A Flexible FPGA communication

Shubha Hiremath[1], Meghana Kulkarni[2]

[1]MTech student, Department of VLSI Design and Embedded systems, VTU Belgavi, Karnataka, India
[2]Associate Professor, Department of VLSI Design and Embedded systems VTU Belgavi, Karnataka, India

Abstract *— This work is aimed at design and implementation of VGA controller. Video Graphics Interface is widely used as standard display interface. The design consists of top layer module design and the timing function simulation. Hardware architecture is implemented on a NEXYS 3 SPARTAN 6 Field Programmable Gate Array chip, which has gained considerable traction for its application in conjunction with VGA controllers [1]. The focus is on system architecture, hardware design and software programming. This controller is developed using only VERILOG (hardware description language) based on the IEEE standards, to ensure the portability for any manufacturer. The system can display various color strips, Chinese characters, images and even perform certain image processing techniques such as data compression, noise removal etc. The results show that this proposed algorithm gives good performance with optimal time and low resource utilization of up to 54% registers and 31% functional units. Since, the data can be sent directly to monitors, the design speeds up data processing, improve system reliability in real-time and conserve hardware resources.*

Keywords *— FPGA, VERILOG, VGA Controller, Nexys 3 Spartan 6.*

I. INTRODUCTION (SIZE 10 & BOLD)

The most popular display interface for the applications like video conference systems, face-recognition systems, surveillance and remote vehicle guidance systems, is Video Graphics Array (VGA). VGA Controller is a system which works with high frequency signals, possessing a property of hardware exclusivity. Initially, the implementation and corresponding usage was carried out directly on the printed circuit boards, however the solutions corresponding to these are accompanied with parameters like over sizing and high power dissipation.

The tools which are quite common are DSP, or GPU. FPGA has better potential in terms of parallelism than DSP. GPU has very high power consumption, though it provides a flexible environment for parallelism. With the advancements in the field of semiconductor technology, FPGA's now hosts the capabilities like impacted size and low power utilization. During the span of 2014 to 2020, it is expected that the FPGA market will possess a CAGR growth rate of 9.1 % [2].

An extensive work and research has been conducted when it comes to image recognition topics based on FPGA and corresponding algorithms have been deployed and tested widely. In this work, the solution is implemented on a FPGA hardware unit and a top-down programming methodology is adopted in the integration tools (XILINX ISE 14.3). Each module can be downloaded into the FPGA, post execution of the following steps: implementation, compiling, function simulation, layout and timing simulation. The advantages of this work are compacting circuit board size, accompanied with enhancement in reliability of the system, providing for versatile design flexibility, thus leading to lower costs [3].

II. TOOLS AND TECHNIQUES

The prototype application for communication between FPGA and PC, and vice versa is as shown in block diagram below, Fig.1. It has VGA interface between FPGA and PC2 and JTAG interface between PC1 and FPGA. Custom designed Verilog module along with mapped text file generated through MATLAB is dumped on to FPGA and displayed on the VGA monitor on PC2. The corresponding tools and techniques used to develop this working prototype are listed below.

Fig. 1: Block Diagram of Prototype functioning

A. Hardware Unit1: Nexys3 Spartan-6

The Nexys3 is an FPGA hardware unit manufactured by Xilinx, which provides a digital circuit development platform, complete and ready to use. This hardware is based on the Xilinx Spartan-6 LX16 FPGA. The Field programmable gate arrays these days are entirely system on chip (SoC). The FPGA board used for VGA controller implementation in this project is a Spartan 6 kit by Nexys 3 [4].

It is a digital circuit development platform based on the Xilinx Spartan-6 LX16 FPGA. Its features include 2,278 slices, each slice contains four 6-input LUTs and eight flip-flops, 576kbits of fast block RAM, two clock tiles (four DCMs & two PLLs), 32 DSP slices and 500MHz+ clock speeds [4].

