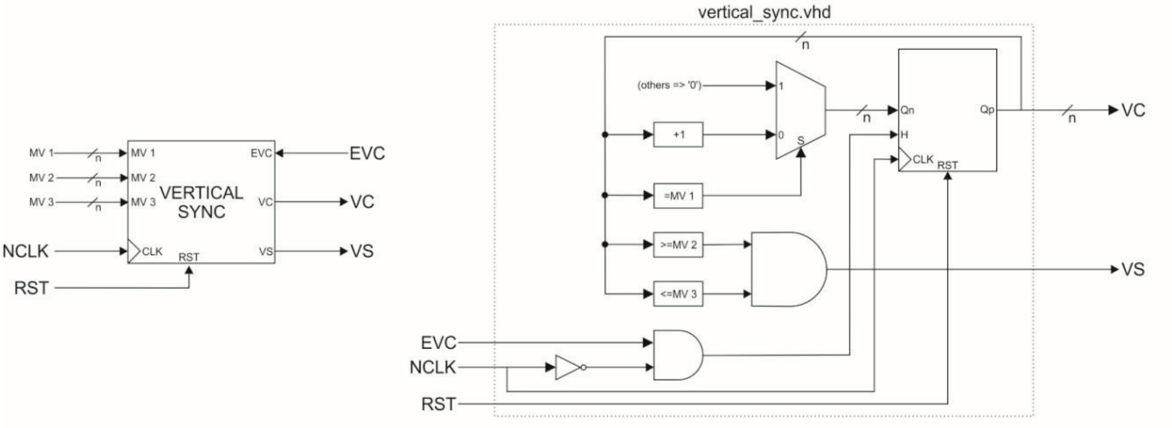


Fig. 9. (a) Vertical Sync Description; (b) Vertical Sync Internal Description.

The signals **MV1, MV2, MV3** and **VC** are data buses of size **n** and its values are obtained with respect to the indications of table 2 from the vertical sync, its values are obtained from a similar way to the equations 1-4 previously mentioned. These calculations are shown on equations 5-8.

$$MV1 = T_S - 1 \tag{5}$$
$$MV2 = T_{DISP} + T_{FP} - 1 \tag{6}$$
$$MV3 = T_{DISP} + T_{FP} - T_{PW} - 1 \tag{7}$$
$$VC = T_{DISP} \tag{8}$$

In Fig. 9 (b) the internal wiring of **Vertical Sync** is shown. It is observed that the descriptions is similar to Fig. 8 (b), with the difference that the register where the actual count is stored, requires an activate signal **H** generated with the signal **EVC** and the signal **NCLK** complemented. This means that when its value is high, the value of the signal **VC** is updated increase by 1 or reset to 0. The sync signal **VS** is generated while the value of the count **VC** is less or equal than M**V2** or greater or equal than **MV3**.

In Fig. 10, the comparator module that generates the signal **EDS** is observed. This module indicates the moment in which the current pixel will be displayed on the monitor in the case it is situated within the resolution range of 640x480 pixels.

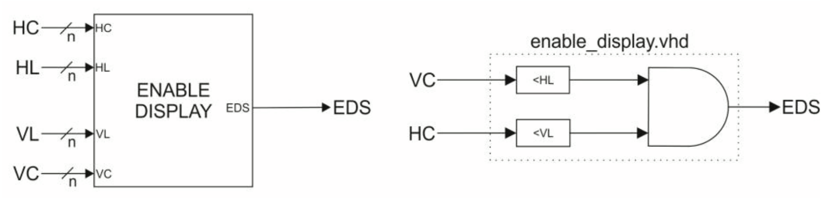


Fig. 10. Enable pixel component.

4. Testing and Results

To test the proposed driver, an 8 basic RGB colors display were carried out. Table 3 shows the color code from each of the development boards mentioned previously [13 – 16].

Table 3. Color code from each of the development boards used in the test.