It includes 16Mbytes of Cellular RAM, 32Mbytes of micron's latest phase change non-

volatile memory, an USB-UART port, also an USB host port for mice and keyboards, a 10/100 ethernet PHY, and an improved high-speed expansion connector. These exclusive peripherals make the Nexys3 board an ideal host for a wide range of applications. Nexys3 uses Digilent's Adept USB2 system, which offers both FPGA and ROM programming and is compatible with all Xilinx CAD tools including EDK, ChipScope, and free WebPack [4].

B. Hardware Unit2: VGA Controller

VGA can be referred to as an Analog computer display or the 15 pin D-subminiature VGA connector or to the 640x480 resolution itself. The VGA Signal Timings have been tabulated in below Table.1 [5]. A 25 MHz pixel clock and 60 Hz ± 1 refresh was employed to derive these signal timings for a display (640 pixel by 480 row). The relation between each timing symbol is depicted in Fig.2. Through examination of different VGA display, the timing for sync pulse width (TPW) and front and back porch intervals (TFP and TBP) are derived. Back porch parameters are the pre-and post-sync pulse times. During these intervals, data cannot be displayed [5].

Symbol	**Parameter**	**Vertical Sync**			**Horizontal Sync**	
		Time	Clocks	Lines	Time µs	Clocks
TS	Sync Pulse time	16.7 ms	416,800	521	32	800
TDISP	Display time	15.36 ms	384,000	480	25.6	640
TPW	Pulse width	64 µs	1,600	2	3.84	96
TFP	Front porch	320 µs	8,000	10	640	16
TBP	Back porch	928 µs	23,200	29	1.92	48

Table 1: 640x480 VGA Mode timing

Generally, the horizontal timing is controlled by a counter clocked by the pixel clock. The HS signal is generated by decoded counter values. On a provided row, current pixel display is tracked by this counter. Vertical timing is tracked by a separate counter. Further, the VS signal is generated by decoded values. With every HS pulse, the vertical-sync counter gets incremented. Display row is tracked by this counter. The address into a video display buffer is formulated by these two simultaneously running counters. In between onset of HS pulse and onset of VS pulse, there is no time relationship specified. Consequently, the counters can be arranged to reduce the decoding logic for sync pulse generation or easy formation of video RAM addresses.

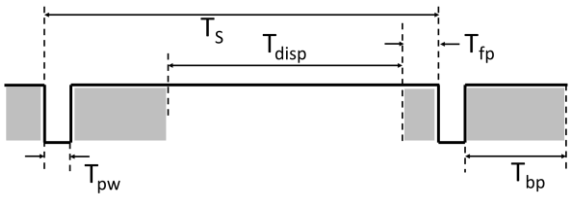

Fig. 2: VGA Control Timing

C. Software tool1: MATLAB 7.10 R2010A

The MATLAB tool was used to read the image which is supposed to be dumped on the FPGA kit and displayed on the VGA monitor screen

D. Software design tool2: Xilinx ISE 14.3

The Xilinx ISE 14.3 tool was used as a base IDE to run and execute Verilog code that was primarily done as a pre-test before hardware deployment

E. Tool Chain

The tool chain required to execute the project was established on factors like, no clash which may lead to compatibility and other unknown issues. The tool chain is displayed in the Fig.3 below.

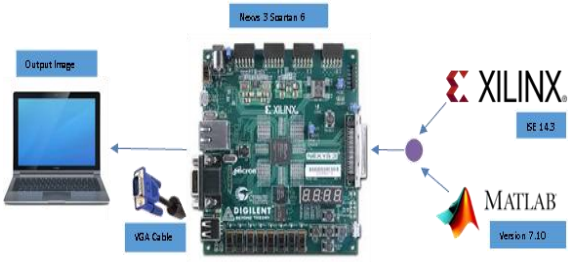

Fig. 3: Tool Chain for Prototype Application

In this work, a top model was formulated in Xilinx, which constituted of all the programs encompassing horizontal scan, vertical scan, color code, and clock division. MATLAB was used for pre-processing of the image, wherein the image was read and accompanied with conversion into text file, which was further dumped into the FPGA hardware unit. Next, FPGA sent across this processed image to the VGA monitor via VGA cable and the output was displayed in the monitor.

A 24-bit sample picture is taken. It is converted to a 3-bit format which is compatible with both Spartan 6 FPGA board and VGA display controller. The application uses three fundamental colors Red, Green and Blue. Each color is of 8bits, when these

colors are merged the combination will be a 24-bit image. Since the image should be made compatible with both the Spartan 6 FPGA and the VGA port, it is scaled down to 8bits. After scaling down each of the red and green colors, flatter down to 3bits each and blue color takes a 2bit value which when juxtaposed becomes an 8-bit image. The text file is the MSB value of R, G and B colors of each pixel. The 3-bit display color code is as shown in Table.2 below [5].

GA_RED	VGA_GREEN	VGA_BLUE	Resulting Color
0	0	0	Black
0	0	1	Blue
0	1	0	Green
0	1	1	Cyan
1	0	0	Red
1	0	1	Magenta
1	1	0	Yellow
1	1	1	White

Table 2: RGB values for color code 1

As shown in the above tabulation Table.2, the MSB value of R, G and B colors forms the text file indicating the resulting color. For example, if both Green and Blue are 0 and MSB Red is 1 then the color would be Red and so on the color strips corresponding to the bits are generated. When associated with the MATLAB code through which the image is read, the image is been dumped and displayed on the VGA screen.

III.PROPOSED ARCHITECTURE

Fig.4 depicts the proposed block diagram of VGA Controller, consisting reset generator block, clock generator block and VGA Controller block. Clock pulse of 100MHz and reset signal is provided as input to the block and RGB (red, green, blue), horizontal synchronization, vertical synchronization and blanking signals are obtained as outputs. The objective of reset block is to produce the reset signal and that of clock generator is to decrease the frequency of input clock from 100 MHz to 25 MHz, to realize 640 x 480 resolution [6], [8].

The VGA synchronization, vga_sync signal generates the timing and synchronization signals, vga_hs which specifies the time to scan a row and vga_vs specifies the time required to scan the entire screen. The vga_blank indicates retrace period of the display [5], [7].

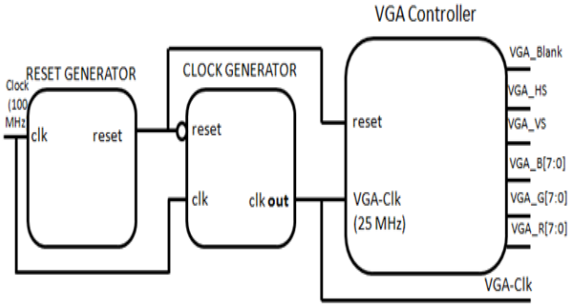

Fig. 4: Proposed Block Diagram

IV. IMPLEMENTATION AND RESULTS

A. Steps for Implementation

i. Take a Nexys3 Spartan-6 Development Board.

ii. Connect 12V supply through adapter to turn ON/OFF the board.

iii. Connect JTAG cable to PC1 and also VGA cable to PC2.

iv. Open Xilinx ISE14.3 software, by double clicking on Xilinx 14.3 icon and compile Verilog code in the software.

vi. Assign the pins at pin planner, by using user manual of the development board.

vii. In order to dump the program on board, select the device program, verify and observe the color strips on the VGA monitor screen of PC2.

viii. To display and observe the image associate and read the MATLAB code with the Verilog module through read mem function.