Spartan 3 3 bit RGB			Spartan 3AN 12 bit RGB			Nexys 2 8 bit RGB			Basys 2 8 Bit RGB			Color
R	G	B	R	G	B	R	G	B	R	G	B	
0	0	0	0000	0000	0000	000	000	00	000	000	00	Black
0	0	1	0000	0000	1111	000	000	11	000	000	11	Blue
0	1	0	0000	1111	0000	000	111	00	000	111	00	Green
0	1	1	0000	1111	1111	000	111	11	000	111	11	Cyan
1	0	0	1111	0000	0000	111	000	00	111	000	00	Red
1	0	1	1111	0000	1111	111	000	11	111	000	11	Magenta
1	1	0	1111	1111	0000	111	111	00	111	111	00	Yellow
1	1	1	1111	1111	1111	111	111	11	111	111	11	White

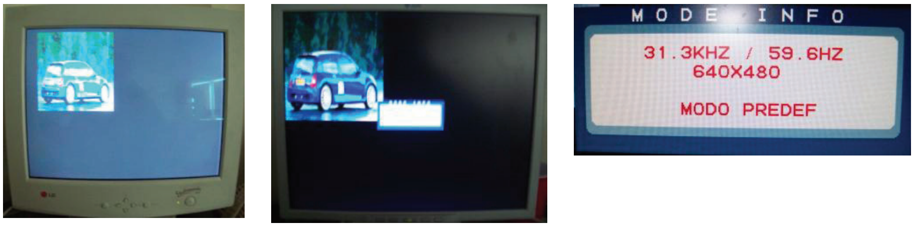
The colors displayed with the controller used in the test are shown on table 4. The monitor used is a VGA port CRT. However, the system can be connected to any VGA port LCD monitor.

Table 4. Example showing the monitor with the proposed driver.

Black Blue Cyan Green

Red Magenta Yellow White

Another example employed to test the performance of the proposed controller, consist in reading an image stored on an external RAM memory [19]. The result of this test is shown on Fig. 11.



MODE INFO

31.3KHZ / 59.6HZ
640X480

MODO PREDEF

Fig. 11. Image stored on an external RAM memory and displayed on a monitor; (a) CRT Monitor, (b) LCD Monitor, (c) Information of the operation mode.

The Fig. 12 shows a time diagram, obtained by the hardware simulation using Active-HDL, it shows the timing of each main signal on VGA Driver in one second. On Fig. 12 (a) VS signal is in "high" when the vertical count finished and restarts to 0, one pulse on high of VS signal is generated in 15.68 ms, it means that in one second, we have 60 screens displayed on monitor, like it see in Fig 11 (c). Fig. 12 (b) shows the cycle time to generate every horizontal synchronous (HS signal) each 26.21 µs.

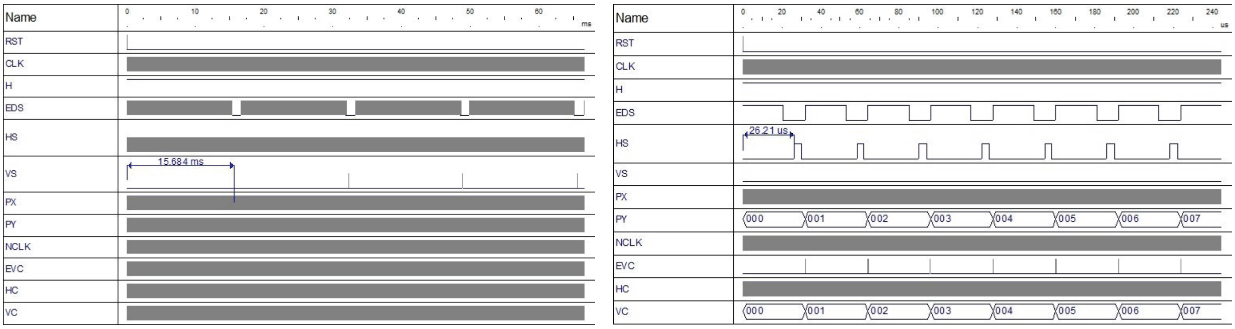


Figure 12. VGA Driver Simulation using Ative-HDL, (a) VS Signal Timing, (b) HS Signal Timing.