B. Flow Chart

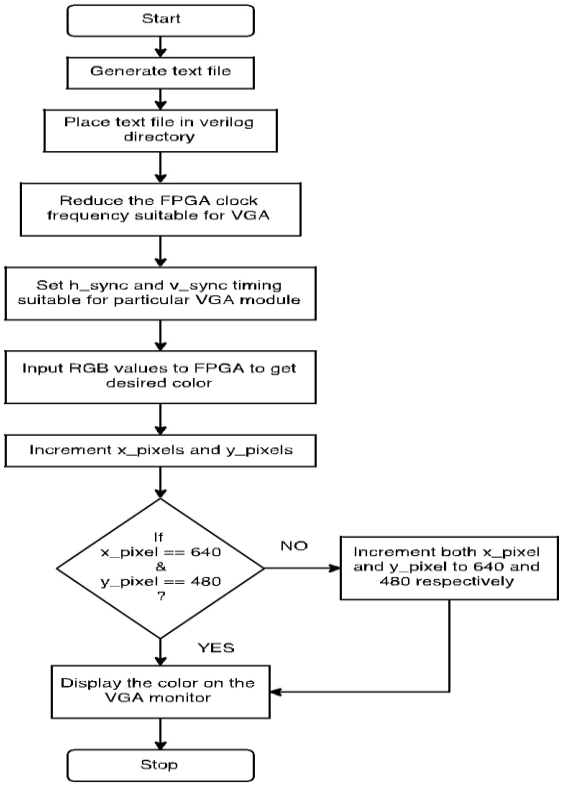

Fig. 4: Flow chart for VGA controller implementation

The flowchart specified in Fig.5 describes the operation of VGA controller and the verilog module flow. The following algorithm is used to describe the flowchart:
Step.1 Run the matlab code and generate a text file.
Step .2 Place the text file in the verilog directory
Step.3 Reduce the 100MHz FPGA clock frequency to 25MHz
Step.4 Set the horizontal and vertical synchronization signal timings suitable for the respective VGA controller mode
Step.5 Input the RGB values of 8bits to display the color strips on the VGA monitor
Step.6 Increment the location of x_pixes and y_pixels
Step.8 If x_pixel is equal to 640 and y_pixel is equal to 480, the condition satisfies and the respective image is displayed on the VGA monitor screen
Step.9 Else the pixels are incremented to 640x480 resolution and then the image is displayed.

C. Results

The results as compared with the existing system show all the possible combinations of RGB colors. This reduces the area required for the colors to be displayed and thus the power consumption [9]. Also the methodology involves generating text file of the image to be displayed which again saves the time required for hex file generation [10].

The verilog module is designed in Xilinx ISE 14.3. It includes codes for clock division that reduces the clock frequency to 25MHz, color display which is responsible for the display of the RGB colors of 8bits, horizontal and vertical synchronization codes for scanning the horizontal and vertical lines of the VGA screen.

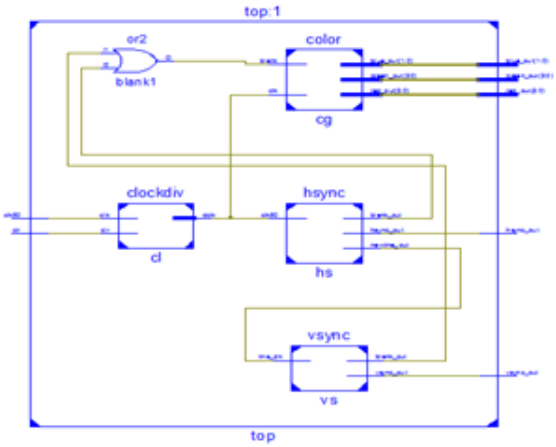

Fig. 6: RTL schematic of Verilog module

This verilog module can be observed in Fig.6 which shows the RTL schematic of the module. This module when simulated and dumped on to the Spartan 6 FPGA kit displays color strips on the VGA monitor screen of PC2 which can observed in Fig.7.

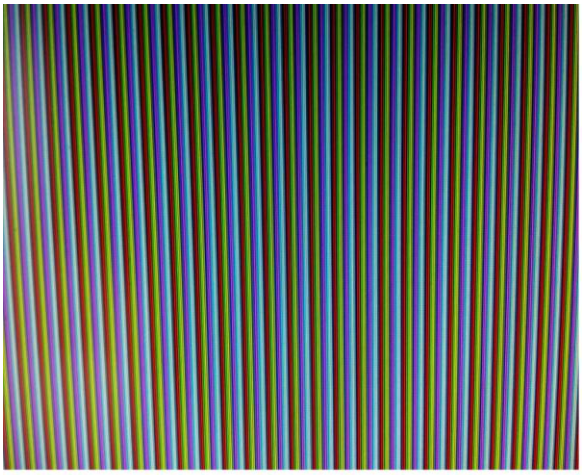

Fig. 7: Colour strips on VGA screen

Further coding is done in MATLAB to read a specific image and convert it into text file. This text while when placed in the directory of the verilog module can be read by the module through a read_mem file. This integrated code when dumped on to the Spartan 6 kit display the image read by the MATLAB on the VGA screen of PC2 which is as shown in Fig.9.