The proposed driver allows modification to other resolutions which are shown on table 4. The video controller proposed in this work can be used for several video resolutions [20]. To do that, a change for MH1, MH2, MH3, HC, MV1, MV2, MV3 and VC are needed, according with the specifications of the resolution. A limitation for the video controller is the clock frequency, because with higher resolution, a higher clock frequency is needed. The user has to compute the value for the frequency divider and get the frequency needed. If the system has the exact frequency needed, the Frequency Divider module can be omitted. In figure 10 it is observed a good synchrony and display of an image in the monitor regardless the type of monitor (CRT or LCD), even though the projection scheme is different. Also, the operation mode indicates the resolution and the current frequency of the monitor, whose resolution is 640x480 with a refresh rate of 59.6Hz., the horizontal path per column last 31.97µs, whereas the total path of the monitor is16.64ms.

Table 4. VGA resolutions parameters.

		SVGA 800x600 72Hz 50MHz				
Symbol	Parameter	Vertical Sync			Horizontal Sync	
		Time	Clocks	Lines	Time	Lines
T_S	Sync Pulse	13.85 ms	692, 500	666	20.79 µs	1040
T_{DISP}	Display Time	12.5 ms	625, 000	600	16 µs	800
T_{PW}	Pulse Width	124.8 µs	6, 240	6	2.39 µs	120
T_{FP}	Front porch	769.6 µs	38, 480	37	1.27µs	64
T_{BP}	Back porch	478.4 µs	23, 920	23	1.11 µs	56
		XGA 1024x768 60Hz 65MHz				
T_S	Sync Pulse	16.66 ms	1, 082, 900	806	20.67 µs	1344
T_{DISP}	Display Time	15.87 ms	1, 031, 550	768	15.75 µs	1024
T_{PW}	Pulse Width	124.06 µs	8, 064	6	2.09 µs	136
T_{FP}	Front porch	62.03 µs	4, 032	3	0.36 µs	24
T_{BP}	Back porch	599.63 µs	38, 976	29	2.46 µs	160

5. Conclusions

The contribution of this work is an open architecture and standardization of a video VGA controller with all the required signals for correct display and performance and the architecture of how to obtain each signal. This architecture may be used in any FPGA device regardless of the brand or model and is scalable to any resolution such as: SVGA, XGA or WUXGA. Furthermore, this architecture minimizes the developing time of a controller according to the datasheet, with respect to the design and planning of the controller. Also, a worth noticing that there are IP cores created by each FPGA maker. Nevertheless, to use this modules or libraries requires an extra cost and are no portable to every FPGA.

The block diagram used in this work can help to anybody who wants to implement a VGA driver to get a successful result. Major of designs provided in the open literature only shows the code in VHDL, Verilog, etc, but a graphical description is not presented. The Block Diagram Hardware Description presented in this work, let us to implement the driver VGA controller easily with any hardware description language.

Acknowledgment

We would like to thank Consejo de Ciencia y Tecnología del Estado de Querétaro (CONCYTEQ) by the 2012-01 science and technology activities programs, and the Universidad Autónoma de Querétaro (UAQ) for the facilities and support.

References

[1] Dubey Rahul, Introduction to Embedded System Design Using Field Programmable Gate Arrays, Springer-Verlag London Limited, ISBN: 978-1-84882-015-9, Londres, 2009.

[2] Ramos Arreguín Carlos Alberto, Cora Gallardo Orlando Marcos, Ramos Arreguín Juan Manuel, Pedraza Ortega Jesús Carlos, Canchola Magdaleno Sandra Luz, Vargas Soto José Emilio, Metodología para Manejo de Imágenes en FPGA, 6º Congreso Internacional de Ingeniería, pp. 219-226, ISBN: 978-607-7740-39-1, Querétaro, Qro., Abril 2010

[3] Quintero M. Alexander, Vallejo R. Eric, Image Processing Algorithms using FPGA, Revista Colombiana de Tecnologías de Avanzada, Vol. 1; No. 7; pp. 11-16, ISSN: 1692-7257, 2006.