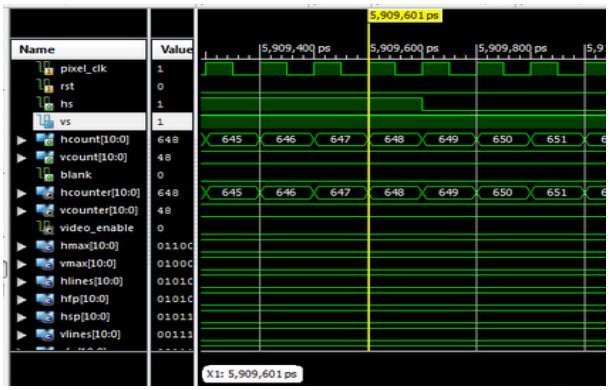

Fig.8: Simulation results

Fig. 9: Image displayed on VGA screen

V. CONCLUSION

A flexible communication is thus established between two PCs and the required image is transmitted. This communication defines and describes a VGA controller as to how it is used to establish interconnection between the VGA monitor screen and Spartan-6 FPGA through VGA port, in order to display the color strips formed by basic RGB colors and read the required image. Individual modules of VGA controller are developed using verilog HDL and functionally verified using Xilinx ISE. The synthesis is done using Spartan-6 FPGA and the results for the system development specify reduced number of resources utilized that is upto 54% of registers and 31% of functional units. In this proposed design, it can be said that FPGA is very feasible and convenient method in implementing VGA controllers as it requires new data to change the design and display each time. Verilog HDL makes the design flexible, reliable and convenient. The text file generation through MATLAB and VGA controller together are used for image processing and its results are better than the conventional BRAM method that consumes much processing time for image conversion [10].

REFERENCES

[1] FPGA Based Multi Resolution Graphics Controller, Abhijith S, International Journal of Engineering Trends and Technology (IJETT) – Volume 14 Number 6 – Aug 2014.

[2] http://economictimes.indiatimes.com/topic/CAGR FPGAmarket growth

[3] Philip H.W. Leong, Dept. of Computer Science and Engineering, Recent Trends in FPGA Architectures and Applications, 4th IEEE International Symposium on Electronic Design, Test & Applications

[4] Nexys3™Board Reference Manual Revision, (April 3, 2013), www.digilentinc.com

[5] Spartan-3,Starter Kit Board User Guide, Chapter 5 VGA Port, www.xilinx.com (online)

[6] Renuka A. Wasu, Vijay R. Wadhankar, Research Scholar, Dept. of ENC, Agnihotri College of Engineering, Nagthana Road, Wardha(M.S), India, H.O.D, Dept. of ENC, Agnihotri College of Engineering, Nagthana Road, Wardha (M.S), India, Design and Implementation of VGA Controller on FPGA, International journal of Innovative research in computer engineering.

[7] *W. James MacLean,* Department of Electrical & Computer Engineering, University of Toronto, Toronto, Ontario, M5S 1A1, maclean@eecg.toronto.edu, Evaluation of the Suitability An of FPGAs for Embedded Vision Systems

[8] Niveditha Yadav M, Yaseen Basha, Rohith S, Venkateshkumar H, Algorithm to Design VGA controller on FPGA Board

[8] V. Betz, J. Rose, and A. Marquardt, Eds., Architecture and CAD for Deep-Submicron FPGAs. Kluwer Academic Publishers,1999

[9] Video Graphics array interfacing through Artix-7 FPGA Mr. Naga V Satyanarayana Murthy, Asst. Professor, department of ECE GNITC, Hyderabad-501506, India

[10] Image processing using FPGA. By: Sumitha Ajith Saicharan, Bandarupalli and Mahesh Borgaonkar http://web.cecs.pdx.edu/~mperkows/temp/KALMAN/ECE 590_ProjectReport[1].pdf