[4] Bravo Muñoz Ignacio, Arquitectura basada en FPGA para la Detección de Objetos en Movimiento, utilizando Visión Computacional y Técnicas PCA, Tesis Doctoral, Departamento de Electrónica de la Escuela Politécnica de la Universidad de Alcalá, España 2007.

[5] A. Castillo, J. Vázquez, J. Ortegón, C. Rodríguez, Prácticas de Laboratorio para Estudiantes de Ingeniería con FPGA, IEEE Latin America Transactions, pp. 130-136, Junio 2008.

[6] Ramos Arreguín Carlos Alberto, Moya Morales Juan Carlos, Ramos Arreguín Juan Manuel, Pedraza Ortega Jesús Carlos, Metodología de una Etapa Básica de un Sistema de Procesamiento de Imágenes basado en FPGA, 9º Congreso Nacional de Mecatrónica, pp. 235-240, ISBN: 978-607-95347-2-1, Octubre 2010.

[7] Kalomiros J. A., Lygouras J., Design and evaluation of a hardware/software FPGA-based system for fast imageprocessing, Microprocessors and Microsystems, Elsevier, pp. 95– pp. 106, doi:10.1016/j.micpro.2007.09.001.

[8] ChaikalisD., Sgouros N. P., Maroulis D., A Real Time FPGAArchitecture for 3D reconstruction from integral images,Journal of Visual Communication & Image Representation,Elsevier, pp. 9 - pp. 16, doi:10.1016/j.jvcir.2009.09.004.

[9] SiélerL., Tanougast C., Bouridane A., A scalable andembedded FPGA architecture for efficient computation of graylevel co-ocurrence matrices and Haralick textures features,Microprocessors and Microsystems, Elsevier, pp. 14 - pp. 24,doi:10.1016/j.micpro.2009.11.001.

[10] Krill B., Ahmad A., Amira A., Rabah H., An efficient FPGAbaseddynamic partial reconfiguration design flow andenvironment for image and signal processing IP cores, SignalProcessing: Image Communication, Elsevier, pp. 377 - pp. 387,doi:10.1016/j.image.2010.04.005.

[11] Satake Shin-ichi, SorimachiGaku, Masuda Nobuyuki, ItoTomoyoshi, Special-purpose computer for particle imagevelocimetry, Computer Physics Communications, Elsevier, pp.1178 – pp. 1182, doi:10.1016/j.cpc.2011.01.022.

[12] Pong P. Chu, FPGA Prototyping by VHDL Examples XilinxSpartan 3 Version, Wiley Interscience, pag. 257 – pag. 266, ISBN: 978-0-470-18531-5, USA 2008.

[13] Xilinx, Spartan-3 FPGA Starter Kit Board User Guide, www.xilinx.com.

[14] Xilinx, Spartan-3A/3AN FPGA Starter Kit Board User Guide, www.xilinx.com.

[15] DigilentNexys 2 Board Reference Manual, www.digilentinc.com.

[16] DigilentBasys 2 Board Reference Manuel, www.digilentinc.com.

[17] Romero Troncoso René de Jesús; Electrónica Digital y Lógica Programable; Universidad de Guanajuato, 2ª. Edición; ISBN: 968-864-449-8; Guanajuato México, 2007.

[18] Marco Antonio Aceves Fernández, Juan Manuel Ramos Arreguín; Fundamentos de Sistemas Embebidos, Mediante Lenguajes Descriptivos de Hardware; Asociación Mexicana de Mecatrónica A. C.; ISBN: 978-607-95347-4-5; Santiago de Querétaro, Qro., México, 2011.

[19] C. A. Ramos Arreguín, J. C. Moya Morales, J. M. Ramos Arreguín, J. C. Pedraza Ortega, M. A. Aceves Fernández, J. E. Vargas Soto, S. Tovar Arriaga; Metodología para almacenamiento de imágenes en Memorias externas de tipo RAM, empleando FPGA; IX Congreso sobre Innovación y Desarrollo Tecnológico, CIINDET 2011; ISBN: 978-607-95255-3-8; Cuernavaca, Morelos, México, 2011.

[20] http://tinyvga.com/vga-timing, last consult 12/08/11.